applied sciences

MDPI

Article

Deep Neural Network Modeling for CFD Simulations: Benchmarking the Fourier Neural Operator on the Lid-Driven Cavity Case

Paulo Alexandre Costa Rocha [1,2], Samuel Joseph Johnston [3], Victor Oliveira Santos [1,*], Amir A. Aliabadi [1], Jesse Van Griensven Thé [1,3,4] and Bahram Gharabaghi [1]

[1] School of Engineering, University of Guelph, 50 Stone Rd E, Guelph, ON N1G 2W1, Canada
[2] Mechanical Engineering Department, Technology Center, Federal University of Ceará, Fortaleza 60020-181, CE, Brazil
[3] Lakes Environmental Research Inc., 170 Columbia St W, Waterloo, ON N2L 3L3, Canada
[4] Department of Mechanical & Mechatronics Engineering, University of Waterloo, 200 University Avenue West, Waterloo, ON N2L 3G1, Canada
* Correspondence: volive04@uoguelph.ca

Abstract: In this work we present the development, testing and comparison of three different physics-informed deep learning paradigms, namely the ConvLSTM, CNN-LSTM and a novel Fourier Neural Operator (FNO), for solving the partial differential equations of the RANS turbulence model. The 2D lid-driven cavity flow was chosen as our system of interest, and a dataset was generated using OpenFOAM. For this task, the models underwent hyperparameter optimization, prior to testing the effects of embedding physical information on performance. We used the mass conservation of the model solution, embedded as a term in our loss penalty, as our physical information. This approach has been shown to give physical coherence to the model results. Based on the performance, the ConvLSTM and FNO models were assessed in forecasting the flow for various combinations of input and output timestep sizes. The FNO model trained to forecast one timestep from one input timestep performed the best, with an RMSE for the overall x and y velocity components of 0.0060743 $m \cdot s^{-1}$.

Keywords: physics-informed neural operator; partial differential equations; turbulence; Reynolds-averaged Navier–Stokes; deep learning; pytorch library

Citation: Costa Rocha, P.A.; Johnston, S.J.; Oliveira Santos, V.; Aliabadi, A.A.; Thé, J.V.G.; Gharabaghi, B. Deep Neural Network Modeling for CFD Simulations: Benchmarking the Fourier Neural Operator on the Lid-Driven Cavity Case. *Appl. Sci.* **2023**, *13*, 3165. https://doi.org/10.3390/app13053165

Academic Editors: Gholamreza Kefayati and Hasan Sajjadi

Received: 27 January 2023
Revised: 21 February 2023
Accepted: 23 February 2023
Published: 1 March 2023

1. Introduction

Throughout history, numerical modeling has emerged as an important tool for technological development; it is used for nearly all human innovations. In the ecosystem of numerical models, Computational Fluid Dynamics (CFD) is a class of tools that have dominated many scientific and engineering fields, from microfluid flows [1,2] to large scale flows [3–5], as well as specific device evaluation [6]. Despite this success, modern CFD approaches have drawbacks that limit their effectiveness. For example, there is no unique and general-purpose turbulence closure model, and the computational cost, in terms of time and CPU requirements, is usually very high, even for simple problems [7,8].

In this context, Machine Learning (ML) models appear as a candidate to overcome some of those issues. Recent works have shown that the ML paradigm can be employed in different problems, such as microscale transport between fluids [9], fluid flow through nanochannels [10], enzyme production optimization [11] and solid fuel performance prediction [12], attesting its multidisciplinary applications.

Recently, Deep Neural Networks (DNNs) have been used to solve Partial Differential Equations (PDEs) of related problems [13–15]. In this context, CFD studies have emerged as a natural path for implementing and testing those techniques. The examples are many, and they come because the evaluation of DNN models are applied to directly map the PDE

solution [16–19], as well as the insertion of ML models in parts of the PDE solution [15,20], especially for the closure problem [13,21]. The research reports solution acceleration in the order of 40–80 times [22].

As a recent advance in the use of DNNs to solve PDE-related problems, a new approach has been developed by researchers in the field. It consists of embedding physical information into the DNN model, which can be achieved in several ways [23], improving the model's computational performance when compared with traditional finite element methods [24]. One of these new approaches includes the insertion of physical constraints directly in the DNN models, which is known as a hard constraint. One can also impose a soft constraint, whereby the loss function used to assess the model performance is augmented to impose physical restraints [25]. Both practices were reported to improve accuracy and generalizability, while also reducing training times [26].

In this work, we applied both approaches, where the soft constraints were imposed by the boundary conditions (BCs), as well as the mass conservation law to assess its contribution to the model's predictions. On the other hand, as a hard constraint, we tested the Fourier Neural Operator (FNO) [27], which is a type of Neural Network (NN) layer with high physical appeal. It pertains to a new class of NN layers, which maps fields between different spaces (not necessarily Euclidean) and thus can learn the parametric dependence of the solution directly to an entire family of PDEs. To our best knowledge, there are still few results in the literature regarding the implementation and testing of FNO for turbulent flow simulations [28].

It is common in CFD studies to test and assess the proposed methodology on classical problems [29,30], whose main characteristics are their ubiquitous validation by the community, their simplicity to be set, their representation of complex characteristics and their indication of real-world problems. Considering this, we chose the lid-driven cavity flow problem to be modeled by the proposed DNNs.

The main objective of this work was to calibrate and test different DNN paradigms for the simulation of a flow under laminar and turbulent regimes. We tested their behavior by adding soft (boundary conditions and mass conservation law) and hard (Fourier Neural Operator) constraints. Finally, we discovered and assessed the best model, its hyperparameters and setup, including the number of timesteps to use when learning the flow dynamics.

From what was presented, the main contribution of this work is that recent research on turbulence is following a path of improving DNN results toward making them useful for real flow simulations. In this sense, our results give hints of which approaches/architectures may perform better, taking into consideration the accuracy as well as the computational load.

After this introduction, this paper presents the methodology (Section 2) regarding the data generation, as well as the DNNs' test configuration. In Section 3, we present the results, and we discuss them in Section 4. The conclusion and final remarks are provided in Section 5.

2. Materials and Methods

2.1. Data Generation

We used the lid-driven cavity problem [31–34] to generate data for the models. Lid-driven cavity flow is well known in the CFD research field because it provides complex flow structures and interactions and can be run in both laminar and turbulent flow regimes. For our cases, Reynolds numbers of 1000, 2000, 4000, 6000, 10,000 (test dataset), 14,000, 16,000, 18,000 and 20,000 were considered by the transient incompressible RANS solver (pisoFoam), which is part of the OpenFOAM CFD package [35,36]. The Re = 8000 and 12,000 cases were not used, aiming to leave a reasonable gap between the training and testing datasets. The spatial domain was divided into a 128×128 mesh, in a 0.1×0.1 m^2 cavity. This research work adopted the present resolution for the sake of computational load, which also impacts the tested ML models. Specific cavity lid velocity, in the range of

0.1 to 2.0 $m \cdot s^{-1}$, established predefined Reynolds numbers. For calculation convenience, the kinematic viscosity was taken as 10^{-5} $m^2 \cdot s^{-1}$, which is near the value for air.

Although the simulation time step was 5×10^{-4} s, the flow variables were recorded at every 0.1 s interval. The solution started with a fully static fluid and proceeded to simulate the flow up to 20 s, which was sufficient to reach a steady state for all the cases, before using the solution for the models.

A complete Direct Numerical Simulation (DNS) flow solution was generated employing OpenFOAM. This approach required large time and computer resources, when compared with the simpler RANS k-ε turbulence model for a minimal gain on the proof of concept for our case. Therefore, we proceeded with the analysis, employing the classical RANS k-ε model for validation, which was carried out using the data from [37], whose authors implemented a numerical method for 2D Steady Incompressible Navier–Stokes equations in streamfunction and vorticity formulations. For this task, the authors solved the lid-driven cavity flow to Reynolds numbers in the range of 1000–21,000. The authors provided the normal velocity components along a vertical and a horizontal line in the middle of the domain, i.e., the x component on the vertical line and the y component on the horizontal line. They used a high-order simulation to generate highly accurate data for the investigated Reynolds numbers. This approach permits taking values at critical points, which present highly variable shear stresses and vorticity.

2.2. Models' Architecture Setup

To compare the capacity of the ML models to precisely predict the flow characteristics, three DL models were built and tested using the PyTorch library for the Python language. These were, namely, Convolutional Long Short-Term Memory (ConvLSTM), 2D CNN + LSTM and the Fourier Neural Operator (FNO).

