Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

IoT workload offloading efficient intelligent transport system in federated ACNN integrated cooperated edge-cloud networks

Abdullah Lakhan[1], Tor-Morten Grønli[1], Paolo Bellavista[2], Sajida Memon[3], Maher Alharby[4] and Orawit Thinnukool[5*]

Abstract

Intelligent transport systems (ITS) provide various cooperative edge cloud services for roadside vehicular applications. These applications offer additional diversity, including ticket validation across transport modes and vehicle and object detection to prevent road collisions. Offloading among cooperative edge and cloud networks plays a key role when these resources constrain devices (e.g., vehicles and mobile) to offload their workloads for execution. ITS used different machine learning and deep learning methods for decision automation. However, the self-autonomous decision-making processes of these techniques require significantly more time and higher accuracy for the aforementioned applications on the road-unit side. Thus, this paper presents the new offloading ITS for IoT vehicles in cooperative edge cloud networks. We present the augmented convolutional neural network (ACNN) that trains the workloads on different edge nodes. The ACNN allows users and machine learning methods to work together, making decisions for offloading and scheduling workload execution. This paper presents an augmented federated learning scheduling scheme (AFLSS). An algorithmic method called AFLSS comprises different sub-schemes that work together in the ITS paradigm for IoT applications in transportation. These sub-schemes include ACNN, offloading, scheduling, and security. Simulation results demonstrate that, in terms of accuracy and total time for the considered problem, the AFLSS outperforms all existing methods.

Keywords Edge AI, ACNN, Federated learning, Cloud, Transport applications

Introduction

Internet of Things (IoT) enabled public transport has been increasing daily [1]. Different modes of public transport, such as the metro, train, and bus, are operated in many metropolises around the world [2]. IoT-based applications like traffic prediction, ticketing, and trip planning are widely utilized in public transport. In European countries, it is common for trams and buses to be ridden on the road, while the metro and train are ridden on distinct routes [3, 4]. As a result, public transport providers collaborate to offer user-friendly services to passengers. Intelligent transport systems (ITS) are an emerging paradigm comprising artificial intelligence (AI) schemes, vehicles, traffic, fog, and cloud nodes. ITS offers automated services such as traffic prediction,

*Correspondence:
Orawit Thinnukool
orawit.t@cmu.ac.th
[1] Ubiquitous Computing Technology Laboratory, Kristiania University College, Kirkegata 24-26, 0153 Oslo, Norway
[2] Department of Computer Science and Engineering, Alma Mater Studiorum - Universitá di Bologna, Bologna, Italy
[3] Department of Computer System Engineering, Dawood University of Engineering and Technology, Karachi, Pakistan
[4] Computer Science Department, Taibah University, Medina, Saudi Arabia
[5] Embedded System and Computational Science Lab, College of Arts, Media and Technology, Chiang Mai University, Chiang Mai 50200, Thailand

vehicle collision detection, and pedestrian and vehicle detection based on mobility services in fog cloud networks [5].

Recently, significant advancements have been made in the Intelligent Transport Systems (ITS) paradigm, utilizing cooperative edge and cloud networks to achieve various objectives. Several studies such as [6–8] suggested secure and road traffic maintenance ITS systems based on ubiquitous fog cloud services in smart cities. However, these studies only considered one type of transport mode in their architectures. These studies investigated many modes of transport-enabled road traffic prediction [9–13]. The different modes, such as buses and vans, bikes, and cars, are considered in these studies. The pedestrian, road traffic, and collision types of objectives were investigated in these studies. The secure and big data-enabled processing of mobility-aware vehicle services for multi-mode transports was investigated in these studies [14–20]. These studies processed the heterogeneous data that was collected from different IoT sources. Using edge computing, these studies have implemented federated learning and convolutional neural network (CNN) schemes to process data from IoT sources. Edge computing is a subset of cloud computing that brings computing and storage services to radio networks. However, all the studies above deepened upon the black box of AI to make any decision without involving humans.

There are many research questions in ITS for IoT applications in edge cloud networks. (i) Existing ITS paradigms only consider roadside transport modes and provide services with a homogeneous routing map in smart cities. (ii) The presented AI schemes implicitly decide on IoT data with the explicit interaction of humans. (iii) Existing ITS are straightforward and need to consider the many constraints of IoT applications in their research paradigms.

This paper presents the novel Federated-IoT-enabled Intelligent Transport System (ITS) in Road Augmented Convolutional Neural Network (ACNN) Integrated Edge Networks. We consider the cooperative edge and cloud nodes together to perform workload offloading of vehicles and pedestrians on them. The objective is to optimize the time, cost, and security of IoT applications in edge cloud networks. The paper makes the following novel contributions to the research work:

- In this paper, we present an augmented federated learning scheduling scheme (AFLSS). Whereas AFLSS is an algorithmic methodology and consists of different sub-schemes: ACNN, offloading, scheduling, and security schemes in the ITS paradigm for transport IoT applications.
- In the paper, we present the federated learning-enabled ITS paradigm, which comprises different public transport modes in edge cloud networks. IoT applications such as traffic detection, ticketing, and trip planning are considered in this work.
- We introduce a security scheme based on a secure hashing algorithm (SHA-256) [21] to process the different routing data securely.
- We present the ACNN scheme, in which all parameters are configured explicitly to improve transparency, security, time, and cost for IoT applications in edge cloud networks.

The paper has the following subsections: "Related work" section discusses the existence of ITS paradigms and their findings and limitations. "Proposed intelligent transport system" section shows the components of the proposed ITS paradigm. "Proposed methodology" section shows the algorithm framework for the considered problem. "Experimental design" section shows the performance evaluation and results discussion. "Conclusion and future" section is the research conclusion and future work of the study.

Related work

These days, the usage of IoT transportation applications has been growing progressively. For instance, traffic prediction, ticketing validation, accident monitoring, map routing searching, and trip planning require real-time data on transports from the mobility environment. ITS obtained the optimal results in these network-enabled IoT applications. In related work, we discussed road-unit-side services based on cooperative edge cloud networks. The network management-enabled networking position enabled services with the mobility suggested in [1]. IoT applications offload workloads for execution and can invoke network-enabled routing services from any location in smart cities. However, this work focused on fixed services and scheduling decisions made during application design. The cooperative vehicle and pedestrian-enabled ITS paradigms presented in these studies [2–4]. These studies suggested cooperative schemes in which vehicles can offload their workload and communicate with each other to avoid any collisions with pedestrians in smart cities. These studies presented the trajectory positioning enabled locations, collected the content-aware IoT applications, and processed services in ITS

networks. However, these studies consider a single type of transport mode and fixed pedestrian data. The security constraints for the data have yet to be considered in these studies.

The offloading ITS paradigms for the security of IoT data are presented in these studies [5–7]. The federated learning integrated with the graph neural network and optimized the ITS services for IoT applications. These applications collect and offload different types of data: traffic, IoT location, and vehicle and pedestrian data from different sources in smart cities. Federated learning enabled ITS to process secure data among edge cloud networks. Computing nodes, such as edge and cloud computing, are placed in different places in smart cities to support transport data and allow seamless invoking of services in other places in networks. SHA-256 schemes are implemented in federated learning to process and offload cooperative data among computing nodes.

The secure big data offloading of traffic data among cooperatives processed by ITS paradigms in these studies [8–10]. The spatial big data networks of traffic signals and vehicle IoT devices are processed on different edge cloud networks in the networks. The Hadoop framework was deployed in these studies. The data was trained and tested based on the CNN scheme. Due to different modalities such as audio, video, and text, these studies processed the data on other CNN channels in the edge networks. However, due to heterogeneous nodes, data security is a critical challenge in ITS paradigms. The data security in heterogeneous nodes, such as edge and cloud nodes, is based on centralized and blockchain technologies presented in these works [11–15]. The ITS paradigms utilize various security methods, such as proof of work, proof of credibility, and Byzantine fault-tolerant schemes, to ensure the safety of IoT applications running on diverse computing nodes. However, data modality training and validation on the centralized incurred higher power consumption, resources, and costs for IoT applications and service providers in ITS paradigms.

