European Conference on EDGE AI Technologies and Applications - EEAI

Connecting the future and driving the next wave of technological advancements for a better world.

21-23 October 2024, Cagliari, Sardinia, Italy

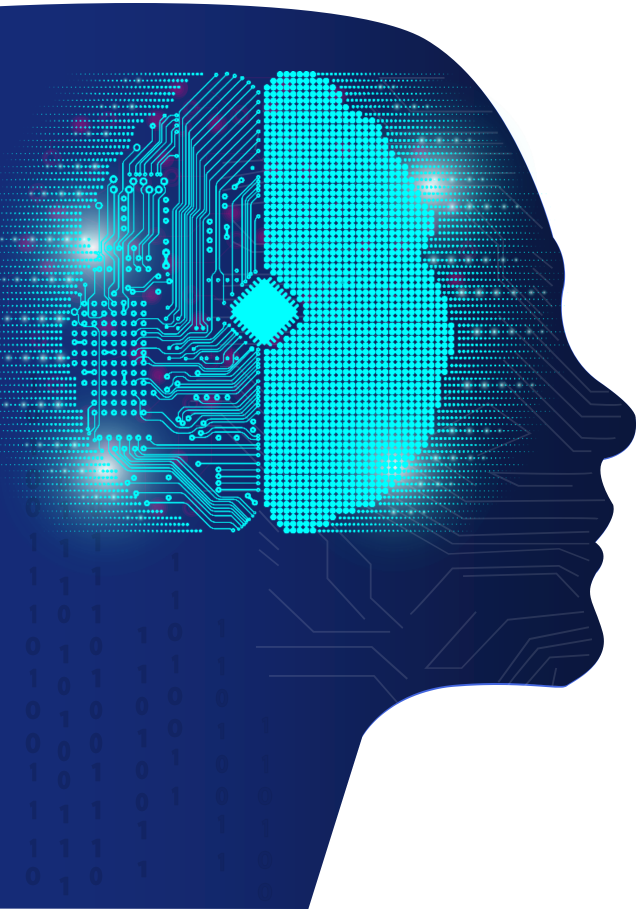
European Conference on EDGE AI
Technologies and Applications - EEAI

BAYESIAN APPROACHES TO UNCERTAINTY ESTIMATION IN EDGE AI COLLABORATIVE LEARNING

Gleb Radchenko, Silicon Austria Labs, Austria

21-23 October 2024 Cagliari, Sardinia, Italy

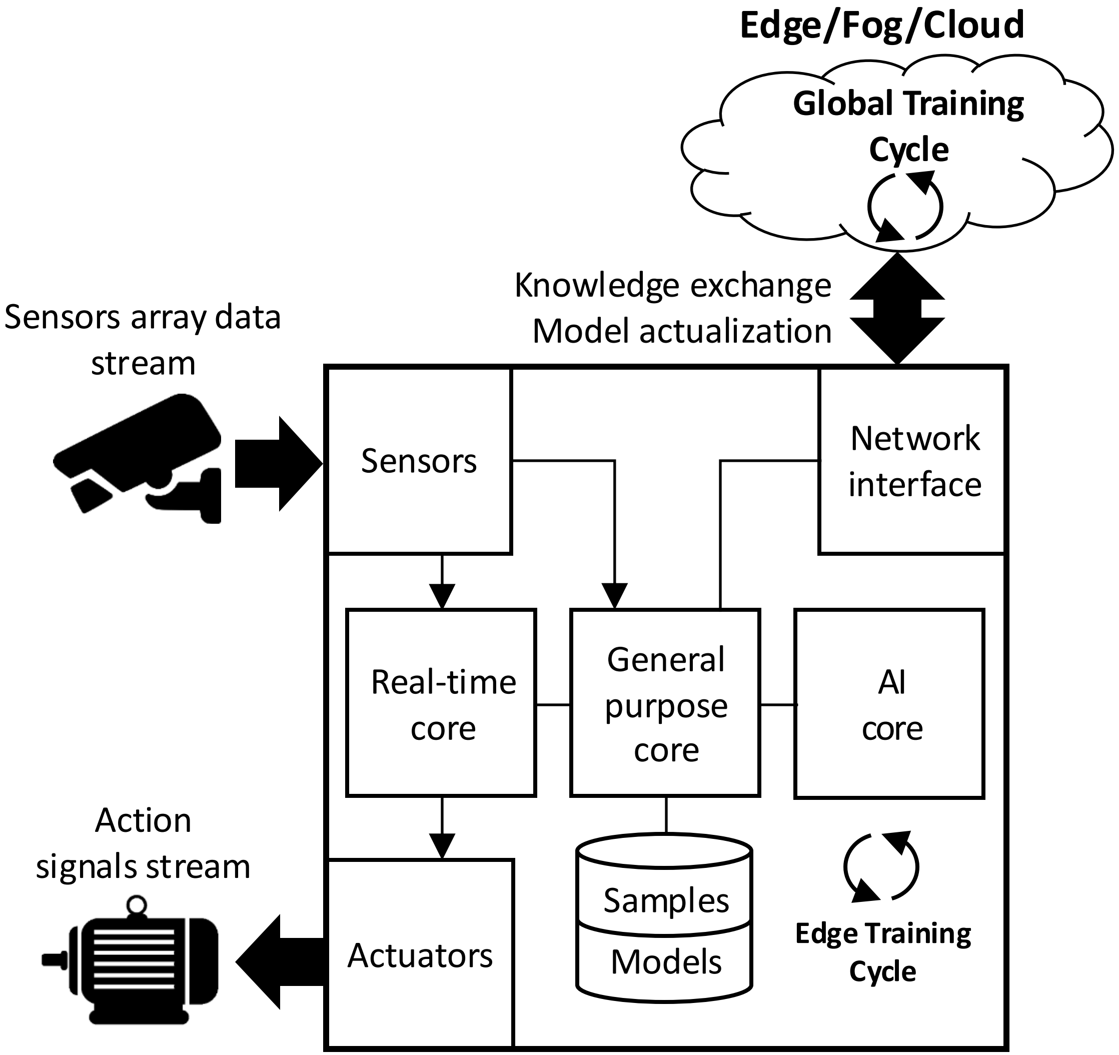
Presentation Outline



- Introduction
- State-of-the-art
- Research Goal
- Methodology: ADMM and BNN
- Design and Implementation: collaborative mapping case
- Experimental Setup and Results
- Future Research
- Conclusion

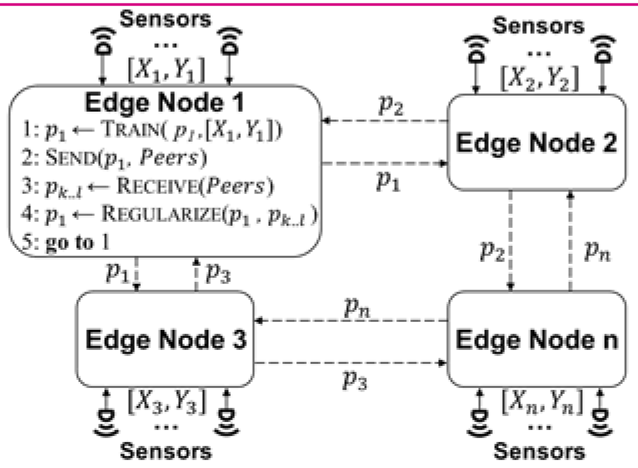
Decentralized learning on AI-Enabled edge devices

- We explore the potential for enabling decentralized learning and knowledge sharing among AI-Enabled Edge Devices (AEEDs).
- An AEED is an agent device situated at the network edge, directly interfacing with data streams from various sensors.
- It may also control actuators to interact with its environment.
- Beyond standard computational capabilities, these devices feature an AI Core capable of conducting both inference **and model training (tuning)** directly on the device.



