Master Thesis

The Development and Optimization of Surrogate Deep Learning Models to Predict the Response of Very Large Floating Structures

R.R. Rutgers

Delft University of Technology

TU Delft

Master Thesis

The Development and Optimization of Surrogate Deep Learning Models to Predict the Response of Very Large Floating Structures

by

R.R. Rutgers

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Monday February 10, 2025 at 11:00 AM.

Student number:	4957946	
Project duration:	August 1, 2024 – February 10, 2025	
Thesis committee:	Dr. ir. O. Colomés Gené,	TU Delft, Chairman
	Dr. ir. H. Wang,	TU Delft, supervisor
	Dr. ir. S. Agarwal,	TU Delft, supervisor

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.
The majority of the codes used in this thesis is available in the following GitHub repository:
`https://github.com/RafaelRutgers/thesis-deep-learning-models.git`

TU Delft

Preface

This master's thesis will complete my Master's degree in Civil Engineering at the Technical University of Delft, specializing in Hydraulic and Offshore Structures. When I began my bachelor's degree in 2018, I was uncertain about the direction my career and studies would take. Gradually, throughout my undergraduate program, I discovered a strong interest in the dynamic field of offshore engineering. The innovative nature of floating structures particularly captivated me, and I immediately knew I wanted to specialize in this area.

After taking the course Data Science and Artificial Intelligence for Engineers, my areas of interest expanded. I recognized that machine learning holds tremendous potential in engineering and that engineers will increasingly utilize machine learning in the coming decades. I believe it can fundamentally transform the engineering industry, compelling me to integrate it into my professional skill set.

Both of these fields are areas I aspire to work in for the foreseeable future, with the aim of becoming an expert in each. Given the complexity of these domains, I have learned that as long as I remain interested in a topic, I will pursue it diligently until I fully understand its intricacies.

First and foremost, I would like to thank Oriol Colomés for his supervision during my master's thesis and for making time for me despite his busy schedule. I am also grateful for the courses I took under his guidance, especially Floating and Submerged Structures, which reinforced my desire to continue in this field. Additionally, I extend my gratitude to Hongrui Wang for overseeing the machine learning component of my thesis and for providing me with valuable insights and a thoughtful approach to machine learning. Lastly, I would like to thank Shagun Agarwal for providing the dataset for this thesis and for offering valuable feedback throughout the research process.

Enjoy reading!

R.R. Rutgers
Delft, February 2025

Abstract

Offshore floating photovoltaics (OFPVs) show great potential for future renewable energy generation by complementing existing wind energy. To maximize power output, it is essential to accurately predict the response of these structures under wave-induced forces.

Although finite-element fluid-structure interaction (FE-FSI) models can simulate the response of very large floating structures (VLFSs) with high accuracy, they are computationally expensive and time-intensive. To address this challenge, this thesis develops and optimizes three surrogate deep learning models: a Convolutional Neural Network (CNN), a Long Short-Term Memory (LSTM) network, and a hybrid CNN-LSTM model. An iterative trial-and-error approach was employed to fine-tune the hyper-parameters and improve model performance.

The CNN model accurately captured spatial features but struggled with generalization across varying sea states. The LSTM model captured temporal patterns well, but had lower accuracy and longer training times. The hybrid CNN-LSTM model combined the strengths of both approaches, achieving a low weighted absolute percentage error (WAPE) of 0.98%. All models performed well under typical wave conditions but exhibited reduced accuracy in extreme scenarios, particularly for low-amplitude tilts. A sensitivity analysis indicated that these errors were likely caused by insufficient representation of extreme sea states in the training data.

The hybrid CNN-LSTM model was selected as the model that most accurately predicts the response of VLFSs due to its balanced performance, minimal prediction time, and generalization capability across typical operating conditions. While the model shows high reliability for typical sea states, its performance declines for extreme conditions, such as extremely large waves or very small tilts. Despite reduced reliability in extreme conditions, the model provides a promising tool for real-time OFPV response prediction and preliminary studies.

Contents

1 Introduction **1**
- 1.1 Background . . . 1
- 1.2 Irradiance-Tilt Coupling . . . 4
- 1.3 Underlying Physics . . . 6
  - 1.3.1 Linear Wave Theory . . . 6
  - 1.3.2 Fluid-structure interaction . . . 8
- 1.4 Current model . . . 11
- 1.5 Limitations: Introduction to Machine Learning . . . 12
- 1.6 Research Questions . . . 16
- 1.7 Research Scope . . . 16
- 1.8 Research Methodology . . . 16

2 Dataset **19**
- 2.1 Environmental Setup . . . 19
- 2.2 Simulation Data . . . 20
- 2.3 Data Preparation . . . 21

3 Model configuration **26**
- 3.1 Convolutional Neural Network . . . 27
  - 3.1.1 Input parameters . . . 28
  - 3.1.2 Model architecture & Hyperparameters . . . 28
- 3.2 Long Short-Term Memory . . . 31
  - 3.2.1 Input parameters . . . 31
  - 3.2.2 Model architecture & Hyperparameters . . . 32
- 3.3 Convolutional Neural Network - LSTM hybrid . . . 35
  - 3.3.1 Input parameters . . . 35
  - 3.3.2 Model architecture & Hyperparameters . . . 36

4 Model Performance **41**
- 4.1 Convolutional Neural Network Performance . . . 41
  - 4.1.1 Spatial and Temporal Tilt Analysis . . . 41
  - 4.1.2 Evaluation Metrics Assessment . . . 43
- 4.2 Long Short-Term Memory Performance . . . 45
  - 4.2.1 Spatial and Temporal Tilt Analysis . . . 45
  - 4.2.2 Error Metrics Assessment . . . 47
- 4.3 CNN-LSTM Hybrid Performance . . . 49
  - 4.3.1 Spatial and Temporal Tilt Analysis . . . 49
  - 4.3.2 Error Metrics Assessment . . . 51
- 4.4 Model Comparison . . . 53

5 Error Quantification & Reliability **58**

6 Conclusions **62**

7 Discussions **64**

8 Recommendations for Future Work **66**
- 8.1 Expanding Input Cases . . . 66
- 8.2 Generalizing the CNN Model . . . 66
- 8.3 Incorporating Physics-Based Approaches . . . 66
- 8.4 Fatigue and Long-Term Performance . . . 67

References **68**

A Dataset **71**

B First-order Taylor Expansion **73**

C Visualization of the Models **77**

D Results **82**
D.1 CNN Results 82
D.2 LSTM Results 84
D.3 CNN - LSTM Hybrid Results 86

List of Figures

1.1 Robust coefficient of variation for the mean annual (upper panel) and mean seasonal wind power density for the Mediterranean Sea in winter (middle left panel), spring (middle right panel), summer (lower left panel), and autumn (lower right panel) [14]. 2
1.2 Mean annual (upper panel) and mean seasonal solar irradiance for the Mediterranean Sea in winter (middle left panel), spring (middle right panel), summer (lower left panel), and autumn (lower right panel) [14]. 3
1.3 Monthly values of energy output of an 8-MW offshore wind turbine, an 8-MW floating solar farm, and their combination at the Alboran Sea (upper left), the central Aegean Sea (upper middle), the Gulf of Lion (upper right), the Gulf of Sidra (lower left), the Northeastern part of Sicily (lower middle), and the Southern part of Cyprus (lower right) [14]. . . . 4
1.4 Global horizontal (reflected) irradiance (I_{GHI}), direct normal irradiance (I_{DNI}), and diffuse horizontal irradiance (I_{DHI}) [30]. 4
1.5 Ranges of applicability of different wave theories. With H being the wave height, h being the water depth, τ being the wave period, and g the gravitational constant [5]. 6
1.6 The Laplace and linearized Bernoulli equations and boundary conditions for linear wave theory, in terms of the velocity potential ϕ applied to a domain Ω [15]. 7
1.7 The global response of floating structures under fluid forces [33]. 9
1.8 A schematic of the boundary value problem of the fluid domain Ω, bounded by the inlet Γ_{in}, outlet Γ_{out}, sea bed Γ_{sb}, free surface Γ_{fs}, and floating platform Γ_b[1]. 10
1.9 Offshore floating photovoltaic system with 3 floaters and 2 joints [1]. 11
1.10 A sketch of the discretized fluid domain Ω bounded by the inlet Γ_{in}, outlet Γ_{out}, sea bed Γ_{sb}, free surface Γ_{fs}, and floating platform Γ_b used in the FE-FSI model [1]. 12
1.11 Comparison of the overall accuracy of the LSTM model and the 1D-CNN model, capturing the response over left) 2.5 s and right) 5.0 s [29]. 13
1.12 The hybrid CNN-LSTM architecture. The data coming from the sensors are all processed individually. The time-series is divided into multiple time windows, resulting in a feature map per window. The feature maps belonging to the same time window are concatenated and chronologically used in the subsequent RNN [6]. 14
1.13 The hybrid CNN-GRU architecture used to forecast ship motion. Including a 1D CNN layer, Gated Recurring Unit layer, and an attention layer. 15
1.14 Flowchart of the methodology of this research. 17

2.1 Boxplots per season of a) hourly significant wave height values and b) hourly peak period values. 22
2.2 Quartile JONSWAP spectra for each season 23
2.3 5th percentile JONSWAP spectra for each season 24

3.1 Schematic of an OFPV platform showing (a) one floater, (b) multiple floaters interconnected through hinges, and (c) multiple floaters with variable stiffness. 27
3.2 Layer configuration of the CNN model architecture. Each black box corresponds to the name of the layer, with its corresponding input and output shape noted in the white box. Note that the output of one layer serves as the input for the subsequent layer. 30
3.3 An LSTM memory cell, including a forget gate, input gate, and output gate [18]. 33
3.4 Layer configuration of the LSTM model architecture. Each black box corresponds to the name of the layer, with its corresponding input and output shape noted in the white box. Note that the output of one layer serves as the input for the subsequent layer. 35

3.5 Layer configuration of the CNN-LSTM hybrid model architecture. Each black box corresponds to the name of the layer, with its corresponding input and output shape noted in the white box. Note that the output of one layer serves as the input for the subsequent layer, except for the concatenation layer where an additional input is included. The layers are in order from a to g. 39

4.1 Learning curve of the CNN model, including the training and validation sets. 42
4.2 Predicted and actual values of the inclination angle φ along the floating platform at time instance $t = 10$ s for a sample drawn from winter (left) and summer (right). 42
4.3 Predicted and actual values of the tilt φ over the first and last 100 s of a storm at $x = 51$ m for a sample drawn from winter (top) and summer (bottom). 43
4.4 The mean absolute error (MAE) 50th, 75th, 90th, and 95th percentiles over time for each location along the floating structure for a sample drawn from winter (left) and summer (right). 44
4.5 Learning curve of the LSTM model, including the training and validation sets. 46
4.6 Predicted and actual values of the inclination angle φ along the floating platform at time instance $t = 10$ s for a sample drawn from winter (left) and summer (right). 46
4.7 Predicted and actual values of the tilt φ over the first and last 100 s of a storm at $x = 1$ m for a sample drawn from winter (top) and summer (bottom). 47
4.8 The mean absolute error (MAE) at the 50th, 75th, 90th, and 95th percentiles over time for each location along the floating structure for a sample drawn from autumn (left) and summer (right). 48
4.9 Learning curve of the CNN-LSTM Hybrid model, including the training and validation sets. 49
4.10 Predicted and actual values of the inclination angle φ along the floating platform at time instance $t = 10$ s for a sample drawn from autumn (left) and summer (right). 50
4.11 Predicted and actual values of the tilt φ over the first and last 100 s of a storm at x = 1 m for a sample drawn from autumn (top) and summer (bottom). 51
4.12 The mean absolute error (MAE) at the 50th, 75th, 90th, and 95th percentiles over time for each location along the floating structure for a sample drawn from autumn (left) and summer (right). 52
4.13 Actual versus Predicted scatter plots of the CNN model for three locations along the structure, including an error magnitude color map. For a sample with the largest standard deviation: $4.15°$ (a), and a sample with the smallest standard deviation: $1.48° \times 10^{-2}$ (b). 54
4.14 Actual versus Predicted scatter plots of the CNN-LSTM Hybrid model for three locations along the structure, including an error magnitude color map. For a sample with the largest standard deviation: $4.24°$ (a), and a sample with the smallest standard deviation: $1.82° \times 10^{-2}$ (b). 55
4.15 Actual versus Predicted scatter plots of the CNN model for the three next to lowest standard deviation samples (upper plots), and for three samples of the most frequent standard deviation in the test dataset (lower plots). 56
4.16 Actual versus Predicted scatter plots of the CNN-LSTM Hybrid model for the three next to lowest standard deviation samples (upper plots), and for three samples of the most frequent standard deviation in the test dataset (lower plots). 56

5.1 The 95th percentile MAE (left), the 95th percentile MAPE (middle), and WAPE (right) along all locations of the structure under different input constraints. 58
5.2 The 95th percentile MAE (left), the 95th percentile MAPE (middle), and WAPE (right) along all locations of the structure under different input constraints. 59
5.3 The 95th percentile MAE (left), the 95th percentile MAPE (middle), and WAPE (right) along all locations of the structure under different input constraints. 60

A.1 Boxplot of hourly significant wave height values for each month. 71
A.2 Boxplot of hourly peak period values for each month. 72
A.3 Boxplot of the standard deviation of the tilt for each month. 72

C.1 CNN Architecture Visualization. 78
C.2 LSTM Architecture Visualization. 79

C.3 CNN-LSTM hybrid Architecture Visualization (1). 80
C.4 CNN-LSTM hybrid Architecture Visualization (2). 81

D.1 The predicted and actual values (CNN) of the inclination angle φ along the floating platform at time instance t = 10 s for a random sample in spring (left) and autumn (right). . . 82
D.2 Predicted and actual values (CNN) of the tilt φ over the first and last 100 s of a storm at x = 51 m for a sample drawn from spring (top) and autumn (bottom). 83
D.3 The mean absolute error, and mean absolute percentage error 50th, 75th, 90th, and 95th percentile (CNN) in time for every location along the floating structure for a sample drawn from spring (left) and autumn (right). 84
D.4 The predicted and actual values (LSTM) of the inclination angle φ along the floating platform at time instance t = 10 s for a random sample in spring (left) and autumn (right). 84
D.5 Predicted and actual values (LSTM) of the tilt φ over t = 0 - 100 s and t = 3300 - 3400 s of a storm at x = 1 m for a sample drawn from spring (top) and autumn (bottom). 85
D.6 The mean absolute error, and mean absolute percentage error 50th, 75th, 90th, and 95th percentile (LSTM) in time for every location along the floating structure for a sample drawn from spring (left) and autumn (right). 86
D.7 The predicted and actual values (CNN-LSTM hybrid) of the inclination angle φ along the floating platform at time instance t = 10 s for a random sample in winter (left) and spring (right). 86
D.8 The predicted and actual values (CNN-LSTM hybrid) of the inclination angle φ along the floating platform at time instance t = 500 s for a random sample in winter (left) and spring (right). 87
D.9 The predicted and actual values (CNN-LSTM hybrid) of the inclination angle φ along the floating platform at time instance t = 3000 s for a random sample in winter (left) and spring (right). 87
D.10 Predicted and actual values (CNN-LSTM hybrid) of the tilt φ over t = 0 - 100 s and t = 3500 - 3600 s of a storm at x = 1 m for a sample drawn from spring (top) and autumn (bottom). 88
D.11 The mean absolute error, and mean absolute percentage error 50th, 75th, 90th, and 95th percentile (CNN-LSTM hybrid) in time for every location along the floating structure for a sample drawn from spring (left) and autumn (right). 89

List of Tables

2.1 Environmental Properties. 19
2.2 Structural Properties of the floating platform. 20

3.1 Three distinct input scenarios that could significantly affect the tilt response of OFPVs. 26
3.2 Fixed parameter and hyperparameter specifications for the CNN architecture. 31
3.3 Fixed parameter and hyperparameter specifications for the LSTM architecture. 36
3.4 Fixed parameter and hyperparameter specifications for the CNN-LSTM hybrid architecture. 40

4.1 Evaluation metrics subjective to the CNN model. 45
4.2 Evaluation metrics for the LSTM model. 48
4.3 Evaluation metrics for the CNN-LSTM Hybrid model. 52

5.1 Reliability thresholds based on the MAPE and WAPE. 61
5.2 Reliability quantification under certain input constraints. 61

Nomenclature

Abbreviations

CNN	Convolutional Neural Network
DC	Direct Current
DHI	Diffuse Horizontal Irradiance
DNI	Direct Normal Irradiance
DNN	Dense Neural Network
FE-FSI	Finite Element Fluid-Structure Interaction
FSI	Fluid-Structure Interaction
GHI	Global Horizontal Irradiance
GRU	Gated Recurrent Unit
HDPE	High-Density Polyethylene
ISA	International Standard Atmosphere
JONSWAP	Joint North Sea Wave Project
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MLP	Multi-Layer Perceptron
OFPV	Offshore Floating Photovoltaic
OOM	Out-Of-Memory
PV	Photovoltaic
RNN	Recurrent Neural Network
VLFS	Very Large Floating Structure
WAPE	Weighted Absolute Percentage Error

Symbols

α	$\mathrm{m^2 s^{-4}}$	Empirical constant
β / φ	$^\circ$	Tilt angle

δ	–	Damping coefficient
η	m	Free surface elevation
η_{mod}	–	Module conversion efficiency
γ	–	JONSWAP peak enhancement factor
Γ_b		Beam boundary
Γ_{fs}		Free surface boundary
Γ_{in}		Inlet boundary
Γ_{out}		Outlet boundary
Γ_{sb}		Sea bed boundary
λ	m	Wavelength
λ_c	m	Characteristic wavelength
Λ_e		Free edge boundary
Λ_i		Location of the joints
Ω		Fluid domain
ω	$\mathrm{rad\,s^{-1}}$	Angular frequency
ϕ	$\mathrm{m\,s^{-1}}$	Velocity potential
ρ	$\mathrm{kg\,m^{-3}}$	Density
ρ_a	–	Albedo term
$\sigma(f)$	–	Spectral width parameter
τ	s	Wave period
θ	rad	Phase shift
θ_i	rad	Angle of incidence
A	$\mathrm{m^2}$	Area of PV module
a	m	Wave amplitude
d	m	Water depth
E	Pa	Young's modulus
EI	$\mathrm{N\,m^2}$	Bending stiffness
f	Hz	Frequency
F	N	Force
g	$\mathrm{m\,s^{-2}}$	Gravitational acceleration constant

h_b	m	Beam thickness
H_s	m	Significant wave height
I	$\mathrm{kg\,m^2}$	Moment of inertia
k	$\mathrm{m^{-1}}$	Wave number
L_{pv}	m	PV module length
L_{tot}	m	Total structural length
m_b	$\mathrm{kg\,m^{-2}}$	Beam mass per unit area
p	N	Transverse load
P_{DC}	W	DC Power output
Re	$-$	Reynolds number
$S(f)$	$\mathrm{m^2\,Hz^{-1}}$	Spectral density
t	s	Time
T_p	s	Peak period
u	m	Beam deflection
v	$\mathrm{m\,s^{-1}}$	Velocity
W	m	Width of structure

1

Introduction

The introductory chapter of this thesis begins with background information on the significance of offshore floating photovoltaics (OFPVs) in Section 1.1, emphasizing their role in renewable energy generation. In Section 1.2, the coupling between the irradiance and the tilt angle of OFPV systems is examined, highlighting how the tilt relates to the power output. Section 1.3 explores the underlying physics essential for understanding the behavior of OFPV systems, including Airy wave theory in Section 1.3.1, which describes the propagation of gravity waves on deep water surfaces, and Fluid-Structure Interaction (FSI) in Section 1.3.2, which highlights how fluid forces interact with floating structures. Section 1.4 discusses the current computational model used to simulate the response of very large floating structures (VLFSs), specifically the Finite Element Fluid-Structure Interaction (FE-FSI) model, which serves as a baseline for this project. Recognizing the limitations of existing models, Section 1.5 introduces machine learning as a potential solution, setting the foundation for the surrogate models proposed in this research. Section 1.6 outlines the research questions guiding this thesis, addressing the identified research gaps. Section 1.7 defines the scope of the project, specifying the objectives and boundaries of the study. Finally, Section 1.8 presents an outline of the methodology employed to achieve the research objectives.

1.1. Background

Due to the swift energy transition, the demand for renewable energy is increasing rapidly. Action is essential to mitigate the effects of global climate change. The European Union has set a renewable energy target of 42.5% by 2030, placing pressure on governments to develop additional renewable energy sources [8]. Photovoltaic energy is considered one of the most promising renewable energy sources for meeting these targets. However, photovoltaic systems require large surface areas, which makes inland deployment challenging, particularly in densely populated regions [7]. In addition to spatial constraints, previous studies have shown that the annual energy output of offshore photovoltaic systems is 12.96% higher than that of similar land-based systems [12]. While large offshore wind farms are now widely deployed, OFPV systems still need to expand their global footprint. Offshore wind farms can provide substantial energy throughout the year [2]. However, due to the limitations of current energy storage technologies and the intermittency of wind power, complementary renewable sources are needed to ensure a stable energy supply. Many ongoing plans aim to install OFPV systems alongside existing wind farms, optimizing the use of offshore space, smoothing the inter-annual variability of wind energy, and providing more consistent power generation across seasons.

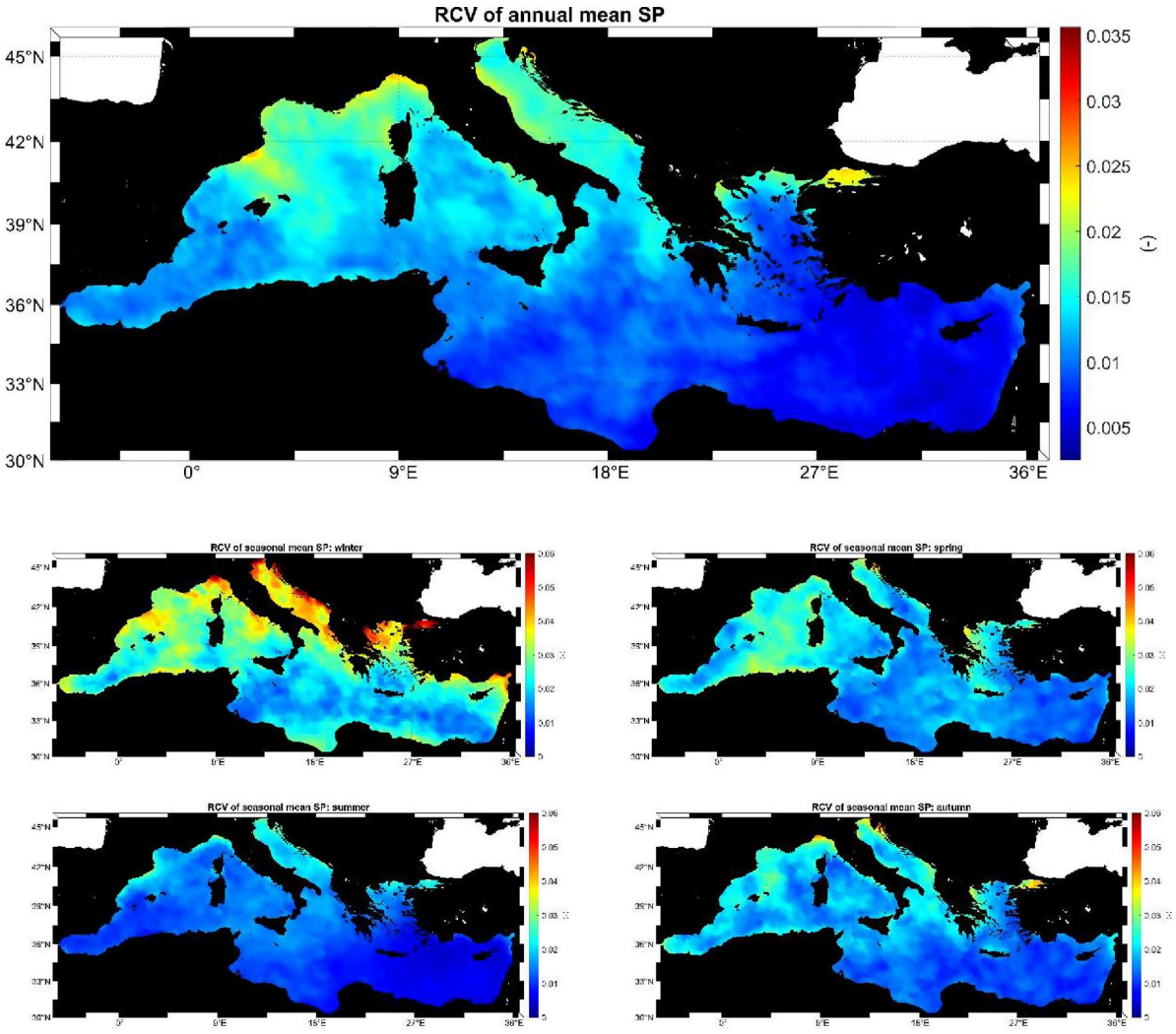


Figure 1.1: Robust coefficient of variation for the mean annual (upper panel) and mean seasonal wind power density for the Mediterranean Sea in winter (middle left panel), spring (middle right panel), summer (lower left panel), and autumn (lower right panel) [14].

Wind energy and solar energy are negatively correlated, meaning that during windy seasons, wind energy can supply the majority of power, whereas photovoltaic energy can maintain renewable energy output during periods of low wind velocity [11]. In Figure 1.1, the coefficient of variation of wind power density around the Mediterranean Sea is presented. It is evident that wind power density is highest in winter, while Figure 1.2 shows that solar irradiance is lowest during the same season. The opposite trend is observed in spring and summer [32].

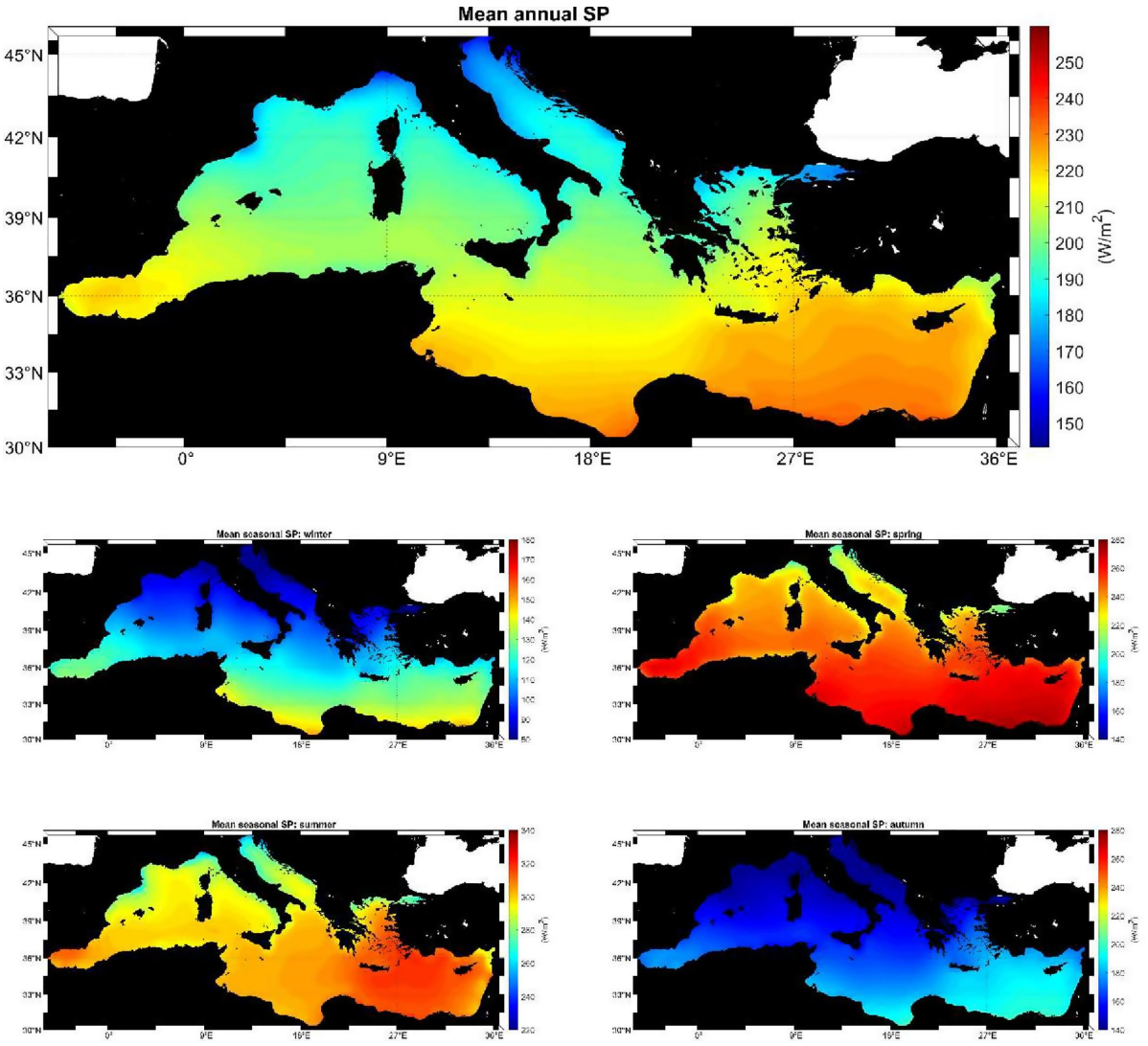


Figure 1.2: Mean annual (upper panel) and mean seasonal solar irradiance for the Mediterranean Sea in winter (middle left panel), spring (middle right panel), summer (lower left panel), and autumn (lower right panel) [14].

Soukissian et al. [14] conducted a study to evaluate the complementarity of offshore wind and solar resources. Monthly energy outputs of wind, photovoltaic, and their combination were analyzed at six different locations around the Mediterranean Sea. The results, presented in Figure 1.3, show that while photovoltaic energy exhibits a parabolic trend and wind energy follows an inverse parabolic trend, their combination produces a more uniform energy output. This flattening of the curves indicates that the hybrid system can provide a more consistent renewable energy supply throughout the seasons. It is therefore important to optimize solar energy capture and enhance the power output of photovoltaic panels.

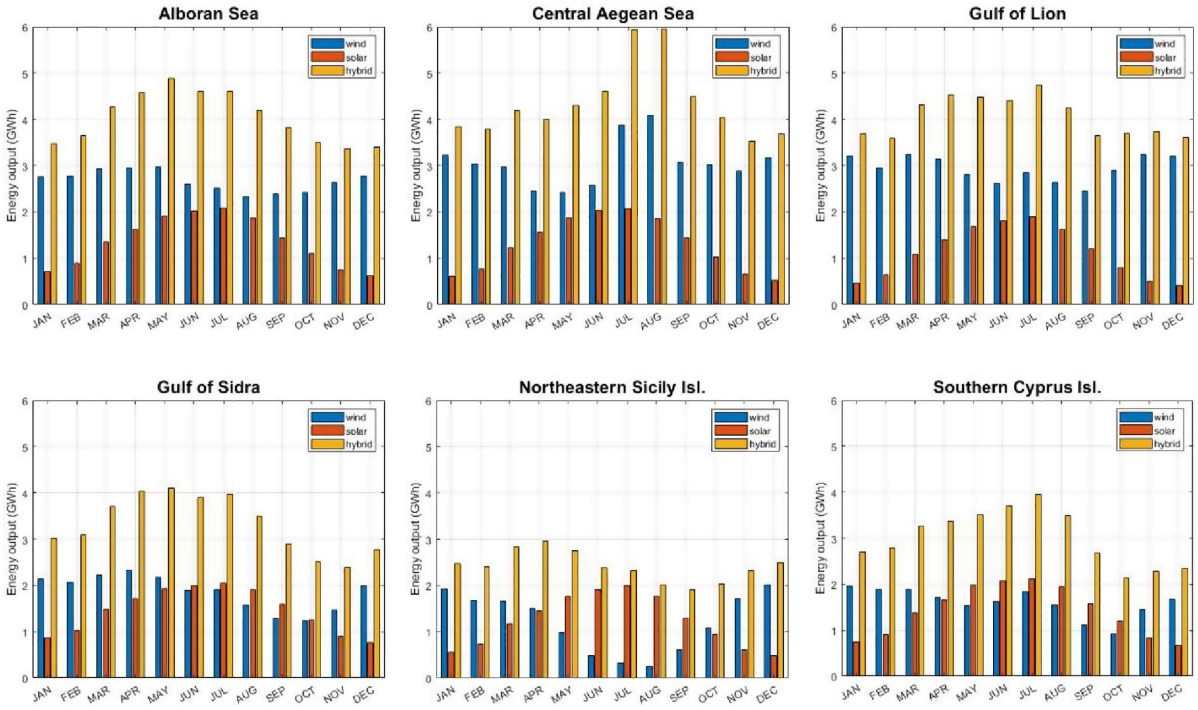


Figure 1.3: Monthly values of energy output of an 8-MW offshore wind turbine, an 8-MW floating solar farm, and their combination at the Alboran Sea (upper left), the central Aegean Sea (upper middle), the Gulf of Lion (upper right), the Gulf of Sidra (lower left), the Northeastern part of Sicily (lower middle), and the Southern part of Cyprus (lower right) [14].

1.2. Irradiance-Tilt Coupling

OFPV systems float on the sea surface, and their motion is closely related to the properties of the waves. The interaction between the floater and the waves is crucial for analyzing the structure's motion, as the floater's movement directly affects the tilt angle, β. As illustrated in Figure 1.4, the irradiance (an instantaneous measurement of solar power received per unit area (in W/m^2)) on a module comprises three types of radiation: Global Horizontal Irradiance (GHI) (I_{GHI}), Direct Normal Irradiance (DNI) (I_{DNI}), and Diffuse Horizontal Irradiance (DHI) (I_{DHI}).

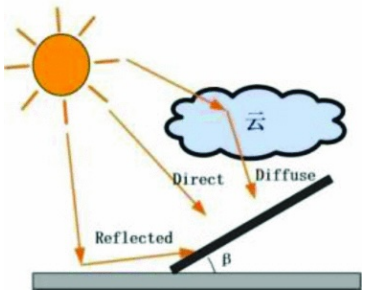


Figure 1.4: Global horizontal (reflected) irradiance (I_{GHI}), direct normal irradiance (I_{DNI}), and diffuse horizontal irradiance (I_{DHI}) [30].

These components are all influenced by the tilt angle, β, before contributing to the total irradiance ($G_{\text{total}}(\beta)$), which consists of direct beam irradiance ($G_{\text{beam}}(\beta)$), diffuse irradiance ($G_{\text{diffuse}}(\beta)$), and reflected irradiance ($G_{\text{reflected}}(\beta)$) [30]:

$$G_{\text{total}}(\beta) = G_{\text{beam}}(\beta) + G_{\text{diffuse}}(\beta) + G_{\text{reflected}}(\beta) \tag{1.1}$$

The total irradiance is directly proportional to the direct current (DC) power output of the PV module. A

simple approximation of the DC power output in watts (W) is given by:

$$P_{DC} = A_{PV} \times \eta_{\text{mod}} \times G_{\text{total}}(\beta) \tag{1.2}$$

Where:

- A_{PV}: area of the PV module (m^2)
- η_{mod}: overall module conversion efficiency (-)

So how does a change in β relate to a change in the total irradiance?

When assuming an isotropic sky, the diffuse irradiance can be approximated using the Liu and Jordan model, as shown in Equation 1.3. A smaller tilt angle, β, results in a larger cosine term, thereby increasing the diffuse irradiance.

Reflected irradiance can be estimated using ground reflectance, as demonstrated in Equation 1.4. Reflected irradiance depends on the ground reflectance (ρ_a) and ranges from 0 (no reflection) to 1 (total reflection), known as the albedo. In offshore environments, ρ_a represents the water surface reflectivity, which typically has a low albedo (< 0.1). Unlike diffuse irradiance, reflected irradiance includes a negative cosine term. Therefore, only large tilt angles result in significant reflected irradiance. However, since the pontoons supporting OFPV systems are moored to buoys or the seabed, tilt angles are often limited to a maximum of 20° [10]. Consequently, the reflected irradiance component remains very small in the calculation of the total irradiance for OFPVs.

$$G_{\text{diffuse}}(\beta) = I_{DHI} \times \frac{1 + \cos(\beta)}{2} \tag{1.3}$$

$$G_{\text{reflected}}(\beta) = I_{GHI} \times \rho_a \times \frac{1 - \cos(\beta)}{2} \tag{1.4}$$

The last term, the direct beam irradiance, can be approximated as following:

$$G_{\text{beam}}(\beta) = I_{DNI} \times \cos(\theta_i) \tag{1.5}$$

The angle of incidence, θ_i, is defined as the angle between the sun's rays and the normal to the PV surface. It relates to the tilt angle, β, in a complex manner, as θ_i depends on several factors: solar declination (δ), solar zenith angle (θ_z), solar azimuth angle (γ_s), hour angle (ω), surface azimuth angle (γ), and latitude (ϕ). For south-facing PV panels in the Northern Hemisphere, the relationship between θ_i and β can be simplified to:

$$\cos(\theta_i) = \cos(\beta) \times (\sin(\delta)\sin(\phi) + \cos(\delta)\cos(\phi)) \tag{1.6}$$

From the simplified equation, it can be seen that the angle of incidence (θ_i) and the tilt angle (β) are proportional to each other. This means that a smaller tilt angle, β, results in a larger direct beam irradiance. For full derivations, approximations, and simplifications of these formulas, the reader is referred to the book *Solar Engineering of Thermal Processes* [9].

