https://doi.org/10.31436/iiumej.v27i1.3717

GNN-based Skyline Query Processing for Large-Scale and Incomplete Graphs

HASAN KHAIR ADZMAN, RAINI HASSAN*, DINI OKTARINA DWI HANDAYANI

Department of Computer Science, Kulliyyah of Information and Communication Technology, International Islamic University Malaysia, Kuala Lumpur, Malaysia

**Corresponding author: hrai@iium.edu.my*

(Received: 5 February 2025; Accepted: 1 October 2025; Published online: 12 January 2026)

ABSTRACT: Skyline queries are crucial in database management, selecting optimal points from multi-dimensional datasets based on dominance relationships. They are widely used in decision-making, recommendation systems, and data filtering. However, traditional skyline algorithms struggle with large volumes and missing data, leading to high computational costs and inefficiencies. This research proposes a hybrid approach that integrates the ISkyline dominance graph technique with Graph Neural Networks (GNNs) to improve skyline query performance under such conditions. The GNN component is utilized to predict skyline tuples in the presence of missing or incomplete data. Evaluation on both synthetic and real-world datasets demonstrates improved accuracy and efficiency compared with established methods such as ISkyline, SIDS, and OIS. This research demonstrates the potential to improve query processing efficiency and to support applications in e-commerce, finance, and smart data systems.

ABSTRAK: Pertanyaan latar langit adalah penting dalam pengurusan pangkalan data, iaitu dengan memilih titik optimum daripada set data berbilang dimensi berdasarkan hubungan dominasi. Ia digunakan secara meluas dalam membuat keputusan, sistem pengesyoran, dan penapisan data. Walau bagaimanapun, algoritma latar langit tradisional bergelut dengan kuantiti data yang besar dan data menghilang, membawa kepada peningkatan kos pengiraan dan ketidakcekapan. Kajian ini mencadangkan pendekatan hibrid yang mengintegrasi teknik graf penguasaan ISkyline dengan Rangkaian Graf Neural (GNNs) bagi meningkatkan prestasi pertanyaan latar langit berkeadaan sedemikian. Komponen GNN digunakan bagi meramalkan tupel latar langit dengan kehadiran data menghilang atau tidak lengkap. Penilaian pada kedua-dua set data sintetik dan dunia nyata menunjukkan peningkatan ketepatan dan kecekapan jika dibandingkan dengan kaedah sedia ada seperti ISkyline, SIDS dan OIS. Kajian ini menunjukkan potensi bagi mencipta pemprosesan pertanyaan yang lebih cekap, menyokong aplikasi e-dagang, kewangan dan sistem data pintar.

KEY WORDS: *Skyline query processing, Graph Neural Networks (GNNs), Incomplete data, Pareto optimality, Machine learning.*

1. INTRODUCTION

Skyline query processing is widely used in multi-criteria decision-making applications, including route planning, product recommendation, and health diagnostics. However, existing skyline methods face major challenges when applied to large and incomplete datasets, conditions that are increasingly common in real-world scenarios.

This research introduces a hybrid approach that combines Graph Neural Networks (GNNs) with the ISkyline dominance graph technique to enhance skyline query performance. The proposed method is designed to handle missing data and scale efficiently, thereby improving the prediction of skyline tuples even in complex, incomplete environments. Experimental results on both synthetic and real-world datasets demonstrate that this method outperforms state-of-the-art techniques in accuracy and efficiency.

Skyline queries aim to retrieve data records that are not dominated by any others across multiple dimensions, often referred to as Pareto-optimal points. While powerful, these queries are computationally expensive, particularly when applied to large graph-based datasets or those with incomplete attributes. Existing solutions, such as ISkyline and SIDS, aim to address scalability, but they still struggle to predict under uncertainty or in the presence of data loss.

Recent advances in deep learning, particularly Graph Neural Networks (GNNs), offer promising capabilities for learning from structured and incomplete data. By integrating GNNs into the skyline processing workflow, the proposed method leverages graph-based feature learning to support more robust and intelligent skyline selection.

Skyline queries are essential for identifying optimal data points from multi-dimensional datasets based on dominance relationships. However, traditional skyline query algorithms face significant limitations when processing large-scale, incomplete graph datasets. These methods often encounter challenges related to scalability, computational overhead, and inefficiencies in handling dynamic database environments [1], [2]. Furthermore, approaches such as Bucket and ISkyline struggle to integrate missing data effectively, resulting in suboptimal accuracy and high processing costs [3]. Current solutions lack a comprehensive framework that integrates state-of-the-art advances in machine learning, particularly Graph Neural Networks (GNNs), which have the potential to address these challenges by improving scalability, accuracy, and adaptability [4].

Despite the promising capabilities of machine learning, including its ability to model complex relationships in graph-structured data, its application in optimizing skyline queries remains underexplored [5]. Existing research does not adequately leverage the dynamic adaptability and efficiency of machine learning techniques, leaving a critical gap in addressing the computational and data-handling shortcomings of traditional methods. This study aims to bridge these gaps by introducing a novel framework that integrates Pareto optimality principles with advanced machine learning methods, providing robust solutions for scalable and efficient skyline computation in real-world settings.

Despite the potential of integrating Graph Neural Networks (GNNs) with skyline query processing, several limitations must be acknowledged. First, GNN-based approaches are resource-intensive, often resulting in high memory usage and longer query response times, particularly when dominance graphs are dense. Second, these models face challenges in interpretability, as it can be difficult to explain how dominance relationships are learned and applied in classification. Third, the computational requirements may restrict scalability when applied to large-scale, real-world datasets using modest hardware. These limitations frame the motivation for this study and highlight the importance of balancing accuracy improvements with efficiency and practical deployment considerations.

The research aims to accomplish three objectives. First, to develop a unified framework that integrates Pareto optimality principles with advanced machine learning techniques, particularly Graph Neural Networks (GNNs), to improve skyline query processing over large-scale and attribute-incomplete graphs. Second, to evaluate the performance of the proposed framework across real-world and synthetic datasets using comprehensive metrics, including

accuracy (target > 99%), F1-score (target > 99%), AUC-ROC (target > 99%), query response time, and memory usage, with a focus on ensuring scalability, efficiency, and adaptability in dynamic environments. Third, to compare the effectiveness of the proposed framework against traditional skyline algorithms and alternative machine learning models to identify the most suitable method for skyline query processing on incomplete graph-structured data.

Additionally, this research contributes significantly to academia and real-world applications by introducing a novel framework that integrates Pareto optimality with Graph Neural Networks (GNNs) to enhance skyline query processing in large-scale, incomplete datasets. Traditional methods often struggle with scalability and missing data, but this hybrid approach leverages Pareto optimality to identify non-dominated points and GNNs to learn from graph-structured, incomplete data. Their complementary strengths result in improved accuracy, F1-score, and AUC-ROC. The research also establishes standardized benchmarks using synthetic and real datasets, evaluating performance using metrics such as accuracy, F1-score, AUC-ROC, query response time, and memory usage. Practically, the solution is scalable and adaptive, benefiting industries such as e-commerce, finance, and smart infrastructure by enabling efficient, real-time decision-making even with incomplete data.

2. RELATED WORKS

Skyline queries identify optimal choices from a large dataset, such as selecting the top products that are inexpensive, fast, and well-rated. GNNs are useful when product information is incomplete or when relationships are important.

Prior research on skyline query processing has produced several key algorithms. Borzsony et al. [6] introduced the skyline operator to filter non-dominated points, inspiring methods like ISkyline [2], which partitions incomplete data using bitmaps, and SIDS [7], which applies sorted-based pruning. More recent work by Wang et al. [1] proposed Skyline Preference Queries (SPQ) for large, incomplete datasets.

Despite these advances, most traditional methods struggle with scalability, particularly in the presence of missing data. Few studies have explored machine learning approaches for skyline computation. Recent developments in Graph Neural Networks (GNNs) show promise for learning dominance structures directly from incomplete graphs [4], but their application in skyline queries remains underexplored. This research builds upon these foundations, proposing a hybrid GNN-based framework to address the identified gaps.

