An AI-Driven Intelligent Transportation System: Functional Architecture and Implementation

Árpád Huszák, Vilmos Simon, László Bokor, László Tizedes, and Adrian Pekar

Abstract—**The surge in urbanization and the concomitant growth of the urban population have exacerbated issues such as traffic congestion and air pollution across cities globally. While Intelligent Transportation Systems (ITS) offer promise for improving urban mobility, existing solutions predominantly exhibit limitations in scalability and adaptability, thus falling short in delivering city-wide traffic management. This unaddressed gap necessitates the development of a robust, scalable, and adaptive system that can manage the intricacies of urban traffic. Our work introduces CityAI, an automated, AI-driven framework designed to operate on a city-wide scale. The system harvests data from diverse sensing infrastructures, employing machine learning algorithms to predict future traffic states and patterns. Furthermore, it proposes real-time interventions, including adaptive traffic light control and V2X-based solutions. The architecture and components of CityAI not only incorporate state-of-the-art techniques but are also applied in real-world environments. The CityAI framework was implemented in the city of Pécs, Hungary, as a proof-of-concept ITS system. The framework enables city authorities to implement proactive measures, thus preventing traffic issues before they manifest. The paper focuses on practical development aspects of an ITS system undertaking R&D on new technologies, applications, and techniques which may facilitate future product development.**

Index Terms—**data analytics, Intelligent Transportation Systems, machine learning, traffic light control, vehicular communication**

I. Introduction

THE escalating trend of urbanization across the globe places enormous demands on existing infrastructure, most significantly on road traffic management systems [1]. Challenges arising from this include elevated energy consumption, increased air pollution, and an adverse impact on the quality of life for city inhabitants [2], [3]. Intelligent Transportation Systems (ITS) have emerged as promising tools to mitigate these issues, incorporating technologies such as machine learning, data analytics, and advanced communication systems [4]–[7].

However, these ITS solutions commonly suffer from limitations in their scope, scalability, and adaptability. They are often tailored for specific segments of a city or particular use cases, thereby lacking the versatility required for comprehensive, city-wide applications [8]–[10]. Additionally, the technical complexity of these systems poses a significant barrier for traffic operators who may not have expertise in data science or software engineering. These shortcomings are further compounded by regional and legal constraints such as stringent data protection regulations.

This paper introduces the CityAI system, an innovative ITS framework empowered by machine learning to achieve scalable and adaptable management of urban traffic networks. The aim of our work was to implement a proof-of-concept ITS system that includes state-of-the-art technologies, techniques, and applications that may facilitate future product development by giving guidelines for technical system design. The contributions of this work are as follows:

- it elucidates a novel approach for multi-modal transport integration using machine learning,
- it develops an adaptive traffic prediction model that can scale with the complexity of growing cities,
- it presents a comprehensive data-driven decision-making process, enhanced by a diverse set of data sources, and
- it proposes an adaptive and resilient architecture capable of real-time monitoring and rapid response to unforeseen events.
- it introduces a real-life implementation of a proof-of-concept ITS system deployed in the city of Pécs.

The remainder of this paper offers a comprehensive exposition of CityAI, focusing on its architecture and the functionalities of its key components to provide an in-depth understanding of its capabilities and its potential role in shaping the future of urban transportation management.

II. Related Work

In recent years, significant advancements in ITS have been driven by the integration of machine learning, data analytics, and advanced communication systems. Existing solutions like City Brain [11], developed by Alibaba Cloud and deployed in cities such as Hangzhou (China) and Kuala Lumpur (Malaysia), and European ITS software suites like Yunex Traffic [12], an independent company specializing in intelligent traffic systems after spinning off from Siemens Mobility, and Urban Traffic Management (UTM) [13] by SWARCO exemplify large-scale applications of artificial intelligence (AI) in urban management. However, the specific details of these systems' AI-based methodologies remain sparse, highlighting a gap in comprehensively documented, adaptive, and scalable AI-driven traffic management solutions. This statement is

Á. Huszák, V. Simon, L. Bokor, and A. Pekar are with the Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary.

Á. Huszák and L. Bokor are also with the HUN-REN-BME Cloud Applications Research Group, Budapest, Hungary.

A. Pekar is also with the HUN-REN-BME Information Systems Research Group, Budapest, Hungary.

L. Tizedes with the HUN-REN-SZTAKI Machine Perception Research Laboratory, Budapest, Hungary

Corresponding author: Árpád Huszák (huszak@hit.bme.hu).

DOI: 10.36244/ICJ.2024.3.2

also supported by collections of current V2X deployment activities in recent surveys (e.g., [14], [15]) highlighting that cooperative ITS solutions are in their early phases of adopting AI technologies.

Traditional machine-learning approaches have been widely applied to traffic forecasting and classification tasks using roadside sensors. Methods such as Hidden Markov models, gradient boosting regression trees, artificial neural networks, decision trees, support vector machines, Gaussian mixture models, and Bayesian networks have been successfully employed for short-term traffic prediction and travel time estimation [16]–[19]. These foundational techniques, while effective, often face challenges in scalability and adaptability for real-time, city-wide applications.

Recent advances have shifted towards deep learning models that capture spatial and temporal dependencies in traffic data. Long Short-Term Memory (LSTM) neural networks, stacked autoencoders, and fuzzy-based convolutional neural networks have shown promise in improving prediction accuracy under dynamic conditions [20]–[22]. Hybrid methods combining neural networks with statistical or optimization approaches, such as swarm intelligence and evolving fuzzy neural networks, further enhance the robustness and adaptability of traffic flow models [23]–[26].

Recent studies have focused on incremental learning and data stream processing techniques to address the growing need for real-time traffic management. For instance, trajectory clustering using hyperdimensional computing and smart traffic management platforms leveraging online incremental machine learning represent efforts to detect and adapt to real-time changes in traffic patterns [27], [28]. These approaches underscore the importance of handling the dynamic and streaming nature of urban traffic data.

Despite these advancements, many ITS solutions remain limited by their specificity to particular urban segments or technical complexities that hinder broader applicability. CityAI addresses these gaps by proposing a comprehensive, data-driven, and adaptive ITS framework that integrates multi-modal transport data, supports scalable traffic prediction models, and enables real-time monitoring and rapid response to urban traffic dynamics.

III. System Architecture

The architecture of CityAI is intricately designed to facilitate a comprehensive traffic management solution. It is organized around three major functional components, aligning with the focus of the upcoming sections: Data Collection (Section IV), Data Analytics (Section V), and Informed Traffic Governance and Visualization (Section VI). A schematic representation of the architecture is depicted in Fig. 1. In the following subsections, these functional groups are briefly overviewed. Detailed discussions concerning individual system components will be presented in subsequent sections.

Data Collection

Data Collection is primarily concerned with the acquisition and preprocessing of data. This functional group incorporates modules such as the Sensor Hub, which takes responsibility for data collection, standardization, and forwarding.

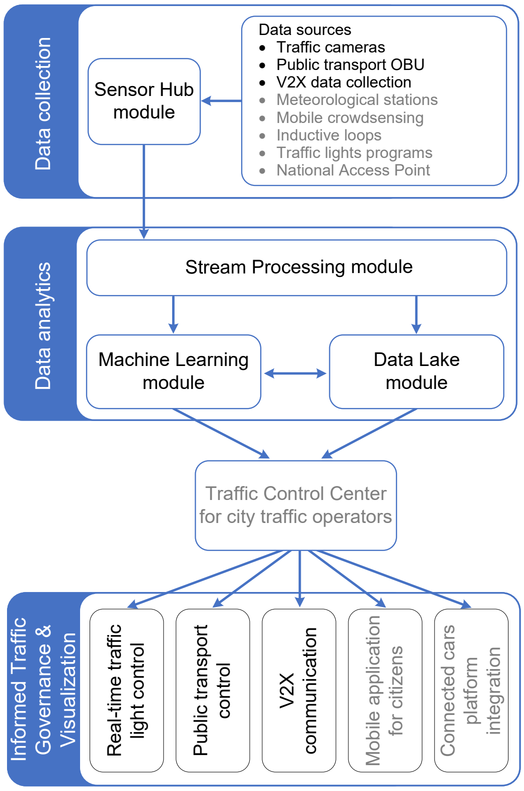

Fig. 1. CityAI general architecture, aligned with the thematic components of Data Collection, Data Analytics, and Informed Traffic Governance and Visualization.

Data Analytics

This functional group is tasked with intensive computational activities. It includes the Machine Learning and Data Lake modules, which engage in complex data analysis, traffic pattern recognition, and actionable insight generation.