For each one, several architecture configurations were tested, with the aim to identify the ones that provide the greatest balance between performance and computational load. The models were implemented in a standard form, already including the boundary conditions data in the input dataset. A set of configurable hyperparameters was proposed, and because exhaustively testing possible configurations would incur a prohibitive computational time, 10 random configurations were tested for up to 50 epochs instead. We could perceive that it was enough to achieve the best results for each base model. The best architectures were chosen to be more deeply trained, using up to 300 epochs. Further details about the tested architectures for each model are presented in the next subsections.

During all the steps of the work, the training parameters were fixed. We selected the cases of Re = 1000, 2000, 4000, 6000, 14,000, 16,000, 18,000 and 20,000 for the data for training and the Re = 10,000 case for testing, to develop the models' architecture and compare their overall performance under the same conditions. The training and testing data included only the first 10 s of the simulation, as most flow dynamics were contained in this time frame. This simulation, with timesteps of 0.1 s, was divided into sequences of 3 inputs and 1 output, so our models were trained to forecast t_{i+3} given the input sequence $\{t_i, t_{i+1}, t_{i+2}\}$.

The learning rate was set to 1×10^{-3}, and a scheduler was used to reduce its value by a factor of 2, whenever the validation loss stopped improving. The number of epochs could vary, because we used a "scheduler patience" of 10 epochs and implemented early stopping with a patience of 50 epochs. We used the overall Root-Mean-Square Error (RMSE), including the velocity components as well as the pressure, to assess the models' performance on the training and testing sets, as well as to plot predictions to visualize the models' forecasting accuracy. It is important to clarify that, even though the pressure field was also used to calculate the RMSE along with the x and y velocity fields, its value is always about 10 times smaller than theirs. In this sense, we opted to present the RMSE in units of $m \cdot s^{-1}$, which gives a physical meaning to the error without incurring any perceptible deviation.

2.2.1. ConvLSTM

A ConvLSTM is essentially an LSTM where the internal gates are modified to operate over two-dimensional data, which allows it to be used for spatial-temporal prediction. This is achieved by using convolution operations in the state-to-state and input-to-state transitions of the traditional LSTM gates [38].

A multi ConvLSTM layer network was built and tested. The general structure was built by stacking multiple ConvLSTM cells in sequence, to test how the number of hidden dimensions, the kernel size and the number of layers affected performance.

2.2.2. Two-dimensional CNN + LSTM

For this DNN paradigm, 2D CNN (convolution + max pooling) layers were used to create a deep autoencoder network, with an LSTM network added at the end of the network (Figure 1). An additional linear layer was used to map the flattened output of our convolutional encoder to the latent dimension selected for use in the LSTM. In the same way, an additional linear layer was used to map the output of the LSTM back to the appropriate size such that the data could be reshaped back.

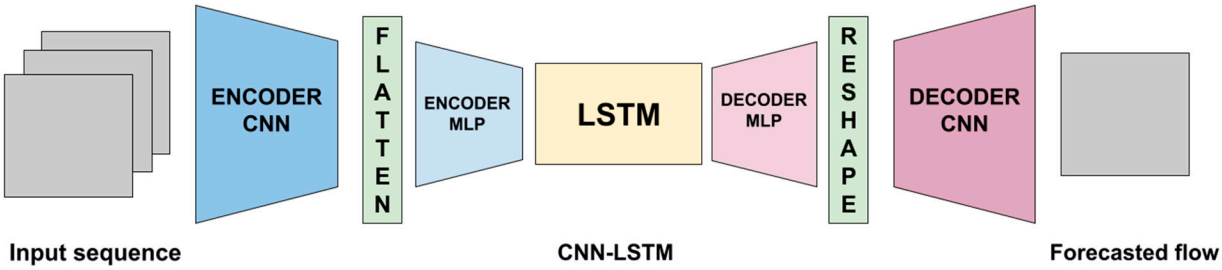


Figure 1. General layout of the tested 2D CNN + LSTM architecture.

Like the other models, we randomly tested the optimal quantity of contiguous CNN layers, as well as the hidden channels and the dimension of the latent space of the DNN, i.e., how much information is retained and processed between the encoder and decoder MLPs (Figure 1). Based on previous tests, only one LSTM layer was added. In [39], the authors applied a similar procedure, successfully testing a similar DNN paradigm for solar irradiance modeling.

2.2.3. Fourier Neural Operator

The Fourier Neural Operator (FNO) has one important reported advantage over classical methods (all CFD as well as ML methods). The FNO models the solution operator, for not only one instance but also multiple instances at a time [27]; this means that its tuning is not restricted to a specific solution case, being more generalizable than other ML approaches. In our work, we implemented this capability by training and testing the Physics-Informed Neural Network (PINN) for different Reynolds numbers.

The structure of the tested models was mounted by stacking contiguous FNO layers sequentially, up to 3. The current research evaluated the number of Fourier modes to keep (modes) in each layer (up to 16, in powers of 2), as well as the number of transforms to apply to these Fourier modes (width, up to 32, in powers of 2). It was expected that the FNO could capture the intrinsic PDE information, because its mapping occurs in the Fourier space (based on frequency modes), not Euclidean (based on spatial coordinates), as usual. This commonly may represent, or be near, the real solution of the PDE.

2.3. Application of Physical Information to the Models

After choosing the best architecture of each NN paradigm, we tested the effects of imposing physical constraints onto the models. This was performed via soft constraints [23], namely adding an error formula to drive the solution to mass conservation through the usual loss function. It was performed by calculating the mass flow imbalance (Equation (1))

for each grid cell and taking the arithmetic mean of their absolute values for all of the computational domain. It should be further noted that the values at the boundaries are already in the datasets, which also gives physics information to the models. The aim of the mass conservation error evaluation is to insert physical coherence into the models, improving interpretability and accuracy.

$$Mass\ imbalance = Mass\ flux_{out} - Mass\ flux_{in} \quad (1)$$

where the overall mass flux can be defined as

$$Mass\ flux_{out} - Mass\ flux_{in} = \oint_{boundaries} \rho \vec{V} \cdot d\vec{A} \quad (2)$$

Because Cartesian coordinates are being used, it is straightforward to solve the above integral only using the velocity components and the side lengths of each cell.

The governing equations, in a non-dimensional form, for the 2D lid-driven problem are [31,34]

$$(\nabla \cdot \boldsymbol{u}) = 0 \quad (3)$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\nabla \mathrm{P} + \frac{1}{Re}\left(\nabla^2 \boldsymbol{u}\right) \quad (4)$$

where $\boldsymbol{u} = (u, v)$ are the velocities for the 2D case, and P and t are pressure and time, respectively. Re represents the Reynolds number, defined as a ratio between the product of velocity and lid length over the kinematic viscosity. For the boundary conditions, the velocities were set to zero over the lateral walls and to finite non-zero over the upper wall. For the domain walls, both epsilon and P were set as having zero gradient. Last, k was set to zero over the domain's walls.

At this stage, several weights were tested for the calculated errors of mass conservation, trying to optimize the PINNs results. The tested values were 0.0, 0.1, 1.0, 2.0 and 5.0.

2.4. Number of Timesteps (Moving Time Window)

Aiming to quantify generalizability to assess how performance degrades with the degree of extrapolation, under unseen initial conditions [23], we tested the influence of the training time window size on the models' performance, following a similar approach that is usually performed in CFD. Once the improved PINNs were developed at this stage, we took the best two models, namely ConvLSTM and FNO, to assess their performance under several time window configurations. This was performed first by forecasting 10 timesteps in advance, each one separately, i.e., we ran the model to forecast the next timestep, and then we use this forecasted field to predict the next one, and so on. In this stage, we tested input timestep windows with sizes of 1, 3, 6, 10 and 15, aiming to mimic different orders of time discretization, a standard approach in CFD. It is worth mentioning that the implementation of high-order time discretization (e.g., from 3 and above) is usually a complex task in CFD, whereas in DNN, it is almost straightforward, being just a matter of training set selection.

After this stage, we tested another approach, where we took the best input time window size and forecasted different size outputs, this time being the prediction directly obtained by the models in a single step. The tested output sizes were 1, 3 and 10, and they were applied to directly predict 10 timesteps in advance. It is worth mentioning that, based on the time resolution (0.1 s) and the lid velocity for the testing set (1.0 $\mathrm{m \cdot s^{-1}}$), 10 timesteps correspond to a complete pass of the lid by the solution domain.

