Traffic Accident Prediction and Warning System: Integration Use Case

Amirhossein Ghaffari[1,2,*], Huong Nguyen[1], Alaa Saleh[1], Lauri Lovén[1] and Ekaterina Gilman[1]

[1]Center for Ubiquitous Computing, University of Oulu, Oulu, Finland
[2]Infotech Oulu, University of Oulu, Oulu, Finland

Abstract
This paper presents a system for predicting and warning about traffic accidents in smart cities, aimed at enhancing urban safety through advanced data analysis and explained warning and reporting. Our system emphasizes computational efficiency and data privacy, predicting traffic accident severity with good accuracy. By integrating real data with external knowledge sources, the system produces detailed, contextually relevant reports and warnings. Implemented with effective task orchestration, our system ensures seamless integration and resource management. Evaluation results demonstrate high accuracy and scalability, highlighting its potential for practical application in smart city environments. Future work will focus on further enhancing model efficiency, exploring transfer learning for broader applicability, and conducting real-world deployments to validate system performance.

Keywords
Smart City, Transportation, Federated Learning, Edge Computing, Generative AI, RAG

1. Introduction

By 2050, over two-thirds of the global population is projected to live in urban areas [1]. Urbanization, driven by population growth and migration towards cities, presents both opportunities and challenges such as overpopulation and traffic congestion [2]. Developing smart cities is a strategic approach to mitigate these challenges.

A "smart city" integrates information and communication technology to enhance urban living [3]. This concept emphasizes the interconnection of community, people, and technology, aiming to prioritize human needs [4]. Urban mobility and transportation are significant challenges, with traffic congestion and accidents being major concerns. Annually, traffic accidents result in 1.35 million deaths globally, underscoring the critical need for effective accident prevention measures [5].

In large-scale Internet of Things (IoT) ecosystems, efficient data processing is crucial. Centralized cloud servers face latency and security challenges for many application domains, making real-time processing difficult [6]. Edge computing aims to address these limitations by bringing computational resources closer to data sources, enabling timely processing and reducing latency [7, 8]. When edge computing is integrated with AI, known as EdgeAI, real-time urban decision-making could be facilitated [9, 10]. For example, models trained to predict road weather [11, 12] or traffic congestion can operate on edge devices located closer to the sites, providing immediate insights to traffic management systems. Nonetheless, due to the resource constraints of edge devices, certain computationally intensive tasks might still be offloaded to the cloud or other powerful nodes within the city network. That approach requires loosely-coupled architectures and distributed algorithms [13, 14].

A lot of research proposes AI support for smart transportation systems from various perspectives. For example, Bortnikov et al. [15] detect accidents by training a 3D Convolutional Network on the data generated by a video game, Uma and Eswar's [16] develop yawning detection of the drivers, Liu et al [17] concentrate on traffic flow prediction. However, the majority of related work focuses on a single type of AI module specifically developed for the task at hand, neglecting the capabilities of integrating their approach with other kinds of AI modules to create a more comprehensive support system. Additionally, response time is often overlooked in the assessment of related work, with a primary focus on accuracy. To our knowledge, no existing work incorporates multiple types of AI modules, raising questions about the integration and applicability of these separate modules into a cohesive framework.

To address these gaps, we present our integrated system, containing two AI modules: first, Federated Learning (FL) [18] model to predict traffic accident occurrences and estimate severity and second, Generative Artificial intelligence (GenAI) to generate reports and warnings. Moreover, we utilized k0s, a lightweight Kubernetes dis-

KiL'24: Workshop on Knowledge-infused Learning co-located with 30th ACM KDD Conference, August 26, 2024, Barcelona, Spain
*Corresponding author.
amirhossein.ghaffari@oulu.fi (A. Ghaffari); huong.nguyen@oulu.fi (H. Nguyen); alaa.saleh@oulu.fi (A. Saleh); lauri.loven@oulu.fi (L. Lovén); ekaterina.gilman@oulu.fi (E. Gilman)
0009-0006-9264-8681 (A. Ghaffari); 0000-0001-9067-3396 (H. Nguyen); 0000-0002-6725-7383 (A. Saleh); 0000-0001-9475-4839 (L. Lovén); 0000-0001-9816-2240 (E. Gilman)

tribution, for efficient task orchestration [19]. The task orchestration capabilities of k0s are crucial for seamlessly integrating the FL models and Retrieval-Augmented Generation (RAG) processes across multiple edge nodes. This enables automated deployment, scaling, and management of tasks, ensuring high availability, fault tolerance, and robust performance monitoring for our accident prevention warning system.

The contributions of this work can be summarized as follows:

1. We integrate two different kinds of AI modules into a coherent distributed system supporting accident prevention. We comprehensively evaluate this system and analyze the related challenges and opportunities.
2. We orchestrate tasks and monitor our system, examining its feasibility for real-world smart city environments.

The remainder of this article is organized as follows. sec:relatedwork discusses related work, while sec:design describes the system design, and sec:implementation details the implementation. sec:eval then provides a detailed system evaluation and metrics, sec:discussfuture discusses our findings, implications, and future research directions, and sec:conclusion concludes the work.

2. Related Work

2.1. Intelligent Transportation System in Smart City

Intelligent Transportation Systems (ITS) are essential for the advancement of smart cities, with many recent studies dedicated to improving urban traffic management and safety. Here, we discuss several key works that have made significant contributions to this field. As an example, Hasan et al. [20] used the Google Distance Matrix and Directions APIs to provide advanced traffic jam alerts. Their Internet of Vehicles (IoV) module detects accidents and, with the assistance of the National Data Warehouse and a GPS module, notifies the nearest clinic. They developed an Android application for routing suggestions and employed an Arduino with a Sonar sensor, temperature sensor, gyroscope, piezo sensor, and GSM module as the core processing unit.