AI-enabled Edge device

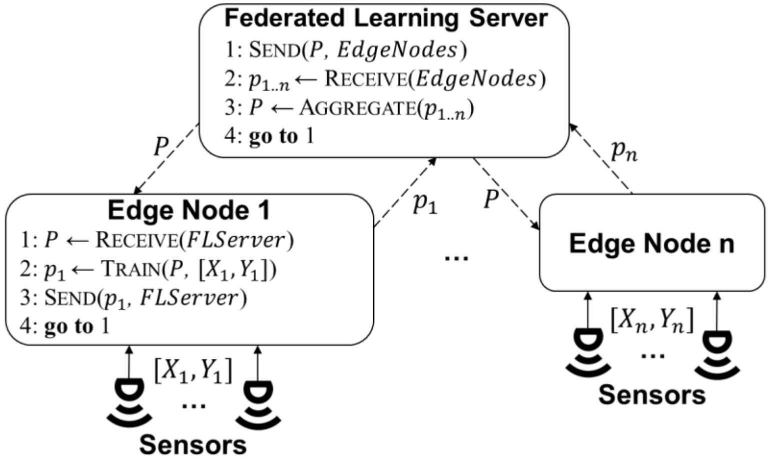
Edge Learning Methods (SoTA)



Edge Node 1

```
$p_1 \leftarrow \text{TRAIN}(p_l, [X_1, Y_1])$
$\text{SEND}(p_1, Peers)$
$p_{k..l} \leftarrow \text{RECEIVE}(Peers)$
$p_1 \leftarrow \text{REGULARIZE}(p_1, p_{k..l})$
go to 1
```

Alternating Direction Method of Multipliers (ADMM)



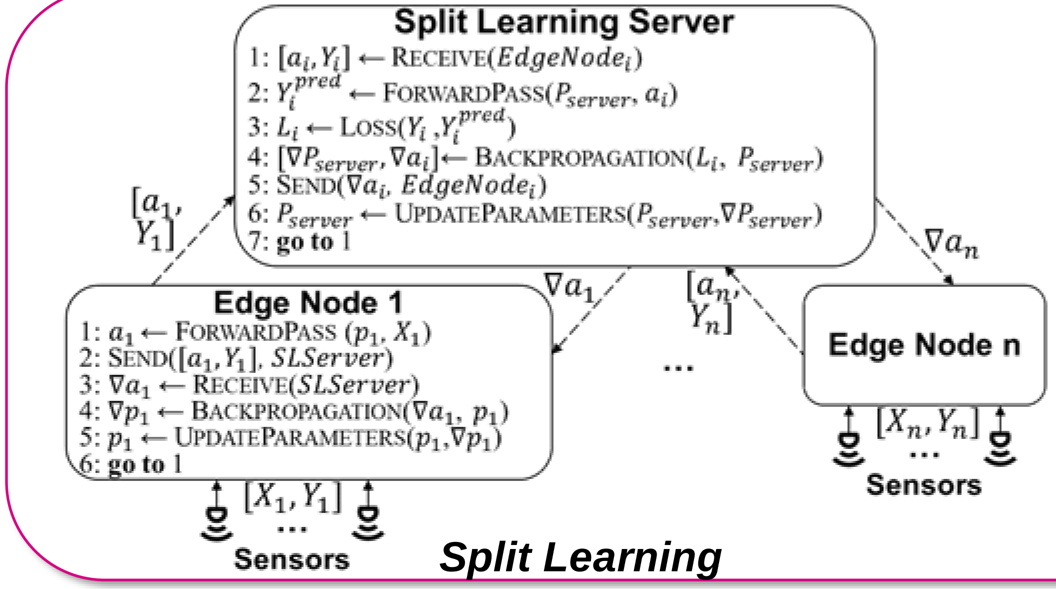
Federated Learning Server

```
$\text{SEND}(P, EdgeNodes)$
$p_{1..n} \leftarrow \text{RECEIVE}(EdgeNodes)$
$P \leftarrow \text{AGGREGATE}(p_{1..n})$
go to 1
```

Edge Node 1

```
$P \leftarrow \text{RECEIVE}(FLServer)$
$p_1 \leftarrow \text{TRAIN}(P, [X_1, Y_1])$
$\text{SEND}(p_1, FLServer)$
go to 1
```

Federated Learning



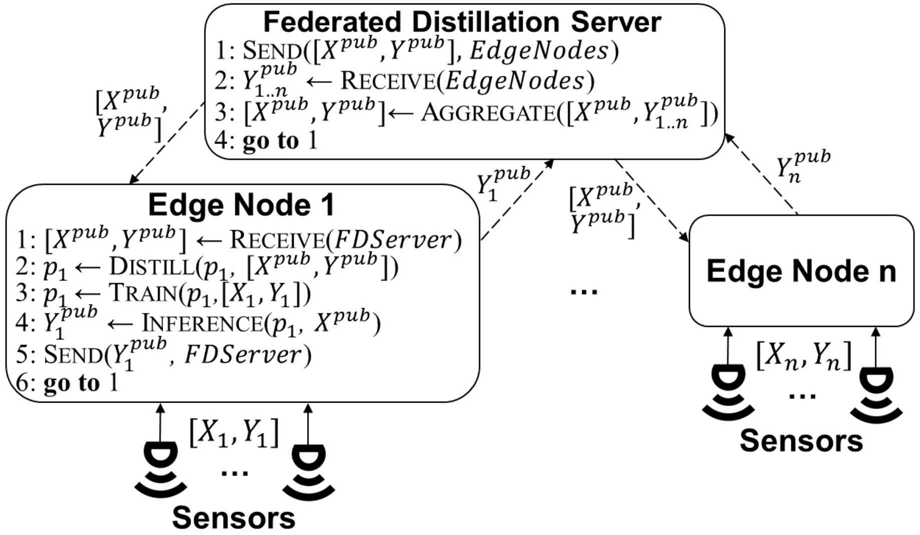
Split Learning Server

```
$[a_i, Y_i] \leftarrow \text{RECEIVE}(EdgeNode_i)$
$Y_i^{pred} \leftarrow \text{FORWARDPASS}(P_{server}, a_i)$
$L_i \leftarrow \text{LOSS}(Y_i, Y_i^{pred})$
$[\nabla P_{server}, \nabla a_i] \leftarrow \text{BACKPROPAGATION}(L_i, P_{server})$
$\text{SEND}(\nabla a_i, EdgeNode_i)$
$P_{server} \leftarrow \text{UPDATEPARAMETERS}(P_{server}, \nabla P_{server})$
go to 1
```

Edge Node 1

```
$a_1 \leftarrow \text{FORWARDPASS}(p_1, X_1)$
$\text{SEND}([a_1, Y_1], SLServer)$
$\nabla a_1 \leftarrow \text{RECEIVE}(SLServer)$
$\nabla p_1 \leftarrow \text{BACKPROPAGATION}(\nabla a_1, p_1)$
$p_1 \leftarrow \text{UPDATEPARAMETERS}(p_1, \nabla p_1)$
go to 1
```

Split Learning



Federated Distillation Server

```
$\text{SEND}([X^{pub}, Y^{pub}], EdgeNodes)$
$Y^{pub}_{1..n} \leftarrow \text{RECEIVE}(EdgeNodes)$
$[X^{pub}, Y^{pub}] \leftarrow \text{AGGREGATE}([X^{pub}, Y^{pub}_{1..n}])$
go to 1
```

Edge Node 1

```
$[X^{pub}, Y^{pub}] \leftarrow \text{RECEIVE}(FDServer)$
$p_1 \leftarrow \text{DISTILL}(p_1, [X^{pub}, Y^{pub}])$
$p_1 \leftarrow \text{TRAIN}(p_1, [X_1, Y_1])$
$Y_1^{pub} \leftarrow \text{INFERENCE}(p_1, X^{pub})$
$\text{SEND}(Y_1^{pub}, FDServer)$
go to 1
```

