UNIVERSITY
OF SUSSEX

Decentralized Traffic Congestion Prediction: Integrating AI, Blockchain, and IPFS for Smart Transportation Systems

By
Candidate No: 276236

Supervisor:
Dr. Naercio Magaia

Word count: 14819 words

MSc. Artificial Intelligence and Adaptive Systems
Department of Informatics
University of Sussex
2024

Statement of Originality and Intellectual Property Rights:

Signed: **Date: 27/08/2024**

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Dr. Naercio Magaia, for his invaluable guidance, support, and encouragement throughout this research project. His insightful feedback during our weekly meetings was crucial in shaping the direction of this work. I am also thankful to the faculty and staff of the Department of Informatics at the University of Sussex, particularly for the access to well-equipped labs and the technical support provided by the lab staff, which greatly facilitated my research. My appreciation also extends to my colleagues and fellow students in the MSc. Artificial Intelligence and Adaptive Systems program for their support and the many fruitful discussions that helped refine my ideas. My heartfelt thanks go to my family and friends for their unwavering support and understanding throughout my studies. Finally, I acknowledge the data providers and the open-source community for their essential contributions to this research.

Contents

Contents **iii**

List of Tables **vi**

List of Figures **vii**

List of Acronyms **ix**

Abstract **x**

1 Introduction **1**
- 1.1 Overview and Motivation . . . 1
- 1.2 Problem Statement and Research Objectives . . . 3
- 1.3 Structure of the Document . . . 5

2 Literature Review **6**
- 2.1 Artificial Intelligence (AI) . . . 6
- 2.2 Federated Learning . . . 7
  - 2.2.1 How Federated Learning Works . . . 7
- 2.3 Blockchain . . . 8
  - 2.3.1 Role of Blockchain Technology . . . 8
  - 2.3.2 Ganache for Blockchain Simulation . . . 8
  - 2.3.3 Smart Contracts in Blockchain . . . 9
- 2.4 InterPlanetary File System (IPFS) . . . 10
  - 2.4.1 Overview of IPFS . . . 10
  - 2.4.2 File Chunking and Cryptographic Hashing . . . 10
  - 2.4.3 Pinata IPFS: Enhancing Decentralized Storage . . . 10
  - 2.4.4 Content Identifiers (CIDs) and Merkle DAG . . . 11
  - 2.4.5 Distributed Storage . . . 11
  - 2.4.6 Merkle Tree in IPFS . . . 11
  - 2.4.7 Data Retrieval . . . 12
  - 2.4.8 Pinata IPFS in Decentralized Storage Solutions . . . 12
- 2.5 Locust . . . 13
  - 2.5.1 Key Features . . . 13
- 2.6 Internet of Vehicles (IoV) . . . 13
- 2.7 Artificial Intelligence (AI) in Traffic Management and Security . . . 14
- 2.8 Blockchain Technology in IoV Systems . . . 15
- 2.9 Decentralized Storage in IoV . . . 16
- 2.10 Integration of AI and Blockchain for Enhanced IoV Systems . . . 17

2.11 Traffic Prediction Datasets . . . 18
2.12 Summary . . . 19

3 Design & Implementation 20
3.1 System Architecture . . . 20
3.1.1 Data Collection Layer . . . 21
3.1.2 Data Processing Layer . . . 21
3.1.3 Blockchain Layer . . . 21
3.1.4 Decentralized Storage Layer . . . 22
3.1.5 Application Layer . . . 23
3.2 Dataset . . . 24
3.2.1 Citation and Credit . . . 25
3.3 Data Processing . . . 25
3.3.1 Data Processing Workflow . . . 26
3.3.2 Data Cleaning . . . 26
3.3.3 Feature Engineering . . . 27
3.3.4 Feature Selection . . . 28
3.3.5 Data Visualization . . . 28
3.4 AI Model Development . . . 31
3.4.1 Model Selection . . . 31
3.4.2 Data Splitting . . . 32
3.4.3 Training and Hyperparameter Tuning . . . 32
3.4.4 Final Model Evaluation . . . 32
3.4.5 Ganache Implementation and Configuration . . . 33
3.5 Smart Contracts and IPFS Integration for Federated Learning . . . 34
3.5.1 Smart Contracts . . . 34
3.5.2 Integrated Workflow . . . 35
3.6 Limitations . . . 36
3.7 Summary . . . 37

4 Performance Evaluation 38
4.1 AI Model Evaluation and Selection . . . 38
4.1.1 Methodology Overview . . . 39
4.1.2 Model Performance . . . 39
4.1.3 Discussion . . . 44
4.2 Storage Service Performance Comparison . . . 45
4.3 System Performance and Evaluation . . . 45
4.3.1 Computational Performance . . . 45
4.3.2 Blockchain Operations and Efficiency . . . 47
4.3.3 Scalability and Load Testing with Locust . . . 50
4.3.4 Summary of Results . . . 51

5 Conclusion and Future Work 52
5.1 Conclusion . . . 52
5.2 Future Works . . . 53

Bibliography 55

A Feature Sets 59

List of Tables

2.1 Comparison of AI Techniques in Traffic Management 15
2.2 Blockchain and AI Integration in IoV 18
2.3 Key Features of LargeST Dataset 19

3.2 Description of the Traffic Events Dataset Columns [1] 25
3.3 Final Selected Features after Feature Engineering 28

4.1 Performance Metrics of Evaluated Models 39
4.2 Performance Comparison of Google Drive and Pinata IPFS 45
4.3 Contributor Reward Distribution 48

A.1 Comparison of Initial, Feature Engineered, and Final Selected Feature Sets 59

List of Figures

1.1 Overview of the Internet of Vehicles (IoV) communication model (Extracted from [2]). . . . 1
1.2 Comparison of Centralized, Distributed, and Federated Learning models (Extracted from [3]). . . . 2
1.3 Architecture of the blockchain system integrated with IPFS for IoV systems. . 3

2.1 Interaction of Smart Contracts within the Ganache Blockchain Environment. 9
2.2 IPFS Data Storage and Retrieval Mechanism with Merkle DAG 12
2.3 Blockchain Architecture for IoV Systems 16
2.4 IPFS Architecture for Decentralized Data Storage 17

3.1 Overview of the System Architecture 20
3.2 System Architecture for IoV with Federated Learning and Blockchain Integration 24
3.3 The entire data processing workflow is depicted in the flowchart. 26
3.4 Distribution of Traffic Event Types 29
3.5 Number of Events Per Hour After Preprocessing 29
3.6 Number of Events by Day of the Week After Preprocessing 30
3.7 Geographical Distribution of Events After Preprocessing 30
3.8 Correlation Matrix of Selected Features 31

4.1 Gradient Boosting Regressor CPU & Memory Utilization 39
4.2 Random Forest Regressor CPU & Memory Utilization 40
4.3 K-Nearest Neighbours Regressor CPU & Memory Utilization 41
4.4 LSTM CPU & Memory Utilization 41
4.5 Linear Regression CPU & Memory Utilization 42
4.6 XGBoost Regressor CPU & Memory Utilization 42
4.7 Decision Tree Regressor CPU & Memory Utilization 43
4.8 Extra Tree Regressor CPU & Memory Utilization 44
4.9 MSE Comparison of AI Models 44
4.10 Local Model Training for Contributor 1 46
4.11 Local Model Training for Contributor 2 46
4.12 Local Model Training for Contributor 3 46
4.13 Local Model Training for Contributor 4 47
4.14 Aggregated Training 47
4.15 Ganache Interface Displaying Mined Blocks and Transactions 48
4.16 Ganache Interface Showing Account Balances and Transaction Counts . . 49
4.17 Ganache Interface Showing Transaction Details and Gas Used 49
4.18 Scalability Testing with Locust - Total Requests 50

4.19 Scalability Testing with Locust - Number of Users 50
B.1 CPU & Memory utilization for Data Preprocessing 60

List of Acronyms

Acronym	**Description**
AI	Artificial Intelligence
FL	Federated Learning
IoV	Internet of Vehicles
IPFS	InterPlanetary File System
ITS	Intelligent Transportation Systems
DL	Deep Learning
RL	Reinforcement Learning
MADRL	Multi-Agent Deep Reinforcement Learning
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
XAI	Explainable Artificial Intelligence
ML	Machine Learning
VANET	Vehicular Ad Hoc Network
DSN	Decentralized Storage Network
IDS	Intrusion Detection System
GBT	Gradient Boosted Trees
RF	Random Forest
DDoS	Distributed Denial of Service
TMC	Traffic Message Channel
LSTW	Large-Scale Traffic and Weather Events
SVM	Support Vector Machine
VT	Vision Transformers
GAN	Generative Adversarial Network
MSE	Mean Squared Error
RMSE	Root Mean Squared Error

Abstract

This dissertation explores the development of a decentralized, secure, and efficient traffic congestion prediction system by integrating Artificial Intelligence (AI), Federated Learning (FL), and blockchain technology within the Internet of Vehicles (IoV) framework. The research addresses key challenges in urban traffic management, such as data centralization, privacy concerns, scalability issues, and inefficient data storage. The proposed system leverages blockchain for secure data storage and transparency, IPFS for decentralized data management, and FL to preserve data privacy while enabling collaborative AI model training across multiple nodes.

The methodology involves designing a multi-layered architecture that incorporates data collection, processing, blockchain, decentralized storage, and application layers. The system's performance is evaluated through various metrics, including data retrieval time, scalability, and the efficiency of AI models for traffic prediction. The study demonstrates that the integration of AI with blockchain and decentralized storage enhances the accuracy of traffic predictions while ensuring data security and user privacy.

The results indicate that the Random Forest Regressor outperforms other models in predictive accuracy and resource efficiency, making it the optimal choice for real-time traffic management in IoV systems. The dissertation concludes with a discussion on the limitations of the current implementation and suggests directions for future work, including deploying the system on a global scale and exploring more sophisticated blockchain frameworks to further improve scalability and performance.

Chapter 1

Introduction

1.1 Overview and Motivation

The rapid advancement of intelligent transportation systems (ITS) has been fundamentally driven by the evolution of vehicular networks, notably the transition from Vehicular Ad Hoc Networks (VANETs) to the Internet of Vehicles (IoV). The IoV represents a sophisticated network where vehicles, infrastructure, and pedestrians interact through a seamless exchange of information, aiming to enhance road safety, optimize traffic flow, and improve the overall efficiency of transportation systems. This evolution is underpinned by advanced communication technologies such as IEEE 802.11p and 5G, which are essential for vehicle-to-everything (V2X) communications. These technologies facilitate real-time data exchange between vehicles and roadside infrastructure, enabling more dynamic and responsive traffic management solutions[4][2]. Figure 1.1 provides an overview of the IoV communication model, illustrating the interconnectedness of modern vehicular networks.

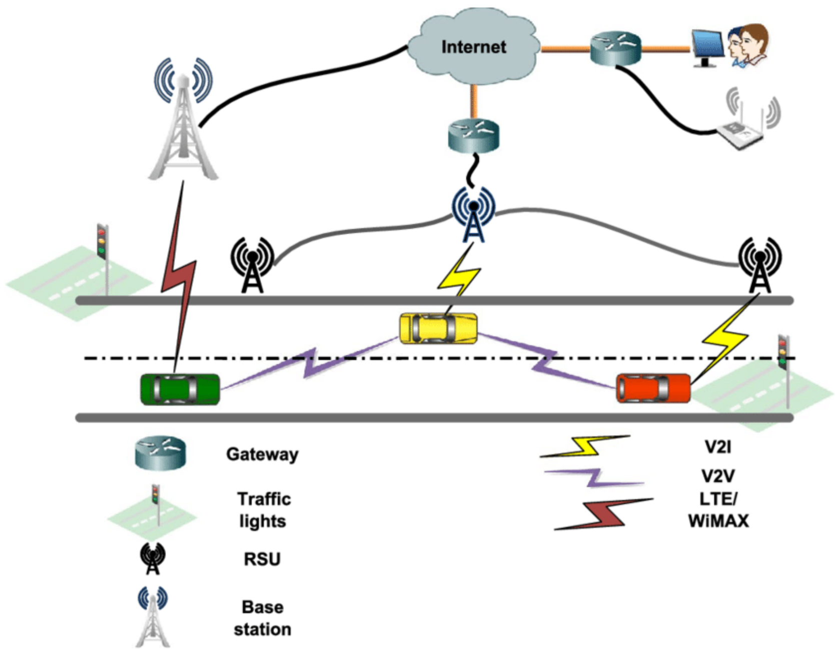


Figure 1.1: Overview of the Internet of Vehicles (IoV) communication model (Extracted from [2]).

Despite these technological advancements, urban traffic congestion continues to pose significant challenges. The increasing volume of vehicles in urban areas has led to severe congestion, resulting in longer travel times, higher fuel consumption, and increased emissions[5]. Traditional traffic management systems, which rely heavily on centralized models, are often inadequate in addressing these issues effectively. These systems suffer from several key limitations, including the centralization of data, which raises serious privacy concerns[6]; scalability challenges that hinder the ability to manage growing urban environments[7]; and inefficiencies in data storage and retrieval that complicate real-time traffic management[8].

The motivation for this research arises from the pressing need to develop innovative solutions that can overcome these challenges. Decentralized technologies, such as Federated Learning (FL), offer a promising approach to enhancing the scalability, privacy, and security of traffic management systems. FL enables the collaborative training of machine learning models across multiple decentralized devices, such as vehicles and roadside units (RSUs), without requiring the transfer of raw data to a central server[3]. This method preserves data privacy by ensuring that sensitive information, such as vehicle locations and driving behaviors, remains localized. Moreover, FL facilitates the aggregation of locally trained models into a global model, which improves the accuracy and robustness of traffic predictions. Figure 1.2 illustrates the distinctions between centralized, distributed, and Federated Learning models, highlighting the advantages of FL in privacy-preserving data management[3].

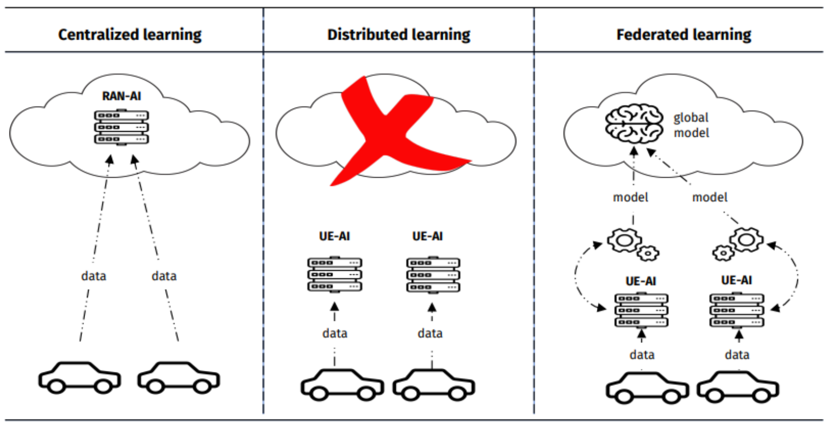


Figure 1.2: Comparison of Centralized, Distributed, and Federated Learning models (Extracted from [3]).

In addition to FL, the integration of decentralized storage solutions, such as the InterPlanetary File System (IPFS), is crucial for managing the vast amounts of data generated by IoV systems. IPFS offers a distributed method for data storing and sharing among nodes in a network, improving data security, resilience, and availability.[8]. Unlike traditional centralized storage systems, IPFS divides data into smaller chunks and distributes them across multiple nodes, ensuring that data remains accessible even in the event of network failures. This decentralized approach is particularly beneficial for IoV environments, where network reliability and data integrity are paramount. The combination of IPFS with blockchain technology further strengthens the security of data storage and management. Blockchain provides a decentralized, immutable ledger that records all transactions related to data storage, model aggregation, and access control, ensuring transparency and tamper-proof management of traffic data[9][10]. The architecture of IPFS and its integration with blockchain for IoV systems is depicted in Figure 1.3.

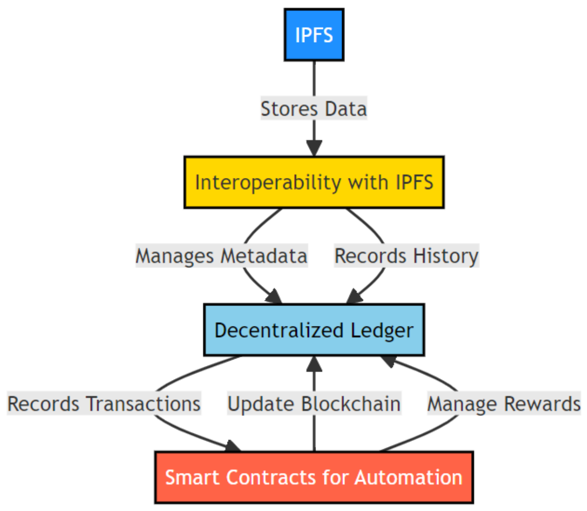


Figure 1.3: Architecture of the blockchain system integrated with IPFS for IoV systems.

The convergence of these advanced technologies presents an opportunity to overcome the limitations of traditional traffic management systems by enabling more secure, scalable, and efficient solutions. This research is driven by the need to explore and harness these technologies to address the pressing challenges of urban traffic congestion, laying the groundwork for the development of a novel traffic congestion prediction system.

1.2 Problem Statement and Research Objectives

Despite the significant advancements in ITS and the development of IoV, urban traffic management remains a challenging problem, especially in rapidly growing cities where the existing infrastructure is struggling to keep up with increasing traffic volumes. Cities such as Los Angeles, Beijing, and Mumbai are notorious for their severe traffic congestion, which not only increases travel times but also contributes to higher fuel consumption, greater emissions, and a decline in overall transportation efficiency. Traditional traffic management systems, which often rely on centralized models, have repeatedly failed to adapt to the dynamic and complex nature of urban traffic in these environments, leading to frequent bottlenecks and gridlock situations.

For instance, in Los Angeles, one of the most traffic-congested cities in the world, centralized traffic management systems have struggled to efficiently manage traffic flow during peak hours, leading to significant economic losses and increased environmental pollution[11]. Similarly, Mumbai, with its dense population and inadequate road infrastructure, faces chronic traffic congestion, and centralized traffic management systems have often failed to provide timely interventions to alleviate these issues[12]. These examples highlight several key issues with traditional traffic management systems:

1. **Data Centralization and Privacy Concerns:** Centralized systems for traffic management need enormous amounts of data to be collected and processed centrally,

raising significant privacy concerns. Sensitive information, such as vehicle locations and routes, can be vulnerable to breaches, leading to a loss of trust among users[6].

2. **Scalability and System Resilience:** As urban environments grow and the number of connected vehicles increases, centralized systems face challenges in scaling effectively. The central servers become single points of failure, and any disruption can lead to significant delays and inefficiencies in traffic management. Additionally, the centralized nature makes these systems more susceptible to cyber-attacks, which can compromise the entire network[7].
3. **Inefficient Data Storage and Access:** Traditional data storage solutions are often not equipped to handle the large volumes of data generated by IoV systems. These solutions struggle with efficient storage, retrieval, and sharing of data across multiple entities, leading to latency issues and reduced effectiveness in real-time traffic management[8][13].
4. **Limited Integration of Advanced Technologies:** While technologies such as Federated Learning, blockchain, and decentralized storage systems like IPFS offer promising solutions, their integration into existing traffic management systems is still limited. Without proper integration, the benefits of these technologies—such as enhanced data privacy, improved scalability, and secure data management—cannot be fully realized[9][3].

To address these challenges, this research aims to develop a robust, secure, and decentralized traffic congestion prediction system by:

1. **Implementing Secure and Decentralized Data Storage:** Leverage blockchain technology and IPFS to ensure secure and tamper-proof storage and sharing of traffic data, providing data immutability, transparency, and resistance to cyber-attacks[14].
2. **Developing a Decentralized and Transparent Contributor Reward System:** Design and implement a blockchain-based reward system to incentivize contributors (e.g., vehicles, sensors) for their data contributions. This system will use smart contracts to ensure transparency and fairness in reward distribution, encouraging active participation and continuous data sharing within the network[15].
3. **Enhancing System Scalability and Data Handling Efficiency:** Develop a system architecture that efficiently handles large volumes of traffic data, leveraging blockchain and decentralized storage technologies like IPFS. The design will focus on reducing latency and ensuring quick access to stored data, allowing the system to scale to accommodate the growing number of connected vehicles and data sources in urban environments[8][13].
4. **Incorporating Federated Learning for Privacy-Preserving Data Analysis:** Implement Federated Learning techniques to enable collaborative data analysis across distributed nodes without the need to centralize sensitive data. This approach will help preserve user privacy while still allowing the system to make accurate and timely traffic predictions based on a broad dataset[3].

5. **Facilitating Real-Time Traffic Data Management and Prediction:** Develop methods and algorithms for processing and managing traffic data in real-time, leveraging decentralized computing paradigms such as edge and fog computing. The goal is to enable dynamic responses to changing traffic conditions, thereby improving traffic flow and reducing congestion[4][16].

