AI GENERATED TRACK OPTIMIZATION FOR INDIAN RAILWAY

PROJECT REPORT

submitted by

ADRIN K JIGY

PTA20CS004

NANDAJA P

PTA20CS041

SONU S

PTA20CS055

TIJO K ABRAHAM

PTA20CS064

to

the APJ Abdul Kalam Technological University
in partial fulfilment of the requirements for the award of the
Degree of Bachelor of Technology in Computer Science and Engineering

Department of Computer Science & Engineering

College of Engineering Kallooppara

Pathanamthitta.

May 2024

DECLARATION

We undersigned hereby declare that the project report **AI GENERATED TRACK OPTIMIZATION FOR INDIAN RAILWAY** submitted for partial fulfilment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of **Assistant Professor Mrs. Shilpa Radhakrishnan.** This submission represents our ideas in our own words and ideas or words of others have been included where we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in our submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the university and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other university.

Place: Kallooppara

Date:29/04/2024

ADRIN K JIGY

NANDAJA P

SONU S

TIJO K ABRAHAM

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

COLLEGE OF ENGINEERING KALLOOPPARA

PATHANAMTHITTA- 689603

CERTIFICATE

This is to certify that the project design report entitled **AI GENERATED TRACK OPTIMIZATION FOR INDIAN RAILWAY** submitted by **Nandaja P** (PTA20CS041), **Sonu S** (PTA20CS055), **Adrin K Jigy** (PTA20CS004), **Tijo K Abraham** (PTA20CS064) in partial fulfillment with the requirements of the award of the Degree of Bachelor of Technology in Computer Science and Engineering of APJ Abdul Kalam Technological University is a bonafide work carried out by them under our guidance and supervision. The report in any form has not been submitted to any other University or Institute for any purpose.

Co-Coordinator	**Guide**	**Head of the Department & Coordinator**
Ms. Nimitha Mary Mohan	**Mrs. Shilpa Radhakrishnan**	**Dr. Renu George**
Assistant Professor, Department of Computer Science & Engineering	Assistant Professor, Department of Computer Science & Engineering	Assistant Professor, Department of Computer Science & Engineering

ACKNOWLEDGEMENT

We take this opportunity to express our deepest sense of gratitude and sincere thanks to everyone who helped me to complete this work successfully. We express our sincere thanks to **Dr. Nisha Kuruvilla**, Principal for the constant support and help. We also thank **Dr. Renu George**, Head of the Department Computer Science and Engineering, College of Engineering Kallooppara for providing me with all the necessary facilities and support. We would like to express our sincere gratitude to **Ms. Nimitha Mary Mohan** and **Mrs. Shilpa Radhakrishnan**, department of Computer Science and Engineering, College of Engineering Kallooppara for their support and cooperation. We would like to place on record our sincere gratitude to my project guide, **Mrs**. **Shilpa Radhakrishnan**, Assistant Professor, Computer Science and Engineering, College of Engineering Kallooppara for the guidance and mentorship throughout the course. Finally, we thank our families, and friends who contributed to the successful fulfilment of this project work.

ADRIN K JIGY
NANDAJA P
SONU S
TIJO K ABRAHAM

ABSTRACT

The Indian railway system faces critical challenges including overcrowding, financial losses, delays, and safety concerns due to inadequate track optimization. This paper proposes an innovative solution to address these issues by introducing an AI-enabled track optimization system utilizing genetic algorithms. The system aims to enhance efficiency, safety, and service quality within the Indian railway network. The approach involves encoding train schedules and routes into genetic algorithm populations, defining a fitness function to evaluate solutions based on objectives like minimizing delays, optimizing throughput, and ensuring safety. Genetic operators such as selection, crossover, and mutation are applied to generate improved solutions. The system simulates and evaluates performance, iterates for continuous improvement, and eventually validates and implements optimized solutions. Furthermore, the implementation strategy involves harnessing AI to enable real-time decision-making, utilizing advanced data analytics to predict and mitigate disruptions, and integrating the AI-enabled system seamlessly into existing railway operations. Anticipated benefits encompass enhanced punctuality, reduced overcrowding, increased efficiency, cost savings, improved safety, optimal resource utilization, real-time adaptability, and strategic planning insights. The proposed AI-enabled track optimization system not only presents a transformative solution but also outlines a comprehensive implementation roadmap that has the potential to revolutionize the Indian railway network, positioning it as a global leader in terms of efficiency, safety, and passenger experience.

CONTENTS

LIST OF FIGURES

ABBREVATIONS

- DEAP: Distributed evolutionary algorithm in python
- GA: Genetic Algorithm
- ML: Machine Learning
- AI: Artificial Intelligence
- NEAT: Neuro Evolution of Augmenting Topologies
- ANN: Artificial Neural Network

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

The envisioned AI-enabled track optimization system represents a groundbreaking solution poised to revolutionize the operational landscape of the Indian railway system. Currently grappling with pervasive issues of overcrowding, financial losses, delays, and safety concerns, the Indian Railways finds itself at a critical juncture where a transformative intervention is imperative. The absence of advanced track optimization technologies, coupled with organizational errors, has not only resulted in unfavourable train timetables and operational inefficiencies but has also led to catastrophic instances of train collisions. In response to this pressing challenge, our proposed system leverages the power of artificial intelligence, specifically employing genetic algorithms, to meticulously optimize train tracks, mitigate delays, streamline schedules, and avert potential collisions. At the core of this technological intervention is a systematic approach that encompasses key components such as encoding, fitness function development, genetic operators implementation, collision prevention mechanisms, simulation, and ongoing evaluation. The encoding phase ensures that intricate details of train schedules, routes, and relevant factors are represented as chromosomes or individuals within the genetic algorithm population. Subsequently, a carefully crafted fitness function evaluates the quality of solutions against defined objectives, including minimizing delays, maximizing train throughput, optimizing travel time, and ensuring safety by preventing collisions. The implementation of genetic operators, encompassing selection, crossover, and mutation, facilitates the generation of new and potentially improved candidate solutions.

Crucially, the system incorporates sophisticated collision prevention mechanisms, employing efficient algorithms and data structures to swiftly identify and prevent potential train collisions. Simulation and evaluation processes allow for a comprehensive assessment of the system's performance, taking into account factors such as train speeds, track capacities, station dwell times, and signalling systems. Through iterative genetic algorithm

processes, the system progressively converges towards optimal solutions, systematically reducing delays and enhancing overall efficiency.

The validation and implementation phase ensures the real-world viability of the optimized solutions, with a focus on seamless integration into the existing railway system while minimizing disruption. Continuous monitoring and adaptation mechanisms enable the system to evolve dynamically, collecting feedback, updating genetic algorithms with new data, and adjusting parameters to adapt to changing conditions, thereby ensuring a sustained trajectory of efficiency improvement. Anticipated benefits include significant improvements in punctuality, reduced overcrowding through optimized train routes, enhanced efficiency for both freight and passenger services, and cost optimization leading to financial savings for the Indian Railways. The transformative impact of this AI-enabled track optimization system extends beyond mere operational improvements; it envisions the Indian Railways not only as the largest but also as the preeminent railway system globally, setting new benchmarks in reliability, safety, and efficiency.

1.2 EXISTING SYSTEM

Let's consider the characteristics of the existing railway system before the implementation of this advanced system.

Manual Scheduling and Optimization:

The current system relies on manual scheduling and optimization processes, which are prone to human errors and does not efficiently handle the complexities of a vast railway network.

Limited Technology Integration:

There is a lack of advanced technologies, such as artificial intelligence and genetic algorithms, in the existing railway system. This absence contributes to inefficiencies, delays, and challenges in optimizing train tracks effectively.

Safety Concerns:

The existing system faces safety concerns, as demonstrated by the occurrence of catastrophic instances and train collisions. Safety measures rely on conventional signalling systems, and incidents have highlighted the need for a more robust approach.

Financial Losses and Overcrowding:

Financial losses are incurred due to inefficiencies in the system, including delays, suboptimal schedules, and inadequate capacity management. Overcrowding is a significant issue, affecting passenger comfort, safety, and overall system efficiency.

Inefficient Collision Prevention:

Collision prevention mechanisms in the existing system may not be as efficient, leading to a higher risk of accidents. The manual nature of the current processes makes it challenging to respond in real-time to potential collision scenarios.

Limited Data-Driven Decision-Making:

Decision-making in the existing system is primarily based on historical data and manual analyses. Real-time data utilization for dynamic decision-making is limited, resulting in suboptimal operational performance.

Infrastructure Limitations:

The existing infrastructure may have limitations in handling the increasing demands of passenger and freight transportation. Station capacities and track maintenance schedules might not be effectively optimized.

Challenges:

As the existing system within the Indian railway network, numerous challenges contribute to its inefficiencies, financial losses, and safety concerns. The foremost issue is the pervasive overcrowding that strains the system's capacity, leading to delays, unfavourable train schedules, and compromised safety measures. The absence of a robust track optimization mechanism exacerbates these challenges, resulting in suboptimal utilization of resources and increased collision risks. Technological shortcomings and organizational errors further hinder the system's ability to respond effectively to dynamic operational demands. The lack of advanced technology and optimization strategies has contributed to catastrophic instances of train collisions, highlighting a critical need for a comprehensive solution. Station capacity limitations, inadequately scheduled track maintenance, and adherence to safety standards pose additional hurdles to the seamless functioning of the railway system. These challenges collectively undermine the punctuality, efficiency, and overall performance of the Indian railway network, limiting its potential for economic growth and productivity.

The proposed AI-enabled track optimization system aims to address these challenges by leveraging genetic algorithms to minimize delays, enhance schedules, and prevent collisions, ushering in a transformative era for Indian Railways.

1.3 PROBLEM STATEMENT

The major issue faced by the Indian railway is overcrowding and yet it is running at financial loss. The absence of track optimization is to blame for this issue, which causes delays, unfavourable train timetables, and safety issues. Due to a lack of technology and organizational errors, there have been a number of catastrophic instances with numerous trains colliding in recent years. To address this issue, we propose the implementation of an AI-enabled system that utilizes genetic algorithms to optimize train tracks, minimizing delays, optimizing schedules, and avoiding collisions. The aim is to enhance the effectiveness of the Indian railway network while taking limitations like station capacity, track maintenance schedules, and safety standards into account. Its main goals are optimizing train tracks, minimizing delays, streamlining schedules, and proactively preventing collisions. The overarching goal is to elevate the efficiency and efficacy of the Indian railway network while meticulously considering constraints such as station capacity, track maintenance schedules, and stringent safety standards. By harnessing the power of advanced artificial intelligence and genetic algorithms, our solution aspires not only to transform the Indian railway system into the largest but also the most advanced and safest railway network globally. Through the prevention of delays, meticulous timetable optimization, and a steadfast commitment to safety, this system promises substantial benefits for both the Indian Railways and its passengers, heralding a new era of punctuality, reduced overcrowding, enhanced efficiency, and cost optimization.

1.4 OBJECTIVE

As the AI-enabled track optimization system envisioned for the Indian Railway, the primary objective is to revolutionize the current railway operations by addressing the critical issues of overcrowding, financial losses, delays, and safety concerns. The overarching goal is to enhance the efficiency of the Indian railway network while adhering to constraints such as station capacity, track maintenance schedules, and safety standards. The proposed system aims to achieve this by employing advanced technologies, particularly genetic algorithms, to optimize train tracks effectively. The key objectives include the development of a robust encoding scheme to represent train schedules, routes, and relevant factors as individuals in

the genetic algorithm population. This encoding will capture essential information such as departure times, routes, and speeds, laying the foundation for comprehensive optimization. A crucial aspect of the system is the formulation of a fitness function that meticulously evaluates solutions based on defined objectives. These objectives encompass minimizing delays, maximizing train throughput, optimizing travel time, and ensuring safety by avoiding collisions. The assignment of higher fitness values to solutions aligning with these objectives will drive the evolution of more efficient and secure railway systems.

The genetic operators, including selection, crossover, and mutation, play a pivotal role in generating new candidate solutions. Through these mechanisms, the system favours the survival of the fittest solutions, combining genetic information from successful parents to create potentially improved offspring. The introduction of mutation injects an element of exploration, facilitating the discovery of novel solutions.