The fine-grained and coarse-grained offloading and scheduling enabled semi-vertical federated learning based on CNN efficient ITS paradigms are presented in [16–20]. The main goal is to meet the training and validation of data modality at different nodes with the minimum resource consumption and time at heterogeneous computing nodes. Federated learning allows different edge computing nodes to train different modality data and aggregate them on the centralized node for making the final decision. Data security, time, and cost constraints are optimized in these paradigms. These studies [21–25] studies suggested joint offloading and scheduling schemes for vehicular applications in cooperative edge cloud networks. The goal was to minimize application processing services' response time, energy, and cost. These studies [26, 27, 25, 28–30] suggested fine-grained and coarse-grained offloading and scheduling schemes for vehicular applications. These studies considered security constraints during the offloading and scheduling of IoT vehicular applications on edge cloud networks. The secure coarse-grained offloading scheme (SCOS) and fine-grained secure offloading scheme (SFOS) in these studies suggested the security and time mechanism on different nodes such as local vehicle nodes, wireless nodes, and cloud nodes.

To our knowledge, Federated-IoT-enabled Intelligent Transport Systems (ITS) in Road Augmented Convolutional Neural Network (ACNN) Integrated Edge Networks have yet to be studied. The augmented CNN in federated learning added the setting of a method in which humans and machines perform together instead of decisions made by the machine learning algorithms. We integrated federated learning on different edge cloud networks to train and test data based on CNN methods.

Proposed intelligent transport system

This paper considers the different transport modes such as trams, buses, metros, and trains with different routing operations and data in smart cities. This paper presents the ITS system as shown in Fig. 1 consisting of different levels and components. The local routing-level servers collect and provide infrastructure-level information in ITS. These are distinct local servers such as cars, pedestrians, buses, and trams managed by their local routing operation in their regions. However, the local routing level only collects information from vehicles and pedestrians through the signal and cameras.

The operation level consisted of heterogeneous, scalable edge nodes in distributed ITS at the smart city level. Each edge node trains the data collected from the routing level using an augmented convolutional neural network (ACNN). Due to the volume of data, we divided the transport data into local datasets based on federated learning. Level 3 is the highly scalable, centralized cloud-based server that aggregates all trained data offloaded from edge computing and aggregates them as a single

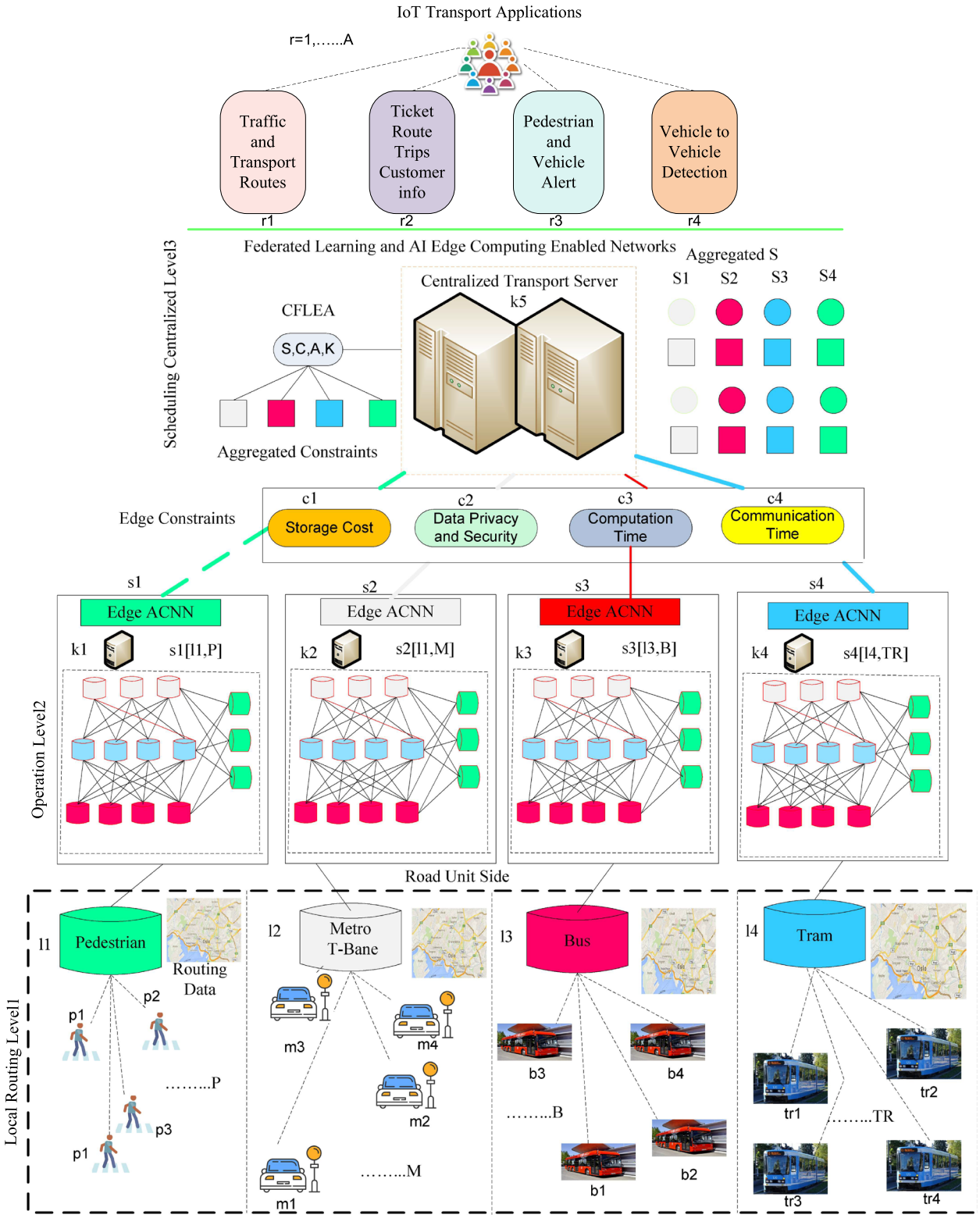

Fig. 1 IoT workload offloading efficient intelligent transport system in federated ACNN integrated cooperated edge-cloud networks

large dataset. We detail the proposed ITS paradigm's process in the problem formulation.

We differentiate all transport by their local routes in the proposed ITS paradigm. In our work, we assume that L number of route data points is generated by local routing. Each local server has speed and fixed resources, e.g., $\epsilon_z \in Z$ and $\zeta_z \in Z$. We consider the $l1 = \{p = 1, \ldots, P\}$ number of pedestrians walking on the road-unit-side on the designated routes in smart cities. In ITS, we are also considering ticketing IoT applications. Therefore, passengers can pay and validate tickets in different transports. Therefore, we differentiate the transport modes by their types. The number of cars is represented by $l1 = \{m = 1, \ldots, M\}$. The number of public buses is represented by $l3 = \{b = 1, \ldots, B\}$. We present the trams in this expression $l4 = \{tr = 1, \ldots, TR\}$. As defined above, each component (e.g., pedestrian, car, tram, and bus) has its own generated local routes. Therefore, for the operation, all data is offloaded to the number of edge nodes, e.g., $\{k = 1, \ldots, K\}$. Each edge and cloud node $k \in K$ is scalable and heterogeneous and contains the resources ϵ_k and speed ζ_k in the ITS paradigm. Each edge node trained the offloaded routing data based on ACNN deep learning methods based on given parameters. We consider the tuple of parameters of ACNN in the following way: $< input, conv, pool, FC, softmax >$. We represent the trained data based on their features by $\{s = 1, \ldots, S\}$. The particular dataset expressed as s and S shows the aggregated dataset of all merged trained datasets into one centralized server. We assume that the considered problem has different constraints, e.g., $\{c = 1, \ldots, C\}$ as shown in Fig. 1.