Informed Traffic Governance and Visualization

This functional group is devoted to the effective application of acquired knowledge and insights for a range of tasks. These tasks include real-time traffic management, network optimization, and enhanced visualization. The analytics from the Data Analytics group are transformed into actionable interventions and also channeled into a visual interface for a more holistic understanding of urban mobility patterns.

IV. Data Collection

Data collection is one of the crucial elements of an ITS architecture, which produces the required input for all other modules of the system. Although the Sensor Hub module allows different data sources, such as meteorological stations and mobile application data (crowd sensing), in the deployed CityAI framework, three different real-time data sources are used currently: traffic cameras, public bus trajectories, and V2X information.

A. Traffic Cameras

The CityAI systems modules rely on estimated statistical information regarding vehicle traffic (speed, flow, occupancy) in urban environments by processing the images of preinstalled cameras. It is worth mentioning that the primary aim of these PTZ (pan-tilt-zoom) surveillance cameras - owned by the local authority of Pécs - is to ensure public safety. The position and view angle of the cameras made it very challenging to use them for traffic monitoring. Therefore, the system applied in this work has different characteristics from speed, flow, and occupancy (SFO) information extracted by standard traffic monitoring camera systems [29]. Here, we used cameras from a pre-installed network of urban surveillance cameras with preset positions that monitor only a particular section of the traffic path at a time. Due to preset changes, only periodical data acquisition from the monitored area was possible during intervals when the surveillance camera preset was monitoring a particular section. Also, an automatic detection algorithm was needed to determine the camera's current preset position.

1) Implementation of the system: After evaluating the data from the preliminary tests, our design choice was to use a distributed system architecture, where we deployed NVIDIA Jetson Nano embedded computers to process each camera image locally (*cf.* Fig. 2). In the processing pipeline, incoming camera images are pre-processed to determine which preset the camera is currently in, and then the Yolo neural network [30], [31] is applied to detect the objects visible in the image. After filtering by class, the vehicle object (auto, bus, truck, motorcycle) instances are fed into a tracker module to establish which objects in the current frame correspond to past object displacements. The camera image is calibrated to the real-world scene since we measured the projection of the camera image onto the road surface plane using the homography transformation [32]. Using this information, the speed of the tracked objects is computed by counting the pixel displacement on consecutive images. Also, we have set trigger and occupancy zones on the images. Therefore, using these zones one can calculate the SFO [33], [34] values of the passing objects as follows:

Speed is the current specified object speed for a given trigger zone (*cf.* Fig. 3).

$$Speed = \frac{s_c}{t_f}, \tag{1}$$

where s_c is the distance the center of the same object in two consecutive calibrated frames, t_f is the time that has passed between taking two consecutive frames.

$$Flow = \frac{d}{t} \cdot t_p, \tag{2}$$

where d is the numbered tracked objects belonging to the given trigger zone, t is the elapsed time (end of measurement - start of measurement), and t_p is the Flow time window rate (a multiplier calculated from the preset cycle time).

The occupancy statistical information is calculated using the occupancy zones shown in Fig. 4. The occupancy is the median value of the velocity of the objects within the zone divided by the number of objects at a time instant. Namely,

$$Occupancy = \frac{n}{Med_{i=0}^{n} v_i}, \tag{3}$$

where n is the number of objects (the number of vehicles in the occupancy zone at the moment of the measured time), v_i is the speed of the i-th object, and $Med_{i=0}^{n} v_i$ is the median speed of n objects. The calculated value is normalized between 0 and 100. The value is 0 if there is no traffic and 100 if the band is saturated. If the median speed is 0 and n is greater than 0, then a value of 100 is transmitted.

The data measured by the distributed Jetson Nanos are aggregated and periodically transmitted to higher-level components of the system for processing. Data communication relies on stream processing, utilizing a distributed streaming platform for efficient data ingestion through message queuing. This approach effectively manages data streams and promotes seamless communication between various system components. Asynchronous information transmission enables real-time pro-

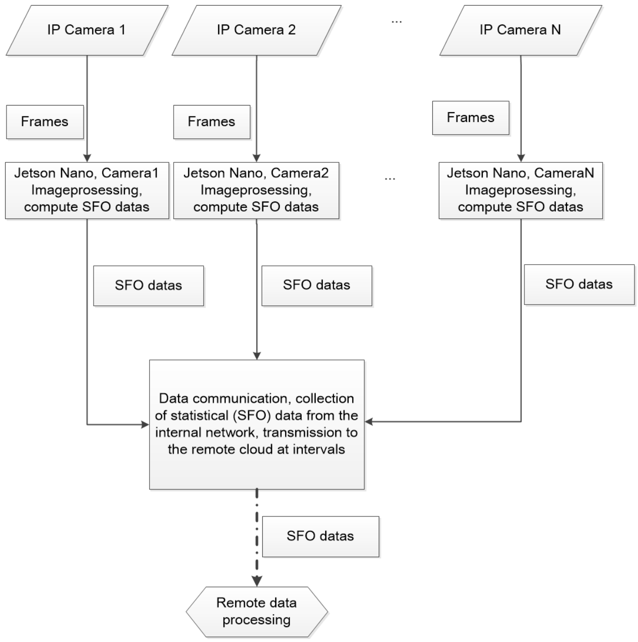

Fig. 2. Image processing data flow.

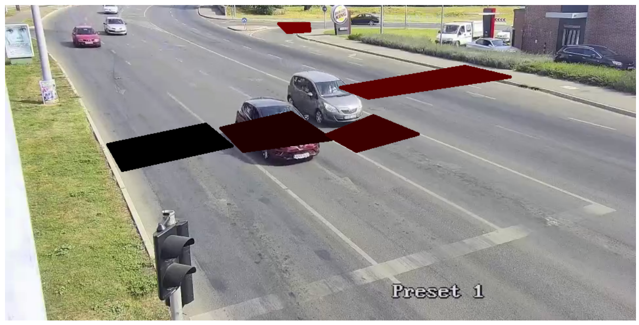

Fig. 3. The used trigger zones.

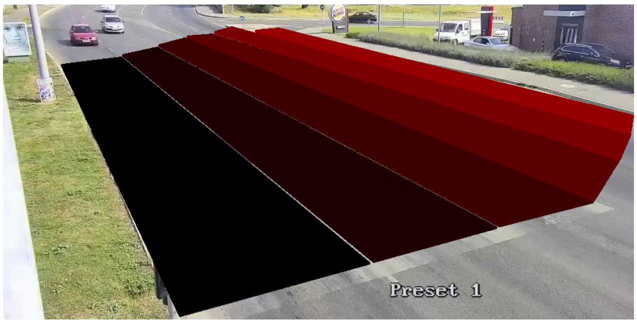

Fig. 4. The used occupancy zones.

cessing and analysis of data as it is generated. Additionally, the streaming platform ensures load balancing and load sharing for ingested data, boosting the system's scalability and fault tolerance.

The algorithm was tested under various weather and lighting conditions to assess the performance of the automatic SFO measurement system. We manually counted the number of vehicles across different time intervals to compare with the automated measurements. The evaluation was conducted at eight different measurement locations. Under daylight and favorable lighting conditions, there was no significant discrepancy between the manual and automated counts. However, in low visibility conditions, such as rain and twilight, an error margin of approximately 5-10% was observed, varying by location. The error primarily stemmed from the algorithm undercounting vehicles compared to manual counts. Additionally, the camera's viewing angle relative to the road had a minor influence on the algorithm's accuracy. The system also considers real-time weather data during operation. The error introduced by weather conditions and twilight can be mitigated by applying compensatory estimation values (5-10%) when making traffic decisions.

B. Public Transport Trajectories

The advancements in technology have led to an increase in the capabilities of in-vehicle sensors and on-board units, allowing them to collect and periodically report trajectory data [35], [36]. A vehicle trajectory, defined as the path generated by a moving vehicle in space [37], is recorded by vehicle trajectory data, which captures the movement of individual vehicles [38]. These trajectory data have become a crucial element in modern traffic management [39]. However, the collection of diverse data still poses a challenge due to factors such as privacy concerns and the administration of sensors by different entities.

Our system currently employs bus trajectories to improve traffic flow forecasting and congestion detection. Specifically, we utilize the trajectories of public buses operating in the city of Pécs, Hungary's 5th largest city. The buses are operated by TükeBusz Zrt., the local public transport service provider. At the time of system implementation, the company operated 202 buses on a 300 km network. The on-board units installed in the buses periodically collect and record various data. These data are then transmitted to a remote collection unit which forwards it to our system in a comma-separated format.

