PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
الجمهـوريـــة الجزائـريـــة الديـمـقراطيـــة الشـعبيـــة
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
وزارة التعليـم العالـي والبحـث العلمـي
HIGHER SCHOOL OF COMPUTER SCIENCE AND DIGITAL TECHNOLOGIES-BEJAIA
المـدرسة العـليا في علـوم وتكنـولوجيـات الاعـلام الالـي بجـاية

Dissertation Submitted to the Department Of Computer Science in Partial Fulfillment of the Requirements for Engineer's Degree in Computer Science

Specialty: Artificial Intelligence and Data Sciences

Submitted By:

Madadi Mounia

Recommender systems for multimodal transportation systems in smart cities

Supervised by:

Dr. Dr. Houda Elbouhissi
Litan laboratory - ESTIN

Members of jury:

▪ **Pr.** AZOUAOU Faical	President	Litan laboratory - ESTIN
▪ **Dr.** LEKEHALI Somia	Examiner	Litan laboratory - ESTIN
▪ **Mme.** Bouali Meriem	Examiner	Litan laboratory - ESTIN
▪ **Mme.** Ziani Laldja	Examiner	Litan laboratory - ESTIN

Academic year: 2023/2024

To my dear mother, who has always been my source of strength and without whom I would have achieved nothing,

To my late father, whom I hope to make proud with every step I take in my life,

For my husband Amine, who is a constant source of support at every step I take,

For my brother Boualem and my sister Ikram, my companions in life who walk beside me in every step of life,

For my grandfather and all my uncles and aunts who have shown me support as if I were their own daughter,

For my in-laws who have quickly become a second family,

I dedicate this humble work,

In the hope of making more contributions in the future, InshAllah.

Acknowledgments

I wish to extend my sincere thanks to those who have supported me throughout my journey and enabled me to complete this work.,

I would like to warmly thank my supervisor, Dr. Elbouhissi Houda, for her invaluable assistance, expert advice, and consistent guidance throughout this project,

I am deeply grateful to the jury members, Bouali Meriem, Ziani Laldja, Dr. LEKEHALI Somia, and Pr AZOUAOU Faical, for their willingness to review this thesis,

Furthermore, I wish to express my genuine appreciation to all my previous ESTIN teachers, whose individual contributions have provided me with invaluable guidance and collectively enabled me to undertake this endeavor,

I also extend my thanks to all my friends and classmates throughout my university journey for their supportive advice and significant role in shaping my path forward.,

I also thank my entire family for their constant support, which has been crucial throughout my journey and has contributed significantly to my achievements.

Abstract

Transportation recommendation systems have been trending for quite some time due to their continuous potential for improvement. Among the most interesting advancements are multimodal transportation recommendation systems, which provide suggestions for traveling from one location to another using a combination of available transportation modes. In this thesis, we present a multimodal transportation recommendation system that recommends trajectories to users based on their personal preferences. Our system consists of two main phases. The first phase involves trajectory generation, where we search for optimal trajectory combinations between the starting point and the destination using Particle Swarm Optimization, followed by post-processing on the trajectories. Once the trajectories are generated, we rank them using the RankNet model trained on previously selected user trajectories, employing a content-based approach.After testing our system, we observed that the generated trajectories were quite feasible and the recommendations were highly accurate.

keywords:
Recommendation system, multimodal transport, RankNet, Particle Swarm Optimization, content-based approach, public transport.

Table of Contents

List of Abbreviations

CB Content based systems. 13, 16

CF Collaborative filtering systems. 14, 16

HBF Hybrid based filtering systems. 14, 16

ICT Information and communication technologies. 13

MAE Mean Absolute Error. 17

MTR Multi-modal transportation recommendations. 23

PSO Particle Swarm Optimization. 5, 11, 28, 31–33, 41

RMSE Root Mean Squared Error. 17

RS Recommendation Systems. 13–17

SC Smart Cities. 18–20

List of Algorithms

List of Figures

List of Tables

Chapitre 1 : Introduction

The diversity of public transportation options in urban environments, including metro, tramway, buses, trains, and taxis, provides a multitude of travel choices. Navigating through these alternatives can be complex, especially for those unfamiliar with the city, whether they are tourists or residents from other metropolitan areas. Finding the optimal public transport trajectory from one point to another, considering factors such as cost, comfort, and travel duration, can be challenging. In this context, the crucial importance of recommendation systems becomes evident, especially within the framework of 'smart cities'.

The transportation sector, in particular, significantly benefits from the rapid evolution of these systems, which demonstrate an increasing ability to provide personalized recommendations to each user. A concrete example of this utility lies in proposing the best trajectory, taking into account the user's past travel choices, budget preferences, and time and comfort preferences. In connected urban environments, this convergence between combination of available transportation modes and intelligent recommendations offers a particularly relevant solution to simplify and optimize city travel.

This thesis addresses the issue of recommending trajectories by integrating various modes of transportation within a city and presenting them to users based on their personal preferences regarding cost, time, and preferred mode of transport. We will first construct the trajectories using Particle Swarm Optimization; once the trajectories are refined to ensure presentability, we will rank them for the user using a RankNet model trained on the user's previous journeys.

We summarize our contribution in the following main points:

- first, we conducted a state-of-the-art review discussing approaches in the field of transport recommendation, focusing specifically on multimodal transport recommendations.
- Next, we proposed our approach, which involves generating trajectories using PSO and then ranking them using the RankNet model.
- Finally, we implemented our method and tested it on a dataset from the public transport network of the city of Paris

We have structured our thesis into six essential chapters to enhance understanding and presentation of our approach. The first chapter serves as the introduction. The second chapter provides background on the pillars of our theme: recommendation systems, smart cities, and multimodal transportation, setting the stage within the current context.

The third chapter presents a literature review on recent developments in recommendation systems to identify current trends. It also explores various approaches used in multimodal transportation recommendation systems, synthesizing the presented works.

In the fourth chapter, we discuss our approach to multimodal transportation recommendation. Our method combines PSO for generating trajectories and RankNet for recommending them. We highlight the key aspects of our approach, focusing on its potential to improve trajectory recommendation systems and enhance user experience in urban transportation.

In the fifth chapter, we delve into the practical implementation and evaluation of our system. We start by elaborating on the datasets used in both the trajectory generation and recommendation phases. Following this, we discuss the development environment used for system implementation, highlighting the utilization of Streamlit for system presentation. Lastly, we present the results obtained from the evaluation conducted for both phases.

The sixth and final chapter, Conclusion and Perspectives, wraps up our study by presenting the conclusions drawn from our research. We provide an overview of the work accomplished throughout the thesis, highlighting the contributions, discoveries, and limitations of the developed recommendation system.

Chapitre 2 : Background

2.1 Introduction

In this section, we cover three main topics: recommendation systems, smart cities, and multi-modal transportation. Recommendation systems personalize user experiences, smart cities use Information and communication technologies (ICT) to optimize urban services, and multi-modal transportation integrates diverse transit modes for efficient mobility.

2.2 Recommendation Systems

Recommendation Systems (RS) have enjoyed success since their first appearance in the 1990s Adomavicius and Tuzhilin 2005. Despite the extensive work done to develop new approaches in RS, research has not waned. Every day, we witness the increasing indispensability of RS in our lives due to their involvement in various activities Iftikhar et al. 2023. In today's world, where there is a vast array of choices across all domains ranging from selecting a movie to watch, choosing a meal to eat, to determining which trajectories to take, RS provides invaluable support in making these decisions, thus reducing information overload Rodríguez-Hernández et al. 2015; Saini and Singh 2023. Therefore, each year sees improvements in the implementation of RS to achieve optimal results Saifudin and Widiyaningtyas 2024. Furthermore, even the current generation of RS requires further improvement to be more effective and applicable in broader domains of real life Adomavicius and Tuzhilin 2005.

2.2.1 Definition

In its most common definition, a Recommander System is an algorithm that rates items, aiming to identify those with the highest ratings, as they are likely the ones preferred by the user. Subsequently, the system recommends these highly-rated items to the user Adomavicius and Tuzhilin 2005; Saifudin and Widiyaningtyas 2024; Rodríguez-Hernández et al. 2015. Additionally, it's important to mention that the method of rating varies among different RS. Each system employs its own unique approach to analyzing data sources in order to establish connections between users and items Melville and Sindhwani 2002.

2.2.2 Types of Recommendations Systems

As the methods of generating recommendations vary from one system to another, RS are categorized into three main classes as shown in the figure 1 that describe the general approach: Collaborative Filtering, Content-based Filtering, and Hybrid filtering, which combines elements of both Adomavicius and Tuzhilin 2005; Rodríguez-Hernández et al. 2015; Melville and Sindhwani 2002.

Content based systems (CB): This method operates on the intuition that if a user liked a particular item, they'll probably like similar ones as well. It works by identifying similarities between items and then recommending those with features similar to ones the user has liked before Saifudin and Widiyaningtyas 2024; Rodríguez-Hernández et al. 2015. To determine the items the user liked in the past, various methods are employed. These can be explicit, such as directly asking the user, or implicit, which involves analyzing user behavior Adomavicius and Tuzhilin 2005. In this approach, dealing with cold starts 2.2.3 is usually not an issue, as every item/user has its attributes/preferences defined once it's entered in the database. However, if there isn't a sufficient amount of information about the user's preferences, he'll receive fewer recommendationsSaifudin and Widiyaningtyas 2024.

Collaborative filtering systems (CF): In this second approach, the RS suggests items to users based on ratings from others whose behaviors align closely with theirs Adomavicius and Tuzhilin 2005; Rodríguez-Hernández et al. 2015. This method stands out as the most popular and widely utilized one, with significant advancements in both academia and industry Adomavicius and Tuzhilin 2005. While this approach is widely used, it has both advantages and disadvantages. One advantage is that it doesn't require additional information about items or users when adding them to the database; rather, it relies on the interaction between users and items. However, if there isn't enough interaction, the system may generate fewer recommendations Saifudin and Widiyaningtyas 2024. Collaborative filtering systems, in turn, are divided into two general classes: Memory-based, which uses the database each time for recommendation, and Model-based, which constructs a model for predictions Breese, Heckerman, and Kadie 1998. Similar to the previous approach, the ratings can be either explicit or implicit Breese, Heckerman, and Kadie 1998.

Hybrid based filtering systems (HBF): Both of the instinctive approaches mentioned have their own set of advantages and disadvantages. To address the limitations of each while capitalizing on their strengths, hybrid collaborative systems were developed. These RS types combine elements of both collaborative filtering and content-based systems into a single structure Adomavicius and Tuzhilin 2005; Saifudin and Widiyaningtyas 2024; Rodríguez-Hernández et al. 2015. We have various methods available for constructing hybrid collaborative systems, which vary according to the data employed Saifudin and Widiyaningtyas 2024. These methodologies can be categorized into four main classes Adomavicius and Tuzhilin 2005:

1. Implementing collaborative and content-based methods independently, then combining their predictions.
2. Integrating specific content-based features into a collaborative approach.
3. Incorporating selected collaborative characteristics into a content-based approach.
4. Developing a comprehensive unified model that incorporates both content-based and collaborative attributes.

The article Adomavicius and Tuzhilin 2005 provides an extensive review of various approaches utilized in the construction of hybrid recommendation systems, drawing from a wide range of scholarly works.

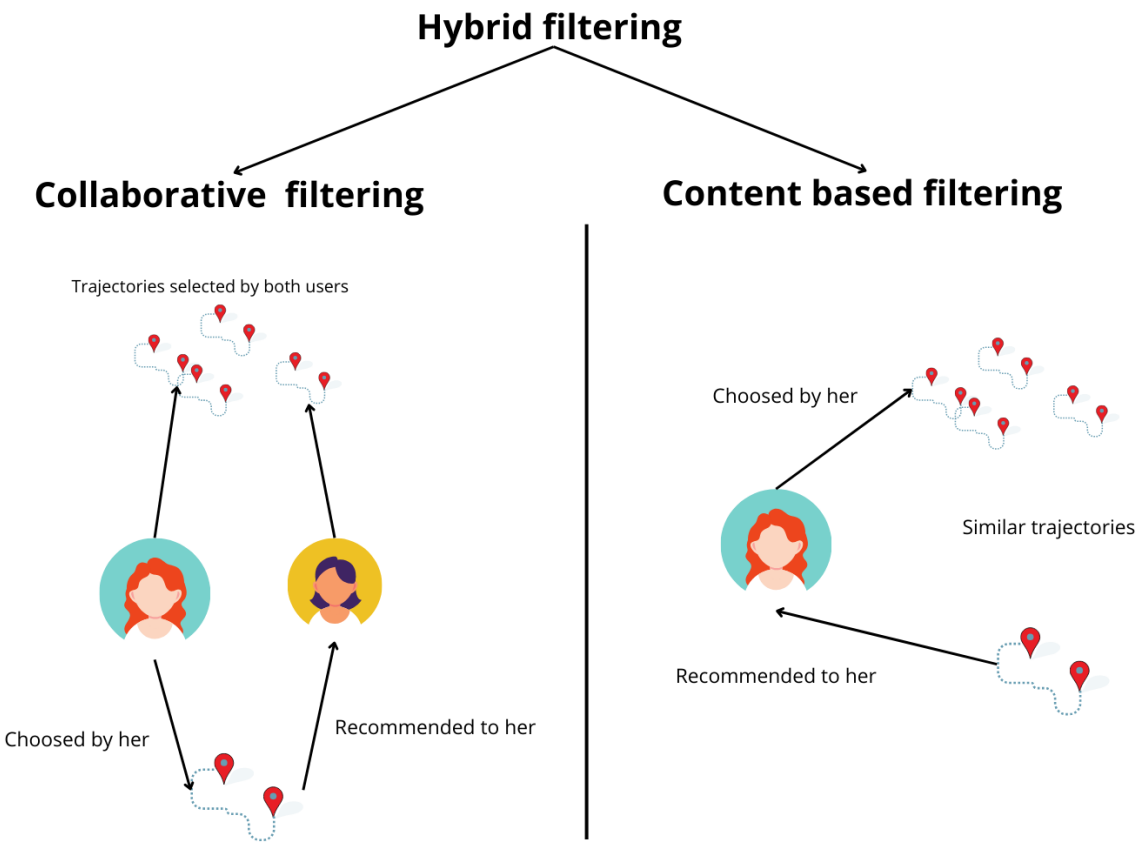

Figure 1: The three types of recommender systems: Collaborative Filtering, Content-Based Filtering, and Hybrid Systems

2.2.3 Challenges and Limitations

Recommender systems, like any other system, come with their own set of challenges and limitations. These challenges arise at various stages of the recommendation process, such as scalability during the recommendation process itself, difficulties in finding suitable datasets prior to recommendation, and issues related to fraud outside of the recommendation process Saifudin and Widiyaningtyas 2024. Let's take a brief overview of the most common limitations encountered.

Cold start Problem: One of the primary challenges faced by RS occurs when they encounter a lack of information or data absence when making predictions for new items or users Saifudin and Widiyaningtyas 2024. This issue is also referred to as the new-item problem or the first-rater problem Melville and Sindhwani 2002. While it is a concern across all types of RS, it poses a particularly significant challenge for collaborative filtering recommenders due to their reliance on historical ratings for recommendations.

Sparsity Problem : This problem arises when there's insufficient information available, as only a few items in the database are evaluated by the user. This results in sparse user-item matrices, making it challenging to find successful resemblances and provide effective recommendations Saifudin and Widiyaningtyas 2024; Melville and Sindhwani 2002. Additionally, data sparsity leads to coverage issues, where only a small percentage of items in the system can be used for recommendations Saifudin and Widiyaningtyas 2024. This poses challenges for collaborative filtering systems as it reduces the likelihood of finding similarity between users Melville and Sindhwani 2002. This problem often occurs when there's a high ratio of items to users or during the initial stages of system use. To address this issue, additional information about the topic can be considered, or assumptions can be made about how the data was created. These steps help improve the accuracy of filling in missing information Melville and Sindhwani 2002.

Scalability Problem : Scalability poses a significant challenge for many recommendation systems Saini and Singh 2023, especially as the number of users and items grows Saifudin and Widiyaningtyas 2024. Efficiently producing recommendations becomes increasingly difficult with this expansion. While developing a filtering system that can scale effectively requires careful consideration Saini and Singh 2023, there are dimension reduction techniques available to address scalability issues and accelerate recommendation processes, such as Singular Value Decomposition Saifudin and Widiyaningtyas 2024.

