Privacy-preserving federated learning for robust approximate MPC

Joshua Adamek[†] [*] **Janis Adamek**[†] [**] **Moritz Schulze Darup** [**]
Sergio Lucia [*]

[*] *Department of Biochemical and Chemical Engineering, TU Dortmund University, 44227 Dortmund, Germany (e-mail: joshua.adamek@tu-dortmund.de)*
[**] *Department of Mechanical Engineering, TU Dortmund University, 44227 Dortmund, Germany (e-mail: janis.adamek@tu-dortmund.de)*

Abstract: Approximate model predictive control based on imitation learning methods enables real-time implementation of optimal constrained control even for large-scale systems under uncertainty. Training the underlying neural networks often requires large datasets of, which can be challenging for a single process operator to gather. A federated learning scheme, where multiple operators combine smaller datasets, could alleviate this issue. Yet, off-the-shelf federated learning may conflict with privacy requirements of the participants. In this paper, we present a collaborative federated learning scheme for robust approximate model predictive control. To protect data privacy with respect to the central computing server, we integrate homomorphic encryption, allowing for encrypted learning.

Keywords: Model predictive control, Artificial Intelligence and Machine Learning, Process control

1. INTRODUCTION

Model predictive control (MPC) optimally controls nonlinear systems under state and input constraints (Rawlings et al., 2017). Robust MPC methods handle parametric uncertainties to ensure constraint satisfaction despite model mismatches (Campo and Morari, 1987). For nonlinear systems, common approaches include multi-stage MPC (Lucia et al., 2013) and tube-based MPC (Mayne et al., 2011). A key challenge in robust MPC, especially for nonlinear systems, is real-time feasibility, as the computational complexity of the optimization problem grows exponentially with uncertainties and prediction horizon. Explicit MPC (Bemporad et al., 2002) precomputes solutions offline but are impractical for complex nonlinear systems due to storage and computation constraints.

Approximate MPC (AMPC), an approach that has recently attracted much interest, using neural networks to approximate MPC control laws (Chen et al., 2018), (Karg and Lucia, 2020). While AMPC reduces online computation, generating sufficient offline samples for accurate control law approximation is computationally expensive. Another challenge of AMPC is the difficulty to adapt to changes in parameters. Addressing new operating conditions requires either retraining the network or expanding the sampling space to include new parameters, as in Abu-Ali et al. (2022). This paper addresses the challenges of AMPC in settings where multiple process operators (participants) control similar systems under different conditions. Training an approximate controller for all participants is inefficient, as retraining is costly when conditions change. Instead, we propose pooling participant data to create a controller that generalizes across various conditions. However, directly sharing full datasets is inefficient and raises privacy concerns. Federated learning mitigates these issues (Zhang et al., 2021) by training models locally and sharing only model updates (McMahan et al., 2017). To reduce communication overhead, a third-party server aggregates updates, but as shown by Gu et al. (2022), even these updates can reveal underlying data distributions, leading to privacy risks. Privacy-preserving methods such as differential privacy (Dwork, 2008) and homomorphic encryption (HE) (Marcolla et al., 2022) can counteract this. We opt for HE, as it provides stronger confidentiality guarantees than differential privacy. While Xu and Wu (2024) explored privacy-preserving federated learning for MPC, their approach focuses on data communication between subsystems and approximates the model rather than the MPC law itself. Our key contribution is the first encrypted federated learning algorithm for training parametric robust MPC controllers. Figure 1 illustrates the setup, featuring multiple approximate controllers and encrypted federated learning. This method improves controller performance across parameter sets without resampling while preserving data privacy, enabling scalable robust MPC deployment. The remainder of this paper is structured as follows:

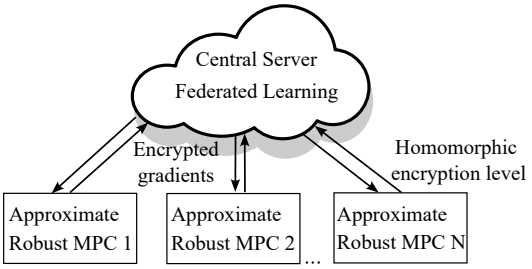

Fig. 1. Overview of privacy-preserving federated learning for approximate MPC. The central server updates encrypted gradients from each local network, enabling training across all approximate MPC networks.

[⋆] Financial support by the German Research Foundation (DFG) under the grant SCHU 2940/5-1 is gratefully acknowledged.
[†] Both authors contributed equally to this work

The next section revisits parametric robust multi-stage MPC approximation using neural networks. After introducing federated learning and HE, we present our privacy-preserving federated learning algorithm. We then evaluate the approach on a nonlinear case study, demonstrating functionality and sampling efficiency. Finally, we conclude with future research directions.

