Journal of
Marine Science and Engineering

MDPI

Article

Federated Learning for Maritime Environments: Use Cases, Experimental Results, and Open Issues

Anastasios Giannopoulos [1,*], Panagiotis Gkonis [2], Petros Bithas [2], Nikolaos Nomikos [1], Alexandros Kalafatelis [1] and Panagiotis Trakadas [1]

1 Department of Ports Management and Shipping, National and Kapodistrian University of Athens, 34400 Euboea, Greece; nomikosn@pms.uoa.gr (N.N.); alexkalafat@pms.uoa.gr (A.K.); ptrakadas@pms.uoa.gr (P.T.)
2 Department of Digital Industry Technologies, National and Kapodistrian University of Athens, 34400 Euboea, Greece; pgkonis@dind.uoa.gr (P.G.); pbithas@dind.uoa.gr (P.B.)
* Correspondence: angianno@uoa.gr

Abstract: Maritime transportation is crucial for global trade and responsible for the majority of goods movement worldwide. The optimization of maritime operations is challenged by the complexity and heterogeneity of maritime nodes. This paper presents the emerging deployment of federated learning (FL) in maritime environments to address these challenges. FL enables decentralized machine learning model training, ensuring data privacy and security while overcoming issues associated with non-i.i.d. data. This paper explores various maritime use cases, including fuel consumption reduction, predictive maintenance, and just-in-time arrival. Experimental results using real datasets demonstrate the superiority of FL in predicting the fuel consumption of large cargo ships in terms of accuracy and spatiotemporal complexity over traditional collaborative machine learning approaches. The findings indicate that FL can significantly improve the performance of fuel consumption models in a collaborative way, while ensuring data privacy preservation and no data transmission during the learning process. Finally, this paper discusses open issues and future research directions necessary for the widespread adoption of FL in maritime transportation and settings.

Keywords: federated learning; machine learning; maritime communication networks; maritime industry; maritime transportation; security and privacy

check for updates

Citation: Giannopoulos, A.; Gkonis, P.; Bithas, P.; Nomikos, N.; Kalafatelis, A.; Trakadas, P. Federated Learning for Maritime Environments: Use Cases, Experimental Results, and Open Issues. *J. Mar. Sci. Eng.* **2024**, *12*, 1034. https://doi.org/10.3390/jmse12061034

Academic Editor: Mihalis Golias

Received: 16 May 2024
Revised: 14 June 2024
Accepted: 19 June 2024
Published: 20 June 2024

1. Introduction

Maritime transportation, an indispensable part of the world economy, relies on various associated functionalities that are related to fleet management, fuel consumption minimization, effective routing, logistics, etc. In this context, Internet of Things (IoT)-based maritime services leverage the huge amount of collected, stored, and processed data [1]. Hence, real-time monitoring and automated optimization procedures are expected to play a key role in future maritime transportation systems for ship path optimization or energy-consumption-dependent and speed optimization purposes [2–4]. To this end, traditional optimization approaches might be inefficient, as they might lead to non-convex problems, which are solved via iterative optimization approaches.

Machine learning (ML) algorithms have recently emerged as a promising solution for large-scale dataset manipulation and process optimization. In the vast majority of related applications, neural networks (NNs) are properly trained in order to have an accurate representation of the input variables to the output vectors. However, if the output variables have distinct values, a clustering strategy might be more appropriate, offering significantly reduced training time compared to NNs. In general, ML training is made feasible via supervised learning (SL), unsupervised learning (UL), and deep reinforcement learning (DRL) [5]. In SL, data associations are known a priori to the system, and the algorithm is trained with a large amount of training data. In UL, there is no pattern for the association of input variables with output metrics. As in the SL case, having a large dataset as an input prompts UL algorithms to detect any association pattern among the inputs and outputs of the system. Finally, in DRL, a mobile agent interacts with the wireless environment and tries to define the optimum set of polices via rewards or penalties associated with specific actions. After a sufficient number of training samples, it is possible to derive the optimum set of actions for all potential states.

In the maritime sector, however, proper ML training involves data collection and processing from an increased number of distributed sources that cover a large geographical area with diverse propagation conditions. Hence, centralized ML approaches might not be effective, especially for highly demanding ultra-reliable low-latency communications (URLLC) services due to the large propagation distances required for efficient data gathering. In the same context, proper data manipulation is required to mitigate the effects of diverse propagation conditions. Finally, in many cases, training data might contain sensitive information for a particular maritime component, making it preferable to be stored locally and not transmitted to centralized premises [6]. In this context, and in an effort to deal with the aforementioned issues, the concept of federated learning (FL) has been proposed as an alternate ML approach that can speed up ML execution times while keeping data localized [7,8]. In FL, maritime nodes exploit shared models trained from excessive amounts of data without the need to centrally store them.

Considering the above, this paper aims to explore the diverse applications of federated learning within maritime environments. It underscores the necessity of implementing FL in the maritime sector, highlighting its advantages in predicting critical parameters and identifying faults while ensuring security and data privacy. To demonstrate the significance of FL in maritime transportation systems, this paper includes an experimental study on fuel consumption prediction, utilizing real maritime industry datasets. The findings reveal that FL enhances prediction accuracy without increasing complexity compared to other distributed and centralized learning methods. As FL is an emerging field, this paper also addresses various open issues and research directions, aiming to foster further interest and development in the context of 6G-enabled maritime services. The main contributions of this article are as follows:

- The motivation behind using FL in maritime environments and differences with applying FL in other industrial settings is discussed.
- Maritime transportation, security and privacy, fault detection, and underwater applications are discussed together with the current state of the art in conventional ML- and FL-aided solutions;
- An illustrative use case aiming at ship fuel consumption reduction using FL with real datasets is investigated, highlighting a high accuracy and a reduced complexity over centralized ML approaches;
- Several open issues are discussed for stimulating further research in FL-assisted maritime environments.

It must be emphasized that since FL can handle non-independent and identical distributed datasets (non-i.i.ds.) [9], it can be applied in a plethora of maritime scenarios; for example, supporting fault diagnosis, predictive maintenance, just-in-time arrival, weather forecasting, and wind pattern prediction, as long as appropriate FL objective functions are defined.

The rest of this article is organized as follows. Section 2 provides an architectural description of FL and the motivation behind its use in maritime settings. In Section 3, FL-aided maritime applications are discussed while Section 4 presents our experimental results with multiple vessels participating in the FL process. Then, Section 5 includes relevant open issues, while conclusions are given in Section 6.

2. Federated Learning

2.1. Description

FL has been proposed as an efficient approach for handling distributed datasets and decoupling model training from requiring direct access to raw training data. In FL, users exploit shared models trained from excessive amounts of local data without needing centralized storage [10]. Here, devices represent clients in a federation, targeting to solve the learning task under the coordination of a central server. Each client maintains a local training dataset and only computes and transmits an update to the server's global model. FL is well suited for applications where training relies on already available data at each client, thus guaranteeing high privacy and security levels, as attacks will only affect individual devices, and not the whole federation [11].

In Figure 1, FL operation in different maritime scenarios is depicted, in which deep neural networks (DNNs) are trained at edge nodes and then the aggregated information is sent to the central processing node. Afterwards, the edge nodes are informed for the updated global model, and hence they can adapt to various operational scenarios. Also, communication links with varying levels of bandwidth, reliability, and coverage are shown, such as satellite support for long propagation distances and maritime links, including Unmanned Aerial Vehicle (UAV)-aided communications [12]. The depicted application domains include Internet of Ships (IoS), maritime IoT with buoys, and Internet of Underwater Things (IoUWT).

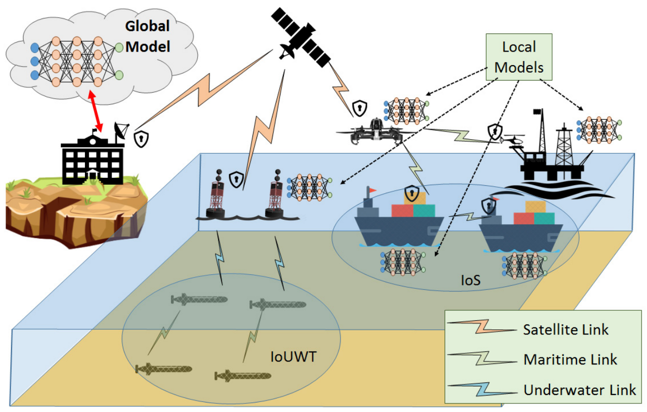

Figure 1. Federated learning in various maritime scenarios, including Internet of Ships (IoS), maritime Internet of Things (IoT) with buoys, and Internet of Underwater Things (IoUWT).

