Online training for fuel oil consumption estimation: A data driven approach

Dimitrios Kaklis
Department of Informatics and Telematics
Harokopio University of Athens,
Danaos RC, NCSR Demokritos
Athens, Greece

Iraklis Varlamis
Department of Informatics and Telematics
Harokopio University of Athens
Athens, Greece

George Giannakopoulos
Institute of Informatics & Telecoms
NCSR Demokritos
Athens, Greece

Constantine Spyropoulos
Institute of Informatics & Telecoms
NCSR Demokritos
Athens, Greece

Takis J. Varelas
Danaos Research Center
Piraeus Greece

***Abstract*—Estimating the Fuel Oil Consumption (FOC) of a vessel is a critical task for the maritime industry, affecting route planning and the overall management of the vessel's operation and maintenance. Consumption is strongly coupled with the operation of the Main Engine (ME), but also with the environmental conditions (i.e., weather, ocean-energy spectrum) and the hydrodynamic features (i.e., resistance, propulsion) of the vessel. Current research shows that a multitude of features collected either from the AIS (Automatic Identification System) or on-board sensors can assist to the continuous prediction of FOC. Even when a FOC estimation model is perfectly trained on a specific vessel, its performance may degrade over time, when new weather conditions apply or when the hydrodynamics of the vessel change over time, due to fouling, aging and negligent maintenance. This work presents an online learning framework that employs a custom encoding-decoding Neural Network scheme and real-time data from various on-board sensors, to appropriately update FOC estimation models. The model is able to adapt to newly acquired data using a temporally-aware batch scheme, that samples from the initial training set using a custom auto-encoder.**

***Index Terms*—online learning, concept shift, FOC estimation, multivariate spline regression, LSTM**

I. Introduction

The survey of related literature on the estimation of vessel operational costs and more specifically of Fuel Oil Consumption reveals a shift from mathematical (white box) models to more data driven approaches (black box models), which combine big data and machine learning in order to train the appropriate predictive models [2]. Even when these models are fine-tuned on a specific vessel, or type of vessel, with respect to its hydrodynamics performance and operational profile, their performance quickly degrades because of the changing conditions (weather, load, vessel status etc.) that occur during the vessel life-cycle. These changes may trigger completely different prediction tasks that the trained model has never experienced before, resulting in poor performance.

The ability to learn different tasks in a sequential manner is crucial for neural networks and it has been a research topic for many years. The problem arises from the fact that it is difficult to incrementally train a neural network to learn a function that is approximated from non-stationary data generated from different distributions. This happens mainly because Neural Networks, especially complex ones, tend to over-fit on the data patterns they see more frequently and "forget" others, a problem that leads to poor generalization performance over time, which is described as "catastrophic forgetting" [16]. The standard way of dealing with catastrophic forgetting in machine learning is by jointly mixing new training examples with old ones, and re-training the model offline. As it is obvious, this process may require a vast amount of computational resources, especially for large high dimensional datasets, and represents a time-consuming and non-scalable solution.

An ideal online learning system would be able to absorb new information without the need to store the entire training set or repeating the training process from scratch on the whole dataset. This paper proposes an online learning framework that resembles the way the Hippocampal complex (HC) of the brain adapts to new tasks, while remembering past knowledge through selective sampling and retraining.

The novelty of the proposed scheme lies in the use of a custom recurrent encoding-decoding scheme that automates the procedure of discovering temporal patterns between tasks, which are useful in the long term process of continuous training of the model and reduction of catastrophic forgetting. With the proposed approach the model is able to consolidate recent memories into a long term buffer-rehearsal storage [18]. With a pseudo-rehearsal (memory-replay) strategy we avoid storing previous training samples, resulting in a computationally more efficient predictive scheme.

Section II provides an overview of the related work in the field. Section III presents the proposed approach for efficient estimation of FOC, under the online learning setup. Section IV illustrates the experimental results achieved so far and Section V briefly summarizes our findings and discusses the next steps of this work.

