Traffic density estimation with IoT edge computing in smart city: a case study

Hande Okutucu[1]
Şükrü Mustafa Kaya[*2]

Geliş tarihi / *Received:* 25.09.2025
Düzeltilerek geliş tarihi / *Received in revised form:* 13.11.2025
Kabul tarihi / *Accepted:* 18.11.2025
DOI: 10.17932/IAU.ABMYOD.2006.005/abmyod_v20i72006

Abstract
In the digitalizing world, smart city management and applications have become an integral part of our lives in recent years. With the increase in innovative sensor-based devices, concepts such as smart environment, energy, transportation, healthcare, and traffic have emerged within smart cities, improving the quality of life for citizens through smart city management. This study focuses on the concept of smart transportation and traffic management, which is a subcategory of smart cities. The concept of traffic management in smart cities has advanced thanks to the integration of IoT (Internet of Things) and Edge Computing technologies. This system provides more realistic traffic density predictions. IoT devices, traffic sensors, cameras, and GPS-enabled devices collect real-time data such as traffic density, vehicle speeds, and road conditions in smart cities. In our study, we aim to predict hourly vehicle density for a specific day. A three-year dataset was utilized, consisting of 24-hour vehicle density data for each day of the year. Using time series algorithms, hourly vehicle density predictions were made for a future date. Algorithms such as ANN (Artificial Neural Network), KNN (K-Nearest Neighbors), LSTM (Long Short-Term Memory), Random Forest, Prophet, and XGBoost (Extreme Gradient Boosting) were employed for predictions. The error rates of the algorithms were analyzed to identify the most accurate prediction method. The vehicle density prediction data produced by this algorithm was considered the closest to reality. The results were discussed and evaluated in the final section of the article.

[1] *İstanbul Aydin Üniversitesi, handeokutucu@stu.aydin.edu.tr, ORCID: 0009-0006-5381-9804*
[2] *İstanbul Aydin Üniversitesi, mustafakaya@aydin.edu.tr, ORCID: 0000-0003-2710-0063*

Keywords: *Internet of Things; Edge Computing; Smart City; Traffic Density; Machine Learning*

Akıllı şehirde IoT uç bilişim ile trafik yoğunluğu tahmini: bir vaka çalışması

Özet

Dijitalleşen dünyada akıllı şehir yönetimi ve uygulamaları son yıllarda birçok alanda hayatımıza girmiştir. Üretilen inovatif sensör tabanlı cihazların artmasıyla birlikte akıllı şehirlerde akıllı çevre, enerji, ulaşım, sağlık, trafik gibi kavramlar ortaya çıkmış ve akıllı şehir yönetimi ile vatandaşların hayat kalitesi artmaya başlamıştır. Çalışmada akıllı şehirlerin alt başlığı olan akıllı ulaşım ve trafik yönetimi kavramına odaklanılmaktadır. Akıllı şehirlerde trafik yönetimi kavramı IoT (Nesnelerin İnterneti) ve Edge Computing (Uç Bilişim) teknolojisinin entegrasyonu sayesinde gelişme kaydetmektedir bu sistemle trafik yoğunluğu tahmini daha gerçekçi bir sonuç ortaya koymaktadır. IoT cihazları, trafik sensörleri, kameralar ve GPS destekli cihazlar aracılığıyla akıllı şehirlerde trafik yoğunluğu, araç hızları ve yol koşulları gibi eş zamanlı veriler toplamaktadır. Çalışmamızda belirli bir günde, bir saatlik araç yoğunluğu tahminlemesi hedeflenmiştir. 3 yıllık veri setinden yararlanılmış ve yılın her günü 24 saatlik araç yoğunluğu verileri kullanılmış olup, zaman serisi algoritmaları kullanılarak ileri bir tarih için saatlik araç yoğunluğu tahmini yapılmıştır. Zaman serisi analizlerinden ANN (Artificial Neural Network), KNN (K-Nearest Neighbors), LSTM (Long Short-Term Memory), Random Forest, Prophet, XGBoost (Extreme Gradient Boosting) gibi algoritmalar kullanılarak tahminleme yapılmıştır. Algoritmaların hata oranlarına incelenip en doğru tahmin hangi algoritma ile bulunduğu ortaya koyulmuş ve bu algoritmayla yapılan araç yoğunluğu tahmin verisi gerçeğe en yakın kabul edilmiştir. Çıkan sonuçlar makalenin son kısmında tartışılmış ve değerlendirilmiştir.

Anahtar Kelimeler: *Nesnelerin İnterneti, Uç Bilişim, Akıllı Şehirler, Trafik Yoğunluğu, Makine Öğrenmesi*

Introduction

Smart cities are modern urbanism that aims to make city life more efficient, sustainable and livable by using information and communication technologies. This concept aims to optimize the infrastructure and services of cities with the help of technology and to provide a better quality of life for city residents.

The concept of smart cities is defined by Zhao et al. as the application of urban modernization and the integration of information technologies in the process to promote the sustainability of cities and improve the well-being of citizens (Zhao, Su, Li, Zhu and Zhang, 2025).Smart cities aim to solve challenges through the integration of technology and innovative developments. Building a smart city means improving urban life, increasing resource efficiency and creating sustainable environments through the integration of advanced technologies and data-driven solutions (Karri, Machado,Tavares and Jain, 2024). Abu Rayash and Dinçer stated in their study that the transformation of homes and cities into smart ones with the development of technology has made the lives of citizens living in the city easier. They emphasized that smart transportation systems, which are one of the basic building blocks of smart cities, are of critical importance in increasing the quality of life of people living in big cities with traffic density management. The success of these systems largely depends on the accurate and instant collection and analysis of traffic density data. They state that in order for smart cities to be effective and sustainable, the collected data needs to be managed, sustainable solutions provided, and simultaneous data needs to be transferred to the system, and control and analysis needs to be provided (Abu-Rayash and Dinçer, 2025). Due to the increasing population due to migration in recent years, traffic density has increased significantly, which has a very negative impact on the daily lives of individuals. Many technologies and strategies continue to be developed to reduce traffic density problems and make transportation more efficient by making cities smart. By integrating traffic lights, sensors and cameras with real-time traffic data, vehicle flow is made more sustainable. Thanks to real-time data collection, information from mobile devices, GPS and traffic cameras is analyzed and alternative route suggestions are offered to drivers where there is no traffic congestion (Kheder and Mohammed, 2024).In public transportation systems, smart transportation vehicles are optimized and made more efficient, and it is planned to ease traffic by reducing the use of private vehicles. In addition, autonomous vehicles and

vehicle-vehicle and vehicle-infrastructure communication technologies contribute to safer and more relaxed traffic flow (Dahooie, Mohammadian, Qorbani and Daim, 2024). The introduction of artificial intelligence into our lives makes it possible to analyze past traffic data, predict future densities, and develop solutions for this situation. In addition, within the scope of sustainable transportation planning, bicycle paths and pedestrian paths are encouraged, and shared vehicle service applications and micro-mobility solutions are encouraged. Thanks to these systems, it not only relieves traffic congestion but also contributes to nature by reducing carbon emissions. In order for smart cities to work effectively, it is of great importance that both technological investments increase and users perceive how much these systems contribute to their lives (Idrissi, Lachgar and Hrimech, 2024). Urban mobility, which is one of the main components of smart cities, causes the problem of finding a parking space in parking lots and traffic congestion with the increase in the number of vehicles. Tekouabou et al. stated that vehicle drivers looking for parking spaces add 30% extra density to traffic density. In their study, they propose a system that also integrates the prediction model to solve this problem. The system aims to reduce traffic congestion by predicting the available parking spaces in parking lots for citizens in advance. In their tests on the Birmingham dataset, the Mean Absolute Error (MAE) was measured as 0.06% with the Bagging Regression (BR) algorithm. In addition, the complexity of the system has been significantly reduced (Tekouabou, Alaoui, Cherif and Silkan, 2022).