By adopting FL, the computational load is balanced among various devices, reducing latency, while local data are not sent to the central processing node, maximizing the privacy of the participating entities. As a result, local devices can also exploit the information observed and processed by other learners using the global model. Moreover, contrary to other distributed learning approaches, FL can operate with unbalanced and non-i.i.d. datasets in scenarios where the selected data differ in type and size [9]. This capability of FL increases its applicability to heterogeneous maritime scenarios and its potential in the maritime domain. In practice, FL can be applied considering various ML training modes per distant location, such as fully connected NNs, convolutional NNs, clustering approaches, support vector machines, etc. FL can provide cost- and time-effective solutions in highly demanding applications in the context of sixth-generation (6G) networks, such as autonomous driving, augmented reality, advanced e-health applications, etc. Due to the harsh propagation conditions met in long-distance communications for underwater segments, we consider that IoUWTs are passive during the FL training process, in the sense that Unmanned Surface Vehicles (USVs) are responsible for collecting their data. As a result, USVs convey the data collected by IoUWTs to train local models and participate in the FL loop.

2.2. Why FL for Maritime Environments?

Maritime environments have some unique characteristics when compared to other Industry 4.0 settings, rendering the adoption of FL-aided solutions vital [13]. More specifically, various maritime services must be provided in highly mobile and geographically dispersed topologies, comprising ships and port facilities. FL is decentralized by design, allowing models to be trained locally on each edge node without requiring constant connectivity to a central server. For example, FL can be significantly useful in UAV-aided remote [14] or offshore maritime environments with limited or intermittent wireless connectivity, significantly reducing the need for data transfer towards centralized premises and leading to energy efficiency gains for the aerial and the maritime nodes [14]. On the contrary, in more static Industry 4.0 scenarios, as is the case for factories, the advantages of FL might not be pronounced as much, since reliable power supply and wireless connectivity are usually guaranteed.

Another critical aspect is related to data privacy and regulatory compliance, considering the international nature of maritime operations, adhering to various regulations, data protection laws and standards. Here, FL can address data privacy concerns by allowing individual vessels or port facilities to maintain control over their data while contributing to global model updates [15]. In addition, the maritime industry involves different stakeholders, including vessel operators, port authorities, and regulatory bodies, and FL can foster knowledge sharing and global model improvement without neglecting data ownership and operational autonomy. Nonetheless, other industrial settings do not have to deal with possibly conflicting regulations and data locality needs, since the desired service might not impose stringent privacy constraints, only involving national stakeholders.

Finally, in intelligent maritime transportation systems, real-time decision-making must be supported for collision avoidance and route optimization purposes, especially when unmanned vessels are used. FL promotes computation task offloading by distributing the task among the edge nodes. In this way, the possible resource and energy constraints of respected models are more efficiently updated. Moreover, FL offers high scalability, as it does not require a centralized infrastructure to handle data from a diverse set of maritime nodes [16].

3. Maritime Federated Learning Use Cases

Here, FL-aided solutions for maritime transportation, security and privacy, fault detection, and underwater applications are discussed. Table 1 summarizes these solutions and highlights their main points.

Table 1. FL-aided techniques for maritime applications.

Work	Application Scenario	Method	Limitations—Open Issues
[15]	Maritime transportation	FL-aided fuel prediction and ship speed optimization	Comparisons only among FL and individual optimization
[16]	Maritime transportation	FL-aided PEP prediction	Focus only on PEP prediction accuracy
[17]	Security and privacy	Part-FL	Additional sets of data
[18]	Security and privacy	CNN, MLP federated aggregation	Realistic channel modelling
[19]	Security and privacy	FL-based gradient aggregation and credibility mechanisms	Extension to more complex orientations
[20]	Security and privacy	FL-aided anti-tampering blockchain technology	Extension to more complex orientations
[21]	Fault detection	Adaptive FL	Convergence improvement
[22]	Fault detection	Interpretable FL	Network in a multi-level federated center
[23]	Underwater applications	Federated meta-learning	Limited number of nodes

3.1. Maritime Transportation

As shipping is responsible for the vast majority of global trade, reductions in fuel consumption and CO_2 emissions are necessary to reduce costs and improve environmental sustainability. Conventional data-driven and centralized learning approaches have been proposed in various works. For example, the authors in [24] focused on energy-efficient ship management, in terms of shaft power and fuel consumption rate per unit time. Then, the authors in [25] presented a collision avoidance system relying on data analysis and risk modeling. In inland waterways, a recent study [26] developed a data-driven solution leveraging precise position, navigation, and timing data to provide bridge-collision warnings, mooring, and autonomous guidance.

However, centralized approaches have limited scalability, since vessels may be geographically dispersed and so centralized solutions might be infeasible due to high communication overheads and increased delays. In this context, a small number of works have recently presented FL-based techniques to improve the performance and sustainability of maritime transportation. The study in [15] introduced a two-phase FL-based algorithm to predict ship fuel consumption and optimize ship sailing speed. Based on the experimental results, the initial phase of the algorithm led to fuel consumption prediction with improved accuracy. In the second phase, this prediction model was leveraged to derive the optimal ship speed. The experiments revealed a reduction in ship fuel consumption between 2.5% and 7.5% when compared to models relying on individual initial data. In addition, ref. [16] developed a maritime FL framework for decentralized on-ship intelligence to predict the primary engine power (PEP) of large cargo vessels, being directly related to their fuel consumption. By using real datasets containing both weather- and ship-related data, various ML models were assessed as local learners. Among these models, artificial neural networks (ANNs) provided the best PEP prediction accuracy. Finally, comparisons were included among FL, centralized ML, and decentralized transfer learning, with FL offering the best prediction accuracy, data privacy, and communication efficiency balance.

3.2. Security and Privacy Protection

Security and privacy protection requirements in maritime applications restrict the transmission of sensitive data to centralized nodes for model training purposes. Therefore, FL has received significant attention in the literature. The authors of [17] proposed a part-FL (PFL) approach where the collected data might not be identically distributed, and the algorithm only used a portion of the data that were uploaded to the general global model. Hence, the privacy of the algorithm was improved, while signaling overheads were reduced. PFL divided the original model into a shared model part and a non-shared model part. Optimization included the optimal allocation of the segmentation points to reduce the total communication and computation delay between the client and the server when the channel information is known. Finally, the advantages of PFL in processing non-i.i.d. data were shown through the experimental results, while its feasibility was verified by simulation analysis of the parameter optimization process.

In [18], the authors deal with intrusion detection in maritime transportation systems. Compared with centralized servers, IoT devices are more vulnerable to cyber-attacks due to power, memory, and processing limitations. Here, a convolutional neural network (CNN) model based on a multi-layer perceptron (MLP) was used, where the CNN provided data feature extraction and the MLP performed attack classification. Moreover, the straggler problem of FL was studied, referring to the inability of the involved devices to upload their data in a timely manner. As a mitigation measure, batch federated aggregation was proposed and evaluated in both i.i.d. and non-i.i.d. scenarios, showing that batch federated aggregation suppressed model parameter oscillations during FL.

Next, ref. [19] used FL to protect a maritime transportation system against Byzantine attackers. A privacy-preserving gradient aggregation algorithm was developed using secure two-party calculation. A key novelty of the proposed method was the introduction of credibility to measure the trustworthiness of a ship's sharing knowledge, updating its credibility based on the sharing information in each epoch. Then, a guiding gradient vector relying on the credibility was designed, comparing the similarity between a ship's gradients and the guiding gradients to efficiently determine whether the ship was a Byzantine node or not.

In a maritime IoT setting, ref. [20] combined FL with blockchains to ensure privacy protection and proper selection among the edge nodes. The blockchain served as a decentralized method, which stored FL workers to guard against tampering and increase security. FL processed the original data and only stored the calculation results, thus reducing the cost of storage resources and addressing the issue of the limited energy of edge devices. Also, a proof-of-parameter quality consensus mechanism for the opinion blockchain was designed, aiming to add higher-quality workers to the blockchain.

3.3. Fault Detection

Fault detection and predictive maintenance are crucial in the maritime industry, since the lifetime of vessels and port infrastructure and the security of assets and personnel are maximized. In this domain, FL has many benefits, as multiple vessels and assets participate in the training process and reduce the total execution time. Ref. [21] considered the problem of fault detection in the IoS, adopting FL to increase privacy, where the central server adjusted the aggregation interval based on the information from the shipping agents. Furthermore, to deal with the harsh maritime environment, a control algorithm was proposed to adapt the model aggregation interval during training, thus reducing cryptography, computation, and communication costs.