It is assumed that IoT applications, such as ticketing, pedestrian and vehicle detection, the timetable of transport routing, and searching by $\{r = 1, \ldots, A\}$, are utilized. Therefore, each IoT application has many constraints, such as $\{c = 1, \ldots, C\}$, during their execution in the ITS paradigm on the road-unit side. We determined the local routing time in the following way:.

$$Local = \sum_{l=1}^{L} \sum_{z=1}^{Z} \frac{l}{\zeta_l}. \tag{1}$$

Equation (1) determines the local processing time of local routing of different transports in level 1.

$$Edge = \sum_{s=1}^{S} \sum_{k=1}^{K} \frac{s}{\zeta_k}. \tag{2}$$

Equation (2) determines the edge processing time of operation on different offloaded data from the local routing in ITS.

$$Cloud = \sum_{s=1}^{S} S = \frac{s}{\zeta_k}. \tag{3}$$

Equation (2) determines the edge processing time of operation on different offloaded data from the local routing in ITS.

$$Com = \sum_{=1}^{L} \sum_{k=1}^{K} \sum_{z=1}^{Z} z \sim k \frac{l}{Up - bandwidth} + \frac{l}{download} \tag{4}$$

Equation (4) shows the communication time between local routing to the edge and cloud nodes during offloading and downloading data.

$$Enc = \sum_{l=1}^{L} \sum_{s=1}^{S} \sum_{z=1}^{Z} \sum_{k=1}^{K} (PK, l, s, p, q). \tag{5}$$

Equation (5) determines the encryption process of all device requests with public and private keys.

$$Dec = \sum_{l=1}^{L} \sum_{s=1}^{S} \sum_{z=1}^{Z} \sum_{k=1}^{K} Enc(PV, l, s, p, q). \tag{6}$$

Equation (6) determines the encryption process of all requests on all devices with public and private keys.

The accuracy of the prediction is determined in the following way:.

$$accuracy = \sum_{l=1}^{L} \sum_{s=1}^{S} \sum_{r=1}^{A} \sum_{k=1}^{K} \frac{s}{\zeta_k} r \sim S. \frac{TP + TN}{TP + TN + FP + FN}. \tag{7}$$

Equation (7) determined the accuracy of the applications for the given tasks on the request data.

The total time for the IoT applications is determined in the following way:

$$Total = \sum_{r=1}^{A} Local + Com + Ede + Cloud + Enc + Dec. \tag{8}$$

Equation (8) shows the total processing and communication time on different levels during processing in the ITS paradigm.

Proposed methodology

This paper proposes the AFLSS algorithm methodology, which consists of different schemes. AFLSS consisted of offloading, CNN, and scheduling schemes to solve the combinatorial problem with the given constraints.

Algorithm 1 AFLSS algorithm methodology

```
Input : L, S, K, Z, A
begin
    foreach (r=1 to R) do
        Call federated learning scheme to merge the local datasets S;
        Invoke all services of IoT transport applications A;
        Call Offloading Scheme for local routing L;
        Call ACNN scheme to process local routing data L ∼ S in secure form;
        Call scheduling scheme for prediction and allocation;
End process;
```

In AFLSS, all IoT applications are connected to the centralized server and invoke services from the centralized server. The centralized server is federated learning-enabled, which offers transport services based on trained and merged data from different datasets. IoT applications perform parallel offloading of data, invoke services from the local routing server, and are centralized during execution in the ITS paradigm. AFLSS Algorithm 1 has the following steps:

- The IoT applications offload the data securely to the local routing servers. At the operation level, the operation servers at the edge nodes train and validate the offload.
- The ACNN trained the data locally at the edge servers and offloaded the centralized server for further decision with the explicit parameters.
- All the edge servers and centralized servers in the ITS paradigm are collaborated based on oriental federated learning schemes.

Due to the many sub-schemes, we discussed the AFLSS methodology in detail based on the given scenario, as shown in Fig. 2. The local routing servers collected the vehicle, pedestrian, and signal data from their distinct routes. The data collection process collects data from different sources, such as pedestrian devices, vehicle devices, traffic signals, and routes. It is assumed to

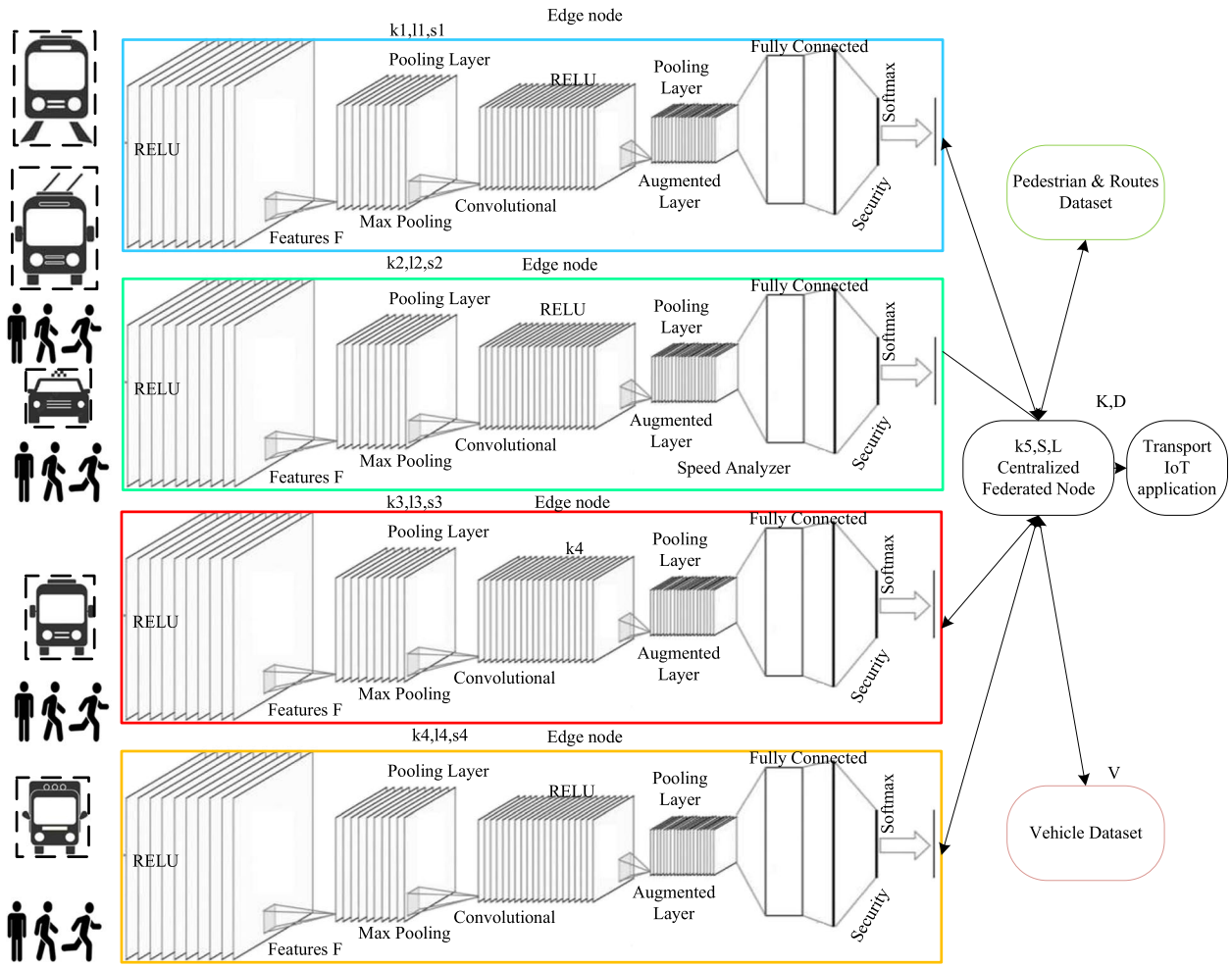

Fig. 2 Scenario: AFLSS algorithm methodology working schemes

be the data input for the edge nodes. The edge nodes are heterogeneous and process the locally generated raw data with the pre-processing technique and train them based on an augmented CNN scheme before offloading the centralized server for further decision-making. The augmented CNN is explainable and has customized layers, such as training and security, added to the CNN algorithm. The decision parameters, such as training, security, and offloading, are explicitly set in our ACNN scheme. All the locally trained datasets at edge nodes, e.g., *S*, *L*, *K*, are offloaded to the centralized server for the final decision and invoking services for IoT applications. The centralized process scheduled the merged trained datasets for the IoT applications and offered them many services in the ITS paradigm, as shown in Fig. 2.

