UNIVERSIDADE D
COIMBRA

João António Correia Vaz

Railway Traffic Management System

Dissertation in the context of the Master in Informatics Engineering, specialization in Software Engineering, advised by Professor Vasco Nuno Sousa Simões Pereira and Engineer Paulo Miguel Pacheco Carreiro and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

July 2024

DEPARTAMENTO DE
ENGENHARIA INFORMÁTICA
FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

João António Correia Vaz

Railway Traffic Management System

Dissertation in the context of the Master in Informatics Engineering, specialization in Software Engineering, advised by Professor Vasco Nuno Sousa Simões Pereira and Engineer Paulo Miguel Pacheco Carreiro and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

July 2024

DEPARTAMENTO DE
ENGENHARIA INFORMÁTICA
FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

João António Correia Vaz

Sistema de Gestão de Tráfego Ferroviário

Dissertação no âmbito do Mestrado em Engenharia Informática, especialização em Engenharia de Software, orientada pelo Professor Vasco Nuno Sousa Simões Pereira e pelo Engenheiro Paulo Miguel Pacheco Carreiro e apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Julho 2024

Acknowledgements

This master's thesis would not have been possible without the guidance, encouragement, emotional support and help of several individuals who, in some way, contributed and extended their valuable assistance during this study. It is a pleasure and honour to thank those who made it possible.

First, I am indebted to my two supervisors whose guidance, support, and expertise enabled me to complete my thesis. I want to thank Professor Vasco Pereira for guiding me and helping me in everything he could. Considering the initial events, Professor Vasco's rigour and demands made this thesis possible and always on time. These valences also ensured that the thesis was written correctly and appropriately. Next, I want to thank Engineer Paulo Carreiro for his guidance and help in the railway market. Engineer Paulo's expertise and knowledge about the project can be considered a lighthouse at night.

Although I didn't have a formal collaborator in this project, I would like to thank all the colleagues who warmly welcomed me into their work group and integrated me into their two-week sprints, where we shared our progress during the daily meetings. There were moments when I struggled with development, and they promptly helped me with some advice and ideas. They also immediately made themselves available to read my final report so they could give me their opinion on my writing style and its understandability.

Then, I would like to thank my platoon's project manager supporters (PMS), Carolina T. and Vânia P., who welcomed me from the start and with whom I could share my breaks from work and other memorable moments. Outside of CSW, I would like to thank my closest friends, David L., Diogo B., Guilherme J., Maria C. and Neuza E., with whom I was able to share some moments of relaxation that allowed me to escape from my routine daily life's routine from the past months.

Next, I would like to thank my family, especially my uncles, aunts, and cousins, for always caring about me and trying to give me a good time when we are together. All the opportunities we had to be together and the respective moments are undoubtedly happy memories and give me some of the good energy I need to keep going.

Finally, I would like to thank the three most meaningful people: my father, Silvio Vaz; my mother, Teresa Correia; and my girlfriend, Inês Ramos, my three pillars that always support me and make me want to keep going. The counselling they give me is vital and contributes enormously to making my best decisions. In fact, without them, I wouldn't have been able to complete this thesis, and I'm sure they played an essential role in helping me become a better person and be more focused on my goals. At last, no words can describe how grateful and lucky I am to have them in my life.

Abstract

In recent years, we have seen an evolution in the use of Artificial Intelligence (AI) to solve known problems faster and more efficiently. One of the areas of application is traffic management. We have more and more people in large cities who use public transport, which leads to the need for a more efficient network, which directs to an increase in the quality of services available.

Critical Software (CSW) has been working for more than eight years with one of the largest companies in the rail market to improve and update their products. These improvements had as their final objective the use of these systems to improve the quality of the service and, consequently, the passengers' satisfaction. In 2020, CSW was invited to a new project to add new functionalities to a product that deals with rail traffic management. One of the most critical parts of this type of product is the algorithms they use to optimise and automate the railway network. Also, in 2020, the company sold part of its business, and the product came under a new company. This internship improved the optimisation algorithms used in this product.

These algorithms are used to solve problems on the rail network, for example, to force a train to change its route or minimise possible delays. With the growth of evolutionary algorithms, we aim to create and apply a new AI technique in the product that fully utilises its capabilities. This new algorithm should improve the speed and quality of the responses and enable more trains to operate on the railway network. The trainee initially produced a contextualisation of the railway concepts, followed by market research for existing solutions and a search in the literature for optimisation algorithms and neural networks that could be connected to resolving these problems. After the research tasks, all the information gathered was analysed and compared. A set of risks and requirements, functional and non-functional, were elaborated before going through the project's development, testing and evaluating. This internship allowed the trainee to assess the performance of an Artificial Intelligence (AI) model inside the railway scheduling context by adjusting the dwell time of trains. It was possible to compare multiple models to evaluate the best among them and then compare the selected model against a mathematical algorithm used in the product of the CSW. The results revealed that the performance can be increased and an acceptable accuracy level can be maintained.

Keywords

Traffic Management Systems, Network Optimisation Problems, Route Automation Problems, Optimisation Algorithms, Reinforcement Learning, Artificial Intelligence, Neural Networks, Long Short-Term Memory, Recurrent Neural Networks, Feed Forward Neural Networks, Linear Neural Network

Resumo

Nos últimos anos temos assistido a uma evolução na utilização de Inteligência Artificial de modo a resolver problemas conhecidos de um modo mais rápido e eficiente. Uma das suas áreas de utilização é a gestão de tráfego. Cada vez temos mais pessoas nas grandes cidades que usam meios de transporte o que leva à necessidade da utilização de uma rede mais eficaz que leve a um aumento na qualidade dos serviços disponibilizados.

A Critical Software (CSW) trabalhou, há mais de oito anos, com uma das maiores empresas do mercado ferroviário para melhorar e atualizar os seus produtos. Estas melhorias tinham como objetivo final a utilização destes sistemas para melhorar a qualidade do serviço e, consequentemente, a satisfação dos passageiros. Em 2020, a CSW foi convidada para um novo projeto com o objectivo de adiconar novas funcionalidades a um produto que trata da gestão de tráfego ferroviário. Uma das partes mais importantes deste tipo de produto são os algoritmos utilizados para otimizar e automatizar a rede ferroviária. Ainda em 2020, a empresa parceira acabou por vender parte do seu negócio, passando o produto a estar sob a alçada de uma nova empresa. Este estágio surgiu com o objetivo de melhorar os algoritmos de otimização utilizados neste tipo de produto. Estes algoritmos são usados para resolver problemas que surjam na rede ferroviária e que obriguem à mudança de rota dos comboios ou para minimizar eventuais atrasos. Com o recente crescimento dos algoritmos evolutivos pretende-se desenvolver e implementar um novo algoritmo para que o produto tire o máximo proveito das suas capacidades e que melhore a velocidade de resposta aos problemas, a qualidade das repostas dadas e que seja capaz de suportar mais comboios na rede ferroviária.

O estagiário fez inicialmente uma contextualização dos conceitos ferroviários, seguida de uma pesquisa de mercado para soluções existentes e de uma pesquisa na literatura sobre algoritmos de otimização e redes neuronais que pudessem estar ligados à resolução destes problemas. Após as tarefas de investigação, toda a informação recolhida foi analisada e comparada. Foi elaborado um conjunto de riscos e requisitos, funcionais e não funcionais, antes de se passar ao desenvolvimento, teste e avaliação do projeto. Este estágio permitiu avaliar o desempenho de um modelo de Inteligência Artificial (IA) no contexto da programação ferroviária, ajustando o tempo de permanência dos comboios nas plataformas. Foi possível comparar vários modelos para avaliar o melhor de entre eles e depois comparar o modelo selecionado com um algoritmo matemático que é utilizado no produto da CSW. Os resultados revelaram que o desempenho pode ser melhorado e manter mesmo assim um bom nível de precisão.

Palavras-Chave

Sistema de Gestão de Tráfego, Problema de Otimização de Rede, Problema de Automação de Rede, Algoritmos de Otimização, Aprendizagem Reforçada, Inteligência Artificial, Redes Neuronais, Long Short-Term Memory, Rede Neuronal Recorrente, Rede Neural Feed Forward, Rede Neuronal Linear

Contents

Acronyms

ACK Acknowledgement.

ACO Ant Colony Optimisation.

AI Artificial Intelligence.

API Application Programming Interface.

ARS Automatic Route Setting.

ATC Automatic Train Control.

ATO Automatic Train Operation.

ATP Automatic Train Protection.

ATR Automatic Train Regulation.

ATS Automatic Train Supervision.

CBTC Communication-Based Train Control.

CSW Critical Software.

CTC Centralized Traffic Control.

DEI Department of Informatics Engineering.

DNN Deep Neural Networks.

DQN Deep Q-Network.

ERTMS European Rail Traffic Management System.

ETCS European Train Control System.

FFNN Feed Forward Neural Network.

FIFO First In First Out.

FTCMS Full Train Control and Management System.

GoA Grade of Automation.

GSM-R Global System for Mobile Communications-Railway.

IAO Inteligent Optimisation Algorithms.

LMA Limit of Movement Authority.

LNN Linear Neural Network.

LSTM Long Short-Term Memory.

MAE Mean Absolute Error.

MILP Mixed-Integer Linear Programmimg.

ML Machine Learning.

MQ Message Queue.

MSE Mean Squared Error.

NGTC Next Generation Train Control Systems.

NN Neural Network.

PoC Proof-of-Concept.

ReLU Rectified Linear Unit.

RNN Recurrent Neural Network.

SoA State of the Art.

STDEV Standard Deviation.

SVM Support Vector Machines.

TMS Traffic Management System.

UITP Union Internationale des Transports Publics.

UK United Kingdom.

XML Extensible Markup Language.

List of Figures

List of Tables

Chapter 1

Introduction

The railway sector plays a crucial role in modern transportation by facilitating the movement of goods and passengers efficiently, safely, and environmentally consciously. However, two major challenges need to be addressed to make significant progress towards the potential of public transport: network optimisation and route automation. By solving these well-known problems, we can improve the overall customer experience and provide a more practical, economical, and eco-friendly alternative to private transport, such as cars. Furthermore, this can lead to more efficient and safe travel while meeting customers' demands during critical hours and avoiding train delays and cancellations.

In recent years, the biggest and most important railway companies have identified and recognised the pressing need to address these challenges. Recently, many innovative and pioneering initiatives have emerged intending to improve environmental sustainability. These efforts often rely on advanced technologies like artificial intelligence, machine learning, and advanced sensor systems. Companies invest heavily in research and development to create intelligent algorithms to solve environmental problems. These solutions use advanced sensor systems and machine learning capabilities to optimise network layouts and automate routing decisions, directing to a new era of operational efficiency and reduced delays.

Critical Software (CSW)[SA., 2024], a company specialising in providing mission-critical information systems, software and engineering solutions, is currently working on an Traffic Management System (TMS) for a project from one of its clients. The product itself is self-contained, and it was created at the end of the century when it still belonged to another company. For the last few years, CSW has been developing some improvements/customisation for this specific TMS via the Full Train Control and Management System (FTCMS) project. The company that initially owned the product ended up selling part of its business, and the FTCMS became the property of a new company.

When this project was proposed to CSW, it was demanded that the FTCMS must be able to communicate with an external system inflow relating to information, resulting in the need to make changes to around 25% of the product's components. However, some critical components have not evolved for over 15 years. Inside the FTCMS, there is a TMS that aims to automate the routes and support the trains running in the network. The TMS uses mathematical models to maintain the disruptions that happen in the network.

During the second semester, triggering a mitigation plan for one of the risks was necessary, which led to a significant change in the development plan and partially in the State of the Art (SoA). As expected, the initial chapters submitted in the intermediate delivery were updated with the required changes to present and explain the new content introduced in the report. In the second semester, the author could decode and fully understand the messages that contained the information that would serve as input for the Proof-of-Concept (PoC). This moment of understanding revealed that the component studied and improved does less than expected, and our implementation was slightly off target. Therefore, it was required to research and analyse Neural Networks (NN) due to a shortening in the development plan, which will be better explained in Section 5.3 when the study is presented.

Additionally, due to confidentiality affairs, the names of the components, the product and the CSW's client were omitted. Therefore, the names in the report are more generalist versions, so they cannot be linked to the actual names.

1.1 Motivation

The TMS's mathematical models have been used in almost all product releases and have not changed since 2006 when the product was in the initial versions and still belonged to the first company. The product is currently in version 5.15, and some components have suffered some changes. Still, these optimisation algorithms have not been updated; they have only been subjected to minor fixes.

Considering the latest improvements in the machine learning area, there is the possibility that the algorithms may be outdated. This outdatedness is not a sure thing, as the algorithms can still deliver some results. Still, there is the desire, by CSW, to improve these results given that nowadays it is possible to handle more data faster. The consequences of outdated algorithms may be reflected in different aspects, such as not producing the best solutions, not achieving speed performance, or even being incapable of adequately supporting a specific train capacity in the network by not solving the disruptions that appear during execution in the timeout given.

1.2 Objectives

Based on the limitations presented earlier and the desire to improve the FTCMS, CSW created this internship and proposed it to Department of Informatics Engineering (DEI). The final goal of this internship is to develop a proof-of-concept that demonstrates how Artificial Intelligence (AI) can be implemented in TMS to improve network optimisation problems solving. The remaining objectives of this internship are related to the limitations identified in Section 1.1 and to some requirements made by the client. As such, this project's goals are the following:

- The final client requires that the product must be able to maintain and support up to sixteen trains in each direction, making a total of thirty-two trains. CSW wants to take a step forward and develop an algorithm that is capable of supporting, without disruptions, twenty trains in each direction, provoking an improvement of eight trains;
- Decode and analyse the messages exchanged between components that will serve as input and output of the PoC. It is also required to develop a PoC capable of replicating the behaviour of one of the components using an AI model to execute the same tasks.
- Match the speed performance by making the algorithm provide solutions faster than the mathematical algorithms that the FTCMS's TMS uses;
- Equalise the quality of the solution provided by the application. Since we are competing against a mathematical model, one of their most substantial advantages is that they can always give the best solution for the problem regardless of its current state. The objective for Proof-of-Concept's model is to provide the best (or correct) answer in at least 65% to 70% of the cases. The model should be rejected if it cannot consistently achieve at least 60% of correct answers.
- Develop an algorithm endowed with the capability to learn continually and enhance its performance over time, leveraging insights gained from its training data and refining its responses based on past interactions and outcomes.

1.3 Structure

The remaining chapters, Chapters 2 to 11, are organised in the following way:

Chapter 2 defines the methodology and plan used during the first semester and presents the plan for the next semester. A Gantt Chart accompanies the methods and planning of each semester.

Chapter 3 presents the risks identified for this project. Each risk is defined by its name, impact, and probability. In appendix's Chapter C, there is a detailed analysis of each risk with the risk itself, its origin, the impact, the probability and the mitigation plan.

Chapter 4 explains all the railway concepts that were considered relevant and highly important. Additionally, it demonstrates the architecture of the FTCMS, the simulator used to retrieve crucial information about the messages exchanged between some components and will be used to test the algorithm's developments.

Chapter 5 presents a brief explanation of the solutions found in the market. Further, literature research is presented to select the optimisation algorithms that are possible candidates for implementation. At the end of the chapter, there is a literature review. During the second semester, a section about Neural Networks (NN) was added to the literature research and an analysis of what led to that addition.

Chapter 6 describes the set of requirements, functional and non-functional, that were defined to achieve the goals of the project successfully. In this chapter, a table summarises the requirements identified. In appendix's Chapter F, there is a complete description of each requirement, with the description of the requirement, the MoSCoW classification and the estimated effort for it.

Chapter 7 describes a fundamental explanation of the messages of the FTCMS and how the information should be interpreted to be the input of the Proof-of-Concept (PoC).

Chapter 8 will present all the developed work during the second semester and how the classes developed are linked to the requirements defined earlier.

Chapter 9 will explain the detail of all the tests made to the PoC and the results produced when comparing NNs between them and against the **clientSolver** from **clientSolverApp**.

Chapter 10 will present some approaches that can be taken for future implementations and ways to continue the work developed in this internship.

Chapter 11 will conclude the report by summing up what the reader learned about the project, highlighting the work produced and stating the contributions made to the project's theme.

Chapter 2

Methodology & Planning

This chapter will address methodology and work planning. This internship will be ten months long and divided into two semesters. A methodology and an idealised work plan will be used each semester, presented in Sections 2.1 and 2.2.

2.1 First Semester

In the first period of this internship, a group of research tasks that needed to be developed in parallel were defined and did not follow a formal methodology. These tasks were supported by biweekly meetings held with Department of Informatics Engineering (DEI) mentor, Professor Vasco Pereira, to keep him updated on the work being done and to evaluate if the proposed objectives were being achieved. Three of these meetings were attended by Engineer Paulo Carreiro, from Critical Software (CSW), to allow the mentors to discuss important matters related to the internship and its objectives. In addition to these meetings, the student (and trainee) held weekly meetings with Engineer Paulo Carreiro so that he could help and guide him on more specific issues related to work that needed to be produced.

During the first semester, the tasks were related to multiple kinds of research required to build the necessary foundations. Next are listed the tasks made during the first semester:

- Understand the market and the railway concepts (Chapter 4)
- Conduct a market research to assemble a State of the Art (SoA) composition (Section 5.1)
- Examine the literature for optimisation algorithms that might be possible candidates (Section 5.2)
- Analyse the results obtained from the literature research (Section 5.4)
- Assemble the intermediate report

The Figure 2.1 presents a simple version of a Gantt Diagram produced for the first semester. The full version can be visualised in Appendix A, Figure B.1.

Figure 2.1: Work plan of the first semester

2.2 Second Semester

For the second semester, the author was integrated into CSW's team that is working on FTCMS project that follows an agile methodology, **scrum**: a project management framework that facilitates collaboration in teams. This methodology allows independent work from each team member and, at the same time, a partnership between all to help solve the problems of a specific member. The tasks are evaluated and selected considering a fixed time frame, also known as **sprint**. In this case, each **sprint** corresponds to two weeks. In each sprint, the collaborators must analyse and choose the tasks they will be working on and estimate the time they will spend on them. Incorporating this project with a methodology like this allowed the author to speak with other collaborators and expose some difficulties. It also gave him more expert guidance to plan his tasks and better control over what must be done.

During this semester, the main tasks of the author were the development of the Proof-of-Concept (PoC), writing the report and analysing the messages from FTCMS. In early sprints, the author focused more on two tasks: debugging the messages since it was vital for successfully developing the PoC and starting the development itself by producing other parts of the PoC that were not related to the algorithms. When the debugging ended, the focus shifted to researching a new concept to be inserted in the State of the Art (SoA): Neural Networks. After researching and implementing some NNs, the PoC reached a stable point in the development so the author could start the training and testing the model. Finally, the report writing task was always present in the sprints but only became a priority closer to the final delivery date. In Figure 2.2, it is possible to envision a simpler version of the Gantt Diagram of the work plan for the second semester. A slightly more detailed plan can be found in the Gantt Diagram of Figure B.2.

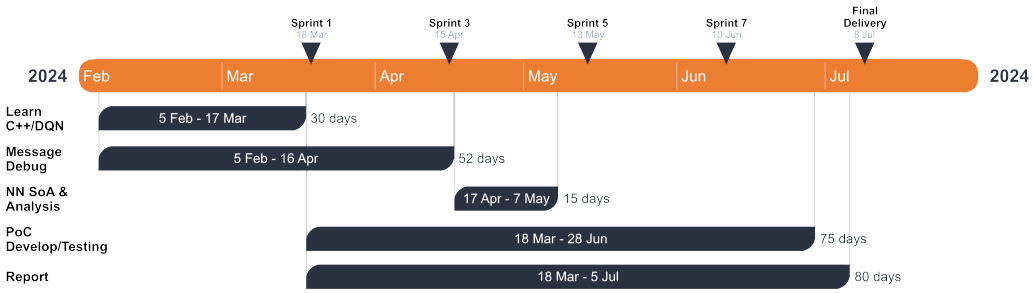

Figure 2.2: Work Plan of the second semester

Chapter 3

Risks

This chapter presents the risks identified during the internship. Each risk will be analysed, considering its impact and probability of occurring. A mitigation plan is also prepared for each of them.

As with all projects, risks are always present, and they must be considered. These risks were deemed necessary, and preventing them from making the project fail was meaningful. The analysis of the risks must be a continuous activity to keep in order what might happen to the project and what mitigation plan should be followed to minimise their impact after they occur.

Each risk is characterised by an impact level (low, medium, or high) representing its importance on the project and its success. This criterion means that a risk with high impact has a lot of dependency on the project's success and, if not mitigated, might provoke the project's failure. A medium-level impact represents a risk that cannot induce failure, but achieving the project's success is much more challenging. Finally, a risk with low impact does not affect the project's success or its conclusion but will produce a step back in the development.

In addition to the impact level, the risk analysis includes a probability of each risk becoming a reality. Similarly to the impact level, this characteristic has three tiers (low, medium and high). The probability of happening and the impact level will help us outline a proper mitigation plan for each. A more significant impact and a more prominent likelihood make the risk more dangerous to the project, meaning defining a well-established plan is vital. All the identified risks were summarised in Table 3.1, and Chapter C of the appendix presents a more detailed analysis and a mitigation plan.

Non-functional Requirements			
ID	**Name**	**Impact**	**Probability**
Risk 1	Lack of documentation	Medium	High
Risk 2	Lack of time to learn the programming language	Medium	Low
Risk 3	Lack of time to learn the algorithm	High	Low
Risk 4	Tasks delivered late	High	Low
Risk 5	Wrong estimations	Medium	Medium
Risk 6	Changes in the development plan due to unknown information	High	High

Table 3.1: Risks

At the beginning of the second semester, in one of the first revisions of the risks, a new risk (Risk 6) was identified related to preventing an eventual change in the development plan. This risk was identified because the application we were working on was considered a black box. This classification occurred after confirming that no formal documentation of the messages existed and that no Critical Software (CSW) employee knew how it worked. Therefore, the development plan was built based on uncertain aspects because no one knew the content of the messages exchanged between components.

The next chapter will present some concepts necessary to understand the project's scope and the application into which it is inserted.

Chapter 4

Background Knowledge

This section will give an overview of the railway concepts of this research and how they are related or compared. A clear view of these concepts is essential because they are used across this report and must be explained beforehand. The Section 4.1 is logically organised where the first is a more independent component, which does not need to refer to other concepts, and then we will start approaching more complex definitions that will require the previously presented concepts.

4.1 Concepts

Understanding complex system definitions is necessary to comprehend the product and market solutions better. Understanding the more straightforward concepts is essential before digging into the complicated concepts. In addition to this section, Chapter A of the appendix presents more concepts that were important to get to know better the railway world but were not considered crucial for this project.

Dwell Time

The total time from when a train stops in a station until it resumes moving. This information is usually passed to the train when it arrives at a station[RailSystem].

Automatic Train Operation

The Automatic Train Operation (ATO) replaces some of the functions carried out by the Train Operator. Thinking of a manually driven train, it is possible to comprehend that the driver initiates the train's starting, allowing its acceleration to the permitted speed, slowing it where necessary for speed restrictions, and stopping at designated stations in the correct location. The ATO system will manage the train's tractive effort (motoring or braking) and stop the train at the next stopping point, commonly a station. The train operator will only need to start the locomotion. A usage example

of this subsystem is the ATO spots. These spots give the train its station stop commands, where usually the last one in a station stop sequence transmits data about the time the train should stop (the dwell time) at the station and may tell it how fast to go to the next station [Gubler, 2023]. This system operates under information usually received from the trackside signalling equipment or traffic control system [RailSystem].

Automatic Train Protection

The Automatic Train Protection (ATP) is responsible for ensuring the safe movement of all trains on the railway. Its function is to monitor the position of all trains and infrastructure on the railway network and establish the Limit of Movement Authority (LMA) and maximum safe speed for each train. This boundary is based on the train's current speed, its braking capability and the distance it can go before it must stop. The ATP applies the emergency brakes if it detects unsafe conditions.

ATP is implemented via ATP control units and ATO spots, that are also called beacons or balises. The control units are positioned between two different rail blocks, i.e., an extract of the railroad. These control units communicate with each other to share information about the railroad. Usually, the ATP control unit receives data from blocks ahead of its position to understand the next rail blocks' conditions.

Regarding a manually driven train, the driver manages this through a combination of route and stock knowledge and the visual information received from wayside signals. On the automated train, the data for the LMA is transmitted from the track to the train, where the onboard computer registers the current speed and calculates the target speed that the train must reach and by when. This is electronically plotted in the form of a Braking Curve. If the train can exceed the curve profile, the brakes will automatically apply to bring the train to a stand or within the permitted speed [Gubler, 2023].

Automatic Train Supervision

The Automatic Train Supervision (ATS) monitors the state of the railway and requests ATO and ATP functions to activate train movements according to the timetable. The ATS receives the data sent by ATO and ATP and compares it with the schedule to determine if the train is running according to schedule or is late. To adjust the train's timing, the ATS can send commands to the ATO spots along the track if they transmit flexible data to the train or the ATP Control Unit [Gubler, 2023]. Figure 4.1 represents an excellent example of how the systems work together.

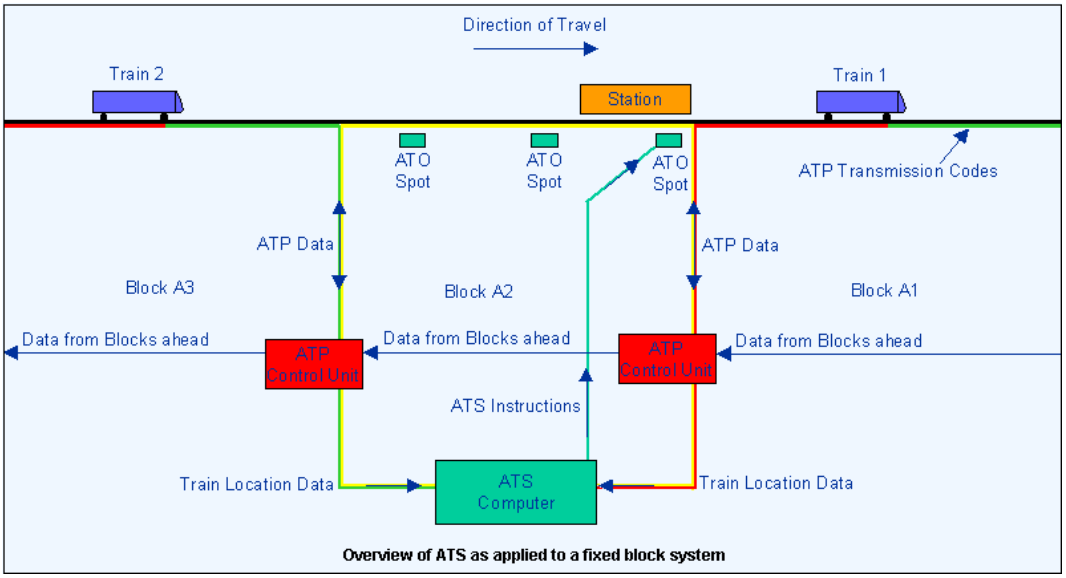

Figure 4.1: ATO, ATP and ATS cooperation in a railway track [Gubler, 2023]

Automatic Train Regulation

The Automatic Train Regulation (ATR) employs changes to train movement profiles and stopping times to deliver the service according to defined strategies, typically either to provide the timetable, to regulate service intervals, or a combination of both. It guarantees the train service returns to the timetabled operation or regular, fixed headways following disruption. The ATR subsystems adjust the performance of individual trains to maintain schedules and are typically a subsystem ATS [RailSystem].