3. METHODOLOGY

3.1. Introduction

This section details the methodologies employed in this study, drawing on the design science research cycle as outlined by Hevner [8]. The approach integrates three cycles to enhance the identification and comprehension of design science research initiatives. Figure 1 illustrates the adapted design science research framework based on the research of von Brocke et al. [9].

https://doi.org/10.31436/iiumej.v27i1.3717

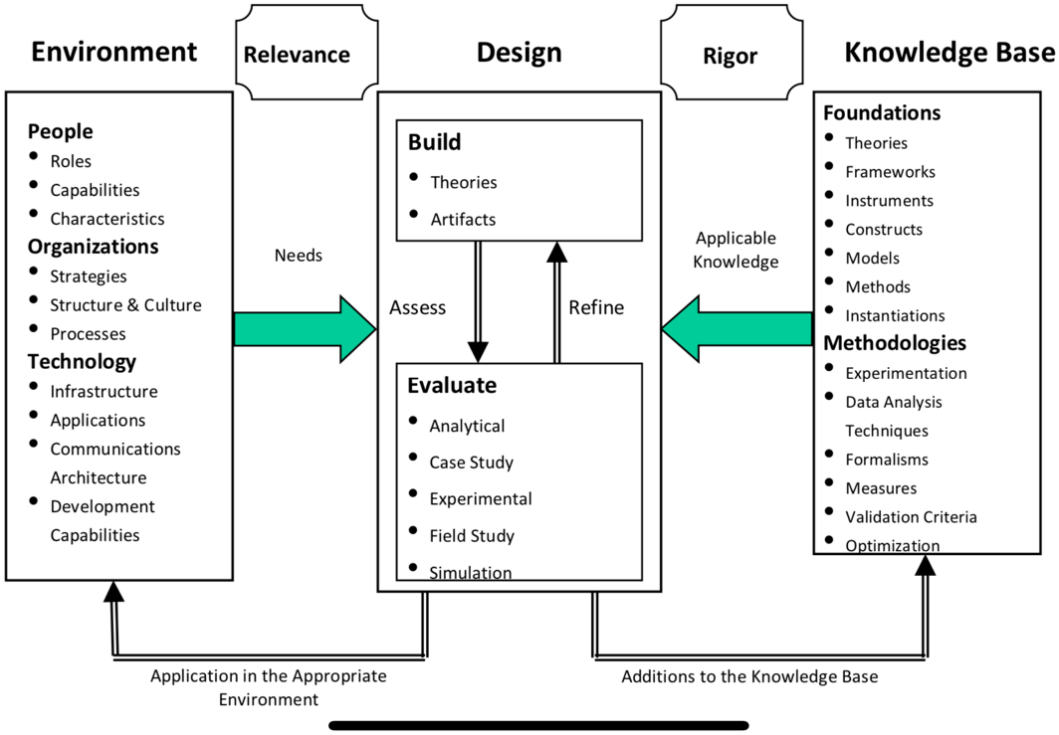

Figure 1. Design Science Research Framework [19]

The Relevance Cycle connects the research project's contextual environment with design science activities, ensuring that the needs and requirements for achieving the research objectives are adequately identified. The Rigor Cycle links design science activities to a knowledge base of scientific principles, expertise, and prior experiences that inform and guide the research process [10]. At the core is the Design Cycle, which focuses on developing and evaluating design artifacts and research processes. This cycle plays a pivotal role in describing the research activities. Figure 2 illustrates the sequential flow of the research process within this framework.

The research flow begins with a literature review that provides a foundational understanding of traditional skyline algorithms, machine learning frameworks, and graph-based methodologies. The process then advances to testing synthetic datasets as the initial step, followed by real datasets. Both workflows incorporate data preprocessing, including tasks such as data normalization, handling missing values, and splitting datasets into training, validation, and test sets. For synthetic datasets, the research involves selecting traditional algorithms (e.g., ISkyline) and machine learning frameworks (e.g., Graph Neural Networks), and integrating them to leverage the strengths of both approaches. The unified framework, along with traditional algorithms, is tested extensively, and the performance of these methods is compared using metrics such as processing time, memory usage, and accuracy. Results are analyzed to evaluate the effectiveness of the unified approach. Similarly, the real datasets testing follows the same workflow but includes an additional step of tuning the unified machine learning framework to optimize its performance for real-world applications. This tuning involves adjusting hyperparameters and evaluating alternative architectures (e.g., GraphSAGE). Finally, the results from both workflows are consolidated, discussed comprehensively, and used to propose a final algorithm that demonstrates superior performance in terms of scalability, robustness, and adaptability. This expanded explanation ensures a clear understanding of how each step in the methodology contributes to achieving the research objectives.

https://doi.org/10.31436/iiumej.v27i1.3717

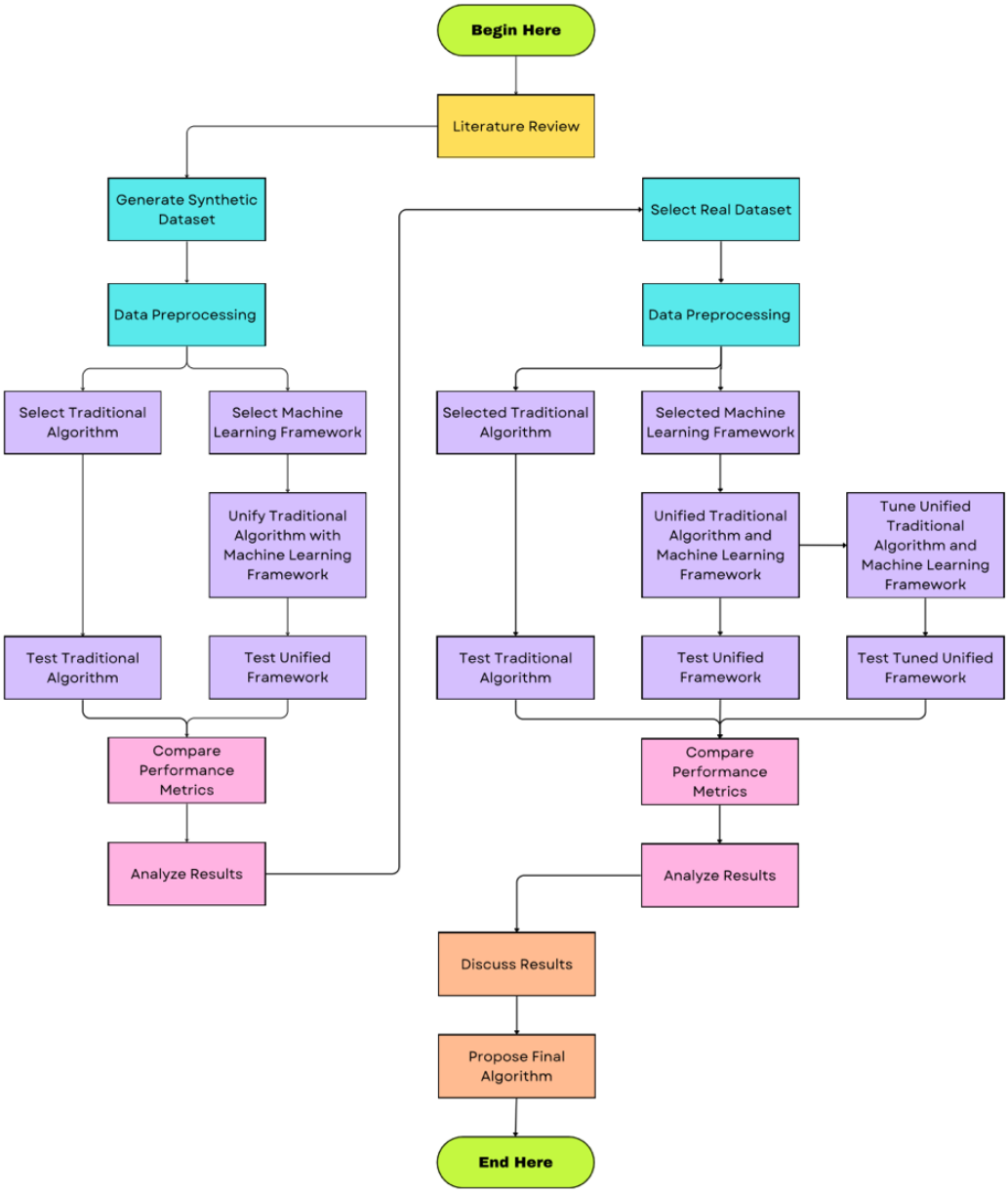

Figure 2. Design Cycle

3.2. Proposed Method: GNN + ISkyline

This research proposes a hybrid method that integrates the traditional skyline query algorithm ISkyline with a Graph Neural Network (GNN) to address attribute-level incompleteness in graph-structured data. The goal is to identify dominant relationships among data points and to improve skyline prediction in incomplete datasets using graph-based deep learning.

The proposed methodology begins by computing the ground-truth skyline from normalized and imputed data, then applies a modified ISkyline algorithm that handles missing values by skipping comparisons with None entries. Dominance is determined through a custom function that checks if one point is greater than or equal to another across all comparable attributes and strictly greater in at least one. Each data point is then labeled as skyline or non-skyline. To address class imbalance, skyline points are oversampled to match non-skyline points, and a directed dominance graph is constructed in which nodes represent products and

https://doi.org/10.31436/iiumej.v27i1.3717

edges represent dominance relationships. This graph feeds into a 3-layer Graph Convolutional Network (GCN), which uses normalized attributes as node features and binary skyline labels for supervision. The model is trained using binary cross-entropy loss with class imbalance adjustments and evaluated over 200 epochs on standard classification and efficiency metrics. Designed to handle incomplete data, this hybrid approach combines ISkyline logic with GNN-based pattern recognition to infer skyline membership effectively. Figure 3 below illustrates the full pipeline of this hybrid methodology.