Working on one of the most trendy applications, Bortnikov et al. [15] developed a 3D Convolutional Neural Network (CNN) to recognize accidents automatically. They trained the CNN using a custom video game to create accident scenes with various weather and lighting conditions, adding noise to diversify the data. The model was then tested on real traffic videos from YouTube. The novelty of this research lies in the use of video games to generate datasets, which are challenging to replicate in real-life scenarios. Yu et al. [21], with the same aim, proposed a Deep Spatio-Temporal Graph Convolutional Network for traffic accident prediction for Beijing traffic data, which was collected hourly over three months and includes accident records (time and location), vehicle speeds, meteorological conditions and points of interest. Recent research has considered informing other vehicles after detecting traffic accidents using IoT, IoV, and related technologies. Zhou et al. [22] proposed an accident detection algorithm based on spatio-temporal feature encoding with a multilayer neural network. This method first detects border frames as potential accident frames, then encodes the spatial relationships of detected objects to confirm an accident. The process involves using Histogram of Oriented Gradients and ordinal features initially, followed by CNN feature encoding and object relationship detection with a multilayer neural network. A trained Support Vector Machine then confirms the presence of an accident.

Another approach involves efforts to reduce accidents before they occur is the work of Uma and Eswari [16], which developed a prototype using a Raspberry Pi and Pi Camera, along with sensors to monitor driver's eye movements, detect yawning, and identify toxic gases and alcohol consumption. This system, employing the Haar Cascade algorithm for face detection and calculation of Eye Aspect Ratio and Mouth Aspect Ratio, estimates risk through these feature analysis. Besides, to identify accident hot spots, Le et al. [23] used Road Traffic Accident data over three years in Hanoi, Vietnam, to develop a GIS-based statistical analysis technique. This method assesses the influence of accident severity on temporal-spatial patterns, identifying accident hotspots in relation to specific times of day and seasons.

Beyond the mention in [24] of the potential service supports of cloud to autonomous vehicles applications, edge computing is playing a pivotal role in reshaping traffic management in smart cities. Within this domain, Mohamed's [25] and Zhou's research groups [26] demonstrated substantial improvements in traffic management and reduced congestion durations through an edge-based model for real-time traffic data analysis. Besides, to achieve low latency and high prediction accuracy on vehicle identification at the edge, Wan et.al [27] have eliminated redundant frames from collected videos and presented an approach for real-time video processing.

In a similar manner, Ke et al. [28] developed a multi-thread system for real-time detection of near-crash events in traffic, using video analytics on dashcams. Leveraging edge power, their system efficiently performs object detection and tracking directly from the video feeds on board. This approach involves removing irrelevant video to conserve bandwidth and storage while collecting diverse and valuable data for traffic safety such as road user type, vehicle trajectory, vehicle speed, brake switch, and throttle. The approach from Ke et al. demonstrates considerable promise for widespread application due to its low cost, real-time processing, high accuracy, and broad compatibility with various vehicles and camera types.

Additionally, a recent work by Nguyen et al. [29] utilized Blockchain technology alongside edge computing to develop a reliable and transparent situational awareness system for autonomous vehicles. Their system broadcasts notifications and alternative route suggestions from the nearest edge station when congestion or accidents are detected by other vehicles, using various sensing data sources, including dashcam images and environmental factors like weather, temperature, and humidity. The use of Blockchain in their study ensures the data validity and integrity, as well as facilitates collaboration among different service providers. However, despite the recognized vision and applications, Zhou et al. [30] emphasized that employing edge computing in ITS always comes with inherent challenges related to sensor failure, and privacy protection concerns, which must be addressed for effective implementation.

2.2. FL in ITS

Building on the challenges identified by Zhou et al. [30] particularly concerning privacy protection, FL recently has been used more in smart cities. Amongst many applied domains within urban environments, the extension of FL applications in traffic systems is mostly leveraged for traffic monitoring and accident predictions.

FedGRU - FL-based Gated Recurrent Unit (GRU) neural network [17] is one of the pioneering works for traffic flow prediction (TFP) with federated deep learning that comparably performs to other advanced competing methods without compromising the privacy and security of data. Additionally, as proved by experiments, the joint announcement protocol proposed in this paper helps in reducing communication overhead by 64.10% compared with centralized models, implicating the scalability of FedGRU for bigger networks.

With the same motivation to address the privacy exposure risk of centralized machine learning, Qi et al. [31] presented a fully decentralized FL network, utilizing a Blockchain-based FL architecture as opposed to the conventional vanilla framework. The authors employed the local differential privacy technique to protect vehicle location and utilized GRU to achieve accurate TFP. Performance and security comparisons were also made among different machine learning models and with/without the use of blockchain. Qi et al. also conducted comparative analyses in terms of both performance and security, examining various machine learning models and contrasting scenarios with and without blockchain implementation. Concerning the monitoring of traffic congestion, typical systems begin by detecting vehicles and subsequently estimating traffic flow density.

In their research, Xu et al. [32] employed remote sensing images for this purpose, while Chougule et al. [33] continuously used the estimated traffic density from intersection-captured images to dynamically adjust the duration of green light and schedule the timing of signals across all lanes.

As one of the highlights in the narrow field of applying FL on ITS: risk detection, Yuan et al. [34] introduced FedRD, a framework combining edge-cloud computing, FL, and differential privacy techniques for intelligent road damage detection and warning. The framework not only improves detection performance and coverage area but also addresses privacy concerns through Individualized Differential Privacy with pixelization technique.

Comprehensive evaluations demonstrate FedRD's capability to deliver high detection accuracy and wider coverage while preserving user privacy, even in scenarios where edge devices have limited data. This groundbreaking effectiveness sets a new benchmark in the field.

2.3. GenAI in ITS

Recently, GenAI has garnered significant attention in several applications, including ITS, due to its advantages and flexibility. By analyzing data from various sources, such as roadside sensors, vehicles, and traffic signals, GenAI enhances urban operations by detecting patterns, identifying trends, and providing accurate predictions and advice. With the leverage of natural language processing, GenAI can present these predictions in human-understandable language, making these technologies more accessible and practical for smart services [35]. See prior works [36, 37] for examples of how GenAI integrated into many services within cities. As another example in ITS, Impedovo et al. [38] propose a deep generative model to predict weekday vehicular traffic flow to prevent accidents in the most critical areas and improve continuity by reducing traffic. More notably, RAG, first introduced by Lewis et al. in 2020 [39]l, stood out as a part of this GenAI world, representing a distinct approach to generating text, informed reasoning, and supporting decision-making.

