DTSA 5511 Introduction to Machine Learning: Deep Learning

Final Report: Using Deep Learning to Model Ocean Surface Elevation using Real World Data

Andrew Simms
University of Colorado Boulder

2024-12-09

Table of contents

1 Problem Description

Ocean wave prediction is used for maritime safety, wave energy conversion, and coastal engineering applications. This research explores deep learning approaches for predicting ocean surface elevation time-series using historical buoy measurements. While traditional wave analysis relies on statistical parameters including significant wave height (H_{m_0}) and energy period (T_e), many applications could benefit from more accurate wave-by-wave predictions of surface elevation.

The need for accurate wave prediction is particularly evident in wave energy applications, where J. V. Ringwood [1] highlights challenges in control system optimization that depend on reliable wave forecasts. O. Abdelkhalik *et al.* [2] demonstrates how wave predictions enable real-time optimization of energy extraction, showing that accurate forecasting directly impacts system performance and efficiency.

This project addresses the fundamental need for accurate near real-time wave prediction by developing deep learning models to forecast three-dimensional surface elevation time-series, focusing on maintaining both prediction accuracy and computational efficiency through models trained on previously collected measurements.

1.1 Data Sources

The study utilizes surface elevation wave measurements from the Coastal Data Information Program (CDIP) focusing on two strategic United States locations, Kaneohe Bay, Hawaii and Nags Head North Carolina. These locations were chosen because they have many years of realtime measurement and the sites have significant seasonal variations in wave conditions.

Figure 1 shows the Kaneohe Bay, Hawaii buoy location (CDIP 225) This buoy is located within the U.S. Navy's Wave Energy Test Site (WETS) [3]. This deep water deployment at 84m depth experiences a mixed wave climate with trade wind waves, North Pacific swell, and South Pacific swell. The site generally maintains consistent wave conditions due to trade wind dominance.

```
<folium.folium.Map at 0x147699e50>
```

Figure 1: Map of CDIP Buoy 225 Location at the Wave Energy Test Site, Kaneohe Bay, Oahu, HI

Figure 2 shows the Nags Head, North Carolina buoy location (CDIP 243) [4]. This buoy is located near the Jennettes's Pier Wave Energy Test Center. This site, at an intermediate water depth of 21m, experiences primarily wind-driven wave conditions. The wave climate is highly variable due to the influence of Cape Hatteras weather systems, with conditions ranging from calm seas to severe storm events including tropical cyclones.

```
<folium.folium.Map at 0x31ea199d0>
```

Figure 2: Map of CDIP Buoy 243 Location in Nags Head, NC

Table 1 compares the characteristics of the chosen measurement locations.

Table 1: CDIP 225 vs CDIP 243 site comparisons

Characteristic	Kaneohe Bay, Hawaii (CDIP 225)	Nags Head, NC (CDIP 243)
Water Depth	Deep water (84m)	Intermediate water (21m)
Wave Climate	Mixed: trade wind waves, North & South Pacific swell	Primarily wind-driven
Wave Conditions	Generally consistent due to trade winds	Highly variable due to Cape Hatteras weather
Weather Events	Seasonal variations from Pacific storms	Tropical cyclones, severe storms
Research Site	Wave Energy Test Site (WETS)	Jennette's Pier Wave Energy Test Center
CDIP Link	CDIP 225	CDIP 243
Data Record	Available since 2012	Available since 2017

1.2 Data Acquisition Hardware

Surface elevation measurements are collected using Datawell Waverider DWR-MkIII buoys, detailed in Datawell BV [5]. These specialized oceanographic instruments capture three-dimensional displacement measurements at a sampling frequency of 1.28 Hz, providing high-resolution data of vertical, northward, and eastward wave motion. All measurements undergo CDIP's standardized quality control and processing pipeline before being archived and made available for analysis.

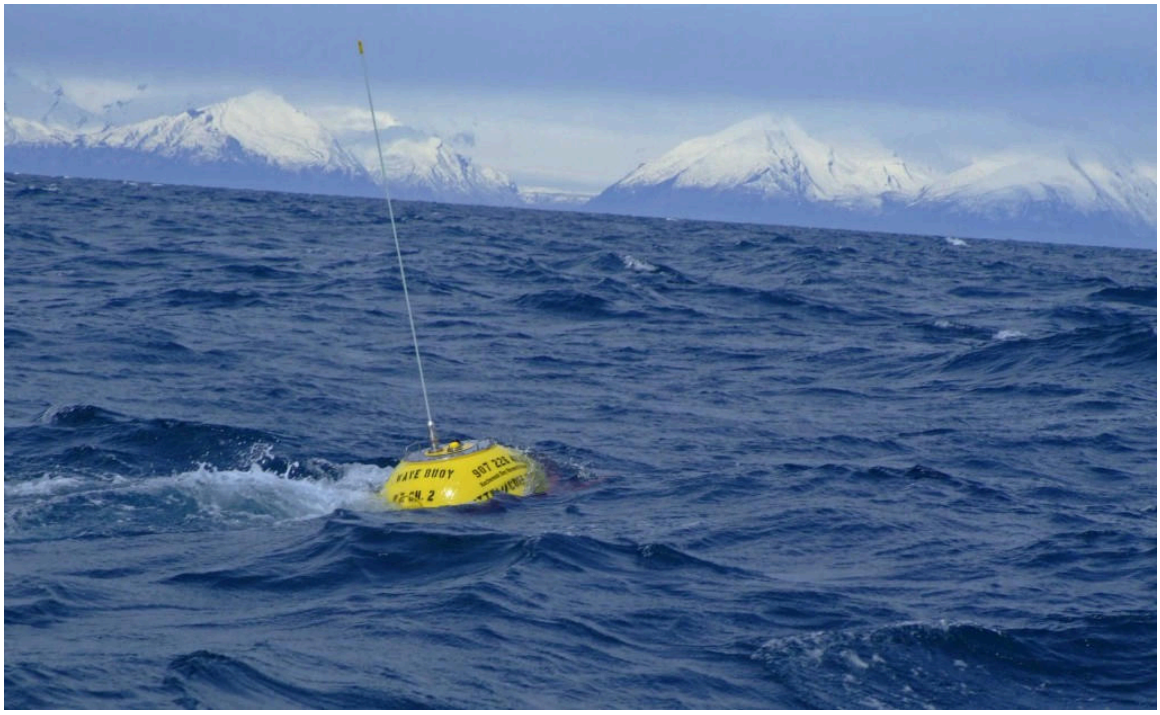


Figure 3: CDIP Buoy 204 Deployed in Lower Cook Inlet Alaska. Photo from AOOS

1.2.1 Directional Reference Frame

As illustrated in Figure 4, the buoy's movement is tracked in a three-dimensional reference frame, measuring displacements in vertical (Z), east-west (X), and north-south (Y) directions. Our prediction task focuses on forecasting these three displacement components in real-time as measurement data streams from the buoy. This multivariate time series prediction approach uses historical measurements of all three displacement components to forecast their future values over specified time horizons, providing a comprehensive representation of wave motion at each location.