In a similar setting, ref. [22] presented an interpretable multi-level FL network for ship fault diagnosis, dealing with non-i.i.d. data threatening FL performance. Specifically, this study focuses on the detection of rolling bearing faults in ships using an interpretable multi-level FL network. Rolling bearings are critical components in maritime machinery, and their failure can lead to significant operational issues. The types of faults detected in this context include (i) Outer Race Faults (i.e., damage to the outer raceway of the bearing), (ii) Inner Race Faults (i.e., damage to the inner raceway), (iii) Rolling Element Faults (i.e., defects in the rolling elements, balls or rollers), and (iv) Cage Faults (i.e., issues with the bearing cage that holds the rolling elements in place). These faults can result in vibrations, noise, and operational inefficiencies. The FL approach improves fault detection accuracy by addressing the challenges posed by non-i.i.d. (non-independent and identically distributed) data, ensuring robust and reliable diagnosis across different ships and operational conditions. To this end, the concept of multi-level sharing was introduced where in the first step, each client performed local training and in the second step, different parameters were sent to different layers according to calculated network values. Experimental evaluation revealed that the interpretable FL solution provided improved fault detection accuracy over other approaches, overcoming the issue of non-i.i.d. data.

3.4. Underwater Applications

Another important maritime area comprises IoUT deployments and underwater vehicles. Currently, most studies are based on centralized learning paradigms that might pose difficulties for resource-constrained edge nodes to participate in model training processes; for example, when autonomous underwater vehicle (AUV) swarms are employed in search-and-rescue or pollution-monitoring applications [27–29].

Regarding FL-aided techniques, in an ocean-of-things setting, relying on an acoustic network to acquire underwater information, ref. [23] employed federated meta-learning (FML) to tackle the lack of insufficient training data at a single buoy. FML improved the performance of DNN-based acoustic receivers by leveraging the model parameters from multiple buoys. The convergence performance was analyzed and a closed-form expression for the convergence rate was derived, considering the impact of scheduling ratios, local epochs, and data volumes on a single node. The experiments showed that FML offered better bit error rate performance and reduced complexity compared to conventional matched filter-based detectors. Also, FML achieved improved generalization performance over conventional FL.

4. FL-Aided Fuel Consumption Prediction

This section presents a practical implementation of FL, exploiting a dataset provided by a commercial maritime enterprise. The goal of the presented experimental results is to quantify the degree to which an FL-based decentralized scheme is superior against benchmark approaches, namely against a centralized, a transfer learning, and an ensemble learning scheme. Specifically, a multi-variate regression scheme is considered to model and predict the on-road fuel consumption of large cargo ships. Given that fuel consumption is proportional to the ship's consumed Main Engine Power (MEP), the latter was used as the target variable. Since an accurate fuel consumption prediction can allow the adoption of a proactive policy and timely corrective actions towards reducing both the ship's operational costs and environmental footprint, MEP prediction is considered a crucial optimization task in maritime environments. Coupled with the previous point, proactive adjustment of fuel management acts beneficially for both the ship operator (since it saves fuel) and the environmental footprint (reduces pollution), perfectly fitting with the timely concerns faced by the whole maritime era, namely the "Fit-for-55" concept.

A deep learning solution using fully connected Artificial Neural Networks (ANNs) is adopted to estimate the MEP based on multiple ship- and weather-related features as model predictors. Given that the relationship between predictors and the target variable is not known a priori, ANNs were selected as the regression models because of their ability to accurately estimate both linear and non-linear complex functions.

Noteworthily, the fuel consumption prediction paradigm comprises an indicative practical use case for comparative purposes. Without a loss of generality, other objective-specific use cases could be also considered (e.g., weather forecasting, wind pattern predictions, prediction of the vessel propulsion engine condition, just-in-time arrival modeling, etc.) for deploying the FL model and comparing it with other baselines. The following simulation setups were conducted in Python 3.8 using the Sci-kit learn and Tensorflow (version 2.4) libraries. The training phase of all algorithms ran on a PC (CPU i7-8700; 3.2 GHz; RAM 8 GB; no GPU usage).

4.1. Dataset Description

The raw form of the dataset contained information about six twin cargo ships provided by a maritime enterprise. Ten time-series per ship were recorded with a sampling period of 1 min, including multiple voyages of the same ships, as well as ship- and weather-related data. The recordings had a duration of 1 year (from 1 July 2021 to 30 June 2022), resulting in 514,525 and 525,005 total samples per time-series for ships 1 and 2, respectively.

In specific, the following variables are available for each cargo ship:

1. **Speed over ground (SOG)**: It is the speed of the vessel relative to the surface of the Earth, measured in knots.
2. **Speed through water (STW)**: It is the speed of the vessel relative to the water currents, measured in knots.
3. **Heading**: It is the direction in which the ship is pointing and it is expressed as the angular distance relative to north (0°), clockwise through 359°.
4. **Continuous wind speed (CWS)**: It is the wind velocity amplitude expressed in units of m/s.
5. **Discretized wind speed (DWS)**: It is the quantized version of CWS, expressed in Beaufort scale (bft), and takes the categorical values of 0–12, mapping with the 13 wind types (from calm to hurricane-force wind categories).
6. **Wind direction (WD)**: It is the direction of the true wind relative to the ship's on-board heading. It varies from 0° to 360° (wind on the bow at 0°, wind on the beam at 90°, wind on the stern at 180°).

7. **Draft forward (DF)**: The draft forward (bow) is measured (depth in meters) at the perpendicular of the bow according to predefined depth levels.
8. **Draft aft (DA)**: The draft aft (stern) is measured (depth in meters) at the perpendicular of the stern according to predefined depth levels.
9. **Trim**: The trim of a ship is the difference between the DF and DA (depth in meters) relative to the designed waterline located at the middle of the ship. It determines the minimum depth of water a ship can safely navigate.
10. **Main Engine Power (MEP)**: It is the total power supplied by the prime mover(s) installed on a ship to provide propulsion and is expressed in kW. Obviously, it is positively correlated with the electricity requirements of the ship's diesel engine, thus being directly proportional to the fuel oil consumption.

4.2. Preprocessing and Dimensionality Reduction

Here, we note the preprocessing steps carried out to increase the signal-to-noise ratio of the raw data. Given the low sampling period (1 min), multiple successive samples of the time-series show zero or slow variations. This allowed us to simply replace the missing values by interpolation. Independently for each ship and variable, the missing values were identified and replaced by the average of the previous and next available samples. The missing values did not exceed the 8% of the total sample size for all variables. To maintain only on-board samples, the rows corresponding to stationary ship instances (SOG < 0.2 knots) were totally rejected. Subsequently, downsampling by a factor of 15 (1 sample per 15 min) was applied to the whole dataset for the purposes of reducing the data redundancy and facilitating the model execution time. Before downsampling, a moving-average filter was applied to smooth the curves and mitigate the distortion caused by aliasing. Then, a visual inspection of the parameter distributions was performed to identify additional outliers (samples exceeding the mean by three standard deviations). Finally, all features and target variables were normalized to take values in the range [0, 1], according to the following formula [30]:

$$x'_{i,j} = \frac{x_{i,j} - \min\{X_j\}}{\max\{X_j\} - \min\{X_j\}}, \qquad (1)$$
$$\forall i \in [1, 2, \ldots, |X_j|], j \in [1, 2, \ldots, 10],$$

where $x'_{i,j}$ and $x_{i,j}$ denote the scaled and non-scaled, respectively, sample i of variable j, and X_j is the column vector of variable j. The operator $|\cdot|$ performs the vector element count.

As an initial attempt to relax the regression model complexity and avoid feature redundancy, specific objective and multi-collinearity criteria were followed to retain or reduce the model dimensions (number of input neurons). First, the heading parameter was initially excluded as completely unrelated factor to the ship's MEP. DF and DA features were additionally avoided, as they are directly reflected in Trim. Finally, only the DWS was selected to represent the wind velocity amplitude. Note that, both SOG and STW were retained, since STW indirectly contains information about sea currents and flows and it is possible to be mismatched with the SOG (e.g., when the boat is drifting in a 5-knot current, then SOG = 5 kn and STW = 0). This was also confirmed by computing the correlation coefficient between SOG and STW (Pearson's $R_{coef} = 0.23$). Further elimination of the number of features to be included in the MEP models is presented in Section V.D.