Traffic intersection signal optimization is one of the most important applications to be done to control traffic density in smart cities. With this system, reducing traffic density is of critical importance. In this direction, a smart city traffic intersection signal control optimization method has been examined with the adaptive artificial whale swarm algorithm. In order to understand the real situation of traffic flow, the traffic flow status system at the intersections has been designed. Signal control parameters, minimum average delay and number of stops have been designed as target functions and an optimization model has been created for signal control (Zhang, Jin, Chang and Huang, 2024). By combining with chaotic search theory, improving the artificial whale swarm algorithm, the smart city traffic intersection signal control optimization model was solved based on the adaptive artificial whale swarm algorithm and the smart intersection signal control optimization was realized. According to the study, the average

delay obtained with this method was found to be 7.8 ms, the number of stops was 2, and the travel time was 68.4 seconds. Thus, they proved that with the proposed method, they achieved good results for traffic signaling at smart intersections and thus could reduce traffic congestion (Wei and Ju, 2024). Every passing day, a new one is added to smart city management. In this case, it makes the lives of city residents easier and offers a comfortable life. Thanks to IoT and sensor-based tools, a new device is integrated into our lives every passing day. Researchers are also looking for how to make these devices more efficient and integrated into life (Doğaroğlu and Çalışkanelli, 2022).

Smart city management

Rapid population growth, unsafe lifestyle, waste of natural resources, indiscipline in human behaviours, urgent needs in the field of medicine, patient information security, and issues related to agriculture, and automation requirements in industries are the main reasons for inventing technologies. Smart cities aim to address these challenges through the integration of technology, data, and innovative applications. Building a smart city requires integrating advanced technologies and data-driven solutions to improve city life, increase resource efficiency, and create sustainable environments (Karri, Machado, Tavares, Jain, Dannana, Gottapu and Gandomi, 2024). In recent years, significant progress has been observed in the scope of smart city applications in developing countries, and municipalities are fulfilling their responsibilities in many areas, especially in transportation, energy, water and waste management (Figure 1). Within the scope of smart city management; smart transportation and traffic applications such as traffic density maps, smart parking applications, automatic passage systems, and electronic inspection systems, vehicle tracking systems, smart meter readings, and chip-equipped garbage containers are available (Khemakhem and Krichen, 2024). For example; within the scope of smart environment, solid waste facilities that produce electricity from methane gas, air quality monitoring systems, noise control and warning systems, excavation vehicle tracking systems, smart water network applications, solar lighting of stops, electric buses and solar-powered meteorological stations are also used in environmental smart management systems (Özcan, 2023). One of the most important problems in smart city management is waste management. The rapid increase in population in big cities has made this problem even more important. Waste management, thanks to smart sensors and IoT devices, ensures that the trash can is emptied when necessary according to waste

levels. This reduces the frequency of waste collection, creates unnecessary traffic congestion, and thus minimizes operational costs (Ahmed, Dubey, Kumar and Dubey, 2024). Technological advances have paved the way for the Internet of Things (IoT) concept to become widespread worldwide. Applications of IoT for smart cities are particularly effective in environmental and urban management. In smart cities, IoT devices are used to connect supply and demand information for smart homes and energy management in smart grids. However, power scheduling and information security transmission stand out as important problems in complex smart systems, but these problems can be solved with various machine learning techniques and data analytics methods (Awan, Khan, Rahmani, Tahir, Alam, Alturki and Ullah, 2020) .

IoT and edge computing

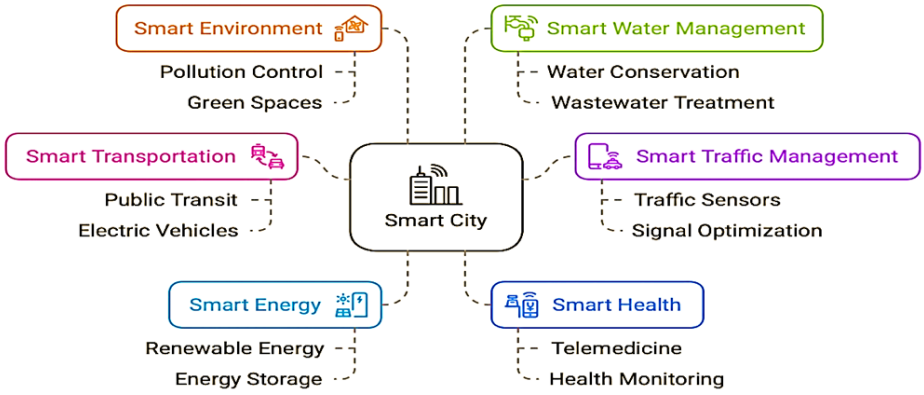

Figure 1: Smart city component

This structure contributes to the management of cities in a more efficient, sustainable way and in a way that increases the quality of life of citizens (Yalli, Hasan, Jung and. Al-Selwi, 2025). In recent years, with the widespread use of Internet of Things (IoT) devices in many areas, data production has also become very high. After this data is collected, it is transferred to the cloud computing system for analysis. However, due to the remoteness of the cloud nodes in the system, problems such as network congestion, data transfer delay and security arise. These problems are solved by processing at a point closer to the data sources thanks to edge computing, which offers alternative solutions. Edge computing is superior to IoT devices in terms of processing power, but it is more limited than cloud systems because tasks that require high processing power still

need to be directed to the cloud. At this point, the Autoencoder algorithm reduces the delay in data traffic. The Autoencoder algorithm is an artificial neural network algorithm that is one of the unsupervised learning methods and is generally used to reduce data size, make meaningful inferences from data or detect anomalies. Autoencoders try to reproduce the input (Azzalini, Flammini, Emanuele, Guadagno, Ragaini and Amigoni, 2025). It optimizes data transfer by compressing and encoding data with artificial neural network technique. In this way, it reduces the data flow between edge and cloud computing and reduces network congestion and latency problems (Tekin, Kakız and Çoban, 2021).

The Internet of Things (IoT) Architecture consists of four layers as seen in Figure 2 The first of these is the interface layer, the others are the service layer, the network layer and the perception layer, respectively. In the architecture (Figure 4), it is recommended to integrate the edge computing system between the network layer and the perception layer. In this section, it is recommended that the data be passed through the filter, processed and sent to the last layer before being sent to the perception layer.