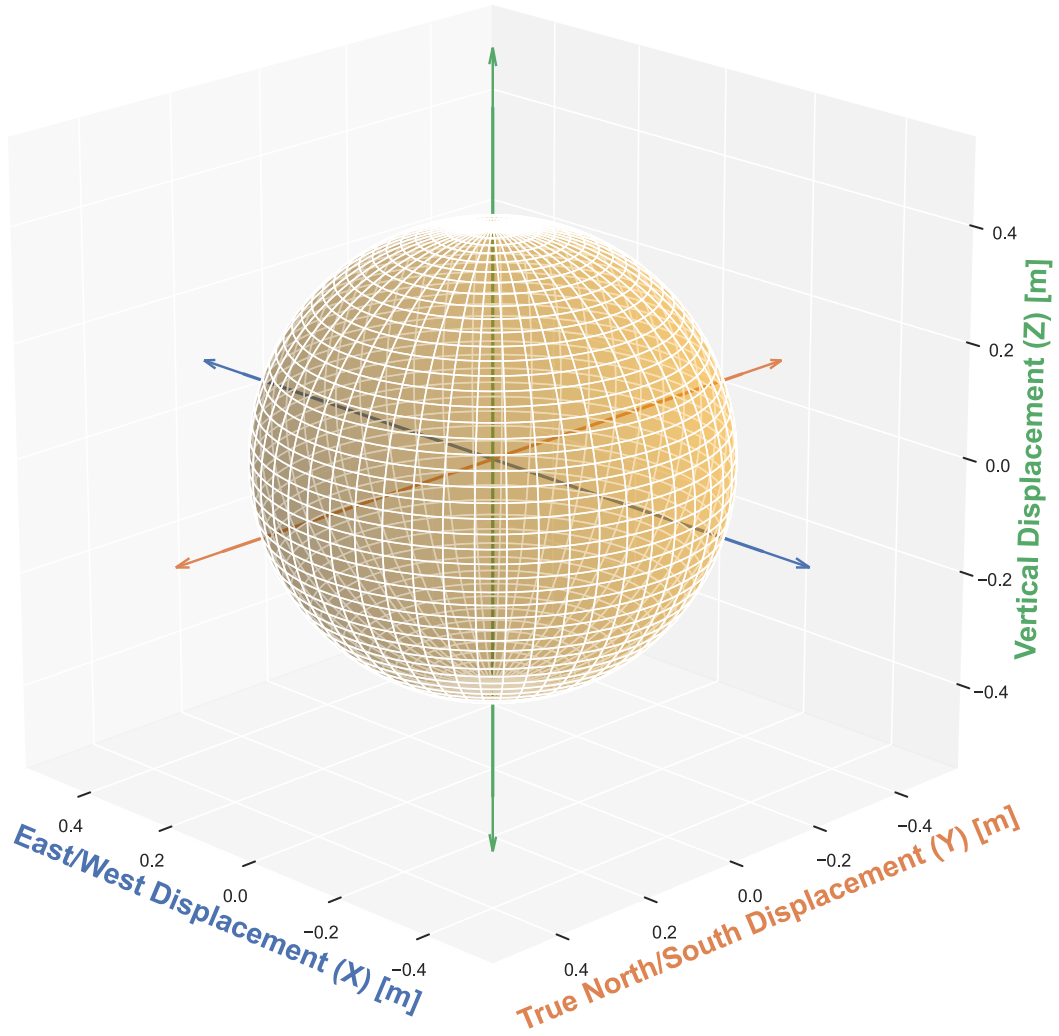


Figure 4: Directional Reference Frame of Datawell Waverider DWR-MkIII Buoy

1.3 Deep Learning Architecture Overview

This project leverages both Long Short-Term Memory (LSTM) networks and Transformer architectures to predict ocean surface elevation measurements. LSTMs, first introduced by S. Hochreiter and J. Schmidhuber [6], have demonstrated success in temporal sequence learning through their ability to capture long-term dependencies. The Transformer architecture, developed by A. Vaswani *et al.* [7], offers an alternative approach using self-attention mechanisms to process sequential data without recurrence.

Previous work in ocean wave forecasting has shown promise using neural network approaches. S. Mandal and N. Prabaharan [8] demonstrated effective wave height prediction using recurrent neural networks, while N. K. Kumar, R. Savitha, and A. A. Mamun [9] explored sequential learn-

ing algorithms for regional wave height forecasting. Building on these foundations, our approach implements both LSTM and Transformer models using the PyTorch framework by J. Ansel *et al.* [10], and the LSTM Module described by H. Sak, A. Senior, and F. Beaufays [11], allowing for direct comparison of their performance in predicting three-dimensional surface elevation time series.

1.4 Project Workflow

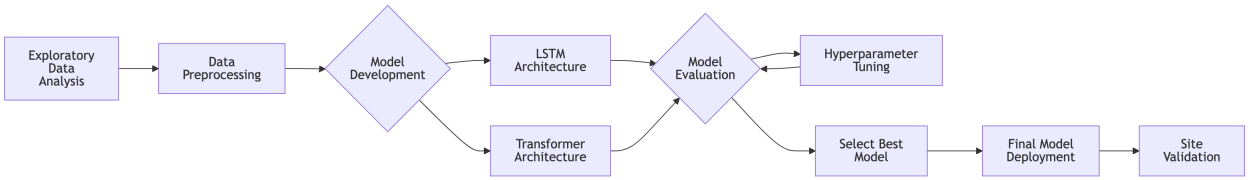


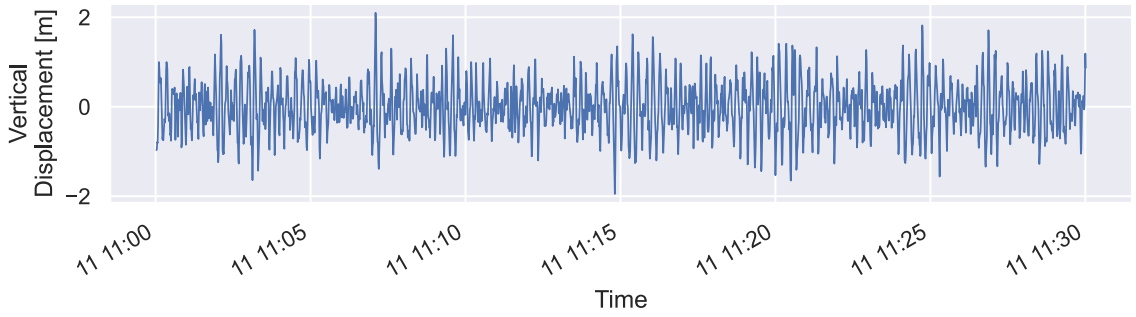
Figure 5: Wave Prediction Modeling Workflow