2. PARAMETRIC ROBUST APPROXIMATE MPC

2.1 Robust multi-stage MPC

This Section summarizes the main idea of multi-stage nonlinear MPC as described by Lucia et al. (2013) with the addition of changing but known process parameters c. Within this framework, uncertainties are handled by a tree of discrete scenarios, where each scenario reflects a concrete uncertainty realization. In this work, we consider the following discrete-time nonlinear system

$$x_{k+1}^j = f(x_k^{p(j)}, u_k^j, d_k^{r(j)}, c_k), \tag{1}$$

where the next state $x_{k+1}^j \in \mathbb{R}^{n_x}$ for each scenario j is computed as a nonlinear function of the parent state $x_k^{p(j)}$, the input vector u_k^j, its respective realization r of the uncertainty $d_k^{r(j)} \in \mathbb{R}^{n_d}$ and parameters c_k such as process conditions that can be changed but are known. The set of indices (j,k) in the scenario tree is denoted by I and S_i denotes the i-th scenario which is the path from the root node x_0 to one of the leaf nodes. A reasonable strategy to build a scenario tree is to consider as branches all possible combinations of the extreme and nominal values of the parameters assumed to be uncertain. The number of combinations grows therefore exponentially with the number of uncertainty parameters. Furthermore, the combination of all uncertainty realizations leads to an exponential growth of the scenario tree with the prediction horizon N_{pred} of the MPC $N_{\text{scen}} = n_d^{N_{\text{pred}}}$. Typically the scenario tree is only branched up to a certain horizon, called robust horizon N_R, which is much smaller than the prediction horizon to limit the exponential growth. After the robust horizon, uncertainties are considered fixed. Then the multi-stage optimization problem can be formulated as

$$\min_{x_k^j, u_k^j, \forall (j,k) \in I} \sum_{i=1}^{N_{\text{scen}}} \omega_i J_i(X_i, U_i) \tag{2a}$$

subject to:

$$x_{k+1}^j = f\left(x_k^{p(j)}, u_k^j, d_k^{r(j)}, c_k\right), \qquad \forall (j,k+1) \in I, \tag{2b}$$

$$0 \geq g\left(x_{k+1}^j, u_k^j, d_k^{r(j)}, c_k\right), \qquad \forall (j,k) \in I, \tag{2c}$$

$$u_k^j = u_k^l \text{ if } x_k^{p(j)} = x_k^{p(l)}, \qquad \forall (j,k),(l,k) \in I, \tag{2d}$$

where X_i, U_i are the set of states and control inputs that belong in the scenario S_i with the probability of occurrence ω_i. State and input constraints are denoted by $g(\cdot)$. Note the non-anticipativity constraints in (2d), as the control inputs cannot anticipate the current realization of the uncertainty. The cost of each scenario $J_i(\cdot)$ is the summation of the cost function value $L(\cdot)$ for each predicted timestep in the respective scenario with a terminal cost $T_i(\cdot)$

$$J_i(X_i, U_i) = \sum_{k=0}^{N_{\text{pred}}-1} L(x_k^j, u_k^j, c_k) + T(x_{N_{\text{pred}}}), \quad \forall x_k^j, u_k^j \in S_i. \tag{3}$$

2.2 Parametric approximate MPC

The first input $u_0^\star$ of an MPC optimization problem that is applied to the system as the control policy Π_{MPC} can be regarded as a nonlinear function over the current state x_0 and previous input u_{-1}. Approximate MPC suffers from the fact that the solution is only optimal for a fixed set of parameters c_0 of the model. We therefore explicitly consider the process parameters, which could change during the online application of the controller, as inputs of the control policy:

$$\Pi_{\text{MPC}} = u_0^\star(x_0, u_{-1}, c_0), \tag{4}$$

where $u_0^\star$ is the first input value of the solution of equation (2). Note that the uncertainty realizations for the uncertain parameters d_0 are fixed and therefore not part of the function space. We define the approximate MPC controller as a feedforward neural network, parameterized by a number of layers l, the number of neurons m, a nonlinear activation function, and its network parameters θ. The control law

$$\Pi_{\text{approx}} = \mathcal{N}(x_0, u_{-1}, c_0; \theta), \tag{5}$$

is generated by imitating Π_{MPC} on a dataset

$$\mathcal{D} = \{\{x_j, u_{j-1}, c_j\} \to \{u_j^\star\} | j \in \{0, N_{\text{data}}\}\}. \tag{6}$$

This dataset is sampled by offline computation of (2). The training of the neural network parameters θ is done by minimizing the mean squared error between $\Pi_{\text{approx}}(\theta)$ and Π_{MPC} on the dataset

$$\min_\theta \frac{1}{N_{\text{data}}} \sum_{j=0}^{N_{\text{data}}} |\Pi_{\text{MPC}}(x_j, u_{j-1}, c_j) - \Pi_{\text{approx}}(x_j, u_{j-1}, c_j; \theta)|^2. \tag{7}$$