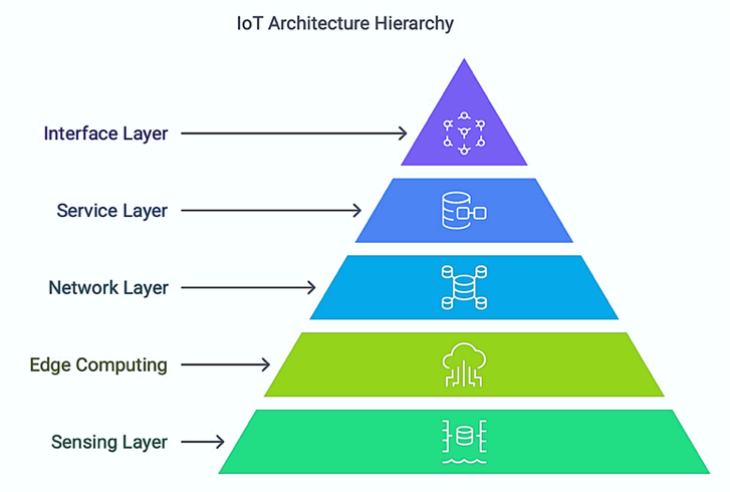

Figure 2: IoT basic architecture

The Interface Layer provides the integration between IoT devices and the application layer, allowing devices to communicate with each other and users. This layer manages device connections, monitors their status, and is responsible for security. It provides data transmission between devices through different protocols and APIs, remotely controls and monitors them. It manages data efficiently by ensuring compatibility between devices, but

it can also face challenges such as security and protocol diversity (Donta, Srirama, Amgoth and Annavarapu, 2022).

The Service Layer is a layer that provides functional services in the IoT system. This layer analyzes the collected data and derives meaningful results from this data. The service layer provides the infrastructure that various services such as data management, data analysis, data processing and decision making need to be performed. The layer usually works integrated with cloud computing or edge computing technologies and processes large data sets and transmits these results to users or applications (Arachchige, Murtaza, Cheng , Albahlal and Cheng-Chi, 2024).

The Network Layer is the layer that provides data transmission between IoT devices. This layer plays a role in the secure transmission of data by using network communication protocols that enable data exchange between sensors and devices. It allows devices to communicate with each other or with a central data processing unit (for example, cloud or edge computing) and generally uses wireless communication technologies such as Wi-Fi, ZigBee, 5G. The reliability of this layer is critical for the proper operation of IoT systems because devices cannot function effectively without data transmission and network connectivity (Alkhaldi, Darem and Alhashmi, 2024).

Edge Computing is the processing and analysis of data at a point close to the source, i.e. at the edge where IoT devices and sensors are located, before reaching the center. This approach allows data to be processed on local devices or networks before being sent to a central cloud server, thus reducing delays that may occur during data transmission and optimizing network bandwidth usage. Edge computing is important for real-time applications and IoT scenarios where latency should not be a problem. Moreover, processing data locally also offers advantages in terms of security and privacy, because sensitive data is analysed and stored locally without being carried on the network (Rajagopal and Buyya, 2025).

The Sensing Layer is the lowest layer in the IoT architecture and completes the task of collecting data from the physical environment. This layer includes devices such as sensors and actuators, detects environmental parameters (e.g. temperature, light, motion, humidity, pressure), and converts them into digital data. This layer is the foundation of IoT systems because other layers only act based on the data obtained from this layer.

The sensing layer is critical to the accuracy of sensors and data quality (Dui, Li , Dong and Wu, 2025).

Edge computing

Edge computing is a computing model that allows data to be processed at a point close to the source of the data before being sent to the center. In this method, data transmission delay is reduced by directly processing the data collected from the devices, bandwidth usage is optimized and faster response times are provided in real-time applications. Local processing of data both facilitates the protection of privacy and reduces the load on central servers, creating a more efficient computing infrastructure (Tran-Dang and Kim, 2025).

The Edge Computing architecture in Figure 3 reduces the dependency on central data centers by processing this data at a point close to the data source. In this way, data transmission delay is reduced, the load on the network is reduced, and real-time decision-making processes are accelerated. Especially in applications that require instant intervention such as traffic management, energy optimization, and emergency alerts, the use of Edge Computing creates a more efficient and secure smart city structure (Rahmani, Tanveer, Gharehchopogh, Rajabi and Hosseinzadeh, 2025).

With the integration of IoT and Edge Computing technologies, the closest estimate of traffic data to reality is provided. Before the data is sent to the point where it will be stored in the system, it means that the data is processed locally with edge computing systems close to the source, making it more real, complete, noiseless and easy to analyze. In addition, this local processing reduces data transmission latency and allows for real-time estimates with instant traffic density analysis. With these estimates, traffic density can be reduced or alternative routes can be suggested by optimizing traffic signals and controlling traffic flow. Real-time traffic density estimates made with this technology provide cities with a more sensitive and compatible transportation system, making cities smarter, more efficient and sustainable (Katsigiannis and Mykoniatis, 2024).

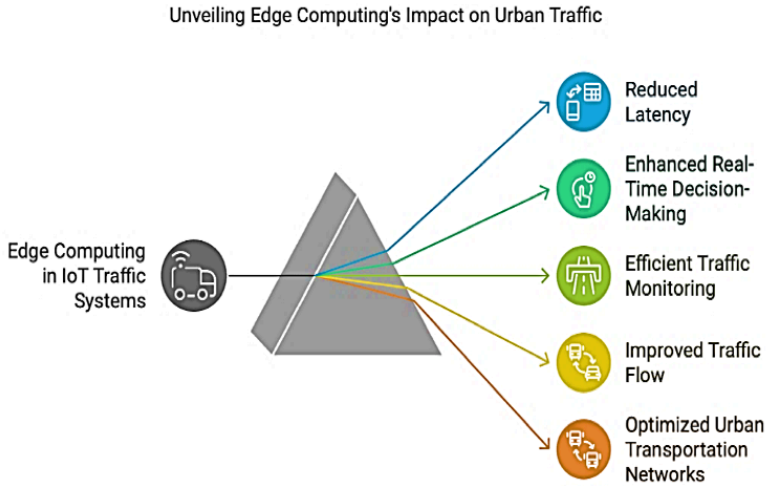

Figure 3: Edge computing

In their article, Kaya et al. define Edge Computing as a mechanism developed to overcome the limitations of cloud-based data processing in traditional IoT systems. The sensing layer is insufficient in traditional systems for high-performance data processing, and the physical distance of cloud technology creates a disadvantage in simultaneous data processing. With edge computing, data from sensors is processed at the closest point to the source, reducing latency and completing data processing with high performance. This system works in harmony with IoT layers and offers real-time solutions (Kaya, İşler, Mahfouz, Rasheed and AlShammari, 2023). Moura et al. mentioned that smart transportation systems in smart cities sometimes encounter communication delay problems in the real world, and therefore cybersecurity vulnerabilities related to real-time data processing and centralized data storage occur. In his study, he proposes an innovative architecture using Edge Computing and Distributed Ledger Technology (DLT) to solve these problems. Edge Computing is moving cloud services to units closer to data sources, such as moving edge computing to the roadside if vehicle data is stored. Thanks to this approach, delays and network congestion are reduced, and the DLT architecture ensures that this data is stored on a secure platform. The proposed architecture is to create a decentralized platform for data management and to ensure that data from sources in the transportation network is processed, stored and shared securely (Moura , Aquino and Loureiro, 2024).