C. Vendor-independent V2X information collection/dissemination sub-system

The purpose of our proposed CityAI framework's V2X-based data collection and intervention modules is twofold. On the one hand, it aims at implementing standardized vehicular data exchange to support dynamic, adaptive, and fine-grained information gathering and dissemination tasks in the ITS domain. On the other hand, it provides solution portability by ensuring that the implementation works independently from the V2X device manufacturer's application programming interface and other vendor-specific details, making information exchange of data collection and intervention both feasible in a generic manner, independently of V2X implementations.

Our V2X sub-system is to be able to store and process the data generated by on-board and road-side units - the two basic infrastructure elements of vehicle communication – and present the resulting data set to other processing components in the framework. The proposed solution can act as an integration point in any complex ITS architecture where vehicular communication is considered: it converts manufacturer-specific V2X data into a vendor-independent format, creates/maintains connection with other backend components, and performs further data conversion so that the connected modular elements can easily process the data in a bidirectional way. Fig. 5 shows the general architecture of the V2X sub-system, highlighting the integration links and the most essential modules briefly introduced below.

- On-board Unit (OBU): its communication relies on CAM (Cooperative Awareness Message) and DENM (Decentralized Environmental Notification Message) services, which the RSU (Road Side Unit) receives and forwards to the data management component.
- Human Machine Interface (HMI): it can trigger various DENM messages and display the received traffic/accident information using the Google Maps API.
- Road-side Unit (RSU): RSUs forward the data received from the OBU to the centralized, vendor-independent data management component. We added a particular module to the RSU to help this operation by converting the manufacturer-specific data representation into general, device-independent data models.
- V2X Dashboard: to visualize the data of the V2X sub-system for testing, evaluation, and demonstration purposes, we have implemented a web dashboard interface that displays the received CAM and DENM messages and their explicit content (*cf.* Fig. 5)
- Traffic Control Center V2X interfaces: data can be sent and received through the Stream Processing module and also the REST services offered by the TCC implementation. The V2X sub-system can integrate with the dispatch center through both available interfaces.
- Vendor-independent data management framework module: the central component of the V2X sub-system realizes the data management functions of device-independent facilities-layer protocol data models of CAM

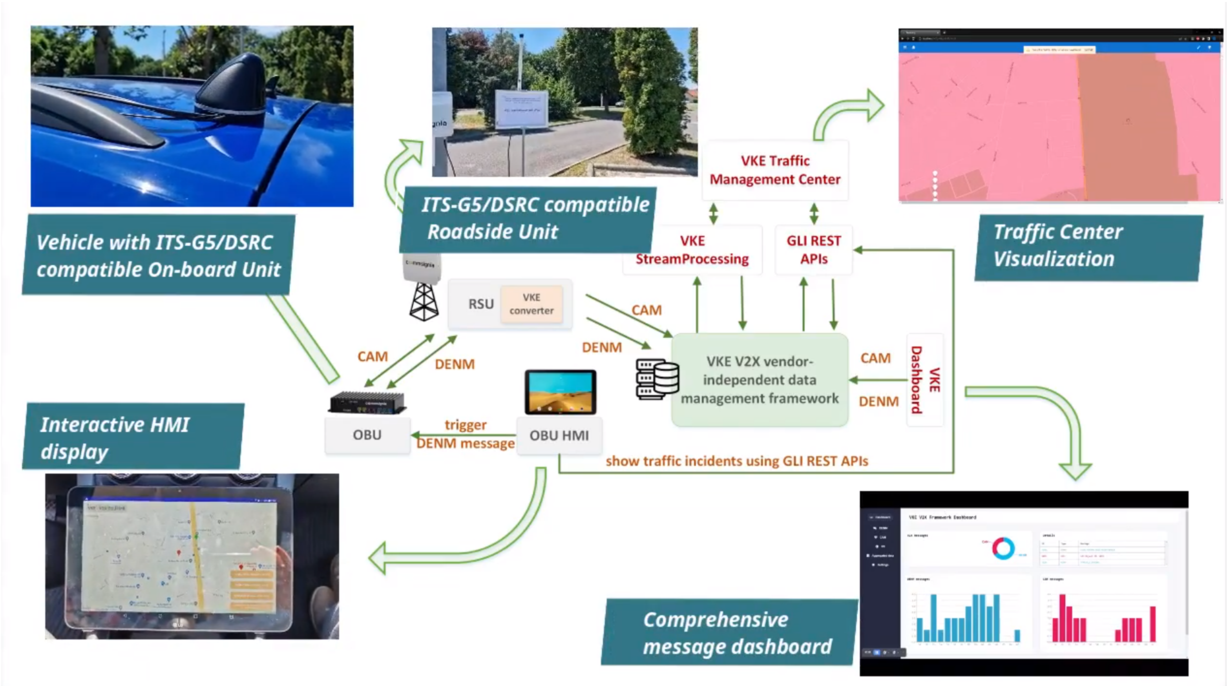

Fig. 5. Proof-of-concept implementation of the V2X sub-system.

and DENM, corresponding to the standards. It implements the device-independent, Apache AVRO scheme-based data models designed for the framework. It also provides the interfaces to related systems so that they can access the device-independent V2X data. Using these interfaces, the converter middleware implemented in the RSU can create data and provide data to other systems through them.

V. Data Analytics

The raw data gathered by different sensors must be unified and pre-processed to make it adaptable for data analytics and other services. The main modules that handle the incoming data flows and make it available for other CityAI modules are the Stream Processing-, Machine Learning-, and Data Lake modules.

A. Stream Processing Module

Sensor data, originating from an array of sources and devices, is transmitted in various formats. The high speed, volume, and diversity of these data types render traditional processing methods, such as batch-based approaches, inefficient, unscalable, and unreliable.

Stream processing [40] offers an innovative technique for the effective extraction and analysis of heterogeneous data. This method perceives data as continuous, never-ending streams and boasts the primary advantage of immediate data processing upon availability. Stream processing alleviates the burden on storage systems by requiring minimal resources through real-time data processing, enabling the extraction of valuable features on the fly without requiring extensive measurement data storage.

Several solutions exist in this context, encompassing message brokers, Pub/Sub services, WebSocket, event-driven architecture, and reactive programming. However, we have chosen Apache Kafka[1] as CityAI's backbone. As a reliable, scalable, and high-throughput message broker, Kafka adeptly manages substantial data stream volumes with minimal latency. Furthermore, Kafka offers robust fault tolerance, message ordering, and real-time data processing capabilities, rendering it an exemplary choice for constructing a complex system such as an intelligent transportation system.

Data from disparate sources are stored in distinct Kafka topics. Kafka Producer applications or Kafka Source connectors write data into these topics, while Kafka Consumer applications or Kafka Sink connectors read data from them. We also employ Kafka Stream applications to execute data pre-processing for the system's other components.

In conclusion, leveraging Apache Kafka has enabled us to develop a high-performance and dependable proof-of-concept system.

B. Machine Learning Module

The main goal of the CityAI Machine Learning module (MLM) is to obtain valuable information from the gathered raw traffic data (originating from the city's traffic sensing infrastructure) with machine learning-based prediction and anomaly detection algorithms [41]. It communicates with the Stream Processing and the Data Lake modules, the former provides the raw real-time traffic data, and it helps to disseminate

[1] https://kafka.apache.org

the information produced by the MLM. The latter has a role in model training and visualization of the traffic data, as it provides the historical data for the MLM.

Fig. 6 shows the components of the Machine Learning module. As mentioned above, due to the architecture of the system, the MLM communicates directly only with the Stream Processing and the Data Lake modules. Based on the timing of the communication between the modules, we can distinguish between real-time and demand-driven communication. The MLM accesses the stored/historical data that is required through the DataLakeInterface. Historical data are used for training and monitoring the prediction models. It is worth mentioning that the data is not needed all the time but is frequently accessed because of its multiple uses. The MLM accesses the required real-time data streams through the StreamProcessingInterface. Real-time data are used for real-time traffic behavior identification, forecasting, and outlier detection. The data is received asynchronously and continuously at varying frequencies, which are subsequently resampled and sent in uniform time units.

The MLM consists of three main blocks:

- *Apache Flink Cluster*, where real-time functionalities are executed. These functionalities are defined as separate Flink jobs, and their current states can be monitored through a Web Dashboard. Our choice of Apache Flink was driven by the unique demands of our use case. It provides an optimal combination of performance, scalability, and compatibility features that best meet our project requirements. Additionally, implementing the cluster using Flink ensures seamless integration with the Apache ecosystem, including Kafka.
- *Monitoring Component*, which offers a platform for tracking prediction tasks and models. This component enables the training of new models or the retraining of existing ones as needed.
- *ModelServing Component*, which oversees prediction tasks, stores trained models, and provides access to them.