Considering that reflected irradiance is minimal for OFPV systems, and combining the equations above, the DC power output increases as the tilt angle, β, decreases. To accurately predict this tilt response, it is essential to understand the motion dynamics of OFPV systems.

1.3. Underlying Physics

This section introduces the underlying physics essential for analyzing the motion of OFPVs. The equations presented here form the foundation of the dataset used in the surrogate model, which is further detailed in Section 1.4. The hydrodynamics governing the motion of OFPVs are based on fundamental equations derived from conservation laws, as discussed in Subsection 1.3.1. The interaction between fluids and solid structures, known as Fluid-Structure Interaction (FSI), describes how these interactions affect the motion of both the fluid and the structure. Subsection 1.3.2 analyzes this phenomenon in the context of offshore environments.

1.3.1. Linear Wave Theory

The motion of offshore floaters is highly influenced by wave-induced forces, resulting in tilting. These forces are characterized by wave motion, typically described using wave theory. In offshore environments, waves exhibit non-linear behavior; however, in certain cases, their motion can be approximated as linear, as described by linear wave theory. The primary assumption underlying this theory is the *small-amplitude approximation*, which assumes that the wave elevation η, expressed as a fraction of the wavelength λ and the water depth d, is relatively small (Equations 1.7 and 1.8). These ratios are commonly referred to as wave steepness and relative depth. Another method to assess the applicability of linear wave theory is by consulting the graph in Figure 1.5, which indicates whether linear wave theory is appropriate or if a non-linear wave theory should be applied. Additionally, the graph shows the breaking limit of waves.

$$\frac{\eta}{\lambda} \ll 1 \tag{1.7}$$

$$\frac{\eta}{d} \ll 1 \tag{1.8}$$

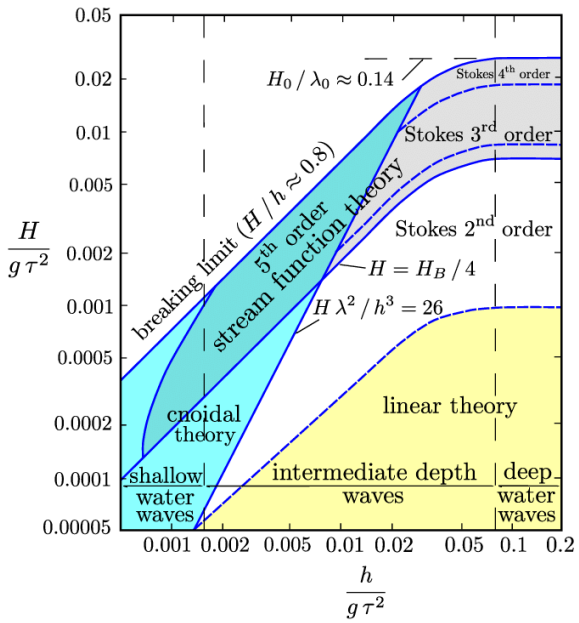


Figure 1.5: Ranges of applicability of different wave theories. With H being the wave height, h being the water depth, τ being the wave period, and g the gravitational constant [5].

Other key assumptions in Linear wave theory are:

- The fluid has zero viscosity
- The fluid is incompressible
- The flow is irrotational
- The fluid properties are uniform (homogeneous)
- The depth is constant in the domain considered
- The bottom is rigid and impermeable
- Gravitation is the sole restoring force
- The flow is only considered in a two-dimensional plane

Linear wave theory, also known as Airy wave theory, is derived from two fundamental conservation equations: the mass balance and the momentum balance. Under the aforementioned assumptions, these equations can be simplified to the Laplace and linearized Bernoulli equations by introducing a velocity potential function, whose spatial derivatives represent the velocities of the water particles (Equation 1.9). Figure 1.6 presents these equations along with the kinematic and dynamic boundary conditions applied to the domain Ω. For a detailed derivation, the reader is referred to the textbook *Waves in Oceanic and Coastal Waters* [15]

$$\phi(x,y,z,t) \; with \; u_x = \frac{\partial \phi}{\partial x}, \; u_y = \frac{\partial \phi}{\partial y} \; and \; u_z = \frac{\partial \phi}{\partial z} \tag{1.9}$$

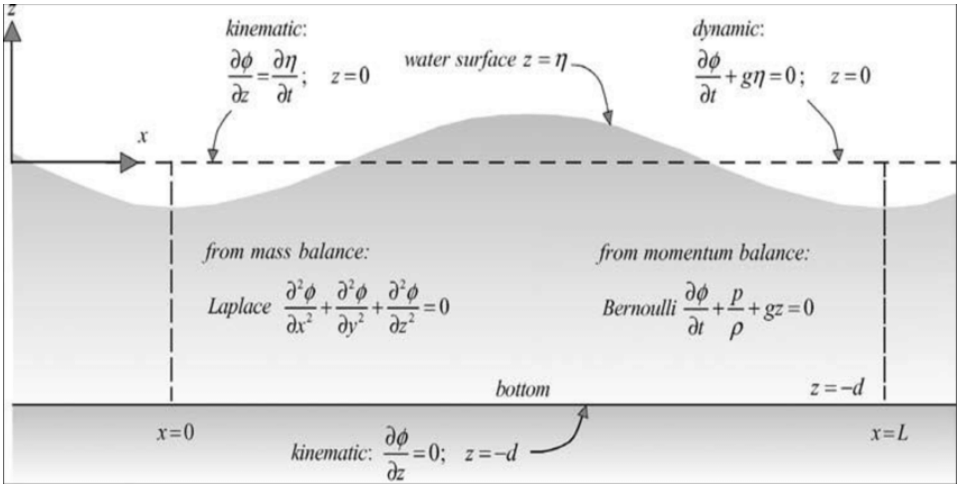


Figure 1.6: The Laplace and linearized Bernoulli equations and boundary conditions for linear wave theory, in terms of the velocity potential ϕ applied to a domain Ω [15].

$$\eta(x,t) = a\sin(\omega t \; - \; kx \; + \; \theta) \tag{1.10}$$

$$\phi(x,z,t) = \hat{\phi(z)}\cos(\omega t \; - \; kx \; + \; \theta) \;\; with \;\; \hat{\phi(z)} = \frac{\omega a}{k}\frac{\cosh[k(d+z)]}{\sinh(kd)} \tag{1.11}$$

An analytical solution to the linearized equations can be expressed as a propagating harmonic wave $\eta(x,t)$ (Equation 1.10), characterized by the wave amplitude a, wave number k, angular frequency ω, and phase shift θ. This solution is associated with a velocity potential, as shown in Equation 1.11, which also depends on the depth d. The propagating harmonic wave represents only one of the many waves present in offshore waters. In Linear Wave Theory, the total wave motion in offshore waters can be represented as a superposition of harmonic waves with varying amplitudes, wave numbers,

frequencies, and directions. Considering N linear waves, each with distinct characteristics, the total wave elevation η at a given position x and time t is given by:

$$\eta(x,t) = \sum_{i=1}^{N} a_i \sin\left(\mathbf{k_i x} - \omega_i t + \theta_i\right) \quad with \;\; \mathbf{k_i} \;=\; k_i \; \mathbf{n_i} \tag{1.12}$$

Where:

- $\eta(x,t)$: Total wave elevation at position x and time t.
- a_i: Amplitude of the i-th wave.
- ω_i: Angular frequency of the i-th wave, $\omega_i = \dfrac{2\pi}{T_i}$, with T_i being the period.
- θ_i: Phase shift of the i-th wave.
- k_i: Wavenumber of the i-th wave, $k_i = \dfrac{2\pi}{\lambda_i}$, with λ_i being the wavelength.
- $\mathbf{n_i}$: Unit normal factor indicating the direction of the propagation of the i-th wave.

The combination of these waves is commonly represented by a wave spectrum, such as the Pierson-Moskowitz or JONSWAP spectrum, which describes the distribution of wave energy across different frequencies and directions. Further details are provided in Section 2.2.

1.3.2. Fluid-structure interaction

FSI refers to the interplay between fluid motion and a structural body, where the fluid influences the structure's motion, and conversely, the structure affects the fluid's motion. This multi-physics phenomenon requires solving both fluid dynamics and structural mechanics equations simultaneously. FSI typically involves two types of interactions. The first is *Rigid Body Interaction*, where the structure moves as a whole without deforming, such as a ship moving through water. The second type is *Flexible Body Interaction*, in which the structure undergoes partial deformation due to fluid forces, a common scenario in VLFSs. Zhang et al. [33] introduced a third type: *Very Flexible Body Interaction*, where the structure almost entirely deforms under fluid forces, often characterized by very small structural thickness.

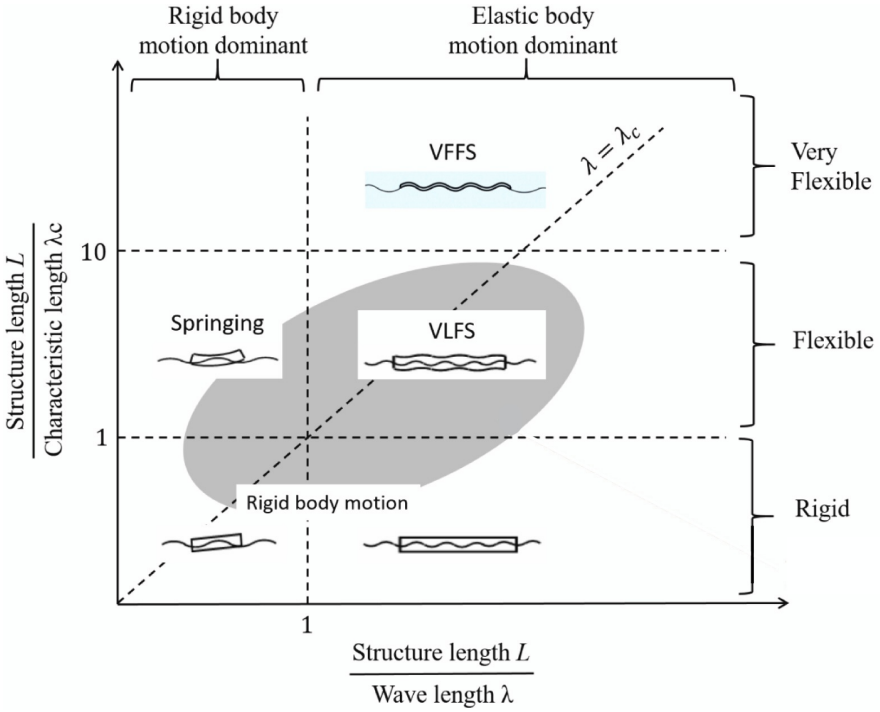


Figure 1.7: The global response of floating structures under fluid forces [33].

A quick way to determine the dominant type of interaction is by consulting the graph in Figure 1.7. The key parameter in this graph is the characteristic length λ_c, which is defined as:

$$\lambda_c = 2\pi\Big(\frac{EI}{\rho g}\Big)^{\frac{1}{4}}. \tag{1.13}$$

Here, the product ρg represents the hydrostatic stiffness of the structure, while EI denotes the bending stiffness, where E is the Young's modulus and I is the moment of inertia. When $\lim_{EI\to 0}$, the structure lacks rigidity, causing its motion to fully follow the fluid motion, making it 100% *Very Flexible Body Motion dominant*. Conversely, when $\lim_{EI\to\infty}$, the structure becomes perfectly rigid and does not deform under fluid forces, making it 100% *Rigid Body Motion dominant*. For VLFSs, the ratio of structural length L to characteristic length λ_c typically ranges between 1 and 10. In this range, the coupling between fluid dynamics and structural mechanics equations determines the magnitude of the response.

To model this FSI, it can be convenient to look at the same domain Ω as discussed in the previous subsection. However, in this case the kinematic boundary condition

$$\frac{\partial \phi}{\partial z} = \frac{\partial \eta}{\partial t} \;\; on \;\; \Gamma_{fs} \tag{1.14}$$

and dynamic boundary condition

$$\frac{\partial \phi}{\partial t} \; + \; g\eta = \; 0 \;\; on \;\; \Gamma_{fs} \tag{1.15}$$

are no longer valid, as there is a floating structure instead of a free surface. Consequently, the boundary Γ_{fs} is replaced by Γ_b, where the governing equations couple the fluid dynamics with the structural mechanics at this boundary. It is important to note that the boundary conditions adjacent to the platform at Γ_{fs} remain unchanged. Figure 1.8 illustrates the boundary value problem.

Assuming, as in Airy wave theory, that the flow is two-dimensional and focusing on the transverse deflection of the beam, the Euler-Bernoulli beam theory can be used to model the beam. Given that both the velocity potential ϕ and the propagating wave η are harmonic, it is reasonable to assume that the beam's response u is also harmonic, resulting in complex-valued solutions. Therefore, the deflection of the beam, which varies in both time and space, is defined as:

$$u(x,t) = u(x)\exp(-i\omega t). \tag{1.16}$$

Which can be separated into a time-dependent and space-dependent component, simplifying both the equation of motion and the boundary conditions. Assuming the beam is homogeneous, the equation of motion is given by:

$$EI\frac{\partial^4 u}{\partial x^4} \; - \; m_b\omega^2 u = \; p \tag{1.17}$$

where EI is the beam's bending stiffness, m_b the beam's mass per unit area and p the transverse load.

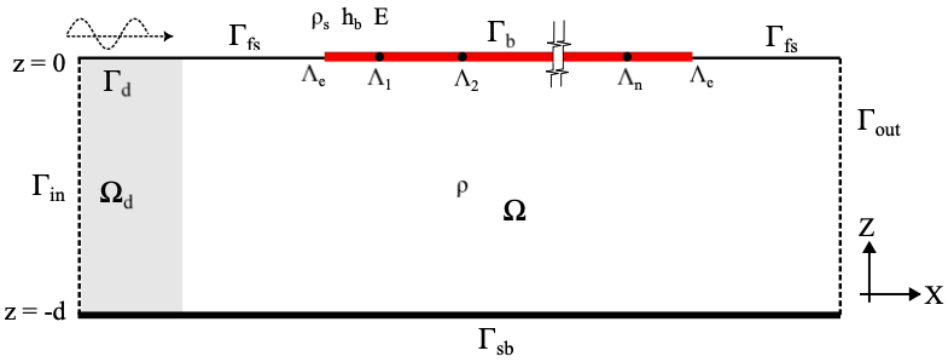


Figure 1.8: A schematic of the boundary value problem of the fluid domain Ω, bounded by the inlet Γ_{in}, outlet Γ_{out}, sea bed Γ_{sb}, free surface Γ_{fs}, and floating platform Γ_b[1].

With the assumption that there is no gap between the beam and the fluid, the kinematic boundary condition 1.14 can be simplified to

$$\frac{\partial \phi}{\partial z} \; + \; i\omega u \; = \; 0 \;\; on \;\; \Gamma_b. \tag{1.18}$$

The dynamic boundary condition couples the fluid dynamics with the structural mechanics by equating the transverse load in the beam's equation of motion to the pressure exerted by the fluid beneath the beam, which is derived from the linearized dynamic Bernoulli equation. Consequently, the dynamic boundary condition becomes:

$$EI\frac{\partial^4 u}{\partial x^4} \; - \; m_b\omega^2 u \; - \; i\omega\rho\phi \; + \; \rho g u \; = \; 0 \;\; on \;\; \Gamma_b \tag{1.19}$$

Here, g denotes the gravitational constant, ρ represents the density of water, and the remaining parameters are as previously defined. Incorporating the beam's equation of motion into the fluid domain requires additional boundary conditions for the beam. These conditions can be determined by defining the boundaries Λ_e as free-edge boundaries, where the moments and shear forces are set to zero.

$$EI\frac{\partial^2 u}{\partial x^2} \; = \; 0 \;\; on \;\; \Lambda_e. \tag{1.20}$$

$$EI\frac{\partial^3 u}{\partial x^3} = 0 \ \ on \ \ \Lambda_e. \tag{1.21}$$

In case multiple floaters are present, instead of just a singular floater, equation 1.20 is also applicable at the joints Λ_i between the floaters, setting again the bending moment to zero.

$$EI\frac{\partial^2 u}{\partial x^2} = 0 \ \ on \ \ \Lambda_i. \tag{1.22}$$

It should be noted that this section focuses solely on the influence of the fluid on the structural body, without considering the reverse effect. Factors such as added mass and added damping, which influence the response of VLFSs and affect the fluid around the structure, are acknowledged. In this problem, the effect is mitigated by introducing a damping zone Ω_d to absorb waves reflected by the floating platform. For a detailed description of this approach, the reader is referred to [24]. The equations presented in this section serve as the foundation and input for the FE-FSI model discussed in the next section.

1.4. Current model

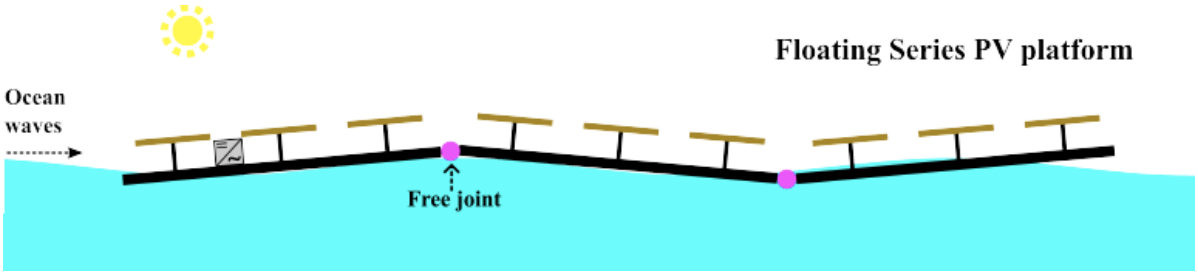


Figure 1.9: Offshore floating photovoltaic system with 3 floaters and 2 joints [1].

Alcañiz et al. [1] conducted an optoelectrical analysis of the response of OFPVs, including a structural analysis of the underlying VLFS, as schematized in Figure 1.9. The response of these VLFSs is solved numerically using a Finite Element (FE) model combined with the FSI equations presented in the previous section. This analysis builds on the complex hydroelastic FSI model developed by Colomés et al. [24] and employs a monolithic finite element scheme, where the coupled boundary value problem is formulated as a single unified weak form rather than treating each governing equation separately. The problem is first discretized in space using the same domain Ω defined in Section 1.3 (see Figure 1.10), followed by time discretization using a frequency domain approach, as the focus is on harmonic or steady-state solutions. This approach simplifies the coupling of the equations and facilitates the inclusion of damping zones to absorb reflected waves.

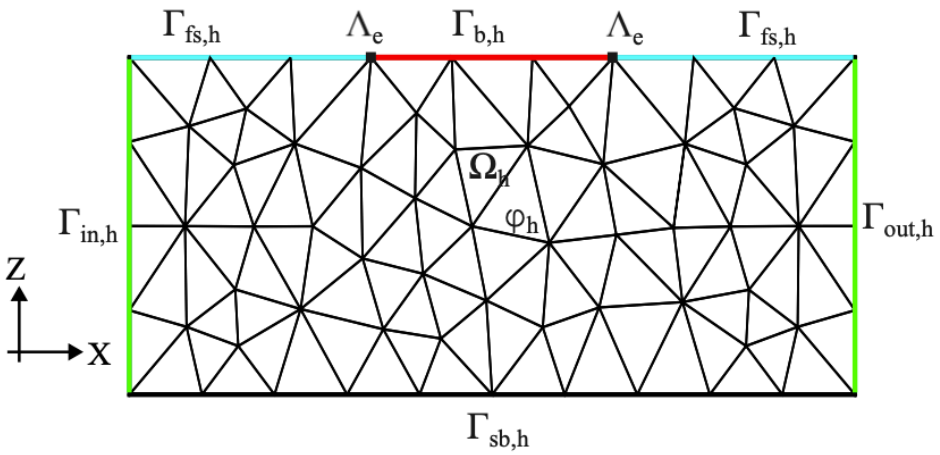


Figure 1.10: A sketch of the discretized fluid domain Ω bounded by the inlet Γ_{in}, outlet Γ_{out}, sea bed Γ_{sb}, free surface Γ_{fs}, and floating platform Γ_b used in the FE-FSI model [1].

The FE-FSI model uses environmental conditions and platform properties as input, while keeping the water depth d and structure length L constant. For multiple floaters, the floaters are connected through joints. The output consists of complex-valued solutions for beam deflection u, free-surface elevation η, and velocity potential ϕ, providing both the magnitude and phase of the response. The various configurations and magnitudes of the environmental conditions and platform properties are discussed in Chapter 2, as they serve as input to the model proposed in the next section.

1.5. Limitations: Introduction to Machine Learning

A numerical simulation tool, such as the FE-FSI model described in the previous section, is invaluable when real-life sensor data is unavailable or impractical. It can accurately predict specific phenomena and responses with negligible errors. However, despite advancements in processing power, such simulations remain computationally expensive and time-intensive, particularly when numerous simulations with varying parameters are required. As the offshore solar industry continues to grow, a computationally efficient and faster alternative is increasingly desirable for enhancing the performance of OFPVs.

Machine learning, which has gained significant attention across various industries and simplified complex engineering problems, offers a promising alternative. Several machine learning techniques can be employed to develop surrogate models, including supervised learning methods such as Linear and Polynomial Regression, Decision Trees, and Random Forests, as well as unsupervised learning methods like K-Means Clustering and Principal Component Analysis. Each method has distinct strengths and is suited to specific types of problems. This research focuses on deep learning, a branch of machine learning capable of modeling complex linear and non-linear phenomena by extracting patterns from input data and generalizing them into continuous outputs [27]. This introduction to machine learning will explore different deep learning methods, their proven utility in various engineering applications, and their potential for predicting the response of VLFSs.

A dense neural network (DNN), also known as a Multi-Layer Perceptron (MLP), is a neural network architecture in which every neuron in one layer is fully connected to every neuron in the subsequent layer. DNNs are widely used for identification and classification tasks [20], and many offshore floating structures can be effectively monitored using this approach. Wang et al. [3] applied a DNN to infer information about an offshore floating wind turbine at the location of a faulty sensor using data from functioning sensors. Similarly, Kwon et al. [19] predicted the structural responses of a submerged floating tunnel using a DNN. In addition, DNNs have been employed to improve the accuracy of significant wave height predictions in offshore environments, an essential parameter in offshore data processing. Studies have shown that DNNs can reduce the prediction error of significant wave heights from 26% (with numerical models) to 12% [13].

Although DNNs have proven useful, convolutional neural networks (CNNs) are often preferred due to their ability to recognize local patterns and dependencies more effectively. This capability is particularly advantageous when local dependencies are more critical than global ones, or when working with image-based input data, such as satellite images or structural health monitoring imagery [28]. CNNs also provide translational invariance, making them suitable for offshore environments where the exact location of features may vary. Compared to DNNs, CNNs excel at extracting damage-sensitive features due to their local pattern recognition capabilities. Consequently, CNNs are frequently applied in structural health monitoring, including damage detection in offshore jacket structures [4] and damage identification and classification of mooring lines in offshore floating wind turbines [16]. Additionally, CNNs can be employed for continuous input data. Kawai et al. [17] utilized a CNN architecture to estimate the sea state by deriving frequency features from response spectra, demonstrating its utility in offshore applications.

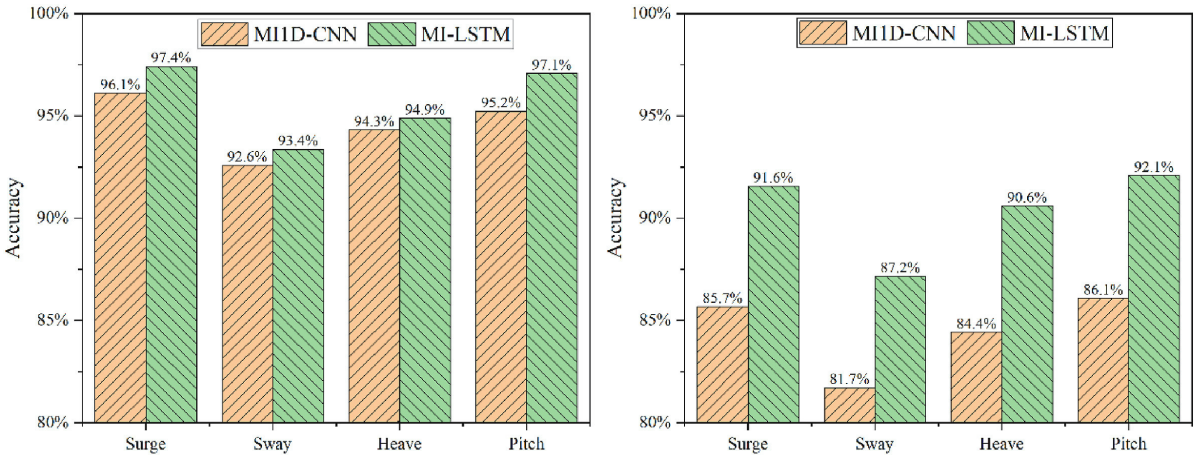


Figure 1.11: Comparison of the overall accuracy of the LSTM model and the 1D-CNN model, capturing the response over left) 2.5 s and right) 5.0 s [29].

Although CNNs are effective for damage recognition, they are less suited for capturing temporal dependencies, as structural health changes typically occur over extended time intervals. While CNNs can handle time series data, converting the input from the time domain to the frequency domain is often recommended. Many offshore phenomena, such as incoming wind waves, tidal waves, currents, and temperature, are time-dependent and can be modeled using deep learning techniques. Recurrent neural networks (RNNs) are well-suited for such tasks, as they can capture sequential data and temporal dynamics by remembering previous inputs. However, a major limitation of RNNs is the vanishing or exploding gradient problem during training, which makes learning long-term dependencies challenging. To address this, variations of the RNN architecture, such as Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and bidirectional RNNs, have been developed. Bidirectional RNNs process input data in both forward and backward directions, making them useful when future information is relevant to the current state.

Shi et al. [29] used an LSTM architecture to predict the short-term motion of a floating offshore wind turbine, incorporating second-order hydrodynamic effects. Their study compared the LSTM model to a multi-input 1D-CNN trained on the same dataset, aiming to evaluate the impact of time dependency on structural response and patterns between time steps. Figure 1.11 presents a comparison of the models' accuracies for different forecast times. At a forecast time of 2.5 s (left figure), both models show similar accuracy, with a difference of about ±1%. However, at a forecast time of 5.0 s (right figure), the accuracy of the LSTM model improves by approximately 5%. This demonstrates that temporal neural network architectures, such as LSTMs, can outperform CNNs in time series prediction, particularly as the forecast interval increases.

Although RNN architectures generally handle time-series data more effectively, important features ex-

tracted by CNN models can also be valuable when using raw sensor data in time-series applications. Ordóñez et al. [26] were among the first to combine CNNs with RNNs for human activity recognition, a task that involves complex motor sequences and requires capturing temporal dynamics. To address this challenge, they developed a hybrid DeepConvLSTM model. Following this, hybrid models found applications in engineering fields. Canizo et al. [6] proposed a CNN-RNN architecture for anomaly detection in high-value machinery, where the CNN extracts meaningful features from sensor data, and the RNN learns temporal patterns.

Figure 1.12 illustrates the architecture of the hybrid model. Rather than processing the entire time-series data from all sensors at once, the data is segmented into smaller windows, allowing feature extraction to occur iteratively for each window. This approach enables the model to focus on distinct phases within the time series. The extracted features from all windows are subsequently combined and processed by the RNN component of the model.

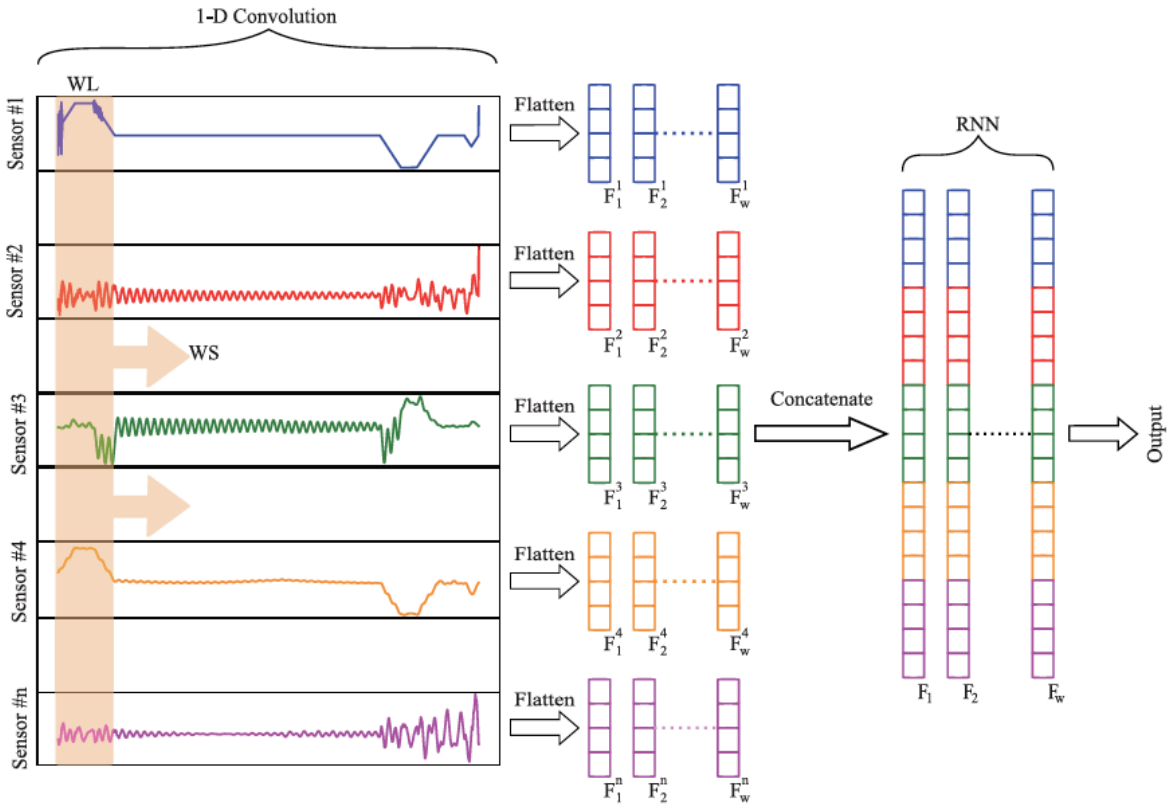


Figure 1.12: The hybrid CNN-LSTM architecture. The data coming from the sensors are all processed individually. The time-series is divided into multiple time windows, resulting in a feature map per window. The feature maps belonging to the same time window are concatenated and chronologically used in the subsequent RNN [6].

LSTM architectures have proven effective in combination with CNNs, but GRUs can also perform well in hybrid models. Li et al. [22] proposed a model for forecasting ship motion that combines a CNN with a GRU, followed by an attention layer. The attention layer assigns higher weights to important time steps, enhancing the model's focus on significant events. The architecture from that study is schematized in Figure 1.13. The order of components mirrors that of the DeepConvLSTM model: first, a one-dimensional CNN extracts key features through successive convolution and pooling layers, reducing dimensionality. The resulting vectors are passed to the GRU layer, which captures temporal features. An attention layer follows, weighting critical temporal features before applying a softmax function. This approach is particularly useful when specific past events, such as tidal waves or recurring currents, have a greater impact on the present state. Finally, the output layer combines all learned features.

Bidirectional RNNs can be useful when future data is relevant; however, in structural health monitoring

and wave or weather forecasting, this is rarely the case. As a result, bidirectional architectures are generally unsuitable for offshore engineering problems. Wang et al. [31] employed a bidirectional LSTM network to predict ship roll angles, where the bidirectional setup allowed feature extraction from both forward and reverse sequences of the time series. Despite this, LSTM and GRU architectures remain more common in offshore engineering due to their simplicity and ability to effectively capture temporal patterns.

While literature demonstrates the successful prediction of ship motion using hybrid models, no existing studies have modeled the motion of floating offshore structures using CNN-LSTM or CNN-GRU architectures. Therefore, the hybrid model proposed by Li et al. represents a state-of-the-art approach in this domain.

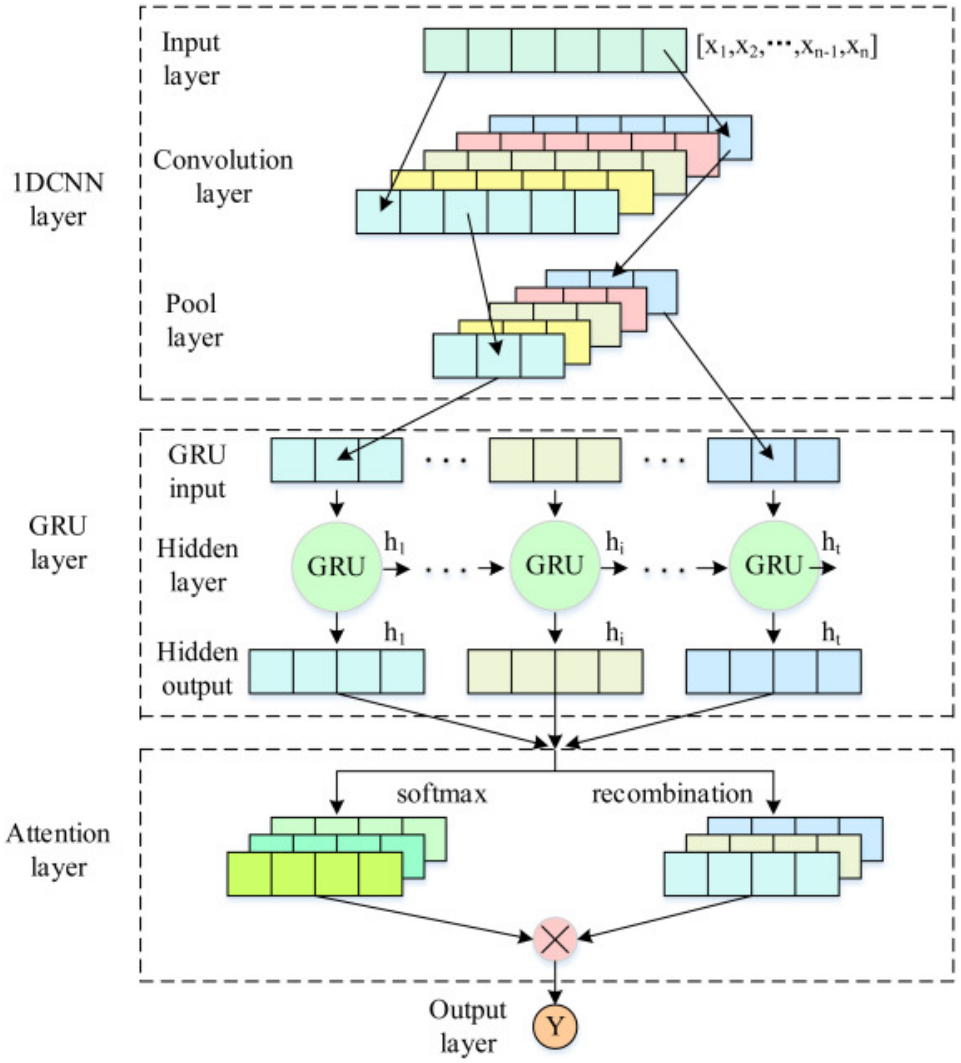


Figure 1.13: The hybrid CNN-GRU architecture used to forecast ship motion. Including a 1D CNN layer, Gated Recurring Unit layer, and an attention layer.

The combination of these studies demonstrates that deep learning models can be highly complex and challenging to manage. A model that appears suitable for a specific problem does not always yield the best results. Proper tuning of input parameters is crucial when employing different approaches, as it can significantly enhance the model's performance. Additionally, increasing the amount of data does not always improve the results. Nevertheless, existing literature indicates that there remains considerable potential for further exploration in applying deep learning to offshore floating engineering problems.