Local routing data processing

Local routing is a complex process in smart cities. In particular, along with pedestrians, different vehicle modes are used on the road. For instance, electric scooters, cars, bicycles, trucks, buses, and other vehicles. Therefore, safety for pedestrians is paramount. Consequently, we suggest the life safety ITS paradigm, which collects information on all transport modes and traffic data from different sources as part of the local routing. In this paper, the local routing shows the basis of data collection from various IoT devices during execution in intelligent cities. We present the local routing scheme with other steps as shown in Algorithm 2. The local routing servers only collect information in the form of data from different sources. The data has various forms, including images, location coordinates, and text. Algorithm 2 shows data collection from other sources, and servers offload data in real-time to the edge servers for further preprocessing and training based on given requirements.

Algorithm 2 AFLSS algorithm methodology

```
Input : L, Z
begin
    foreach (l=1 to L) do
        Determine collecting data time based on equation (1) and equation (7);
        if l ∼ z then
            Collect the data from different sources;
        if (Offload.True=1) then
            Offload all generated data from local routing to edge servers;
        Offloading End;
    Invoke Edge Servers;
End process;
```

Edge nodes ACNN scheme

We integrated the security in the ACNN scheme, where the security of offloaded workloads was validated based on deadline, resource, and time constraints in the system. In our framework, we consider the different heterogeneous edge nodes that are placed in the local smart cities. These edge nodes are heterogeneous, which means they have different computing capabilities and resource thresholds. We connected all edge nodes through federated learning as operation nodes, where distinct datasets are trained at local edges, and the security of their datasets is maintained at the edge nodes. Algorithm 3 determines the training of datasets on different edge nodes based on augmented CNN. We have integrated the different datasets for transport applications on different edge nodes, pedestrians, vehicle detection, trams, and traffic. We trained datasets on local autonomous edge nodes using augmented CNN and federated learning. The trained weights of the datasets were then sent to the aggregated node so that they could be run. We define the steps of Algorithm 3 in the following way:

- The Algorithm 3 takes the input as different computing nodes, datasets, and hyper-parameters on different channels on different computing nodes.
- This expression *S* are federated weights, where trained datasets are offloaded to the aggregated for computing.
- We set the different hyper-parameters of augmented CNN, such as convolutional, hidden, dense, and fully connected layers, with the modification. Here, we added the security layer for encrypting the trained models before sharing them with the aggregated node.
- Each trained model has weight and security with a distinct index, as shown in steps 1 to 6.
- Each edge node implements an augmented CNN where channels process the different dataset values and extract their features, as shown in steps 7 to 34.
- We encrypted the trained model with additional two-way verification based on advanced standard encryption (AES-256) asymmetric schemes. Both public and private keys are used to encrypt and decrypt trained models for merge and execution.
- The softmax function gets the results based on the probability function for different channels on different edge nodes. The softmax function has different probability values, such as 0.3, 0.7, and 0.6. The probability value, e.g., 0.3 in the softmax function, determined that we trained according to the given requirements, and it is the same for all nodes.

Algorithm 3 Secure edge assisted ACNN scheme

Input : L, S, K
begin
  Federated learning layers as CNN channels;
  S are federated weights and initialization index=0;
  foreach *(index=0 to S)* **do**
    Get ready for Pedestrian CNN channel one input;
    $s \sim index = 1$;
    $s1[l1, P, k1]$;
    Apply Convolutional layer;
    Extracted the security features;
    AES.Encrypt $s1[l1, P, k1]$;
    AES.Decrypt $s1[l1, P, k1]$;
    Fully Connection layer;
    Softmax Probability;
    $s1[l1, P, k1]$=0.3;
    Get ready for Car CNN channel two input;
    $s \sim index = 2$;
    $s2[l2, M, k2]$;
    Apply Convolutional layer;
    AES.Encrypt $s2[l2, M, k2]$;
    AES.Decrypt $s2[l2, M, k2]$;
    Fully Connection layer;
    Softmax Probability;
    $s2[l2, M, k2]$=0.7;
    Get ready for Bus CNN channel 3 input;
    $s \sim index = 3$;
    $s3[B, P, k3]$;
    Apply Convolutional layer;
    AES.Encrypt $s3[B, P, k3]$;
    AES.Decrypt $s3[B, P, k3]$;
    Fully Connection layer;
    Softmax Probability;
    $s3[B, P, k3]$=0.6;
  End Data Modality Training;
  Merge $S = s1 + s2 + s3$;
End Main;

Federated learning enabled services and scheduling scheme

We implemented federated learning with additional nodes to improve the efficiency of transport services for different kinds of IoT applications. For instance, ticket validation and pedestrian and vehicle detection help avoid collisions in smart cities. We trained all workloads at different layer edge nodes and compared them to cloud computing for service providing to the applications. The scalability and interoperability of cooperated edge and cloud nodes are flexible and can be increased and decreased at runtime. We have defined the federated learning process in Algorithm 4, which consists of various steps. In steps 1 to 5, the locally trained datasets from different edge nodes are offloaded and merged at the aggregated node for decision-making and analysis for other applications. From steps 11 to 14, we performed federated learning. In our case, federated learning is the central server, where different edge nodes and applications are integrated to perform their tasks. Algorithm 4 performs the different functions in federated learning, such as the *Total* function and *accuracy* for different applications based on given task data. The decision will be based on a trained aggregated model for the assigned tasks. Federated learning is implemented at the central worker nodes, offering different services and transport modes to pedestrians in smart cities.

Algorithm 4 Federated learning enabled scheduling and aggregation scheme

Input : $MergeS = s1 + s2 + s3$
begin
  Creation Iterations for updating weights S;
  Iteration=500;
  foreach $(s = 1$ *to* $S)$ **do**
    Determined the aggregation;
    $S = s + iteration$;
    Updated S;
    foreach *()* **do**
      Call aggregated function;
      $k.Total$;
      $k.Accuracy$;
    End Aggregation;
  End Federated Updating;
End process;

Time complexity and space complexity

This paper considers workload offloading and scheduling in cooperative edge cloud networks. We considered the workloads, such as pedestrians, transport modes, and training data. To solve the problem, we presented the AFLSS algorithm scheme, which consisted of different schemes. Therefore, we determined the time complexity by $O(NlogN)$. Meanwhile, N shows the different number of schemes involved in AFLSS, and $log(N)$ shows the number of operations performed in the various schemes during workload offloading and scheduling in cooperated edge cloud networks.

Experimental design

In the performance evaluation, we show the performance of proposed methods for different transport applications on road-unit side services. The simulator consisted of different parameters, as shown in Table 1. We consider the different simulation parameters for conducting the different experiments for the problem. We repeated the experiments many times due to leveraging new data into the simulator and obtaining optimal results on real case scenarios. Simulation parameters shown in Table 1 consisted of different parameter details such as developing

Table 1 Simulation parameters

Value	Description
Platform	X86 Runtime
Resource	Docker Container
Language	JAVA, KOTLIN, Python and C
L	20
K	5
I	1-TB to 2 TB, 128 GB to 256 GB RAM
k	5 TB to 10 TB GB, 512 GB RAM
P	20,000
B	1000
M	2000
TR	500

languages, computing configuration, data, and different configuration setting values for experiments. We configured various JAVA, Python, Kotlin, and C programming parameters.

We exploited the real-time transport data of Reuter Oslo, which is available publicly for experiments. The data has a variety of parameters. However, we use only a few parameters according to our research problem. The considered parameters are transport type, positioning (start and end), route type with distinct identification numbers (ID), and traffic signals in smart cities. Table 2 illustrates the dataset of roadside unit paths in Oslo city. We downloaded the dataset of OSLO city from the Reuter website, the biggest public transport company in Oslo, Noways. Table 2 shows different routers for different transports and pedestrians and single waiting times in Oslo city. The implemented data is publicly available, updated, and leveraged after 24 hours.