To address safety concerns, the incorporation of collision prevention mechanisms is imperative. Efficient algorithms and data structures are employed to identify potential collisions between trains and trigger preventive actions, mitigating the risk of catastrophic incidents. The subsequent simulation and evaluation processes simulate the railway system using generated solutions, considering factors such as train speeds, track capacities, station dwell times, and signalling systems. This comprehensive assessment aims to accurately gauge the impact of different routes and schedules on efficiency, collision avoidance, and overall system performance. Through iterative refinement, the genetic algorithm process converges towards optimal solutions, continually reducing delays and improving efficiency over multiple iterations. The validation and implementation phase involves testing the system with real-world scenarios and historical data, ensuring a seamless integration into the existing railway infrastructure while minimizing disruption.

Continuous monitoring and adaptation are essential components of the system, allowing for real-time adjustments based on performance feedback. Regular updates to the genetic algorithm with new data and parameter adjustments ensure adaptability to changing conditions, contributing to a continuous improvement cycle. The anticipated benefits of this transformative initiative include significant improvements in punctuality, reduced overcrowding, enhanced efficiency for both freight and passenger services, and cost optimization for Indian Railways, ultimately positioning it as not only the biggest but also the best railway system globally.

1.5 SCOPE

The primary focus lies in addressing the critical challenges of overcrowding, financial losses, delays, unfavourable timetables, and safety concerns within the existing railway network. By employing cutting-edge technology, specifically genetic algorithms, the system aims to optimize train tracks, ensuring minimal delays, efficient schedules, and the prevention of collisions. The scope encompasses the intricate encoding of train schedules, routes, and pertinent factors, followed by the development of a sophisticated fitness function that evaluates solutions based on predefined objectives such as punctuality, throughput maximization, travel time optimization, and safety assurance. The incorporation of genetic operators, including selection, crossover, and mutation, introduces a dynamic and adaptive element to generate improved candidate solutions over iterative processes. Furthermore, the system is designed to detect and prevent collisions using efficient algorithms and data structures. Through simulation and evaluation, the impact of different routes and schedules on efficiency, collision avoidance, and overall system performance will be accurately assessed. The validation and implementation phases involve real-world scenarios, historical data, and the seamless integration of optimized routes and schedules into the existing railway infrastructure, ensuring minimal disruption. Continuous monitoring and adaptation are integral, allowing the system to evolve, refine, and adapt to changing conditions through regular updates of the genetic algorithm with new data and parameter adjustments. The expected benefits, ranging from punctuality improvements to cost optimization, underscore the transformative potential of the system, positioning the Indian Railway as not only the largest but also the most advanced and efficient railway system globally.

CHAPTER 2

LITERATURE SURVEY

1. **H. J. Kaleybar, M. Davoodi, M. Brenna and D. Zaninelli, "Applications of Genetic Algorithm and Its Variants in Rail Vehicle Systems[1]**:

   Railway systems are time-varying and complex systems with nonlinear behaviours that require effective optimization techniques to achieve optimal performance. Evolutionary algorithms methods have emerged as a popular optimization technique in recent years due to their ability to handle complex, multi-objective issues of such systems. In this context, genetic algorithm (GA) as one of the powerful optimization techniques has been extensively used in the railway sector, and applied to various problems such as scheduling, routing, forecasting, design, maintenance, and allocation. This paper presents a review of the applications of GAs and their variants in the railway domain together with bibliometric analysis. The paper covers highly cited and recent studies that have employed GAs in the railway sector and discuss the challenges and opportunities of using GAs in railway optimization problems. Meanwhile, the most popular hybrid GAs as the combination of GA and other evolutionary algorithms methods such as particle swarm optimization (PSO), ant colony optimization (ACO), neural network (NN), fuzzy-logic control, etc with their dedicated application in the railway domain are discussed too.

2. **P. Wang and R. M. P. Goverde, "Train trajectory optimization of opposite trains on single-track railway lines,"[2]:**

   This paper studies the train trajectory optimization problem of two opposite trains on single-track railway lines. A two-train trajectory optimization problem is formulated using the multiple-phase optimal control model, where the timetable as well as train performance parameters, track gradients, curves, speed limits and energy-saving requirements are taken into account. Multiple-tracks at stations are reflected in the model, so that trains may use different tracks at stations. A meeting constraint is developed to avoid head-on conflicts between opposite trains on open tracks. The trajectories of the two opposite trains are optimized simultaneously by a pseudospectral method with minimizing energy consumption as the objective function. In addition, an optimization method is developed for energy-efficient time-distance paths for two opposite trains based on the two-train trajectory optimization model. The method is applied in case studies of two opposite trains running on

a Dutch single-track railway corridor. The results show that our method is able to produce optimal trajectories and time-distance paths for both trains within a short time.

3. **M. T. Lazarescu and P. Poolad, "Asynchronous Resilient Wireless Sensor Network for Train Integrity Monitoring,"[3]:**

   To increase railway use efficiency, the European Railway Traffic Management System (ERTMS) Level 3 requires all trains to constantly and reliably self-monitor and report their integrity and track position without infrastructure support. Timely train separation detection is challenging, especially for long freight trains without electrical power on cars. Data fusion of multiple monitoring techniques is currently investigated, including distributed integrity sensing of all train couplings. We propose a wireless sensor network (WSN) topology, communication protocol, application, and sensor nodes prototypes designed for low-power timely train integrity (TI) reporting in unreliable conditions, like intermittent node operation and network association (e.g., in low environmental energy harvesting conditions) and unreliable radio links. Each train coupling is redundantly monitored by four sensors, which can help to satisfy the train collision avoidance system (TCAS) and European Committee for Electrotechnical Standardization (CENELEC) software integrity level (SIL) 4 requirements and contribute to the reliability of the asynchronous network with low rejoin overhead. A control centre on the locomotive controls the WSN and receives the reports, helping the integration in railway or Internet-of-Things (IoT) applications. Software simulations of the embedded application code virtually unchanged show that the energy-optimized configurations check a 50-car TI (about 1-km long) in 3.6-s average with 0.1-s standard deviation and that more than 95% of the reports are delivered successfully with up to one-third of communications or up to 15% of the nodes failed. We also report qualitative test results for a 20-node network in different experimental conditions.

4. **N. Bešinović , "Artificial Intelligence in Railway Transport: Taxonomy, Regulations, and Applications[4]:**

   Artificial Intelligence (AI) is becoming pervasive in most engineering domains, and railway transport is no exception. However, due to the plethora of different new terms and meanings associated with them, there is a risk that railway practitioners, as several other categories, will get lost in those ambiguities and fuzzy boundaries, and hence fail to catch the real opportunities and potential of machine learning, artificial vision, and big data analytics, just to name a few of the most promising approaches connected to AI. The scope of this paper is to introduce the basic concepts and possible applications of AI to railway academics and practitioners. To that aim, this paper presents a structured taxonomy to guide researchers

and practitioners to understand AI techniques, research fields, disciplines, and applications, both in general terms and in close connection with railway applications such as autonomous driving, maintenance, and traffic management. The important aspects of ethics and explainability of AI in railways are also introduced. The connection between AI concepts and railway subdomains has been supported by relevant research addressing existing and planned applications in order to provide some pointers to promising directions.

5. **S. D. Immanuel and U. K. Chakraborty, "Genetic Algorithm: An Approach on Optimization,"[5]:**

   Solutions for both constrained and unconstrained problems of optimization pose a challenge from the past till date. The genetic algorithm is a technique for solving such optimization problems based on biological laws of evolution particularly natural selection. In simple terms, a genetic algorithm is a successor to the traditional evolutionary algorithm where at each step it will select random solutions from the present population and labels those as parents and uses them to reproduce to the next generation as children with a series of biological operations, namely reproduction, selection, crossover and mutation.

CHAPTER 3

SYSTEM ANALYSIS

3.1 HARDWARE REQUIREMENTS

Computer or Device: a PC or a Laptop that meets the minimum requirement is essential

Processor: Intel Core i5 or higher, AMD Ryzen 5 or above

RAM: 8 GB or higher recommended

Storage: At least 100GB of available storage space

Graphics Card: Any modern graphics card with OpenGL support

Internet Connectivity: Required for accessing online resources and datasets

3.2 SOFTWARE REQUIREMENTS

Operating System: Windows 10 or above, macOS Catalina or above, Ubuntu 20.04 or above

Web Browsers: Google Chrome, Mozilla Firefox, Safari, Microsoft Edge

Development Environment: Visual Studio Code

Programming Language: Python 3.7 or higher

Frameworks/Libraries: Django, Pygame, NEAT Python, DEAP, Scikit Learn

Database: SQLite

Other Dependencies: NumPy, Pandas, Matplotlib, Seaborn, Sklearn's Random Forest Regressor, Sklearn's Gradient Boosting Regressor

3.2.1 Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 1980s as a successor to the ABC

programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages. For this project ,we use python 3.11.7 version.

3.2.2 NumPy, Pandas and Sklearn

NumPy and pandas are two powerful libraries in the Python ecosystem that are widely used for data manipulation and analysis. NumPy, short for Numerical Python, is a fundamental package for scientific computing with Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. NumPy is essential for tasks involving numerical operations and is the foundation for many other scientific computing libraries. On the other hand, pandas is a high-level data manipulation library built on top of NumPy. It provides easy-to-use data structures such as DataFrame and Series, which are ideal for handling structured data. With pandas, you can efficiently manipulate, clean, and analyse data, making it a crucial tool for data scientists and analysts. Pandas excels in tasks like data cleaning, exploration, and transformation, making it an essential part of the data science toolkit. In summary, NumPy is the backbone for numerical operations in Python, while pandas adds a layer of abstraction for convenient and efficient data manipulation and analysis, making them both indispensable for data science and scientific computing tasks.

scikit-learn (sklearn) is a versatile and widely-used machine learning library in Python, offering a range of tools for predictive data analysis. It provides efficient implementations of a large variety of machine learning algorithms, making it accessible for both beginners and experts in the field.

One of the key advantages of **scikit-learn** is its ease of use. It provides a consistent and simple interface for working with different machine learning models, making it easy to experiment with various algorithms and techniques.

scikit-learn includes a wide range of functionalities, including:

Data preprocessing: scikit-learn provides tools for preprocessing data, including handling missing values, scaling features, and encoding categorical variables.

Supervised learning: It offers a variety of supervised learning algorithms for classification, regression, and time series forecasting. This includes algorithms like Support Vector Machines (SVM), Random Forests, Gradient Boosting, and more.

Unsupervised learning: scikit-learn also includes algorithms for unsupervised learning tasks such as clustering, dimensionality reduction, and outlier detection. Algorithms like KMeans, PCA, and Isolation Forest are part of its toolkit.

Model evaluation: It provides tools for evaluating the performance of machine learning models, including metrics like accuracy, precision, recall, F1-score, and more. It also supports techniques like cross-validation for robust model evaluation.

Hyperparameter tuning: scikit-learn includes utilities for hyperparameter tuning, such as GridSearchCV and RandomizedSearchCV, which help in finding the best hyperparameters for a given model.

3.2.3 Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It's designed to help developers build web applications quickly and with less code. Here are some key features and concepts of Django:

MTV Architecture: Django follows the Model-Template-View (MTV) architecture pattern.

- Model: Defines the data structure. Django provides an ORM (Object-Relational Mapping) to interact with the database without writing SQL queries.
- Template: Manages the presentation layer, allowing the developer to build HTML templates with Django's template language.
- View: Controls the logic and handles user requests. Views are Python functions or classes that receive web requests and return web responses.

Admin Interface: Django provides a built-in admin interface that allows developers and administrators to manage site content without directly interacting with the database.

URL Routing: Django uses a URL dispatcher to direct incoming web requests to the appropriate view based on the request URL.

Form Handling: Django simplifies form handling and validation. It provides tools to create HTML forms, validate user input, and process form data.

Security Features: Django includes built-in protection against common web security threats, such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).

Authentication and Authorization: Django provides a robust authentication system for user authentication and a flexible authorization system for controlling access to different parts of the application.

Internationalization and Localization: Django supports internationalization (i18n) and localization (l10n) features, allowing developers to create applications that support multiple languages and regions.

Middleware: Middleware is a framework of hooks into Django's request/response processing. It's a lightweight, low-level plugin system for globally altering Django's input or output.

Testing Framework: Django provides a testing framework to create and run tests to ensure your application works as expected.

Template Engine: Django includes its own template engine to separate the design from Python code. Templates are HTML files that allow Python-like expressions to generate dynamic content.