1.6. Research Questions

This research will explore the potential of machine learning in engineering by predicting the tilt response of VLFSs, which serve as the floating foundations for OFPVs. The goal of this research is to answer the following research question:

How to develop and optimize a surrogate machine learning model to accurately predict the tilt response of very large floating structures?

In order to answer this main research question, it is divided in the following sub-questions that will be answered throughout this thesis:

- How effectively do temporal (e.g., LSTM) and spatial (e.g., CNN) neural networks address the interplay between wave-induced motions and tilt response in offshore floating solar platforms?
- How do variations in environmental input parameters (e.g., wave height, peak period) affect the performance of the surrogate model?
- What challenges and advantages arise when implementing real-time deep learning models for predicting VLFS tilt responses under varying environmental conditions?
- How do variations in deep learning model architectures (e.g., CNN, LSTM, hybrid models) influence accuracy and computational efficiency in tilt response prediction?
- To what extent can the final surrogate machine learning model be considered reliable for predicting VLFS tilt responses across varying scenarios?

1.7. Research Scope

Literature such as [24] and [1] has explored the response of floating structures and identified variables critical to the design of OFPVs. Where these traditional FE-FSI simulations, while highly accurate, are resource-intensive and impractical for large-scale or real-time applications. This research aims to extend these studies by proposing multiple deep learning frameworks that can predict the tilt response of a specific OFPV system instantaneously (once trained), for varying environmental scenarios. Such a model opens the door for site-adapted operational rules, near real-time operational decision support, and integrated digital twin solutions. However, the proposed model is not meant to replace FE-FSI simulations entirely. Rather, it augments them by serving as a surrogate for repeated or real-time predictions, thus substantially reducing computational overhead during the conceptual design phase or routine monitoring.

The scope of this research excludes the effect of mooring lines on the structure, as well as the direct calculation of power output. Instead, the model focuses on predicting the tilt response of the structure under irregular wave conditions, based on linear wave theory. While the calculation of power output is outside the scope of this thesis, it can be addressed in future studies by conducting an optoelectrical analysis that incorporates the tilt response predicted by this model as an input. For the purposes of this thesis, maximizing the power output of OFPVs is equivalent to minimizing the tilt angle, φ (previously denoted as β), as demonstrated in Section 1.2.

1.8. Research Methodology

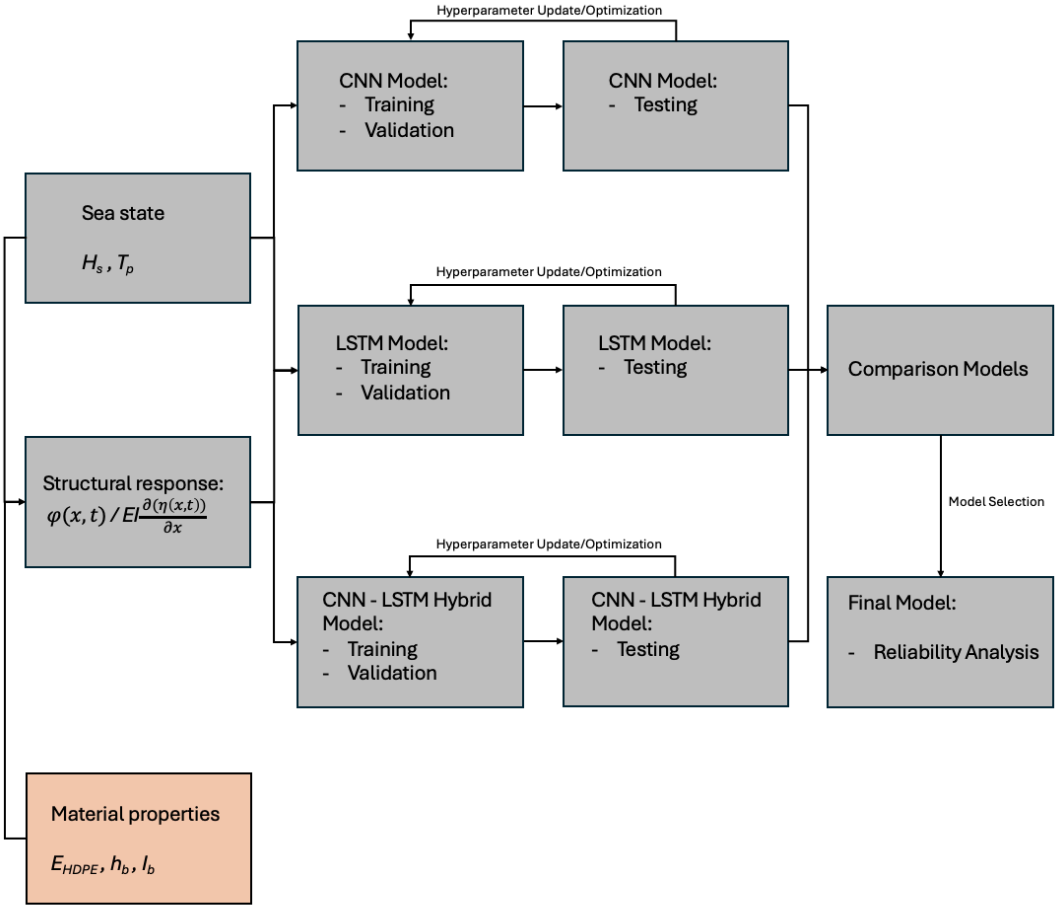
The research questions will be addressed using a conceptual framework consisting of seven sequential steps, from data acquisition to model comparison. The structural response data (target data) is already available and depends on the material properties of the structure and the sea state parameters (input data). The objective is to develop and compare multiple surrogate machine learning models for predicting the tilt response of VLFSs. These models include a CNN, an LSTM, and a CNN-LSTM hybrid model. The optimization of these models will involve tuning their hyperparameters.

Automated tuning methods, such as Bayesian Optimization, Grid Search, Hyperband, and Random Search, are often preferred because they rely on probabilistic models, systematic enumeration of possible combinations, or (partial) random sampling from the hyperparameter space. However, since this

thesis serves not only as a research project but also as a learning experience, an iterative trial-and-error approach is employed to optimize hyperparameters and gain a better understanding of their sensitivity.

Figure 1.14 presents the flowchart of the methodology, followed by a detailed description of each step.

Figure 1.14: Flowchart of the methodology of this research.



These seven sequential steps, involved in following the flowchart, together with their required necessities, are enumerated below.

1. **Data Acquisition**: Collect the material properties, sea state parameters, and structural response data of the OFPVs from simulations performed using the FE-FSI model. The majority of the data that is needed for this research is already available. Therefore, this step of the methodology will not be a part of the research.

2. **Data Preprocessing**: In this step, the available data will be analyzed, processed, and prepared for model development. If necessary, downsampling will be performed to reduce data size while preserving important features. The data will then be split into training, validation, and test sets, followed by data scaling and other preprocessing steps as required. Additionally, the data will be converted to tensors for compatibility with machine learning frameworks such as $TensorFlow$ and $Keras$.

   Required software, equipment, access, training, accreditation:

   - *DelftBlue*

- *Python*
- *Python packages (*$TensorFlow, Keras, Scikit-learn, Matplotlib$, **etc.**)

3. **Model Development**: In this step the deep-learning model will be created. This process will start with building the backbone architecture. The initial hyperparameters will be based on assumptions and engineering intuition.

   Required software, equipment, access, training, accreditation:

   - *Python*
   - *Python packages (*$TensorFlow, Keras, Scikit-learn, Matplotlib$, **etc.**)
   - *GitHub repositories*

4. **Model Training and Validating**: The model will be trained using the training set, while the validation set will be used to monitor performance during training. Performance metrics such as loss and accuracy will be tracked to guide hyperparameter adjustments later on.

   Required software, equipment, access, training, accreditation:

   - *GPU Resources*
   - *Python*
   - *Python packages (*$TensorFlow, Keras, Scikit-learn, Matplotlib$, **etc.**)

5. **Model Testing**: The model will be tested using the unseen test set to evaluate its generalization performance. Prediction errors will be analyzed, and a small sensitivity analysis will be conducted to assess the impact of key features and hyperparameters on the model's output.

   Required software, equipment, access, training, accreditation:

   - *Python*
   - *Python packages (*$TensorFlow, Keras, Scikit-learn, Matplotlib$, **etc.**)

6. **Model Adjustment**: To ensure the model's reliability, it is crucial to enhance its accuracy. Therefore, this step circles back to step 3 of this methodology, and involves revisiting the model architecture and hyperparameters to improve performance. Adjustments will be made iteratively, and the process will continue until no further significant improvements are observed.

   Required software, equipment, access, training, accreditation:

   - *GPU Resources*
   - *Python*
   - *Python packages (*$TensorFlow, Keras, Scikit-learn, Matplotlib$, **etc.**)

7. **Model Comparison and Error Quantification**: In this final step, the developed and optimized deep learning models will be compared based on their prediction accuracy, computational efficiency, and error metrics. The model with the best overall performance will be selected as the final surrogate model. Finally, an error quantification analysis will be conducted to assess the reliability of the selected model.

   Required software, equipment, access, training, accreditation:

   - *Python*
   - *Python packages (*$TensorFlow, Keras, Scikit-learn, Matplotlib$, **etc.**)

2

Dataset

This chapter introduces the dataset generated using the FE-FSI model, which will serve as the input and target data for the deep learning models developed in this research. Section 2.1 discusses the environmental conditions and structural properties used in the simulations. In Section 2.2, the generated simulation data is reviewed, and the process of converting simulation output into input data is explained, including any assumptions or simplifications made. Finally, Section 2.3 outlines the preprocessing steps required to prepare the input data for the deep learning models.

2.1. Environmental Setup

The analysis is based on environmental data from an offshore meteorological station in the North Sea (53°24'42.0"N, 6°11'57.0"E) for the year 2017. Hourly wind data are used to compute the significant wave height (H_s) and peak period (T_p), with these environmental properties constrained by the values listed in Table 2.1. The tilt (target data) is induced by waves generated using the JONSWAP (Joint North Sea Wave Project) spectrum and a random phase for each storm. It should be noted that combining all possible phase shifts with each storm would result in an unmanageably large input dataset, making model training infeasible. Incorporating uncertainty quantification or a probability density function for this parameter could improve the reliability of the input set. However, this enhancement lies beyond the scope of this research. Therefore, the spectral density input values are used with a single set of random phase shifts.

Table 2.1: Environmental Properties.

Property	Symbol	Value
Water depth	d	$30\,\mathrm{m}$
Significant wave height	H_s	$H_s \in [0.050\,\mathrm{m},\ 14.869\,\mathrm{m}]$
Peak wave period	T_p	$T_p \in [0.628\,\mathrm{s},\ 16.250\,\mathrm{s}]$

To simulate the required input and target data, the floating platform is hypothetically positioned in the North Sea with a constant water depth of $d = 30,\mathrm{m}$ and a South-facing orientation ($180°$). For this study, the platform length is also fixed at $L_{\text{tot}} = 100,\mathrm{m}$. Existing OFPVs typically consist of multiple interconnected floaters, which help to reduce overall structural stress and material usage. In the case of $L_{\text{tot}} = 100,\mathrm{m}$, the platform configuration can be defined as:

$$L_f = \begin{cases} 100\ m, & \text{if } N_f = 1 \\ 50\ m, & \text{if } N_f = 2 \\ 20\ m, & \text{if } N_f = 5 \\ 10\ m, & \text{if } N_f = 10 \\ 4\ m, & \text{if } N_f = 25 \\ 2\ m, & \text{if } N_f = 50 \end{cases} \quad \text{for } N_f \in \{1, 2, 5, 10, 25, 50\}$$

As this research focuses primarily on evaluating the performance of deep learning models under varying environmental conditions, the dataset corresponding to a single floater ($N_f = 1$) is used as input and target data.

The floating platform is constructed from high-density polyethylene (HDPE) with a standard thickness of $h_b = 0.2, \text{m}$. Although the weight of the photovoltaic (PV) panels is negligible compared to the floating structure, it is still included in the platform's mass. The selected PV module design has a length of $L_{pv} = 1.420, \text{m}$. A summary of the structural properties of the floating platform is provided in Table 2.2.

Table 2.2: Structural Properties of the floating platform.

Property	Symbol	Value
Floater thickness	h_b	$0.2\,\text{m}$
Cross-section Moment of Inertia per unit width	$I = \frac{1}{12}h_b^3$	$6.667 \times 10^{-4}\text{m}^3$
Young's Modulus	$E = E_{\text{HDPE}}$	$500\,\text{MPa}$
Mass per unit area	m	$192.956\,\text{kgm}^{-2}$
Characteristic hydro-elastic length	λ_c	$15.076\,\text{m}$
Total length floating platform	L_{tot}	$100\,\text{m}$
Length PV module	L_{pv}	$1.420\,\text{m}$

2.2. Simulation Data

In Section 1.5, various deep learning approaches relevant to this topic were introduced. Consequently, Section 1.8 focuses on investigating multiple deep learning architectures with increasing model complexity, aiming to develop a model that best predicts the response of VLFSs.

Since different deep learning models require different inputs and targets, the CNN model, which excels at recognizing spatial patterns, uses a JONSWAP spectrum as input. The significant wave height (H_s) and peak period (T_p) provided in Section 2.1 are used to compute the JONSWAP spectrum (Equation 2.1), which is subsequently converted into a one-dimensional array containing spectral density values across various frequencies.

$$S(f) = \alpha g^2 f^{-5} \exp\left(-1.25\left(\frac{f_p}{f}\right)^4\right)\gamma^{\exp\left(-\frac{[f-f_p]^2}{2\sigma(f)^2 f_p^2}\right)} \tag{2.1}$$

Where:

- $S(f)$: Spectral density at frequency f (in m^2/Hz).
- α: Empirical constant for the JONSWAP spectrum, calculated as:

$$\alpha = 0.076\left(\frac{H_s^2}{T_p^4}\right)$$

- g: Acceleration due to gravity ($g = 9.81\,\text{m/s}^2$).
- f: Frequency (in Hz).
- f_p: Peak frequency, calculated as $f_p = \dfrac{1}{T_p}$.
- γ: Peak enhancement factor (default value $\gamma = 3.3$).
- $\sigma(f)$: Spectral width parameter, defined as:

$$\sigma(f) = \begin{cases} 0.07, & \text{if } f \leq f_p \\ 0.09, & \text{if } f > f_p \end{cases}$$

Since the primary objective is to predict the tilt response of the VLFS over time, a time series of the tilt is the most appropriate form of target data. Accordingly, the FE-FSI model generates a time series output of the tilt at various points along the floating platform. Virtual sensors are placed at 2-meter intervals, starting from $x = 1,\text{m}$ and ending at $x = 99,\text{m}$. The position of the i-th sensor is given by:

$$x_i = i\,\text{m}, \quad \text{for } i = 1, 3, 5, \ldots, 99.$$

For this CNN model, the input data X_{CNN} and target data Y extracted from the simulations will have the shapes

$$X_{\text{CNN}} \in \mathbb{R}^{N_{\text{samples}} \times N_{\text{freq\_bins}} \times 1}$$

$$Y \in \mathbb{R}^{N_{\text{samples}} \times N_{\text{time\_steps}} \times N_{\text{sensors}}}$$

Here, $N_{\text{samples}} = 8760$ corresponds to the number of samples derived from hourly wind data over a full year (24×365); $N_{\text{freq\_bins}}$ represents the number of frequency bins used in the JONSWAP spectrum; $N_{\text{time\_steps}}$ denotes the number of time steps, with a default value of 14400 time steps per hour at intervals of $0.25\,\text{s}$; and $N_{\text{sensors}} = 50$ represents the number of virtual sensor locations along the floating platform.

It should be noted that when dealing with power output, the total number of samples N_{samples} is less than 8760, as solar energy production is not continuous throughout the day.

When using an LSTM model, which excels at capturing temporal patterns, it is crucial to adjust the input and target data to fully utilize the model's strengths. While spatial information about the floating platform may be relevant, the platform's response is assumed to be harmonic and primarily time-dependent. Therefore, both the input and output data should be time-dependent to effectively capture temporal patterns. The target data for the LSTM model can retain the same shape and form as that used for the CNN model, as it already represents sequential data, making it well-suited for LSTM models. However, the input parameters H_s and T_p lack temporal resolution, as they are summary statistics describing the overall sea state during a specific time interval, without detailed information on how the sea state evolves within that interval. To address this limitation, a different type of input is needed to provide time-resolved information about the sea state.

The wave elevation η (Equation 1.10), which varies with time, can serve as a suitable input. Assuming a time step $\Delta t = 0.25,\text{s}$, the wave elevation provides detailed information about the sea state at intervals of 0.25 seconds. Let $\eta_{-\infty}(t)$ and $\eta_0(t)$ denote the undisturbed far-field and undisturbed upstream wave elevations, respectively. With the target data Y unchanged, the input data X_{LSTM} will have the shape:

$$X_{\text{LSTM}} \in \mathbb{R}^{N_{\text{samples}} \times N_{\text{time\_steps}} \times 2}$$

This generalizes to the case where more locations of known wave elevation can be included:

$$X_{\text{LSTM}} \in \mathbb{R}^{N_{\text{samples}} \times N_{\text{time\_steps}} \times N_{\text{wave\_elevation\_locations}}}$$

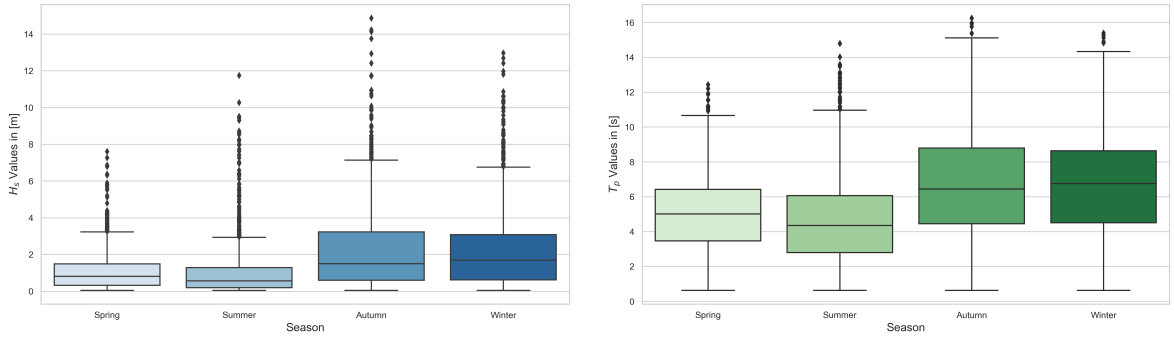
In a hybrid model that combines both CNN and LSTM architectures, the inputs for both models are utilized. The initial input, X_{CNN}, is processed by the CNN component of the hybrid model, after which the LSTM input is incorporated at a later stage. Further details regarding the architecture, as well as the input and output shapes, are provided in Section 3.3.

2.3. Data Preparation

With the input and target data defined, the next step is to prepare the data for model training. A key consideration regarding the target data is its size. A large target dataset increases computational load and model complexity, potentially reducing generalization capability. Downsampling the dataset can significantly reduce its size, improving training efficiency.

While the number of sensors (N_{sensor}) cannot be reduced, as monitoring the response along the structure is essential, the number of samples (N_{samples}) can be adjusted. Using 8,760 samples per structural configuration would generate an enormous dataset, likely exceeding available RAM/VRAM capacity. The number of selected samples may vary depending on the model and will be discussed in Chapter 3. Nonetheless, examining the sample distribution is necessary.

The magnitudes of significant wave height (H_s) and peak period (T_p) vary considerably, as shown by the bounds in Table 2.1. However, extreme values are rare in the North Sea. Figure 2.1 presents the seasonal distribution of H_s and T_p using boxplots. The figure shows that these bounds are outliers, with over 50% of H_s values below 2.0 m and more than 50% of T_p values below 8.5 s.



(a) **(b)**

Figure 2.1: Boxplots per season of a) hourly significant wave height values and b) hourly peak period values.

A more detailed visualization of the data is provided in Appendix A, where the distribution is shown per month. The majority of outliers occur during the autumn and winter months, except for June, which likely experienced a few stormy days. While outliers are often not representative of the overall dataset, they are relevant in this case, as such extreme weather conditions occasionally occur. When significant wave heights reach nearly 15 meters, the storms are considered extreme, often linked to cyclonic activity. It is highly unlikely that OFPVs operate under these conditions, particularly given the accompanying cloud coverage. Therefore, in terms of power output, the response of OFPVs under such conditions is irrelevant. However, these extreme events are crucial for assessing the structural integrity of the platform.

Due to the limited availability of information on maximum wave conditions during OFPV operations, a maximum significant wave height of 7.5 meters is selected. This threshold is based on the maximum operational wave height for offshore floating wind turbines [21] and constraints from existing OFPVs [25]. Consequently, all samples with $H_s > 7.5$ meters will be excluded from the dataset. Additionally, samples where the tilt φ remains below 0.1° at all time instances will be removed, as such values are negligible and can be treated as zero tilt.

These extreme values are rare and likely underrepresented in the dataset, which can hinder the model's ability to handle them effectively during training. Removing these outliers is expected to enhance the model's performance.

Since seasonal variations significantly influence the response of OFPVs, it is essential to ensure that the selected samples are distributed evenly across the entire year to maintain model generalizability. Figure A.3 in Appendix A presents a boxplot of the standard deviation of the tilt per month. This figure, consistent with other boxplots in the appendix, demonstrates a positive correlation between H_s, T_p, and the tilt φ. Consequently, larger input values correspond to larger output values.

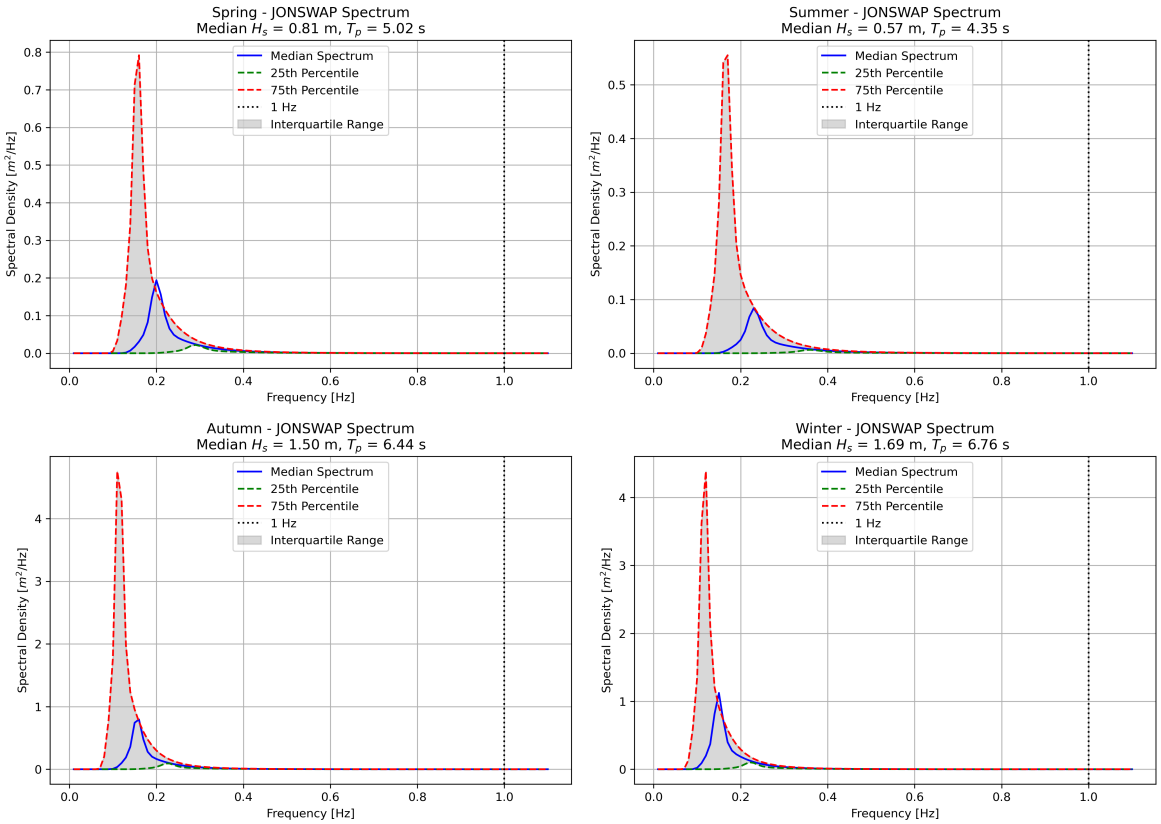


Figure 2.2: Quartile JONSWAP spectra for each season

Figure 2.2 presents the quartile JONSWAP spectra for each season, illustrating how the spectrum evolves with varying values of H_s and T_p. The median JONSWAP spectrum for each season is represented by the blue line. When the values are below the median, the spectrum aligns more closely with the green dotted line, while values above the median correspond to the red dotted line. In 75% of the cases, the spectrum shape lies within the grey area, representing the interquartile range.

Notably, the peak heights of the spectra vary significantly across seasons, indicating that average wave heights are larger in autumn and winter compared to spring and summer. Additionally, the spectra become more peaked with increasing wave heights and more flattened for smaller waves. This effect is particularly evident in the 5th percentile JONSWAP spectrum shown in Figure 2.3.

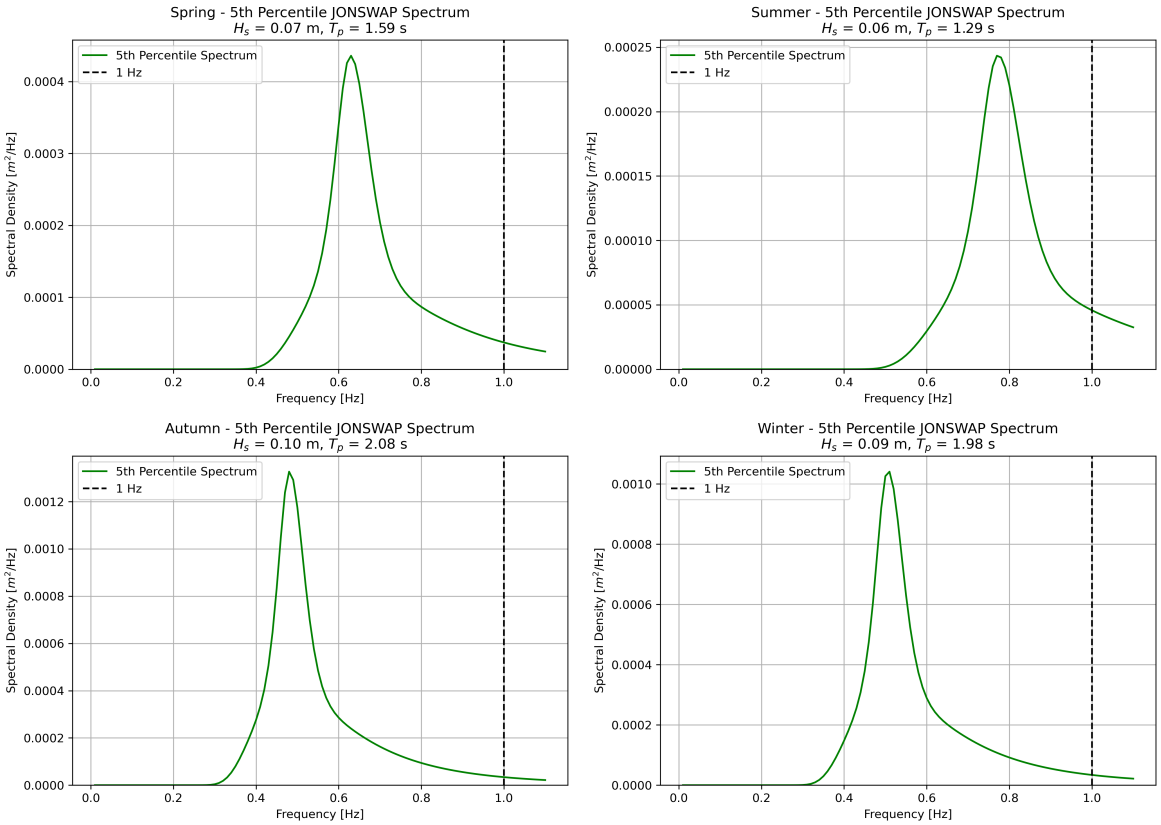


Figure 2.3: 5th percentile JONSWAP spectra for each season

In these graphs, less than 5% of the data exhibits a spectral density value $S(f)$ greater than 1×10^{-4}, m^2/Hz for frequencies above 1 Hz. Therefore, it is reasonable to assume a maximum frequency of 1 Hz, corresponding to a 1-second period. Each sample from the FE-FSI model consists of 14,400 time steps with a time interval of $\Delta t = 0.25$, s, resulting in a sampling frequency of 4 Hz. Consequently, each sample can be downsampled by a factor of 4, reducing its size while retaining nearly all relevant information.

Another essential preprocessing step is scaling. Large datasets often contain features with varying magnitudes, where those with larger values can dominate the learning process, distorting the model's performance. Scaling ensures that all features contribute uniformly, enhances optimization efficiency, and prevents numerical instability. The most commonly used scaling method is Min-Max scaling (normalization), which transforms features to a fixed range [0, 1]. However, since the target values (the tilt φ) can be both positive and negative, standard scaling (standardization) is preferred. Standard scaling centers the data around a mean μ of zero and scales it to have a standard deviation σ of one, as defined by:

$$X_{\text{scaled}} = \frac{X - \mu_X}{\sigma_X}. \tag{2.2}$$

$$Y_{\text{scaled}} = \frac{Y - \mu_Y}{\sigma_Y}. \tag{2.3}$$

Before standardization, it is essential to split the dataset into training (70%), validation (15%), and test (15%) sets. To maintain alignment between inputs and outputs after randomization, all data is split simultaneously. Given the wide dispersion of values in the dataset, it is crucial to ensure that each set contains a balanced representation across various value intervals. For instance, if the training

set contains a disproportionate number of small values, these values may be underrepresented in the validation and test sets, potentially degrading model performance.

Stratified sampling addresses this issue by dividing the data into bins—value intervals typically based on the mean of a sample. However, since the dataset contains both positive and negative values, using the mean directly is inappropriate. Instead, using the absolute value of the mean provides a more accurate measure of the sample's magnitude.

The number of bins (N_{bins}) should be selected based on the dataset, as it can significantly influence model performance. Since each model operates with its own dataset and architecture, N_{bins} must be treated as a variable. Furthermore, as N_{bins} can either enhance or degrade performance, it is considered a hyperparameter.

Once the data is split, the scaling parameters are determined using only the training set and applied uniformly to all sets. It is important to fit the scaling parameters exclusively on the training set to avoid data leakage.

With the data properly preprocessed, it is now ready for use in model training.

3

Model configuration

As discussed in Section 1.5, comparing different deep learning models is crucial. Each model responds differently to specific inputs and often requires tailored input formats for optimal performance. Nevertheless, deep learning models also share common characteristics. This chapter focuses on building and analyzing the different models, with an emphasis on their differences, which may explain why a particular model is best suited for this problem.

The first model, a CNN, is detailed in Section 3.1, covering its architecture, input and output types (and shapes), and relevant hyperparameters. Section 3.2 introduces a fundamentally different model: an LSTM. This section highlights key differences in architecture, input, and hyperparameters between a model designed for grid-like data and one intended for sequential data. Finally, Section 3.3 presents a hybrid model that integrates both CNN and LSTM architectures.

The objective of the models is to accurately predict the tilt response of a VLFS—a complex task due to the numerous parameters involved in fluid-structure interaction. Among other inputs, environmental conditions, the number of floaters, and the structural stiffness all significantly influence the response of the VLFS. The combination of these parameters can simulate an VLFS well, but it can result in large datasets and high model complexity. In this thesis, the input is limited to Case 1: *Base Case*, as shown in Table 3.1 and Figure 3.1. While the table and figures outline additional aspects for future investigation, this thesis focuses exclusively on the base case due to the primary research objective and time constraints. Further research could explore Case 2 and Case 3 in greater depth to gain a more comprehensive understanding of the VLFS response. The values of N_f in Case 2 and the values of both N_f and EI_f in Case 3 are left open, as they can be chosen arbitrarily depending on the researcher's focus. Please note that the length of the floater, L_f, can never be smaller than the length of the PV module, L_{pv}.

Table 3.1: Three distinct input scenarios that could significantly affect the tilt response of OFPVs.

Case	Variable/Constant Input
Case 1: *Base Case*	$H_s \in [0.050\,\mathrm{m},\ 14.869\,\mathrm{m}],\ T_p \in [0.628\,\mathrm{s},\ 16.250\,\mathrm{s}]$ $N_f = 1$ $EI_f = EI_{\mathrm{HDPE}} = 333.35\,\mathrm{kNm}$
Case 2: *Multiple Floaters*	$H_s \in [0.050\,\mathrm{m},\ 14.869\,\mathrm{m}],\ T_p \in [0.628\,\mathrm{s},\ 16.250\,\mathrm{s}]$ $N_f \in [..., ...]$ $EI_f = EI_{\mathrm{HDPE}} = 333.35\,\mathrm{kNm}$
Case 3: *Multiple Floaters + Varying Stiffness*	$H_s \in [0.050\,\mathrm{m},\ 14.869\,\mathrm{m}],\ T_p \in [0.628\,\mathrm{s},\ 16.250\,\mathrm{s}]$ $N_f \in [..., ...]$ $EI_f \in [..., ...]\ \mathrm{kNm}$

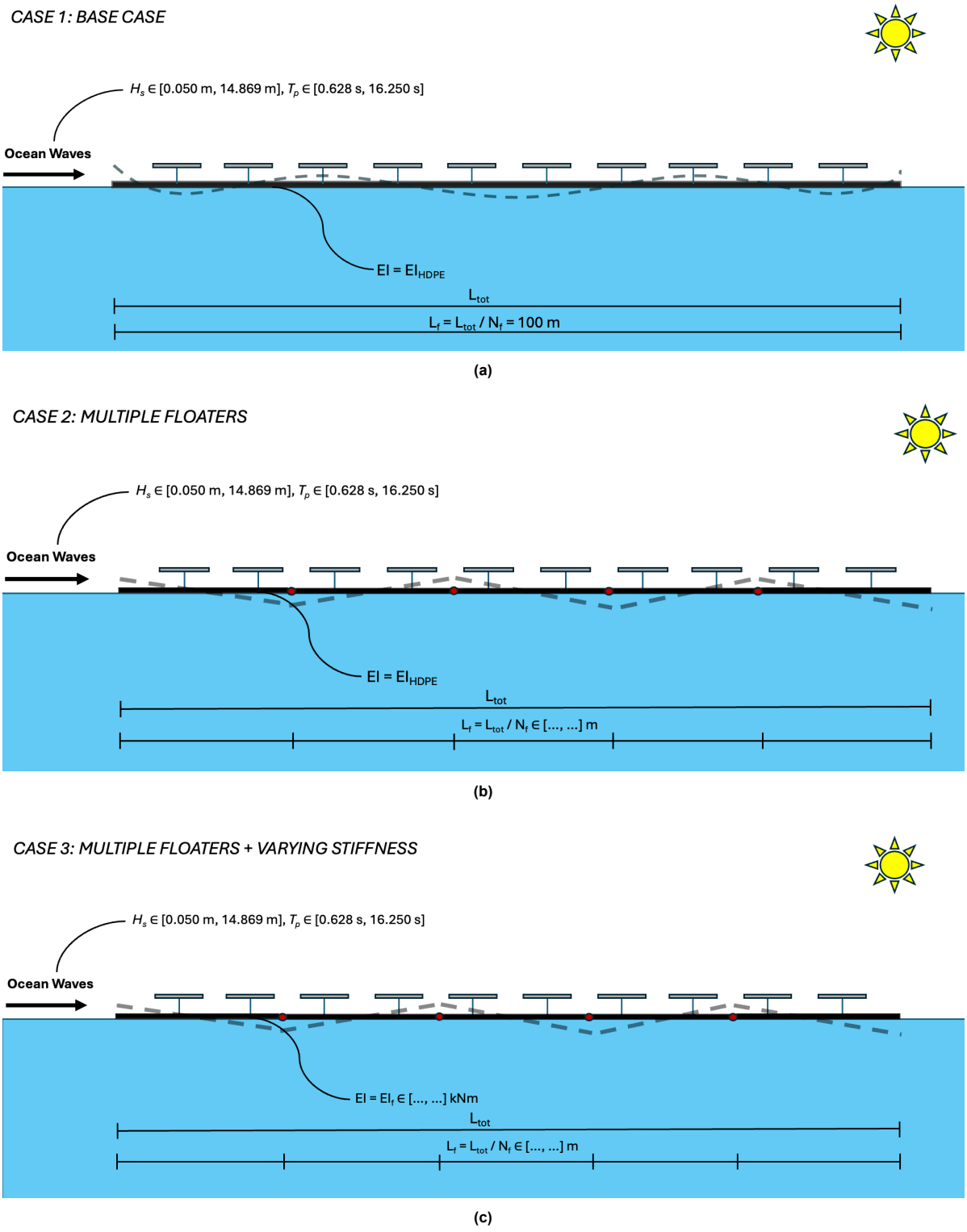


Figure 3.1: Schematic of an OFPV platform showing (a) one floater, (b) multiple floaters interconnected through hinges, and (c) multiple floaters with variable stiffness.