3. Results

3.1. Numerical Data Validation

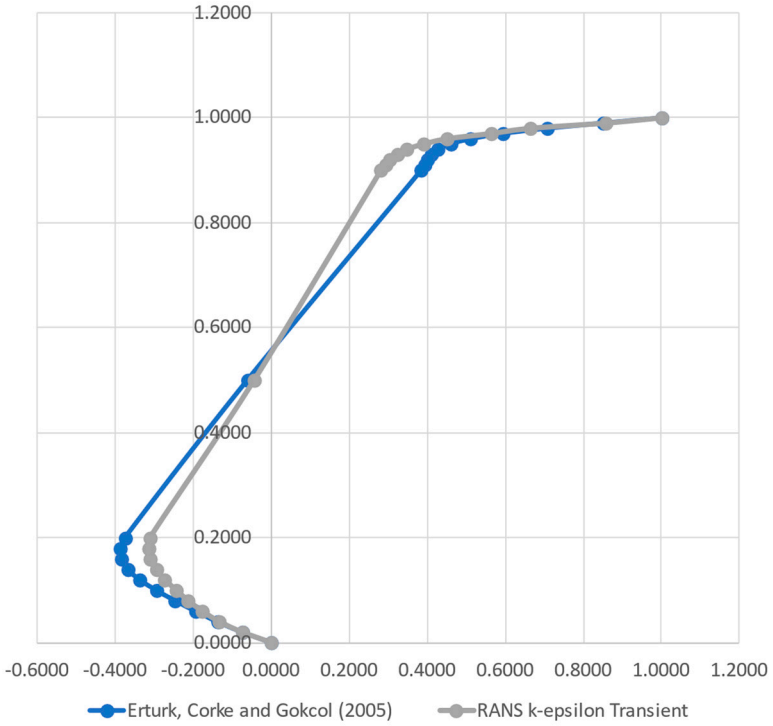
The velocity profiles calculated by the models, as well as the reference ones found in [37], are presented in Figures 2 and 3. Figure 2 depicts the profiles of the x component of velocity over a vertical, center line, for the Reynolds numbers of 1000 (a), 10,000 (b) and

20,000 (c), and Figure 3 shows the equivalent plots of the y component of velocity over a horizontal, center line. As can be seen, our results replicate the reference model reasonably well. The distance between our results and the reference model slightly increases with the Reynolds number, but the flow structure remains the same, as can be noticed by the curvatures, especially near the cavity walls.

3.2. Models' Architecture Setup

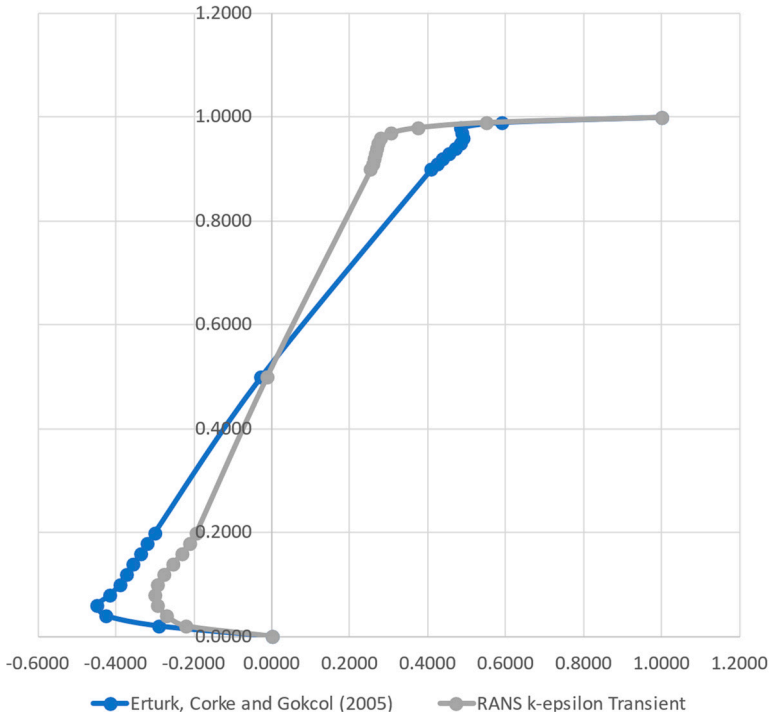
After parameter testing and optimization, two models outperformed similarly, namely ConvLSTM and FNO. The flow predictions for the best architectures of each model are presented in Figures 4 and 5. The figures shown are the specific case for the prediction of t = 0.3 s, given an input of t = 0.0–0.2 s, even though all the tests were implemented for the time ranging from 0 to 10 s, where a steady state was achieved. We chose this timestep because it represents the worst results of all tested models. The CFD simulated data are on the top row, where the first three were used as inputs to the model and the fourth was the ground truth. The predicted output lies directly below, to facilitate the comparisons.

According to the results of Table 1, the model that performed the worst was CNN-LSTM. It could be noticed that, although it could predict the increase in the main vortex, the model was not capable of rotating it. This did not happen to the other models in the same intensity. On the other hand, the remaining models performed better and with the same error magnitude, with an improved performance of the FNO, which achieved its best results with only one layer, which had a direct impact on the computational load. This can be confirmed by the overall RMSE results for the flow velocity, presented in Table 1. Based on these results, we opted to continue with the tests using only the ConvLSTM and FNO models.

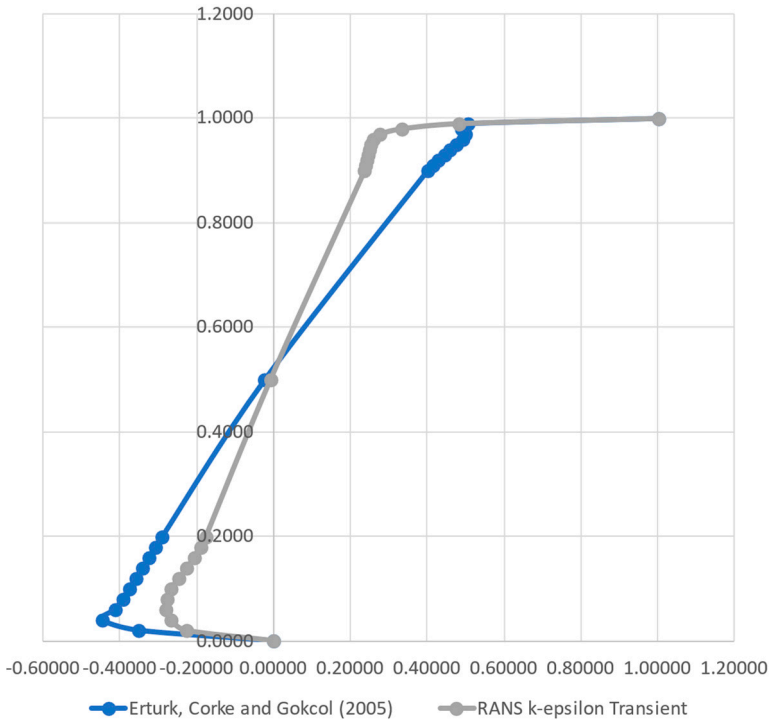


(**a**)

Figure 2. *Cont.*

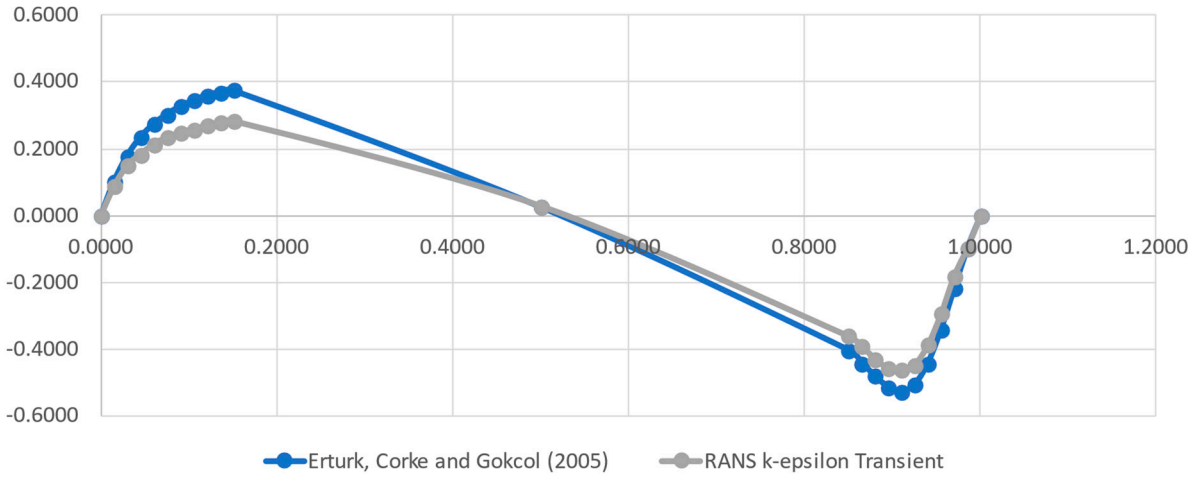


(**b**)