3. BACKGROUND ON PRIVACY IN COLLABORATIVE LEARNING

3.1 Basics of federated learning

Federated learning is a collaborative machine learning algorithm designed for training a deep neural network on N distributed datasets. We consider N datasets of input-output data pairs $\{z_j^i, y_j^i\}$ for $i = 1 \ldots N$ and $j = 1 \ldots K_i$ with K_i being the local dataset sizes. The machine learning task is to jointly train a deep neural network $\mathcal{N}(z_j^i; \theta)$ to match the labels y_j^i over all datasets by finding

$$\theta = \text{argmin} \frac{1}{\sum_{i=1}^N K_i} \sum_{i=1}^N \sum_{j=1}^{K_i} \ell(z_j^i, y_j^i), \tag{8}$$

for some loss function $\ell(\cdot)$. To prevent the exchange of local data, the participants perform gradient updates on their local data, which are then aggregated to a central update step at a third-party server. The most general framework for describing federated learning algorithms is the FedOpt framework, presented by Reddi et al. (2020), which we extended by using the ideas of Li et al. (2019) for non-uniform dataset sizes. In communication round r, a subset $\mathcal{P}_r$ of participants is used to update the global model weights θ_r stored at the server. The participants do a batch gradient descent of E epochs in which B batches of data, expressed by the set $\mathcal{B}_{r,e,b}$ (the lower indexes representing the communication round, epoch, and batch of the dataset), are drawn. To update the local parameters, which at the beginning of each iteration are set to the global model weights, the participants use the the function

$$\theta_{r,e,b+1} = f_{\text{Participant}}(\theta_{r,e,b}^i, g_{r,e,b}^i) \tag{9}$$

based on the old weights and the new gradient $g^i_{r,e,b}$ from the drawn batch. This function could include a learning rate scheduling based on previous gradients, as in Adam (Kingma and Ba, 2014) and AdamW (Loshchilov, 2017) optimizers. For simplicity, we omit the updates in the internal parameters of the optimizer function in our notation. Afterward, the difference to the global model

$$\Delta\theta^i_r = \theta^i_{r,E+1,B+1} - \theta_r \tag{10}$$

is uploaded to the server, which aggregates the weight differences to a pseudo-gradient

$$\Delta\theta_r = \frac{N}{|\mathcal{P}_r|} \sum_{i \in \mathcal{P}_r} \frac{K_i}{\sum_{i=1}^N K_i} \Delta\theta^i_r. \tag{11}$$

This pseudo-gradient includes the information of all datasets from participants in $\mathcal{P}_r$ and is subsequently used to update the model parameters with the server update function

$$\theta_{r+1} = f_{\text{Server}}(\theta_r, \Delta\theta_r) \tag{12}$$

in which again internal parameters are not included for readability.

3.2 Essentials on homomorphic encryption

A method for achieving privacy in collaborative machine learning is homomorphic encryption (HE), which allows arithmetic operations on encrypted data, a feature that is typically not available for standard cryptosystems. As with any public key cryptosystem, an HE scheme defines a public key $\mathtt{pk}$ and an encryption primitive $\mathtt{ct}(a) = \mathsf{Enc}(a, \mathtt{pk})$ to encrypt a message a into a ciphertext $\mathtt{ct}(a)$. Only with the possession of the secret key, the message can be recovered using the decryption primitive $a = \mathsf{Dec}(\mathtt{ct}(a), \mathtt{sk})$. All proper HE schemes present confidentiality guarantees of ciphertext messages and ensure the hardness of inferring the secret key from the public key, such that the public key can be made publicly available.

Different from standard encryption, partially HE methods like the Paillier cryptosystem (Paillier, 1999) define operations $\oplus$ and $\odot$ on ciphertexts that are equal to additions of the ciphertexts $\mathtt{ct}(a_1)$ and $\mathtt{ct}(a_2)$ and multiplications with a plaintext scalar s

$$\begin{aligned} \mathsf{Dec}(\mathtt{ct}(a_1) \oplus \mathtt{ct}(a_2), \mathtt{sk}) &= a_1 + a_2 \\ \mathsf{Dec}(\mathtt{ct}(s) \odot \mathtt{ct}(a_1), \mathtt{sk}) &= s \cdot a_1. \end{aligned}$$

We can therefore simply write

$$\mathtt{ct}(a_1 + a_2) = \mathtt{ct}(a_1) \oplus \mathtt{ct}(a_2), \quad \mathtt{ct}(s \cdot a_1) = s \odot \mathtt{ct}(a_1).$$

Since the Paillier cryptosystem has a message space containing integer numbers from a finite field, we will turn to the more powerful fully HE method CKKS (Cheon et al., 2017). It uses a message space $v \in \mathbb{R}^n$ and further defines an elementwise multiplication operation

$$\mathtt{ct}(v_1) \otimes \mathtt{ct}(v_2) = v_1 v_2$$

on the two ciphertexts corresponding to $v_1 \in \mathbb{R}^n$ and $v_2 \in \mathbb{R}^n$. As updates and model parameters of neural networks are large vectors, the implementation of privacy-preserved collaborative learning is according to Ma et al. (2022) significantly faster using CKKS. This is because multiple elements can be processed in parallel as they are encrypted in a single ciphertext. In addition, the cryptosystem also provides the possibility for additional privacy benefits.