Federated Distillation

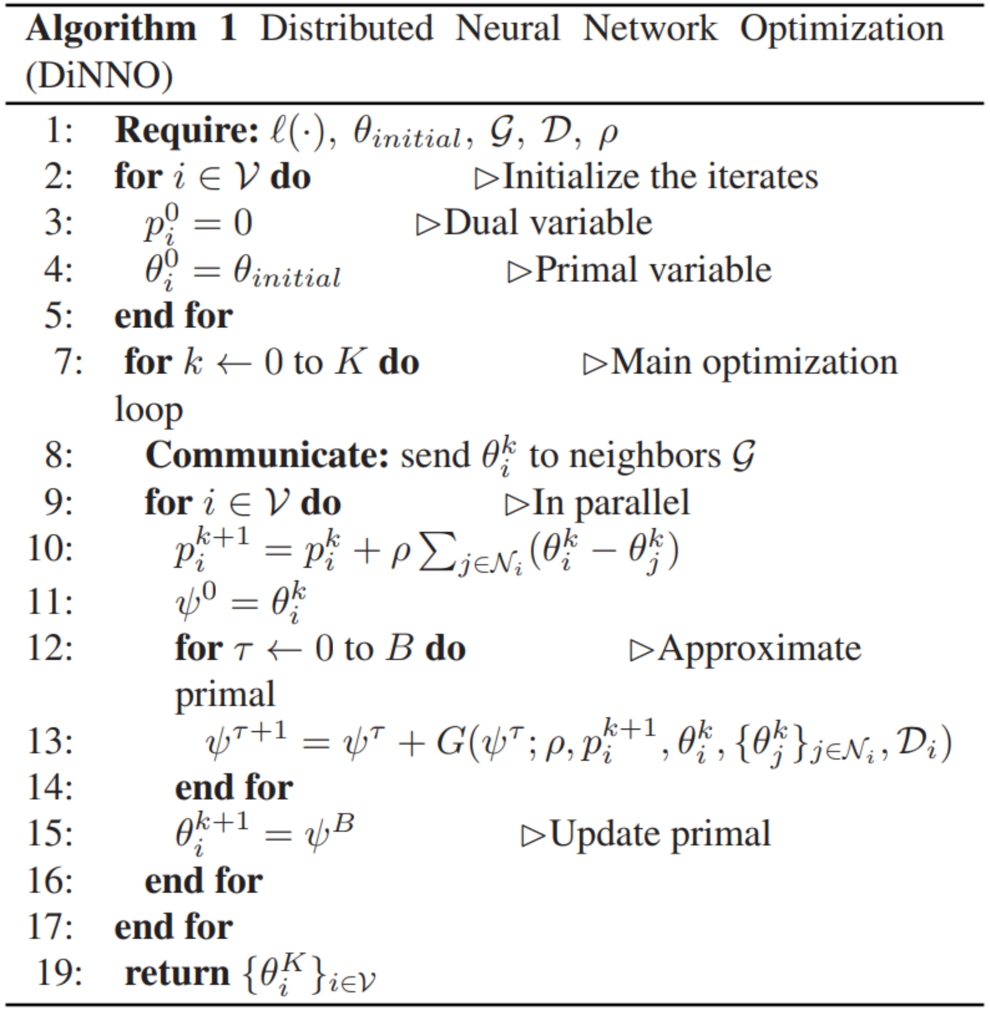
Radchenko, G., & Fill, V. A. (2024). Edge AI Collaborative Learning: Bayesian Approaches to Uncertainty Estimation. *arXiv preprint arXiv:2410.08651*. DOI:10.48550/arXiv.2410.08651

Research Goal

- Investigate the algorithms and methods for deploying **distributed machine learning** within the framework of autonomous, network-capable, sensor-equipped, AI-enabled edge devices.
- Within the framework of this study specifically, we focus on determining **confidence levels in learning outcomes**, considering the spatial and temporal variability of data sets encountered by independent agents.
- To achieve this, we address the following tasks:
  - Decouple the ADMM-based Distributed Neural Network Optimization (DiNNO) algorithm implementation into independent processes, enabling asynchronous network communication for distributed learning
  - Integrate distributed uncertainty estimation into the resulting models by applying **Bayesian Neural Networks (BNN)**
  - Implement and evaluate the proposed approaches within a case: simulation of robots navigating a 3D environment using the Webots platform, augmented with advanced LiDAR sensors for environmental mapping

Alternating Direction Method of Multipliers (ADMM)

- ADMM is an algorithm that solves convex optimization problems by breaking them into smaller pieces, each of which are then easier to handle.
- We take the Distributed Neural Network Optimization (DiNNO) algorithm as a basis for our research



Algorithm 1 Distributed Neural Network Optimization (DiNNO)

```
**Require:** $\ell(\cdot)$, $\theta_{initial}$, $\mathcal{G}$, $\mathcal{D}$, $\rho$
**for** $i \in \mathcal{V}$ **do**                    ▷Initialize the iterates
    $p_i^0 = 0$                                      ▷Dual variable
    $\theta_i^0 = \theta_{initial}$                  ▷Primal variable
**end for**
**for** $k \leftarrow 0$ to $K$ **do**               ▷Main optimization loop
    **Communicate:** send $\theta_i^k$ to neighbors $\mathcal{G}$
    **for** $i \in \mathcal{V}$ **do**               ▷In parallel
        $p_i^{k+1} = p_i^k + \rho \sum_{j \in \mathcal{N}_i} (\theta_i^k - \theta_j^k)$
        $\psi^0 = \theta_i^k$
        **for** $\tau \leftarrow 0$ to $B$ **do**    ▷Approximate primal
            $\psi^{\tau+1} = \psi^\tau + G(\psi^\tau; \rho, p_i^{k+1}, \theta_i^k, \{\theta_j^k\}_{j \in \mathcal{N}_i}, \mathcal{D}_i)$
        **end for**
        $\theta_i^{k+1} = \psi^B$                    ▷Update primal
    **end for**
**end for**
**return** $\{\theta_i^K\}_{i \in \mathcal{V}}$
```

J. Yu, J. A. Vincent and M. Schwager, "DiNNO: Distributed Neural Network Optimization for Multi-Robot Collaborative Learning," in IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 1896-1903, April 2022, DOI: 10.1109/LRA.2022.3142402.

Bayesian Neural Networks (BNNs)

- In a conventional neural network architecture, a linear neuron is characterized by a weight (w), a bias (b), and an activation function (f_{act}). Given an input x, a single linear neuron performs the following operation:

$$y = f_{act}\,(w \cdot \ x \ + \ b)$$

- Bayesian Neural Networks (BNNs) employ a Bayesian approach to train stochastic neural networks
- Instead of deterministic weights and biases, they utilize probability distributions, denoted $P(w)$ for weights and $P(b)$ for biases.
- Typically, these distributions are approximated as Gaussian, with mean and standard deviation derived from the training data. So, the operation of a Bayesian Linear neuron can be described as:

$$P(y|x) = f_{act}\left(\sum P(w) \ \times \ x + P(b)\right)$$

- For inference, BNNs might conduct multiple forward passes. The standard deviation of the inference values distribution indicate the model's uncertainty for each point in the input data space.

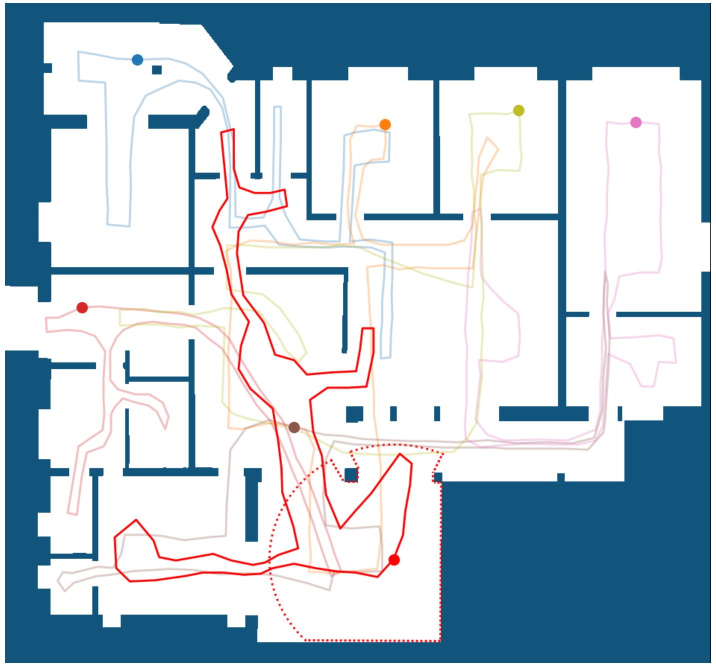
Implementation: Collaborative mapping case

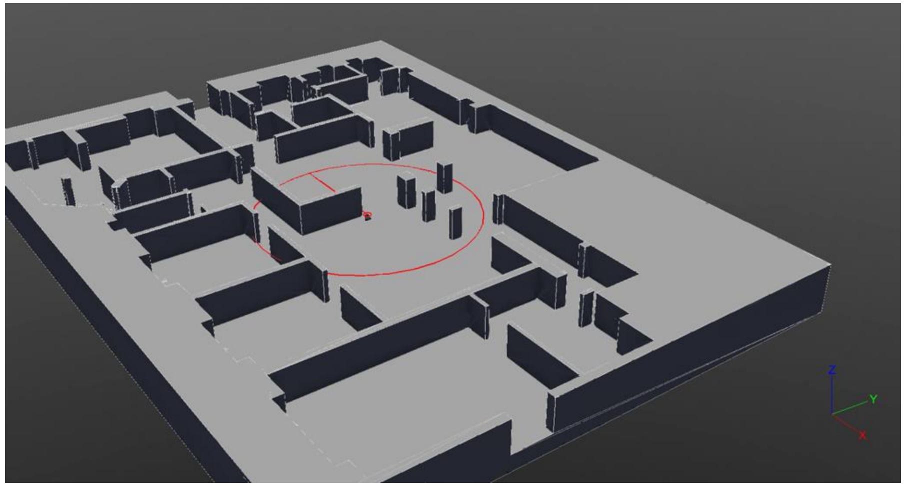
- We test our approach based on collaborative mapping task. This task involves deploying a network of independent, robotic edge devices (robots) at various starting points.