Traffic Management System

The Traffic Management System (TMS) is the main component that controls everything that happens. The TMS is a system that helps improve railway service performance. It is designed to manage train operations to ensure speedy and efficient recovery when unexpected traffic disruptions occur. As a well-established, highly automated hands-off system, the TMS supervises trains and manages railway traffic. It makes trains run following timetables of the day and conducts real-time replanning to handle disruptive situations [Hitachi, b]. The TMS refers to the whole system, which includes all the other automatic functions and might include a degree of manual intervention for some of these functions. The TMS, therefore, might be a package which consists of the following:

- Automatic Train Protection
- Automatic Train Operation
- Automatic Train Supervision

In all systems that use TMS, like UITP, CBTC, and ETCS, the only component that is considered mandatory is ATP. The ATO and ATS are either optional components inside the TMS or complementary but separate components of the TMS [Martin, 2020]. The TMS aims to provide a safe and efficient movement of trains through a series of track circuits and integrated logic for routing and speed controls. The TMS can be considered the general train protection system that responds to external inputs. TMS can have a similar name: ATC.

Automatic Route Setting

The Automatic Route Setting (ARS) is a subsystem of railway TMS that automatically sets routes according to the trains' planned timetables, train descriptions, and facility situations. The system considers conditions like train operations, track layouts, and facilities' gear [H. Teshima, 2014]. The subsystem aims to reduce the workload on signalers at their workstations by running the timetable, setting routes, and regulating trains automatically [Ford, 2020]. This subsystem only works if the train and railroad communicate with a system.

4.2 Full Train Control and Management System

The product used as a work base is the Full Train Control and Management System (FTCMS), a TMS whose main objective is to monitor train operations, ensure that all of them are on time, and maintain the network without any enforcement. This product is currently under development by Critical Software (CSW) for a project of one of its clients. The main concern is that the algorithms used to solve the train delays haven't been evolving since a deployment made in 2004 in Porto's Metro. The Figure 4.2 presents a simpler version of the architecture created for the FTCMS. This diagram was made to preserve the privacy of the project [Yang, 2023].

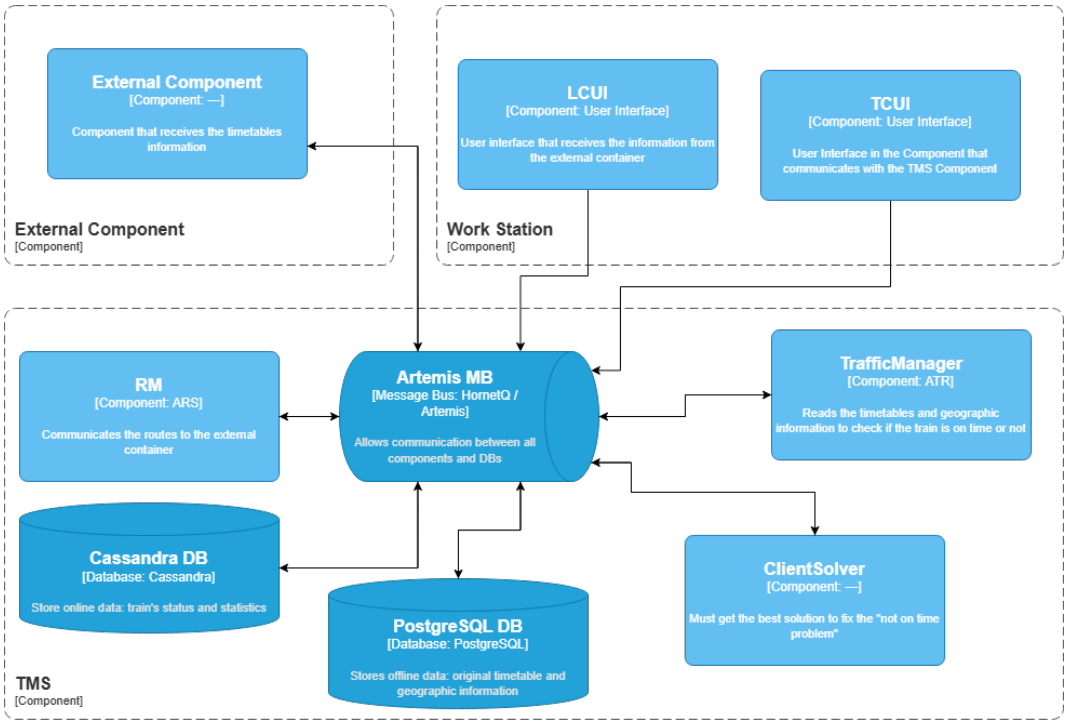

Figure 4.2: FTCMS architecture simplification

Inside this TMS system, there are crucial components that will be critical to the development of this thesis and require a more detailed explanation.

- **routeManager:** a component that can be identified as an ARS that communicates the routes to an external component;
- **trafficManager:** a component that can be identified as an ATR that reads the timetable and the geographic information to check if the train is on time. In case the train is behind schedule, the component sends a message to the **clientSolverApp** to fix the problem that has been raised. An answer message is returned with a possible way to resolve the issue, and then the **trafficManager** needs to evaluate and select the best option to act;

- **clientSolverApp:** a component whose only function is to solve the problems sent by the **trafficManager**. It uses mathematical algorithms to achieve a purpose and solve the issues. Inside this component, multiple **clientSolvers** will receive the message from the **clientSolverApp**, which can be considered the server of the **clientSolver**;
- **Cassandra:** a database that stores online data like information about train status and traffic statistics. It can be accessed via **trafficManager** component;
- **Postgresql:** a database that stores offline data like timetable and geographic information and can be accessed via **trafficManager** and **routeManager** components;

More precisely, this master's thesis will focus on developing a new optimisation algorithm to maintain the network without disturbances and solve eventual delay problems. Since both **trafficManager** and **clientSolverApp** use the algorithms, it was decided that it would be good to analyse the messages exchanged between these components and the **clientSolver** inside the **clientSolverApp**. All the messages are sent through a common HornetQ/Artemis message bus.

4.2.1 Simulation Environment

To retrieve some vital information about the messages exchanged between the **trafficManager** and the **clientSolverApp**, the FTCMS uses an internal simulator. This simulator was created to let the developers try their changes in the product in a more controlled environment. The simulator comprises a virtual machine where the previous components are installed with a specific configuration. The virtual machine is where the TMS is up and running, waiting to receive data. Additionally, there is a user interface where it is possible to visualise the trains running and receiving extra information from the TMS. Additionally, this internal simulator allowed us to perform the proper tests to the Proof-of-Concept (PoC) developed.

First, we initiate the virtual machine to start executing the TMS and the **clientSolvers** to prepare them to start solving the problems that arise in the network.

Secondly, the user interface is started and will try to connect to the virtual machine and the TMS inside it. Figure D.1 shows an image of the simulator after establishing a successful connection to the TMS. It is possible to see that the left-top corner shows the user that it is connected to the TMS. After the connection, the user selects the scenario he wants to execute in the simulator, as shown in Figure D.2. The simulation emits a floating message indicating how many trains were allocated in the scenario. In the bottom left corner of the application, it is possible to set the starting date and time of the simulation and the time increment wanted.

After all these steps, the user is finally ready to start the simulation. In Figure D.3, it is possible to see how the traffic view is presented. In it, the geographic network is presented with some squares. Each station has a three-letter abbreviation, and each square represents a train with a unique number. At the top of the view, it is possible to see all the unique train numbers and a map of the whole network if the user pretends to amplify to a specific network zone. Each train has its specifications, and it is possible to check them if the user places the cursor on top of the train's square in the network or if the user places the cursor on the top square, like Figure D.4 exemplifies.

4.2.2 Algorithms

Next, it is crucial to analyse and understand more about the objective functions and algorithms used to solve the delay issues of the trains. Usually, this matter is solved using only one of the functions, depending on the chosen configuration. From the little documentation available, it was possible to identify three objective functions in the FTCMS and some variants. Each function follows a mathematical model to achieve a specific purpose [Wiatr, 2023].

Timetable Mode

The objective is to minimise the deviation from the timetable, if possible, from the current train's position and progressively adjust trains to the timetable. The ATR searches the regulated values that minimise the difference to the planned timetable. With this, the function wants the regulated departure time close to the scheduled timetable. By using a coefficient, it is possible to give particular importance to fulfilling planned timetables on specific train platforms.

Headway Mode

Trains on the line are distributed regularly, and a passing time difference between two trains through a point (headway) is adjusted to a specified value. The ATR searches the regulated departure times for the current station and the next one to keep the headway at a certain level. To maintain the headway, adjusting the departure time of one train and the following one is necessary. The headway regulation is based on minimising passenger waiting times at stations, which leads to evenly distributed traffic. The headway target depends on the number of trains in the line and the line's running times.

Waiting Times Adjustment Mode

The last function considers the *dwell* concept. In a fast recap, this value is when a train stops at a station. The ATR searches the set of dwell values that minimises the difference to the default time defined in the system. This operation suggests approximating the real platform's train waiting time to the platform's time configured in the system.

4.3 Client & Competitors

Before analysing the market and looking for solutions already in use, it is vital to understand which companies rule the market and how significant their influence is [Research and Markets, 2023] [Intelligence, 2022]. It is essential to clearly understand which companies are in the market and how they are inserted. This section will help us to get to know a bit more about our client and other companies that are considered competitors. After getting to know the companies, it will be easier to present and organise the various market solutions found in Section 5.1.

4.3.1 Alstom

Alstom is a French multinational company specialising in manufacturing railway equipment and systems. It is a leading rolling stock manufacturer, including high-speed, suburban, regional, and urban trains and trams. The company also provides signalling, infrastructure, integrated system solutions, modernisation, maintenance, parts and repair, and support services for the transport sector [Wiki, b]. It is in 63 countries with a talent base of over 80,000 people from 175 nationalities and 250 sites [Alstom, a].

4.3.2 Thales Group

A French multinational company that operates in various industries, including rail transport. Thales Group provides a wide range of rail solutions, including rolling stock, rail automation, electrification, and digitalisation. The company offers products and services related to trams, light rail, metro, commuter and regional trains, and high-speed and very high-speed trains. It also provides systems to highly demanding national, regional, and international rail networks located on every continent [Thales, a]. It also offers transport security systems for railways, including electronic communication products and rail safety solutions. The company's manufacturing and installation capabilities are cornerstones of its ability to meet the challenges of any rail safety or communication project, large or small [Thales, b].

4.3.3 Siemens

A German multinational company that operates in various industries, including rail transport. Siemens Mobility is a leading provider of rail solutions, including rolling stock, rail automation, electrification, and digitalisation. The company offers a wide range of products and services related to trams, light rail, metro, commuter and regional trains, and high-speed trains [Siemens, a]. Siemens Mobility has provided seamless, sustainable, reliable, and secure transport solutions for more than 160 years [Siemens, b].

4.3.4 Hitachi Rail

A Japanese multinational company that operates in various industries, including rail transport. Hitachi has a division called Hitachi Rail that operates in the Railway market. It is a leading rolling stock system manufacturer, including high-speed trains, commuter trains, and metro trains [Wiki, a]. The company has a long history of manufacturing railway carriages and maintaining high-speed bullet trains to digital signalling infrastructure [Hitachi, a]. Other key points are Rail signalling systems, Rail infrastructure and Global reach.

4.3.5 Resonate

Founded in 2008, Resonate is a technology company specialising in rail and connected transport solutions. They offer digital railway and connected transport solutions through powerful digital platforms for signalling, traffic management, and operational intelligence. Resonate offers digital railway and connected transport solutions through powerful digital platforms for signalling, traffic management and operational intelligence. They provide software applications, development, support, and advice on signalling technology [Railway-Technology]. With over 50 years of industry experience and a background in British Rail, Resonate has a strong understanding of signalling control, operations management, and logistics [Resonate, b].

Resonate developed Luminate, a digital platform that provides traffic management solutions for the railway industry. Luminate Traffic Management provides a digital intelligence layer that improves performance, increases capacity, and manages disruption. By implementing planning models based on powerful predictive analytics, They expect to deliver advanced automation and a more efficient network [Resonate, a].

Chapter 5

State of the Art

After explaining the background concepts and presenting the companies, it is time to start looking for market solutions that the companies have developed over the years. This research aimed to acquire better knowledge about what is available in the market and what optimisation algorithms should be considered for implementation in this project. It is also necessary to look in the literature for possible ways to achieve the objectives of this project.

Section 5.1 will examine the solutions found in the market, and Sections 5.2 and 5.3 will present literature research on some of the best optimisation algorithms and Neural Networks (NN) to implement in a network optimisation problem and route automation. At the end of the State of the Art (SoA), the information gathered from the literature is evaluated, and the algorithms are ranked (Section 5.4).

5.1 Market Solutions

This section will present the products found in the railway market that might compete with the Full Train Control and Management System (FTCMS). In some companies, multiple products were found. Because of that, it was decided to organise the product by company.

5.1.1 Alstom

Alstom can offer scalable Communication-Based Train Control (CBTC) that supports the full range of automatic train control technologies. Alstom also presents its comprehensive CBTC portfolio: Urbalis^{TM} and Urbalis Flo^{TM}. These solutions are suitable for conventional and intelligent solutions and for any complexity, brown- or green-field system, and at all grades of automation. Upgrades and optimisation can be done without service disruption, which provides a new overlay capability. The system can be deployed quickly, allowing big cities to meet their transportation needs rapidly and reliably, and uses energy-saving strategies, which reduces consumption to be reduced by 30% [Alstom, b].

5.1.2 Thales Group

Thales Group has a Traffic Management System (TMS) that provides permanent control across the network, automatically sets routes for trains, logs train movements and detects and solves potential conflicts. Some benefits are related to competitive operating costs and improving customer services [Thales-Group].

The first benefit enables practical functionality in local or regional systems. It can be operated in older networks, either as an overlay or with the full integration of the route control systems to automate the train routing. It also contains powerful evaluation mechanisms to help to optimise and reduce the cost of activities. The second benefit allows the integration of modern interlocking technology with advanced TMS to increase the throughput on the network due to intelligent conflict detection and solution functionality, helping the operators to make effective and result-oriented decisions [Thales-Group].

Thales Group presents the following solutions [Thales-Group]:

NetTrac MT $\longrightarrow$ the most advanced, open architecture solution available for central control of urban rail systems applying either conventional fixed block signalling or CBTC technology. The system can be configured to support stand-alone Centralized Traffic Control (CTC) or highly integrated applications;

Aramis^{TM} $\longrightarrow$ the most advanced and proven system for intelligent Operation Control Centres. It controls and automates the railway traffic operation and sets routes, supervises the infrastructure, envisions the status of the railway network in real-time, performs a comparison with the planned targets, shows KPIs, optimises resources and calculates forecasts based on the actual data.

NetTrac 6614 Automatic Train Routing System (ZL L2000) $\longrightarrow$ an ARS that automates the operation and discharges signallers and dispatchers from routine work. Together with other systems, it can overlay local interlocking systems or remote control systems for CTC and assumes the automatic train setting of routes.

5.1.3 Siemens

Siemens offers a CBTC system Trainguard^{MT} that helps solve the challenges of many cities with greater train automation in public transport. The objective is to get 100% punctuality, maximum network capacity and energy savings of up to 20% using optimised speed profiles. The Trainguard^{MT} is a versatile and modular CBTC system that can be customised to the railway operator's needs and that is possible to implement different train control and automation levels both in new and existing networks, depending on the client's requirements for performance and functionality [Siemens, c].

5.1.4 Hitachi

Hitachi's TMS is an advanced system that helps improve railway service performance. It is designed to manage train operations to ensure speedy and efficient recovery when unexpected traffic disruptions occur. As a well-established, highly automated, hands-off system, the TMS supervises trains and manages railway traffic, including at depots and sidings. Data management is necessary to coordinate railway operations, so the TMS supports the efficient operation by enabling optimum data management [Hitachi, b].

The key benefits are linked to efficient infrastructure capacity utilisation, reduced workload through repetitive tasks, and the ability to detect and manage potential disruptions proactively. Finally, Hitachi can achieve this with extensive experience installing, maintaining, and monitoring TMS solutions for other railway companies and agencies [Hitachi, b].

5.1.5 Resonate

Resonate has signalling control systems extensively used in the United Kingdom (UK). This system provides comprehensive fault diagnostics and monitoring, plus an ARS that optimise traffic flow, even under disrupted conditions. Their systems control over half of UK mainline train services [Railway-Technology].

5.1.6 Bombardier

In 2010, an article was published about a product offered by Bombardier Transportation that would meet the necessities of an overwhelmed network. The FLEXX Tronic, including roll-compensation (WAKO) and active wheelset-steering (ARS), is characterised by its high reliability and numerous benefits in performance and economy. The product would increase the transport capacity, achieve the optimal speed, and reduce travel time. To achieve the goal of reducing travel time, improving all three previous measures is necessary. The first series-ready version was tested in the framework of the Gröna Taget project in Sweden for a Regina train and certified according to the International Union of Railways for 200km/h [Schneider, 2010]. Although this product belonged to Bombardier, it is now owned by another company.

5.2 Optimisation Algorithms

After looking for solutions available in the market, it was time to look for possible candidates to implement the optimisation algorithm. After extensive literature research and analysis, selecting eight algorithms and comparing them with the one used in the FTCMS was possible. These eight algorithms were the ones that were more connected to network optimisation and route automation when looking for potential solutions in the literature.

These algorithms can improve the speed and quality of solving problems using memory structures, probabilistic techniques, and evolutionary principles. Their capabilities make them well-suited candidates for this project. Next, it will be presented and explained each of these algorithms.

5.2.1 Backtracking

Backtracking is a problem-solving algorithmic technique that involves finding a solution incrementally by trying different options and undoing them if they lead to a dead end. It is commonly used when exploring multiple possibilities to solve a problem. The algorithm backtracks to the previous decision point when a dead end is reached. It explores a different path until a solution is found or all possibilities have been exhausted [GeeksforGeeks, b].

This graph-based algorithm can determine the routes for a given direction by leading the graph edges to ensure proper orientation. It can be used in ARS, which releases the signaller from unnecessary pre-setting of the elements required for establishing the route. Using the backtracking algorithm allows for checking all possible routing solutions. By adding a graphical user interface, this method could be successfully used as a design tool (for the interlocking system) for any railroad station [Catrina, 2015].

5.2.2 Deep Q-Network

The Deep Q-Network (DQN) algorithm combines the Q-Learning algorithm with Deep Neural Networks (DNN). As it is well known in AI, DNN are excellent non-linear function approximations and are used to approximate the Q-function, replacing the need for a table to store the Q-values. This algorithm uses two DNN to stabilise the learning process [Ausin, 2020].

The DQN can optimise the train routing system by prioritising each route. This algorithm can be used in ARS systems to generate a route list based on the route distance and track element availability [Lövétei et al., 2022]. The Q-value function in DQN is represented with weights, and the Q network works like the Q table in Q-learning when selecting actions [De-Yu, 2021].

5.2.3 Tabu Search

Tabu search is an iterative neighbourhood search algorithm where the neighbourhood changes dynamically. Tabu search enhances local search by avoiding points in the search space that have already been visited. Loops in search space are avoided by preventing already visited points, and local optima can be escaped. Solutions can be optimised with Tabu Search [GeeksforGeeks, g].

The basic idea of Tabu Search is to penalise moves that take the solution into previously visited search spaces. Tabu Search, however, does deterministically accept non-improving solutions to prevent getting stuck in local minimums [Liang, 2020].

5.2.4 Mixed-Integer Linear Programming-based Heuristic

A Mixed-Integer Linear Programmimg (MILP) is a problem with the linear objective function, bounds and linear constraints but no non-linear constraints and restrictions on some components to have integer values [MATLAB].

This algorithm can solve real-time railway problems like rerouting and rescheduling trains to minimise delay propagation due to traffic perturbations. This algorithm has proven effective in various contexts [Pellegrini et al., 2019].

The algorithm makes scheduling decisions and heuristically shrinks the search space. Then, these are transformed into a significantly reduced MILP model that can be solved fast, while its results imply further decisions. The algorithm runs many small MILP until the final solution is determined entirely [Élesa et al., 2018].

5.2.5 Neuro-fuzzy moddelling

Neuro-fuzzy is a term used to describe a type of artificial intelligence that combines neural networks and fuzzy logic elements. Neural networks are a type of machine learning algorithm that is used to model complex patterns in data. At the same time, fuzzy logic is a type of logic that allows for approximate reasoning. Combining these two technologies can create more flexible and efficient systems than those that use either technology alone. Neuro-fuzzy systems have been used in various applications, including control systems, image recognition, and data mining [Autoblocks].

These models can work in a hybrid way with other algorithms, like genetic algorithms, to form train routes. This combination can construct an automated control system to form optimal routes for passengers and freight trains. A system like this might be able to create more economical and technological train routes, which will provide the passenger and freight transportation systems with flexibility and adaptability in changing working conditions [Dolgopolov et al., 2019].

5.2.6 Genetic Algorithm

Genetic Algorithms are adaptive heuristic search algorithms based on the ideas of natural selection and genetics. These are intelligent exploitations of random searches provided with historical data to direct the search to the region of better performance in solution space. They are commonly used to generate high-quality solutions for optimisation problems and search problems [GeeksforGeeks, d].

This algorithm has been used to solve railway scheduling problems that imply the optimisation of trains on a railway line that is occupied (or not) by other trains with fixed timetables. The timetable for the new trains is obtained with a Genetic Algorithm that includes a guided process to build the initial population, and the results obtained suggest that this algorithm is an appropriate method to explore the search space of this complex problem and can lead to reasonable solutions in a short amount of time [Tormos et al., 2008].

5.2.7 Ant Colony Optimisation

Ant Colony Optimisation (ACO) technique is purely inspired by the foraging behaviour of ant colonies. Pheromones are organic chemical compounds secreted by the ants that trigger a social response in members of the same species. These are chemicals capable of acting like hormones outside the body of the secreting individual, impacting the receiving individual's behaviour. The whole scenario can be realised through weighted graphs where the ant colony and the food source act as vertices (or nodes); the paths serve as the edges, and the pheromone values are the weights associated with the edges [GeeksforGeeks, c].

ACO has advantages such as heuristic, positive feedback and distribution. ACO algorithm has been widely used to solve the travelling salesman problem. This algorithm was proposed to solve a tourism route planning problem similar to rerouting a train in case of a disruption [Liang, 2021].

5.2.8 Greedy Algorithm

Greedy is an algorithmic paradigm that builds up a solution gradually by always choosing the next step that offers the most prominent and immediate benefit. The problems where choosing locally optimal leads to global solutions are the best fit for Greedy [GeeksforGeeks, e]. This algorithm ignores the fact that the current best option may not yield optimal results. Even if the initial decision is incorrect, the algorithm never reverses it [Simplilearn].

The greedy algorithm is a trendy choice for scheduling problems. It can be used for route setting to find the most efficient path between two points and solve scheduling problems, such as interval scheduling, where tasks or activities must be scheduled on a machine or resource [Johanna, 2011].

5.2.9 Mathematical-theoretical Analysis

A mathematical algorithm can be defined as an algorithm or procedure utilised to solve a mathematical problem or problem that can be solved using data structures and Algorithms. It is a handy topic for solving computer science problems, such as problem-solving, competitive programming, and statistics, and it has vast applications in competitive programming [GeeksforGeeks, f].

These algorithms can be used in a two-step optimisation method that combines with Inteligent Optimisation Algorithms (IAO) to reduce the arrival interval by avoiding unnecessary stops in front of the home signal and increasing the running speed of trains through the throat area. In this application, the recommended approaching speed and position are obtained by analytical calculation [et al., 2023]. This algorithm is currently being used in FTCMS.

5.3 Neural Networks

As explained in Chapter 1, it was necessary to reformulate the development plan due to a better understanding of the messages of the FTCMS that represent the input for the Proof-of-Concept (PoC). This section was required after that development change, where it was decided to implement a Neural Network (NN) instead of a whole optimisation algorithm. This decision is based upon the foundations of work already done in the first semester, during which an analysis of optimisation algorithms was conducted. The Neural Networks (NN) are designed to reproduce what is known of the human brain's neural connections. Therefore, NNs can learn linear and non-linear relationships. Another advantage is the wide range of types of neural networks and layers that can be used to build them. Each presented NN offers different capabilities for treating data and mapping the relationship between the input and the output.

5.3.1 Feed-Forward Neural Network

A Feed Forward Neural Network (FFNN) is a type of neural network in which the data flows unidirectionally from the input to the output. It consists of an input layer, where the data is received, one or more hidden layers, and finally, an output layer. The input layer receives the data, the hidden layers process the data, and the output layer produces the predictions. Every neuron in one layer is connected to all neurons in the next. During training, the weights are adjusted using a backpropagation algorithm to minimise the error between predicted and actual outputs [Fred, 2023] [Turing, 2024] [Pytorch, 2024].

Although they can represent the simplest type of neural networks, they can also model non-linear relationships between input and output. It is possible to identify the non-linearity relationship using non-linear activation functions like the Rectified Linear Unit (ReLU), Sigmoid, or Tanh functions [Fred, 2023] [Turing, 2024].

5.3.2 Linear Regression Neural Network

The Linear Neural Network (LNN) is a neural network that uses linear regression to model relationships between the input and the output.

Like the feed-forward, the LNN is one of the simplest models for identifying a linear relation between input and output. The model is less complex than others, leading to a more straightforward interpretation. However, its simplicity limits its ability to identify more complex and non-linear relations between variables [Fred, 2023].

5.3.3 Recurrent Neural Network

The Recurrent Neural Network (RNN) is a neural network that handles sequential data, where the output at a given time step depends on the input of previous time steps. This model has a hidden state with feedback connections capable of storing information of earlier steps, allowing it to capture long-term dependencies in the data [Alshehri, 2021] [Fred, 2023].

The disadvantages of RNNs are often linked to the vanishing or exploding gradient problem. In the first case, the gradient becomes smaller at each step of the backpropagation, while in the second case, the opposite situation occurs. Another disadvantage of the RNN is its training difficulties for large sequences [Alshehri, 2021].

5.3.4 Long Short-Term Memory Neural Network

The Long-Short-Term Memory (LSTMNN) neural network is a variant of RNN designed to handle sequential data. The main differences between these two models are how they handle the data and learn long-term dependencies.

Like the RNN, the LSTM has a hidden state capable of storing information. However, the LSTMNN uses a particular cell, called LSTM, that can maintain the information for extended periods and works on a read, write and forget basis. The memory cell uses gates to control the flow of the data inside of it. Inside a cell, there are three gates: the forget gate that filters data that is not relevant, the input gate that controls the data that enters the cell, and the output gate that maintains the flow between the cell and its output [Alshehri, 2021].

The LSTM is also less prone to the gradient problems presented for the RNN but can suffer from overfitting, where it adapts too much to the training data and then performs poorly in testing when facing new data. Additionally, the LSTM can require more computational resources than other neural networks and can become more challenging to explain the results obtained [Fred, 2023] [Coursera, 2024].