By integrating these three components, the MLM can serve the city traffic operators in the traffic management process with several crucial functionalities. As discussed in Section I, the technical complexity of management systems poses a significant barrier for traffic operators who may not have expertise in data science or software engineering. Therefore, it is imperative that this module pre-processes the raw traffic data and complements missing or flawed measurements automatically. It can provide real-time traffic state predictions for different road sections and time horizons, which helps traffic operators plan interventions on time. The traffic state predictions appear in the control center, together with intervention suggestions generated by artificial intelligence-based solutions. However, the final decision about the type and volume of the intervention is made by humans, only the suggestion is generated by AI, which leaves control in the hands of the central authority.

A special type of prediction focuses on traffic congestion. The first phase of the congestion is recognized by anomaly detection algorithms, which are trained to find these specific patterns in the traffic times series. This way, the detection time can be kept low, and the intervention can be made on time, before the initial congestion evolves into a traffic jam, on a wider scale.

The Monitoring Component is crucial to have a constant measurement of the precision of the above-mentioned prediction models. The city infrastructure and, therefore, the traffic patterns change dynamically (*e.g.*, closing/opening lanes, building new roads, maintenance works, and mass events). Thus, if the utilized models are not precise enough, they should be retrained with the novel traffic time series, the module does it automatically. Another useful feature of the module is that it uses public transport data to make predictions more precise. Furthermore, it can provide predictions of the arrival and departure times of public transport vehicles for passengers.

In our proof-of-concept, we validated multiple machine learning algorithms using a comprehensive dataset collected from the city's traffic sensing infrastructure. After comparing their performance on various prediction horizons, we chose XGBoost and SVM as our top selections. For instance, we evaluated the models' performance at short intervals, such as one or two minutes, and longer intervals, such as days. Depending on the SFO input data, the algorithms performed very similarly, with the only discrepancies appearing across different prediction horizons.

For the learning process, we divided the dataset into training and validation subsets, using a 70/30 split. We trained the models on the training set and validated their performance on the validation set. During the training phase, we applied techniques such as cross-validation and hyperparameter tuning to optimize the performance of each algorithm.

We assessed the performance of the selected models using metrics such as mean absolute error (MAE), mean squared error (MSE), and coefficient of determination (R-squared).

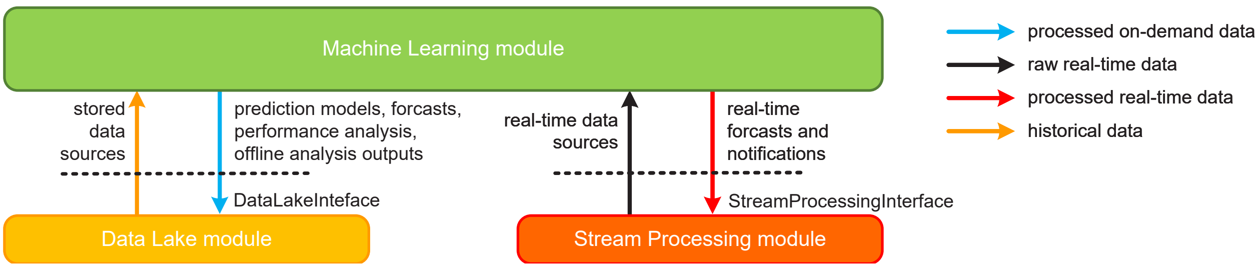

Fig. 6. Relationship of the *Machine Learning* module with other modules.

These metrics helped us to continuously monitor the models' precision through the Monitoring Component, allowing us to retrain them whenever necessary. This ensured that our system remained up-to-date and capable of delivering reliable traffic predictions.

C. Data Lake Module

The main function of our Data Lake [42] component is to consolidate data from multiple sources in a single location, enabling the exploration of complex relationships between data from different sources. It receives data from the corresponding topics in the Stream Processing module, writes them to the Data Lake tables corresponding to the topic, and indexes them. In our prototype implementation, the Machine Learning, Traffic Management, and Dispatch Center components are directly linked for data retrieval, providing a central repository that is accessible to various applications.

Our Data Lake design comprises of two main components: a data warehouse optimized for efficient data insertion and storage and an indexed data warehouse that supports dynamic data retrieval. The former is implemented using Cassandra[2], and the latter uses ElasticSearch[3]. Additional tools such as event monitoring, alerting, and data archival were also integrated into the solution to enhance its functionality.

The primary concern in the design of the Data Lake connectivity was data security and consistency. Write operations are performed indirectly through the Stream Processing component. To achieve this, corresponding Kafka Consumers were developed, which connect to the topics, transform the messages published in these topics, and write them into the database. The data written into the database are subsequently indexed.

We differentiate between two distinct methods of data reading. Read operations are primarily performed by directly connecting to the relevant data store and executing native queries. Currently, there is no abstraction layer or proxy for data querying. Instead, a data warehouse solution is utilized to retrieve specific stored data based on unique identifiers or other predefined (indexed) fields. The second method of querying data is through Kafka, where the desired data can be obtained using appropriate Kafka Source components.

VI. INFORMED TRAFFIC GOVERNANCE AND VISUALIZATION

The section discusses how the different types of data flows and AI-based predictions can be used for interventions and value-added services. Within the CityAI system, we deployed a pilot implementation of reinforcement learning-based traffic light control and introduced a traffic data dissemination service based on V2X and public transport congestion detection service using incremental learning.

A. Reinforcement Learning-based Traffic Light Control

An advantage of reinforcement learning-based traffic light control approaches over conventional signal control techniques, such as traffic theory-based and heuristic methods, is that it does not rely on pre-defined rules but learns the appropriate actions based on the feedback they receive from the observations. The adoption of reinforcement learning for traffic signal control has become very popular recently. Finding the appropriate formulation of states and rewards is crucial to achieving training stability and providing rapid convergence to derive the best policies [43]. Several survey papers were published [44]–[47] that categorize hundreds of research papers in this field. According to the results overviewed in these papers, reinforcement learning has shown superior performance over conventional methods. Although, the performance of these solutions was investigated in a simulated environment that was SUMO traffic simulator in most of the cases. Although some of the papers investigate real-world scenarios, such as [48], to the best of our knowledge, there are no reinforcement learning-based traffic light control approaches implemented in real-life environments.

To make interventions in the traffic, we used reinforcement learning as a goal-oriented machine learning technique, which can learn how to attain a complex objective and maximize along a particular dimension. The approach is concerned with how the agent should take action in the current state of the environment and maximize the overall reward gained. To make the method successful, a lot depends on how the action, the state of the environment, and the reward function are constructed.

The uniqueness of our RL-based traffic control method is that it was deployed in real life as a proof-of-concept solution. Therefore, our hands were tied, and we had to adapt to the existing conditions when defining the environment states reward function, and possible actions. The Hungarian road operator allowed us to run our RL-based scheme in one specific junction in the downtown of Pécs, on the main road that crosses the city. The Rákóczi rd. - Alsómalom rd. junction (Fig. 7) is part of a green-wave traffic control system that significantly limits the allowed signal program changes. Also, the Swarco ACTROS traffic light controller and the traffic management system used by the road operator made it not possible to dynamically create and upload new signal programs in run-time. Instead, only pre-defined and previously uploaded signal programs to the controller can be activated. Moreover, the signal program slots in the controller are also limited. Due to all of these restrictions, the road operator allowed four additional signal programs as modifications of the original one. In these programs, the start time and end time of the green phase were modified by ± 2 seconds, respectively. E.g., in the allowed signal programs, the green phase of signal group $J1$ can be in the 24-78, 24-76, 22-78, 22-80, and 20-80 seconds time ranges, while the cycle time is constantly $Cl = 105$ s. Fig. 7 shows the traffic light program for the first example ($J1$ green phase: 24-78 s). The action of our reinforcement learning-based approach was to select one of the five available traffic light programs. Moreover, the system used by the Hungarian road operator also limited the frequency of program changes to one program change every 15 minutes. Due to all these limitations we had to adapt, we can declare that the allowed traffic light program changes are only enough to fine-tune the current traffic light setup, but not sufficient to

[2]https://cassandra.apache.org
[3]https://www.elastic.co/elasticsearch/

make fundamental changes in the traffic flow.