- Each device is tasked with building a coherent map of the environment, utilizing installed LiDAR, and exchanging knowledge about the environment with other devices.

- These devices are designed to update a local ML model with newly acquired data samples and implement inter-device communication via a network interface

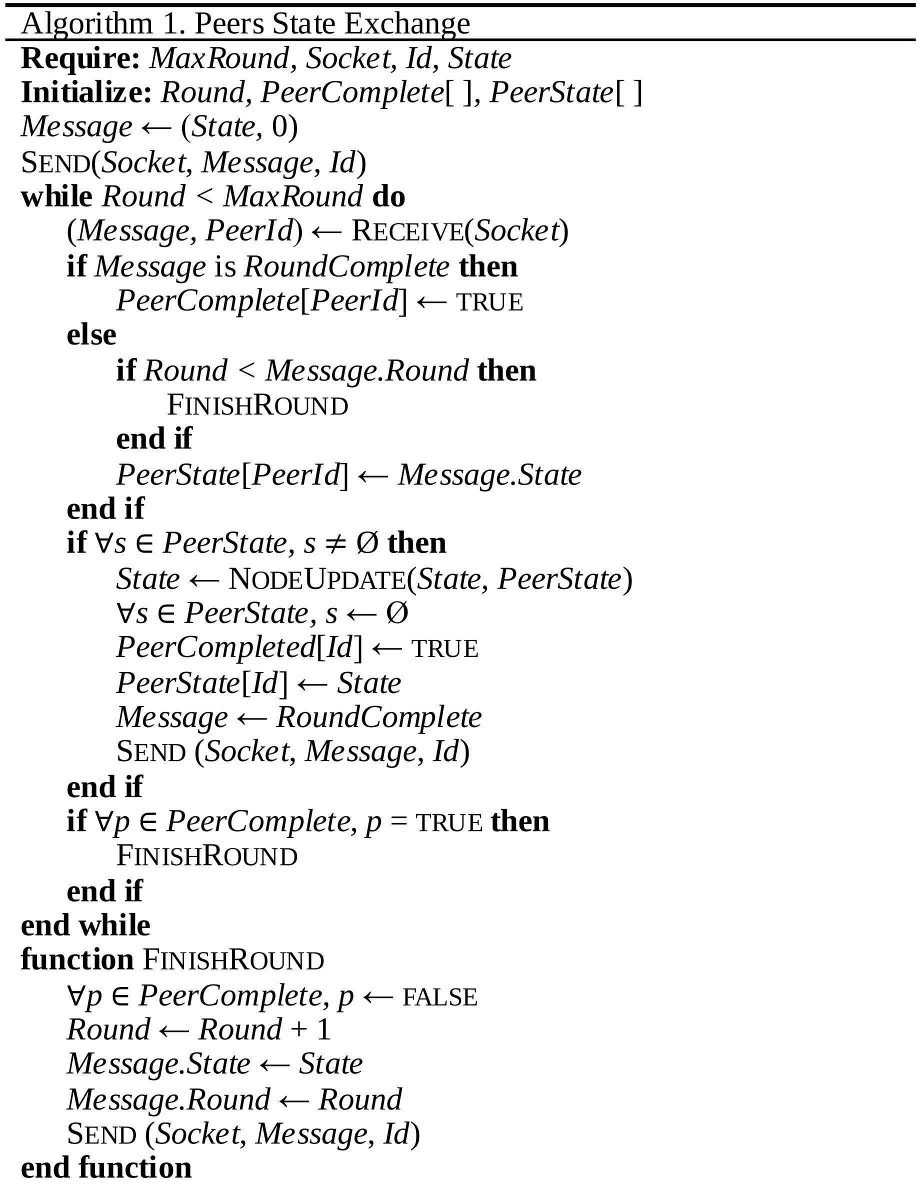
- The CubiCasa5K data set was used as a reference for the floor plans generation





Implementation: Decentralized State Exchange Algorithm

- Original DiNNO implementation is a centralized learning framework that relies on sequential learning processes based on shared agents’ memory.
- We have introduced an epoch-based algorithm to support the decentralized peer-to-peer exchange of NN parameters among agents. Generally, the following steps are implemented:
- Edge NN training using local data set
- P2P exchange of NN parameters
- Regularization of local NN parameters based on the parameters, received from the peers
- This version of the algorithm operates under the assumption that each message sent will eventually be received by its intended recipient.
- In that conditions, all the peers would eventually reach the NodeUpdate state and proceed to the next round of communication



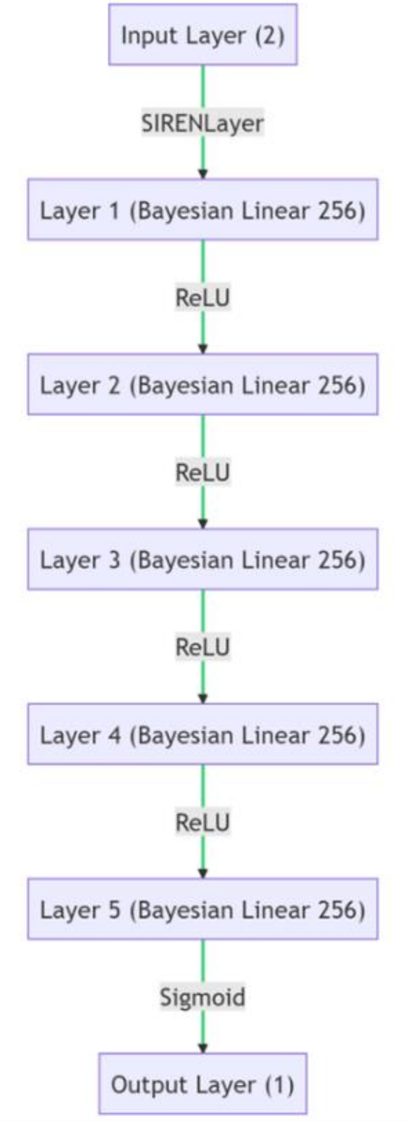
Algorithm 1. Peers State Exchange

```
Require: MaxRound, Socket, Id, State
Initialize: Round, PeerComplete[ ], PeerState[ ]
Message ← (State, 0)
SEND(Socket, Message, Id)
while $Round < MaxRound$ do
    (Message, PeerId) ← RECEIVE(Socket)
    if Message is RoundComplete then
        PeerComplete[PeerId] ← TRUE
    else
        if $Round < Message.Round$ then
            FINISHROUND
        end if
        PeerState[PeerId] ← Message.State
    end if
    if $\forall s \in PeerState, s \neq \emptyset$ then
        State ← NODEUPDATE(State, PeerState)
        $\forall s \in PeerState, s \leftarrow \emptyset$
        PeerCompleted[Id] ← TRUE
        PeerState[Id] ← State
        Message ← RoundComplete
        SEND (Socket, Message, Id)
    end if
    if $\forall p \in PeerComplete, p =$ TRUE then
        FINISHROUND
    end if
end while
function FINISHROUND
    $\forall p \in PeerComplete, p \leftarrow$ FALSE
    Round ← Round + 1
    Message.State ← State
    Message.Round ← Round
    SEND (Socket, Message, Id)
end function
```

Radchenko, G. and Fill, V. (2024). Uncertainty Estimation in Multi-Agent Distributed Learning for AI-Enabled Edge Devices. In *Proceedings of the 14th International Conference on Cloud Computing and Services Science - CLOSER*; ISBN 978-989-758-701-6; ISSN 2184-5042, SciTePress, pages 311-318. DOI: 10.5220/0012728500003711

Implementation: BNN Model

- To address uncertainty estimation in the distributed mapping problem, we implement BNN model, introducing Bayesian Linear Layers in the NN architecture.
- The architecture of the BNN is detailed as follows
  - *Input Layer (2):* x, y – an input coordinate representing the global position on the environment map.
  - *SIRENLayer (256):* a layer with a sinusoidal activation function suitable for Neural Implicit Mapping.
  - *4 x Bayesian Linear Layers (256):* four Bayesian Linear layers with 256 nodes each, activated by the ReLU function. These layers are probabilistic and support uncertainty estimation.
  - *Output Layer (1):* a linear layer with one node activated by the Sigmoid function.
- This approach introduces probabilistic inference to the model, allowing for estimating uncertainty in the network's predictions.