Algorithm 1 Privacy-preserving federated learning algorithm

Require: Initial parameters θ_1
  for $r = 1 \dots r_{\max}$ **do**
    for participant $i = 1 \dots N$ in parallel **do**
      $\theta^i_{r,1,1} = \theta_r$
      for $e = 1 \dots E$ **do**
        for $b = 1 \dots B$ **do**
          $g^i_{r,e,b} = \nabla \frac{1}{|\mathcal{B}^i_{r,e,b}|} \sum_{(z^i_j, y^i_j) \in \mathcal{B}^i_{r,e,b}} \ell(z^i_j, y^i_j)$
          $\theta^i_{r,e,b+1} = f_{\text{Participant}}(\theta^i_{r,e,b}, g^i_{r,e,b})$
        $\theta^i_{r,e+1,1} = \theta^i_{r,e,B+1}$
      $\Delta\theta^i_r = \theta^i_{r,E+1,B+1} - \theta_r$
      Encrypt $\mathtt{ct}(\Delta\theta^i_r)$ and send to server
    Server: $\mathtt{ct}(\Delta\theta_r) = \bigoplus_{i=1}^N \left(\frac{K_i}{\sum_{i=1}^N K_i} \odot \mathtt{ct}(\Delta\theta^i_r) \right)$
    for participant $i = 1 \dots N$ in parallel **do**
      Obtain $\mathtt{ct}(\Delta\theta^i_r)$ from server and decrypt
      $\theta_{r+1} = \theta_r + \Delta\theta_r$

4. PRIVACY-PRESERVING FEDERATED LEARNING FOR APPROXIMATE MPC

4.1 Federated learning for parameterized AMPC tasks

In the following, we present our proposed approach to apply privacy-preserving federated learning to learn robust approximate MPC controllers. We consider a scenario where N participants control different instances of the same process using an approximation of a robust MPC. Small differences between the processes can be dealt with as uncertainties within the robust MPC design. Typically, each participant would sample data locally and train its neural network only on this data. However, this local approach is only reasonable for smaller systems where the sampling effort is manageable and is furthermore only applicable for a fixed set of operating parameters c.

To enable the potential use of approximate MPC for large-scale systems and different operating parameters, there is a need to cooperate to use the computation resources of all participants. As sharing the sampled data is inefficient and can cause privacy concerns, we propose the following privacy-preserving federated learning method. In our approach, the gradients of the local training are exchanged and then encrypted. The update step on the central server is computed only on the encrypted gradients. These updates are passed back to the local participants, where they get decrypted and the local approximation updated. This enables privacy-preserving learning on the entire sample dataset such that each participant can use its obtained solution on the entire parameter set.

In the following, we adapt the principle federated learning algorithm of Reddi et al. (2020) as explained in Subsection 3.1 to the new algorithm presented in Algorithm 1, which incorporates privacy through HE. The equations (7) and (8) show that federated learning is applicable in the case of distributed parameterized robust approximate MPC datasets. The inputs to the neural network are $z_k = \{x_k, u_{k-1}, c_k\}$ and the outputs (labels during training) are the corresponding optimal inputs computed via MPC $y_k = \Pi_{\text{MPC}}(z_k)$.

First, we assume an industrial-type cooperation with few and reliant participants but a strong need for privacy of the exchanged information. Therefore, we do not sample subsets of

the participants such that $\mathcal{P}_r$ is always the full number of participants and subsequently grant all participants access to the global model weights in all communication rounds. As can be seen in Algorithm 1, the local update step of each participant updating its own local parameter set $\theta^i_{r,E+1,B+1}$ as well as the difference to the global parameter set $\Delta\theta^i_r$ remains as described in (9) and (10). The fundamental difference is the encryption of said difference using HE such that $\mathtt{ct}(\Delta^i_r)$ is sent to the central server. Therefore, the calculation of the weighted sum as in eq. 11 will be done on the encrypted differences

$$\mathtt{ct}(\Delta\theta_r) = \bigoplus_{i=1}^{N} \left(\frac{K_i}{\sum_{i=1}^{N} K_i} \odot \mathtt{ct}(\Delta\theta^i_r) \right). \tag{13}$$

Note that different from typical federated learning where the update of the global parameter θ_r from this weighted sum is calculated on the server as in (12), this encrypted weighted sum is sent back to the participants where each one decrypts the value and updates the global parameter set by simply adding the weighted sum $\theta_{r+1} = \theta_r + \Delta\theta_r$. The second major modification to the federated learning approach of Reddi et al. (2020) is a more complex update rule $f_{\text{Participant}}(\cdot)$. We now use the effectiveness of batch gradient descent with adaptive solvers like Adam (Kingma and Ba, 2014) or AdamW (Loshchilov, 2017), which is used for $f_{\text{Server}}(\cdot)$ in the original algorithm.