Running Time Problem: Improving running time is an important aspect of every research effort. According to Saifudin and Widiyaningtyas 2024 by using algorithms with both the CF and HBF abbreviations, changes can be implemented to accelerate the time taken compared to previous methods.

Accuracy Problem: It's one of the most common challenges faced by recommendation system algorithms, each of which endeavors to enhance accuracy through various research methods. The ongoing efforts to improve the accuracy of RS are both continuous and exciting Saifudin and Widiyaningtyas 2024.

Fraud Problem: The fraud problem, while less technical compared to others, remains a significant concern in RS. Sellers, for example, may manipulate RS to benefit from recommendations, often seen in commercial systems where they aim to lower the average ratings of competitors while boosting their own. CB methods are less susceptible to this issue as they are not reliant on ratings. However, CF methods often face challenges in dealing with this manipulation Melville and Sindhwani 2002.

The article Saifudin and Widiyaningtyas 2024 presents a ranking of the occurrence of the problems mentioned in the RS across all systems.

2.2.4 Evaluating Recommender System Performance

To properly assess our RS, evaluation is essential. We can gauge its effectiveness by comparing its recommendations to a predefined set of user ratings Melville and Sindhwani 2002. The article Saifudin and Widiyaningtyas 2024 outlines six key metrics commonly used for this purpose, such as :

- Weighted Precision : Measures the proportion of true positives among the items predicted as positives, considering class weights. Useful for imbalanced classes.

  Formula:

$$\text{Weighted Precision} = \frac{\sum_{i=1}^{k} \text{Precision}_i \cdot n_i}{\sum_{i=1}^{k} n_i}$$

- Recall : Measures the proportion of true positives among all actual positives. Indicates the model's ability to detect true positives.

  Formula:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- Weighted F1 Score : The harmonic mean of weighted precision and recall, considering class weights. Balances precision and recall for imbalanced data.

$$\text{Weighted F1 Score} = \frac{2 \cdot \text{Weighted Precision} \cdot \text{Weighted Recall}}{\text{Weighted Precision} + \text{Weighted Recall}}$$

- Mean Reciprocal Rank (MRR) : Measures search/recommendation effectiveness, averaged over the reciprocal ranks of the first relevant results. Useful for ranking tasks.

  Formula:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}$$

- AUC-ROC (Area Under the Receiver Operating Characteristic Curve) : Evaluates binary classification performance by measuring the area under the ROC curve, plotting true positive rate against false positive rate.

  Formula: AUC is typically calculated using numerical integration methods.

And others like Normalized Discounted Cumulative Gain (NDCG). Interestingly, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) seem to be the most frequently used, suggesting their comparable nature. However, it's worth noting that neither metric has a solid theoretical justification, often leading to a choice based on practical considerations Saifudin and Widiyaningtyas 2024.

2.2.5 Multcriteria Ratings

When discussing predictions made by RS, we often think about categorizing items based on single criteria, like taste for food or interest for books, which are examples of single-rating RS. However, in our daily lives, we often encounter situations where multiple factors need to be considered, not just one. In the article Adomavicius and Tuzhilin 2005, various solutions have been proposed, including finding Pareto optimal solutions, using a linear combination of criteria, optimizing the most important criterion, and successive optimizations of individual criteria. Despite being an intriguing subject in RS, there hasn't been extensive research in the recommender systems literature Adomavicius and Tuzhilin 2005, indicating a need for further exploration in this area.

2.3 Smart cities

Smart cities represent a major evolution in urban management, integrating information and communication technologies (ICT) to improve citizens' quality of life. This section explores the definition of smart cities, their key technologies, their challenges, and provides some recommendations.

2.3.1 Definition

The term ”Smart Cities (SC)” was initially introduced in a book published in the United States in 1992 Nguyen, Nawara, and Kashef 2024, highlighting the growing adoption of information and communication technology (ICT) in contemporary urban infrastructures. However, its significance extends beyond technological innovation; it involves the utilization of various sensors, actuators, cameras, and devices within vehicles, buildings, or houses to monitor their physical structure and integrate with other related applications, while also considering human values alongside technological advancements Nguyen, Nawara, and Kashef 2024. Despite numerous attempts to clarify its meaning, a universally agreed-upon definition remains difficult to define Lai and Cole 2022. Additionally, we find multiple approaches to building a SC Nguyen, Nawara, and Kashef 2024, with variations depending on regional contexts Lai and Cole 2022.

2.3.2 Key Technologies

SC rely on essential technologies that form the fundamental framework for their operations in the modern world. These technologies enable connectivity, data collection, and smart decision-making processes, serving as the backbone of urban development. And together they drive the development of SC, making them more sustainable, resilient, and responsive to citizens needs. Among these key technologies we find the following :

- The Internet of Things : refers to the process of connecting physical objects to the internet or alternative network protocols. These objects are equipped with sensors, processing units, and software, enabling them to interact with other devices and systems by accessing, exchanging, and processing data. The collected data is stored in databases for processing, allowing administrators to track the city’s progress and make informed decisions regarding the planning, control, and coordination of devices and systems Nguyen, Nawara, and Kashef 2024.

- Big Data : SC generate massive volumes of data from diverse sources, making the role of big data management crucial. The ”5 V’s” of big data—Volume, Variety, Velocity, Veracity, and Value—underscore the complexities and potentials linked with handling extensive and varied datasets Nguyen, Nawara, and Kashef 2024.

- Cloud computing : Cloud computing, encompassing the provision of computing services such as servers, storage, databases, networking, software, and analytics over the internet, facilitates rapid innovation, flexibility in resource allocation, and economies of scale. By embracing cloud-based applications, SC can alleviate the burden of maintaining internal databases or physical servers. Cloud providers like Amazon, Microsoft, or Google offer services and resources that are accessible and shareable across multiple devices simultaneously, thereby streamlining data management and enhancing scalability for SC administrators Nguyen, Nawara, and Kashef 2024.

- Blockchain : Blockchain, with its decentralized and immutable ledger system, plays a pivotal role in SC . By safeguarding transactions from alteration or control by any single entity, blockchain ensures security and integrity. This technology enhances the efficiency and transparency of urban governance by facilitating secure data sharing among city departments and simplifying processes such as permitting and procurement through the implementation of smart contracts. Furthermore, blockchain encourages the development of creative citizen engagement initiatives. Ultimately, blockchain holds the promise of transforming smart cities into more resilient, inclusive, and sustainable urban environments Nguyen, Nawara, and Kashef 2024.

- Machine learning : Machine learning involves the use of algorithms and statistical models to enable SC applications, such as data analytics, sensor information fusion, anomaly detection, and emotion recognition. It plays a crucial role in optimizing SC operations, leading to enhanced efficiency, effectiveness, and overall performance Fadhel et al. 2024.

2.3.3 Challenges

SC use advanced technology to address various challenges such as traffic congestion, environmental pollution, and energy efficiency. However, in this process, they encounter many challenges, including the following:

- Smart Sensors in Smart Cities: Sensors are vital components in the creation of SC, as they measure the physical attributes of devices and enable diverse functionalities. However, integrating sensors into SC infrastructure encounters challenges such as safety and security management, privacy issues, and energy constraints in wireless network sensors Nguyen, Nawara, and Kashef 2024.

- Security and Privacy: SC gather lots of data, including details about citizens, which needs to be kept safe from unauthorized access or tampering. However, this data can be at risk of cybercrime as it moves through different stages. Challenges include preventing unauthorized access to stored data and ensuring data streams are not disrupted during transmission. It's important to protect citizens private information Nguyen, Nawara, and Kashef 2024.

- The cost of SC : The cost of implementing SC is a significant concern, affecting both the initial development expenses and ongoing operational costs. Development costs encompass the implementation of SC initiatives, including planning strategies to meet citizens needs and installing various devices, sensors, and software, which can be both time-consuming and expensive. Operational costs relate to the ongoing maintenance needed to ensure SC operate efficiently throughout their lifespan. It's crucial to allocate sufficient resources to maintain smart facilities and ensure uninterrupted benefits for citizens Nguyen, Nawara, and Kashef 2024.

- Data extraction and management : Data extraction from various sources, alongside the management of urban data, presents labor-intensive tasks, particularly due to the diverse sources and complex data types involved. These challenges underscore the essentiality of implementing efficient data management strategies to support the development of SC Fadhel et al. 2024.

- Energy consumption : the challenge of energy consumption in IoT systems remains pertinent. These systems, integral to the functioning of SC, involve the collection, analysis, and transmission of large quantities of data. This process, essential for various SC functionalities, demands considerable resources such as storage capacity, cloud computing, and wide bandwidth, leading to significant energy consumption. Additionally, the energy needs of the sensing devices used in SC infrastructure further contribute to this challenge Omrany et al. 2024.

- Scalability, adaptability, and reliability : another significant challenge in the development of SC. These systems must seamlessly adjust to evolving user demands, incorporate new services and devices, and maintain consistent performance. Additionally, they need to evolve to meet the dynamic requirements of SC, accommodating a diverse range of devices with varying capabilities, all while ensuring the robustness and reliability of IoT systems to deliver high-quality service despite scaling up to integrate more devices and services Omrany et al. 2024.

- Ethical concerns : Ethical considerations in IoT applications for SC involve a range of challenges such as intellectual property rights and data privacy. Five primary issues have been identified in this context: informed consent, privacy, information security, physical safety, and trust Omrany et al. 2024.

2.3.4 Recommendation

To address the many challenges of SC, several strategies can be put in place. Article Fadhel et al. 2024 offers various recommendations to make the functioning of smart cities more efficient. Here are some of them:

- Utilizing a machine learning-driven intrusion detection system to bolster communication security within mobile cloud computing environments.
- Utilizing supervised machine learning methods to harmonize diverse data sources, enhancing decision-making efficacy in urban planning.
- Suggesting the adoption of location-based filtering and global descriptor similarity-based ranking methods for recognition tasks to improve recognition outcomes, surpassing the effectiveness of local descriptor-based re-ranking approaches.
- Taking into account spatial-temporal correlations among various local regions within a city to attain precise prediction results, leading to a deeper understanding of urban dynamics and enhancing prediction accuracy and reliability.
- Adopting advanced techniques like dual-attention deep reinforcement learning, lifelong learning, and multi-granularity fusion techniques in IIoT environments to enhance resource utilization, system performance, and accuracy.
- Suggesting the implementation of explainable recommender systems to increase user acceptance rates by providing clear explanations for energy-saving actions. This strategy aids in reducing energy wastage and promoting sustainability initiatives.

2.4 Multi-modal transportation

In the realm of SC, transportation has experienced a significant revolution. Whether it's in private vehicles with systems for selecting optimal trajectories or finding available parking spaces, or in public transportation, be it unimodal or multimodal, where options range from selecting the most appropriate mode of transport to identifying stops with minimal crowds or the closest proximity. These advancements continue to evolve due to the numerous technologies and advantages facilitated by SC initiatives.

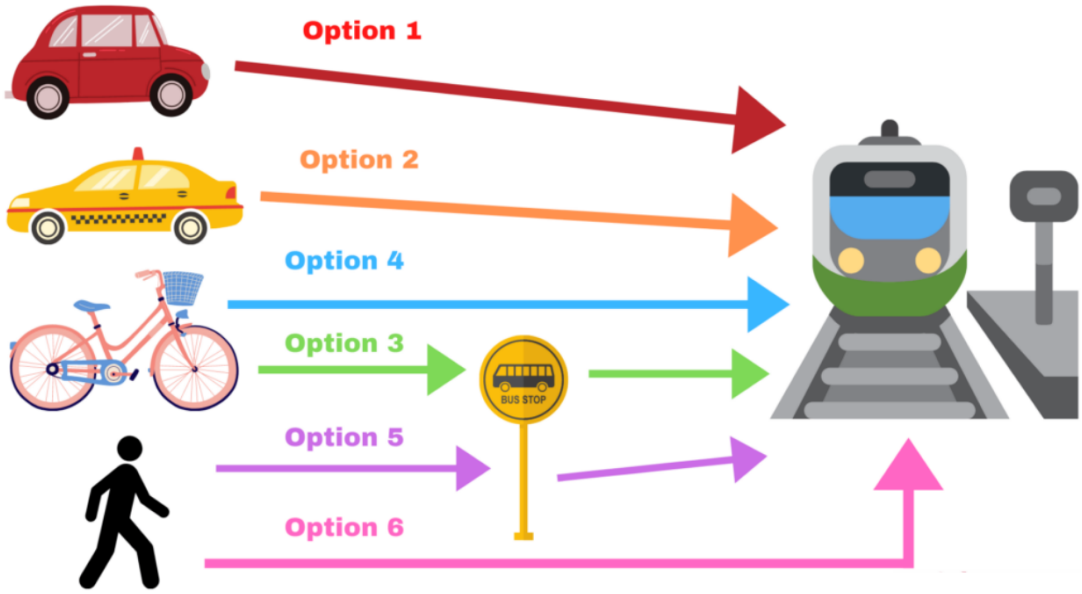

Figure 2: Multimodal transportation

2.4.1 Definition

The evolution of multimodal transportation systems is a key indicator of the advancement of smart transportation. These systems are defined in various ways, such as "the shipment of cargo and the movement of people involving more than one mode of transport during a single seamless journey," or "movement in which two or different transport modes are linked end to end in order to move freight and/or people from point of origin to point of destination." Another definition is "the combination of modes, usually ship, truck, or rail, to transport freight." Dua and Sinha 2015 All these definitions simply refer to systems that use a combination of different transportation methods to travel from a starting point to a destination.

2.4.2 Transition from Conventional to Multimodal Transportation Planning

Transitioning to a multimodal transportation approach involves addressing various issues and adapting organizational structures to support diverse transportation modes. Conventional planning primarily focuses on automobile travel, promoting high vehicle ownership and low land-use density, while often neglecting broader impacts such as public health and environmental concerns. This approach uses models designed for highway evaluation, emphasizing speed and congestion reduction. In contrast, multimodal planning integrates various transportation modes, encouraging higher land-use density and mixed-use development. It values modal diversity and social equity, requiring more complex analysis tools to assess the quality and interconnectivity of different transportation modes. Effective organizational structures are crucial for coordinating efforts and achieving stability. Most State Departments of Transportation (DOTs) use a divisional structure to manage functions by transportation mode. To incorporate multimodal approaches, these structures must evolve, considering non-highway modes alongside traditional highway functions. This evolution requires reengineered processes, skilled staffing, and advanced analytical tools Smith 2013.

2.4.3 Best Practices in Multimodal Transportation Planning

The evolution of multimodal transportation planning has led states to adopt best practices that emphasize equitable consideration of various transportation modes, organizational restructuring, and enhanced public involvement. Successful approaches include fostering strong modal advocacy within departments and implementing flexible funding mechanisms. States have also improved methods for assessing transportation needs and selecting projects, with a focus on integrating different modes through simultaneous analysis and advanced planning models. Public involvement is crucial, with efforts to engage stakeholders and users of the transportation system, ensuring their input shapes the planning process. Additionally, the use of performance measures, such as accessibility, travel time reduction, and multimodal options, helps evaluate the effectiveness of transportation systems and supports data-driven decision-making. These comprehensive strategies underscore the importance of evolving organizational structures and enhancing coordination to achieve an efficient and interconnected multimodal transportation network Smith 2013.

2.4.4 Challenges

Statewide multimodal transportation planning faces several key challenges. First, there is a growing need for performance-based planning, where transportation agencies are accountable for measurable results, not just outputs. This requires developing neutral performance measures using readily available data. Second, greater public and stakeholder involvement is essential, ensuring that diverse interests, including minority and low-income populations, are fairly represented. Another major challenge is integrating advanced technology, which can revolutionize data processing and analysis but needs better tools and methods. Environmental sustainability and fair distribution of transportation benefits are also becoming important. Additionally, coordinating planning efforts across local, regional, and multistate levels is crucial to avoid political gridlock and ensure comprehensive solutions. Finally, recruiting, training, and retaining skilled professionals with the necessary technical expertise remains a significant hurdle. Pedersen Chairman: Neil J. Pedersen

2.5 Conclusion

In conclusion, recommendation systems, smart cities leveraging ICT, and multi-modal transportation contribute crucially to shaping modern urban environments, offering enhanced decision-making, sustainable service optimization, and efficient mobility solutions, respectively, towards smarter cities for the future.