Table 2 Dataset: Transports and road-unit Oslo

Transport-Type	Starts	RouteID	Signal(minutes)
Tram	Frognerseteren	1	10
Bus	Voksenkollen	1	10
Bus	Lillevann	1	10
Car	Skogen	1	10
Car	Holmenkollen	1	10
Tram	Skadalen	1	10
Tram	Vettakollen	1	10
Tram	Gaustad	1	10
Tram	Vinderen	1	10
Tram	Steinerud	1	10
Tram	Osteras	2	10
Tram	Osteras	2	10
Metro	Lijodet	2	10
Bus	Hovseter	2	10
Pedestrian	Borgen	2	10
Pedestrian	Kolsas	3	10
Pedestrian	Bergkrystallen	4	10
Car	Sognsvann	2	10

Implementation of AFLSS framework

We denoted the implementation components of the simulator in different classes, as shown in Fig. 3. We told the elements of object-oriented programming (OOP). In the simulator, we can reuse components from top-level abstraction to implementation, such as overloading and overriding. The method overloading shows that different transport methods, such as trams, trains, buses, and metros, can be implemented as inherited methods. In essence, Object-Oriented Programming (OOP) and Unified Modeling Language (UML) collaborate to encourage the generation of reusable elements by providing an organized and visual method for software design. OOP principles steer the crafting of modular and encapsulated classes, whereas UML diagrams function as tools for communication and visualization, facilitating the design and depiction of these reusable elements within a system. We denoted the simulation of the proposed work in different unified modeling languages (UML). The simulator AFLSS starts with editable encapsulated interfaces, which can be changed with depreciation. These interfaces are inherited from sub-classes such as routing, operation, and aggregated classes. The polymorphism allows multimode public transport methods to be implemented with different parameters for a single objective in IoT transport applications.

Object-Oriented Programming (OOP) and Unified Modeling Language (UML) allow the simulator to change the different versions, such as reusable elements, through various classes and object-based concepts. After implementing the simulator, the method depreciation and version will be changed at runtime. Therefore, the code with different versions will be provided in future updates, and a new programming interface application (API) will be upgraded for new features in the simulator.

Case study augmented CNN

We executed the different cases in the simulator, such as the vehicle-to-vehicle information and pedestrian-to-vehicle information on the other road unit sides, as shown in Fig. 4. Therefore, all transport applications have various services for public transport traffic in smart cities. It allows pedestrians and users to get information about public transport from other complex streets available for travel. The pedestrian can choose the optimal path for walking with less traffic, and the vehicle can choose the best route for driving and executing in intelligent cities. Therefore, the simulator is flexible, and more road unit-side services can be implemented, as shown in Fig. 4.

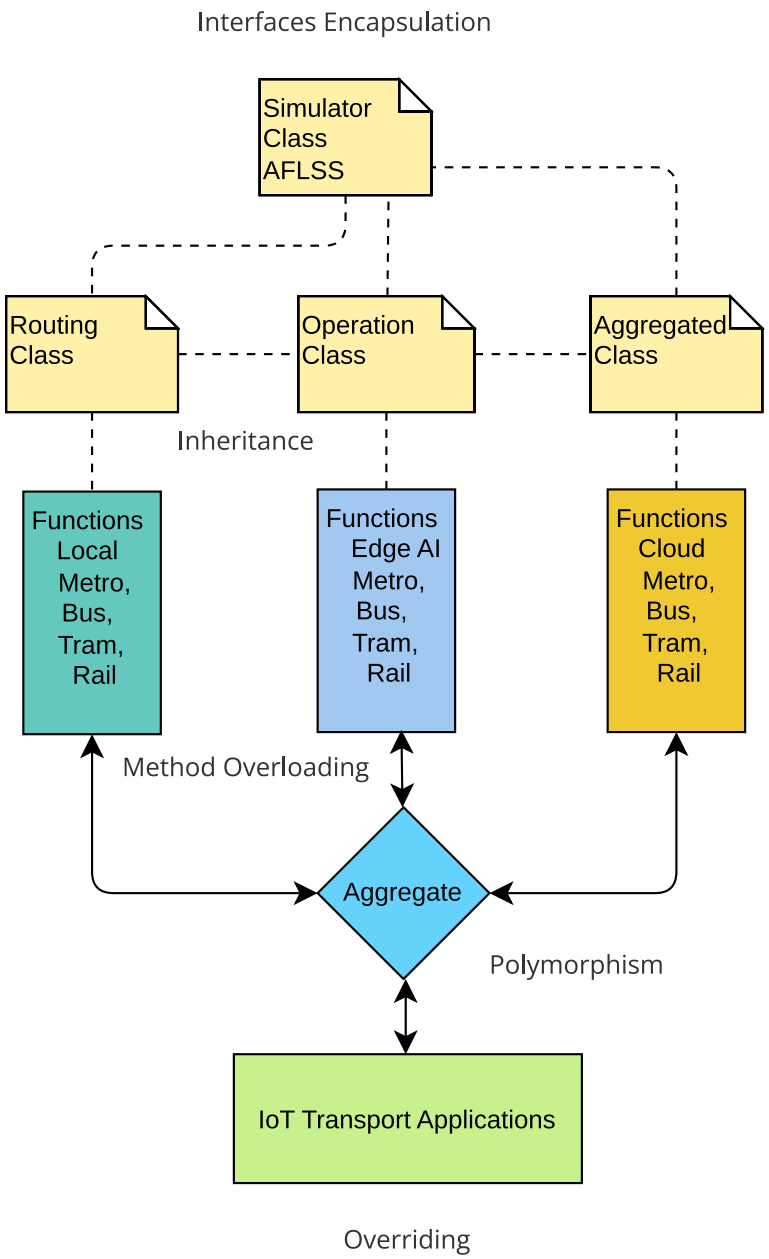

Fig. 3 Unified modeling language-assisted simulator classes

Result analysis

We design the setting of the experiments with different constraints such as accuracy, precision, recall, time, security, and resources during offloading and scheduling for vehicular applications. In the result analysis, we conducted different experiments according to the proposed architecture 1. The architecture consisted of local routing, operational, and scheduling data for IoT transport applications. Table 3 shows the analysis of different metrics for all IoT applications based on different methods. We compare the performances of methods with metrics such as method, application, total (minutes), accuracy, F1 score, recall, and precision. We determined the total time in minutes, and accuracy and other metrics are determined in percentages. Table 3 shows that AFLSS outperforms, has higher accuracy due to the augmented amendment in methods, and has a lower total time due to dividing the tasks offloaded into different edge and cloud networks. Table 4 shows the performances of different vehicles during pedestrian detection and vehicle

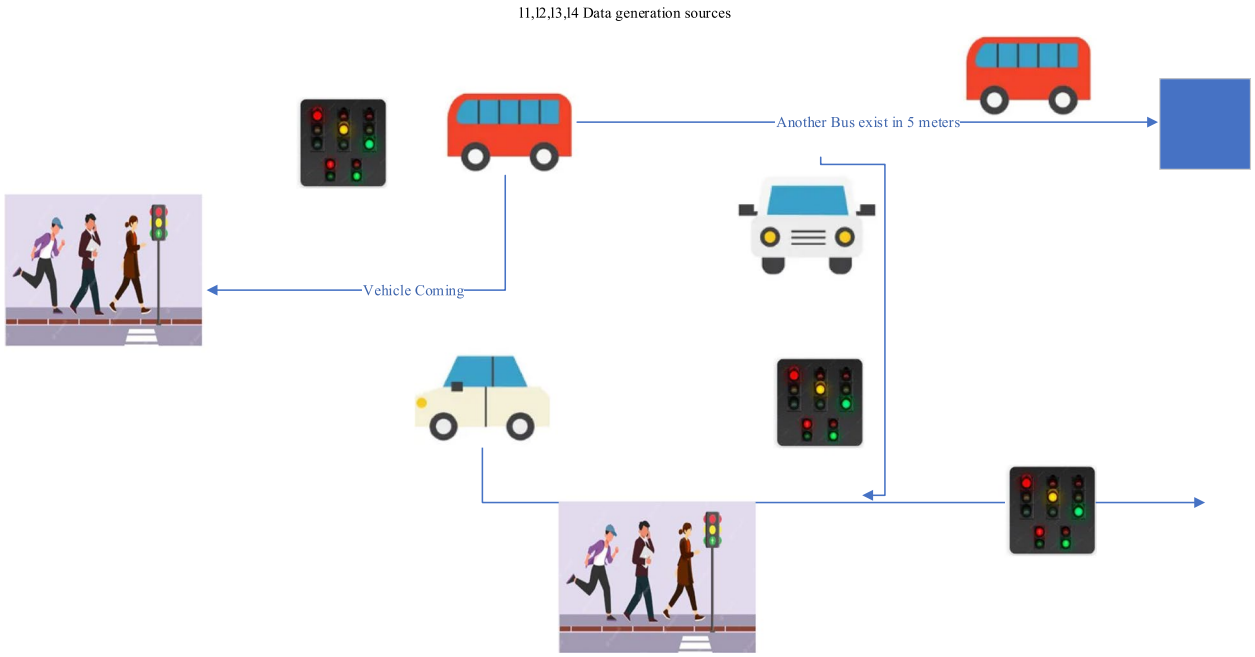

Fig. 4 Cooperative edge cloud road services for pedestrian and vehicle detection

Table 3 IoT Transport application results analysis

Method	Application	Total (minutes)	Accuracy	F1-Score	Recall	Precision
AFLSS	r1	7	98	0.97	0.96	0.98
AFLSS	r2	9	98	0.98	0.97	0.97
AFLSS	r3	6	99	0.97	0.96	0.97
FCNN	r1	13	92	0.91	0.92	0.93
FCNN	r2	17	93	0.92	0.93	0.92
FCNN	r3	16	94	0.93	0.92	0.93
CNN	r1	21	85	0.85	0.88	0.91
CNN	r2	25	83	0.84	0.89	0.91
CNN	r3	30	84	0.82	0.86	0.92