Both steps identify the weighted sum as the central need for exchanging the user information and therefore being the only operation that has to be calculated in an encrypted fashion at the server. The reasoning is that we avoid very long iterative calculations on the encrypted values on the central server, which are hard for HE schemes due to the limited amount of successive multiplications (see (Marcolla et al., 2022) for more details). By shifting the more complex gradient update to $f_{\text{Participant}}(\cdot)$, we can express all HE operations by multiplications and additions.

4.2 Establishing privacy

We model all participants and the server as semi-honest, such that they will honestly follow the algorithm but try to infer as much information as possible. To obtain the privacy of the gradient updates against the server, we rely on the confidentiality guarantees of the CKKS HE scheme. In Algorithm 1, the weights of the different gradients are available in plaintext to the server. To keep the sizes of the datasets private, the participants can exchange this side information beforehand and only send the encrypted weights to the server, which then uses the $\otimes$ operation instead of $\odot$. The CKKS secret key is generated by one of the participants and shared with the other participants. Since every participant would be able to decrypt the messages of other participants, we need another symmetric encryption like AES to encrypt all communication with the central server to achieve privacy towards the other participants. Since we model all agents as semi-honest we assume no exchange of the secret key between the server and the participants. However, this cooperation can be counteracted by the use of a threshold variant of CKKS like it is implemented in the OpenFHE (Al Badawi et al., 2022) library.

5. CASE STUDY

5.1 Nonlinear robust MPC of a CSTR

We test the algorithm described in Section 3 on the case study of a parametric robust approximate MPC applied to the continuous stirred tank reactor described by Klatt and Engell (1998). We will allow to adapt the temperature of the inlet T_{in} that is fixed during one operation but could change in different operation settings. The system dynamics for the four states c_A (concentration of substance A), c_B (concentration of substance B), T_R (Temperature of the reactor), and T_K (Outlet temperature of the coolant) can be described as:

$$\begin{aligned}
\frac{dc_{\text{A}}}{dt} &= F\left(c_{\text{A},0} - c_{\text{A}}\right) - k_1 c_{\text{A}} - k_3 c_{\text{A}}^2, \quad \frac{dc_{\text{B}}}{dt} = -F c_{\text{B}} + k_1 c_{\text{A}} - k_2 c_{\text{B}}, \\
\frac{dT_{\text{R}}}{dt} &= F\left(T_{\text{in}} - T_{\text{R}}\right) + \frac{k_{\text{W}} A_{\text{R}}}{\rho C_{p,\text{R}} V_{\text{R}}}\left(T_{\text{K}} - T_{\text{R}}\right), \\
&\quad - \frac{k_1 c_{\text{A}} \Delta H_{R,1} + k_2 c_{\text{B}} \Delta H_{R,2} + k_3 c_{\text{A}}^2 \Delta H_{R,3}}{\rho C_{p,\text{R}}} \\
\frac{dT_{\text{K}}}{dt} &= \frac{\dot{Q} + k_{\text{W}} A_{\text{R}}\left(T_{\text{R}} - T_{\text{K}}\right)}{m_{\text{K}} C_{p,\,\text{K}}}, \\
k_1 &= \beta k_{0,bc} \exp\left(\frac{-E_{A,bc}}{R\left(T_{\text{R}} + T_0\right)}\right), \; T_0 = 273.15\,\text{K} \\
k_2 &= k_{0,ab} \exp\left(\frac{-E_{A,ab}}{R\left(T_{\text{R}} + T_0\right)}\right), \; k_3 = k_{0,ad} \exp\left(\frac{-\alpha E_{A,ad}}{R\left(T_{\text{R}} + T_0\right)}\right),
\end{aligned}$$

where the cooling power $\dot{Q}$ and the dilution rate F are the input variables. The values of the constant parameters of the system model are the same as in (Klatt and Engell, 1998). The parameters α and β are considered uncertain.