II. RELATED WORK

The majority of solutions for FOC estimation in the maritime sector is found in the field of operational research under the broader task of weather routing, with more recent works focusing on data-driven approaches [7, 11, 12]. They are supported by the exponential growth in vessel data availability in the maritime sector either from on-board monitoring systems or from AIS (Automatic Identification System) that describe the state of the vessel during a voyage. These approaches aim to improve estimation accuracy, whilst disregarding more practical problems, such as performance and long term robustness.

This streamlined process, from data collection to model training, faces two major issues. The first is the lack of data for all vessels and all types of navigation states and the second is the high-volatility of factors that affect the prediction performance (e.g. vessel condition, load, weather conditions etc.). Consequently, pre-trained static models are incapable to incorporate newly acquired data-instances and appropriately adapt to new conditions and environments. Since on-board sensors and AIS can provide useful information about the vessel and its environment at real time, and this information is continuously updated, data-driven methods [1] must be properly adapted to take advantage of new data.

A potential solution in this direction is online learning [9]. In online learning, a model that can be initially trained on historical data, is being trained continuously to adapt to the new states and conditions of the vessel, or even adapt to different vessels capturing their inherent characteristics. The main idea and motivation behind adopting online training in the maritime sector is to train and deploy a large generic model that can be applied to different vessel types, engine specs and weather states in order to cover the business need for continuous and accurate estimation and prediction of a vessel's FOC while on route.

The pertinent literature that deals with the problem of online learning over sequentially and disjointedly observed data falls in the following categories: 1) works that modify the neural network architecture on demand, by mapping the input space to a network structure, such as Self Organizing Incremental Neural Networks (SOINNs) [20]; 2) works that adapt the NN weights by adding a regularization term in the loss function [4]. that minimizes the distance between the weights distributions assigned to the model on each training batch; [14]. 3) methods that leverage information from different tasks and use it at training time (multitask learning paradigm) [15], or store previous samples in a rehearsal buffer; [3]. 4) works that use auto-encoders to generate streams that resemble data batches previously seen by the model [13].

Our contributions compared to related work are summarized in the following: 1) the first application of online learning for FOC estimation, 2) a novel hybrid auto-encoding/translating scheme, which can be utilized as a concept drift detection tool; 3) a fully automated "memory induced" sample generation process that replaces the need to use a buffer for memory replay, since "memory" samples are generated on-the-fly from the newly acquired samples; 4) a Recurrent Deep NN architecture that estimates FOC using real-time measurements from on-board monitoring systems of the vessel.

III. PROPOSED APPROACH

A. FOC estimation model

The dynamic estimation of FOC based on vessel state and environmental conditions can be examined as a multivariate time-series prediction problem, that takes into account the actual values, as well as their recent history, and captures the information hidden in the values evolution over time. Based on the superiority of Long short-term memory (LSTM) models over traditional time-series prediction methods (e.g. ARIMA) [19], LTSM NNs are chosen as the basis of our solution.

1) The basic LSTM NN for FOC estimation: LSTM is an artificial Recurrent Neural Network (RNN) architecture [8], which has been extensively used for time-series prediction [10] tasks, especially because it is not so sensitive to the length of gaps between important events in a time-series.

The input of the LSTM network at timestep t_u comprises N time-series, one for each feature of interest (speed through waterW, wind speed, wind angle etc) and in order to use the recent history of values in each feature, we employ a fixed-length time-window (time-lag of length m). As a consequence, the window contains the values for each time step $t_i \in [t_{k-m}, t_u]$ for the weather and vessel state features that are used for the estimation of FOC at time t_u, resulting in N time-series, of length $m+1$, of the form $[F_{N(u-m)}, ..., F_{N(u-1)}, F_{N(u)}]$, for each feature F_N. Given a sequence of consecutive time-steps, and a multivariate feature set, we get the following correspondence between the input and the output of the LSTM:

$$\begin{bmatrix} [F_{1(u-m)} \dots F_{j(u-m)} \dots F_{N(u-m)}] \\ \vdots \\ [F_{1(i)} \dots F_{j(i)} \dots F_{N(i)}] \\ \vdots \\ [F_{1(u)} \dots F_{j(u)} \dots F_{N(u)}] \end{bmatrix} \longrightarrow \begin{bmatrix} FOC_{u-m} \\ \vdots \\ FOC_i \\ \vdots \\ FOC_u \end{bmatrix}, \quad (1)$$

where N is the number of monitored features (and respectively of the time-series fed to the LSTM), m is the window length, and $F_{j(i)}$ is the value of feature $j \in [1, N]$ at timestamp t_i. FOC_i is the FOC value that we want to predict.

2) SplineLSTM*:* In a previous work [11] we demonstrated the approximation capabilities of spline-based regression models [5] and their ability to adapt to the linear and non-linear patterns that exist between dependent and independent variables, as those that describe the underlying function that approximates FOC. A spline regression of degree d partitions the input space in sub-domains separated by k knots. Each domain is approximated by different polynomial of degree up to d. Splines of order d have continuous $d-1$ derivatives, a property that balances the trade off between goodness-of-fit and smoothness of the spline interpolant, and results in a predictive scheme with good generalization capabilities.

An example of spline regression and the polynomials that are constructed in training time given a multi-variate feature set: $x_1, \ldots, x_N$ and a target variable y is described as follows:

$$y = f(x) = \begin{cases} H_{1i}(x_{1i}, ..., x_{1N_i})b_{1i}, & x_{1i} \in [t_i, t_i + \Delta t] \\ \vdots & \\ H_{ki}(x_{ki}...x_{kN_i})b_{ki}, & x_{ki} \in [t_i + (k-1)\Delta t, t_i + k\Delta t], \\ 0 & otherwise \end{cases} \tag{2}$$

where t_i is the corresponding time-step of observation x_i, k is the number of knots of the pre-trained SplineModel$_i$, $H(x)$ are piece-wise continuous hinge functions of order $d \geq 1$ defined on subsequent time intervals, and b_{in} are the regression coefficients of the pre-trained spline models, where $i \in [1, k]$.

The proposed spline-based LSTM network uses the knowledge gained in the pre-spline-training step for providing an extended input vector of size $k + N$ in each case, where k is the number of knots of the respective spline function. The proposed Spline network evaluates each feature x_i on the corresponding hinge function H_i generated from Spline regression creating a k dimensional vector that quantifies the impact of each feature in FOC estimation. This vector guides the network to form spatial-aware embeddings that help the model learn a different set of functions for different sub-domains of interest.

With this transformation the LSTM network *looks back* m time steps to form the hidden state units h_{t-1}. The hidden state acts as the NN memory for it holds information on data the network has *seen* before. The input vector is constructed by moving time windows that comprise: 1) $(F_1, \ldots, F_N)$ values, 2) k values generated from evaluating our feature set $(F_1, \ldots, F_N)$ values at each of the k knots of the pre-trained Spline model, and 3) the corresponding FOC values. The resulting input vector at time instance t_i takes the form:

$$\begin{bmatrix} [F_{1(1)}, \ldots, F_{N(1)}, H_{1(1)}(F_1), \ldots, H_{1(k)}(F_{1'})] \\ \vdots \\ [F_{1(i)}, \ldots, F_{N(i)}, H_{i(1)}(F_i), \ldots, H_{i(k)}(F_{i'})] \\ \vdots \\ [F_{1(m)}, \ldots, F_{N(m)}, H_{m(1)}(F_m), \ldots, H_{m(k)}(F_{m'})] \end{bmatrix} \rightarrow FOC_i \tag{3}$$

where m is the number of the previous time steps used to form the initial $2D$ vector of velocity and its mean value, m is the step used to set-up the time window vectors for the hidden LSTM units and H_{ik} is the i-th Hinge function of the pre-trained spline regression model.