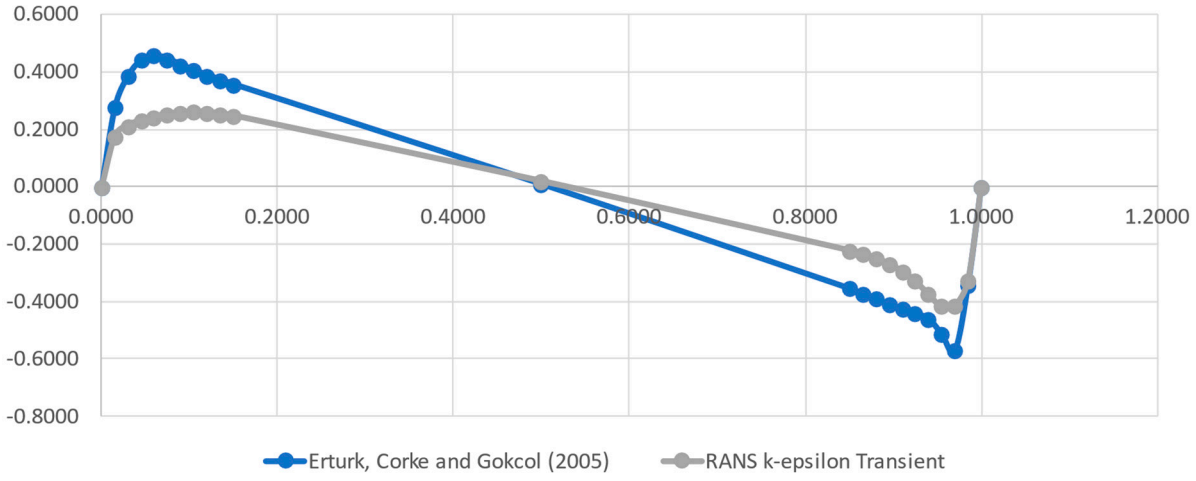


(**c**)

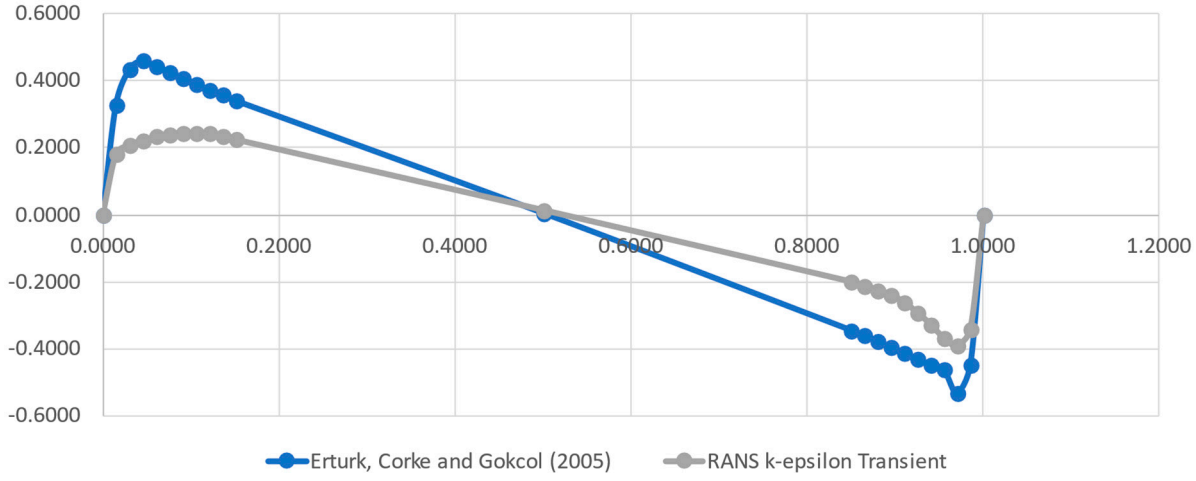
Figure 2. Comparison of the horizontal velocity component over a vertical line obtained by the CFD model (RANS k-ε) and the reference values from [37] for the Reynolds numbers of 1000 (**a**), 10,000 (**b**) and 20,000 (**c**).



(**a**)



(**b**)



(**c**)

Figure 3. Comparison of the vertical velocity component over a horizontal line obtained by the CFD model and the reference values from [37] for the Reynolds numbers of 1000 (**a**), 10,000 (**b**) and 20,000 (**c**).

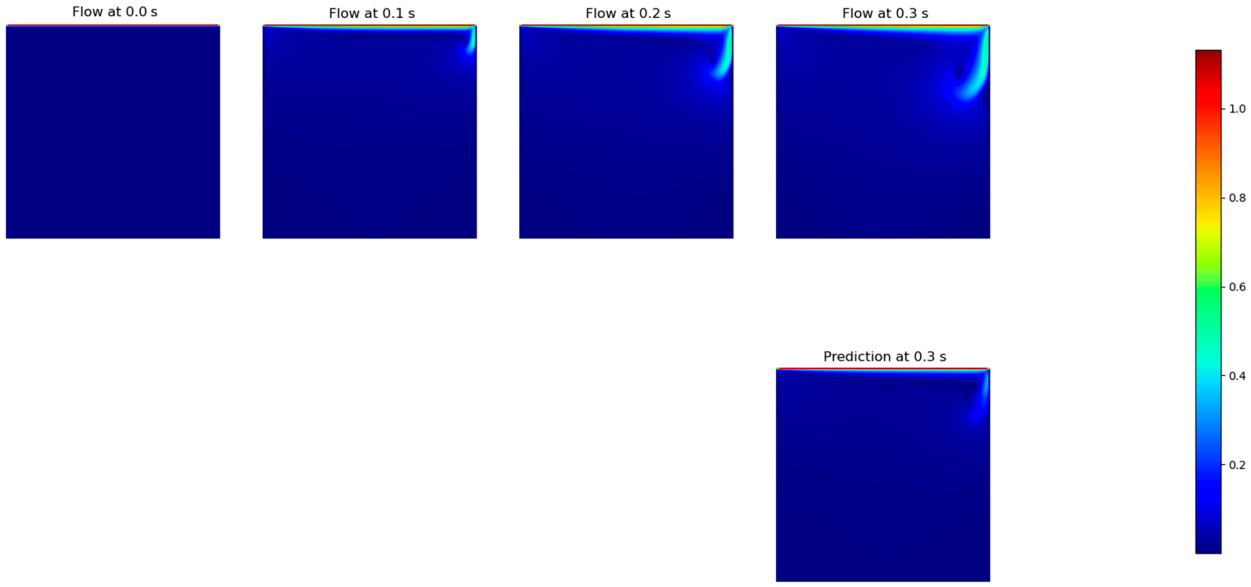


Figure 4. Flow velocities obtained by the ConvLSTM model after tuning, for t = 0.3 s.

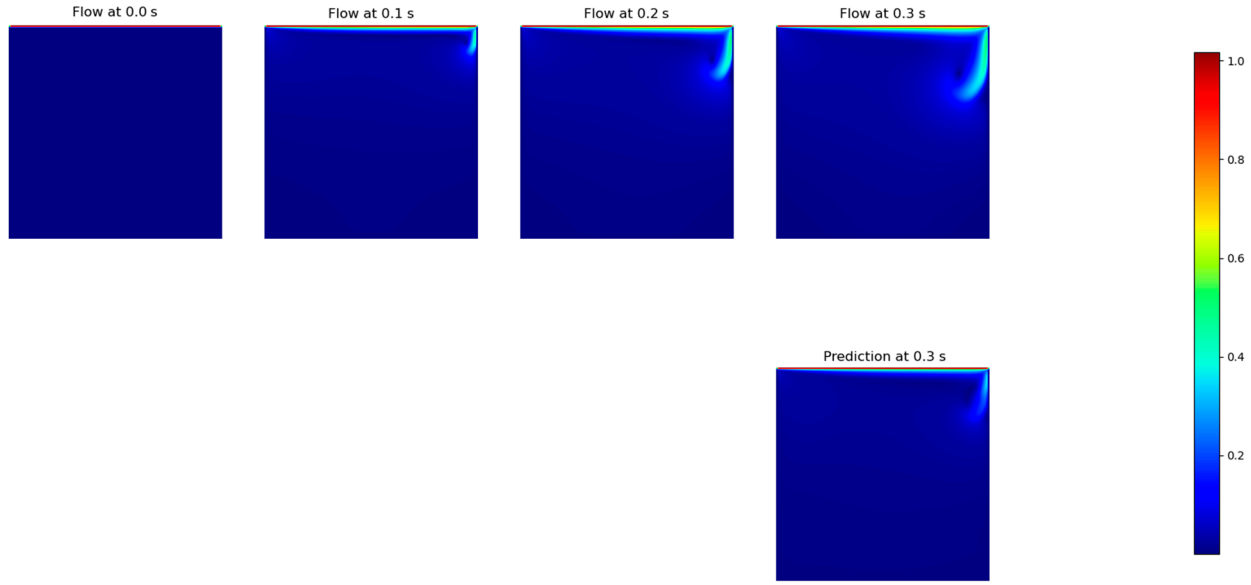


Figure 5. Flow velocities obtained by the FNO model after tuning, for t = 0.3 s.