3.1. Convolutional Neural Network

This section outlines the configuration of the CNN model, focusing on three key criteria: input parameters, model architecture, and hyperparameters. Each aspect will be described and analyzed to evaluate how the model handles generalization and increasing data size.

3.1.1. Input parameters

After the preprocessing steps in Section 2.3, the data is prepared for use in the model. Since this research focuses exclusively on Case 1, only the environmental input varies, with changes in significant wave height (H_s) and peak period (T_p) resulting in different sample inputs. Each storm is associated with a specific H_s and T_p, leading to distinct JONSWAP spectra. The number of frequency bins, $N_{\text{freq\_bins}}$, is set to 300, starting from 0.01 Hz with a step size of 0.01 Hz. Consequently, each sample consists of an array of 300 spectral density values.

To balance computational efficiency with model complexity, N_{samples} is set to 3000, ensuring sufficient training samples across various value ranges for effective generalization. The resulting input shape for the CNN is $X_{\text{CNN}} = (3000, 300, 1)$.

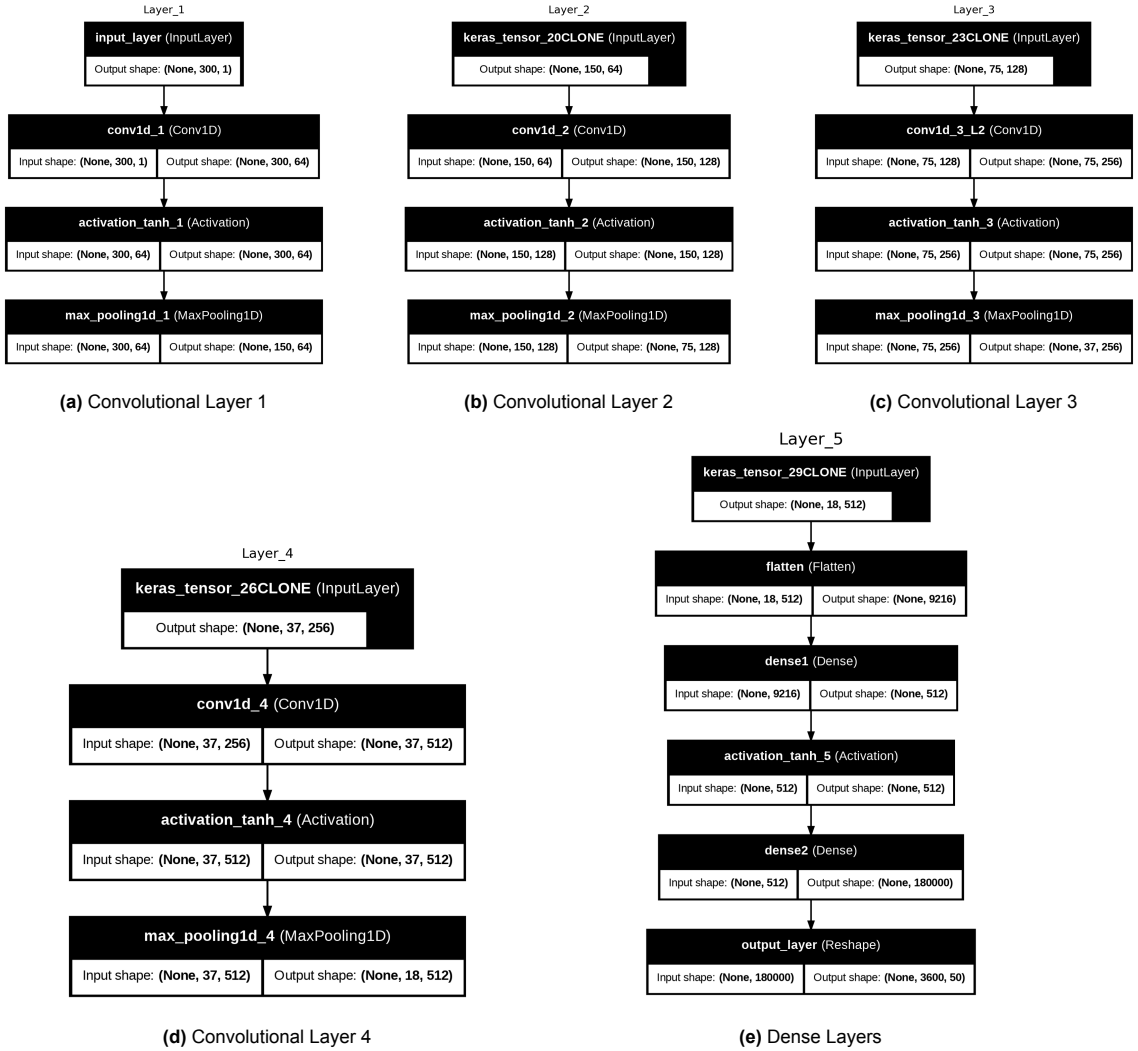
3.1.2. Model architecture & Hyperparameters

The architecture of the model can be described as a classic CNN with 4 convolutional layers and 2 dense layers, before reshaping it to the appropriate output format. A brief summary of all the layers is provided below. All the layer configurations and their corresponding shapes are illustrated in Figure 3.2.

1. **Input Layer**: Serves as an entry point in the model for the spectral density values derived from the JONSWAP spectrum.
2. **First Convolutional Block**
   - **Layers**:
     - *Conv1D*: 64 filters, kernel size 3, padding: 'same'
     - *Activation function*: Hyperbolic Tangent (tanh)
     - *MaxPooling1D*: Pool size 2
   - **Function**: Captures the initial features with a moderate number of filters, followed by a tanh activation function that introduces non-linearity into the network. The tanh activation function is particularly useful when dealing with both positive and negative values, ensuring that negative values are appropriately transformed and propagated through the network. The max pooling layer reduces the dimensions of the feature maps, thereby decreasing the number of parameters, ensuring translational invariance, and increasing computational efficiency. Since the output values are centered around zero, the maximum values provide more information than the average values, hence the choice of max pooling.
3. **Second Convolutional Block**
   - **Layers**:
     - *Conv1D*: 128 filters, kernel size 3, padding: 'same'
     - *Activation function*: Hyperbolic Tangent (tanh)
     - *MaxPooling1D*: Pool size 2
   - **Function**: With an increasing number of filters, this layer captures more complex patterns. The same activation and pooling strategies are applied to enhance the model's feature extraction capabilities.
4. **Third Convolutional Block**
   - **Layers**:
     - *Conv1D*: 256 filters, kernel size 3, padding: 'same', L2 Regularizer: 0.001
     - *Activation function*: Hyperbolic Tangent (tanh)
     - *MaxPooling1D*: Pool size 2
   - **Function**: Further increases the number of filters and feature extraction capabilities. A weight decay algorithm (L2 Regularizer) is introduced, to add a penalty to the gradient and reduce overfitting, leading to an enhanced model generalization.
5. **Fourth Convolutional Block**

- **Layers**:
  - *Conv1D*: 512 filters, kernel size 3, padding: 'same'
  - *Activation function*: Hyperbolic Tangent (tanh)
  - *MaxPooling1D*: Pool size 2
- **Function**: This last convolutional block maximizes the network's capacity to learn high-level features. The large number of filters enhances the model's ability to make accurate predictions.

6. **Flattening Layer**: This layer serves as a preparatory step for the upcoming fully connected layer. It reduces the multi-dimensional output of the convolutional blocks to a single vector.
7. **First Dense Layer**
   - **Layers**:
     - *Dense*: 512 units
     - *Activation function*: Hyperbolic Tangent (tanh)
   - **Function**: This first dense layer interprets the features extracted by the convolutional layers and learns complex combinations and interactions between those features. While convolutional layers are useful for detecting local patterns, dense layers are effective for capturing global patterns. The number of units is chosen to match the number of filters in the last convolutional layer, balancing model size and complexity.
8. **Second Dense (Output) Layer**
   - **Layers**:
     - *Dense*: 180,000 units
   - **Function**: Produces the final prediction output, which is then reshaped to the desired temporal and spatial dimensions of $(3600, 50)$. This represents the tilt at 50 locations along the floating platform for every second in an hour. This layer accounts for more than 90% of the total trainable parameters, making it the most critical layer. Note that no activation function is used in the output layer as it constrains the output to a certain range, which is undesirable.



(a) Convolutional Layer 1 **(b)** Convolutional Layer 2 **(c)** Convolutional Layer 3

(d) Convolutional Layer 4 **(e)** Dense Layers

Figure 3.2: Layer configuration of the CNN model architecture. Each black box corresponds to the name of the layer, with its corresponding input and output shape noted in the white box. Note that the output of one layer serves as the input for the subsequent layer.

The training of this architecture is fully supervised and conducted in batches using Mini-Batch Stochastic Gradient Descent with the Adam optimizer. The number of convolutional and dense layers is selected to optimize performance while maintaining a manageable model size to avoid Out-Of-Memory (OOM) errors. The data is stratified into six bins ($N_{\text{bins}} = 6$). A batch size of 8 is chosen to balance computational efficiency and memory consumption. The Adam optimizer employs the Huber loss function (Equation 3.1), which combines mean squared error and mean absolute error. It behaves quadratically for small errors and linearly for large errors, making it less sensitive to outliers—a crucial feature given the significant presence of outliers in the dataset, as shown in Figure 2.1.

During training, the learning rate is managed using the ReduceLROnPlateau scheduler, starting at 0.001. This scheduler reduces the learning rate by half if the validation loss does not improve for three consecutive epochs. The maximum number of epochs is set to 30, with early stopping applied after 10 epochs to prevent overfitting. An overview of all the fixed training parameters and hyperparameters used in this architecture is provided in Table 3.2.

$$L_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \leq \delta, \\ \delta \cdot \left(|y - \hat{y}| - \frac{1}{2}\delta\right) & \text{otherwise.} \end{cases} \tag{3.1}$$

Where:

- y is the true value.
- $\hat{y}$ is the predicted value.
- δ is a threshold parameter that determines the point where the loss function transitions from quadratic to linear.

Table 3.2: Fixed parameter and hyperparameter specifications for the CNN architecture.

(Hyper)parameter	Value
Loss function	Huber (δ = 2.0)
Optimizer	Adam
Learning rate (ReduceLROnPlateau)	0.001 (start value)
Nr. of Epochs	30
Nr. of Conv. Layers	4
Nr. of Dense Layers	2
Nr. of Input Channels	1
Activation Function	Tanh
N_{bins}	6
Early Stopping (Patience)	10
L2 Regularizer	0.001 (3$^{\text{rd}}$ layer)
Kernel Size	3
Stride	1
Padding	'same'
Number of learnable parameters	97,576,352
Model size	372.22 MB
Training time	220 s
Evaluation time	2.31 s
Memory usage (RAM)	17.71 GB
GPU utilization	22%
GPU memory usage (VRAM)	12.57 GB

3.2. Long Short-Term Memory

This section discusses the configuration of the Long Short-Term Memory (LSTM) model, which is particularly suited for sequential data. Consequently, both the model architecture and input data differ from those of the CNN model. Subsection 3.2.1 covers the various inputs used for the LSTM model, while Subsection 3.2.2 introduces the model architecture and its corresponding hyperparameters.

3.2.1. Input parameters

In the previous chapter, two wave elevations, $\eta_{-\infty}(t)$ and $\eta_0(t)$, were introduced. Both parameters contribute to the LSTM input X_{LSTM}. Since the target data is downsampled to a 1-second time interval, the wave elevations are similarly downsampled, resulting in an input shape of $(3000, 3600, 2)$.

Although LSTMs are designed to handle long sequences, excessively long sequences can complicate the training process and increase computational cost. To address this, a sliding window approach is employed to segment the data into manageable, fixed-length windows, enabling the model to focus on relevant time horizons. Given that wind-induced waves typically have peak periods around 16 seconds, using the entire 3600-second sequence is unnecessary. Since the dataset is filtered, a peak period exceeding 12 seconds is unlikely. Using this peak period as a reference, the wave number can be calculated via the dispersion relation:

$$\omega^2 = gk\tanh(kd) \implies (\frac{2\pi}{12})^2 = 9.81 \cdot k\tanh(30 \cdot k). \tag{3.2}$$

Using the wave number and peak period, the phase speed can be calculated, resulting in c = 14.75

$\mathrm{m\,s^{-1}}$. Consequently, a wave will traverse the 100-meter platform in under 7 seconds. This indicates that a sliding window of at least 7 seconds is sufficient, making the use of the entire 3600-second sequence unnecessary and inefficient. However, reducing the window size too much increases computational complexity. Therefore, a well-balanced window size must be selected.

In the sliding window approach, a window W of length W_L is moved across the entire time series with a step size W_S. Each window is processed independently, and the outputs are reshaped back into their original chronological order. The sliding window is defined as:

$$X_{\text{windows}} = \{X_{\text{original}}[n, t : t + W_L, :] \mid n = 0, \ldots, N_W - 1;\ t = 0, W_S, 2W_S, \ldots, T - W_L\}, \tag{3.3}$$

$$y_{\text{windows}} = \{y_{\text{original}}[n, t : t + W_L, :] \mid n = 0, \ldots, N_W - 1;\ t = 0, W_S, 2W_S, \ldots, T - W_L\}. \tag{3.4}$$

Where T is the total time series length, and the number of windows N_W is defined as $(\frac{T-W_L}{W_S} + 1) \times N_{\text{samples}}$. The input will now have a shape $(213000, 100, 2)$, with $W_L = 200s$, and $W_s = 190s$. The output will be of the shape $(56685, 200, 50)$, before reshaping it to $(3000, 3600, 50)$.

In this case the step size is smaller than the window size, resulting in overlapping windows. This overlap ensures that every last bit of the window also influences the next window. The values of $W_L = 200s$, and $W_s = 190s$ are chosen to create overlapping windows, and to balance computational efficiency with the minimum window size, as mentioned before. To ensure the overlapping windows don't affect the output shape, the average of the overlapping values is taken.

3.2.2. Model architecture & Hyperparameters

The architecture of the LSTM model is an expansion on the classic RNN model, with the aim at mitigating the vanishing gradient problem in long-term dependencies. The first stage in the LSTM architecture is the forget gate, This is defined as f_t in Figure 3.3, where:

- f_t : Forget gate vector at time step t .
- σ : Sigmoid activation function, which outputs values between 0 and 1.
- W_f : Weight matrix for the forget gate.
- h_{t-1} : Hidden state from the previous time step.
- x_t : Current input vector.
- b_f : Bias vector for the forget gate.

In this part of the architecture the model determines which elements of the cell state is relevant based on the previous time step and the new input data. This stage can be seen as a filter stage, where the filter determines what information, and how much, is being discarded or retained based on a value provided by the sigmoid activation function.

The following stage is the input gate, where:

- i_t : Input gate vector at time step t .
- $\tilde{C}_t$: Candidate cell state vector (proposes new information).
- W_i, W_C : Weight matrices for the input gate and candidate cell state.
- b_i, b_C : Bias vectors for the input gate and candidate cell state.

This gate act as a gatekeeper for new information. It determines to what extent new information is added from the candidate cell state, which creates new potential candidate values for the cell state. The candidate values are proposed through a tanh activation function.

The final stage is the output gate. The output gate determines what the output should be as the hidden state for the current time step. This hidden state is used for predictions and passed to the next time step. This final stage is also shown in Figure 3.3, where:

- o_t : Output gate vector at time step t .
- h_t : Hidden state (output) at time step t .
- W_o : Weight matrix for the output gate.
- b_o : Bias vector for the output gate.

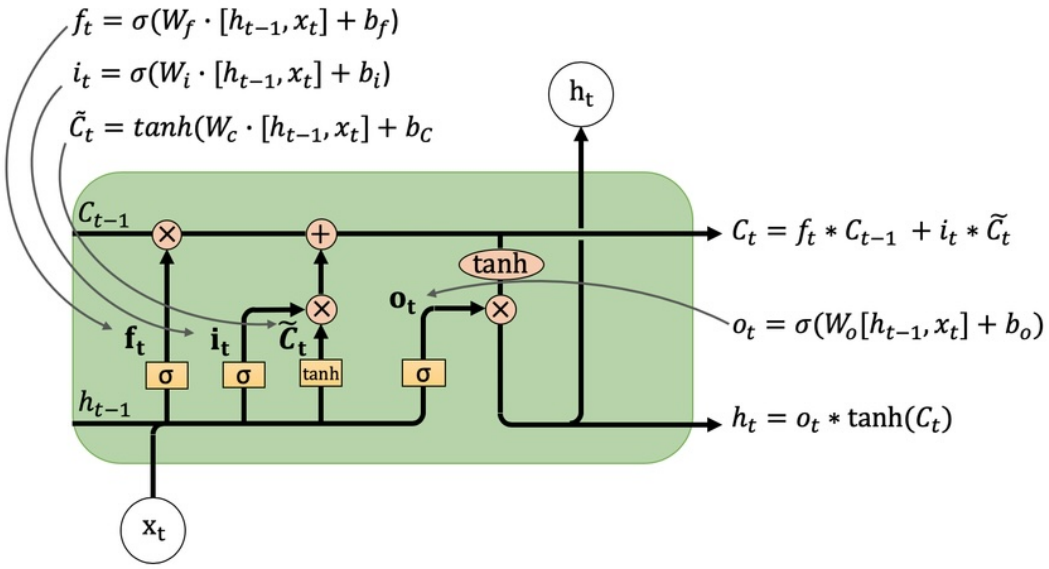


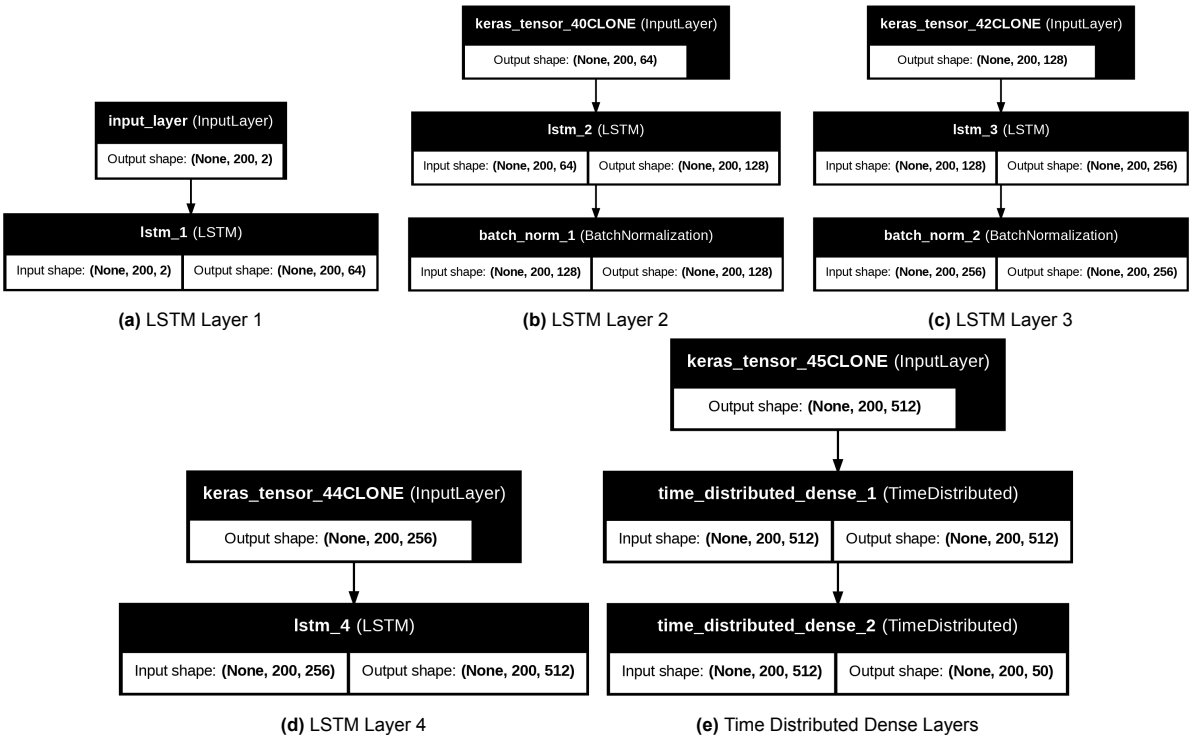
Figure 3.3: An LSTM memory cell, including a forget gate, input gate, and output gate [18].

The full sequence of these three stages complete one time step. The selected information will then be stored and transferred to the next time step, until all the time steps in a window are predicted. Increasing the number of LSTM layers, increases the model's complexity and number of learnable parameters. Therefore multiple LSTM layers are selected, with each an increasing number of units. A short summary of the model's architecture and layers is given below. The layer configurations, and corresponding shapes are depicted in Figure 3.4.

1. **Input Layer**: Serves as an entry point in the model for the wave height values $\eta_{-\infty}(t)$ and $\eta_0(t)$, which are the only two features in this model.
2. **First LSTM Block**
   - **Layers**:
     - *LSTM*: 64 units
     - *Activation function*: Hyperbolic Tangent (tanh)
     - *Recurrent Activation function*: Sigmoid
   - **Function**: Captures the temporal dependencies of the input data and maintains a hidden state of 64 dimensions. This layer learns to retain important information from the input sequence and pass it forward to the subsequent layers.
3. **Second LSTM Block**
   - **Layers**:
     - *LSTM*: 128 units
     - *Activation function*: Hyperbolic Tangent (tanh)
     - *Recurrent Activation function*: Sigmoid
     - *Batch Normalization*

- **Function**: Further processes the temporal information extracted by the first LSTM layer and increases the model's capacity to learn with 128 hidden units. Includes a batch normalization to stabilize and accelerate the training process.

4. **Third LSTM Block**
   - **Layers**:
     - *LSTM*: 256 units
     - *Activation function*: Hyperbolic Tangent (tanh)
     - *Recurrent Activation function*: Sigmoid
     - *Batch Normalization*
   - **Function**: Further deepens the network to capture more complex sequences. With 256 units the model's capacity to learn also further increases. Again a batch normalization is added to normalize the output and ensures subsequent layers have consistent statistical properties.
5. **Fourth LSTM Block**
   - **Layers**:
     - *LSTM*: 512 units
     - *Activation function*: Hyperbolic Tangent (tanh)
     - *Recurrent Activation function*: Sigmoid
   - **Function**: Last sequential layer. Adds significant depth to the model, allowing it to capture very complex temporal patterns.
6. **First Time Distributed Dense Layer**
   - **Layers**:
     - *Dense*: 512 units
     - *Activation function*: Hyperbolic Tangent (tanh)
   - **Function**: Applies a fully connected layer at each time step individually. It facilitates the modeling of time-dependent patterns before the production of the final output.
7. **Second Time Distributed Dense Layer**
   - **Layers**:
     - *Dense*: 50 units
   - **Function**: Maps the transformed features from the previous dense layer to the desired output dimensionality (e.g., 50), representing the tilt at 50 locations along the floating platform.



(a) LSTM Layer 1 **(b)** LSTM Layer 2 **(c)** LSTM Layer 3

(d) LSTM Layer 4 **(e)** Time Distributed Dense Layers

Figure 3.4: Layer configuration of the LSTM model architecture. Each black box corresponds to the name of the layer, with its corresponding input and output shape noted in the white box. Note that the output of one layer serves as the input for the subsequent layer.

The training of this LSTM model is also fully supervised. The window size is 200 seconds, with a step size of 190 seconds. This ensures a consistent 10-second overlap, adhering to the minimum overlap time discussed earlier. The Adam optimizer is used for gradient descent, with a Huber loss function where the parameter δ is set to 1.0. The learning rate scheduler ReduceLROnPlateau is employed with an initial learning rate of 0.001. The stratified sampling parameter N_{bins} is set to 8. A batch size of 64 is chosen to balance computational efficiency and model generalization. Early stopping is set at 15 epochs, as overfitting is unlikely due to the relatively limited input data compared to the CNN model. All fixed training parameters and hyperparameters used in this architecture are summarized in Table 3.3.

3.3. Convolutional Neural Network - LSTM hybrid

In this final model configuration section, the two previously discussed models are combined. The spatial pattern recognition capabilities of the CNN are integrated with the sequential processing strengths of the LSTM. The model architecture largely follows a sequential combination of the two models, utilizing both inputs. Subsection 3.3.1 analyzes the model's inputs and specifies where the two inputs are incorporated. Subsection 3.2.2 presents the combined architecture and outlines the modifications made by merging the two models.

3.3.1. Input parameters

Combining the CNN and LSTM models requires merging their respective inputs, which presents a challenge since the CNN input is time-independent while the LSTM input is time-dependent. The same inputs, X_{CNN} and X_{LSTM}, are used as in the previous models. The input X_{LSTM} has a shape of $(3000, 3600, 50)$, and as discussed in Section 3.2, including 3600 time steps per epoch is computationally infeasible and may result in Out-Of-Memory errors. Therefore, a windowing approach is again employed for the hybrid CNN-LSTM model.

Ideally, windowing would be applied only to X_{LSTM}, as it is primarily needed for the LSTM part. However, the machine learning package $TensorFlow$ requires that multiple inputs have the same length along

Table 3.3: Fixed parameter and hyperparameter specifications for the LSTM architecture.

(Hyper)parameter	Value
Loss function	Huber (δ = 1.0)
Optimizer	Adam
Learning rate (ReduceLROnPlateau)	0.001 (start value)
Nr. of Epochs	30
Nr. of LSTM Layers	4
Nr. of Dense Layers (Time Distributed)	2
Nr. of Input Channels	1
Activation Function	Tanh
Recurrent Activation Function	Sigmoid
N_{bins}	8
Early Stopping (Patience)	15
Number of learnable parameters	2,374,962
Model size	9.06 MB
Training time	3,942 s
Evaluation time	6.20 s
Memory usage (RAM)	22.02 GB
GPU utilization	85%
GPU memory usage (VRAM)	8.42 GB

the first dimension (number of samples). Applying windowing solely to X_{LSTM} results in a mismatch, as its first dimension becomes larger than that of X_{CNN} due to the reduction of the second dimension to length W_L. Consequently, windowing must be applied to both inputs.

The second dimensions of X_{CNN} and X_{LSTM} also differ: X_{CNN} has a second dimension of length 300, representing spectral density values, while X_{LSTM} has a length of 3600, representing time steps. To reconcile this discrepancy, X_{CNN} is concatenated 12 times along its second dimension, ensuring that the correct spectral density values correspond to the appropriate samples. The window size W_L and step size W_S are both set to 300, matching the number of spectral density values per sample. This ensures that the entire JONSWAP spectrum influences each window. The trade-off is the absence of overlapping windows, meaning each time step does not influence subsequent time steps.

Following Equation 3.3 from the previous section, the resulting shapes of X_{CNN} and X_{LSTM} are $(36000, 300, 1)$ and $(36000, 300, 2)$, respectively. After windowing, the data is scaled to ensure that standardization occurs per window. Note that the first dimension of these tensors represents the total number of samples, which are subsequently divided into training, validation, and test sets as described in Section 2.3.

3.3.2. Model architecture & Hyperparameters

The hybrid model combines a CNN and a LSTM architecture in series. The CNN part captures spatial patterns from the visual representation of the spectral density values. Once processed, the output is passed to the LSTM part, where it is concatenated with the wave elevation input X_{LSTM} along the feature axis. This combined tensor is then processed as sequential data per window to capture temporal dependencies.

The CNN part comprises four convolutional layers, each followed by a pooling layer. The resulting tensor is flattened and passed through two dense layers. The output vector is reshaped to match the window size along the second dimension and concatenated along the third dimension with the wave elevation input. This combined tensor is then passed through two LSTM layers, followed by a time-distributed dense layer. The final output represents the tilt of the floating platform at 50 locations over time, with a shape of $(36000, 300, 50)$. A summary of the model's architecture is provided below, with the layer configurations illustrated in Figure 3.5.

1. **First Input Layer**: Serves as an entry point in the model for the spectral density values derived from the JONSWAP spectrum.

2. **First Convolutional Block**
    - **Layers**:
        - *Conv1D*: 64 filters, kernel size 3, padding: 'same'
        - *Activation function*: Hyperbolic Tangent (tanh)
        - *MaxPooling1D*: Pool size 2
    - **Function**: This block performs initial feature extraction by applying convolutional filters to capture local patterns in the spectral density data. The hyperbolic tangent activation introduces non-linearity, enabling the model to learn complex representations. The 'MaxPooling1D' layer downsamples the feature maps, reducing their temporal dimension by half and retaining the most important features, which helps in minimizing computational complexity and controlling overfitting.
3. **Second Convolutional Block**
    - **Layers**:
        - *Conv1D*: 128 filters, kernel size 3, padding: 'same'
        - *Activation function*: Hyperbolic Tangent (tanh)
        - *MaxPooling1D*: Pool size 2
    - **Function**: This block deepens the feature extraction process by increasing the number of convolutional filters to 128, allowing the model to capture more complex and abstract patterns in the data.
4. **Third Convolutional Block**
    - **Layers**:
        - *Conv1D*: 256 filters, kernel size 3, padding: 'same'
        - *Activation function*: Hyperbolic Tangent (tanh)
        - *MaxPooling1D*: Pool size 2
    - **Function**: This advanced convolutional block further amplifies the feature extraction capability by utilizing 256 filters, enabling the capture of highly intricate and abstract features from the spectral density inputs.
5. **Flattening Layer**: This layer serves as a preparatory step for the upcoming fully connected layer. It reduces the multi-dimensional output of the convolutional blocks to a single vector, allowing the dense layers to process the extracted features effectively.
6. **First Dense Layer**
    - **Layers**:
        - *Dense*: 512 units
        - *Activation function*: Hyperbolic Tangent (tanh)
    - **Function**: This dense layer transforms the flattened feature vector into a higher-dimensional space with 512 units, enabling the model to learn complex, non-linear relationships within the data. The 'tanh' activation introduces non-linearity, facilitating the modeling of intricate patterns necessary for accurate tilt predictions.
7. **Second Dense Layer**
    - **Layers**:
        - *Dense*: 76,800 units
    - **Function**: This expansive dense layer further transforms the feature representations into a very high-dimensional space (76,800 units), effectively preparing the data for temporal modeling by the LSTM layers. This large number of units ensures that the rich feature set is adequately captured and retained for subsequent sequential processing, allowing the LSTM

layers to access a comprehensive set of features for accurate tilt prediction. The number of units is chosen such that it can be reshaped in the following layer to the desired dimensions.

8. **Reshaping Layer**: Serves as a layer that reshapes the output of the previous layer to the desired dimensions: window size W_L in the second dimension, and number of features $N_{\text{features}}^{\text{LSTM}}$ in the third dimension. Note that $N_{\text{features}}^{\text{LSTM}}$ is arbitrarily chosen to be 256, and has nothing to do with the number of units in the following (first) LSTM block.
9. **Second Input Layer**: Serves as an entry point in the model for the wave height values $\eta_{-\infty}(t)$ and $\eta_0(t)$, which is only used in the LSTM part of the hybrid model.
10. **Concatenation Layer**: This layer concatenates the output of the CNN part with the second input layer over the third dimension.
11. **First LSTM Block**
    - **Layers**:
      - *LSTM*: 256 units
      - *Activation function*: Hyperbolic Tangent (tanh)
      - *Recurrent Activation function*: Sigmoid
      - *Batch Normalization*
    - **Function**: This LSTM layer is responsible for capturing short-term temporal dependencies in the combined feature set from the dense layers and the wave height inputs. With 256 units, it processes the sequential data, maintaining a memory of previous time steps through its gating mechanisms (as described in Section 3.2). The 'Batch Normalization' layer normalizes the activations, accelerating training and improving model stability by reducing internal covariate shift.
12. **Second LSTM Block**
    - **Layers**:
      - *LSTM*: 512 units
      - *Activation function*: Hyperbolic Tangent (tanh)
      - *Recurrent Activation function*: Sigmoid
    - **Function**: This advanced LSTM layer deepens the model's capacity to capture long-term temporal dependencies by utilizing 512 units. It processes the output from the first LSTM block, further refining the temporal representations and enabling the model to understand more extended sequences of wave height variations.
13. **Time Distributed Dense Layer**
    - **Layers**:
      - *Dense*: 50 units
    - **Function**: This layer applies a dense (fully connected) transformation to each time step individually, mapping the rich temporal features learned by the LSTM layers to the desired output dimensionality. With 50 units, it generates the tilt predictions for 50 distinct locations along the floating platform. The 'TimeDistributed' wrapper ensures that the same dense layer is applied to every time step in the sequence, maintaining consistency and enabling the model to produce a sequence of predictions corresponding to the required sequence length.

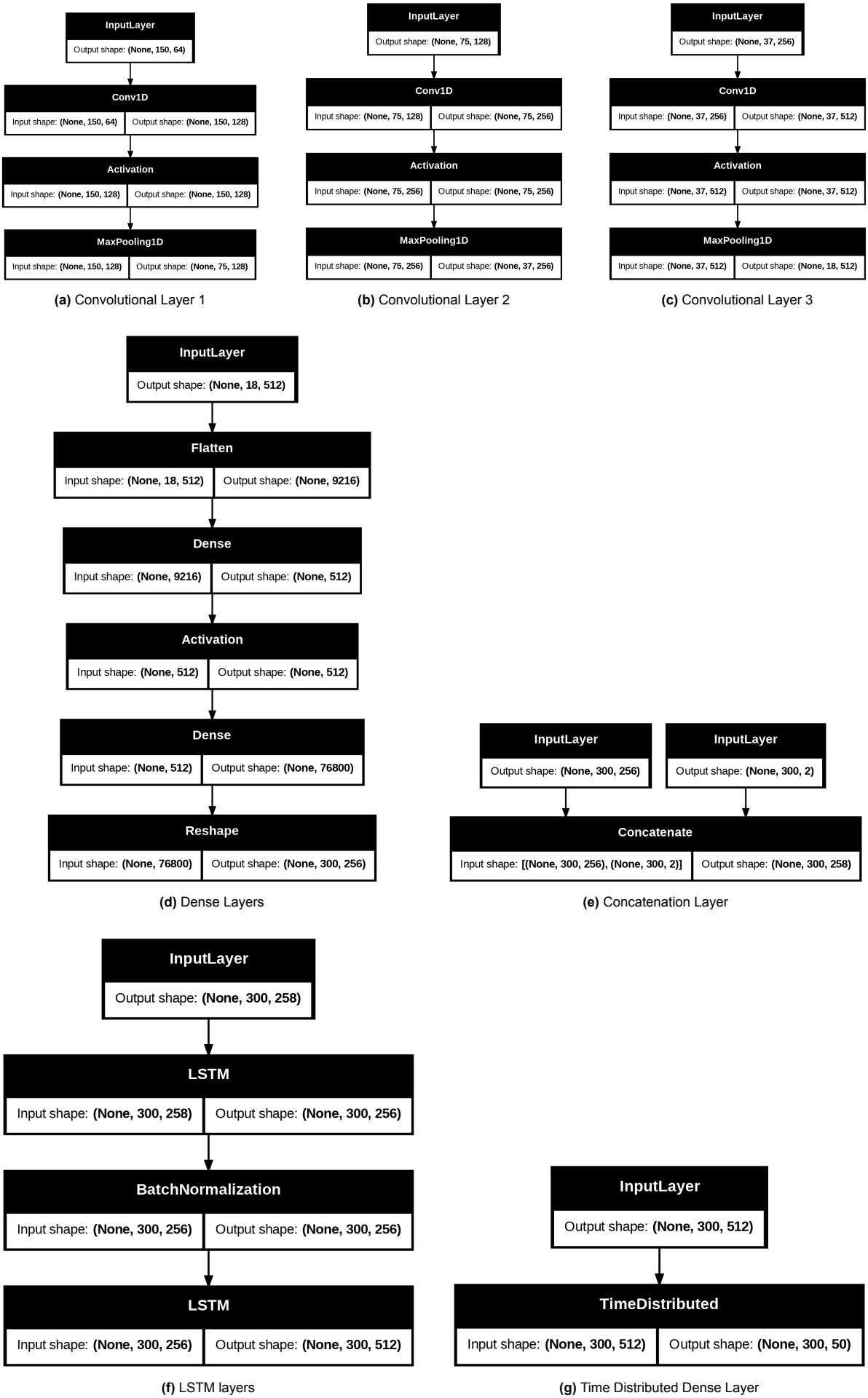


Figure 3.5: Layer configuration of the CNN-LSTM hybrid model architecture. Each black box corresponds to the name of the layer, with its corresponding input and output shape noted in the white box. Note that the output of one layer serves as the input for the subsequent layer, except for the concatenation layer where an additional input is included. The layers are in order from a to g.