Chapter 3 : State of the Art

With the rising popularity of Multi-modal transportation recommendations (MTR), numerous studies have been conducted, each with its unique approach aiming to achieve improved results. Researchers are actively contributing to this field, incorporating their own perspectives into their work. Below are some of the most notable recent studies conducted in this area.

3.1 Comparative Table

The table below provides a comprehensive summary of proposed Multimodal Transport Recommendation Systems, highlighting key studies and their methodologies. Each entry includes details such as authors, publication year, title, dataset used, recommendation methods employed, trajectory generation techniques, and outcomes achieved. This table serves as a structured overview, showcasing how different approaches contribute to enhancing decision-making and optimizing trajectories within multimodal transport networks.

Table 1: Summary of Multimodal Transport Recommendation Systems

Approach proposed in	**Focus**	**Used dataset**	**Used methods**	**Evaluation Results**
Campigotto et al. 2016	Personalized and situation aware.	Travel choices made by 40 participants from diverse demographic groups.	Bayesian learning. Transfer learning. Stated preference survey.	Mean predictive accuracy: 0.723
H. Liu, Li, et al. 2019	Personalized recommendations for multimodal transportation trajectories, accommodating various modes.	Travel events recorded from map queries and user feedbacks on the Baidu Map in Beijing and Shanghai.	Anchor embedding technique. Joint representation learning framework.	NDCG: 0.893 PREC: 0.700 REC: 0.770 F1: 0.711

Continued on next page

Approach proposed in	Focus	Used dataset	Used methods	Evaluation Results
Abedalla et al. 2019	Recommend the most appropriate transport mode to each user.	The public dataset from the Context Aware Multi-Modal Transportation Recommendation challenge, released on April 10th, 2019, which includes historical user behavior data from Baidu Map.	Convolutional Neural Network. Gradient Boosted Decision Trees.	Weighted F1-score: 0.68898702
H. Liu, Tong, et al. 2020	Multimodal transportation plan that adapts to various situational contexts.	Baidu Maps data, covering user behavior data from September to November 2018.	Light-weight gradient boosting decision tree . Multi-task wide and deep learning .	F1: 0.414 PRE: 0.286 REC: 0.748 NDCG: 0.805
Y. Liu et al. 2021	Various combinations, while extracting features related to user preference, mode accessibility, and location popularity.	Real travel mode choice data from Baidu Maps, covering around 2 million historical recommendations and user behaviors.	Bipartite graph. Graph embedding technique.	Weighted F1 between 0.6041 and 0.7936
Xu et al. 2023	Recommend the most suitable transport mode for users.	Navigation data from Baidu Maps, including user attributes, origin, destination, and transport mode data.	Heterogeneous graph Attention. Graph neural networks. Graph embedding.	F1: 0.7566 PRE: 0.797 REC: 0.7848 NDCG: 0.9523

Continued on next page

Approach proposed in	Focus	Used dataset	Used methods	Evaluation Results
H. Liu, Han, et al. 2023	Unified route representation for multimodal transportation recommendation, leveraging spatiotemporal dependencies and semantic coherence of historical route.	Real-world transportation data from Beijing and Shanghai, provided by a leading navigation application.	Graph-based contextual encoder. Spatiotemporal pretraining strategies.	Hit between 0.8716 and 0.9235 NDCG between 0.8735 and 0.8983
Yang et al. 2024	Outputs a K shortest path planning framework for intercity door-to-door travel.	Urban public transport data, including bus and metro lignes information, railway data detailing train schedules and station locations, and transfer data indicating transfer times between stations.	Algorithm inspired by the ripple-spreading phenomenon.	Average Path-finding time/s: 80 s Minimum Path-finding time/s: 0.04 s Maximum Path-finding time/s: 530 s

3.2 Synthesis

After reviewing the various studies presented in the current chapter, it is evident that these works extensively utilize advanced machine learning and deep learning techniques to enhance recommendation systems. For instance, convolutional neural networks (CNN) and gradient-boosted decision trees (GBDT) are commonly employed to handle complex data and improve model performance. Additionally, many studies incorporate graph-based methodologies to model the relationships between users, origin-destination (OD) pairs, and transport modes. This approach is exemplified in the use of bipartite graphs Y. Liu et al. 2021 and heterogeneous graphs Xu et al. 2023.

However, each study adopts a different approach in its recommendation and trajectory generation methods. For recommendation generation, the following approaches were identified:

- Bayesian Learning and Transfer Learning: Utilized in Campigotto et al. 2016 to continuously update user profiles and apply pre-existing knowledge for trajectory recommendations.
- Optimization Modeling: Employed in Hao and Yue 2016 who use dynamic programming for optimizing container multimodal transport systems.
- Joint Representation Learning: Implemented in H. Liu, Li, et al. 2019 who employ a multi-modal transportation graph to learn representations for recommendation.
- Multi-Task Deep Learning Systems: Utilized in H. Liu, Tong, et al. 2020 who use a combination of light-weight GBDT and multi-task wide and deep learning (MTWDL) for context-aware recommendations.

For trajectory generation, the following methods were identified:

- Graph Embedding Techniques: Constructed bipartite graphs and transformed nodes into feature vectors as done in Y. Liu et al. 2021.
- Heterogeneous Graph Attention Networks: Utilized a hierarchical attention mechanism to generate node embeddings as done in Xu et al. 2023.
- Time-Dependent Multi-View Transportation Graphs: Transformed transportation networks into multi-view graphs and employed a graph-based contextual encoder as done in H. Liu, Han, et al. 2023.
- Ripple-Spreading Algorithm: Utilized an improved ripple-spreading algorithm to find the K shortest paths in intercity multimodal transport networks as done in Yang et al. 2024.

The datasets used also vary significantly from one study to another, with some studies focusing on city-specific data while others utilize broader datasets covering multiple cities and transport modes. Additionally, many datasets are confidential.

The outputs of these systems also differ, aligning with the objectives of the respective applications:

- Personalization and Context Awareness: Achieved through dynamic user profiles and consideration of situational factors like weather H. Liu, Tong, et al. 2020; Campigotto et al. 2016.

- Optimal Transport Strategies: Effective strategies for container transport systems and other multimodal transport scenarios Hao and Yue 2016.
- Unified transport network Representation: Leveraging spatiotemporal dependencies for robust recommendations H. Liu, Han, et al. 2023.
- Efficiency in Computation: Improved algorithms for faster and optimal path planning Yang et al. 2024.

3.3 Conclusion

In conclusion, while all the studies aim to improve multimodal transport recommendation systems, they differ significantly in their methodologies, data utilization, and specific outcomes. The common thread across these studies is the use of advanced machine learning techniques and the integration of diverse data sources to provide accurate, personalized, and context-aware recommendations. These similarities and differences highlight the evolving landscape of multimodal transport recommendation systems and the continuous efforts to enhance their performance and user satisfaction.

Chapter 4 : Contribution

4.1 Key Concepts

In this section, we introduce the key terms and concepts essential for our study. These foundational elements provide the necessary theoretical framework to explore the core aspects addressed herein.

- *Stations or stops* : Stations are the constituent elements of our lines, referring to all public transport stations within the transport network.
- *Lines* : Lines refer to the different routes of the transport network, regardless of the mode of transport, composed of a set of stations arranged in a specific order.
- *Trajectories or Routes* : Trajectories represent the sequence of lines, each encompassing all the stops through which one passes, thereby forming a continuous link from the starting point to the destination.

4.2 Introduction

In this chapter, we detail our contribution to the domain of multimodal transportation recommendation systems.. Our proposed approach integrates PSO for trajectory generation and RankedNet for trajectory recommendation. This approach addresses the challenge of efficiently guiding users to their destinations, taking into account factors such as travel time, cost, and user preferences regarding transportation modes. We provide a detailed breakdown of each step in our approach, including capturing origin-destination pairs, generating optimal paths using PSO, calculating trajectory attributes for recommendation, and utilizing RankedNet for trajectory recommendation. Through this comprehensive approach, our goal is to enhance the effectiveness and accuracy of trajectory recommendation systems, thereby improving the overall user experience in navigating transportation networks.

4.3 Proposed approach

Recognizing that in certain cities, selecting the optimal mode of transportation from one location to another is no longer straightforward due to the array of available options and numerous potential trajectories, travelers may find themselves unsure when deciding on their itinerary. Therefore, our system offers users the optimal trajectory utilizing public transportation, whether it involves a single mode or a combination, tailored to their preferences regarding cost, time, and preferred mode of travel.

To offer this recommendation, our system follows essential steps outlined as follows:

- Capture the user's starting and destination locations in the format (longitude, latitude), then add these two points as the starting and ending nodes, respectively, to our transportation network.
- Generate some optimal paths in terms of distance between the starting node and the destination node using PSO.
- Once the optimal trajectories have been selected, calculate their necessary attributes for the recommendation, including the percentage of each mode of transportation, the total time, and the total cost.

- Proceed to recommend the best trajectory using RankNet. Subsequently, present the user with the top recommendations, indicating the recommended one and providing details of the trajectories.

These essential steps are depicted in the figure 3.

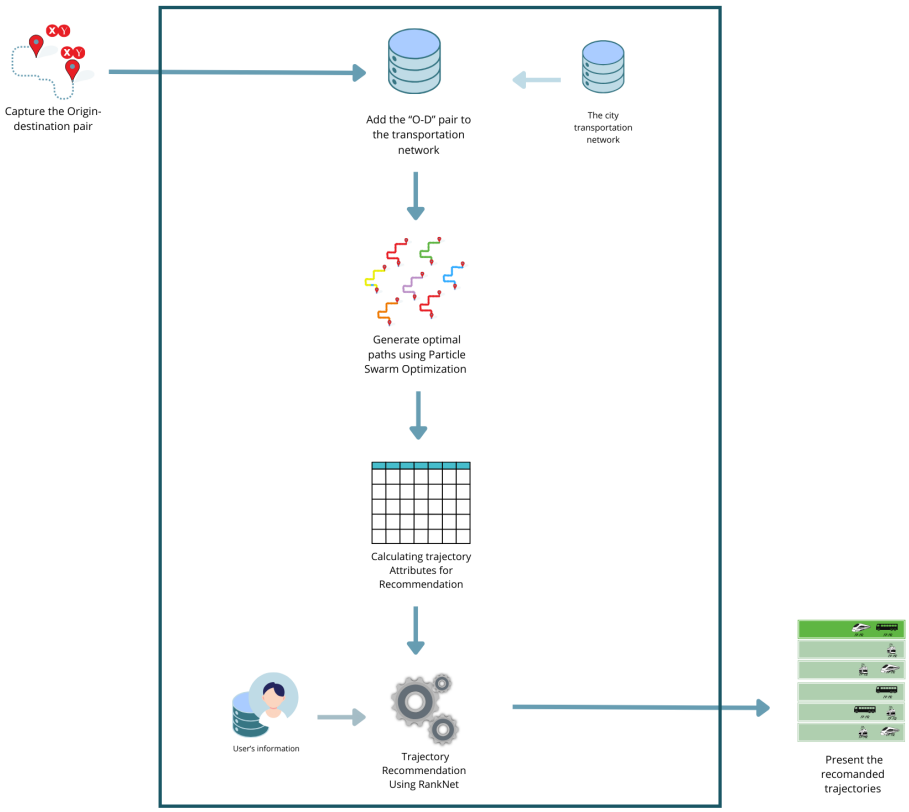

Figure 3: The approach essentiel steps

4.4 Breakdown of Each Step

We will now outline each step to clarify its role and significance in the process :

4.4.1 Capture the Origin-destination pair

As soon as the request begins, the user indicates on the city map their starting and destination points. From their input, we retrieve the coordinates of these two points, namely their longitude and latitude. Next, we establish links between each of these two points and the nearest stations to them. Our goal is to select nearby stations in the network and ensuring that we select at least one station. To achieve this, we will calculate the distance of all stations from our departure and destination locations, sort these distances,

and select the first stations. We will stop when the distance becomes significantly larger than the average distance of the stations already selected. We even create a walking link between them if they are close enough. We then designate these two new nodes in our study as the starting and destination nodes. And we apply a sort of gridding as shown in the figure 4 to only keep the stations that are present in the box delimited by the higher and lower longitudes and latitudes of the origin-destination pair. This step is essential to reduce the search space. The detailed steps are elaborated in the algorithm 1, and the Figure 5 illustrates the establishment of these links where we observe that the algorithm will select the stations that are close to the departure and arrival locations, regardless of the type of station.

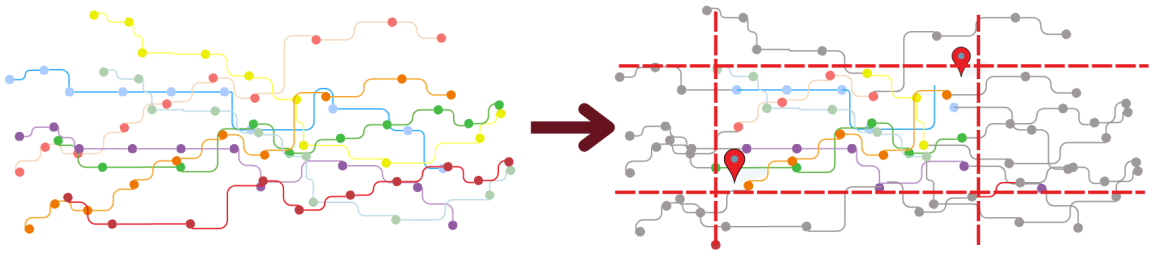

Figure 4: Apply grids to form a box around the origin and destination

Algorithm 1: Neighbour stations selection

Input: Origin-Destination (O-D) coordinates, Transportation network
Output: Two lists of selected stations

Start;
Calculate the distance of each station from the departure station O and the destination station D;
Sort the stations by distance from the departure station;
Initialize an empty list of selected stations (*selected_stations_O*);
Initialize a variable *Avr* as the maximum distance initially that will contain the average distance of the added stations;
while *the distance of the next station is not significantly larger than Avr* **do**
    Add the new stations to *selected_stations_O*;
    Update the value of *Avr* as the new average of distances of the selected stations;
end
Sort the stations by distance from the destination station;
Initialize an empty list of selected stations (*selected_stations_D*);
Initialize *Avr* as the maximum distance initially;
while *the distance of the next station is not significantly larger than Avr* **do**
    Add the new stations to *selected_stations_D*;
    Update the value of *Avr* as the new average of distances of the selected stations;
end
Return the two lists: *selected_stations_O* and *selected_stations_D*;

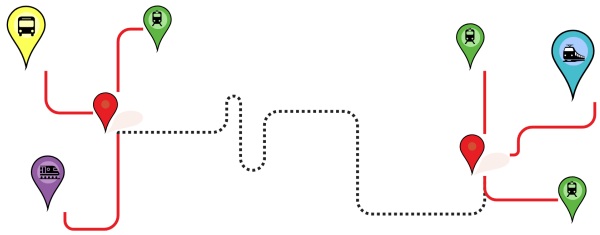

Figure 5: Link creation between the origin-destination pair and the closest stations to them

4.4.2 Generate possible trajectories using PSO

To calculate all possible trajectories, our network is stored as a graph G=(U,V), where each station is represented by a node Ui. These nodes capture the station's location and line ID. Connections between stations are represented by edges Eij. These edges signify the adjacency of stations on the same line. And are characterized by the cost of traveling between stations Cij, the time it takes to travel between them Tij, and the identifier of the line that connects them. This graph structure that we illustrated in the figure 6 enables us to explore various trajectories within the transportation network efficiently where line is represented by a color, and we can clearly see the arc link between each successive station on the same line in the network. Finally, the representation of the trajectories that will be generated is shown as a series of arcs between the departure point and the arrival point. The generated trajectories will be a list of nodes that need to be visited in that order to reach the destination Trj = [U1, U2, . . . , Un].

