Part III

Blockchain Solutions for Trusted Edge Intelligence in IoT Systems

11

Decentralized Strategy for Artificial Intelligence in Distributed IoT Ecosystems: Federation in ASSIST-IoT

Eduardo Garro[1], Ignacio Lacalle[2], Karolina Bogacka[3,4], Anastasiya Danilenka[3,4], Katarzyna Wasielewska-Michniewska[3], Charalambos Tassakos[5], Anastasia Theodouli[6], Anastasia Kassiani Blitsi[6], Konstantinos Votis[6], Dimitrios Tzovaras[6], Marcin Paprzycki[3], and Carlos E. Palau[2]

[1]Prodevelop S. L., Spain
[2]Communications Department of Universitat PolitÃĺcnica de València, Spain
[3]Systems Research Institute, Polish Academy of Sciences, Poland
[4]Warsaw University of Technology, Poland
[5]TwoTronics GmbH, Germany
[6]The Centre for Research & Technology, Greece

Abstract

Decentralization of the IoT ecosystems poses several challenges whenever AI is applied in a shared fashion. Diverse locations, alongside privacy concerns, require the use of holistic strategies, where various environments effectively collaborate while avoiding data disclosure. In this context, this chapter proposes a use case to demonstrate the appropriateness of the solution brought by the ASSIST-IoT project. Specifically, multiple geographic and computing locations, which are close to the automotive surface defects detection scanners, work together to improve AI outcomes, scaling those to a large fleet of vehicles.

keywords: Federated learning, Internet of Things, decentralization, edge-cloud continuum, surface defects detection.

11.1 Introduction

The current transition from cloud-like centralized datacenters to more decentralized systems, where geographically dispersed edge devices live, fosters an unprecedented paradigm shift with disruptive effects in the convergence between the physical and digital world. Here, orchestrating intelligence (AI) promises to be a key driver for enabling low-latency applications with high reliability in multitude of use cases (e.g., automotive, industrial automation, personalized health, etc.). In addition, moving intelligence closer to the edge, relaxing the dependence from a central location could contribute to bandwidth savings, and energy-efficiency and help to preserve data security/privacy [14].

The previous is aligned with reference to European entities in the field. First, the European Strategy for Data includes at its heart the need for decentralization to ensure flexibility and agility in matching demand/supply, and responsiveness while reducing resource consumption through flexible federation and a "fair business offer" [3]. Besides, according to the Alliance for the Internet of Things Innovation (AIOTI) roadmap [7], the next release (v6) of its high-level architecture will focus on artificial intelligence and machine learning (AI) for the next-generation IoT systems (NG-IoT). The success of using AI/ML to solve NG-IoT problems will highly depend on the quality and quantity of available training data. However, while traditional ML approaches typically rely on the central management of training data, such an approach does not seem to be feasible or practical in the next era of IoT. The reasons for this are, on the one hand, data privacy and regulatory compliance and, on the other hand, technical burdens associated with the growing amount of data to be collected and transferred to "a central location." In this context, decentralized AI solutions are needed.

11.1.1 Decentralized AI

The term distributed intelligence has at least two meanings: (a) **collective intelligence** and (b) **decentralized AI**.

The main mechanisms of the collective intelligence are: (a) *cognition* in terms of sensing, (b) *cooperation* as multiple (semi-) autonomous entities exchanging data to jointly establish what needs to be done, and (c) *coordination*, conceptualized as a mechanism crucial for the realization of workflows, where specific actions depend on the results of other actions. If those mechanisms are understood in the most convenient way, it is not very difficult to envision scenarios, in which collective intelligence can be claimed to materialize within NG-IoT ecosystems.

On the other hand, decentralized AI (sometimes also called distributed AI) is a subfield of AI research, dedicated to the development of distributed solutions for problems. It is often seen as a predecessor to the research devoted to software agents and (multi-)agent systems. Still, within the scope of this book, we referred this action to *distributed problem solving*. The main idea is that, for example, completing the training of neural networks with (very) large datasets would require on a single node a substantial amount of time (hours, days, or even weeks) and resources, whereas if multiple computing nodes are tightly coupled, the "training work" can be divided among them, leading to more efficient use of resources in lower operational time. In this way, distributed AI is somehow related to parallel computing. Here, it is important to realize that the most common parallel computing methods and approaches have been designed for a single stakeholder (i.e., a single user, or a company), being the sole owner of all of the data used for model training. However, it has to be realized that, for the past few years, the situation has been rapidly evolving. Among others, the following trends brought about the changes:

- Proliferation of powerful handheld devices with multiple sensors, which generate streams of data that users may want to control.
- Fast drop of price and size of sensors (and actuators), which can be placed "everywhere" and can belong to "anybody."
- Availability of small and inexpensive processors designed for machine learning (e.g., NVIDIA Jetson Nano series devices), which can be placed in almost any location within the IoT ecosystem.
- Increase in the number of wireless networks with high bandwidth and range, which are used to establish communication channels between sensors, actuators, edge devices, computing nodes, gateways, cloud(s), etc.
- Progress in research, development, and deployment of the IoT ecosystems, in almost all areas of day-to-day activities.
- Advances in methods, and their implementations, that can be used in various ML scenarios.

As a result, the vision of a single owner of data, which is stored in a centralized location and used to train model(s) to realize its own (individual) goals, starts to be supplanted by approaches that can facilitate coopetition. Here, coopetition is understood as a scenario where multiple entities (e.g., data owners) compete in one context (e.g., as producers of medicines) and cooperate in another, e.g., as providers of knowledge for the development of shared machine learning models. Notably, this implies certain orchestration and harmonization workload that must be performed among topologically– and likely geographically– disperse devices, therefore becoming larger than single-node parallelization.

11.1.2 Federated learning

Federated learning (FL) [10], [12] is one of the most recent developments in the area of decentralized AI. FL is an approach to train AI/ML models involving multiple datasets stored in "local nodes." In other words, in FL, a shared (global) model is trained collaboratively by multiple parties, which protect their (private) training datasets. After each "round" of local training, the model parameters are "combined into the central model." After the update is completed, the updated central model is redistributed and used either in the training or in the inference processes. Typically, the updated version of the global model is sent back to the nodes that participated in the training. However, there exist FL scenarios, in which "new nodes" participate in each training round (see, for instance, [11]). The training process is completed, when the common model meets specific stopping criteria. Here, it should be noted that while the typical training of a neural network is reported, FL is model-independent; i.e., any model that can be trained on local data and updated centrally can be used.

It should be noticed that the notion of parties participating in FL training might refer to a wide spectrum of possibilities; starting from small edge devices, cameras, or mobile phones, up to enterprise-scale data centers located in different countries or even different companies and organizations. With that scope in mind, ASSIST-IoT project [1] moves forward in this decentralized AI direction, by providing an FL infrastructure to be used to instantiate FL in future NG-IoT systems. This infrastructure is under construction and is being deployed in a real-life industrial scenario. This chapter presents the ASSIST-IoT FL system in detail, in the context of a specific use case of the project focused on automotive sector. Here, the deployment will realize an FL-based surface defect detection, applied without compromising the data privacy of a large fleet of vehicles that pass through the scanners in their individual locations.

The remainder of the chapter is organized as follows. Section 11.2 introduces the different concepts of federated learning, while section 11.3 presents the adopted ASSIST-IoT FL architecture, detailing the different enablers designed and implemented within the scope of the project. Next, Section 11.4 presents the specific case study and the current deployment situation. Finally, conclusions are drawn in Section 11.5.

11.2 Federated Learning Principles

In order to successfully design the appropriate ASSIST-IoT FL system, the project has followed the FL taxonomy identified in [9] that relies on five main aspects:

- **Communication architecture:** While in a centralized design the parameter updates on the global model are always done in a central manager, also called aggregator or collector, in a decentralized design, there is not a single point of truth (there is no manager element). The most commonly known example of a centralized FL architecture is the Google Keyboard - Gboard for mobile keyboard predictions [8].
- **Scale of federation:** The FL systems can be categorized into two typical types by the scale of federation: *cross-silo* and *cross-device*. The differences between them lie in the number of parties and the amount of data stored in each party. In *cross-silo*, the parties can be either independent organizations or independent data centers of a single organization. In *cross-device*, on the contrary, the number of parties is relatively large and each party has a relatively small amount of data as well as computational power, the parties usually being IoT devices.
- **Data partitioning:** FL systems are also categorized in horizontal or vertical data partitioning based on how data are distributed over the sample and feature spaces. In *horizontal data partitioning*, the datasets of different parties have the same feature space but little intersection on the sample space, so that the parties can train the local models using their local data with the same model architecture. In *vertical FL*, the datasets of different parties have the same sample space but differ in the feature space.
- **ML model:** Since FL is used to solve ML problems, the parties usually want to train state-of-the-art ML models. The most popular ML models are *neural networks (NN)*, which achieve state-of-the-art results in many AI tasks, like image classification and word prediction; *decision trees*, which are highly efficient to train and easy to interpret compared with NNs; and *linear models* (e.g., linear regression, logistic regression, and