B. The formulation of the online learning task

Concept Drift detection and Elastic Weight Consolidation (EWC) [14] are two frequently combined tasks in online learning that aim to detect when re-training is needed and adjust the ratio between samples retrieved from memory and new samples in each re-training step. According to EWC, if we assume that data comes from n statistically independent distributions $A_1 \ldots A_n$ it is possible to separate the task in n consecutive sub-tasks with the overall loss function of EWC taking the form:

$$L(\theta) = \sum_{i=1}^{n} L_{A_i}(\theta_i) + \frac{1}{2}\sum_{i=1}^{n-1} \lambda_i * F_i(\theta_{A_{i+1}} - \theta_{A_i})^2 \tag{4}$$

where θ_i denotes the set of the learned parameters of the system that correspond to task A_i, $L_{A_i}(\theta)$ is the loss function for task A_i, θ_{A_i} are the parameters for task A_i, λ_i is a factor that quantifies the memory window of the system while traversing from task A_i to task A_{i+1}, and F_i is the Fisher information matrix [17] that approximates the expected overlapping information between two consecutive tasks (i.e. A_{i-1} and A_i).

The parameter λ in Eq. 4 introduces the notion of memory to the learning system while it adapts to new data samples. Based on the work presented in [6], given a specific training policy P and a given domain with periodic shifts of interest in the training set T_s, we model the expected performance $\bar{f}$ of a learning system P as a function of signal to noise ratio Z between different tasks on this domain of interest:

$$\bar{f} = E(f(Z)) = \sum_{i=0}^{\rho T_s - 1} f_i q(f_i) \tag{5}$$

where $\rho = \frac{r}{T_s}$ with r being the memory window of the system, f_i is the distinct value of $f(Z)$ and $q(f_i)$ is the probability distribution of f over Z.

For $\rho > 1$ equation (5) becomes

$$\bar{f} = m + (\bar{M} - m)(\frac{1}{\rho T_S}\sum_{i=0}^{n} \bar{f}_N(Z_i)) \tag{6}$$

where $\bar{m}$ and $\bar{M}$ is the minimum and maximum performance of the algorithm respectively and $f_N(Z)$ is the normal characteristic function of the system as described in [6] depending only from signal to noise(N) ratio Z and some constants a, b. Replacing the term $\rho T_s = r$ in equation (6) with λ we can observe that the expected performance of the learning system $E(f)$, in a domain where periodic concept shifts are defined, is a function depending only from the memory window λ.

The use of a deep learning model ($SplineLSTM$) in an online learning setup raises the need to deal with abrupt or gradual shifts in sample distribution statistics. Using a drift detector to detect such changes, the online learning framework introduces samples from past memory in order to assist the model to adapt FOC estimation to newly acquired batches of data.

C. A custom Recurrent Auto-Encoder (RAE) for concept drift detection and sample generation

Auto-Encoders (AE) have been extensively used in numerous applications from image generation to language modeling. An extension of AEs are the Variational Auto Encoders (VAE), generative models in statistical terms, that aim to approximate the posterior distribution of the latent space generated by the encoder using neural networks. VAEs map a single point of the input space to a distribution in latent space, and then try to approximate the posterior probability distribution of the decoder in order to re-construct the input space. They differ from AEs in the loss function. The loss function of a standard VAE comprises of two terms and is defined as follows:

$$L_{VAE} = ||x - \bar{x}||^2 + KL[N(\mu_x, \sigma_x), N(0, 1)] \quad (7)$$

where the first term represents the reconstruction error, and the second term is the Kullback-Leibler (KL) divergence confining the latent space distribution to a Gaussian, normal like distribution $\simeq N(0, 1)$. The first term is minimizing the difference between the input space and the generated reconstructed distribution of the decoder. The second term is responsible for a regularized latent space by ensuring the distributions generated by the encoder are as close as possible to a normal Gaussian like distribution.

In order to develop an efficient sampling strategy for our $SplineLSTM$ model we introduce a NN-based scheme that, given an evolving stream of data ($A \rightarrow B \rightarrow C \rightarrow \ldots Z$), extracts "meaningful" samples from older batches of data. The scheme minimizes the generalization error on newly acquired batches and maintains an acceptable accuracy on already processed ones.