Overall, Django's goal is to simplify the creation of complex, database-driven websites by emphasizing reusability, modularity, and the principle of don't repeat yourself (DRY)

3.2.4 NEAT and Artificial Neural Networks (ANNs)

NEAT is an evolutionary algorithm used to train artificial neural networks. It works by creating a population of neural networks with varying structures and connections. These networks are then evaluated based on their performance in a specific task (in this case, controlling the train). The networks with the best performance are then used to breed a new generation, with the goal of continually improving the population's overall performance. Artificial Neural Networks (ANNs) are computational models inspired by the biological neural networks of the human brain. They consist of interconnected nodes (neurons) organized into layers. Each neuron receives input signals, processes them using an activation function, and passes the output to other neurons. ANNs are capable of learning complex patterns in data and are used in various applications, including image recognition, natural language processing, and robotics.

NEAT employs a genetic algorithm to evolve neural networks. It starts with a population of simple neural networks and evolves them over generations. During each generation, NEAT evaluates the performance of each network using a fitness function and selects the best-performing networks for reproduction. Through mutation and crossover operations, NEAT creates new offspring with modified structures, allowing for the exploration of different network topologies.

The key advantage of NEAT is its ability to handle the complexity of evolving neural network structures. By starting with simple networks and gradually adding complexity, NEAT can discover optimal architectures for a given task without the need for manual design. This makes NEAT a powerful tool for developing sophisticated neural network models that can solve complex problems effectively.

3.2.5 SQLite

SQLite is a lightweight, serverless, self-contained SQL database engine that is used by default in Django for development purposes. It is a popular choice for small to medium-sized applications due to its simplicity and ease of setup. SQLite stores data in a single file, making it portable and convenient for development and testing. However, it may not be suitable for high-traffic or large-scale applications due to its limited concurrency support compared to other database engines. In Django, SQLite is easy to configure and use, allowing developers to quickly prototype applications without the need for a separate database server.

3.2.6 Internet Connectivity:

Broadband Connection: Recommended for faster data transfer and smoother browsing experience

Mobile Data: Can be used as an alternative, but may result in slower loading times and limited data caps

CHAPTER 4

METHODOLOGY

4.1 PROPOSED SYSTEM

AI-enabled system that utilizes genetic algorithms to optimize train tracks, minimizing delays, optimizing schedules, and avoiding collisions. The aim is to enhance the effectiveness of the Indian railway network while taking limitations like station capacity, track maintenance schedules, and safety standards into account.

Solution Approach:

1. Encoding

○ Represent train schedules, routes, and other relevant factors as chromosomes or individuals in the genetic algorithm population.

○ Design an encoding scheme that captures the necessary information for each train's departure time, route, and speed.

2. Fitness Function

○ Develop a fitness function that evaluates the quality of each solution based on defined objectives.

○ Consider factors such as minimizing delays, maximizing train throughput, optimizing travel time, and ensuring safety by avoiding collisions.

○ Assign higher fitness values to solutions that align with these objectives.

3. Genetic Operators

○ Implement genetic operators, including selection, crossover, and mutation, to generate new candidate solutions.

○ Apply selection to favour individuals with higher fitness values, ensuring the survival of the fittest solutions.

○ Utilize crossover to combine genetic information from two parent solutions, creating offspring solutions with potentially improved fitness.

○ Introduce mutation to explore new solutions by introducing random changes to the encoded variables.

4. Simulation and Evaluation

○ Simulate the railway system using the generated solutions to evaluate their performance.
○ Consider factors such as train speeds, track capacities, station dwell times, and signalling systems.

○ Accurately assess the impact of different routes and schedules on efficiency, collision avoidance, and overall system performance.

5. Iteration and Improvement

○ Iterate the genetic algorithm process by applying genetic operators, evaluating fitness, and generating new solutions.

○ Over multiple iterations, the AI system will converge towards more optimal solutions, continuously reducing delays and improving efficiency.

6. Validation and Implementation

○ Validate the performance of the optimized solutions using real-world scenarios and historical data.

○ Implement the optimized routes and schedules in the actual railway system, ensuring smooth integration and minimal disruption.

7. Continuous Monitoring and Adaptation

○ Monitor the performance of the implemented system and collect feedback to further refine the AI model. Regularly update the genetic algorithm with new data and adjust parameters as needed to adapt to changing conditions and continuously improve efficiency.

CHAPTER 5

SYSTEM DESIGN

5.1 ARCHITECTURE

A distributed evolutionary algorithm in Python, comprises a robust architecture designed to address the complex challenges of minimizing delays, optimizing schedules, and avoiding collisions. The system architecture involves a distributed setup to harness the power of evolutionary algorithms across multiple nodes. The system is divided into several components, each responsible for specific tasks. The encoding module transforms train schedules, routes, and pertinent information into chromosomes or individuals, forming the genetic algorithm population. The fitness evaluation component develops a comprehensive fitness function, considering factors such as minimizing delays, maximizing train throughput, optimizing travel time, and ensuring safety by avoiding collisions. This fitness function assigns higher values to solutions aligning with defined objectives.

The genetic operator module encompasses selection, crossover, and mutation processes, facilitating the generation of new candidate solutions. Selection favours individuals with higher fitness values, ensuring the survival of the fittest solutions. Crossover combines genetic information from parent solutions, creating potentially improved offspring solutions. Mutation introduces random changes to encoded variables, exploring new solution spaces.

The collision prevention mechanism, integrated into the system, employs efficient algorithms and data structures to identify and prevent potential collisions between trains. The simulation and evaluation module simulates the railway system using generated solutions, considering variables such as train speeds, track capacities, station dwell times, and signalling systems. This enables a thorough assessment of different routes and schedules on efficiency, collision avoidance, and overall system performance. The iterative process involves the genetic algorithm evolving over multiple iterations, converging towards more optimal solutions. The validation and implementation component ensures the performance of optimized solutions through real-world scenarios and historical data. These solutions are then seamlessly integrated into the actual railway system, minimizing disruption and

ensuring smooth implementation. Continuous monitoring and adaptation are critical aspects of the architecture. The system monitors performance, collects feedback, and refines the AI model. Regular updates to the genetic algorithm with new data and parameter adjustments adapt to changing conditions, ensuring continuous improvement in efficiency. The distributed nature of the evolutionary algorithm enhances scalability and computational efficiency, contributing to the system's effectiveness in transforming the Indian Railway into a world-class, efficient, and safe transportation network.

5.2 DIAGRAMS

5.2.1 DATA FLOW DIAGRAM LEVEL-0

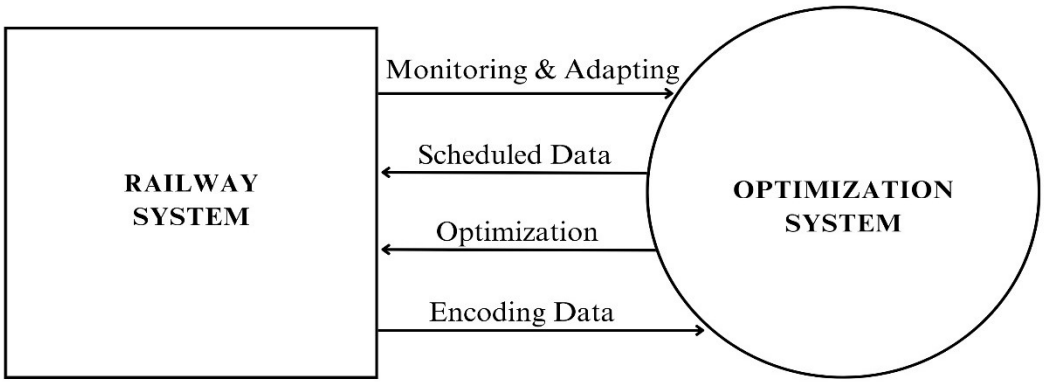

Figure 5. 1 Zeroth level dfd

The optimization system is encoded with time and other railway system data. The AI optimization system generate an optimized solution which includes scheduling and proper evaluation. It also detects collision in the simulated solution. The AI system can monitor the changes in the railway system and can adapt to the changes.

5.2.2 DATA FLOW DIAGRAM LEVEL-1

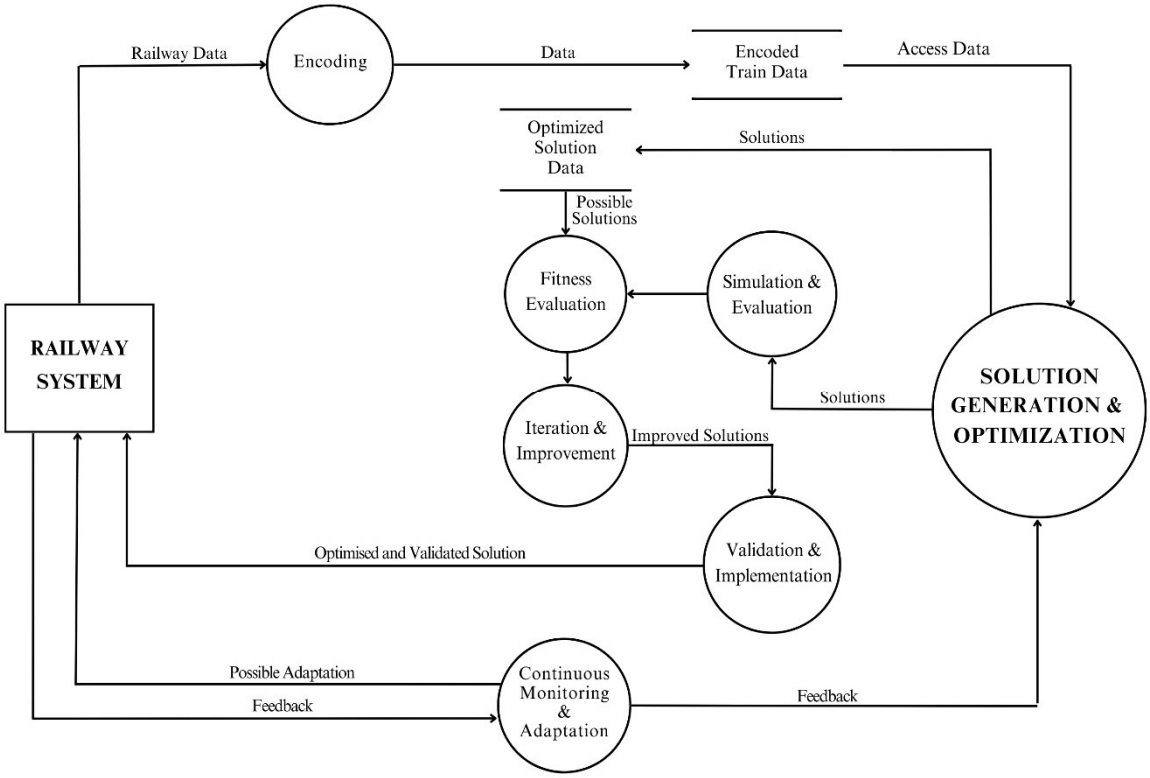

Figure 5. 2 First level dfd

In a first-level data flow diagram (DFD) for the AI-enabled track optimization system for Indian Railways, the main components are;

External Entities:

Indian Railways System: Represents the existing railway infrastructure, including tracks, stations, and trains.

Processes:

Encoding Process: Responsible for representing train schedules, routes, and relevant factors as chromosomes or individuals in the genetic algorithm population.

Solution generation: Generates all possible solutions.

Fitness Evaluation Process: Involves the development of a fitness function that evaluates the quality of each solution based on defined objectives.

Simulation and Evaluation Process: Simulates the railway system using generated solutions to evaluate their performance.

Iteration and Improvement Process: Involves iterating the genetic algorithm process by applying genetic operators, evaluating fitness, and generating new solutions.

Validation and Implementation Process: Validates the performance of optimized solutions using real-world scenarios and historical data, then implements them in the actual railway system.

Continuous Monitoring and Adaptation Process: Monitors the performance of the implemented system, collects feedback, and updates the genetic algorithm with new data.

Data Stores:

Encoded Train Data: Stores the encoded information about train schedules, routes, and speeds.

Optimized Solutions Data: Stores data related to the optimized schedules and routes generated by the genetic algorithm

5.2.3 USE CASE DIAGRAM

UML (Unified Modelling Language) use case diagram for the AI-enabled track optimization system for Indian Railways, various components can be identified to represent the functionalities and interactions of the system. Here's a description of key components:

Actors:

- **Railway system:** Represents the administrative personnel responsible for overseeing and managing the entire railway system.