6. **Improving Urban Transportation Networks:** The integration of these advanced technologies will not only reduce congestion and optimize traffic flow but also foster trust and transparency among users by safeguarding sensitive data [17, 18]

By achieving these objectives, this research aims to develop a robust, secure, and efficient traffic congestion prediction system that utilizes blockchain technology and Federated Learning to improve urban transportation networks. The outcomes of this research will contribute to more effective traffic management, improved route planning, and enhanced data security and transparency, ultimately benefiting smart city environments.

1.3 Structure of the Document

This document is organized into the following sections:

- **Chapter 1: Introduction** - Provides an overview and motivation for the research, outlines the problem statement, and sets forth the research objectives.
- **Chapter 2: Literature Review & Background** - Reviews the relevant literature, discussing technological advances in the Internet of Vehicles (IoV), AI in traffic management, and blockchain technology. It also covers the integration of these technologies and the importance of user privacy in IoV systems.
- **Chapter 3: Methodology** - Describes the system architecture, including the data collection, processing, and blockchain layers. It also covers data processing techniques, AI model development, and the integration of IPFS for decentralized storage.
- **Chapter 4: Results and Discussions** - Presents the implementation of the system, evaluates AI models, and discusses the results obtained from the integration of AI, blockchain, and IPFS.
- **Chapter 5: Conclusion and Future Work** - Summarizes the key findings of the research and suggests areas for future work.
- **Bibliography** - Lists the references and sources cited throughout the document.
- **Appendices** - Includes additional material such as detailed feature sets, CPU and memory utilization plots, and other relevant data that support the main content of the dissertation.

Each chapter builds upon the previous ones, gradually leading to the development and evaluation of a decentralized, secure, and efficient traffic congestion prediction system, ultimately contributing to the advancement of smart transportation systems.

Chapter 2

Literature Review

The transition from Vehicular Ad Hoc Networks (VANETs) to the Internet of Vehicles (IoV) marks a significant advancement in intelligent transportation systems, designed to improve safety and efficiency in urban environments[4]. As vehicular networks evolve, the integration of advanced technologies becomes essential to address the increasing complexity of traffic management, security, and data privacy. This literature review explores key technologies, including blockchain, Artificial Intelligence (AI), Federated Learning (FL), and decentralized storage solutions like the InterPlanetary File System (IPFS), which support the development of secure and scalable IoV systems. The review highlights current challenges and discusses future opportunities for integrating these technologies into effective traffic management solutions.

2.1 Artificial Intelligence (AI)

The development of AI models for traffic congestion prediction plays a critical role in modern Intelligent Transportation Systems (ITS). This process involves selecting and training machine learning models to forecast traffic patterns and congestion levels. A key consideration in this phase is the model's ability to handle the complexity of traffic data, including non-linear relationships and temporal dependencies. For instance, Deep Learning (DL) techniques have shown superior capabilities in forecasting traffic congestion compared to traditional methods due to their ability to process and analyze large volumes of data [19]. Additionally, the integration of external factors like weather conditions into these models further improves their prediction accuracy [20].

In the realm of traffic congestion prediction, various AI models are employed, each offering distinct advantages and facing specific challenges. **Linear Regression** is a simple and computationally efficient supervised learning technique that models the relationship between dependent and independent variables through a linear equation. However, its assumption of linear relationships often limits its ability to capture complex patterns in traffic data [20]. **Decision Trees** offer a more flexible approach, capable of handling non-linear relationships by splitting the data into branches based on feature values. This interpretability makes Decision Trees a popular choice in various applications. Nonetheless, they are prone to overfitting, especially when the trees are deep, necessitating techniques like pruning to improve generalization [10]. On the other hand, **Support Vector Machines (SVMs)** are powerful in high-dimensional spaces and are robust against outliers. Despite their effectiveness, SVMs can be computationally intensive and sensitive to parameter settings, which may limit their scalability in large datasets [21].

Ensemble learning methods like **Random Forest** and **Gradient Boosting Machines (GBMs)** further enhance predictive accuracy by combining multiple models. Random Forest reduces overfitting by aggregating the predictions of various decision trees, making it robust to noisy data. However, it is computationally expensive and can be less interpretable compared to simpler models [21]. GBMs build models sequentially, with each new model correcting the errors of the previous ones. This method provides high predictive accuracy and is resistant to overfitting, but it requires careful tuning of hyperparameters and is also computationally demanding [21].

In deep learning, **Convolutional Neural Networks (CNNs)** are particularly effective for tasks involving spatial data, such as traffic congestion prediction, where they can capture spatial dependencies within the data. In practical use, however, CNNs could provide a major challenge since they require significant amounts of data and computer power to train. [10]. Similarly, **Long Short-Term Memory (LSTM)** networks are well-suited for time-series forecasting due to their ability to capture temporal dependencies. LSTMs are particularly effective in predicting traffic patterns over time but share the challenge of requiring large datasets and extensive computational power [20].

Finally, **Multi-Agent Deep Reinforcement Learning (MADRL)** is gaining attention for its ability to manage dynamic environments, such as coordinating traffic signals across multiple intersections. MADRL adapts well to complex scenarios, where multiple agents must learn and make decisions simultaneously. Despite its potential, MADRL is resource-intensive and requires extensive training, making it challenging to implement at scale [22]. These AI models collectively advance the field of traffic congestion prediction, with each model offering unique benefits that cater to different aspects of the prediction process.

2.2 Federated Learning

Federated Learning (FL) is a decentralized approach to machine learning designed to enhance data privacy and reduce the need for data centralization. It enables multiple nodes, such as vehicles or roadside units (RSUs), to collaboratively train a machine learning model while keeping their local data private [23][14]. This section provides a detailed explanation of the Federated Learning process.

2.2.1 How Federated Learning Works

Federated Learning operates through a systematic process involving several key steps:

1. **Local Model Training**: Each node in the network trains a machine learning model on its own local data. This process occurs independently at each node, allowing the model to learn from the local dataset without transmitting the data to a central server [23].

2. **Model Aggregation**: After local training is completed, the nodes send their updated model parameters (e.g., weights and biases) to a central server or aggregator. Importantly, the raw data remains on the local nodes, enhancing privacy.

3. **Global Model Update**: The central server aggregates the received model parameters to create a global model. This aggregation typically involves averaging the

parameters from all local models to form a unified global model that reflects the knowledge gathered across all nodes [23].

4. **Distribution of Global Model**: The updated global model is then sent back to each node. Each node integrates this global model with its local model, incorporating the improvements and knowledge gained from other nodes[14].

5. **Iterative Training and Aggregation**: The process of local training, model aggregation, and global model distribution is repeated iteratively. This iterative cycle continues until the global model reaches the desired level of performance and accuracy.

By maintaining local data on individual nodes and only sharing model updates, Federated Learning enables collaborative model training while preserving data privacy. This approach facilitates the development of robust machine learning models across distributed data sources without compromising sensitive information.

2.3 Blockchain

The integration of blockchain technology plays a pivotal role in ensuring data integrity, transparency, and security within the project [24]. Blockchain's decentralized and immutable ledger system is crucial for managing data exchanges in vehicular networks, where maintaining the accuracy and trustworthiness of data is paramount.

2.3.1 Role of Blockchain Technology

One of the main features of blockchain technology is its immutability means the data stored on the blockchain cannot be removed or changed and its decentralized structure. This inherent property of blockchain provides several key benefits:

- **Data Integrity**: Blockchain ensures that all transactions and data entries are secure and tamper-proof. Each block in the chain contains a cryptographic hash of the previous block, creating a continuous and unalterable chain of records[14].

- **Transparency and Trust**: The decentralized nature of blockchain means that all participants in the network have access to the same ledger. Since everyone can independently confirm the integrity of transactions, this transparency increases confidence among participants.

- **Security**: Blockchain uses advanced cryptographic techniques to secure data. This makes it highly resistant to fraud and unauthorized access, protecting sensitive information within the network.

2.3.2 Ganache for Blockchain Simulation

Ganache is a blockchain simulator that provides a controlled environment for testing and developing blockchain-based applications [25]. It is particularly useful for simulating the blockchain environment without the complexities of a live network. In this project, Ganache is employed to:

- **Simulate Blockchain Operations**: Ganache allows for the creation and management of a private blockchain network where blockchain operations can be simulated. This includes creating blocks, transactions, and contracts in a test environment.

- **Develop and Test Smart Contracts**: Ganache supports the deployment and execution of smart contracts. This enables developers to test the functionality of smart contracts in a simulated environment before deploying them to a live blockchain.

- **Analyze Performance and Behavior**: By using Ganache, developers can observe how the blockchain behaves under different conditions and analyze the performance of blockchain operations and smart contracts.

2.3.3 Smart Contracts in Blockchain

Smart contracts are self-executing contracts with the terms of the agreement directly written into code[14]. These contracts automate processes and ensure that transactions and agreements are executed based on predefined conditions. In this project, smart contracts are used to:

- **Manage Data Sharing**: Automate the process of data-sharing agreements, ensuring that data is shared and accessed according to the specified rules and conditions.

- **Track Metadata**: Record and manage metadata related to model updates, data provenance, and transaction timestamps. This includes details such as which node submitted an update and when it was submitted.

- **Implement Incentive Mechanisms**: Define and execute reward systems for nodes that contribute valuable data or model updates. Incentives are distributed automatically based on the criteria set in the smart contracts.

As illustrated in Figure 2.1, these smart contracts operate within the Ganache Blockchain Environment, facilitating the automation and management of data sharing, metadata tracking, and incentive mechanisms within a decentralized framework.

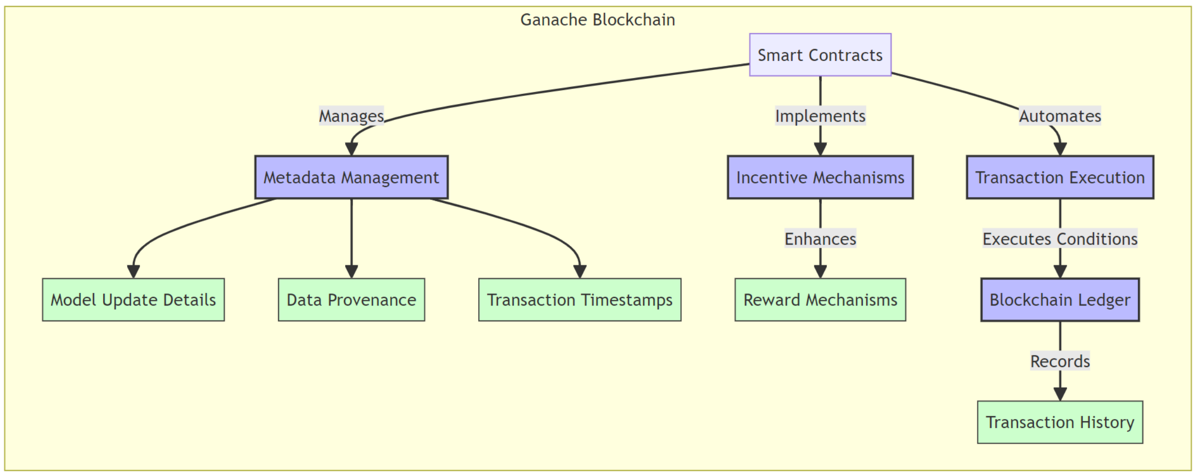


Figure 2.1: Interaction of Smart Contracts within the Ganache Blockchain Environment.

2.4 InterPlanetary File System (IPFS)

The InterPlanetary File System (IPFS) is a distributed file system designed to make the web more resilient, efficient, and secure by decentralizing data storage and retrieval [26]. IPFS addresses several limitations of traditional web infrastructure by creating a peer-to-peer network for storing and sharing files.

2.4.1 Overview of IPFS

IPFS aims to improve data management and retrieval through a decentralized approach. Instead of relying on centralized servers, IPFS distributes data across a network of nodes, which ensures that files are not subject to single points of failure and enhances both the resilience and efficiency of data retrieval. The key components and concepts of IPFS include:

- **Decentralization:** Data is stored across multiple nodes, which mitigates the risks associated with central server failures and provides a more robust system [26].
- **Content Addressing:** Files are identified by their content rather than their location. This is achieved through cryptographic hashing, which ensures that each file has a unique identifier based on its content [26].
- **Versioning and Immutability:** IPFS supports versioning of files and ensures that data cannot be altered once it has been uploaded. This is crucial for maintaining the integrity of data and tracking changes over time [26].

2.4.2 File Chunking and Cryptographic Hashing

When a file is uploaded to IPFS, it undergoes a process called chunking, where the file is split into smaller, manageable chunks. Each chunk is typically a few kilobytes in size. This chunking process is crucial as it allows IPFS to efficiently manage, distribute, and retrieve large files across a distributed network.

2.4.3 Pinata IPFS: Enhancing Decentralized Storage

While IPFS provides a robust framework for decentralized storage, managing and ensuring the persistence of data within the IPFS network can be challenging for developers. This is where Pinata comes into play. Pinata IPFS offers an enhanced interface and set of tools specifically designed to facilitate the storage, management, and retrieval of data on IPFS. By providing a user-friendly API and robust content management features, Pinata allows developers to easily pin files to the IPFS network, ensuring that the content remains available and accessible over time [27].

Pinata is particularly advantageous for developers working with decentralized applications (dApps) and blockchain projects, where data immutability and accessibility are paramount. It simplifies the process of interacting with IPFS, enabling users to manage their data through a centralized dashboard while still benefiting from the underlying decentralized infrastructure of IPFS.

2.4.4 Content Identifiers (CIDs) and Merkle DAG

Each chunk generated from the file is hashed using a cryptographic hash function, typically SHA-256, resulting in a unique identifier known as a Content Identifier (CID). These CIDs are used to locate and retrieve the chunks from the IPFS network.

The chunks are then arranged into a Merkle Directed Acyclic Graph (Merkle DAG). In this structure, each node (representing a chunk) is linked to its CID. The Merkle DAG ensures efficient verification of data integrity, where any change in any chunk of the file results in a completely different root hash, making it easy to detect tampering [26].

2.4.5 Distributed Storage

Once the file is chunked and hashed, the chunks are distributed across multiple nodes within the IPFS network. These nodes can be located anywhere globally, ensuring no single entity controls the storage, thereby increasing the resilience and integrity of the data.

2.4.6 Merkle Tree in IPFS

A Merkle tree, or more precisely a Merkle DAG in IPFS, is a data structure used to ensure data integrity and efficient data retrieval. The structure of the Merkle tree in IPFS can be described as follows:

- **Leaves:** The leaf nodes represent the individual chunks of the file, each identified by its CID.
- **Intermediate Nodes:** The intermediate nodes contain hashes of their child nodes, combining the CIDs from the lower levels of the tree.
- **Root Node:** The root node is the top-most node in the tree, representing the entire file. The hash of this node is the CID used to retrieve the file from IPFS.

The Merkle tree structure allows IPFS to efficiently verify the integrity of the data. Any alteration in a chunk will propagate up the tree, altering the root hash and making tampering easy to detect. As shown in Figure 2.2, this structure is crucial for maintaining the security and reliability of data stored and retrieved through the IPFS network.

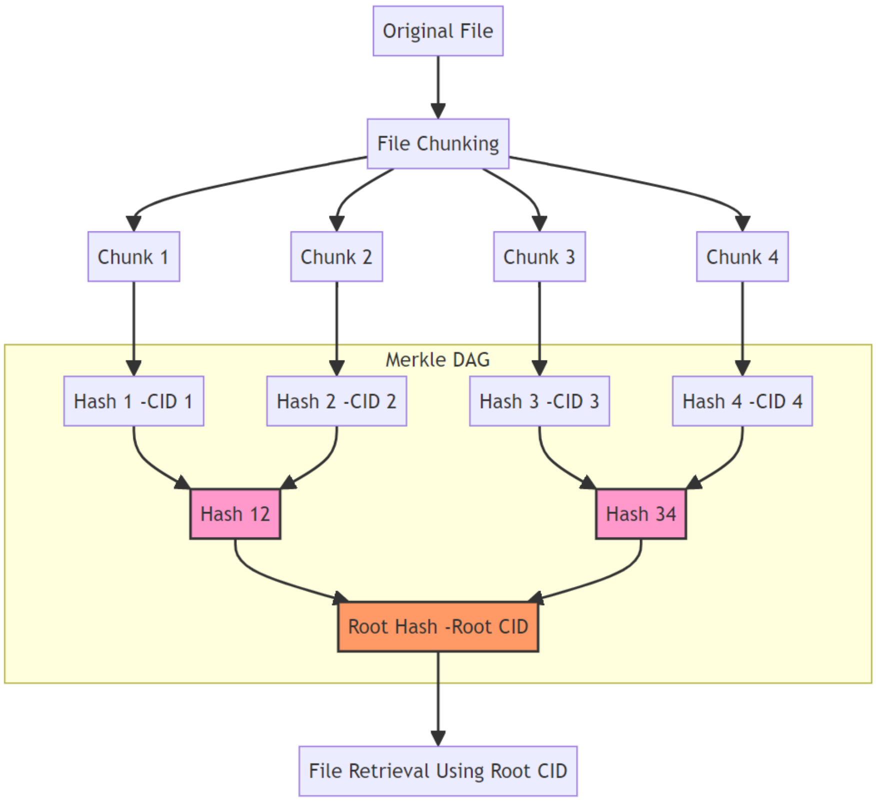


Figure 2.2: IPFS Data Storage and Retrieval Mechanism with Merkle DAG

2.4.7 Data Retrieval

When a user or node requests data stored on IPFS, the system uses the unique cryptographic hash assigned to each file (or file chunk) to locate the data across the distributed IPFS network. This process is decentralized, meaning that the data can be retrieved from the node or nodes closest to the requester, minimizing latency.

The retrieval process, illustrated in Figure 2.2, involves the following steps:

- **Hash Lookup**: The requester uses the cryptographic hash, known as a Content Identifier (CID), to query the IPFS network. Nodes that hold the corresponding data chunk respond to the request.

- **Data Assembly**: Once all chunks of the file are retrieved, IPFS reassembles the data in its original form. This ensures that even if the data is distributed across multiple nodes, the requester can retrieve the complete file seamlessly.

This process guarantees that data retrieval from IPFS is both efficient and secure, leveraging the strengths of a distributed network for fast and reliable access to information [26].

2.4.8 Pinata IPFS in Decentralized Storage Solutions

Pinata IPFS provides a scalable and user-friendly interface for the InterPlanetary File System (IPFS), which is a decentralized storage protocol. IPFS is designed to store and share hypermedia in a distributed network, thus eliminating the need for centralized servers. Pinata enhances IPFS by offering tools for efficient content management, making it ideal for decentralized applications (dApps) and blockchain projects. By leveraging Pinata, users can easily pin files to IPFS, ensuring data persistence and accessibility across the network [27].

2.5 Locust

Locust is an open-source tool used for performance testing and load testing of web applications and systems. It helps simulate multiple users interacting with a system to evaluate its performance under various loads. Locust is known for its ease of use, flexibility, and ability to scale tests across multiple machines [28].

2.5.1 Key Features

- **Python-Based Testing:** Tests are written in Python, making it easy to script complex user behaviors.
- **Distributed Load Testing:** Supports running tests from multiple machines to simulate high user loads.
- **Real-Time Metrics:** Provides live feedback on performance, including response times and request rates.

Locust will be used to test the performance of the integrated AI models, blockchain, and IPFS systems by simulating various load scenarios. This will help identify potential performance issues and ensure the system's reliability under realistic conditions.

2.6 Internet of Vehicles (IoV)

The Internet of Vehicles (IoV) represents a significant evolution in intelligent transportation systems, expanding upon traditional Vehicular Ad Hoc Networks (VANETs) by integrating vehicles, infrastructure, and various connected devices into a unified, internet-like network. This interconnected environment enables real-time data exchange, aiming to enhance traffic safety, efficiency, and the overall driving experience [4]. The IoV architecture typically includes several layers: the perception layer for gathering environmental data, the network layer for facilitating communication between vehicles and infrastructure, and the application layer where services like traffic management, autonomous driving, and smart parking are implemented [4].

IoV applications are diverse, ranging from real-time traffic management that optimizes traffic flow and reduces congestion, to autonomous driving systems that rely on continuous communication with surrounding infrastructure and other vehicles. Additionally, IoV supports smart parking solutions, which help drivers locate available parking spaces more efficiently, and provides enhanced infotainment services tailored to passenger needs [9].

However, the widespread adoption of IoV comes with significant security challenges. The continuous exchange of sensitive information, such as vehicle locations and user data, raises serious privacy concerns [29]. Moreover, IoV systems are increasingly vulnerable to cybersecurity threats, including Distributed Denial of Service (DDoS) attacks and unauthorized data access, which can compromise the safety and reliability of the network

[6]. Effective trust management between vehicles, infrastructure, and service providers is also critical to ensure the integrity and security of IoV operations [30]. Addressing these challenges is essential for the successful deployment and scalability of IoV systems.