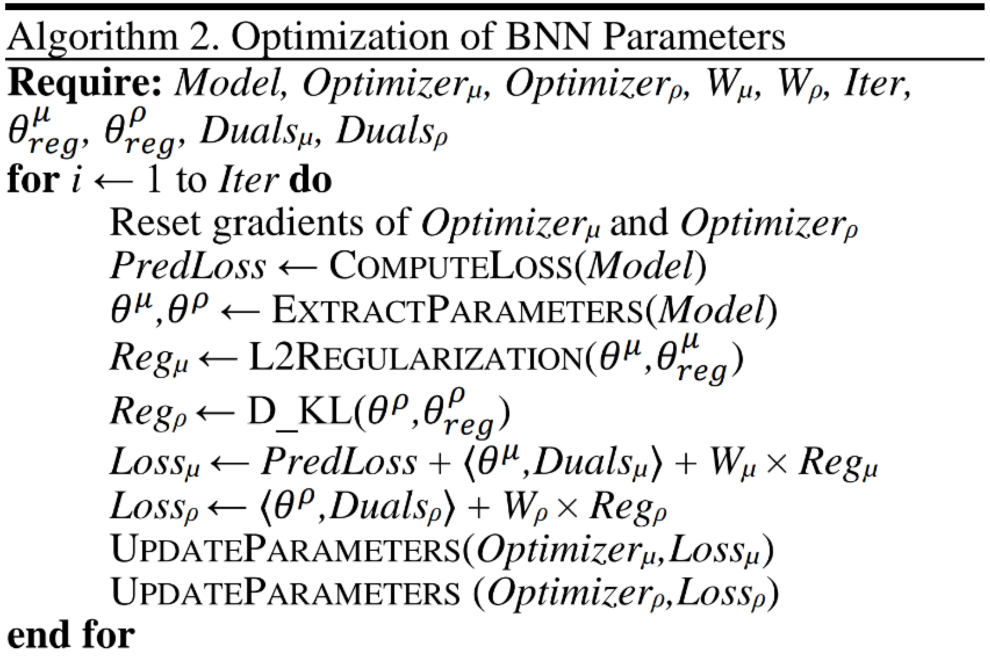

Input Layer (2)
SIRENLayer
Layer 1 (Bayesian Linear 256)
ReLU
Layer 2 (Bayesian Linear 256)
ReLU
Layer 3 (Bayesian Linear 256)
ReLU
Layer 4 (Bayesian Linear 256)
ReLU
Layer 5 (Bayesian Linear 256)
Sigmoid
Output Layer (1)

BNN parameters regularization

- To ensure correct regularization of the BNN parameters during the distributed learning regularization phase, Algorithm 2 has been developed to consider the semantics of median (μ) and standard deviation (ρ) parameters of BNN neurons.
- We utilize Kullback-Leibler Divergence (KL Divergence) for the regularization of BNN ρ -parameters between the models of individual actors.
- KL Divergence is employed to account for the difference between the Gaussian distributions that represent the parameters of the BNN. KL Divergence serves as a measure to quantify the dissimilarity between two probability distributions and can be generally computed as:

$$D_{KL}(g \parallel h) = \int g(x) \log \frac{g(x)}{h(x)} \mathrm{d}x$$

- Within the BNNs, applying KL Divergence helps quantify the deviation of the neural network's parameter distribution from a specified prior distribution

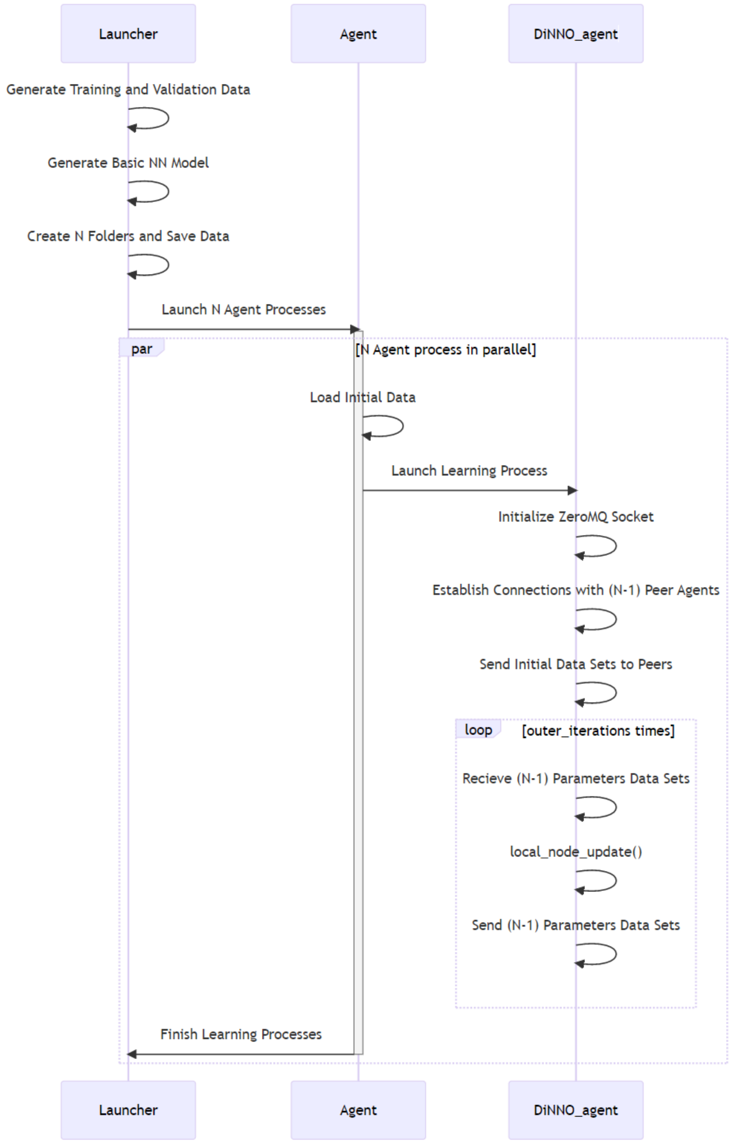


Algorithm 2. Optimization of BNN Parameters

```
**Require:** $Model$, $Optimizer_\mu$, $Optimizer_\rho$, $W_\mu$, $W_\rho$, $Iter$, $\theta^\mu_{reg}$, $\theta^\rho_{reg}$, $Duals_\mu$, $Duals_\rho$
for $i \leftarrow 1$ to $Iter$ do
    Reset gradients of $Optimizer_\mu$ and $Optimizer_\rho$
    $PredLoss \leftarrow$ ComputeLoss($Model$)
    $\theta^\mu, \theta^\rho \leftarrow$ ExtractParameters($Model$)
    $Reg_\mu \leftarrow$ L2Regularization($\theta^\mu, \theta^\mu_{reg}$)
    $Reg_\rho \leftarrow$ D_KL($\theta^\rho, \theta^\rho_{reg}$)
    $Loss_\mu \leftarrow PredLoss + \langle \theta^\mu, Duals_\mu \rangle + W_\mu \times Reg_\mu$
    $Loss_\rho \leftarrow \langle \theta^\rho, Duals_\rho \rangle + W_\rho \times Reg_\rho$
    UpdateParameters($Optimizer_\mu, Loss_\mu$)
    UpdateParameters ($Optimizer_\rho, Loss_\rho$)
end for
```

Experiment setup

- The experiment involves launching seven independent agents that gradually collect information from LiDAR sensors while exploring a virtual interior space
- Each agent runs as a separate Python process
- Agent communication is handled through direct TCP connections among the processes within the same virtual local network
- The ZeroMQ framework is used for asynchronous data exchange
- Containerization of agent processes is achieved using Singularity containers equipped with GPU access
- In the experiments outlined, we initiate all processes on GPU-enabled computing nodes managed by the SLURM workload manager