Table 1. RMSE obtained by all optimized model architectures after 50 epochs.

Model	CNN-LSTM	ConvLSTM	FNO
RMSE (m·s^{-1})	2.26×10^{-2}	7.36×10^{-3}	6.83×10^{-3}

3.3. Application of Physical Error Metrics to the Models

After the addition of a mass imbalance error term in the model tuning, all models presented a validation RMSE that was essentially unchanged (Table 2), with the exception of the CNN-LSTM model, which increased its RMSE. Nevertheless, the addition of mass conservation was able to stabilize the errors through the epochs (Figure 6), specifically for the FNO model. For the case without mass conservation information, the mass balance achieved a minimum testing RMSE and then started to increase monotonically, which does not happen in the case with mass conservation.

Table 2. Impact of mass conservation weights on validation RMSE obtained by all optimized model architectures after a maximum of 300 epochs.

	No MC Error	0.1 × MC Error	1.0 × MC Error	2.0 × MC Error	5.0 × MC Error
FNO	6.56×10^{-3}	6.43×10^{-3}	6.46×10^{-3}	6.49×10^{-3}	6.48×10^{-3}
CNN-LSTM	1.82×10^{-2}	1.91×10^{-2}	2.57×10^{-2}	2.58×10^{-2}	8.09×10^{-2}
ConvLSTM	7.06×10^{-3}	7.03×10^{-3}	7.21×10^{-3}	7.31×10^{-3}	7.31×10^{-3}

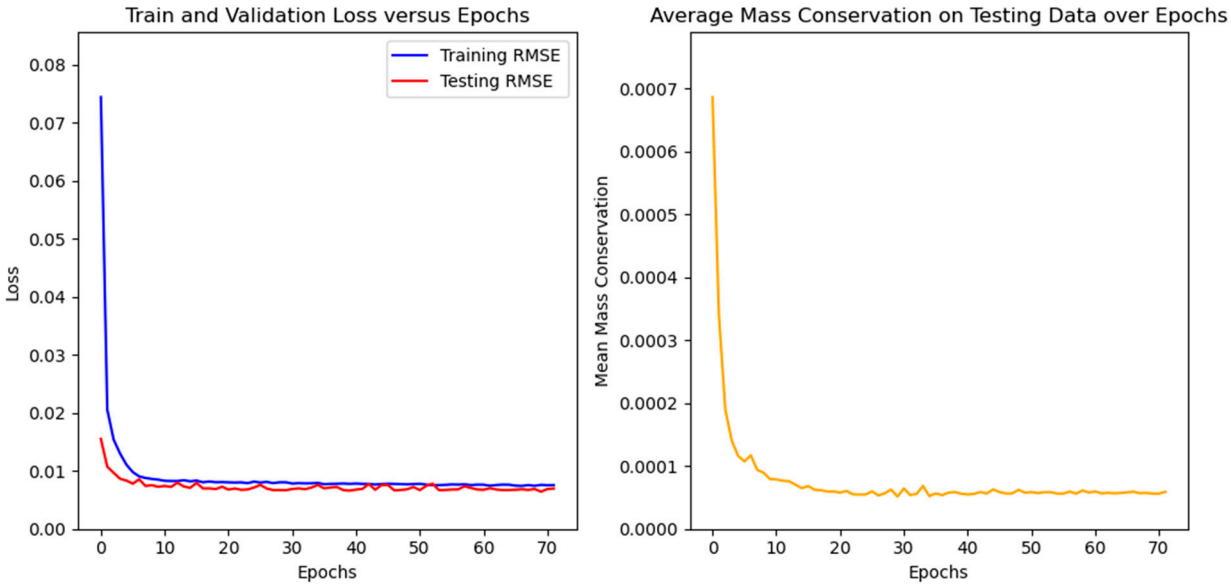


Figure 6. Loss function and mass imbalance error over the training epochs of the FNO model. Testing RMSE is also presented (red line, left plot).

The addition of the mass imbalance error also improved the overall mass conservation, and the impact of the weight on it is shown in Table 3. As can be seen, the presence of the error term itself already worked to diminish the mass imbalance, and an increase in the weight did not imply a further improvement, except for the CNN-LSTM model, which was less sensible to the weight and needed a higher value.

Table 3. Impact of mass conservation weights on mean cell mass imbalance ($kg \cdot s^{-1} \cdot m^2$) obtained by all optimized model architectures after a maximum of 300 epochs.

	No MC Error	0.1 × MC Error	1.0 × MC Error	2.0 × MC Error	5.0 × MC Error
FNO	1.24×10^{-4}	5.90×10^{-5}	4.11×10^{-5}	4.07×10^{-5}	3.96×10^{-5}
CNN-LSTM	2.38×10^{-4}	1.12×10^{-3}	2.67×10^{-4}	2.47×10^{-4}	7.56×10^{-5}
ConvLSTM	7.62×10^{-5}	5.49×10^{-5}	5.54×10^{-5}	4.06×10^{-5}	3.70×10^{-5}

To further illustrate the beneficial effect of the addition of a mass conservation error term in the model training, Figure 7 depicts the specific case of the mass imbalance (Equation (1)) field for the prediction of t = 1.3 s. When we refer to the mass imbalance field, we actually present the absolute calculated mass imbalance for each grid cell, which should always approximate to zero. This constraint was added to the model training. On the left plots, one can see the ground truth (CFD results), and on the right, the predicted fields are shown, without adding the mass conservation error to the models (top), and with its application to the solution (bottom). The improvement was remarkable, being the predicted field with the mass conservation constraint essentially equal to the ground truth.