5.4 State of the Art Analysis

Unfortunately, when it came to looking a bit deeper into the solutions found in the market (Section 5.1), it was not possible to check them in more detail, specifically on what algorithms they used. This impossibility happens because this part of the product is considered a core business, making the companies not reveal many details about what we need. However, creating a more visual comparison between the products presented and the Full Train Control and Management System (FTCMS) is still possible. This comparison is made considering the concepts presented in the Section 4.1, that addresses the multiple components that construct a TMS, the difference between UITP, CBTC, and ETCS, and the different levels of GoA.

In parallel, after selecting eight algorithms, a second analysis will be done with a more profound evaluation to rank them considering the problem they are set to solve. After all the research looking for good optimisation algorithm possibilities to implement, it is time to start evaluating and ranking the algorithms according to the best option for route automation and network optimisation. Next is a brief explanation of the pros and cons of the previously selected algorithms.

5.4.1 Products Comparison

After analysing all the products in the market, it is possible to compare them with each other and the FTCMS considering the definitions they follow (CBTC, ETCS or UITP), the type of the product and the GoA level. This kind of comparison allows us to place our product alongside the ones in the market and helps us understand what the real competitors of our product might be. Table 5.1 compares all the products regarding the concepts explained in the Section 4.1.

From the Table 5.1, we can realise that the FTCMS is nearly matched with multiple products in the market, which makes it even more vital to take a step forward in other terms like the optimisation algorithms. This optimisation would improve the system and close the gap between the different products even more or beat them in the best-case scenario.

Product	Product definition	Product type	GoA Level
FTCMS (CSW's Client)	CBTC	TMS	GoA 4
Urbalis^{TM} (Alstom)	CBTC	TMS	GoA 4
Urbalis Flo^{TM} (Alstom)	CBTC	TMS	GoA 4
FLEXX Tronic (Bombardier)	Not a system	Technology with ARS	Not applicable
NetTrac TM (Thales)	CBTC	TMS	GoA 4
Aramis^{TM} (Thales)	ETCS	TMS	GoA 2
ZL L2000 (Thales)	Not a full system	ARS	Not applicable
Trainguard^{MT} (Siemens)	CBTC	TMS	GoA 4
TMS (Hitachi)	CBTC	TMS	GoA 4
Signalling control systems (Resonate)	ETCS	TMS with ARS	GoA 2

Table 5.1: Comparison between market solutions and the FTCMS

5.4.2 Algorithms and Neural Networks

While researching and learning about the potential prospects for the implementation, some desired specifications, like the usage of memory structures to continuously learn, were detected. On the other hand, other characteristics that were not wanted, like the possibility of quickly becoming a black-box model, were also identified.

Firstly, some algorithms, like greedy algorithm and MILP-based, may not find the optimal solution. The reason behind this may vary, whether it is related to performing local moves (greedy algorithm) [Simplilearn], the nature of the problem (MILP-based) [Pellegrini et al., 2015]. The backtracking algorithm has a significant drawback: it can be inefficient as it often explores the same parts of the search space multiple times. This repeated exploration can lead to redundant calculations, significantly slowing down the time it takes to find a solution [GeeksforGeeks, a].

Secondly, the Mathematical-theoretical Analysis, used in FTCMS, is supported by highly developed mathematical theory but lacks attention to the model solutions' theoretical characterisation and the actuation function space [You, 2022]. Similarly, neuro-fuzzy modelling is often criticised for its lack of transparency since the fuzzy logic relies heavily on human knowledge and expertise to function effectively [Goyal, 2022].

Contrariwise, Tabu Search, ACO and Genetic Algorithm have been applied to train scheduling and rerouting operations, justifying that the first selection was well based [Corman et al., 2010a] [Sahana, 2014] [Wegele et al., 2003]. In addition, the DQN has also been successfully applied to train scheduling problems and leverages deep reinforcement learning to make informed decisions selecting action which corresponds to the maximum Q-value [Gong et al., 2020], offering a promising approach for our concerns of network optimisation and route automation problems.

To summarise the previous paragraphs, a table was created to help compare the algorithms. Additionally, a three-level (high, moderate and low) computational complexity of the algorithms was added and if the algorithms have already been used in this type of problem. This information will enhance and help us make a comparison. The Table 5.2 presents each algorithm's pros, cons, usability in this context and computational complexity level.

Algorithm	Advantages	Disadvantages	Previously used	Computational Complexity
ACO	Adaptability; Near-Optimal Solutions	Slow Convergence Speed	Yes [Sahana, 2014]	High [Li, 2015]
Backtraking	Simplicity	Low Performance;	No	High [Carter et al., 1985]
DQN	Experience Replay	Training Time	Yes [Kim et al., 2023]	High [Du, 2016]
Genetic Alogrithm	Adaptability; Global Optimization	Computational Cost; Parameter Tuning	Yes [Vijini, 2020]	High [Lissovoi and Oliveto, 2019]
Greedy Algorithm	Simplicity; Incremental Solution	Incapacity of learning; Local move actions	No	Low [Codility, 2021]
Mathematical-theoretical Analysis	Predictive capability; Near-Optimal Solutions	Complexity; Easily Black-box	Yes [Wiatr, 2023]	Moderate [Tantrasuwan, 2018]
MILP-based Heuristic	Effective; Feasible Solutions;	Problem-specific; Easily Black-box	No	Moderate [Pellegrini et al., 2019]
Neuro-fuzzy Modelling	Hybrid Model; Predictive capability;	Complexity; Easily Black-box	No	Moderate [Lark, 2023]
Tabu Search	Avoid Local Optima; Memory Utilization	Slow Convergence Speed	Yes [Corman et al., 2010b]	High [GeeksforGeeks, g]

Table 5.2: Comparison of algorithm features

After analysing the pros and cons of the algorithms, considering the explanations given in this chapter and their applicability in the context of railway operations, it was possible to understand that some algorithms are better for network optimisation problems than others. Additionally, most of the algorithms were only used in similar cases. The analysis has revealed that some algorithms are more promising and successful when addressing complex routing problems and improving railway network efficiency. It was possible to rank the algorithms initially considered using all this information. The following list presents the rank made to the literature review.

1. Deep Q-Network
2. Tabu Search
3. Ant Colony Optimisation
4. Genetic Algorithms
5. Backtracking Algorithms
6. MILP-based Heuristic
7. Neuro-fuzzy Modelling Algorithms
8. Greedy Algorithms
9. Mathematical-theoretical Analysis Algorithm

Unfortunately, a deeper comparison between the algorithms could not be performed without effectively implementing them. With this information, it was decided that the algorithm to be implemented would be selected considering the analysis and the rank made after that. With this classification, the author decided that the initial development plan was to implement a DQN because it allows us to choose which type of neural network can be implemented and adapt its input and output to such a complex environment.

The author shortened the implementation after the update on the development plan by only developing a Neural Network (NN) instead of creating a full DQN. Despite this change, the objective of understanding how an Autonomous Learning Model can be used in the railway world remains. While analysing the NNs presented in Section 5.3 since all of them have their strengths and weaknesses, the author decided to implement all four NNs to test and evaluate them so the NN that would perform better in testing could be selected. Section 8.2 will explain how the NNs were built in the PoC. The next chapter, Chapter 6, will identify this project's functional and non-functional requirements considering the State of the Art produced.

Chapter 6

Requirements & Risks

The following chapter will present the software requirements, functional and non-functional, identified to develop the proof of concept. These requirements represent the functionalities or attributes related to performance and usability. The first section of this chapter will explain the functional requirements, while the second will present the non-functional requirements necessary to achieve the project's goals.

6.1 Functional Requirements

These requirements are connected to the functionalities that must be developed and represent the defined functionalities to be implemented in the PoC.

Firstly, to enable the integration of the new model in the Full Train Control and Management System (FTCMS), the model must receive and send messages to the message bus. For this, a serialiser must be developed to decipher the message received. It is also necessary to create a component capable of inverting this process.

Secondly, the model must be able to create a problem object. This class places the content sent in the messages in their respective structures and, later in the execution, can add any additional information sent in other messages, like the Solve message. With all the data processed and structured in the PoC, it is necessary to start building the solution. For that, the PoC creates an empty solution, defined as a copy of the current state of the problem, and then solves it to find the solution to the problem.

Lastly, the model must solve the problem received: adjusting dwell times to fix any timetable disruption, whether a delay or an overrun. After solving the problem, the model must update and return the parameters in a new message.

The functional requirements previously explained are summarised in Table 6.1, and it is possible to find the complete details about each requirement in Section F.1 of the appendices.

Functional Requirements			
ID	**Name**	**MoSCoW**	**Table**
FR1	Read messages from the message bus	Must	Table F.1
FR2	Send messages to the message bus	Must	Table F.2
FR3	Serialise and Deserialise messages	Must	Table F.3
FR4	Create a problem instance	Must	Table F.4
FR5	Add additional data to a problem	Must	Table F.5
FR6	Create an empty solution	Must	Table F.6
FR7	Find a solution to the problem	Must	Table F.7
FR8	Update the content received in the problem	Must	Table F.8

Table 6.1: Functional Requirements

6.2 Non-Functional Requirements

While the functional requirements are more connected to the software's development and functionalities, the non-functional requirements are more related to the application's usage, performance and usability.

The first non-functional requirement is related to the programming language that could be used for the PoC. Since the entire FTCMS is built in **C++**, it was considered a possible requirement that the PoC could be built in **C++** too.

The following requirements are more connected to the performance of the PoC. First, the model must avoid delay propagation by solving and adjusting the dwell times. Then, the model should also minimise the impact of new traffic by keeping the trains running according to the timetable.

Then, this project could improve the speed performance by responding faster than the current **clientSolver** being used, and the results must be comparable with the ones produced by the **clientSolver**. The requirement explicitly covers the answer's speed and the value obtained after solving the timetable disruption. This value represents the remaining disruption after the **clientSolver** execution. Finally, the model must achieve in testing the non-rejection accuracy defined in one of the objectives of this project.

The Table 6.2 presents a summary of the non-functional requirements of this project. The entire content of these requirements can be found in Section F.2 of the appendices.

The Chapter 7 will present all the information about the messages exchanged between the **trafficManager** and the **clientSolverApp** and how this information provoked a change in the development plan.

Non-functional Requirements			
ID	**Name**	**MoSCoW**	**Table**
NFR1	C++ as Programming Language	Could	Table F.9
NFR2	Avoid delay propagation	Must	Table F.10
NFR3	Minimise the impact of new traffic	Should	Table F.11
NFR4	Improve the speed performance	Could	Table F.12
NFR5	Results comparability	Must	Table F.13
NFR6	Model's Accuracy	Should	Table F.14

Table 6.2: Non Functional Requirements

Chapter 7

Full Train Control and Management System Messages

This chapter will present all the information and explain the messages exchanged between the two main components of the Full Train Control and Management System (FTCMS), the **trafficManager** and the **clientSolverApp**. This chapter aims to clarify the content of the black box previously identified in Chapter 3 that led to the creation of Risk 6. The first section will comprise multiple subsections, each presenting a type of message and its function in the complete process of the FTCMS.

7.1 Types of messages

The messages traded between the **trafficManager** and the **clientSolverApp** are naturally related to the delay concerns. When the system detects a disruption to the timetable, the **trafficManager** constructs the problem and broadcasts it to the **clientSolver**'s instances of the **clientSolverApp**. The **clientSolvers** must acknowledge the problem so the **trafficManager** can seal the problem. Then, the **clientSolvers** will send a work request to the **trafficManager**, and one of the **clientSolvers** will receive a second message with additional information about the problem.

The **clientSolver** will then find a solution to that problem and return it to the **trafficManager**. After receiving the solution, the **trafficManager** will broadcast a message so the **clientSolver** can drop the problem and reset themselves for the subsequent disruption detection.

The Figure 7.1 shows the message flow that happens between the **trafficManager** and the **clientSolvers** from **clientSolverApp**.

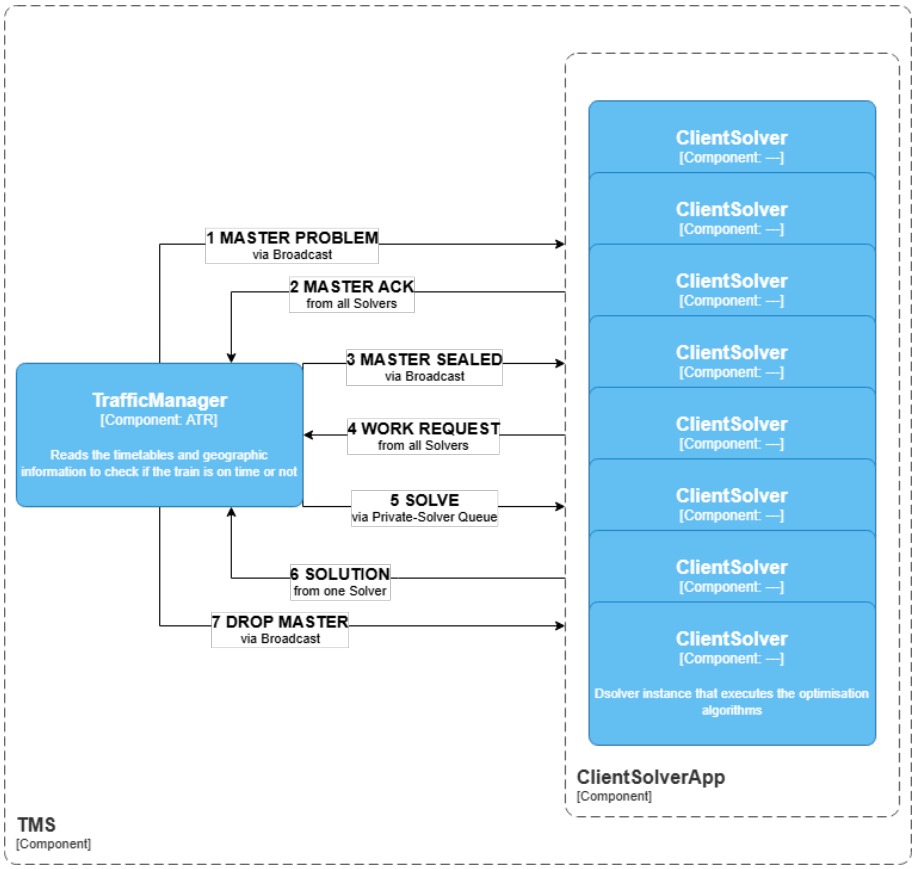

Figure 7.1: FTCMS message flow

In addition to this flow of messages, the **trafficManager** and the **clientSolvers** exchange heartbeat messages to check if the services are working and operating correctly. The **trafficManager** broadcasts a Solver Hello Request, and each **clientSolver** responds with a Solver Hello that is acknowledged by the **trafficManager**.

The following subsections (Sections 7.1.1 to 7.1.7) will explain each message present in Figure 7.1. The information about the content of the messages was obtained in the middle of the second semester when part of the implementation was already underway.

7.1.1 Master Problem

This message is the first created and sent by the **trafficManager** when a problem arises. A MasterID identifies this message type and contains three main components: Vars, Sets and Params.

Vars

A group of elements that contains the forecast for each service's following three stations.

Each element is labelled by a global unsigned identifier (GUID) representing a station and a platform. Additionally, it contains a default current dwell time (Cur), a minimum (Min) and a maximum (Max) value for the new dwell value that will be set bet the **clientSolver**.

For each service trip, there will be six elements. The first three represent the original forecast, which will remain untouched, and the other three are the elements that will be modified by the **clientSolver** with the solution.

Sets

The **trafficManager** defines a group of sets to indicate which elements the **clientSolver** must use to handle the problem correctly. Later in the process, when the **clientSolver** receives the Solve message, the **trafficManager** sends additional information with a set ID to let the **clientSolver** know what it should use to solve this part of the problem.

Params

Several parameters are used for the **clientSolvers**' linear algorithms in **clientSolverApp**. Since the algorithm will be different, these parameters do not have much interest for the Proof-of-Concept (PoC).

In Listing E.1 is an example of a **Master Problem** message.

7.1.2 Master Acknowledgement

When the **clientSolvers** receive by broadcast a Master Problem, they must respond to it with a simple Master Acknowledgement with the Master ID from the Master Problem and their private queue to identify themselves.

7.1.3 Master Sealed

After sending the Master Problem, the **trafficManager** sends a Master Sealed message to confirm the MasterID of the Master Problem and how many sub-problems there are in the first message.

7.1.4 Work Request

When the **clientSolvers** receive a Master Sealed message via broadcast, they have to send a work request to the **trafficManager** to receive the second part of the problem, the Solve message in their private queue. The **trafficManager** will select one of the clientSolvers, usually the first to send the work request, to send the subsequent message to its private queue.

7.1.5 Solve

After selecting the **clientSolver**, the **trafficManager** will send a Solve message to the private queue of the **clientSolver**. This message has two identifiers: the Master ID from the Master Problem and the UID that uniquely identifies the Solve message.

Additionally, the message contains the number of constraints and a static penalty, which must be added at the end of the calculus.

The main body of this message is composed of a set of objective functions (Fun), the application's algorithms explained in Section 4.2.2. Inside the objective function is a group of stations (Sta), one or two for each service. Each station is composed of a weight coefficient (WeC) and a group of polynomials (Pols) where each polynomial (Pol) has a constant (Con) and a VarSetID to indicate which Set from the Master Problem the **clientSolver** has to use.

Combining the two messages (Master Problem and Solve), the **clientSolver** can find a solution and send a response message with new information.

In Listing E.2 is an example of a **Solve** message.

7.1.6 Solution

The Solution message contains the Master ID from the Master Problem message, the UID from the Solve message, and the **clientSolver**'s private queue.

The **clientSolver** must also send the value obtained when adding the parts of the linear formula and the static penalty.

Finally, after solving the problem comprising two messages, the **clientSolver** must update the Vars from the Master Problem with new dwell times. These values will replace the default current values, and the **clientSolver** will only change the elements used to solve the problem.

In Listing E.3 is an example of a **Solution** message.

7.1.7 Drop Master

Finally, when the **trafficManager** receives the Solution message from a single **clientSolver** instance, it broadcasts a Drop Master message so the remaining **clientSolvers** can clear the data they had stored about that problem sent earlier. After this message, the components return to the regular exchange of messages to check their status.

7.2 Message Analysis

After discovering all the information about these messages, it was possible to understand that the **clientSolvers** and the **clientSolverApp** do not consider the geography to make their calculations and only update the dwell times to compensate for any disruption from the original timetable. It was also possible to reproduce manually the calculations made by the **clientSolver** to confirm the value sent in the Solution message.

The next chapter will explain the implementation approach. Some topics are the programming language selected and its libraries and modules, the types of Neural Network (NN) that exist and which were implemented and, finally, how the classes and functions created can achieve the functional and non-functional requirements earlier.

Chapter 8

Proof-of-Concept Implementation

This chapter will describe the implementation of the Proof-of-Concept (PoC) where a new **clientSolver** was created to be integrated into the Full Train Control and Management System (FTCMS) in the future. The first section, Section 8.1, will explain topics related to the programming language used, its libraries and their usefulness in helping the development process.

In this PoC, it was necessary to link the new **clientSolver** to the application Message Queue (MQ) via topic subscription and develop a listener to read the messages sent to the topics. For each message received, the PoC must output the corresponding message to replicate the behaviour of the **clientSolver** as shown in Figure 7.1.

When thePoC receives the messages about the problem, it is required to process the information inside and start building the problem that the Neural Network (NN) must solve. After that, the PoC must create the solution and send it to the **trafficManager**. Section 8.2 will explain how the whole process was developed and designed to achieve the goals of the requirements.

Lastly, Section 8.3 was reserved for describing the usage of secondary files that might be found in the main folder of this PoC.

8.1 Language and Framework

After deciding the algorithm to implement in the PoC, it was necessary to understand if one of our non-functional requirements (NFR1 from Table F.9) is suitable for this implementation.

8.1.1 C++ vs Python

While looking for suitable programming languages to implement a machine learning algorithm, specifically a deep learning algorithm like DQN, it was possible to conclude that **C++** (the language initially chosen for the task) was unsuitable. **C++** lacks sufficient libraries and frameworks for deep learning. Although some libraries offer **C++** Application Programming Interfaces (API), they are less developed or robust than their counterparts, which may limit the functionality and flexibility of the DQN developed. Moreover, **C++** is not widely used in deep learning, leading to a lack of accessible resources for learning and troubleshooting the code produced. These two aspects would make the development of this project take more time than allowed, so it was decided to develop it in a programming language that was more suitable and capable for this type of project.

On the other hand, **Python** has a rich ecosystem of libraries and frameworks that are particularly well-suited for implementing deep learning models, such as TensorFlow and PyTorch. These libraries can provide many functionalities for building and training deep learning models, making **Python** an appropriate choice for implementing a DQN. Additionally, **Python** is known for its simplicity and ease of use, making the learning curve less steep than other programming languages, such as **C++**.

8.1.2 TensorFlow vs PyTorch

TensorFlow and PyTorch are the most famous Python libraries for developing deep-learning algorithms. They constantly improve and strive for excellence in deep neural networks, making them well-suited libraries for this implementation.

TensorFlow is a highly regarded open-source platform offering end-to-end machine learning support. Before becoming open-sourced, a group of researchers and engineers from Google Brain was developing it. There are several benefits to using this library, many of which are associated with its being open-source, scalability, and compatibility with other libraries (such as Keras) and other programming languages. Although TensorFlow is a popular library for machine learning, it is slower and less user-friendly than other libraries. In addition, it also requires more expertise in the library since it adds a level of complexity related to its dependency on other libraries and tools, such as CUDA and TensorFlow Serving, which are different components [Boesch, 2023].

PyTorch is a more recent library developed by a Facebook research group. Its initial plan was to use it for probabilistic programming. However, it became a popular tool in deep learning by combining a focus on usability with careful performance considerations. Some advantages rely on its *Pythonic* programming style since the library was developed in native Python, making it more straightforward to learn and debug when the user is familiar with the programming language. Lastly, PyTorch can distribute computational work among multiple CPU or GPU cores. On the other hand, PyTorch is less extensive than TensorFlow because it is not an end-to-end tool, which requires the help of different tools to deploy applications to servers or workstations, making PyTorch less scalable [Boesch, 2023].

After analysing both libraries, it is believed that despite PyTorch's disadvantages, its advantages can lead to a smoother implementation for a user familiar with the programming language but unfamiliar with neural network implementation.

8.2 Proof-of-Concept Classes

This section will present the solutions to successfully enforce the functional and non-functional requirements. Each subsection will explain how a requirement or group of requirements was implemented in the PoC.

8.2.1 Deserialiser and Listener Classes

The first three functional requirements (FR1, FR2 and FR3) are related to the capability of reading and writing messages in the message bus of the Full Train Control and Management System (FTCMS), the Artemis Message Queue (MQ). Along with these functions are those of deserialising and serialising, respectively. A Listener class and a DeSerializer class were implemented to meet these requirements.

The Listener class extends the **ConnectionListener** class from the *Stomp* python library, which provides functions to handle messages written in one or multiple topics or queues. In addition to the built-in functions of the *stomp* class, other functions were built to place the messages received in the First In First Out (FIFO) queues, to retrieve the messages from the FIFOs to be processed properly and to check if the FIFOs is empty or not. To properly use this class, a **Connection** (also from *Stomp* library) was set to the IP and port of the internal simulator of the FTCMS (presented in Section 4.2.1) and subscribed to the broadcast topic of the **trafficManager** and to the queue of the **clientSolverApp**, where all **clientSolvers** write their messages. This class implements the requirements FR1 (Table F.1) and FR2 (Table F.2).

The DeSerializer class has two functions: to deserialise a new message received from a subscribed queue or topic and to serialise a message to publish in a queue or topic from the message bus. These functions were constructed with the help of *xmltodict*, a python library that converts an Extensible Markup Language (XML) string message to a dictionary and contrariwise. This class implements the requirement FR3 (Table F.3).

8.2.2 Problem Class

The problem construction is made in two different steps. The first is when the **trafficManager** broadcasts the Master Problem message, and the **clientSolver** receives the forecast of each trip and the workable sets to solve. The Problem class constructor is built to receive the data from the Master Problem message as a parameter. The class itself has functions to handle each part of the message (Vars, Sets and Params) that create the attributes for each. This part of the class implements the FR4 (Table F.4)

Later, when the Solve message is received, it can be added to the Problem object to finalise the problem's construction. After this addition, the object can create a new Solution instance, passing some necessary information to the Solution constructor. This part of the class implements the FR5 (Table F.5) and FR6 (Table F.6).

8.2.3 Solution Class

The Solution class has multiple methods to implement the functional requirements for finding a solution and all the necessary steps. Each problem is divided into subproblems for each platform received in the Solve message.

In this class, some functions can calculate and preprocess the input data of the Neural Network (NN) model to solve the input using a given NN model, to process the output and prepare it to be converted and sent in the Solution message. All these functions were made with the help of *PyTorch* library.

The preparation of the input data requires some steps. First, it is necessary to create the linear spaces for each GUID that will be used to solve the problem. If a GUID has already been calculated, the linear space will be represented by one element with the value of it. Otherwise, the GUID will be represented by 1500 numbers between its minimum (Min) and maximum (Max) value. After the creation of the linear spaces, it is required to calculate all the possible combinations between those linear spaces. That can be achieved by calculating the Cartesian product, something that *PyTorch* allows us to do. The last step is to add the platform's constant and weight coefficient to every item of the Cartesian product. Finally, in some cases, it is necessary to shape the data according to the input features of the NN model. One example of an element of the input:

$$[x_i, y_i, z_i, c_p, w_p] \tag{8.1}$$

Where x_i, y_i, z_i is *i-th* combination of the Cartesian product, c_p is the constant for platform p and w_p is the weight for platform p as well.

After these steps, the data is ready to go through the model, and the result will be retrieved. After getting the output, it is necessary to check the minimum value of the outcome and get the combination that originated that value. The values of the GUIDs are then updated with the ones in the combination. After solving all subproblems, the data is prepared to be sent in the Solution message after being converted to a XML message. This last part of the class implements the FR8 (Table F.8).

8.2.4 Neural Network Classes

After researching to identify the most appropriate Neural Networks (NN), a subsequent study was conducted to find examples of their implementation. This investigation confirmed that PyTorch could streamline the implementation process, leading to the decision to implement multiple NNs. These models aim to implement the FR7 (Table F.7)

Four models were developed to identify the most suitable NN for solving timetable disruptions by adjusting the dwell time. The purpose of creating these models was to comprehensively evaluate their capabilities by comparing their learning capacity while training with a dataset and verifying the accuracy when faced with a new dataset. By developing multiple models and analysing their outcomes, we aimed to determine their behaviour through the training process and how to adapt their outputs to the correct ones. These classes were designed to inherit the capabilities of a PyTorch *nn.Module* container. The PyTorch container *torch.nn.Module* serves as the base class for all neural network modules in PyTorch. By inheriting this class, the developed classes acquire all the necessary methods and attributes required for creating and managing a neural network [Pytorch, 2024].

The following list aims to explain how the models were built.

Linear Neural Network

The Linear Neural Network (LNN) class comprises four layers: one input, one output and two hidden layers. This Neural Network (NN) has no non-linear activation function since it aims to understand if it can map the relation between the input and the output with a linear approach. The first hidden layer comprises 64 neurons, and the second hidden layer is comprised of 32 neurons.

Feed Forward Neural Network

In a similar approach to the LNN, the Feed Forward Neural Network (FFNN) class is also composed of four layers with the addition of one non-linear activation function. Like the previous NN, 64 and 32 neurons form the hidden layers, respectively. The objective of having two NNs very similar with only one difference is to compare the results between a linear approach and a non-linear one.