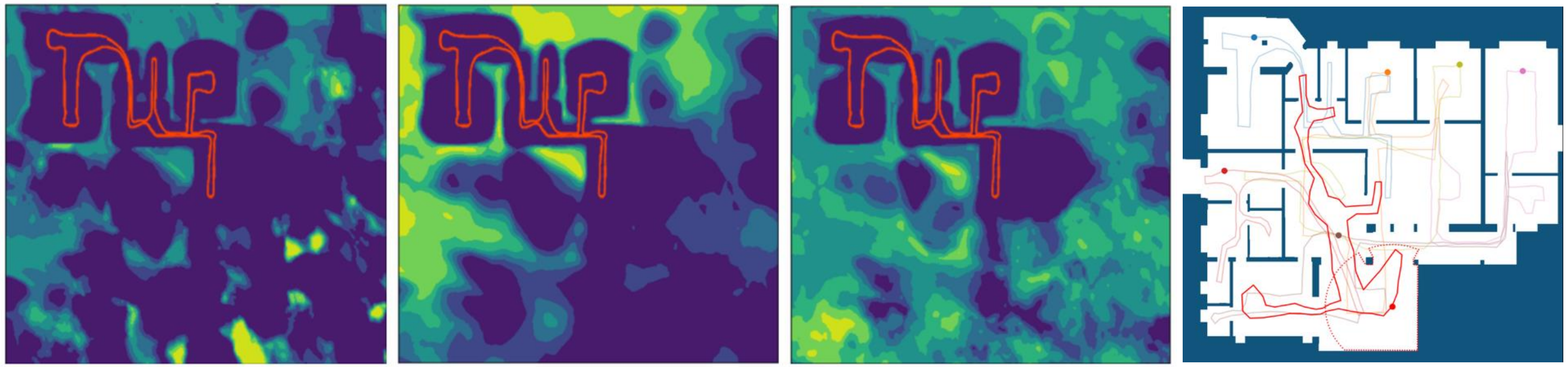

Launcher
Agent
DiNNO_agent
Generate Training and Validation Data
Generate Basic NN Model
Create N Folders and Save Data
Launch N Agent Processes
par
[N Agent process in parallel]
Load Initial Data
Launch Learning Process
Initialize ZeroMQ Socket
Establish Connections with (N-1) Peer Agents
Send Initial Data Sets to Peers
loop
[outer_iterations times]
Recieve (N-1) Parameters Data Sets
local_node_update()
Send (N-1) Parameters Data Sets
Finish Learning Processes
Launcher
Agent
DiNNO_agent

Single-Agent Uncertainty Estimation (Offline Learning)

- To generate outputs from the Bayesian neural network, 50 queries were made for each pair of input coordinates (x,y).

- Subsequently, a visualization was created to illustrate the mean values and standard deviations of the neural network responses.

- The kl_{weight} learning hyperparameter should be correctly “fine-tuned” if we want to distinguish the "hallucinations" of the neural network from areas with sufficient data

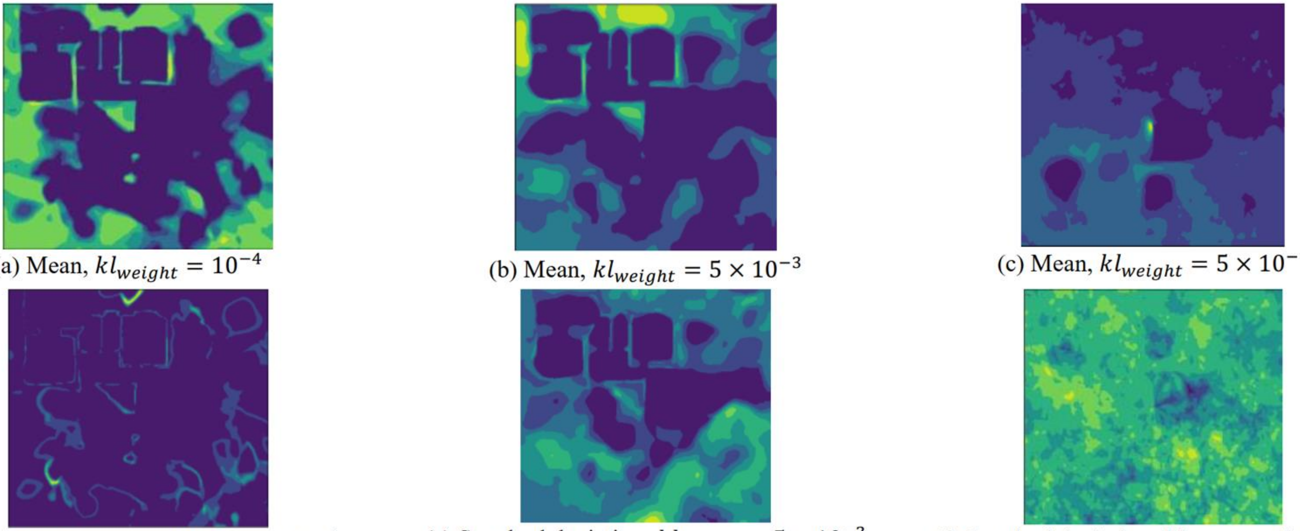


Single forward pass

Mean of 50 forward passes

Standard deviations of 50 forward passes

Environment map



(a) Mean, $kl_{weight} = 10^{-4}$

(b) Mean, $kl_{weight} = 5 \times 10^{-3}$

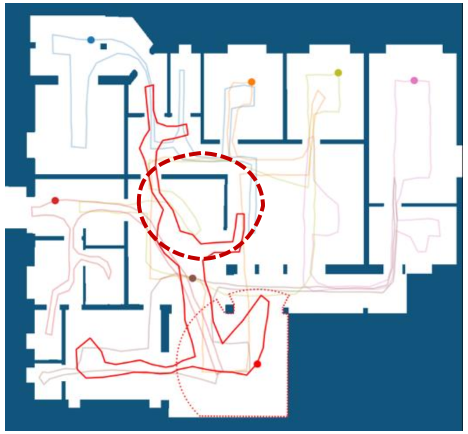
(c) Mean, $kl_{weight} = 5 \times 10^{-1}$

(d) Standard deviation, $kl_{weight} = 10^{-4}$

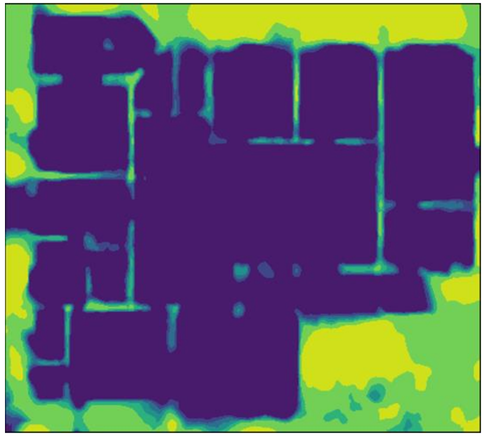
(e) Standard deviation, $kl_{weight} = 5 \times 10^{-3}$

(f) Standard deviation, $kl_{weight} = 5 \times 10^{-1}$

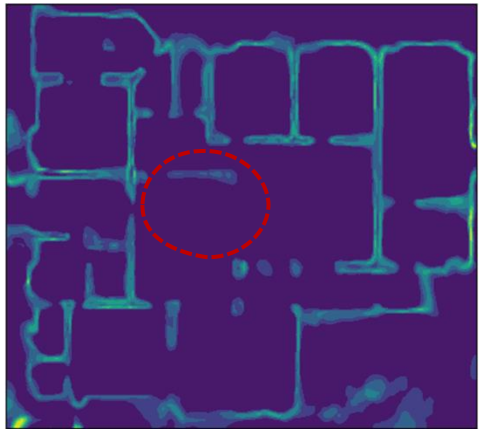
Multi-Agent BNN Learning



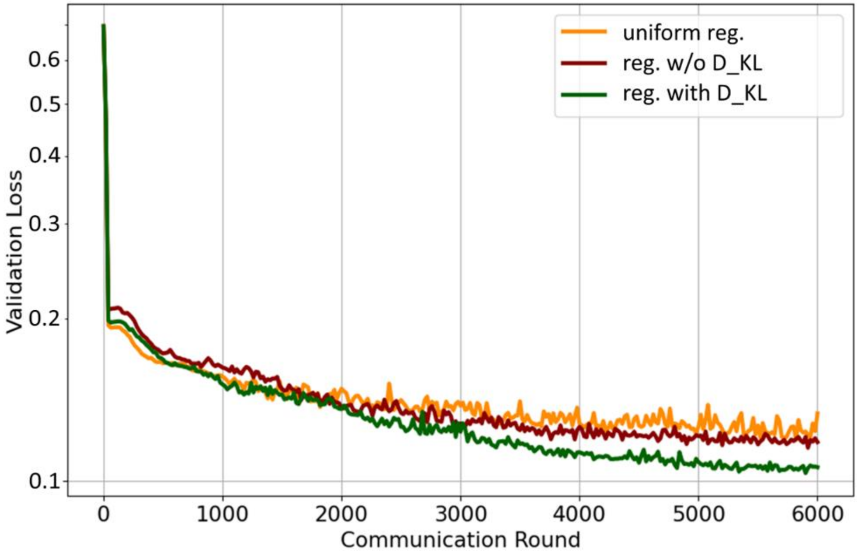
Environment map



Mean



Standard Deviation



- We evaluated the impact of different regularization approaches on the training quality of Bayesian neural networks within the decentralized environment
- We observe that applying Kullback–Leibler divergence for parameter regularization leads to a 12-30% decrease in the validation loss of the distributed BNN training compared to other regularization strategies
- We are currently evaluating applicability of BNN to the case of novelty detection: we evaluate the uncertainty in case if **half** of the agents are browsing the altered map, where the wall in the middle is removed (see the highlighted area)