Its application in ITS is not really popular, however, there are some notable works. For instance, Dai et al. [40] integrated RAG into autonomous driving systems to enhance decision-making processes. According to the authors, the use of RAG in their work addresses the problem of impractical generated content from the mainstream foundation models nowadays, such as GPT4 or LLaMa. It helps these models enhance the reliability of their outputs during the generation phase by dynamically retrieving accurate contextual information from outer databases (e.g. updated traffic rules, driving experiences, or human pref-

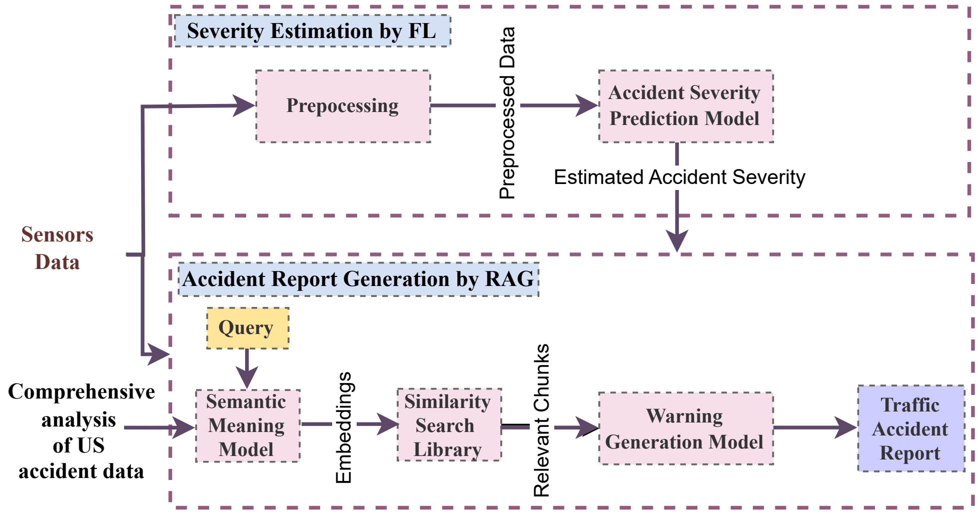

Figure 1: System workflow. This figure illustrates the key components of the core system, Federated Learning (FL) and Retrieval-Augmented Generation (RAG). Using preprocessed weather and road traffic sensors, FL predicts accident severity. Within the RAG framework, the Semantic Meaning Model creates embeddings for documents and queries. The Similarity Search Library selects the most relevant document chunks based on similarity. Finally, the Warning Generation Model generates a traffic accident report that incorporates data analysis and future recommendations.

erence). Similarly, Ding et al. [41] utilized RAG for more controlled generation of traffic scenarios. Specifically, RealGen [41] synthesizes new scenarios by combining behaviors from multiple retrieved examples in a gradient-free manner, using templates or tagged scenarios. This in-context learning framework provides versatile generative capabilities, including scenario editing, behavior composition, and the creation of critical scenarios, thus enhancing the adaptability and precision of synthetic data generation for various applications. Most recently, in his Master's thesis, Mohanan [42] evaluated eight embedding RAG models for a chatbot tailored to Indian Motor Vehicle Law.

As can be seen, prior research typically focuses on a single module, such as risk estimation or warning generation, limiting possible support for ITS. This raises an open question: *"Is it possible to integrate all diverse components into a cohesive and comprehensive ITS framework?"* This is where our work positions.

3. System Design

This article presents a system for predicting and preventing traffic accidents. It is capable of predicting the possible accidents based on the traffic conditions and other available data, and provides detailed textual comments to the user explaining the grounds leading to such estimation. Figure 1 illustrates the overall system flow, highlighting the interplay between the key components: Federated Learning (FL) and Retrieval- Augmented Generation (RAG).

This integrated system combines the strengths of RAG and FL to ensure high-quality outputs while maintaining data privacy and relevance. FL enhances the accident severity prediction model while maintaining data privacy. The RAG system uses integration between the warning generation model and the knowledge retrieval model to enhance the generation process with relevant external data, improving context and accuracy.

Our training approach starts from data preprocessing. The preprocessed dataset is then used to train the FL model for traffic accident risk estimation. The predictions, along with the sensors' real-time data, are utilized as input for the RAG model. The RAG model integrates advanced retrieval mechanisms with state-of-the-art language generation capabilities to produce detailed warnings and reports for traffic accidents.

To efficiently manage and deploy these components, we use a task orchestration tool. This tool ensures seamless integration and coordination among the various models, automates deployment, and scales the system as needed. Additionally, it facilitates robust performance monitoring, ensuring high availability and fault tolerance across the system.

3.1. Dataset

This study uses US Accidents (2016-2023) dataset [1][43] from Kaggle, distributed under CC BY-NC-SA 4.0 license. This dataset comprises a vast collection of over 7.7 million (7,728,394) traffic accident records, covering 49 states of the USA from February 2016 to March 2023. The accident data were collected using multiple APIs that provide streaming traffic incident data captured by various entities, including the US and state departments of transportation, law enforcement agencies, traffic cameras, and traffic sensors within the road networks. The data includes detailed information on accident severity, location, time, and weather conditions. This dataset was utilized to train the FL models for traffic accident prediction.

3.2. Federated Learning

Our application relies on FL model for accident risk estimation. FL was selected based on two primary considerations: data privacy and collaborative enhancement.

1. **Privacy:** Addressing privacy concerns, vehicles in a real scenario do not transmit raw data, which could potentially reveal sensitive information. Instead, only model parameters will be sent, ensuring that individual data remains secure and private. This cannot be done with traditional centralized learning when all data need to be sent to a central server for training.
2. **Collaboration:** When a vehicle updates and shares its model parameters, it contributes to the overall learning process. This collective effort leads to an improvement in the overall model's performance, as it can learn from a wide range of diverse and localized inputs. The shared knowledge enables more accurate and robust risk estimation.