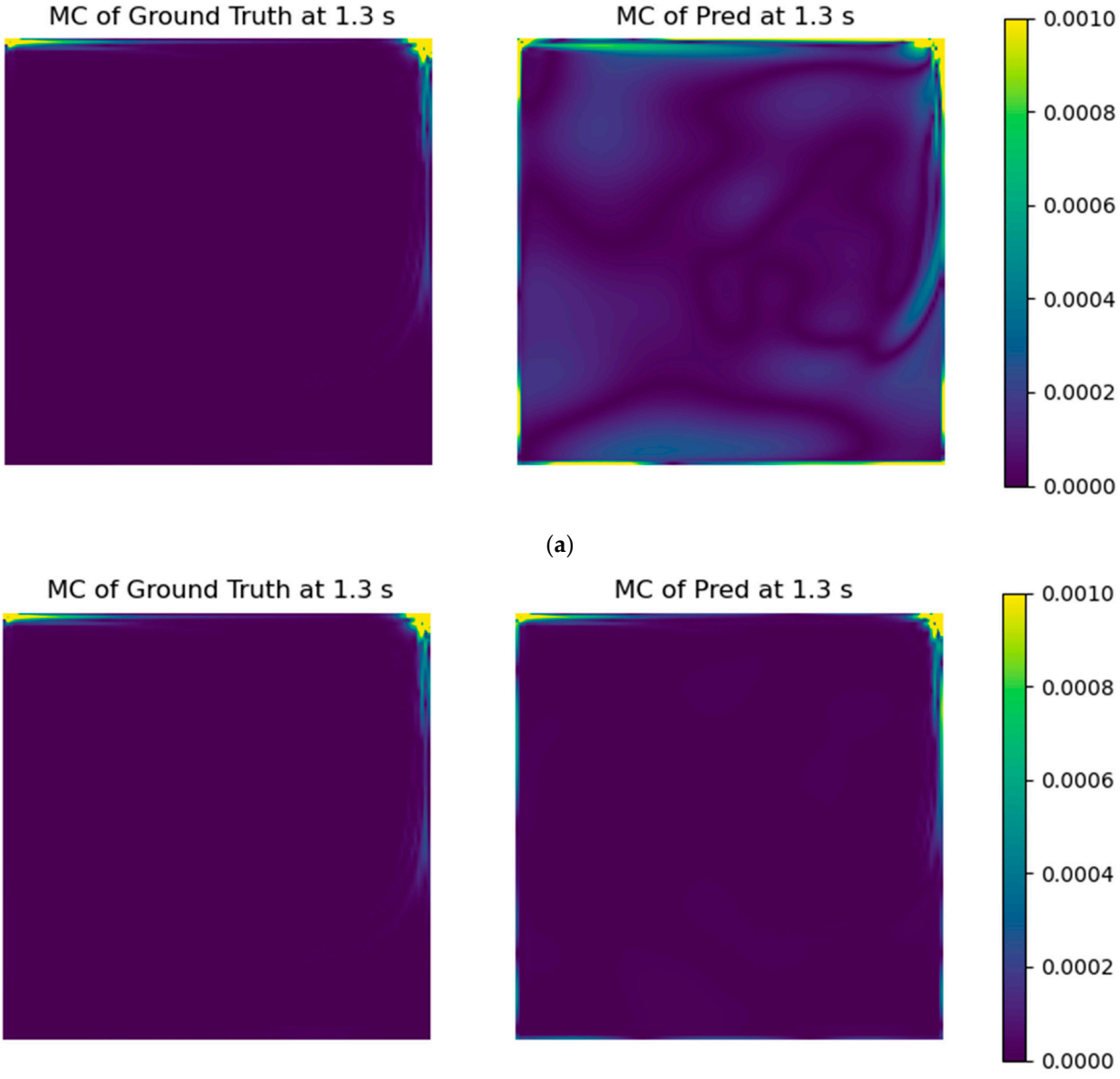


Figure 7. Mass imbalance ($kg \cdot s^{-1} \cdot m^2$) fields without (**a**) and with (**b**) the mass conservation error. The plots on the left are the ground truth fields, and the ones on the right were obtained using the DNN FNO model.

3.4. Accuracy Assessment

3.4.1. Input Time Window Size

After analyzing the results of the previous step, we decided to maintain the weight of the mass conservation error as 0.1 to keep for the tests with the FNO model, once it consistently presented the best performance over all the tested models. This likely happened because the FNO is able to learn the parameters of the equations that underlie the physics of the problem, because the data are transformed to a representation on the solution domain, and other DNN models try to obtain the direct link between the data and the results of the flow. In this sense, the model obtains the ability to learn the physical conservation law, which is supposed to give the general solution of the flow for each variable. Therefore, to keep improving the model, we tested five different input time windows sizes (1, 3, 6, 10 and 15 timesteps) to predict 1 output timestep (Table 4), also "un-rolling" 10 timesteps (Table 5), using the previously forecasted fields when applicable.

Even though the RMSE did not change significantly, it is possible to state that the inclusion of previous timesteps in the training dataset is not able to improve the results, which differs from usual CFD results, at least for the case under study. Figure 8 presents the prediction for the specific time t = 2.3 s for the case of three input timesteps (3:1).

Table 4. FNO errors in predicting one timestep in advance, for several input time window sizes, after a maximum of 300 epochs.

	1:1	**3:1**	**6:1**	**10:1**	**15:1**
RMSE ($m \cdot s^{-1}$)	6.07×10^{-3}	6.43×10^{-3}	7.15×10^{-3}	6.81×10^{-3}	6.70×10^{-3}
Mass imbalance ($kg \cdot s^{-1} \cdot m^2$)	5.71×10^{-5}	5.89×10^{-5}	6.10×10^{-5}	5.14×10^{-5}	5.27×10^{-5}

Table 5. FNO errors for predicting ten timesteps in advance, for several input time window sizes, after a maximum of 300 epochs.

	1:10	**3:10**	**6:10**	**10:10**	**15:10**
RMSE ($m \cdot s^{-1}$)	1.60×10^{-2}	1.07×10^{-2}	9.28×10^{-3}	8.43×10^{-3}	6.59×10^{-3}
Mass imbalance ($kg \cdot s^{-1} \cdot m^2$)	1.99×10^{-4}	1.52×10^{-4}	1.12×10^{-4}	6.68×10^{-5}	6.71×10^{-5}

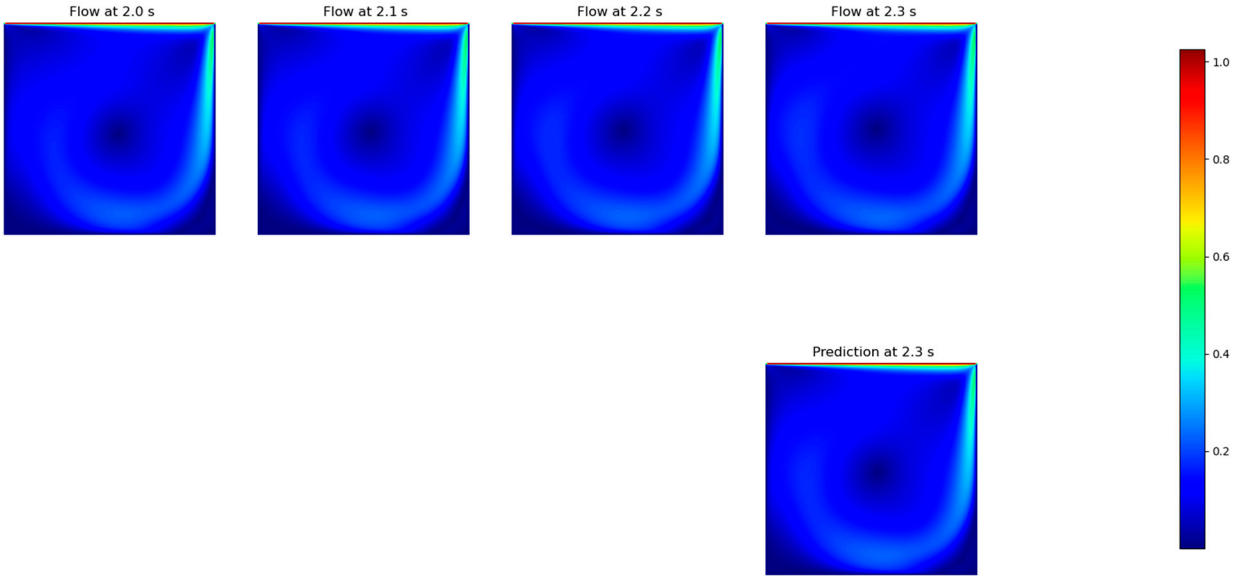


Figure 8. FNO prediction of the resulting speed for the 3 timesteps input. The ground truth plots are above, and the forecasted field is below.

3.4.2. Output Time Window Size

Because the results of the previous subsection were obtained by "un-rolling" the solution over the predicted fields, we also tested the model to directly forecast a predefined amount of timesteps, composing a prediction of 10 timesteps at the end. The input time window was kept at size three because, even though it was not the best result (see Table 4), we opted to maintain more temporal information. The results are presented in Table 6. To give insight into the results, Figure 9 shows the predictions for the specific timesteps with a range of 1.5–1.7 s, for the case of three input and three output timesteps.

Table 6. FNO testing errors for predicting several timesteps in advance, for three input time window sizes, after a maximum of 300 epochs.

	3:1	**3:3**	**3:10**
RMSE ($m \cdot s^{-1}$)	1.07×10^{-2}	8.79×10^{-3}	1.07×10^{-2}
Mass imbalance ($kg \cdot s^{-1} \cdot m^2$)	1.52×10^{-4}	7.24×10^{-5}	1.41×10^{-4}

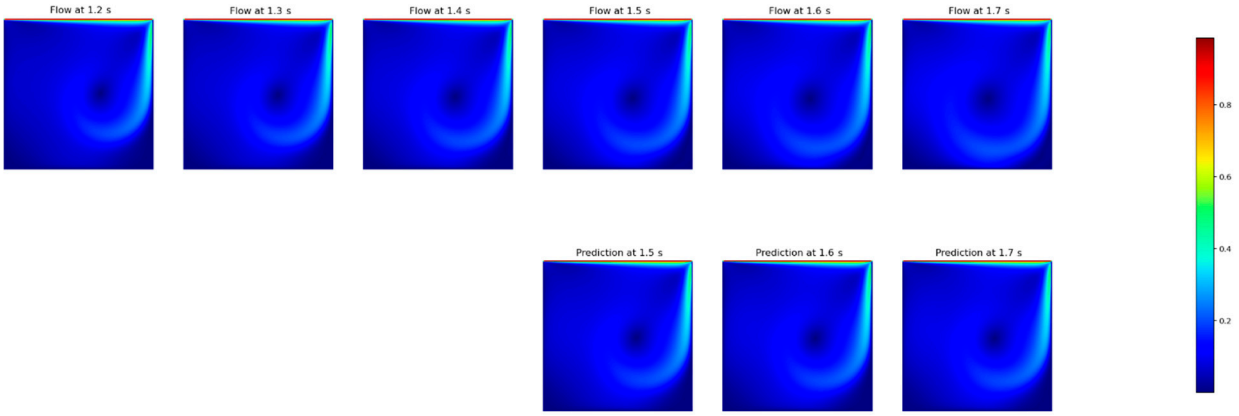


Figure 9. FNO prediction of the resulting speed for the 3 timesteps input and 3 timesteps output. The ground truth plots are above, and the forecasted fields are below.