Future Research

- We are currently exploring how distributed learning with BNNs can be tailored for embedded AI hardware, including such methods as distributed distillation. We also evaluate how BNN could be tailored to identify the novelty or inconsistency in the training data between the nodes

- We plan to compare the efficiently of distributed and decentralized NN training using Federated Learning, ADMM-based and Federated Distillation approaches, in cases of centralized and decentralized environments

- We also explore task management and offloading strategies within the multi-layered fog and hybrid edge-fog-cloud environments to improve computational efficiency and resource utilization

Conclusion

- We addressed a problem of uncertainty estimation within distributed machine learning based on AI-enabled edge devices:
  - we set up a simulation of a collaborative mapping problem using the Webots platform;
  - introduced an epoch-based algorithm to support the decentralized peer-to-peer exchange of NN parameters among agents;
  - integrated distributed uncertainty estimation into our models by applying Bayesian neural networks;
  - evaluated applicability of Cumulative Data Retention (CDR) and Data Refresh (DR) approaches in online learning
- BNNs can effectively support uncertainty estimation in a distributed learning context.
- Applying Kullback–Leibler divergence for parameter regularization resulted in a 12-30% reduction in validation loss during the training of distributed BNNs compared to other regularization strategies.

EDGE AI

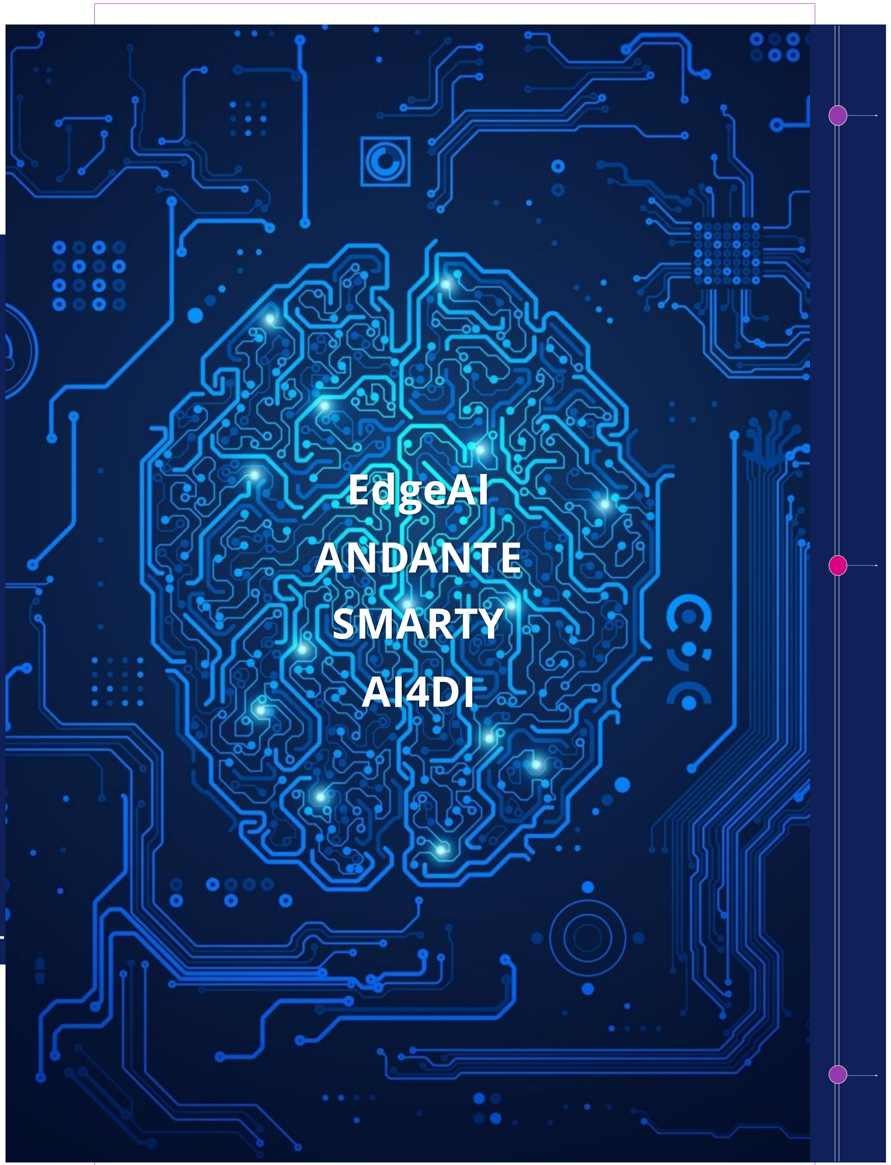
Thank You

For your attention

@ gleb.radchenko@silicon-austria.com

The research reported has been partly funded by the European Union’s Horizon 2020 research and innovation program within the framework of Chips Joint Undertaking (Grant No. 101112268). This work has been supported by Silicon Austria Labs (SAL), owned by the Republic of Austria, the Styrian Business Promotion Agency (SFG), the federal state of Carinthia, the Upper Austrian Research (UAR), and the Austrian Association for the Electric and Electronics Industry (FEEI)

Event Organisers



EdgeAI

ANDANTE

SMARTY

AI4DI

EdgeAI (Edge AI Technologies for Optimised Performance Embedded Processing) develops new electronic components and systems, processing architectures, connectivity, software, algorithms, and middleware through the combination of microelectronics, edge AI, embedded systems, and edge computing. www.edge-ai-tech.eu

The ANDANTE project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876925. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Belgium, France, Germany, The Netherlands, Portugal, Spain, Switzerland. www.andante-ai.eu

SMARTY - Scalable and Quantum Resilient Heterogeneous Edge Computing enabling Trustworthy, focuses on cloud-edge continuum for heterogeneous systems, that protects data-in-transit and data-in-process, employing novel accelerators for quantum resilient communications, confidential computing, and software defined perimeters. https://www.smarty-project.eu/

The AI4DI project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 826060. The JU receives support from the European Union's Horizon 2020 research and innovation programme and the national authorities. www.ai4di.eu