The training data features provide a detailed view of accident records, including the specifics of the accidents, the geographic locations, the prevailing weather conditions at the time of the accidents, and various environmental and contextual factors that may be relevant to analyzing the accidents. In a real scenario, the vehicle's onboard computing system uses these inputs to continuously update its local model, learning from real data. Once the training is done, the model parameters will be sent to the nearby edge server. The server, after receiving a sufficient amount of models will start doing the aggregation to get the global model, which is then sent back to the participating vehicles. When this whole process is complete, we finish one communication round and continue to the next round.

[1]https://www.kaggle.com/datasets/sobhanmoosavi/us-accidents

3.3. Retrieval-Augmented Generation

RAG combines an information retrieval component with a text generator model to provide situational information and guidance [44]. In the ITS context, RAG can integrate various external data sources to analyze and report traffic accidents, identifying risk factors and details [45]. This makes the system more dynamic and adaptable to new information. In our system, see Figure 1, RAG provides textual accident warnings to the end user, along with explanations of how the estimates were derived.

Knowledge retrieval model It is designed to find the most relevant information from an external knowledge base in response to the query. This enhances FL model output and sensor data with relevant information. We use SentenceTransformers[2] as a retrieval model based on similarity search.

Warning generation model: It is designed to generate new content using language models. It uses the retrieved information by the retrieval model and FL-output details to generate a response. For our system, we use **gpt-3.5-turbo-0613**[3] to create contextually relevant warnings and detailed reports. The accident report includes the severity of the accident, the location and traffic control procedures, and guidance and actions.

3.4. Task Orchestration and Monitoring

Effective resource management and device health monitoring are essential for enhancing the responsiveness of smart city services. This requires comprehensive system monitoring that spans from edge devices to the cloud. The deployment of applications on edge devices necessitates advanced task orchestration platforms, which must be carefully selected based on specific requirements. Given that edge devices typically have limited resources, the chosen tool must operate smoothly under such constraints. For the proposed system, k0s[4] has been selected. We selected k0s because of its minimal resource consumption on edge devices and its straightforward and rapid implementation process, supported by comprehensive documentation and active developer forums. It typically operates with as little as 1 CPU and 512 MB of RAM on each controller node and 1 GB of RAM on each worker node, which aligns well with the capabilities of edge devices. However, the minimum requirements increase when the number of worker nodes is increased. Additionally, numerous monitoring options compatible with k0s are available. k0s is packaged as a single, self-extracting binary which embeds Kubernetes binaries. It has many benefits, such as it has no OS level dependencies and everything can be, and is, statically compiled.

[2]https://sbert.net/
[3]https://platform.openai.com/docs/models/gpt-3-5-turbo
[4]https://docs.k0sproject.io/stable/

4. System Implementation

4.1. Risk Estimation with FL

4.1.1. Preprocessing

The preprocessing phase for our system includes a series of essential data preparation steps to ensure the quality of the dataset for further analysis:
1. Data Cleaning: Duplicated and missing values were removed.
2. Feature Engineering: To enhance the informativeness of the dataset, a new feature, called "Comfort_Index" following Equation 1 is created.

$$Comfort\_Index = (Temperature - 32) * (Humidity/100) \tag{1}$$

3. Data Resampling: To address the imbalance issue, both random oversampling and undersampling of the data was done to ensure that each label had an equal distribution.
4. Data Transformation: Done according to feature type:

- **Categorical Data:** One-hot encoding was applied to categorical columns, except for "Street," "State," and the target label "Severity".
- **Boolean Data:** Columns with two distinct values were binarized, converting them to 0 and 1.
- **Numeric Data:** Columns containing numeric data were left unchanged, preserving their original values.

5. Standardization: The dataset was then subjected to StandardScaler standardization. This process ensured that all features had consistent scales and values within a particular range.

4.1.2. FL Training and Prediction

To simulate a real-world scenario using our chosen dataset, we distributed the data across several nodes and established certain assumptions. This section will elaborate on those details.
Distribution: The data is divided into five equal parts, corresponding to five nodes in the system. We also make sure the number of samples of each label is distributed equally among clients.
Model Training: Each client trains its local model, consisting of three fully connected layers. Training specifications include the use of the cross-entropy loss function, Adam optimizer with a learning rate of 1e-3, and a batch size of 32. After ten training epochs, the locally trained models are aggregated by the server into a global model, and the global parameters are saved at each checkpoint, here at each communication round, before being sent back to the participants for training in the next round. The FL training process concludes after ten communication rounds. At this stage, various model architectures, encompassing differing layer counts and hyperparameters, were evaluated over 50 communication rounds to observe the trend and convergence in via its performance. The selected model outperformed alternatives; models with reduced layers demonstrated inferior outcomes (3-4%), while configurations with additional layers, despite a 3% accuracy improvement, incurred prolonged training duration and converged to local, rather than global, optima. See Table 1 for details.

Table 1
Risk estimation models comparison: accuracy (%) and training time (hours)

	Simple	Chosen	Complex
Acuracy (%)	67.09	71.15	74.42
Time (hours)	3.461	4.042	5.603

Prediction: The input is a sensors real-time data. This data goes first through the 5-step preprocessing process (refer to Sub-section 4.1.1) to get the feature vector, which will be fed as input for the model to predict.

4.2. Warning Generation with RAG

Using the RAG model, we retrieve text passages using an input sequence. During the generation of the target sequence, we include these passages as additional context. Our model leverages two components, which are implemented in LangChain[5]. A retriever that retrieves relevant text snippets in response to a user's query or prompt based on knowledge source which is uploaded using built-in document loader from LangChain.

In our system, we rely on the US traffic accident database as an external knowledge source, containing a comprehensive analysis of US traffic accident data [46]. This report provides insight into preventive measures and policy recommendations for decreasing traffic accidents in the US based on detailed analyses by state, time, and contributing factors such as weather. The retrieval process begins with loading documents using a tool in LangChain. This process is enhanced by a splitter tool, also integrated into LangChain, designed to segment extensive texts into smaller chunks based on a specified chunk size by examining characters recursively which is crucial for the efficient handling of large textual data.

For the creation of text embeddings, we employ HuggingFaceEmbeddings, a specialized embedding model from the Hugging Face library[6] within LangChain. This model transforms the segmented text chunks into numerical vectors, facilitating their computational handling.