Recurrent Neural Network

Considering a different procedure, a Recurrent Neural Network (RNN) was built, suitable for sequential data. The model comprises an input layer, a single RNN layer and an output layer.

Two styles of inputs and outputs were tested for this NN. The first one was explained in Solution Class subsections, where the input represents the combinations of numbers that should lead to a value. Instead of giving the combinations of numbers to get the value, the second type of input was passed to the values of the combinations, and the model must select the correct one. This input alternative was elaborated to give the RNN something more alike to sequential data.

Long Short Term Neural Network

Lastly, Long Short-Term Memory (LSTM) Neural Network was built. Like the RNN, this model comprises an input layer, followed by a LSTM layer, ending with an output layer.

In this case, we want to compare the capabilities of a RNN and a LSTM and check which models can perform better in training and testing.

By building these four models, it was possible to test them against each other and see how they work with different inputs and see which of the inputs is more aligned with the objectives of this project, whether in terms of speed or terms of training well the models so they can have a good performance in test and against the **clientSolver** from **clientSolverApp**.

8.3 Other Files

The last section of the chapter is reserved for other created files that were used to continuously improve the Proof-of-Concept or served as secondary functions necessary to ease one task. The following list aims to explain more about these secondary files that were also crucial to developing a good Proof-of-Concept (PoC).

train_test.py

This file defines the train and test functions to teach and evaluate the models. In each function, it was necessary to simulate the steps required to prepare the input, pass it through the model, and retrieve its output.

On the one hand, in the training function, the output retrieved is compared to the expected output, the loss value is calculated, and the model suffers some weight adjustment to improve its performance.

On the other hand, in the test function, the output is evaluated to verify whether the correct combination was identified. In this case, the right combination is the one that leads to the minimum value according to the mathematical formula that is being used in the FTCMS.

logger.py

In this file, a Logger class was developed to retrieve some information during the execution. This class was implemented with the help of the Python library *logging*. The capacity of verbosity was also implemented, where the logger writes the information in the terminal and a file for later consultation.

message_generator.py

This file contains functions for preparing data structures to parse them into XML messages. Specific functions were created for each type of message to build the header and the data dictionaries with relevant information tailored to that type of message.

save_load.py

This file contains the function to save and load the trained and tested models and can be used in execution scenarios against the **clientSolver** from **clientSolverApp**.

write_graph.py

Lastly, this file includes the functions to write the models' loss and accuracy results into files and plot them in graphs so they can be analysed easily.

Chapter 9

Testing

This chapter aims to present all the tests and comparisons made between the four Neural Networks (NN) developed in the Proof-of-Concept (PoC) and the **clientSolver** from FTCMS's **clientSolverApp**. It will explain which NN was selected and why. Initially, it will compare the two input styles experimented in the NNs and which one was chosen to proceed with the NN's evaluation in Section 9.1.

Secondly, the results obtained from the two recurrent NN models will be shown, and what analysis can be made from that (Section 9.2). Then, the results gathered from the Feed Forward Neural Network (FFNN) and the Linear Neural Network (LNN) will be presented and compared in (Section 9.3).

Finally, the best model is selected and compared against the **clientSolver** from the **clientSolverApp** and, finally, the objectives and requirements are compared to the results acquired to understand if the goals were met (Section 9.4).

The models were analysed using two metrics: the loss value and the accuracy. The loss value denotes the difference between the outputs, the one expected to be outputted by the NN and the one outputted by the NN. There are multiple functions to calculate the loss. The Mean Absolute Error (MAE) and the Mean Squared Error (MSE) were chosen among all options. The MAE loss function calculates the absolute difference between the predicted and actual values rather than squaring the differences as in MSE. This non-squaring approach also makes the MAE more robust to outliers and their impact. The simplicity of MAE also makes its value easier to analyse and interpret. On the other hand, in the case of a model's wrong output, the MAE cannot penalise them more heavily than minor errors, something that the MSE can do very well. While testing, it was noticed that the MAE produced better results than the MSE, which is why it was abandoned.

The accuracy is represented by a percentage defined by the number of cases where the model found the correct answer over the total number of cases as Equation (9.1) shows. In Chapter G, it is possible to see the graphics produced with the results acquired from all the tests made.

$$Accuracy = \frac{Cases\ with\ Correct\ Answer}{Total\ Cases} \tag{9.1}$$

The model's training and testing used a dataset with almost 2 300 problems, each with ten trains running. This combination of several problems and the number of trains per problem gives nearly 23 000 cases for the models to train and test. After some experiments, it was found an optimal division 85%/15% for the training and testing, meaning that 85% of the dataset would be used for training and the remaining 15% would saved for performing the tests and accuracy evaluation. In addition, a dataset with about 225 problems with 21 trains running was also used, which gives practically 5 000 cases for training and testing.

Both datasets were crucial to better understanding the models' behaviour and the time they take to solve the problems of different sizes. In both datasets, it was experimented to pass the training sequence through the model several times. This experiment revealed that, in some cases, the models started to overfit the training sequence, and then in the testing, the model would perform poorly, ending up with low accuracy. It was verified that the training sequence should not pass more than three times. In Table 9.1 is a summary of the tests made to each model.

	Testing Datasets	
	10 Trains Dataset	**21 Trains Dataset**
Number of Problems	2 300	225
Total Cases	23 000	5 000
Train/Test Percentage	85%/15%	
Sequence Epochs	[1, 2, 3]	
Train Batch Size	[10, 15, 20]	
Number of executions	3	5
Total tests	27	45

Table 9.1: Summary of tests made to each model

9.1 Input Comparison

As briefly explained in Section 8.2, two styles of inputs were experimented with due to the strengths of the recurrent NNs of handling sequential data. The first input represents the combination of the service forecast and a platform's weight and constant, as exemplified in Equation (8.1). As output, the NN emits the combination's value, the same that the **clientSolver** calculates.

The second input is all the values that the **clientSolver** computes, and the NN must select their minimum value. This selection is the next step the **clientSolver** performs after calculating the possible values.

The tests revealed that the NN was extremely slow when processing data shaped in the second input style. It was estimated that only a single run of training and testing a NN would take more than 15 hours. Because of that, it was decided to abandon this input style and keep the remaining tests and evaluations with only combination input.

9.2 Recurrent Models

It was time to experiment with the four models after selecting the input for the NNs. This section presents the results acquired for the Recurrent Neural Network (RNN) and the Long Short-Term Memory (LSTM).

These results confirmed what was expected: the recurrent models have strengths in dealing with sequential data rather than isolated numbers. From countless tests, both recurrent models could not achieve the minimum required to avoid rejection. The outcomes presented in Chapter G of the appendix show that both models could not reach an accuracy higher than 50% and could not stabilise the loss below average. However, the LSTM proved to be better than RNN by achieving a higher accuracy than RNN. Both could not continuously reduce the loss value and get closer to the expected output. In Tables 9.2 and 9.3, is possible to verify a resume of the accuracy obtained for the RNN and LSTM models using the two datasets previously presented. At the same time, in Sections G.1.1 and G.1.2 of the appendix, it is possible to glimpse all the loss and accuracy results obtained by the RNN and the LSTM.

With these results, it was clear that the strengths of LSTM and RNN could not be fully maximised because of the type of input chosen and the time consumed by a more suitable input. The models were discarded from the possibility of being selected as the best model to compete against the **clientSolver**.

	RNN		LSTM	
Number Epochs x Batch Size	**Mean**	**STDEV**	**Mean**	**STDEV**
1 *x* 10	23%	7%	30%	12%
1 *x* 15	26%	7%	33%	7%
1 *x* 20	24%	6%	30%	9%
2 *x* 10	14%	12%	36%	9%
2 *x* 15	24%	5%	37%	7%
2 *x* 20	18%	5%	39%	5%
3 *x* 10	20%	4%	28%	11%
3 *x* 15	20%	4%	41%	6%
3 *x* 20	16%	4%	36%	13%

Table 9.2: Accuracy results acquired for RNN and LSTM using 10 Trains Dataset

	RNN		LSTM	
Number Epochs x Batch Size	**Mean**	**STDEV**	**Mean**	**STDEV**
1 *x* 10	4%	5%	23%	10%
1 *x* 15	4%	3%	19%	9%
1 *x* 20	2%	2%	7%	9%
2 *x* 10	9%	12%	18%	12%
2 *x* 15	1%	1%	20%	10%
2 *x* 20	1%	1%	21%	8%
3 *x* 10	7%	10%	26%	9%
3 *x* 15	1%	1%	24%	9%
3 *x* 20	1%	0%	24%	10%

Table 9.3: Accuracy results acquired for RNN and LSTM using 21 Trains Dataset

9.3 Linear versus Non-Linear

While the recurrent models were being tested, FFNN and LNN were also subjected to testing. This section, together with the Chapter G, aims to show and analyse the results of all the tests and evaluations.

From tests performed to LNN models, it was possible to verify that the model achieved the minimum required to avoid rejection with an accuracy higher than 60%. However, it was also likely to ascertain that, after overcoming the rejection threshold, the model could not improve its performance in the tests or reduce the value of the loss function.

In contrast, the FFNN exceeded the expectations and could achieve outstanding performance in testing and reducing many loss values. The model was skilled in reducing the loss value to approximately 75% of its value. At the same time, the model could perform in tests consistently with an accuracy of 90% except for one outlier, where the model obtained an accuracy of only 70%. In Tables 9.4 and 9.5, it is possible to see an overview of the accuracy results for the FFNN and LNN using the two datasets. Similarly to the other models, in Sections G.1.3 and G.1.4 of the appendix, it is possible to see the loss and accuracy results acquired for the LNN and FFNN models.

After analysing the results from LNN and FFNN, it was possible to conclude that the best model from all four implemented is the Feed Forward Neural Network since it was the only model capable of achieving a high accuracy consistently in tests and consistently reducing the loss value while training.

	LNN		**FFNN**	
Number Epochs x Batch Size	**Mean**	**STDEV**	**Mean**	**STDEV**
1 *x* 10	62%	3%	79%	12%
1 *x* 15	67%	1%	79%	14%
1 *x* 20	65%	2%	77%	12%
2 *x* 10	66%	1%	87%	8%
2 *x* 15	66%	1%	87%	5%
2 *x* 20	65%	1%	70%	7%
3 *x* 10	65%	2%	77%	13%
3 *x* 15	66%	1%	81%	11%
3 *x* 20	65%	3%	73%	7%

Table 9.4: Accuracy results acquired for LNN and FFNN using 10 Trains Dataset

	LNN		FFNN	
Number Epochs x Batch Size	**Mean**	**STDEV**	**Mean**	**STDEV**
1 *x* 10	61%	7%	66%	11%
1 *x* 15	61%	6%	67%	12%
1 *x* 20	59%	10%	63%	11%
2 *x* 10	61%	8%	65%	14%
2 *x* 15	62%	7%	72%	11%
2 *x* 20	62%	4%	76%	14%
3 *x* 10	59%	8%	71%	18%
3 *x* 15	62%	3%	72%	14%
3 *x* 20	63%	3%	72%	16%

Table 9.5: Accuracy results acquired for LNN and FFNN using 21 Trains Dataset

9.4 Model versus ClientSolver

After selecting the best model, it was necessary to compare it against the **clientSolver**, the component that this PoC is trying to replicate. Two aspects were analysed to adequately make that comparison: time performance and the final value sent in the solution, where all the values obtained from the problem's calculations are summed up.

The time performance was very tricky to analyse. This difficulty was due to the cluster of points created on the graph you want to analyse. Because of all these overlaps, it was feasible to conclude that the time performance neither improved nor worsened. The plots show that the record of solving problems in a few milliseconds has been maintained and that the differences found can be considered inconsequential. These values can be quickly confirmed in Table 9.6 where the minimum, the maximum, the mean and the STDEV time are presented for the problems from the two datasets used. In Section G.2.1 of the appendix, the plots show the time performance results of the model and the **clientSolver** and their difference.

	FFNN Model		**FTCMS clientSolver**	
	10 T. Dataset	**21 T. Dataset**	**10 T. Dataset**	**21 T. Dataset**
Minimum (sec.)	0.03	0.09	0.00	0.00
Maximum (sec.)	1.21	1.15	2.96	0.71
Mean (sec.)	0.15	0.24	0.07	0.10
STDEV (sec.)	0.12	0.14	0.09	0.10

Table 9.6: Time comparison between best model (FFNN) and **clientSolver**

On the other hand, the values obtained at the end of the calculations were slightly different than the ones obtained from the **clientSolver**. However, the combinations were correctly found, and the stop times were adequately adjusted. The non-zero loss function values at the end of the training also explain this difference between the model and the **clientSolver**. In Table 9.7, the differences between the values given by the FFNN model and the **clientSolver** can be seen. In Section G.2.2 of the appendix, it is the plots that show the difference between the values given by the model and the **clientSolver** when solving the problems.

	Values Difference	
	10 T. Dataset	**21 T. Dataset**
Minimum (sec.)	−11 674.17	−5 305.88
Maximum (sec.)	224 224.56	451 370.62
Mean (sec.)	26 397.20	87 496.46
STDEV (sec.)	34 675.95	65 038.77

Table 9.7: Solution Value difference between best model (FFNN) and **clientSolver**

Lastly, it is necessary to bring back the non-functional requirements and verify if they were all met. The functional requirements were already analysed and paired with the PoC's classes in section 8.2. Next is a brief conclusion about each NFR:

- **NFR1** - C++ as Programming Language (Table F.9)

  It was not met due to various reasons explained and presented in Section 8.1, where it was concluded that **Python** would be a more suitable programming language for this implementation.

- **NFR2** - Avoid delay propagation (Table F.10)

  Getting a model that achieved a 90% accuracy rate implies that the problems are solved correctly in most cases, which authorises to affirm that the NFR2 goal was achieved.

- **NFR3** - Minimise the impact of new traffic (Table F.11)

  Unfortunately, due to the simulator's limitations, verifying if the model could handle new traffic was impossible when it was added.

- **NFR4** - Improve the speed performance (Table F.12)

  Although the speed performance did not improve, it was possible to keep identical records even with a different programming language. Considering this, it is possible to state that the minimum requirements of NFR4 were achieved.

- **NFR5** - Results comparability (Table F.13)

  The PoC's results were correctly defined to be comparable to those from the **clientSolver**, which met the goals of NFR5.

- **NFR6** - Model's Accuracy (Table F.14)

  Lastly, it was possible to conceive multiple models that overcame the minimum accuracy defined for not being rejected, which made it possible to achieve the objectives of NFR6.

Chapter 10

Future Work

After developing the Proof-of-Concept (PoC), opening up horizons for what's next is necessary since there was no time to continue evolving the PoC due to the changes made during the second semester. Considering the theme of this project, the current state of the PoC and everything that happened during its timeframe, it was discussed with CSW's mentor two possibilities for the next steps. This chapter aims to unveil some of the considerations made about these possibilities. Finally, it will present some insights into what can be done with the already developed models.

10.1 Creating Timetables

At Critical Software (CSW), the possibility of creating new and improved timetables in FTCMS by using Artificial Intelligence (AI) methods was debated. Those methods are related to using optimisation algorithms, like the ones studied in Section 5.2 or to Machine Learning (ML) models, like a Neural Network (NN) or a Support Vector Machines (SVM).

On the one hand, optimisation algorithms can be used to improve the timetables, considering the geography and some path constraints (for example, train speed, train type, junctions priority management and closed or blocked parts of the rail). The optimisation objective may differ, considering the stakeholders' desires in developing a project like this. Although the overall goal is the same, optimising the timetable can be achieved by focusing on a specific matter (or objective function) like passenger waiting time, travel time, or energy consumption while respecting the geography's constraints. One example of implementation is the combination of a Mixed-Integer Linear Programmimg (MILP) with a heuristic technique that uses Genetic Algorithms since the creation of a new timetable is considered an NP-hard problem that can be solved by the MILP in more straightforward cases and by the Genetic algorithm in more complex cases [Garrisi and Cervelló-Pastor, 2019].

On the other hand, a ML technique can be trained to learn the relationship between services running or some of the patterns linked to delay problems and how they evolve. This type of implementation could help the models to optimise the timetables by adjusting them accordingly. It was found in an academic paper a suggestion of an innovative approach to optimise scheduling in educational institutions by using AI and ML [Nagtilak et al., 2023].

10.2 Informed Estimations

The second approach to a possible future step for the theme and project is to consider making estimations with more information than simply a platform's constant and weight coefficient. This style of approach opens a complete set of possibilities.

One of the ideas debated with CSW's mentor was the possibility of counting passengers to understand its relationship with the project's main focus: solving delay problems by adjusting the dwell time. By doing some research, it was possible to find a paper about a study to investigate the relationship between passenger volume and frequency and the amount of delay in the dwell time in Stockholm. For this paper, the authors used a dataset created by an automatic system that counts the passengers inside a train. The study concludes that the relationship is non-linear and depends on the schedule [Kuipers and Palmqvist, 2022].

In addition, an example of a counting passenger system is the one produced by Luminator Technology Group. This system can count the number of passengers that enter and leave automatically and present some metrics to provide more information to the passengers [Luminator, 2024]. This system can provide substantial insights into the passengers' affluence patterns (grouping by services or platforms) to help the **clientSolver** make better estimations while adjusting the dwell time.

10.3 Parameter exploration

The last natural approach is the exploration of parameters inside the PoC. As is well known, constructing a neural network depends on multiple variables. Understanding that parameter customisation is possible, one way to move forward and continue developing the PoC is by trial and error to discover if a better parameter combination leads to better results. It is possible to try other loss functions, other values for the learning rate, different activation functions, adding more layers to the NN, and increasing or reducing the number of input and output features.

Chapter 11

Conclusion

The work performed during the first semester of this internship was primarily exploratory. This type of work, related to research and investigation, is naturally necessary to understand the concepts when entering a new market like the railway. During this process, it was also necessary to analyse the project's risks and start elaborating on the objectives and requirements.

After understanding distinct system definitions, it was possible to correlate them and know how they can work together. Learning about the railway companies and their placement in the market was also crucial. While going through this process, it was necessary to comprehend the FTCMS's components, messages and algorithms in use.

The final task of the first semester was the construction of the State of the Art (SoA), where it was necessary to do research in the market for existing products and in the literature for optimisation algorithms that would look suitable for our network optimisation problem. The author faced the first challenge during the market research because the content pursued was core business and, consequently, private. A more generalist comparison of the products and the FTCMS was made to bypass this privacy issue. In addition to the analysis, ten algorithms were analysed and ranked based on their performance in similar matters and limitations compared to the other algorithms.

Decoding the messages was much more challenging and complex than expected. The author had to spend much more time and effort than initially thought. He had to analyse the **trafficManager** code and the **clientSolver** code comprehensively to understand the origin of the content of the various message's attributes. This decoding delay occurred because it was confirmed that no formal documentation existed about it, and the colleagues who developed it no longer worked for CSW. After decoding them, the author had to activate a mitigation plan and adjust the development plan, add more information to the SoA, and adequately document the information discovered. In addition to the fact that the messages were a crucial aspect for developing the PoC, this documentation was something the CSW has been looking for for a while, and this project could contribute to that.

The development of the PoC involved selecting the programming language and the libraries to do it, looking for examples of NNs, implementing the NNs and developing other classes to make it possible to communicate with other components, process the data received and reply with new information.

For the tests, two different inputs were experimented but one was revealed to be too slow and was discarded. The tests performed to the multiple NNs highlighted that the recurrent models could achieve the objectives defined in the beginning. The models were not capable of reaching the minimum accuracy and were rejected. The other two models (LNN and FFNN) were both capable of achieving, and the FFNN delivered better results than the LNN. After finding the best model, this one was compared against the **clientSolver** from **clientSolverApp**. This phase revealed that the PoC's model was equally fast and could send identical answers to the **clientSolver**.

Regarding the objectives defined at the beginning of the report, it was impossible to verify if the PoC could handle the desired number of trains due to external limitations. It was possible to match the speed of the **clientSolver** and nearly equalise the answers given by the **clientSolver**. Lastly, creating a model capable of learning was possible since it could continuously reduce the loss value and improve its accuracy.

This project took the first steps in implementing Artificial Intelligence (AI) to help manage railway schedules by adjusting the dwell times of trains. Unfortunately, it was impossible to continue developing the PoC, but an analysis of possible ways to continue the work done so far was conducted to ensure its continued growth.

References

Hanan Abdullah Alshehri. *Deep Learning for Electricity Forecasting Using Time Series Data*. PhD thesis, Science and Mathematics of Montclair State University, 2021. URL https://core.ac.uk/download/386155399.pdf. [Accessed 30-04-2024].

Alstom. Alstom company. alstom.com, a. URL https://www.alstom.com/company. [Accessed 23-10-2023].

Alstom. Alstom CBTC range: World leading high-capacity signalling. alstom.com, b. URL https://www.alstom.com/solutions/signalling/alstom-cbtc-range-world-leading-high-capacity-signalling. [Accessed 30-10-2023].

ARC-Advisory. When will we see etcs level 3 train control systems. arcweb.com. URL https://www.arcweb.com/industry-best-practices/when-will-we-see-etcs-level-3-train-control-systems. [Accessed 19-12-2023].

Markel Sanz Ausin. Introduction to reinforcement learning. part 3: Q-learning with neural networks, algorithm dqn. *medium.com*, 2020. URL https://shorturl.at/admM9. [Accessed 23-10-2023].

Autoblocks. What is neuro-fuzzy? autoblocks.ai. URL https://www.autoblocks.ai/glossary/neuro-fuzzy. [Accessed 25-10-2023].

Keith Barrow. Automatic train operation takes to the main line. *Internation Railway Journal*, 2018. URL https://www.railjournal.com/in_depth/automatic-train-control-takes-to-the-main-line/. [Accessed 19-12-2023].

Gaudenz Boesch. Pytorch vs tensorflow: A head-to-head comparison. viso.ai, 2023. URL https://viso.ai/deep-learning/pytorch-vs-tensorflow/. [Accessed 20-04-2024].

Carter, Stockmeyer, and Wegman. The complexity of backtrack searches. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, page 449–457. Association for Computing Machinery, 1985. ISBN 0897911512. doi: 10.1145/22145.22195. URL https://doi.org/10.1145/22145.22195. [Accessed 20-02-2024].

Maria Catrina. Railway station routing algorithm using the backtracking method. www.scientificbulletin.upb.ro, 2015. URL https://www.scientificbulletin.upb.ro/rev_docs_arhiva/fulla34_189839.pdf. [Accessed 23-10-2023].

Codility. Greedy algorithms. codility.com, 2021. URL https://codility.com/media/train/14-GreedyAlgorithms.pdf. [Accessed 20-02-2024].

Francesco Corman, D'Ariano, Andrea, Dario Pacciarelli, and Marco Pranzo. A tabu search algorithm for rerouting trains during rail operations. *Transportation Research Part B: Methodological*, 2010a. doi: 10.1016/j.trb.2009.05.004. [Accessed 06-12-2023].

Francesco Corman, Andrea D'Ariano, Dario Pacciarelli, and Marco Pranzo. A tabu search algorithm for rerouting trains during rail operations. *Transportation Research Part B: Methodological*, 2010b. doi: https://doi.org/10.1016/j.trb.2009.05.004. URL https://www.sciencedirect.com/science/article/pii/S0191261509000708. [Accessed 11-12-2023].

Coursera. What is an lstm neural network? *https://www.coursera.org/*, 2024. URL https://www.coursera.org/articles/lstm-neural-network. [Accessed 29-04-2024].

Chao De-Yu. Deep q-network, with pytorch. *Towards Data Science*, 2021. URL https://towardsdatascience.com/deep-q-network-with-pytorch-146bfa939dfe. [Accessed 23-10-2023].

Peter Dolgopolov, Denis Konstantinov, Liliya Rybalchenko, and Ruslans Muhitovs. Optimization of train routes based on neuro-fuzzy modelling and genetic algorithms. *Procedia Computer Science*, 2019. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2019.01.101. URL https://www.sciencedirect.com/science/article/pii/S1877050919301073. [Accessed 25-10-2023].

Yunshu Du. Introduction to deep q-network. Washington State University, 2016. URL https://www.railway-technology.com/contractors/signal/delta-rail/. [Accessed 20-02-2024].

Wenxing Wu et al. An integrated method for reducing arrival interval by optimizing train operation and route setting. *MDPI*, 2023. URL https://www.mdpi.com/2227-7390/11/20/4287. [Accessed 25-10-2023].

Roger Ford. Automatic route setting key to successful tm. *Modern Railways*, 2020. URL https://www.modernrailways.com/article/automatic-route-setting-key-successful-tm. [Accessed 31-10-2023].

Olamide Fred. Deep learning for beginners: A comprehensive guide. *https://nicholasidoko.com/blog*, 2023. URL https://nicholasidoko.com/blog/2023/03/09/deep-learning-for-beginners-a-comprehensive-guide/. [Accessed 28-04-2024].

Gianmarco Garrisi and Cristina Cervelló-Pastor. Train-scheduling optimization model for railway networks with multiplatform stations. *Sustainability*, 2019. URL https://api.semanticscholar.org/CorpusID:214198722. [Accessed 18-06-2024].

GeeksforGeeks. Backtracking meaning in dsa. geeksforgeeks.org, a. URL https://www.geeksforgeeks.org/backtracking-meaning-in-dsa/. [Accessed 06-12-2023].

GeeksforGeeks. Introduction to backtracking – data structure and algorithm tutorials. geeksforgeeks.org, b. URL https://www.geeksforgeeks.org/introduction-to-backtracking-data-structure-and-algorithm-tutorials/. [Accessed 22-12-2023].

GeeksforGeeks. Introduction to ant colony optimization. geeksforgeeks.org, c. URL https://www.geeksforgeeks.org/introduction-to-ant-colony-optimization/. [Accessed 30-11-2023].

GeeksforGeeks. Genetic Algorithms. geeksforgeeks.org, d. URL https://www.geeksforgeeks.org/genetic-algorithms/. [Accessed 25-10-2023].

GeeksforGeeks. Greedy Algorithms. geeksforgeeks.org, e. URL https://www.geeksforgeeks.org/greedy-algorithms/. [Accessed 25-10-2023].

GeeksforGeeks. Mathematical Algorithms. geeksforgeeks.org, f. URL https://www.geeksforgeeks.org/mathematical-algorithms. [Accessed 25-10-2023].

GeeksforGeeks. What is TABU Search? geeksforgeeks.org, g. URL https://www.geeksforgeeks.org/what-is-tabu-search/. [Accessed 23-10-2023].

Intaek Gong, Sukmun Oh, and Yunhong Min. Train scheduling with deep q-network: A feasibility test. *Applied Sciences*, 2020. ISSN 2076-3417. doi: 10.3390/app10238367. URL https://www.mdpi.com/2076-3417/10/23/8367. [Accessed 10-12-2023].

Kechit Goyal. Fuzzy logic in artificial intelligence: Architecture, applications, advantages & disadvantages, 2022. URL https://www.upgrad.com/blog/fuzzy-login-in-artificial-intelligence/. [Accessed 06-12-2023].

D Gubler. Automatic train control. railway-technical.com, 2023. URL http://www.railway-technical.com/signalling/automatic-train-control.html. [Accessed 05-11-2023].

A. Shimura & N. Sato H. Teshima, S. Hori. Railway track layout modelling and its application to an automatic route setting system. witpress.com, 2014. URL https://www.witpress.com/Secure/elibrary/papers/CR14/CR14006FU1.pdf. [Accessed 06-11-2023].