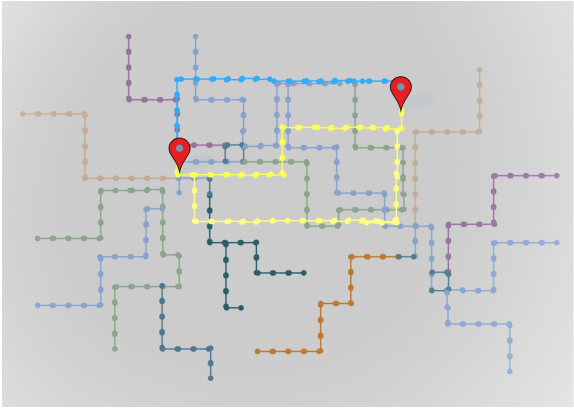

Figure 6: The transportation network representation

We then proceed to the step of generating possible trajectories. For this purpose, we have chosen to work with PSO, which provides an optimal method for our study. Representing a city's transportation

network as a graph, where stations are nodes and links between stations are edges, the search for possible trajectories quickly becomes exponential. To optimize this search, we have employed the following approach using PSO.

The principle of PSO : PSO is an optimization algorithm inspired by the social behavior of birds flocking or fish schooling. In PSO, a group of particles (potential solutions) moves around the search space to find the optimal solution. Each particle adjusts its position based on its own experience and the experience of neighboring particles. The movement of each particle is influenced by its best-known position and the best-known positions of the swarm as shown in the figure 7, with positions being adjusted using velocity equations. This allows the particles to converge towards the optimal solution over time. We chose to work with PSO because swarm intelligence is widely used for its simplicity and effectiveness in solving complex optimization problems.

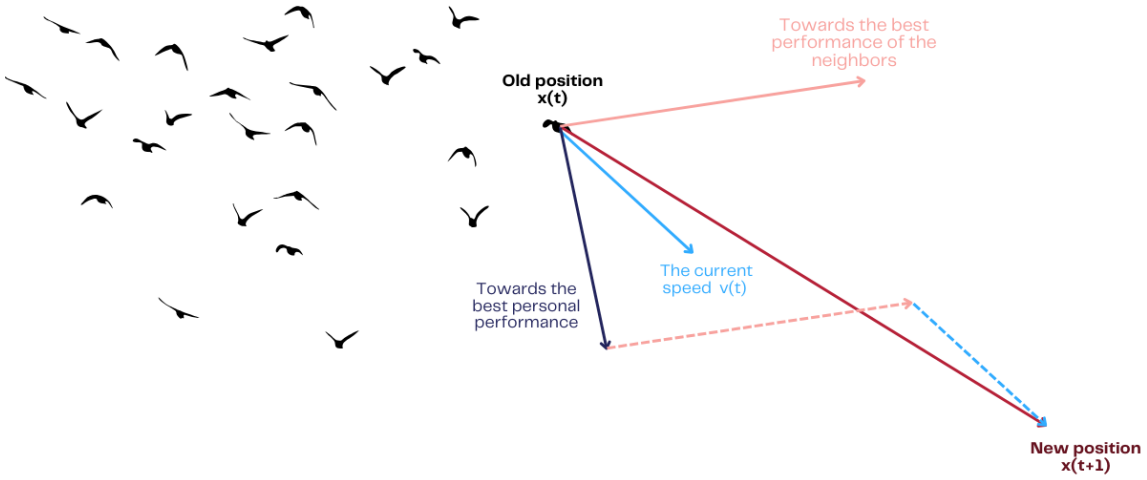

Figure 7: Practile Swarm Optimization principle

Update Velocity:

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot (pbest_i^t - x_i^t) + c_2 \cdot r_2 \cdot (gbest^t - x_i^t) \tag{1}$$

Where:

- v_i^{t+1}: the new velocity of particle i at iteration $t+1$.
- w: the inertia weight.
- v_i^t: the current velocity of particle i at iteration t.
- c_1 and c_2: the learning coefficients.
- r_1 and r_2: random numbers between 0 and 1.
- $pbest_i^t$: the best position attained by particle i until iteration t.

- x_i^t: the current position of particle i at iteration t.
- $gbest^t$: the best position attained by the swarm until iteration t.

Update Position:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2}$$

Where x_i^{t+1} is the new position that should take the particle i at iteration $t+1$. However, instead of maintaining these new positions, the nearest stations to these positions were selected as the new positions, since it is not possible to position directly in space. Specifically, we do not simply choose the nearest station; like illustrated in the figure 8 rather, we select the station closest in the direction of the velocity vector. This ensures that the trajectory aligns with the direction indicated by the velocity vector, optimizing the movement towards the destination.

Our adaptation of the PSO : To adapt the algorithm to our study, we made the following adjustments.

In our case, the optimization function represents the distance between the current point and the destination.

For initialization, the particles were positioned at stations close to the origin rather than being initialized randomly in space. At each iteration, the new positions of the particles were calculated using the velocity that also been calculated using the PSO equations.

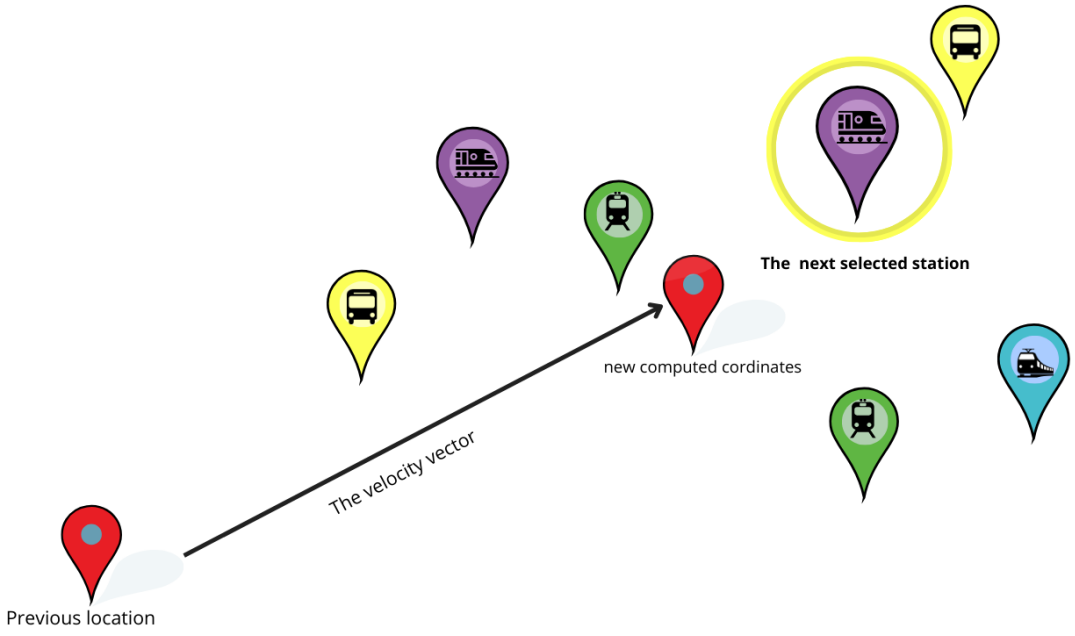

Figure 8: Next station selection

Starting from the initialization, at each iteration, we add to each trajectory the new station taken by its particle if it does not move away from the destination. Finally, we return the 8 trajectories whose last station is closest to the destination. And we only return thoose who are not duplicated. All the steps involved in generating trajectories using our adaptation of PSO are detailed in algorithm 2.

Algorithm 2: Generating initial trajectories using PSO

Input: Origin and destination identifier, List of the neighbours of the origin and the destination, SwarmSize, MaxIterations, w, c_1, c_2, The transportation network

Output: List of best trajectories found

```
Start;
Initialize a swarm of particles at the origin neighbors;
For each particle, compute its distance from the destination and set each one as the pbest of the particles;
Sort the distances and store the smallest 8 as the most optimal ones and set the smallest one as the gbest;
while iter ¡ MaxIterations or at least 3 of the optimals are less than 1km from the destination do
    for each particle in the swarm do
        Update the velocity of the particle using the PSO equations;
        Update the position of the particle based on the new velocity;
        Update the particle's best-known position (pbest) if the fitness improves;
        Update the list of the 8 best particles if the new pbest is better than the 8th one in the list;
        Update gbest if the particle's fitness is better than the current gbest;
        if the particle has reached the destination then
            Remove it from the swarm and do not work on it in next iterations;
        end
    end
end
Return the list of best trajectories found so far as the solution to the trajectories finding problem;
End
```

Since the result is a list of eight trajectories, each represented by a sequence of stations from the starting point to the destination, it is not certain that there is a link between each pair of successive stations in these trajectories. Additional treatments shown in the figure 9 had to be added to these stations. These treatments are as follows:

- First, the destination was added as the final station in the trajectories that had not reached the destination yet.
- Then, combinations of pairs belonging to the same line were searched for. If they existed, then intermediate stations were removed.
- Next, links between each pair of successive stations were searched for as follows:
  - First, I checked if they belonged to the same line; if so, their link was marked as 'same line.'

- If they did not belong to the same line, then I searched if there was a station that belonged to both a line passing through the first station and another passing through the second station. If such stations existed, then their link was marked as 'line intersection.'
- If no link from the two previous types was found, then the link between them was marked as 'walk.'

- Then, the intermediate stations were added in both 'intersection of station' and 'same line' types. Duplicated stations have been removed as well, and if there was any cycle in the trajectory, it was also eliminated. This was done to finally obtain the complete trajectories and a dictionary describing all the lines, each with its stations, to go from the origin to the destination.

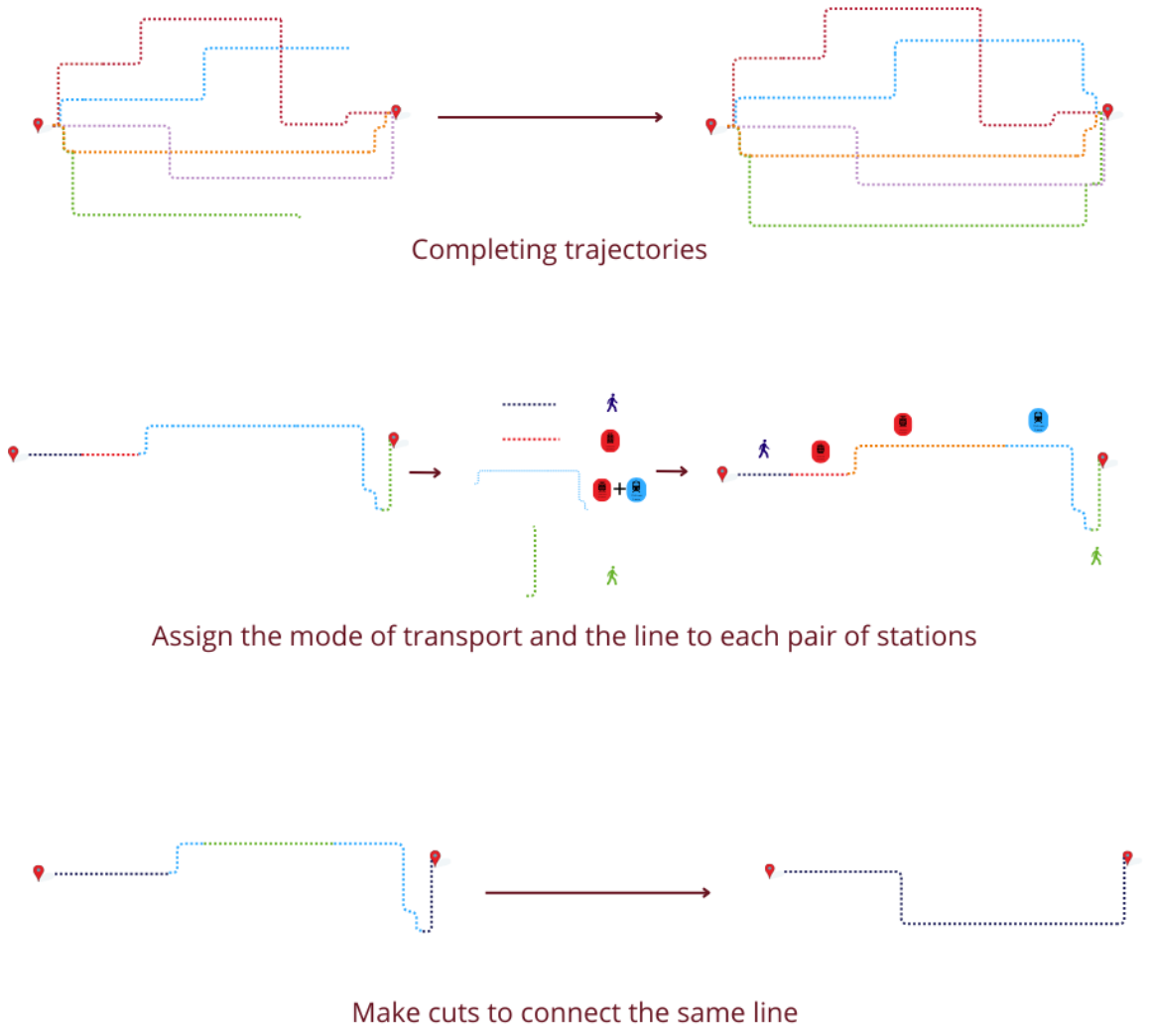

Figure 9: Post processing steps

All these post-processing steps are summarized in algorithm 30.

Algorithm 3: Finalizing the trajectories

```
Input: Origin and destination identifier, List of the 8 best trajectories, The transportation network
Output: List of best trajectories found
Start;
Remove from the 8 trajectories the duplicated ones;
for each trajectory of the remaining ones do
    repeat
        for each pair of stations in the trajectory do
            if they belong to the same line then
                Remove all intermediate stations between them;
            end
        end
    until no more modifications are made;
    for each successive pair of nodes do
        if they belong to the same line then
            Mark their link as 'same line';
        end
        else
            Search for a station that belongs to both a line passing through the first station and
              another passing through the second station;
            if such stations exist then
                Mark their link as 'line intersection';
            end
            else
                Mark the link between them as 'walk';
            end
        end
    end
    Combine the successive nodes that have the same line;
    Add the intermediate stations between two stations connected by a 'same line' or 'line
      intersection' link;
    Organize the trajectory in a dictionary of lines of the trajectory, each one with the type of
      transportation, the line id and the list of stations;
end
Return the final trajectories;
End
```

4.4.3 Calculating trajectory Attributes for the recommendation phase

In this step, we essentially calculate, for each possible trajectory, the numerical value of the criteria that matter to us during the recommendation phase.

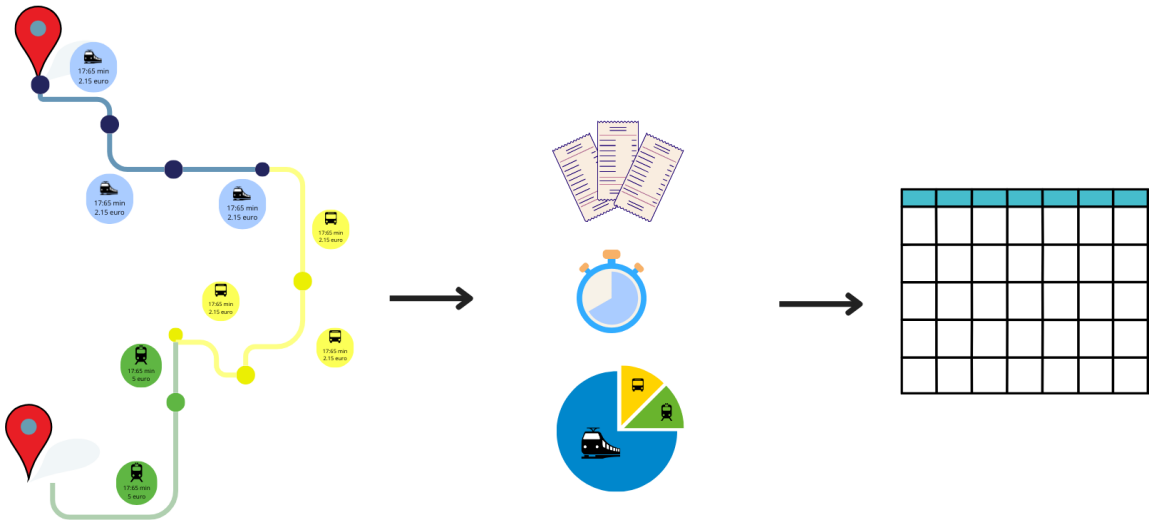

Figure 10: Trajectories attribute computation.

for this, we need to compute first the proportion of time that each line takes from the entire trajectory, for that we will need the entire time of the trajectory :

Total time of the trajectory:

$$\text{Total\_time} = \sum_{k=0}^{n} T_k \tag{3}$$

Where:

- T_k is the time spent on ligne k.
- n is the total number of lignes in a trajectory.

