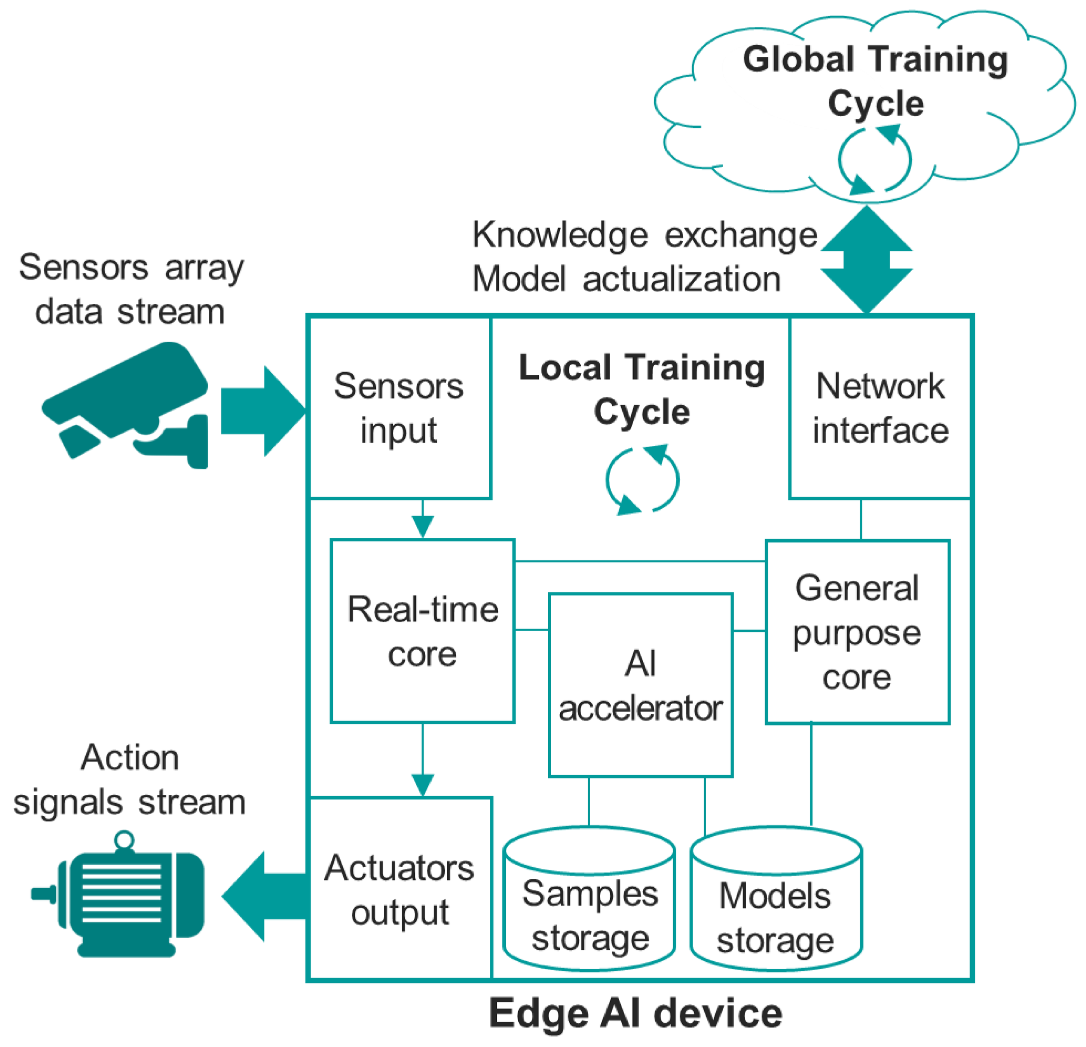
Uncertainty Estimation in Multi-Agent Distributed Learning

Gleb Radchenko
Victoria Andrea Fill

Embedded Systems Division
Silicon Austria Labs
gleb.radchenko@silicon-austria.com

- 6G enables distributed learning among independent edge agents
- We address the issue of collaborative learning of edge AI devices in distributed network environments
- Particularly, to better estimate the uncertainty in collaborative learning results, we are exploring the use of Bayesian Neural Networks in distributed learning frameworks

Distributed Collaborative Learning: Uncertainty Issue



Machine learning is an effective mechanism for **distributed processing** of data streams, providing increased data privacy, scalability and flexibility.
Nonetheless, distributed neural network (NN) training introduces several challenges, including:

- Defining the concept of "knowledge" and establishing protocols for its exchange among Edge AI devices
- Addressing the identification and management of spatial and temporal heterogeneities in the input data.