The workflow in Figure 5 is a systematic deep learning workflow optimized for wave prediction modeling. Beginning with exploratory data analysis of CDIP buoy measurements, the data undergoes preprocessing including normalization and temporal windowing. The model development phase explores multiple neural network architectures in parallel, followed by rigorous evaluation and hyperparameter tuning. The final model undergoes cross-site validation to assess its generalization capabilities across different ocean environments.

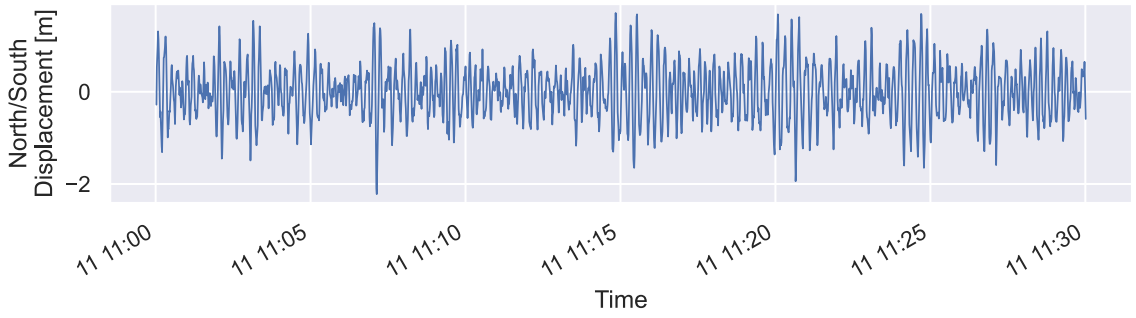
2 Exploratory Data Analysis

2.1 Displacement Measurement

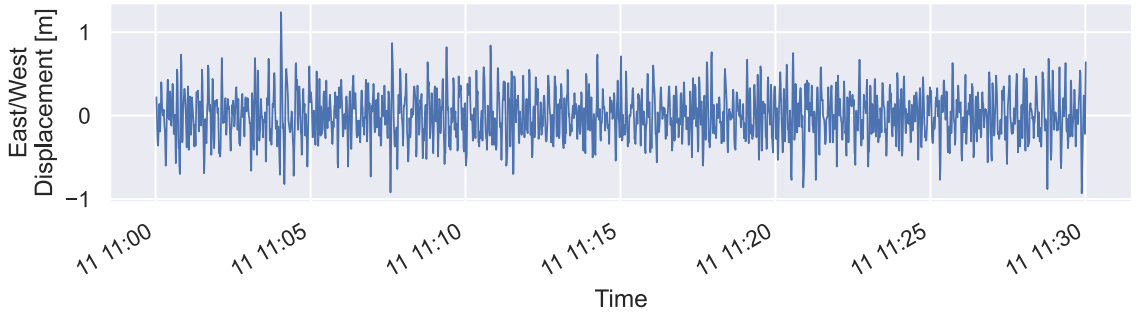
Ocean wave measurements from CDIP buoys utilize sophisticated sensor arrays including accelerometers, magnetometers, and GPS sensors to track three-dimensional wave motion Datawell BV [5]. These instruments output vertical displacement (commonly referred to as surface elevation), along with northward and eastward displacements. Figure Figure 6 illustrates these three displacement components using a 30-minute sample from CDIP 225, demonstrating the typical wave motion captured by a Datawell Waverider buoy.



(a) Vertical (Z) Displacement



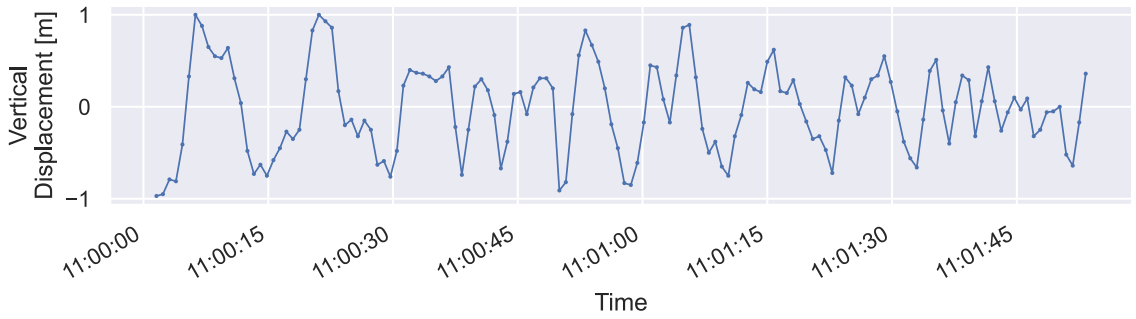
(b) North/South (Y) Displacement



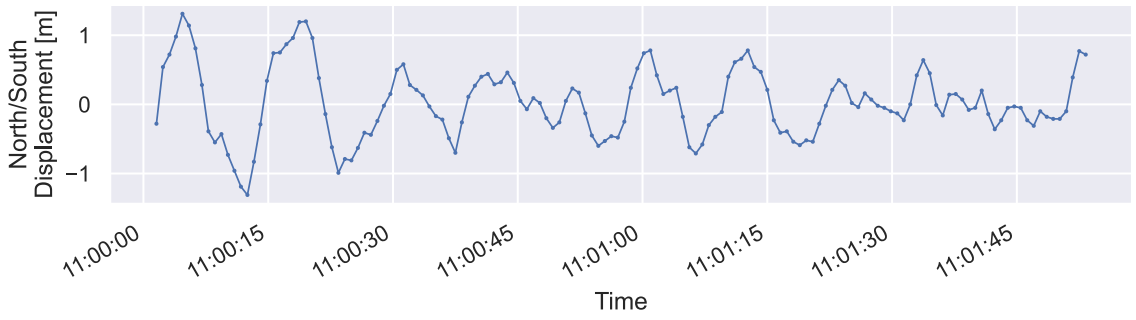
(c) East/West (X) Displacement

Figure 6: CDIP 225 30 minutes of Displacement - November 11, 2017 @ 11 am. Data from Coastal Data Information Program [3]