Time spent in each type of transportation: Then we need to compute the proportion of time for each ligne.

$$\text{Time\_in\_type} = \sum_{k=0}^{n} \text{Type}_k \cdot t_k \tag{4}$$

Where:

- t_k is the proportion of time spent on edge k.
- Type_k is a binary variable that takes 1 if we took that type of transportation mode on that ligne, otherwise it is set to 0.

- n is the total number of lignes in a trajectory.

Proportion of each type of transportation:

$$\text{Proportion\_of\_type} = \frac{\text{Time\_in\_type}}{\text{Tota\_time}} \tag{5}$$

Where:

- Time_in_type is the time spent in each type of transportation.
- Tota_time is the total time of the trajectory.

Cost of the whole trajectory: we also need the cost of the whole trajectory.

$$\text{Cost} = \sum_{k=0}^{n} C_k \tag{6}$$

Where:

- C_k is the cost on ligne k.
- n is the total number of lignes in a trajectory.

finally we will obtain a data-frame where each line contains the above information about a generated trajectory.

4.4.4 Trajectory Recommendation Using RankNet

The entire process of sorting trajectories is presented in the algorithm. For the recommendation, we implemented a machine learning solution using a neural network called RankNet to rank our trajectories. The model takes as input the list of trajectories generated for a specific origin-destination pair, each represented by the attributes counted in the previous step. As a result, the model returns the same list of trajectories but ordered according to the trained model. To evaluate the model, we divided the initial dataset into training and testing data, with 80% allocated for training and 20% for testing.

RankNet principle : RankNet is a pairwise learning-to-rank algorithm developed by Microsoft Research, primarily used in information retrieval and search engines. It uses a neural network to predict the rank order of items by comparing pairs of items and learning their relative preferences. The network is trained using a gradient descent method to minimize the loss function, which measures the difference between the predicted and actual rankings. During training, RankNet adjusts the weights of the neural network to improve the accuracy of its predictions. This method allows RankNet to effectively handle large-scale ranking problems and produce highly relevant search results.

The process that s shown in the figure 11 begins by creating pairs for each origin-destination pair. For each pair, multiple trajectories are generated, and among these, only one is selected. For each non-selected trajectory, we create a pair where the selected trajectory is the first element and the non-selected trajectory is the second element. These pairs are then integrated into DataLoader objects to enable efficient batch processing.

We developed a RankNet model comprising three fully connected layers to predict scores for input feature vectors representing each trajectory. To evaluate the model's performance and facilitate training, we use a custom loss function specifically designed for RankNet.

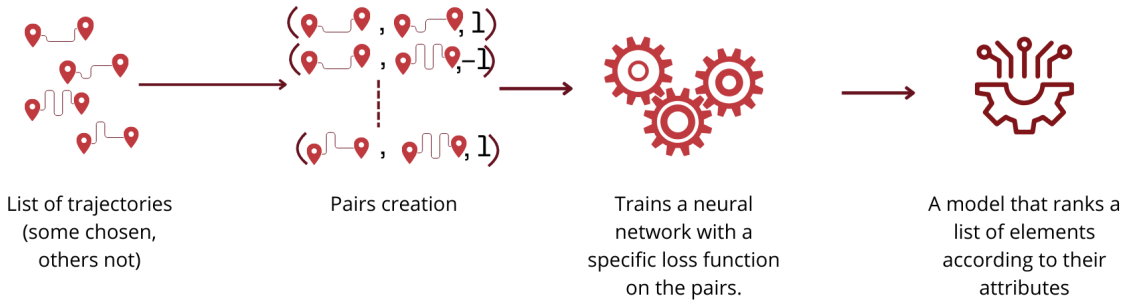

Figure 11: The principe of Ranknet

The entire process of sorting trajectories is presented in algorithm 29.

Algorithm 4: Training and Evaluating a RankNet Model

Input: Initial trajectories dataset, Origin-Destination pairs, Number of epochs
Output: Trained RankNet model

```
Step 1: Split data into training and testing sets;
train_data, test_data = split_data(initial_dataset);
Step 2: Create pairs of trajectories for each origin-destination pair;
pairs = [];
for each od_pair in origin_destinations do
    trajectories = retrieve_trajectories(od_pair,the dataset);
    selected_trajectory = select_trajectory(trajectories);
    for each traj in trajectories do
        if traj ≠ selected_trajectory then
            pairs.append((selected_trajectory, traj));
        end
    end
end
Step 3: Create DataLoader for efficient batch processing;
data_loader = DataLoader(pairs);
Step 4: Define RankNet model with three fully connected layers;
model = RankNetModel(layers=3);
Step 5: Train the model using a custom RankNet loss function;
for each epoch in range(num_epochs) do
    for each batch in data_loader do
        selected_traj, non_selected_traj = batch;
        score1 = model(selected_traj);
        score2 = model(non_selected_traj);
        loss = ranknet_loss(score1, score2);
        optimize_model(loss);
    end
end
Step 6: Evaluate model performance on the test data;
evaluate_model(model, test_data);
```

4.5 The main algorithm

Algorithm 5 outlines all the steps from capturing the origin and destination to displaying the recommendations.

Algorithm 5: Recommending the best trajectories

Input: Origin-Destination (O-D) coordinates
Output: Ordered list of recommended trajectories with the user's favourite predicted one

1 **Start**;
2 Add the O and D to our network as the departure and destination nodes;
3 Add links between them and the nearest stations using the Neighbour Station Selection Algorithm;
4 Select the 8 optimal trajectories using the Practice Swarm Optimization Algorithm;
5 Update and finalize the trajectories and return the dictionary of lines of each one using the finalizing trajectories algorithm;
6 Use the returned dictionary to make a dataframe of attributes with a trajectory in each line;
7 Use the RankNet model to recommend one of the generated trajectories represented in the generated dataframe to the user and display the other options;
8 Take the final choice of the user and update their profile for further recommendations;
9 **End**

4.6 Conclusion

In summary, in this chapter we introduced a novel approach that integrates PSO and RankNet for trajectory recommendation in multimodal transportation systems. By considering factors such as travel time, cost, and user preferences, our approach aims to enhance the effectiveness and accuracy of trajectory recommendation systems, ultimately improving the overall user experience in navigating transportation networks.

Chapter 5 : Realization and evaluation

5.1 Introduction

In this chapter, we explore the practical implementation and evaluation of our system. We begin by detailing the datasets utilized in both the trajectory generation and recommendation phases. Next, we discuss the development environment employed for system realization, including the use of Streamlit for system presentation. Finally, we present the results of the evaluation conducted for both phases.

5.2 Dataset description

5.2.1 Transportation network dataset

For our transportation network dataset, we utilized data sourced from the "Open Data portal of Île-de-France Mobilités" which endeavors to offer comprehensive open data concerning mobility in the Île-de-France area. Specifically, we selected a dataset comprising 72 804 rows, detailing all lines within the Île-de-France network and the corresponding stops served by each line. Initially, this dataset contained the following attributes:

- route id: the identifier of the ligne;
- route long name: the name of the ligne;
- stop id: the identifier of a station;
- stop name: the name of the station;
- stop lon: the longitude of the station;
- stop at: the latitude of the station;
- OperatorName: the name of the operator;
- Nom commune: the name of the municipality;
- Code insee: the municipality code;
- Pointgeo: the geographic point of the station.

The figure 12 illustrate the initial state of the dataset.

	route_id	route_long_name	stop_id	stop_name	stop_lon	stop_lat	OperatorName	Pointgeo	Nom_commune	Code_insee
0	IDFM:C00001	482	IDFM:2679	Lycée Marianne	2.427407	48.730014	Keolis Ouest Val-de-Marne	48.730013592071785, 2.4274068162228435	Villeneuve-le-Roi	94077
1	IDFM:C00001	482	IDFM:2709	EDF	2.438983	48.729511	Keolis Ouest Val-de-Marne	48.729510906921725, 2.4389825601620467	Villeneuve-le-Roi	94077
2	IDFM:C00001	482	IDFM:461967	Ecoles	2.440496	48.732918	Keolis Ouest Val-de-Marne	48.73291790476698, 2.440496016072866	Villeneuve-le-Roi	94077
3	IDFM:C00001	482	IDFM:463501	Marché	2.420823	48.736440	Keolis Ouest Val-de-Marne	48.736439826111166, 2.420822990662051	Villeneuve-le-Roi	94077
4	IDFM:C00001	482	IDFM:21660	Place Amédée Soupault	2.418142	48.734232	Keolis Ouest Val-de-Marne	48.734232149423555, 2.4181416099829285	Villeneuve-le-Roi	94077
...	...	...	...	...	...	...	...	...	...	...
72799	IDFM:C01841	P	IDFM:472235	Changis - Saint-Jean	3.024129	48.958124	SNCF	48.958123608993006, 3.024129201681532	Changis-sur-Marne	77084
72800	IDFM:C01841	P	IDFM:472311	Mortcerf	2.908207	48.788625	SNCF	48.78862538612734, 2.908206738274958	Mortcerf	77318
72801	IDFM:C01841	P	IDFM:483833	Sainte-Colombe Septveilles / Place du Général ...	3.254041	48.529332	SNCF	48.52933156603378, 3.2540414434793448	Sainte-Colombe	77404
72802	IDFM:C01841	P	IDFM:470684	Nangis	3.012481	48.561285	SNCF	48.56128481096427, 3.012481441812586	Nangis	77327
72803	IDFM:C01841	P	IDFM:6756	Gare de Verneuil l'Etang	2.825089	48.644848	SNCF	48.644848150315916, 2.8250892153714657	Verneuil-l'Étang	77493

72804 rows × 10 columns

Figure 12: Initial state of the dataset

The city choice : Initially, the dataset included stations and transport lines from several cities in France. However, for our study, we want to focus on only one city, as it is sufficient for our research. Moreover, the city itself has more than 6,000 stations, making it preferable to limit the number of stations from other cities to reduce complexity. Since it is preferable to choose a city with a large number of stations, given that we want to generate multiple trajectory options, We displayed the number of stations available for each line in the figure 13 and mapped the stations on a map of France to observe their distribution. The results are ilustrated in the figure 14.

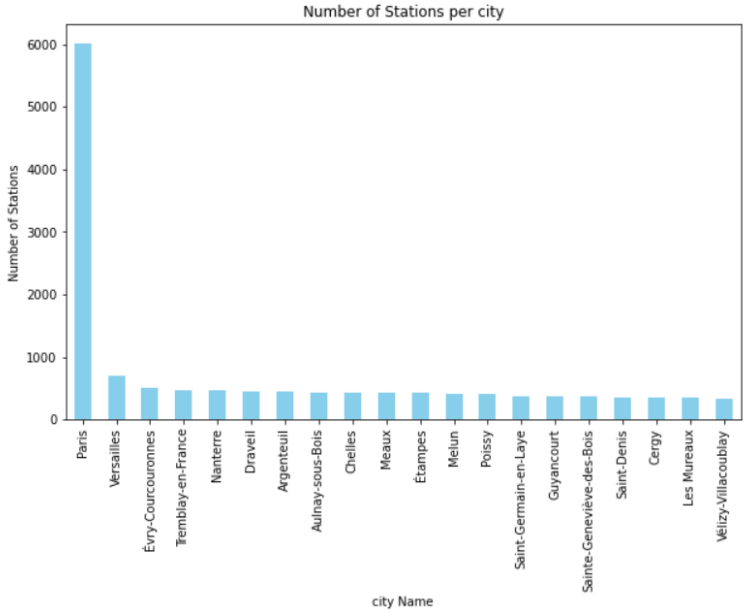

Figure 13: Distribution of stations in cities

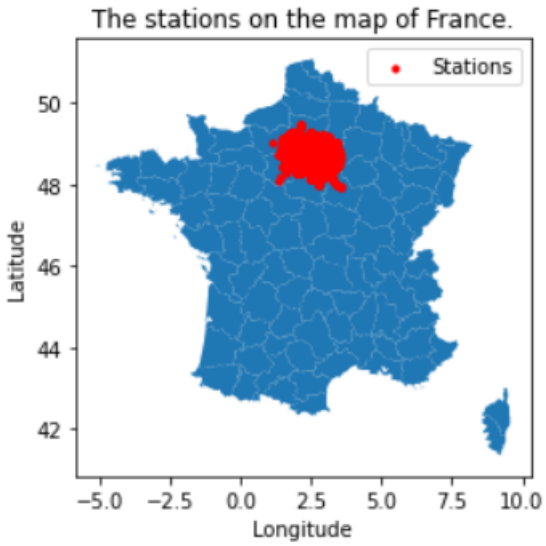

Figure 14: Location of stations on the map of France

By observing the results, we can clearly see that most of the stations are located in the city of Paris. That is why we decided to continue with this city and keep only its lines and stations, reducing the dataset from 72 804 stations to 6019.

Data cleaning Once the city was identified, we moved on to the data-cleaning phase by removing the unnecessary data and adding the missing information.

Adding missing data

1. In the context of our study, we need the mode of transportation of each line, which is not present in the current dataset. To address this, for each line, we will retrieve the name of one of its stations and manually verify the type of public transportation for the station. Then, we assign this type of transportation to the whole line. At the end, we obtained the dataset of the lines highlighted in figure 15.

	route_id	route_long_name	stop_name	Transport_Type
179	IDFM:C01371	1	Porte de Vincennes	metro
191	IDFM:C01372	2	Colonel Fabien	metro
199	IDFM:C01373	3	Quatre Septembre	metro
210	IDFM:C01374	4	Gare de l'Est	Bus
219	IDFM:C01375	5	Ourcq	metro
...	...	...	...	...
58054	IDFM:C01261	ORLYBUS	Denfert-Rochereau - Métro-RER	RER
58070	IDFM:C01264	289	Porte de Saint-Cloud - Métro	metro
59873	IDFM:C01731	R	Gare de Lyon	metro
59929	IDFM:C01745	TER Bourgogne - Franche-Comté	Paris-Bercy Bourgogne - Pays d'Auvergne	train
63556	IDFM:C01846	J	Pont Cardinet	metro

249 rows × 4 columns

Figure 15: Assigning a transportation type to each line in the dataset.

For the lines that still have no assigned transport type, which are only 5, we assigned them the second most common transport type, which is the Bus, according to the distribution of lines in the figure 16

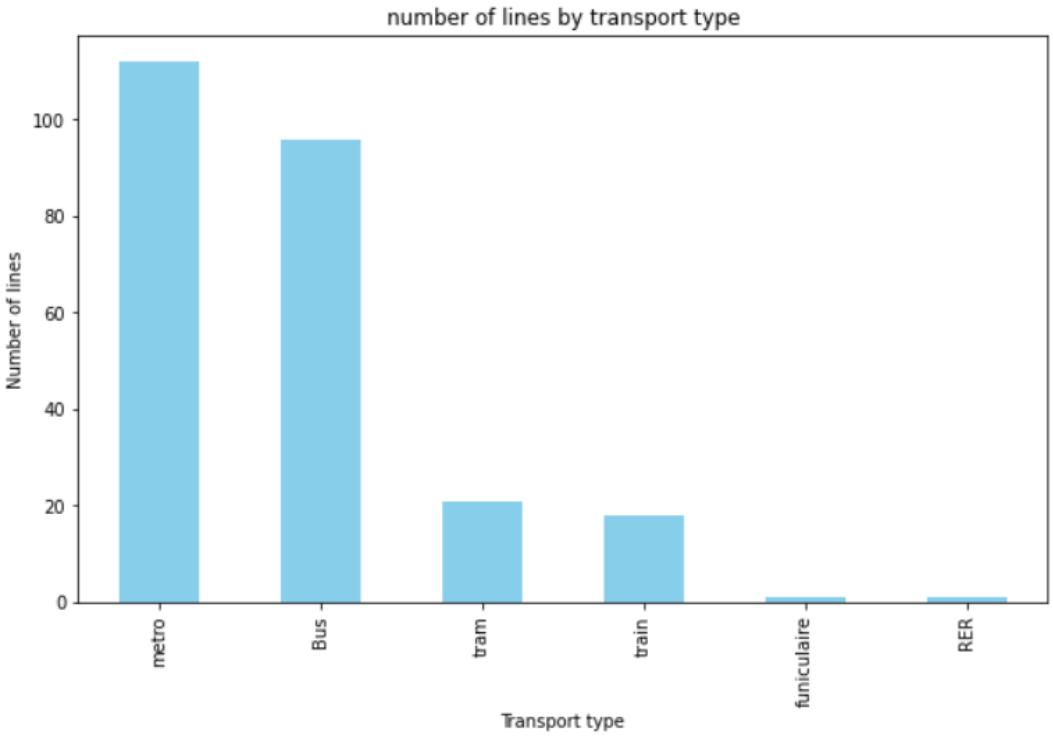

Figure 16: Number of lines by mode of transport

2. Another missing piece of data is the order of stations in each line. Considering the large number of stations and lines we have, it would be very inconvenient to assign their order in the line one by one. Therefore, we decided to proceed by distance. We will designate the first station as the head of the line, and then successively assign the closest station to the head the second position, continuing this process for each subsequent station.