Consequently, a key task is the quantification of a neural network's **confidence** in the result.

Bayesian neural networks (BNNs) employ a Bayesian approach to train a stochastic neural network. They utilize probabilities, denoted $P(w)$ for weights and $P(b)$ for biases.

The operation of a Bayesian Linear neuron can be described as:

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

To estimate the response, BNNs might conduct multiple forward passes on a single input value. The standard deviation of these outputs are interpreted as the model's uncertainty for each point in the input data space.

Distributed Training

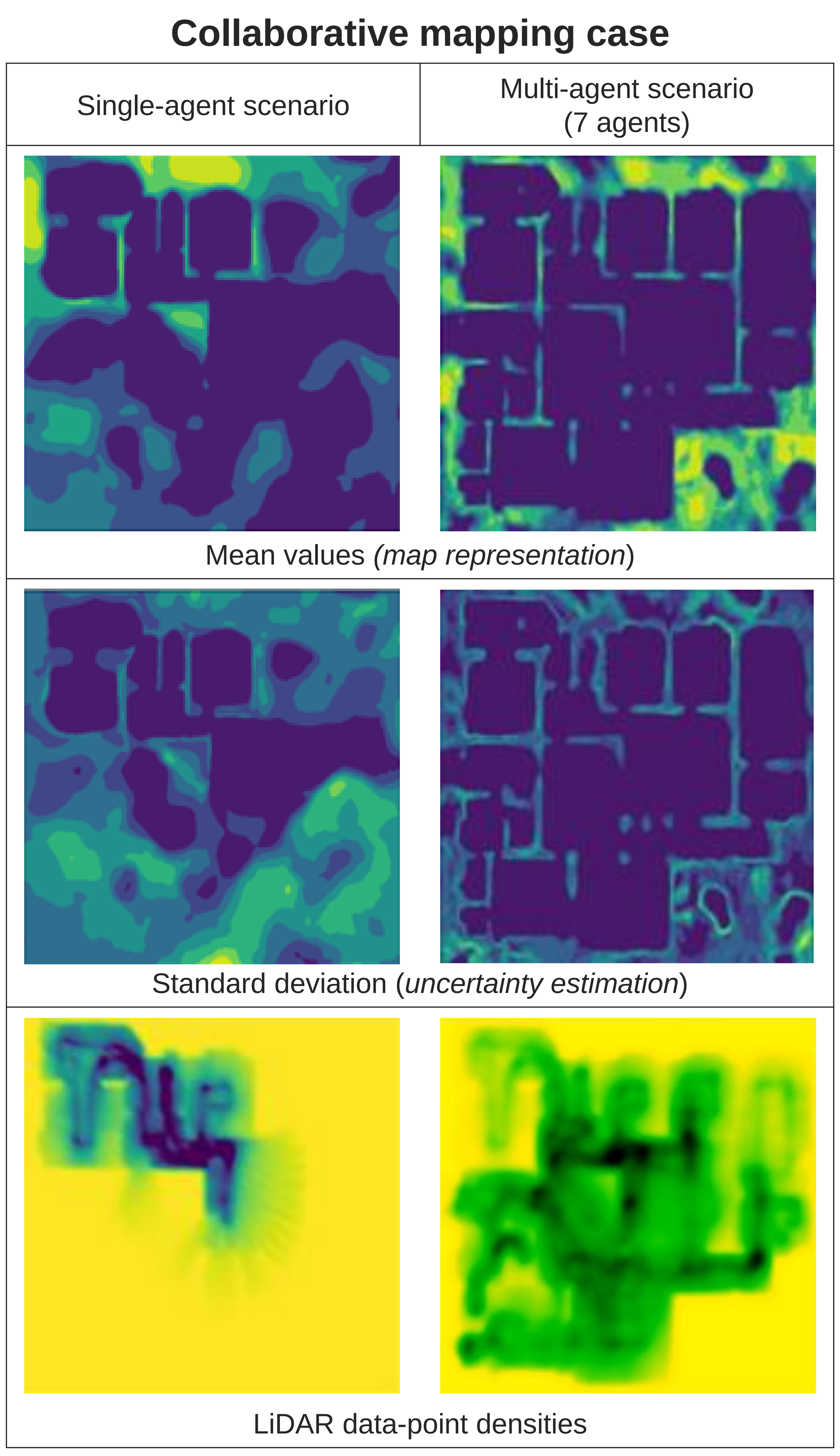
We expand the Distributed Neural Network Optimization (DiNNO) [1] algorithm by tailoring it for compatibility with BNNs.