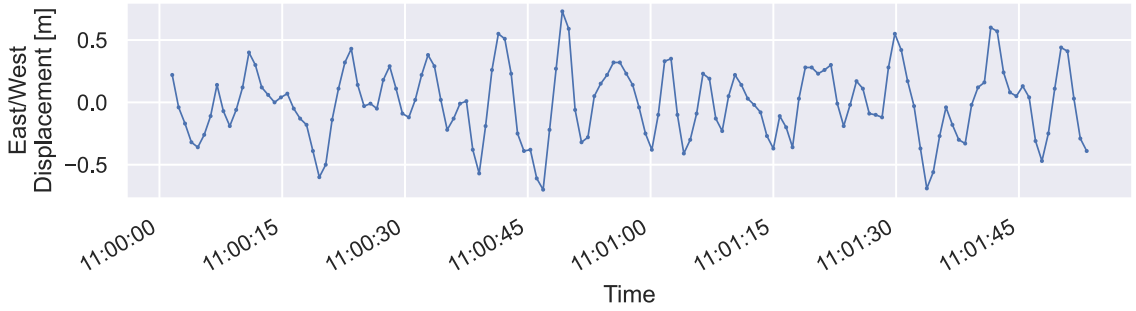
In Figure 7, a detailed view of approximately 90 seconds of the same time series reveals the fine-scale structure of wave motion. While the overall pattern exhibits typical wave behavior, the vertical displacement shows distinct non-sinusoidal characteristics, particularly during directional transitions. These abrupt changes in vertical motion highlight the complex, non-linear nature of real ocean waves compared to idealized wave forms.



(a) Vertical (Z) Displacement



(b) North/South (Y) Displacement



(c) East/West (X) Displacement

Figure 7: CDIP 225 1.5 minutes of Displacement - November 11, 2017 @ 11 am. Data from Coastal Data Information Program [3]

2.2 Data Source and Download

The displacement data used in this study was obtained from CDIP's THREDDS data server. Data for both locations - the Wave Energy Test Site (CDIP 225) and Nags Head (CDIP 243) - is hosted on CDIP's archive server with standardized displacement time series documentation. The data is provided in NetCDF format.

Each data file contains three-dimensional displacement measurements sampled at 1.28 Hz, along with corresponding quality control flags. The raw NetCDF files were processed using xarray by S. Hoyer and H. Joseph [12] for efficient handling of the multidimensional data. Timestamps were generated according to CDIP specifications, accounting for sampling rates and filter delays. The processed measurements were then consolidated into pandas DataFrames, developed by The pandas development team [13], and stored in parquet format for efficient access during model development.

Listing 1: CDIP Download Implementation

```
from pathlib import Path

import requests
import xarray as xr
import numpy as np
import pandas as pd
import seaborn as sns

sns.set_theme()

# NAGS HEAD, NC - 243
# station_number = "243"
# KANEOHE BAY, WETS, HI - 225
station_number = "225"
# 1.28 hz * 30
SAMPLES_PER_HALF_HOUR = 2304


def get_cdip_displacement_df(station_number, dataset_number):
    fname = f"{station_number}p1_d{dataset_number}.nc"

    nc_path = Path(f"./data/00_raw/{fname}").resolve()
    print(f"Opening {nc_path} if it exists...")

    if nc_path.exists() is False:
      nc_url = f"https://thredds.cdip.ucsd.edu/thredds/fileServer/cdip/archive/
{station_number}p1/{fname}"
        print("Downloading", nc_url)
        # Download the NetCDF file using requests
        response = requests.get(nc_url)
        with open(nc_path, "wb") as f:
            f.write(response.content)

    # Open the downloaded NetCDF file with xarray
    ds = xr.open_dataset(nc_path)

    # Extract the relevant variables from the dataset
    xdisp = ds["xyzXDisplacement"]  # North/South Displacement (X)
    ydisp = ds["xyzYDisplacement"]  # East/West Displacement (Y)
    zdisp = ds["xyzZDisplacement"]  # Vertical Displacement (Z)
    qc_flag = ds["xyzFlagPrimary"]  # Quality control flag

   # For some reason all of these are missing one sample. So we remove the last
section

    xdisp = xdisp[:-(SAMPLES_PER_HALF_HOUR)]
    ydisp = ydisp[:-(SAMPLES_PER_HALF_HOUR)]
    zdisp = zdisp[:-(SAMPLES_PER_HALF_HOUR)]
    qc_flag = qc_flag[:-(SAMPLES_PER_HALF_HOUR)]

    filter_delay = ds["xyzFilterDelay"].values
    start_time = ds["xyzStartTime"].values  # Start time of buoy data collection
    sample_rate = float(
        ds["xyzSampleRate"].values
    )  # Sample rate of buoy data collection
```

The CDIP download, shown in Listing 1, demonstrates the automated download and processing pipeline. This code handles the retrieval of NetCDF files, extraction of displacement measurements, timestamp generation, and data organization into a structured format suitable for analysis and model training. Quality control flags are preserved throughout the processing pipeline to ensure data integrity.

2.3 Available Data

This section examines the temporal extent and characteristics of the available wave measurements.

2.3.1 Duration

Table 2: Temporal Details of Downloaded CDIP Data

	Start Date [UTC]	End Date [UTC]	Duration
0	2016-08-26 22:00	2024-09-11 19:00	8 years, 17 days
1	2018-08-26 15:00	2023-07-13 00:00	4 years, 10 months, 21 days

Based the information in Table 2, the CDIP buoy datasets provide extensive coverage for both locations: WETS (CDIP 225) spans approximately 8 years (2016-2024), while Nags Head (CDIP 243) covers nearly 5 years (2018-2023). Both datasets contain three-dimensional displacement measurements at 1.28 Hz sampling frequency.

2.4 Displacement Statistics

For each location, we analyzed the statistical characteristics of the three-dimensional displacement measurements.