The trade-off between the accuracy (or error) achieved in two consecutive data batches $\mathcal{A}$ and $\mathcal{B}$ of the following form

$$\mathcal{A}: \begin{bmatrix} [x_{11}, \ldots, x_{1n}] \\ \vdots \\ [x_{i1}, \ldots x_{in}] \\ \vdots \\ [x_{k1}, \ldots x_{kn}] \end{bmatrix} \quad and \quad \mathcal{B}: \begin{bmatrix} [y_{11}, \ldots, y_{1n}] \\ \vdots \\ [y_{i1}, \ldots y_{in}] \\ \vdots \\ [y_{k1}, \ldots y_{kn}] \end{bmatrix}, \quad (8)$$

with k being the number of features and n being the number of samples, can be modeled as a bounding error increase problem, using a base learner B_L as follows:

$$|\mathcal{L}(B_L, x_{\mathcal{A}}) - \mathcal{L}(B_L, y_{\mathcal{B}})| < \phi \quad (9)$$

where ϕ is the maximum accepted error difference between two consecutive tasks A, B and $\mathcal{L}$ is the evaluation function.

The model architecture of the neural sampling scheme consists of two AEs, we call them translating auto-encoders $trAE_A$ and $trAE_B$ since they reconstruct information from one task to the internal memory layer and then back to another task, and one intermediate memory layer M as depicted in Figure 1. More specifically, the first auto-encoder ($trAE_A$) is responsible for compressing the information to represent useful time instances for task A and decoding it back to the memory layer. The memory layer M holds information for both tasks A and B as serves as the last layer of the $trAE_A$ and the first layer of the $trAE_B$. Intuitively the memory layer M represents the latent space representation of the translating scheme. The second auto-encoder ($trAE_B$) is responsible for compressing the information from the memory layer and decoding it back to represent task B.

Based on the regularization term of VAE's loss function as given in Equation 7 we introduce a custom regularization term for the loss function of the translating scheme that

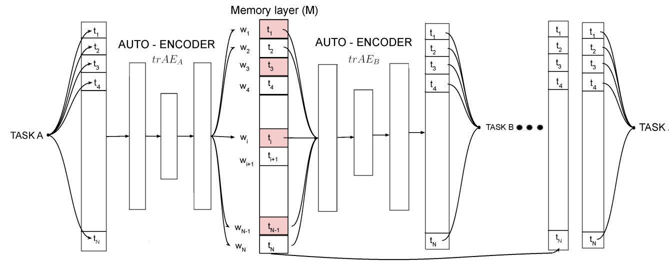


Figure 1: Sampling scheme for fixed length sequences

quantifies the amount of information (time-steps) the model needs to remember in order to optimally reconstruct the chain of different tasks (distributions) that is given at training time. Consequently, the loss function of the proposed model for two tasks A, B is modeled as follows:

$$L_{tr_{A->B}} = ||\bar{x} - \bar{y}||^2 + KL[N(\mu_{\hat{x}}, \sigma_{\hat{x}}), N(\mu_{\bar{y}}, \sigma_{\bar{y}})] \quad (10)$$

where $N(\mu_{\hat{x}}, \sigma_{\hat{x}})$ is the output distribution of the first translating scheme $trAE_A$, which is fed to the memory layer M, and $N(\mu_{\bar{y}}, \sigma_{\bar{y}})$ is the reconstructed output distribution of the second translating scheme $trAE_B$. Intuitively the loss function of Equation 10 is responsible for guiding the model to construct a latent space that represents a memory module holding information from past tasks in order to be able to adapt to new ones ideally.