4. Discussion

Research efforts have been carried out in order to model turbulent flows. The approach of using classical problems that, in a simple way, manage to represent complex phenomena is often used [40]. In this context, cavity flow is commonly applied under various conditions and for different purposes. In [41], the authors proposed the combined solution of CFD with Back Propagation for modeling a cavity with a backward-facing step, and as can be seen in our work, the technique is promising and deserves investigation.

In the present work, we focused on comparing classic DNN architectures with the recent FNO approach. In general, the DNN approach tends to dissipate flow energy, which can be evidenced by a reduction in eddy viscosity [42], as well as a reduction in vorticity, which can be shown, as we present here, in the difficulty of the models to maintain the rotation of the flow. In this regard, we were able to indicate that the investigated FNO-based DNN architecture did not show improvements in the flow prediction capability for timesteps greater than three, as presented in Table 6.

Regarding the performance of the developed architecture, it managed to reach levels equivalent or even superior to those presented in the literature and required reduced training time due to its simpler implementation. Our MSE of 4.13×10^{-5} is in the same order of magnitude as that of [43], where the authors obtained their values for Burgers' equation, the Sine-Gordon equation, the Allen–Cahn equation and Kovasznay flow. For the cavity, our results are much better than the best result reported by them, MSE = 8.9456×10^{-2}.

It is common in works involving DNNs to use different performance evaluation metrics, which makes direct comparison between approaches difficult. In [28], the authors used the L2 relative error to evaluate several DNN architectures, specifically based on DeepONet and FNO. Despite this, as we also point out here, the approaches based on the FNO stood out for their superior performance.

Given the above, the feasibility of the technique is strongly indicated, as also found in the recent literature [14], where there was no quantitative presentation of the error, but which qualitatively showed the robustness of the Physics-Informed Neural Network solution. This viability is also already shown with realistic problems, such as the cavity flow studied here. The authors of [24] presented MAE in the order of 1×10^{-3}, which is at the same level of our reached RMSE. It is noteworthy that the RMSE metric always tends

to be greater than the MAE for the same problem, which indicates that the FNO is fully qualified for the development of research of Physics-Informed Neural Networks.

5. Conclusions

This paper presents a novel approach to solve the Navier–Stokes equations using Physics-Informed Neural Networks. Three DNN paradigms (Fourier Neural Network, Convolutional LSTM and CNN-LSTM) were implemented and tested against the CFD solution of the cavity lid-driven flow problem. We used several Reynolds numbers to test and forecast the Re = 10,000 case, aiming to benchmark the Fourier Neural Network (FNO). The advantage of this new class of DNN architecture is the mapping of fields between different spaces (not necessarily Euclidean), which allows the learning of the parametric dependence of the solutions directly to an entire family of PDEs. From the results, the following conclusions can be highlighted:

- A RANS k-ε CFD solution was performed to generate data (training and testing) to be fed to the models. A comparison with the results found in the literature was able to attest the data quality. The evaluation of the k-ε turbulence model against a full Direct Numerical Simulation indicated that the simpler CFD model, for this simple case, accurately represented the turbulence phenomena.
- After the tests for the models' architectures setup, the FNO and ConvLSTM paradigms performed better, with a consistent small advantage of FNO.
- With the selected models' architectures, a custom error parcel regarding the mass conservation error was added to the training step, using several weight values. Even though the RMSE of the test case did not improve, the resultant fields presented a notable improvement in physical coherence.
- The FNO paradigm was finally assessed to predict the solutions of the flow under several input/output situations, giving a testing RMSE of 0.008792 m/s for the best configuration (three timesteps for the input and three timesteps for the output), which was at least two orders of magnitude of the reference lid velocity (1.0 m/s).

For the tested case, the FNO was able to consistently perform better than the other models, which qualifies it as an option to further develop DNN solutions of partial differential equations. Although presenting good performance, the FNO has some limitations, as presented by Lu et al. [28], and DeepONet is an option to address such hindrances. For further work, implementation of the momentum conservation equation to the model could be performed to assess its contribution over the model's results, as well the effect of different image resolutions in the final model's prediction. Moreover, a deeper understanding of FNO limitations can be assessed by testing other problem configurations and the effect of turbulence levels, not only for RANS models, but also for Large-eddy Simulations or Direct Numerical Simulation (DNS) results. For instance, other flow geometries can be tested, which would produce more transient solutions of the PDEs, such as a bluff body or wake flows. Moreover, greater Reynolds numbers can be considered, which may be indicative of environmental flows.

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

Funding: This research was funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) Alliance, Grant No. 401643, in association with Lakes Environmental Software Inc., and by the Conselho Nacional de Desenvolvimento Científico e Tecnológico—Brasil (CNPq), Grant No. 305456/2019-9.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The code and the data used for this work can be found, respectively, at https://drive.google.com/drive/folders/13e5lnKy_UVtLNIytdJzwV7rQjC-PiQ-w?usp=sharing (code) and https://drive.google.com/drive/folders/1v-ORCjsH1v9GAUsTK_lZ3rDPFZYS7Pjp?usp=sharing (data), accessed on 25 January 2023.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nguyen, N.T.; Bochnia, D.; Kiehnscherf, R.; Dötzel, W. Investigation of Forced Convection in Microfluid Systems. *Sens. Actuators A Phys.* **1996**, *55*, 49–55. [CrossRef]
2. Wang, Z.; Li, B.; Luo, Q.-Q.; Zhao, W. Effect of Wall Roughness by the Bionic Structure of Dragonfly Wing on Microfluid Flow and Heat Transfer Characteristics. *Int. J. Heat Mass Transf.* **2021**, *173*, 121201. [CrossRef]
3. Castorrini, A.; Gentile, S.; Geraldi, E.; Bonfiglioli, A. Increasing Spatial Resolution of Wind Resource Prediction Using NWP and RANS Simulation. *J. Wind Eng. Ind. Aerodyn.* **2021**, *210*, 104499. [CrossRef]
4. Machniewski, P.; Molga, E. CFD Analysis of Large-Scale Hydrogen Detonation and Blast Wave Overpressure in Partially Confined Spaces. *Process Saf. Environ. Prot.* **2022**, *158*, 537–546. [CrossRef]
5. Li, R.; Wang, Y.; Lin, H.; Du, H.; Wang, C.; Chen, X.; Huang, M. A Mesoscale CFD Simulation Study of Basic Wind Pressure in Complex Terrain—A Case Study of Taizhou City. *Appl. Sci.* **2022**, *12*, 10481. [CrossRef]
6. Carneiro, F.O.M.; Moura, L.F.M.; Costa Rocha, P.A.; Pontes Lima, R.J.; Ismail, K.A.R. Application and Analysis of the Moving Mesh Algorithm AMI in a Small Scale HAWT: Validation with Field Test's Results against the Frozen Rotor Approach. *Energy* **2019**, *171*, 819–829. [CrossRef]
7. Menter, F.; Hüppe, A.; Matyushenko, A.; Kolmogorov, D. An Overview of Hybrid RANS–LES Models Developed for Industrial CFD. *Appl. Sci.* **2021**, *11*, 2459. [CrossRef]
8. Waschkowski, F.; Zhao, Y.; Sandberg, R.; Klewicki, J. Multi-Objective CFD-Driven Development of Coupled Turbulence Closure Models. *J. Comput. Phys.* **2022**, *452*, 110922. [CrossRef]
9. Kumar, G.; Mondal, P.K. Application of Artificial Neural Network for Understanding Multi-Layer Microscale Transport Comprising of Alternate Newtonian and Non-Newtonian Fluids. *Colloids Surf. A Physicochem. Eng. Asp.* **2022**, *642*, 128664. [CrossRef]
10. Ismayeel, M.; Mehta, S.K.; Mondal, P.K. Prediction of Electrodiffusio-Osmotic Transport of Shear-Thinning Fluids in a Nanochannel Using Artificial Neural Network. *Phys. Fluids* **2023**, *35*, 012018. [CrossRef]
11. Kumar, G.; Saha, S.P.; Ghosh, S.; Mondal, P.K. Artificial Neural Network-Based Modelling of Optimized Experimental Study of Xylanase Production by Penicillium Citrinum Xym2. *Proc. Inst. Mech. Eng. Part E: J. Process Mech. Eng.* **2022**, *236*, 1340–1348. [CrossRef]
12. Kumar, G.; Thampi, G.; Mondal, P.K. Predicting Performance of Briquette Made from Millet Bran: A Neural Network Approach. *Adv. J. Grad. Res.* **2020**, *9*, 1–13. [CrossRef]
13. Zhao, Y.; Akolekar, H.D.; Weatheritt, J.; Michelassi, V.; Sandberg, R.D. RANS Turbulence Model Development Using CFD-Driven Machine Learning. *J. Comput. Phys.* **2020**, *411*, 109413. [CrossRef]
14. Guo, Y.; Cao, X.; Liu, B.; Gao, M. Solving Partial Differential Equations Using Deep Learning and Physical Constraints. *Appl. Sci.* **2020**, *10*, 5917. [CrossRef]
15. Iskhakov, A.S.; Dinh, N.T.; Chen, E. Integration of Neural Networks with Numerical Solution of PDEs for Closure Models Development. *Phys. Lett. A* **2021**, *406*, 127456. [CrossRef]
16. Bhatnagar, S.; Afshar, Y.; Pan, S.; Duraisamy, K.; Kaushik, S. Prediction of Aerodynamic Flow Fields Using Convolutional Neural Networks. *Comput Mech* **2019**, *64*, 525–545. [CrossRef]
17. Ringstad, K.E.; Banasiak, K.; Ervik, Å.; Hafner, A. Machine Learning and CFD for Mapping and Optimization of CO_2 Ejectors. *Appl. Therm. Eng.* **2021**, *199*, 117604. [CrossRef]
18. Mohammadpour, J.; Husain, S.; Salehi, F.; Lee, A. Machine Learning Regression-CFD Models for the Nanofluid Heat Transfer of a Microchannel Heat Sink with Double Synthetic Jets. *Int. Commun. Heat Mass Transf.* **2022**, *130*, 105808. [CrossRef]
19. Wang, R.; Kuru, E.; Yan, Y.; Yang, X.; Yan, X. Sensitivity Analysis of Factors Controlling the Cement Hot Spot Temperature and the Corresponding Well Depth Using a Combined CFD Simulation and Machine Learning Approach. *J. Pet. Sci. Eng.* **2022**, *208*, 109617. [CrossRef]
20. Sirignano, J.; MacArt, J.F.; Freund, J.B. DPM: A Deep Learning PDE Augmentation Method with Application to Large-Eddy Simulation. *J. Comput. Phys.* **2020**, *423*, 109811. [CrossRef]
21. Xu, X.; Waschkowski, F.; Ooi, A.S.H.; Sandberg, R.D. Towards Robust and Accurate Reynolds-Averaged Closures for Natural Convection via Multi-Objective CFD-Driven Machine Learning. *Int. J. Heat Mass Transf.* **2022**, *187*, 122557. [CrossRef]
22. Kochkov, D.; Smith, J.A.; Alieva, A.; Wang, Q.; Brenner, M.P.; Hoyer, S. Machine Learning–Accelerated Computational Fluid Dynamics. *Proc. Natl. Acad. Sci. USA* **2021**, *118*, e2101784118. [CrossRef]
23. Kashinath, K.; Mustafa, M.; Albert, A.; Wu, J.-L.; Jiang, C.; Esmaeilzadeh, S.; Azizzadenesheli, K.; Wang, R.; Chattopadhyay, A.; Singh, A.; et al. Physics-Informed Machine Learning: Case Studies for Weather and Climate Modelling. *Phil. Trans. R. Soc. A.* **2021**, *379*, 20200093. [CrossRef] [PubMed]