The training of the hybrid model is supervised, with a batch size of 36. This value is selected to balance noise in gradient estimates while maintaining training stability. Additionally, the batch size is a multiple of 12, ensuring that each batch contains an integer number of storms during training. With a window size of 300 and a step size of 300, one storm is processed every 12 batches. The Adam optimizer is utilized for gradient descent, employing a Huber loss function with δ set to 1.0. The optimizer uses the learning rate scheduler ReduceLROnPlateau from $TensorFlow$, starting with an initial learning rate of 0.001. The stratified sampling parameter N_{bins} is set to 5.

Compared to the previous two models, the number of training epochs is reduced to 20 to improve computational efficiency. Due to the increased complexity of the hybrid model and its higher susceptibility to overfitting, early stopping is set at 5 epochs. All the fixed parameters and hyperparameters used in this hybrid architecture are summarized in Table 3.4.

Table 3.4: Fixed parameter and hyperparameter specifications for the CNN-LSTM hybrid architecture.

(Hyper)parameter	**Value**
Loss function	Huber (δ = 1.0)
Optimizer	Adam
Learning rate (ReduceLROnPlateau)	0.001 (start value)
Nr. of Epochs	20
Nr. of Conv. Layers	3
Nr. of LSTM Layers	2
Nr. of Dense Layers	2
Nr. of Dense Layers (Time Distributed)	1
Nr. of Input Layers	2
Activation Function	Tanh
Recurrent Activation Function	Sigmoid
N_{bins}	6
Kernel Size	3
Stride	1
Padding	‘same’
Early Stopping (Patience)	10
Number of learnable parameters	46,763,186
Model size	178.39 MB
Training time	2690 s
Evaluation time	6.46 s
Memory usage (RAM)	23.21 GB
GPU utilization	88%
GPU memory usage (VRAM)	11.54 GB

4

Model Performance

In Chapter 3, multiple models were developed, each with distinct architectures and input parameters. These models were trained and validated using their respective training and validation datasets. This chapter evaluates the models' performance on the test set. First, the results of each model will be analyzed to assess their accuracy in both spatial and temporal dimensions. Since differences in model performance may not always be visually apparent in graphs, various evaluation metrics will be employed to aid in the analysis. Finally, an overview of all models' performances will be presented, highlighting their differences and similarities, and concluding with a comparative analysis to determine the model best suited for predicting the response of a very large floating structure.

Although a small error in predictions may seem favorable, the primary concern lies not in the accuracy of tilt angle predictions themselves but in how these errors influence the power output predictions of OFPV modules. Therefore, before assessing the performance of the models, it is essential to understand how errors in tilt angle predictions translate into errors in power output.

In Section 1.2, the coupling between power output and tilt angle of PV modules is presented. Each irradiance term is influenced by the cosine of the tilt angle. Since cosine functions are continuous and differentiable, a small perturbation in the tilt angle leads to a small change in power output. This relationship is confirmed by applying a first-order Taylor expansion with a small perturbation $\Delta\varphi$ around a reference tilt angle φ_0. The first-order Taylor expansion and a numerical example are illustrated in Appendix B. These calculations demonstrate that a small error in tilt angle prediction results in only a minor error in power output.

4.1. Convolutional Neural Network Performance

This section analyzes the performance of the CNN model. The predicted values ($\hat{y}$) will first be inverse-scaled to their original scale and then compared with the actual values (y) from the test set. Given that the response variable (tilt φ) must be predicted across the entire platform and over time, the results will be examined both temporally and spatially.

4.1.1. Spatial and Temporal Tilt Analysis

Beginning with the spatial predictions, the learning process appears steady. Referring to the learning curve in Figure 4.1, both the training and validation losses gradually decrease towards zero. The plot shows no indications of underfitting or overfitting. Additionally, the magnitude of the loss on the y-axis suggests a reasonable performance.

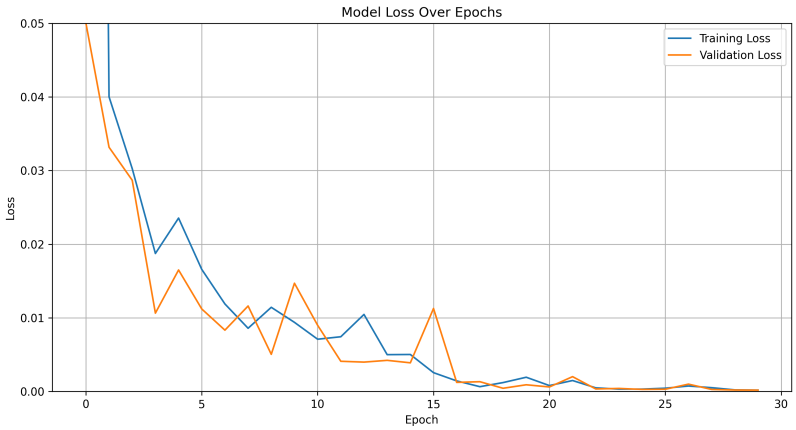


Figure 4.1: Learning curve of the CNN model, including the training and validation sets.

This observation is further supported by Figure 4.2, which presents two plots. In these plots, the blue dots represent actual values from the test set, while the red line illustrates the predicted values generated by the model. The left plot shows a randomly selected winter sample, demonstrating near-perfect prediction. At every location along the floating platform, the model closely matches the actual values. In contrast, the right plot, depicting a summer sample, reveals that the model does not capture the trend as accurately. However, the red line follows the general trend of the blue dots, indicating that although the predictions are less precise, the model successfully identifies some underlying patterns.

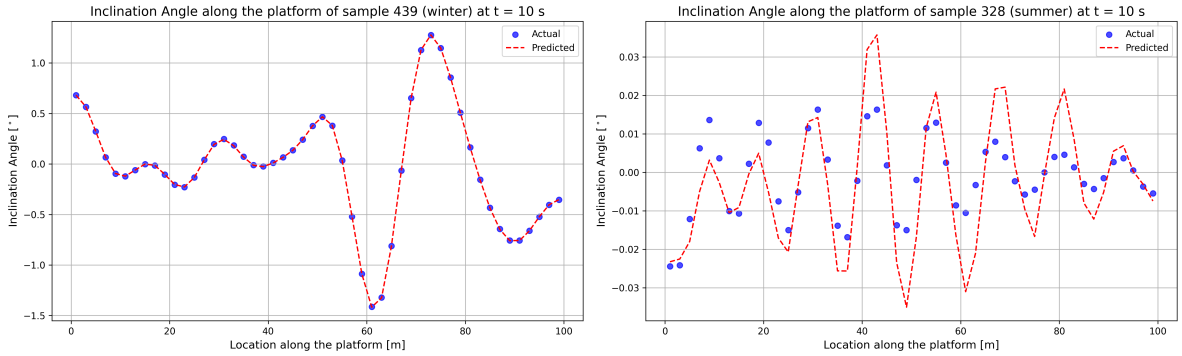


Figure 4.2: Predicted and actual values of the inclination angle φ along the floating platform at time instance $t = 10$ s for a sample drawn from winter (left) and summer (right).

The summer sample presented is not randomly selected; it was chosen because it has one of the lowest standard deviations, $\sigma_Y = 0.018°$. Compared to the winter sample, this value is approximately 1/100th of the corresponding winter values. Referring to the boxplot of tilt standard deviations in Appendix A, a value of 0.014 lies at the lower end, indicating that such cases are rare in the dataset. This imbalance likely hinders the model's ability to learn patterns effectively, resulting in poor generalization for values in this range.

From a physics perspective, this observation is expected. Larger wave heights induce larger tilts due to stronger forces, while very small tilts result from minimal forces, which are often less predictable and introduce more variability and noise into the data.

The similarity between actual and predicted values is further illustrated in the temporal predictions of the CNN model. Figure 4.3 shows the predicted tilt over time for the same seasonal samples. The upper plots display predictions at $x = 51$ m for a winter sample, demonstrating high accuracy throughout

the time period. Even with an irregular signal, the model predicts it almost perfectly. In contrast, the lower plots, representing summer samples, show less accurate predictions. While the model captures the general trend, it struggles with precision for samples exhibiting smaller tilt magnitudes. However, inaccurate predictions are infrequent, as the model generalizes well and yields accurate results for spring and autumn samples. Appendix D.1 provides similar graphs for these seasons, where the model, like in the winter sample, almost perfectly predicts the response.

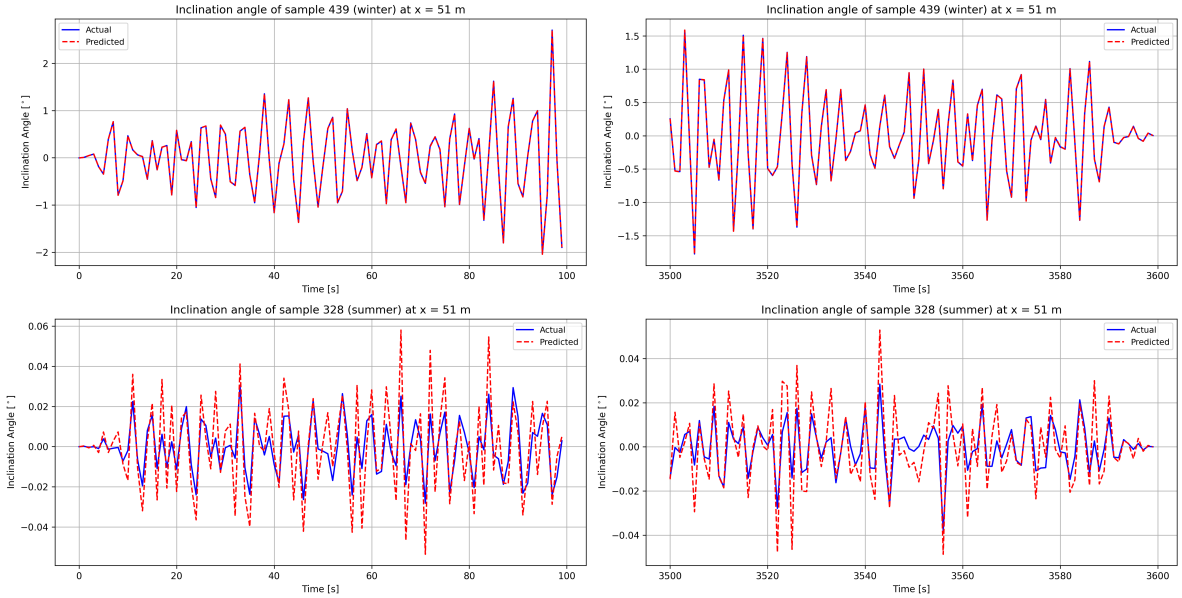


Figure 4.3: Predicted and actual values of the tilt φ over the first and last 100 s of a storm at $x = 51$ m for a sample drawn from winter (top) and summer (bottom).

4.1.2. Evaluation Metrics Assessment

The accuracy of the spatial and temporal predictions is clearly illustrated in Figure 4.4, which plots the percentiles of the mean absolute error (MAE) and mean absolute percentage error (MAPE) over time for each location along the floating platform. The percentiles indicate, for example, that the 50th percentile corresponds to the value below which the MAE or MAPE falls for 50% of the time. The MAE provides a direct and intuitive interpretation of the average error, and can be calculated as following:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{4.1}$$

It is the absolute error between the predictions ($\hat{y}_i$) and the actual values (y_i). The MAPE measures the average absolute percentage difference between the predicted and the actual values, see equation 4.2. Since the MAPE is scale-independent, it is useful for comparing models across different datasets. Division by zero is prevented by dividing by $\max(y_i,\ \epsilon)$, with $\epsilon = 1 \times 10^{-7}$ as a standard value.

$$\text{MAPE} = \frac{100\%}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{\max(y_i,\ \epsilon)}\right| \tag{4.2}$$

However, when considering this value, an actual tilt of $1° \times 10^{-7}$ is unrealistic. Such small tilts are effectively horizontal and do not need to be assessed using an error metric. This is primarily because when a predicted value $\hat{y}_i$ deviates from a very small actual value y_i, the denominator becomes very small, inflating the error term. For example, if $y_i = 0.001°$ and $\hat{y}_i = 0.002°$, the MAPE for that specific point

would be 100%. Even though the predicted and actual values are close, the overall MAPE increases drastically, which is not representative. Therefore, in this research, the epsilon term is set to 1×10^{-3}. Note that this change does not affect the MAPE percentile plots, as these small values probably only constitute the smallest 5% of the MAPE.

In the left plots, representing the winter sample, even for small tilts (maximum 1.5°), 95% of the values have a MAE below 0.008° and a MAPE below 2.20°, indicating extremely accurate predictions.

In contrast, the right plots, representing the summer sample, show that 95% of the values have a MAE below 0.025°, which is still relatively small. However, this low error is mainly due to the significantly smaller tilt values in the sample, resulting in a proportionally smaller MAE. The lower right plot highlights a large MAPE, indicating that the model struggles to accurately predict this sample despite the low absolute error, as the percentage error is amplified by the small magnitude of the target values.

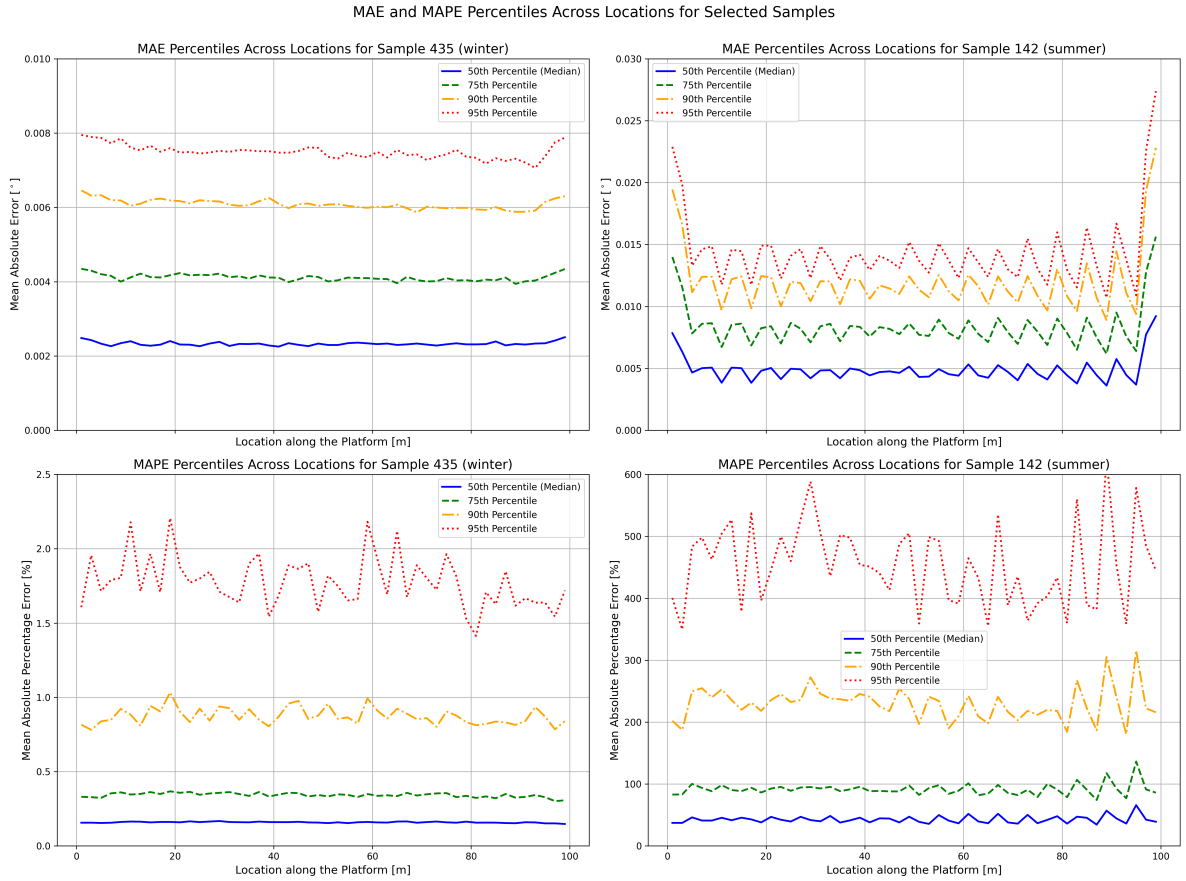


Figure 4.4: The mean absolute error (MAE) 50th, 75th, 90th, and 95th percentiles over time for each location along the floating structure for a sample drawn from winter (left) and summer (right).

Another notable observation in both figures is that the MAE is slightly higher at the beginning of the structure ($x = 1$ m). This is where the wave first impacts the floating platform and where boundary conditions are applied, making predictions more challenging. Beyond this point, the MAE fluctuates around a relatively constant value (depending on the percentile) until the end of the structure. Since the platform consists of a single homogeneous floater, it is expected that the MAE remains consistent without sudden peaks. However, at the end of the structure ($x = 99$ m), where different boundary conditions apply, the MAE increases slightly. To demonstrate that the MAE remains small across all samples, similar plots for the spring and autumn seasons are included in Appendix D.1.

Table 4.1 presents three overall metrics used to further assess the model's performance: MAE, MAPE, and WAPE. The MAE is an easy to comprehend metric, used to give insight into the absolute error.

However, this also means that when y_i is large, the MAE will tend to be larger as well. The MAPE is relative error metric, it offers insight into the error magnitude relative to the actual values. For example, while the predictions for the autumn sample (Appendix D.1) appear more accurate than those of the summer sample, the MAE is lower for the summer sample due to the smaller magnitude of the tilt values. In contrast, the MAPE is lower for the autumn sample, reflecting its relative accuracy despite higher absolute values.

The final metric is the Weighted Absolute Percentage Error (WAPE), which, like MAPE, calculates the relative error:

$$\text{WAPE} = \left(\frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{\sum_{i=1}^{N} |y_i|} \right) \times 100\% \tag{4.3}$$

However, unlike MAPE, WAPE is less sensitive to near-zero values that can disproportionately inflate the error. By aggregating the relative error across all samples, WAPE provides a more stable metric. Each actual value y_i implicitly acts as a weight for its corresponding error $|y_i - \hat{y}_i|$. Larger values contribute more to both the numerator and the denominator, ensuring that errors on larger deviations have a proportionate impact on WAPE, which makes sense considering very small deviations can be considered negligible on OFPVs. The lower the WAPE, the better the model's overall performance, as it represents the proportion of total error relative to the sum of actual values.

Table 4.1: Evaluation metrics subjective to the CNN model.

Season	MAE	MAPE	WAPE
Winter	3.23×10^{-3}	3.23%	0.43%
Spring	2.98×10^{-3}	3.80%	0.54%
Summer	3.08×10^{-3}	3.90%	0.56%
Autumn	3.54×10^{-3}	3.77%	0.44%
Overall	3.22×10^{-3}	3.58%	0.48%

The overall MAE, MAPE, and WAPE scores are exceptional. A forecast is considered reasonable when the MAPE is between 20% and 50%, a forecast is considered good when a MAPE is between 10% and 20%, and highly accurate when the MAPE is below 10% [23]. The WAPE does not have universal thresholds applicable across all industries or use cases. It is therefore context-dependent. In the context of this thesis, a WAPE of 0.48% is considered exceptionally good. To put this number into perspective: 99.52% of the tilts are predicted correctly across 3,600 time steps, 50 locations, and 450 samples.

4.2. Long Short-Term Memory Performance

This section evaluates the performance of the Long Short-Term Memory (LSTM) model. Given that the model specializes in temporal predictions, it is expected to perform better over time than across spatial dimensions. After training, the predicted values ($\hat{y}$) will be inverse-scaled to their original scale and compared side by side with the target test values (y) to assess the model's performance. The results will be analyzed both temporally and spatially.

4.2.1. Spatial and Temporal Tilt Analysis

Examining the training progress in Figure 4.5, the learning process appears steady, with both the training and validation losses decreasing towards zero. There is a sudden peak at epoch 10, which is likely due to a gradient explosion. However, since this peak is immediately followed by a return to the previous trend in the subsequent epoch, it should not adversely affect the learning process.

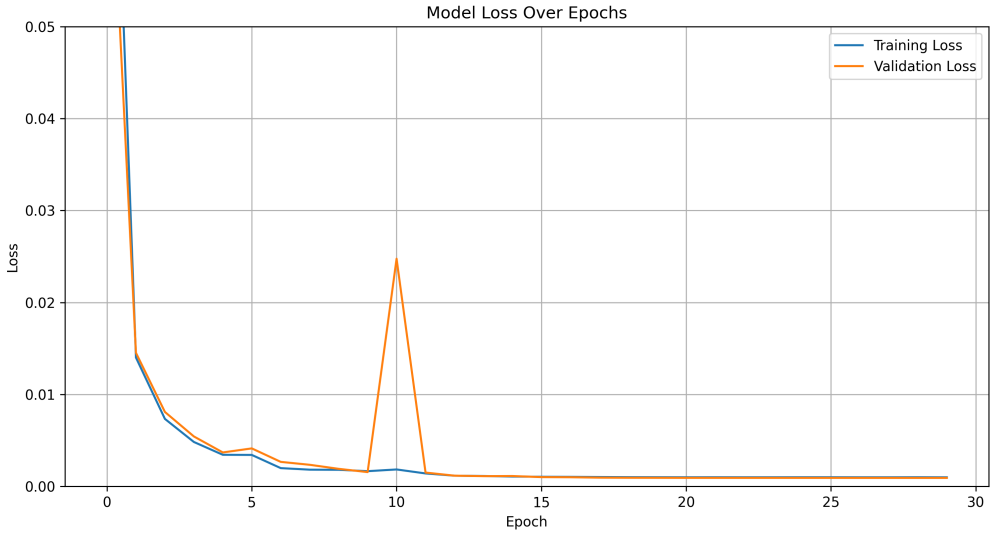


Figure 4.5: Learning curve of the LSTM model, including the training and validation sets.

Despite the training and validation losses trending toward zero, the results reveal a different outcome. Figure 4.6 presents two plots of the predicted and actual inclination angles for samples taken from winter and summer at a snapshot of $t = 10$ s. The left plot shows a winter sample, where the predicted red-striped line closely follows the actual blue-dotted line, indicating a near-perfect prediction. Across various locations along the platform, the model demonstrates strong alignment with the actual values, suggesting that it effectively captures the underlying trends and patterns.

In contrast, the right plot shows a summer sample, where the model generalizes poorly, with minimal alignment between predicted and actual values. The actual values, represented by the blue-dotted line, exhibit a cyclic movement that gradually dampens, while the predicted values, represented by the red-striped line, display an irregular response with some cyclic characteristics. Although the model captures certain patterns, as indicated by the consistent spacing between peaks, its predictions lack precision for this sample.

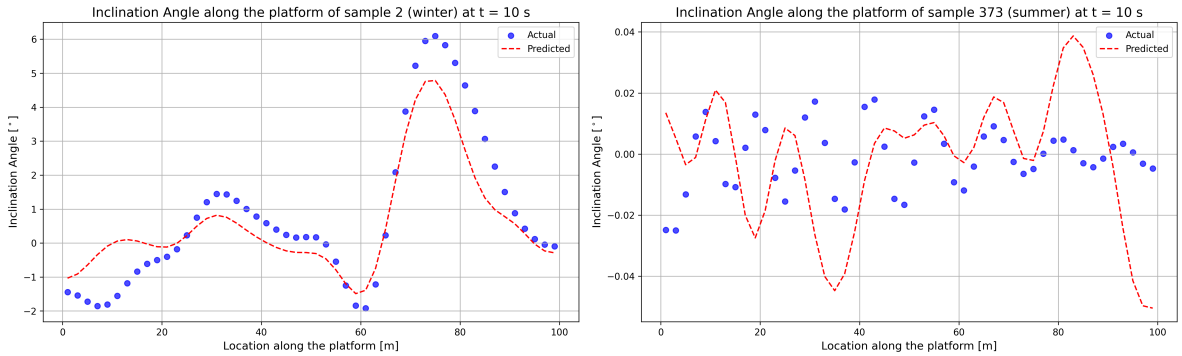


Figure 4.6: Predicted and actual values of the inclination angle φ along the floating platform at time instance $t = 10$ s for a sample drawn from winter (left) and summer (right).

The results for the summer sample show significantly smaller magnitudes compared to the winter sample. Similar plots for samples taken from spring and autumn are provided in Appendix D.2 for snapshot $t = 10$ s. In these plots, the model appears to converge toward the correct solution up to a certain point, after which it follows the shape of the actual solution but does not perfectly align with the blue-dotted

line, similar to the winter sample in Figure 4.6.

Since this model is designed to capture temporal dependencies, weaker spatial performance was anticipated. The key question, then, is how well the LSTM model performs over time. Figure 4.7 presents the results for the same samples plotted over time, showing the first 100 time steps and the time steps from $t = 3300$ s to $t = 3400$ s. Note that the final window cannot be fully predicted, as overlapping windows require time steps up to 3,619 seconds to complete the last window.

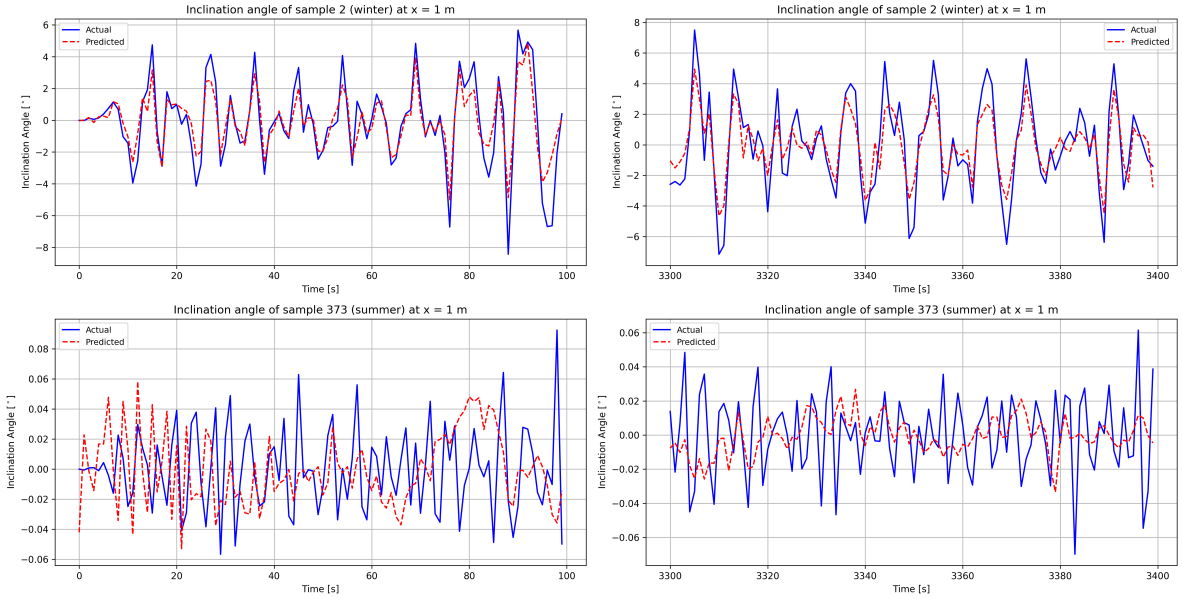


Figure 4.7: Predicted and actual values of the tilt φ over the first and last 100 s of a storm at $x = 1$ m for a sample drawn from winter (top) and summer (bottom).

The top two plots represent the results over time for the winter sample. Both plots demonstrate that the model can capture the high non-linear behavior of tilt over time. Although the red-striped line does not perfectly align with the blue-dotted line, it reflects well-learned behavior on unseen data. In contrast, the predictions for the summer sample are less accurate. While the model identifies some patterns, it is not as precise for samples with larger tilt magnitudes, as illustrated in Appendix D.2.

4.2.2. Error Metrics Assessment

The model's performance can also be quantified by examining the percentile graphs of MAE and MAPE in Figure 4.8. The top two plots display the MAE for the winter and summer samples. The MAE for the summer sample is lower than that of the winter sample, despite the earlier graphs indicating a better prediction for the winter sample.

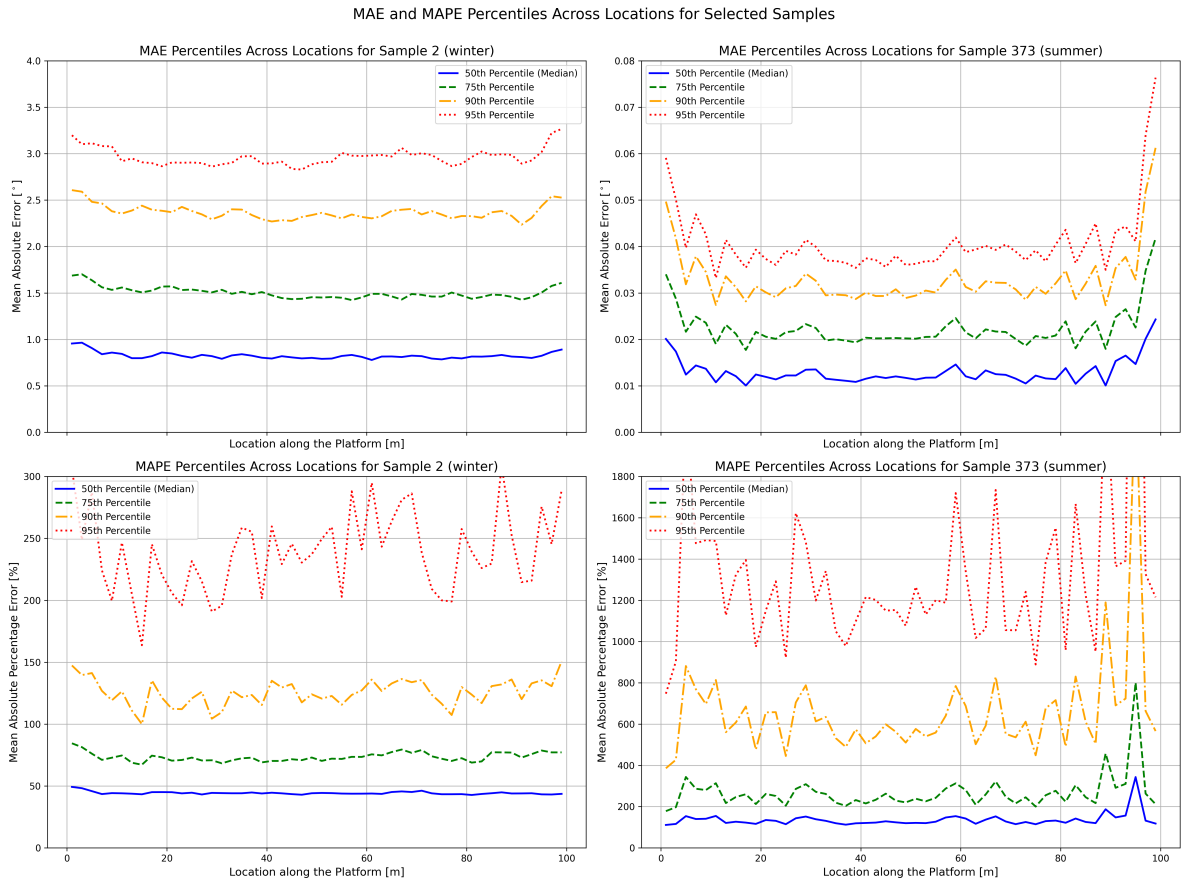


Figure 4.8: The mean absolute error (MAE) at the 50th, 75th, 90th, and 95th percentiles over time for each location along the floating structure for a sample drawn from autumn (left) and summer (right).

This discrepancy becomes evident in the lower two plots, which display the MAPE. For the winter sample, the 50th percentile MAPE remains below 50% across most of the platform, whereas for the summer sample, it is significantly higher, indicating a smaller relative error in the winter sample. Additionally, the summer sample is not representative of the entire dataset, as shown by the corresponding graph in Appendix D.2. Most samples exhibit a median MAPE below 100%, with the summer sample being an extreme case selected for illustrative purposes.

Table 4.2 presents the overall MAE and MAPE across seasons. While the MAEs are relatively consistent across different seasons, the MAPE varies significantly. This confirms the earlier observations from the plots, underscoring the high relative error. An overall MAPE of 49.48% suggests that the model's reliability is reasonable, but limited. When considering the WAPE, the LSTM model still achieves a reasonably low error, with over 97% of all values in the test set being predicted correctly.

Table 4.2: Evaluation metrics for the LSTM model.

Season	MAE	MAPE	WAPE
Winter	2.52×10^{-2}	39.94%	2.40%
Spring	2.49×10^{-2}	53.08%	3.44%
Summer	2.64×10^{-2}	69.83%	4.18%
Autumn	2.52×10^{-2}	33.44%	2.23%
Overall	2.54×10^{-2}	49.48%	2.89%

4.3. CNN-LSTM Hybrid Performance

In this section, the performance of the CNN-LSTM hybrid model is evaluated using the test set by comparing the predicted values ($\hat{y}$) with the actual values (y). Various evaluation metrics, as outlined in Subsection 4.2.1, are used to assess the model's accuracy. While the hybrid model is expected to outperform the CNN and LSTM models due to its increased complexity, literature suggests that higher complexity does not always guarantee better performance. Therefore, a detailed evaluation is necessary. Subsection **??** will further analyze the model's strengths by examining its input parameters and hyperparameter configuration.

4.3.1. Spatial and Temporal Tilt Analysis

Figure 4.9 illustrates the learning curve during the training progress of the hybrid model. After a few epochs, both the training and validation losses decrease below 5×10^{-4}. Subsequently, the losses continue to decline towards zero. The learning curve clearly indicates that the model does not require a large number of epochs to improve performance, suggesting that 20 epochs are sufficient.

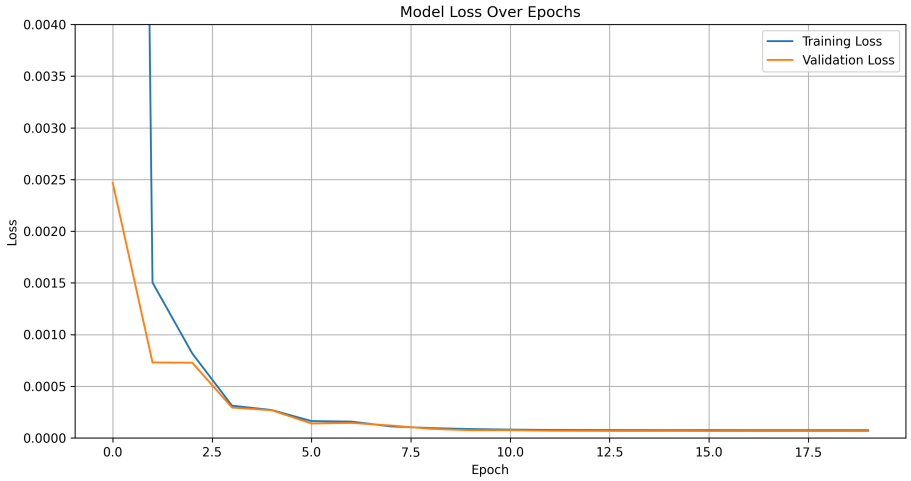


Figure 4.9: Learning curve of the CNN-LSTM Hybrid model, including the training and validation sets.

To accurately assess the model's performance, the results are analyzed both spatially and temporally. Figure 4.10 compares the actual and predicted tilt values along the floating platform at a specific time instance ($t = 10$ s). The selected samples represent scenarios with the highest and lowest standard deviations, thereby illustrating the model's performance under extreme conditions. The figure demonstrates a near-perfect prediction for the autumn sample, whereas the prediction for the summer sample is noticeably less accurate. This outcome aligns with expectations, as the majority of the data corresponds to higher tilt ranges, making these ranges more prominently represented in the training set.

In both plots, the blue-dotted line exhibits non-linear behavior. For the autumn sample, the hybrid model closely follows the actual values, accurately predicting the tilt across the platform. In contrast, the summer sample prediction deviates significantly, with the model output resembling a piecewise linear trend rather than capturing the true non-linear (cyclic) pattern. This suggests that the model struggles to capture the complex fluid-structure interaction dynamics associated with small tilt amplitudes. The underlying reason for this limitation is likely linked to the variability of forces at lower amplitudes. As shown in Equation 1.17, the equation of motion for a beam includes a transverse load, primarily driven by wave pressure on the submerged portion of the floater and the vertical component of the orbital motion of water particles.