2.7 Artificial Intelligence (AI) in Traffic Management and Security

Artificial Intelligence (AI) plays a pivotal role in enhancing traffic management, particularly in urban areas where congestion and traffic-related issues are prevalent. Deep Learning (DL) techniques, a subset of AI, have demonstrated superior capabilities in forecasting traffic congestion compared to traditional methods[19]. The ability of DL models to process and analyze large volumes of data enables them to predict traffic patterns with high accuracy, which is crucial for mitigating congestion and improving traffic flow in urban environments.

Several layers of artificial neurons are used in deep learning approaches to model complex patterns found in the data. These models take the input feature vector, which includes various factors like traffic density, time, and location, and process it through successive layers of transformations. Each layer applies a weighted sum of the inputs, adds a bias, and passes the result through an activation function, ultimately producing a predicted output.

DL techniques, such as Vision Transformers (VTs) combined with Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, effectively capture spatiotemporal characteristics of traffic data[19]. These models consider various factors, including time, location, and traffic density, to predict future traffic conditions. Additionally, integrating external factors like weather conditions into these models further improves their prediction accuracy, making them more reliable for real-world applications[20]. However, the complexity of these models, coupled with the need for continuous updates and high-quality data, poses significant challenges[10].

Reinforcement Learning (RL) is another AI technique that has shown considerable promise in traffic signal control. The fundamental RL model is based on the Bellman equation, which is defined as:

$$Q(s, a) = r + \gamma \max_{a'} Q(s', a') \tag{2.1}$$

where $Q(s, a)$ represents the quality of action a in state s, r is the reward, and γ is the discount factor.

Unlike traditional traffic signal systems that rely on fixed schedules or simple adaptive methods, RL-based systems learn and adapt in real-time by interacting with the environment[31]. This dynamic approach allows for more efficient traffic signal control, reducing congestion and improving overall traffic flow. When combined with deep learning, RL systems can handle complex scenarios involving multiple intersections and varying traffic conditions[31]. Multi-Agent Deep Reinforcement Learning (MADRL) is particularly effective in such scenarios, as it enables the coordination of multiple traffic signals to optimize traffic flow across a network[22]. Despite its advantages, MADRL introduces additional computational complexity and requires significant resources to implement effectively[22].

Explainable AI (XAI) is a branch of AI that aims to increase the understanding and transparency of AI models' decision-making processes. In the context of traffic management, XAI is particularly important as it helps in building trust in AI systems by providing insights into how and why certain decisions are made. This is crucial when deploying AI in critical applications like traffic management, where understanding the rationale behind decisions can improve stakeholder confidence and facilitate better decision-making processes. However, the implementation of XAI techniques often comes with challenges related to scalability and the complexity of providing explanations that are both accurate and comprehensible [16].

Table 2.1: Comparison of AI Techniques in Traffic Management

AI Technique	**Advantages**	**Challenges**
Deep Learning (DL)	High prediction accuracy	High computational cost
Reinforcement Learning (RL)	Adaptive signal control	Complex multi-agent scenarios
Explainable AI (XAI)	Transparency	Limited scalability

In the realm of vehicular network security, Machine Learning (ML) enhances the ability of IoV systems to detect and mitigate cyber threats. The increasing reliance on interconnected systems in vehicular networks has made them more vulnerable to cyberattacks, including Distributed Denial of Service (DDoS) attacks and unauthorized access[6]. ML techniques are employed to develop Intrusion Detection Systems (IDSs) that can autonomously monitor network traffic and detect anomalies indicative of a security breach[6]. For example, in Vehicular Ad Hoc Networks (VANETs), ML models such as gradient-boosted trees (GBT), Logistic Regression (LR), and Random Forest (RF) have demonstrated high accuracy in identifying and mitigating DDoS attacks[32]. However, the dynamic nature of vehicular environments and the need for real-time processing pose significant challenges to the effectiveness of ML-based security solutions[33].

2.8 Blockchain Technology in IoV Systems

Blockchain technology, first introduced by Satoshi Nakamoto as the underlying technology of Bitcoin, offers a robust framework for addressing the challenges associated with data security, privacy, and trust management in IoV systems [24]. A core component of blockchain's security is its use of cryptographic hash functions, represented as:

$$h = H(x) \tag{2.2}$$

where H is the hash function and x is the input data, such as a block of transactions.

As IoV networks involve the continuous exchange of sensitive information, ensuring the integrity and security of this data is paramount. Blockchain's decentralized and immutable ledger system provides a high level of security, making it an ideal solution for managing data exchanges within vehicular networks[29].

One of the key advantages of blockchain technology is its ability to create a tamper-proof and traceable record of transactions. This feature is particularly useful in IoV systems, where the authenticity and accuracy of data are critical[29]. For instance, blockchain can be used to record and verify vehicle identities, ensuring that only authorized vehicles can access certain network services or infrastructure[29]. Moreover, blockchain's decentralized nature eliminates the need for a central authority, reducing the risk of a single point of failure and enhancing the overall resilience of the network[30].

To address high mobility and communication costs in vehicular networks, an efficient multi-sharding protocol has been proposed, reducing communication complexity without compromising blockchain security[29]. The integration of attribute encryption methods and collaborative on-chain and off-chain data storage solutions further enhances data security and reduces system overhead[34]. Utilizing smart contracts for log tracking ensures that data-sharing records are transparent and immutable, crucial for maintaining real-time identity tracking and securing data exchanges[35]. These blockchain-based approaches significantly improve system security, data privacy, and communication efficiency, making them suitable for real-world vehicular applications[35].

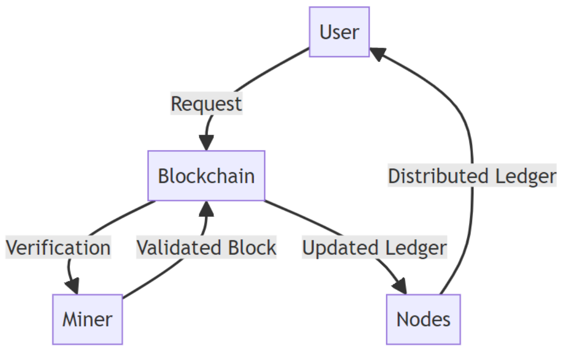


Figure 2.3: Blockchain Architecture for IoV Systems

Blockchain's role in trust management within IoV systems is equally important. Traditional centralized trust management systems often struggle to meet the dynamic and decentralized needs of vehicular networks. Blockchain technology, with its ability to provide a decentralized framework, offers enhanced resiliency, traceability, and tamper-proof data verification[30][36]. In such systems, blockchain can be used to manage trust between different entities, such as vehicles, infrastructure, and service providers, ensuring that all parties involved can trust the data being shared[30].

Incentive mechanisms facilitated by the use of blockchain play an important part in encouraging data sharing within IoV systems. Traditional data-sharing models often rely on centralized third-party platforms, which can be vulnerable to data tampering and lack transparency[15]. Blockchain-based incentive mechanisms, on the other hand, provide a decentralized solution that ensures data security and transparency while giving users control over their data[15]. Smart contracts, which are self-executing contracts with the terms of the agreement directly written into code, automate the process of reward distribution, incentivizing participants to contribute high-quality data by linking rewards to the quality and quantity of data shared[15][14]. This approach is essential for maintaining an active and trustworthy network of participants within the IoV ecosystem.

2.9 Decentralized Storage in IoV

The integration of blockchain technology with decentralized storage solutions, such as the InterPlanetary File System (IPFS), presents a robust framework for enhancing data security, trust, and efficiency within Internet of Vehicles (IoV) systems. Blockchain's decentralized and immutable ledger system, as originally proposed by Nakamoto [24], is particularly beneficial for managing the vast amounts of data generated by IoT devices, offering significant security benefits for low-power IoT networks that are often vulnerable to Distributed Denial of Service (DDoS) attacks[37].

Blockchain networks are typically categorized into public (permissionless), private (permissioned), and hybrid types, each offering distinct security and access control features[38]. Public blockchains provide open access and decentralization, while private blockchains restrict access to authorized participants, enhancing security and control[38]. The use of decentralized storage solutions like IPFS within these blockchain networks further enhances data sharing efficiency and security by distributing data across multiple nodes[8].

IPFS operates as a peer-to-peer distributed file system, meaning that data is stored across multiple nodes rather than on a centralized server. This approach not only improves data security by reducing the likelihood of data breaches but also enhances data retrieval speeds by allowing users to access information from the nearest node[8]. The integration of IPFS with blockchain technology further strengthens the security and efficiency of IoV systems, making it a valuable tool for managing large volumes of data generated by connected vehicles.

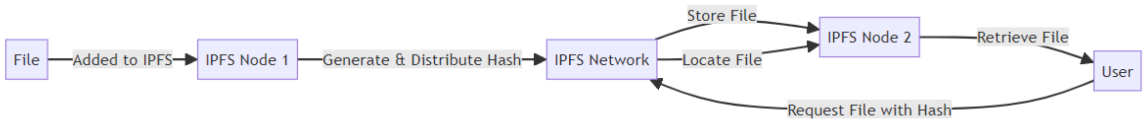


Figure 2.4: IPFS Architecture for Decentralized Data Storage

As shown in Figure 2.4, IPFS architecture enables decentralized data storage, which is critical for ensuring data integrity and availability in IoV systems. The distributed nature of IPFS, combined with blockchain, addresses the scalability and security challenges often associated with centralized storage solutions.

Decentralized Storage Networks (DSNs) leverage blockchain technology to enhance security, trust, and transparency in data storage. These networks allow users to rent out unused storage space, creating a peer-to-peer system that is facilitated by blockchain[13]. Incentive mechanisms, such as those based on native tokens and smart contracts, encourage participation in these networks, while novel consensus mechanisms like Proof-of-Spacetime reward users for providing storage space[13]. This approach addresses trust and security challenges, making blockchain an ideal solution for secure data management.

Scalability and performance are critical concerns in blockchain storage architecture. Experimental test beds with multiple blockchain nodes have been used to evaluate scalability issues such as network latency, block transaction rate, and bandwidth usage[38]. Two-tier blockchain frameworks, which manage transactions locally and globally, have been proposed as a solution to these scalability challenges[38]. The integration of blockchain with emerging technologies like Federated Learning can further enhance data privacy and security, particularly in sensitive domains like healthcare and the Industrial Internet of Things (IIoT)[13].

2.10 Integration of AI and Blockchain for Enhanced IoV Systems

The integration of Blockchain and Artificial Intelligence technologies represents a significant advancement in the development of secure and efficient IoV systems. By combining the data processing capabilities of AI with the security features of blockchain, these technologies can address many of the challenges associated with managing large-scale, decentralized networks[7][39].

AI plays a crucial role in analyzing and processing the vast amounts of data generated by IoV systems. Machine learning models, for example, can be trained to predict traffic patterns, detect anomalies, and optimize network performance in real-time[7]. However, the centralized nature of traditional AI systems can pose risks to data privacy and security. This is where the use of blockchain technology becomes significant. By decentralizing data storage and processing, blockchain can ensure that sensitive information is protected from unauthorized access and tampering[7][39]. Moreover, blockchain's immutable ledger can be used to record and verify AI model updates, ensuring that the models are accurate and trustworthy.

Decentralized AI approaches, such as Federated Learning (FL), are particularly effective in IoV systems. The model updates in FL can be aggregated as:

$$w_{t+1} = w_t - \eta \sum_{i=1}^{n} \frac{n_i}{N} \nabla \mathcal{L}(w_t; x_i) \tag{2.3}$$

where w_t represents the model parameters at time t, and $\mathcal{L}$ is the loss function.

FL allows for the training of AI models across multiple decentralized nodes, ensuring that sensitive data remains local while still contributing to a global model[3]. This approach not only enhances data privacy but also improves the scalability and resilience of IoV systems, as it reduces the reliance on central servers and mitigates the risks associated with central points of failure[3]. The combination of FL and blockchain technology creates a powerful framework for developing intelligent, secure, and scalable IoV systems.

The convergence of AI and blockchain also facilitates privacy-preserving machine learning, particularly in sensitive domains like healthcare and the Industrial Internet of Things (IIoT)[3]. By integrating FL with blockchain, organizations can ensure that machine learning models are trained on decentralized data while maintaining a secure and immutable record of model updates[3]. This approach is ideal for privacy-preserving traffic management systems, where data privacy is a major concern[3].

Table 2.2: Blockchain and AI Integration in IoV

Technology	**Application in IoV**	**Benefits**
AI	Traffic prediction, anomaly detection	Real-time data analysis
Blockchain	Data security, trust management	Decentralization, immutability
Federated Learning (FL)	Decentralized AI model training	Enhanced privacy, scalability

2.11 Traffic Prediction Datasets

Traffic prediction datasets are crucial for developing and evaluating models that forecast traffic conditions in Intelligent Transportation Systems (ITS). Real-world datasets enable a deeper understanding of traffic complexities and support the creation of robust models that can handle incomplete data, which is a common issue in traffic prediction[40]. The LargeST dataset, which includes data from 8,600 sensors across California over five years, is a significant resource for studying long-term traffic patterns and training deep learning models[41]. The rich metadata in LargeST enhances the reliability and interpretability of traffic data, providing additional context for improving model predictions[41].

Table 2.3: Key Features of LargeST Dataset

Feature	Description	Importance
Number of Sensors	8,600	High coverage across California
Duration	5 years	Long-term traffic pattern analysis
Metadata	Highway categories, lanes	Enhances model reliability

User privacy is a critical concern in IoV systems, particularly in the context of blockchain-based vehicular networks. Blockchain technology offers a decentralized solution to these concerns by storing records in an immutable ledger, which eliminates the need for a central authority and enhances trust among network participants[42]. Privacy preservation techniques such as pseudonyms, which associate user identities with addresses or public keys that can be changed for each transaction, help maintain user anonymity within the network[43]. Additionally, blockchain-based monetary incentive schemes and distributed key-policy attribute-based encryption further enhance privacy by protecting user locations and securely controlling access to forensic data[43].

In conclusion, the integration of AI, blockchain, and decentralized storage solutions like IPFS represents a transformative approach to enhancing the security, efficiency, and scalability of IoV systems. Federated Learning (FL) further enhances data privacy by enabling decentralized training of AI models, ensuring that sensitive data remains within the local environment. Continued research and innovation are essential to address existing limitations and enhance the scalability and performance of these systems, ensuring their viability in real-world applications[42][43].

2.12 Summary

This chapter explored the evolution of the Internet of Vehicles (IoV), focusing on how it enhances traffic management and safety through the integration of advanced technologies like Artificial Intelligence (AI), blockchain, and decentralized storage systems such as the InterPlanetary File System (IPFS). AI techniques, including Deep Learning (DL) and Reinforcement Learning (RL), were highlighted for their effectiveness in predicting traffic congestion and optimizing traffic signals, while Explainable AI (XAI) was emphasized for increasing transparency in decision-making processes. The chapter also examined blockchain's role in securing data exchanges, managing trust, and supporting decentralized storage, which is critical for handling the vast amounts of data generated by IoV systems. Additionally, the integration of AI and blockchain was discussed as a promising approach to developing secure, scalable, and efficient IoV systems, with Federated Learning (FL) further enhancing data privacy. Finally, the chapter introduced performance testing tools like Locust, which will be crucial for assessing the reliability and scalability of these integrated systems under real-world conditions.

Chapter 3

Design & Implementation

3.1 System Architecture

The system architecture, depicted in Figure 3.1, is designed to support a decentralized, secure, and efficient Internet of Vehicles (IoV) environment. This architecture is divided into several interconnected layers, each serving a specific function within the overall system. The following sections provide a detailed explanation of each layer and how they were implemented in the system.

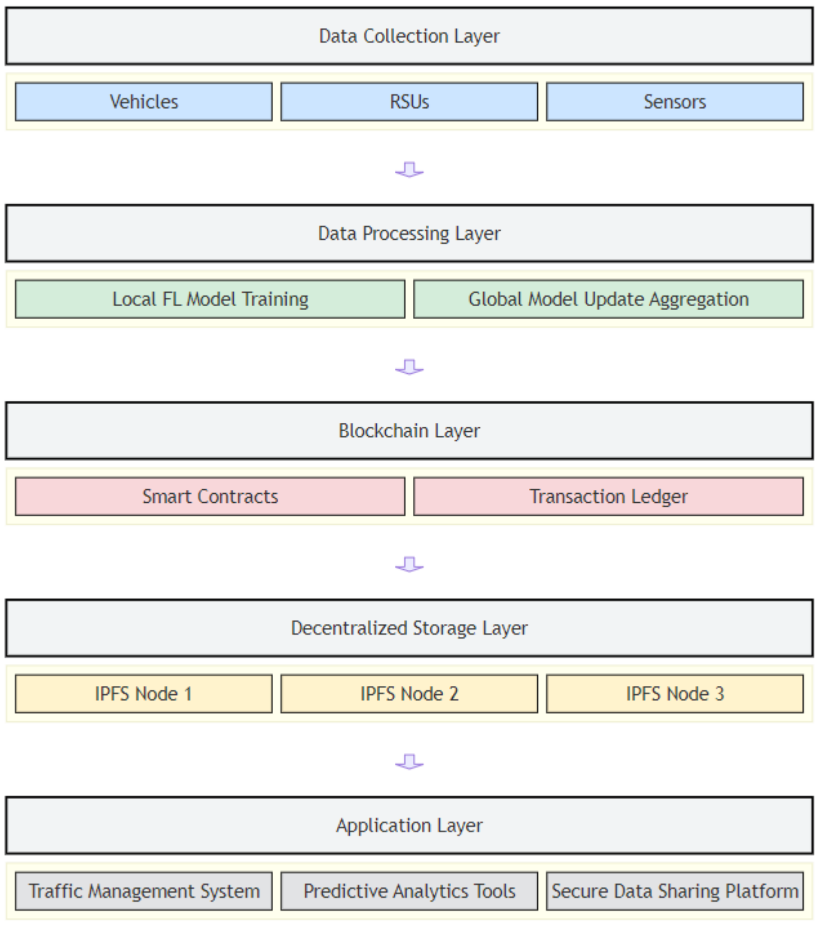


Figure 3.1: Overview of the System Architecture

3.1.1 Data Collection Layer

Description: The Data Collection Layer is the foundation of the IoV system, responsible for gathering data from various sources such as vehicles, Roadside Units (RSUs), and environmental sensors. The data collected includes traffic flow information, vehicle speed, location data, and environmental conditions. This layer ensures real-time data capture, providing a comprehensive view of the operating environment.

Implementation: Since the data is provided in a pre-collected format, the implementation of this layer involves loading and pre-processing the dataset. The dataset is loaded into the system, and pre-processing steps such as data cleaning, normalization, and feature selection are applied. These steps ensure that the data is in a suitable format for further processing in the next layer.

Data Flow: Once the data is pre-processed, it is forwarded to the Data Processing Layer for further analysis. This local pre-processing step is crucial for reducing the amount of data transmitted across the network and for enhancing the efficiency of the system.

3.1.2 Data Processing Layer

Description: The Data Processing Layer is central to the analytical capabilities of the system. It uses Federated Learning (FL) to process the data while preserving privacy. Each node (e.g., vehicle, RSU) trains an AI model locally without sharing raw data, which ensures that sensitive information is kept private.

Implementation: Local models are trained on each node using the collected and pre-processed data. The training process is decentralized, with each node independently developing its model. Once training is completed, the nodes send the model updates (rather than the raw data) to a central aggregator. The aggregator combines these updates to create a global model that reflects the collective learning of all nodes.

Data Flow: After local model training, the updates are sent to a central aggregator. The aggregator merges these updates to form a global model, which is then distributed back to the nodes. This approach ensures that all nodes benefit from the collective intelligence without compromising the privacy of individual datasets.

3.1.3 Blockchain Layer

Description: The Blockchain Layer is implemented to ensure the security, integrity, and transparency of the system. It leverages blockchain technology, particularly Ethereum-based smart contracts, to create a decentralized ledger where all transactions, including model registrations and data exchanges, are securely recorded.

Implementation:

- **Smart Contracts:** Multiple smart contracts are deployed to manage different aspects of the system:
  - `LocalModelRegistry` registers and tracks local models contributed by individual nodes.
  - `AggregatedModelRegistry` manages the registration of aggregated models created from multiple local models.

  - `ReputationSystem` rewards contributors based on the performance of their models.
  - `FileMetadataRegistry` stores metadata for files stored on IPFS, ensuring data is retrievable and its integrity can be verified.

- **Transaction Workflow:** Nodes register their models on the blockchain by submitting a transaction that includes the model's IPFS hash and other relevant metadata. This transaction is processed by the corresponding smart contract and recorded on the blockchain, ensuring that the registration is immutable and publicly verifiable.
- **Integrity Checks:** The system includes a feature for verifying the integrity of registered models by comparing stored hashes with provided hashes, ensuring data consistency and preventing tampering.

Data Flow: Every model registration, update, and reward distribution is recorded on the blockchain. These transactions are governed by smart contracts, which enforce validation rules and ensure that all actions are securely logged and cannot be altered.

3.1.4 Decentralized Storage Layer

Description: The Decentralized Storage Layer is responsible for the long-term storage of data within the system. It is implemented using the InterPlanetary File System (IPFS), a peer-to-peer distributed storage network that provides efficient and reliable storage without relying on a central server.