Hitachi. About hitachi rail. wikipedia.org, a. URL https://www.hitachirail.com/our-company/about-hitachi-rail/. [Accessed 23-10-2023].

Hitachi. Traffic Management System (TMS): Social Infrastructure Information Systems. hitachi.com, b. URL https://www.hitachi.com/products/it/society/product_solution/mobility/passenger_information/Traffic_Management_System.html. [Accessed 23-10-2023].

Mordor Intelligence. Automatic train control market size & share analysis - growth trends & forecasts (2023 - 2028). 2022. URL https://www.mordorintelligence.com/industry-reports/automatic-train-control-market. [Accessed 23-10-2023].

Törnquist Johanna. Greedy algorithm for railway traffic rescheduling during disturbances: A swedish case. *Intelligent Transport Systems, IET*, 2011. doi: 10.1049/iet-its.2009.0122.

Kyung-Min Kim, Hag-Lae Rho, Bum-Hwan Park, and Yun-Hong Min. Deep q-network approach for train timetable rescheduling based on alternative graph. *Applied Sciences*, 2023. doi: 10.3390/app13179547. URL https://www.mdpi.com/2076-3417/13/17/9547. [Accessed 11-12-2023].

Ruben Alaric Kuipers and Carl-William Palmqvist. Passenger volumes and dwell times for commuter trains: A case study using automatic passenger count data in stockholm. *Applied Sciences*, 2022. ISSN 2076-3417. doi: 10.3390/app12125983. URL https://www.mdpi.com/2076-3417/12/12/5983. [Accessed 19-06-2024].

Lark. Neuro fuzzy. larksuite.com, 2023. URL https://www.larksuite.com/en_us/topics/ai-glossary/neuro-fuzzy#pros-&-cons-of-neuro-fuzzy. [Accessed 20-02-2024].

Munan Li. Efficiency improvement of ant colony optimization in solving the moderate ltsp. *Journal of Systems Engineering and Electronics*, 2015. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7375763. [Accessed 20-02-2024].

Frank Liang. Optimization techniques — tabu search. *Towards Data Science*, 2020. URL https://towardsdatascience.com/optimization-techniques-tabu-search-36f197ef8e25. [Accessed 23-10-2023].

Shengbin Liang. An improved ant colony optimization algorithm based on context for tourism route planning. *Pub Med Central*, 2021. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8445481/. [Accessed 25-10-2023].

Andrei Lissovoi and Pietro Oliveto. Computational complexity analysis of genetic programming. Department of Computer Science University of Sheffield, 2019. URL https://arxiv.org/pdf/1811.04465.pdf. [Accessed 20-02-2024].

Luminator. Track your most important metrics, 2024. URL https://luminator.com/en-uk/products/passenger-counting.html. [Accessed 19-06-2024].

István Lövétei, Bálint Kővári, Tamás Bécsi, and Szilárd Aradi. Environment representations of railway infrastructure for reinforcement learning-based traffic control. *Applied Sciences*, 2022. ISSN 2076-3417. doi: 10.3390/app12094465. URL https://www.mdpi.com/2076-3417/12/9/4465. [Accessed 23-10-2023].

Laura Martinez & Ullrich Martin. Terminology, differences, and challenges of communications-based train control and european train control systems. witpress.com, 2020. URL https://www.witpress.com/Secure/elibrary/papers/CR20/CR20002FU1.pdf. [Accessed 08-11-2023].

MATLAB. Mixed-Integer Linear Programming (MILP) Algorithms. mathworks.com. URL https://www.mathworks.com/help/optim/ug/mixed-integer-linear-programming-algorithms.html. [Accessed 23-10-2023].

Suraj Nagtilak, Nivrutee Dongare, Anand Salave, Pranav Dongare, Ganesh Yeole, Prajakta A. Satarkar, and Students. Smart timetable system using machine learning and artificial intelligence. *International Journal of Advanced Research in Science, Communication and Technology*, 2023. URL https://api.semanticscholar.org/CorpusID:259761568. [Accessed 18-06-2024].

Giovanni Neglia, Sara Alouf, Abdulhalim Dandoush, Sebastien Simoens, Pierre Dersin, Alina Tuholukova, Jerome Billion, and Pascal Derouet. In *Performance Evaluation of Train Moving-Block Control*, 2016. ISBN 978-3-319-43424-7. doi: 10.1007/978-3-319-43425-4_23. URL https://www.researchgate.net/publication/305799816_Performance_Evaluation_of_Train_Moving-Block_Control. [[Accessed 18-12-2023]].

NetworkRail. A £1.86bn investment in digital technology, gsm-r paves the way for more efficient signalling systems. networkrail.co.uk. URL https://www.networkrail.co.uk/running-the-railway/gsm-r-communicating-on-the-railway/. [Accessed 18-12-2023].

Paola Pellegrini, Grégory Marliere, Raffaele Pesenti, and Joaquin Rodriguez. RECIFE-MILP: An effective MILP-based heuristic for the real-time railway traffic management problem. *IEEE Transactions on Intelligent Transportation Systems*, 2015. doi: 10.1109/TITS.2015.2414294. URL https://hal.science/hal-01471388v2/file/doc00021712.pdf. [Accessed 06-12-2023].

Paola Pellegrini, Raffaele Pesenti, and Joaquin Rodriguez. Efficient train rerouting and rescheduling: Valid inequalities and reformulation of recife-milp. *Transportation Research Part B: Methodological*, 2019. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2018.12.008. URL https://www.sciencedirect.com/science/article/pii/S0191261518307276. [Accessed 23-10-2023].

Pytorch. Pytorch torch.nn module. https://pytorch.org/docs, 2024. URL https://pytorch.org/docs/stable/nn.html. [Accessed 10-05-2024].

RailSystem. Railway Signalling Terms. railsystem.net. URL https://railsystem.net/railway-signalling-terms/. [Accessed 08-11-2023].

Railway-Technology. Resonate - smart digital transport solutions. railway-technology.com. URL https://www.railway-technology.com/contractors/signal/delta-rail/. [Accessed 31-10-2023].

Research and Markets. Alstom, bombardier, and siemens lead the global train control and management systems market to 2032. *PR Newswire*, 2023. URL https://shorturl.at/wIKTU. [Accessed 23-10-2023].

Resonate. Traffic management solutions on our luminate digital platform. a. URL https://www.resonate.tech/ui/content/content.aspx?ID=88. [Accessed 31-10-2023].

Resonate. Our journey. resonate.tech, b. URL https://www.resonate.tech/ui/content/content.aspx?ID=85. [Accessed 31-10-2023].

Critical Software SA. Csw about us, 2024. URL https://criticalsoftware.com/pt/our-world/cultura-historia. [Accessed 21-12-2023].

Sudip Sahana. Ant colony optimization for train scheduling: An analysis. *I.J. Intelligent Systems and Applications, ISSN-2074-904X(print),2074-9058(online)*, 2014. doi: 10.5815/ijisa.2014.02.04. [Accessed 06-12-2023].

Richard Schneider. Combining capacity with track-friendly technology: Flexx tronic wako and ars from bombardier. *Global Railway Review*, 2010. URL https://shorturl.at/knpJK. [Accessed 30-10-2023].

Siemens. Rail. mobility.siemens.com, a. URL https://www.mobility.siemens.com/global/en/portfolio/rail.html. [Accessed 23-10-2023].

Siemens. About us. mobility.siemens.com, b. URL https://www.mobility.siemens.com/global/en/company/about-us.html. [Accessed 23-10-2023].

Siemens. Communications-Based Train Control System. mobility.siemens.com, c. URL https://www.mobility.siemens.com/global/en/portfolio/rail/automation/mass-transit/communications-based-train-control-system.html. [Accessed 23-10-2023].

Simplilearn. What is Greedy Algorithm: Example, Applications and More. simplilearn.com. URL https://www.simplilearn.com/tutorials/data-structure-tutorial/greedy-algorithm. [Accessed 25-10-2023].

Supawat Tantrasuwan. Design manual sa automatic train regulation. Technical report, Alstom, 2018. [Accessed 10-12-2023].

Thales. Main line rail. thalesgroup.com, a. URL https://www.thalesgroup.com/en/markets/transport/main-line-rail. [Accessed 23-10-2023].

Thales. Transport security system (railways). thalesgroup.com, b. URL https://www.thalesgroup.com/en/transport-security-systems-railways. [Accessed 23-10-2023].

Thales. European train control system (etcs). thalesgroup.com, c. URL https://www.thalesgroup.com/en/markets/transport/signalling/signalling-solutions-main-line-rail/european-train-control-system-etcs. [Accessed 19-12-2023].

Thales-Group. Traffic Management Systems. thalesgroup.com. URL https://www.thalesgroup.com/en/europe/germany/transportation/traffic-management-systems. [Accessed 23-10-2023].

María Tormos, A. Lova, F. Barber, Laura Ingolotti, M. Abril, and Miguel Salido. *A Genetic Algorithm for Railway Scheduling Problems*. 2008. ISBN 978-3-540-78984-0. doi: 10.1007/978-3-540-78985-7_10.

Turing. Understanding feed forward neural networks with maths and statistics. *https://www.turing.com/kb*, 2024. URL https://www.turing.com/kb/mathematical-formulation-of-feed-forward-neural-network. [Accessed 02-05-2024].

UNIFE. About us. unife.org. URL https://www.unife.org/about-us/. [Accessed 06-11-2023].

M. Vijini. Using genetic algorithms to schedule timetables. *Towards Data Science*, 2020. URL https://towardsdatascience.com/using-genetic-algorithms-to-schedule-timetables-27f132c9e280. [Accessed 25-10-2023].

Stefan Wegele, Roman Slovák, and Eckehard Schnieder. Automated train operation planning using genetic algorithms. *IFAC Proceedings Volumes*, 2003. doi: https://doi.org/10.1016/S1474-6670(17)32440-0. URL https://www.sciencedirect.com/science/article/pii/S1474667017324400. [Accessed 06-12-2023].

Krystian Wiatr. Instalation and configuration manual. Technical report, Alstom, 2023. [Accessed 27-11-2023].

Wiki. Hitachi. wikipedia.org, a. URL https://en.wikipedia.org/wiki/Hitachi. [Accessed 23-10-2023].

Wiki. Alstom. wikipedia.org, b. URL https://en.wikipedia.org/wiki/Alstom. [Accessed 23-10-2023].

Lu Yang. Software architecture specification. Technical report, Alstom, 2023. [Accessed 27-11-2023].

Yilling You. Mathematical models and control algorithms for traffic automation. *UC Berkeley Electronic Theses and Dissertations*, 2022. URL https://escholarship.org/uc/item/35n5f8gw. [Accessed 06-12-2023].

András Élesa, Heriberto Cabezasb, and István Heckla. Heuristic algorithm utilizing mixed-integer linear programming to schedule mobile workforce. *Transportation Research Part B: Methodological*, 2018. ISSN 2283-9216. doi: https://doi.org/10.3303/CET1870150. URL https://www.aidic.it/cet/18/70/150.pdf. [Accessed 23-10-2023].

Appendices

Appendix A

Railway Concepts

The first chapter of the appendix aims to present another secondary railway concept that was also studied while producing the Background Knowledge (Chapter 4). Next is a list of relevant concepts to understand the railway world.

Track Circuit

A track circuit is an electrical system that detects the absence of a train on a section of trail. This information then helps the signalling system know if it is safe for another train to proceed [RailSystem].

Axle Counter

A system used in railway signalling to detect the status of a section of track between two points (cleared or occupied). The system generally consists of a wheel sensor (one for each end of the section) and an evaluation unit for counting the axles of the train both in and out of the section[RailSystem]. They often replace the Track Circuit.

Fixed Block Operation

The railway tracks are divided into small block sections and are often protected by sideline signals. In this operation, Track Circuits or Axle Counters are mounted on the tracks, indicating whether a block is free. Only one train is allowed to move in each block section[Neglia et al., 2016].

Moving Block operation

This operation works under the principle that the following danger point location is precisely known: the rear of the train ahead or movable track elements, like junctions. Following this principle, the minimum separation between trains is determined by the safe braking distance of the train plus an additional safety distance. In other words, the separation distance between trains is not fixed. Instead, it changes continuously depending on the trains' speeds or movable infrastructure elements [Gubler, 2023].

This operation can potentially increase the network capacity, with more trains per hour, without significant investment in the current infrastructure. However, its implementation could lead to considerable investments in train equipment, like telecommunication infrastructure, and onboard equipment maintenance [Gubler, 2023].

Braking Curve

A graphical representation of the Braking Distance of a train concerning the Gradient, the braking characteristics and the train's speed [RailSystem].

Balise

A track-mounted spot transmission unit that uses transponder technology. Its function is to transmit/receive messages to/from the train passing overhead [RailSystem]. These transponders can also be called beacons, and a good usage example is the Automatic Train Operation (ATO) spot [Gubler, 2023].

Global System for Mobile Communications-Railway

The Global System for Mobile Communications-Railway (GSM-R) delivers digital, secure, dependable communications between drivers and signallers. This helps to increase safety, reduce delays and improve performance – providing a better experience for passengers. The GSM-R supports secure and reliable driver-signaller communication by bringing together the most practical combination of technology, processes and people [NetworkRail].

European Rail Traffic Management System

A system for managing rail traffic, enabling it to operate on compatible Signalling Systems across European borders. Eight UNIFE members [UNIFE] developed this major industrial project in close cooperation with the European Union, railway stakeholders and the GSM-R industry. Members involved were Alstom Transport, AZD Praha, Bombardier Transportation, CAF, Hitachi Rail STS, Mermec, Siemens Mobility, and Thales Group.

European Train Control System

The European Train Control System (ETCS) is a train protection system designed to replace some incompatible systems used by European railways. It is part of European Rail Traffic Management System (ERTMS) and has been successfully adopted outside Europe. The ETCS contains three levels of automation, where the first level (ETCS Level 1) can be superimposed on the existing signalling system (Figure A.1), and the second level (ETCS Level 2) displays the train movement and other signalling aspects on onboard computers, with a digital radio-based signalling system like GSM-R (Figure A.2). However, train detection, track vacancy, and train integrity supervision must be monitored using a radio block centre from the trackside [ARC-Advisory].

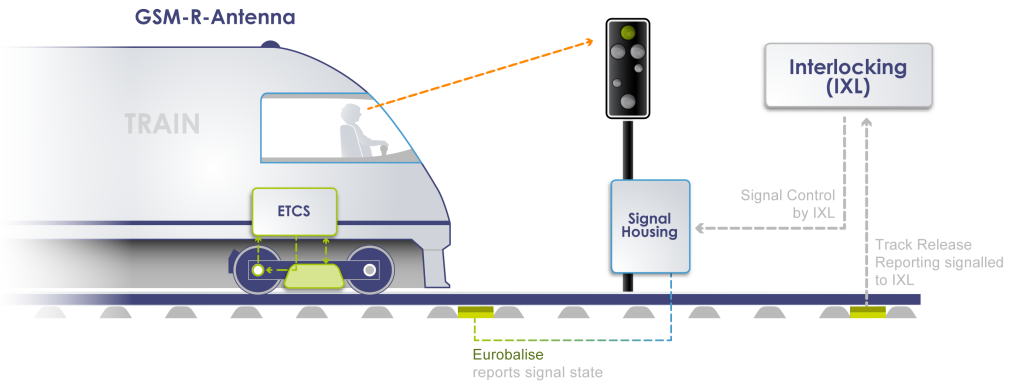

Figure A.1: ETCS Level 1 operation [Thales, c]

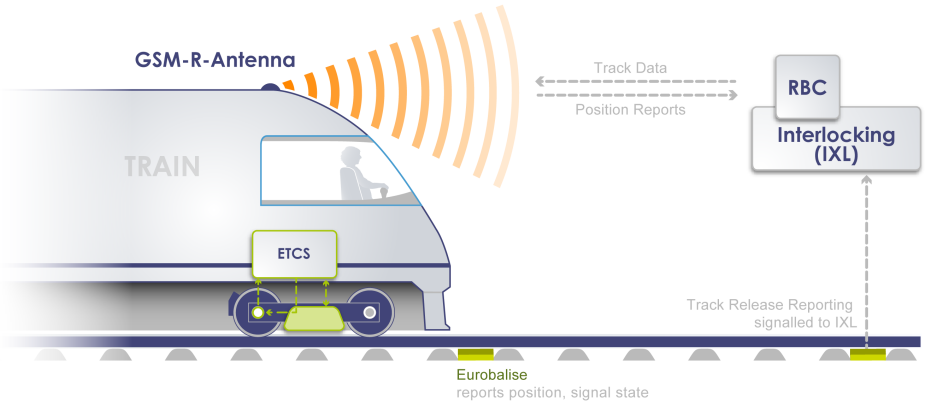

Figure A.2: ETCS Level 2 operation [Thales, c]

A third level (ETCS Level 3) is being developed and studied. The ETCS Level 3 eliminates the need for track vacancy detection components and operates at fixed intervals because it uses the onboard equipment to obtain short positioning data signals (Figure A.3). This level allows continuous line-clear authorisation and the possibility of implementing fixed or moving block operation. It maintains the GSM-R technology used in the previous level [ARC-Advisory] [Thales, c].

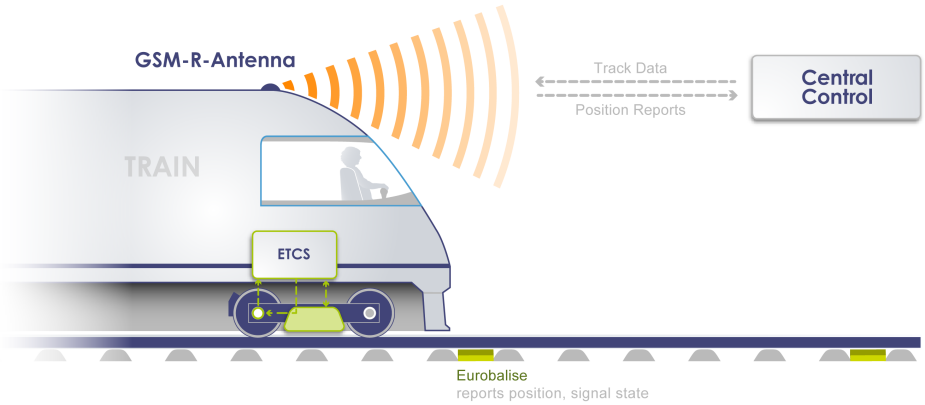

Figure A.3: ETCS Level 3 operation [Thales, c]

To sum up, different ETCS Levels can be achieved by having specific components in the system. These components will process the data received and optimise the network status. In parallel, the ETCS must receive the data needed to automate the operations. That information can only be passed with the right technologies. For example, suppose a train does not have a GSM-R antenna. In that case, achieving ETCS Level 2 and 3 is impossible because the interlocking cannot correctly communicate with the train.

Communication-Based Train Control

Communication-Based Train Control (CBTC) is a railway signalling system that uses the telecommunications between the train and trackside equipment for traffic management and infrastructure control. The system operates under the logic of the moving block separation principle, and its foundation is based on track-to-train communication. CBTC can determine the precise train position at any given time independent of using Track Circuits or Axle Counters and implementing functions related to safe train operation and optional functions related to driverless operation (e.g. speed regulation) and dispatching rules. This system implements many automation functionalities faster than ETCS for urban mass transit applications in Vancouver, Canada and Beijing, China [Martin, 2020].

Union Internationale des Transports Publics

The Union Internationale des Transports Publics (UITP) is a global network that brings more sustainable transport modes with more than nineteen hundred members over more than one hundred countries. This company's scale makes this organisation the only worldwide network to unite all public transport stakeholders. UITP is inserted in the railway world as a member-led organisation for public transport authorities, networks and operators, policy decision-makers, scientific institutes, and the public transport supply and service industry that works to advance sustainable urban mobility. UITP has proposed the definition of Grade of Automation (GoA).

Grade of Automation

A classification of the levels that make the automation of railway systems possible by functionalities and responsibility. The lowest level of automation is GoA 1, where the driver performs all the main functionalities (setting the train in motion, stopping the train, door closure and operation in the event of a disruption) and uses only the ATP subsystem. On the other hand, at the highest level, GoA 4, the train control system regulates all the main functionalities. Only assigning a responsible person is necessary to ensure the correct system functioning. This was suggested by UITP [Barrow, 2018], and the Table A.1 shows the levels of GoA.

Level	**Operation**	**Motion**	**Stopping**	**Door Closure**	**Disruptions**	**Examples**
GoA 1	ATP with Driver	Driver	Driver	Driver	Driver	ETCS Level 1
GoA 2	ATP and ATO with Driver	Automatic	Automatic	Driver	Driver	ETCS Level 2 and 3
GoA 3	Driverless	Automatic	Automatic	Train Attendant	Train Attendant	
GoA 4	Automatic	Automatic	Automatic	Automatic	Automatic	CBTC at full capacity

Table A.1: Levels of GoA [Martin, 2020]

UITP vs CBTC vs ETCS

After introducing all the concepts, it is crucial to compare how the different systems and organisations define the TMS. The Table A.2 shows how UITP, CBTC and ETCS define it, what components are integrated and what is optional [Martin, 2020].

For UITP, the ATP has to ensure basic safety, whereas the ATO assumes the functions of the driver except for the door-closing action. UITP defines TMS as a general class of ATP that is in charge of route setting and train regulation [Martin, 2020].

In the CBTC case, concerning the standards previously described by UITP, it is added the term ATS, whose function is to monitor trains and adjust their performance to maintain schedules. Regarding ATP and ATO functionalities, CBTC systems have similar meanings when compared with the ones used by UITP. Although CBTC Standard distinguishes ATP, ATO, and ATS as different subsystems inside the TMS, it must include ATP. It may have ATO, which allows driverless operation, and ATS, which allows rescheduling operation, but it needs to have the three functionalities to achieve complete automation, GoA Level 4 [Martin, 2020].

Finally, the ETCS as a Train Control System is closer related to the definition of ATP of CBTC Standards and UITP. In contrast with CBTC, ERTMS does not contemplate the whole automation of ETCS as an objective. The ETCS is considered a fail-safe signalling component primarily independent of other optimisation objectives [Martin, 2020].

	UITP	**CBTC**	**ETCS**
TMS Mandatory Functions	ATP	ATP	ATP
TMS Optional Functions		ATO	
Crucial Functions	ATO		ATO
Optional Functions		ATS	

Table A.2: TMS components [Martin, 2020]

ETCS Levels vs CBTC

To develop future train control systems (believed the evolution of ETCS system) with higher compatibility to CBTC, it was created a project called Next Generation Train Control Systems (NGTC). This project focuses on the standard functions between the ETCS and CBTC, which means the ATP functions. The Table A.3 permits us to evaluate and compare the different characteristics [Martin, 2020].

Levels	**ETCS Level 1**	**ETCS Level 2**	**ETCS Level 3**	**CBTC**
Application	(mostly) Heavy	(mostly) Heavy	(mostly) Heavy	(mostly) Light
Operation	Fixed Block	Fixed Block	Moving Block	Moving Block
Lineside Signals	Yes	Not needed	Not needed	Not needed
Interoperability & Interchangeability	Yes	Yes	Yes	No
Functionalities	ATP	ATP with ATO	ATP with ATO	ATP, ATO, ATS
Standard	ERTMS	ERTMS	Not yet specified	Partially by IEEE
Route Interlocking	Yes	Yes	Not yet specified	Yes
Type Data Transmission	Spot and Unidirectional	Continuous Bidirectional Track-to-Train	Continuous Bidirectional Track-to-Train	Continuous Bidirectional Track-to-Train
Vital Information Transmission	Lineside signals, Electronic Unit, controlled balises	Radio communication via GSM-R	Radio communication (to be defined)	Radio communication via Wi-Fi
Track Clearance Detection	Track Circuit / Axle Counter	Track Circuit / Axle Counter	On-board (to be defined)	Continuous message

Table A.3: ETCS and CBTC characteristics [Martin, 2020]

Appendix B

Gantt Charts

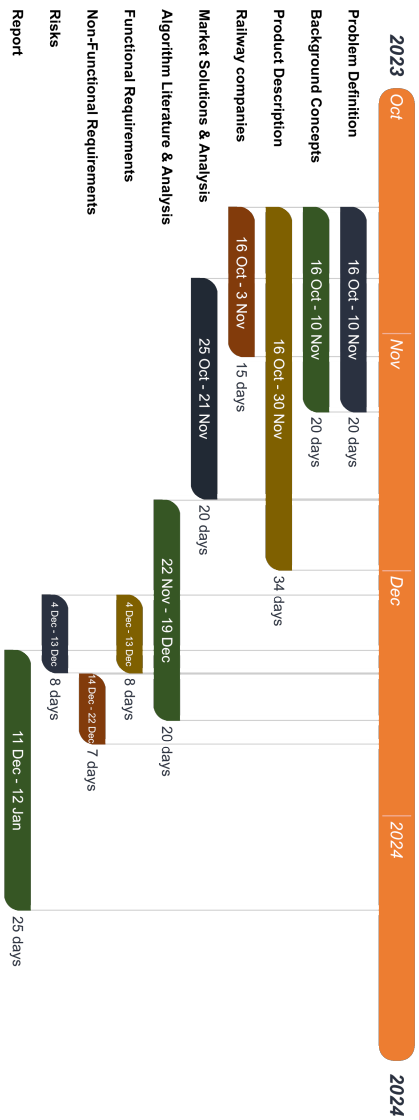

Figure B.1: Work Plan of the first semester

Figure B.2: Work Plan of the second semester

Appendix C

Risks

Name	Lack of documentation
Risk	The producers of FTCMS's algorithms no longer work in CSW, and there is no formal documentation about the messages that are exchanged internally. Therefore, it may not be easy to understand what is inside the messages received as input and what is supposed to be the output.
Impact	Medium
Probability	High
Origin	Identified from CSW mentor's concern where it was identified that the producers of **trafficManager** and **clientSolverApp** have already left the company and there is no formal document explaining how things were constructed and what they mean.
Mitigation Plan	Define a period in the work plan to be fully dedicated to learning the undocumented content. Additionally, ask for the help of the CSW mentor and consider allocating more time in the work plan to understand the functions of the FTCMS.

Table C.1: Risk 1 - Lack of documentation

Name	No time to learn about the programming language
Risk	The author is unfamiliar with the programming Language of the FTCMS. Therefore, he might not have time to learn it properly.
Impact	Medium
Probability	Low
Origin	Although the NFR1 is not a top priority, it was identified as a possibility to produce the PoC since all code in FTCMS is in C++. Consequently, this risk was identified to demonstrate that it was considered, and a mitigation plan was prepared.
Mitigation Plan	Devise tasks in the work plan to learn the language and go to CSW C++ training. Alternatively, evaluate the possibility of using another programming language that suits the project's needs (Python or Java, for example).

Table C.2: Risk 2 - No time to learn about the programming language

Name	No time to learn about the chosen model
Risk	The author is unfamiliar with the model chosen to implement in the PoC. Therefore, he might not have time to learn it properly.
Impact	High
Probability	Low
Origin	Since the most crucial part of the PoC relies on something new to the author, this risk represents a situation improbable to happen but with a huge impact on the project.
Mitigation Plan	Allocate time in the work plan to study the model and its multiple variations of types and prepare the implementation by creating some examples based on courses on the internet or from DEI subjects bibliography.