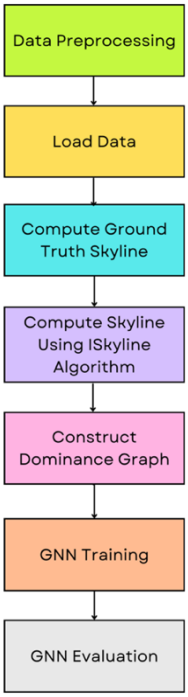

Figure 3. Proposed GNN + ISkyline Framework

By structuring the methodology in this modular fashion, the proposed approach provides transparency in its design while allowing each step to be assessed and improved independently. The integration of dominance relations with graph learning bridges classical skyline computation and modern neural representation learning, yielding a more scalable and accurate approach to handling incomplete multi-criteria datasets.

3.3. Dataset Preparation

This study evaluates the proposed framework using three real-world datasets: CoIL 2000 [11], NBA Stats [12], and MovieLens [13], selected for their diversity in domain, data volume, and feature complexity. These datasets are commonly cited in the literature on multi-criteria decision-making, incomplete data handling, and recommendation systems. To simulate real-world scenarios involving incomplete information, 20% attribute-level incompleteness was introduced into each dataset, either by randomly removing attributes or by building on pre-existing missing values.

CoIL 2000, with 5,822 complete customer records and 86 attributes divided into sociodemographic data (attributes 1–43) and product ownership (attributes 44–86). The NBA Stats dataset comprises 18,381 player regular-season records across 17 features, spanning the period 1946 to 2005. Features include standard demographics and performance statistics such as field goals, free throws, assists, and points per game. The MovieLens dataset consists of

https://doi.org/10.31436/iiumej.v27i1.3717

1,000,209 ratings from 6,040 users for approximately 3,900 movies. Each user contributed at least 20 ratings on a 5-point scale. Each row of the dataset records a user ID, movie ID, rating, and timestamp.

Table 1 summarizes the key characteristics of each dataset and details the approach taken to introduce and define incompleteness. The focus is on attribute-level incomplete graphs, in which node features may be partially missing while maintaining structural relationships via similarity-based edges.

Table 1. Summary of Real-World Datasets and Incompleteness Characteristics

Dataset	No. of Nodes	No. of Columns	Original Completeness	Incompleteness Introduced	Incomplete Graph Type	Graph Representation
CoIL 2000	5,822	86	Fully complete	20% of attribute values were removed randomly	Attribute-level incompleteness	Nodes: customers; Edges: feature similarity
NBA Stats	18,381	17	Partially incomplete	Removed attribute values to reach 20% of total missing values	Attribute-level incompleteness	Nodes: customers; Edges: feature similarity
MovieLens	1,000,209	4	Fully complete	20% of attribute values were removed randomly	Attribute-level incompleteness	Nodes: customers; Edges: feature similarity

To evaluate the robustness of the proposed framework under varying levels of attribute-level incompleteness, four synthetic e-commerce datasets were generated, each with 5,000 product nodes and six attributes: 1) Product ID, 2) Price, 3) Rating, 4) Availability, 5) Shipping Time, and 6) Category. Products in the same category are connected, forming 1,249,533 edges, yielding an edge density of approximately 0.1 (9.998%) of the possible 12,497,500 node-to-node connections. The three datasets introduce random incompleteness across all attributes at 10%, 50%, and 90%, enabling controlled robustness testing under progressively increasing levels of missing data, as shown in Table 2.

Table 2. Summary of Synthetic E-Commerce Datasets and Incompleteness Characteristics

Dataset	No. of Nodes	No. of Edges	Max Possible Edges	Edge Density	Attribute Missingness
Synthetic-10%	5,000	1,249,533	12,497,500	~0.1 (9.998%)	10%
Synthetic-50%	5,000	1,249,533	12,497,500	~0.1 (9.998%)	50%
Synthetic-90%	5,000	1,249,533	12,497,500	~0.1 (9.998%)	90%

These datasets provide a comprehensive testing environment for benchmarking the robustness of skyline algorithms and the proposed framework. By varying the degree of incompleteness while holding the graph topology constant, the research ensures that performance differences across methods are directly attributable to their handling of missing data. This controlled design is essential for deriving reliable conclusions about algorithmic behavior in incomplete graph settings.

3.4. Baseline Models

To evaluate the effectiveness of the proposed machine learning-based framework, three traditional skyline query algorithms, ISkyline, SIDS, and OIS, were selected as baselines due

to their prominence in handling incomplete data and large-scale graphs. ISkyline uses bitmap representations and shadow skylines to reduce dominance comparisons and manage missing values effectively. SIDS employs a round-robin sorting mechanism to prune dominated tuples, offering efficiency for static datasets with partial incompleteness. OIS, although not designed for missing data, minimizes I/O overhead and performs well in large-scale environments. These diverse, well-established methods provide a solid benchmark for assessing improvements in robustness, scalability, and processing efficiency offered by the proposed learning-based approach.

3.5. Evaluation and Benchmarking

To assess the performance of the proposed GNN + ISkyline framework and baseline models in classifying skyline and non-skyline points, five key metrics are used: 1) accuracy, 2) precision, 3) recall, 4) F1-score, and 5) AUC-ROC. These metrics are crucial in imbalanced datasets, where skyline points are often the minority. Eq. (1) shows that the accuracy measures overall correctness:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{1}$$

where TP is true positives, TN is true negatives, FP is false positives, and FN is false negatives.

Eq. (2) shows that the precision or the ratio of correctly predicted skyline points to all predicted skyline points, is given by:

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

Eq. (3) shows that the recall reflects the model's ability to detect actual skyline points:

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

Eq. (4) shows that the F1-score balances precision and recall using the harmonic mean:

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \tag{4}$$

AUC-ROC measures the model's ability to distinguish between classes across thresholds, with 1.0 indicating perfect classification and 0.5 representing random guessing. These metrics, computed on the test set, provide a comprehensive evaluation of model performance, particularly under varying degrees of data incompleteness.

Query response time is a critical performance metric for evaluating the efficiency of skyline query processing methods, particularly when dealing with large and incomplete datasets. In this study, query response time is defined as the total time from the initiation of query processing to the generation of the final results.

Memory usage is an essential metric for evaluating the computational efficiency of skyline query processing methods, particularly when applied to large-scale, incomplete datasets. It provides insights into the resource intensity of each technique, which is particularly relevant for deployment in memory-constrained environments. In this study, memory usage is measured as the peak memory usage observed during the complete execution cycle of each method.

3.6. Setup and Configuration

All experiments were conducted locally on a MacBook Air with an Apple M2 chip (8-core CPU, 8-core GPU, 16-core Neural Engine), 8 GB unified memory, and a 256 GB SSD running macOS Sequoia. Despite modest hardware, the system efficiently supported the framework's

https://doi.org/10.31436/iiumej.v27i1.3717

implementation in Python 3.11 using PyTorch (MPS backend), PyTorch Geometric, and Scikit-learn, with data handled via Pandas and NumPy, and visualizations done using Matplotlib and Seaborn. Model and batch optimizations enabled the smooth execution of experiments on both synthetic and real-world datasets, ensuring reproducibility and consistent performance across all tests.

4. RESULTS AND DISCUSSIONS

4.1. Experimental Setup and Dataset Description

To ensure the validity of the proposed GNN–ISkyline framework, experiments were conducted using both real-world and synthetic datasets. A detailed description of these datasets and the preprocessing steps is provided in Section 3. However, this section reiterates key aspects relevant to the experimental pipeline.

The real-world datasets used in this study include CoIL 2000, NBA Stats, and MovieLens. CoIL 2000 is a customer dataset with 5,822 records and 86 attributes, containing sociodemographic data and product ownership, originally completed before 20% of attribute values were removed to simulate incompleteness. The NBA Stats dataset contains season-based player statistics from 1946–2005, which are inherently partially incomplete and were further modified by removing attribute values to achieve 20% incompleteness. The MovieLens dataset contains over 1 million user ratings, initially complete, with 20% of attributes removed to simulate missing data. All datasets were transformed into graphs where nodes represent entities (e.g., customers, players, or users) and edges represent similarity based on features.

In addition to these real-world datasets, four synthetic e-commerce datasets were generated, each containing 5,000 product nodes. These datasets were constructed with six attributes: product ID, price, rating, availability, shipping time, and category. The three synthetic datasets were generated with uniform incompleteness levels of 10%, 50%, and 90%, respectively, by randomly removing attribute values across all columns.

4.2. Method Execution

The experiment involved several stages, beginning with data preprocessing, in which missing values were imputed with zeros and attributes were normalized using MinMaxScaler. Skyline labels were generated via dominance checks, and oversampling skyline points balanced the dataset. A dominance graph was then constructed with directed edges from dominating to dominated nodes, along with the feature and label tensors, and was fed into a 3-layer GCN model. The model was trained on 80% of the data and tested on the remaining 20%, with performance evaluated using accuracy, precision, recall, F1 score, and AUC-ROC, alongside query response time and peak memory usage as efficiency metrics.