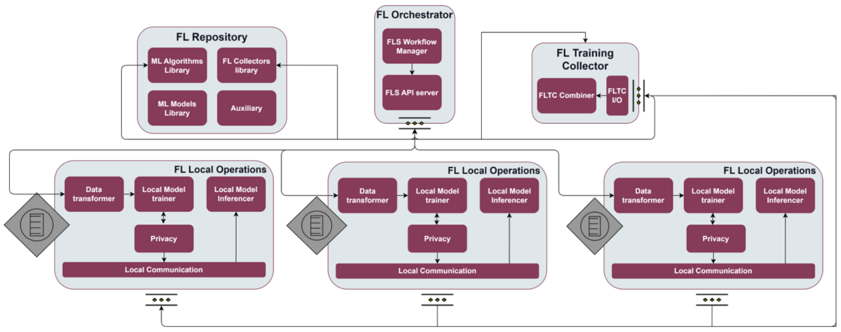


Figure 11.1 ASSIST-IoT FL system formed by four enablers.

support vector machines), which are well-known and easy-to-use ML models.

- **Privacy mechanism:** Although, ideally, local data is not expected to be exposed in FL, the exchanged model parameters may still leak sensitive information about the data. The most well-known privacy mechanisms include *cryptographic methods* or *differential privacy*.

11.3 Federated Learning System of ASSIST-IoT Project

According to the previously described categorization, and feature implementation options, the proposed ASSIST-IoT FL system for the automotive pilot uses the following configuration:

- Communication architecture: Centralized
- Scale of federation: Cross-device
- Data partitioning: Horizontal
- ML model: Neural Network
- Privacy mechanism: Differential privacy

The proposed ASSIST-IoT FL system block diagram and flow chart are shown in Figure 11.1. As it can be seen, four main functional blocks can be distinguished. These functional blocks are named **enablers** and are used as an abstraction term in the project acting as the cornerstone elements of the ASSIST-IoT architecture. In essence, an enabler is a collection of software *components* – running on nodes – that work together to deliver a specific functionality of a system, that is, ASSIST-IoT enablers are not atomic but presented as a set of interconnected components. It should be noticed that multiple enablers may be used in a system to deliver a more complex *service*, leveraging features of the involved enablers. Additionally, one of the most important design principles that distinguish *components* from *enablers* is that the components from different enablers cannot directly communicate unless a RESTful API endpoint has been explicitly developed for that purpose.

Regarding the regular call flow in this particular deployment, it starts with the model training. To do so, a proper training job configuration is submitted to FL orchestrator that propagates it to FL training collector and candidate FL local operations to execute the job. Then, FL training collector collaborates with FL local operations to finally obtain new global model aggregated from successive local updates. To support the process, FL repository is used to store all required intermediate and final information and metadata. After successfully finalizing the training job, the new global model can be used for local inference by FL local operations.

The following sections describe the four ASSIST-IoT FL enablers in detail [13].

11.3.1 FL enablers

11.3.1.1 FL Orchestrator

FL orchestrator is the enabler responsible for specifying and managing FL workflow(s)/pipeline(s), including:

- FL job scheduling;
- Management of the FL lifecycle;
- Selection and delivery of initial version(s) of the shared model;
- Delivery of the version(s) of models used in various stages of the process, such as training stopping criteria;
- Handling the different "error conditions" that may occur during the FL process.

It is formed by two components:

- **FLS API server:** Offers a REST API to allow for the communication and interaction with the other enablers of the FL system. Although the communication of model updates and configuration between the FL training collector is carried out via gRPC, all traffic between the FL orchestrator or the FL repository and the rest of the enablers is exchanged using a RESTful API. Hence, it allows to retrieve information or perform FL management actions, to FL local operations, FL training collector, and FL repository.