Event Organisers

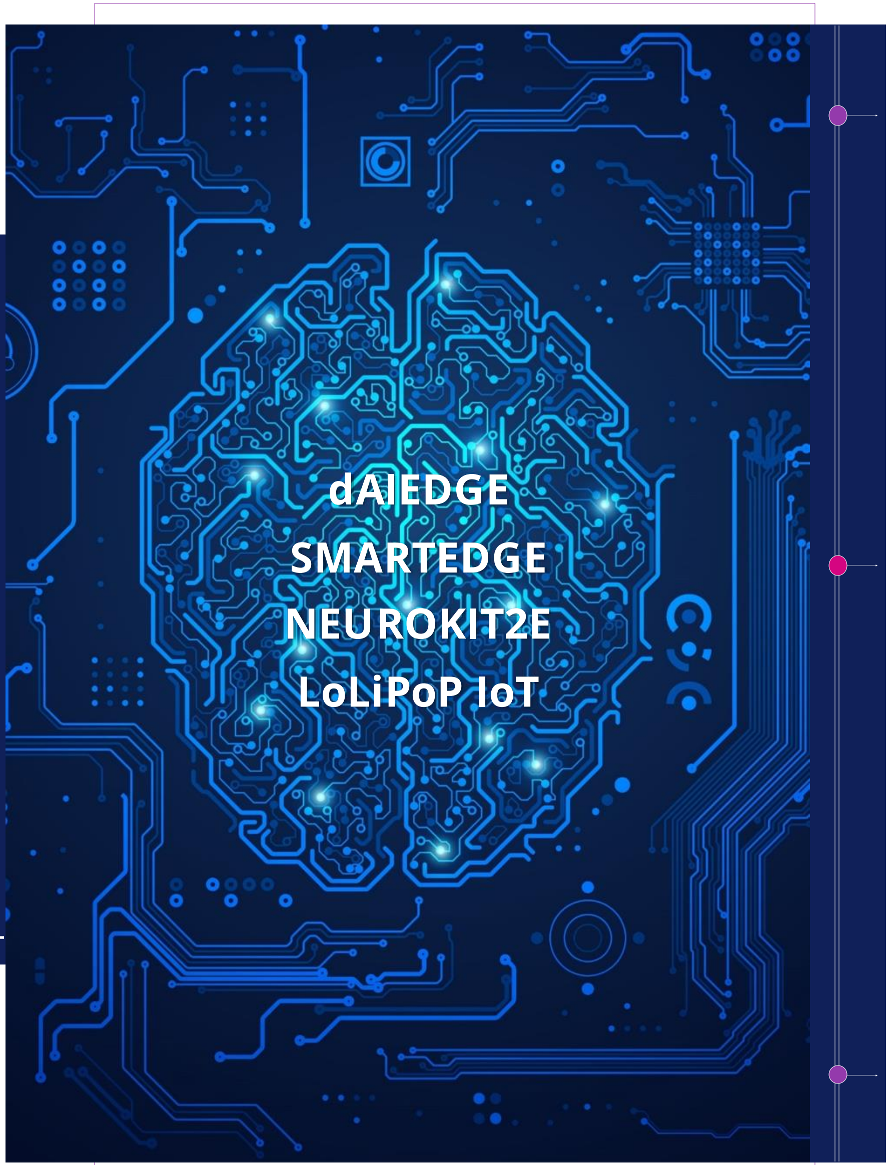
AI4CSM (Automotive Intelligence for Connected Shared Mobility) develops advanced electronic components and systems (ECS) and architectures for future mass-market ECAS vehicles based on sensor fusion and perception platforms, efficient propulsion and energy modules, advanced connectivity and trustworthy AI techniques and methods. www.ai4csm.eu

The ANDANTE project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876925. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Belgium, France, Germany, The Netherlands, Portugal, Spain, Switzerland. www.tristan-project.eu

CLEVER (Collaborative edge-cLoud continuum and Embedded AI for a Visionary industry of thE futuRe) proposes innovations in hardware accelerators, design stack, and middleware software that revolutionize the ability of edge computing platforms to operate federated, leveraging sparse resources that are coordinated to create a powerful swarm of resources. www.cleverproject.eu

REBECCA (Reconfigurable Heterogeneous Highly Parallel Processing Platform for safe and secure AI) aims to democratize the development of edge AI systems and create a complete hardware and software stack centered around a RISC-V CPU, which offer higher performance, energy efficiency, safety, and security than existing systems. www.rebecca-chip.eu

Event Organisers



dAIEDGE

SMARTEDGE

NEUROKIT2E

LoLiPoP IoT

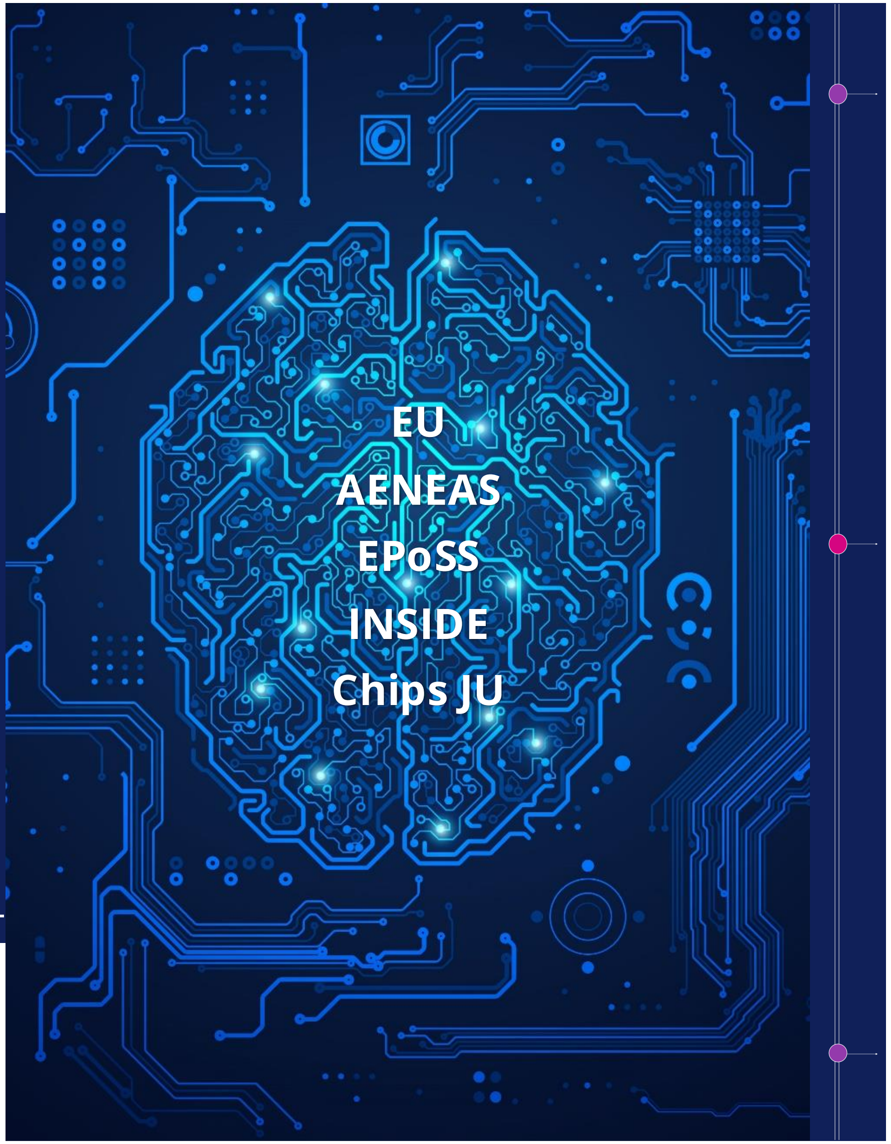
dAIEDGE is the European Network of Excellence for distributed, trustworthy, efficient, and scalable AI at the Edge and promotes the application, development, and deployment of Artificial Intelligence (AI) on edge computing platforms. https://daiedge.eu/

The SmartEdge project aims to achieve dynamic integration of decentralized edge intelligence while prioritizing reliability, security, privacy, and scalability. The SmartEdge solution includes a low-code tool programming environment with three main tools: Continuous Semantic Integration, Dynamic Swarm Network, and Low-code Toolchain for Edge Intelligence. https://www.smart-edge.eu/

NEUROKIT2E (Open source deep learning platform dedicated to Embedded hardware and Europe) proposes a Deep Learning Platform for Embedded Hardware around an established European value chain (AI HW/SW). The solutions developed support neural network design, optimisation, and implementation on constrained HW. https://www.neurokit2e.eu/

The objectives of LoLiPoP IoT (Long Life Power Platforms for Internet of Things) are to develop energy harvesting-based innovative Long Life Power Platforms that enable retrofit of wireless sensor network edge devices for asset tracking, condition and performance monitoring. www.lolipop-iot.eu

Supporting Organizations



EU

AENEAS

EPoSS

INSIDE

Chips JU

AENEAS standing for Association for European NanoElectronics ActivitieS, is an industrial Association, established in 2006, providing unparalleled networking opportunities, policy influence & supported access to funding to all types RD&I participants in the field of micro and nanoelectronics enabled components and systems. https://aeneas-office.org/

The European Technology Platform on Smart Systems Integration is an industry-driven policy initiative, defining research, development and innovation needs as well as policy requirements related to Smart Systems Integration and integrated Micro- and Nanosystems. The main objective is to develop a vision and to set up a Strategic Research Agenda. www.smart-systems-integration.org

Inside Industry Association is the European Technology Platform for research, design and innovation on Intelligent Digital Systems and their applications. The Association is a membership organisation for the European research and innovation actors with more than 200 members and associates from all over Europe. www.inside-association.eu

Chips Joint Undertaking supports research, development, innovation, and future manufacturing capacities in the European semiconductor ecosystem. Launched as part of the Chips for Europe Initiative, it confronts semiconductor shortages and strengthens Europe's digital autonomy, engaging a significant EU, national/regional and private industry funding of nearly €11 billion. https://portal.chips-ju.europa.eu/