Implementation:

- **Model Storage:** After training, models are serialized, compressed, and uploaded to IPFS using the Pinata API[27]. This process generates a unique IPFS hash, which serves as the model's address in the IPFS network.
- **Model Retrieval:** To retrieve a model, the system uses the IPFS hash stored on the blockchain. The model is fetched from IPFS, decompressed, and deserialized for further use.
- **Metadata Management:** Metadata associated with the models, such as descriptions, contributor information, and IPFS hashes, is managed by the `FileMetadataRegistry` smart contract on the blockchain.
- **Data Integrity:** The decentralized nature of IPFS, along with cryptographic hashing, ensures that stored data is tamper-proof. Any alteration in the data would result in a different hash, easily detectable during retrieval.

Data Flow: Data (specifically machine learning models) is stored in a decentralized manner on IPFS. The corresponding metadata and IPFS hashes are recorded on the blockchain to ensure the data is easily retrievable and verifiable, maintaining data integrity over time.

3.1.5 Application Layer

Description: The Application Layer utilizes the processed data and the secure blockchain framework to deliver various services to the end-users. This layer includes applications such as real-time traffic management systems and secure data-sharing platforms that use the underlying data and models to optimize traffic flow and ensure privacy and trust among participants.

Implementation: Applications at this layer consume the data processed and secured in the lower layers. For instance, traffic management applications use the global model generated by the Data Processing Layer to predict traffic patterns and adjust traffic signals in real-time. Additionally, secure data-sharing platforms ensure that participants can share data without compromising privacy, leveraging the blockchain for transparent and trustworthy transactions.

Data Flow: Processed and secured data from the lower layers is fed into various applications for decision-making, analytics, and service delivery. These applications leverage the robustness of the underlying architecture to provide reliable and efficient services to end-users.

The overall integration and interaction of these layers within the system are crucial for achieving a decentralized, secure, and efficient IoV environment. Figure 3.2 provides a detailed view of the system architecture, highlighting how data flows from collection through processing, blockchain management, decentralized storage, and finally, to the application layer where it is utilized for real-time decision-making and secure data sharing.

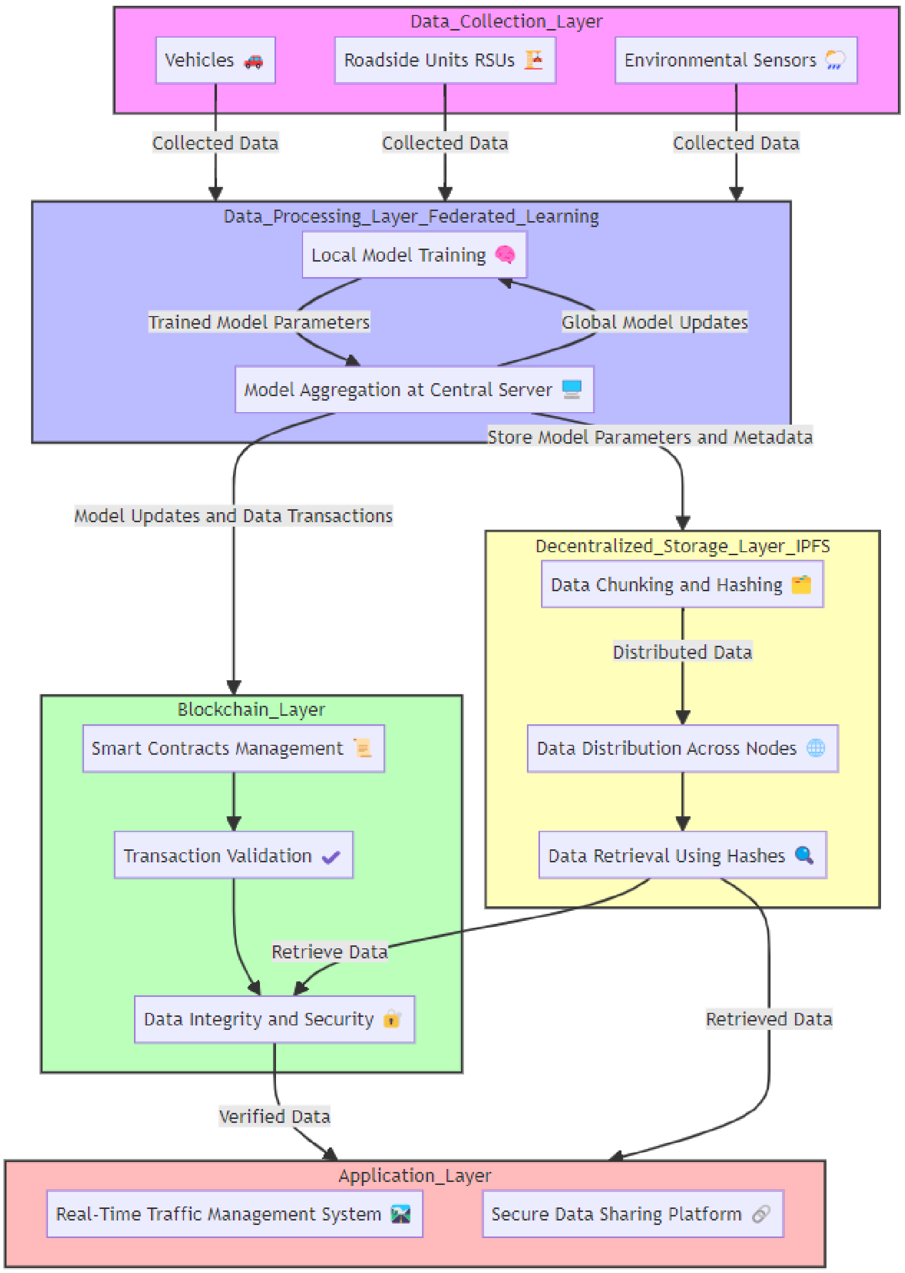

Data_Collection_Layer
Vehicles
Roadside Units RSUs
Environmental Sensors
Collected Data
Collected Data
Collected Data
Data_Processing_Layer_Federated_Learning
Local Model Training
Trained Model Parameters
Global Model Updates
Model Aggregation at Central Server
Store Model Parameters and Metadata
Model Updates and Data Transactions
Decentralized_Storage_Layer_IPFS
Data Chunking and Hashing
Distributed Data
Data Distribution Across Nodes
Data Retrieval Using Hashes
Blockchain_Layer
Smart Contracts Management
Transaction Validation
Retrieve Data
Data Integrity and Security
Retrieved Data
Verified Data
Application_Layer
Real-Time Traffic Management System
Secure Data Sharing Platform

Figure 3.2: System Architecture for IoV with Federated Learning and Blockchain Integration

3.2 Dataset

The Large-Scale Traffic and Weather Events (LSTW) dataset provides comprehensive traffic event data across the United States, including the state of Texas. The dataset includes records of various traffic incidents such as accidents, congestion, and road hazards, collected since August 2016. For this project, we focus on the subset of data specific to Texas. This dataset will enable an in-depth analysis of traffic patterns and safety concerns within the state, supporting applications such as predictive modeling, traffic management, and safety analysis.

3.2.1 Citation and Credit

The dataset is made available by Sobhan Moosavi[1] and is licensed under a Creative Commons license.

- **Citation:** Sobhan Moosavi, "Large-Scale Traffic and Weather Events Dataset (LSTW)," available at `https://smoosavi.org/datasets/lstw`.

#	Attribute	Description	Nullable
1	`EventId`	This is the identifier of a record.	No
2	`Type`	The type of an event; examples are accident and congestion.	No
3	`Severity`	The severity is a value between 0 and 4, where 0 indicates the least impact on traffic and 4 indicates a significant impact on traffic.	No
4	`TMC`	Each traffic event has a Traffic Message Channel (TMC) code which provides a more detailed description of the event.	No
5	`Description`	The natural language description of an event.	No
6	`StartTime (UTC)`	The start time of an event in UTC time zone.	No
7	`EndTime (UTC)`	The end time of an event in UTC time zone.	No
8	`TimeZone`	The US-based timezone based on the location of an event (eastern, central, mountain, and pacific).	No
9	`LocationLat`	The latitude in GPS coordinates.	Yes
10	`LocationLng`	The longitude in GPS coordinates.	Yes
11	`Distance (mi)`	The length of the road extent affected by the event.	Yes
12	`AirportCode`	The closest airport station to the location of a traffic event.	Yes
13	`Number`	The street number in the address record.	Yes
14	`Street`	The street name in the address record.	Yes
15	`Side`	The relative side of a street (R/L) in the address record.	Yes
16	`City`	The city in the address record.	Yes
17	`County`	The county in the address record.	Yes
18	`State`	The state in the address record.	Yes
19	`ZipCode`	The zipcode in the address record.	Yes

Table 3.2: Description of the Traffic Events Dataset Columns [1]

3.3 Data Processing

The data processing workflow for this project involved multiple steps to clean, engineer features, select relevant features, and visualize the data. Below is a detailed description of each stage.

3.3.1 Data Processing Workflow

The entire data processing workflow is depicted in the flowchart below:

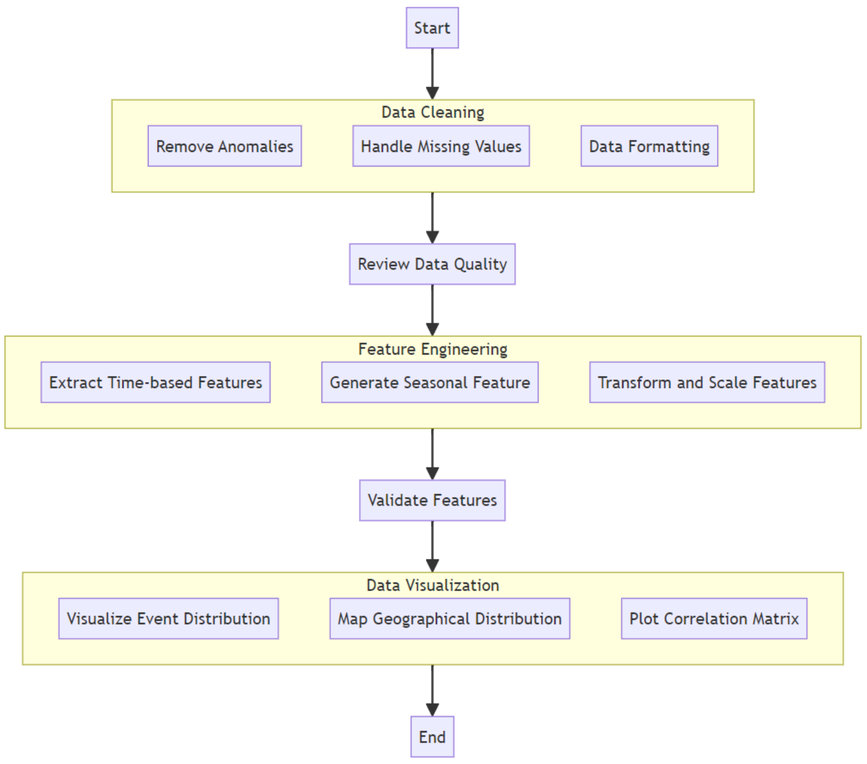


Figure 3.3: The entire data processing workflow is depicted in the flowchart.

3.3.2 Data Cleaning

The first step in data processing was to clean the raw data, which involved several tasks to ensure the dataset was in a suitable format for analysis:

- **Removing Anomalies:** The dataset initially contained 1,500,000 records across 21 columns. To improve data quality, the cleaning process began with the removal of anomalies. For instance, any rows with null values in critical columns like 'Description' were dropped, resulting in the removal of 32 records. Additionally, columns such as 'Number', which contained a large number of missing values, were entirely removed from the dataset. This step ensured that the data did not include misleading or incomplete entries that could skew the analysis.

- **Handling Missing Values:** Missing data was addressed through a combination of imputation and deletion strategies. For fields like 'ZipCode' and 'City', which had missing values, a mapping based on related fields such as 'AirportCode' was used to fill in the gaps. Specifically, the most frequent non-missing value for 'ZipCode' corresponding to each 'AirportCode' was used for imputation. Similarly, the 'TimeZone' field, which had 457 missing values, was filled by mapping it to the most frequent 'TimeZone' associated with each 'City' or 'AirportCode'. Rows where neither mapping nor imputation was feasible were removed from the dataset to ensure consistency.

- **Formatting:** Consistency in the dataset, particularly with date and time formats, was crucial for the subsequent analysis. Therefore, after imputing missing time zones, the 'StartTime' and 'EndTime' fields were standardized to the UTC time zone to ensure uniformity across all records. This step was essential for aligning events chronologically across different time zones.

After completing the data cleaning process, the dataset size was slightly reduced to 1,499,968 records across 20 columns. This cleaning process ensured that the remaining data was accurate, consistent, and ready for further

3.3.3 Feature Engineering

The feature engineering process involved creating new features and transforming existing ones to better represent the data for modeling purposes. Below are the detailed steps taken:

- **Time-based Features:**
  - `Hour`: Extracted from the `StartTime(UTC)` field, this feature represents the hour of the day when the event started.
  - `DayOfWeek`: Derived from the `StartTime(UTC)`, this feature indicates the day of the week on which the event occurred (0 = Monday, 6 = Sunday).
  - `Month`: Extracted from `StartTime(UTC)`, this feature indicates the month during which the event occurred.
  - `Duration`: Calculated as the difference between `EndTime(UTC)` and `StartTime(UTC)`, representing the duration of the event in hours.
- **Seasonal Feature:**
  - `Season`: A categorical feature representing the season during which the event occurred, derived from the month (e.g., Winter, Spring, Summer, Fall).
- **Label Encoding:**
  - `City`: The `City` field was label-encoded to convert categorical city names into numerical values.
  - `County`: Similarly, the `County` field was label-encoded.
- **One-Hot Encoding:**
  - Categorical features with low cardinality such as `TimeZone`, `Side`, `State`, and `Season` were one-hot encoded to create binary variables.
- **Numerical Feature Scaling:**
  - Numerical features like `Duration` and `ZipCode` were normalized using a standard scaler to ensure that the data was on a comparable scale.
- **Timestamp Conversion:**

  - The `StartTime(UTC)` was converted into a timestamp to facilitate temporal analysis.

- **Dropping Irrelevant Features:**
  - Features such as `EventId`, `Type`, `TMC`, `Description`, `EndTime(UTC)`, `Distance(mi)`, `AirportCode`, and `Street` were dropped from the dataset as they were not relevant for further analysis.

This extensive feature engineering process transformed the raw data into a more structured format, enhancing its suitability for predictive modeling and analysis.

3.3.4 Feature Selection

After the feature engineering process, a final set of features was selected for analysis and modeling. The selected features are summarized in Table 3.3.

Table 3.3: Final Selected Features after Feature Engineering

Feature Name	**Description**
`Hour`	Hour of the day when the event started.
`DayOfWeek`	Day of the week when the event occurred.
`Month`	Month during which the event occurred.
`Duration`	Duration of the event in hours.
`Season`	Season during which the event occurred (Winter, Spring, Summer, Fall).
`City`	Encoded city name where the event occurred.
`County`	Encoded county name where the event occurred.
`TimeZone`	One-hot encoded timezone of the event location.
`Side`	One-hot encoded side of the street (R/L).
`State`	One-hot encoded state where the event occurred.
`Duration`	Normalized duration of the event.
`ZipCode`	Normalized zipcode of the event location.
`timestamp`	Timestamp representing the start time of the event in Unix format.

The selected features represent a comprehensive set of variables that capture the temporal, geographical, and categorical aspects of the traffic events, making them well-suited for further analysis and predictive modeling.

3.3.5 Data Visualization

To validate the preprocessing steps and gain insights into the dataset, several visualizations were created. These visualizations not only helped confirm the effectiveness of the cleaning process but also provided a deeper understanding of the underlying patterns and relationships in the data.

- **Distribution of Event Types:** The distribution of different types of traffic events was visualized to understand the frequency of each event type within the dataset. As shown in Figure 3.4, this bar chart highlights the prevalence of each event category, such as congestion, accidents, and road closures. This distribution helps identify which types of events are most common, ensuring that the dataset includes a diverse range of event types, which is crucial for building a robust predictive model.

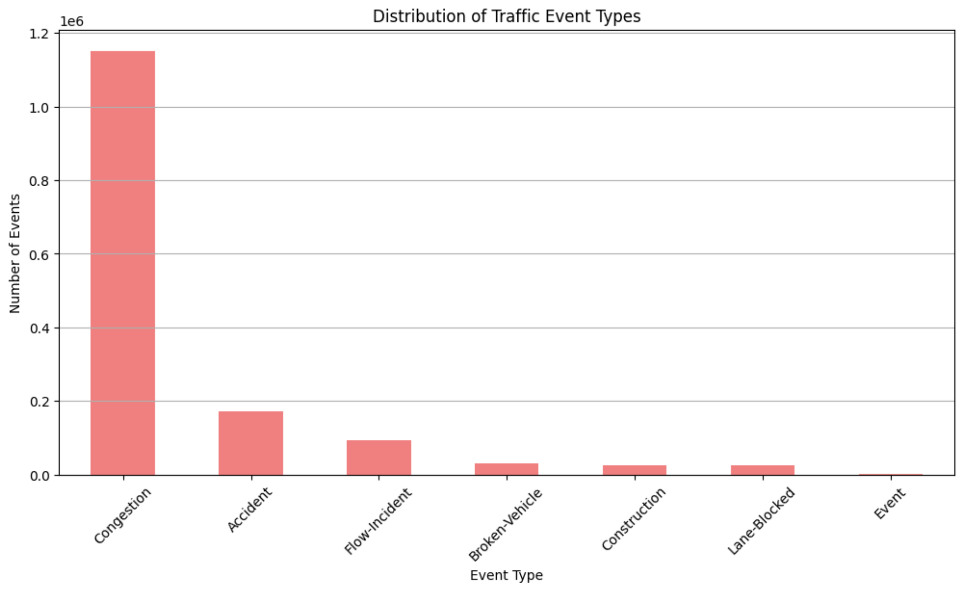


Figure 3.4: Distribution of Traffic Event Types

- **Event Distribution by Hour:** To explore the temporal patterns of traffic events, the number of events per hour was plotted, as shown in Figure 3.5. This bar chart illustrates how event occurrences vary throughout the day, revealing peak hours when traffic events are most likely to happen. Such patterns are crucial for time-based predictive modeling, as they highlight the times of day when traffic incidents are more frequent, potentially guiding resource allocation for traffic management and emergency response.

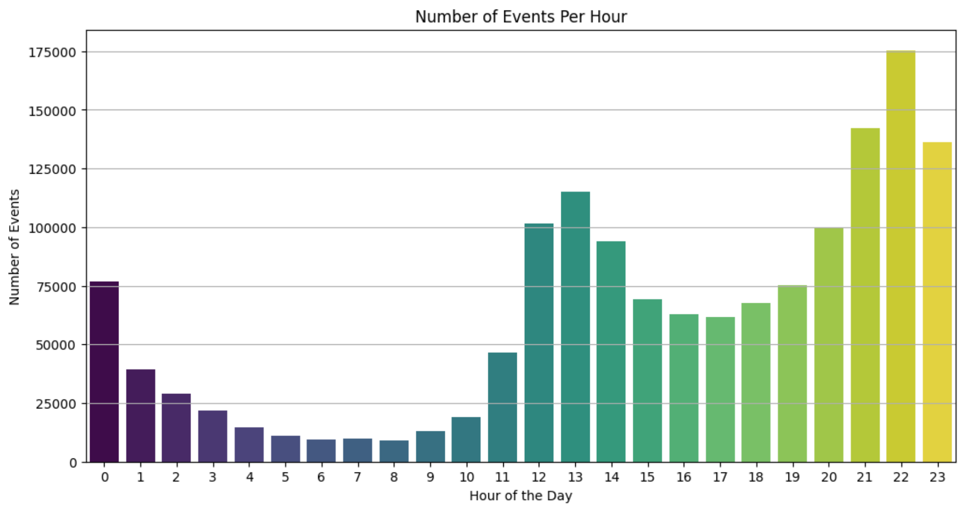


Figure 3.5: Number of Events Per Hour After Preprocessing

- **Event Distribution by Day of the Week:** The distribution of events by day of the week, depicted in Figure 3.6, provides insights into weekly patterns in traffic incidents. This bar chart reveals whether certain days, such as weekends or weekdays, are associated with higher or lower frequencies of events. Understanding these patterns is important for planning weekly traffic management strategies and for anticipating potential traffic disruptions based on the day of the week.