In a real-life environment, the accuracy of deployed sensors is also limited. Moreover, in the case of traffic cameras, it can vary due to weather conditions and light intensity. In our pilot system, surveillance cameras were used to measure the speed [km/h], road occupancy [%], and flow [vehicle/hour] metrics. The camera observation zones in the controlled intersection are presented in Fig. 7. These PTZ (pan-tilt-zoom) cameras were deployed to ensure security for citizens and not for traffic monitoring purposes. Therefore, the perspective was not ideal, and only the flow values were reliable enough to be used as the state descriptor for the reinforcement learning algorithm. Although, the image processing module was able to distinguish and separately monitor the lanes' traffic, the measured values for lanes having the same direction were merged. The merged observation zones are illustrated with the same color code in Fig. 7. In order to conceal the variations of the measured speed (v) values, moving average with 15 minutes window size was deployed to determine the state (s_t) used as input for the reinforcement learning agent.

$$s_t = \{\overline{v_1}, \overline{v_2}, \overline{v_3}, \overline{v_4}, \overline{v_5}, \overline{v_6}\} \tag{4}$$

The reward function plays a significant role in the learning phase, while during the execution of the learned model, it is used only for monitoring purposes. There are two main methods to learn the model for real-life control: (*i.*) the actions are performed in the real environment, (*ii.*) a simulated environment is used.

In the first case, the determined actions can be more accurate and effective, but on the other hand, the learning phase is quite challenging. The reason is that the performed actions during the learning phase can be random and lead to unwanted situations. Traffic control is very sensitive from this aspect because we can not afford to cause traffic jams during the training phase.

The only option we had was to use a simulated environment to learn the agent and later use it for real-life traffic signal control. Therefore, we modeled not just the specific junction controlled by the reinforcement learning agent but also several neighboring junctions on the main road that crosses the city of Pécs, as illustrated in Fig. 8. Including neighboring junctions in the simulated environment was necessary because our aim was to achieve global improvement in the traffic flow and not just in the controlled intersection. Moreover, we had to take the pre-configured signal programs of other intersections into account in order not to disrupt the green-wave provision. The objective of the reward function was to maximize the average speed of the vehicles in the region of all nine intersections shown in Fig. 8. The simulation of the environment was performed in SUMO [49] by setting up traffic demands based on validated O-D (Origin-Destination) trip tables provided by the road operator.

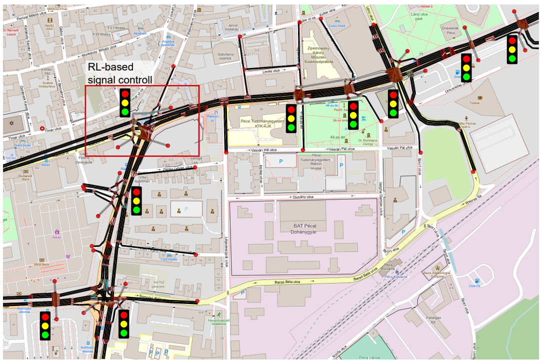

Fig. 8. SUMO simulation of the main road traffic in the center of Pécs.

In order to make the environment compatible with different reinforcement learning-related Python packages, we used the OpenAI Gym framework to create the SUMO-based environment. Although the real-life reinforcement learning-based traffic light control was demonstrated in a single intersection, the implemented environment also supports multiple intersection control using multi-agent reinforcement learning techniques. There are several reinforcement learning algorithm packages available (*e.g.*, KerasRL, Tensorforce, and StableBaselines3), but currently, only RLlib has multi-agent support. Moreover, RLlib is actively maintained, has a large community, and also offers other advanced features, such as hyperparameter optimization and action masking. We tested the performance of different algorithms, such as PPO, A3C, and PG, but the best results were achieved by DQN. The DQN model was trained for 40 simulated days in the SUMO environment. The trained DQN model was deployed in the CityAI domain and

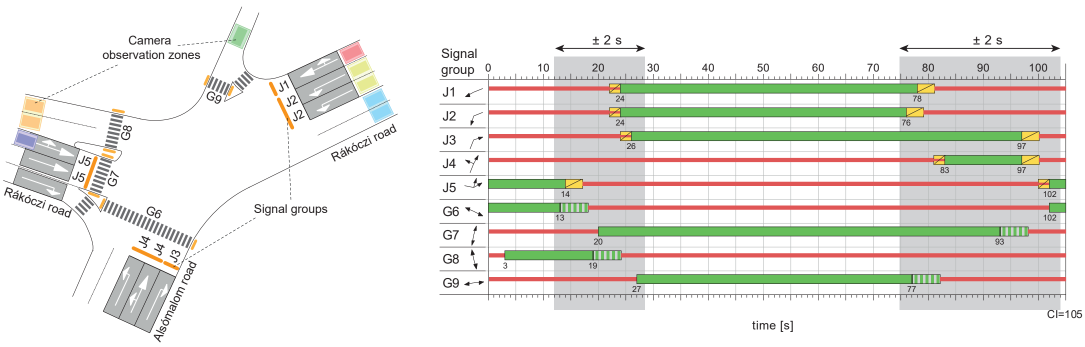

Fig. 7. Topology and signal programs of the RL-controlled junction.

fed with real-life flow [vehicle/hour] values extracted from camera video streams.

The introduced reinforcement learning-based traffic light control (*RL-TLC*) algorithm that uses the trained DQN model, gathers the requested input from the Kafka platform (Stream Processing module) and pushes the proposed traffic light program ID as illustrated in Fig. 9. The proposed program ID is consumed by the Dispatch Center, which confirms the traffic light program automatically or by human operators and pushes its ID back to the Stream Processing module. We used the automatic confirmation setup during real-life experiments. In order to activate the selected signal program, we used the REST API provided by the road operator (MK), which is accessible from their own domain. The new signal program ID data entries were immediately forwarded through the REST API of the road operator to the traffic signal management tool (*JTR-controller*). The *JTR-controller* is responsible for sending the signal program ID to the local intersection traffic signal controller and activating it. The CityAI traffic light control architecture overview and data flows are presented in Fig. 9.

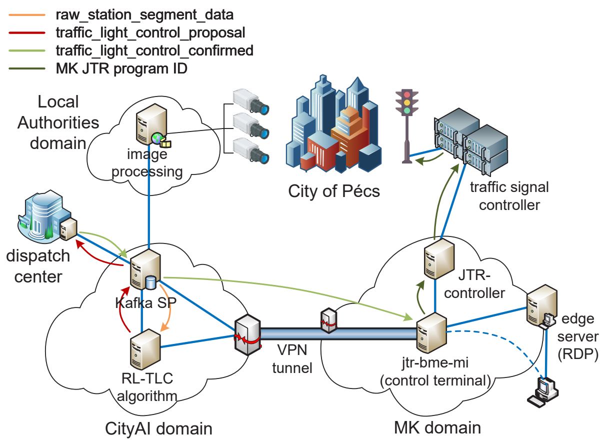

Fig. 9. RL-based traffic light control architecture and data flow.

In cooperation with the Hungarian road operator, we were able to test the presented CityAI reinforcement learning-based traffic light control system in real-life conditions. The proof-of-concept demonstration was running on a regular working day (Monday, 11 Apr 2022) between 6:30 AM and 8:00 PM. In order to compare the performance of the RL-based scheme with the default signal program, we collected the speed [km/h] and flow [vehicle/hour] values from the Kafka platform measured on the demonstration day and one week before (also a Monday). According to the series of flow values, the traffic was very similar on the two examined days. For the speed values, we found that instead of 40.8 km/h average speed, the reinforcement learning-based traffic light control scheme increased the average speed to 44.5 km/h. Although the duration of the real-life test was only one-day long that is not sufficient to justify the performance improvement as a scientific result, we believe that the implemented proof-of-concept solution is very promising, especially if reliable data sources are available and more flexible changes in the traffic signal program are allowed. To the best of our knowledge, this was the first real-life implementation of a reinforcement learning-based signal control scheme.

B. Vendor-independent V2X information collection/dissemination sub-system

The proposed V2X sub-system enables vehicles to communicate with other vehicles, the CityAI infrastructure, and other road users through wireless communication protocols. Besides the highly efficient, timely, and disseminated data collection support, one of the significant capabilities of this sub-system is the ability to implement various intervention types, such as cooperative awareness, and cooperative decision-making.

Cooperative awareness refers to the ability of V2X capable road users to share information about their speed, location, and other relevant data with other road users and infrastructure, thereby improving the overall situational awareness of all parties. This type of intervention allows for the real-time exchange of information between vehicles, such as traffic congestion, roadwork, weather conditions, and any other events that could affect the driver's safety. This information can be used to alert drivers of potential hazards on the road and make informed decisions to avoid collisions and improve traffic flow.

Cooperative decision-making refers to the ability of road users to rely on the information shared through cooperative awareness to make more informed decisions, such as adjusting speed or changing lanes to avoid a potential collision. This intervention also allows vehicles to make decisions that optimize traffic flow, such as forming platoons of vehicles to increase the capacity of highways, enhance the throughput of intersections, and reduce congestion. Additionally, this type of intervention can optimize electric vehicles' energy consumption by allowing them to communicate and coordinate their recharging schedules.