Table C.3: Risk 3 - No time to learn about the chosen model

Name	Not having the task delivered on time
Risk	The deadline is tight. Therefore, the tasks may be delivered later than planned.
Impact	High
Probability	Low
Origin	Prevalent risk, although some projects do not consider it. In this case, it is important to identify it because it has a paramount impact on the project.
Mitigation Plan	Review the work plan regularly to keep the time of tasks well adjusted to the development made at that point.

Table C.4: Risk 4 - Not having the task delivered on time

Name	Wrong estimations on the task duration
Risk	The author is not an expert in performing estimations. Therefore, he may misestimate the duration of the tasks.
Impact	Medium
Probability	Medium
Origin	Typical risk in many projects. It does not have a significant impact but is relatively likely, so a mitigation plan is necessary.
Mitigation Plan	Review the work plan regularly and ask for advice from senior colleagues when estimating the duration of the tasks.

Table C.5: Risk 5 - Wrong estimations on the task duration

Name	Changes in the development plan due to unknown information
Risk	We are working on a black-box application. Therefore, understanding the application may lead to a change in the development plan.
Impact	High
Probability	High
Origin	Identified at the beginning of the 2^{nd} semester that was not considered at the beginning of the project. It was mandatory to identify it since it may greatly impact the project's future steps.
Mitigation Plan	Call a meeting with both mentors to discuss the changes that might be done in the development and check if both mentors approve the changes. Further, analyse what might change in the initial chapters of the report that were delivered earlier. Also, explain what caused this change.

Table C.6: Risk 6 - Changes in the development plan due to unknown information

Appendix D

Simulation Environment

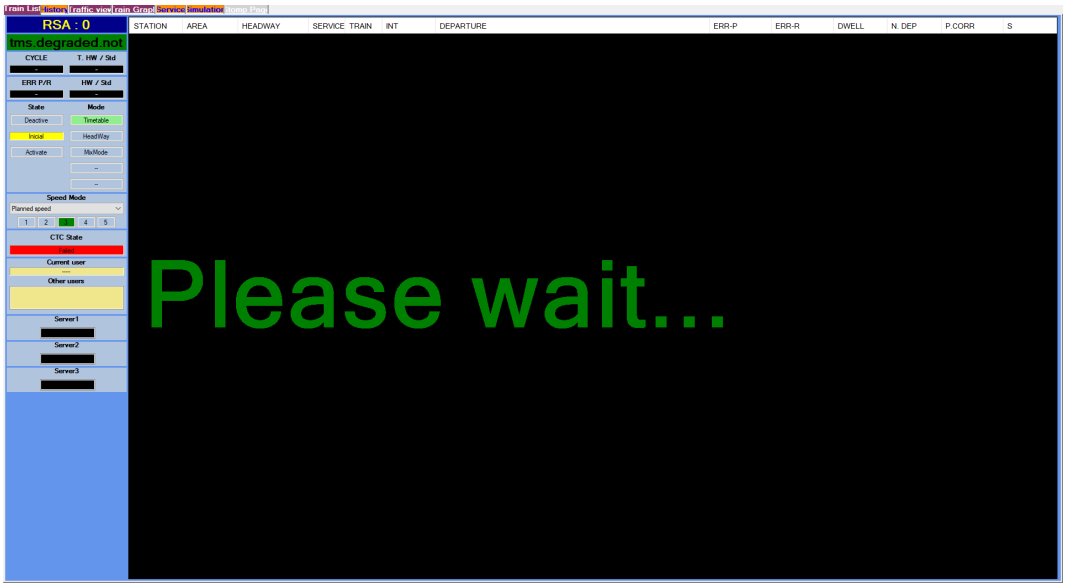

Figure D.1: Successful connection to the TMS

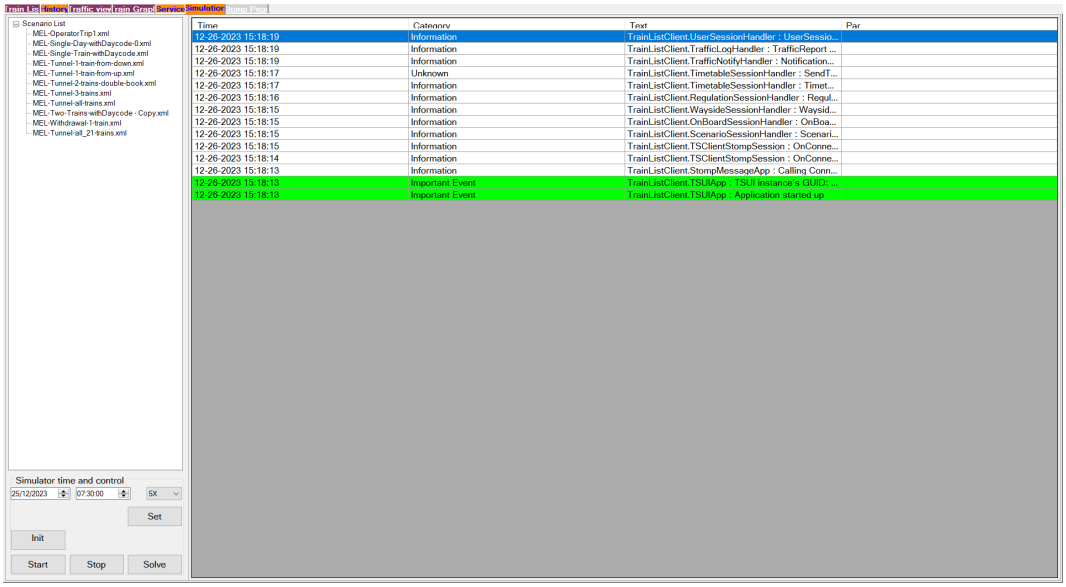

Figure D.2: Scenario loaded into the simulator

Figure D.3: Geographic map of the network with the trains

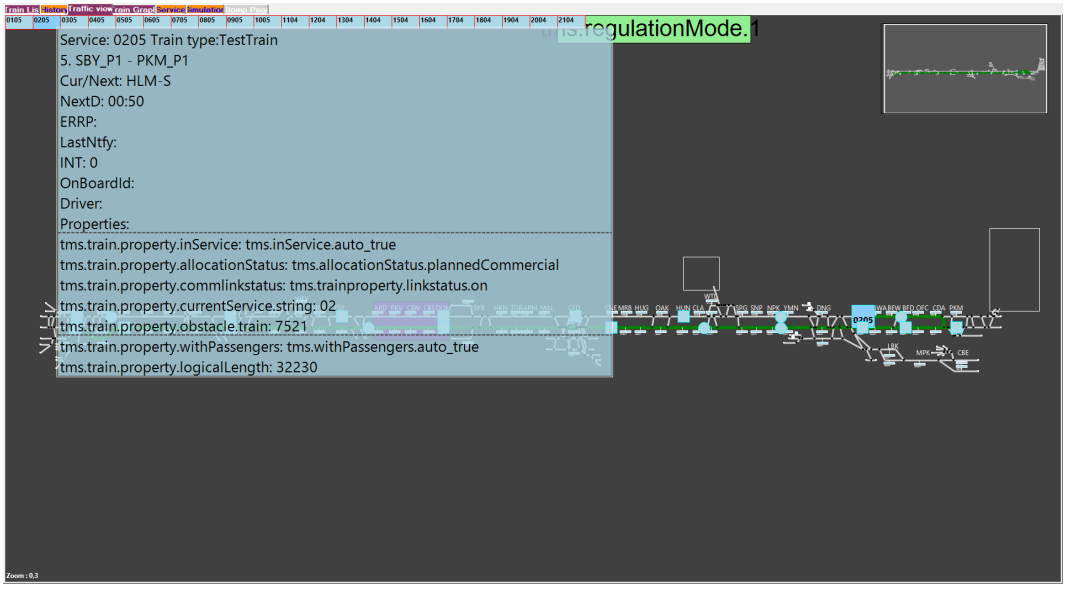

Figure D.4: Information about a specific train

Appendix E

Messages Structure

Listing E.1: Master Problem Message

```
<MasterProblem>
    <MasterID> 43 </MasterID>
    <SharedObjective> false </SharedObjective>
    <Vars>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="1"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="2"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="3"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="63"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="64"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="65"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="141"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="142"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="143"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="203"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="204"/>
        <X Cur="20.00" Min="15.00" Max="40.00" GUID="205"/>
    </Vars>
    <Sets>
        <S N="1">
            <GUID>1</GUID>
            <Vars> <X>1</X> </Vars>
        </S>
        <S N="2">
            <GUID>2</GUID>
            <Vars> <X>1</X> <X>2</X> </Vars>
        </S>
        <S N="3">
            <GUID>3</GUID>
            <Vars> <X>1</X> <X>2</X> <X>3</X> </Vars>
        </S>
        <S N="1">
            <GUID>4</GUID>
            <Vars> <X>63</X> </Vars>
        </S>
        <S N="2">
```

```
            <GUID>5</GUID>
            <Vars> <X>63</X> <X>64</X> </Vars>
        </S>
        <S N="3">
            <GUID>6</GUID>
            <Vars> <X>63</X> <X>64</X> <X>65</X> </Vars>
        </S>
        <S N="1">
            <GUID>7</GUID>
            <Vars> <X>141</X> </Vars>
        </S>
        <S N="2">
            <GUID>8</GUID>
            <Vars> <X>141</X> <X>142</X> </Vars>
        </S>
        <S N="3">
            <GUID>9</GUID>
            <Vars> <X>141</X> <X>142</X> <X>143</X> </Vars>
        </S>
        <S N="1">
            <GUID>10</GUID>
            <Vars> <X>203</X> </Vars>
        </S>
        <S N="2">
            <GUID>11</GUID>
            <Vars> <X>203</X> <X>204</X> </Vars>
        </S>
        <S N="3">
            <GUID>12</GUID>
            <Vars> <X>203</X> <X>204</X> <X>205</X> </Vars>
        </S>
    </Sets>
    <Params>
        <Abs> 300.00 </Abs>
        <IniPhi> 50.00 </IniPhi>
        <IncPhi> 1.10 </IncPhi>
        <Theta> 0.00 </Theta>
        <AbOjb1> 0.00 </AbOjb1>
        <AlphaX> 0.00 </AlphaX>
        <MaxIte> 2000 </MaxIte>
        <MaxImp> 100 </MaxImp>
        <MaxUf> 100 </MaxUf>
        <SendAll> true </SendAll>
    </Params>
</MasterProblem>
```

Listing E.2: Solve Message

```
<Solve>
    <MasterID> 43 </MasterID>
    <UID> 43 </UID>
    <StaticPenalty> 0.00 </StaticPenalty>
    <Constraints N="0"/>
    <Obj>
        <Ys>
            <Fun Type="2" N="2" S="0">
                <Sta Id="2243">
                    <WeC> 0.50 </WeC>
                    <Pols N="1">
                        <Pol>
                            <Con> -94.82 </Con>
                            <VarSetID> 10 </VarSetID>
                        </Pol>
                    </Pols>
                </Sta>
                <Sta Id="2138">
                    <WeC> 0.50 </WeC>
                    <Pols N="1">
                        <Pol>
                            <Con> -17.41 </Con>
                            <VarSetID> 7 </VarSetID>
                        </Pol>
                    </Pols>
                </Sta>
            </Fun>
        </Ys>
    </Obj>
</Solve>
```

Listing E.3: Solution Message

```
<Solution>
    <MasterID> 43 </MasterID>
    <UID> 43 </UID>
    <PrivateClientQueue>
        jms. queue.TMS.Solver2.rsa-r21-tms1
    </PrivateClientQueue>
    <Value> 6030.22 </Value>
    <StaticPenalty> 0.00 </StaticPenalty>
    <Vars>
        <X Cur="20.00" GUID="1"/>
        <X Cur="20.00" GUID="2"/>
        <X Cur="20.00" GUID="3"/>
        <X Cur="20.00" GUID="63"/>
        <X Cur="20.00" GUID="64"/>
        <X Cur="20.00" GUID="65"/>
        <X Cur="17.41" GUID="141"/>
        <X Cur="20.00" GUID="142"/>
        <X Cur="20.00" GUID="143"/>
        <X Cur="15.00" GUID="203"/>
        <X Cur="20.00" GUID="204"/>
        <X Cur="20.00" GUID="205"/>
    </Vars>
</Solution>
```

Appendix F

Requirements

F.1 Functional

Functional Requirement	
ID	FR1
Name	Read messages from the message bus
MoSCoW	Must
Effort	Medium
Type	Use Case
Description	The Proof-of-Concept must be able to read messages sent to the queues from the Artemis message bus so they can be processed.

Table F.1: FR1 - Read messages from the message bus

Functional Requirement	
ID	FR2
Name	Send messages to the message bus
MoSCoW	Must
Effort	Medium
Type	Use Case
Description	The Proof-of-Concept must be able to send messages to the message bus' topics to transmit necessary information to the other components.

Table F.2: FR2 - Send messages to the message bus

Functional Requirement	
ID	FR3
Name	Serialise and Deserialise messages
MoSCoW	Must
Effort	Medium
Type	Use Case
Description	The algorithm must be able to deserialise messages received from the message bus and serialise new messages to send as a reply.

Table F.3: FR3 - Serialise and Deserialise messages

Functional Requirement	
ID	FR4
Name	Create a problem instance
MoSCoW	Must
Effort	High
Type	Use case
Description	The algorithm must be able to create a problem object with the content of a master problem message received earlier.

Table F.4: FR4 - Create a problem instance

Functional Requirement	
ID	FR5
Name	Add additional data to a problem
MoSCoW	Must
Effort	High
Type	Use case
Description	The algorithm must be able to add to a problem instance any additional data received later in the execution.

Table F.5: FR5 - Add additional data to a problem

Functional Requirement	
ID	FR6
Name	Create an empty solution
MoSCoW	Must
Effort	Low
Type	Use Case
Description	The algorithm must be able to create an empty solution instance that represents the starting point of the solving process.

Table F.6: FR6 - Create an empty solution

Functional Requirement	
ID	FR7
Name	Find a solution to the problem
MoSCoW	Must
Effort	High
Type	Use Case
Description	The algorithm must solve the problem received by finding the optimal solution to the status of the problem.

Table F.7: FR7 - Find a solution to the problem

Functional Requirement	
ID	FR8
Name	Update the content received in the problem
MoSCoW	Must
Effort	High
Type	Use Case
Description	The algorithm must partially replicate the content of the master problem message with some of the information updated with the solution results to send it as a reply.

Table F.8: FR8 - Update the content received in the problem

F.2 Non-Functional

Non-Functional Requirement	
ID	NFR1
Name	C++ as Programming Language
MoSCoW	Could
Effort	Medium
Type	Constraint
Description	To ease the integration of the Proof-of-Concept in the product, it could be developed in **C++**. Alternatively, it can be developed in another language, like **Python**.

Table F.9: NFR1 - C++ as Programming Language

Non-Functional Requirement	
ID	NFR2
Name	Avoid delay propagation
MoSCoW	Must
Effort	High
Type	Scalability
Description	The model must avoid delay propagation among trains by correctly solving the problems that arise.

Table F.10: NFR2 - Avoid delay propagation

Non-Functional Requirement	
ID	NFR3
Name	Minimise the impact of new traffic
MoSCoW	Should
Effort	High
Type	Scalability
Description	The model should be able to minimise the impact of adding more trains to the network. Adding new traffic should not delay trains already running in the network.

Table F.11: NFR3 - Minimise the impact of new traffic

Non-Functional Requirement	
ID	FR4
Name	Improve the speed performance
MoSCoW	Could
Effort	High
Type	Performance
Description	The model developed could improve the speed performance compared to the other algorithms of the FTCMS in normal work conditions. The performance can also not improve if the PoC is made in another programming language.

Table F.12: NFR4 - Improve the speed performance

Non-Functional Requirement	
ID	NFR5
Name	Results comparability
MoSCoW	Must
Effort	Medium
Type	Performance
Description	The results produced by the Proof-of-Concept (PoC) must be able to be compared with the results of other mathematical algorithms of the product. This comparability metric refers to time execution and disruption value results after solving.

Table F.13: NFR5 - Results comparability

Non-Functional Requirement	
ID	NFR6
Name	Model's Accuracy
MoSCoW	Must
Effort	High
Type	Performance
Description	To be accepted as a possible candidate, the Proof-of-Concept's model must achieve a minimum of 50% of accuracy in testing.