3. Finally, we also needed to add a dataFrame containing information about each transport type because it's important for the upcoming phase of trajectory evaluation and recommendation. That's why we collected information about them and created the dataFrame illustrated in the figure 17

	Type	Time per 1 kilometer (min)	Ticket price (€)
0	metro	2.5	2.15
1	Bus	5.5	2.15
2	tram	3.7	2.15
3	train	1.0	5.00
4	RER	0.6	5.00
5	walk	10.0	0.00

Figure 17: Transportation information dataset

Deleting unnecessary data

1. After adding the missing data, we started deleting the unnecessary data, and we started by removing the columns that we don't need, including:
   • The Nom commune column, which is not useful since we only kept the Paris municipality.
   • The operatorname, which will not serve our study.
   • The route name, which is nothing but a duplication since we kept it in the lines table.
   • The INSEE code, which is the municipality code.
   • The geographical point, which is just a concatenation of the columns stop lon and stop lat.

2. Then after an observation of the number of station in each line, We saw that there are stations with more than 80 stations, which is too high. This suggests a possibility of having duplicate stations. Additionally, there are stations with fewer than 5 stations, which is too few.That is why we decided to delete all lines with less than 10 stations with all their stations. And we did not have any duplicated stations in the same line, but we had one duplicated line so we deleted it too.

3. In our study, we considered each road as bidirectional to avoid representing the same trajectory twice and then represent its directions. For this, we will remove the rows and all their stations if these rows have the same names but different IDs, meaning the trajectory is defined more than once.

Since we want to consider our transport network as a graph and we already have the nodes (the dataset of stations), all we need are the links between these nodes, which are the edges. That's why we added an Edges dataset in which we created an edge between each pair of successive stations on the same line.

We finally download all the created and updated datasets to use them in the next steps.

5.2.2 The trajectory recommendation used dataset

For trajectory recommendation, since our model is content-based, focusing on the analysis of item attributes, it is necessary to have trajectories previously chosen by the user from a group of trajectories to utilize it.

Therefore, we generated trajectories for 100 origin-destination pairs using our trajectory generation algorithm. The algorithm returned between 1 and 8 trajectories for each pair.

Subsequently, we manually selected one trajectory as the user's choice using a form for each pair as highlited in the figure 18.

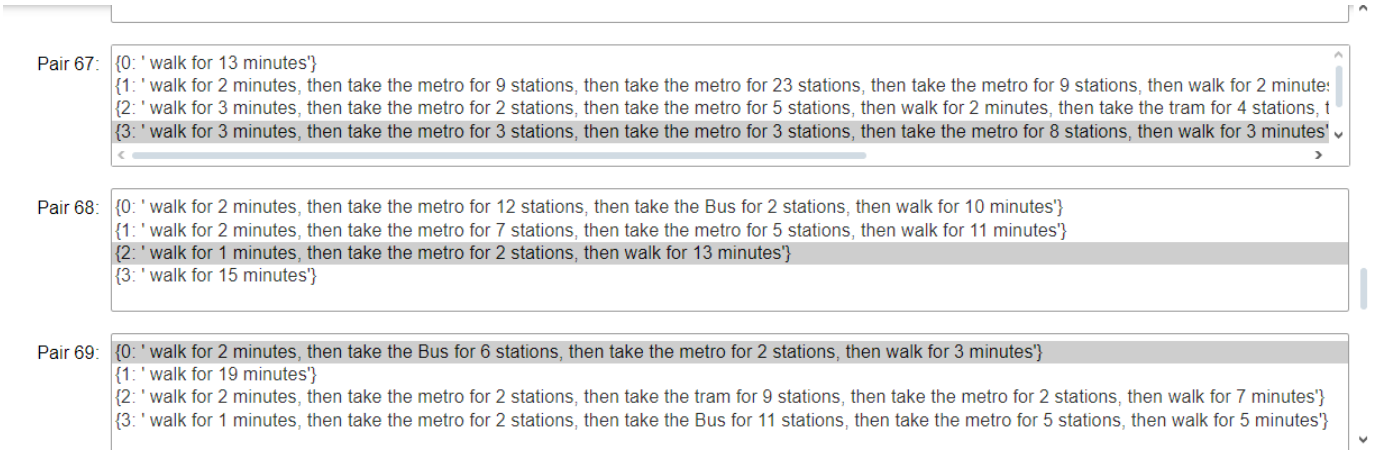

Figure 18: Form for trajectory selection

As a result, we have a dataset of 514 trajectories presented in the figure 19, with 100 chosen (one for each pair) and the rest not chosen. ne

	Origin	Destination	Duration (Hour)	Distance	Cost (€)	Proportion walk	Proportion Bus	Proportion metro	Proportion tram	Proportion train	Proportion RER	Index details	chosed
0	[48.877759, 2.343653]	[48.850646, 2.329514]	0.852467	8.999856	4.30	0.071636	0.928364	0.000000	0.000000	0.000000	0.0	0.0	True
1	[48.877759, 2.343653]	[48.850646, 2.329514]	0.531438	3.188628	0.00	1.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.0	False
2	[48.877759, 2.343653]	[48.850646, 2.329514]	1.852912	22.788332	10.75	0.196468	0.362958	0.002038	0.438537	0.000000	0.0	0.0	False
3	[48.877759, 2.343653]	[48.850646, 2.329514]	1.260198	12.982958	12.90	0.366677	0.542170	0.091153	0.000000	0.000000	0.0	0.0	False
4	[48.877759, 2.343653]	[48.850646, 2.329514]	2.407995	26.023515	10.75	0.020770	0.979230	0.000000	0.000000	0.000000	0.0	0.0	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...
509	[48.864054, 2.400308]	[48.853082, 2.306027]	1.809955	28.326307	7.15	0.142068	0.747146	0.000000	0.000000	0.110787	0.0	98.0	False
510	[48.842209, 2.404122]	[48.877083, 2.30986]	1.321627	7.929762	0.00	1.000000	0.000000	0.000000	0.000000	0.000000	0.0	99.0	False
511	[48.842209, 2.404122]	[48.877083, 2.30986]	2.996158	35.808063	10.75	0.432501	0.188966	0.148578	0.229954	0.000000	0.0	99.0	False
512	[48.842209, 2.404122]	[48.877083, 2.30986]	2.329327	29.327942	8.60	0.005197	0.864395	0.130408	0.000000	0.000000	0.0	99.0	True
513	[48.842209, 2.404122]	[48.877083, 2.30986]	2.224627	20.340995	6.45	0.359646	0.640354	0.000000	0.000000	0.000000	0.0	99.0	False

514 rows × 13 columns

Figure 19: The recommendation dataset

Each entry contains the attributes returned by our trajectory generation algorithm, which are as follows:

- Origin : Origin coordinates
- Destination : Destination coordinates
- Duration (Hour) : Travel duration
- Cost : Total cost
- Distance : Distance of the journey
- Proportion ”mode” : Proportion of each mode of transport from 0 to 1
- choosed : Boolean variable indicating whether the trajectory was taken or not.

5.3 Development Environment

5.3.1 Hardware Environment

The model was executed on the following hardware:

- Computer: DELL
- Processor: Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz, 1992 MHz

- Installed Memory (RAM): 8.00 GB
- System Type: 64-bit Operating System, x64-based processor
- Operating System: Windows 11 Professional

5.3.2 Software Environment

Anaconda Anaconda is a distribution of Python and R designed for scientific computing, data science, and machine learning. It simplifies package management and deployment across Windows, Linux, and macOS. Anaconda includes over 250 pre-installed packages and supports installing more than 7,500 additional packages via PyPI or conda. Anaconda Navigator, its GUI, provides a user-friendly way to manage packages, environments, and applications without using the command line.

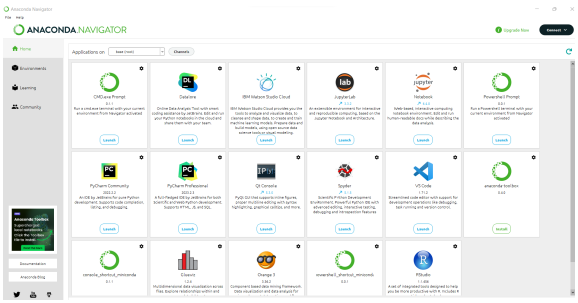

Figure 20: Anaconda interface

Jupyter notebook The Jupyter Notebook stands as an open-source web application designed for creating and sharing documents comprising live code, equations, visualizations, and textual content. Originating as a spin-off project from IPython, which previously hosted its IPython Notebook project, Jupyter derives its name from the core supported programming languages: Julia, Python, and R.

Initially shipping with the IPython kernel for Python programming, Jupyter has since expanded its compatibility to include over 100 other kernels, allowing users to work with various languages beyond Python. This versatility empowers users to harness a wide array of programming languages within the same notebook environment, enhancing flexibility and facilitating interdisciplinary collaboration.

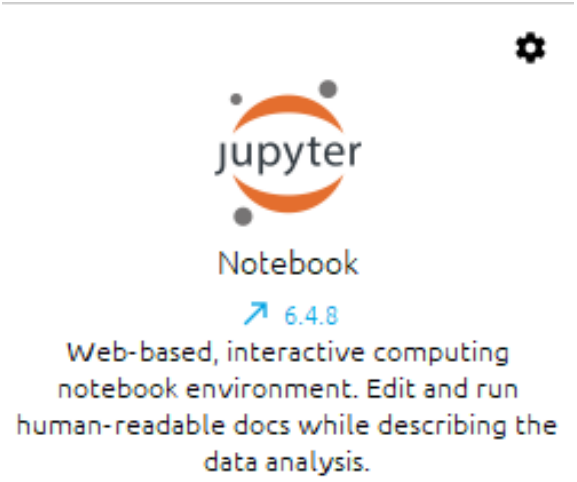

Figure 21: Jupyter notebook

Python Python is a high-level, dynamically-typed programming language known for its readability and versatility in Rapid Application Development (RAD). It supports modularity, code reuse, and integrates well with existing components. Python's interpreter and extensive standard library are freely available across major platforms, driving its widespread adoption and community support.

Figure 22: Python

Python librairies

- pandas : pandas employs fast, flexible, and expressive data structures tailored to simplify working with relational or labeled data. It emerged as one of the most utilized libraries in this project.
- numpy : Renowned for its simplicity yet unparalleled usefulness in data manipulation within this project.
- math : This library proved indispensable by providing access to a plethora of mathematical functions.

- GeoPy : a Python library, streamlines geographical calculations, which were pivotal within the context of our project.
- geopandas : An open-source endeavor aimed at facilitating geospatial data manipulation in Python.
- matplotlib : Matplotlib serves as a comprehensive tool for generating static, animated, and interactive visualizations in Python, significantly aiding various decision-making processes with its visualization capabilities.
- seaborn : As a Python library for crafting statistical graphics, seaborn complements matplotlib seamlessly and integrates closely with pandas data structures.
- sklearn.model selection : Scikit-learn, a Python library, offers a wide array of unsupervised and supervised learning algorithms. Its role was paramount, particularly in the data splitting phase.
- torch : A suite of APIs extending PyTorch's core library of operators. Its significance was highlighted, especially in conjunction with Ranknet.
- Streamlit: an open-source Python library that facilitates the creation and sharing of custom web applications for machine learning and data science.

Figure 23: Pythons librairies

5.4 Presentation of the Recommendation System

To present our trajectory recommendation system, we utilized the Streamlit library in Python. Our interface primarily features a map of the city of Paris that shows in the figure 24, which is directly displayed to

the user, prompting them to select their starting point and then their destination on the map.

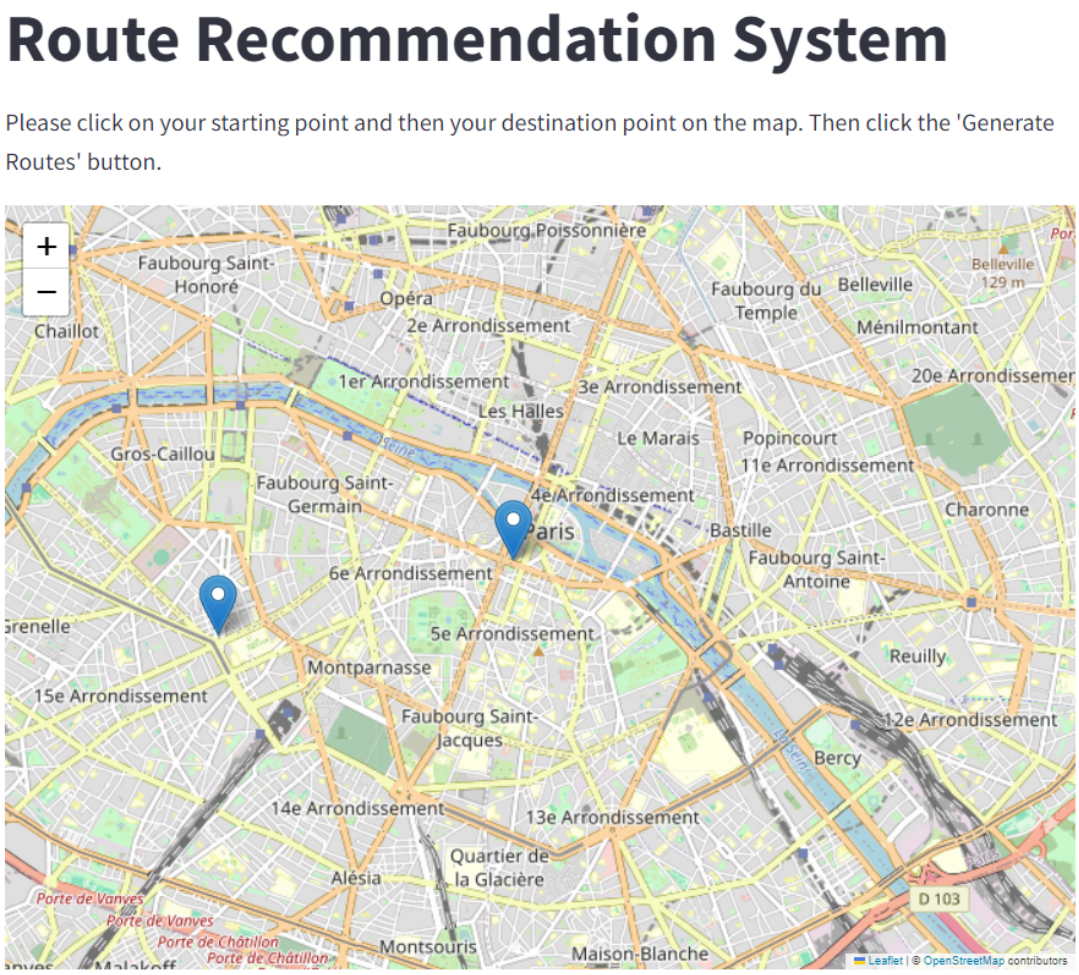

Figure 24: Interface for Selecting Starting and Destination Points on the Map

After the user selects the coordinates, the system retrieves these coordinates and displays them back to the user as illustrated in the figure 25.