Materials and methods

The aim of this study is to make predictions using time series algorithms for vehicle density prediction in smart traffic management and to examine the error rates of these algorithms and to determine which algorithms give the closest results to the truth. In the study, a data set consisting of four attributes such as date, time, number of data, and vehicle density is classified. Although the data obtained from Rtms (Remote Traffic Microwave Sensor) devices is processed, the received data stream is sent directly to the data storage center. As far as it is determined, there is no edge computing system in this type of system. In case there is a preprocessing layer that will undertake the edge computing task, the data will be more accurate, reliable and will contribute to the formation of more meaningful analyses in real time.

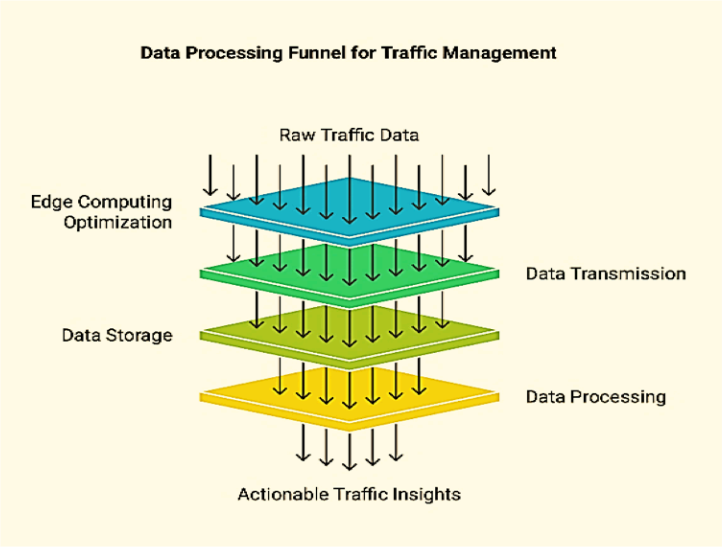

Figure 4: IoT layers and edge computing

Case study

In the study, a location with high traffic (Istanbul, July 15 Martyrs Bridge) was selected for vehicle density data and the dataset was obtained from data provided by the Istanbul Metropolitan Municipality through Rtms sensors at the bridge entrance was used. Vehicle density estimation will be made for a future date using the vehicle density (number) data of the years 2022, 2023 and 2024 (3-year data) at every hour of the day. There are 4 attributes in the study such as day, hour, number of data and number

of vehicles. According to the algorithms, some of the data was used as a training data set and some as a test data set in an optimum way.

Rtms (Remote traffic microwave sensor)

This device is a radar-based sensor system used to monitor and analyze traffic flow on the road in real time. It uses microwave radar signals to detect the number, class, speed and density of vehicles. It collects real-time data from multi-lane roads and transmits this data to management centers, allowing traffic flow to be monitored, signaling systems to be optimized and traffic volume to be reduced (Zhao, Tang, Gao and Zhu, 2018). It is widely used in smart city projects due to its features such as high accuracy and low maintenance cost. Rtms (Remote Traffic Microwave Sensor) is a radar-based sensing technology used in traffic monitoring and management (Figure 5). These sensors use a technology that emits microwaves to monitor the movement of vehicles and the flow of traffic. This technology (Figure 6) is effectively used for speed measurement, vehicle counting and traffic density.

Figure 5: Rtms sensor

Since system measurement is important, the angles must be adjusted correctly, therefore, situations such as sensor angle changes and obstructive foreign objects that cause sensor data to be inaccurate are tried to be prevented through periodic maintenance works carried out by the Traffic Branch Directorate (Yıldırım and Çataltepe, 2007).

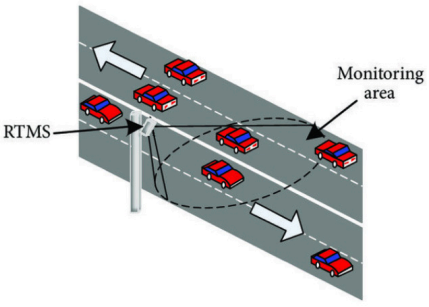

Figure 6: Rtms application area

The estimation models to be selected according to the structure of the data set are extremely important. For example, parametric models (such as Linear Regression) may be suitable for data with a normal distribution, while more flexible models (such as Random Forest) should be preferred for data with an unknown distribution (İlyas and Albayrak, 2023). Since time series analysis will be performed in this study, similar studies in the literature have been examined and the most commonly used algorithms in traffic density estimations have been used. With these algorithms, hourly vehicle count estimates will be made on a future date and the error rate of the algorithms will be compared with the accuracy. The obtained outputs are evaluated in the conclusion section of the study.

Dataset

The data set used in the study consists of 4 columns: day, hour, number of data, and number of vehicles. The data set consists of approximately 25,000 rows, 100,000 data, and is 190 kb in size. Although the training and test data sets differ according to the algorithms in the data, approximately 80% training and 20% test data were used in general for the algorithms. The raw data received was read using the Pandas Library in Python and transferred to the data frame. The "Number of data" column from the attributes in the data set expresses the number of hourly vehicles passing every 2 minutes. In other words, the number of hourly data is expected to be 30. However, it was observed that some of the data in the data set was missing. This number of missing data needs to be completed to 30. Because the missing number of data means that the "Number of Vehicles" data is also missing. The missing data was completed using the Pandas Library in Python. In addition, since the different scales of the attributes in the data set will negatively affect the performance of some algorithms, min-max normalization was applied to the data. And results closer to reality were obtained.

Simulation model

- In order to say that the number of data in the dataset is complete, the data count column must be 30, but some data counts are less than 30, which means that the vehicle count data is missing, so the data was filtered and the data count in Python was completed to 30, thus the vehicle count data column was completed.
- Real-time incoming data is received and processed simultaneously in the data flow shown in Figure 5. The Information Processing Directorate,

which uses this data flow as a source, stores the data sets and then the data is sent to the Traffic Branch Directorate for processing.

- There is no roadside edge computing system for traffic density data to determine whether the data generated from rtms sensors is abnormal or normal (Figure 7).If found, incomplete, erroneous, meaningless data will not be stored unnecessarily, and it is anticipated that it will not create additional storage costs allocated to the municipality budget.
- In the study, sensor data were generally used for training (optimally) at 70% and the remaining 30% for testing and prediction according to the algorithms.
- As can be expected, vehicle density in traffic is observed especially during work commute hours between 06:00 and 08:00 in the morning and between 18:00 and 19:00 in the evening.
- The data was first studied through algorithms without normalization, but results that were far from reality emerged. Later, since the data needed to be scaled within a certain range (usually between 0 and 1), min-max normalization was applied to the data set. And more successful results were obtained.