According to Figure 1.7, given the structural properties, the floating platform operates in the flexible body motion dominant region (a typical behavior for VLFS). However, body dominance can vary depending on the storm characteristics. The autumn sample, with the highest standard deviation, likely includes longer wavelengths, placing it firmly in the flexible region. Conversely, the summer sample,

with the lowest standard deviation, likely corresponds to shorter wavelengths. This distinction is also evident from the plots: the blue-dotted line for the summer sample crosses the zero line frequently, indicating elastic body motion dominance, while the autumn sample shows fewer zero crossings, suggesting less frequent variation and a behavior closer to rigid body motion dominance. Even though the structure remains in the flexible region overall, the shift toward rigid body motion dominance under such conditions explains the observed discrepancies in prediction accuracy.

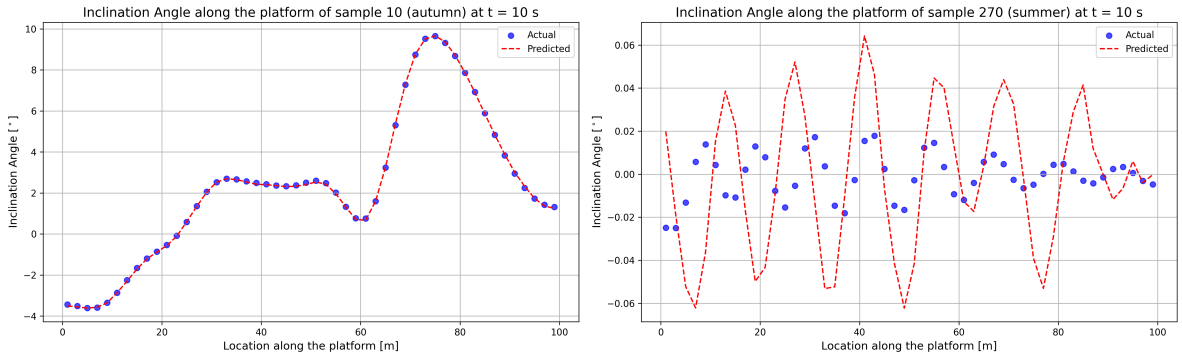


Figure 4.10: Predicted and actual values of the inclination angle φ along the floating platform at time instance $t = 10$ s for a sample drawn from autumn (left) and summer (right).

Small forces are easily triggered and can occur simultaneously, resulting in a superposition of minor forces. This often leads to complex, non-linear behavior in the structure. In contrast, large forces occur less frequently and, when present, are typically dominant. The simultaneous presence of multiple large forces is uncommon, and if smaller forces occur alongside larger ones, they are generally negligible in comparison. Therefore, from a physics standpoint, the observed differences in prediction accuracy for varying tilt magnitudes are logical.

When examining the space-tilt plots in Appendix D.3, which present two different samples at time instances $t = 10$ s, $t = 500$ s, and $t = 3000$ s, the model produces nearly identical results for both samples, differing primarily in tilt magnitude. Despite this variation in magnitude, the shape of the plots remains consistent, highlighting the model's remarkable pattern recognition capability. This suggests that the JONSWAP spectra of these two samples are nearly identical, differing only in the magnitude of the spectral density values. The consistent shape across different time instances demonstrates that the model captures the underlying physics effectively.

A similar conclusion can be drawn from the hybrid model's temporal predictions shown in Figure 4.11 and Appendix D.3. The model predicts medium to large tilt angles with high accuracy but struggles with small tilt angles, where predictions become less precise. Visually, the time series response for both small and large tilts appears non-cyclic, lacking distinct repeating patterns corresponding to specific wave impacts. Instead, various waves impact the floater at different time instances, resulting in non-cyclic, non-linear behavior. Despite these complexities, the model successfully identifies the underlying patterns, achieving a near one-to-one alignment between the actual and predicted values.

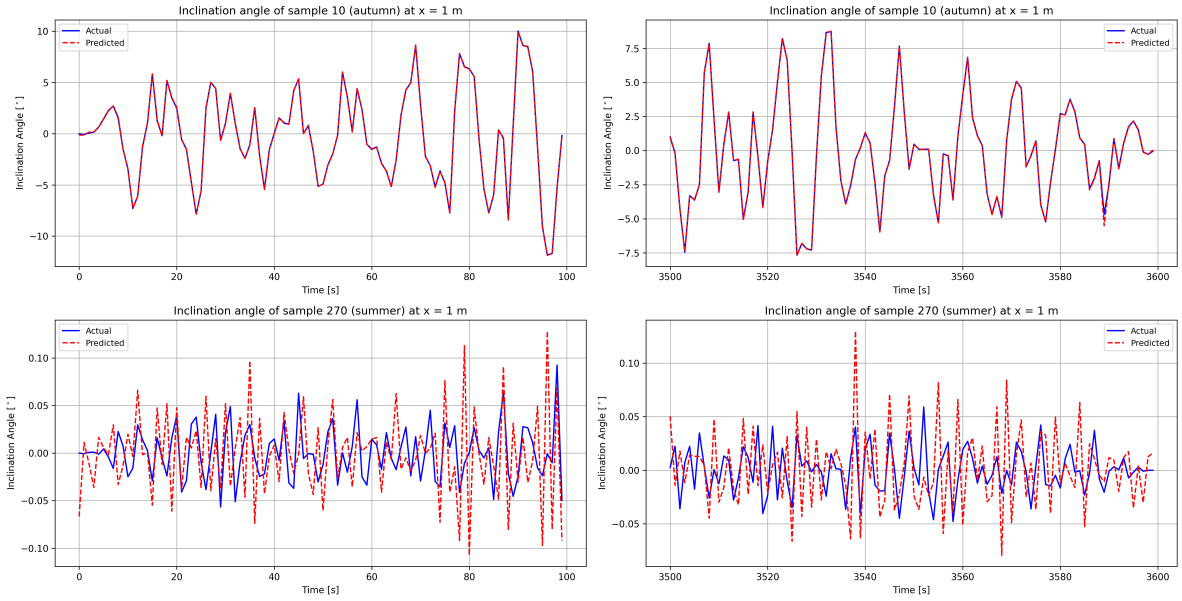


Figure 4.11: Predicted and actual values of the tilt φ over the first and last 100 s of a storm at x = 1 m for a sample drawn from autumn (top) and summer (bottom).

4.3.2. Error Metrics Assessment

The model's performance can be further evaluated using the percentile graphs of the MAE and MAPE shown in Figure 4.12. Both plots indicate that the MAE remains consistently low across the platform. However, the MAPE plots reveal a disparity in accuracy between the two samples: the summer sample exhibits higher relative errors, while the autumn sample demonstrates high accuracy, with 95% of all time steps during this storm having a MAPE below 10%. To further validate the model's performance, similar plots for samples drawn from winter and spring are provided in Appendix D.3.

Figure 4.12: The mean absolute error (MAE) at the 50th, 75th, 90th, and 95th percentiles over time for each location along the floating structure for a sample drawn from autumn (left) and summer (right).

Another notable observation in these figures is that the error tends to increase at the boundaries of the floaters. While the model effectively captures the fluid-structure interaction dynamics in the middle of the floater—where structural behavior is more uniform—it faces challenges at the ends of the floater. This difficulty is likely due to complex boundary effects and structural constraints that introduce additional variability, making accurate predictions more challenging in these regions.

The overall performance of the hybrid model is summarized in Table 4.3, which presents the MAE, MAPE, and WAPE, both overall and by season. The MAPE indicates reasonable to good results, while the WAPE shows that over 99% of the tilt values are predicted correctly, reflecting high accuracy. Despite this discrepancy, the results underscore the importance of evaluating the model using multiple error metrics. Even with a larger epsilon, the MAPE can be sensitive to near-zero values, which inflate the error percentage, whereas the WAPE provides a more balanced measure of the model's overall predictive accuracy.

Table 4.3: Evaluation metrics for the CNN-LSTM Hybrid model.

Season	MAE	MAPE	WAPE
Winter	5.20×10^{-3}	18.81%	0.73%
Spring	5.43×10^{-3}	23.06%	1.19%
Summer	6.52×10^{-3}	32.91%	1.61%
Autumn	5.12×10^{-3}	15.02%	0.75%
Overall	5.50×10^{-3}	22.11%	0.98%

4.4. Model Comparison

Now that the results of each of the three deep learning models are known, the models can be compared based on their performance. The effectiveness of the training process will be assessed, as well as the outcome of the model. After the comparison of the different models, one model will be selected that can be presented as the best model, that is, the model with the best accuracy and applicability. In the next chapter, certain criteria will be set on whether and when this model can be used in real-life scenarios.

First, let's examine the training process. The learning curve of the CNN is quite different from the learning curves of the LSTM and CNN-LSTM hybrid. This is mostly due to the batch size of 8 compared to batch sizes of 64 and 36, respectively. With more batches being processed simultaneously, the learning process seems to be steadier. This can clearly be seen in the shapes of the three learning curves. Also, the learning curve of the CNN tends toward zero after 20 epochs, while the learning line of the LSTM is almost horizontal after 10 epochs. In the hybrid model, this is already the case after 8 epochs. These quick drops indicate that the models effectively capture the patterns during the training process, especially in the hybrid model. However, this is difficult to compare since the batch sizes differ.

In terms of training time, the LSTM model scores the worst. Due to the many LSTM blocks, each with its corresponding gates, it takes a while to train: $\pm$ 65 minutes. The hybrid model has fewer LSTM layers but includes some convolutional blocks as well, leading to a training time of approximately 45 minutes. The CNN model is very efficient, and even with more epochs, the training time is only around 4 minutes.

In the sense of matching complexity with effectiveness, the training process of the CNN model seems to be very efficient. However, the number of trainable parameters is very large in this model and can lead to out-of-memory errors when the layers are not handled correctly.

When looking at the results of the models, the LSTM model seems to perform the worst. The space-tilt graphs show some patterns, but none of the red-striped lines follow the blue-dotted lines very accurately. Especially when considering small tilt angles, the predictions are off. The space-tilt graphs of the CNN and CNN-LSTM hybrid closely align. Only small tilt angles are harder to predict, but this is the case for both models. This is probably due to the underrepresentation of the smaller values, but also because of the Huber loss function used, a loss function where large outliers are penalized more. So from the space graphs, it is very hard to distinguish a more accurate model.

When looking at the time-tilt graphs, again the LSTM model doesn't perform well. The CNN and CNN-LSTM hybrid again go head-to-head since all the time predictions seem to be nearly perfect. However, the error metrics favor the CNN model since it has an overall MAE of 3.22×10^{-3}, an overall MAPE of 3.58% and an overall WAPE of 0.48%. While the CNN-LSTM hybrid model has an overall MAE of 5.50×10^{-3}, an overall MAPE of 22.11%, and an overall WAPE of 0.98%. But the error metrics do not say everything since the MAE doesn't give any information on the relative error, the MAPE can be sensitive to near-zero values, and the WAPE favors the large tilts.

Other powerful visualization tools to compare the two models, which complement these quantitative error metrics, are actual-versus-predicted plots. These scatter plots offer a more intuitive understanding of how the model behaves on individual samples and how well the model aligns with or deviates from the perfect prediction line.

Since both the CNN model and the CNN-LSTM hybrid model perform well on various storms, selecting more specific and representative samples can be particularly useful in explaining why one model might outperform or behave differently from the other. Therefore, in Figure 4.13 and Figure 4.14, different scatter plots are shown for samples representing: the largest and smallest standard deviations at the beginning of the structure ($x = 1\ m$), the middle of the structure ($x = 51\ m$), and the end of the structure ($x = 99\ m$) for both models.

From the top three plots in both figures, there are not many values or patterns to distinguish. With a large standard deviation, i.e., a large response of the structure, both models seem to follow the perfect prediction line. There are few to no predicted values that deviate from the actual values. Only in the scatter plot of the hybrid model do some outliers deviate from the perfect prediction line, but this amount is minimal. Another point to note is the error magnitude. It is difficult to quantify this error magnitude

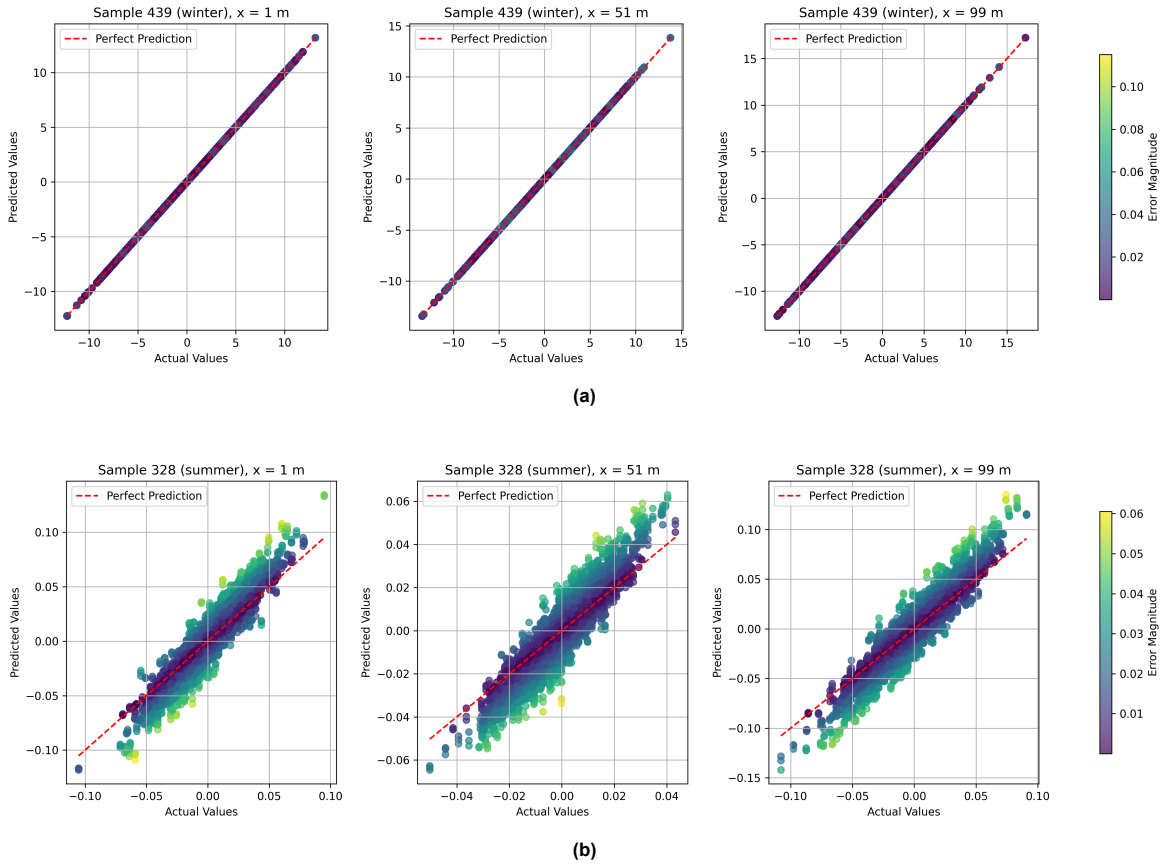


Figure 4.13: Actual versus Predicted scatter plots of the CNN model for three locations along the structure, including an error magnitude color map. For a sample with the largest standard deviation: 4.15° (a), and a sample with the smallest standard deviation: $1.48^\circ \times 10^{-2}$ (b).

since most of the values lie on the x-y line, but in the hybrid model plot, some single values exceed an error magnitude of 0.4°, which is not the case in the CNN model plots.

In the lower plots of both figures, illustrating the same scatter plots for the samples with the smallest standard deviation, a significantly larger spread is visible. In the CNN plots, the elongated 'cloud' clusters around the diagonal line, indicating that the model captures the patterns well. The lower plots of Figure 4.14 show a more scattered cloud, with an increased width (i.e., how much the points spread out perpendicular to the diagonal). This indicates that the model performs worse on smaller standard deviations compared to the CNN model. Also, the min/max of the axes are the same in both figures, but the error in the hybrid plots reaches larger values.

Another noteworthy observation in both figures is the systematic shift or bias. In the lower plots of the CNN model, one may notice a slight upward or downward tilt further away from the center, as if the clusters had rotated a few degrees away from the perfect prediction line. This shows that the model overestimates at specific bands. While the model's prediction is accurate around zero values, it overestimates (in absolute terms) further away from zero. However, this is not the case in the scatter plots of the CNN-LSTM hybrid model. In these plots, although the accuracy is worse, the cloud does not show any form of underestimation or overestimation further away from zero values. The CNN model thus exhibits a small bias when estimating smaller values, which is not observed in the CNN-LSTM hybrid model. The LSTM layers in the hybrid model can capture sequential dependencies that a standalone CNN might miss. By modeling potential time- or sequence-based relationships, the hybrid architecture may correct for biases that occur when those relationships are ignored. Hence, the bias in the CNN predictions for small tilts.

In the previous sections, the MAE plots indicated a larger error at the boundaries of the structure.

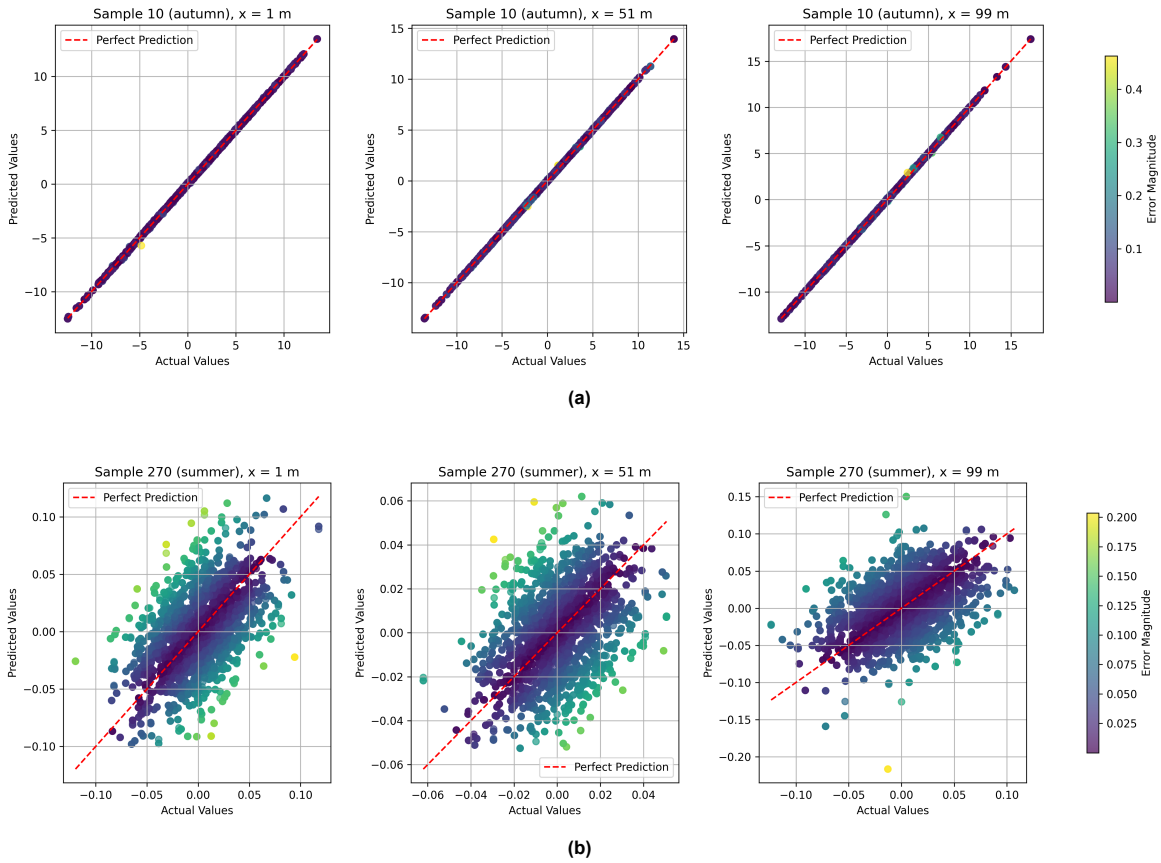


Figure 4.14: Actual versus Predicted scatter plots of the CNN-LSTM Hybrid model for three locations along the structure, including an error magnitude color map. For a sample with the largest standard deviation: $4.24°$ (a), and a sample with the smallest standard deviation: $1.82° \times 10^{-2}$ (b).

However, when examining the three locations in these scatter plots, the differences are so small that these scatter plots cannot confirm this. The min/max on the axes in the middle plots only show a smaller response in the middle of the structure. However, this only indicates the fluid-structure response and not the model's performance.

It is interesting to assess the model's performance in extreme cases. However, operating conditions are the most common scenarios the structure (and the model) will encounter. Therefore, it is also relevant to examine how the model performs in the majority of cases. In Figure 4.15 and Figure 4.16, three of the same scatter plots are shown from samples selected based on the most frequent standard deviation in the test dataset. This is done by selecting an arbitrary number of bins (10 in this case) in which the samples are classified based on the standard deviation. The three samples are selected from the bin with the highest count. Additionally, since the previous scatter plots with the lowest standard deviation revealed a lot of information about the models, the next three lowest standard deviation samples are also shown.

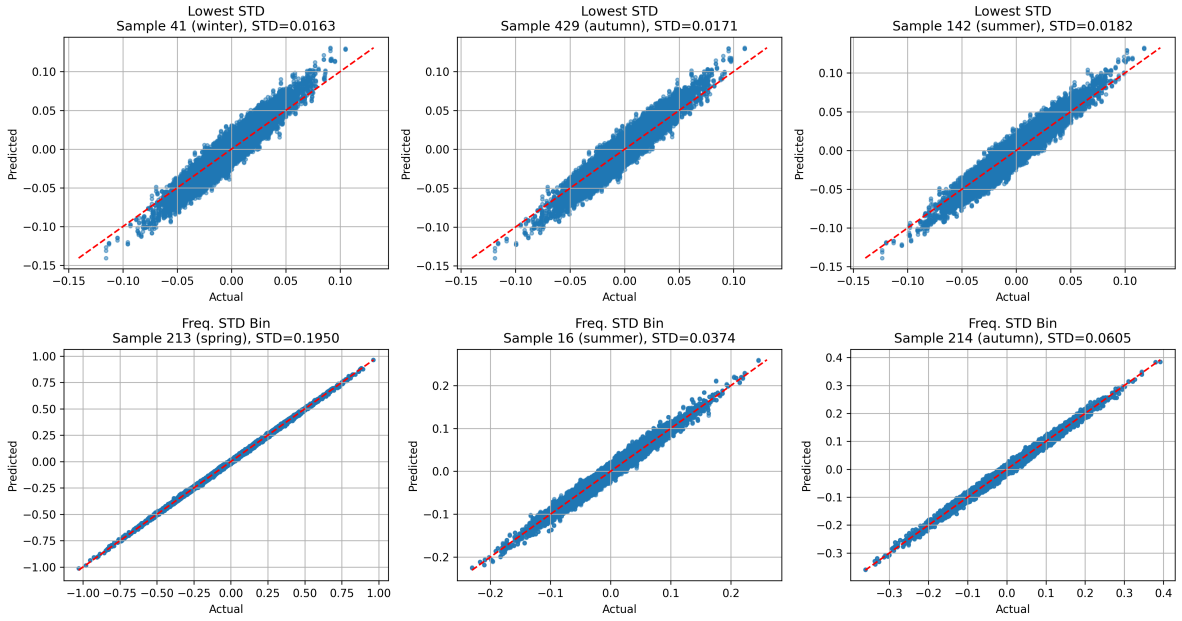


Figure 4.15: Actual versus Predicted scatter plots of the CNN model for the three next to lowest standard deviation samples (upper plots), and for three samples of the most frequent standard deviation in the test dataset (lower plots).

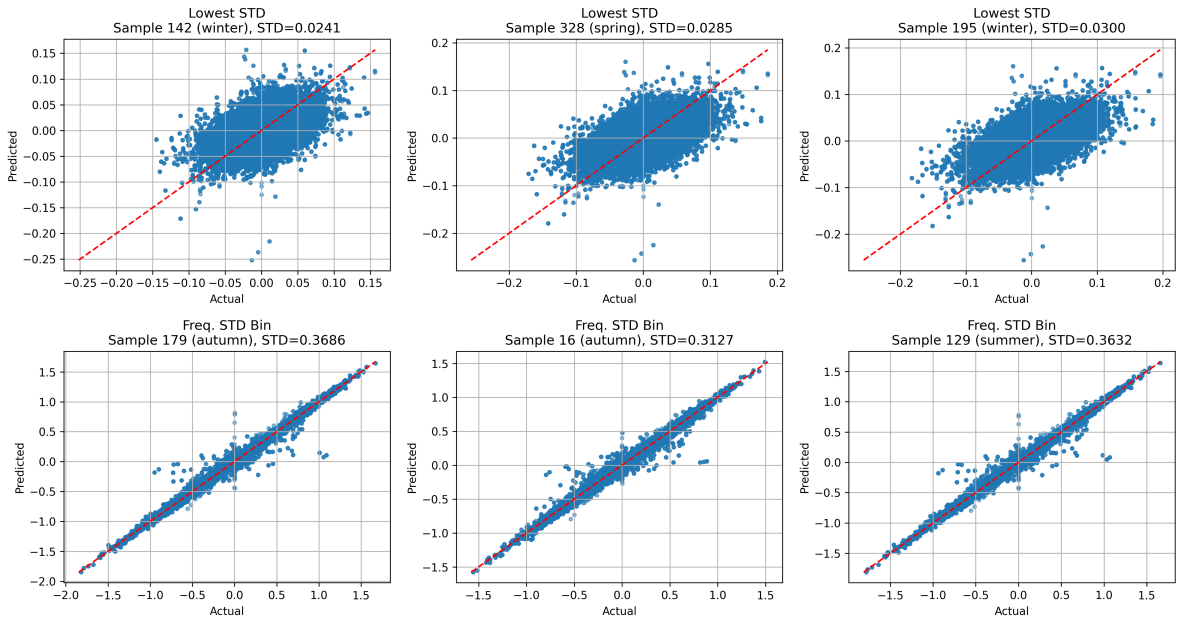


Figure 4.16: Actual versus Predicted scatter plots of the CNN-LSTM Hybrid model for the three next to lowest standard deviation samples (upper plots), and for three samples of the most frequent standard deviation in the test dataset (lower plots).

For the plots with the lowest standard deviation, there is not much additional information visible. However, from the top plots of the CNN model, it can be confirmed that the model has a bias towards overestimating small values away from zero. Again, the width of the clusters in the CNN-LSTM hybrid model is consistent on both sides of the perfect prediction line, indicating that there is no bias.

The lower plots definitely show more accurate results. The width of the cluster is much smaller, especially in the CNN-LSTM hybrid model. However, they also reveal a significant number of outliers away from the cluster. The majority of these single values lie around the zero predicted line, with most of them

between 0 and 0.5. This indicates that the model frequently predicts values in this range incorrectly. In this case, with more outliers around the zero predicted line, the model is underestimating values in this range. Further away from the center, the width of the cluster narrows and eventually forms a pointy end, demonstrating a more accurate prediction of larger values.

The lower plots of the CNN model clearly show some variation in the most frequent standard deviation bin. Even with a lower standard deviation, the model is more accurate in its predictions. The lower left plot shows a standard deviation comparable to those in the CNN-LSTM hybrid plots. It demonstrates excellent prediction, with almost all values lying on the perfect prediction line.

All of these plots show very good results for the majority of storms for both models. The CNN model excels in accurate prediction for medium to large tilts. It also shows decent results for smaller tilts but exhibits a bias towards over-predicting when slightly increasing these small values. The CNN-LSTM hybrid model shows good results in predicting medium to large tilts but remains inaccurate when predicting small tilts.

In terms of applicability, the CNN model shows excellent potential but cannot be used since it needs to be trained on input data containing multiple phase shifts, as discussed in Section 2.1. The CNN-LSTM hybrid model, on the other hand, contains information about the incoming waves—the wave elevation at two locations in front of the structure. While it shows good results, its accuracy still depends on the input. Since this model can be used in real-life scenarios, it is chosen as the best-suited model for predicting the tilt response of very large floating structures. But when can this model be used? In the next and final chapter, an error quantification will be performed, and the final research sub-question will be answered: to what extent is the final model reliable?

5

Error Quantification & Reliability

Currently, all data samples with a significant wave height exceeding 7.5 meters have been removed from the dataset, as well as samples with tilts smaller than 0.1°. Existing floating photovoltaics are operational only up to a maximum wave height of 4 meters, including both OFPVs and those deployed on inland waters. However, for the model to be applicable to both inland and offshore waters, multiple input constraints need to be analyzed to assess its performance in calm conditions and its behavior under extreme conditions. This evaluation can demonstrate whether the model is reliable under operating conditions or if it can even be used to assess the ultimate strength limit during severe weather. Sensitivity testing of these inputs will reveal how input constraints affect the model's performance and will help quantify the error under specific input conditions.

To assess the reliability of the final model, multiple sets of input constraints are created. The first set of input constraints includes samples that reach at least a certain tilt once during a storm. Each storm lasts for one hour, so only the samples that reach a specific tilt at least once during that hour are selected, while keeping the maximum significant wave height at 7.5 meters. The selected tilt thresholds are: 0.5°, 1.0°, 1.5°, 2.0°, and 5.0°. In Figure 5.1, the 95% confidence errors are presented over the length of the platform, with location index 0 corresponding to 1 meter and location index 49 corresponding to 99 meters. Three error metrics are shown: the MAE, MAPE, and WAPE.

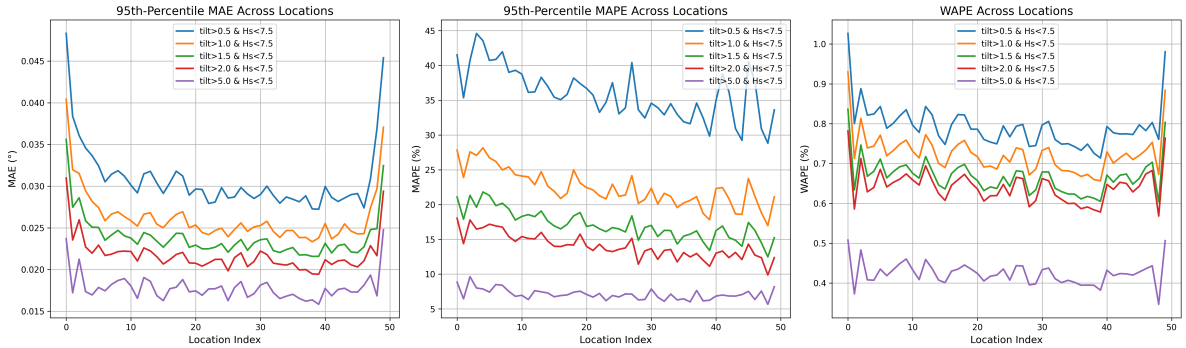


Figure 5.1: The 95th percentile MAE (left), the 95th percentile MAPE (middle), and WAPE (right) along all locations of the structure under different input constraints.

The three graphs show a clear correlation between small tilt values and the error metrics. Excluding small tilt values significantly reduces the overall error. Increasing the exclusion threshold from 0.5° to 1.0° reduces the MAPE by almost 50%. This reduction continues with each subsequent increase in the threshold. Including only tilt values greater than 5.0° further decreases the MAPE to below 10%, indicating that the model is highly accurate for larger tilts. However, as previously mentioned, the MAPE is very sensitive to near-zero values and may underestimate the model's performance for samples with

small tilts. These samples tend to fluctuate around the zero line and therefore include more values close to zero. The WAPE graph provides additional insight into the model's reliability. For instance, the blue line representing tilt values from 0.5° onward shows that the model is reasonably reliable. Moreover, the WAPE graph indicates that, for the majority of the structure, the model correctly predicts the tilts in 99.2% of cases. Increasing the tilt threshold further demonstrates even greater reliability.

Determining the cutoff for "small" target values (i.e., small inclination angles) warrants further discussion. Very small inclination angles can be considered nearly horizontal, while an increase in tilt can already reduce power output. Establishing a threshold between "horizontal" and "significant tilt" angles requires examining the relationship between inclination angle and power output. Alcañiz et al. [1] found that horizontal PV systems provide higher annual energy yields than floating panels or MPPT systems. Therefore, including samples with small inclination angles in the model may be unnecessary, as calm conditions already yield the highest energy output. In this research, a conservative threshold of 0.1° has been chosen. Increasing this threshold to 0.5°, 1.0°, 1.5°, 2.0°, or 5.0° further improves model performance.

The second set of input constraints is the significant wave height, H_s. This set includes only input and target values corresponding to significant wave heights below 12 meters, 10 meters, 8 meters, and 6 meters. In Figure 5.2, the same graphs are plotted for this set of input constraints. While a trend of decreasing error was observed when excluding smaller tilt values in the previous plots, this trend is not apparent here. Excluding larger significant wave height values does not necessarily lead to a reduction in MAE or MAPE. In fact, the middle plot suggests that the model performs better when larger significant wave height values are included. However, since this set of input constraints includes even more near-zero tilt values, the MAPE could again be misleading.

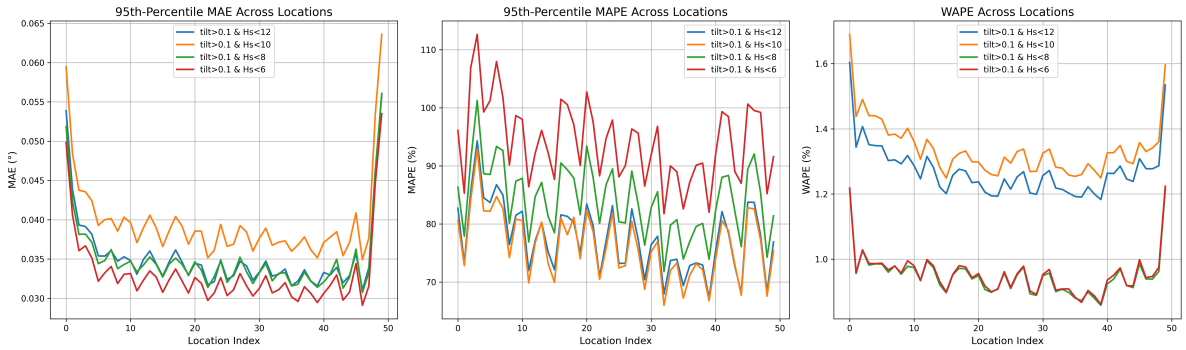


Figure 5.2: The 95th percentile MAE (left), the 95th percentile MAPE (middle), and WAPE (right) along all locations of the structure under different input constraints.

The WAPE shows reasonable results. The model appears to generalize better when including storms with H_s smaller than 12 meters rather than 10 meters. Beyond this, there is an error jump in the prediction for storms with H_s smaller than 8 or 6 meters, which does not seem to significantly impact the results. Given the variety in these graphs, it can be inferred that including storms with medium or large significant wave heights does not notably affect the model's performance, especially when comparing this set of input constraints to the previous set. Nevertheless, achieving a correct prediction rate of 98.7% when including all storms with H_s smaller than 12 meters and a tilt of 0.1° is still highly reasonable.

Since the model also performs well when storms with large significant wave heights are included, how will it perform if all storms with H_s smaller than 12 meters are included, while the tilt varies between 1.5°, 2.0°, 2.5°, and 3.0°? In Figure 5.3, the same graphs are presented for this final set of input constraints.

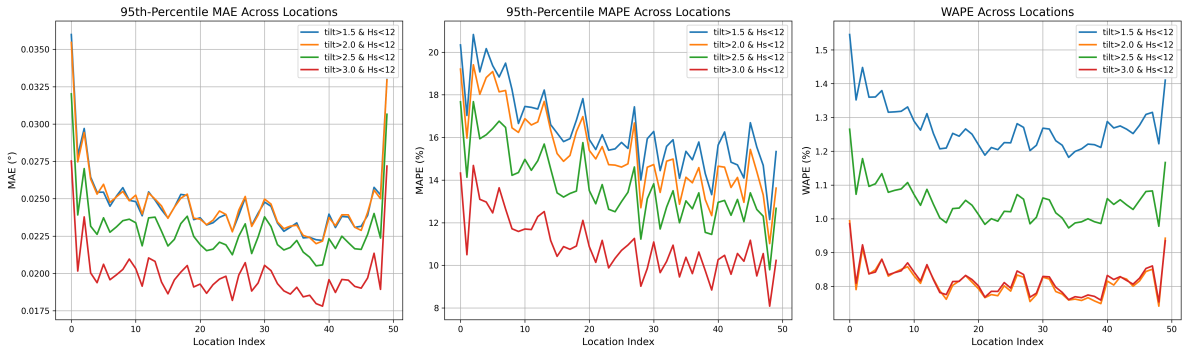


Figure 5.3: The 95th percentile MAE (left), the 95th percentile MAPE (middle), and WAPE (right) along all locations of the structure under different input constraints.