4.3. Tuning of Learning Hyper-Parameters

When training a multi-layer perceptron-based ANN model, the selection of the hyper-parameters has to be cautiously considered, as their values can significantly impact the prediction error convergence. Among several learning hyper-parameters, different values of the learning rate (a, it influences the back-propagation and ANN weight adjustment) and ANN deepness (number of hidden layers, it impacts the model complexity) were found to cause serious variations in the model performance. To that end, extensive experimentation on those parameters was performed to achieve their best value tuning. As the activation function of the ANN units, the ReLu function was adopted, whereas an Adam optimizer was applied for implementing the Stochastic Gradient Descent of the back-propagation iterations. Initially, we separated the dataset into training/validation data using a 90/10 split. For each model training setup, the Mean Absolute Error (MAE) was extracted over the validation samples. To quantify the best hyper-parameter configuration, the Normalized Accuracy (A_{norm}) of each derived model was inversely proportional to MAE and referenced to the minimum MAE model ($A_{norm} = 1$ for the most well-tuned model), according to the formula:

$$A^{i}_{norm} = \frac{\min_{j \in M} \{\sum_{n=1}^{N_{val}} |y^{n,j}_{pred} - y^{n,j}_{real}|\}}{\sum_{n=1}^{N_{val}} |y^{n,i}_{pred} - y^{n,i}_{real}|}, \qquad (2)$$

where A^{i}_{norm} is the Normalized Accuracy of model i, $y^{n,i}_{pred}$ and $y^{n,i}_{real}$ are the predicted and actual values of validation sample n and N_{val} is the count of validation data. Indices i and j run over the set of models M derived by different hyperparameter configuration (i.e., count of models is a levels multiplied by deepness levels or 5·5 = 25 models). Thus, the accuracy of all models is referenced to the best model performance, since the denominator reflects the MAE resulted from model i, whereas the nominator calculates the minimum MAE across all models.

Figure 2 shows the Normalized Accuracy for 30 pretrained models considering (five) varying values a for each of the six ships, considering fixed ANN deepness for all ships of four hidden layers. Evidently, local ANN models for all ships are optimally configured for $a = 0.001$ and four hidden layers. The neurons contained in each hidden layer were, respectively, [1000, 800, 600, 400].

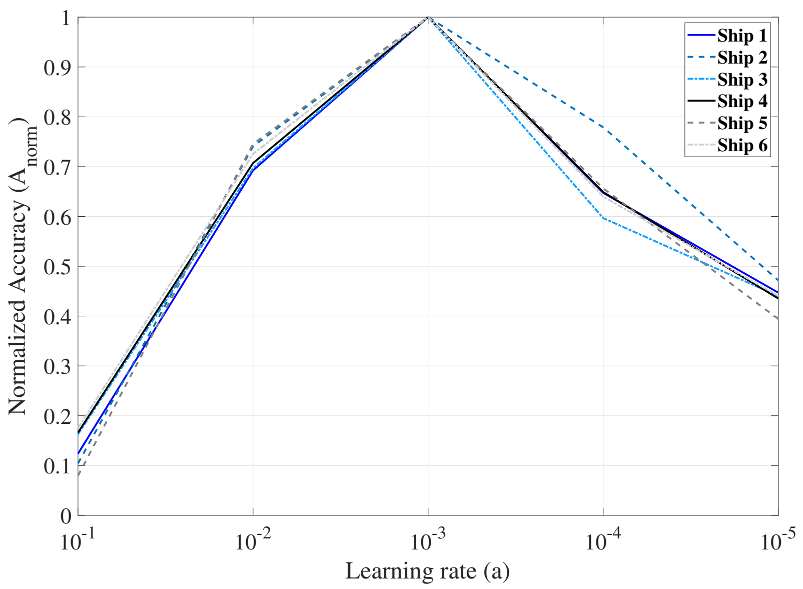

Figure 2. Model performance as a function of the learning rate a for each local ship model. All models have their optimal ANN deepness of 4 hidden layers.

4.4. Recursive Feature Elimination

For testing any further model complexity relaxation, a Recursive Feature Elimination (RFE) method was adopted. To sort the five features (SOG, SWT, DWS, WD, and Trim) by their importance in predicting the MEP, the RFE involved an XGBoost-based feature importance test. Specifically, independently for each ship model, an XGBoost regressor was fitted to the training data and the F-values of the correlation statistical tests (feature vs. target variable) were extracted to indicate the feature importance. XGBoost was preferred against other feature importance methods, given its fast execution time and superiority against linear models. All ship models showed the same sorted-by-importance version of features as follows: [Trim, SOG, WD, STW, DWS] (from highest- to lowest-importance features). As expected, Trim (depends on the cargo load) and SOG (depends on fuel consumption) have a high degree of association with MEP, with the WD being the next most important feature (see Figure 3). Having the sorted feature list, we recursively applied the steps included in Algorithm 1 below.

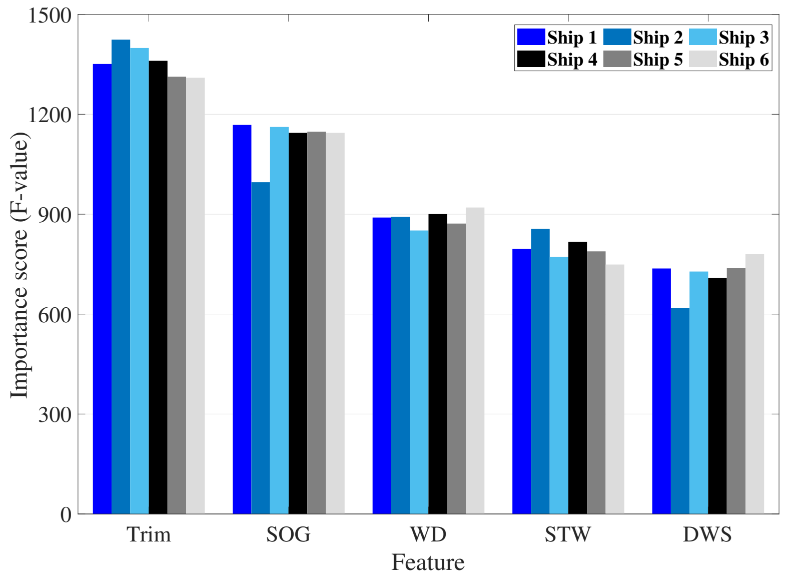

Figure 3. Sorted importance scores of features derived by XGBoost model for all ship models.

Algorithm 1 Feature elimination algorithm

```
1:  input Number of ship models $N$
2:  for each ship $i = 1$ to $N$ do
3:      input Number of features $K = 5$
4:      Set the ship model to the optimal hyper-parameters.
5:      while $K >= 1$ do
6:          Select the $K$ most important features as model inputs.
7:          Train the model with $K$-feature input.
8:          Extract the validation MAE.
9:          Reduce the number of features $K \leftarrow K - 1$.
10:     end while
11:     output Reference the accuracy of all ship-specific models to the best one.
12: end for
13: output Plot the referenced accuracy scores of all ship as bar plots in Figure 4.
```

Figure 4 depicts the Normalized Percentage Accuracy bars for all feature-specific models of each ship. This metric was calculated as A_{norm} (%) to concretely demonstrate the percentage accuracy decrease as features were eliminated. As shown in Figure 4, there is a gradual performance decrease with the number of features included in the model. This is attributed to the fact that each feature related (highly or slightly) to the MEP can improve the model performance at the expense of increasing the model complexity (performance–complexity trade-off). Given that four features do not significantly reduce the model accuracy (lower than 10% decrement than accuracy with all features), we finally eliminated the number of features used for input to four (Trim, SOG, WD, STW), so as to mitigate the model complexity.

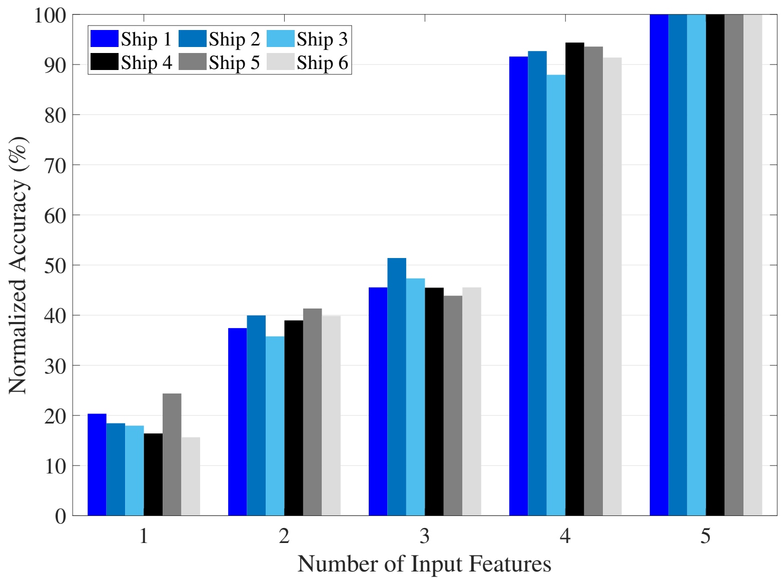

Figure 4. Model performance of each ship model for different number of input features.