detection on the streets. IoT transport applications show pedestrian information availability to vehicles. Therefore, traffic, pedestrian, and vehicle information is collected at runtime. Table 4 shows that AFLSS outperformed all other vehicle and pedestrian detection methods in smart cities. We processed the 50,000 pedestrian image data and 50,0000 image data for training methods and detecting their object nature in smart cities. Figure 5 shows the performances of different methods in routing local data processing for ticket validation among different transport modes. These three methods, such as AFLSS, CNN, and FCNN, were integrated into the simulation environment to analyze the performance of local routing cooperation among different transport modes. The local routing computing nodes offload their data to the edge nodes for further analysis and training based on designated parameters. The parameters are data size, offloading time, ticket validation, and training accuracy to perform the different tasks. We consider the different tasks, such as vehicle cooperation among different transport modes due to customers traveling with the same ticket. Figure 5 shows that AFLSS outperformed CNN and FCNN regarding ticket validation among transport modes. The main reason is that we trained and divided the process into different parts, such as local routing, operation, and scheduling, and trained data on edge nodes with the minimum delay and higher accuracy. The aggregation server combines different transports from different edge servers and offers

Table 4 Pedestrian and vehicle detection results analysis

Method	Pedestrian	Vehicle	Accuracy	F1-Score	Recall	Precision
AFLSS	10,0000	20,000	99	0.98	0.97	0.96
AFLSS	20,0000	30,000	98	0.96	0.98	0.97
AFLSS	50,0000	50,000	99	0.98	0.99	0.99
FCNN	10,0000	10,000	92	0.91	0.92	0.93
FCNN	20,000	20,000	93	0.92	0.93	0.92
FCNN	50,000	50,000	94	0.93	0.92	0.93
CNN	10,000	10,000	85	0.85	0.88	0.91
CNN	20,000	20,000	83	0.84	0.89	0.91
CNN	50,000	50,000	84	0.82	0.86	0.92

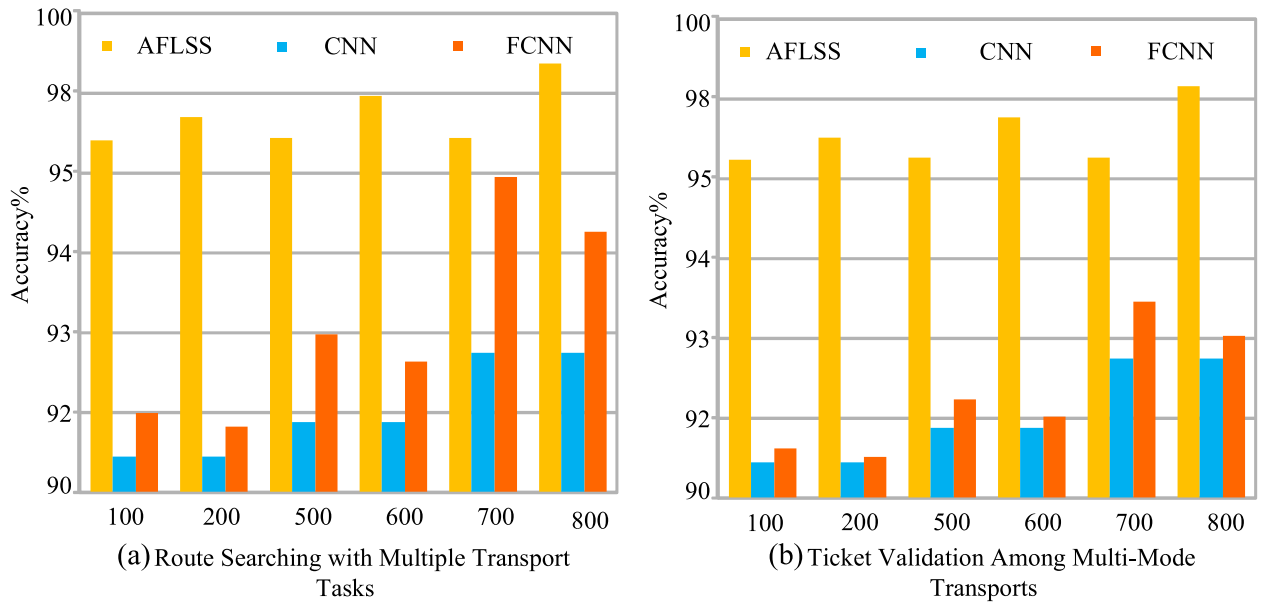

Fig. 5 Performance of methods local routing and IoT ticket validation application among different transport modes

services to the IoT application. It gained optimal results in terms of ticket validation accuracy as compared to existing federated learning and CNN methods for IoT transport applications. The main reason for the AFLSS algorithm is to perform optimally than existing CNN and FCNN is that, AFLSS augments and changes the parameter and feature settings at the runtime of offloading and scheduling when tasks are missing their deadlines. However, existing CNN and FCNN are fixed and do not allow to change the parameters, therefore, many tasks missed their deadlines and consumed much more resources as compared to AFLSS in simulation.

Figure 6 shows the performances of different methods for road cooperation between vehicles in cooperative edge cloud networks. We simulated this situation with different configuration parameters, such as 500 locations, 2000 road paths, and 100 to 200 meters for vehicle detection and collision avoidance. Figure 6 shows that AFLSS has higher accuracy in complex locations to detect vehicles and avoid collisions than CNN FCNN studies. The existing methods suffered from augmented properties, where deep learning and machine learning models depend on black-box decision models. The augmented properties allow the methods to decide with machines and humans to avoid any wrong vehicle collision. Therefore, the proposed work has higher accuracy than existing studies. We conducted the experiments based on real-time levering data from the dataset, to determine the traffic of transports on the road-unit side in smart cities. Therefore, we are considering the different cases such as pedestrian detection on the road-unit side, vehicle ratio, and transport detection on the road-side. AFLSS, outperformed in all cases as compared to existing methods to run the transport in the simulation.