[5]https://www.langchain.com/
[6]https://huggingface.co/

To store these embedding vectors in a vector store, we utilize the FAISS library[7], a robust vector database. It enables effective similarity search by identifying text chunk vectors most similar to the question vector. This process is vital to determine which portions of the knowledge source are most pertinent to the input query. This is for later retrieval at query time based on the k argument which finds the top k most relevant text chunk vectors for each query. Table 2 summarizes the RAG parameters used.

The generator creates a more detailed, factual, and relevant response based on the original input and retrieved documents. The original input represents the severity of an accident, derived from the FL output and complemented by sensor real-time data. For the generation of coherent and contextually relevant text, the original input and the retrieved documents are fed into **gpt-3.5-turbo-0613**, a sophisticated pre-trained language model. Based on the content of these documents, the model generates coherent and contextually relevant text grounded in real-world information. Figure 2 illustrates an example of a traffic accident report generated by RAG.

Table 2
Summary of RAG parameters used

Parameter	Value
Text splitter type	RecursiveCharacterTextSplitter
Chat model	ChatOpenAI
ChatOpenAI model name	gpt-3.5-turbo-0613
Vector store	FAISS
Embeddings type	HuggingFaceEmbeddings
Embeddings model name	sentence-transformers/all-mpnet-base-v2
Search type	similarity
Chunk size	2000

4.3. Task Orchestration and Monitoring

As discussed in Sub-section 3.4 we opted for k0S, which is ideal for our needs and simple in implementation. We used Lens IDE[8] which is a Kubernetes IDE to manage the cluster and monitoring of the whole system. It allows for comprehensive oversight of nodes, pods, and resource monitoring. Monitoring involves tracking the usage of CPU, memory, storage, and network bandwidth, and monitoring device safety and functionality to detect any potential problem. We containerized our application using Docker[9] and deployed our application using Lens IDE and k0s task orchestration tool. We used Cluster metrics in the Lens IDE to monitor the resources efficiently.

[7]https://faiss.ai/index.html
[8]https://k8slens.dev/
[9]https://www.docker.com/

```
Traffic Accident Report:
Accident Data:
- Street: US Highway 22
- State: NJ
- Start Latitude: 40.65562
- Start Longitude: -74.40149
- Crossing: False
- Give Way: False
- Junction: False
- No Exit: False
- Railway: False
- Roundabout: False
- Station: True
- Stop: False
- Traffic Calming: False
- Traffic Signal: False
- Distance (mi): 0.167
- Temperature (F): 56.09
- Wind Chill (F): 72.98
- Humidity (%): 42.97
- Pressure (in): 29.59
- Visibility (mi): 9.99
- Wind Direction: NNW
- Wind Speed (mph): 9.19
- Precipitation (in): 0.0016
- Weather Condition: Fair
- Sunrise/Sunset: Night
- Comfort Index: 10.35
- Severity: 2
Analysis of the Accident: Based on the accident data, the accident occurred on US Highway 22
in NJ. The accident did not occur at a crossing, junction, or railway. However, there was a
station nearby. The distance of the accident location from the starting point was approximately
0.167 miles. The weather conditions at the time of the accident were fair, with a temperature
of 56.09°F, wind chill of 72.98°F, humidity of 42.97%, pressure of 29.59 in, and visibility
of 9.99 miles. The wind was coming from the NNW direction at a speed of 9.19 mph, and there
was a slight precipitation of 0.0016 inches.
Recommendations for Future Prevention:
1. Increase Public Awareness: It is crucial to educate the public about safe driving practices
and the importance of following traffic rules and regulations. Public awareness campaigns
should be conducted to promote responsible driving behavior and reduce the likelihood of
accidents.
2. Improve Road Safety Measures: Implementing traffic calming measures, such as speed bumps
or roundabouts, can help reduce the risk of accidents. Additionally, installing traffic signals
at appropriate locations can improve traffic flow and prevent collisions.
3. Enhance Emergency Response: Ensure that emergency services, such as fire and medical teams,
are well-equipped and trained to handle road traffic accidents efficiently. Regular drills
and training sessions should be conducted to improve response times and minimize casualties.
4. Regular Safety Inspections: Regular inspections of roads, signage, and traffic signals
should be conducted to identify and address any potential safety hazards. Prompt repairs and
maintenance should be carried out to ensure the safety of drivers and pedestrians.
5. Collaboration with Law Enforcement: Collaborate with local law enforcement agencies to
enforce traffic laws and regulations effectively. Increase police presence on the roads to
deter reckless driving and enforce speed limits.
6. Continuous Monitoring of Weather Conditions: Implement a system to continuously monitor
weather conditions and provide real-time updates to drivers. This will help drivers make
informed decisions and adjust their driving behavior accordingly during adverse weather
conditions.
By implementing these recommendations, we can work towards preventing similar accidents in
the future and ensuring the safety of all road users.
```

Figure 2: An example of a traffic accident report generated by RAG

5. System Evaluation

To assess the system's performance, several key metrics were employed. We want to ensure that all the components work perfectly both independently and in the integrated system. First, we monitored the accuracy of the FL model for risk estimation, assessing its ability to predict traffic accident severity. This evaluation utilized the dataset for training the model. Additionally, the quality and relevance of warnings and reports generated by the RAG model were assessed. The system's prompt responsiveness was also tested, particularly how quickly it can generate alerts and warnings based on incoming data. Furthermore, the resource management aspect was evaluated to ensure that the system's resource usage is optimized and well-maintained. The developed system was deployed and tested on a real cluster of three nodes with k0s equipped with the monitoring application.

5.1. Risk Estimation Evaluation

5.1.1. Accuracy

We monitor the training process of the FL model in terms of accuracy, loss, and convergence. The training for 50 communication rounds with 5 training clients takes up to 4.042 hours.

Figure 3 plots the training accuracy in the upper graph and the training loss in the lower graph. The model demonstrates convergence approximately by round 30 at 71.15%, as depicted in the upper plot. Initially, model accuracy exhibits an upward trend from round 0 to 30, albeit with fluctuations observed around rounds 15-17 and 21. Subsequently, after round 30, the risk estimation model appears to have reached a plateau in accuracy, becoming converged. This is also reflected in the lower graph of training loss.