4.5. Performance vs. Spatial Complexity Comparison

Upon completion of hyper-parameter value stabilization and feature elimination to aim towards a reasonable performance–complexity equilibrium, this section presents comparative results amongst several MEP prediction schemes. To quantitatively investigate the prons and cons of different centralized and decentralized approaches, four prediction schemes are considered and contrasted in terms of their prediction accuracy and complexity. The analytical high-level architecture of the considered schemes is shown in Figure 5, where the learning operation performed at the port site in FL scheme (Figure 5D) represents the aggregation step across the local models. Specifically, the following schemes were trained and tested:

1. *Centralized Learning and Inference (CLI)*: This is the conventional computationally expensive scheme, in which all data samples (from all ships) are collected in the central server to build a large and powerful model. Inference takes place locally upon vertical communication between local ship clients and the central server. The ship-independent central model constantly requires data exchange with the local clients. The estimation model of this scheme was derived by training an ANN using the data from all ships as a whole.

2. *Local Learning and Model Sharing (LLMS)*: Based on the principles of transfer learning, one ship-specific model is trained locally and then the rest of the ships inherit the pretrained model for future inference purposes. In this scheme, we tested all ships inheriting from all ships (i.e., ship 1 transfers to the rest, ship 2 transfers to the rest, and so on). The central server is only used for model sharing operations across the local ship clients. The estimation model of this scheme was derived by training only one local model and then copying it to the rest of the agents.

3. *Local Learning and Collaborative Inference (LLCI)*: Similar to the ensemble learning technique, this scheme trains a local ship-specific model for each ship and then, during the inference phase, each ship can take the average prediction calculated over all local model predictions, given a particular testing sample. To make this scheme work, each local client has to store all the six models locally for the purposes of mean prediction extraction without communication overhead.

4. *Federated Learning (FL)*: This scheme follows the typical federation process proposed in the FedAvg algorithm [10]. In brief, each client trains the local model decentrally and only the learned model parameters are sent to a trusted center. The latter combines the

parameters to aggregate the global model before circulating it back to all local clients. The formula used for the aggregation operation is the following [10]:

$$W_{Global}^{t+1} = \sum_{m=1}^{N_m} \frac{n_m}{n_{total}} \cdot W_{Local,m}^{t}, \tag{3}$$

where W_{Global}^{t+1} are the global model parameters at aggregation round $t+1$ and $W_{Local,m}^{t}$ are the parameters of the local model $m \in [1, 2, ..., N_m]$ at the aggregation round t. Further, n_m is the sample count of the local model m and n_{total} is the total sample count of all FL local models.

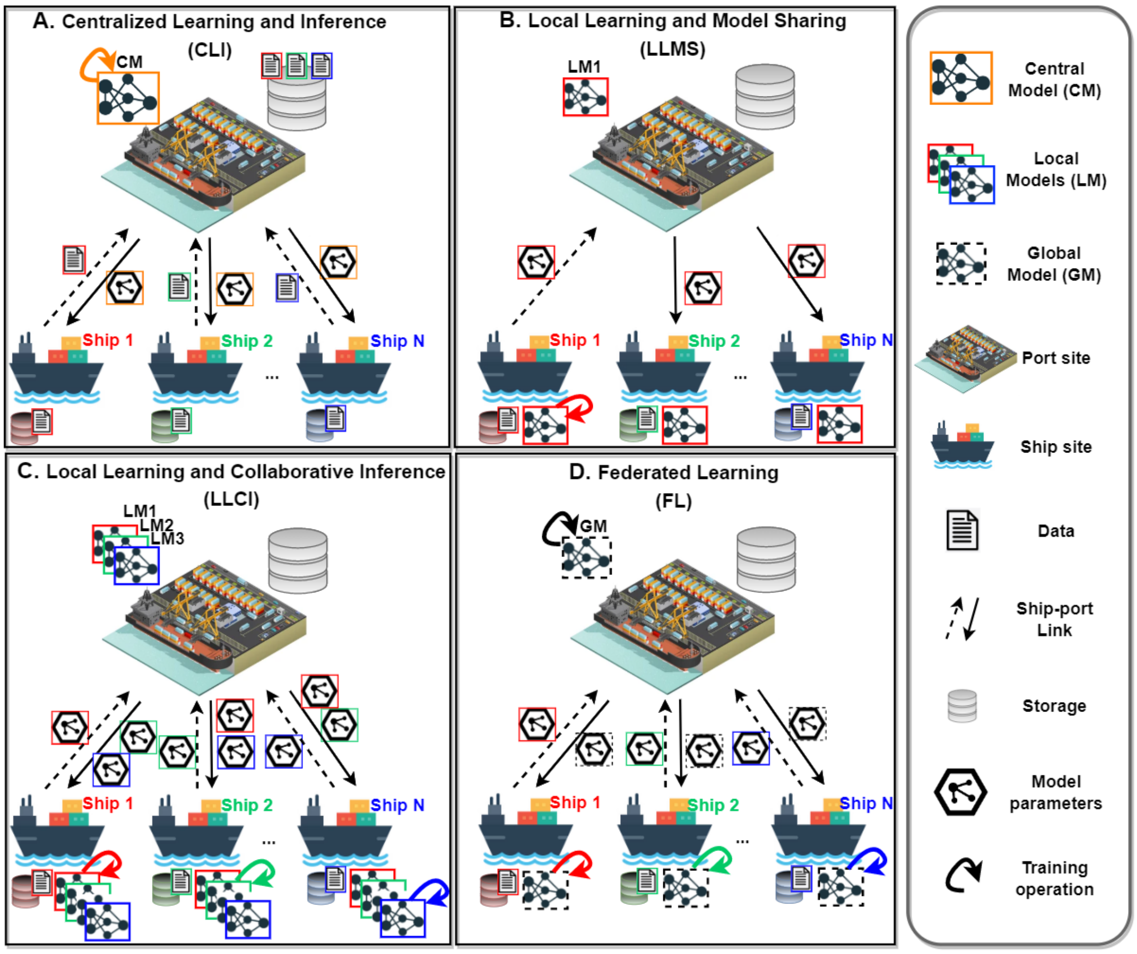

Figure 5. Architecture of the four (**A**–**D**) centralized and decentralized schemes considered for distributed ships' MEP prediction.

The accuracy A_i of each deployed scheme i is measured as the MAE calculated over the validation data (unseen during training), whereas the corresponding complexity C_i is quantified via the following formula [31]:

$$C_i = \sum_{m=1}^{N_m} (E_m \cdot U_m + S_m), \tag{4}$$

where the total complexity C_i (in a.u.) of scheme i (that includes N_m ANN models) is equal to the aggregated complexity of all individual models included in the scheme. Each individual model m $(m = 1, \ldots, N_m)$ exhibits a complexity that is equal to the sum of ANN density (i.e., the product between the hidden edges E_m and the hidden units U_m) and ANN storage requirements S_m (in KB).

For testing the LLMS, LLCI, and FL schemes, the optimal configuration presented in Sections V.C-D was adopted for the local ship models, whereas the CLI scheme showed a minimization of the MEP mean prediction error for $a = 0.001$ and five hidden layers upon extensive training simulations.

Figure 6 demonstrates the performance (in terms of accuracy) and the complexity of the deployed schemes. The validation samples included in the accuracy/complexity assessment of all schemes concerned data samples from all ships, all drawn from a dataset that was not encountered during training. To concretely show the relative performance/complexity decrement between schemes, we used normalized measures: all schemes' accuracy ($1/MAE_i$) was divided by the best-accuracy ($1/MAE_{min}$) scheme (i.e., CLI), whereas all schemes' complexity was divided by the lowest-complexity scheme (i.e., LLMS or FL). As clearly evidenced by Figure 6, CLI presents the highest prediction accuracy, with FL exhibiting 11% accuracy reduction relative to CLI.