Therefore, a robust MPC is designed that aims to control the concentration to the setpoint $c_a = 0.7$ and $c_b = 0.6$ without violating the state constraints. The cost for each scenario as in (3) can be described as:

$$L(x_k, u_k) = (x_k - \bar{x})^T Q (x_k - \bar{x}) + \Delta u_k^T R + \Delta u_k, \tag{14}$$

with $Q = \text{diag}(1,1,0,0)$, $\bar{x} = (0.7, 0.6, 0, 0)^T$, $R = \text{diag}(0.1, 0.001)$.

The state and input constraints can be found in Table 1. This table also shows the range for the two uncertain parameters α and β as well as the range of possible values of the known parameter T_{in} that can be changed during the runtime of the reactor. The prediction horizon of the robust MPC is $N_{\text{pred}} = 20$, and the robust horizon is $N_R = 1$. For the MPC, the continuous dynamics are discretized using orthogonal collocation, with $\Delta t = 0.005\,\text{h}$. The case study is implemented in the do-mpc (Fiedler et al., 2023) framework and the code is openly available [1].

Table 1. Lower (LB) and upper bounds (UB) for states, inputs, and variable parameters.

	c_A	c_B	T_R	T_K	$\dot{Q}$	F	α	β	T_{in}
LB	$0.1\,\text{molL}^{-1}$	0.1	50 °C	50 °C	−8500	5	0.95	0.9	125
UB	$2\,\text{molL}^{-1}$	2	140 °C	140 °C	0	100	1.05	1.1	135

5.2 Federated learning results

We compare the federated learning approach against the training on local datasets. The dataset is sampled as closed loop trajectories with random trajectory starting points within the state space and a trajectory length of $N_{\text{traj}} = 20$. We assume $N_{\text{part}} = 6$ participants that each sample 1000 random trajectories with different fixed values of T_{in} for each participant. As some trajectories will be infeasible, each local training dataset $\mathcal{D}_i$ will

[1] https://github.com/JoshuaAda/2024_privacy_federated_learning_approx_mpc

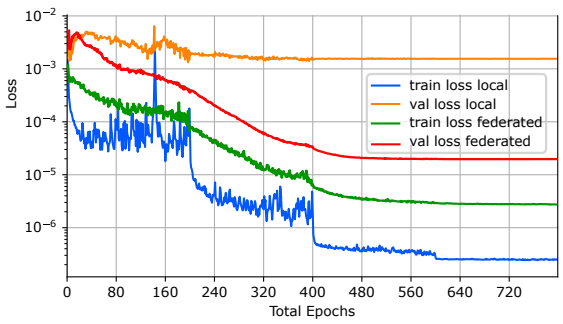

Fig. 2. Training and validation loss for the local approach trained on a dataset with fixed $T_{\text{in}} = 125\,°\text{C}$ in comparison to the encrypted federated training and validation loss.

contain $K_i = 8000$ sampling values. All other sampling values are collected in the validation dataset (21880 sampling values), which is then validating the loss for different values of T_{in}. For a simpler training of the neural networks on the datasets, a box-based scaling using the state, input, and parameter bounds is performed. While the privacy-preserving federated learning algorithm combines the information of all datasets $\mathcal{D}_i$ in the way it is described in Section 3, the local training for participant i is only based on $\mathcal{D}_i$. For both the local as well as the federated approach, the neural networks are initialized with the same weights for three fully connected layers with 500 neurons. The training is done within the Pytorch library using the AdamW solver (Loshchilov, 2017) with a learning rate scheduling (dropping the learning rate by a factor of 10 every 200 epoch). This training corresponds to the local update step $f_{\text{Participant}}(\cdot)$ in Algorithm 1. We use the OpenFHE (Al Badawi et al., 2022) library for the CKKS implementation. The training and validation loss for the approximate MPC training can be found in Figure 2 both for one of the local training and for one of the federated learning networks. While the training loss of the local networks reaches in general a lower value than the federated approach, the validation loss remains high. This is expected, since changes in T_{in} are not accounted for. For the federated approach, the validation loss is related to the training loss as both decrease during the training.

To incorporate the information of all participants in each local controller, every participant would have to extend their dataset by sampling the whole dataset separately, which needs roughly 5 hours of additional computational time on an Intel i7 processor. Since the federated learning approach parallelizes the local updates, it is almost as fast the compared centralized training on the concatenated dataset. The training for both approaches took around 15 minutes on an NVIDIA Geforce RTX 3050 Ti GPU. Therefore, our method provides significant computational advantages even in this simple example with only one process parameter. For this reason, we believe that our proposed methodology is a promising direction to achieve large-scale approximate MPC.