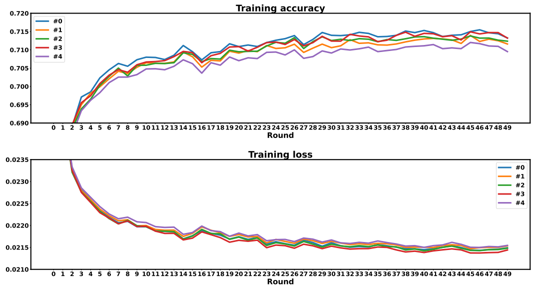

Figure 3: Risk estimation training accuracy (top) and loss (bottom)

It is, however, possible for low power-resource devices to terminate the training process at an earlier stage, such as after round 10 or 20, with negligible tradeoffs in accuracy.

5.1.2. Total latency trends

The bar graph (referred to Fig. 4) depicting the total latency for predictions reveals a clear trend: as the number of inputs processed simultaneously increases, so does the time required for prediction.

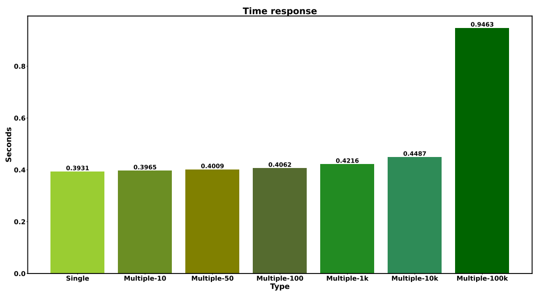

Figure 4: Total latency trends of risk estimation

Starting from a swift 0.3931 seconds for a single input, the latency moderately rises for batches of 10 and 100 inputs, reaching 0.4062 seconds, suggesting the model handles small to moderate increases in input size efficiently.

However, as input sizes increase to 1,000 and 10,000, the total latency grows more substantially, hitting 0.4487 seconds for 10,000 inputs. This increment continues, even more sharply, with the model taking 0.9463 seconds to predict outcomes for 100,000 inputs concurrently.

Overall, this evaluation outcome underscores the FL model's scalability with a total latency, not only for small input batches but also optimized for larger ones. Nevertheless, it should be noted that the measured time can be different among different working devices.

5.2. Accident Warning Report Evaluation

To evaluate the quality of accident warning report generated by RAG, we have used correctness, relevance, and faithfulness as criteria to assess LLM outputs[10]. We used gpt-3.5-turbo-0613 for the evaluation task to contextually analyze and interpret generated reports according to the criteria.

Correctness is based on the LLM's internal knowledge. However, given the potential unreliability of the LLM's knowledge base, we enhanced the evaluation method by incorporating reference labels. This provides an external benchmark for correctness. The evaluation process produces a dictionary containing key metrics: "score", a binary integer from 0 to 1 indicating compliance with the criteria, "value", which is either "Y" (Yes) or "N" (No) based on the score, and "reasoning", which outlines the LLM's chain of thought. Relevance evaluates the relevance and focus of the generated answer in relation to the provided prompt. Faithfulness assesses the factual consistency of the generated answer against the given context and reference documents. Using this approach, we ensure not only that the generated content meets the prompt's specific requirements. It also remains true to the factual information provided in the reference material. Figure 5 illustrates an example of RAG output evaluation.

Based on correctness, relevance, and faithfulness criteria, the evaluation shows that the output accurately represents an actual quote. Throughout the evaluation output, all necessary elements are addressed in a comprehensive, well-structured, and well-written manner. Based on the evaluation output, the response summarizes accident data and provides a comprehensive analysis of weather conditions at the time of the accident, including visibility and severity. Additionally, it provides recommendations for preventing accidents in the future relevant to the reference.

[10] https://python.langchain.com/docs/guides/evaluation/string/criteria_eval_chain

```
Correctness_criteria:

{'reasoning': To determine if the submission meets the criteria, we need to evaluate the correctness, accuracy, and factual nature of the submission.

1. Check if the submission correctly presents the accident data, including the street, state, latitude, longitude, and various factors related to the accident.

2. Verify if the submission accurately describes the weather conditions at the time of the accident, including temperature, wind chill, humidity, pressure, visibility, wind direction, and precipitation.

3. Assess whether the submission accurately provides information about the severity of the accident, distance, sunrise/sunset, and comfort index.

4. Evaluate if the recommendations for future prevention are reasonable and relevant to the accident scenario.

Based on the above reasoning, the submission meets the criteria if all the above conditions are satisfied. 'score': 1, 'value': 'Y'}

Relevance_criteria:

{'reasoning': To determine if the submission meets the criteria of relevance, we need to compare the content of the submission with the provided data.

We will check if the submission accurately refers to a real quote from the text.

- The submission provides a detailed analysis of the accident data, including the street, state, and various accident factors. It also mentions the weather conditions, severity, and recommendations for future prevention based on the given data.

- The submission accurately reflects the information provided in the data.

- Therefore, the submission meets the criteria of relevance.

Based on the above reasoning, the conclusion is that the submission meets all the criteria.'score': 1, 'value': 'Y'}

Faithfulness_criteria:

The assistant's response is faithful to the reference context. It accurately summarizes the accident data provided in the user question and provides a detailed analysis of the accident. It also offers recommendations for future prevention. The response is comprehensive and covers all the relevant aspects of the accident data.
```

Figure 5: An example of RAG output evaluation criteria

5.3. Task Orchestration and Monitoring

We utilized a simplified demonstration setup comprising one controller node and two worker nodes to test the deployment of the system to the distributed environment. The technical characteristics of our system are as follows: The controller node is equipped with an Intel Core i7-6700HQ CPU, an NVIDIA GeForce GTX 960M GPU, and 16 GB of RAM. One of the worker nodes is identical to the controller node, featuring an Intel Core i7-6700HQ CPU, an NVIDIA GeForce GTX 960M GPU, and 16 GB of RAM. The other worker node is equipped with an Intel Core i5-1135G7 CPU and 16 GB of RAM. The system was successfully deployed and operated as expected, effectively generating warnings in response to simulated input data. Additionally, we employed Lens IDE to monitor data outputs and to oversee the resource usage on the controller node. A screenshot of the Lens IDE is provided in Figure 6 to demonstrate how the cluster is controlled.