Use cases:

- **Encode train info**: The railway system provides necessary descriptions about the train schedules and other details.
- **Optimization:** the genetic algorithm system evaluate the encoded data and produce the possible solutions

- **Evaluate Fitness:** The system evaluates the fitness of generated schedules using the defined fitness function, considering objectives like minimizing delays, maximizing throughput, and ensuring safety.
- **Generate solutions:** Based on the evaluation the algorithm produces a optimal solution
- **Simulation and Evaluation:** Simulates the railway system using generated solutions to evaluate performance based on various factors.
- **Validation and Implementation:** Validates the performance of optimized solutions using real-world scenarios and historical data. Implements the optimized routes and schedules in the actual railway system.
- **Iteration and Improvement:** Involves iterating the genetic algorithm process by applying genetic operators, evaluating fitness, and generating new solutions.
- **Monitor and adapt:** Monitors the performance of the implemented system, collects feedback, and updates the genetic algorithm with new data for continuous improvement.

The UML use case diagram in fig 5.2 provides a high-level view of the AI-enabled track optimization system, illustrating the interactions between key actors and the system's major functionalities. It helps in understanding how different components collaborate to achieve the overall goal of enhancing the efficiency and safety of the Indian Railway system.

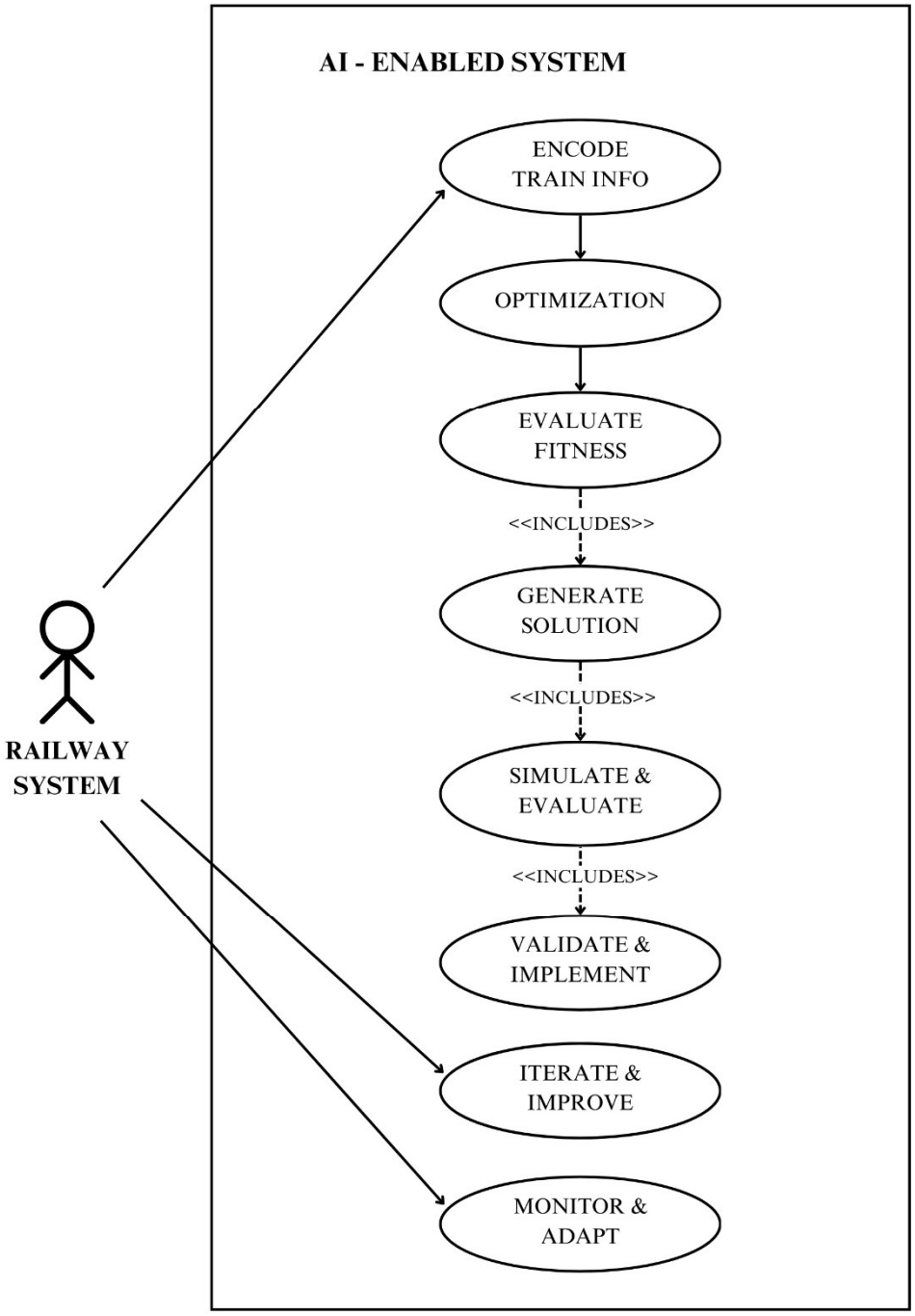

Figure 5. 3 Use case diagram

5.2.4 ACTIVITY DIAGRAM

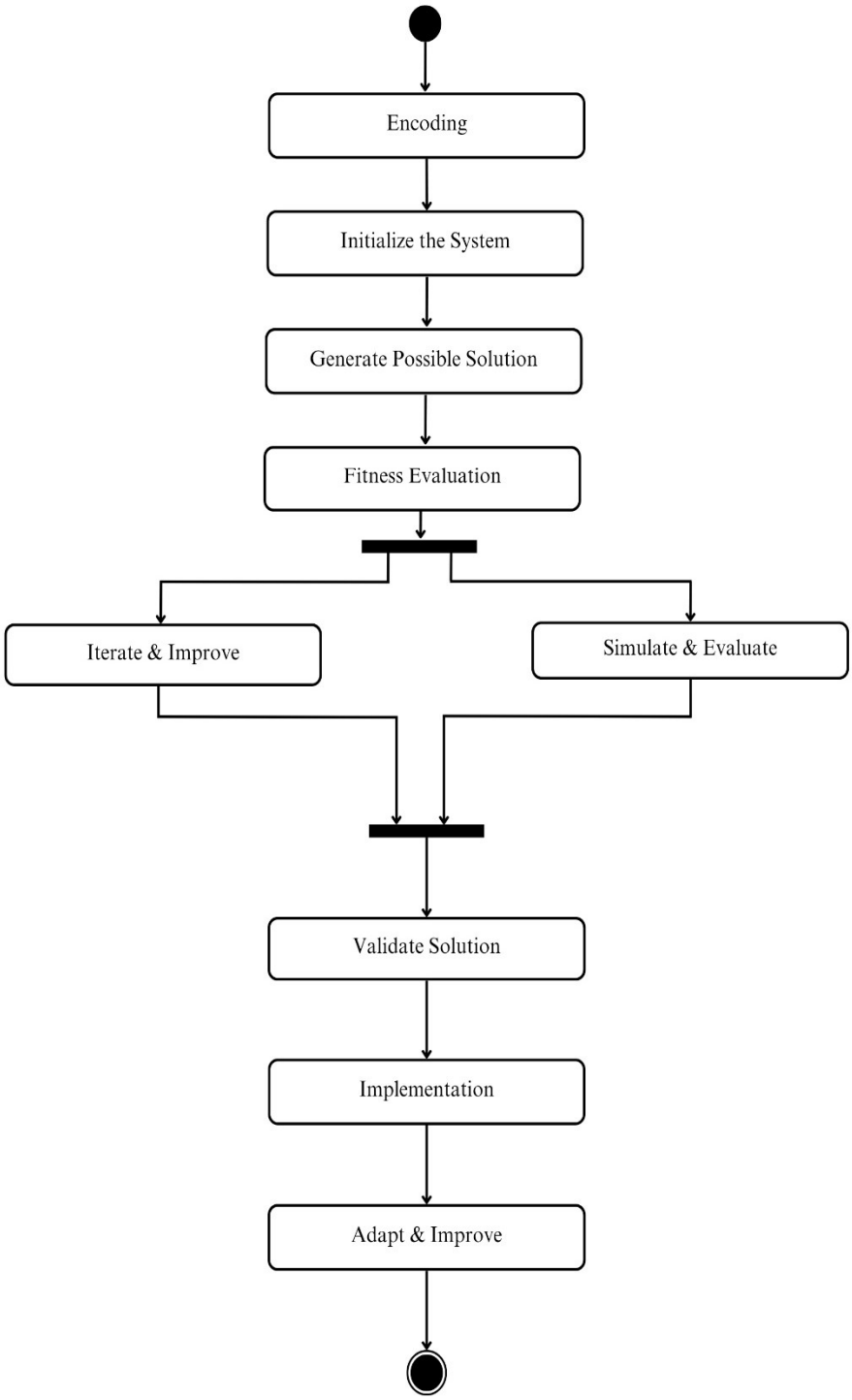

Figure 5. 4 Activity diagram

The UML activity diagram for the proposed AI-enabled track optimization system using genetic algorithms involves representing the flow of activities and interactions within the system. Below is a simplified representation of the key steps in the solution approach:

Encoding: Represents the encoding of train schedules, routes, and relevant factors as chromosomes in the genetic algorithm population

Initialization: Initializes the genetic algorithm process, including the initial population of solutions.

Generate solution: This process generates all possible solutions to the given inputs.

Fitness evaluation: Develop a fitness function that evaluates the quality of each solution based on defined objectives.

Iterate and improve: Iterates the genetic algorithm process by applying genetic operators, evaluating fitness, and generating new solutions. Over multiple iterations, converges towards more optimal solutions.

Simulate and evaluate: Simulates the railway system using generated solutions to evaluate their performance.

Validation: Validates the performance of optimized solutions using real-world scenarios and historical data.

Implementation: implements optimized routes and schedules in the actual railway system, ensuring smooth integration.

Adapt and improve: Regularly updates the genetic algorithm with new data and adjusts parameters for continuous improvement.

CHAPTER 6

SYSTEM IMPEMETATION

This section contains the technical details and implementation aspects of the system.

6.1 TECHNOLOGIES AND TOOLS

The system is divided into 3 modules based on functionality.

A. Train delay prediction using random forest regression
B. Self driving train using NEAT python
C. Train Timetable Optimization Using Genetic Algorithm

6.1.1 TRAIN DELAY PREDICTION USING RANDOM FOREST REGRESSION

The "Train-Delay-Prediction" module focuses on analyzing historic train data to gain insights and develop a predictive model for predicting train delays. By leveraging machine learning algorithms like Random Forest Regressor and Gradient Boosting Regressor, the project aims to predict whether a train will be delayed based on various characteristics. The objective of the project is to perform analysis of the historic train data to gain valuable insights and build a predictive model to predict whether a train will be delayed or not given a set of train characteristics. The objective of the predictive model is to build a model to predict whether a train will be delayed or not based on certain characteristics of the train. Such a model may help both passengers as well as railways to predict future delays and minimize them. The model is trained with a dataset which contains train data including train names, station names, timings and historical data about train delays. The output shows whether the train is delayed or not depends on the time; if the train is late by 15 minutes or more then it is declared as delayed.

Technologies Used:

- Numpy
- Pandas
- Matplotlib
- Seaborn
- Sklearn
- Random Forest Regressor from Sklearn

- Gradient Boosting Regressor from Sklearn

Random Forest Regression Random Forest Regression is a machine learning algorithm used for regression tasks, which involves predicting a continuous value based on input features. It is an ensemble learning method, meaning it combines the predictions of multiple individual models (trees) to improve the overall accuracy and robustness of the model. Here's how the Random Forest Regression algorithm works:

i. Random Forest Overview: A Random Forest is made up of a collection of decision trees. Each tree in the forest is trained independently on a random subset of the data (bootstrap sample) and a random subset of the features. This randomness helps to ensure that each tree in the forest is different from the others.

ii. Decision Trees: Each decision tree in the Random Forest makes a prediction by recursively splitting the data into subsets based on the feature values. The splits are chosen to minimize the variance of the target variable (i.e., reduce the error in prediction). The final prediction of the tree is the average (for regression) of the target values of the training instances in the leaf node.

iii. Combining Predictions: In a Random Forest, the predictions of all the trees are combined to make the final prediction. For regression tasks, this is typically done by averaging the predictions of all the trees in the forest.

In the train delay prediction, Random Forest Regression model is implemented using the RandomForestRegressor class from the sklearn.ensemble module. Here's a breakdown of how it is implemented:

• Loading Data: The loadData function reads the data from CSV files (trains.csv and stations.csv) using pandas. It preprocesses the data by dropping unnecessary columns and encoding categorical variables.