5.3 Closed-loop performance evaluation

We evaluate the local and federated approach in a closed-loop application of the resulting controllers on the system. We consider 100 closed loop trajectories of length 0.2 h with different values of T_{in} and random uncertainty realizations within the described bounds in Table 1. We evaluate the mean tracking cost (MTC), which is the sum of the difference between the

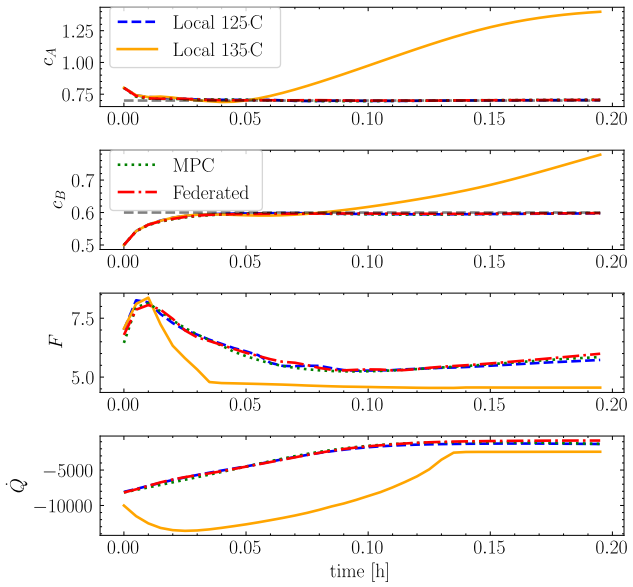

Fig. 3. Concentration and input trajectories for one random initial state vector with a trajectory length of 0.2 h and $T_{\text{in}} = 125\,°\text{C}$. In the figure the closed-loop trajectories are compared against each other for the exact MPC, the federated approach, and the local approaches trained on a dataset with $T_{\text{in}} = 125\,°\text{C}$ and $T_{\text{in}} = 135\,°\text{C}$. The black line represents the desired concentrations to be tracked.

Table 2. Mean tracking cost (MTC), the relative frequency of constraint violations (RFV), percentage deviation from the constraints (PDV) over 100 trajectories for all six locally trained networks, the federated approach, and the exact MPC.

Network	MTC [−]	RFV states [%]	PDV state[%]
Local $T_{\text{in}} = 125\,°\text{C}$	0.103	4.31	4.82
Local $T_{\text{in}} = 127\,°\text{C}$	1.22	2.93	7.24
Local $T_{\text{in}} = 129\,°\text{C}$	0.394	1.88	4.14
Local $T_{\text{in}} = 131\,°\text{C}$	0.598	0.22	1.18
Local $T_{\text{in}} = 133\,°\text{C}$	0.118	0.00	0.00
Local $T_{\text{in}} = 135\,°\text{C}$	2.90	0.00	0.00
Federated	0.031	0.005	1.03
Exact MPC	0.025	0.00	0.00

actual concentration and the setpoint throughout the trajectory. Furthermore, state constraints violations are evaluated with the relative frequency of occurrence (RFV) as well as the percentage deviation from the constraints (PDV) defined as

$$\text{PDV} = \begin{cases} \frac{x_k - x_{\text{ub}}}{x_{\text{ub}} - x_{\text{lb}}}, & \text{if } x_k > x_{\text{ub}} \\ \frac{x_{\text{lb}} - x_k}{x_{\text{ub}} - x_{\text{lb}}}, & \text{if } x_k < x_{\text{lb}} \end{cases} \tag{15}$$

with x_{lb} and x_{ub} being the lower and upper bound for the respective state. Table 2 shows the results of the mean tracking cost over all scenarios. The tracking cost of the federated approach is close to the cost of the exact MPC, indicating that it is a valid approximation of the controller. The tracking costs of all six locally trained approximations are much higher, showing that for this case study, it is indeed necessary to train on the entire dataset. Furthermore, this table shows the relative number of constraint violations and the mean percentage deviation to the constraint bound for state constraints. While there exist few state constraint violations for the federated approach because of the approximate nature of the controller, the mean percentage deviation from the constraint is small in comparison to the

local approximations. For some of the local approaches, the number of constraint violations is much larger. For the local networks with $T_{\text{in}} = 133\,^{\circ}\text{C}$ and $T_{\text{in}} = 135\,^{\circ}\text{C}$, no constraint violations occur but the resulting performance is very conservative when compared to the exact MPC or the proposed federated approach. One exemplary trajectory for this behavior can be seen in Figure 3. Figure 3 shows the example trajectory for $x_0 = (0.8, 0.5134.14, 130)^T$ and $T_{\text{in}} = 125\,^{\circ}\text{C}$ for the federated approach, the local approach trained on a dataset with $T_{\text{in}} = 125\,^{\circ}\text{C}$ and $T_{\text{in}} = 135\,^{\circ}\text{C}$ as well as the exact robust MPC in comparison. As expected, the performance of the local approach trained on the correct T_{in} is satisfactory with a tracking cost of 0.0155, while the tracking cost for the local approximation on a different training set is much worse with a tracking cost of 5.59. Additionally, input constraint violations for the cooling power can be observed. In contrast, the federated approach has an tracking cost of 0.0158 on this trajectory, which is similar to the tracking cost of the exact MPC (0.0153) with almost similar control actions, showing the validity of the approach.