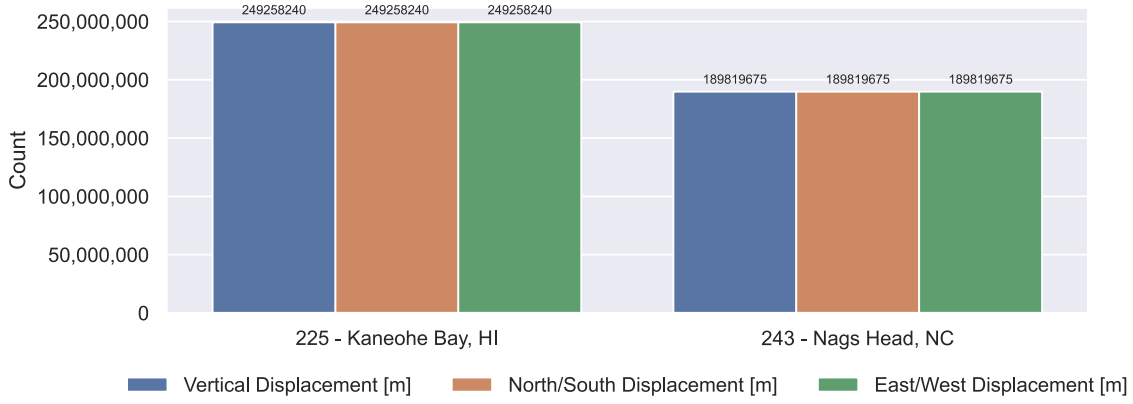


Figure 8: Available Data: Sample Count

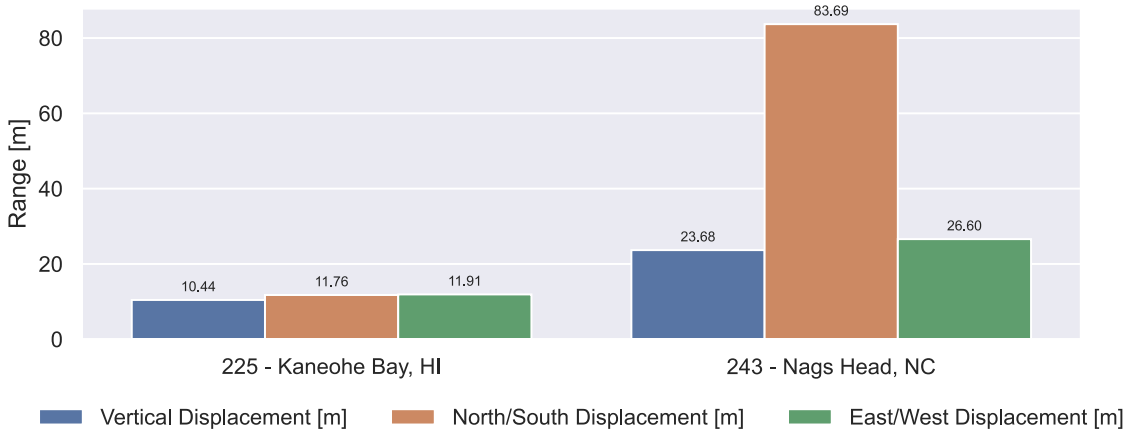


Figure 9: Available Data: Range [m]

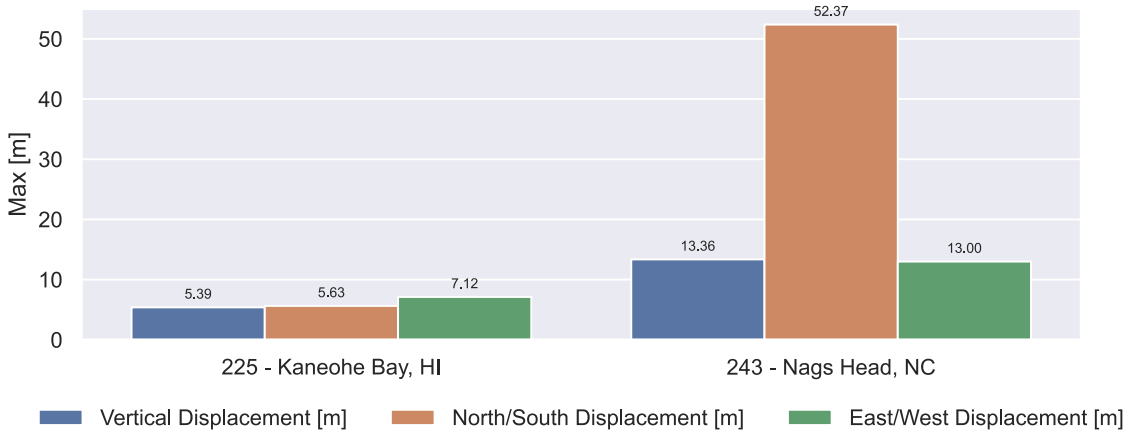


Figure 10: Available Data: Maximum [m]

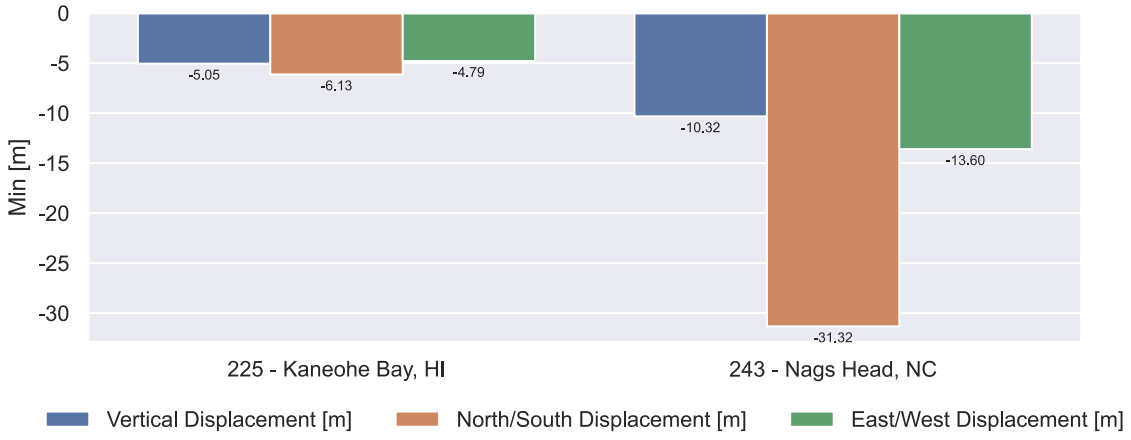


Figure 11: Available Data: Minimum [m]