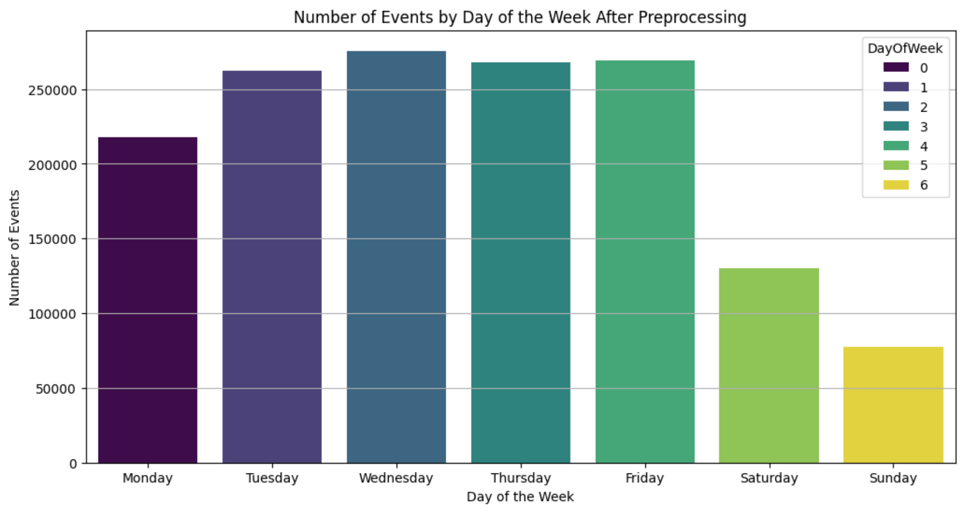


Figure 3.6: Number of Events by Day of the Week After Preprocessing

- **Geographical Distribution of Events:** A scatter plot was created to show the geographical distribution of traffic events, as seen in Figure 3.7. This plot is color-coded by the severity of the events, allowing for an analysis of how traffic incidents are distributed across different regions. This visualization is particularly useful for identifying hotspots—areas with a high concentration of severe events—which can inform targeted interventions and infrastructure improvements.

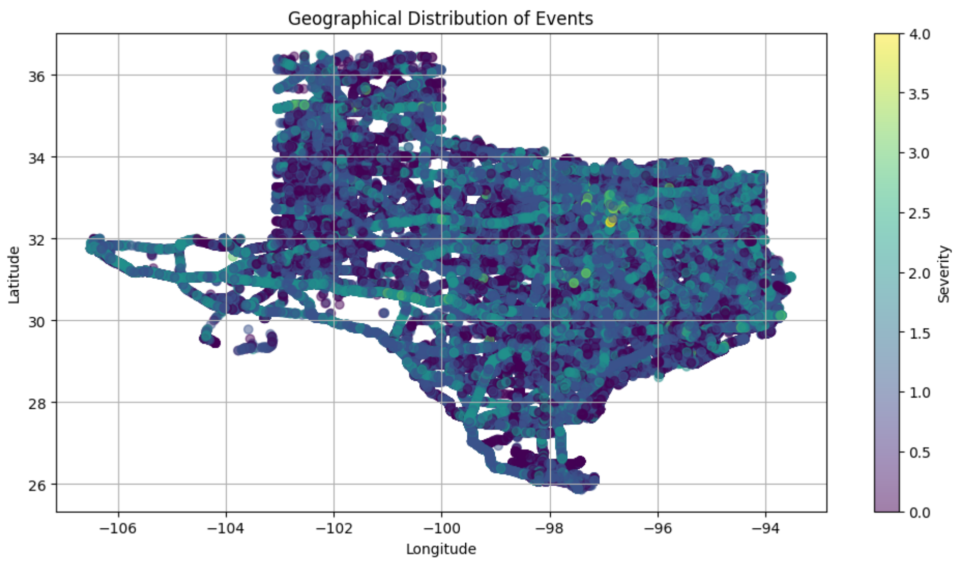


Figure 3.7: Geographical Distribution of Events After Preprocessing

- **Correlation Matrix:** A heatmap of the correlation matrix was generated to assess the relationships between numeric features in the dataset, as shown in Figure 3.8.

This visualization helps identify which features are strongly correlated, either positively or negatively. Understanding these correlations is essential for feature selection and engineering, as highly correlated features might indicate redundancy, while weakly correlated features might be more valuable for predictive modeling.

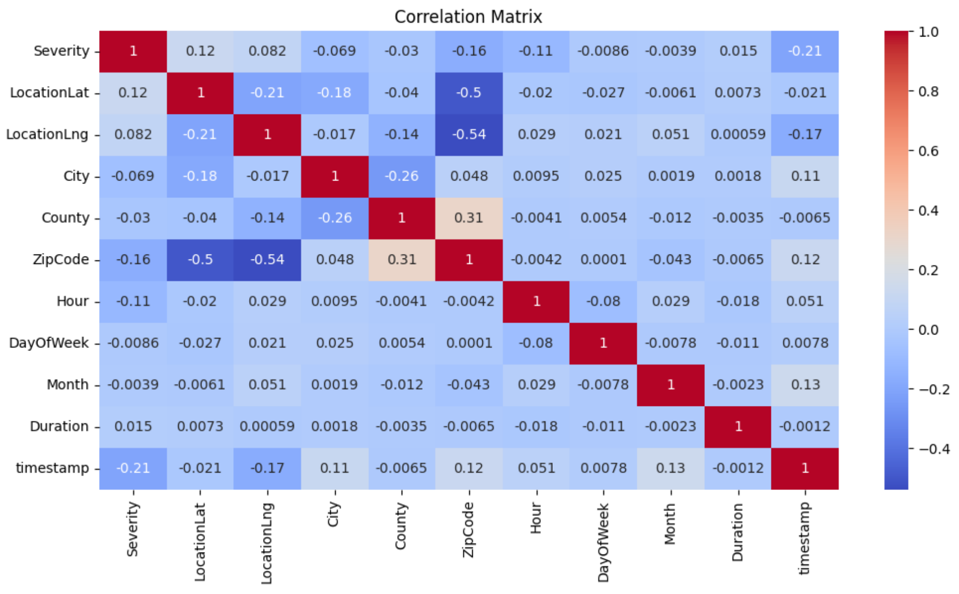


Figure 3.8: Correlation Matrix of Selected Features

These visualizations not only validate the preprocessing steps but also provide a solid foundation for subsequent analysis and modeling. By exploring the distributions, correlations, and patterns within the data, these visual tools help uncover insights that are crucial for effective decision-making and model development.

3.4 AI Model Development

The AI model development for traffic congestion prediction involved careful selection and training of models that could handle the complexities of traffic data, such as non-linear relationships and temporal dependencies.

3.4.1 Model Selection

The selection of models was guided by the specific characteristics of the traffic data. The following models were considered:

- **Decision Tree Regressor**: Chosen for its ability to model non-linear relationships and provide interpretable decision-making processes. Decision Trees are particularly effective in scenarios where the relationship between features and the target variable is complex and non-linear.

- **Extra Trees Regressor**: This ensemble method was selected to improve robustness and accuracy by combining the predictions of multiple decision trees. Extra Trees help reduce the variance of the model, making it more stable and reliable.

- **Long Short-Term Memory (LSTM) Networks**: Included due to its strength in handling time-series data, which is critical for predicting future traffic conditions based on past observations. LSTMs are particularly adept at capturing temporal dependencies in data, making them ideal for forecasting tasks.
- **Random Forest Regressor**: Similar to Extra Trees, Random Forest uses an ensemble of decision trees with the added benefit of bagging, which further reduces overfitting. This model was chosen for its balance between accuracy and robustness.

3.4.2 Data Splitting

To train and evaluate the models effectively, the dataset was split into two subsets:

- **Training Set**: 80% of the data was used to train the models. This subset was crucial for fitting the models and learning the underlying patterns in the traffic data.
- **Test Set**: The remaining 20% of the data was set aside as the test set. This set was not used during training and was reserved solely for evaluating the final model's performance. The use of a single test set ensured that the model's Mean Squared Error (MSE) was assessed on unseen data, providing a realistic measure of its predictive accuracy.

3.4.3 Training and Hyperparameter Tuning

Each model was trained on the training set, with hyperparameters optimized to enhance performance. The optimization was done using techniques such as Grid Search and Random Search. The goal was to minimize the Mean Squared Error (MSE) on the test set, as MSE was the primary metric for evaluating model performance. For example:

- **Decision Tree Regressor**:

  Hyperparameters such as `max_depth` and `min_samples_split` were fine-tuned to control the complexity of the tree and prevent overfitting.
- **LSTM Network**:

  The number of LSTM units, learning rate, and batch size were optimized to ensure the network effectively captured the temporal dynamics of the traffic data.

3.4.4 Final Model Evaluation

The final model was selected based on its performance on the test set, with the MSE serving as the key metric. The model that achieved the lowest MSE was deemed the most effective at predicting traffic congestion, and was chosen for deployment. This approach ensured that the selected model was both accurate and reliable in real-world scenarios.

3.4.5 Ganache Implementation and Configuration

Description: Ganache is a personal blockchain for Ethereum development that allows developers to deploy contracts, develop applications, and run tests on a local blockchain environment. It simulates the features of the Ethereum mainnet, providing full control over network parameters, which is essential for testing and development purposes.

Implementation: For the development and testing of the smart contracts within our IoV system, Ganache was used to simulate a local Ethereum blockchain. The following steps outline the implementation and configuration process:

1. **Installation:**
    - Ganache was installed on the local development environment. Both the command-line interface (CLI) and graphical user interface (GUI) versions are available. For this project, the GUI version was selected to easily visualize blockchain operations and account balances.
2. **Network Configuration:**
    - The local blockchain was configured with the following parameters:
        - **Port**: The default port (8545) was used to run the Ganache instance.
        - **Network ID**: A custom network ID (5777) was chosen to avoid conflicts with other networks.
        - **Accounts**: Ganache was configured to generate 20 accounts, each pre-funded with 100 ETH to simulate transactions and smart contract deployments without financial limitations.
        - **Block Time**: The block time was set to 0 seconds for immediate mining, ensuring that transactions are processed without delay during testing.
3. **Smart Contract Deployment:**
    - The smart contracts were deployed to the Ganache blockchain using a development framework such as Truffle. Deployment scripts were configured to connect to the local blockchain instance running on Ganache.
    - After deployment, the contract addresses and transaction hashes were logged, and the contract's state was inspected using the Ganache GUI to verify successful deployment.
4. **Testing and Simulation:**
    - Various test scenarios were executed on the Ganache blockchain, including registering models, updating metadata, and distributing rewards. These tests were performed to validate the smart contract logic under simulated conditions.
    - Ganache's built-in block explorer and transaction history features were used to monitor and verify each transaction, ensuring that the contracts behaved as expected.
5. **Interacting with the Blockchain:**

- Interactions with the deployed smart contracts were carried out using the appropriate tools and libraries configured to communicate with the Ganache instance.
- The development environment was configured to connect to the Ganache blockchain, facilitating seamless interactions for testing and development.

3.5 Smart Contracts and IPFS Integration for Federated Learning

3.5.1 Smart Contracts

Smart contracts are integral to the decentralized architecture of our traffic prediction system, ensuring secure, transparent, and automated management of the federated learning process. The following smart contracts have been implemented to manage various aspects of the system, including model registration, metadata handling, and contributor rewards.

1. **Local Model Registry:**

   The LocalModelRegistry smart contract is responsible for registering local machine learning models trained on individual nodes. After training, models are uploaded to the InterPlanetary File System (IPFS), and the resulting IPFS hash (Content Identifier, CID) is registered on the blockchain via this contract. This ensures the integrity and provenance of each model.

   - **Model Registration:** After training, each node calls the 'registerLocalModel' function in the LocalModelRegistry contract. This function stores the model's IPFS hash along with the contributor's address on the blockchain, ensuring traceability and integrity.
   - **Data Integrity Verification:** The contract also allows for verifying the integrity of a registered model by comparing the stored IPFS hash with a newly generated hash from the model data. This ensures the model has not been tampered with post-registration.

2. **Aggregated Model Registry:**

   The AggregatedModelRegistry contract manages the registration of models aggregated from multiple local models. This contract is essential for tracking different versions of aggregated models and ensuring their integrity.

   - **Model Aggregation and Registration:** Once local models are aggregated, the resulting global model is uploaded to IPFS, where a new CID is generated. This CID, along with the version number and reference to the last local model used, is registered on the blockchain via the 'registerAggregatedModel' function.
   - **Version Control:** The contract maintains a history of all aggregated model versions, linking each version to its respective IPFS hash and metadata, thus ensuring a complete and transparent record over time.

3. **File Metadata Registry:**

   The FileMetadataRegistry contract stores and manages metadata related to models and datasets. This contract enhances the searchability and traceability of models by associating additional information with each IPFS hash.

   - **Metadata Addition:** Contributors can add metadata to their files using the ‘addFileMetadata‘ function. This metadata, including file name, description, tags, and the IPFS hash, is stored on the blockchain, ensuring that each file is easily searchable and its context is well-documented.
   - **Search Functionality:** The contract provides a ‘searchFilesByTag‘ function, enabling users to search for files based on specific tags, making it easier to retrieve related models or datasets.

4. **Reputation System:**

   The ReputationSystem smart contract manages the incentive mechanism within the federated learning framework. Contributors are rewarded based on the quality and usefulness of their models.

   - **Contributor Rewards:** Upon registering a local or aggregated model, contributors are rewarded with reputation points. The ‘rewardContributor‘ function assigns points inversely proportional to the Mean Squared Error (MSE) of the model, encouraging the development of high-quality models.
   - **Reputation Tracking:** The contract tracks the total reputation points for each contributor, which can be queried via the ‘getReputation‘ function, promoting transparency and a competitive environment.

3.5.2 Integrated Workflow

The integration of these smart contracts, combined with IPFS, creates a secure and seamless workflow for federated learning and model management in the decentralized traffic prediction system. This workflow ensures that all processes are transparent, secure, and incentivized, fostering a robust and reliable decentralized model training environment.

1. **Model Training and Upload:**

   - Each node independently trains a local machine learning model using its private dataset. The trained model is serialized, compressed, and uploaded to IPFS. IPFS generates a unique CID for the model, ensuring its immutability and facilitating future retrieval and verification.
   - The model's CID is then registered on the blockchain using the **LocalModelRegistry** smart contract, establishing a tamper-proof record of the model's existence and ownership.

2. **Model Aggregation:**

   - Multiple local models are retrieved from IPFS using their CIDs for aggregation. This ensures the exact version of each model is retrieved as originally uploaded.

- The aggregation process involves extracting decision trees from each Random Forest model and combining them to create a global model, representing the collective knowledge of all participating nodes.
- The aggregated global model is uploaded to IPFS, generating a new CID. This CID, along with version information, is registered on the blockchain via the **AggregatedModelRegistry** contract, preserving the version history and integrity of the global model.

3. **Metadata Management:**
    - Metadata for both local and aggregated models is managed through the **FileMetadataRegistry** contract. This metadata includes the model's name, description, relevant tags, and its IPFS hash, enhancing traceability and searchability within the system.
    - The system allows users to search for models by tags or keywords using the 'searchFilesByTag' function, making it easier to locate related models or datasets.

4. **Reward Distribution:**
    - Contributors are incentivized through a reputation-based reward system managed by the **ReputationSystem** contract. Reputation points are awarded based on model performance, with higher-quality models (lower MSE) earning more points.
    - Reputation points are tracked on the blockchain, allowing contributors to see their standing within the network. The 'getReputation' function provides transparency and encourages continuous improvement.

This integrated workflow, leveraging IPFS for decentralized storage and blockchain for secure management, ensures that the federated learning system operates efficiently and transparently. The combination of these technologies not only protects data privacy but also maintains the integrity and traceability of the models, providing a solid foundation for decentralized and collaborative model training.

3.6 Limitations

While the use of Ganache and IPFS provided valuable insights during development and testing, several limitations should be noted:

1. **Ganache Blockchain Simulation:** Ganache offers a controlled environment for testing, but it doesn't fully replicate the complexities of a live blockchain network. Real-world conditions such as network latency, variable transaction fees, and potential congestion are not present in Ganache, which may result in an overestimation of the system's performance under actual conditions.

2. **IPFS Integration:** Testing IPFS locally ensured smooth data storage and retrieval processes. However, in a global, distributed IPFS network, issues like data availability and retrieval latency could arise. These potential challenges were not fully captured in the local environment, leading to potentially optimistic performance expectations.

3. **Scalability Testing:** The scalability of the system was evaluated under simulated conditions, but the absence of real-world network variability and user load means that these results may not fully reflect the system's behavior in a live deployment. Further testing on a more realistic network setup, such as the Ethereum testnet, is recommended to better understand the system's performance in real-world scenarios.

While the local testing environment has been essential for initial validation, the results should be interpreted with caution. Additional testing in more realistic environments is needed to accurately assess the system's performance and scalability under real-world conditions.

3.7 Summary

In this chapter, we detailed the design and implementation of a decentralized, secure, and efficient Internet of Vehicles (IoV) system. The system architecture was divided into multiple layers—Data Collection, Data Processing, Blockchain, Decentralized Storage, and Application—each contributing to the overall functionality of the system. The architecture ensures that data is collected, processed, and managed securely across a distributed network.

We utilized a subset of the Large-Scale Traffic and Weather Events (LSTW) dataset focused on Texas to provide real-world traffic data for our system. The data processing workflow involved extensive cleaning, feature engineering, and visualization to prepare the data for AI model development. The AI models, particularly using Federated Learning, were developed to predict traffic congestion while maintaining data privacy across distributed nodes.

The Blockchain Layer was implemented using Ethereum-based smart contracts to manage various processes within the system, including model registration, metadata management, and contributor rewards. The integration of the InterPlanetary File System (IPFS) was critical for decentralized storage, ensuring data integrity and secure retrieval.

We also discussed the use of Ganache as a local blockchain simulator to test the smart contracts and the system's performance. However, the limitations of this local testing environment were noted, including the absence of real-world conditions like network latency and variable transaction costs. Future work will involve testing the system on more realistic networks to better understand its performance in a live deployment.

Overall, this chapter established the foundation for a robust and scalable IoV system, demonstrating the integration of advanced technologies to achieve secure and efficient traffic management.

Chapter 4

Performance Evaluation

The implementation of this project was carried out using a well-defined computational environment, ensuring smooth execution of all processes involved in data handling, model training, and system integration. Detailed descriptions of the methodologies used can be found in the Methodology section. Below is a summary of the key components of the environment setup:

The project was implemented on a local machine with the following configuration:

- **Operating System:** Windows 11
- **Programming Language:** Python 3.11.4
- **Machine Learning Libraries:**
  - TensorFlow 2.16.1
  - Scikit-learn 1.3.0
- **Blockchain Simulation:** Ganache v7.9.1 for Ethereum
- **Smart Contracts Development:** Truffle v5.11.5
- **Decentralized Storage:** Pinata (IPFS) API
- **Smart Contracts:** Written in Solidity v0.8
- **Load Testing:** Locust 2.31.1 for simulating user interactions and assessing system scalability

4.1 AI Model Evaluation and Selection

To determine the most suitable model for predicting traffic congestion, several machine learning models were evaluated. The models considered include Gradient Boosting Regressor, Random Forest Regressor, K-Nearest Neighbors Regressor, LSTM, Linear Regression, XGBoost Regressor, Decision Tree Regressor, and Extra Trees Regressor. The evaluation metrics used were Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), as these metrics are critical for assessing the predictive accuracy of regression models.

4.1.1 Methodology Overview

Each model was trained and evaluated using a consistent dataset, with the same training/test split and hyperparameter tuning strategy. This consistency ensures that the performance comparison is fair and reflects the true capabilities of each model.

4.1.2 Model Performance

The performance of each model was evaluated, and their MSE values were compared. The results are summarized in Table 4.1.

Table 4.1: Performance Metrics of Evaluated Models

Model	**MSE**	**RMSE**
Gradient Boosting Regressor	0.56	0.7483
Random Forest Regressor	0.34	0.5831
K-Nearest Neighbors Regressor	0.79	0.8889
LSTM	0.58	0.7616
Linear Regression	0.72	0.8485
XGBoost Regressor	0.43	0.6557
Decision Tree Regressor	0.66	0.8124
Extra Trees Regressor	0.38	0.6164

Gradient Boosting Regressor:

- **Performance:** The Gradient Boosting Regressor achieved an MSE of 0.56 and an RMSE of 0.7483. This model is known for its ability to handle complex data patterns by sequentially building weak learners.
- **Resource Utilization:** The Gradient Boosting model exhibited moderate CPU usage, with frequent fluctuations reflecting its iterative training process. The memory usage started at a baseline and gradually increased over time, indicating that the model progressively utilized more memory as the number of trees in the ensemble grew.