Algorithm 1 enables asynchronous data interchange during the decentralized training process among autonomous agents.

Algorithm 1 Peers State Exchange

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

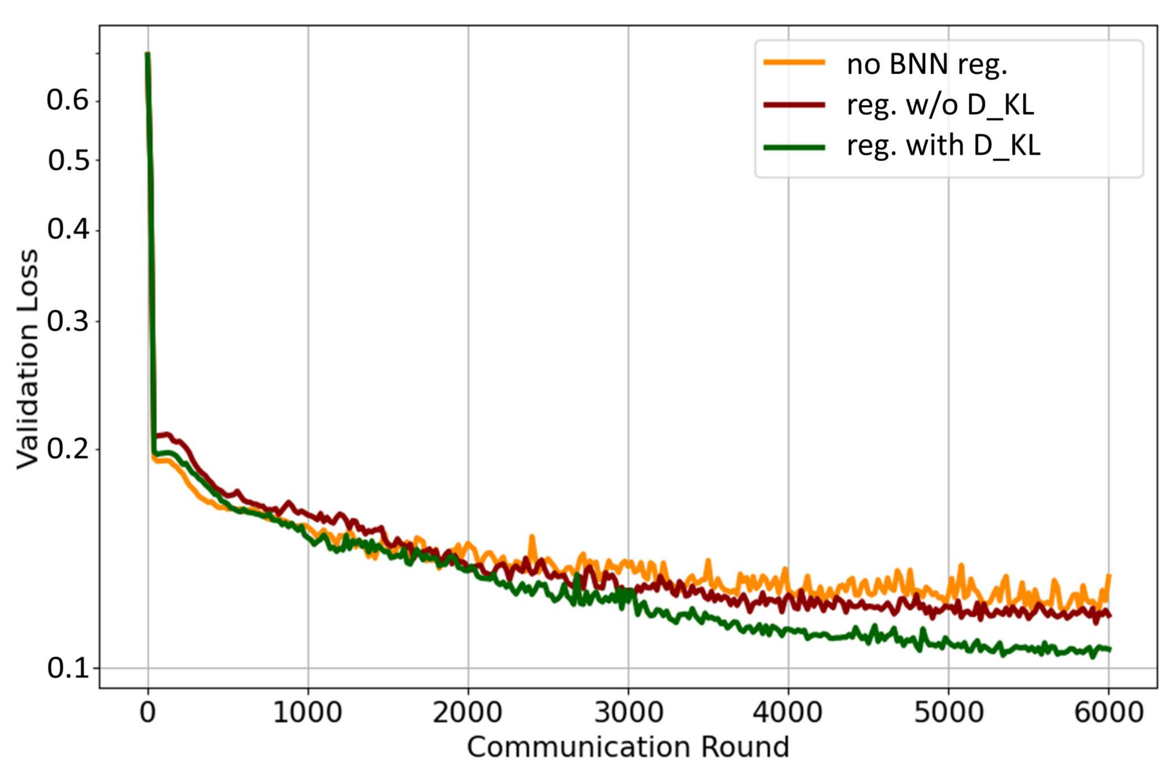
Evaluation of Distributed BNN



Implementation of Kullback–Leibler divergence for the parameter regularization (*Algorithm 2*) provides a 12-30% reduction in distributed BNN validation loss and improves training process stability.

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



Conclusion and Future Work

BNNs can effectively support uncertainty estimation in distributed learning, while considering the required regularization to maintain learning quality.

Future work:

- Refining the BNN approach and NN architecture to suit the resource constraints of edge devices
- Optimizing the network load for edge devices, given the potential of upcoming network infrastructures like 6G

[1] 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.

Funding was provided by the European Union's Horizon 2020 research and innovation program within the framework of Key Digital Technologies Joint Undertaking (Grant No. 101112268).

Silicon Austria Labs GmbH
Sandgasse 34, 8010 Graz, Austria
www.silicon-austria-labs.com
contact@silicon-austria.com