• Preprocessing: The preprocessing function splits the data into features (X) and target (Y) variables. It further splits the data into training and testing sets using train_test_split function from sklearn.model_selection.

• Training the Random Forest Regressor: The rfg function creates an instance of RandomForestRegressor. It fits the model on the training data using the fit method.

• Accepting User Input: The accept_data function takes user input for month, day, scheduled departure time, distance, arrival delay, train code, origin station code, destination station code, and day of the week.

• Making Predictions: The prediction function prepares the input vector for the trained model based on the user input. It then uses the trained RandomForestRegressor model to predict whether the train will be delayed or not based on the input data.

• Displaying Prediction: The main function is the entry point of the application. It uses Streamlit to create a web interface where users can select the machine learning model (in this case, only the Random Forest Regressor is available) and input their data.

The main use of the project is to provide a predictive model for estimating train delays, which can be valuable for both railways and passengers. By analyzing historical train data and considering various factors such as departure time, railway, and station, the model can forecast the likelihood of a train being delayed by 15 minutes or more. This information can help railways to optimize their scheduling and operations, potentially reducing delays and improving overall efficiency. For passengers, knowing the likelihood of a delay can aid in planning travel itineraries and making informed decisions. Overall, the project aims to enhance the efficiency and reliability of train travel by leveraging machine learning algorithms to predict and mitigate train delays.

6.1.2 SELF DRIVING TRAIN USING NEAT PYTHON

This module simulates a self-driving train using Neuro Evolution of Augmenting Topologies (NEAT) in Python. The core logic for the train's behaviour and NEAT implementation is independent of any specific graphics library. This allows for flexibility in designing the visual representation of the train and its environment using any content creation platform.

1. NEAT and Artificial Neural Networks (ANNs)

NEAT is an evolutionary algorithm used to train artificial neural networks. It works by creating a population of neural networks with varying structures and connections. These networks are then evaluated based on their performance in a specific task (in this case, controlling the train). The networks with the best performance are then used to breed a new generation, with the goal of continually improving the population's overall performance.

2. Project Components:

Train: The train object represents the physical entity within the simulation. It has attributes like position, speed, and direction.

Environment: This defines the world where the train operates. It can include elements like tracks, stations, and obstacles. The environment representation needs to be compatible with the input format of the neural network. (Details on this format will be covered in section.

.NEAT Implementation: This core module utilizes the neat-python library to manage the population of neural networks, their training, and selection for breeding.

Fitness Function: This function determines the performance of a specific neural network during evaluation. In this case, a good fitness function might consider factors like staying on track, reaching the destination efficiently, and avoiding obstacles.

3. Input and Output for the Neural Network

The neural network used in this project takes in information about the train's environment and outputs control signals. Here's a breakdown of a possible structure:

Input:

- Train position (relative to the track)
- Distance to the next station
- Presence of obstacles ahead (binary values for different directions)
- Additional environmental features can be included based on your specific simulation needs.

Output:

- Train acceleration/deceleration
- Change in direction (optional, depending on track complexity)

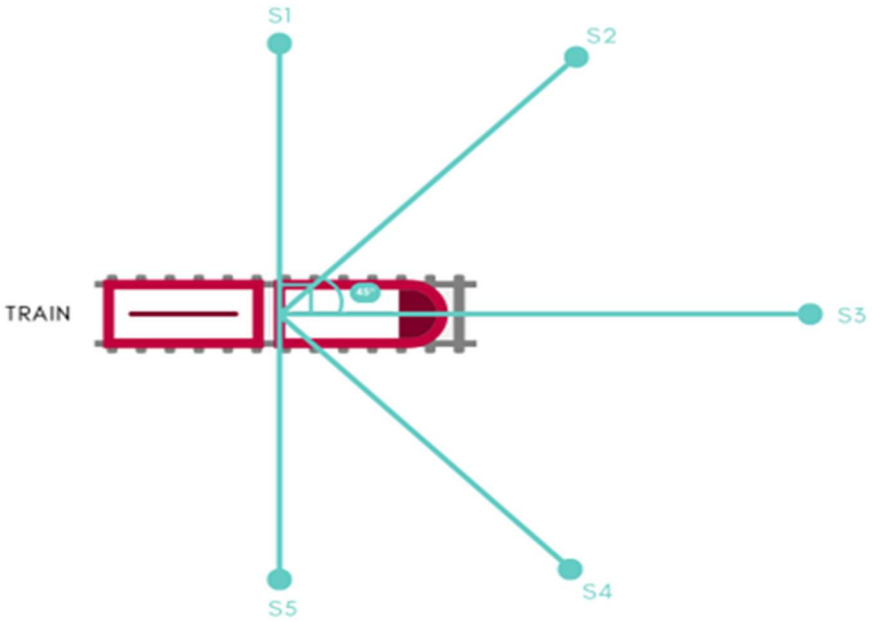

Figure 6. 1 Train model

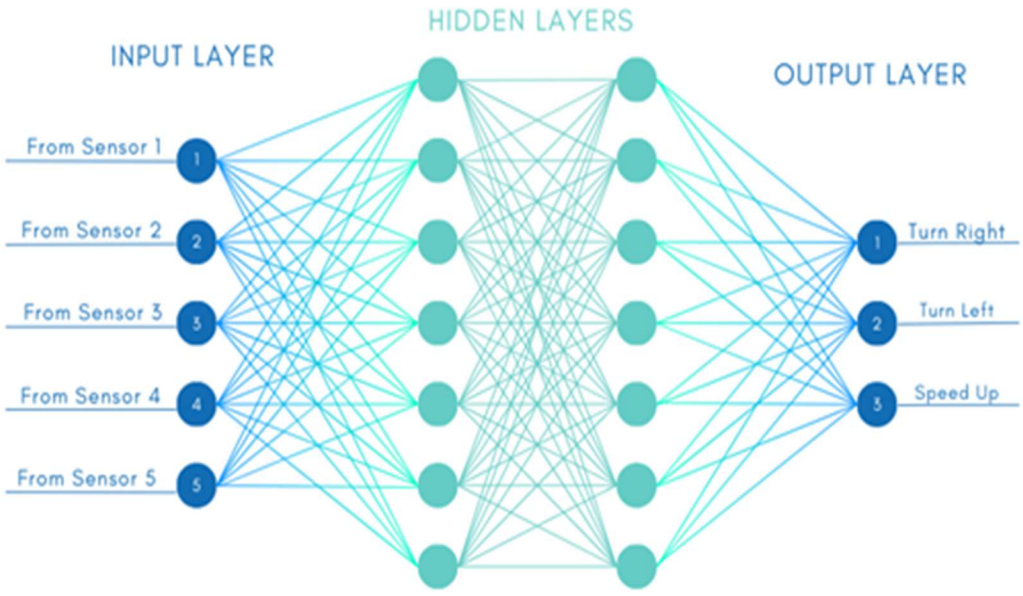

Figure 6. 2 Artificial neural network

4. Training Process

- Initialization: Create a population of neural networks with random topologies.
- Evaluation: For each network in the population: Simulate the train's movement for a certain duration using the network's outputs as control signals. Calculate the fitness score based on the defined fitness function.
- Selection: Choose the top-performing networks (based on fitness) for breeding.
- Breeding: Apply NEAT's genetic algorithms to create a new generation of networks with improved characteristics inherited from the previous generation.
- Repeat: Go back to step 2 and continue training for a set number of generations or until a desired performance level is achieved.

5. Visualization

While the core logic is independent of UI visuals, creating a user interface (UI) to represent the simulation can be beneficial. This UI could be built using any graphics library or framework. It would typically show the train moving on the pre-designed map, potentially highlighting relevant information like the current fitness score or generation number.

This module demonstrates the application of NEAT to train a self-driving train in a simulated environment. The modular design allows for customization of the environment and the neural network's input/output structure. This project serves as a foundation for exploring further complexities in train behavior and environment design.

6.1.3 TRAIN TIMETABLE OPTIMIZATION USING GENETIC ALGORITHM

The train timetable optimization aims to enhance efficiency, safety, and service quality within the Indian railway network. The approach involves encoding train schedules and routes into genetic algorithm populations, defining a fitness function to evaluate solutions based on objectives like minimizing delays, optimizing throughput, and ensuring safety. Genetic operators such as selection, crossover, and mutation are applied to generate improved solutions. The system is implemented using the Django framework.

The key objectives include the development of a robust encoding scheme to represent train schedules, routes, and relevant factors as individuals in the genetic algorithm population. This encoding will capture essential information such as departure times, routes, and speeds, laying the foundation for comprehensive optimization. A crucial aspect of the system is the formulation of a fitness function that meticulously evaluates solutions based on defined objectives. These objectives encompass minimizing delays, maximizing train throughput, optimizing travel time, and ensuring safety by avoiding collisions. The assignment of higher fitness values to solutions aligning with these objectives will drive the evolution of more efficient and secure railway systems.

The genetic operators, including selection, crossover, and mutation, play a pivotal role in generating new candidate solutions. Through these mechanisms, the system favours the survival of the fittest solutions, combining genetic information from successful parents to create potentially improved offspring. The introduction of mutation injects an element of exploration, facilitating the discovery of novel solutions.

The Genetic algorithm implemented as follows;

1. Population**:** The `Population` class represents a collection of schedules. Each schedule within the population is an individual solution to the optimization problem. The size of the population is specified by the `POPULATION_SIZE` constant.

2. Schedule: The `Schedule` class represents a single timetable solution. It contains a list of classes, each representing a scheduled train journey. The fitness of each schedule is calculated based on conflicts such as platform capacity violations and overlapping schedules.

3. Genetic Algorithm*:*

Initialization: The initial population is created with random schedules generated using the `initialize` method of the `Schedule` class.

Selection: The `Tournament Selection` method is used to select schedules from the population for crossover. This method randomly selects a subset of schedules (determined by `TOURNAMENT_SELECTION_SIZE`) and selects the best schedule from this subset based on fitness.

Crossover: The `crossover population` method selects elite schedules directly, then iteratively selects two parent schedules from the population using tournament selection. These selected parent schedules undergo crossover to produce new schedules with characteristics from both parents.

Mutation: After crossover, the resulting population undergoes mutation with a probability determined by the `MUTATION_RATE`. Mutation introduces random changes to individual schedules to maintain diversity and explore new regions of the search space.

4. Fitness Function: The fitness function is defined within the `Schedule` class. It evaluates each schedule based on conflicts such as platform capacity violations and overlapping schedules. The fitness is calculated as the reciprocal of the total number of conflicts plus one, ensuring that higher fitness values correspond to better schedules.

5.Elitism: The topper forming schedules are preserved in each generation without undergoing crossover or mutation. This ensures that the best solutions found so far are retained in subsequent generations.

Overall, the genetic algorithm iteratively evolves a population of schedules over multiple generations, with the aim of finding optimal or near-optimal solutions to the train timetable optimization problem

CHAPTER 7

TESTING

Testing plays a critical role in the development process of the project. It ensures the reliability, accuracy, and functionality of the system by identifying and resolving any issues or bugs. This chapter focuses on the testing phase of the project, outlining the different types of testing conducted and the approaches used to ensure a robust and high-quality application.

7.1 TESTING OBJECTIVES

The testing objectives for the track optimization system are:

1. **Verify Correctness:** Ensure that the system behaves as expected and produces correct results. Use unit testing to verify the correctness of individual components and integration testing to ensure that components work together correctly.
2. **Ensure Reliability:** Ensure that the system operates reliably and consistently under normal and abnormal conditions. Conduct functional testing to verify that the system meets specified requirements and regression testing to ensure that new changes do not introduce bugs or regressions.
3. **Evaluate Performance:** Assess the system's performance in terms of speed, responsiveness, and scalability. Perform performance testing to measure response times, throughput, and scalability under different load conditions.
4. **Validate Safety:** Verify that the system can detect and prevent potential safety hazards, such as train collisions. Conduct safety testing to verify the effectiveness of collision detection mechanisms and other safety features.
5. **Ensure Usability:** Evaluate the system's user interface to ensure it is intuitive and easy to use. Perform usability testing to assess the user interface's usability and identify any usability issues.
6. **Validate Requirements:** Ensure that the system meets all specified requirements. Conduct functional testing to verify that the system meets all specified functional requirements.