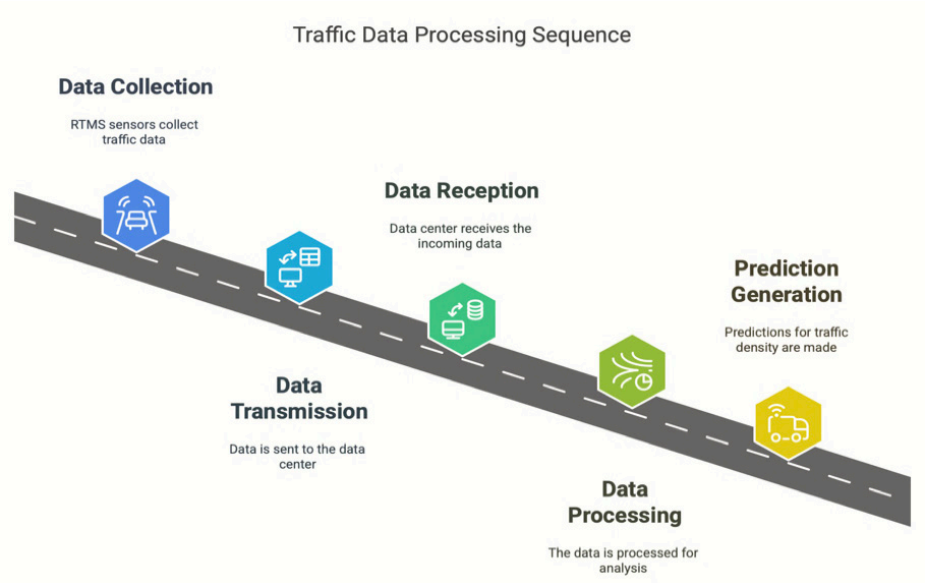

Figure 7: Simulation model

Experimental results

Classification reports

Classification reports are used to evaluate the classification performance of machine learning models and reveal metrics such as Precision, Recall, F1-Score and Support for each classification. This analysis is used to examine the accuracy estimates of the model and to determine the strengths and weaknesses of the algorithms. It provides a balanced evaluation of metrics, especially in imbalanced data sets, and is used to compare the performance of algorithms (Collado-Montañez, Martín-Valdivia and Martínez-Cámara, 2025).

Table 1: Ann classification reports results

Class	Ann classification reports			
	Precision	Recall	F1-score	Support
Accuracy	-	-	1.00	5001
Macro avg	1.00	1.00	1.00	5001
Weighted avg	1.00	1.00	1.00	5001

Since The Table 1 Ann classification report, precision, recall and F1-score: 1.00 are obtained, we can say that the model has made all its classifications correctly. The overall accuracy is calculated as 100%. Also, since the support value is 5001, the number of data can be said to be sufficient.

Table 2: Knn classification reports results

Class	Knn classification reports			
	Precision	Recall	F1-score	Support
0	0.51	1.00	0.68	2534
Accuracy	-	-	0.52	5001
Macro avg	0.75	0.51	0.36	5001
Weighted avg	0.74	0.52	0.37	5001

The Table 2 Knn classification report, precision 0.51, recall 1.00, the model caught all positive examples (high Recall), but some of its predictions were wrong (low Precision). F1-score 0.68 reflects the imbalance between Precision and Recall. Overall accuracy is 52%, meaning that more than half of the model's predictions were correct. Low values in Macro and Weighted Average metrics reveal the difference in performance and imbalance between classes. Parameter optimization may be required to improve the model's performance.

Table 3: Lstm classification reports results

Class	Lstm classification reports			
	Precision	Recall	F1-score	Support
0	1.00	1.00	1.00	25002
Accuracy	-	-	1.00	25002
Macro avg	1.00	1.00	1.00	25002
Weighted avg	1.00	1.00	1.00	25002

The Table 3 Lstm classification report shows that the model has excellent performance (Precision, Recall, F1-score: 1.00) and accuracy is 100%. The model seems to have made error-free predictions for all classes. If the realism of the performance is to be checked, the model should be tested again with new and different data sets.

Table 4: Prophet classification reports results

Class	Lstm classification reports			
	Precision	Recall	F1-score	Support
High	1.00	1.00	1.00	2
Accuracy	-	-	1.00	2
Macro avg	1.00	1.00	1.00	2
Weighted avg	1.00	1.00	1.00	2

Although In the Table 4 Prophet Classification Report states that the model has excellent classification performance (Precision, Recall, F1-score: 1.00) and the accuracy is 100%, the fact that the Support value is only 2 weakens the reliability of this assessment. Since the class examples in the data are very few, these results cannot be generalized and it is difficult to make a sound inference about the real performance of the model. It may need to be retested with a larger and more balanced dataset.

Table 5: Random forest classification reports results

Class	Xboost classification reports			
	Precision	Recall	F1-score	Support
Low	0.00	0.00	0.00	2
Middle	0.00	0.00	0.00	2
High	1.00	1.00	1.00	4163
Accuracy	-	-	1.00	4167
Macro avg	0.33	0.33	0.33	4167
Weighted avg	1.00	1.00	1.00	4167

The Table 5, Random Forest Classification Report shows that the performance of the model is quite unbalanced. While it has excellent results for the high class (Precision, Recall, F1-score: 1.00), all metrics for the low and medium classes are zero (Precision, Recall, F1-score: 0.00) and the number of class examples is very low (2). This shows that the model only correctly predicts the high class and ignores the other classes. As a result, although the overall accuracy is 100%, the classification ability of the model is unbalanced and it only correctly predicts a single class. In this case, the model may have overfitted due to data imbalance and needs improvement.

Table 6: X boost classification reports results

Class	Xboost classification reports			
	Precision	Recall	F1-score	Support
Low	0.00	0.00	0.00	2
Middle	0.00	0.00	0.00	2
High	1.00	1.00	1.00	4997
Accuracy	-	-	1.00	5001
Macro avg	0.33	0.33	0.33	5001
Weighted avg	1.00	1.00	1.00	5001

XGBoost Classification Report Table6, shows that the model predicts the high class perfectly (Precision, Recall, F1-score: 1.00), but all metrics for the low and medium classes are zero (Precision, Recall, F1-score: 0.00) and the number of examples for both classes is very low (2). This result shows that the model only focuses on the high class and cannot correctly classify the other low and medium classes. Although high accuracy (accuracy: 100%) is achieved, there is a situation where the classes produce imbalanced results and the model only learns the dominant class and ignores the others. Improvements may be needed to make this model perform more successfully and balancedly, and it is necessary to review the class imbalance.

Performance values

The studied algorithm performance results are presented below in the Table 7.

Table 7: Comparison of performance values

Algoritma	Mae	Mse	Rmse	R2	Cpu(sn)
Ann	0.5291	0.3582	0.5983	-3.6235	20.4647
Knn	0.2453	0.0771	0.2771	0.0032	30.9216
Lstm	0.2399	0.0747	0.2733	0.0287	16.5026
Random forest	0.0974	0.0202	0.1424	0.6023	621.983
Prophet	0.0604	0.0036	0.0604	-350	6.619
Xboost	0.0753	0.0093	0.0967	0.8147	0.4528

According to table 7, XGBoost gave the best results with the lowest error metrics (MAE: 0.0753, MSE: 0.0093, RMSE: 0.0967) and the highest R^2 value (0.8147), while it also worked quite efficiently in terms of CPU time (0.4528 seconds). Prophet drew attention with the lowest MAE (0.0604), while the R^2 value (-350) showed that the model failed to understand the data. While LSTM and KNN gave reasonable results, the ANN model exhibited low performance with high error metrics and negative R^2. Although Random Forest provided a good R^2 (0.6023), it was inefficient in terms of time with a CPU time of 621.983 seconds. In general, XGBoost stands out as the best algorithm in terms of error and performance balance. In addition, the Prophet model is a regression model, meaning it estimates continuous (numerical) values, so evaluation metrics such as confusion matrix and classification report may not be directly valid.