Table F.14: NFR6 - Model's Accuracy

Appendix G

Test Results

G.1 Models Comparison

G.1.1 Recurrent Neural Network Model

RNN Loss Results

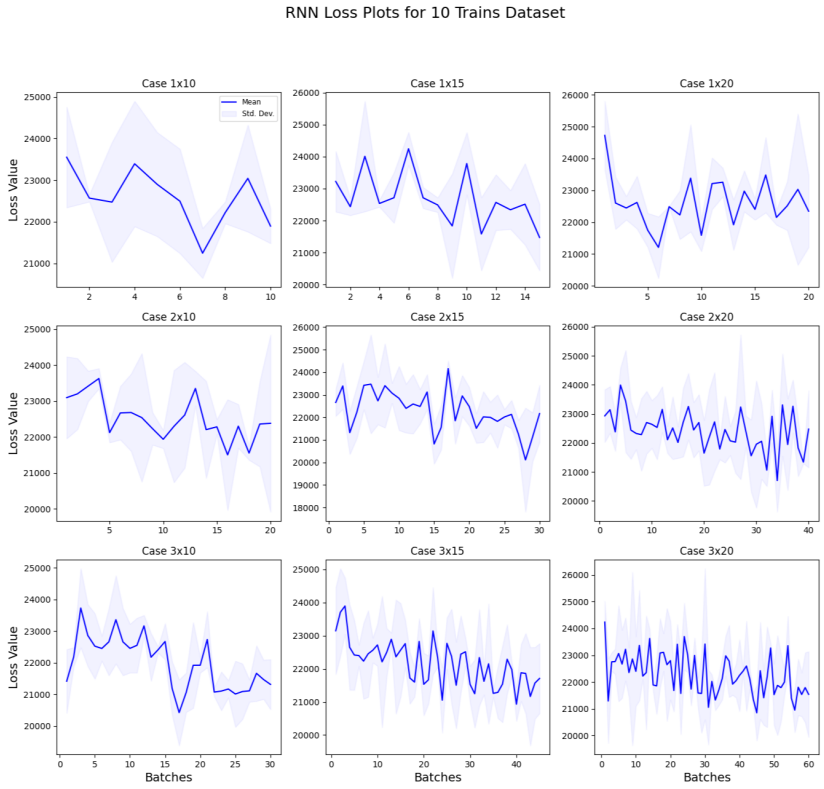

Figure G.1: Loss Results for the RNN in the 10 Trains Dataset

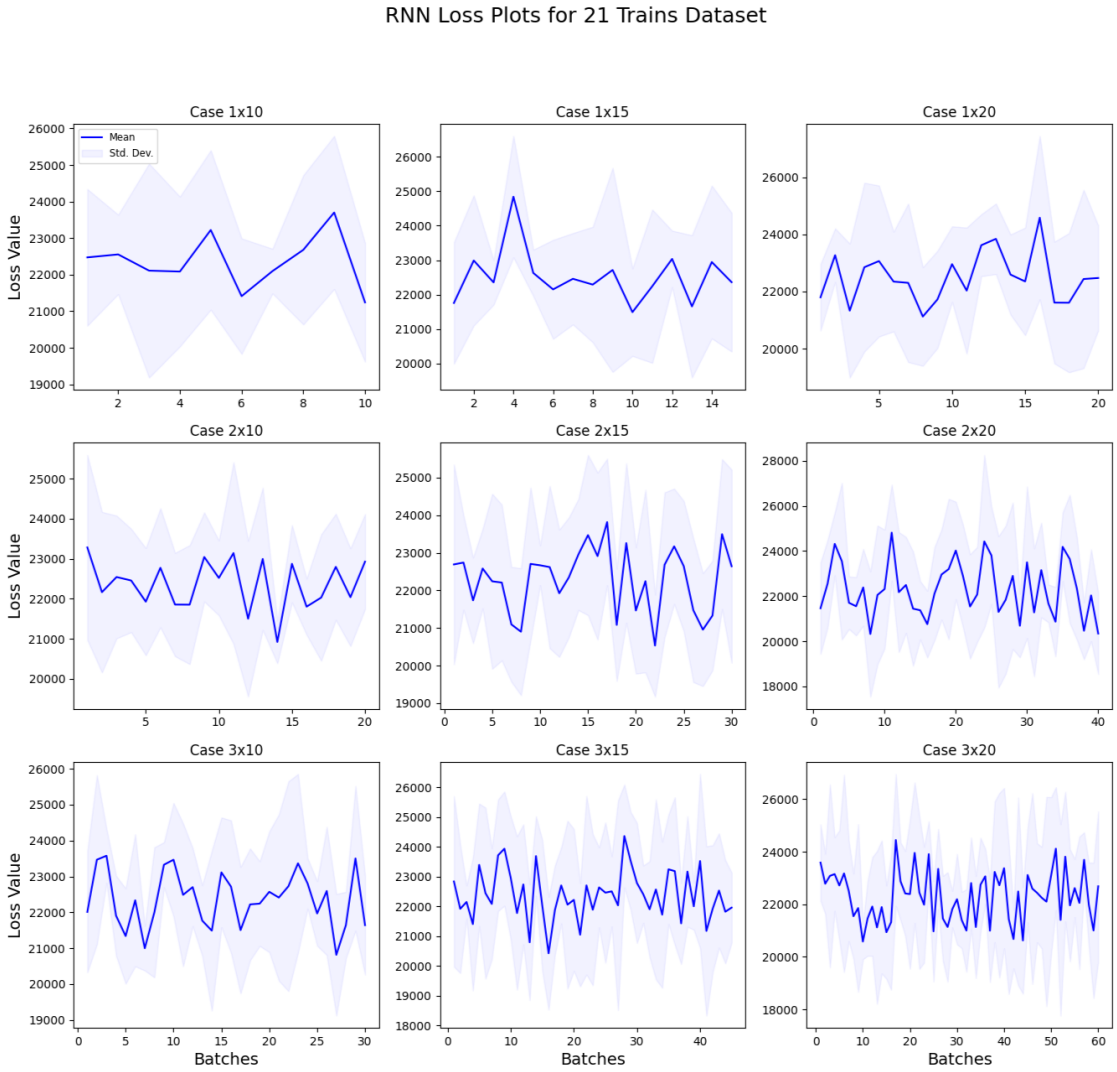

Figure G.2: Loss Results for the RNN in the 21 Trains Dataset

RNN Accuracy Results

Figure G.3: Accuracy Results for the RNN in the 10 Trains Dataset

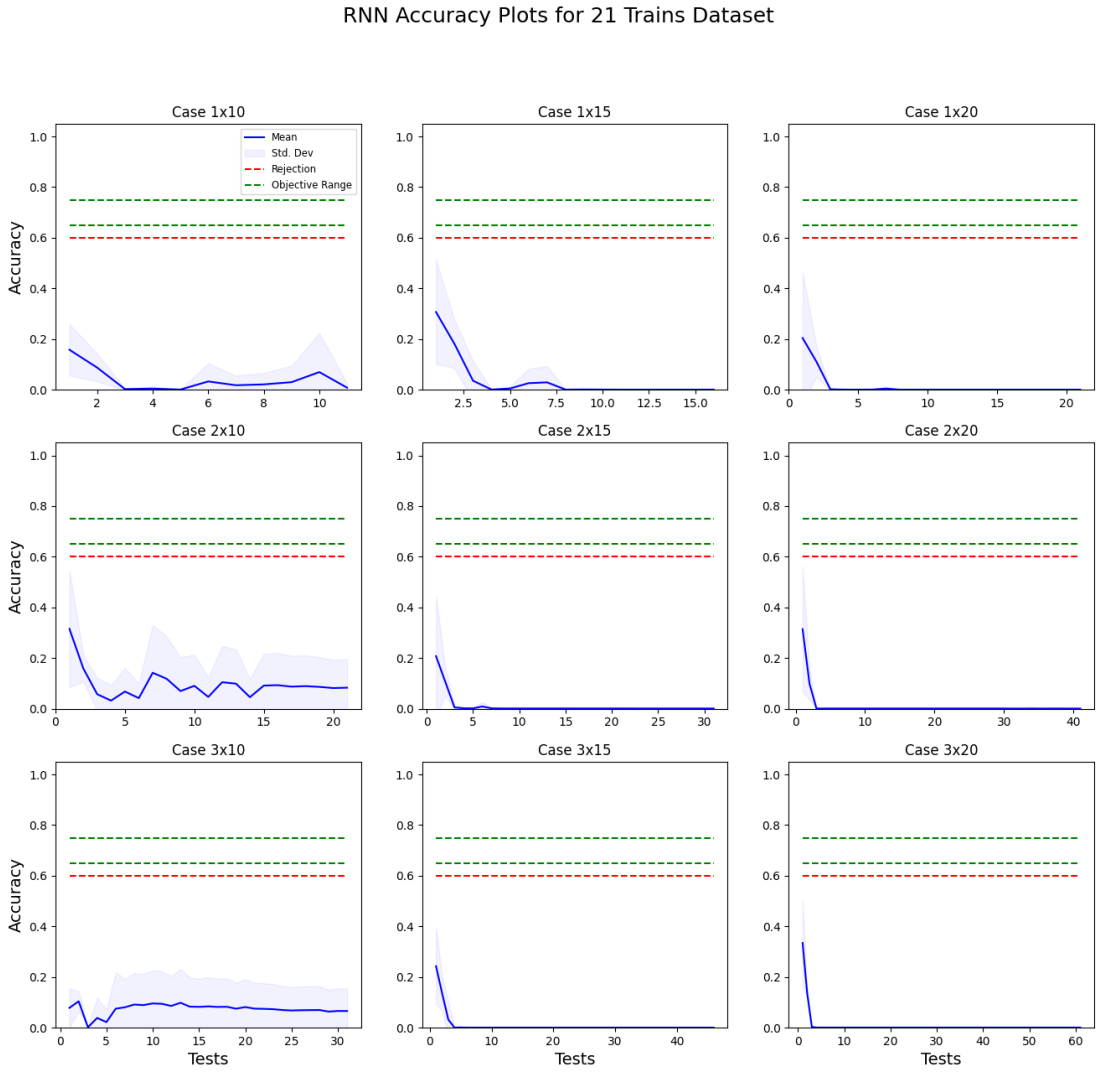

Figure G.4: Accuracy Results for the RNN in the 21 Trains Dataset

G.1.2 Long Short-Term Memory Model

LSTM Loss Results

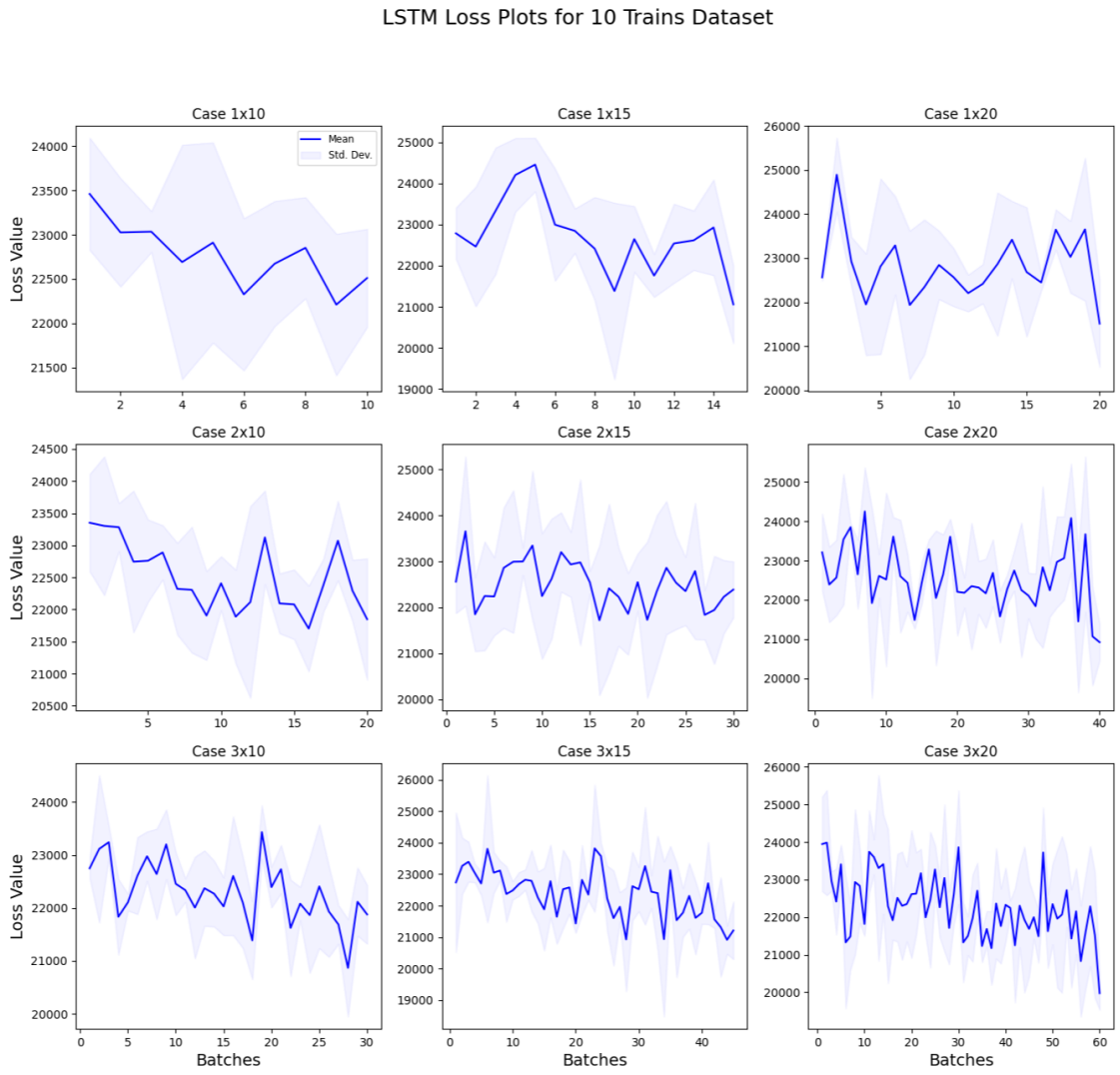

Figure G.5: Loss Results for the LSTM in the 10 Trains Dataset

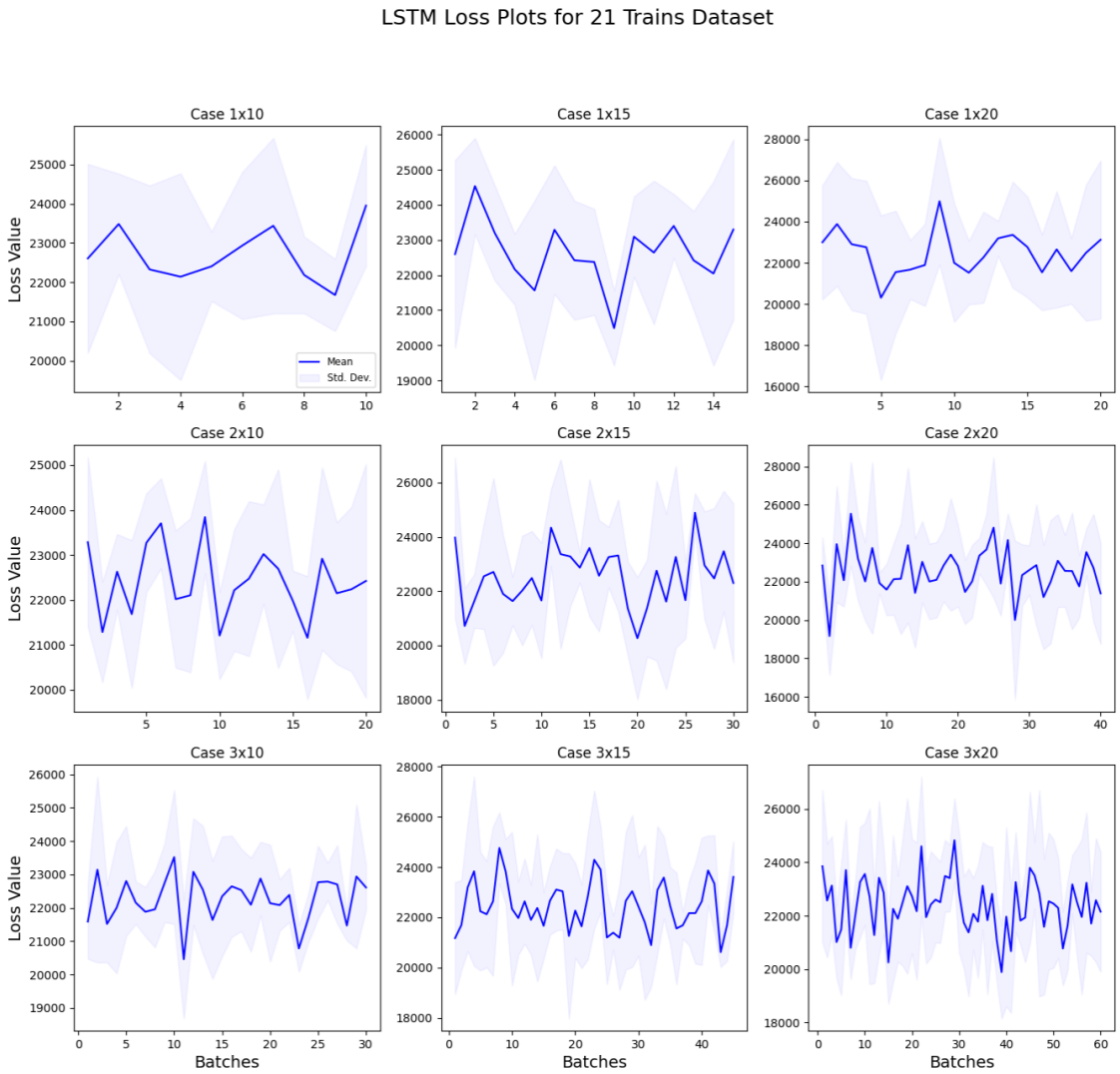

Figure G.6: Loss Results for the LSTM in the 21 Trains Dataset

LSTM Accuracy Results

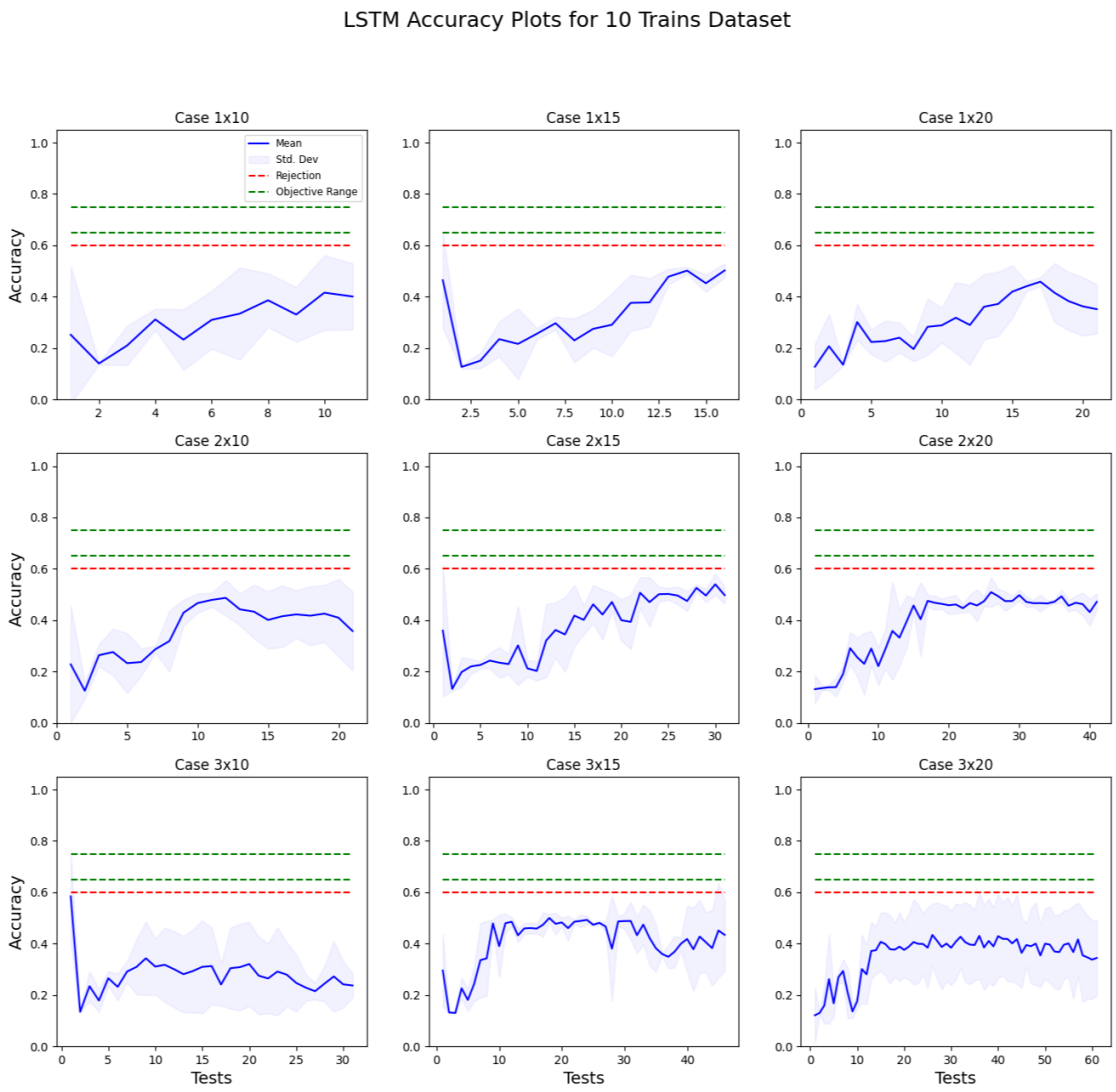

Figure G.7: Accuracy Results for the LSTM in the 10 Trains Dataset

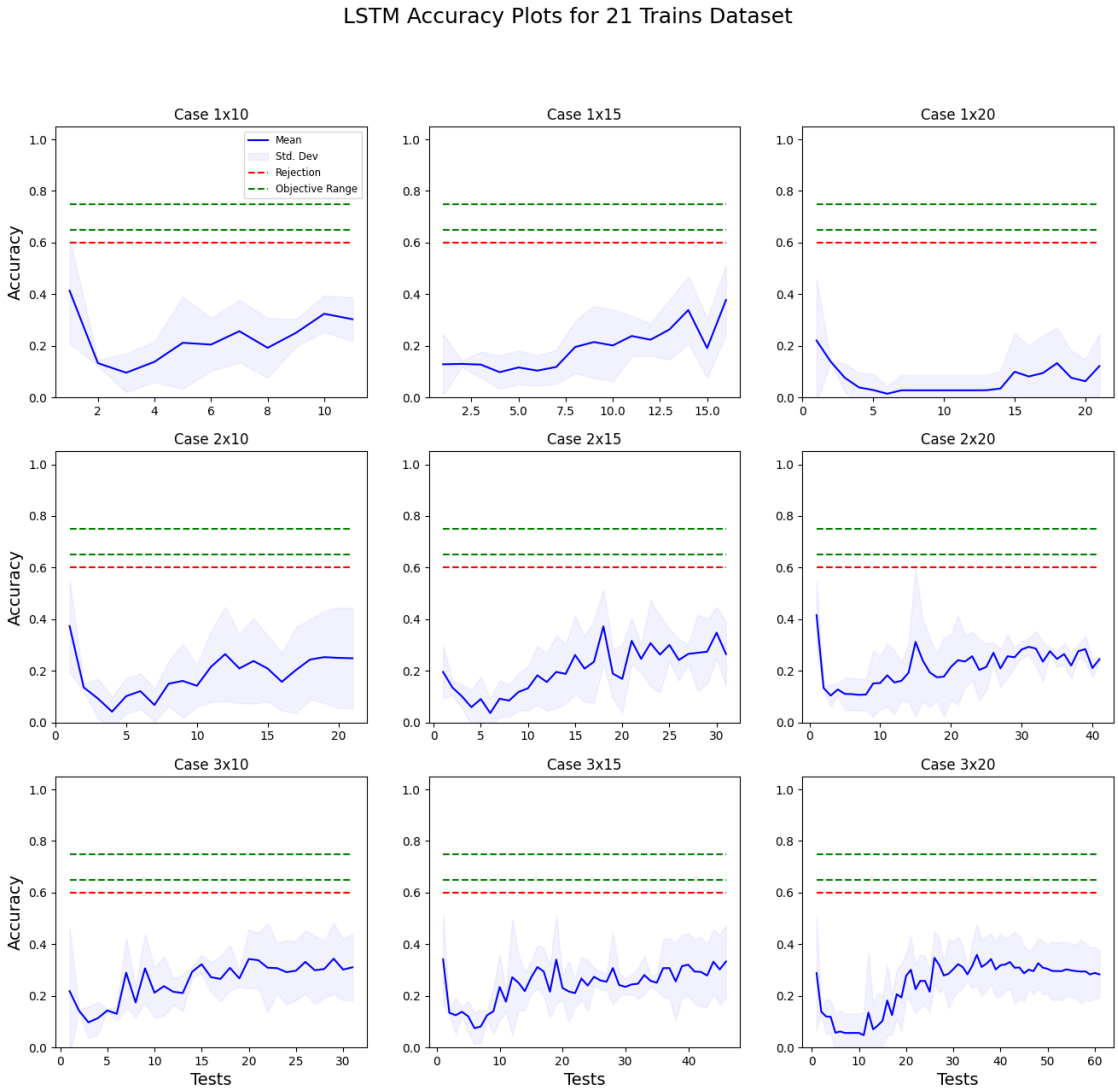

Figure G.8: Accuracy Results for the LSTM in the 21 Trains Dataset

G.1.3 Linear Neural Network Model

LNN Loss Results

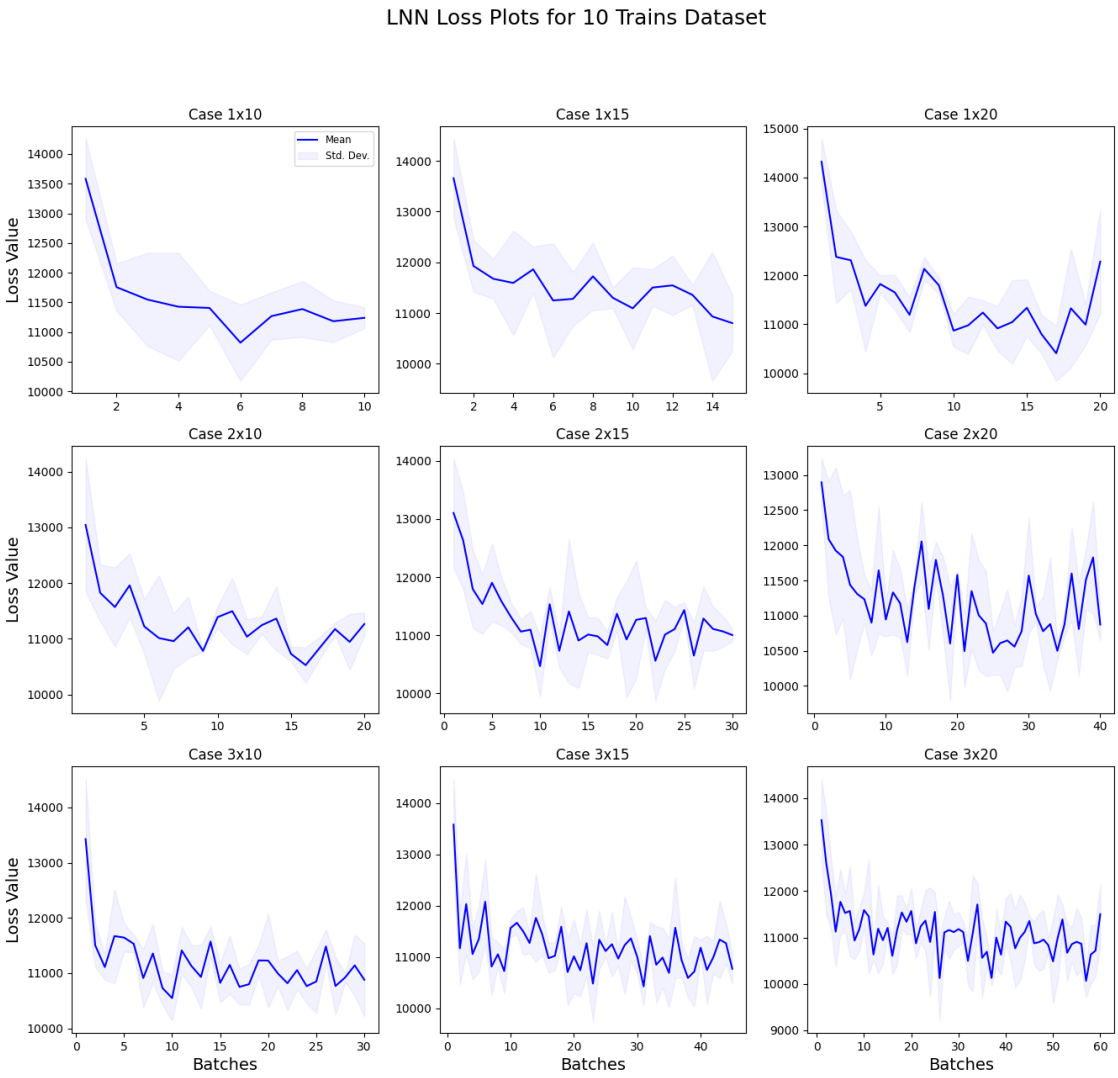

Figure G.9: Loss Results for the LNN in the 10 Trains Dataset

LNN Accuracy Results

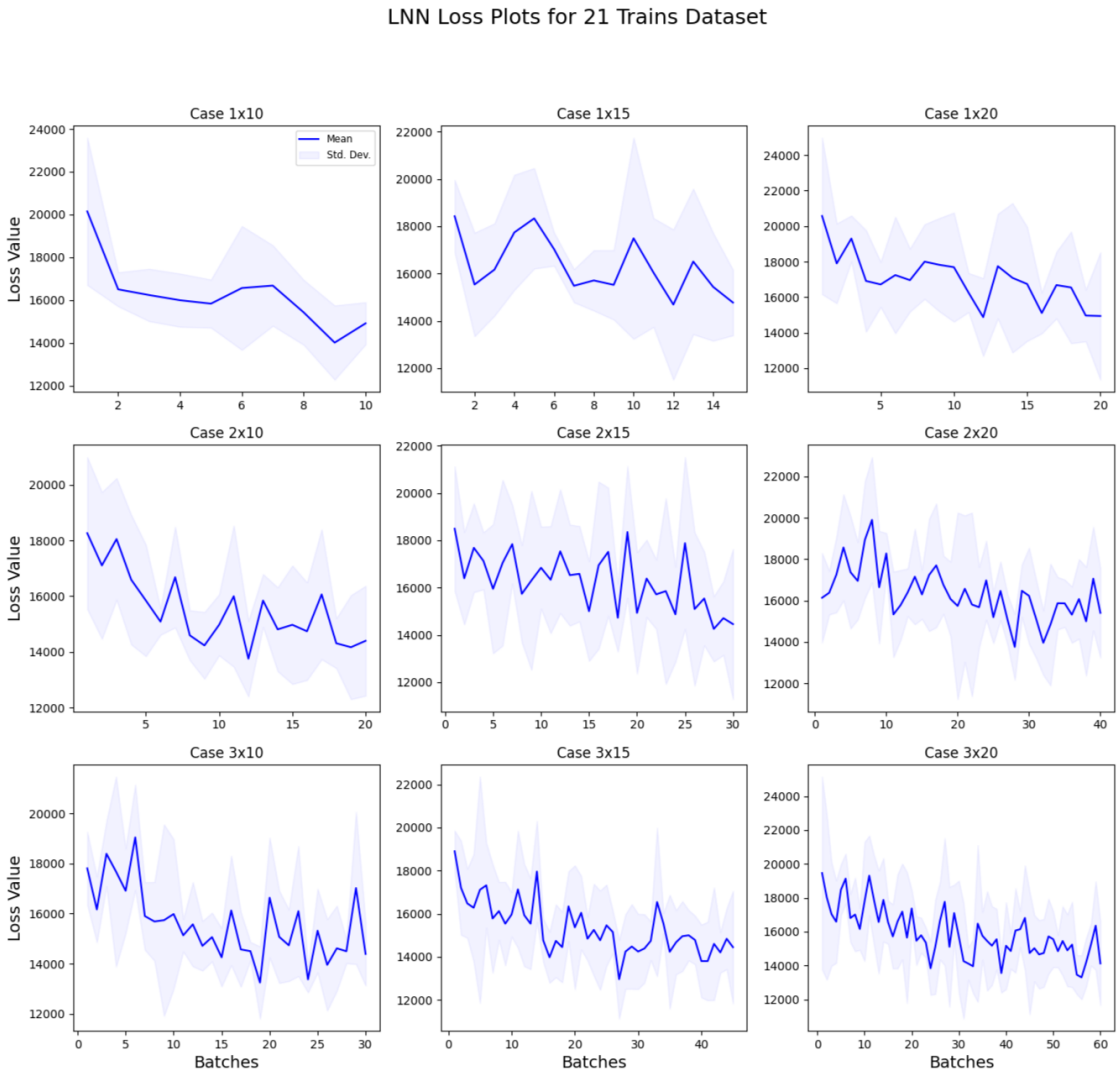

Figure G.10: Loss Results for the LSTM in the 21 Trains Dataset

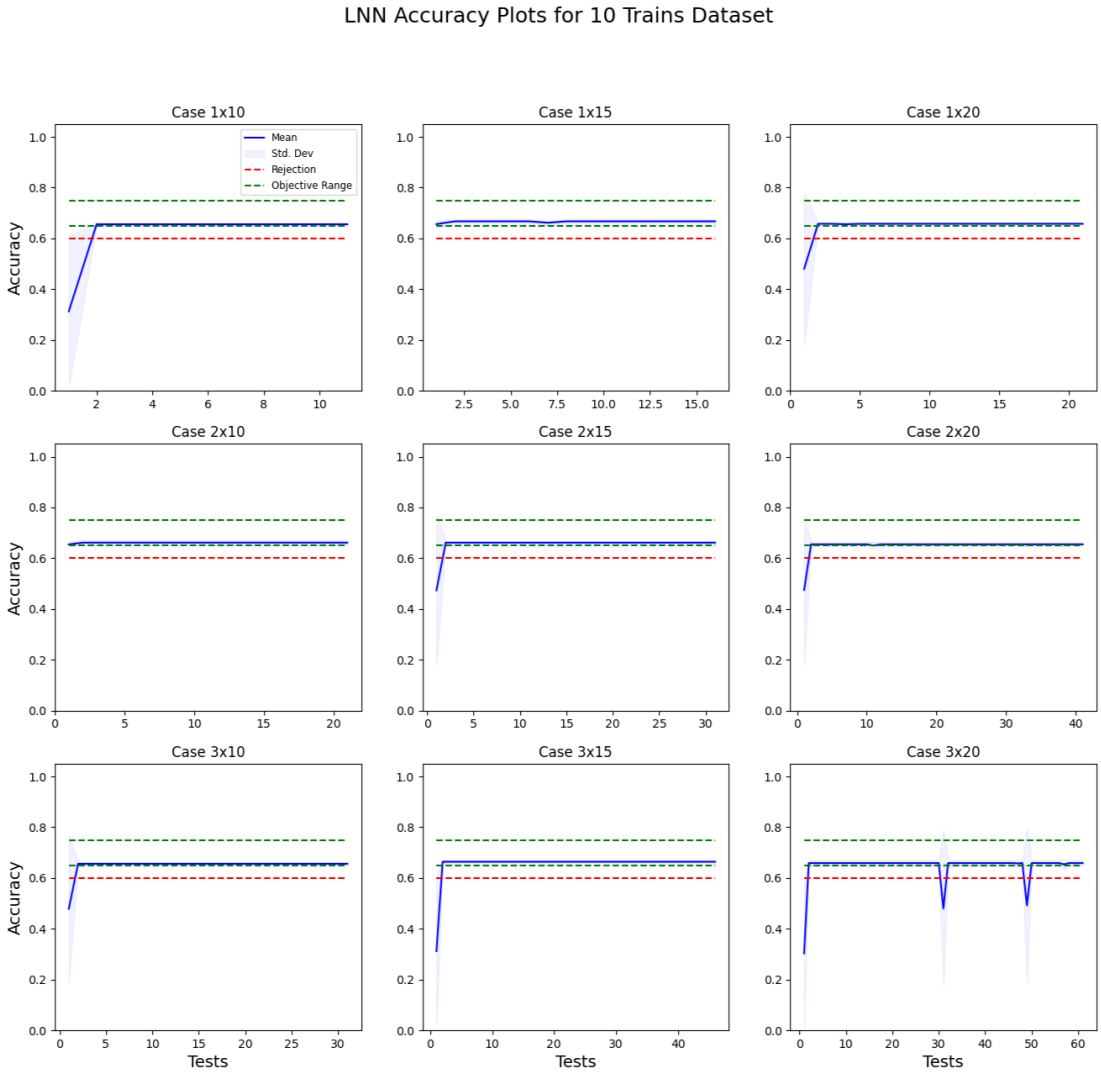