4.3. Comparison of Algorithms Using Synthetic Data with 10%, 50%, and 90% Missingness

To provide a comprehensive evaluation, the results compare both standalone algorithms (GNN, ISkyline, SIDS, OIS) and their hybrid counterparts (e.g., GNN + ISkyline). The standalone models represent the baseline performance of traditional skyline methods or machine learning methods applied independently. In contrast, the hybrid models demonstrate that combining dominance-based algorithms with GNNs can enhance predictive accuracy. This comparison highlights not only which method performs best but also the trade-offs between classical efficiency and deep learning adaptability. The following results address the objectives 1, 2, and 3 stated in the Introduction.

https://doi.org/10.31436/iiumej.v27i1.3717

Table 3 presents a detailed evaluation of various algorithms under synthetic data with increasing levels of missingness (10%, 50%, and 90%), revealing how well each model handles incomplete information. Across all levels, GNN + SIDS consistently performs best on predictive metrics, achieving near-perfect scores in accuracy, precision, recall, F1-score, and AUC-ROC, even with 90% missing data, while maintaining modest memory usage and fast response times. ISkyline performs surprisingly well in terms of accuracy and precision but suffers from extremely low recall and F1-score, indicating poor sensitivity. Interestingly, GNN + ISkyline also maintains high performance but at the cost of very high memory usage and response time, especially under 90% missingness. OIS remains robust with high F1-scores and AUC-ROC, as missingness increases.

Table 3. Comparison of Algorithms Using Synthetic Data with 10%, 50%, and 90% Missingness

10% Missingness							
Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC-ROC**	**Query Response Time (s)**	**Peak Memory Usage (KB)**
GNN	0.5020	0.5020	0.9861	0.6667	0.4842	139.2902	921.39
ISkyline	0.9698	1.0000	0.0195	0.0382	0.5097	0.1603	168.90
GNN + ISkyline	0.9919	0.9841	1.0000	0.9920	0.9878	843.2566	477584.90
SIDS	0.9916	0.0455	1.0000	0.0870	0.9958	0.4705	2971.04
GNN + SIDS	0.9928	0.9860	1.0000	0.9929	0.9943	749.9065	478148.27
OIS	0.9728	0.0145	1.0000	0.0286	0.9864	0.7676	424.48
GNN + OIS	0.0308	0.0308	1.0000	0.0598	0.0009	147.5776	105397.89
50% Missingness							
Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC-ROC**	**Query Response Time (s)**	**Peak Memory Usage (KB)**
GNN	0.4930	0.4930	1.0000	0.6604	0.4919	608.6592	912.55
ISkyline	0.9738	0.0000	0.0000	0.0000	0.5000	0.1257	168.68
GNN + ISkyline	0.9931	0.9870	1.0000	0.9935	0.9956	1147.2394	1027135.04
SIDS	0.9996	0.9957	1.0000	0.9978	0.9998	7.3685	2976.11
GNN + SIDS	0.9995	0.9990	1.0000	0.9995	1.0000	990.1274	1027707.01
OIS	0.9966	0.9647	1.0000	0.9820	0.9981	0.8260	426.06
GNN + OIS	0.0262	0.0262	1.0000	0.0511	0.0007	162.6048	381806.19
90% Missingness							
Algorithm	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC-ROC**	**Query Response Time (s)**	**Peak Memory Usage (KB)**
GNN	0.5110	0.4364	0.0498	0.0894	0.5216	1041.0278	919.07
ISkyline	0.9906	1.0000	0.1296	0.2295	0.5648	0.0365	168.02
GNN + ISkyline	0.9980	0.9960	1.0000	0.9980	1.0000	4667.7921	2157517.52
SIDS	1.0000	1.0000	1.0000	1.0000	1.0000	105.9054	3000.64
GNN + SIDS	0.9995	0.9990	1.0000	0.9995	1.0000	4543.4297	2157766.90
OIS	0.9936	0.9905	1.0000	0.9952	0.9904	6.8788	416.32
GNN + OIS	0.9504	0.1723	0.9444	0.2914	0.9757	191.4423	622770.62

In contrast, GNN + OIS, despite strong recall, underperforms substantially in accuracy, precision, F1-score, and AUC-ROC at lower missingness levels, recovering only somewhat at 90%. This behavior likely arises because OIS does not handle missing attributes well without pre-filtering, and GNNs overfit to the sparse dominance structure, leading to overprediction.

https://doi.org/10.31436/iiumej.v27i1.3717

Additionally, the simple graph structure generated by OIS may lack sufficient complexity for the GNN to extract meaningful patterns, causing ineffective learning.

By contrast, ISkyline constructs richer dominance relationships that provide the GNN with more informative features, improving classification quality. However, this richness comes at a high computational price, since storing and traversing large dominance graphs significantly increases both memory usage and query response time. This trade-off indicates that GNN + ISkyline is highly accurate but not resource-efficient. In contrast, GNN + OIS offers faster runtime and lower memory use but struggles with predictive performance due to a weaker graph representation.

4.4. Comparison of Algorithms Using CoIL 2000

The comparison between standalone and hybrid models is critical here, as it demonstrates how incorporating GNN learning alters the behavior of traditional skyline methods. Standalone algorithms such as ISkyline and OIS reflect their inherent design strengths and weaknesses, whereas hybrids demonstrate the potential improvements, along with the additional costs, of integrating neural network learning. This helps establish whether hybridization truly adds value over efficient standalone baselines.

Table 4 presents a comparative analysis of multiple skyline query processing algorithms using the CoIL 2000 dataset. The results show that traditional methods like ISkyline and SIDS perform poorly in terms of accuracy and F1-score, despite ISkyline achieving perfect precision due to its overly conservative predictions. In contrast, GNN-based models, particularly when combined with optimized skyline algorithms such as OIS, significantly outperform alternatives. The GNN + OIS approach achieves near-perfect performance, with accuracy (0.9698), F1-score (0.9846), and AUC-ROC (0.9950), while maintaining reasonable memory usage. Notably, the standalone OIS algorithm also delivers exceptional results (accuracy of 0.9796) with minimal query time (49.52s), suggesting its standalone strength. However, integrating GNN with OIS marginally increases resource usage but yields the best overall performance. This demonstrates that hybrid models that leverage GNNs with optimized skyline strategies provide the most effective and scalable solution for skyline queries on incomplete data.

Table 4. Comparison of Algorithms Using CoIL 2000

Algorithm	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Query Response Time (s)	Peak Memory Usage (KB)
GNN	0.4884	0.4881	0.8949	0.6316	0.5016	4.6432	336.96
ISkyline	0.0076	1.0000	0.0012	0.0024	0.5006	898.8737	16020.15
GNN + ISkyline	0.7333	0.5556	1.0000	0.7143	1.0000	229.6736	19673.39
SIDS	0.0064	0.0000	0.0000	0.0000	0.5000	1700.9073	70757.20
GNN + SIDS	0.7870	0.7297	0.9123	0.8109	0.8642	724.7528	143958.74
OIS	0.9796	0.9569	1.0000	0.9780	0.9813	49.5192	9158.45
GNN + OIS	0.9698	1.0000	0.9696	0.9846	0.9950	338.3995	38407.19

OIS is inherently optimized for I/O efficiency, which explains why OIS exhibits lower query times and memory consumption than ISkyline. Although GNN + ISkyline achieves a

https://doi.org/10.31436/iiumej.v27i1.3717

strong AUC-ROC, its memory footprint is an order of magnitude higher, making it less practical for real-time applications.

Figure 4 compares the accuracy of various algorithms and their combinations for skyline query processing. Standalone traditional algorithms such as ISkyline and SIDS perform poorly, with extremely low accuracies of 0.0076 and 0.0064, respectively. The GNN alone performs better, achieving an accuracy of 0.4884, indicating its ability to model graph structures but lacking in domain-specific optimization. However, combining GNNs with traditional methods significantly boosts performance. GNN + ISkyline and GNN + SIDS achieve much higher accuracies of 0.7333 and 0.7870, respectively. Notably, the OIS algorithm outperforms all with an accuracy of 0.9796, and when paired with GNNs (GNN + OIS), it maintains a comparably high performance at 0.9698. This indicates that hybrid models, particularly those that incorporate OIS, are the most effective for accurate skyline query processing on complex or incomplete datasets.