From a Traffic Control Center perspective of our CityAI architecture, the intervention capabilities of the proposed V2X sub-system can provide valuable information and tools for traffic optimization and information dissemination. By utilizing the real-time data exchanged between road users and infrastructure, the TCC can comprehensively understand traffic conditions, allowing it to make more informed decisions to optimize traffic flow and reduce congestion by providing V2X-based traffic and advisory information

One of the critical benefits of V2X technology for the TCC is the ability to use cooperative awareness information to provide real-time traffic updates and alerts to drivers, such as roadwork, accidents, and other events that could affect safety. The TCC can also use the information from cooperative decision-making to make decisions that optimize traffic in the area. The TCC can use V2X technology to control traffic lights, manage lane usage, and adjust speed limits in real time to improve traffic flow and reduce delays. Overall, the intervention capabilities of V2X technology can significantly enhance the ability of CityAI to manage and optimize traffic, improving the overall efficiency and safety of the city-wide transportation system.

We implemented a proof-of-concept testbed containing one vehicle and one roadside unit equipment for the functional assessment of integrating the V2X paradigm and our vendor-independent V2X data management solution into the CityAI framework. The proof-of-concept testbed comprises the components introduced in Fig. 5. In this experimental implementation, the RSU is connected to the central components through cellular backhauling, and the OBU-RSU communication is performed over a standardized ITS-G5 V2X interface. In our testbed, we employed Commsignia OB4 and RS4 devices [50] as OBU and RSU nodes, respectively. The specifications of these devices can be found in Table I. The data received by the vehicle OBU is traversed by a local Wi-Fi network to a tablet, running an Android-based HMI application (further details are depicted in Fig. 5). This HMI developed for the proof-of-concept experiments serves two functions:

- Triggering different types of DENM messages in the OBU: The OBU will forward these messages to the RSU, sending the relevant data to the TCC according to the data path depicted in Fig. 5. The HMI can trigger four types of DENM messages (Roadworks Warning – Major Roadworks, Roadworks Warning – Street Cleaning, Hazardous Location Notification – Animal On The Road, and Emergency Vehicle Approaching).
- Visualization of incident information traversed by the TCC via the RSU and ITS-G5 or using the REST API services directly through 4G/5G cellular and displayed for the driver using Google Maps API. Information retrieval is DENM-based in the case of V2X access, while for cellular communications, it is triggered by a periodic query with a parameterizable interval (in the demo scenario, an interval of 5 seconds was set) or by an event-based solution that activates when the vehicle moves (detected by the GPS module built into the HMI tablet).

TABLE I
MAIN TECHNICAL SPECIFICATIONS OF THE USED OBU/RSU DEVICES.

Feature	Specification
CPU	800MHz Freescale
OS	Linux
RAM	2GB DDR3 SDRAM
Flash	4GB eMMc
Antenna	2xV2X, 2xWiFi, 2xLTE/3G, 1xGNSS
Data	1xETH, 2xUSB, 1xCAN, 1xOBD-II
V2X chipset	Autotalks Secton
Hardware Security Module	SLI97
Further references	OB4 [51] / RS4 [52]

The proof-of-concept validation was performed in the city of Pécs, specifically on Road 58, Siklósi út. It is an extensive 2×2 lane road that stretches from the city center of Pécs all the way to the M60 motorway, primarily heading in a southerly direction. It features a few significant curves, roundabouts, and side road branches with traffic lights interrupting the flow of traffic. The road connects six neighborhoods with the city center. Shortly after crossing the Pécs city limits, it connects to the M60 motorway. The location of the test was near the city center section of the road, in close proximity to the public cemetery.

Fig. 10 shows the graphical interface of the HMI. The button responsible for triggering the four DENM messages is located in the lower right part of the screen. The accident/traffic information received from the TCC is displayed using a Google Maps marker, the title of which is the type of event. Additional information can be assigned to the markers on the map. In the example of Fig. 10, we experimented with the V2X-based intervention: the TCC center gathered and disseminated the event through the ITS-G5 interface of the RSU. The OBU received the information, and the HMI presented the marker at the event location together with the source of the given incident (i.e., in this case the TCC as a dispatcher). We can also further differentiate the markers: through the REST API service, we also receive the information that the confirmation of the event is POSSIBLE or VERIFIED for each event. In the first case, the notification represents a possible event, while in the second value, the report shows a confirmed, definitely existing event.

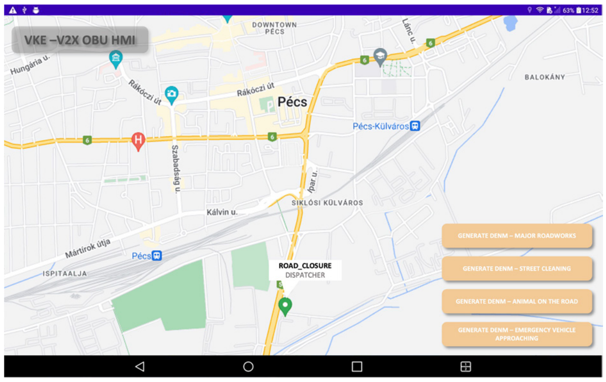

Fig. 10. Example screenshot from the Android-based HMI of the V2 subsystem's proof-of-concept testbed.

In the intervention scenarios of our proof-of-concept experiments, the traffic incident information is generated in the TCC and then transmitted to the vehicle. This information is then visualized on the vehicle's HMI, allowing the driver to see details of the accident, such as location and severity. This allows the driver to make informed decisions about the best route to take, avoiding any potential hazards or delays caused by the incident. It also allows the driver to be better prepared in case they encounter the accident scene on their route.

C. Public Transport Congestion Detection

The standard of living in metropolitan areas is heavily dependent on the ability of transportation systems to move residents, workers, and goods between various locations. However, as urbanization continues to expand worldwide, cities are experiencing a rise in population density. This leads to an increase in the number of vehicles on the road, exacerbating the issue of traffic congestion. This congestion not only hinders economic productivity, but also harms the environment and public safety through increased fuel consumption, air pollution, and increased costs of goods and services. Because of this, the ability to quickly and accurately detect and predict traffic congestion is a crucial task.

Given the importance of addressing traffic congestion in public transportation, we explore the use of incremental learning (IL) [53]–[56] for real-time detection of congestion. Specifically, we investigate the potential of using IL to adapt and scale the detection of congestion in public transport. To achieve this, we utilize long short-term memory (LSTM) in combination with IL to predict short-term bus travel speed by capturing the long-term temporal dependency.

In our experimental implementation, the model is trained in a continuous loop. The data collected from the buses' trajectories is stored in a database and used for training. The data is pre-processed and then separated into sequences. Depending on the number of routes being monitored, the appropriate LSTM model is either trained or updated to handle the trajectories. If the number of inputs and outputs remains unchanged, the LSTM model remains unchanged. Once a new model is trained, it is evaluated against the previous model stored in the database using test data. If the new model shows better performance, it is stored as the current model. If not, it is discarded.

We tested the effectiveness of our solution in Pécs, Hungary. The results showed that our incrementally updated model was able to detect congestion with an accuracy of up to 82.37%. Additionally, we found that the model's accuracy in estimating travel speed can increase up to 221.46% within just six days, demonstrating its versatility. Additionally, resource consumption was found to be similar to traditional learning methods, making it a promising option for use in resource-constrained environments. For a detailed analysis of the achieved results, we refer the reader to discuss [57].

In summary, our solution is adaptive, scalable, and able to operate in real time. It can improve the efficiency of public transportation by providing a more accurate basis for congestion detection while reducing resource consumption. Additionally, as the model is incrementally updated, it can adapt and evolve in tandem with the streaming data, resulting in faster convergence.

D. Scalability Considerations and Performance Evaluation

During the development of the CityAI framework, several key architectural decisions and system components were specifically chosen to ensure the system's ability to handle the demands of a growing city infrastructure. Below, we provide an analysis of these aspects, highlighting their contributions to the overall scalability of CityAI.

1) Sensor Hub and Data Collection Modules: The Sensor Hub module is designed with scalability in mind, incorporating a modular architecture that allows for the easy addition of new sensor types and data sources. Each sensor hub operates independently, ensuring that data collection can be expanded by simply deploying additional hubs across the city. This modular approach not only facilitates scalability in terms of the number of sensors but also in terms of geographic coverage. As more regions of a city are equipped with sensors, the system can absorb and integrate this data without requiring significant modifications to the underlying architecture.