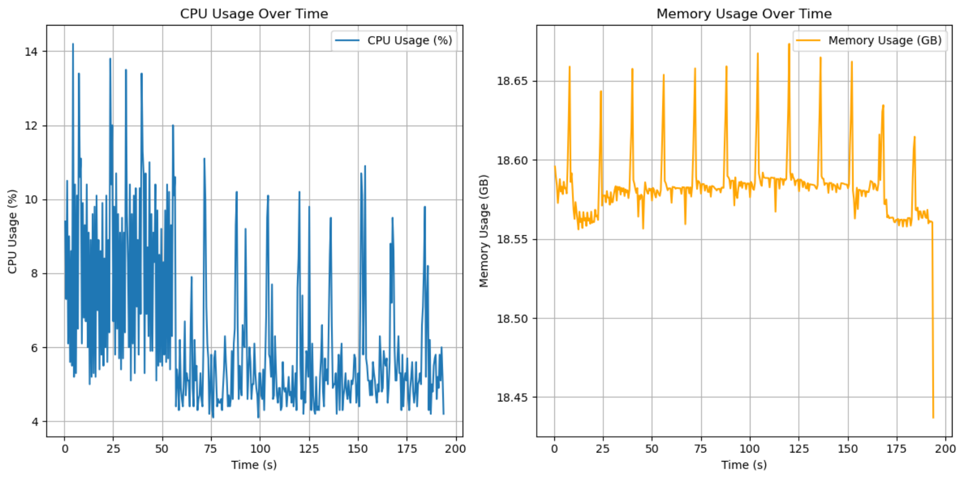


Figure 4.1: Gradient Boosting Regressor CPU & Memory Utilization

Random Forest Regressor:

- **Performance:** The Random Forest model provided the best performance with an MSE of 0.34 and an RMSE of 0.5831. Its ensemble approach, combining multiple decision trees, helps to reduce overfitting and improve predictive accuracy.

- **Resource Utilization:** Random Forest showed relatively stable CPU usage, with moderate spikes corresponding to the processing of individual trees. The memory usage started at a baseline and consistently increased, demonstrating that the model's complexity and data handling requirements grew with the number of trees.

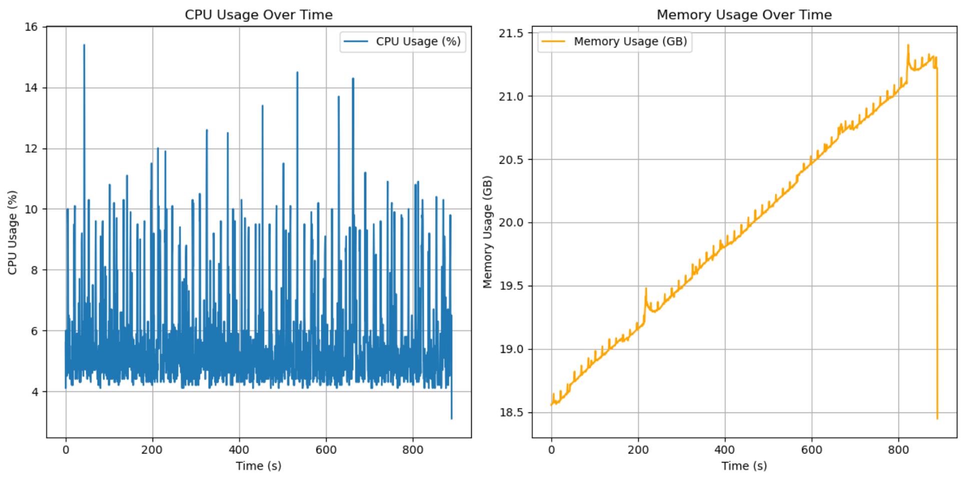


Figure 4.2: Random Forest Regressor CPU & Memory Utilization

K-Nearest Neighbors Regressor:

- **Performance:** The KNN model performed the worst among the evaluated models, with an MSE of 0.79 and an RMSE of 0.8889. It struggled to capture the underlying patterns in the traffic data, resulting in lower accuracy.

- **Resource Utilization:** KNN exhibited high CPU utilization at the beginning of the training process, with memory usage increasing moderately. The resource demand was particularly high during the distance calculations for neighbors, contributing to its inefficiency in handling large datasets.

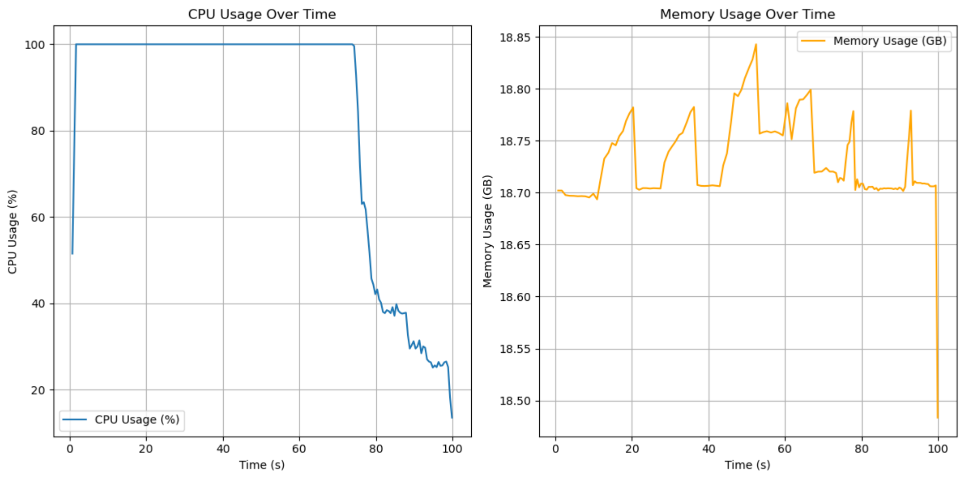


Figure 4.3: K-Nearest Neighbours Regressor CPU & Memory Utilization

LSTM:

- **Performance:** The LSTM model achieved an MSE of 0.58 and an RMSE of 0.7616. While designed to handle temporal dependencies, its performance was moderate, due to the complex and non-linear nature of traffic data.

- **Resource Utilization:** LSTM showed significant CPU usage during training, with memory usage increasing slightly from its starting point. The nature of LSTM's architecture, which processes data in sequences, likely contributed to the continuous CPU usage and the moderate rise in memory consumption.

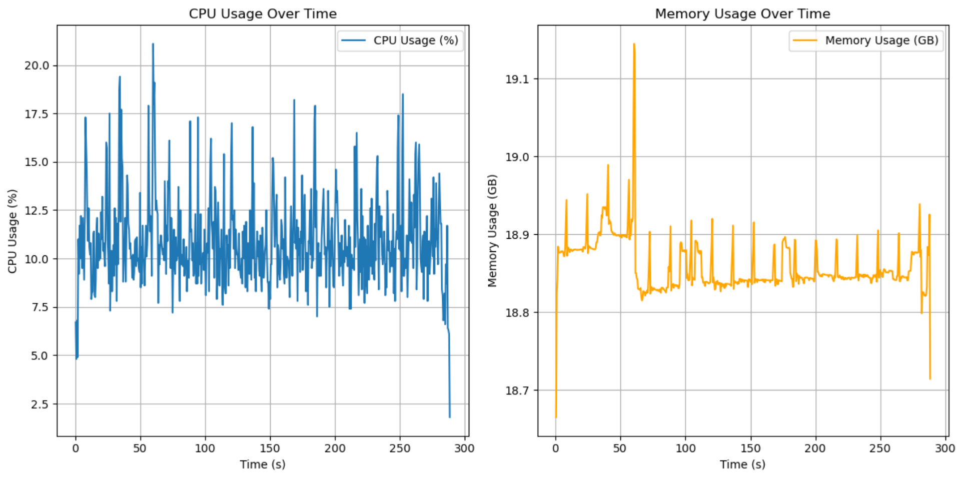


Figure 4.4: LSTM CPU & Memory Utilization

Linear Regression:

- **Performance:** The Linear Regression model produced an MSE of 0.72 and an RMSE of 0.8485, reflecting its limitations in capturing non-linear relationships in the data.

- **Resource Utilization:** Linear Regression demonstrated minimal CPU usage with almost negligible memory increase. As a simple model, it requires few computational resources, which aligns with its straightforward calculation of a linear relationship.

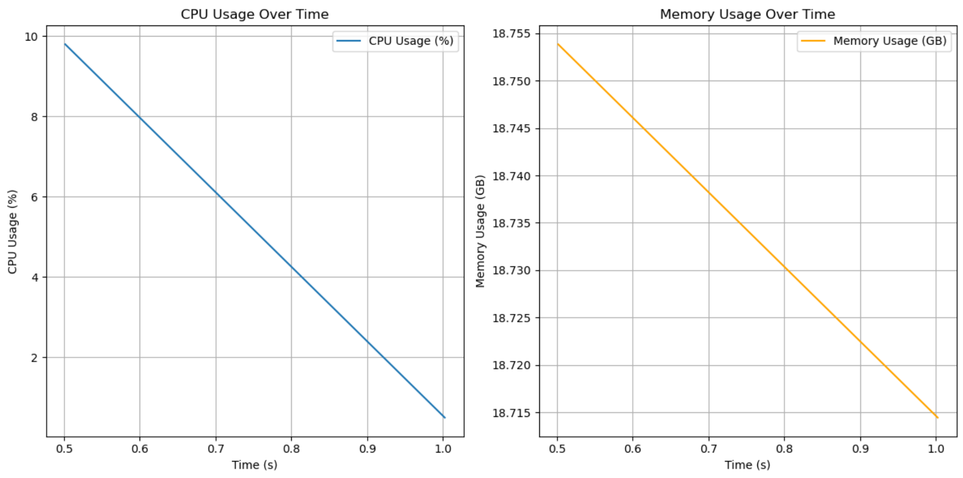


Figure 4.5: Linear Regression CPU & Memory Utilization

XGBoost Regressor:

- **Performance:** XGBoost demonstrated strong performance with an MSE of 0.43 and an RMSE of 0.6557. The model's boosting mechanism, which iteratively improves predictions, contributed to its high accuracy.

- **Resource Utilization:** XGBoost showed significant CPU usage, peaking during training iterations, with memory usage showing a slight increase from the baseline. The high resource demand is due to the model's complexity and the iterative nature of boosting, which requires substantial processing power.

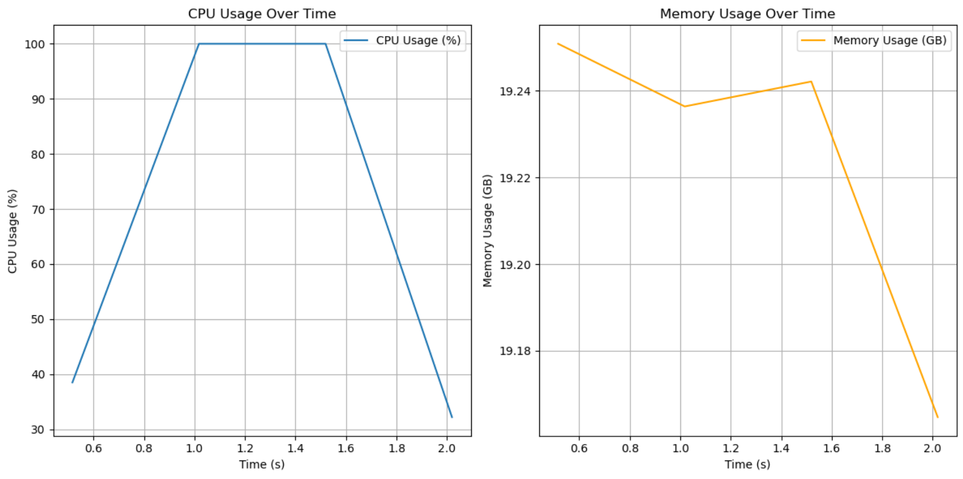


Figure 4.6: XGBoost Regressor CPU & Memory Utilization

Decision Tree Regressor:

- **Performance:** The Decision Tree model yielded an MSE of 0.66 and an RMSE of 0.8124, indicating potential overfitting, a common issue with decision trees when not pruned adequately.
- **Resource Utilization:** The Decision Tree model showed quick spikes in CPU usage, corresponding to the growth of the tree, with memory usage showing minimal increase from the baseline. This pattern reflects the model's efficiency in handling data but also its tendency to overfit if not managed properly.

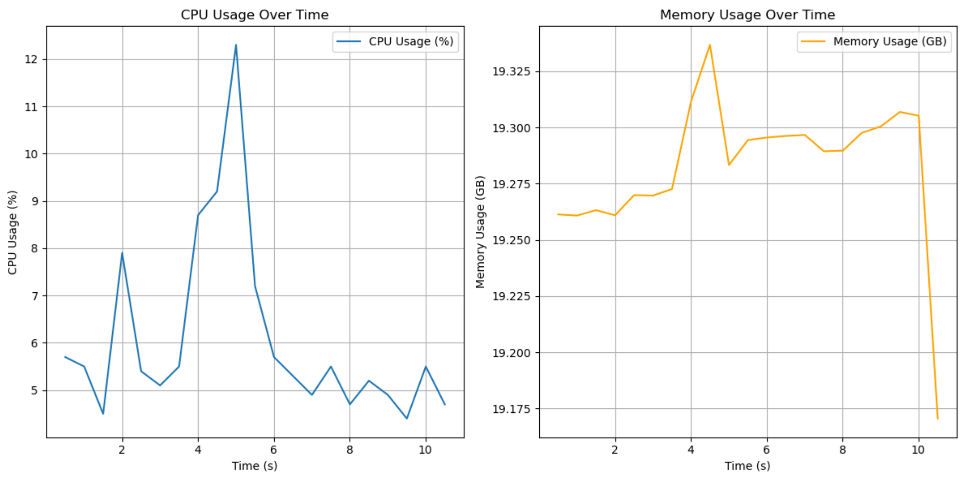


Figure 4.7: Decision Tree Regressor CPU & Memory Utilization

Extra Trees Regressor:

- **Performance:** Extra Trees Regressor performed well, with an MSE of 0.38 and an RMSE of 0.6164. It's similar to Random Forest but introduces additional randomness, leading to robust predictions.
- **Resource Utilization:** Extra Trees exhibited CPU usage similar to Random Forest, with steady memory growth. The model's randomization process adds to its robustness, but it also increases memory usage gradually over time as more trees are added.

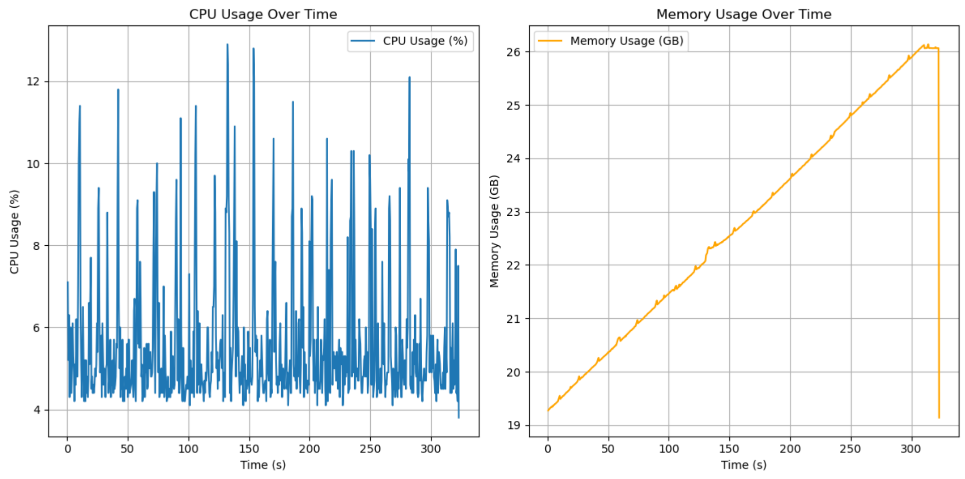


Figure 4.8: Extra Tree Regressor CPU & Memory Utilization

4.1.3 Discussion

The Random Forest Regressor emerged as the best-performing model, offering the lowest MSE and RMSE among all models evaluated, as shown in Figure 4.9. Its ability to handle complex, non-linear relationships within the traffic data, combined with stable and moderate CPU and memory utilization, makes it the ideal choice for real-time traffic management in the Internet of Vehicles (IoV) framework.

While other models like Extra Trees Regressor and XGBoost also performed well, their computational demands may limit their applicability in environments with constrained resources. In contrast, Random Forest's balance between accuracy and resource efficiency ensures its suitability for deployment in real-time systems where both predictive performance and resource utilization are critical considerations.

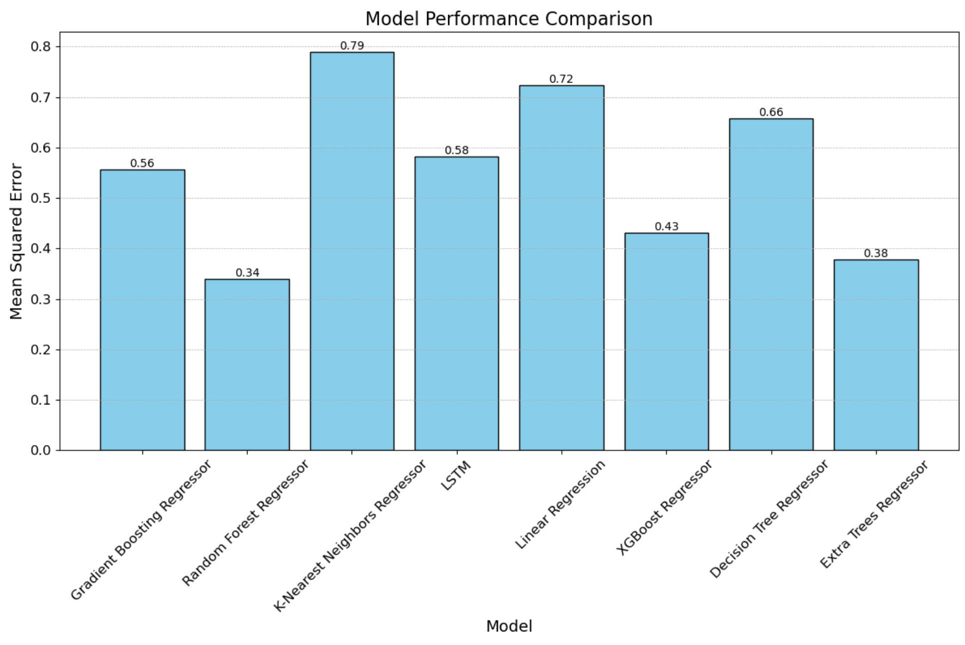


Figure 4.9: MSE Comparison of AI Models

4.2 Storage Service Performance Comparison

In addition to the system performance, a comparison between Google Drive and Pinata IPFS was conducted to assess their suitability for managing model files. The results, summarized in Table 4.2, highlight the differences in upload and download times, file sizes, and network speeds.

Table 4.2: Performance Comparison of Google Drive and Pinata IPFS

Metric	**Google Drive**	**Pinata IPFS**
File Size (MB)	255.59	255.59
Upload Time (s)	8.94	8.86
Download Time (s)	13.71	3.33
Downloaded Size (MB)	255.59	
System's Download Speed (Mbps)	724.12	
System's Upload Speed (Mbps)	229.94	

Discussion:

The comparison reveals that while both services have similar upload times, Pinata IPFS significantly outperforms Google Drive in download times. This makes decentralized storage solutions like Pinata IPFS highly effective for applications requiring rapid data access.

The results demonstrate that decentralized storage solutions can offer competitive, and sometimes superior, performance compared to traditional cloud storage services, particularly in terms of download efficiency.

4.3 System Performance and Evaluation

This section presents a detailed analysis of the blockchain-based federated learning system, simulated using Ganache. The performance metrics include both the computational resource utilization during model training and the blockchain-specific operations such as model registration, contributor reward distribution, and data integrity checking. Additionally, the scalability and performance under simulated user loads were evaluated using Locust.

4.3.1 Computational Performance

The resource utilization during the training of local models was monitored for four distinct nodes within the federated learning network, each referred to as a "contributor." These contributors, which represent individual nodes or participants in the decentralized system, are each responsible for training a local model using their respective datasets. Additionally, each contributor operates with its own blockchain address, allowing for individual tracking and reward distribution.

During the training process, the CPU and memory usage were recorded for each contributor. The following observations were made:

- **CPU Usage:** Across all contributors, the CPU usage exhibited varying patterns during the training process. Contributors 1 and 4 showed significant fluctuations with peaks reaching up to 12%, indicating periods of high computational demand. Contributors 2 and 3, however, demonstrated more stable CPU usage with moderate variations, suggesting consistent processing loads.

- **Memory Usage:** The memory usage across contributors started between 5 GB and 6 GB. Contributor 1 experienced the most substantial increase, reaching up to 8 GB as the complexity of the model grew during training. Contributors 2 and 4 saw a slight increase, stabilizing around 7 GB, while Contributor 3's memory usage remained minimal, indicating efficient memory utilization.