- **FLS workflow manager:** This component is in charge of defining the workflow for a specific instance of the FL lifecycle. Workflow description specifies, among others, the source of initial configuration (e.g., minimum number of FL local operations needed for federated training, number of training rounds for carrying out the federated learning process, the initial shared ML model to be used, evaluation criteria method and required accuracy value, method used for parameter aggregation, and required encryption mechanisms), and lifecycle management (e.g., evaluating the number of FL local operations connected, or the number of training rounds finished provided by the FL training collector).

11.3.1.2 FL Repository

The FL repository is used to store all information necessary to conduct the FL process (configuration, models, algorithms, etc.). It consists of two components, one holding the FastAPI, server which is in constant contact with the second component that encapsulates the MongoDB database.

This database is used to store initial ML models, already trained ML parameters suitable for specific datasets and formats, multiple averaging approaches, as well as additional functionalities that may later be needed, including data transformations and IP addresses of potential client instances present in the FL system of ASSIST-IoT. ML model weights are kept in the form of GridFS chunks in order to allow them to exceed the size of 16 MB (which they sometimes do).

The FastAPI server serves just as a gatekeeper to the MongoDB instance, allowing for the easy performance of specific queries (and only performing those queries).

11.3.1.3 FL Training Collector

The FL training process involves several independent parties that commonly collaborate in order to provide an enhanced ML model. In this process, the different local update suggestions shall be aggregated accordingly. This duty within ASSIST-IoT is tackled by the FL training collector, which resides in a centralized location and is also in charge of delivering back the updated model. Therefore, its functionalities are:

- Aggregation of local updates of the ML model prepared by independent parties as a part of a model enhancement process by means of the specialized FL averaging mechanisms and FL training collector I/O components.

- Supplying specific FL local operations with any additional configuration they might need by communicating via gRPC.
- Configuration of the employment of privacy mechanisms on edge (in the case of differential privacy) or just aggregating the weights in a manner compliant with those mechanisms (in the case of homomorphic encryption).
- Delivering back to the parties the updated model using the established gRPC connection, synchronizing the training, and later obtaining the results of local training.
- In some cases, the FL training collector may also conduct performance evaluation on the global model throughout training. For this purpose, it will also use the data transformation module (in order to pre-process the test data before the evaluation). More information about the data transformation module will be presented in a later section.

11.3.1.4 FL Local Operations

The FL local operations is the enabler embedded in each involved party performing local training. Its components and their respective functionalities are:

- **Data transformer** is used for the verification of local data format compatibility with the data formats required by the models being trained, as well as for application of the required data transformations using predefined transformers if needed. For more details about the data transformation module, please refer to the next section.
- **Local model trainer** is in charge of getting the local results that are later on passed to the FL training collector to carry out the proper aggregation method over the common shared model.
- **Local model inferencer**, as its name suggests, carries out the inference process of the final shared ML model over new incoming data.
- **Privacy**. There are two privacy mechanisms available out of the box provided by ASSIST-IoT enablers: differential privacy with adaptive clipping and homomorphic encryption. The differential privacy mechanism was based on [5] and [6]. Here, the influence of the model update supplied by a given client is not clipped according to a fixed clipping threshold but adaptively modified throughout training. Although the Gaussian noise and clipping is applied on the side of FL local operations, FL training collector is responsible for most of the metric computation needed to adjust the clipping. Homomorphic encryption, on the other

hand, requires a significant additional computation on the side of FL local operations, with only small adjustments needed on the side of the FL training collector. Therefore, the computational and communications overhead introduced by the homomorphic encryption currently prohibits its use beyond the training of very simple models using a specially adapted version of the federated averaging strategy.

- **Local communication** is the RESTful API that acts as the entrance and exit gate of the FL local operations with the rest of the enablers of the FL system. The FL local operations can also additionally establish a gRPC connection with the FL training collector.

11.3.2 Secure reputation mechanism for the FL system via blockchain and distributed ledger

In addition to the baseline FL features, an external distributed ledger enabler will be included in the next iteration of the pilot. It would provide a secure reputation mechanism for all the local operators. The reputation mechanism therefore will constitute a safe guard mechanism that prevents free-riders from freely accessing the global model without contributing to it and also malicious adversaries from poisoning the global model [17]. To do so, blockchain technology has been proposed. This technology allows the secure maintenance of a distributed ledger among several parties without the need of a trusted centralized authority using a consensus algorithm. Blockchain technologies depending on whether we refer to permissionless or permissioned blockchain networks can ensure different security aspects. For permissionless blockchain networks, transparency, decentralization, immutability, and traceability of shared data can be ensured, while permissioned networks can ensure private transactions by granting access to the data of the distributed ledger only to authorized users who have the right permissions [2], [15], [16].