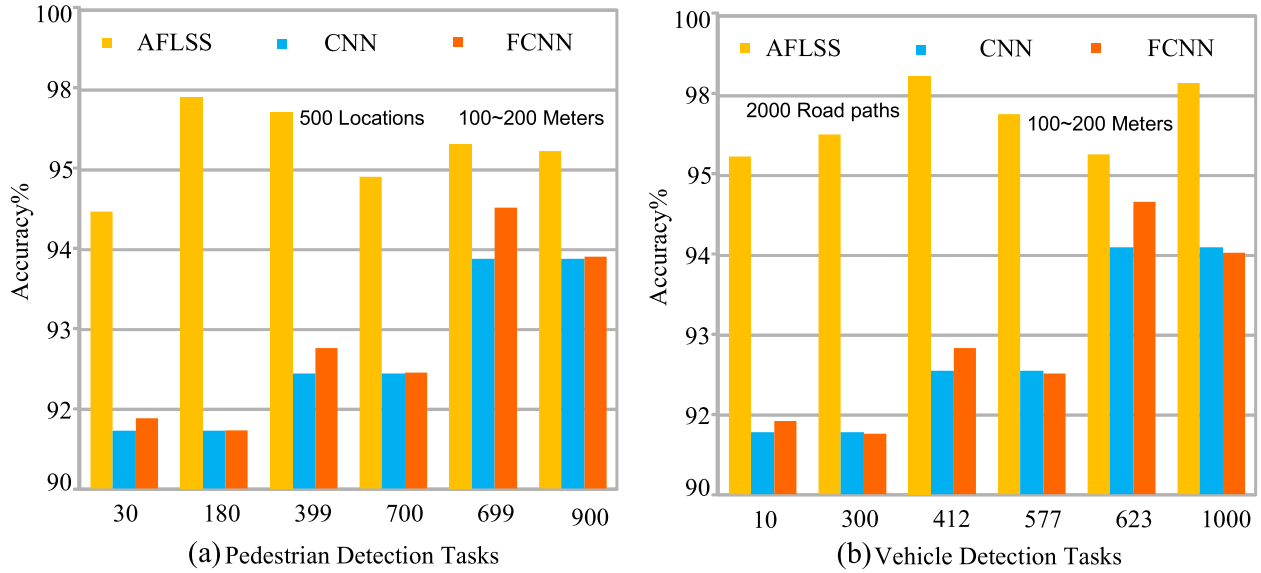

Fig. 6 Road cooperation performances for different IoT transport applications

Figure 7 shows different scenarios such as P2V (pedestrian to vehicle), V2V (vehicle to vehicle), and V2O (vehicle to object) performances in IoT applications in cooperative edge cloud networks. As shown in Fig. 7, the performance determined that road cooperation among moving and stationary objects can be detected and processed to avoid collisions on the road-unit side. Figure 7 shows that the proposed method obtained low total delays compared to existing methods to perform the aforementioned tasks in cooperative edge cloud networks. We implemented different cases where different applications and services could be used to avoid network collision and traffic detection. Traffic prediction and service transport mode availability can be determined for traveling in networking. Predicting traffic on different types of public transportation is a complicated and always-growing field that combines large data sets, advanced machine learning techniques, and real-time data flows. The main goal is to improve transportation systems, make operations more efficient, and make commuting better for everyone. Therefore, optimal results were obtained in different cases during the simulation, as

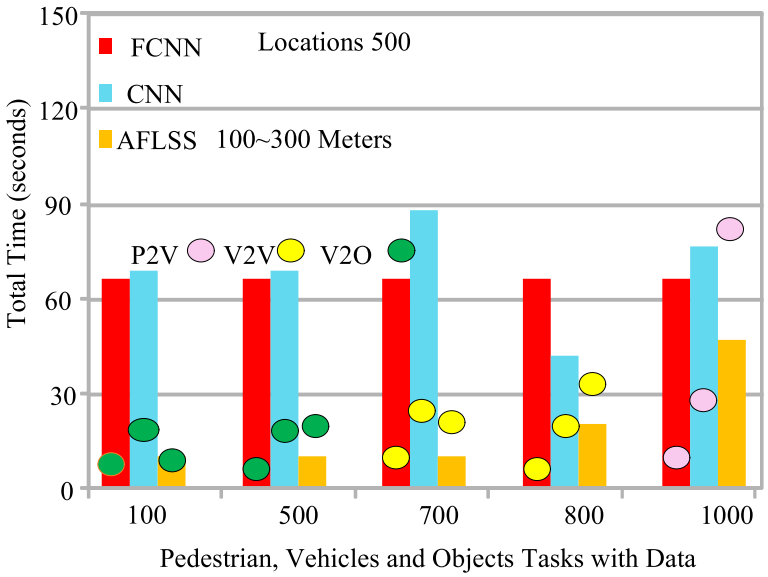

Fig. 7 IoT transport implementation at augmented CNN in smart cities

shown in Fig. 7. The proposed AFLSS outperformed in all cases and obtained the optimal results during the simulation of given data for experiments.

We exploited the different methods as baseline methods. This baseline research [26, 25, 27–30] approach implemented detailed and broad offloading and scheduling plans for vehicular applications. These investigations took into account security limitations while offloading and scheduling IoT vehicular applications on edge cloud networks. The studies introduced the Secure Coarse-Grained Offloading Scheme (SCOS) and Fine-Grained Secure Offloading Scheme (SFOS), outlining security and time mechanisms across various nodes, including local vehicle nodes, wireless nodes, and cloud nodes. Figure 8 shows the performance of security validity with time constraints between vehicles and edge and cloud servers. The x-axis shows the number of different multi-modal transport vehicles offload workloads with the consumer IoT devices with the parallel sequences and y-axis shows the security validity ratio with the time constraints in edge cloud networks. We integrated these baseline approaches with the proposed scheme such as AFLSS, FSOS, and CSOS. Figure 8 shows that AFLSS has a higher ratio of validation according to time constraints as compared to existing coarse-grained and fine-grained offloading schemes for transport applications. With the random stochastic vehicle IoT workloads such as x=32 transport vehicles have lower time constraints ratio with the security validation such as Y 0.727078 with AFLSS as compared to existing studies. However, other approaches with the x=16 and x=32 have higher time delay with the security validity among nodes as compared to proposed AFLSS during offloading and scheduling edge cloud networks. We devised an ACNN security mechanism based on the AES-256 algorithm where time, resource, deadline, and security validity are the features of tasks during offloading and scheduling on different IoT consumers' edge cloud networks. Therefore, AFLSS has optimal results of security validation with the trade-off between time and security among different offloading and scheduling workloads on different computing nodes for vehicular applications.

Finding and limitation

The main finding of this work is to design a practical cooperative ITS system for IoT transport applications. We have enhanced resource management, time utilization, and security constraints for IoT applications. However, resource instability leads to higher costs and overhead in resource allocations. Therefore, we aim to address these factors and constraints in the next version of our work and extend its application to other countries in the intelligent transport system.

Conclusion and future

In contemporary times, the intelligent transport system (ITS) offers various cooperative edge cloud services for roadside vehicular applications. The applications exhibited diverse functionalities, including ticket validation across transport modes and detecting vehicles and objects to prevent road collisions. The offloading process

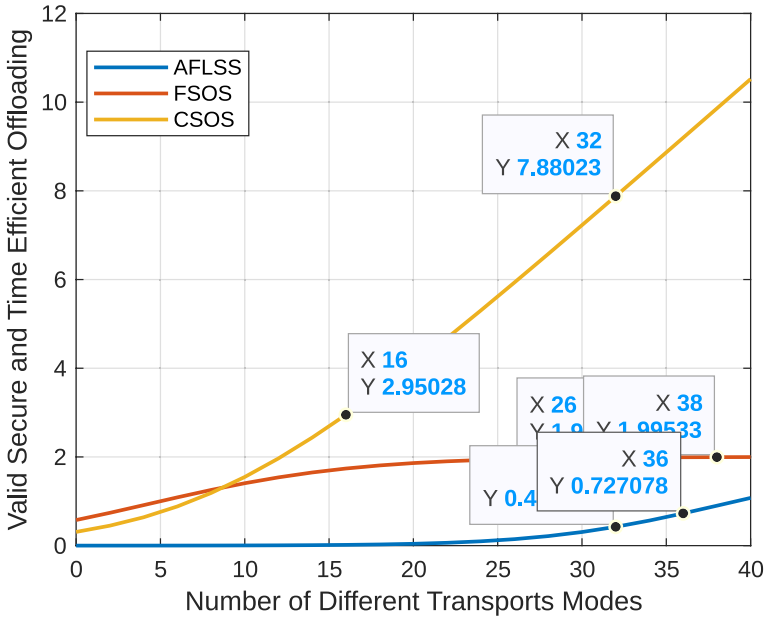

Fig. 8 Security offloading trade-off between time and validity

between cooperative edge and cloud networks was pivotal when devices (e.g., vehicles and mobile devices) had resource constraints, prompting them to offload their workloads for execution.

ITS employed various machine learning and deep learning methods for decision automation. However, the autonomous decisions made by these techniques in the past consumed a significant amount of time and required higher accuracy for the aforementioned applications on the road-unit side.

This research paper presents a novel offloading Intelligent Transport System (ITS) for IoT vehicles within cooperative edge cloud networks. The augmented convolutional neural network (ACNN) was presented as a model that trained workloads on different edge nodes. The ACNN facilitated collaboration between users and machine methods, enabling them to make decisions regarding offloading and scheduling workload execution jointly.