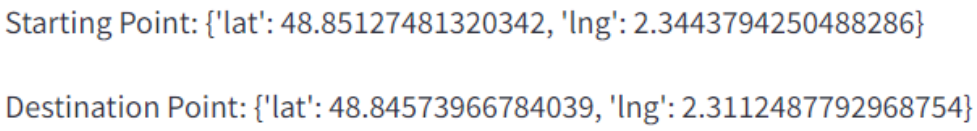

Figure 25: Retrieving Origin and Destination Coordinates

After the coordinates are selected and displayed, the user simply needs to press the button to generate trajectories.

The system will then produce a list of generated trajectories, ordered according to the recommendation model. The trajectories will be displayed with a gradient of green shades as it is highlighted in the figure 26, where lighter shades of green indicate a higher recommendation level.

Route Recommendation System

The trajectory will take 18 minutes and will cost 2.15€

Walk : to IDFM:23662 for `1` minutes.

Take the metro line 56 : for `3` stations from IDFM:23662 to IDFM:23658 for `1` minutes and `2.15` €.

Walk : to destination for `16` minutes.

Sélectionner 0

The trajectory will take 16 minutes and will cost 2.15€

Walk : to IDFM:23662 for `1` minutes.

Take the metro line 56 : for `5` stations from IDFM:23662 to IDFM:23655 for `1` minutes and `2.15` €.

Walk : to destination for `14` minutes.

Sélectionner 1

The trajectory will take 15 minutes and will cost 2.15€

Walk : to IDFM:23662 for `1` minutes.

Take the metro line 56 : for `6` stations from IDFM:23662 to IDFM:23653 for `1` minutes and `2.15` €.

Walk : to destination for `13` minutes.

Sélectionner 1

The trajectory will take 15 minutes and will cost 2.15€

Walk : to IDFM:23662 for `1` minutes.

Take the metro line 56 : for `6` stations from IDFM:23662 to IDFM:23653 for `1` minutes and `2.15` €.

Walk : to destination for `13` minutes.

Sélectionner 2

The trajectory will take 8 minutes and will cost 2.15€

Walk : to IDFM:23662 for `1` minutes.

Take the metro line 56 : for `15` stations from IDFM:23662 to IDFM:36665 for `5` minutes and `2.15` €.

Walk : to destination for `2` minutes.

Sélectionner 3

The trajectory will take 7 minutes and will cost 2.15€

Walk : to IDFM:23662 for `1` minutes.

Take the metro line 56 : for `14` stations from IDFM:23662 to IDFM:426942 for `4` minutes and `2.15` €.

Walk : to destination for `1` minutes.

Sélectionner 4

The trajectory will take 7 minutes and will cost 2.15€

Walk : to IDFM:23662 for `1` minutes.

Take the metro line 56 : for `13` stations from IDFM:23662 to IDFM:426940 for `4` minutes and `2.15` €.

Walk : to destination for `2` minutes.

Sélectionner 5

The trajectory will take 22 minutes and will cost 4.3€

Walk : to IDFM:21910 for `1` minutes.

Take the metro line 9 : for `2` stations from IDFM:21910 to République for `2` minutes and `2.15` €.

Take the metro line 56 : for `5` stations from République to IDFM:23655 for `5` minutes and `2.15` €.

Walk : to destination for `14` minutes.

Sélectionner 6

Go Back

Figure 26: The display of trajectories generated in order

After generating and displaying the trajectories, the user will need to press the 'Select i' button next to the selected trajectory i so that this trajectory is added to his selected trajectories history and to obtain the details of this trajectory printed on a map as the figure 27 shows.

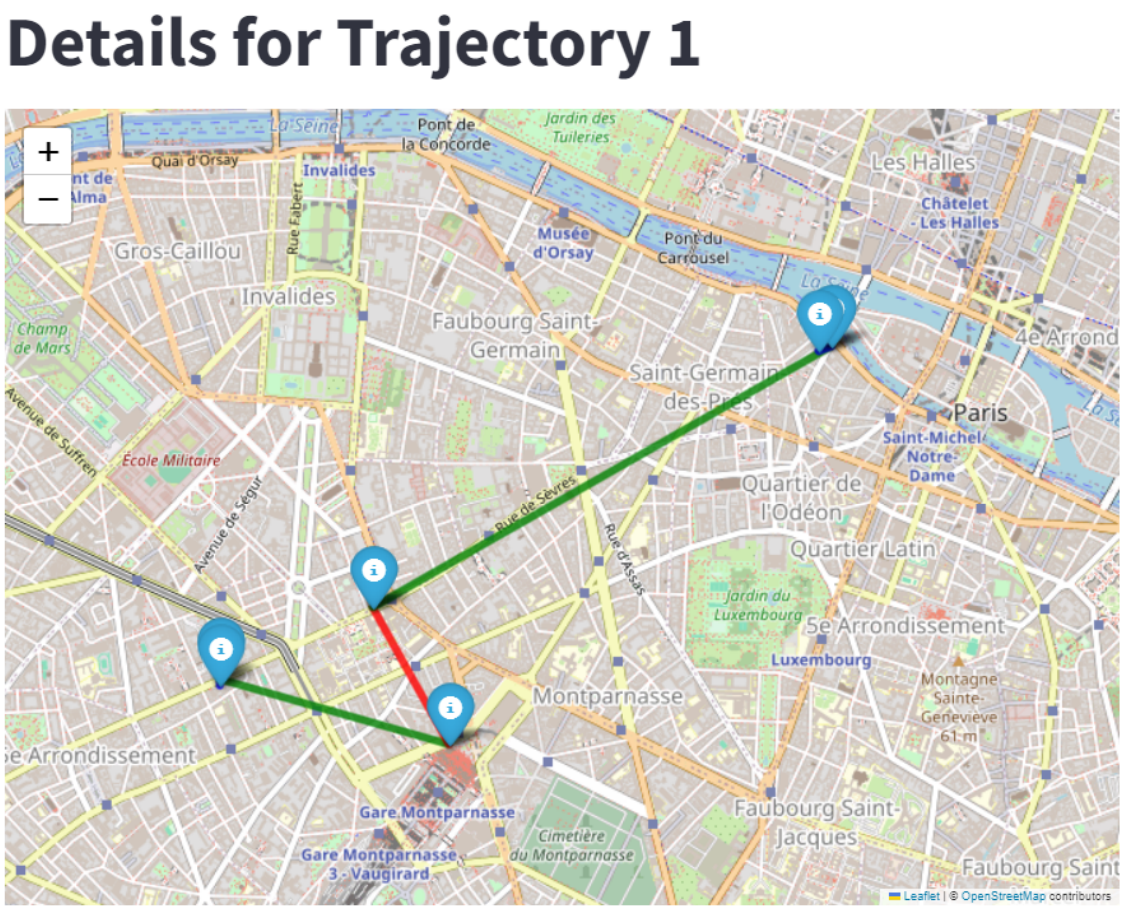

Figure 27: Displaying the details of the trajectory.

To identify the type of transportation for each trajectory, each distinct type is assigned a display color, and clarification of the colors for each type is provided as shown in the figure 28 to the user alongside the display of their trajectories .

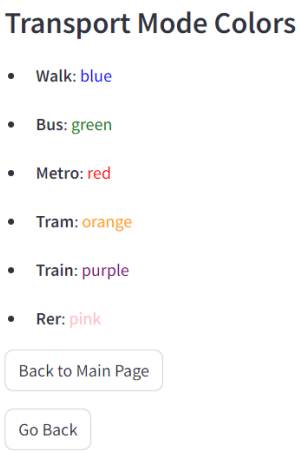

Figure 28: keys for each type of transportation

5.5 Evaluation

In this section, we will present the results of our approach evaluation. Our system consists of two main components: trajectory generation and recommendation of the best trajectory according to user preferences. We evaluated our approach in two ways: first, in terms of the relevance of the generated trajectories, and second, in terms of the recommendations.

5.5.1 Evaluation of Trajectory Generation

To assess the quality of the generated trajectories, we relied on metrics that we consider essential, which are as follows:

- For the first two metrics, we used a dataset consisting of trajectories generated for 100 origin-destination pairs. For these metrics, we evaluated the dataset's coverage of all transportation modes by displaying the frequency of each transportation mode and then showing the frequency of the combinations provided by the system, the results are presented in figures 29 and 30.

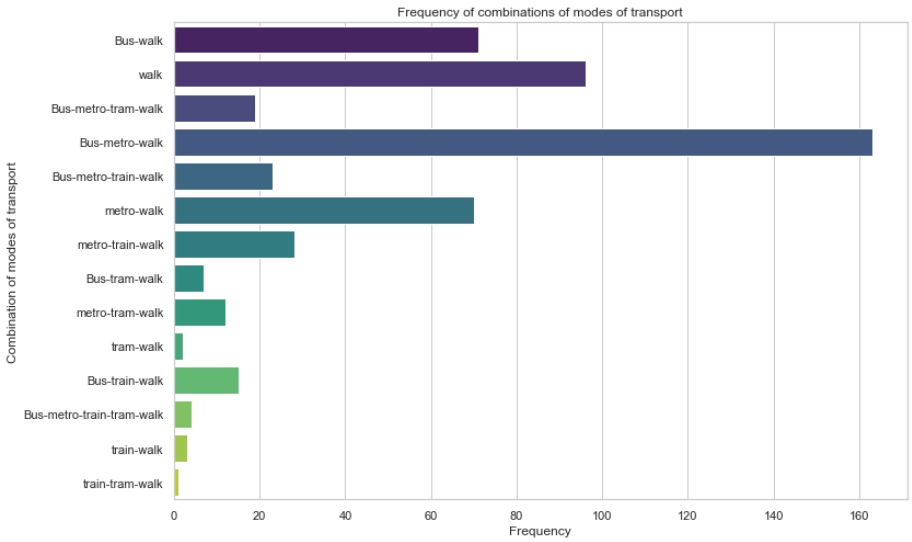

Figure 29: Frequency of combinations of modes of transport

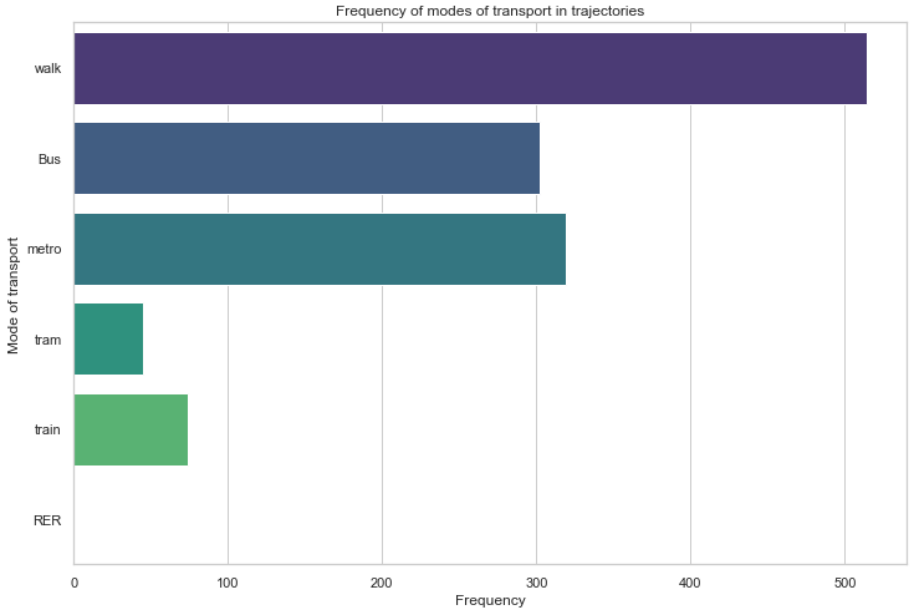

Figure 30: Frequency of modes of transport in trajectories

Upon reviewing the results, we can see that the system covers all modes of transport except for the RER, which is likely due to the small number of lines in the dataset (less than 5 lines). The most common modes of transport are the metro followed by the bus, reflecting the recent trend in Paris

where the metro is the most frequently used mode of transport. Regarding transport combinations, the most frequent combination observed is Bus-Metro, with a total of 9 combinations involving more than one mode of transport, which is quite significant.

- We then recorded the execution time for generating trajectories for 20 origin-destination pairs from the request launch to obtaining the results. the results are illustrated in the figure 31.

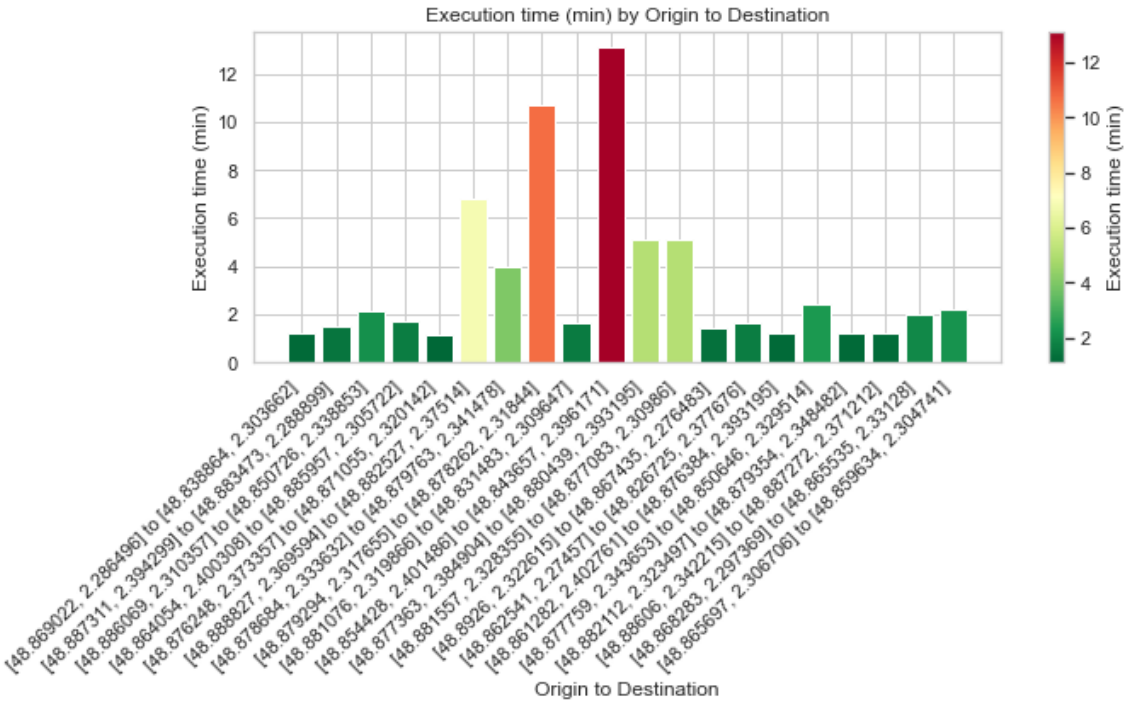

Figure 31: Execution Time for 20 Origin-Destination Pairs

The execution time ranged from 1 minute to 13 minutes. Thus, the generation speed depends on the selected pair.Overall, the algorithm takes time for generation.

- We then displayed the average distance of the generated trajectories for each pair, as well as the minimum distance among these trajectories, based on the straight-line distance between the destination and the origin points, figure 32 and 33 displays the outcomes.