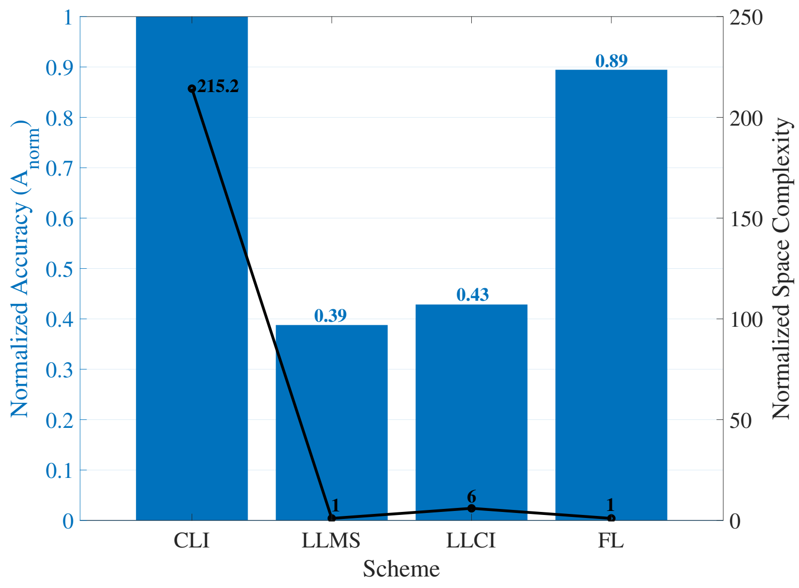

Figure 6. Normalized . accuracy (left y-axis, blue) and complexity (right y-axis, black) for the 4 MEP prediction schemes CLI, $LLMS$, $LLCI$, and FL.

Notably, the LLMS presented the lowest accuracy, attributed to the selfishness of the source model (the model that is transferred to the rest of the models) on its own data. The explanation for the non-effectiveness of this scheme is that it does not have global information about all ships, as opposed to CLI which was trained on the six-ship dataset. It is also observed that the $LLCI$ shows increased accuracy compared with the $LLMS$, given that averaging the predictions reduces the error introduced by inferring only one selfish model. Note that, in the LLMS scheme, models are simply transferred and inferred between ships, without any (soft) retraining in the model receiver site. In that case, the LLMS and LLCI schemes' accuracy would be improved at the expense of longer training and time-consuming processes than for FL.

For completeness, Table 2 contains quantitative information for comparing CLI and FL in terms of seven evaluation metrics: Normalised Accuracy (A_{norm}), Goodness of Fit (GoF), Mean Absolute Error (MAE), Mean Squared Error (MSE), and Mean Absolute Percentage Error (MAPE). To have a clear view of the non-normalized MEP accuracy, Figure 7 depicts the actual and predicted MEP curves for the validation samples, including the two best prediction schemes (CLI and FL).

Table 2. Comparison between FL and CLI performance.

Metric	**Federated Learning (FL)**	**Centralized Learning and Inference (CLI)**
A_{norm} ($a.u.$, 0–1)	0.89	1
GoF ($a.u.$, 0–1)	0.85	0.94
MAE (kW)	13.3	11.9
MSE (kW^2)	171.6	132.2
MAPE (%)	12.7	8.8

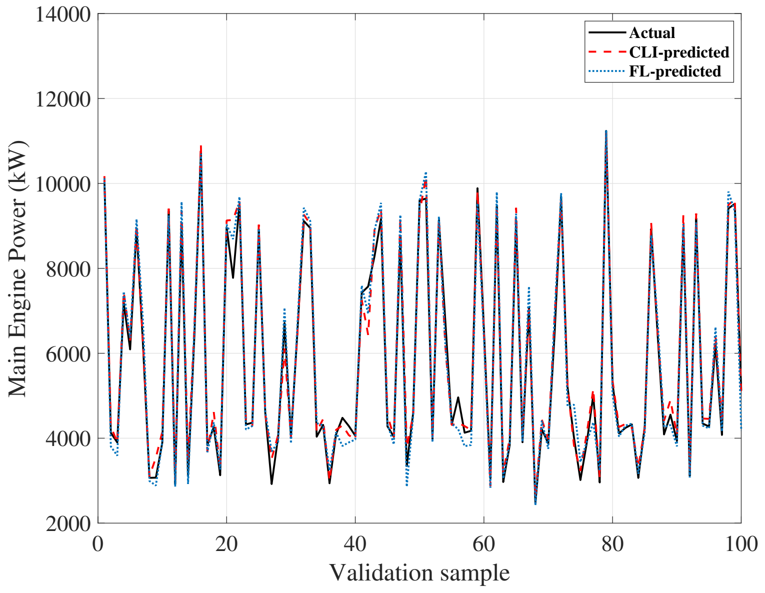

Figure 7. MEP validation curves for the actual, CLI-predicted, and FL-predicted values.

Regarding the complexity, we observed that CLI is 215.2 times more complex than the LLMS and FL schemes, given its high ANN density and size (153.6 $\times$ 10^{12} edges, 3005 units, and 36.9 MB) to achieve the best accuracy. On the contrary, the LLMS and FL schemes show the lowest complexity because they require low-dimensionality ANNs (768 $\times$ 10^9 edges, 2805 units, and 35.4 MB). Note that the LLCI scheme is six times more complex than the LLMS/FL scheme, since six ANNs are deployed at each ship site (to extract the average prediction), where one ANN per site is required for LLMS and FL.

4.6. Runtime and Time Complexity Considerations

The computation complexity of an algorithm can be considered as the amount of the resources needed to accurately execute it from both computation time (*time complexity*) and memory storage requirements (*space complexity*) perspectives. Since space complexity was analyzed in the previous section, this subsection gives an comparative overview of the time complexity characterizing each of the schemes used in this work. Table 3 reports the practical training and inference runtime required for each scheme to run. Notably, the runtime required for training purposes is required only one time (and usually before the real operation of the model), and, once the model is trained, it can—almost effortlessly—be inferred multiple times (inference requires only some arithmetic operations).

Table 3. Runtime and time complexity per scheme.

Scheme	Runtime for Training (h:min:s)	Runtime for Inference (s)
CLI	00:38:43	3.58
LLMS	00:05:19	0.61
LLCI	00:05:21	3.34
FL	00:43:44	0.62
Scheme	**Time Complexity for Training**	**Time Complexity for Inference**
CLI	$\mathcal{O}(E' \cdot f_{GD}(N \cdot n, F, N'_H))$	$\mathcal{O}(N \cdot n' \cdot f_{FF}(F, N'_H))$
LLMS	$\mathcal{O}(E \cdot f_{GD}(n, F, N_H))$	$\mathcal{O}(n' \cdot f_{FF}(F, N_H))$
LLCI	$\mathcal{O}(E \cdot f_{GD}(n, F, N_H))$	$\mathcal{O}(N \cdot n' \cdot f_{FF}(F, N_H) + N \cdot N_O)$
FL	$\mathcal{O}(R \cdot E \cdot f_{GD}(n, F, N_H) + R \cdot \psi)$	$\mathcal{O}(n' \cdot f_{FF}(F, N_H))$

Apart from this, Table 3 tabulates the theoretical time complexity of the compared methods considering worst-case scenarios. To this end, time complexity was quantified as the overall and worst-case numbers of iterations performed by each algorithm, as a function of the input size n and other learning hyper-parameters using the *big O notation* as $\mathcal{O}(f(n))$. We used the following notations for defining the time complexity of both the training and inference phases: n is the number of the input data rows available for each local model for training, n' is the number of the input data rows per local model for inference, F is the number of input features, N_H is the number of hidden neural units of single-agent models, N'_H is the number of hidden neural units of the centralized model, N_O is the number of the output neural units, E is the number of local training epochs of the single-agent model, E' is the number of local training epochs of the centralized model, N is the number of the local ship agents, R is the number of the aggregation rounds performed at the central location (e.g., cloud), ψ is the total number of weights averaged at each aggregation round, $f_{GD}(\cdot)$ is the cost for performing a single GD step (including feedforward and back-propagation steps), and $f_{FF}(\cdot)$ is the cost for performing a single feedforward step.