The integration of the DLT enabler with the FL baseline system is illustrated in Figure 11.2. The DLT enabler will calculate reputation scores for each FL local operator instance, which will be stored on a permissioned blockchain network that allows only authorized users to have access to the scores and also to participate in the consensus algorithm that updates them. This consequently will increase the privacy of the reputation score data. Next, FL training collector will send the weights from the FL local operations and the weights from the global model to the distributed ledger (DLT).

The final reputation score for each FL local operations will be calculated using the cosine similarity between the weights of FL local operations and the aggregated weight [17]. The final reputation score for each local operator

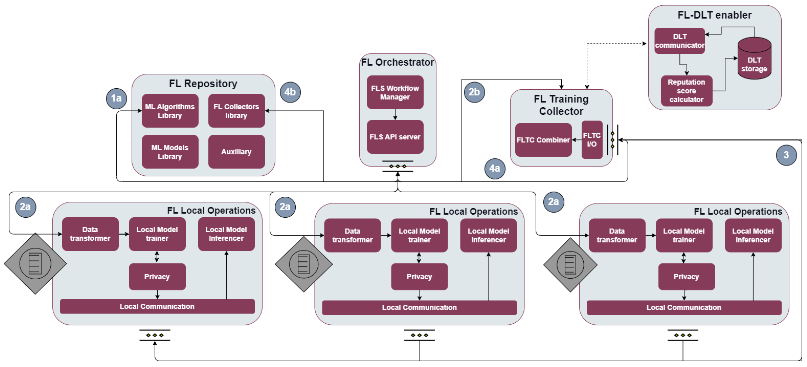


Figure 11.2 ASSIST-IoT FL-DLT enabler.

will be stored in the DLT along with the reputation set that contains the FL local operations who would be considered reputable in this round (if their calculated score is not below a given threshold). The FL training collector will query on an on–off strategy to the reputation scores and reputation set, so that further decisions on the penalties or incentives for the FL local operations may be taken.

In detail, the FL-DLT enabler depicted in Figure 11.2 is composed of three components:

- **Distributed ledger (DLT) communicator:** This component is a RESTful API that receives weights from the training collector, and it also fetches from the DLT storage and sends back to the training collector the reputation scores and reputation set.
- **Reputation score calculator:** This component applies the reputation mechanism and calculates the scores for each local operator in each training round. It also maintains a reputation set containing all the reputable local operators.
- **Distributed ledger (DLT) storage:** This component stores the reputation scores and the reputation set to the distributed ledger.

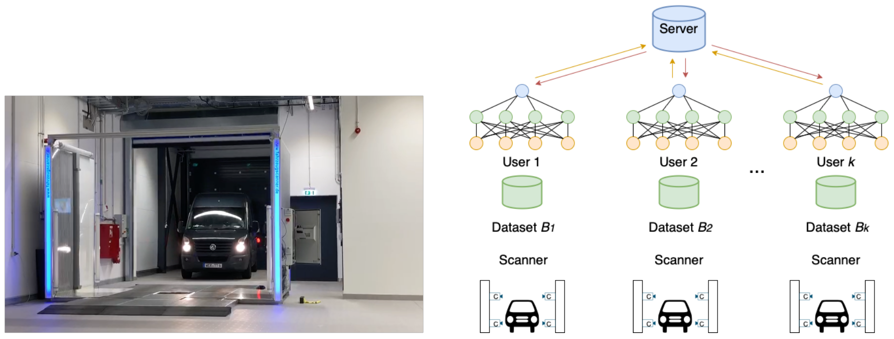
11.4 ASSIST-IoT FL Application in an Automotive Defect Detection Use Case

11.4.1 Business overview and context of the scenario

During the last years, the digitalization pressure and optimization needs are deeply studied in the automotive field. AI-based surface inspection of the vehicle exterior is seen as a well-promised common case of the previous one, where precisely FL can hold a primary role. For instance, in the proposed validation scenario, where images are taken by cameras in arch scanners that are installed in various locations and potentially in garages owned by different entities. First, it relaxes the need of sharing data (privacy and aversion concerns). Second, it reduces the dependency on network connection vís-a-vis a centralized, cloud approach of data gathering and AI application. Third, it moves the focus to the work close to the action (where scanners are), in terms of end-user interaction. This is relevant as this interaction actually inserts labeled training and validation data; thus, efficiency is improved.