24. Wu, H.; Niu, S.; Zhang, Y.; Fu, W. Physics-Informed Generative Adversarial Network-Based Modeling and Simulation of Linear Electric Machines. *Appl. Sci.* **2022**, *12*, 10426. [CrossRef]
25. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
26. Jiang, C.M.; Esmaeilzadeh, S.; Azizzadenesheli, K.; Kashinath, K.; Mustafa, M.; Tchelepi, H.A.; Marcus, P.; Prabhat, M.; Anandkumar, A. MESHFREEFLOWNET: A Physics-Constrained Deep Continuous Space-Time Super-Resolution Framework. In Proceedings of the SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, Atlanta, GA, USA, 9–19 November 2020; pp. 1–15.
27. Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; Anandkumar, A. Fourier Neural Operator for Parametric Partial Differential Equations. In Proceedings of the International Conference on Learning (ICLR), Vienna, Austria, 3–7 May 2021; pp. 1–16.
28. Lu, L.; Meng, X.; Cai, S.; Mao, Z.; Goswami, S.; Zhang, Z.; Karniadakis, G.E. A Comprehensive and Fair Comparison of Two Neural Operators (with Practical Extensions) Based on FAIR Data. *Comput. Methods Appl. Mech. Eng.* **2022**, *393*, 114778. [CrossRef]
29. Carlberg, K.T.; Jameson, A.; Kochenderfer, M.J.; Morton, J.; Peng, L.; Witherden, F.D. Recovering Missing CFD Data for High-Order Discretizations Using Deep Neural Networks and Dynamics Learning. *J. Comput. Phys.* **2019**, *395*, 105–124. [CrossRef]
30. Hanna, B.N.; Dinh, N.T.; Youngblood, R.W.; Bolotnov, I.A. Machine-Learning Based Error Prediction Approach for Coarse-Grid Computational Fluid Dynamics (CG-CFD). *Prog. Nucl. Energy* **2020**, *118*, 103140. [CrossRef]
31. Feldman, Y.; Gelfgat, A.Y. From Multi- to Single-Grid CFD on Massively Parallel Computers: Numerical Experiments on Lid-Driven Flow in a Cube Using Pressure–Velocity Coupled Formulation. *Comput. Fluids* **2011**, *46*, 218–223. [CrossRef]
32. Rajakumar, S.B.; Perumal, D.A.; Yadav, A.K. Computation of Fluid Flow in Double Sided Cross-Shaped Lid-Driven Cavities Using Lattice Boltzmann Method. *Eur. J. Mech.-B/Fluids* **2018**, *70*, 46–72. [CrossRef]
33. Bayareh, M.; Nourbakhsh, A.; Rouzbahani, F.; Jouzaei, V. Explicit Incompressible SPH Algorithm for Modelling Channel and Lid-Driven Flows. *SN Appl. Sci.* **2019**, *1*, 1040. [CrossRef]
34. Filali, A.; Khezzar, L.; Semmari, H.; Matar, O. Application of Artificial Neural Network for Mixed Convection in a Square Lid-Driven Cavity with Double Vertical or Horizontal Oriented Rectangular Blocks. *Int. Commun. Heat Mass Transf.* **2021**, *129*, 105644. [CrossRef]
35. Jasak, H. OpenFOAM: Open Source CFD in Research and Industry. *Int. J. Nav. Archit. Ocean Eng.* **2009**, *1*, 89–94. [CrossRef]
36. OpenFOAM. Available online: https://www.openfoam.com/ (accessed on 13 January 2023).
37. Erturk, E.; Corke, T.C.; Gökçöl, C. Numerical Solutions of 2-D Steady Incompressible Driven Cavity Flow at High Reynolds Numbers. *Int. J. Numer. Methods Fluids* **2005**, *48*, 747–774. [CrossRef]
38. Shi, X.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.; Woo, W. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In Proceedings of the 29th International Conference on Neural Information Processing Systems (NIPS), Montreal, QC, Canada, 7–12 December 2015; pp. 1–9.
39. Rocha, P.A.C.; Santos, V.O. Global Horizontal and Direct Normal Solar Irradiance Modeling by the Machine Learning Methods XGBoost and Deep Neural Networks with CNN-LSTM Layers: A Case Study Using the GOES-16 Satellite Imagery. *Int J Energy Environ. Eng* **2022**, *13*, 1271–1286. [CrossRef]
40. Larsen, B.E.; Fuhrman, D.R. Simulation of Cross-Shore Breaker Bar Development Utilizing a Stabilized Two-Equation Turbulence Model. *Coast. Eng.* **2023**, *180*, 104269. [CrossRef]
41. He, Z.; Sun, T.; Zou, L.; Jiang, Y.; Duan, L. Ventilated Cavity Flows behind a Backward Facing Step with a Combination Computational Fluid Dynamics and Error Back Propagation Algorithm. *Ocean Eng.* **2022**, *260*, 111741. [CrossRef]
42. Liu, Y.; Hu, R.; Kraus, A.; Balaprakash, P.; Obabko, A. Data-Driven Modeling of Coarse Mesh Turbulence for Reactor Transient Analysis Using Convolutional Recurrent Neural Networks. *Nucl. Eng. Des.* **2022**, *390*, 111716. [CrossRef]
43. Tang, S.; Feng, X.; Wu, W.; Xu, H. Physics-Informed Neural Networks Combined with Polynomial Interpolation to Solve Nonlinear Partial Differential Equations. *Comput. Math. Appl.* **2023**, *132*, 48–62. [CrossRef]