2) Stream Processing with Apache Kafka: The CityAI framework leverages Apache Kafka as the backbone of its stream processing architecture. Kafka's high throughput, low latency, and distributed nature make it inherently scalable and capable of handling millions of events per second. This ensures that as the volume of traffic data, sensor inputs, and vehicular communications increase, the system can continue to process and analyze this data in real time without degradation in performance. Kafka's ability to seamlessly integrate with other distributed data processing frameworks further enhances the system's scalability, allowing for the dynamic addition of new data sources or the expansion of processing nodes as required by the growing demands of a city.

3) Data Analytics and Machine Learning Modules: The Data Analytics component, particularly the MLM, is built to scale with increasing data volumes. The use of distributed processing frameworks like Apache Flink allows the system to handle large-scale data streams in real time, making it adaptable to the growing complexity of urban traffic patterns. As the data volume grows, additional processing nodes can be deployed to maintain performance, ensuring that the system's predictive models remain accurate and responsive.

Moreover, the MLM's architecture supports the incremental retraining of models, allowing it to adapt to changing traffic conditions without the need for extensive computational resources. This ability to update models on the fly, combined with distributed processing, ensures that the system can scale efficiently as the city's traffic infrastructure evolves.

4) Informed Traffic Governance and V2X Integration: CityAI's traffic governance framework is designed to be adaptive and responsive, which is critical for scalability. The integration of V2X communication technology enables the system to manage an increasing number of connected vehicles and infrastructure elements. By ensuring that the V2X communication sub-system operates independently of specific hardware vendors, the system can scale to accommodate new vehicles and roadside units as they are deployed, without requiring significant reconfiguration.

The reinforcement learning-based traffic light control system is another testament to the system's scalability. By using a flexible, data-driven approach to traffic management, CityAI can dynamically adjust to the increasing complexity of urban traffic without being constrained by rigid, pre-defined rules.

5) Data Lake and Long-Term Storage: The Data Lake module, built on scalable technologies like Cassandra and Elasticsearch, ensures that the system can manage vast amounts of historical and real-time data. As the volume of data grows, these technologies allow for horizontal scaling, meaning that additional storage nodes can be added to accommodate more data without impacting performance. This scalability is crucial for supporting long-term traffic analysis, anomaly detection, and strategic planning in increasingly complex urban environments.

VII. CONCLUSION

Considering the trends in urbanization and sustainability, there is an urgent need for efficient ITS services that aim to improve mobility and make transportation easier, faster, and more reliable. The proposed CityAI system aims to address these challenges by providing an automated, artificial intelligence-based solution for managing city traffic. This paper introduces key modules' capabilities, architecture, and functions of our proof-of-concept ITS system deployed in the city of Pécs, Hungary. The framework includes state-of-the-art technologies, techniques, and applications but also considers real-world applicability. Since the driving force of all ITS solutions is data, we implemented different data collection modules, such as image processing for extracting vehicle traffic information from roadside cameras, gathering public bus trajectories, and V2X information. To handle the continuous flow of data streams, a Stream Processing module was used in order to make the pre-processed data available for machine learning-based traffic prediction, anomaly detection algorithms, and adaptive traffic control. Although many papers investigate machine learning-based traffic light control in simulated environments, our pilot implementation is the first published reinforcement learning-based traffic light controller that was running in a real-life environment as part of the CityAI system. We also prepared the system to use V2X data to inform drivers about the best route to take, avoiding any potential hazards or delays caused by incidents. Moreover, a public transport congestion detection service was implemented to inform customers and operators about future delays. We can witness that machine learning can be widely used to resolve various issues in modern ITS. A complex system, such as the introduced CityAI system, is required to ensure reliable data gathering, processing, control, and information services to make the concept work in practice.

ACKNOWLEDGEMENTS

This work was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences. Supported by the ÚNKP-23-5-BME-461 New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund. The work presented in this paper was supported by project no. TKP2021-NVA-02. Project no. TKP2021-NVA-02 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021-NVA funding scheme. The authors thank colleagues for their cooperation in the related project activities and research work, especially Norbert Varga, Bernáth Wiandt, and Attila Nagy, who were integral to the initial implementation design.

REFERENCES

[1] J. Lu *et al.*, "Expansion of city scale, traffic modes, traffic congestion, and air pollution," *Cities*, vol. 108, p. 102 974, 2021.

[2] Y. Himeur *et al.*, "Next-generation energy systems for sustainable smart cities: Roles of transfer learning," *Sustainable Cities and Society*, p. 104 059, 2022.

[3] L. Liu and Y. Zhang, "Smart environment design planning for smart city based on deep learning," *Sustainable Energy Technologies and Assessments*, vol. 47, p. 101 425, 2021.

[4] T. S. Gomides *et al.*, "An adaptive and distributed traffic management system using vehicular ad-hoc networks," *Computer Communications*, vol. 159, pp. 317–330, 2020.

[5] J. Wu *et al.*, "Emergency vehicle lane pre-clearing: From microscopic cooperation to routing decision making," *Transportation research part B: methodological*, vol. 141, pp. 223–239, 2020.

[6] S. Chattoraj *et al.*, "Design and implementation of low cost electronic toll collection system in india," in *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, IEEE, 2017, pp. 1–4.

[7] S. Huang *et al.*, "A decentralized etc architecture based on blockchain technology," *Journal of Advanced Transportation*, vol. 2021, 2021.

[8] A. M. Nagy and V. Simon, "Survey on traffic prediction in smart cities," *Pervasive and Mobile Computing*, vol. 50, pp. 148–163, 2018.

[9] K. Lee *et al.*, "Short-term traffic prediction with deep neural networks: A survey," IEEE Access, vol. 9, pp. 54 739–54 756, 2021.

[10] N. Bereczki and V. Simon, "Machine Learning Use-Cases in C-ITS Applications," *Infocommunications Journal*, vol. 15, no. 1, pp. 26–43, 2023.

[11] Alibaba Cloud, *City Brain Now in 23 Cities in Asia*, https://www.alibabacloud.com/blog/city-brain-now-in-23-cities-in-asia_595479, 2019. (visited on 09/08/2024).

[12] Yunex Traffic, *Yunex traffic – intelligent traffic solutions*, https://www.yunextraffic.com, 2024. (visited on 09/08/2024).

[13] SWARCO, *Urban Traffic Management (UTM) Solutions*, https://www.swarco.com/products/software/urban-traffic-management, 2024. (visited on 09/08/2024).

[14] H. Hejazi and L. Bokor, "V2X-Equipped Smart Intersections–Survey of Surveys Use Cases and Deployments," *Journal of Communications*, vol. 18, no. 11, pp. 722–738, 2023. **DOI**: 10.12720/jcm.18.11.722-738.

[15] L. Bréhon-Grataloup *et al.*, "Mobile edge computing for V2X architectures and applications: A survey," *Computer Networks*, vol. 206, p. 108 797, 2022.

[16] Y. Qi and S. Ishak, "A hidden markov model for short term prediction of traffic conditions on freeways," *Transportation Research Part C: Emerging Technologies*, vol. 43, 2014. **DOI**: 10.1016/j.trc.2014.02.007.

[17] R. More *et al.*, "Road traffic prediction and congestion control using artificial neural networks," in *2016 International Conference on Computing, Analytics and Security Trends (CAST)*, 2016, pp. 52–57. **DOI**: 10.1109/CAST.2016.7914939.

[18] T. S. Tamir *et al.*, "Traffic congestion prediction using decision tree, logistic regression and neural networks," *IFAC-PapersOnLine*, vol. 53, no. 5, pp. 512–517, 2020, 3rd IFAC Workshop on Cyber-Physical & Human Systems CPHS 2020. **DOI**: 10.1016/j.ifacol.2021.04.138.

[19] J. Kim and G. Wang, "Diagnosis and prediction of traffic congestion on urban road networks using bayesian networks," *Transportation Research Record*, vol. 2595, no. 1, pp. 108–118, 2016. **DOI**: 10.3141/2595-12. eprint: https://doi.org/10.3141/2595-12.

[20] X. Ma *et al.*, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015. **DOI**: 10.1016/j.trc.2015.03.014.

[21] W. Xiangxue *et al.*, "Data-driven short-term forecasting for urban road network traffic based on data processing and lstm-rnn," *Arabian Journal for Science and Engineering*, vol. 44, pp. 3043–3060, 2019.

[22] J. An *et al.*, "A novel fuzzy-based convolutional neural network method to traffic flow prediction with uncertain traffic accident information," *IEEE Access*, vol. 7, pp. 20 708–20 722, 2019. **DOI**: 10.1109/ACCESS.2019.2896913.