When comparing these graphs with those from Figure 5.1, many similarities emerge. For the same tilt values, the MAE graphs look almost identical, and the overall error magnitudes are comparable. Likewise, the magnitude in the MAPE plots is nearly the same, indicating that samples with large significant wave heights do not drastically affect the MAPE.

However, it is more apparent here that the MAPE decreases farther away from the wave impact location—a trend that was harder to discern in the first set of graphs. This suggests that when large waves are included, the model performs worse near the impact region but improves as the location approaches the end of the structure. At the boundaries, where the tilt is generally largest, the model shows its poorest performance, likely due to the more complex behavior of the structure.

Although these boundary tilt values are already specified in the target dataset, the model still struggles to capture boundary patterns compared to the more stable regions near the center of the platform. While this can be observed in all the graphs, it is especially clear in the MAE graphs because of how the error is defined. The MAE and WAPE both display this trend in error across the structure. Meanwhile, since the MAPE is sensitive to near-zero values, it can cause error spikes at specific points along the structure, leading to deviations from the MAE and WAPE trends and a less stable overall pattern.

The WAPE plot does show some differences in performance. The blue line corresponds to samples with tilts larger than 1.5°. The WAPE for these samples is almost halved when excluding significant wave heights of 7.5 meters or larger. This final plot demonstrates that the model performs better when large significant wave heights are excluded. In real-life scenarios, this is often applicable, as it is currently unrealistic to operate or even deploy OFPVs in offshore areas where significant wave heights of 12 meters can occur. Since determining the operational limitations of OFPVs is a study in itself, this chapter focuses on identifying under which environmental conditions this deep learning model is reliable enough to use.

Each of the three sets of input constraints showed distinct strengths and weaknesses. So, under which conditions can this CNN-LSTM hybrid model be used? The MAPE often exhibits a few extreme values that skew the overall error. Therefore, a 95% percentile MAPE is more representative, as it filters out most of these extreme values. Additionally, since WAPE provides a robust overall error metric, it is used to help determine whether the model is sufficiently reliable under certain constraints. However, because reliability thresholds are domain-specific and no established literature exists for this domain, the decision on whether a WAPE value is reliable will be made by combining the reliability of the MAPE with the values of the WAPE. Table 5.1 provides the reliability threshold range. Since quantifying whether a model is reliable solely based on these thresholds is challenging, this table serves more as a tool to evaluate the reliability of each input constraint and facilitate comparison.

Table 5.1: Reliability thresholds based on the MAPE and WAPE.

WAPE Range	Reliability Level	Interpretation
$WAPE < 0.5\%$	Very Reliable	Model is extremely close to actual values overall.
$0.5\% \leq WAPE < 1.0\%$	Reliable	Model has good accuracy for most practical purposes.
$1.0\% \leq WAPE < 3.0\%$	Moderately Reliable	Errors are somewhat higher, but might still be acceptable.
$WAPE \geq 3.0\%$	Not Reliable	Error is large relative to the signals of interest.

Now that there are thresholds defined that map the WAPE values to reliability labels, the reliability of the model can be mapped under different input constraints/environmental conditions. In table 5.2, the reliability of the CNN-LSTM hybrid model is charted under different input conditions. The MAE and MAPE are also depicted in this table for the sake of completeness.

Table 5.2: Reliability quantification under certain input constraints.

Constraints	MAE	MAPE	WAPE	Reliability
$\varphi > 0.5° \ \& \ H_s < 7.5\ m$	5.70×10^{-3}	37.33%	0.79%	Reliable
$\varphi > 1.0° \ \& \ H_s < 7.5\ m$	5.20×10^{-3}	36.66%	0.73%	Reliable
$\varphi > 1.5° \ \& \ H_s < 7.5\ m$	4.90×10^{-3}	35.50%	0.67%	Reliable
$\varphi > 2.0° \ \& \ H_s < 7.5\ m$	4.70×10^{-3}	35.58%	0.64%	Reliable
$\varphi > 5.0° \ \& \ H_s < 7.5\ m$	3.90×10^{-3}	16.07%	0.42%	Very Reliable
$\varphi > 0.1° \ \& \ H_s < 12\ m$	8.40×10^{-3}	49.66%	1.27%	Moderately Reliable
$\varphi > 0.1° \ \& \ H_s < 10\ m$	9.60×10^{-3}	45.49%	1.34%	Moderately Reliable
$\varphi > 0.1° \ \& \ H_s < 8\ m$	6.20×10^{-3}	50.51%	0.95%	Reliable
$\varphi > 0.1° \ \& \ H_s < 6\ m$	6.00×10^{-3}	52.74%	0.96%	Reliable
$\varphi > 1.5° \ \& \ H_s < 12\ m$	1.01×10^{-2}	33.91%	1.27%	Moderately Reliable
$\varphi > 2.0° \ \& \ H_s < 12\ m$	6.10×10^{-3}	36.27%	0.81%	Reliable
$\varphi > 2.5° \ \& \ H_s < 12\ m$	8.30×10^{-3}	34.59%	1.05%	Moderately Reliable
$\varphi > 3.0° \ \& \ H_s < 12\ m$	6.60×10^{-3}	33.09%	0.82%	Reliable

6

Conclusions

OFPVs show significant potential in renewable energy generation and can complement existing offshore floating wind farms by reducing the inter-annual variability of wind energy. To enhance the performance of these systems, proper research on the structural response of VLFSs that support OFPVs is essential.

Current finite element models accurately simulate the response of VLFSs under wave-induced forces, but they are computationally expensive and time-consuming. This limitation creates a need for surrogate machine learning models, which, once trained, can provide instantaneous predictions and aid in optimizing OFPV design parameters. This thesis aimed to develop such models and assess their accuracy in predicting the tilt response of VLFSs, providing a tool to support finite element models in preliminary design and real-time applications. With the development of these models, this research aims at answering the following research question:

How to develop and optimize a surrogate machine learning model to accurately predict the tilt response of very large floating structures?

In order to answer this main research question, several sub-questions were formulated. Below each sub-question will be answered based on the results of the previous chapters.

How effectively do temporal (e.g., LSTM) and spatial (e.g., CNN) neural networks address the interplay between wave-induced motions and tilt response in offshore floating solar platforms?

Three deep learning models were developed and evaluated: a CNN, an LSTM, and a CNN-LSTM hybrid. The CNN model effectively captured spatial features but was limited to a specific sea state due to the use of a single set of phase shifts. As a result, it generalized poorly across different sea states. Yet it showed an incredible potential, with a WAPE of 0.48%. The LSTM model captured temporal dynamics well, including recurring wave patterns, and generalized better to different sea states due to its input structure. However, it showed lower overall accuracy (WAPE of 2.89%) and required longer training times. The hybrid CNN-LSTM model combined the strengths of both approaches, resulting in better overall performance, with a low WAPE of 0.98%.

How do variations in environmental input parameters (e.g., wave height, peak period) affect the performance of the surrogate model?

All models exhibited similar behavior with respect to varying input parameters: they performed well under typical wave conditions but showed reduced accuracy in extreme sea states, particularly for small wave heights or short peak periods, resulting in small tilt responses. The errors were more pronounced for very low-amplitude tilts, likely due to underrepresentation in the training data and the complex behavior of the structure under minor forces. Sensitivity analysis confirmed that the model's accuracy decreases for extreme scenarios, emphasizing the importance of balanced training data.

What challenges and advantages arise when implementing real-time deep learning models for predicting VLFS tilt responses under varying environmental conditions?

A key challenge in implementing real-time deep learning models is ensuring computational efficiency while maintaining high prediction accuracy. Balancing this trade-off is essential to achieve computational feasibility without compromising accuracy. Another major challenge is generalizing the model across diverse environmental conditions. A model trained on limited environmental data may struggle to perform well in unseen conditions.

While training times varied significantly across models, all demonstrated minimal prediction times once trained, making them suitable for real-time applications. In most cases, the model successfully predicted the full 3600 seconds of a storm, enabling real-time monitoring and control.

How do variations in deep learning model architectures (e.g., CNN, LSTM, hybrid models) influence accuracy and computational efficiency in tilt response prediction?

The CNN model had the fastest training and inference time and demonstrated high accuracy. However, it struggled with generalization. The LSTM model, despite its simple architecture and small model size, required significant training time and delivered less accurate predictions. The hybrid CNN-LSTM model achieved a balance, offering reasonable accuracy with moderate computational demands.

To what extent can the final surrogate machine learning model be considered reliable for predicting VLFS tilt responses across varying scenarios?

Ultimately, the hybrid CNN-LSTM model was selected as the final model due to its consistent performance across typical sea states and its low prediction errors in most scenarios. Excluding extreme conditions, such as large waves ($H_s > 8$ m) and low-amplitude tilts (below 5°), enhances the model's reliability significantly. While the model's reliability decreases in extreme conditions, its generalization capability under typical operating conditions and unseen data makes it a valuable tool for OFPV response prediction.

7

Discussions

This thesis demonstrated the applicability of deep learning models in predicting the response of very large floating structures (VLFSs). The models showed performances ranging from reasonable to excellent under different input constraints. However, the reliability and practical usage of these models remain important points of discussion, beginning with the applicability and limitations of the deep learning models.

The CNN-LSTM hybrid model demonstrates robust overall predictive performance, evidenced by a WAPE of 0.98%. However, it yields a higher MAPE of 22.11%, indicating that while errors at smaller tilt angles are relatively larger, they have less influence on the WAPE due to its implicit weighting by y_i. This aligns with the findings in Section 1.2, which show that small inclination angles minimally affect the DC power output, and are thus less important in real-life application.

Despite the discrepancy between WAPE and MAPE, the model's accuracy is sufficient for preliminary studies and operational decision-making. Its reliability can be further enhanced by integrating the input constraints discussed in Section 5. Since the CNN-LSTM hybrid model was developed and validated for a specific OFPV geometry, its primary application is in operational environments where the platform design is fixed. Operators can use the model to rapidly predict tilt under forecast or real-time environmental conditions, allowing them to make quicker decisions about power output optimization and maintenance scheduling. For instance, if wave height and wind speeds are expected to increase above threshold values, immediate tilt predictions can guide adjustments to mooring lines or panel orientations. Moreover, the approach is beneficial when considering multiple potential locations for deployment of the same platform design; by inputting site-specific wind and wave data, stakeholders can quickly gauge performance and feasibility across different regions without resorting to additional high-fidelity simulations. Nevertheless, any substantial change to the platform's geometry would require re-calibration of the model with updated training data. In addition, FE-FSI simulations remain indispensable for final validation, particularly under extreme or high-risk scenarios.

Another point of discussion is the performance of the CNN model. The CNN model exhibited excellent performance for small to large amplitude tilts, with near-perfect predictions in many cases. It demonstrated that CNN models can be highly accurate while maintaining computational efficiency. However, a key limitation lies in the input: simulating a sea state requires both spectral density values and phase shifts. Since the CNN model only used spectral density values, it was limited to a single set of phase shifts and thus generalized poorly across different sea states. Generalizing the CNN model to account for all possible sea states would require an impractically large dataset and significant memory resources. Given the feasibility constraints and the scope of this research, it was not possible to include such a range of inputs. Consequently, despite its accuracy under specific conditions, the CNN model lacked the reliability required to be selected as the final model.

The LSTM model, on the other hand, used wave elevation data as input and demonstrated a reasonable ability to capture temporal patterns over time. It predicted tilt responses for 50 locations over 3400 seconds using only two wave elevations per second as input. Although the overall predictions were not

highly accurate, the model demonstrated good pattern recognition capabilities. Increasing the number of wave elevation inputs could potentially improve accuracy. However, literature on deep learning indicates that adding more input data does not always enhance performance, and further research would be necessary to validate this hypothesis.

Regarding the input data used for both the CNN and the CNN-LSTM hybrid models, the number of frequency bins (N_{freq_bins}) was set to 300. This choice was reasonable, as it captured approximately 99% of the spectral density values for most sea states. However, some samples likely contained significant spectral density values beyond 3 Hz, corresponding to very small waves. These samples, where inputs may have been padded with zeros for the majority of the frequency range, could have negatively impacted the models' ability to generalize for very small amplitude tilts.

The computational constraints faced during this research represent another key limitation. Since all models were developed and trained on Google Colab, memory and GPU RAM restrictions limited the amount of data that could be used. Although a full year of data (8760 samples) was initially available, out-of-memory errors necessitated reducing the dataset to 3000 samples. Consequently, it remains unclear whether model performance could be improved with the full dataset. Future research could explore training the models on larger datasets using more powerful hardware to determine whether performance gains can be achieved.

Lastly, an iterative trial-and-error approach is employed in the methodology to search for optimal hyperparameters. This approach was adopted to gain better insight into the behavior of different hyperparameters within a deep learning environment. It indeed enhanced the understanding of how specific hyperparameters influence either the accuracy or computational efficiency of the model. However, the trial-and-error method is time-consuming and may overlook optimal configurations that are not immediately apparent. Additionally, the optimal configuration is not derived from quantifiable metrics, rendering the approach less scientific. Although this method provides a better understanding of how hyperparameters function, it is only recommended for those conducting deep learning research for the first time. This is because it offers valuable insights into hyperparameter sensitivity, which are useful as preliminary values, rather than selecting arbitrary initial values.

8

Recommendations for Future Work

This thesis highlights several opportunities for future research, many of which were beyond the scope of this study due to time constraints and limited computational resources. Below, key recommendations are provided to extend the research line and further improve the development of surrogate deep learning models for OFPV systems.

8.1. Expanding Input Cases

In Table 3.1 of Chapter 3, three different input cases were proposed for training the deep learning models. However, this thesis only explored Case 1: Base Case, where a single floater with constant stiffness was considered. The remaining cases—Case 2: *Multiple Floaters* and Case 3: *Multiple Floaters + Varying Stiffness*—represent more realistic scenarios for real-world OFPVs. Incorporating these additional cases into future models would improve the accuracy and practicality of the surrogate models by allowing for variability in stiffness and the interaction of multiple floaters.

Implementing these cases would enable the design of OFPVs to be optimized based on customized objective functions, such as minimizing tilt or maximizing power output across multiple floaters. However, such implementations would require significantly more computational resources, as the models would need to generalize across a much larger and more complex dataset.

8.2. Generalizing the CNN Model

Although the CNN model demonstrated excellent predictive performance with low error metrics, its applicability is currently limited because it was trained on a single set of phase shifts. As a result, it cannot generalize across all possible sea states. Future research should investigate methods to incorporate phase shifts into the model.

One potential approach is to apply a probability density function around the phase shift parameter, quantifying the likelihood of different sea states. By incorporating such a probability distribution, the CNN model could be adapted to account for a broader range of sea states, improving its generalization and reliability. Further exploration of this method could determine how often specific sea states occur and assess the CNN model's reliability under different conditions.

8.3. Incorporating Physics-Based Approaches

Currently, the deep learning models developed in this thesis are purely data-driven. However, the input and target data used for training were generated using finite element-fluid structure interaction (FE-FSI) simulations, which are based on well-established physical equations and boundary conditions.

Future research could explore the use of physics-informed machine learning, where known physical laws are embedded into the learning process. Such hybrid models may enhance performance, particularly in scenarios where purely data-driven models struggle, such as predicting small amplitude tilts. By

incorporating FSI principles directly into the neural network architecture or loss function, these models could potentially offer better generalization and reliability across extreme or underrepresented cases.

8.4. Fatigue and Long-Term Performance

Another important area for future research is the inclusion of fatigue analysis in the performance assessment of OFPVs. While Alcañiz et al. focused on minimizing mismatch losses and this thesis aimed to determine optimal power output locations, both approaches assume that minimizing motion is ideal. However, simply designing an OFPV with one floater and high stiffness, which reduces motion, may negatively impact the system's fatigue life and material usage.

OFPVs are subject to large dynamic forces, and repeated loading cycles can lead to structural fatigue and eventual failure. Therefore, once Case 2: *Multiple Floaters* and Case 3: *Multiple Floaters + Varying Stiffness* are implemented, future research should incorporate fatigue as a parameter in the objective function. This approach would enable the development of designs that balance minimal motion and extended structural lifespan.

References

[1] Alcañiz A. and Agarwal S. “Structural analysis and power losses in floating solar platform in offshore environment”. In: (2024).

[2] International Energy Agency. “Offshore Wind Outlook 2019”. In: *World Energy Outlook Special Report* (2019).

[3] “An identification method of floating wind turbine tower responses using deep learning technology in the monitoring system”. In: *Ocean Engineering* 261 (2022), p. 112105. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2022.112105. URL: https://www.sciencedirect.com/science/article/pii/S0029801822014287.

[4] Xingxian Bao et al. “One-dimensional convolutional neural network for damage detection of jacket-type offshore platforms”. In: *Ocean Engineering* 219 (2021), p. 108293. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2020.108293. URL: https://www.sciencedirect.com/science/article/pii/S0029801820312099.

[5] L.M.M. Bos and Benjamin Sanderse. *Uncertainty quantification for wind energy applications - Literature review*. Aug. 2017.

[6] Mikel Canizo et al. “Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study”. In: *Neurocomputing* 363 (2019), pp. 246–260. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.07.034. URL: https://www.sciencedirect.com/science/article/pii/S0925231219309877.

[7] Chateau et al. “Mathematical modeling suggests high potential for the deployment of floating photovoltaic on fish ponds”. In: *Science of The Total Environment* (2019). DOI: 10.1016/j.scitotenv.2019.05.420.

[8] European Comission. *Renewable energy targets*. 2024. URL: https://energy.ec.europa.eu/topics/renewable-energy/renewable-energy-directive-targets-and-rules/renewable-energy-targets_en.

[9] John A Duffie, William A Beckman, and Nathan Blair. *Solar engineering of thermal processes, photovoltaics and wind*. John Wiley & Sons, 2020.

[10] Sara Golroodbari and Wilfried van Sark. “On the effect of dynamic albedo on performance modelling of offshore floating photovoltaic systems”. In: *Solar Energy Advances* 2 (2022), p. 100016. ISSN: 2667-1131. DOI: https://doi.org/10.1016/j.seja.2022.100016. URL: https://www.sciencedirect.com/science/article/pii/S2667113122000043.

[11] Golroodbari, S Zahra, and et al. “Pooling the cable: A techno-economic feasibility study of integrating offshore floating photovoltaic solar technology within an offshore wind park”. In: *Solar Energy* (2021). DOI: 10.1016/j.solener.2020.12.062.

[12] Golroodbari et al. “Simulation of performance differences between offshore and land□based photovoltaic systems”. In: *Progress in Photovoltaics* (2020). DOI: 10.1002/pip.3276.

[13] S Gracia et al. “Improving accuracy on wave height estimation through machine learning techniques”. In: *Ocean Engineering* 236 (2021), p. 108699. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2021.108699. URL: https://www.sciencedirect.com/science/article/pii/S0029801821001347.

[14] Soukissian Takvor H., Karathanasi Flora E., and Zaragkas Dimitrios K. “Exploiting offshore wind and solar resources in the Mediterranean using ERA5 reanalysis data”. In: *Energy Conversion and Management* 237 (2021), p. 114092. ISSN: 0196-8904. DOI: https://doi.org/10.1016/j.enconman.2021.114092. URL: https://www.sciencedirect.com/science/article/pii/S0196890421002685.

[15] L. H. Holthuijsen. *Waves in oceanic and coastal waters*. Cambridge University Press, 2007. DOI: https://doi.org/10.1017/CBO9780511618536.

[16] K Janas, I A Milne, and J R Whelan. “Application of a convolutional neural network for mooring failure identification”. In: *Ocean Engineering* 232 (2021), p. 109119. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2021.109119. URL: https://www.sciencedirect.com/science/article/pii/S0029801821005539.

[17] Toshiki Kawai et al. “Sea state estimation using monitoring data by convolutional neural network (CNN)”. In: *Journal of Marine Science and Technology* 26.3 (2021), pp. 947–962. ISSN: 1437-8213. DOI: 10.1007/s00773-020-00785-8. URL: https://doi.org/10.1007/s00773-020-00785-8.

[18] Rodney Kizito et al. “Long Short-Term Memory Networks for Facility Infrastructure Failure and Remaining Useful Life Prediction”. In: *IEEE Access* PP (May 2021), pp. 1–1. DOI: 10.1109/ACCESS.2021.3077192.

[19] Do-Soo Kwon, Chungkuk Jin, and MooHyun Kim. “Prediction of dynamic and structural responses of submerged floating tunnel using artificial neural network and minimum sensors”. In: *Ocean Engineering* 244 (2022), p. 110402. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2021.110402. URL: https://www.sciencedirect.com/science/article/pii/S0029801821016929.

[20] Yaguo Lei. “3 - Individual intelligent method-based fault diagnosis”. In: *Intelligent Fault Diagnosis and Remaining Useful Life Prediction of Rotating Machinery*. Ed. by Yaguo Lei. Butterworth-Heinemann, 2017, pp. 67–174. ISBN: 978-0-12-811534-3. DOI: https://doi.org/10.1016/B978-0-12-811534-3.00003-2. URL: https://www.sciencedirect.com/science/article/pii/B9780128115343000032.

[21] Binbin Li. “Operability study of walk-to-work for floating wind turbine and service operation vessel in the time domain”. In: *Ocean Engineering* 220 (2021), p. 108397. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2020.108397. URL: https://www.sciencedirect.com/science/article/pii/S0029801820313044.

[22] Ming-Wei Li et al. “A hybrid approach for forecasting ship motion using CNN–GRU–AM and GCWOA”. In: *Applied Soft Computing* 114 (2022), p. 108084. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2021.108084. URL: https://www.sciencedirect.com/science/article/pii/S1568494621009832.

[23] Juan Montaño et al. “Using the R-MAPE index as a resistant measure of forecast accuracy”. In: *Psicothema* 25 (Nov. 2013), pp. 500–506. DOI: 10.7334/psicothema2013.23.

[24] Colomés O., Verdugo F., and Akkerman I. “A monolithic finite element formulation for the hydroelastic analysis of very large floating structures”. In: *International Journal for Numerical Methods in Engineering* (2022). DOI: 10.1002/nme.7140.

[25] Sara Oliveira-Pinto and Jasper Stokkermans. “Assessment of the potential of different floating solar technologies – Overview and analysis of different case studies”. In: *Energy Conversion and Management* 211 (2020), p. 112747. ISSN: 0196-8904. DOI: https://doi.org/10.1016/j.enconman.2020.112747. URL: https://www.sciencedirect.com/science/article/pii/S0196890420302855.

[26] Francisco Javier Ordóñez and Daniel Roggen. “Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition”. In: *Sensors* 16.1 (2016). ISSN: 1424-8220. DOI: 10.3390/s16010115. URL: https://www.mdpi.com/1424-8220/16/1/115.

[27] Han Rui et al. “SlimML: Removing Non-Critical Input Data in Large-Scale Iterative Machine Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* (2021). DOI: 10.1109/TKDE.2019.2951388.

[28] Smriti Sharma and Vincenzo Nava. “Condition monitoring of mooring systems for Floating Offshore Wind Turbines using Convolutional Neural Network framework coupled with Autoregressive coefficients”. In: *Ocean Engineering* 302 (2024), p. 117650. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2024.117650. URL: https://www.sciencedirect.com/science/article/pii/S0029801824009879.

[29] Wei Shi et al. “Short-term motion prediction of floating offshore wind turbine based on muti-input LSTM neural network”. In: *Ocean Engineering* 280 (2023), p. 114558. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2023.114558. URL: https://www.sciencedirect.com/science/article/pii/S0029801823009423.

[30] Ying Tian et al. “Effect of Wave Conditions on Offshore Floating Photovoltaic Power Generation”. In: (2023), pp. 326–330. DOI: 10.1109/REPE59476.2023.10511473.

[31] Yuchao Wang et al. “Ship Roll Prediction Algorithm Based on Bi-LSTM-TPA Combined Model”. In: *Journal of Marine Science and Engineering* 9.4 (2021). ISSN: 2077-1312. DOI: 10.3390/jmse9040387. URL: https://www.mdpi.com/2077-1312/9/4/387.

[32] Costoya X et al. “Combining offshore wind and solar photovoltaic energy to stabilize energy supply under climate change scenarios: A case study on the western Iberian Peninsula”. In: *Renewable and Sustainable Energy Reviews* 157 (2022), p. 112037. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2021.112037. URL: https://www.sciencedirect.com/science/article/pii/S1364032121012995.

[33] Min Zhang and Sebastian Schreier. “Review of wave interaction with continuous flexible floating structures”. In: *Ocean Engineering* 264 (2022), p. 112404. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2022.112404. URL: https://www.sciencedirect.com/science/article/pii/S0029801822016870.

A

Dataset

In this appendix detailed information on the dataset is given. In Figure A.1 and A.2 the distribution of the input data is given for each month. In Figure A.3 the distribution of the standard deviation of the target data is given for each month.

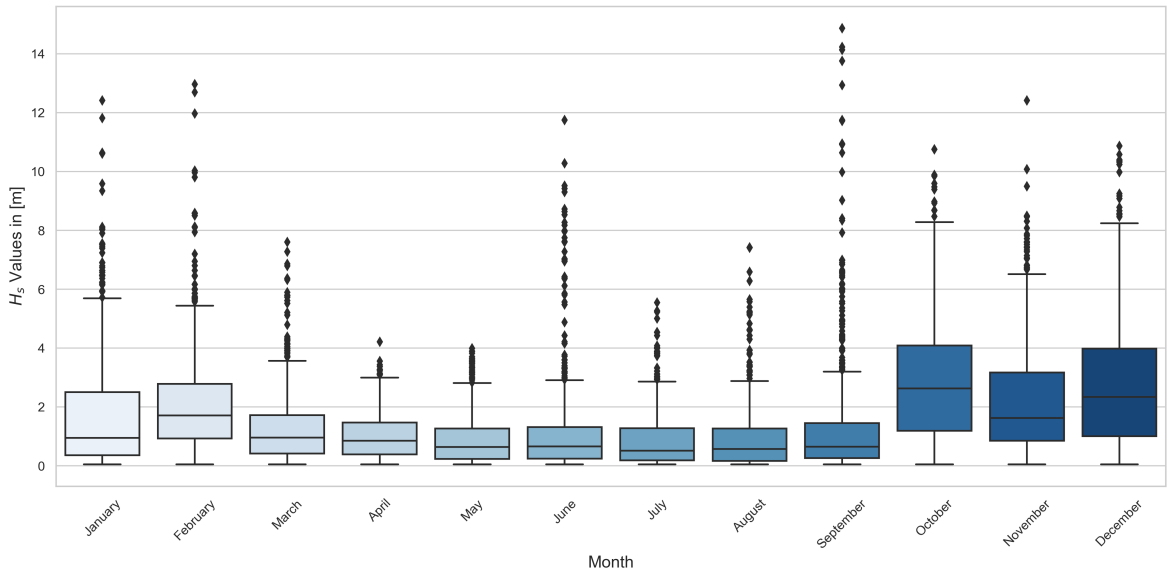


Figure A.1: Boxplot of hourly significant wave height values for each month.

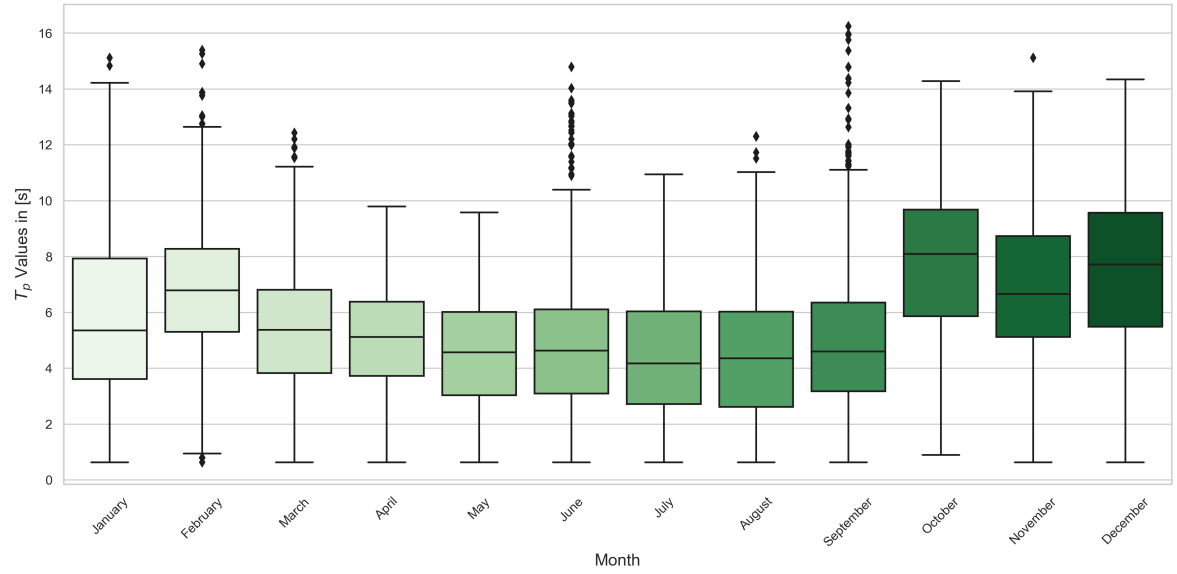


Figure A.2: Boxplot of hourly peak period values for each month.

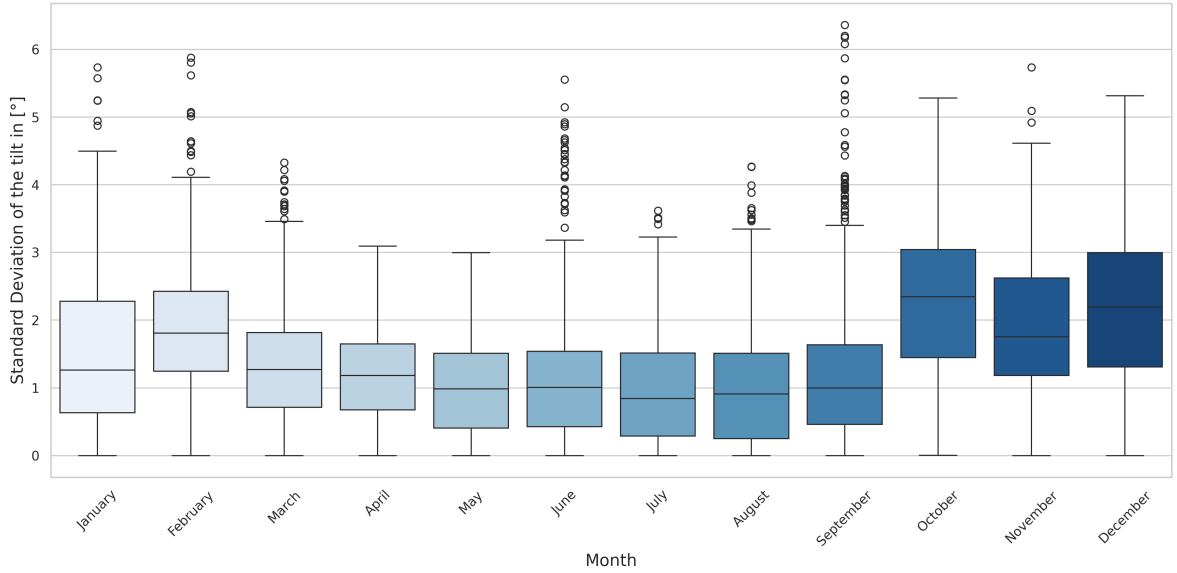


Figure A.3: Boxplot of the standard deviation of the tilt for each month.

B

First-order Taylor Expansion

To effectively assess the performance of the deep learning models, it is crucial to understand how errors in tilt angle prediction translate into errors in power output prediction. In this appendix, a first-order Taylor expansion is employed to analyze the sensitivity of the tilt angle (φ) with respect to the power output (P_{DC}). The formulas presented in Section 1.2 are utilized to demonstrate this relationship through a numerical example, thereby validating the conclusions drawn from the Taylor expansion. For clarity, the tilt angle is denoted as β, as in the formulas, although it is referred to as φ from Chapter 2 onward.

Let $P_{\mathsf{DC}}(\beta)$ be the power output as a function of tilt angle β. A first-order Taylor expansion around a reference tilt angle β_0 states:

$$P_{\mathsf{DC}}(\beta_0 + \Delta\beta) \;\approx\; P_{\mathsf{DC}}(\beta_0) \;+\; \left.\frac{d\,P_{\mathsf{DC}}}{d\beta}\right|_{\beta=\beta_0} \Delta\beta, \tag{B.1}$$

From Equation (B.1), the change in power, ΔP_{DC}, is:

$$\Delta P_{\mathsf{DC}} \;\approx\; \left.\frac{d\,P_{\mathsf{DC}}}{d\beta}\right|_{\beta=\beta_0} \Delta\beta. \tag{B.2}$$

Let's assume:

- $\beta_0 = 10^\circ$ (reference tilt angle),
- A small deviation of $\Delta\beta = +2^\circ$,
- A nominal direct current power output $P_{\mathsf{DC}}(\beta_0) = 313\,\mathrm{W}$ at $\beta_0 = 10^\circ$.

In PV installations, the optimal tilt angle is not a sharp peak but sits on a broad plateau of near-optimal performances. Each component of the total irradiance is a smooth function of β and does not change abruptly around typical operating angles. Lets suppose a (hypothetical) derivative, that is reasonable small:

$$\left.\frac{d\,P_{\mathsf{DC}}}{d\beta}\right|_{\beta=10^\circ} = -0.5\,\mathrm{W}/^\circ.$$

This means that for each additional degree, the power reduces by 0.5 W. Using Equation (B.2), the change in power can be approximated:

$$\Delta P_{\mathsf{DC}} \;\approx\; (-0.5\,\mathrm{W}/^\circ) \times (2^\circ) \;=\; -1.0\,\mathrm{W}.$$

Hence, the new power at $\beta_0 + \Delta\beta = 12^\circ$ is approximately:

$$P_{\mathsf{DC}}(12^\circ) \;\approx\; 313\,\mathrm{W} \;+\; (-1.0)\,\mathrm{W} \;=\; 312\,\mathrm{W}.$$

In this simple example, a 2-degree error in the tilt angle reduces the power by only 1 W, indicating that minor errors in predicting β lead to proportionally minor errors in P_{DC}. This mostly depends on the magnitude of the derivative, let's confirm that the derivative is small with a numerical example.

First, let's define some arbitrarily chosen constants and inputs:

- Direct Normal Irradiance (DNI): $I_{\text{DNI}} = 800\,\text{W}\,\text{m}^{-2}$
- Diffuse Horizontal Irradiance (DHI): $I_{\text{DHI}} = 100\,\text{W}\,\text{m}^{-2}$
- Global Horizontal Irradiance (GHI): $I_{\text{GHI}} = 900\,\text{W}\,\text{m}^{-2}$
- $\rho = 0.06$ (typical for water)
- PV Module Area: $A_{\text{PV}} = 2\,\text{m}^2$
- Module Efficiency: $\eta_{\text{mod}} = 0.18$ (i.e., 18%)
- Nominal (reference) tilt angle: $\beta_0 = 10^\circ$
- A small increment for the difference quotient: $\Delta\beta = +1^\circ$

For the direct beam term, the angle of incidence, θ_i, is needed. For demonstration purposes, the effective angle of incidence is assumed to be $\theta_{i,0} = 15^\circ$ at $\beta_0 = 10^\circ$ (In reality, θ_i depends on solar geometry).