6. CONCLUSION AND OUTLOOK

We presented the first distributed machine learning algorithm for approximate model predictive control tasks to decrease the sampling burden of the participants and to build upon existing datasets and solutions. To ensure privacy of the local data, we develop a privacy-preserving federated learning algorithm based on homomorphic encryption. We show the advantages of the method by reducing the sampling time for each participant as well as a better approximation of the control law for the federated approach in comparison to training only on a local dataset. Future work will extend our method to real-world scenarios in which sampling efficiency and privacy are essential.

REFERENCES

Abu-Ali, M., Berkel, F., Manderla, M., et al. (2022). Deep Learning-Based Long-Horizon MPC: Robust, High Performing, and Computationally Efficient Control for PMSM Drives. *IEEE Transactions on Power Electronics*, 37(10), 12486–12501.

Al Badawi, A., Bates, J., Bergamaschi, F., et al. (2022). Openfhe: Open-source fully homomorphic encryption library. In *Proceedings of the 10th Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, 53–63.

Bemporad, A., Borrelli, F., Morari, M., et al. (2002). Model predictive control based on linear programming˜ the explicit solution. *IEEE transactions on automatic control*, 47(12), 1974–1985.

Campo, P.J. and Morari, M. (1987). Robust model predictive control. In *1987 American Control Conference*, 1021–1026.

Chen, S., Saulnier, K., Atanasov, N., Lee, D.D., Kumar, V., Pappas, G.J., and Morari, M. (2018). Approximating Explicit Model Predictive Control Using Constrained Neural Networks. In *2018 Annual American Control Conference (ACC)*, 1520–1527. IEEE.

Cheon, J.H., Kim, A., Kim, M., and Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology – ASIACRYPT 2017*, 409–437.

Dwork, C. (2008). Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, 1–19. Springer.

Fiedler, F., Karg, B., Lüken, L., Brandner, D., Heinlein, M., Brabender, F., and Lucia, S. (2023). do-mpc: Towards FAIR nonlinear and robust model predictive control. *Control Engineering Practice*, 140, 105676.

Gu, B., Xu, A., Huo, Z., Deng, C., and Huang, H. (2022). Privacy-Preserving Asynchronous Vertical Federated Learning Algorithms for Multiparty Collaborative Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11), 6103–6115.

Karg, B. and Lucia, S. (2020). Efficient Representation and Approximation of Model Predictive Control Laws via Deep Learning. *IEEE Transactions on Cybernetics*, 50(9), 3866–3878.

Kingma, D.P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. URL `https://arxiv.org/abs/1412.6980`. Publisher: arXiv Version Number: 9.

Klatt, K.U. and Engell, S. (1998). Gain-scheduling trajectory control of a continuous stirred tank reactor. *Computers & Chemical Engineering*, 22(4-5), 491–502.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. (2019). On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*.

Loshchilov, I. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lucia, S., Finkler, T., and Engell, S. (2013). Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. *Journal of Process Control*, 23(9), 1306–1319.

Ma, J., Naas, S.A., Sigg, S., and Lyu, X. (2022). Privacy-preserving federated learning based on multi-key homomorphic encryption. *International Journal of Intelligent Systems*, 37(9), 5880–5901.

Marcolla, C., Sucasas, V., Manzano, M., Bassoli, R., Fitzek, F.H., and Aaraj, N. (2022). Survey on fully homomorphic encryption, theory, and applications. *Proceedings of the IEEE*, 110(10), 1572–1609.

Mayne, D.Q., Kerrigan, E.C., Van Wyk, E.J., and Falugi, P. (2011). Tube-based robust nonlinear model predictive control. *International Journal of Robust and Nonlinear Control*, 21(11), 1341–1353.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B.A.y. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. In A. Singh and J. Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, 1273–1282. PMLR.

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In J. Stern (ed.), *Advances in Cryptology — EUROCRYPT '99*, 223–238. Springer Berlin.

Rawlings, J.B., Mayne, D.Q., Diehl, M., et al. (2017). *Model predictive control: theory, computation, and design*, volume 2. Nob Hill Publishing Madison, WI.

Reddi, S., Charles, Z., Zaheer, M., Garrett, Z., Rush, K., Konečnỳ, J., Kumar, S., and McMahan, H.B. (2020). Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*.

Xu, Z. and Wu, Z. (2024). Privacy-preserving federated machine learning modeling and predictive control of heterogeneous nonlinear systems. *Computers & Chemical Engineering*, 187, 108749.

Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., and Gao, Y. (2021). A survey on federated learning. *Knowledge-Based Systems*, 216, 106775.