The memory module M exists as an entity in the latent space representation of the translating scheme from task A to task B. Consequently, given a sequential training of a base learner B_L on a series of consecutive tasks, A to …Z, the memory layer (M_Z) that is responsible for the sampling of task Z can be modeled as a function of all the memory layers that correspond to the tasks preceding task Z, $M_A, \ldots M_Y$, as follows:

$$\begin{aligned} \hat{M_Z}(t) &= (M_Y \circ M_X \circ \ldots M_A)(t) = \\ &= \textstyle\sum_{i=0}^{N} w_i \hat{M}_Z(M_Y(\ldots(M_A(t))\ldots)) \end{aligned} \quad (11)$$

where N is the number of time-steps in task Z, and w_i are the weights learned by the last translation scheme tr_Z.

Each memory module $\hat{M_i}(t)$ is passed as input to another auto-encoding scheme that learns again, to encode and eventually decode the generated memory into the next task. The loss function for the i_{th} translating scheme tr_i is modeled as:

$$L_{tr_i} = ||x_{M_i} - \bar{y_i}||^2 + KL[D(\mu_{M_{i-1}}, \sigma_{M_{i-1}}), D(\mu_i, \sigma_i)] \quad (12)$$

The proposed translating auto-encoding scheme can be used as a concept drift detector by monitoring its loss function (i.e. 10). We can model the reconstruction error between two consecutive batches of data A, B as a bounding error problem as follows:

$$\mathcal{L}(tr_{A->B}) < \lambda \quad (13)$$

When the reconstruction error between two consecutive tasks A, B reaches the maximum error threshold λ we detect a concept drift. It is then possible to identify the boundaries of the input data distribution, where the drift occurred, and enable the generation of past training samples.

IV. Preliminary experimental results

This section demonstrates the experimental results from the use of the $SplineLSTM$ in the online learning task of FOC estimation. For simplicity, we assume that the consecutive tasks are of fixed length, which removes the need for a drift detector. First we evaluate the conventional offline learning methods and then present the performance of the proposed learner, that takes advantage of the memory-based generated samples, when retraining the model between tasks.

A. Dataset description

The dataset of this study contains $24 * 10^5$ of minutely measurements from sensors on board of an existing containership with a carrying capacity of 3000 TEUs [1]. The dataset spans roughly 6 months of the vessel route history and its values correspond to a multitude of features describing a vast majority of different round-trip voyages at different periods and geographical locations[2]. More specifically the feature set comprises: 1) vessel's speed through water, 2) wind speed, 3) wind angle, 4) swell wave height, 5) bearing (vessel course), 6) mid draft of the vessel and 7) and the corresponding FOC.

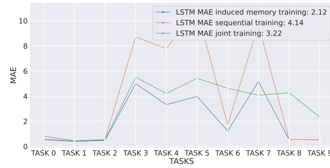
B. Experimental evaluation

The initial dataset is split into five statistically independent subsets with each one comprising $\simeq 5 * 10^4$ (1 month) instances. The statistical independence across different datasets has been preserved utilizing the Kruskal-Wallis which is a non-parametric method for testing whether samples originate from the same distribution. Each subset is then divided into ten fix-length consecutive tasks, each one comprising $\simeq 5{*}10^3$ observations (7 hours).

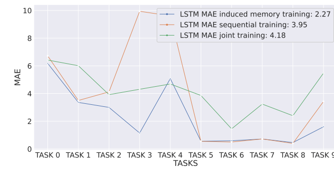
The base learner B_L (*SplineLSTM*) learns online how to estimate FOC using one of the following strategies:

1) B_L is first trained on $task_A$ and then on $task_B$ in a sequential manner, with the risk to forget task A **(Sequential Training)**
2) B_L is first trained on $task_A$ and then trained on $task_A \cup task_B$ **(Joint Training)**
3) B_L is first trained on $task_A$ and then trained on $task_B$ and on a memory batch $task_{AB_S}$ generated as explained in Section III-C (between $task_A$ and $task_B$): $task_{AB_S} \cup task_B$ **(Memory Induced Training)**

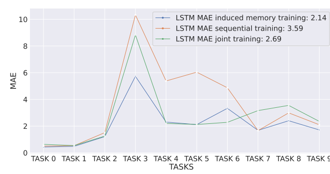
The plots in Figure 2 demonstrate the performance (i.e. MAE) of the $SplineLSTM$ using the three re-training strategies in the 5 subsets. Mean Absolute Error(MAE) was measured separately on each task as we progressively continue the training procedure and overall for all the tasks combined. In the legend of the graphs we display the average value of MAE for all the tasks combined, for each training mode, for all the datasets.