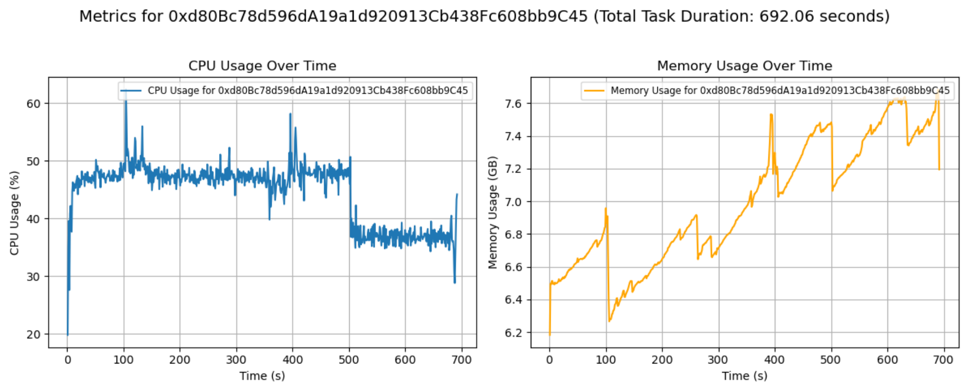


Figure 4.10: Local Model Training for Contributor 1

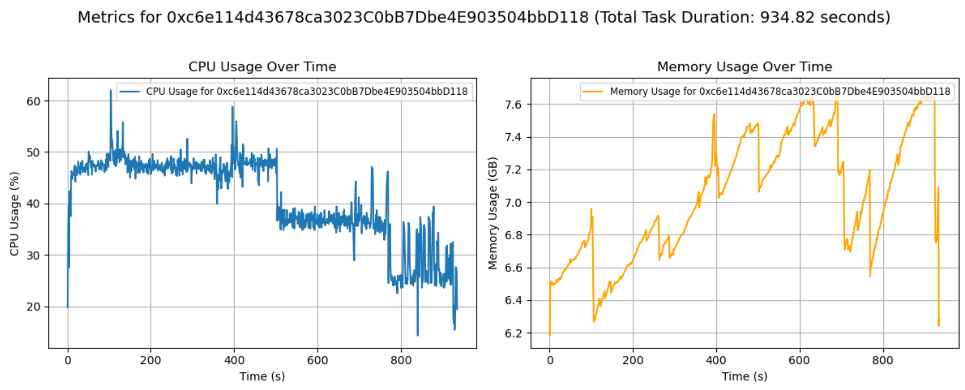


Figure 4.11: Local Model Training for Contributor 2

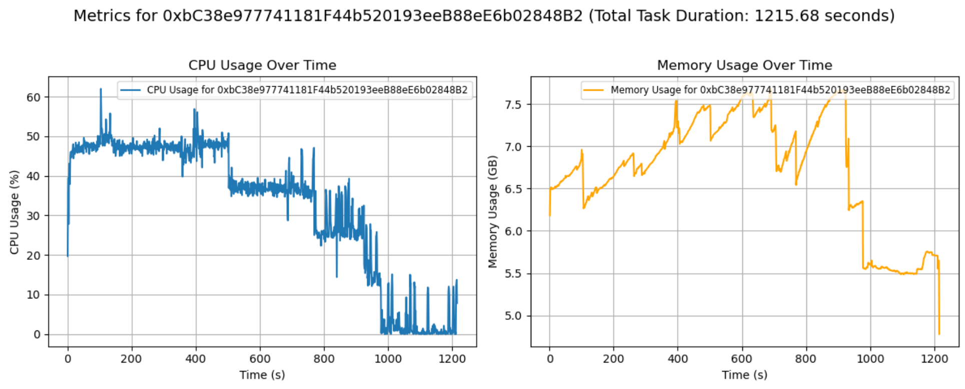


Figure 4.12: Local Model Training for Contributor 3

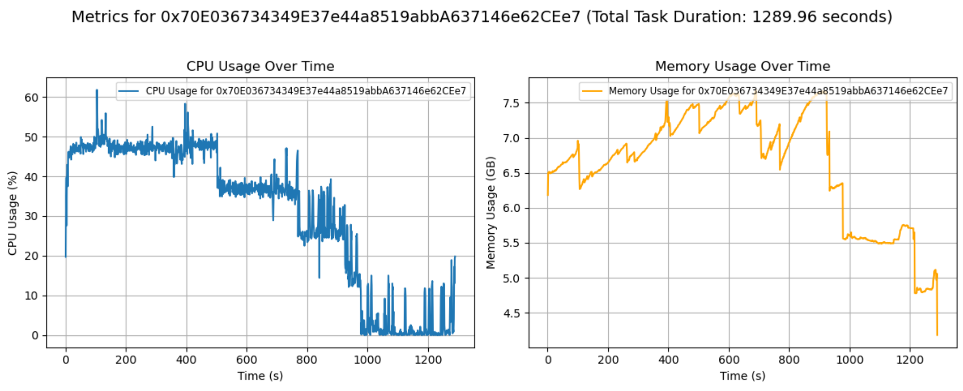


Figure 4.13: Local Model Training for Contributor 4

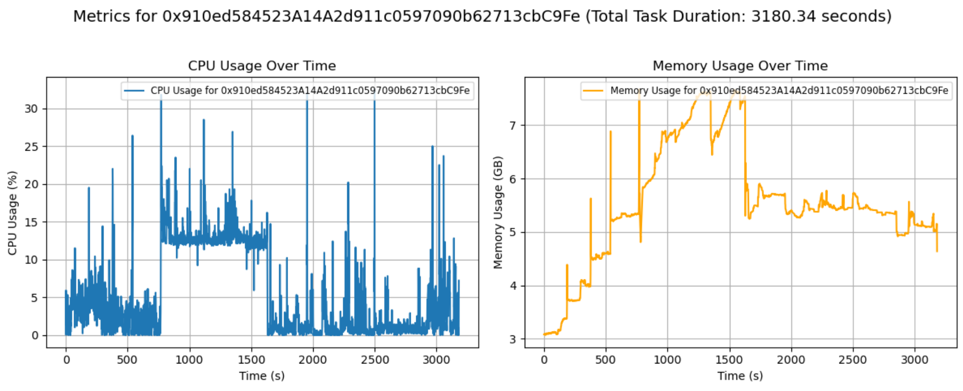


Figure 4.14: Aggregated Training

4.3.2 Blockchain Operations and Efficiency

The blockchain operations involved registering the trained local models on the blockchain, rewarding contributors based on model performance, verifying data integrity, and aggregating models to create a global model. The efficiency of these operations is evaluated based on the time taken for each task, the integrity of the registered models, and the resulting model performance.

Local Model Registration

Each contributor's local model was trained and registered on the blockchain. The registration process involved uploading the model to the IPFS and recording the IPFS hash on the blockchain. The time taken for each model registration varied slightly among contributors, reflecting differences in model complexity and computational resources.

Contributor Reward Distribution

Contributors were rewarded with reputation points based on the performance of their local models. The rewards were inversely proportional to the Mean Squared Error (MSE) of the models, where the MSE measures the error in predicting the traffic severity, a target variable valued from 0 to 4. The reward distribution is summarized in Table 4.3.

Table 4.3: Contributor Reward Distribution

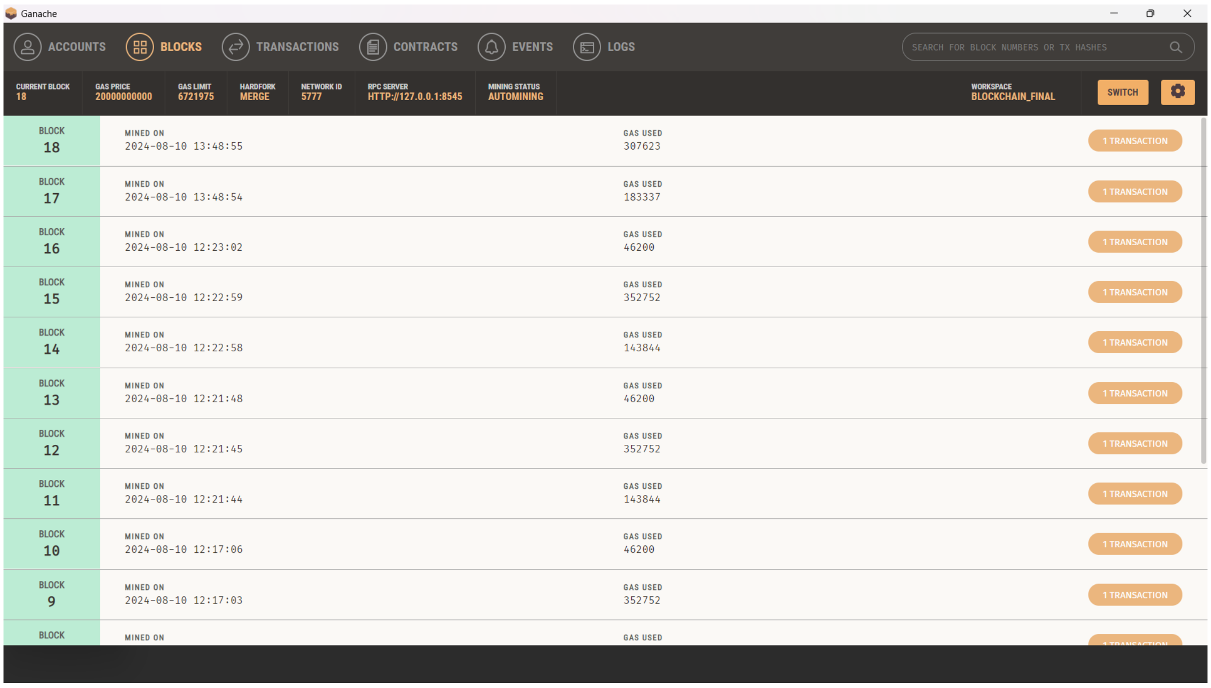
Contributor Address	MSE	Reward Points
0xd80Bc78d596dA19a1d920913Cb438Fc608bb9C45	0.330	302
0xc6e114d43678ca3023C0bB7Dbe4E903504bbD118	0.333	300
0xbC38e977741181F44b520193eeB88eE6b02848B2	0.307	325
0x70E036734349E37e44a8519abbA637146e62CEe7	0.344	290

Data Integrity Checking

An essential aspect of the blockchain-based federated learning system is ensuring the integrity of the models registered by contributors. After the models were trained and registered, the system performed integrity checks by comparing the provided IPFS hashes with the latest registered model hashes for each contributor. The results confirmed that all models matched their respective hashes, ensuring that the data integrity was maintained throughout the process.

Blockchain Transactions Overview

The Ganache interface provides detailed information on the blockchain's operations, including block creation, transaction records, and the gas used during transactions. Below are specific details related to the figures provided:

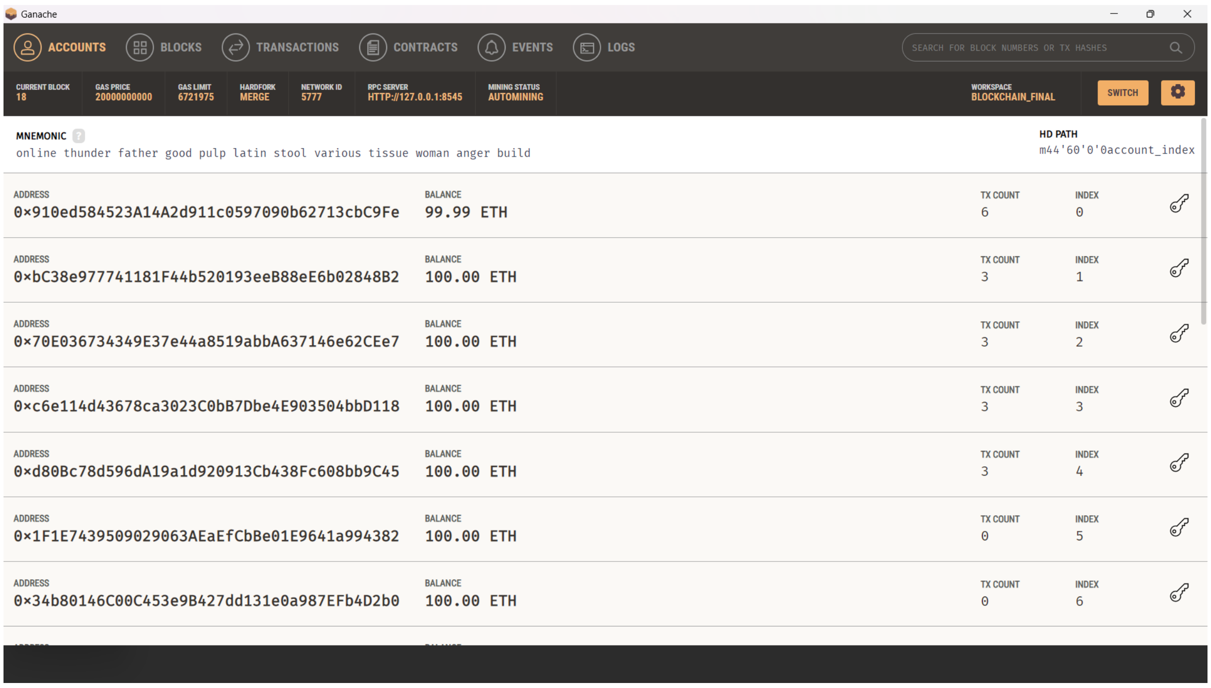


CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
18	20000000000	6721975	MERGE	5777	HTTP://127.0.0.1:8545	AUTOMINING	BLOCKCHAIN_FINAL

BLOCK	MINED ON	GAS USED	
18	2024-08-10 13:48:55	307623	1 TRANSACTION
17	2024-08-10 13:48:54	183337	1 TRANSACTION
16	2024-08-10 12:23:02	46200	1 TRANSACTION
15	2024-08-10 12:22:59	352752	1 TRANSACTION
14	2024-08-10 12:22:58	143844	1 TRANSACTION
13	2024-08-10 12:21:48	46200	1 TRANSACTION
12	2024-08-10 12:21:45	352752	1 TRANSACTION
11	2024-08-10 12:21:44	143844	1 TRANSACTION
10	2024-08-10 12:17:06	46200	1 TRANSACTION
9	2024-08-10 12:17:03	352752	1 TRANSACTION

Figure 4.15: Ganache Interface Displaying Mined Blocks and Transactions

Figure 4.15 shows the Ganache interface listing the blocks that were mined during the simulation. Each block corresponds to specific transactions that include model registration and reward distribution. The block numbers, mining timestamps, and gas usage are displayed, providing insight into the efficiency and cost of each operation.



CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
18	20000000000	6721975	MERGE	5777	HTTP://127.0.0.1:8545	AUTOMINING	BLOCKCHAIN_FINAL

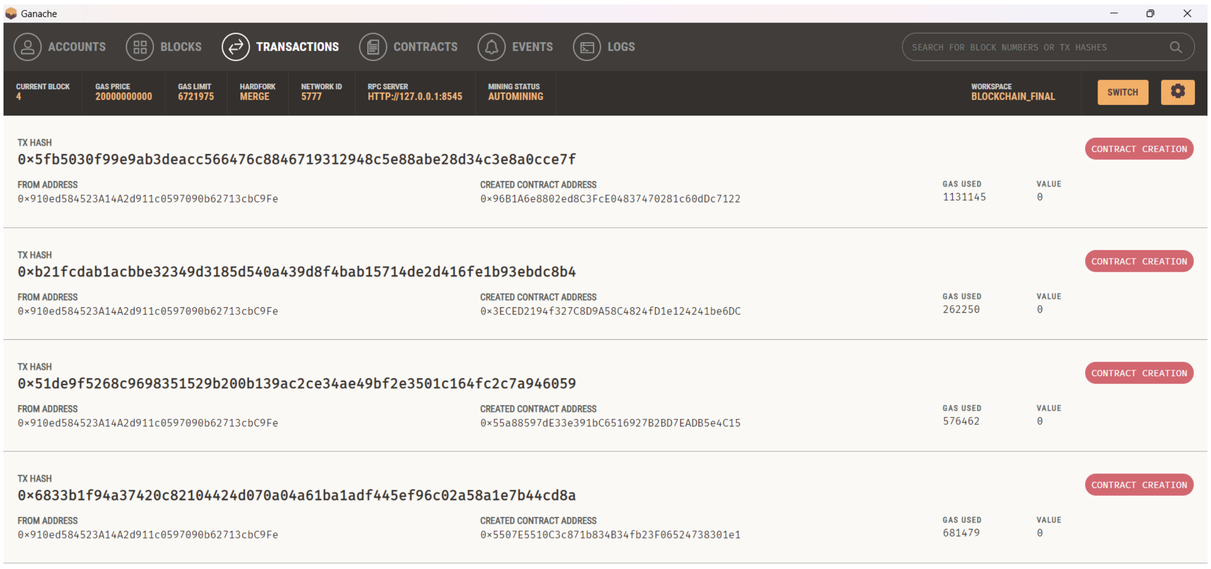
MNEMONIC
online thunder father good pulp latin stool various tissue woman anger build

HD PATH
m44'60'0'0account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0×910ed584523A14A2d911c0597090b62713cbC9Fe	99.99 ETH	6	0
0×bC38e977741181F44b520193eeB88eE6b02848B2	100.00 ETH	3	1
0×70E036734349E37e44a8519abbA637146e62CEe7	100.00 ETH	3	2
0×c6e114d43678ca3023C0bB7Dbe4E903504bbD118	100.00 ETH	3	3
0×d80Bc78d596dA19a1d920913Cb438Fc608bb9C45	100.00 ETH	3	4
0×1F1E7439509029063AEaEfCbBe01E9641a994382	100.00 ETH	0	5
0×34b80146C00C453e9B427dd131e0a987EFb4D2b0	100.00 ETH	0	6

Figure 4.16: Ganache Interface Showing Account Balances and Transaction Counts

Figure 4.16 displays the account balances and transaction counts for each contributor involved in the simulation. This figure highlights the distribution of ETH among the contributors and the number of transactions each account has performed. The consistent balance among accounts reflects the uniform reward distribution mechanism in place.



CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
4	20000000000	6721975	MERGE	5777	HTTP://127.0.0.1:8545	AUTOMINING	BLOCKCHAIN_FINAL

TX HASH	FROM ADDRESS	CREATED CONTRACT ADDRESS	GAS USED	VALUE	
0×5fb5030f99e9ab3deacc566476c8846719312948c5e88abe28d34c3e8a0cce7f	0×910ed584523A14A2d911c0597090b62713cbC9Fe	0×96B1A6e8802ed8C3FcE04837470281c60dDc7122	1131145	0	CONTRACT CREATION
0×b21fcdab1acbbe32349d3185d540a439d8f4bab15714de2d416fe1b93ebdc8b4	0×910ed584523A14A2d911c0597090b62713cbC9Fe	0×3ECED2194f327C8D9A58C4824fD1e124241be6DC	262250	0	CONTRACT CREATION
0×51de9f5268c9698351529b200b139ac2ce34ae49bf2e3501c164fc2c7a946059	0×910ed584523A14A2d911c0597090b62713cbC9Fe	0×55a88597dE33e391bC6516927B2BD7EADB5e4C15	576462	0	CONTRACT CREATION
0×6833b1f94a37420c82104424d070a04a61ba1adf445ef96c02a58a1e7b44cd8a	0×910ed584523A14A2d911c0597090b62713cbC9Fe	0×5507E5510C3c871b834B34fb23F06524738301e1	681479	0	CONTRACT CREATION

Figure 4.17: Ganache Interface Showing Transaction Details and Gas Used

Figure 4.17 shows detailed transaction records from the Ganache interface, including the transaction hashes, sender addresses, and gas used for contract creation. This information is critical for analyzing the efficiency of smart contract executions and their associated costs in terms of gas consumption.

Aggregated Model Performance

The final step involved aggregating the models from all contributors to create a global model. This aggregated model was evaluated on a test dataset to determine its overall performance. The Mean Squared Error (MSE) of the aggregated model was 0.416, indicating a reasonably accurate model that benefits from the contributions of all participants.

4.3.3 Scalability and Load Testing with Locust

The scalability of the blockchain system was evaluated using Locust[28], a load-testing tool, to simulate multiple users interacting with the blockchain and IPFS storage. The goal was to observe how the system performs under varying levels of load, particularly in terms of transaction processing speed on the blockchain and data retrieval times from IPFS.

The results from Locust are summarized as follows:

- **Total Requests per Second:** The system maintained an average of 1-2 requests per second, with occasional spikes reaching up to 2.5 requests per second. This indicates a moderate load-handling capacity.

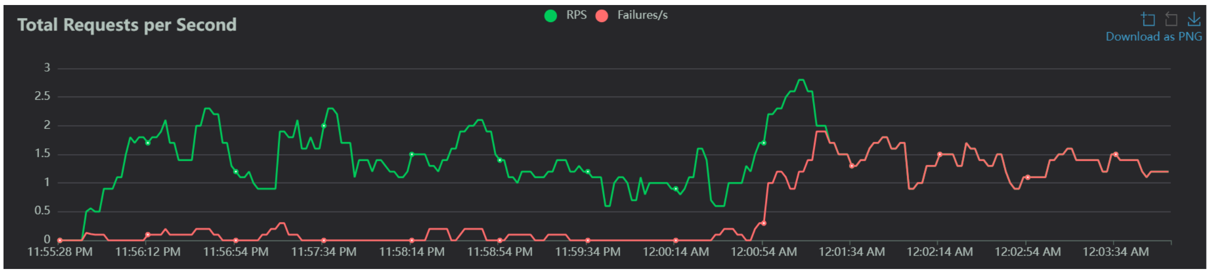


Figure 4.18: Scalability Testing with Locust - Total Requests

- **Response Times (ms):** Response times varied, with most requests being processed within 100 ms. However, under peak load conditions, response times occasionally exceeded 400 ms, indicating potential bottlenecks under high demand.
- **Number of Users:** The number of simulated users gradually increased to nearly 200, with the system managing the growing load effectively until reaching higher thresholds, where performance began to degrade.