As shown in Section 1.2, the total irradiance consists of:

$$G_{\text{total}}(\beta) = G_{\text{beam}}(\beta) + G_{\text{diffuse}}(\beta) + G_{\text{reflected}}(\beta).$$

The direct beam irradiance can be calculated as following.

$$G_{\text{beam}}(\beta_0) = I_{\text{DNI}} \times \cos(\theta_{i,0}).$$

For our assumed values,

$$I_{\text{DNI}} = 800\,\text{W}\,\text{m}^{-2}, \quad \theta_{i,0} = 15^\circ.$$

Hence,

$$\cos(15^\circ) \approx 0.9659,$$
$$G_{\text{beam}}(10^\circ) = 800 \times 0.9659 \approx 773\,\text{W}\,\text{m}^{-2}.$$

Assuming the Liu–Jordan isotropic model, the diffuse irradiance is given as:

$$G_{\text{diffuse}}(\beta_0) = I_{\text{DHI}} \times \frac{1 + \cos(\beta_0)}{2}.$$

Here,

$$I_{\text{DHI}} = 100\,\text{W}\,\text{m}^{-2}, \quad \beta_0 = 10^\circ.$$

So

$$\cos(10^\circ) \approx 0.9848,$$
$$\frac{1 + \cos(10^\circ)}{2} = \frac{1 + 0.9848}{2} \approx 0.9924.$$

Thus,

$$G_{\text{diffuse}}(10^\circ) = 100 \times 0.9924 \approx 99.24\,\text{W}\,\text{m}^{-2}.$$

The reflected irradiance can be calculated as following:

$$G_{\text{reflected}}(\beta_0) = I_{\text{GHI}} \times \rho \times \frac{1 - \cos(\beta_0)}{2}.$$

With:

$$I_{\text{GHI}} = 900\,\text{W}\,\text{m}^{-2}, \quad \rho = 0.06, \quad \beta_0 = 10^\circ.$$

So,

$$1 - \cos(10^\circ) = 1 - 0.9848 = 0.0152,$$

$$\frac{0.0152}{2} = 0.0076.$$

Hence,

$$G_{\text{reflected}}(10^\circ) = 900 \times 0.06 \times 0.0076 \approx 0.41\,\text{W m}^{-2}.$$

Summing the components:

$$G_{\text{total}}(10^\circ) = G_{\text{beam}}(10^\circ) + G_{\text{diffuse}}(10^\circ) + G_{\text{reflected}}(10^\circ).$$

Numerically,

$$G_{\text{beam}}(10^\circ) \approx 773, \quad G_{\text{diffuse}}(10^\circ) \approx 99.24, \quad G_{\text{reflected}}(10^\circ) \approx 0.41.$$

Therefore,

$$G_{\text{total}}(10^\circ) \approx 773 + 99.24 + 0.41 = 872.65\,\text{W m}^{-2}.$$

Now the DC Power at $\beta_0 = 10^\circ$ can be calculated using:

$$P_{\text{DC}}(\beta) = A_{\text{PV}} \times \eta_{\text{mod}} \times G_{\text{total}}(\beta).$$

The P_{DC}, at $\beta_0 = 10^\circ$ is given as:

$$P_{\text{DC}}(10^\circ) = 2 \times 0.18 \times 872.65 \approx 314.15\,\text{W}.$$

Now, consider a small increment:

$$\Delta\beta = +1^\circ \quad \text{(from } 10^\circ \text{ to } 11^\circ).$$

The same steps are repeated to find $G_{\text{beam}}(11^\circ), G_{\text{diffuse}}(11^\circ), G_{\text{reflected}}(11^\circ)$, and hence $P_{\text{DC}}(11^\circ)$.

Assume θ_i goes from 15° to say 15.5° for a 1-degree tilt change (a moderate assumption), the direct beam irradiance is given as:

$$G_{\text{beam}}(11^\circ) = 800 \times \cos(15.5^\circ).$$

$\cos(15.5^\circ) \approx 0.9613$, so

$$G_{\text{beam}}(11^\circ) \approx 800 \times 0.9613 = 769\,\text{W m}^{-2}.$$

The diffuse irradiance is calculated as:

$$G_{\text{diffuse}}(11^\circ) = 100 \times \frac{1 + \cos(11^\circ)}{2}.$$

$\cos(11^\circ) \approx 0.9816$, so

$$\frac{1 + 0.9816}{2} = 0.9908,$$

$$G_{\text{diffuse}}(11^\circ) \approx 100 \times 0.9908 = 99.08\,\text{W m}^{-2}.$$

Finally, the reflected irradiance is calculated as:

$$G_{\text{reflected}}(11^\circ) = 900 \times 0.06 \times \frac{1 - \cos(11^\circ)}{2}.$$

$$1 - \cos(11^\circ) = 1 - 0.9816 = 0.0184,$$

$$0.0184/2 = 0.0092,$$

$$G_{\text{reflected}}(11^\circ) \approx 900 \times 0.06 \times 0.0092 = 900 \times 0.000552 = 0.50\,\text{W m}^{-2}.$$

Summing the irradiance components:

$$G_{\text{total}}(11^\circ) = 769 + 99.08 + 0.50 = 868.58\,\mathrm{W\,m^{-2}}.$$

This leads to a DC Power output at 11° of:

$$P_{\text{DC}}(11^\circ) = 2 \times 0.18 \times 868.58 \approx 312.69\,\mathrm{W}.$$

The derivative around $\beta = 10^\circ$ can now be approximated:

$$\left.\frac{d\,P_{\text{DC}}}{d\beta}\right|_{\beta=10^\circ} \approx \frac{312.69 - 314.15}{1^\circ} = \frac{-1.46}{1} = -1.46\,\mathrm{W\,^\circ{}^{-1}}.$$

This means the slope near $\beta = 10^\circ$ is about $-1.46\,\mathrm{W}/^\circ$ (This negative sign indicates that increasing tilt from 10° to 11° slightly decreases the power).

Hence, we see that a small error of $+1^\circ$ in tilt yields a change of about $-1.46\,\mathrm{W}$, or $\approx 0.46\%$ relative to the original $314.15\,\mathrm{W}$. This confirms the notion that $\frac{dP_{\text{DC}}}{d\beta}$ (the slope) is modest, so minor errors in β translate into minor errors in P_{DC}. In practical terms, this supports the argument that a deep learning model predicting β with a small error will, in turn, produce only a small error in the predicted power output.

C

Visualization of the Models

This appendix presents visualizations of the three developed deep learning models: the CNN, the LSTM, and the CNN-LSTM hybrid (shown over two pages).

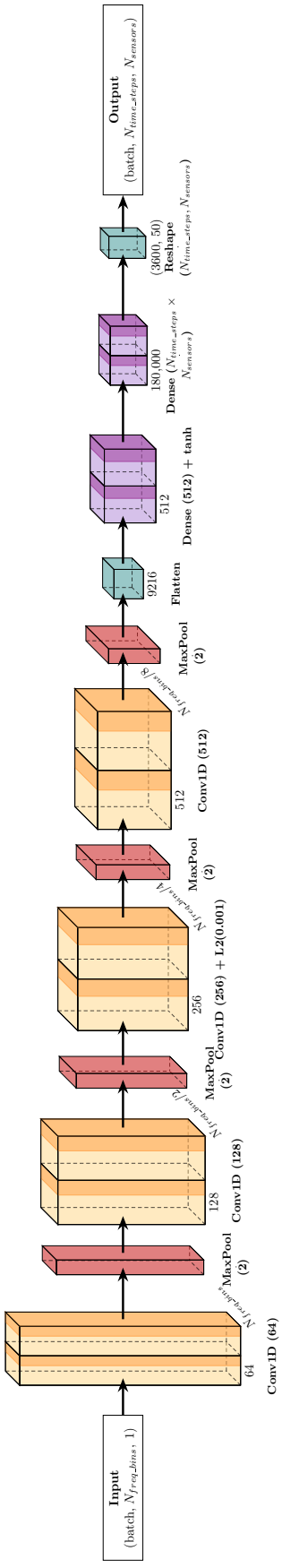


Figure C.1: CNN Architecture Visualization.

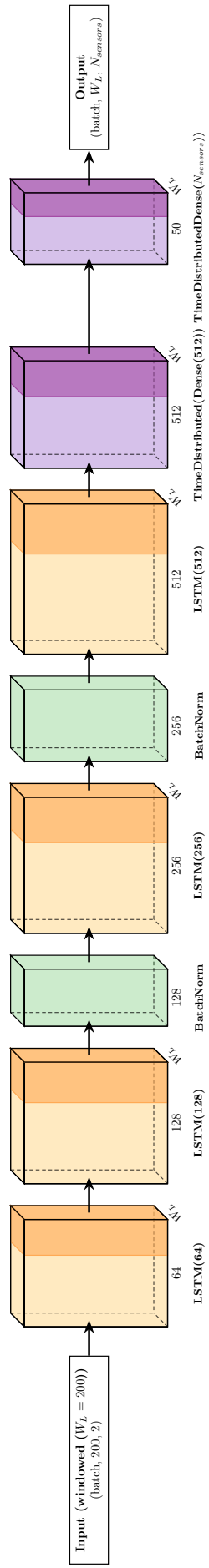


Figure C.2: LSTM Architecture Visualization.

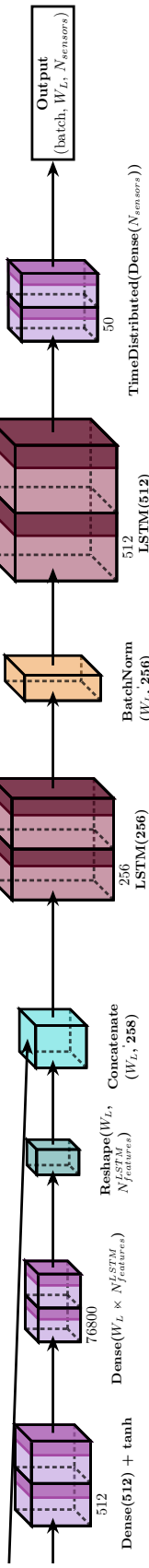


Figure C.3: CNN-LSTM hybrid Architecture Visualization (1).

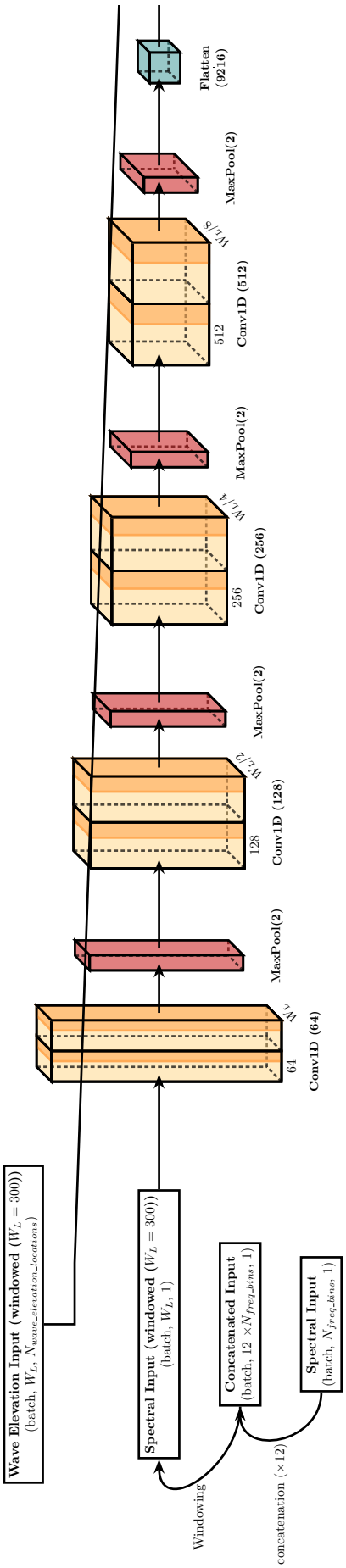


Figure C.4: CNN-LSTM hybrid Architecture Visualization (2).

D

Results

In this Appendix the results are given for the different architecture models. In Appendix D.1 the results of the CNN architecture are given, in Appendix D.2 the results of the LSTM architecture are given, and in Appendix D.3 the results of the Hybrid model are given.

D.1. CNN Results

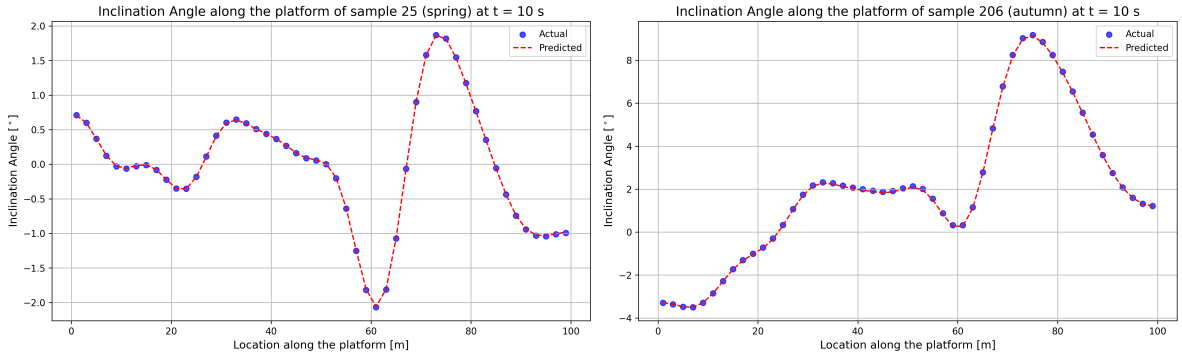


Figure D.1: The predicted and actual values (CNN) of the inclination angle φ along the floating platform at time instance t = 10 s for a random sample in spring (left) and autumn (right).

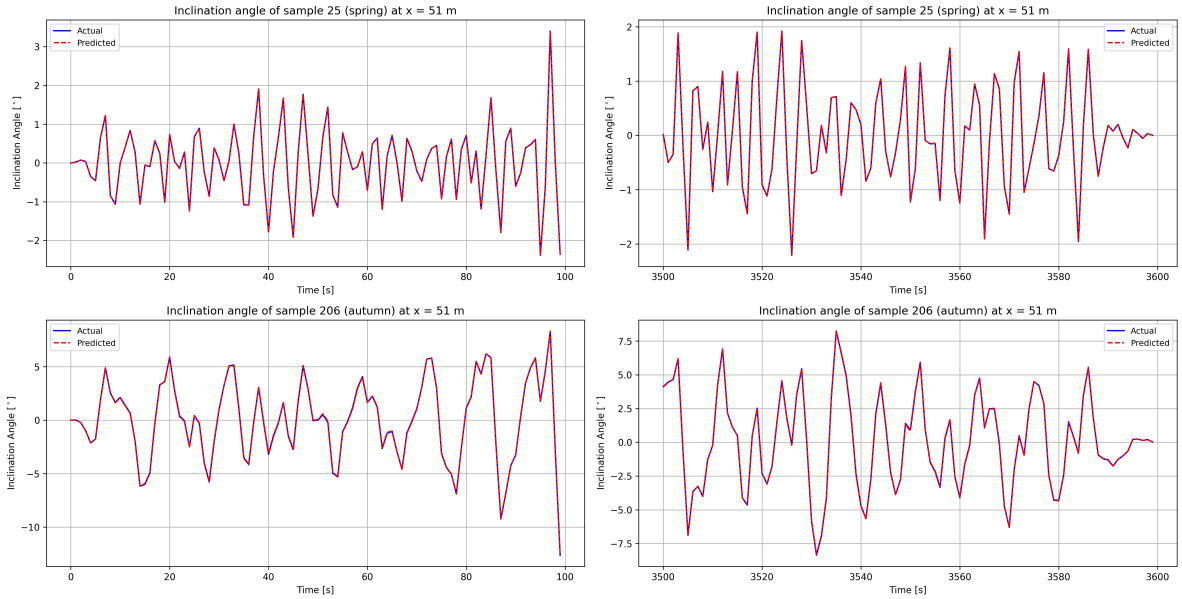


Figure D.2: Predicted and actual values (CNN) of the tilt φ over the first and last 100 s of a storm at x = 51 m for a sample drawn from spring (top) and autumn (bottom).

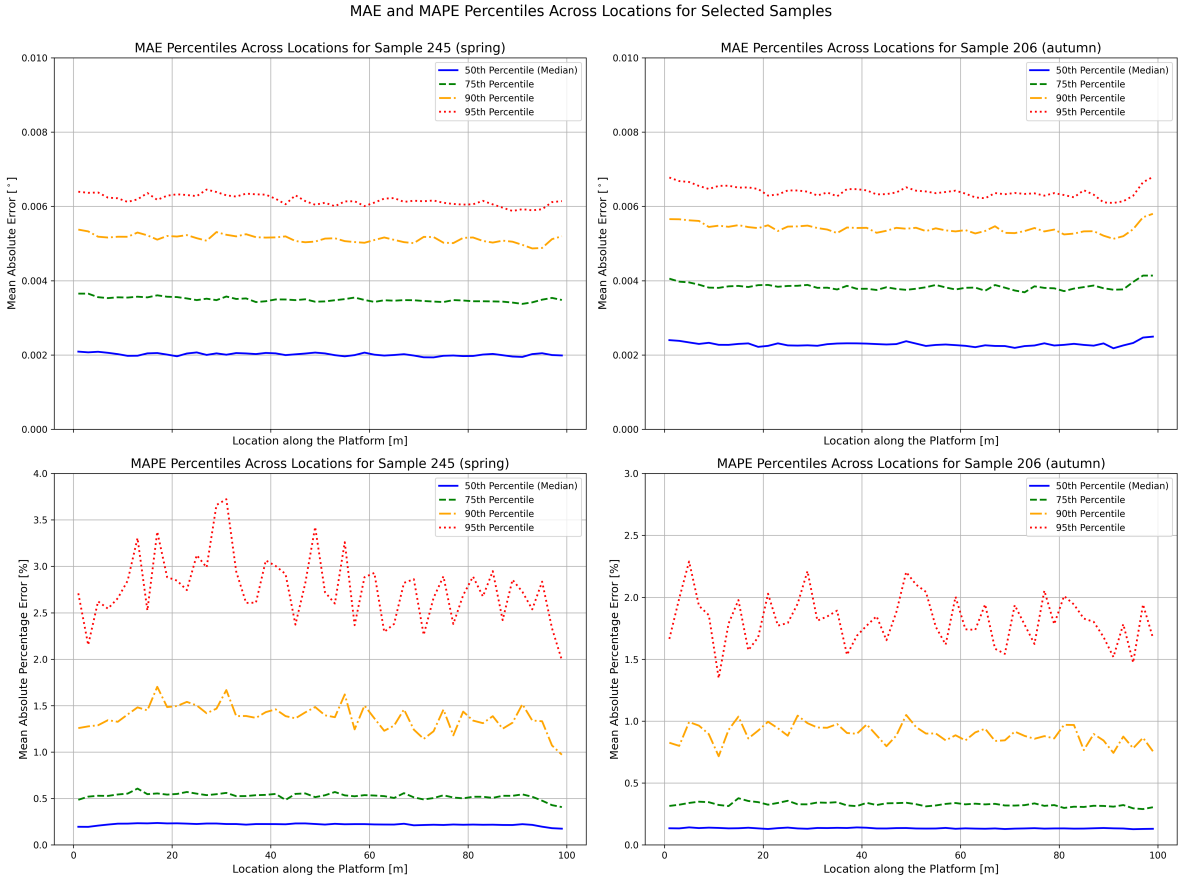


Figure D.3: The mean absolute error, and mean absolute percentage error 50th, 75th, 90th, and 95th percentile (CNN) in time for every location along the floating structure for a sample drawn from spring (left) and autumn (right).

D.2. LSTM Results

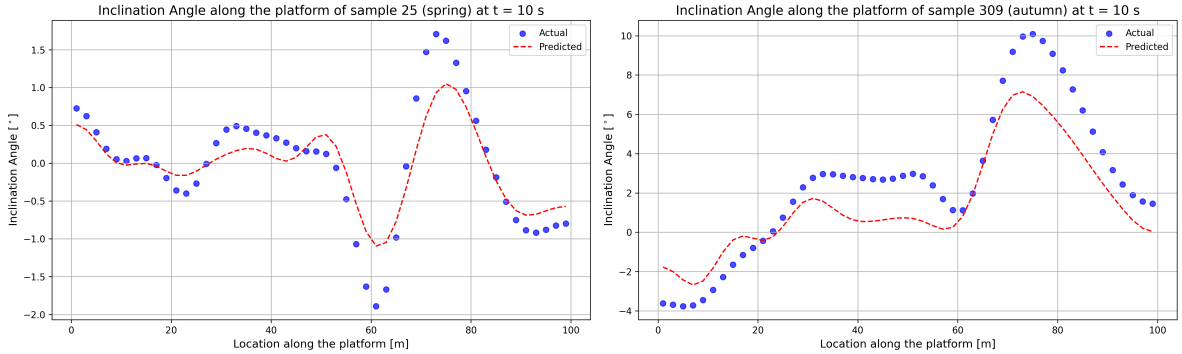


Figure D.4: The predicted and actual values (LSTM) of the inclination angle φ along the floating platform at time instance t = 10 s for a random sample in spring (left) and autumn (right).

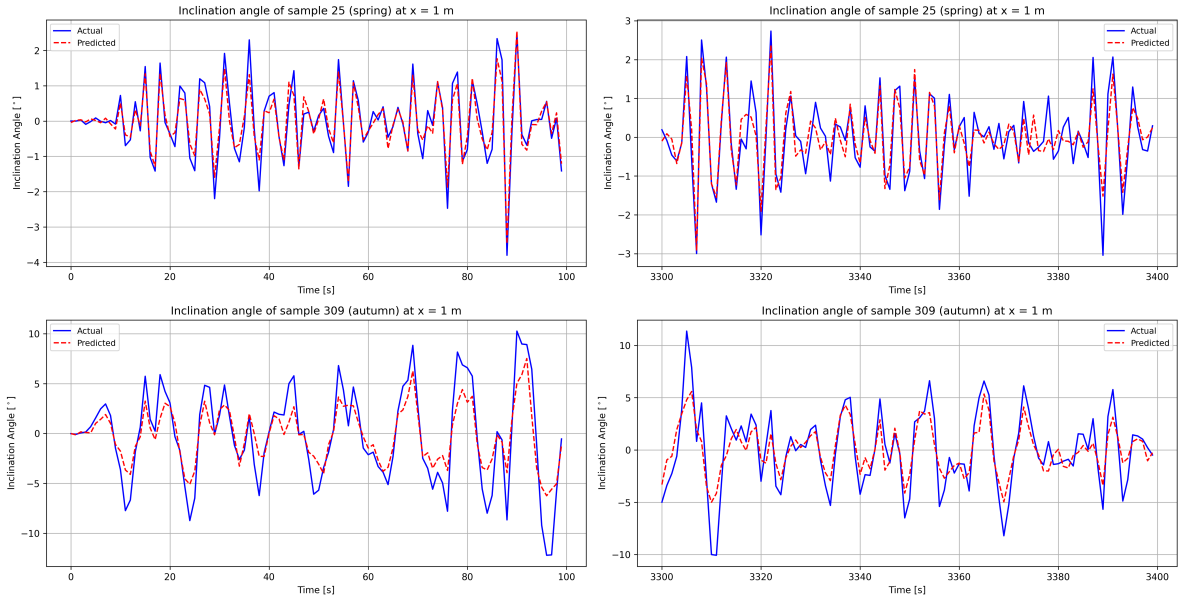


Figure D.5: Predicted and actual values (LSTM) of the tilt φ over t = 0 - 100 s and t = 3300 - 3400 s of a storm at x = 1 m for a sample drawn from spring (top) and autumn (bottom).

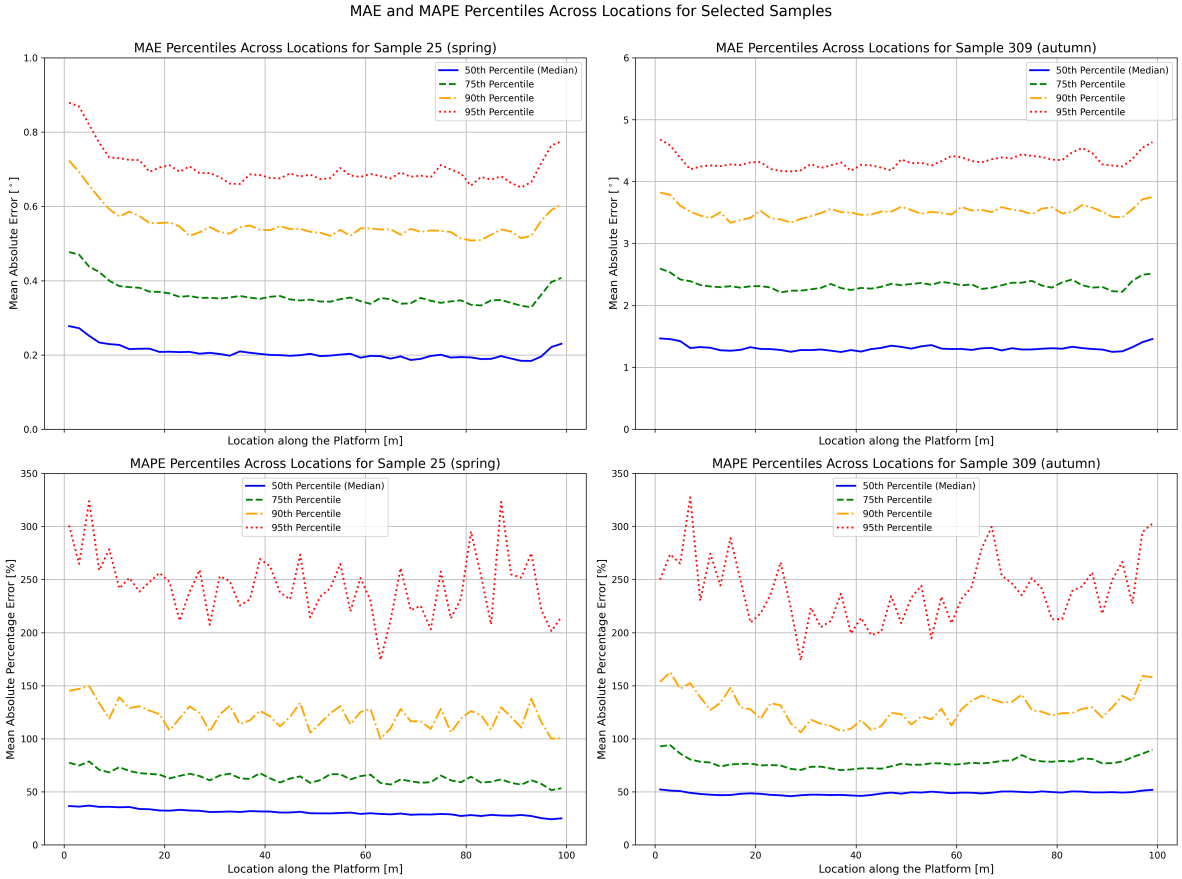


Figure D.6: The mean absolute error, and mean absolute percentage error 50th, 75th, 90th, and 95th percentile (LSTM) in time for every location along the floating structure for a sample drawn from spring (left) and autumn (right).

D.3. CNN - LSTM Hybrid Results

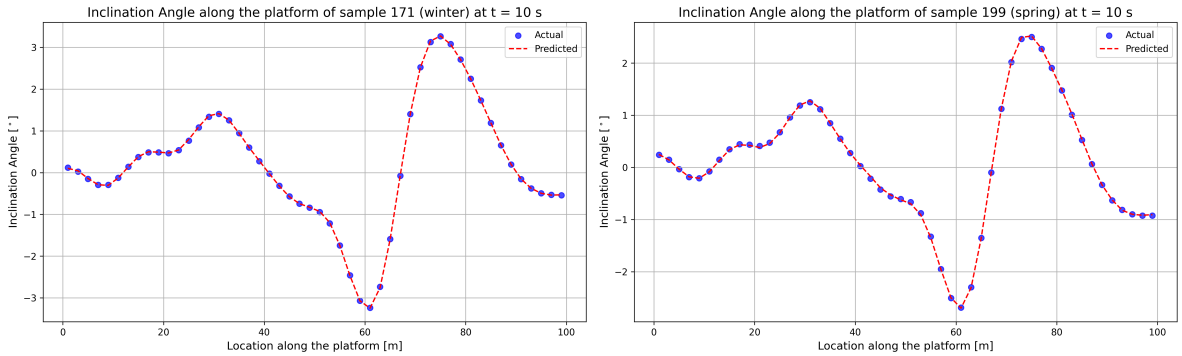


Figure D.7: The predicted and actual values (CNN-LSTM hybrid) of the inclination angle φ along the floating platform at time instance t = 10 s for a random sample in winter (left) and spring (right).

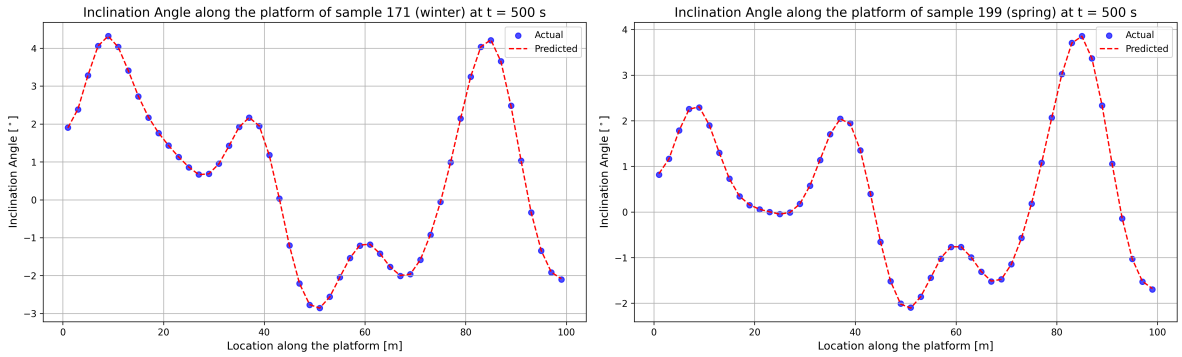


Figure D.8: The predicted and actual values (CNN-LSTM hybrid) of the inclination angle φ along the floating platform at time instance t = 500 s for a random sample in winter (left) and spring (right).

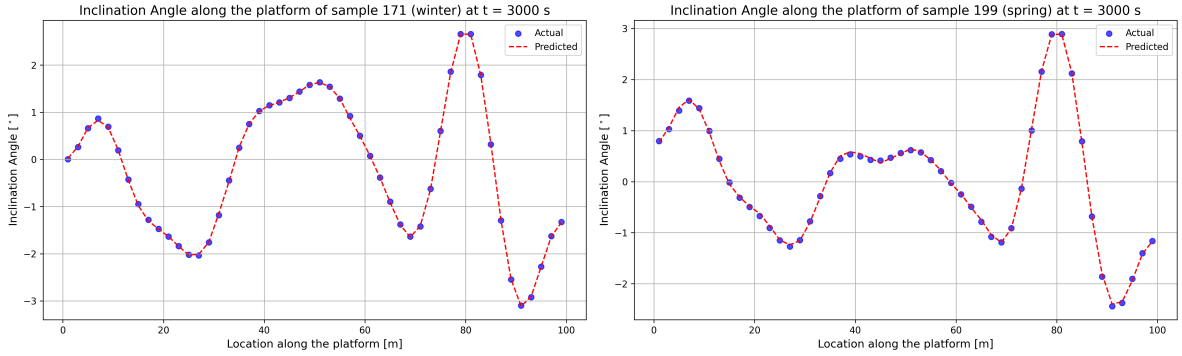


Figure D.9: The predicted and actual values (CNN-LSTM hybrid) of the inclination angle φ along the floating platform at time instance t = 3000 s for a random sample in winter (left) and spring (right).

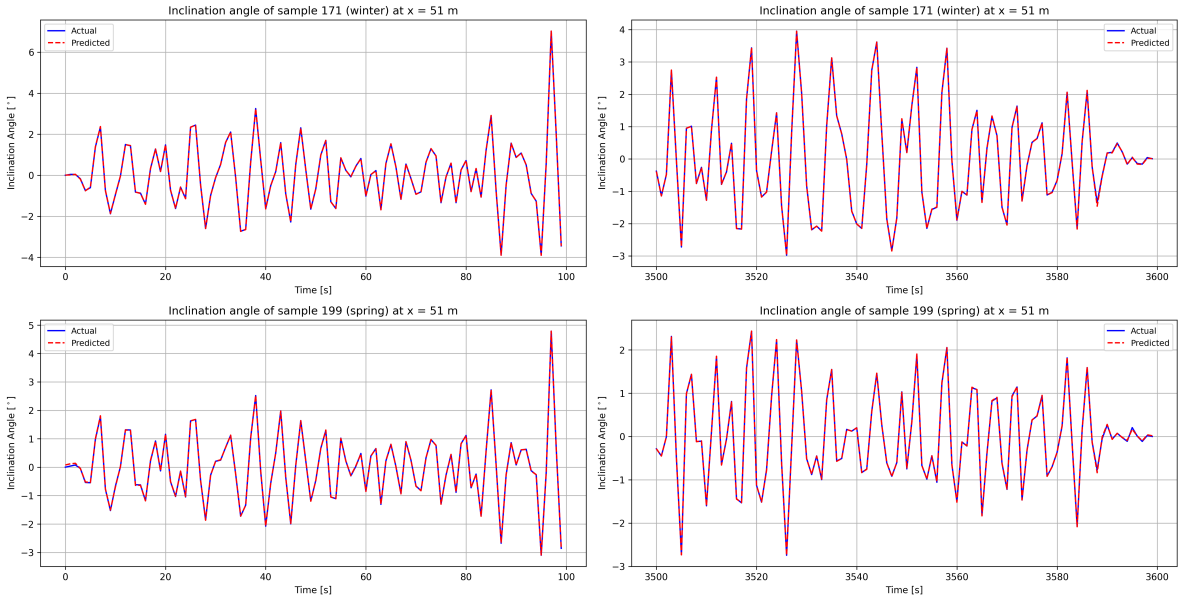


Figure D.10: Predicted and actual values (CNN-LSTM hybrid) of the tilt φ over t = 0 - 100 s and t = 3500 - 3600 s of a storm at x = 1 m for a sample drawn from spring (top) and autumn (bottom).

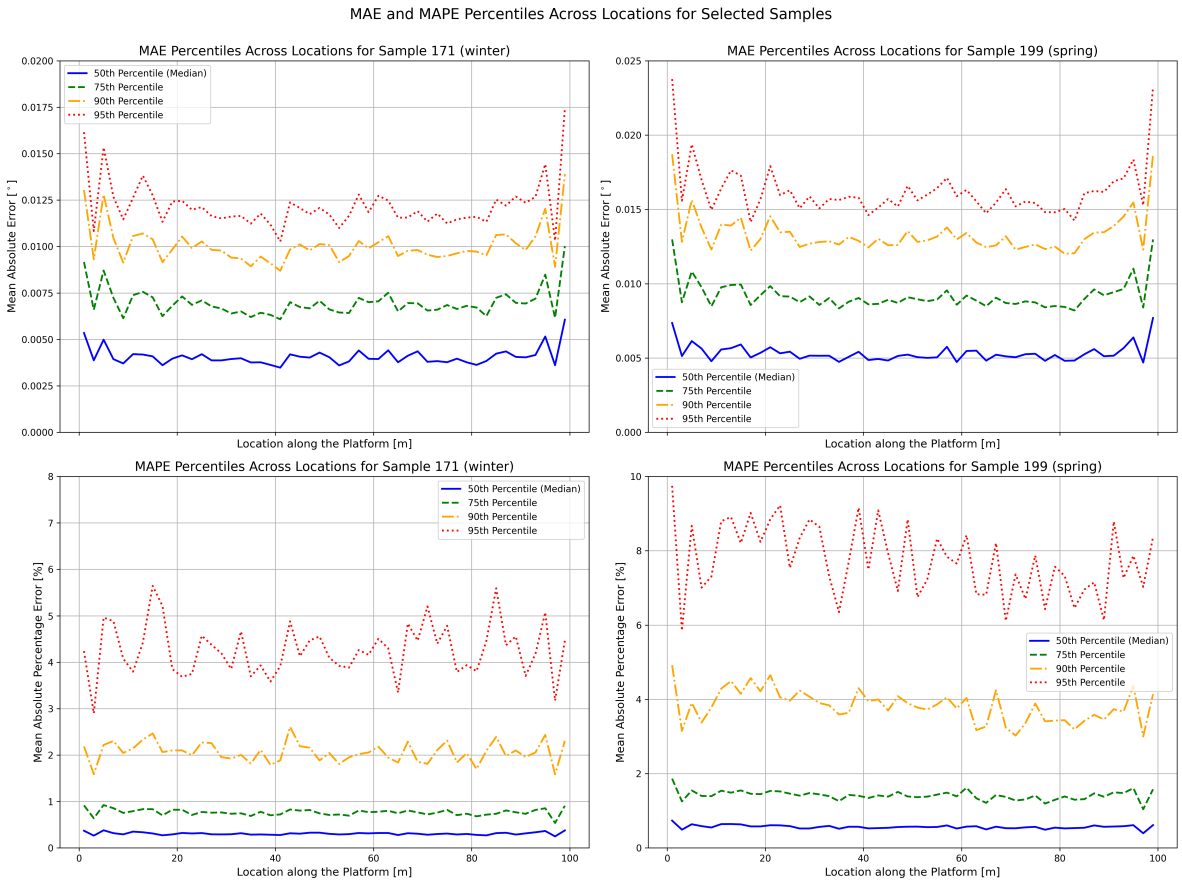


Figure D.11: The mean absolute error, and mean absolute percentage error 50th, 75th, 90th, and 95th percentile (CNN-LSTM hybrid) in time for every location along the floating structure for a sample drawn from spring (left) and autumn (right).