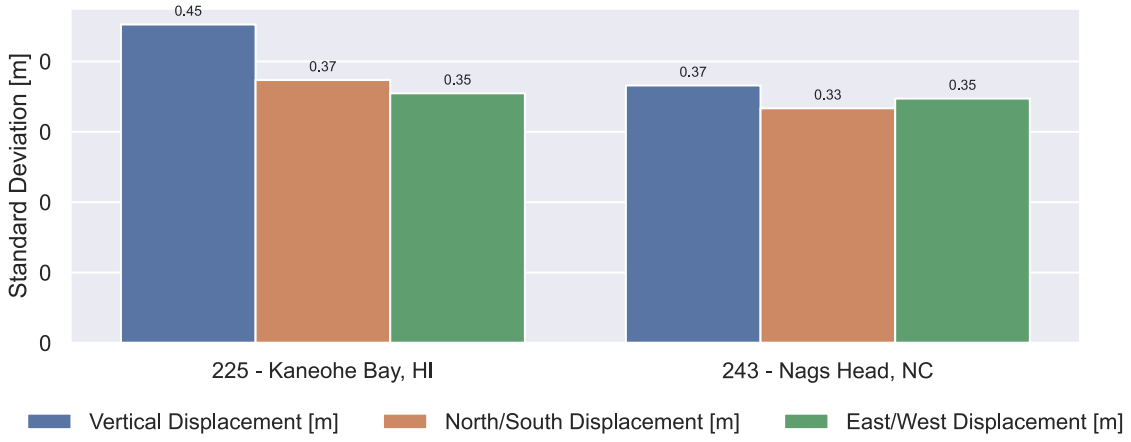


Figure 12: Available Data: Standard Deviation [m]

Figure 9 shows that at WETS (CDIP 225), vertical displacements range approximately ± 5 m, while north-south and east-west displacements show similar ranges of about ± 6 m. The Nags Head site (CDIP 243) experiences larger displacement ranges, with vertical motion reaching ± 13 m and horizontal displacements extending up to ± 52 m, reflecting its more dynamic wave climate. All displacement components at both locations show near-zero means with standard deviations between 0.33 and 0.45 m, indicating symmetric wave motion about the mean position. The anomalous 52 m range in horizontal displacement at Nags Head suggests potential outliers or measurement artifacts that should be filtered prior to model training to ensure data quality.

2.5 Partitioning Data

2.6 Data Partitioning

To facilitate efficient data handling and model development, we partitioned the continuous time series into 30-minute segments using a hierarchical storage structure. Each segment contains 2,304 samples (corresponding to the 1.28 Hz sampling rate) and is organized using a hive partitioning strategy based on temporal metadata (year, month, day, hour, minute) and station number.

The implementation, shown in **?@lst-partition**, creates a systematic file structure where each 30-minute measurement period is stored as an individual parquet file. This organization enables efficient data loading during model training and validation, while maintaining the temporal relationship between segments. The hierarchical structure also facilitates parallel processing and selective data loading based on specific time periods or stations.

2.7 Calculating Statistical Wave Parameters

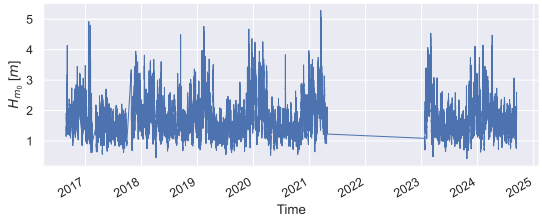
The original multi year displacement data volume is too large to train a model for this project. As such we need to subset the problem. To do this we will use MHKiT-Python by R. Fao *et al.* [14] to transform the surface elevations into 30 minute statistics of wave measurements. These are easier to visualize and understand the wave characteristics.

Transforming raw displacement measurements into 30-minute statistical wave parameters using MHKiT-Python R. Fao, S. Olson, K. Klise, K. Ruehl, Chris-Ivanov, R. Coe, C. A. M. Ströfer, J. McVey, A. Keester, E. Browning, M. Boyd, Bax, M. Bruggemann, C. Kenny, aidanbharath, A. Simms, Cesar, D. Mayes, M. Levin, and S. Pluemer [14] provides a clearer view of the wave conditions at each site. These statistical metrics - significant wave height (H_{m_0}), energy period (T_e), and omnidirectional wave energy flux (J) - help identify unique wave conditions and temporal patterns within the large dataset.

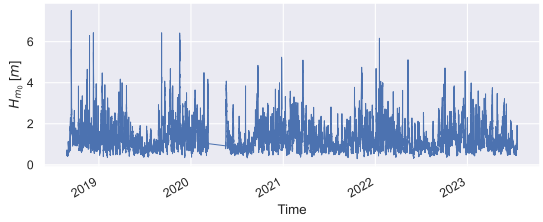
The implementation shown in Listing 2 computes these wave parameters.

Listing 2: Calculation of Wave Quantities of Interest from Displacement

```
import mhkit.wave as wave

def calculate_wave_qoi(input_df, station_number, path):
    input_df = input_df.dropna(axis="index")
    column = "vert_displacement_meters"

    if len(input_df) != 2304:
        return None

    sample_rate_hz = 1.28
    n_fft = 256
    window = "hann"
    detrend = True
    surface_elevation = pd.DataFrame(input_df[column].iloc[:2048])
    spectra = wave.resource.elevation_spectrum(
       surface_elevation, sample_rate_hz, n_fft, window=window, detrend=detrend
    )

    return {
        "time": input_df.index[0],
                                              "significant_wave_height_meters":
wave.resource.significant_wave_height(spectra)["Hm0"].to_list()[0],    #  type:
ignore
                 "energy_period_seconds":  wave.resource.energy_period(spectra)
["Te"].to_list()[0],  # type: ignore
         "omnidirectional_wave_energy_flux": wave.resource.energy_flux(spectra,
np.nan, deep=True)["J"].to_list()[0],  # type: ignore
        "station_number": station_number,
        "path": str(path),
    }
```