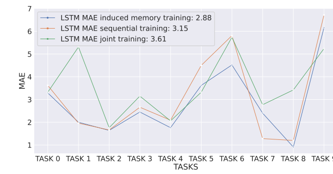
(a) Dataset 1



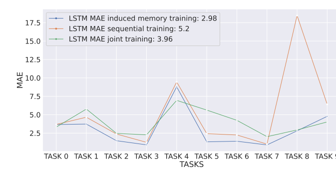
(b) Dataset2



(c) Dataset 3



(d) Dataset 4



(e) Dataset 5

Figure 2: Base learner(*SplineLSTM*) performance through 10 tasks for three modes of online training

Table I depicts the average improvement in Mean Absolute Error (MAE) achieved by Joint or Memory Induced training over the baseline Sequential training approach, in all the tasks of each subset. Negative values denote a reduction in the MAE (i.e. prediction performance improvement). All the improvements are statistically significant at the 99% confidence interval using the 3σ rule ($\bar{x} \pm 3\sigma\sqrt{n}$, $\bar{x}$: the mean of the population, σ: standard deviation, n: the sample size). From the results, we can see that the proposed memory induced training strategy performs better (in average) than joint and sequential training in all subsets. This is very promising, since the proposed method does not need a replay memory buffer and thus is more memory efficient than the joint training method.

Table I: Pairwise MAE Percentage change of *SplineLSTM* on the 5 subsets

	Subset 1	Subset 2	Subset 3	Subset 4	Subset 5
Joint vs Sequential	-22.2%	-5.50%	-25.1%	-12.7%	-23.7%
Memory Induced vs Sequential	-48.8%	-42.5%	-40.4%	-20.2%	-42.6%
Memory Induced vs Joint	-34.2%	-45.7%	-20.4%	-20.2%	-24.7%

V. Conclusions and next steps

After carefully examining the literature on data driven FOC estimation techniques in the maritime sector, we realised the

[1]"The twenty-foot equivalent unit (abbreviated TEU or teu) is an inexact unit of cargo capacity, often used for container ships and container ports. It is based on the volume of a 20-foot-long (6.1 m) intermodal container, a standard-sized metal box which can be easily transferred between different modes of transportation, such as ships, trains, and trucks."

[2]The dataset is available, in sanitized form, upon request to the first of authors.

importance of continuous model updating, and realized the absence of online training methods that are able to overcome the problem of catastrophic forgetting.

The novel SplineLSTM method that we proposed for FOC estimation, allows the integration in a hybrid translating-auto-encoding scheme with a memory module that is able to generate past training samples for more efficient updates through re-training on task changes. This allows the SplineLSTM model to quickly adapt to newly acquired data and changing conditions.

The experimental evaluation assessed the performance of our baseline FOC estimator in ten consecutive tasks using three different model update strategies: sequential model training, joint re-train with all the samples of the previous tasks, and an on-the-fly sample generation strategy using an auto-encoder scheme that is trained to capture information between consecutive tasks.

Using the proposed update strategy, the baseline FOC estimator manages to adapt to task changes and reduce the prediction error across tasks. The average MAE in all datasets across all tasks is improved when memory induced training is applied.

In this work we assumed fixed length tasks and thus the ability of the proposed method to also work as a drift detector has not been employed. It is part of our ongoing work in this field to experiment with unknown boundary tasks in order to evaluate its ability to early detect task changes and quickly confront catastrophic forgetting.

In a different direction, but related to the maritime tasks we examined in this study, we are also working on the development of models that efficiently generalise on different types of vessels. Our next steps in this field will examine the concepts of transfer learning and knowledge distillation, in order to train larger, more abstract and more adaptive models that can easily adapt to different vessel types, without much re-training. Therefore one of our main future objectives is to extend the proposed framework, so that it can build generic cross - vessel - generative predictive models for FOC estimation for different types of vessels.