Confusion matrix

Confusion matrix is a tool used to measure and evaluate the performance of the classification model. It reveals the correct or incorrect predictions of the model. In the table values, it shows the correct and incorrect predictions made by the model for positive and negative classes with values such as True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). This matrix helps us understand in which classes the model is strong or weak by using the calculation of metrics such as Precision, Recall, and Accuracy (Yu and Do, 2024).

The Figure 8 also visualizes the performance of a classification model belonging to the Ann algorithm. According to the visual, all predictions were made correctly for the "Low" class (5001 correct predictions), but no predictions were made for the high class. This indicates that the model focused on only one class and did not consider the other classes. There

may be a class imbalance in the data or the model did not learn the classes well. Performance analysis should be done with a more balanced data set.

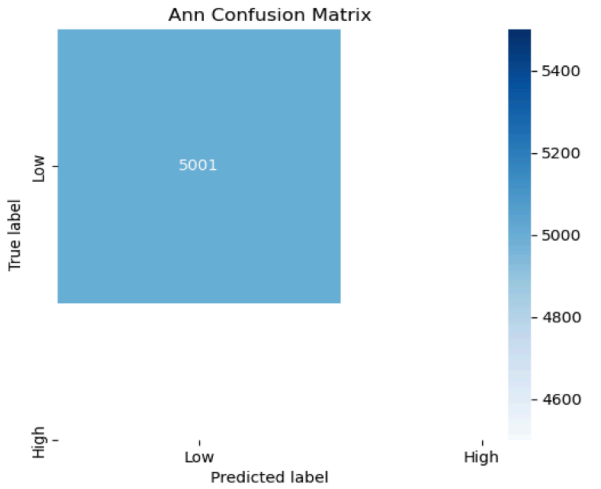

Figure 8: Ann confusion matrix

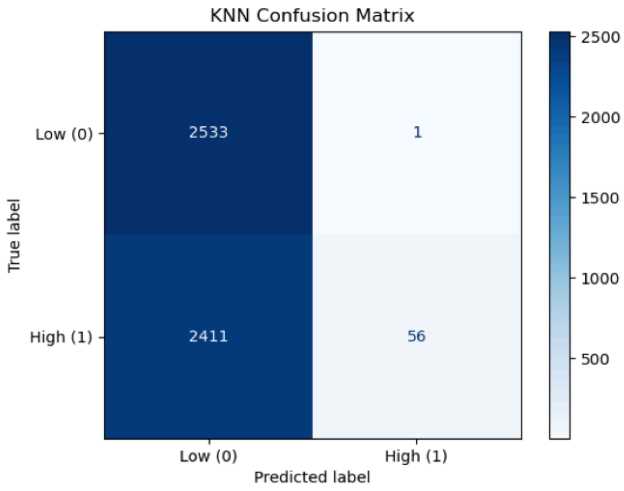

Figure 9: Knn confusion matrix

The Figure 10, Lstm algorithm shows the relationship between the model's actual number of vehicles and the estimated number of vehicles on the time axis. The blue line represents the actual values, and the orange line represents the model's estimated values. The graph shows that the model generally makes estimates close to the actual data, but there are deviations at some time intervals. This shows that the model's predictive performance

is generally consistent over time, but some points may need improvement. Additional features or different optimization methods can be tried to make the model more accurate.

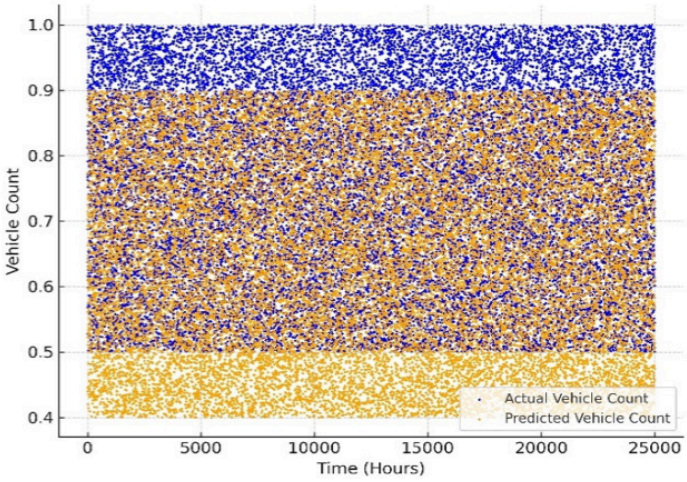

Figure 10: Lstm by time

In the Figure 11, Random Forest performance model, it made 2819 correct predictions and 384 incorrect predictions in the "High (1)" class, while it made 836 correct predictions and 128 incorrect predictions in the "Low (0)" class. In general, the model predicted the "High (1)" class more successfully, but there is room for improvement in false negatives (384).

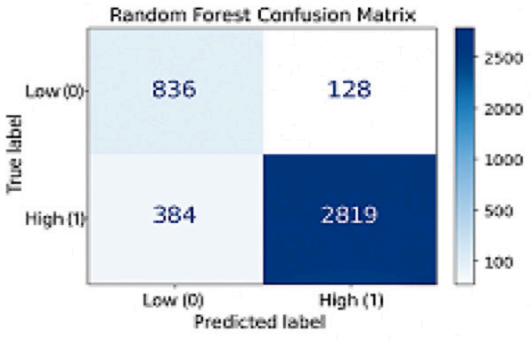

Figure 11: Random forest confusion matrix

The Figure 12, XGBoost model shows that it only correctly predicts the "Low" class (5001) and does not predict the "High" class at all. This indicates that the model may have focused on only one class due to an imbalanced dataset or an incorrect training process. Data imbalance issues

should be addressed and appropriate weighting or retraining methods should be used to improve model performance.

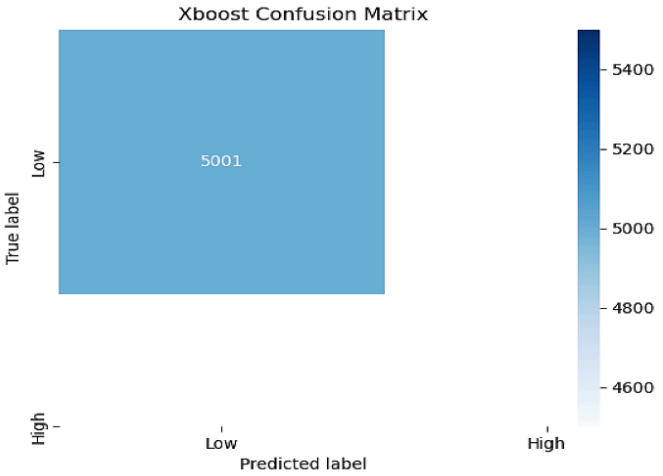

Figure 12: Ann confusion matrix

Discussion and conclusion

This study was prepared using machine learning methods to predict traffic density, analyze trends, and contribute to traffic management**.** Vehicle density in traffic may cease to be linear or predictable due to various factors (such as weather conditions, whether the academic term is in session or not, significant match days, and special occasions like holidays and festivals). In this study, the number of vehicles was predicted using the ANN, KNN, LSTM, Prophet, XGBoost, and Random Forest algorithms, aiming to determine vehicle density at specific days and times. In this way, it becomes possible to prevent traffic congestion, optimize signalization systems, and guide drivers to alternative routes.