In the proposed case, the goal is for a FL-powered deep learning system to relax the bandwidth usage and the overall network dependency and to provide faster and more accurate detection of defects (currently up to 15 minutes). The FL solution will need to deal with cameras that capture data and metadata of 50–300 colored, high-resolution images per vehicle analyzed (Figure 11.3). Then the system mounted forward data via fiber optics, 4G, and 5G to a cloud location. Edge locations (where scanners reside) are equipped with an intelligent storage system with local buffering (but have limited storage capacity) and provide a direct connection to end users that annotate human-visualized defects. There, the associated front-end software must handle a hundred thousand images, offering advanced, application-centered visualization, and display with an optional focus on existing damages and AI proposal. It must be considered that the data can be very heterogeneous due to different models, locations, scanner owner, and indoor/outdoor position, among others. Therefore, the AI-based inspection can strongly support both manual users reviewing or automated inspection and evaluation procedures to monitor and determine the vehicle's exterior conditions. Due to the nature of the task, the consideration of the images of many scanners for the AI-model training has large impact on the overall quality of the global AI models in the current cloud approach

From the federated learning point of view, the task setup may look as follows (Figure 11.3). Either scanners or individual cameras can operate as federated clients, performing both model training and inference tasks, with the central server being responsible for coordinating the processes, like training, testing, aggregating, and distributing the latest version of the global model.



(a) Scanner gate (b) Scanner structure

Figure 11.3 Car damage recognition - scanner gate.

11.4.2 Proposed solution and benefits of decentralized learning strategy

The application of AI and specifically FL to the automotive use case enables to optimize the process of damage recognition with respect to current situation. First, application of AI and inference close to the sources of the data will enable faster processing of the data and recognition of situations that need special handling. Second, the benefits coming with using FL-based approach can be identified compared to the centralized approach. The centralized approach is an alternative in which all data collected on local devices (e.g., scanner cameras) is sent to the server (cloud) and processed there to train the model or inference on using the trained model. Therefore, FL allows the described use case to benefit from the following:

- Significantly reduced or practically non-existing necessity to transfer privacy-sensitive data. As a result, the whole dataset is never stored in a centralized manner, but local datasets are available on client nodes and used there to train the local versions of the model. This increases overall data privacy which is important when considering multi-stakeholder environment with coopetition. This is relevant for the selected application case due to the aforementioned aversion and privacy concern existing while scanners are owned by multiple entities (and even in single scenarios). Cameras are subject to confidentiality rules, as they may contain private information of both the vehicle owner and the company that performs the damage inspection.
- Decreased need for data storage capacities on the server side and reduced data transfer between local devices and server (cloud). This is

specifically important in big data environments. This would help entities using the scanner-based installation to be more efficient and accurate on their predictions, as the need of communication toward cloud would be relaxed and the annotation by company stuff would leverage (limited) edge storage capacities instead of submitting upwards.

- Improved inference speed, as the location of the global model on the local nodes implies that no communication between the source of the data and the server (cloud) for generating predictions is intended and all the inference happens close to the device that generated the data. As indicated in the business case description, this is the main goal of the application of FL system, aiming to reduce the (up to) 15 minutes current timeframe for predictions.
- Personalization availability. Due to the previously described reasons for local data heterogeneity, federated clients may require additional uptraining on their local data for better model performance, and FL provides an easy way to produce a more personalized tool for vehicle damage detection that takes into consideration features of the local dataset, while still benefiting from the generalized knowledge from the multiple entities that participated in the joint training.
- Global model aggregation techniques can be used to mitigate the effect of the heterogeneity of damage and vehicle types present on the client nodes. Here, allowing local training of models, which can grasp more nuances related to usual vehicles in a specific location (e.g., vans), would enhance the depth of knowledge that can be applied to other sites with less volume of such. Therefore, FL system is capable of adapting to task-specific challenges and site-specific data.