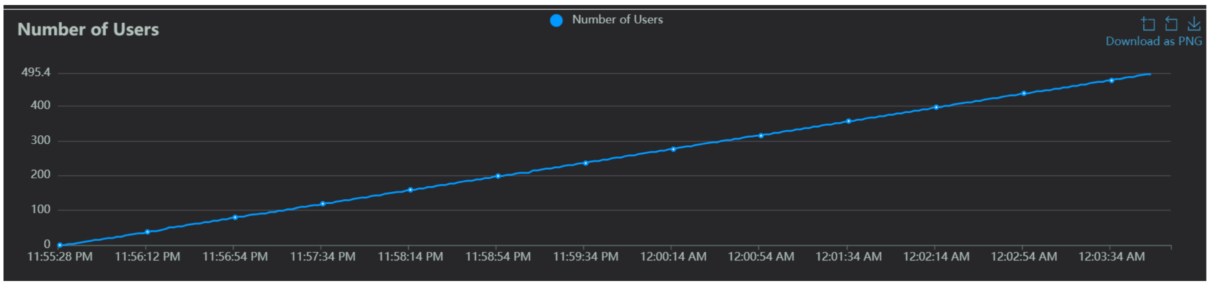


Figure 4.19: Scalability Testing with Locust - Number of Users

4.3.4 Summary of Results

The blockchain-based federated learning system demonstrated effective utilization of computational resources and efficient execution of blockchain operations, as observed in the Ganache simulated environment. The following key results were observed:

- The Random Forest Regressor provided the most accurate local model with the lowest MSE.
- Memory usage was managed efficiently across all contributors, with a gradual increase observed only when necessary.
- The blockchain operations, including model registration, reward distribution, and integrity checks, were executed efficiently, with reasonable timeframes for each task.
- The aggregated model achieved an MSE of 0.416, showcasing the effectiveness of the federated learning approach in improving model accuracy for predicting traffic severity (a target variable valued from 0 to 4).
- The scalability testing with Locust revealed that the system can handle moderate loads effectively, but performance degradation may occur under high load conditions, particularly in response times.

This analysis highlights the potential of integrating blockchain with federated learning to enhance both the accuracy and security of distributed machine learning models, while also noting the limitations observed under simulated load conditions using Ganache.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This research has successfully demonstrated the integration of cutting-edge technologies, including Artificial Intelligence (AI), Blockchain, and the InterPlanetary File System (IPFS), into a comprehensive traffic congestion prediction system within the Internet of Vehicles (IoV) framework. The development of this system addresses several key challenges inherent in modern urban traffic management, such as data privacy, security, scalability, and the need for real-time processing.

The inclusion of Federated Learning (FL) within the system architecture has proven particularly effective in maintaining data privacy while enabling accurate and decentralized traffic predictions. By allowing individual nodes to train models locally and only share model updates rather than raw data, FL minimizes the risk of sensitive information being exposed during transmission. This is crucial in vehicular networks, where data such as vehicle locations and routing information must be protected against potential breaches.

Blockchain technology, integrated as a backbone for security and trust management, provides an immutable and decentralized ledger that ensures data integrity and transparency. The use of smart contracts within the blockchain automates various critical processes, such as reward distribution for data contributors and enforcing access controls. This automation not only reduces the need for manual intervention but also ensures that all transactions are recorded in a secure, tamper-proof manner [44]. Additionally, IPFS has been employed as a decentralized storage solution, which significantly enhances the system's ability to efficiently and securely manage large volumes of data generated by the IoV network [26].

Experimental evaluations of the system demonstrated its effectiveness, particularly in the context of AI model performance. Among the models tested, the Random Forest Regressor achieved the lowest Mean Squared Error (MSE), highlighting its superior capability in predicting traffic congestion accurately. However, scalability tests conducted using the Locust framework revealed potential bottlenecks, especially under conditions of high network load. These bottlenecks were primarily related to transaction throughput and system response times, indicating areas where further optimization is needed.

5.2 Future Works

While the current research has laid a solid foundation, there are several avenues for future work that could further enhance the system's capabilities and address its current limitations:

1. **Real-World Deployment and Testing:** The current system has been validated in a simulated environment, which, while useful, does not fully capture the complexities of real-world deployment. Future work should focus on deploying the system in real-world conditions, such as using the Ethereum testnet for blockchain and a globally distributed IPFS node setup [45]. This deployment would allow for the collection of more accurate performance metrics, including latency, data retrieval times, and transaction costs. Moreover, real-world testing would provide valuable insights into how the system performs under varying environmental conditions and network loads, which are crucial for refining the system before large-scale deployment.

2. **Optimization of Blockchain and Federated Learning Integration:** The interaction between Blockchain and Federated Learning, while effective, introduces computational overhead that can impact the system's overall efficiency. Future research should explore optimizing this integration by investigating alternative consensus mechanisms, such as Proof of Stake (PoS) or Byzantine Fault Tolerance (BFT), which could reduce the computational load [46]. Additionally, exploring more efficient federated learning aggregation techniques could help minimize the communication overhead and improve the speed of model convergence. These optimizations are critical for scaling the system to handle more significant amounts of data and larger numbers of connected vehicles without compromising performance.

3. **Exploration of Advanced AI Models:** The current implementation utilizes several traditional machine learning models for traffic prediction. However, the rapidly evolving field of AI offers new opportunities to enhance predictive accuracy and robustness. Future work should explore integrating more advanced AI techniques, such as reinforcement learning, which can learn optimal traffic management strategies through continuous interaction with the environment [47]. Hybrid models that combine different machine learning approaches, such as combining neural networks with decision trees, could also be investigated to capture complex patterns in traffic data more effectively. These advanced models could potentially offer significant improvements in prediction accuracy and adaptability, particularly in dynamic and complex urban traffic environments.

4. **Scalability and Load Testing in Distributed Environments:** Further research should focus on conducting extensive scalability and load testing in distributed environments that closely mimic real-world conditions. This includes considering factors such as network latency, varying levels of network traffic, and the geographic distribution of nodes across the IoV network. Such testing would help identify and address any scalability issues that may arise as the system is scaled up to manage larger networks of connected vehicles. Additionally, testing in a distributed environment would provide a more accurate assessment of the system's ability to handle real-time traffic data processing and decision-making under different network conditions.

5. **Enhanced Privacy and Security Mechanisms:** Although the current system incorporates several privacy-preserving technologies, such as Federated Learning and Blockchain, there is always room for improvement in this critical area. Future research could focus on integrating advanced cryptographic techniques, such as zero-knowledge proofs, which allow one party to prove to another that they know a value without revealing the actual value itself [46]. Homomorphic encryption, which allows computations to be performed on encrypted data without needing to decrypt it first, could also be explored to further enhance the privacy and security of data within the IoV network [48]. These enhancements would ensure that even as the system scales, it continues to offer robust protection against data breaches and unauthorized access.

6. **Environmental Impact and Sustainability Considerations:** As the deployment of IoV systems increases, so too does the potential environmental impact of the necessary computational infrastructure. Future work could explore ways to reduce the energy consumption of the system, by optimizing the algorithms for energy efficiency or by using more sustainable computing resources. Additionally, the impact of the system on urban mobility patterns and its potential to reduce overall carbon emissions by optimizing traffic flow should be studied to align the development of the IoV with broader sustainability goals.

By pursuing these directions, the system can be refined and expanded to better meet the needs of urban traffic management, offering a more scalable, secure, and efficient solution to the challenges faced in smart city environments. These enhancements would not only improve the system's performance but also ensure that it can be deployed at scale in a manner that is both economically and environmentally sustainable.

Bibliography

[1] S. Moosavi, “Large-scale traffic and weather events dataset (lstw),” https://smoosavi.org/datasets/lstw, 2024, accessed: 2024-06-20.

[2] M. T. Abbas, A. Muhammad, and W.-C. Song, “Road-aware estimation model for path duration in internet of vehicles (iov),” *Wireless Personal Communications*, vol. 109, pp. 715–738, 2019.

[3] F. Bragato, T. Lotta, G. Ventura, M. Drago, F. Mason, M. Giordani, and M. Zorzi, “Towards decentralized predictive quality of service in next-generation vehicular networks,” *arXiv preprint arXiv:2302.11268*, 2023.

[4] L. Silva, N. Magaia, B. Sousa, A. Kobusińska, A. Casimiro, C. X. Mavromoustakis, G. Mastorakis, and V. H. C. De Albuquerque, “Computing paradigms in emerging vehicular environments: A review,” *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 3, pp. 491–511, 2021.

[5] Q.-u.-A. Arshad, W. Z. Khan, F. Azam, M. K. Khan, H. Yu, and Y. B. Zikria, “Blockchain-based decentralized trust management in iot: systems, requirements and challenges,” *Complex & Intelligent Systems*, vol. 9, no. 6, pp. 6155–6176, 2023.

[6] H. Alqahtani and G. Kumar, “Machine learning for enhancing transportation security: A comprehensive analysis of electric and flying vehicle systems,” *Engineering Applications of Artificial Intelligence*, vol. 129, p. 107667, 2024.

[7] H. Kousar, D. D. S. H. Sundara Rajulu Navaneethakrishnan, and D. D. A. M. Tejaswi Vuyyuru, “Integrating artificial intelligence and blockchain technology for secure and efficient data sharing,” *Journal of Survey in Fisheries Sciences*, vol. 10, no. 1S, pp. 5900–5911, 2023.

[8] M. G. Badhe and M. Arjunwadkar, “Decentralised storage systems for blockchain powered applications.”

[9] P. Alvares, L. Silva, and N. Magaia, “Blockchain-based solutions for uav-assisted connected vehicle networks in smart cities: a review, open issues, and future perspectives,” in *Telecom*, vol. 2, no. 1. MDPI, 2021, pp. 108–140.

[10] A.-S. Mihaita, Z. Li, H. Singh, N. Sharma, M. Tuo, and Y. Ou, “Using machine learning and deep learning for traffic congestion prediction: a review,” *Handbook on Artificial Intelligence and Transport*, pp. 124–153, 2023.

[11] P. Lasley, T. Lomax, P. Jha, D. Schrank, B. Eisele, and C. Venditti, “Support for urban mobility analyses (suma) fhwa pooled fund study,” 2019.

[12] A. Uddin, "Traffic congestion in indian cities: Challenges of a rising power," *Kyoto of the cities, Naples*, 2009.

[13] M. I. Khalid, I. Ehsan, A. K. Al-Ani, J. Iqbal, S. Hussain, S. S. Ullah *et al.*, "A comprehensive survey on blockchain-based decentralized storage networks," *IEEE Access*, vol. 11, pp. 10 995–11 015, 2023.

[14] D. Commey, S. Hounsinou, and G. V. Crosby, "Securing health data on the blockchain: A differential privacy and federated learning framework," *arXiv preprint arXiv:2405.11580*, 2024.

[15] W. Zhang, X. Huo, and Z. Bao, "An alliance chain-based incentive mechanism for psg data sharing," *Peer-to-Peer Networking and Applications*, vol. 17, no. 1, pp. 48–67, 2024.

[16] M. Waqas, S. Abbas, U. Farooq, M. A. Khan, M. Alharbi, and M. Asiff, "Reinforcement learning approach for traffic congestion prediction empowered with explainable artificial intelligence (xai)," 2024.

[17] E. Karimi, Y. Chen, and B. Akbari, "Intelligent and decentralized resource allocation in vehicular edge computing networks," *IEEE Internet of Things Magazine*, vol. 6, no. 4, pp. 112–117, 2023.

[18] M. Fardad, G.-M. Muntean, and I. Tal, "Decentralized vehicular edge computing framework for energy-efficient task coordination," in *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)*, 2024, pp. 1–7.

[19] K. Ramana, G. Srivastava, M. R. Kumar, T. R. Gadekallu, J. C.-W. Lin, M. Alazab, and C. Iwendi, "A vision transformer approach for traffic congestion prediction in urban areas," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 3922–3934, 2023.

[20] S. A. Sayed, Y. Abdel-Hamid, and H. A. Hefny, "Artificial intelligence-based traffic flow prediction: a comprehensive review," *Journal of Electrical Systems and Information Technology*, vol. 10, no. 1, p. 13, 2023.

[21] P. A. D. Amiri and S. Pierre, "An ensemble-based machine learning model for forecasting network traffic in vanet," *IEEE Access*, vol. 11, pp. 22 855–22 870, 2023.

[22] R. Zhu, L. Li, S. Wu, P. Lv, Y. Li, and M. Xu, "Multi-agent broad reinforcement learning for intelligent traffic light control," *Information Sciences*, vol. 619, pp. 509–525, 2023.

[23] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics.* PMLR, 2017, pp. 1273–1282.

[24] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[25] T. Suite, "Ganache," https://trufflesuite.com/ganache/, 2022, accessed: 2024-08-01.

[26] J. Benet, "Ipfs-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.

[27] “Pinata: The most powerful ipfs gateway and toolkit,” https://pinata.cloud/, 2024, accessed: 2024-08-26.

[28] Locust, “Locust: A modern load testing framework,” https://locust.io/, 2024, accessed: 2024-08-01.

[29] J. Huang, L. Kong, J. Wang, G. Chen, J. Gao, G. Huang, and M. K. Khan, “Secure data sharing over vehicular networks based on multi-sharding blockchain,” *ACM Transactions on Sensor Networks*, vol. 20, no. 2, pp. 1–23, 2024.

[30] H. Amari, Z. Abou El Houda, L. Khoukhi, and L. H. Belguith, “Trust management in vehicular ad-hoc networks: Extensive survey,” *Ieee Access*, vol. 11, pp. 47 659–47 680, 2023.

[31] S. Bouktif, A. Cheniki, A. Ouni, and H. El-Sayed, “Deep reinforcement learning for traffic signal control with consistent state and reward design approach,” *Knowledge-Based Systems*, vol. 267, p. 110440, 2023.

[32] K. Rashid, Y. Saeed, A. Ali, F. Jamil, R. Alkanhel, and A. Muthanna, “An adaptive real-time malicious node detection framework using machine learning in vehicular ad-hoc networks (vanets),” *Sensors*, vol. 23, no. 5, p. 2594, 2023.

[33] A. Boualouache and T. Engel, “A survey on machine learning-based misbehavior detection systems for 5g and beyond vehicular networks,” *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 1128–1172, 2023.

[34] Z. Wang and S. Guan, “A blockchain-based traceable and secure data-sharing scheme,” *PeerJ Computer Science*, vol. 9, p. e1337, 2023.

[35] V. Upadrista, S. Nazir, and H. Tianfield, “Secure data sharing with blockchain for remote health monitoring applications: a review,” *Journal of Reliable Intelligent Environments*, vol. 9, no. 3, pp. 349–368, 2023.

[36] S. Srivastava, D. Agarwal, B. K. Chaurasia, and M. Adhikari, “Blockchain-based trust management for data exchange in internet of vehicle network,” *Multimedia Tools and Applications*, pp. 1–19, 2024.

[37] K. G. Arachchige, P. Branch, and J. But, “An analysis of blockchain-based iot sensor network distributed denial of service attacks,” *Sensors*, vol. 24, no. 10, p. 3083, 2024.

[38] K. Godewatte Arachchige, P. Branch, and J. But, “Evaluation of blockchain networks’ scalability limitations in low-powered internet of things (iot) sensor networks,” *Future Internet*, vol. 15, no. 9, p. 317, 2023.

[39] B. Y. Kasula, “Synergizing ai, iot, and blockchain: Empowering next-generation smart systems in healthcare,” *International Journal of Sustainable Development in Computing Science*, vol. 5, no. 2, pp. 60–64, 2023.

[40] A. Wang, Y. Ye, X. Song, S. Zhang, and J. James, “Traffic prediction with missing data: A multi-task learning approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 4189–4202, 2023.

[41] X. Liu, Y. Xia, Y. Liang, J. Hu, Y. Wang, L. Bai, C. Huang, Z. Liu, B. Hooi, and R. Zimmermann, "Largest: A benchmark dataset for large-scale traffic forecasting," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[42] N. u. Sehar, O. Khalid, I. A. Khan, F. Rehman, M. A. Fayyaz, A. R. Ansari, and R. Nawaz, "Blockchain enabled data security in vehicular networks," *Scientific Reports*, vol. 13, no. 1, p. 4412, 2023.

[43] I. M. Varma and N. Kumar, "A comprehensive survey on sdn and blockchain-based secure vehicular networks," *Vehicular Communications*, p. 100663, 2023.

[44] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *International journal of web and grid services*, vol. 14, no. 4, pp. 352–375, 2018.

[45] T. T. A. Dinh, R. Liu, M. Zhang, G. Chen, B. C. Ooi, and J. Wang, "Untangling blockchain: A data processing view of blockchain systems," *IEEE transactions on knowledge and data engineering*, vol. 30, no. 7, pp. 1366–1385, 2018.

[46] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future generation computer systems*, vol. 107, pp. 841–853, 2020.

[47] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and trends® in machine learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[48] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Computing Surveys (Csur)*, vol. 51, no. 4, pp. 1–35, 2018.

Appendix A

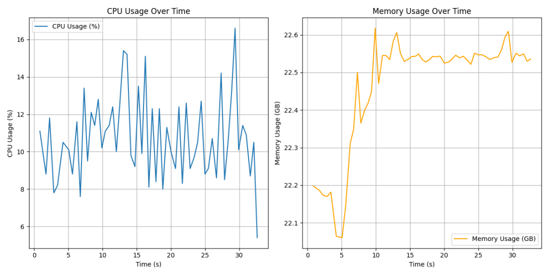
Feature Sets

Table A.1: Comparison of Initial, Feature Engineered, and Final Selected Feature Sets

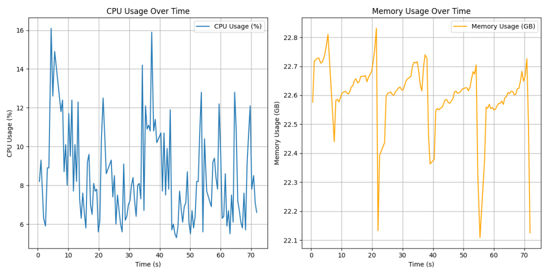
Initial Feature Set	After Feature Engineering	Final Selected Feature Set
`EventId`	`EventId`	`Severity`
`Type`	`Type`	`TimeZone (one-hot encoded)`
`Severity`	`Severity`	`LocationLat`
`TMC`	`TMC`	`LocationLng`
`Description`	`Description`	`Side (one-hot encoded)`
`StartTime (UTC)`	`StartTime (UTC)`	`City (encoded)`
`EndTime (UTC)`	`EndTime (UTC)`	`County (encoded)`
`TimeZone`	`TimeZone (one-hot encoded)`	`State (one-hot encoded)`
`LocationLat`	`LocationLat`	`ZipCode (normalized)`
`LocationLng`	`LocationLng`	`Hour`
`Distance (mi)`	`Distance (mi)`	`DayOfWeek`
`AirportCode`	`AirportCode`	`Month`
`Number`	`Number`	`Duration`
`Street`	`Street`	`Season`
`Side`	`Side (one-hot encoded)`	`Timestamp`
`City`	`City (encoded)`	
`County`	`County (encoded)`	
`State`	`State (one-hot encoded)`	
`ZipCode`	`ZipCode (normalized)`	
	`Hour`	
	`DayOfWeek`	
	`Month`	
	`Duration`	
	`Season`	
	`Timestamp`	

Appendix B

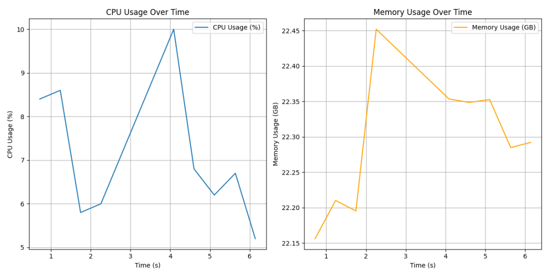
CPU & Memory Utilization Plots



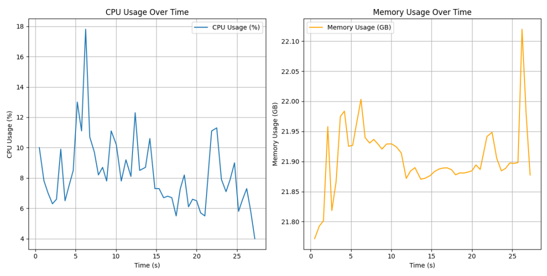
(a) Data Visualization Before preprocessing



(b) Data Cleaning



(c) Feature Engineering



(d) Data Visualization After preprocessing

Figure B.1: CPU & Memory utilization for Data Preprocessing