The CLI training uses all the training samples from all ships, and thus it has $N \cdot n$ training rows in total. Considering E' number of epochs for training the CLI model, the time complexity is $\mathcal{O}(E' \cdot f_{GD}(N \cdot n, F, N'_H))$. The FL algorithm operates according to the following sequence: At each global round, data held on each local agent are either passed row-by-row (Gradient Descent—GD) or split into several batches (Stochastic Gradient Descent—SGD) according to a predefined batch size; each row/batch is passed as a whole to train the local model and an epoch is completed once every row/batch is used for learning. Finally, after multiple epochs, each agent sends the adjusted weights to the server, which then computes the average weights across all agents and distributes it back to them. Thus, assuming the worst case of row-by-row gradient calculation (i.e., the GD case), the data rows are accessed n times per epoch, whereas E epochs are performed for each of the R global aggregation rounds. The cost function $f_{GD}(n, F, N_H)$ is encountered at every epoch, and it includes the cost for the feedforward passing and back-propagation passing of all n samples. Assuming sequential local training rounds for the N ship models (i.e., each local model is trained one after the other), this results in a time complexity of $\mathcal{O}(R \cdot N \cdot E \cdot f_{GD}(n, F, N_H))$, whereas the average extraction across the local models has a complexity of $\mathcal{O}(\psi)$ and is performed R times. Thus, the final complexity of the FL scheme is $\mathcal{O}(R \cdot N \cdot E \cdot f_{GD}(n, F, N_H) + R \cdot \psi)$ (assuming sequential training across ships) or $\mathcal{O}(R \cdot E \cdot f_{GD}(n, F, N_H) + R \cdot \psi)$ (assuming parallel training across ships). In a similar concept, to train a single-ship model, as only required by the LLMS scheme, the complexity is $\mathcal{O}(E \cdot f_{GD}(n, F, N_H))$. This is basically the time complexity met in the FL scheme as a sub-term to train each agent in each global round, thus making the LLMS training the best scheme in terms of time complexity. Finally, the time complexity for training the LLCI scheme is N times the complexity of LLMS training, i.e., $\mathcal{O}(N \cdot E \cdot f_{GD}(n, F, N_H))$, given that N single-ship models are required to be trained. If the ships are trained in parallel, the time complexity is $\mathcal{O}(E \cdot f_{GD}(n, F, N_H))$. Regarding the time complexity for inference, given n' input instances, the complexity is derived by computing the product between the input size and the cost for the feed-forward step (plus the cost for averaging the output predictions only in LLCI).

Based on the previous analysis, some possible ways to reduce the time complexity can be identified: (i) minimization of the number of communications as the gold standard for reducing the complexity in FL; (ii) instead of standard gradient descent, the use of Stochastic Gradient Descent can reduce the time complexity, as the data rows are accessed in batch groups (and not row-by-row), while ensuring performance convergence; (iii) gradient computation can be parallelizable within each ship agent. Note also that it would be beneficial for each untrained model to start training using the pretrained weights of another model, rather than randomly initializing its weights. The latter solution basically corresponds to transfer learning and can significantly reduce the time complexity in multi-agent and multi-round FL.

4.7. Training and Deployment Considerations

Evidently, previous analyses highlighted the performance–complexity trade-off, with the CLI presenting the highest accuracy and the highest complexity, whereas FL showed slightly lower accuracy with significantly lower complexity. One may say that for a completely accuracy-centric scheme, the CLI should be selected, whereas for a complexity-centric scheme, FL would be the best choice. In this selection equation, the training and deployment aspects of the schemes should be also taken into account. First, to make the CLI training feasible, the central server directly receives the data from all ships via an uplink communication protocol. Upon training finalization, the central model is transferred to all the other ships for future inference. As for FL training, ship-specific model updates are performed locally and only the model parameters are sent to the server for FedAvg, preserving the data privacy. All ships have finally the globally averaged model for potential inference purposes. Further, local model training is also the case for the LLMS and LLCI schemes, with the central server being used only for model, and not data, forwarding. Consequently, FL superiority is more pronounced when jointly considering its accuracy, complexity, feasibility, and privacy-preservation attributes.

5. Open Issues

As the integration of FL in maritime settings is still in its early stages, various open issues remain, as discussed below.

5.1. Large-Scale Integration with 6G Networks

Since the discussion of decentralized architectures has started taking place, it is expected that maritime networks will evolve in the 6G era, incorporating key enabling technologies, such as network function virtualization and software-defined networks (SDNs) [32]. Recently, FL-based solutions have emerged to optimize data routing and traffic control in challenged SDNs, being characterized by varying network conditions and the high data volumes of edge devices [33].

5.2. Long Propagation Distances and Diverse Channel Characteristics

Maritime environments involve large propagation distances, multi-modal communication (e.g., radio- or acoustic-based), and the impact of weather conditions and the relative motion of the sea surface. In order to overcome these issues, FL-aided solutions are needed to minimize the amount of data transmitted to centralized premises while keeping high local learning performance. For example, in URLLC collision avoidance scenarios with autonomous surface vehicles, decentralized FL has been adopted to mitigate the impact of environmental parameters on the delay performance [34].

5.3. Non-Identical Training Data from Diverse Sources

Non-identical data manipulation is a crucial step towards data integration from diverse sources. In this case, there might be incompatibilities in the received data, especially in highly correlated environments. Recent solutions have proposed batch federated aggregation, succeeding in suppressing oscillations due to non-i.i.d. data during FL training [18].

5.4. Full-Scale Performance Evaluation

Until now, research on FL integration in maritime environments has mainly considered limited network topologies, statistical propagation environments, and a small number of participating devices in the FL process. This work aimed to present experimental results with real datasets from the maritime industry and involved a varying number of participating vessels. Moreover, it was shown that the application of the proposed FL method can be generalized to other maritime applications by appropriately defining the objective function, e.g., to conduct accurate fault detection. Still, experimental results with a massive number of edge devices are needed to reveal the impact of high-dimensional and high-volume data, heterogeneous networks, and vulnerabilities on FL performance.

5.5. Interpretable ML

The federated network obtained by means of learning is a black box, and it is difficult to understand the internal mechanisms of the network. In this case, an interpretable approach is necessary to prevent malicious attacks on the central server if a hostile client knows the gradient parameters of the other clients. In the same context, as the authors correctly point out in [22], how to build a federal network structure in a multi-level federated center still needs to be further explored in future research.

6. Conclusions

In this paper, various applications of federated learning in maritime environments were presented. First, the need for maritime federated learning was emphasized together with its benefits in predicting key parameters and detecting faults while safeguarding security and data privacy. In order to highlight the importance of federated learning for maritime transportation systems, an experimental investigation related to fuel consumption prediction was performed, using real datasets from the maritime industry. The results showed that federated learning improved prediction accuracy without incurring a higher complexity than other distributed and centralized learning algorithms. Finally, as a highly promising field in its early stages, various open issues and research directions of maritime federated learning were discussed, aiming to stimulate further interest in the era of 6G-enabled maritime services.

Author Contributions: Conceptualization, A.G. and N.N.; methodology, A.G., P.G. and P.B.; software, A.G.; validation, N.N., P.B. and P.T.; formal analysis, A.G. and P.G.; investigation, P.B.; resources, P.T.; data curation, P.T.; writing—original draft preparation, A.G., P.G., P.B. and N.N.; writing—review and editing, A.K. and P.T.; visualization, A.G. and P.B.; supervision, P.T.; project administration, P.T.; funding acquisition, P.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the ICOS (Towards a functional continuum operating system) Project, which is funded by the European Union's HORIZON research and innovation programme under grant agreement No. 101070177 (https://www.icos-project.eu/, accessed on 20 June 2024).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets presented in this article are not freely available due to commercial constraints. However, access to the datasets can be provided upon request to the corresponding author.

Acknowledgments: The authors would like to thank George Ntroulias for his help and Hydrus Engineering S.A. for the data provisioning.

Conflicts of Interest: The authors declare no conflicts of interest. Also, the funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

Acronym	**Meaning**
ANN	Artificial Neural Network
CLI	Centralized Learning and Inference
CNN	Convolutional Neural Network
CWS	Continuous Wind Speed
DA	Draft Aft
DF	Draft Forward
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DWS	Discretized Wind Speed
FL	Federated Learning
GD	Gradient Descent
LLCI	Local Learning and Collaborative Inference
LLMS	Local Learning and Model Sharing
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MEP	Main Engine Power
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
PEP	Primary Engine Power
RFE	Recursive Feature Elimination
SGD	Stochastic Gradient Descent
SL	Supervised Learning
SOG	Speed Over Ground
STW	Speed Through Water
SWT	Ship Wind Turbine
UAV	Unmanned Aerial Vehicle
UL	Unsupervised Learning
URLLC	Ultra-Reliable Low Latency Communications
WD	Wind Direction

References

1. Xia, T.; Wang, M.M.; Zhang, J.; Wang, L. Maritime Internet of Things: Challenges and Solutions. *IEEE Wirel. Commun.* **2020**, *27*, 188–196. [CrossRef]
2. Tao, W.; Zhu, M.; Chen, S.; Cheng, X.; Wen, Y.; Zhang, W.; Negenborn, R.R.; Pang, Y. Coordination and Optimization Control Framework for Vessels Platooning in Inland Waterborne Transportation System. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 15667–15686. [CrossRef]
3. Yu, H.; Murray, A.T.; Fang, Z.; Liu, J.; Peng, G.; Solgi, M.; Zhang, W. Ship Path Optimization That Accounts for Geographical Traffic Characteristics to Increase Maritime Port Safety. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 5765–5776. [CrossRef]
4. Xiao, Z.; Fu, X.; Zhao, L.; Zhang, L.; Teo, T.K.; Li, N.; Zhang, W.; Qin, Z. Next-Generation Vessel Traffic Services Systems—From "Passive" to "Proactive". *IEEE Intell. Transp. Syst. Mag.* **2023**, *15*, 363–377. [CrossRef]