[23] Y. Cong *et al.*, "Traffic flow forecasting by a least squares support vector machine with a fruit fly optimization algorithm," *Procedia Engineering*, vol. 137, pp. 59–68, 2016, Green Intelligent Transportation System and Safety. **DOI**: 10.1016/j.proeng.2016.01.234.

[24] J. Tang *et al.*, "An improved fuzzy neural network for traffic speed prediction considering periodic characteristic," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 9, pp. 2340–2350, 2017. **DOI**: 10.1109/TITS.2016.2643005.

[25] F. Moretti *et al.*, "Urban traffic flow forecasting through statistical and neural network bagging ensemble hybrid modeling," *Neurocomput.*, vol. 167, no. C, pp. 3–7, 2015. **DOI**: 10.1016/j.neucom.2014.08.100.

[26] D. Chen, "Research on traffic flow prediction in the big data environment based on the improved rbf neural network," *IEEE Transactions on Industrial Informatics,* vol. 13, no. 4, pp. 2000–2008, 2017. **DOI**: 10.1109/TII.2017.2682855.

[27] T. Bandaragoda *et al.*, "Trajectory clustering of road traffic in urban environments using incremental machine learning in combination with hyperdimensional computing," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 1664–1670. **DOI**: 10.1109/ITSC.2019.8917320.

[28] D. Nallaperuma *et al.*, "Online incremental machine learning platform for big data-driven smart traffic management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 12, pp. 4679–4690, 2019. DOI: 10.1109/TITS.2019.2924883.
[29] J. I. Engel *et al.*, "A low-complexity vision-based system for real-time traffic monitoring," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1279–1288, 2016.
[30] J. Redmon *et al.*, "You only look once: Unified, real-time object detection," in *2016 IEEE Conf. on Comp. Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.
[31] J. Redmon and A. Farhadi, Yolov3: *An incremental improvement*, 2018. DOI: 10.48550/ARXIV.1804.02767.
[32] D. Barath and L. Hajder, "A theory of point-wise homography estimation," *Pattern Recognition Letters*, vol. 94, pp. 7–14, 2017.
[33] V. T. Arasan and G. Dhivya, "Measurement of occupancy of heterogeneous traffic using simulation technique," *IFAC Proceedings Volumes*, vol. 42, no. 15, pp. 19–24, 2009.
[34] F. L. Hall, "Traffic stream characteristics," *Traffic Flow Theory*. US Federal Highway Administration, vol. 36, 1996.
[35] S. Abdelhamid *et al.*, "Vehicle as a mobile sensor," *Procedia Computer Science*, vol. 34, pp. 286–295, 2014, The 9th International Conference on Future Networks and Communications (FNC'14)/The 11th International Conference on Mobile Systems and Pervasive Computing (MobiSPC'14)/Affiliated Workshops. DOI: 10.1016/j.procs.2014.07.025.
[36] J. Guerrero-Ibáñez *et al.*, "Sensor technologies for intelligent transportation systems," *Sensors*, vol. 18, no. 4, 2018. DOI: 10.3390/s18041212.
[37] S. Kuutti *et al.*, "A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 829–846, 2018. DOI: 10.1109/JIOT.2018.2812300.
[38] Z. Xiao *et al.*, "Trajdata: On vehicle trajectory collection with commodity plug-and-play obu devices," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 9066–9079, 2020. DOI: 10.1109/JIOT.2020.3001566.
[39] L. Li *et al.*, "Trajectory data-based traffic flow studies: A revisit," *Transportation Research Part C: Emerging Technologies*, vol. 114, pp. 225–240, 2020. DOI: 10.1016/j.trc.2020.02.016.
[40] H. Isah *et al.*, "A survey of distributed data stream processing frameworks," *IEEE Access*, vol. 7, pp. 154 300–154 316, 2019. DOI: 10.1109/ACCESS.2019.2946884.
[41] A. Nagy and V. Simon, "Traffic congestion propagation identification method in smart cities," *Infocommunications Journal*, vol. 13, pp. 45–57, 2021. DOI: 10.36244/ICJ.2021.1.6.
[42] P. Sawadogo and J. Darmont, "On data lake architectures and metadata management," *Journal of Intelligent Information Systems*, vol. 56, no. 1, pp. 97–120, 2021. DOI: 10.1007/s10844-020-00608-7.
[43] S. Bouktif *et al.*, "Deep reinforcement learning for traffic signal control with consistent state and reward design approach," *Knowledge-Based Systems*, vol. 267, p. 110 440, 2023. DOI: 10.1016/j.knosys.2023.110440.
[44] A. L. Bazzan and F. Klügl, "A review on agent-based technology for traffic and transportation," *The Knowledge Engineering Review*, vol. 29, no. 3, pp. 375–403, 2014.
[45] P. Mannion *et al.*, "An experimental review of reinforcement learning algorithms for adaptive traffic signal control," *Autonomic road transport support systems*, pp. 47–66, 2016.
[46] H. Wei *et al.*, "Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation," *ACM SIGKDD Explorations Newsletter*, vol. 22, no. 2, pp. 12–18, 2021.
[47] M. Noaeen *et al.*, "Reinforcement learning in urban network traffic signal control: A systematic literature review," *Expert Systems with Applications*, p. 116 830, 2022.
[48] L. N. Alegre *et al.*, "Using reinforcement learning to control traffic signals in a real-world scenario: An approach based on linear func- tion approximation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 9126–9135, 2022. DOI: 10.1109/TITS.2021.3091014.
[49] P. A. Lopez *et al.*, "Microscopic traffic simulation using sumo," in *2018 21st international conference on intelligent transportation systems (ITSC)*, IEEE, 2018, pp. 2575–2582.
[50] Commsignia Ltd., *Official company website*, https://www.commsignia.com, 2023. (visited on 04/19/2023).
[51] Commsignia Ltd., *ITS-OB4 - Product brief*, 2020.
[52] Commsignia Ltd., *ITS-RS4 - Product brief*, 2020.
[53] Z. Chen and B. Liu, "Lifelong machine learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning,* vol. 12, no. 3, pp. 1–207, 2018.
[54] S. S. Sarwar *et al.*, "Incremental learning in deep convolutional neural networks using partial network sharing," *IEEE Access*, vol. 8, pp. 4615–4628, 2020. DOI: 10.1109/ACCESS.2019.2963056.
[55] Y. Wu *et al.*, "Large scale incremental learning," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 374–382. DOI: 10.1109/CVPR.2019.00046.
[56] Y. Luo *et al.*, "An appraisal of incremental learning methods," *Entropy*, vol. 22, no. 11, 2020. DOI: 10.3390/e22111190.
[57] L. A. Makara *et al.*, "Public transport congestion detection using incremental learning," *Pervasive and Mobile Computing*, vol. 91, p. 101 769, 2023. DOI: 10.1016/j.pmcj.2023.101769.

Árpád Huszák received Ph.D. degree in electrical and computer engineering from the Budapest University of Technology and Economics (BME) in 2010. Currently, he is an associate professor with the Department of Networked Systems and Services. Previously, he also joined the Mobile Innovation Center Hungary until 2010. He has done research on mobility management and 3D video streaming, recently his research interests focus on network communications, vehicular networks and machine learning.

Vilmos Simon received his Ph.D. from the Budapest University of Technology and Economics (BME) in 2009. Currently, he is an Associate Professor at the Department of Networked Systems and Services and Head of the Multimedia Networks and Services Laboratory. He has done research on mobility management, energy efficiency in mobile cellular systems, and self-organized mobile networks, recently his research interests include machine learning and data analytics for smart cities and intelligent transportation management systems.

László Bokor received his Ph.D. degree in computer engineering from Budapest University of Technology and Economics in 2014. He is currently an associate professor at the Dept. of Networked Systems and Services. His research interest focuses on V2X communications in Intelligent Transportation Systems. He is a member of several professional organizations, such as the IEEE ITS Society, the Hungarian Standards Institution's Technical Committee for ITS, and the ITS Hungary Association.

László Tizedes received an M.Sc. degree in information technology from the University of Veszprem in 2001. He is a research engineer at the Machine Perception Laboratory, SZTAKI Institute for Computer Science and Control, ELKH Eötvös Loránd Research Network. Previously, he worked as a research and developer engineer at Tateyama Ltd until 2016. Main research areas: image/video processing, biometric image processing, special panoramic optics, hyperspectral image analysis, application development, and coding.

Adrian Pekar received the Ph.D. degree in computer science from the Technical University of Košice, Slovakia, in 2014. Currently, he is a Senior Researcher with the Department of Networked Systems and Services, Budapest University of Technology and Economics. Prior to this, he held research, teaching, and engineering positions in Slovakia and New Zealand. His research interests include network and services management, software-defined networking, network function virtualization, and cloud computing.