7. **Assess Maintainability:** Evaluate the ease with which the system can be maintained and updated. Conduct regression testing to ensure that new changes do not impact existing functionality and end-to-end testing to assess the system's overall maintainability.
8. **Verify Compliance:** Ensure that the system complies with relevant standards and regulations. Conduct compliance testing to verify that the system meets all applicable standards and regulations.
9. **Ensure Security:** Verify that the system is secure and protects sensitive data. Conduct security testing to identify and mitigate potential security vulnerabilities.
10. **Evaluate Scalability:** Assess the system's ability to scale to accommodate a growing number of trains and tracks. Perform performance testing to measure the system's scalability under increasing load conditions.

7.2 TESTING APPROACHES

Testing strategies for the proposed AI-enabled track optimization system for the Indian Railways would be crucial to ensure its reliability, performance, and safety. Here are some key testing strategies that could be employed:

- Unit Testing: Test individual components of the system, such as encoding, genetic operators, collision detection, and simulation, to ensure they function correctly in isolation.
- Integration Testing: Test the integration of different modules and components to ensure they work together as expected. This includes testing the interaction between the genetic algorithm and the collision detection mechanism, as well as the integration of the timetable optimization module with the overall system.
- Functional Testing: Test the system's functionality against the defined requirements. This includes testing the ability of the system to optimize train tracks, mitigate delays, streamline schedules, and prevent collisions as outlined in the requirements.
- Performance Testing: Test the system's performance under different load conditions to ensure it can handle the required number of trains and tracks efficiently. This includes testing the system's response time, throughput, and scalability.

- Safety Testing: Test the system's ability to detect and prevent potential collisions between trains. This includes testing the collision detection mechanisms under various scenarios to ensure they work effectively.
- Usability Testing: Test the system's user interface to ensure it is user-friendly and intuitive for railway operators and administrators to use. This includes testing the system's ability to display relevant information and provide useful insights to users.
- Regression Testing: Test the system after each change or update to ensure that new features do not introduce new bugs or issues and that existing functionality is not affected.
- End-to-End Testing: Test the entire system from start to finish to ensure that all components work together seamlessly to achieve the desired outcomes. This includes testing the system's ability to optimize train timetables, predict train delays, and simulate self-driving trains.

By employing these testing strategies, the proposed AI-enabled track optimization system can be thoroughly tested to ensure its reliability, performance, and safety before deployment in the Indian Railways.

7.3 TEST CASES

Each test case consists of a set of inputs, execution conditions, and expected results. These are designed to validate different aspects of the system, such as functionality, usability, performance, and security. In the context of the AI-enabled track optimization system for Indian Railways, test cases include:

Functionality Test Case:

- Objective: Verify that the system can optimize train tracks based on specified criteria.
- Inputs: Sample train schedules, track capacities, and safety constraints.
- Execution: Run the optimization algorithm.
- Expected Result: Optimized train tracks that minimize delays and collisions.

Usability Test Case:

- Objective: Evaluate the user interface's ease of use.

- Inputs: User interactions with the system.
- Execution: Interact with the user interface to perform common tasks.
- Expected Result: User interface is intuitive and easy to navigate.

Performance Test Case:

- Objective: Assess the system's response time under load.
- Inputs: Simulated high traffic conditions.
- Execution: Run multiple train scheduling scenarios simultaneously.
- Expected Result: System can handle the load without significant performance degradation.

Security Test Case:

- Objective: Verify the system's security measures.
- Inputs: Attempt to access the system without authorization.
- Execution: Try to exploit potential vulnerabilities.
- Expected Result: System is secure and protects sensitive data.

Regression Test Case:

- Objective: Ensure new updates do not introduce regressions.
- Inputs: Updated version of the system.
- Execution: Run existing test cases against the updated version.
- Expected Result: All existing functionalities work as expected in the updated version.

Each test case should be well-documented, including the objective, inputs, execution steps, and expected results. By systematically executing these test cases, we can ensure the system meets its requirements and performs as expected under various conditions.

CHAPTER 8

RESULTS

The AI-enabled track optimization system proposed for the Indian Railways aims to revolutionize operations by leveraging genetic algorithms and AI. The system's core objectives include optimizing tracks, reducing delays and streamlining schedules by encoding train schedules and routes, developing fitness functions, implementing genetic operators, the system systematically improves efficiency and safety.

8.1 DELAY PREDICTION

This system will predict train delays using random forest regression. Train details including train number, source, destination, and timing are given to the system. The model is trained with a dataset which contains train data including train names, station names, timings and historical data about train delays. The output shows whether the train is delayed or not depends on the time; if the train is late by 15 minutes or more then it is declared as delayed.

Figure 8. 1 Delay prediction

8.2 SELF DRIVING TRAIN

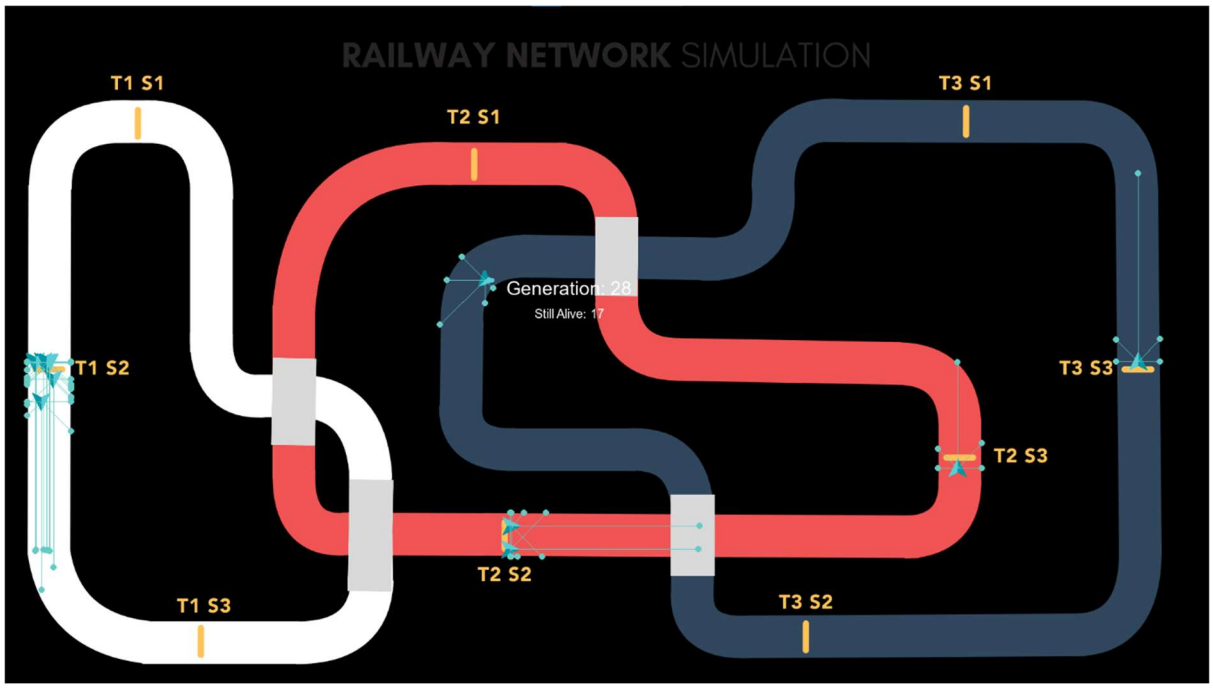

Figure 8. 2 Train simulation map-1

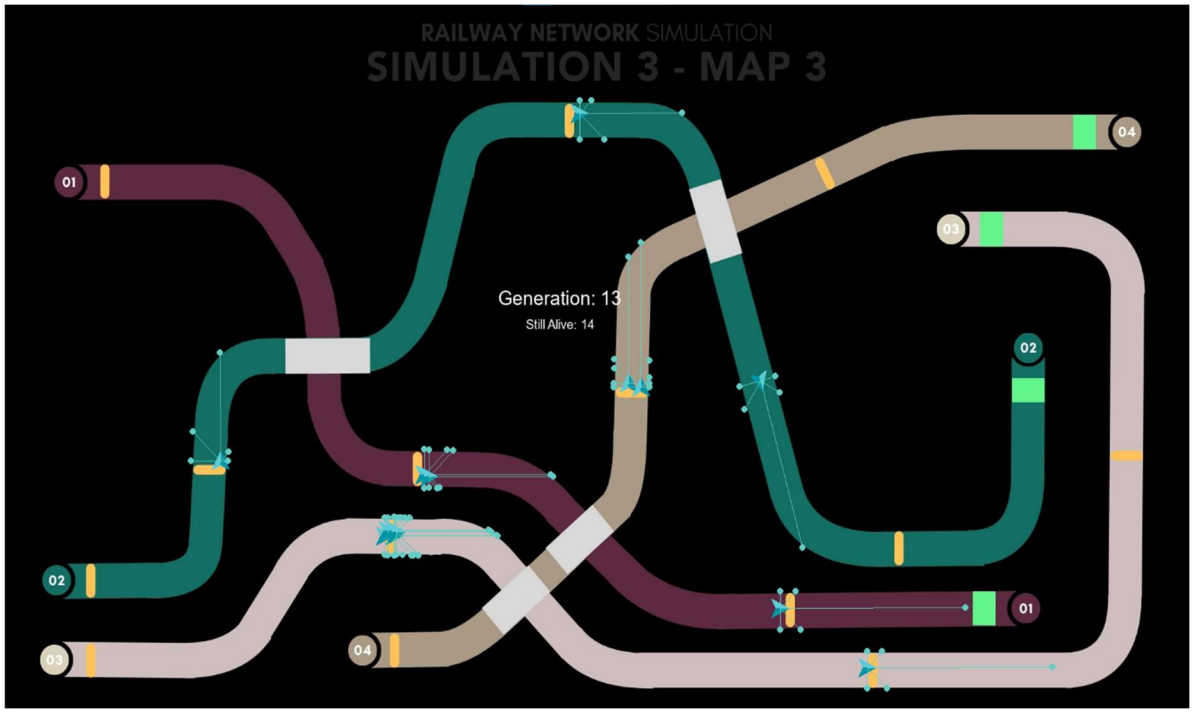

Figure 8. 3 Train simulation map-2

The above figures show simulations of self driving trains with a set of stations. The trains will stop in each station as per the specified time and will continue without collision. The efficiency of the trains will improve after each generation. If a generation has a fault such as collision or irregularity in track, that genome will be discarded and new generations will be created.

```
****** Running generation 25 ******

Population's average fitness: 15754.52267 stdev: 27884.84107
Best fitness: 157416.00000 - size: (5, 14) - species 3 - id 1110
Average adjusted fitness: 0.098
Mean genetic distance 2.153, standard deviation 0.697
Population of 50 members in 3 species:
   ID   age  size  fitness  adj fit  stag
  ====  ===  ====  =======  =======  ====
     1   25    18  59017.3    0.087    23
     3   15    20  157416.0    0.116     0
     4    2    12  29834.7    0.092     0
Total extinctions: 0
Generation time: 16.085 sec (11.570 average)
```

Figure 8. 4 Generation details

From the provided output, we can conclude several key pieces of information about the current state of the NEAT algorithm and its evolution process:

- Generation Information:

  The output indicates that the algorithm is currently running the 25th generation. This means that the population of neural networks has undergone 24 previous rounds of evolution and is currently in its 25th iteration.

- Population Statistics:

  The average fitness of the population in this generation is 15754.52267, with a standard deviation of 27884.84107. This gives us an idea of how well the population as a whole is performing in terms of solving the given problem. The best fitness achieved in this generation is 157416.00000. This represents the fitness score of the best-performing individual (genome) in the population. The size of the best-performing genome is specified as (5, 14), which likely refers to the structure or complexity of the neural network associated with this genome. The best-performing genome belongs to species 3 and has an ID of 1110.

- Species Information:

  The population is divided into different species based on genetic similarity among individuals. In this generation, there are three species (species 1, 3, and 4).Each species has its own statistics, including the age (how many generations it has existed), size (number of individuals), fitness, adjusted fitness, and stagnation count (how many generations the species has remained stagnant without improvement).For example, species 1 has been around for 25 generations, contains 18 individuals, and has a fitness of 59017.3. It has an adjusted fitness of 0.087 and has stagnated for 23 generations without improvement.

- Other Metrics:

  Mean genetic distance and standard deviation provide insights into the genetic diversity within the population. A higher mean genetic distance indicates greater diversity among individuals. The total number of extinctions in this generation is reported as 0, suggesting that no species went extinct during this iteration of evolution. Generation time indicates the time taken to complete the current generation. This information can be useful for performance monitoring and optimization. Overall, this output gives us a snapshot of the current state of the NEAT algorithm, including population performance, species diversity, and evolutionary progress in the 25th generation.