Figure G.11: Accuracy Results for the LNN in the 10 Trains Dataset

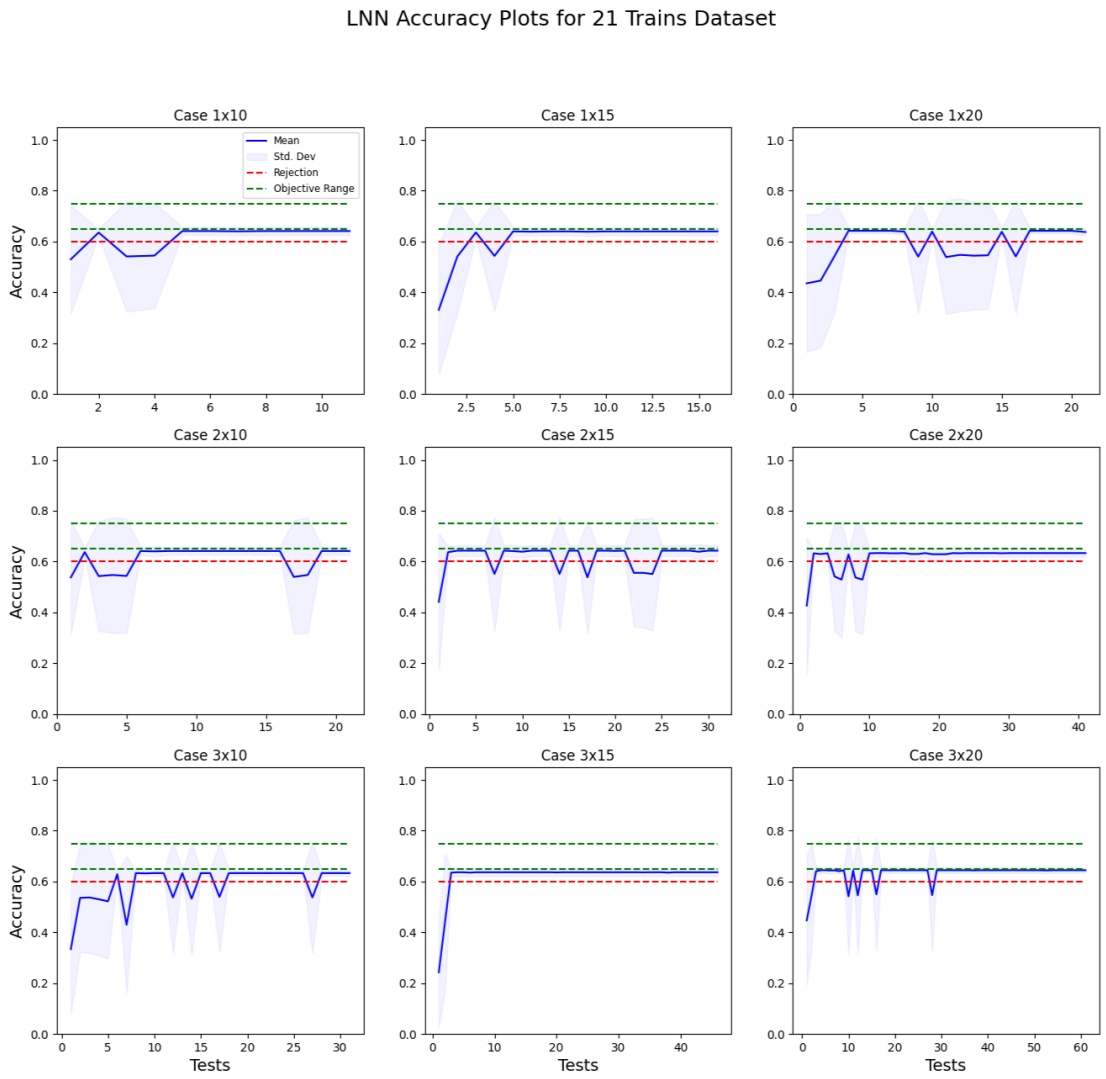

Figure G.12: Accuracy Results for the LNN in the 21 Trains Dataset

G.1.4 Feed Forward Neural Network Model

FFNN Loss Results

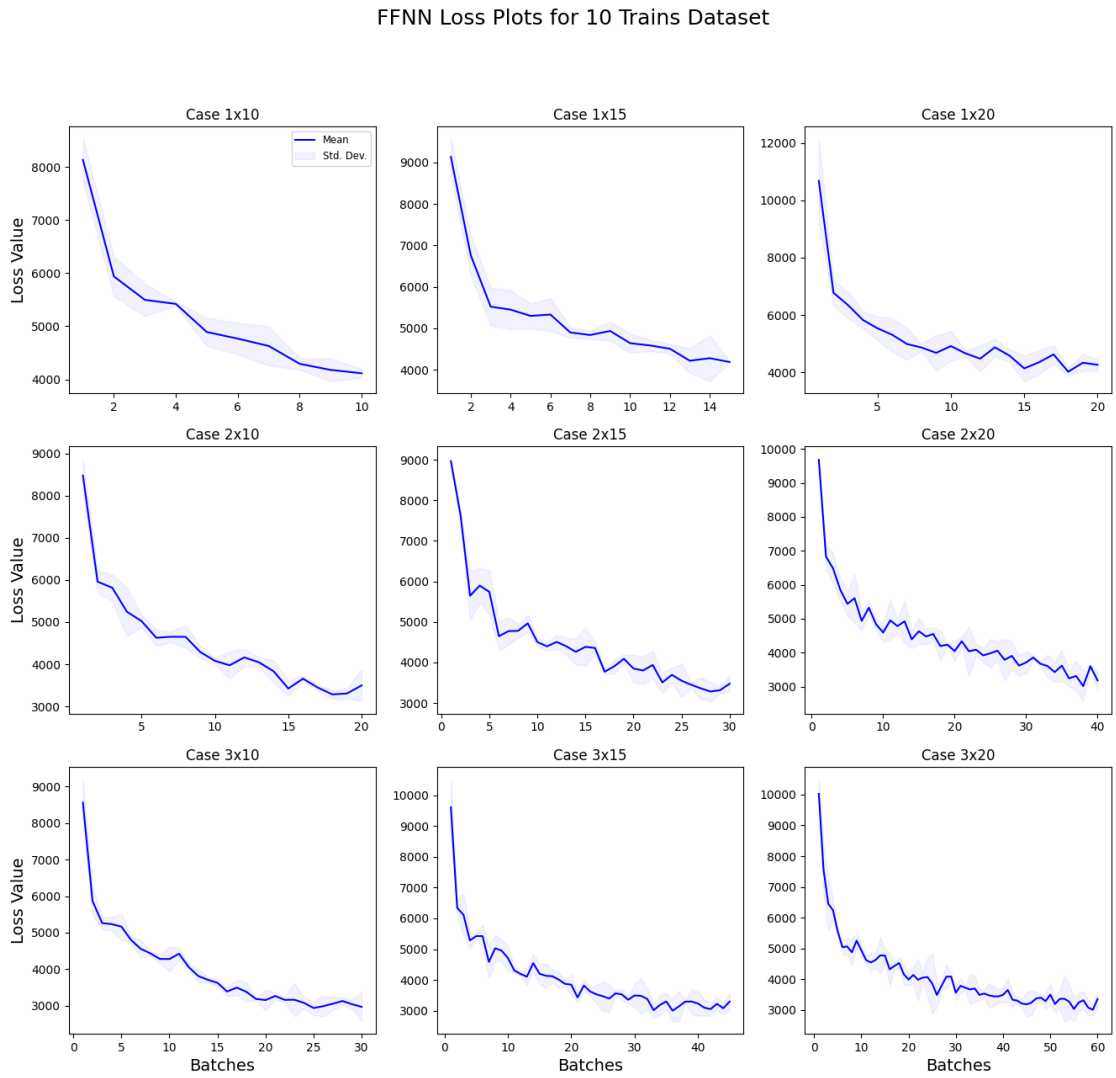

Figure G.13: Loss Results for the FFNN in the 10 Trains Dataset

FFNN Accuracy Results

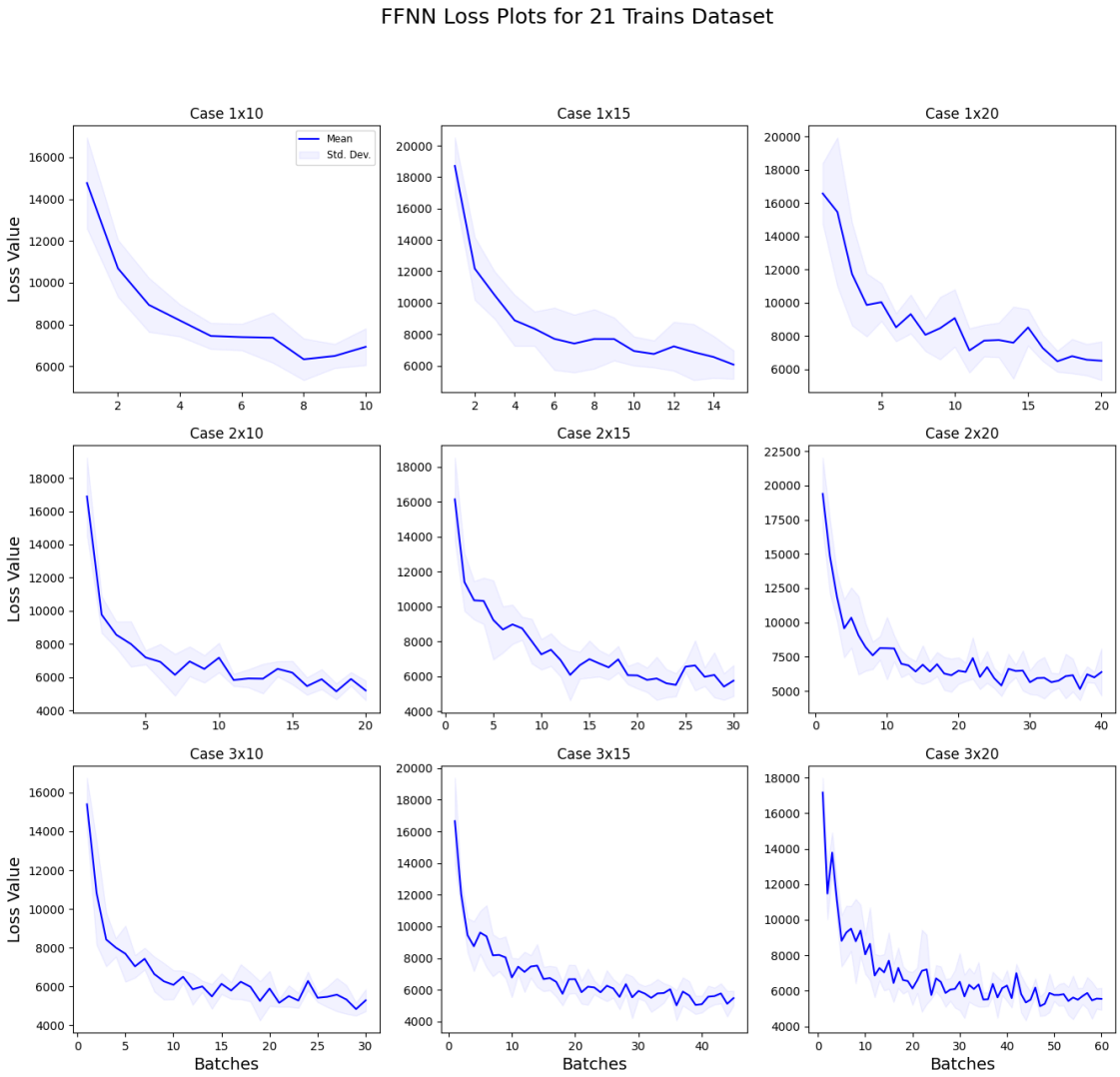

Figure G.14: Loss Results for the FFNN in the 21 Trains Dataset

Figure G.15: Accuracy Results for the FFNN in the 10 Trains Dataset

Figure G.16: Accuracy Results for the FFNN in the 21 Trains Dataset

G.2 Model versus ClientSolver

G.2.1 Time Comparison

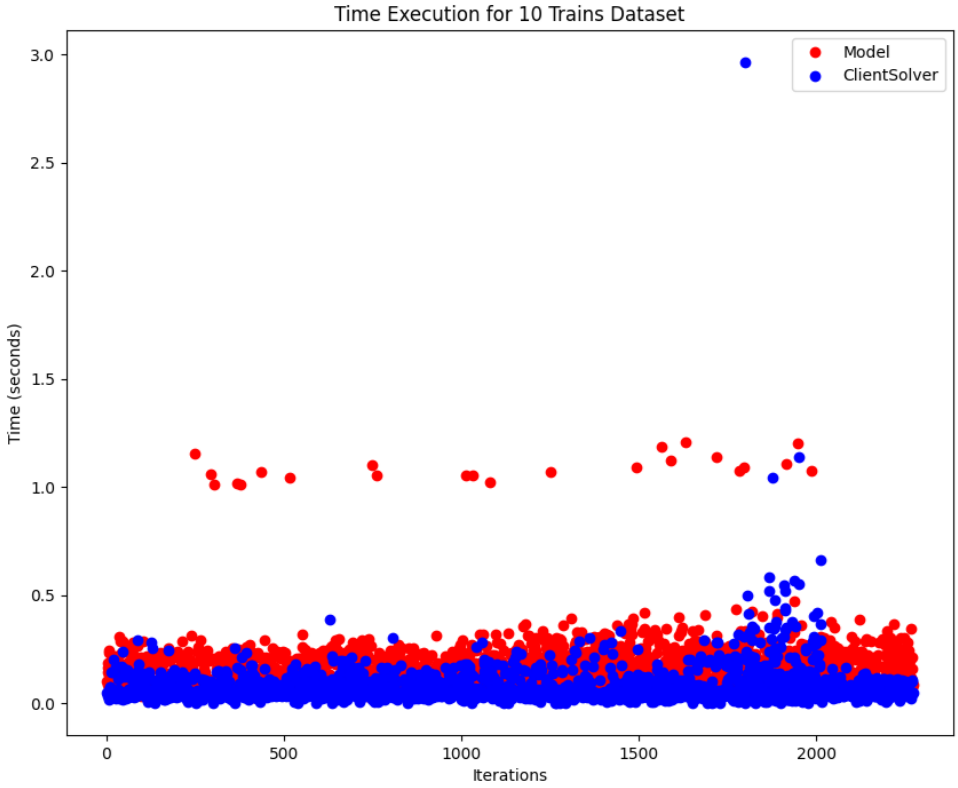

Figure G.17: Scatter Plot with the time executions for 10 Trains Dataset

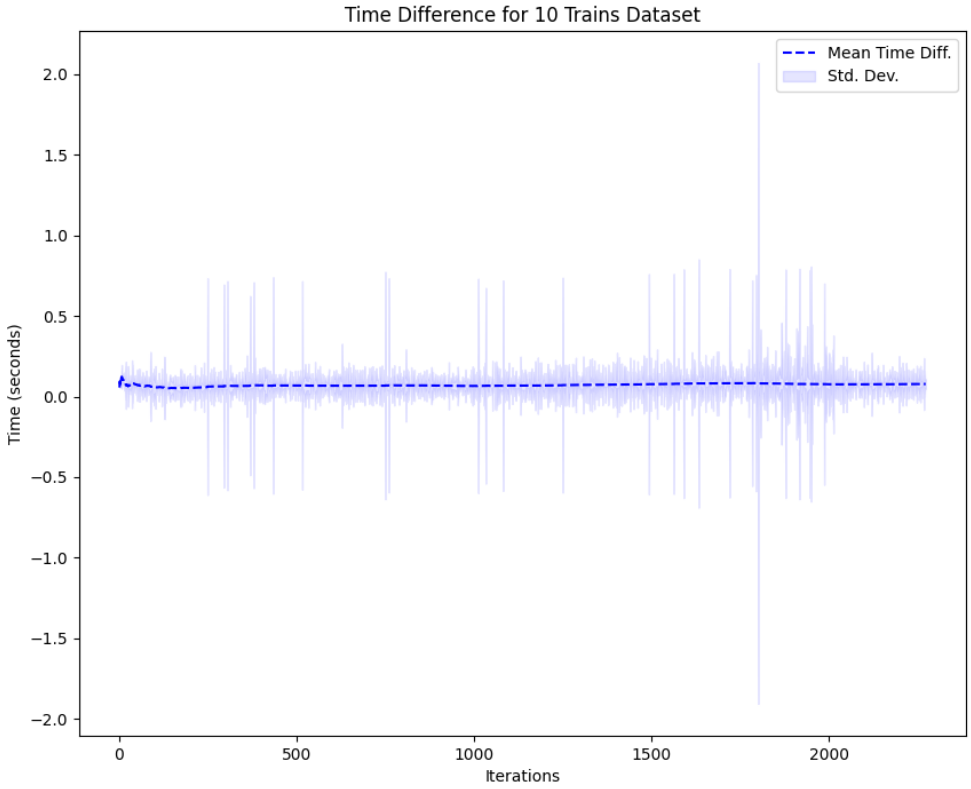

Figure G.18: Time difference between executions for 10 Trains Dataset

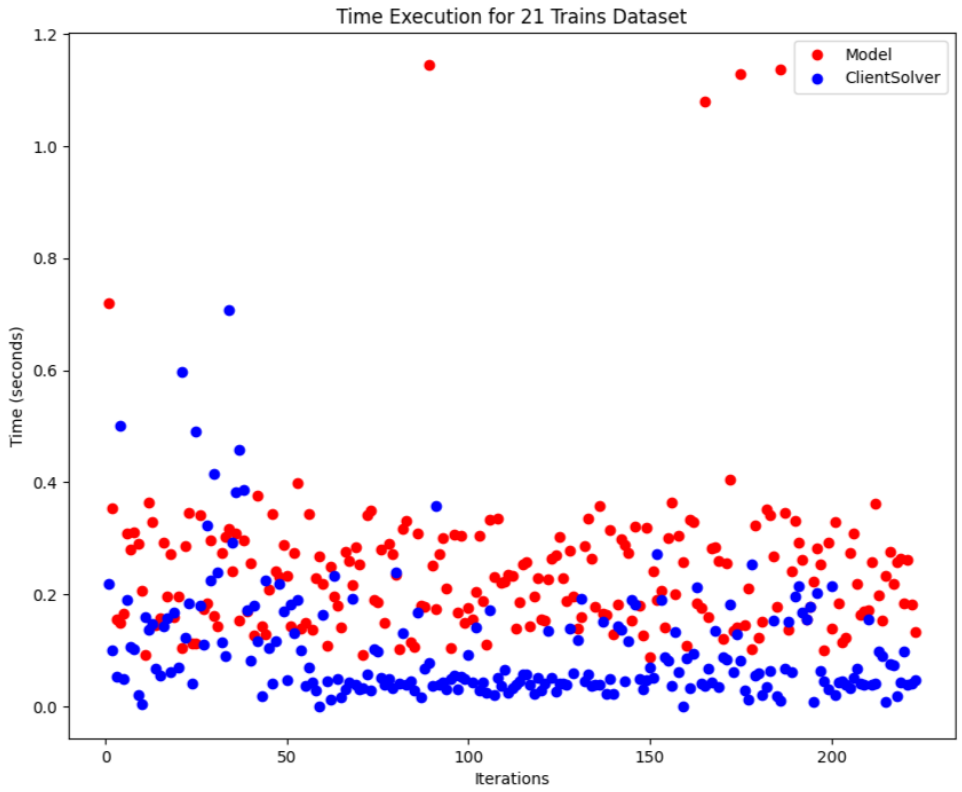

Figure G.19: Scatter Plot with the time executions for 21 Trains Dataset

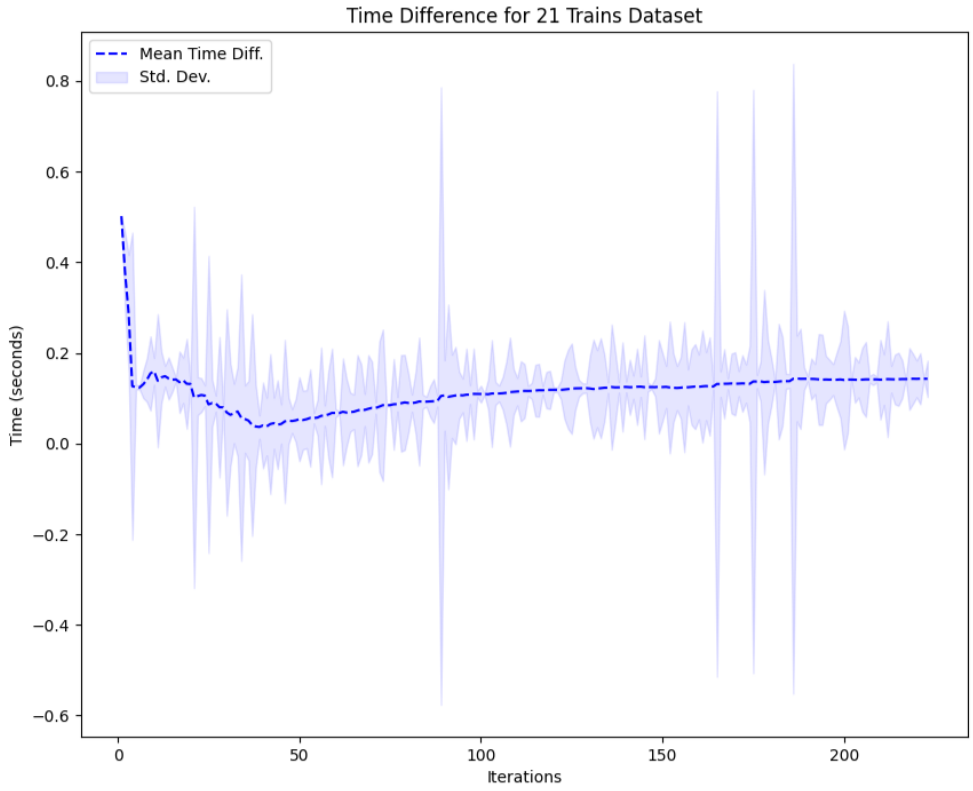

Figure G.20: Time difference between executions for 21 Trains Dataset

G.2.2 Solution Value Comparison

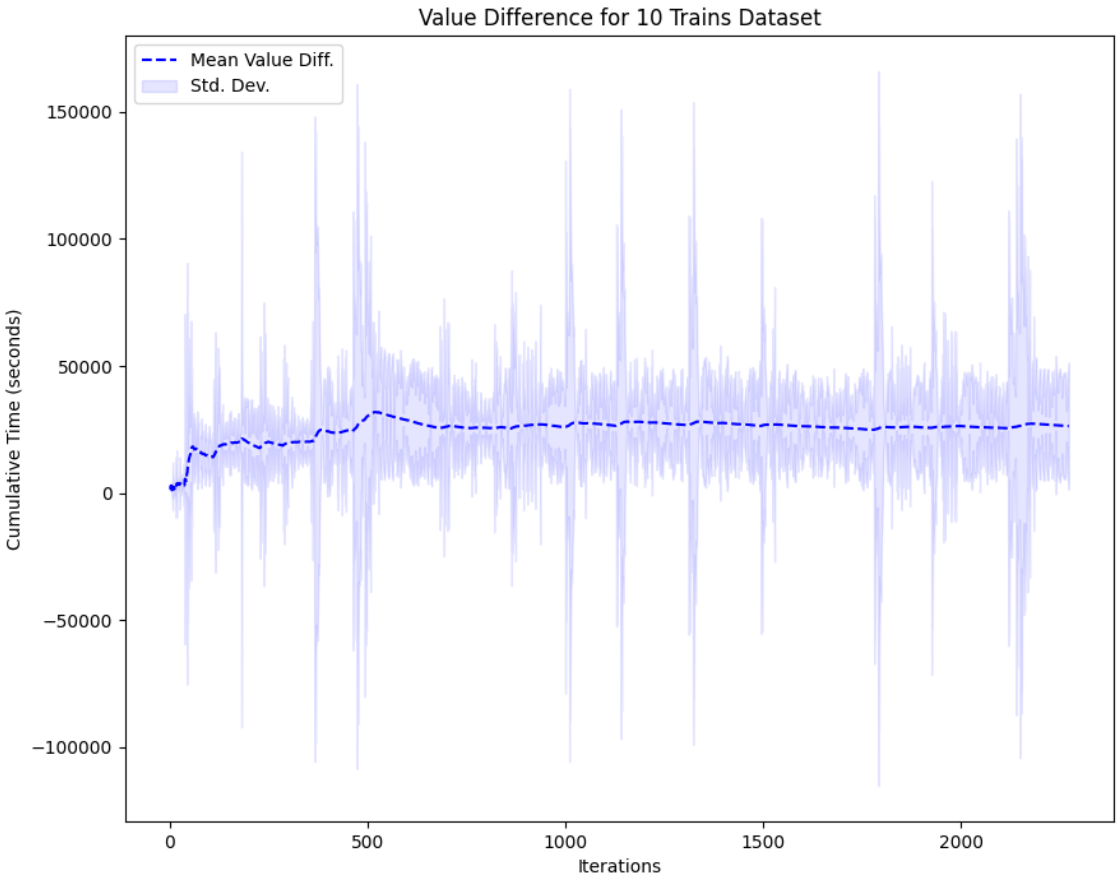

Figure G.21: Value difference between solutions for 10 Trains Dataset

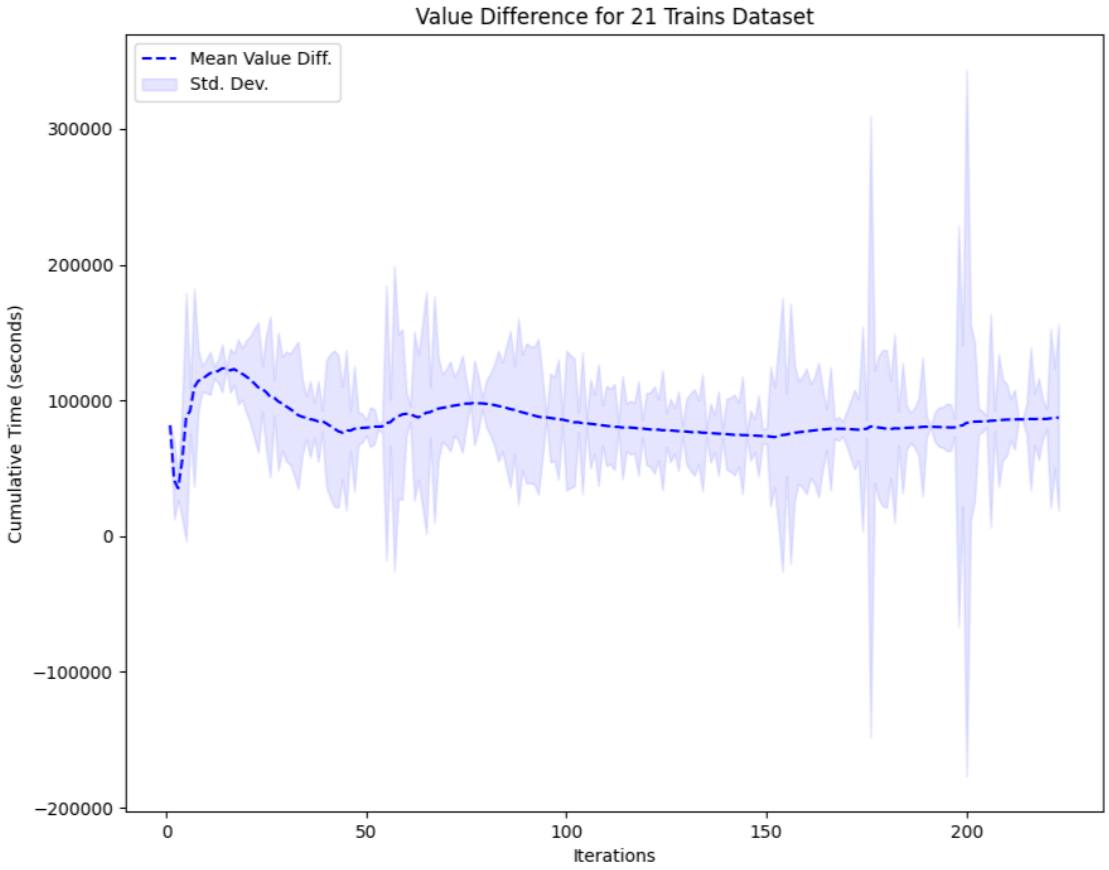

Figure G.22: Value difference between solutions for 21 Trains Dataset