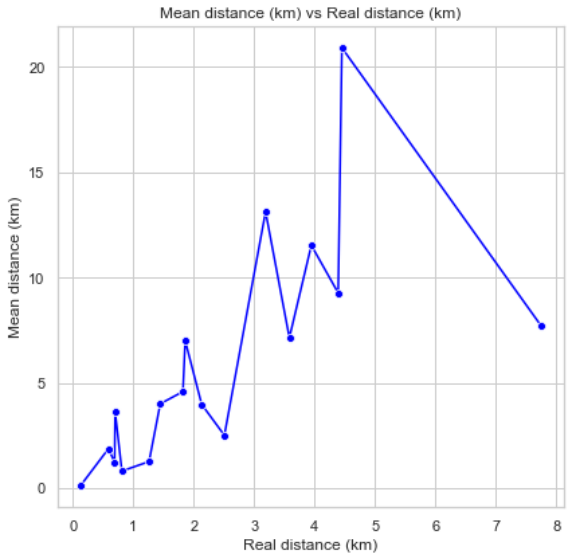

Figure 32: Average Trajectory Distances Compared to Direct Point-to-Point Distance

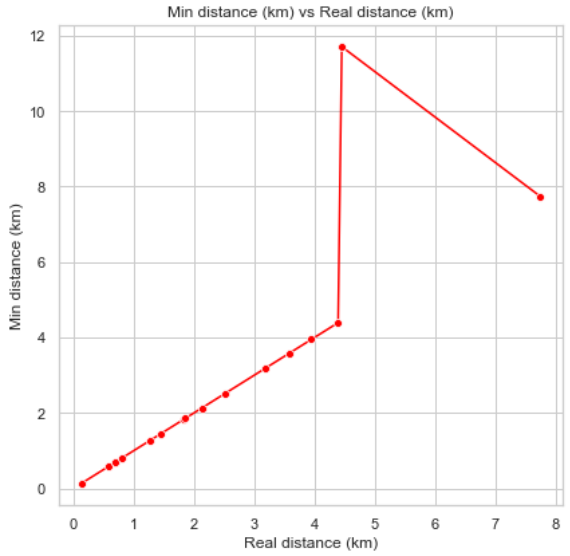

Figure 33: Minimum Trajectory Distances Compared to Direct Point-to-Point Distance

- Finally, we compared the trajectories generated by our algorithm for these 20 pairs with those generated by Google Maps, aiming to evaluate both time and cost of the journeys. We obtained the results from Google Maps by manually searching for the trajectories for each of the 20 origin-destination pairs and recording the minimum travel time and cost.

It's important to note that the comparison is not entirely accurate in terms of time, as our system does not account for waiting times and transport schedules. Similarly, in terms of cost, our system does not consider subscriptions and various discount offers integrated by Google Maps. Additionally, it should be noted that the execution resources are not identical, as our tests were conducted on our own machine infrastructure. Results are depicted in Figures 34 and 35.

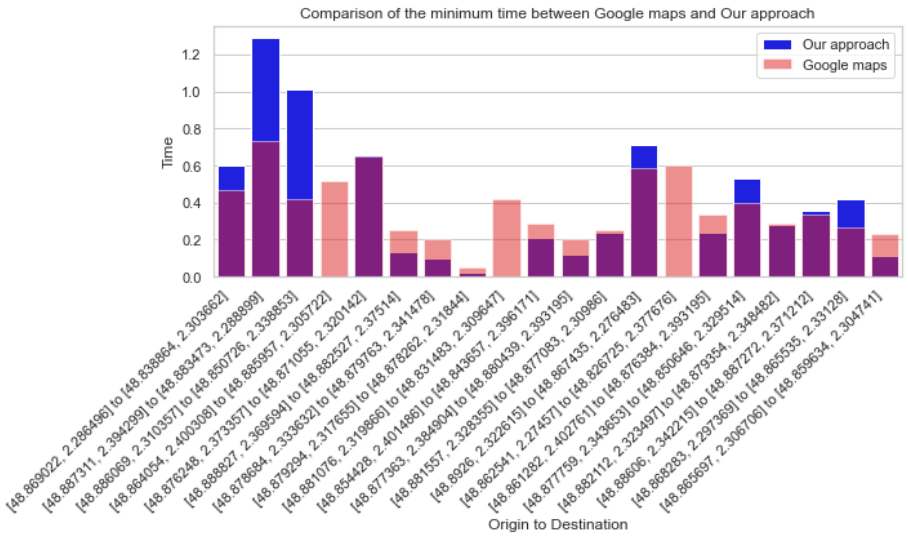

Figure 34: Comparison of the minimum time between Google maps and Our approach

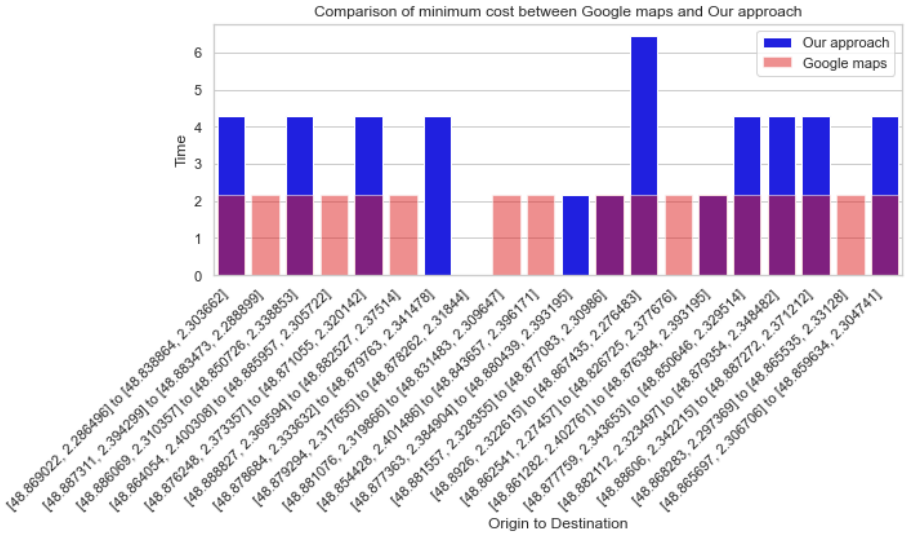

Figure 35: Comparison of minimum cost between Google maps and Our approach

5.5.2 Evaluation of the recommendation

For our ranking recommendation, we used the following metrics. The results are presented in the table 2.

1. Weighted Precision : Measures the proportion of true positives among the items predicted as positives, considering class weights. Useful for imbalanced classes.

   Formula:

   $$\text{Weighted Precision} = \frac{\sum_{i=1}^{k} \text{Precision}_i \cdot n_i}{\sum_{i=1}^{k} n_i}$$

2. Recall : Measures the proportion of true positives among all actual positives. Indicates the model's ability to detect true positives.

   Formula:

   $$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

3. Weighted F1 Score : The harmonic mean of weighted precision and recall, considering class weights. Balances precision and recall for imbalanced data.

   $$\text{Weighted F1 Score} = \frac{2 \cdot \text{Weighted Precision} \cdot \text{Weighted Recall}}{\text{Weighted Precision} + \text{Weighted Recall}}$$

4. Mean Reciprocal Rank (MRR) : Measures search/recommendation effectiveness, averaged over the reciprocal ranks of the first relevant results. Useful for ranking tasks.

   Formula:

   $$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}$$

5. AUC-ROC (Area Under the Receiver Operating Characteristic Curve) : Evaluates binary classification performance by measuring the area under the ROC curve, plotting true positive rate against false positive rate.

   Formula: AUC is typically calculated using numerical integration methods.

Metric	Value
Weighted Precision	1.0000
Recall	0.8646
Weighted F1 Score	0.9267
Mean Reciprocal Rank (MRR)	1.0000
AUC-ROC	0.8457

Table 2: Performance metrics for the ranking recommendation.

As we observed, the model returned very good results. Despite the possibility that these outcomes are partly due to the small size of the dataset, we decided to continue using the model.

Comparing our recommendation results with those of other systems would not be fair, as unlike other systems trained on datasets from real cities with vast amounts of data, our system was trained on a small dataset of 514 manually constructed entries.

5.6 Conclusion

In summary, this chapter has provided an in-depth look at the implementation and evaluation of our system. By detailing the datasets used in trajectory generation and recommendation phases, discussing the development environment, and presenting the evaluation results, we have demonstrated that our system produces achievable and favorable outcomes. These results underscore the utility and efficiency of our approach, setting the stage for further enhancements and applications in real-world settings.

Chapter 6 : Conclusion

Multimodal transport recommendation involves suggesting a trajectory for users from a starting point to a destination, comprising one or a combination of available transport modes, while respecting user preferences. We began by providing context, discussing the background of recommendation systems, their principles, the methods used, and evaluation metrics. We then moved on to smart cities, as their development has significantly enabled the emergence of such systems. Next, we explored the concept of multimodal transport systems and the transformation from conventional to multimodal transport.

We reviewed recent work in the field of recommendation systems to understand the latest trends and innovations, noting the variety of different approaches. We then proposed our approach, which consists of two main steps. First, we described our network as a graph where each station represents a node. We then sought an optimal method to generate trajectories, choosing Particle Swarm Optimization (PSO). Once the trajectories were generated, we applied post-processing to ensure they were feasible. Finally, we recommended these trajectories to users based on their previously selected trajectories by training a ranking model.

We implemented and tested our model using real-world data from the Paris transport network, which required some preprocessing to be usable. We generated trajectories for origin-destination pairs, manually selected choices among the proposals, and evaluated our recommendation model on this dataset. The model's results were generally good, producing feasible trajectories with reasonable prices and travel times. However, execution times varied, sometimes taking longer for certain trajectories and less for others, we also compared our returned trajectories with those from Google Maps in terms of cost and time. Although our recommendation evaluation was based on a small dataset, the results were very promising, with almost perfect accuracy.

To further improve our model, we propose exploring optimization methods to reduce trajectory search times and incorporating transport schedules for better adaptability. Additionally, integrating transport subscription options would provide more realistic and contextually relevant recommendations within the transport network.

References

Adomavicius, G. and A. Tuzhilin (2005). "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions". In: *IEEE Transactions on Knowledge and Data Engineering* 17.6, pp. 734–749. DOI: 10.1109/TKDE.2005.99.

Iftikhar, Arta et al. (2023). "A reinforcement learning recommender system using bi-clustering and Markov Decision Process". In: *Expert Systems With Applications*. Received 12 June 2023; Received in revised form 16 August 2023; Accepted 8 September 2023; Available online 15 September 2023. DOI: 10.1016/j.eswa.2023.121541. URL: https://doi.org/10.1016/j.eswa.2023.121541.

Rodríguez-Hernández, María del Carmen et al. (2015). "Location-Aware Recommendation Systems: Where We Are and Where We Recommend to Go". In: *Proceedings of LocalRec'15*. Copyright held by the author(s). ACKNOWLEDGMENTS: This work has been supported by the CICYT project TIN2013-46238-C4-4-R, DGA-FSE, and a Banco Santander scholarship held by María del Carmen Rodríguez Hernández. University of Zaragoza, Spain. Vienna, Austria.

Saini, Kapil and Ajmer Singh (2023). "A content-based recommender system using stacked LSTM and an attention-based autoencoder". In: *Measurement: Sensors*. Received 4 September 2023; Received in revised form 27 November 2023; Accepted 14 December 2023; Available online 20 December 2023. DOI: 10.1016/j.measen.2023.100975. URL: https://doi.org/10.1016/j.measen.2023.100975.

Saifudin, Ilham and Triyanna Widiyaningtyas (2024). "Systematic Literature Review on Recommender Systems: Approach, Problems, Evaluation Techniques, and Datasets". In: *IEEE Access*. Digital Object Identifier 10.1109/ACCESS.2024.3359274. DOI: 10.1109/ACCESS.2024.3359274.

Melville, Prem and Vikas Sindhwani (2002). "Recommender Systems". In: *IBM T.J. Watson Research Center*.

Breese, John S., David Heckerman, and Carl Kadie (1998). "Empirical Analysis of Predictive Algorithms for Collaborative Filtering". In: *Microsoft Research*. {breese, heckerma, carlk}@microsoft.com.

Nguyen, Hoang, Dina Nawara, and Rasha Kashef (2024). "Connecting the indispensable roles of IoT and artificial intelligence in smart cities: A survey". In: *Journal of Information and Intelligence* xxx. DOI: 10.1016/j.jiixd.2024.01.003.

Lai, Calvin Ming Tsun and Alistair Cole (2022). "Measuring progress of smart cities: Indexing the smart city indices". In: *Urban Geography* xxx. DOI: 10.1016/j.ugj.2022.11.004.

Fadhel, Mohammed A. et al. (2024). "Comprehensive systematic review of information fusion methods in smart cities and urban environments". In: *Information Fusion* 107. DOI: 10.1016/j.inffus.2024.102317. URL: https://doi.org/10.1016/j.inffus.2024.102317.

Omrany, Hossein et al. (2024). "IoT-enabled smart cities: a hybrid systematic analysis of key research areas, challenges, and recommendations for future direction". In: *Discover Cities*.

Dua, Aman and Deepankar Sinha (2015). "The Multimodal Transportation: Research Trend and Literature Review". In: *Indian Institute of Foreign Trade* 39.4. Ph. D scholar at Indian Institute of Foreign Trade, New Delhi. Assistant Professor at University of Petroleum and Energy Studies Dehradun. Phone: +919759134215. Email: adua@ddn.upes.ac.in, om.amandua@gmail.com. Address: Ram Nager -554, Lane No -12 Roorkee. Associate Professor Phone: +919432673733. Indian Institute of Foreign Trade, Kolkata Campus J1/14 EP GP Block, Sector V, Salt Lake, Kolkata 700091, India. E-mail: dsinha@iift.ac.in, October–December.

Smith, Denise A. (2013). "The Evolution of Multimodal Transportation Planning: Key Factors in Shaping the Approaches of State DOTs". In Partial Fulfillment of the Requirements for the Degree Master of Science in the School of Civil and Environmental Engineering. Master's Thesis. Georgia Institute of Technology.

Pedersen, Neil J. (Chairman: Neil J. Pedersen). "Multimodal Transportation Planning at the State Level: State of the Practice and Future Issues". In: *Committee on Statewide Multimodal Transportation Planning*. Maryland Department of Transportation.

Campigotto, Paolo et al. (2016). "Personalized and Situation-Aware Multimodal Route Recommendations: The FAVOUR Algorithm". In: *IEEE Transactions on Intelligent Transportation Systems* 1. Manuscript received April 14, 2015; revised February 29, 2016; accepted April 24, 2016. This work was supported in part by the research projects emporA2 and Crossing Borders through the Climate and Energy Funds (KLIEN) of the Austrian Ministry for Transport, Innovation and Technology (BMVIT). DOI: 10.1109/TITS.2016.2565643.

Liu, Hao, Ting Li, et al. (2019). "Joint Representation Learning for Multi-Modal Transportation Recommendation". In: *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*. AAAI Press.

Abedalla, Ayat et al. (2019). "MTRecS-DLT: Multi-Modal Transport Recommender System using Deep Learning and Tree Models". In: *Proceedings of the IEEE*. Jordan University of Science and Technology, Irbid, Jordan, p. 274. ISBN: 978-1-7281-2946-4. DOI: 10.1109/PROC.2019.2946.

Liu, Hao, Yongxin Tong, et al. (2020). "Incorporating Multi-Source Urban Data for Personalized and Context-Aware Multi-Modal Transportation Recommendation". In: *IEEE Transactions on Knowledge and Data Engineering* 32.12, pp. 2343–2355.

Liu, Yang et al. (2021). "Exploring a large-scale multi-modal transportation recommendation system". In: *Transportation Research Part C: Emerging Technologies* 126, p. 103070.

Xu, Aikun et al. (Feb. 2023). "THAN: Multimodal Transportation Recommendation With Heterogeneous Graph Attention Networks". In: *IEEE Transactions on Intelligent Transportation Systems* 24.2, p. 1533. DOI: 10.1109/TITS.2023.1234567.

Liu, Hao, Jindong Han, et al. (2023). "Unified route representation learning for multi-modal transportation recommendation with spatiotemporal pre-training". In: *The VLDB Journal* 32, pp. 325–342. DOI: 10.1007/s00778-022-00748-y.

Yang, Ruixia et al. (2024). "Door to door space-time path planning of intercity multimodal transport network using improved ripple-spreading algorithm". In: *Computers & Industrial Engineering* 189, p. 109996. DOI: 10.1016/j.cie.2024.109996.

Hao, Congli and Yixiang Yue (2016). "Optimization on Combination of Transport Routes and Modes on Dynamic Programming for a Container Multimodal Transport System". In: *Procedia Engineering* 137, pp. 382–390. DOI: 10.1016/j.proeng.2016.01.272.