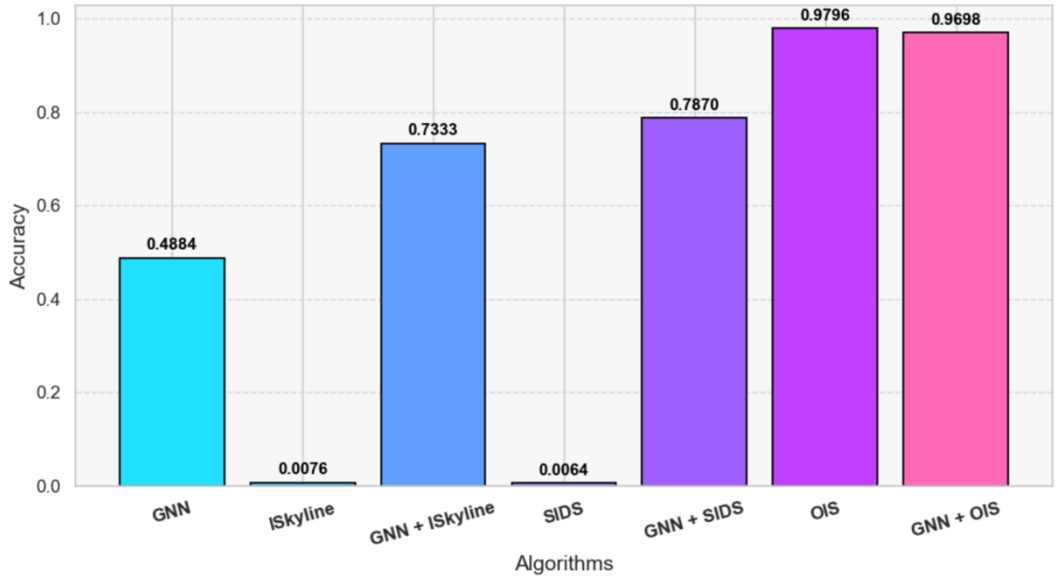

Figure 4. Accuracy Comparison of Algorithms Using CoIL 2000

Figure 5 reveals significant disparities in the performance of different algorithms used for skyline query processing. Traditional methods like ISkyline and SIDS show near-zero effectiveness, with F1-scores of 0.0024 and 0.0000, respectively, indicating their poor balance between precision and recall. The standalone GNN achieves a moderate F1-score of 0.6316, suggesting reasonable performance in capturing skyline points despite data incompleteness. Notably, combining GNN with classical methods significantly enhances performance: GNN + ISkyline achieves 0.7143, and GNN + SIDS further improves to 0.8109. The top-performing models are OIS and GNN + OIS, achieving outstanding F1-scores of 0.9780 and 0.9846, respectively. This underscores that hybrid models, particularly those integrating GNNs with OIS, offer superior capability in maintaining precision-recall tradeoffs, making them ideal for accurate skyline detection in complex datasets.

https://doi.org/10.31436/iiumej.v27i1.3717

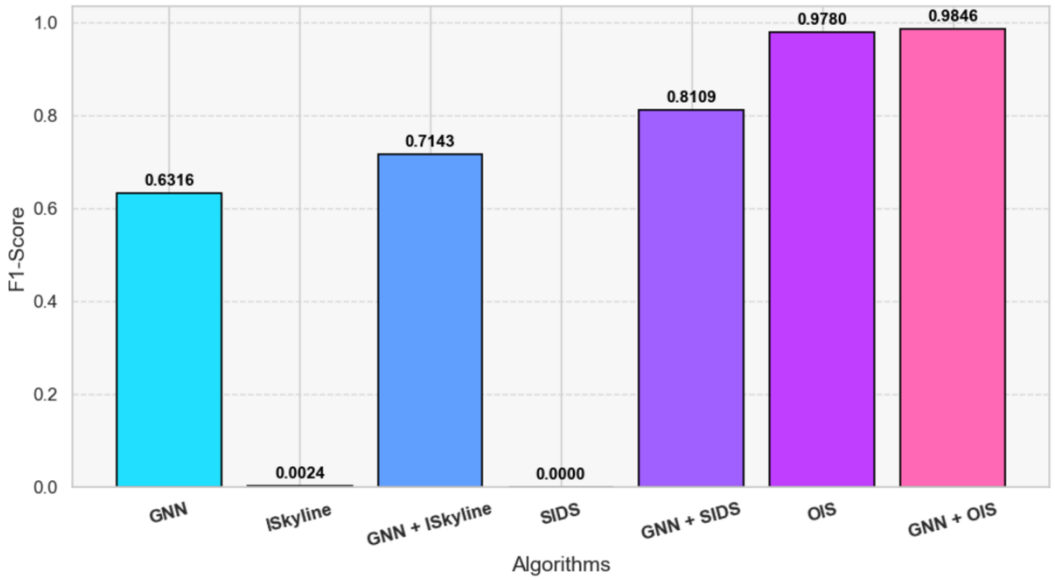

Figure 5. F1-Score Comparison of Algorithms Using CoIL 2000

Figure 6 provides a clear assessment of each algorithm's ability to distinguish between skyline and non-skyline points. Traditional algorithms such as ISkyline and SIDS, along with GNN, hover around a baseline performance of 0.50, indicating they perform no better than random guessing. Interestingly, the combination of GNN + ISkyline achieves an AUC-ROC of 1.0000, which may indicate overfitting or a specific synergy in that dataset. Meanwhile, GNN + SIDS outperforms both methods individually, achieving 0.8642, underscoring the benefits of combining deep learning with classic skyline methods. The OIS model again demonstrates robustness with a high score of 0.9813, and the GNN + OIS hybrid achieves the highest score of 0.9950, reinforcing that integrating GNNs with optimized skyline strategies yields the most reliable and discriminative results across all metrics.

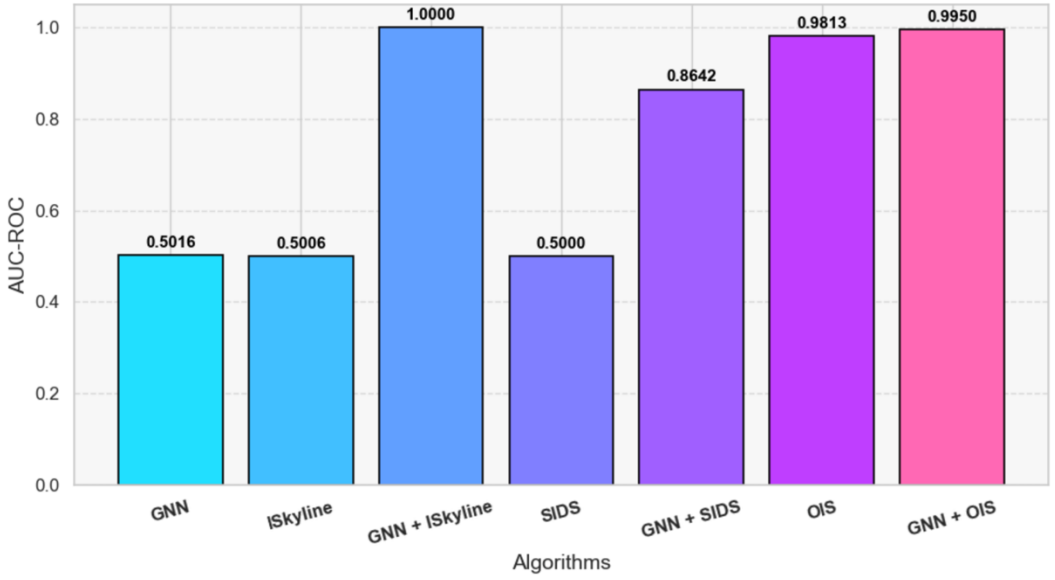

Figure 6. AUC-ROC Comparison of Algorithms Using CoIL 2000

Figure 7 reveals a striking contrast in performance among various algorithms. ISkyline and SIDS perform extremely poorly, with near-zero values across most metrics, indicating that they are ineffective standalone solutions. GNN performs moderately well with a Recall of 0.89

https://doi.org/10.31436/iiumej.v27i1.3717

and an F1-score of 0.63, but struggles with accuracy and precision. Interestingly, GNN + ISkyline achieves perfect precision and AUC-ROC, although its recall (0.56) and accuracy (0.73) limit overall effectiveness. GNN + SIDS shows significant improvement across all metrics, particularly with an F1-score of 0.81. The standout performers are OIS and GNN + OIS, both of which achieve near-perfect or perfect scores across every metric. Notably, GNN + OIS slightly outperforms OIS in AUC-ROC (0.99 vs. 0.98), making it the most robust and balanced model overall, demonstrating the efficacy of hybridizing deep learning with optimized skyline strategies.