In this context, the algorithms that produced the most successful results from the predicted values were identified. The Random Forest algorithm requires a long processing time, which is thought to be due to the model's complexity and its parameters. The ANN model, on the other hand, shows the weakest performance with high error rates and long training time. Among the best-performing models in terms of performance, the Prophet algorithm emerges (with the lowest MAE, MSE, RMSE); in short, if the model with the lowest error rate is preferred, Prophet can be considered. However, the data set should be reviewed with respect to the R^2 value. According to the analysis results, the best-performing models are Prophet

and XGBoost. As a result, if the speed and accuracy of the model are critical, XGBoost can be preferred. In terms of performance, XGBoost has been the best and fastest model (with high R^2 and low MSE).In Utku's study, which utilized Istanbul traffic data, a comparative analysis involving LR, RF, SVM, MLP, CNN, and RNN demonstrated that the LSTM-based model achieved superior performance. The model was reported to attain an R^2 value of approximately 0.9, indicating a high level of predictive accuracy. In contrast, our study evaluated ANN, KNN, LSTM, Prophet, XGBoost, and Random Forest algorithms, with the findings highlighting Prophet as the most effective in terms of low error rates, while XGBoost emerged as the most suitable in terms of both speed and accuracy. The divergence in results between these two studies can be attributed to differences in dataset characteristics, model parameters, and the target variables considered (vehicle count, speed, or density). Consequently, the prominence of tree-based methods and the time series–oriented Prophet model in one study, and the superior performance of the deep learning–based LSTM in the other, is consistent with expectations in the literature (Utku, 2023).

In İlyas and Albayrak's study conducted in Antalya, hourly vehicle count data from each approach lane of two selected smart intersections were utilized, and the data were divided into training and test sets to develop models based on Linear Regression, Polynomial Regression, Support Vector Regression (SVR), and Random Forest Regression. The performance of the models was compared using Mean Absolute Error (MAE) and R^2, and the Random Forest model was found to outperform the other models. In contrast, our study evaluated ANN, KNN, LSTM, Prophet, XGBoost, and Random Forest algorithms, with Prophet exhibiting the lowest error rates and XGBoost emerging as the most suitable model in terms of both speed and accuracy. These differing results can be attributed to the variety of algorithms employed, characteristics of the datasets, and the target variables predicted (vehicle count, speed, or density). Consequently, while the Antalya study is limited to classical regression methods, our study provides a more comprehensive analysis by integrating both time series and advanced machine learning techniques (İlyas and Albayrak, 2023).

In their study, Balcıoğlu and Sezen investigated the performance of different machine learning algorithms for traffic flow prediction on the E80 and E84 highways. The algorithms employed included LSTM, KNN-MLP, and SVR, with their performance evaluated using RMSE and accuracy

(Acc-%). The results revealed that the KNN-MLP model achieved the lowest RMSE and highest accuracy on both highways, outperforming the other approaches; LSTM ranked second, while SVR showed the weakest performance. The proposed KNN-MLP method enhances prediction by selecting spatiotemporally correlated routes through KNN and capturing hidden features and long-term dependencies with MLP. Tested on real traffic data provided by the Istanbul Metropolitan Municipality, the model demonstrated superior predictive capvirgülability compared to LSTM and SVR. Overall, the findings highlight the effectiveness of the KNN-MLP algorithm for traffic flow prediction and suggest that incorporating external factors such as weather conditions and special events could further improve its accuracy and reliability. More broadly, the study underscores the advantage of machine learning approaches over traditional methods in modeling the complex dynamics of traffic flow (Balcıoğlu and Sezen, 2024).

In conclusion, selecting the appropriate model according to the data type increases prediction accuracy. For time series data, developing specific models (ARIMA, LSTM) and investigating their predictive performance on larger and more complex datasets enable more accurate forecasting.

Funding: This research received no funding.

Conflicts of Interest: Author declare no competing interests.

References

[1] Abu-Rayash, A., & Dincer, I., (2025). Development of an integrated model for environmentally and economically sustainable and smart cities. *Sustainable Energy Technologies and Assessments, 73,* 104096. https://doi.org/10.1016/j.seta.2024.104096.

[2] Ahmed, K., Dubey, M. K., Kumar, A., & Dubey, S., (2024). Artificial intelligence and IoT driven system architecture for municipality waste management in smart cities: A review. *Measurement: Sensors, 36,* 101395. https://doi.org/10.1016/j.measen.2024.101395.

[3] Alkhaldi, T. M., Darem, A. A., Alhashmi, A. A., Al-Hadhrami, T., & Osman, A. E., (2024). Enhancing smart city IoT communication: A two-layer NOMA-based network with caching mechanisms and optimized resource allocation. *Computer Networks, 255,* 110857. https://doi.org/10.1016/j.comnet.2024.110857

[4] Arachchige, K. G., Murtaza, M., Cheng, C. T., Albahlal, B. M., & Lee, C. C., (2024). Blockchain-enabled mitigation strategies for distributed denial of service attacks in IoT sensor networks: An experimental approach. *Computers, Materials & Continua, 81*(3), 3679–3705. https://doi.org/10.32604/cmc.2024.059378.

[5] Awan, N., Khan, S., Rahmani, M. K. I., Tahir, M., Alam, N. A., Alturki, R., & Ullah, İ., (2020). Machine learning-enabled power scheduling in IoT-based smart cities. *Computers, Materials & Continua, 67*(2), 1045–1060. https://doi.org/10.32604/cmc.2021.014386.

[6] Azzalini, D., Flammini, B., Emanuele, C. A., Guadagno, A., Ragaini, E., & Amigoni, F., (2024). An empirical evaluation of deep autoencoders for anomaly detection in the electricity consumption of buildings. *Energy and Buildings, 327,* 115069. https://doi.org/10.1016/j.enbuild.2024.115069.

[7] Balcıoğlu, Y. S., & Sezen, B., (2024). K-en yakın komşu (KNN) ve MLP yöntemi ile hibrit bir sistem: Trafik akış tahmini. *Afyon Kocatepe Üniversitesi Sosyal Bilimler Dergisi, 26*(4), 1801–1816. https://doi.org/10.32709/akusosbil.1255897.

[8] Collado-Montañez, J., Martín-Valdivia, M.-T., & Martínez-Cámara, E., (2025). Data augmentation based on large language models for radiological report classification. *Knowledge-Based Systems, 308,* 112745. https://doi.org/10.1016/j.knosys.2024.112745.