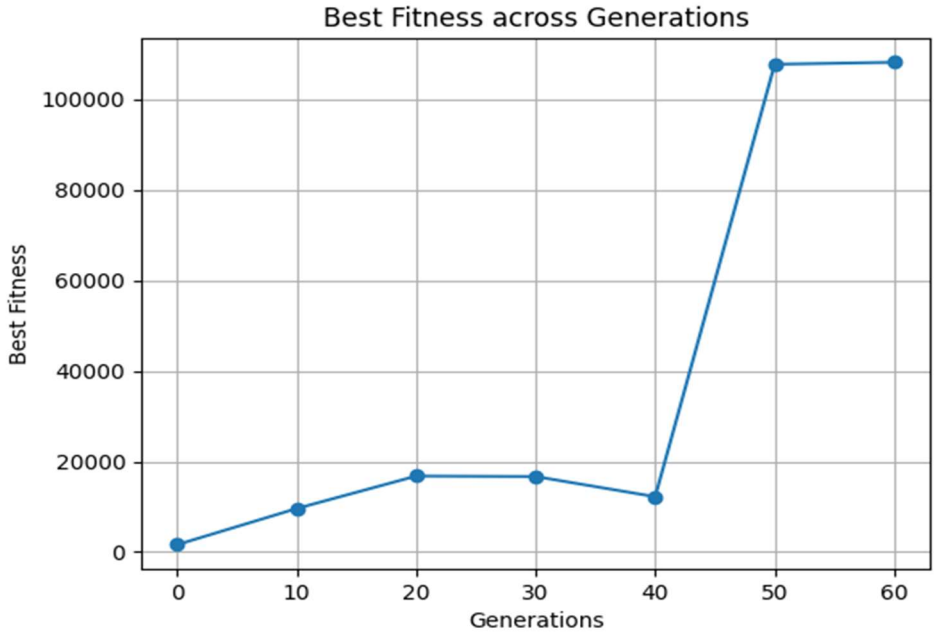

Figure 8. 5 Graph showing fitness after each generation

The graph depicts the progression of the best fitness values across multiple generations of the genetic algorithm optimization process. The x-axis represents the generation and the y-axis represents the best fitness. Fitness is a measure of how well a program performs its task. In this case, the task is for a train to navigate a track without crashing. The higher the fitness, the better the train is at navigating the track. The line in the graph shows that the fitness of the train increases as the number of generations increases. This means that the train is getting better at navigating the track as it goes through more generations.

Here are some other observations that can be made from the graph :

- The fitness starts at around 0 and increases to around 80,000 by generation 50.
- The rate of increase in fitness appears to be slowing down as the number of generations increases. This suggests that the train is approaching its maximum fitness level.
- Overall, the figure shows that the self-driving train program is successful. The train is able to learn how to navigate the track and its performance improves over time.

The project contributes to the advancement of AI applications in transportation. By demonstrating the feasibility of using NEAT to train artificial neural networks for autonomous train control, the project showcases the versatility and adaptability of AI technologies in addressing complex real-world challenges. This innovation not only has implications for the railway sector but also sets a precedent for the integration of AI in other modes of transportation. Moreover, the self-driving train project aligns with broader societal goals such as sustainability and resource optimization. By optimizing train routes and schedules, autonomous trains can reduce energy consumption, minimize delays, and enhance the overall sustainability of railway operations. This aligns with global efforts to create more efficient and environmentally friendly transportation systems.

Overall, the self-driving train project represents a significant step forward in the development of autonomous transportation systems. It embodies the potential of AI to revolutionize traditional industries, improve safety and efficiency, and contribute to a more sustainable future.

8.3 TRAIN TIMETABLE OPTIMIZATION

This module generates time table using genetic algorithm. We can add trains and its types, platforms, stations and its types, and timings. The system will generate suitable time table for each station without conflicts after a number of generations.