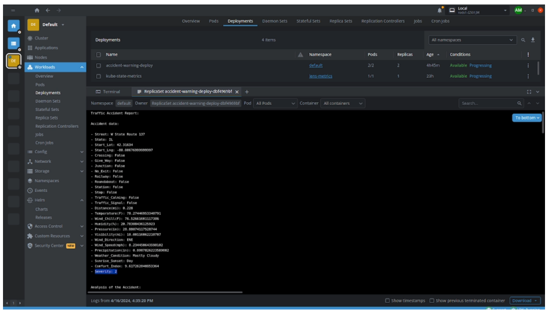

Figure 6: Lens IDE logs output

6. Discussion and Future Work

The development and integration of FL and RAG into an ITS service presents several key findings and areas for further research.

Our FL model demonstrated good performance in predicting traffic accident severity, achieving a convergence point after approximately 30 communication rounds. This suggests that FL can effectively utilize distributed data for predictions while maintaining data privacy. Additionally, the scalability of the FL model was evident from the total latency evaluations, which showed reasonable prediction times even with increasing input sizes, indicating the model's applicability in real scenarios.

The RAG model generated detailed and contextually relevant reports and warnings based on simulated real-time inputs. This was validated through evaluations focusing on correctness, relevance, and faithfulness. The integration of real-time data and FL with external knowledge sources ensured that the generated content was not only accurate but also practical for end-users, such as traffic management authorities.

The use of k0s for task orchestration proved to be effective, enabling seamless integration and management of various system components. The monitoring capabilities provided by Lens IDE ensured the system's robustness and allowed for efficient resource management. Testing on a simulated cluster confirmed the system's reliability and scalability.

While our system shows promising results, several areas warrant further investigation and development. Future work should focus on strengthening privacy-preserving techniques within the FL framework.

In our design of the FL model, we prioritized simplicity and efficiency to predict accident severity. This approach was intended to minimize the computational load. For future work, it would be advantageous to enhance the FL model by exploring other lightweight models. This could potentially improve the accuracy while maintaining the model's efficiency.

Exploring the feasibility of using transfer learning methods to transfer knowledge gained about each state or district to other districts or states can be beneficial.

Developing user-friendly interfaces for traffic management authorities and end-users will be crucial for effective system adoption. This involves designing intuitive dashboards and visualization tools to present predictions and warnings in an accessible manner. Implementing and testing the system in real-world smart city environments will provide valuable insights into its performance and scalability. Collaborations with city authorities can facilitate this process and help refine the system based on practical feedback.

7. Conclusion

This paper presents a service in smart cities integrating FL and RAG to enhance traffic risk prediction and management in smart cities. Our findings demonstrate the system's accuracy, efficiency, and potential for real-world applications. The FL model achieved a good predictive performance while preserving data privacy. The RAG model produced detailed and relevant reports, aiding in effective traffic management.

Task orchestration using k0s ensured seamless integration and robust performance monitoring. Future work will focus on enhancing privacy, scalability, and real-world testing, aiming for broader deployment and integration. Our system offers a promising approach to addressing urban safety challenges, contributing to the development of smarter and safer cities.

Acknowledgments

We would like to thank Prof. Timo Ojala for fruitful discussions on the topic. This research is financially supported by the Research Council of Finland (former Academy of Finland) through the 6G Flagship Program (346208) and UrBOT project (323630), by Business Finland Neural pub/sub research project (diary number 8754/31/2022), by EU Horizon 2020 IDUNN project (101021911), and by the Emerging projects program, Infotech Oulu.

References

[1] Our World in Data, Urbanization, 2023. URL: https://ourworldindata.org/urbanization, accessed: November 13, 2023.

[2] Q. Wang, L. Li, The effects of population aging, life expectancy, unemployment rate, population density, per capita gdp, urbanization on per capita carbon emissions, Sustainable Production and Consumption 28 (2021) 760–774.

[3] E. Gilman, et al., Addressing data challenges to drive the transformation of smart cities, ACM Transactions on Intelligent Systems and Technology (2024).

[4] A. Adel, Future of industry 5.0 in society: Human-centric solutions, challenges and prospective research areas, Journal of Cloud Computing 11 (2022) 1–15.

[5] World Health Organization, Global Status Report on Road Safety 2018: Summary, Geneva: World Health Organization; 2018 (WHO/NMH/NVI/18.20). Licence: CC BY-NC-SA 3.0 IGO, 2018. License: CC BY-NC-SA 3.0 IGO.

[6] T. Alam, Cloud-based iot applications and their roles in smart cities, Smart Cities 4 (2021) 1196–1219.

[7] N. H. Motlagh, et al., Edge computing: The computing infrastructure for the smart megacities of the future, Computer 55 (2022) 54–64.

[8] N. H. Motlagh, et al., Digital twins for smart spaces-beyond iot analytics, IEEE internet of things journal (2023).

[9] M. A. Rahman, M. S. Hossain, A. J. Showail, N. A. Alrajeh, A. Ghoneim, Ai-enabled iiot for live smart city event monitoring, IEEE Internet of Things Journal (2021).

[10] Peltonen, Ella and others, The many faces of edge intelligence, IEEE Access 10 (2022) 104769–104782. doi:10.1109/ACCESS.2022.3210584.

[11] L. Lovén, et al., Mobile road weather sensor calibration by sensor fusion and linear mixed models, PloS one 14 (2019) e0211702.

[12] V. Karsisto, L. Lovén, Verification of road surface temperature forecasts assimilating data from mobile sensors, Weather and Forecasting 34 (2019) 539–558.

[13] H. Kokkonen, et al., Autonomy and intelligence in the computing continuum: Challenges, enablers, and future directions for orchestration, arXiv preprint arXiv:2205.01423 (2022).

[14] L. Lovén, et al., Edison: An edge-native method and architecture for distributed interpolation, Sensors 21 (2021) 2279.