The paper also introduced an augmented federated learning scheduling scheme (AFLSS). AFLSS, an algorithmic method, comprised various sub-schemes that collaboratively operated within the ITS paradigm for IoT applications in transportation. These sub-schemes encompass ACNN, offloading, scheduling, and security. Simulation results indicated that AFLSS surpassed all existing methods regarding accuracy and total time for the problem.

In the future, we plan to apply blockchain technology to address the interoperability and cost issues in the proposed architecture for IoT transport applications.

Acknowledgements
This research was developed at the Ubiquitous Computing Technology Laboratory at Kristiania University College and funded by the Embedded System and Computational Science lab at the College of Arts, Media, and Technology, Chiang Mai University.

Authors' contributions
Abdullah Lakhan: Write the original manuscript, data analysis, and methodology. Tor-Morten Grønli: Supervision. Paolo Bellavista: Data and architecture design. Sajida Memon: Design experiment and results. Maher Alharby: Experiment and Result analysis Orawit Thinnukool: Funding, manuscript reading, improvement, and refinement.

Funding
This research received partial support from both Kristiania University College and Chiang Mai University, which collaborated on this project. The author acknowledges the support from Chiang Mai University, and the research group of Embedded Systems and Computational Science, which received partial support through donations.

Availability of data and materials
The data and code are available on the following URL: https://github.com/ABDULLAH-RAZA/Pedestrian-Vehicle-Non-Dataset. We added the different data additions with the different publications.

Declarations

Ethics approval and consent to participate
For all the algorithms, simulations, figures, and tables we wrote, there is no copyright issue with the existing studies.

Competing interests
The authors declare no competing interests.

Received: 28 January 2024 Accepted: 17 March 2024
Published online: 02 April 2024

References

1. Liu Q, Liu R, Zhang Y, Yuan Y, Wang Z, Yang H, Ye L, Guizani M, Thompson JS (2023) Management of positioning functions in cellular networks for time-sensitive transportation applications. IEEE Trans Intell Transp Syst
2. Autili M, Chen L, Englund C, Pompilio C, Tivoli M (2021) Cooperative intelligent transport systems: Choreography-based urban traffic coordination. IEEE Trans Intell Transp Syst 22(4):2088–2099
3. Ahmed U, Srivastava G, Djenouri Y, Lin JCW (2021) Deviation point curriculum learning for trajectory outlier detection in cooperative intelligent transport systems. IEEE Trans Intell Transp Syst 23(9):16514–16523
4. Richter A, Löwner MO, Ebendt R, Scholz M (2020) Towards an integrated urban development considering novel intelligent transportation systems: Urban development considering novel transport. Technol Forecast Soc Chang 155:119970
5. Gupta BB, Gaurav A, Marín EC, Alhalabi W (2022) Novel graph-based machine learning technique to secure smart vehicles in intelligent transportation systems. IEEE Trans Intell Transp Syst 24(8, August 2023):8483–8491
6. Arthurs P, Gillam L, Krause P, Wang N, Halder K, Mouzakitis A (2021) A taxonomy and survey of edge cloud computing for intelligent transportation systems and connected vehicles. IEEE Trans Intell Transp Syst 23(7, July 2022):6206–6221
7. Telang S, Chel A, Nemade A, Kaushik G (2021) Intelligent transport system for a smart city. Security and privacy applications for smart city development. Springer, p 171–187
8. Fantin Irudaya Raj E, Appadurai M (2022) Internet of things-based smart transportation system for smart cities. In: Intelligent Systems for Social Good: Theory and Practice, Springer, p 39–50
9. Liu C, Ke L (2023) Cloud assisted Internet of things intelligent transportation system and the traffic control system in the smart city. J Control Decis 10(2):174–187. Taylor \& Francis.
10. Kaffash S, Nguyen AT, Zhu J (2021) Big data algorithms and applications in intelligent transportation system: a review and bibliometric analysis. Int J Prod Econ 231:107868
11. Kumar R, Kumar P, Tripathi R, Gupta GP, Kumar N, Hassan MM (2021) A privacy-preserving-based secure framework using blockchain-enabled deep-learning in cooperative intelligent transport system. IEEE Trans Intell Transp Syst 23(9):16492–16503
12. Lv Z, Li Y, Feng H, Lv H (2021) Deep learning for security in digital twins of cooperative intelligent transportation systems. IEEE Trans Intell Transp Syst 23(9):16666–16675
13. Ahmed I, Zhang Y, Jeon G, Lin W, Khosravi MR, Qi L (2022) A blockchain- and artificial intelligence-enabled smart iot framework for sustainable city. Int J Intell Syst 37(9):6493–6507
14. Liao S, Wu J, Bashir AK, Yang W, Li J, Tariq U (2021) Digital twin consensus for blockchain-enabled intelligent transportation systems in smart cities. IEEE Trans Intell Transp Syst 23(11):22619–22629
15. Zhao J, Chang X, Feng Y, Liu CH, Liu N (2022) Participant selection for federated learning with heterogeneous data in intelligent transport system. IEEE Trans Intell Transp Syst 24(1):1106–1115
16. Manias DM, Shami A (2021) Making a case for federated learning in the internet of vehicles and intelligent transportation systems. IEEE Netw 35(3):88–94
17. Zhang C, Zhang S, James J, Yu S (2021) Fastgnn: A topological information protected federated learning approach for traffic speed forecasting. IEEE Trans Ind Inform 17(12):8464–8474
18. Lim WYB, Huang J, Xiong Z, Kang J, Niyato D, Hua XS, Leung C, Miao C (2021) Towards federated learning in uav-enabled internet of vehicles: A multi-dimensional contract-matching approach. IEEE Trans Intell Transp Syst 22(8):5140–5154
19. Zhu Y, Liu Y, James J, Yuan X (2021) Semi-supervised federated learning for travel mode identification from gps trajectories. IEEE Trans Intell Transp Syst 23(3):2380–2391

20. Gadekallu TR, Pham QV, Huynh-The T, Bhattacharya S, Maddikunta PKR, Liyanage M (2021) Federated learning for big data: A survey on opportunities, applications, and future directions. arXiv preprint arXiv:2110.04160
21. Bensalem H, Blaquière Y, Savaria Y (2021) Acceleration of the secure hash algorithm-256 (sha-256) on an fpga-cpu cluster using opencl. In: 2021 IEEE international symposium on circuits and systems (ISCAS), IEEE, p 1–5
22. Tang H, Wu H, Zhao Y, Li R (2021) Joint computation offloading and resource allocation under task-overflowed situations in mobile-edge computing. IEEE Trans Netw Serv Manag 19(2):1539–1553
23. Qu G, Wu H, Li R, Jiao P (2021) Dmro: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing. IEEE Trans Netw Serv Manag 18(3):3448–3459
24. Tang C, Wu H (2022) Joint optimization of task caching and computation offloading in vehicular edge computing. Peer Peer Netw Appl 15:854–869
25. Tang C, Yan G, Wu H, Zhu C (2023) Computation offloading and resource allocation in failure-aware vehicular edge computing. IEEE Trans Consum Electron
26. Huang X, Yu R, Xie S, Zhang Y (2020) Task-container matching game for computation offloading in vehicular edge computing and networks. IEEE Trans Intell Transp Syst 22(10, October 2021):6242–6255
27. Lakhan A, Groenli TM, Muhammad G, Tiwari P (2024) Evolutionary meta-heuristic offloading and scheduling schemes enabled industrial cyber-physical system. IEEE Syst J
28. Lakhan A, Mohammed MA, Abdulkareem KH, Deveci M, Marhoon HA, Nedoma J, Martinek R (2022) A multi-objectives framework for secure blockchain in fog-cloud network of vehicle-to-infrastructure applications. Knowledge-Based Syst 15:854–869
29. Tang C, Wu H (2022) Reputation-based service provisioning for vehicular fog computing. J Syst Archit 131:102735
30. Xu X, Shen B, Yin X, Khosravi MR, Wu H, Qi L, Wan S (2020) Edge server quantification and placement for offloading social media services in industrial cognitive IoV. IEEE Trans Ind Inform 17(4):2910–2918

Publisher's Note