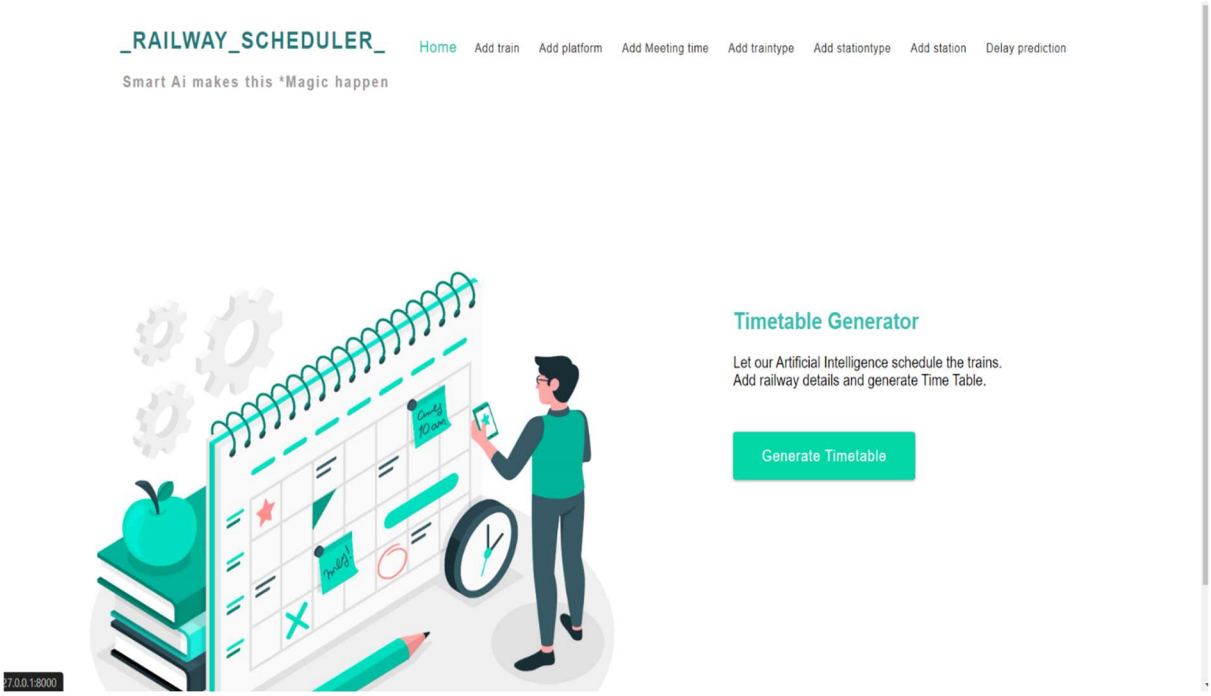

Figure 8. 6 Home page

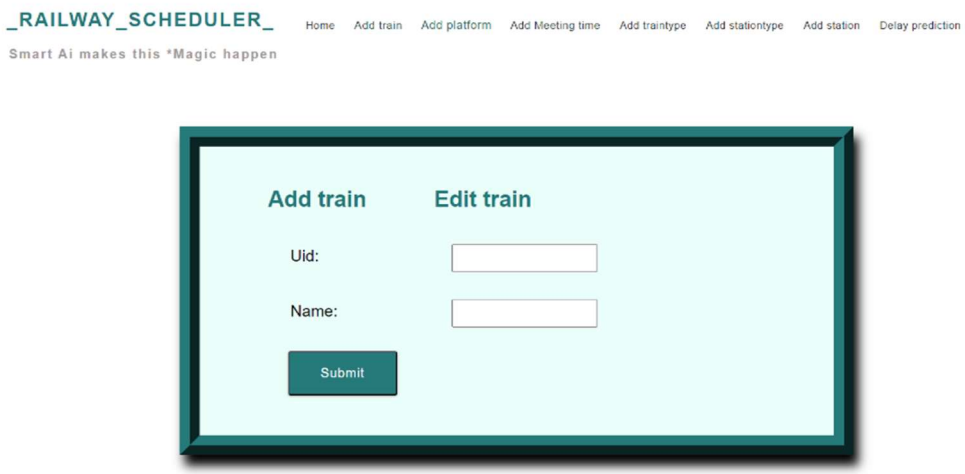

Figure 8. 7 Add train page

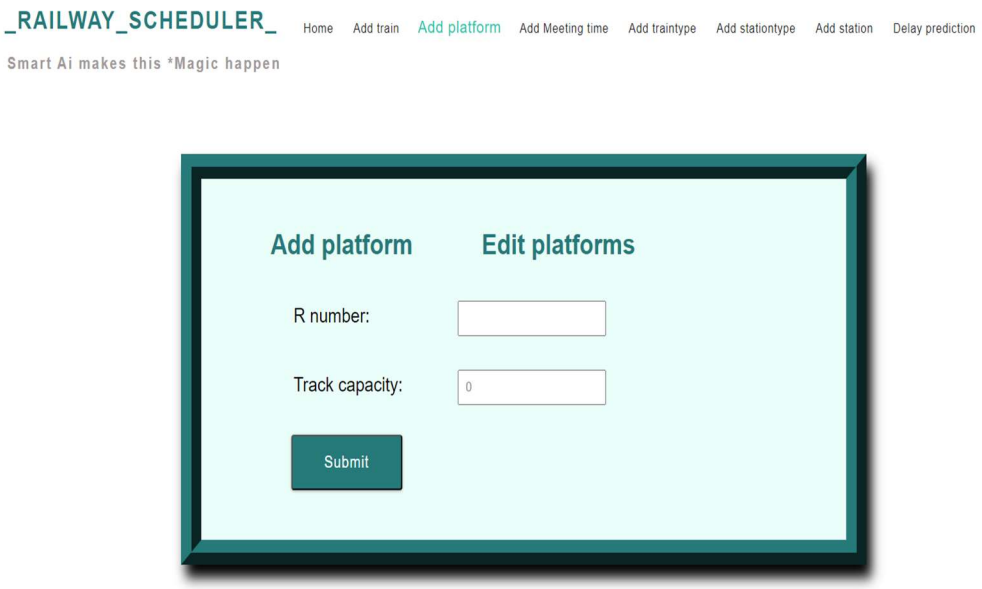

Figure 8. 8 Add platform page

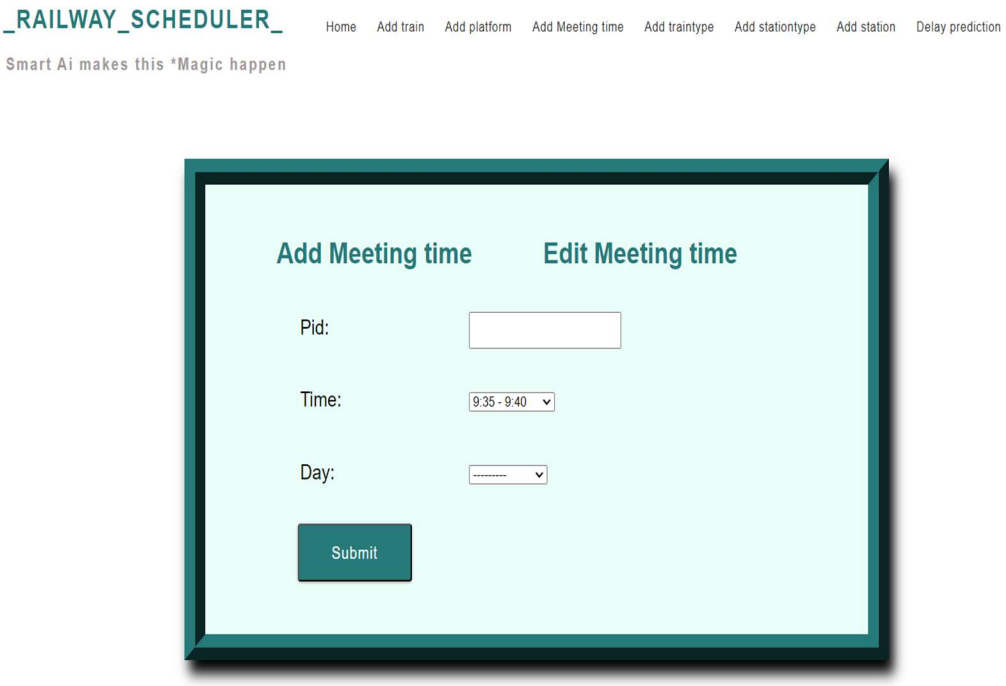

Figure 8. 9 Add timing page

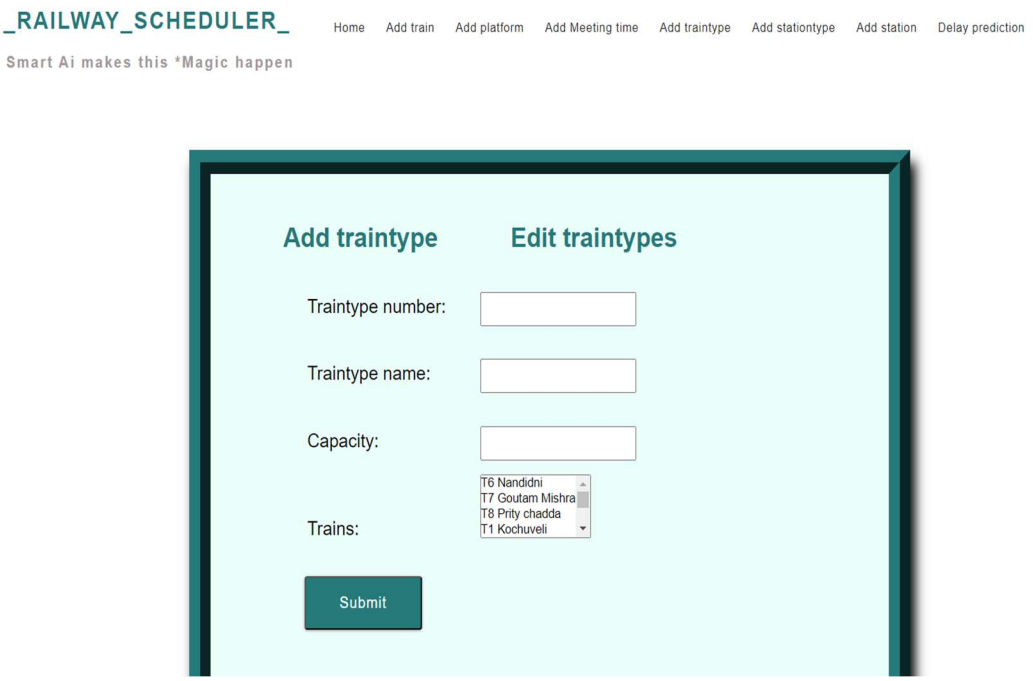

Figure 8. 10 Add train type

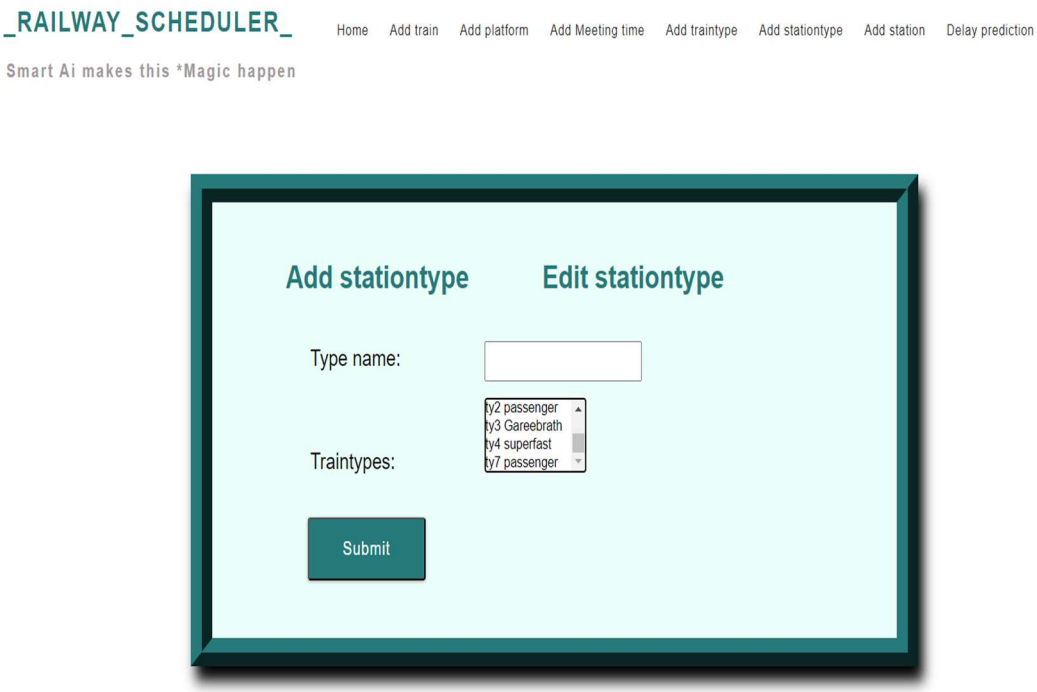

Figure 8. 11 Station type page

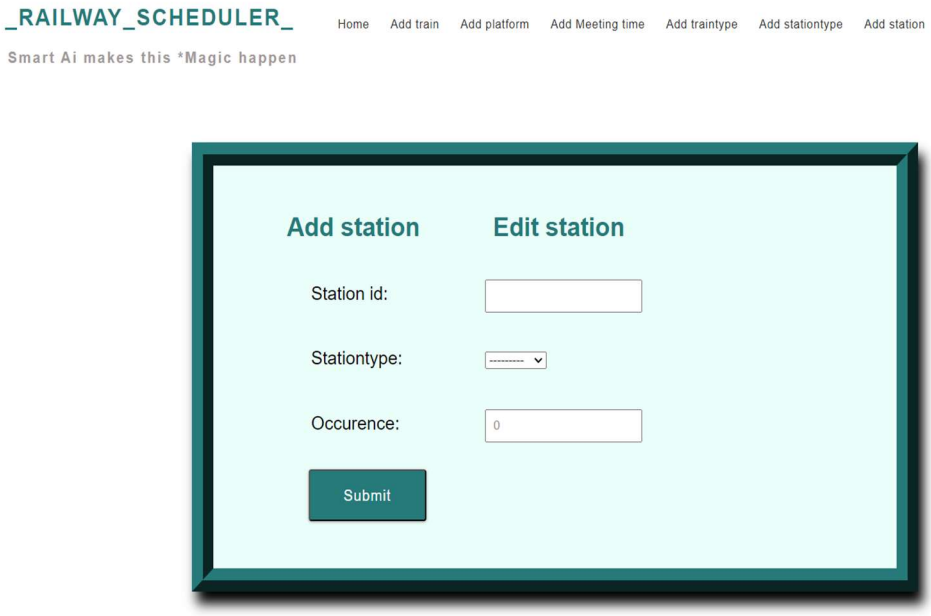

Figure 8. 12 Stations

After entering these details, the system will generate time table for each station as shown below.

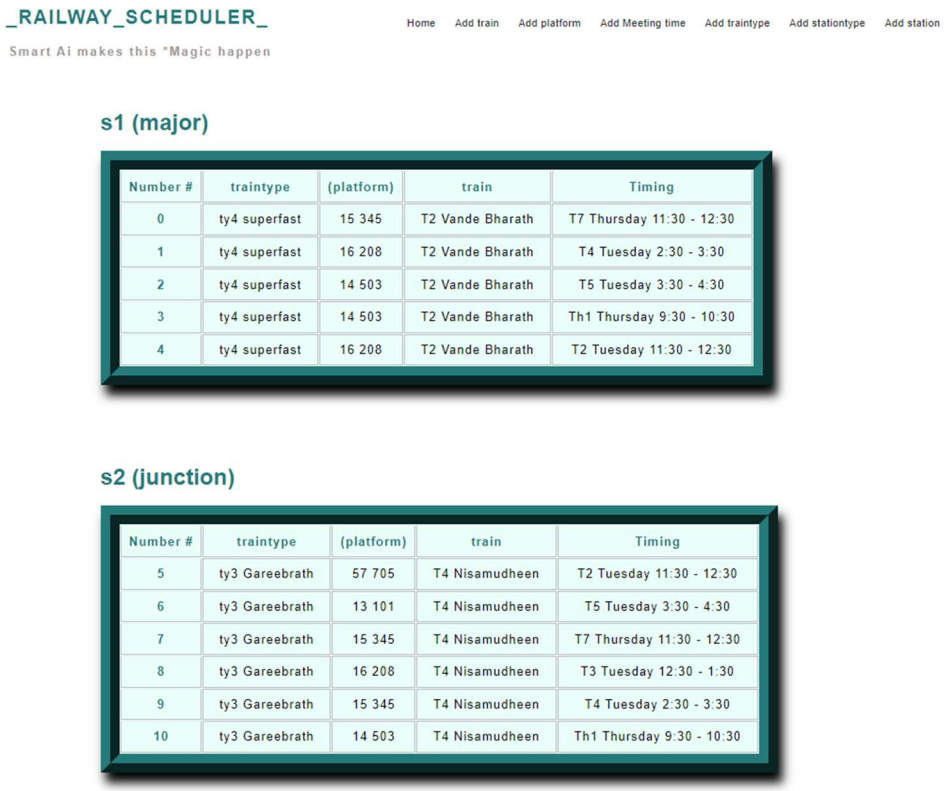

Figure 8. 13 Time table

CHAPTER 9

CONCLUSION

The integration of AI in track optimization for railways marks a transformative leap in the efficiency, safety, and sustainability of rail transportation systems. The AI-generated solutions harness advanced algorithms and predictive analytics to dynamically manage and optimize various aspects of track operations, including maintenance scheduling, fault detection, and resource allocation. This not only enhances the overall reliability of rail networks but also contributes to cost savings and minimizes downtime. The application of AI in track optimization brings about a paradigm shift, enabling proactive decision-making and preventive maintenance strategies. Through continuous monitoring and analysis of data, AI systems can anticipate potential issues, address them in real-time, and even forecast future maintenance needs. This not only extends the lifespan of railway infrastructure but also enhances the safety of operations, reducing the risk of accidents and disruptions. Moreover, the environmental impact of rail transportation is mitigated as AI algorithms optimize energy consumption and reduce unnecessary resource utilization. The implementation of AI-driven track optimization aligns with the broader goal of creating sustainable and smart transportation systems for the future. In essence, the AI-generated track optimization for railways emerges as a pivotal solution, fostering a more resilient, efficient, and environmentally conscious rail network. As technology continues to evolve, the marriage of artificial intelligence and rail transportation promises to redefine the landscape of the industry, offering a glimpse into a future where railways operate with unprecedented precision and sustainability.

REFERENCES

[1] H. J. Kaleybar, M. Davoodi, M. Brenna and D. Zaninelli, "Applications of Genetic Algorithm and Its Variants in Rail Vehicle Systems: A Bibliometric Analysis and Comprehensive Review," in IEEE Access, vol. 11, pp. 68972-68993, 2023, doi: 10.1109/ACCESS.2023.3292790.

[2] P. Wang and R. M. P. Goverde, "Train trajectory optimization of opposite trains on single-track railway lines," 2016 IEEE International Conference on Intelligent Rail Transportation (ICIRT), Birmingham, UK, 2016, pp. 23-31, doi: 10.1109/ICIRT.2016.7588546.

[3] M. T. Lazarescu and P. Poolad, "Asynchronous Resilient Wireless Sensor Network for Train Integrity Monitoring," in IEEE Internet of Things Journal, vol. 8, no. 5, pp. 3939-3954, 1 March1, 2021, doi: 10.1109/JIOT.2020.3026243.

[4] N. Bešinović et al., "Artificial Intelligence in Railway Transport: Taxonomy, Regulations, and Applications," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 9, pp. 14011-14024, Sept. 2022, doi: 10.1109/TITS.2021.3131637.

[5] S. D. Immanuel and U. K. Chakraborty, "Genetic Algorithm: An Approach on Optimization," 2019 International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2019, pp. 701-708, doi: 10.1109/ICCES45898.2019.9002372.

[6] J. Sha and M. Xu, "Applying hybrid genetic algorithm to constrained trajectory optimization," Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology, Harbin, China, 2011, pp. 3792-3795, doi: 10.1109/EMEIT.2011.6023884.

[7] Y. Liu, S. Li and S. Dou, "Time delay prediction and compensation method based on WPD-PSO-LSTM for train wireless network control," 2023 IEEE International Conference on Sensors, Electronics and Computer Engineering (ICSECE), Jinzhou, China, 2023, pp. 183-187, doi: 10.1109/ICSECE58870.2023.10263578

[8] D. Yuan, J. Huang, X. Yang and J. Cui, "Improved random forest classification approach based on hybrid clustering selection," 2020 Chinese Automation Congress (CAC), Shanghai, China, 2020, pp. 1559-1563, doi: 10.1109/CAC51589.2020.932671