5. Giannopoulos, A.; Spantideas, S.; Kapsalis, N.; Karkazis, P.; Trakadas, P. Deep Reinforcement Learning for Energy-Efficient Multi-Channel Transmissions in 5G Cognitive HetNets: Centralized, Decentralized and Transfer Learning Based Solutions. *IEEE Access* **2021**, *9*, 129358–129374. [CrossRef]
6. Trakadas, P.; Masip-Bruin, X.; Facca, F.M.; Spantideas, S.T.; Giannopoulos, A.E.; Kapsalis, N.C.; Martins, R.; Bosani, E.; Ramon, J.; Prats, R.G.; et al. A Reference Architecture for Cloud–Edge Meta-Operating Systems Enabling Cross-Domain, Data-Intensive, ML-Assisted Applications: Architectural Overview and Key Concepts. *Sensors* **2022**, *22*, 9003. [CrossRef] [PubMed]
7. Aledhari, M.; Razzak, R.; Parizi, R.M.; Saeed, F. Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Access* **2020**, *8*, 140699–140725. [CrossRef] [PubMed]
8. Wahab, O.A.; Mourad, A.; Otrok, H.; Taleb, T. Federated Machine Learning: Survey, Multi-Level Classification, Desirable Criteria and Future Directions in Communication and Networking Systems. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 1342–1397. [CrossRef]
9. Niknam, S.; Dhillon, H.S.; Reed, J.H. Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Commun. Mag.* **2020**, *58*, 46–51. [CrossRef]
10. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Artificial Intelligence and Statistics, Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
11. Zetas, M.; Spantideas, S.; Giannopoulos, A.; Nomikos, N.; Trakadas, P. Empowering 6G maritime communications with distributed intelligence and over-the-air model sharing. *Front. Commun. Netw.* **2024**, *4*, 1280602. [CrossRef]
12. Nomikos, N.; Giannopoulos, A.; Kalafatelis, A.; Özduran, V.; Trakadas, P.; Karagiannidis, G.K. Improving Connectivity in 6G Maritime Communication Networks with UAV Swarms. *IEEE Access* **2024**, *12*, 18739–18751. [CrossRef]
13. Angelopoulos, A.; Giannopoulos, A.; Nomikos, N.; Kalafatelis, A.; Hatziefremidis, A.; Trakadas, P. Federated Learning-Aided Prognostics in the Shipping 4.0: Principles, Workflow, and Use Cases. *IEEE Access* **2024**, *12*, 6437–6454. [CrossRef]
14. Nomikos, N.; Giannopoulos, A.; Trakadas, P.; Karagiannidis, G.K. Uplink NOMA for UAV-aided maritime Internet-of-Things. In Proceedings of the 2023 19th International Conference on the Design of Reliable Communication Networks (DRCN), Vilanova i la Geltru, Spain, 17–20 April 2023; pp. 1–6.
15. Wang, H.; Yan, R.; Au, M.H.; Wang, S.; Jin, Y.J. Federated learning for green shipping optimization and management. *Adv. Eng. Inform.* **2023**, *56*, 101994. [CrossRef]
16. Giannopoulos, A.; Nomikos, N.; Ntroulias, G.; Syriopoulos, T.; Trakadas, P. Maritime Federated Learning for Decentralized On-Ship Intelligence. In Proceedings of the Artificial Intelligence Applications and Innovations, León, Spain, 14–17 June 2023; Maglogiannis, I., Iliadis, L., MacIntyre, J., Dominguez, M., Eds.; Springer: Cham, Switzerland , 2023; pp. 195–206.
17. Han, C.; Yang, T. Privacy Protection Technology of Maritime Multi-agent Communication Based on Part-Federated Learning. In Proceedings of the 2021 IEEE/CIC International Conference on Communications in China (ICCC Workshops), Xiamen, China, 28–30 July 2021; pp. 266–271. [CrossRef]
18. Liu, W.; Xu, X.; Wu, L.; Qi, L.; Jolfaei, A.; Ding, W.; Khosravi, M.R. Intrusion Detection for Maritime Transportation Systems with Batch Federated Aggregation. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 2503–2514. [CrossRef]
19. Ma, X.; Jiang, Q.; Shojafar, M.; Alazab, M.; Kumar, S.; Kumari, S. DisBezant: Secure and Robust Federated Learning Against Byzantine Attack in IoT-Enabled MTS. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 2492–2502. [CrossRef]
20. Qin, Z.; Ye, J.; Meng, J.; Lu, B.; Wang, L. Privacy-Preserving Blockchain-Based Federated Learning for Marine Internet of Things. *IEEE Trans. Comput. Soc. Syst.* **2022**, *9*, 159–173. [CrossRef]
21. Zhang, Z.; Guan, C.; Chen, H.; Yang, X.; Gong, W.; Yang, A. Adaptive Privacy-Preserving Federated Learning for Fault Diagnosis in Internet of Ships. *IEEE Internet Things J.* **2022**, *9*, 6844–6854. [CrossRef]
22. Shuangzhong, W.; Zhang, Y. Multi-Level Federated Network Based on Interpretable Indicators for Ship Rolling Bearing Fault Diagnosis. *J. Mar. Sci. Eng.* **2022**, *10*, 743. [CrossRef]
23. Zhao, H.; Ji, F.; Li, Q.; Guan, Q.; Wang, S.; Wen, M. Federated Meta-Learning Enhanced Acoustic Radio Cooperative Framework for Ocean of Things. *IEEE J. Sel. Top. Signal Process.* **2022**, *16*, 474–486. [CrossRef]
24. Zeng, X.; Chen, M.; Li, H.; Wu, X. A Data-Driven Intelligent Energy Efficiency Management System for Ships. *IEEE Intell. Transp. Syst. Mag.* **2023**, *15*, 270–284. [CrossRef]
25. Wu, B.; Li, G.; Zhao, L.; Aandahl, H.I.J.; Hildre, H.P.; Zhang, H. Navigating Patterns Analysis for Onboard Guidance Support in Crossing Collision-Avoidance Operations. *IEEE Intell. Transp. Syst. Mag.* **2022**, *14*, 62–77. [CrossRef]
26. Hesselbarth, A.; Medina, D.; Ziebold, R.; Sandler, M.; Hoppe, M.; Uhlemann, M. Enabling Assistance Functions for the Safe Navigation of Inland Waterways. *IEEE Intell. Transp. Syst. Mag.* **2020**, *12*, 123–135. [CrossRef]
27. Zhang, J.; Han, G.; Sha, J.; Qian, Y.; Liu, J. AUV-Assisted Subsea Exploration Method in 6G Enabled Deep Ocean Based on a Cooperative Pac-Men Mechanism. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 1649–1660. [CrossRef]
28. Wu, J.; Song, C.; Ma, J.; Wu, J.; Han, G. Reinforcement Learning and Particle Swarm Optimization Supporting Real-Time Rescue Assignments for Multiple Autonomous Underwater Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 6807–6820. [CrossRef]
29. Lin, C.; Han, G.; Tao, Q.; Liu, L.; Shah, S.B.H.; Zhang, T.; Li, Z. Underwater Equipotential Line Tracking Based on Self-Attention Embedded Multiagent Reinforcement Learning Toward AUV-Based ITS. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 8580–8591. [CrossRef]
30. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.

31. Hu, X.; Chu, L.; Pei, J.; Liu, W.; Bian, J. Model complexity of deep learning: A survey. *Knowl. Inf. Syst.* **2021**, *63*, 2585–2619. [CrossRef]
32. Moubayed, A.; Shami, A. Softwarization, Virtualization, and Machine Learning for Intelligent and Effective Vehicle-to-Everything Communications. *IEEE Intell. Transp. Syst. Mag.* **2022**, *14*, 156–173. [CrossRef]
33. Sacco, A.; Esposito, F.; Marchetto, G. A Federated Learning Approach to Routing in Challenged SDN-Enabled Edge Networks. In Proceedings of the 2020 6th IEEE Conference on Network Softwarization (NetSoft), Ghent, Belgium, 29 June–3 July 2020; pp. 150–154. [CrossRef]
34. Guan, W.; Wang, K. Autonomous Collision Avoidance of Unmanned Surface Vehicles Based on Improved A-Star and Dynamic Window Approach Algorithms. *IEEE Intell. Transp. Syst. Mag.* **2023**, *15*, 36–50. [CrossRef]