VI. Acknowledgements

This Publication was supported partially by the program of Industrial Scholarships of Stavros Niarchos Foundation, and partially by the SmartShip project, an European Union's Horizon 2020 research and Innovation programme under the Marie Skłodowska-Curie Grant Agreement No 823916. Access to industrial data has been provided by Danaos Shipping Co.

References

[1] F. Ahlgren, M. E. Mondejar, and M. Thern, "Predicting dynamic fuel oil consumption on ships with automated machine learning," *Energy Procedia*, vol. 158, pp. 6126–6131, 2019.

[2] A. Coraddu, L. Oneto, F. Baldi, and D. Anguita, "Vessels fuel consumption forecast and trim optimisation: a data analytics perspective," *Ocean Engineering*, vol. 130, pp. 351–370, 2017.

[3] C. Davalas, D. Michail, C. Diou, I. Varlamis, and K. Tserpes, "Computationally efficient rehearsal for online continual learning," *21st International Conference on Image Analysis and Processing (ICIAP2021)*, 2022.

[4] A. A.-B. David Martinez-Rego, Oscar Fontenla-Romero, "Nonlinear single layer neural network training algorithm for incremental nonstationary and distributed learning scenarios," *Pattern Recognition*, 2017.

[5] J. H. Friedman, "Multivariate adaptive regression splines," *The annals of statistics*, vol. 19, no. 1, pp. 1–67, 1991.

[6] G. Giannakopoulos and T. Palpanas, "Revisiting the effect of history on learning performance: the problem of the demanding lord," *Knowledge and information systems*, vol. 36, no. 3, pp. 653–691, 2013.

[7] C. Gkerekos and I. Lazakis, "A novel, data-driven heuristic framework for vessel weather routing," *Ocean Engineering*, vol. 197, p. 106887, 2020.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online learning: A comprehensive survey," *Neurocomputing*, vol. 459, pp. 249–289, 2021.

[10] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Communications Magazine*, vol. 57, no. 6, pp. 114–119, 2019.

[11] D. Kaklis, G. Giannakopoulos, I. Varlamis, C. D. Spyropoulos, and T. J. Varelas, "A data mining approach for predicting main-engine rotational speed from vessel-data measurements," in *Proceedings of IDEAS 2019*, 2019, pp. 1–10.

[12] P. Karagiannidis and N. Themelis, "Data-driven modelling of ship propulsion and the effect of data pre-processing on the prediction of ship fuel consumption and speed loss," *Ocean Engineering*, vol. 222, p. 108616, 2021.

[13] R. Kemker and C. Kanan, "Fearnet: Brain-inspired model for incremental learning," *In ICLR 2018*.

[14] J. Kirkpatrick, R. Pascanu, and J. V. Neil Rabinowitz, "Overcoming catastrophic forgetting in neural networks," *PNAS*, vol. 114, no. 13, p. 3521–3526, 2017.

[15] M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning," *arXiv preprint arXiv:1511.06114*, 2015.

[16] M. McCloskey and N. J.Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *Psychology of Learning and Motivation*, vol. 24, pp. 109–165, 1989.

[17] R. Pascanu and Y. Bengio, "Revisiting natural gradient for deep networks," *arXiv preprint arXiv:1301.3584*, 2013.

[18] A. Robins, "Consolidation in neural networks and in the

sleeping brain," *Connection Science*, vol. 8, no. 2, pp. 259–276, 1996.

[19] S. Siami-Namini, N. Tavakoli, and A. S. Namin, "A comparison of arima and lstm in forecasting time series," in *2018 17th IEEE ICMLA*. IEEE, 2018, pp. 1394–1401.

[20] C. Wiwatcharakoses and D. Berrar, "Soinn+, a self-organizing incremental neural network for unsupervised learning from noisy data streams," *Expert Systems with Applications*, vol. 143, p. 113069, 2020.