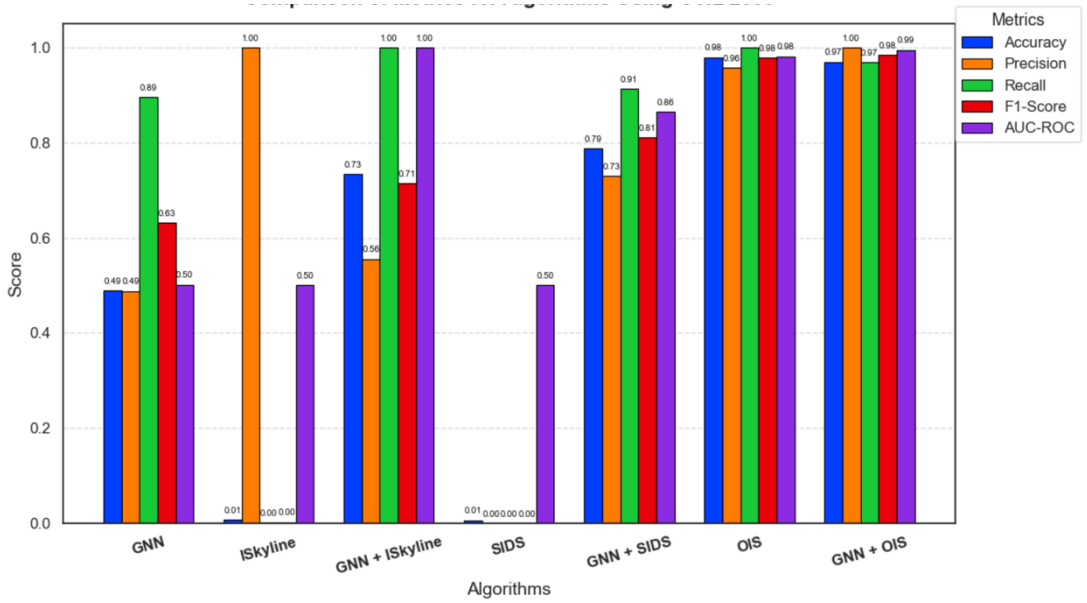

Figure 7. Comparison of Metrics for Algorithms Using CoIL 2000

Table 5. Comparison of Machine Learning Using CoIL 2000

Algorithm	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Query Response Time (s)	Peak Memory Usage (KB)
GNN + OIS	0.9698	1.0000	0.9696	0.9846	0.9950	338.3995	38407.19
GAT + OIS	0.9775	1.0000	0.9774	0.9885	0.9949	351.3499	25964.56
XGBoost + OIS	0.9777	0.9956	0.9819	0.9887	0.8689	97.8628	29138.93
RL + OIS	0.9957	0.9957	1.0000	0.9978	0.6622	875.2617	77173.20
OL + OIS	0.8699	0.9966	0.8721	0.9302	0.7229	106.1407	61728.71
GraphSAGE + OIS	1.0000	1.0000	1.0000	1.0000	1.0000	402.7302	25888.41

Table 5 compares the performance of various machine learning algorithms integrated with the OIS skyline optimization technique using the CoIL 2000 dataset. GraphSAGE + OIS achieves perfect scores (1.0000) across all metrics, making it the top performer in terms of predictive quality, albeit with moderate memory usage (25,888.41 KB) and a relatively high response time (402.73s). RL + OIS also demonstrates near-perfect scores but exhibits the highest memory usage (71,173.20 KB) and the longest response time (875.26s), limiting its practicality. GAT + OIS and XGBoost + OIS achieve strong metric performance and improved efficiency, with XGBoost + OIS having the shortest response time (97.86s) and low memory usage. However, its AUC-ROC is significantly lower (0.8689). GNN + OIS maintains a good balance between accuracy (0.9698), high AUC-ROC (0.9950), and moderate resource use.

https://doi.org/10.31436/iiumej.v27i1.3717

Conversely, OL + OIS, despite a decent precision (0.9966), lags in other metrics, particularly recall (0.8721), and exhibits high memory consumption (61,728.71 KB). In summary, GraphSAGE + OIS offers the best performance.

Figure 8 illustrates the comparative performance of various OIS-enhanced algorithms on the CoIL 2000 dataset across five key metrics: accuracy, precision, recall, F1-score, and AUC-ROC. GraphSAGE + OIS achieves a perfect score of 1.00 across all metrics, demonstrating strong classification performance. RL + OIS also achieves perfect scores for all metrics except AUC-ROC, which drops significantly to 0.66, indicating a weaker ability to distinguish between classes under varying thresholds. OL + OIS performs poorly compared to others, especially in AUC-ROC (0.72) and recall (0.87), despite high precision. GNN + OIS, GAT + OIS, and XGBoost + OIS demonstrate a strong balance of metrics, all hovering close to 0.99, although XGBoost's AUC-ROC lags slightly at 0.87. These results suggest that although several models exhibit competitive predictive performance, GraphSAGE + OIS yields the most robust and consistent performance across all tasks.

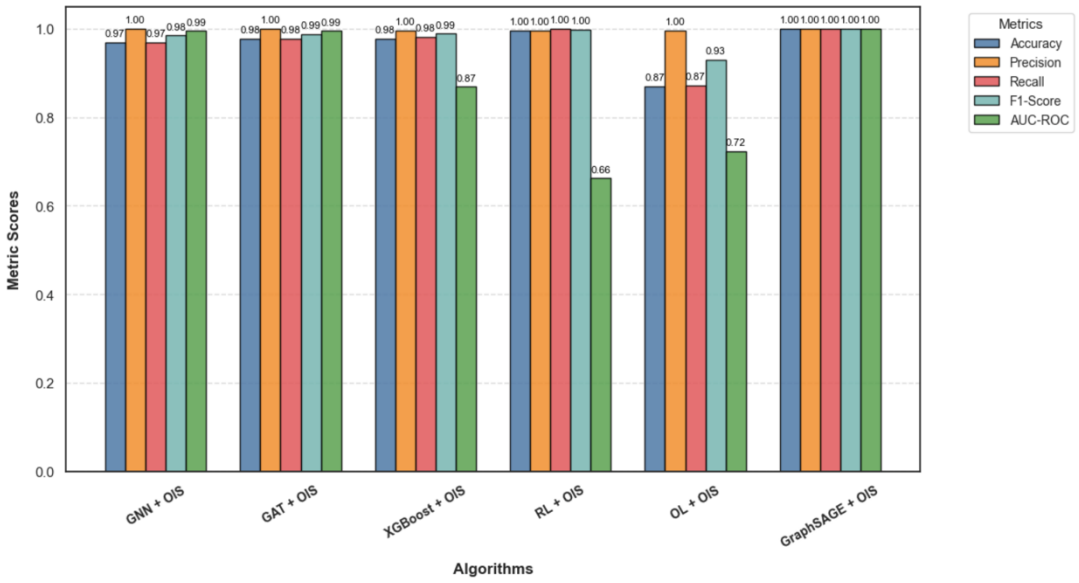

Figure 8. Comparison of OIS Variants Across Metrics Using CoIL 2000

4.5. Comparison of Algorithms Using NBA Stats

In this section, GNN + SIDS and GAT + ISkyline are excluded from the algorithm comparison because they consistently crashed during execution in the Jupyter Notebook. Table 6 presents a performance comparison of various algorithms applied to the NBA Stats dataset, evaluating both classification metrics and computational efficiency. OIS emerges as the top performer overall, achieving near-perfect accuracy (0.9973), perfect recall (1.0), and an excellent AUC-ROC (0.9986), all while maintaining the fastest response time (1.86s) and relatively low memory usage (4324 KB), making it the most balanced and efficient solution. GNN + ISkyline also performs strongly in accuracy (0.9533) and F1-score (0.9515), but at the cost of extremely high query response time (~17,680s) and massive memory usage (2.5 GB), which limits its practicality. ISkyline and SIDS, though faster and lighter, exhibit near-zero recall and F1 Scores, indicating that they miss nearly all relevant results. GNN alone also underperforms with low recall (0.0753) and F1-score (0.1314), despite modest memory demands. Finally, GNN + OIS achieves perfect recall but low accuracy (0.1162) and F1-score (0.2081), suggesting indiscriminate classification, and it also consumes over 1 GB of memory.

https://doi.org/10.31436/iiumej.v27i1.3717

Overall, OIS stands out as the best choice in terms of accuracy, efficiency, and real-world applicability for the NBA Stats dataset.

Table 6. Comparison of Algorithms Using NBA Stats

Algorithm	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Query Response Time (s)	Peak Memory Usage (KB)
GNN	0.5072	0.5150	0.0753	0.1314	0.4936	2279.9321	576.44
ISkyline	0.8840	1.0000	0.0014	0.0028	0.5007	75.5164	1697.09
GNN + ISkyline	0.9533	0.9841	0.9210	0.9515	0.9905	17679.7926	2524215.45
SIDS	0.8838	0.0000	0.0000	0.0000	0.5000	820.3373	36999.28
OIS	0.9973	0.8106	1.0000	0.8954	0.9986	1.8601	4324.18
GNN + OIS	0.1162	0.1162	1.0000	0.2081	0.0159	1650.1123	1031888.89

The NBA dataset, with its high feature sparsity and temporal attributes, exacerbates the inefficiency of ISkyline-based dominance graphs, necessitating excessive resources for GNN + ISkyline. OIS, however, remains efficient because its pruning strategies minimize comparisons, yielding faster, lighter computations while maintaining near-perfect accuracy. This contrast emphasizes the scalability advantage of OIS over ISkyline in real-world, high-volume datasets.