Figure 11.4 shows how enablers proposed in the ASSIST-IoT FL architecture can be combined in the system deployed for automotive defect detection use case. Here, the *FL local operations* run on clients (cameras), whereas *FL orchestrator*, *FL training collector*, and *FL repository* are located in the cloud. The main goal is to distribute the processing, instead of sending all the images to the cloud and processing it centrally. Here, although the centralized topology seems to be a good choice for initial implementation, it can be foreseen that a more complex topology (e.g., hierarchical) may be ultimately needed [4]. One of the reasons is that in an extended deployment, groups of scanners may belong to different stakeholders that all want to benefit from the good detection model but without disclosing their data.

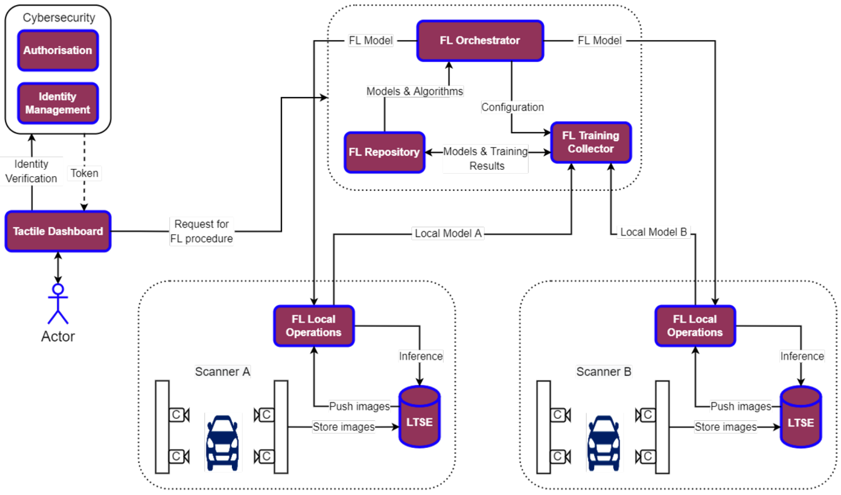


Figure 11.4 FL architecture for the automotive defect detection use case.

Note that, on the diagram, besides aforementioned FL enablers, additional enablers designed and implemented within ASSIST-IoT are included addressing: *cybersecurity* (specifically authentication and authorization), *long-term storage* (the *long-term storage enabler* can provide local storage of images for FL clients), and *tactile dashboard* (for visualizations needed in the system). Upon reflection, it is easy to see that these elements can provide all additional functions needed in the considered ecosystem.

11.4.3 Proposed validation

Federated learning experiments for the car damage detection use case were performed based on the mask-RCNN model for object detection and segmentation. During the federated training, separate cameras were treated as federated clients. Initial experiments were performed with a total of eight cameras, although in the future scenarios, more populated experiments are expected.

The evaluation of the FL model is based on the holdout evaluation dataset, which consists of images, representing a comprehensive set of possible inference scenarios. This dataset also includes images with no detected damages at all, in order to properly test the model's capability to accurately detect both damages and their absence. An example of the damage detected by the model is shown in Figure 11.5.

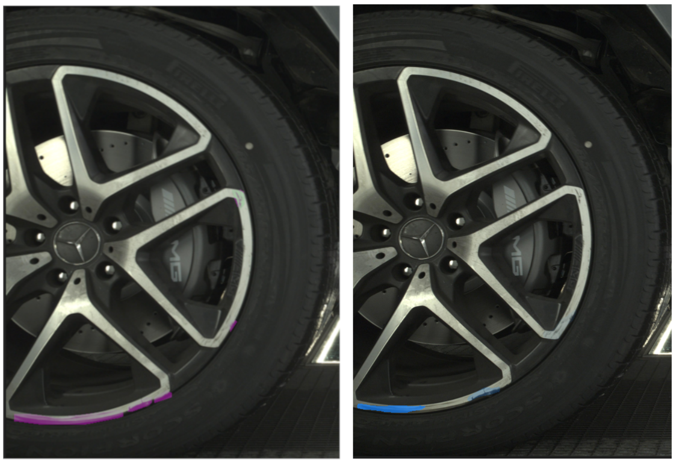


Figure 11.5 Rim damage detection example (target – left; result – right).