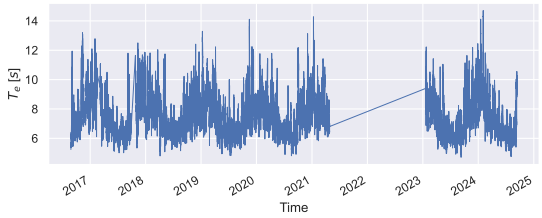


(a) Wave Energy Test Site - Hawaii

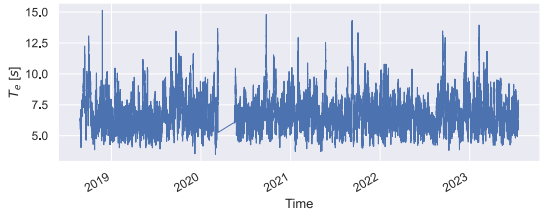


(b) Nags Head - North Carolina

Figure 13: Significant Wave Height, H_{m_0} $[m]$

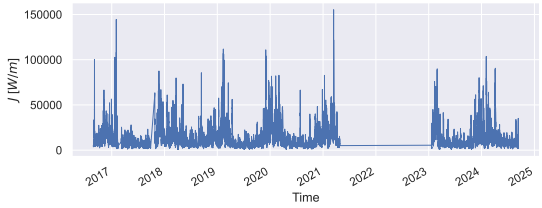


(a) Wave Energy Test Site - Hawaii

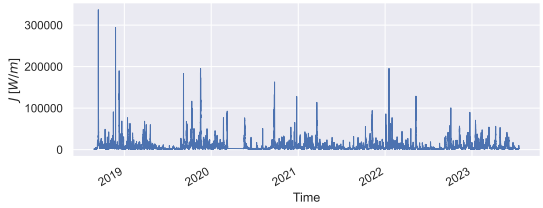


(b) Nags Head - North Carolina

Figure 14: Energy Period, T_e $[s]$

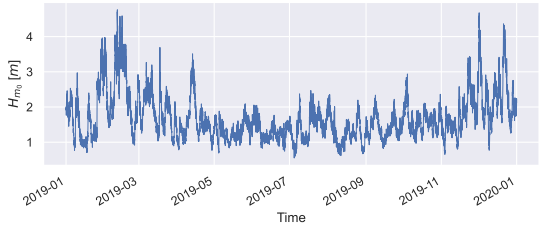


(a) Wave Energy Test Site - Hawaii

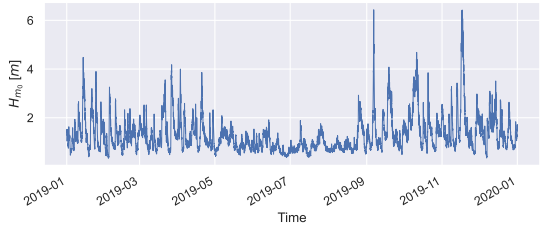


(b) Nags Head - North Carolina

Figure 15: Omnidirectional Wave Energy Flux (Wave Power), J $[W/m]$

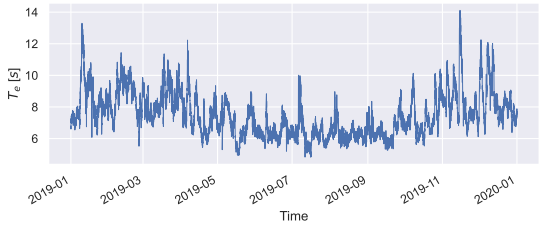


(a) Wave Energy Test Site - Hawaii

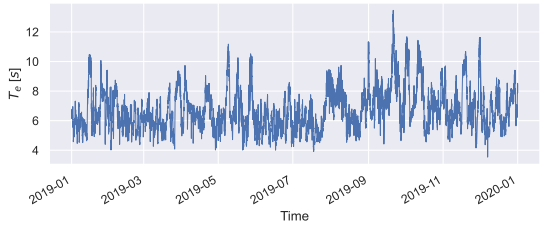


(b) Nags Head - North Carolina

Figure 16: Significant Wave Height, H_{m_0} $[m]$

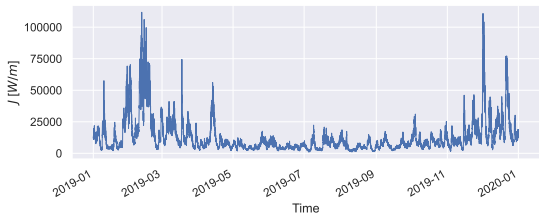


(a) Wave Energy Test Site - Hawaii

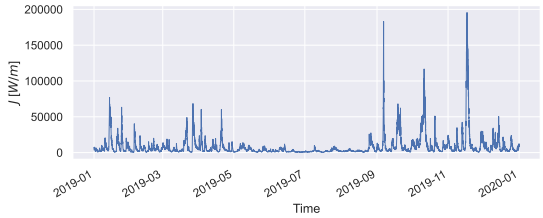


(b) Nags Head - North Carolina

Figure 17: Energy Period, T_e $[s]$



(a) Wave Energy Test Site - Hawaii



(b) Nags Head - North Carolina

Figure 18: Omnidirectional Wave Energy Flux (Wave Power), J $[W/m]$

As shown in Figure 13 through Figure 15, WETS (CDIP 225) exhibits more periodic behavior while Nags Head (CDIP 243) shows greater variability. A notable data gap exists in the WETS measurements from 2021 to 2023. Focusing on 2019 (Figure 16 through Figure 18) highlights that Nags Head experiences higher peak wave power and wave heights but a narrower range of wave periods compared to WETS. These insights will guide our selection of representative data segments for model development.

2.8 Sea State Analysis

To understand the distribution of wave conditions at each site, we developed sea state matrices that bin the data by significant wave height (H_{m_0}) and energy period (T_e). This categorization quantifies the frequency of different wave conditions and helps ensure our training dataset encompasses a representative range of height and period states.