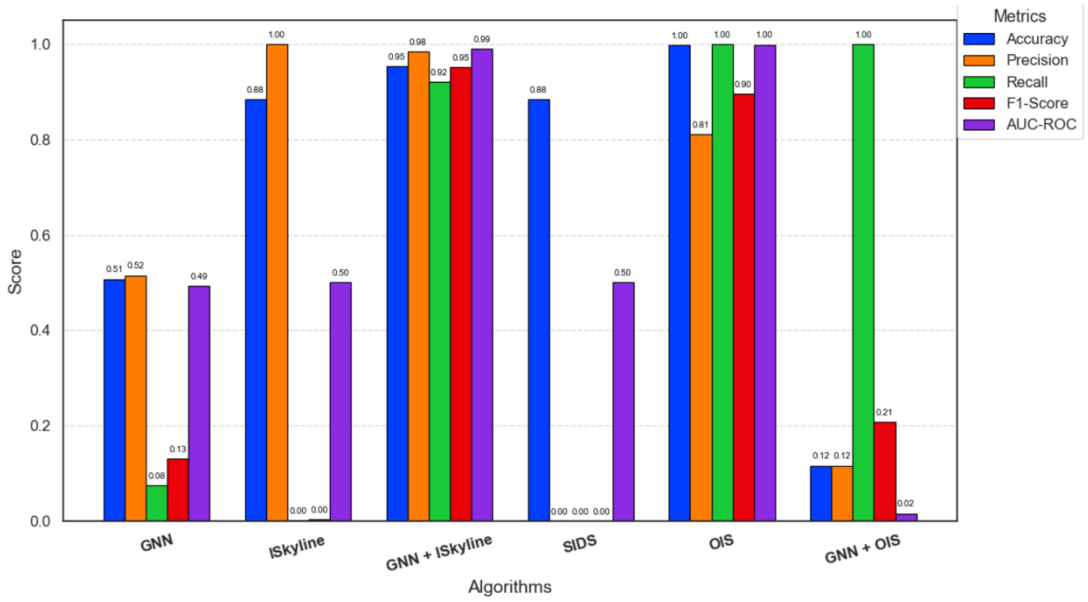

Figure 9. Comparison of Metrics for Algorithms Using NBA Stats

Figure 9 provides a comprehensive side-by-side comparison of multiple performance metrics, accuracy, precision, recall, F1-score, and AUC-ROC, for different algorithms applied to the NBA Stats dataset. OIS is the most balanced and effective model overall, achieving near-perfect or perfect scores across all metrics, including accuracy (1.00), recall (1.00), and AUC-ROC (1.00), making it the top choice for both accuracy and class separation. GNN + ISkyline also performs exceptionally, with high precision (0.9841), strong recall (0.9210), and a very high AUC-ROC (0.9905), showcasing its strong generalization and consistency. On the other hand, ISkyline and SIDS exhibit a complete collapse in recall and F1-score (both ≈0.00),

https://doi.org/10.31436/iiumej.v27i1.3717

despite ISkyline's perfect precision, implying either extreme overfitting or a failure to detect positive cases. GNN performs modestly, with mediocre values across all metrics. In contrast, GNN + OIS achieves high recall (1.00) but very poor performance on all other metrics, particularly accuracy (0.12) and AUC-ROC (0.02), indicating that it indiscriminately labels all points as the skyline. The chart emphasizes the robustness and efficiency of OIS, followed by GNN + ISkyline, while also highlighting the critical weaknesses in standalone traditional models and poorly integrated hybrids.

Table 7 compares various ISkyline-integrated machine learning models on the NBA Stats dataset, evaluating their performance across performance metrics and computational efficiency. GraphSAGE + ISkyline dominates with a perfect score of 1.000 in all performance metrics, accuracy, precision, recall, F1-score, and AUC-ROC, indicating flawless classification. However, this comes at the cost of very high query response time (~8943s) and massive memory usage (~2.5 GB), making it computationally intensive. XGBoost + ISkyline offers the best trade-off, achieving excellent classification results, with accuracy (0.9843), F1-score (0.9844), and AUC-ROC (0.9979), while being far more efficient in time (~482s) and memory (~126 MB). Similarly, RL + ISkyline and OL + ISkyline perform well with decent F1-scores (0.8985 and 0.8347, respectively) and fast response times (<470s), especially OL + ISkyline, which is the most memory-efficient (19.5 MB). In contrast, GNN + ISkyline, while accurate (0.9533), exhibits the longest query time (~17680s) and highest memory consumption (~2.5 GB), thereby limiting its practicality. Overall, GraphSAGE + ISkyline is ideal for scenarios requiring perfect accuracy regardless of resource constraints, whereas XGBoost + ISkyline strikes the best balance between performance and efficiency.

Table 7. Comparison of Machine Learning Using NBA Stats

Algorithm	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Query Response Time (s)	Peak Memory Usage (KB)
GNN + ISkyline	0.9533	0.9841	0.9210	0.9515	0.9905	17679.7926	2524215.45
XGBoost + ISkyline	0.9843	0.9699	0.9994	0.9844	0.9979	481.6498	126111.93
RL + ISkyline	0.9001	0.9072	0.8901	0.8985	0.9663	462.2957	40069.25
OL + ISkyline	0.8209	0.7709	0.9099	0.8347	0.9292	426.5436	19517.34
GraphSAGE + ISkyline	1.0000	1.0000	1.0000	1.0000	1.0000	8942.8623	2524046.19

4.6. Comparison of Algorithms Using MovieLens

When testing the proposed GNN + ISkyline framework on the MovieLens dataset across different dataset sizes (1M, 100k, 30k, 20k, and 10k records), several computational challenges were observed. Processing the 1M, 100k, and even 10k record subsets still required substantial time, indicating that the method remains computationally intensive even at smaller scales. More critically, attempts to run the model on the 30k and 20k subsets led to kernel crashes.

The following results were evaluated on the MovieLens dataset using 5k records, which was sufficient for successful execution. However, the GAT + ISkyline model was excluded from this test because it caused a kernel crash during execution.

As shown in Table 8, GNN + ISkyline achieves the highest overall performance, achieving perfect scores of 1.0000 for accuracy, precision, recall, F1-score, and AUC-ROC. However, this comes with a relatively high query response time (2878.10s) and peak memory usage (1927721.87 KB). ISkyline alone also performs very well in accuracy (0.9978) and precision

https://doi.org/10.31436/iiumej.v27i1.3717

(1.0000), but its recall (0.2778) and F1-score (0.4348) are notably lower, suggesting limited effectiveness in capturing positive cases. OIS achieves excellent accuracy (0.9993) and reasonable AUC-ROC (0.7497), with the benefit of speedy query response time (0.0562s) and low peak memory usage (723.07 KB). GNN + SIDS demonstrates balanced performance, with a high AUC-ROC (0.9051) and relatively efficient resource utilization. Meanwhile, GNN and SIDS alone perform poorly, with zero recall and F1-score values. GNN + OIS struggles significantly, with extremely low accuracy (0.0031) and AUC-ROC (0.0005), despite moderate memory usage.

Table 8. Comparison of Algorithms Using MovieLens

Algorithm	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Query Response Time (s)	Peak Memory Usage (KB)
GNN	0.5171	0.0000	0.0000	0.0000	0.5243	590.1351	375.99
ISkyline	0.9978	1.0000	0.2778	0.4348	0.6389	0.8235	223.54
GNN + ISkyline	1.0000	1.0000	1.0000	1.0000	1.0000	2878.0988	1927721.87
SIDS	0.9969	0.0000	0.0000	0.0000	0.5000	69.2292	3450.52
GNN + SIDS	0.8187	0.8302	0.8010	0.8154	0.9051	2.1972	2283.49
OIS	0.9993	0.5000	0.7500	0.6000	0.7497	0.0562	723.07
GNN + OIS	0.0031	0.0031	1.0000	0.0061	0.0005	212.3288	344711.13

MovieLens data, being user-item interaction-based, produces extremely dense graphs under ISkyline. This enables GNN + ISkyline to learn dominance structures effectively, thereby achieving perfect accuracy. However, the density results in prohibitive memory consumption (approximately 2 GB) and long runtimes. In contrast, GNN + OIS collapses on MovieLens because OIS pruning removes too much relational information, leaving the GNN with insufficient features to generalize.

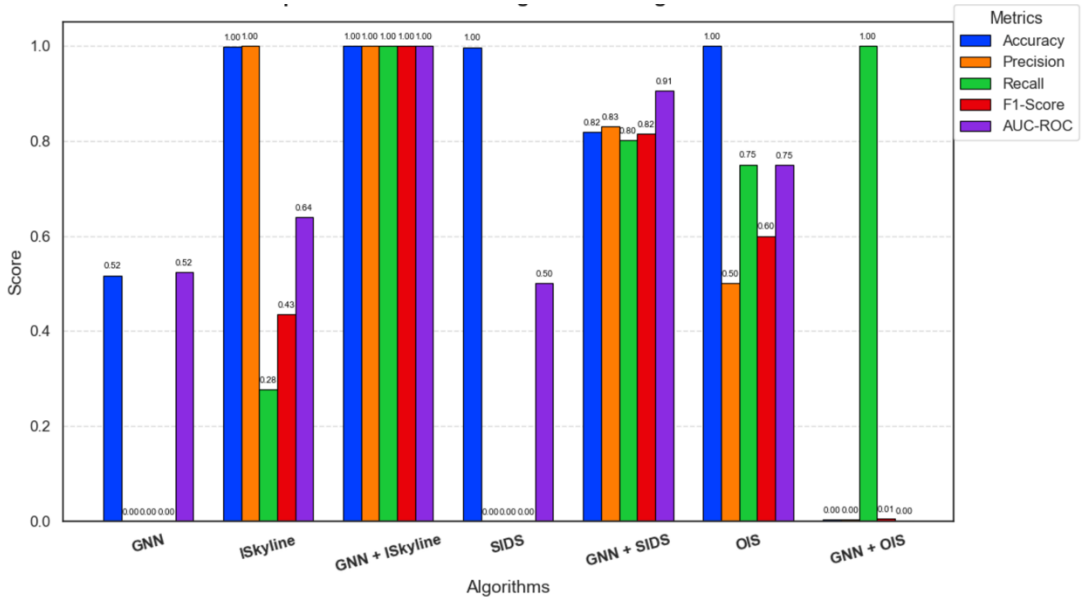

Figure 10. Comparison of Metrics for Algorithms Using MovieLens

Figure 10 highlights GNN + ISkyline as the top-performing approach, achieving perfect scores (1.00) across all five metrics, accuracy, precision, recall, F1-score, and AUC-ROC,