The metrics taken into account are appropriate for the task of object detection and segmentation. The main performance indicators, therefore, are precision, recall, and the resulting F1 score per damage category with the IoU (intersection over union) threshold set at a reasonable value. For the deployed model, the appropriate IoU is expected to be around 0.5.

Apart from the calculated performance indicator, an expert-based evaluation is also implied. As the system is expected to assist human professionals during their damage evaluation activities, their feedback will provide the necessary information for further model improvements.

Finally, for the evaluation of the use case, the following KPIs have been identified and will be controlled and verified: (i) increase of detected defects on the car exterior, (ii) faster vehicle inspection compared to the current process (planned at least 30% increase), and (iii) minimization of data transfer (planned at least 50% increase).

11.5 Conclusions

Decentralized AI promises to be a relevant innovation to be incorporated to Next-Generation IoT deployments. From the viewpoint of decentralized intelligence, ASSIST-IoT has focused on Federated Learning. This technique relies on training machine learning models in coopetition manner –a joint cooperation and competition approach– over heterogeneous nodes located at different locations and with different computing capabilities. For doing so, strategies to locally train the models, centrally orchestrate the averaged updates, and a way to bring back the trained model for either further training or infererence to those edge devices is needed. Rooting on relevant references and based on the novel architecture provided by the project, an FL system has been designed and it is being developed.

One of the many applications that such a system could have is materialized in a real-life use case brought by ASSIST-IoT, consisting of leveraging edge computing in various locations to train ML algorithms that detect defects on vehicles' surfaces. The usage of ASSIST-IoT's FL system allows to improve inference speed as well as reduce the network bandwidth needs to the cloud, while keeping the data in the local environments, thus increasing security and privacy. The use case is currently being trialed and some early evaluation activities are providing optimistic outlooks. Final results of the experiment will be presented in future works.

Acknowledgements

This work is part of the ASSIST-IoT project that has received funding from the EU's Horizon 2020 research and innovation program, under Grant Agreement No. 957258.

References

[1] ASSIST-IoT project. https://www.assist-iot.eu. Accessed: 2023-01-06.

[2] Permissioned blockchain vs. permissionless blockchain: Key differences. https://cointelegraph.com/blockchain-for-beginners/permissioned-blockchain-vs-permissionless-blockchain-key-differences. Accessed: 2023-01-13.

[3] The European Data Strategy, 202. https://ec.europa.eu/commission/presscorner/detail/en/fs_20_283. Accessed: 2023-01-06.

[4] *Introducing Federated Learning into Internet of Things ecosystems – preliminary considerations*, 07 2022.

[5] Galen Andrew, Om Thakkar, Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. *Advances in Neural Information Processing Systems*, 34:17455–17466, 2021.

[6] Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

[7] O. Elloumi, M. Carugi, J. P. Desbenoit, G. Karagiannis, and P. Murdock. AIOTI-WG3-High Level Architecture (HLA) Release 5.0. 2020.

[8] Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *CoRR*, abs/1811.03604, 2018.

[9] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, and M. Hadi Amini. A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24, 2022.

[10] Latif U. Khan, Walid Saad, Zhu Han, Ekram Hossain, and Choong Seon Hong. Federated learning for internet of things: Recent advances, taxonomy, and open challenges. *CoRR*, abs/2009.13012, 2020.

[11] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[12] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016.

[13] Marcin Paprzycki, Maria Ganzha, Katarzyna Wasielewska, and Piotr Lewandowski. Devsecops methodology for ng-iot ecosystem development lifecycle - assist-iot perspective. *Journal of Computer Science and Cybernetics*, 37(3):321-337, Sep. 2021.

[14] Yuji Roh, Geon Heo, and Steven Euijong Whang. A survey on data collection for machine learning: A big data - ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1328–1347, 2021.

[15] Siamak Solat, P. Calvez, and Farid Naït-Abdesselam. Permissioned vs. permissionless blockchain: How and why there is only one right choice. *Journal of Software*, 16:95 – 106, 12 2020.

[16] Muhammad Habib ur Rehman, Khaled Salah, Ernesto Damiani, and Davor Svetinovic. Towards blockchain-based reputation-aware federated learning. In *IEEE INFOCOM 2020 - IEEE Conference on*

Computer Communications Workshops (INFOCOM WKSHPS), pages 183–188, 2020.

[17] Xinyi Xu and Lingjuan Lyu. A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning. *arXiv preprint arXiv:2011.10464*, 2020.