[15] M. Bortnikov, A. Khan, A. M. Khattak, M. Ahmad, Accident recognition via 3d cnns for automated traffic monitoring in smart cities, in: Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 2 1, Springer, 2020, pp. 256–264.

[16] S. Uma, R. Eswari, Accident prevention and safety assistance using iot and machine learning, Journal of Reliable Intelligent Environments 8 (2022) 79–103.

[17] Y. Liu, J. James, J. Kang, D. Niyato, S. Zhang, Privacy-preserving traffic flow prediction: A federated learning approach, IEEE Internet of Things Journal 7 (2020) 7751–7763.

[18] B. McMahan, et al., Communication-efficient learning of deep networks from decentralized data, in: Artificial intelligence and statistics, PMLR, 2017, pp. 1273–1282.

[19] H. Koziolek, N. Eskandani, Lightweight kubernetes distributions: a performance comparison of microk8s, k3s, k0s, and microshift, in: Proceedings of the 2023 ACM/SPEC International Conference on Performance Engineering, 2023, pp. 17–29.

[20] F. Hasan, et al., Iot based traffic management, accident detection, and accident prevention system

using machine learning method, in: Proceedings of the 2nd International Conference on Computing Advancements, 2022, pp. 249–253.
[21] L. Yu, B. Du, X. Hu, L. Sun, L. Han, W. Lv, Deep spatio-temporal graph convolutional network for traffic accident prediction, Neurocomputing 423 (2021) 135–147.
[22] Z. Zhou, et al., Spatio-temporal feature encoding for traffic accident detection in vanet environment, IEEE Transactions on Intelligent Transportation Systems 23 (2022) 19772–19781.
[23] K. G. Le, P. Liu, L.-T. Lin, Determining the road traffic accident hotspots using gis-based temporal-spatial statistical analytic techniques in hanoi, vietnam, Geo-spatial Information Science 23 (2020) 153–164.
[24] S. Sharma, V. Chang, U. S. Tim, J. Wong, S. Gadia, Cloud and iot-based emerging services systems, Cluster Computing 22 (2019) 71–91.
[25] S. A. Elsagheer Mohamed, K. A. AlShalfan, Intelligent traffic management system based on the internet of vehicles (iov), Journal of advanced transportation 2021 (2021) 1–23.
[26] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, J. Zhang, Edge intelligence: Paving the last mile of artificial intelligence with edge computing, Proceedings of the IEEE 107 (2019) 1738–1762.
[27] S. Wan, S. Ding, C. Chen, Edge computing enabled video segmentation for real-time traffic monitoring in internet of vehicles, Pattern Recognition 121 (2022) 108146.
[28] R. Ke, Z. Cui, Y. Chen, M. Zhu, H. Yang, Y. Wang, Edge computing for real-time near-crash detection for smart transportation applications, arXiv preprint arXiv:2008.00549 (2020).
[29] H. Nguyen, T. Nguyen, T. Leppänen, J. Partala, S. Pirttikangas, Situation awareness for autonomous vehicles using blockchain-based service cooperation, in: International Conference on Advanced Information Systems Engineering, Springer, 2022, pp. 501–516.
[30] X. Zhou, R. Ke, H. Yang, C. Liu, When intelligent transportation systems sensing meets edge computing: Vision and challenges, Applied Sciences 11 (2021) 9680.
[31] Y. Qi, M. S. Hossain, J. Nie, X. Li, Privacy-preserving blockchain-based federated learning for traffic flow prediction, Future Generation Computer Systems 117 (2021) 328–337.
[32] C. Xu, Y. Mao, An improved traffic congestion monitoring system based on federated learning, Information 11 (2020) 365.
[33] A. Chougule, et al., A novel framework for traffic congestion management at intersections using federated learning and vertical partitioning, IEEE Transactions on Consumer Electronics (2023).
[34] Y. Yuan, et al., Fedrd: Privacy-preserving adaptive federated learning framework for intelligent hazardous road damage detection and warning, Future Generation Computer Systems 125 (2021) 385–398. URL: https://www.sciencedirect.com/science/article/pii/S0167739X21002302. doi:https://doi.org/10.1016/j.future.2021.06.035.
[35] Y.-C. Wang, J. Xue, C. Wei, C.-C. J. Kuo, An overview on generative ai at scale with edge-cloud computing (2023).
[36] N. Rane, Role of chatgpt and similar generative artificial intelligence (ai) in construction industry, Available at SSRN 4598258 (2023).
[37] R. A. Bakir, S. A. M. Attia, Advancing urban health assessment through generative ai-driven indicators: Gcr case study (2023).
[38] D. Impedovo, V. Dentamaro, G. Pirlo, L. Sarcinella, Trafficwave: Generative deep learning architecture for vehicular traffic flow prediction, Applied Sciences 9 (2019) 5504.
[39] P. Lewis, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, Advances in Neural Information Processing Systems 33 (2020) 9459–9474.
[40] X. Dai, et al., Vistarag: Toward safe and trustworthy autonomous driving through retrieval-augmented generation, IEEE Transactions on Intelligent Vehicles (2024).
[41] W. Ding, Y. Cao, D. Zhao, C. Xiao, M. Pavone, Realgen: Retrieval augmented generation for controllable traffic scenarios, arXiv preprint arXiv:2312.13303 (2023).
[42] M. Mohanan, Competitive Analysis of Embedding Models in Retrieval-Augmented Generation for Indian Motor Vehicle Law Chat Bots, Ph.D. thesis, Dublin Business School, 2024.
[43] S. Moosavi, et al., Accident risk prediction based on heterogeneous sparse data: New dataset and insights, in: Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2019, pp. 33–42.
[44] H. Li, Y. Su, D. Cai, Y. Wang, L. Liu, A survey on retrieval-augmented text generation, arXiv preprint arXiv:2202.01110 (2022).
[45] D. Cai, Y. Wang, L. Liu, S. Shi, Recent advances in retrieval-augmented text generation, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 3417–3419.
[46] N. D. F. FIRE, E. M. R. HANDBOOK, Road traffic accident handbook (JUNE,2009).