https://doi.org/10.31436/iiumej.v27i1.3717

demonstrating outstanding overall performance. GNN + SIDS also performs strongly, achieving high scores across all metrics (0.80–0.91), although slightly lower than GNN + ISkyline. OIS shows good accuracy and recall but lower precision (0.50) and F1-score (0.60), and moderate AUC-ROC (~0.75). In contrast, ISkyline alone achieves excellent accuracy and precision but falls short on recall (0.28) and F1-score (0.43), indicating an imbalance. GNN and SIDS individually struggle, with most metrics 0 except for moderate accuracy (~0.52) and AUC-ROC (~0.52) for GNN. GNN + OIS notably underperforms, with recall the only strong metric (1.00), whereas all other metrics are nearly 0, indicating extremely poor general classification performance. Overall, integrating GNN with ISkyline is the most effective combination, followed by GNN + SIDS.

Table 9 presents a comparison of machine learning models combined with ISkyline on the MovieLens dataset, showing that GNN + ISkyline and GraphSAGE + ISkyline both achieved perfect scores (1.0000) across all evaluation metrics, accuracy, precision, recall, F1-score, and AUC-ROC, demonstrating flawless performance. XGBoost + ISkyline also performed exceptionally, achieving near-perfect metrics with accuracy (0.9996), precision (0.9991), recall (1.0000), F1-score (0.9996), and AUC-ROC (1.0000), while maintaining a low query response time and modest memory usage compared to GNN-based methods. OL + ISkyline showed the lowest performance among the listed models but still maintained decent accuracy (0.9424) and AUC-ROC (0.9842). In terms of resource efficiency, OL + ISkyline was the most lightweight, with the lowest memory usage, whereas the GNN and GraphSAGE variants required significantly more memory. Overall, all models exhibit excellent performance, but the GNN + ISkyline and GraphSAGE + ISkyline combinations stand out for achieving perfect classification at the cost of substantially higher memory and processing demands.

Table 9. Comparison of Machine Learning Using MovieLens

Algorithm	Accuracy	Precision	Recall	F1-Score	AUC-ROC	Query Response Time (s)	Peak Memory Usage (KB)
GNN + ISkyline	1.0000	1.0000	1.0000	1.0000	1.0000	2878.0988	1927721.87
XGBoost + ISkyline	0.9996	0.9991	1.0000	0.9996	1.0000	98.6301	2527.05
RL + ISkyline	0.9832	0.9674	1.0000	0.9834	0.9931	96.9954	51641.99
OL + ISkyline	0.9424	0.9523	0.9309	0.9415	0.9842	93.6479	2007.34
GraphSAGE + ISkyline	1.0000	1.0000	1.0000	1.0000	1.0000	1728.0954	1924217.91

5. CONCLUSION

This research on GNN-based skyline query processing for large-scale and incomplete graphs showed that integrating Graph Neural Networks (GNNs) with Pareto optimality principles improves skyline query performance by enhancing scalability, reducing computational overhead, and effectively handling incomplete data. The proposed GNN-based framework outperformed traditional algorithms such as ISkyline, SIDS, and OIS in accuracy, F1-score, and AUC-ROC, and demonstrated adaptability to dynamic environments with real-time updates. Benchmarks using metrics such as accuracy, F1-score, AUC-ROC, query response time, and memory usage validated its effectiveness across synthetic and real-world datasets.

Furthermore, this research meaningfully supports Sustainable Development Goal 9 (Industry, Innovation, and Infrastructure) by demonstrating scalable, adaptive techniques for data-intensive systems. Handling incomplete and large-scale data robustly ensures the resilience and scalability of digital infrastructures, essential for future smart cities, industrial automation, and sustainable development platforms. By addressing both scalability and data incompleteness, the proposed framework directly contributes to the development of more intelligent, responsive, and reliable infrastructure systems.

Computational limitations constrained this research due to the use of a consumer-grade MacBook Air (M2, 8GB RAM). Limited memory and processing capacity occasionally caused kernel crashes, especially during large-scale graph construction and GNN training on bigger datasets (e.g., NBA Stats, MovieLens). As a result, some hybrid models could not be thoroughly tested with vast datasets. In future work, experiments should be conducted using more powerful computing resources, such as high-memory GPUs or cloud-based platforms (e.g., AWS EC2 P4 instances, Google Cloud TPU Pods). This would enable evaluation of larger graphs, deeper GNN models, and hyperparameter tuning at a larger scale, thereby further validating the proposed framework under realistic, production-level conditions.

Despite the promising results of machine learning models in skyline query processing, several practical limitations remain. First, GNN-based methods exhibit high memory usage, especially when processing large or dense graphs, which may hinder scalability in resource-constrained environments. Second, these deep learning models often suffer from interpretability challenges, making it difficult to understand how specific dominance relationships are learned or how skyline classifications are made, an issue that can limit their adoption in critical decision-making systems where transparency is essential. Third, while synthetic datasets enable controlled experimentation, they may not fully capture the complexity and noise of real-world data, potentially limiting the generalizability of the models trained on them. These limitations highlight the need for further research to optimize resource use, enhance model explainability, and validate findings on more diverse, real-world datasets.

Future research could extend this work in several directions. First, scalability can be improved by investigating lightweight or compressed GNN architectures (e.g., pruning, quantization, or knowledge distillation) to reduce memory and runtime costs without sacrificing accuracy. Second, explainability should be prioritized by integrating interpretable GNN models or post hoc explanation methods, thereby enabling users to understand dominance relationships in skyline predictions better. Third, a broader evaluation on diverse real-world datasets beyond CoIL 2000, NBA Stats, and MovieLens is essential to test generalizability across domains such as healthcare, transportation, and financial risk analysis. Finally, exploring distributed or parallel skyline computation frameworks using cloud-based platforms could further support deployment in large-scale, real-time environments. Together, these directions will help bridge the gap between theoretical contributions and practical deployment of GNN-based skyline query processing.

ACKNOWLEDGEMENT

This research was supported by the Fundamental Research Grant Scheme (FRGS) under Reference Code FRGS/1/2021/ICT01/UIAM/02/2 (Project ID 19574) from the Ministry of Higher Education (MOHE), Malaysia.

REFERENCES

[1] Wang, Y., Shi, Z., Wang, J., Sun, L., & Song, B. (2017). Skyline Preference Query Based on Massive and Incomplete Dataset. IEEE Access, 5, 3183–3192. https://doi.org/10.1109/access.2016.2639558

[2] Khalefa, M. E., Mokbel, M. F., & Levandoski, J. J. (2008). Skyline Query Processing for Incomplete Data. https://doi.org/10.1109/icde.2008.4497464

[3] Bharuka, R., & Kumar, P. S. (2013a). Finding superior skyline points from incomplete data. 35–44. https://doi.org/10.5555/2694476.2694488

[4] Afifi, H., Pochaba, S., Boltres, A., Laniewski, D., Haberer, J., Leonard, P., Poorzare, R., Stolpmann, D., Wehner, N., Redder, A., Samikwa, E., & Seufert, M. (2024). Machine Learning with Computer Networks: Techniques, Datasets and Models. IEEE Access, 1–1. https://doi.org/10.1109/access.2024.3384460

[5] Mohamud, M. A., Ibrahim, H., Sidi, F., Mohd, N., Dzolkhifli, Z., Zhang, X., & Lawal, M. M. (2023). A Systematic Literature Review of Skyline Query Processing Over Data Stream. IEEE Access, 11, 72813–72835. https://doi.org/10.1109/access.2023.3295117

[6] Borzsony, S., Kossmann, D., & Stocker, K. (2001). The Skyline operator. Proceedings 17th International Conference on Data Engineering. https://doi.org/10.1109/icde.2001.914855

[7] Bharuka, R., & Kumar, P. S. (2013b). Finding skylines for incomplete data. 109–117.

[8] Hevner, A., March, S., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. MIS Quarterly, 28(1), 75–105. https://doi.org/10.2307/25148625

[9] vom Brocke, J., Hevner, A., & Maedche, A. (2020). Introduction to Design Science Research. Progress in IS, 1–13. https://doi.org/10.1007/978-3-030-46781-4_1

[10] Hevner, A., & Chatterjee, S. (2010). Design science research in information systems. In Design Research in Information Systems. Springer, Boston, MA., 9–22.

[11] Putten, P. (2000). Insurance Company Benchmark (COIL 2000) [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5630S

[12] (2000) The Basketball-Reference website. [Online]. Available: https://www.basketball-reference.com

[13] (2025) The GroupLens website. [Online]. Available: https://grouplens.org/datasets/movielens/