[9] Dahooie, J. H., Mohammadian, A., Qorbani, A. R., & Daim, T., (2023). A portfolio selection of Internet of Things (IoTs) applications for the sustainable urban transportation: A novel hybrid multi-criteria decision making approach. *Technology in Society, 75,* 102366. https://doi.org/10.1016/j.techsoc.2023.102366.

[10] Doğaroğlu, B., & Çalışkanelli, S. P., (2022). Akıllı otopark sistemlerinde kullanılan araç tanıma teknolojileri üzerine bir inceleme. *Akıllı Ulaşım Sistemleri ve Uygulamaları Dergisi, 5*(2), 85–98. https://doi.org/10.51513/jitsa.1098978

[11] Donta, P. K., Srirama, S. N., Amgoth, T., & Annavarapu, C. S. R., (2022). Survey on recent advances in IoT application layer protocols and machine learning scope for research directions. *Digital Communications and Networks, 8*(1), 1–16. https://doi.org/10.1016/j.dcan.2021.10.004.

[12] Dui, H., Li, H., Dong, X., & Wu, S., (2025). An energy IoT-driven multi-dimension resilience methodology of smart microgrids. *Reliability Engineering & System Safety, 253,* 110533. https://doi.org/10.1016/j.ress.2025.110533.

[13] Idrissi, Z. K., Lachgar, M., & Hrimech, H., (2024). Blockchain, IoT and AI in logistics and transportation: A systematic review. *Transport Economics and Management, 2,* 100002. https://doi.org/10.1016/j.team.2024.09.002.

[14] İlyas, S., & Albayrak, Y., (2023). Comparison of machine learning models for traffic volume estimation at smart intersections. *Journal of Engineering Sciences, 7*(5), S14. https://doi.org/10.30855/gmbd.0705S14.

[15] Kakız, A. T., Kakız, M. T., & Çoban, R., (2022). An evaluation of autoencoder neural network role in IoT edge computing. *Osmaniye Korkut Ata Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 5*(3), 1383–1392. https://doi.org/10.47495/okufbed.1037534.

[16] Karri, C., Machado, J. J. M., Tavares, J. M. R. S., Dannana, S., Gottapu, S. K., Gandomi, A. H., & Jain, D. K., (2024). Recent technology advancements in smart city management: A review. *Computers, Materials & Continua, 81*(3), 2105–2130. https://doi.org/10.32604/cmc.2024.058461.

[17] Katsigiannis, M., & Mykoniatis, K., (2024). Enhancing industrial IoT with edge computing and computer vision: An analog gauge visual digitization approach. *Manufacturing Letters, 41,* 1264–1273. https://doi.org/10.1016/j.mfglet.2024.09.153.

[18] Kaya, Ş. M., İşler, B., Abu-Mahfouz, A. M. A., Rasheed, J., & AlShammari, A., (2023). An intelligent anomaly detection approach for accurate and reliable weather forecasting at IoT edges: A case study. *Sensors, 23*(5), 2426. https://doi.org/10.3390/s23052426.

[19] Kheder, M. Q., & Mohammed, A. A., (2024). Real-time traffic monitoring system using IoT-aided robotics and deep learning techniques. *Kuwait Journal of Science, 51*(1), 1–10. https://doi.org/10.1016/j.kjs.2023.10.017.

[20] Khemakhem, S., & Krichen, L., (2024). A comprehensive survey

on an IoT-based smart public street lighting system application for smart cities. *Franklin Open, 8,* 100142. https://doi.org/10.1016/j.fraope.2024.100142.

[21] Moura, D. L. L., Aquino, A. L. L., & Loureiro, A. A. F., (2024). An edge computing and distributed ledger technology architecture for secure and efficient transportation. *Ad Hoc Networks, 164,* 103633. https://doi.org/10.1016/j.adhoc.2024.103633.

[22] Özcan, K. Y., (2023). Components of smart cities: Smart city applications and smart space management. *Mediterranean Journal of Humanities, 13,* 295–312. https://doi.org/10.13114/MJH.2023.607.

[23] Rahmani, A. M., Tanveer, J., Gharehchopogh, F. S., Rajabi, S., & Hosseinzadeh, M., (2024). Novel offloading strategy for multi-user optimization in blockchain-enabled mobile edge computing networks for improved Internet of Things performance. *Computers and Electrical Engineering, 102,* 109514. https://doi.org/10.1016/j.compeleceng.2024.109514.

[24] Rajagopal, S. M., Supriya, M., & Buyya, R., (2025). Leveraging blockchain and federated learning in edge-fog-cloud computing environments for intelligent decision-making with ECG data in IoT. *Journal of Network and Computer Applications, 204,* 104037. https://doi.org/10.1016/j.jnca.2024.104037.

[25] Tekouabou, S. C. K., Alaoui, E. A. A., Cherif, W., & Silkan, H., (2022). Improving parking availability prediction in smart cities with IoT and ensemble-based model. *Journal of King Saud University - Computer and Information Sciences, 34*(3), 586–595. https://doi.org/10.1016/j.jksuci.2020.01.008.

[26] Tran-Dang, H., & Kim, D.-S., (2025). Digital twin-empowered intelligent computation offloading for edge computing in the era of 5G and beyond: A state-of-the-art survey. *ICT Express, 11*(1), 167–180. https://doi.org/10.1016/j.icte.2025.01.002.

[27] Utku, A., (2023). Derin öğrenme tabanlı trafik yoğunluğu tahmini: İstanbul için bir vaka çalışması. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 11*(3), 1584–1598. https://doi.org/10.29130/dubited.1139534.

[28] Wei, J., & Ju, Y., (2024). Research on optimization method for

traffic signal control at intersections in smart cities based on adaptive artificial fish swarm algorithm. *Heliyon, 10*(10), e30657. https://doi.org/10.1016/j.heliyon.2024.e30657.

[29] Yalli, J. S., Hasan, M. H., Jung, T. L., & Al-Selwi, S. M., (2024). Authentication schemes for Internet of Things (IoT) networks: A systematic review and security assessment. *Internet of Things, 30,* 101469. https://doi.org/10.1016/j.iot.2024.101469.

[30] Yıldırım, Ü., & Çataltepe, Z., (2007). Örüntü tanıma ve öznitelik seçme yöntemleri kullanarak kısa zaman sonraki yol trafik hız öngörüsü. *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi, 3,* 45–53.

[31] Yu, X., & Do, Y., (2024). Perceptual and featural measures of Mandarin consonant similarity: Confusion matrices and phonological features dataset. *Data in Brief, 52,* 109868. https://doi.org/10.1016/j.dib.2023.109868.

[32] Zhang, G., Jin, J., Chang, F., & Huang, H., (2024). Real-time traffic conflict prediction at signalized intersections using vehicle trajectory data and deep learning. *International Journal of Transportation Science and Technology, 13*(4), 122–133. https://doi.org/10.1016/j.ijtst.2024.10.009.

[33] Zhao, J., Gao, Y., Tang, J., & Zhu, L., (2018). Highway travel time prediction using sparse tensor completion tactics and K-nearest neighbor pattern matching method. *Journal of Advanced Transportation, 2018,* 5721058. https://doi.org/10.1155/2018/5721058.