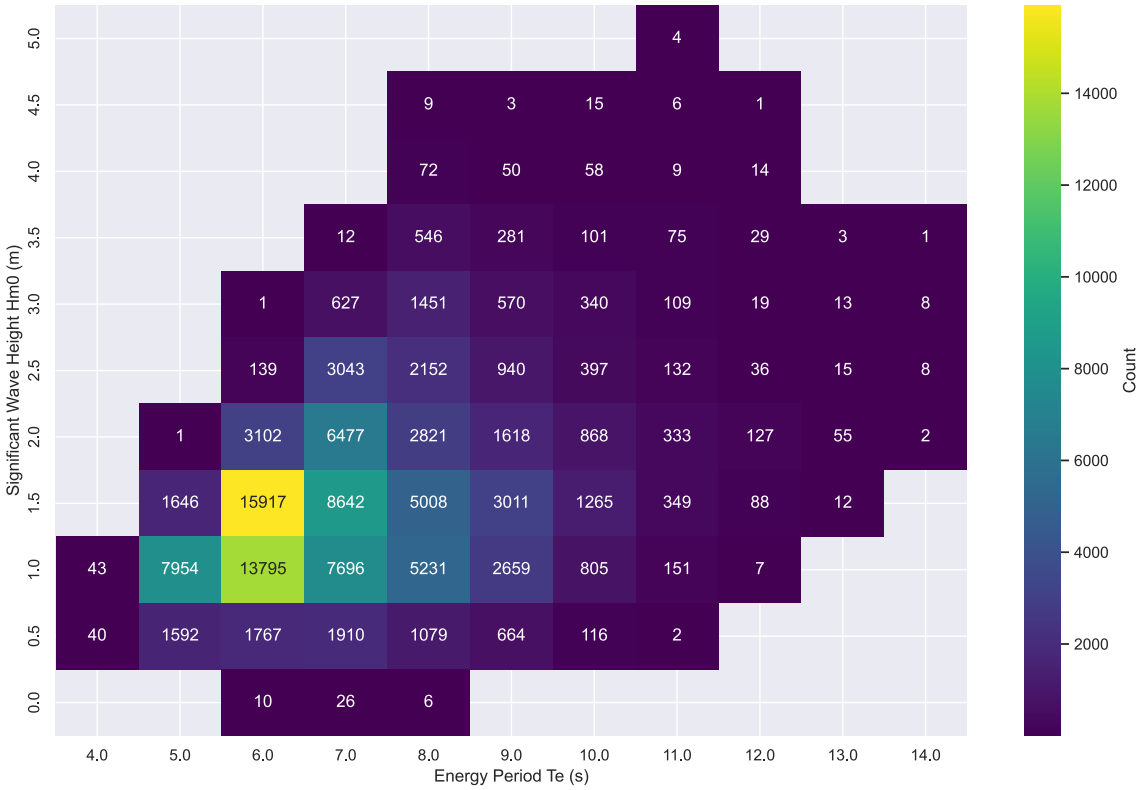


Figure 19: Distribution of Sea States at WETS (CDIP 225), Showing Occurrence Count of Combined Significant Wave Height (H_{m_0}) and Energy Period (T_e) Conditions

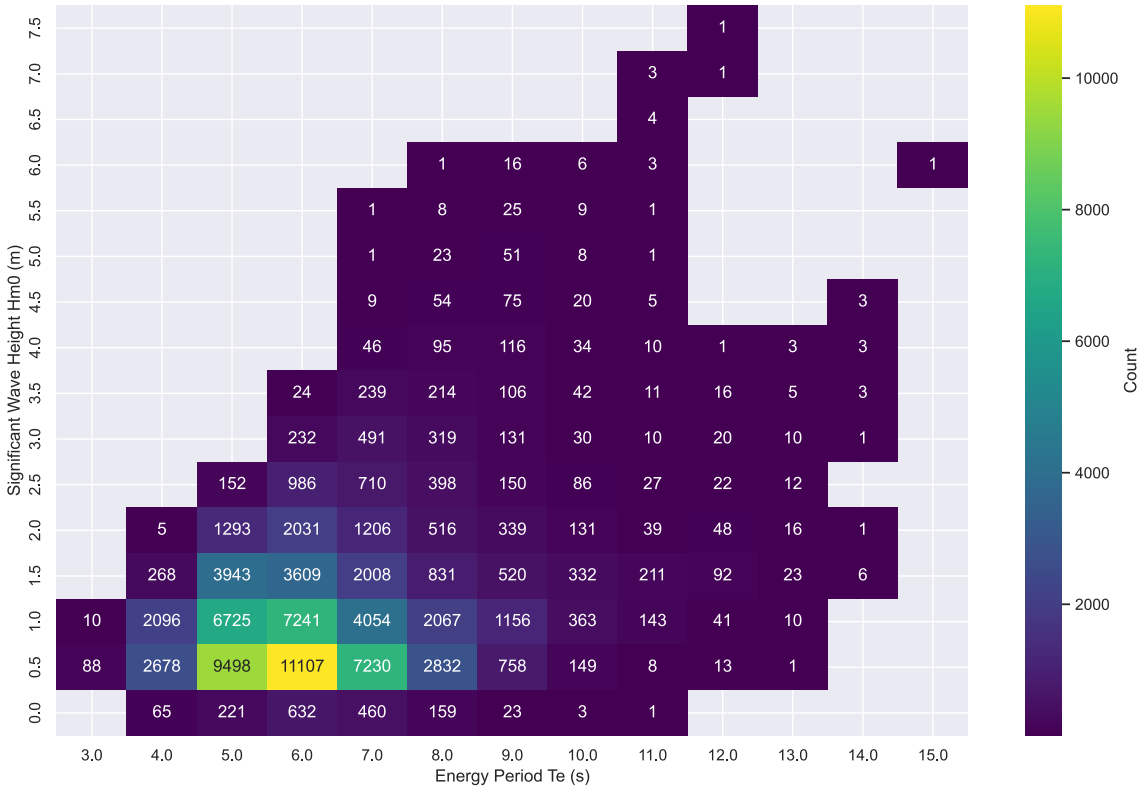


Figure 20: Distribution of Sea States at Nags Head (CDIP 243), Showing Occurrence Count of Combined Significant Wave Height (H_{m_0}) and Energy Period (T_e) Conditions

The sea state matrices reveal distinct wave climates at each location. As shown in Figure 19, the WETS site exhibits wave heights predominantly below 5 m, with energy periods clustered between 6-12 seconds. In contrast, Figure 20 shows that Nags Head experiences a broader range of conditions, with wave heights reaching 7.5 m and energy periods spanning 3-15 seconds. The presence of 3-second periods at Nags Head likely indicates local wind-generated waves, a characteristic absent at WETS.

These differing wave climate characteristics confirm that our dataset captures a diverse range of sea states across both locations, providing a robust foundation for model development. The comprehensive coverage of wave conditions suggests we can proceed with creating training datasets that will expose our models to the full spectrum of expected wave behaviors.

3 Data Cleaning and Dataset Generation

3.1 Structured Sampling Method

Based on the sea state distributions shown in Figure 19 and Figure 20, we developed a systematic sampling method to create balanced training datasets. The approach, implemented in **?@lst-bin-sampling**, samples time series segments from each combination of significant wave height and energy period bins, ensuring representation across the full range of observed wave conditions.

3.2 Training Dataset Creation

station_number	significant_wave_height_meters	energy_period_seconds	omnidirectional_wave_energy_flux	path	hm0_bin	te_bin	hm0_bin_center	te_bin_center
0	225	1.116275	5.640104	3445.602238	/Users/asmacbook/Desktop/Programming/msds/dtsa...	1.0	5.0	1.25	5.5
1	225	1.352471	6.033005	5410.344970	/Users/asmacbook/Desktop/Programming/msds/dtsa...	1.0	6.0	1.25	6.5
2	225	1.087143	7.104669	4116.732805	/Users/asmacbook/Desktop/Programming/msds/dtsa...	1.0	7.0	1.25	7.5
3	225	1.068228	8.150482	4559.810465	/Users/asmacbook/Desk-	1.0	8.0	1.25	8.5

station_number	significant_wave_height_meters	energy_period_seconds	omnidirectional_wave_energy_flux	path	hm0_bin	te_bin	hm0_bin_center	te_bin_center
				top/Programming/msds/dtsa...				
225	1.070689	9.320074	5238.196702	/Users/asmacbook/Desktop/Programming/msds/dtsa...	1.0	9.0	1.25	9.5
4								

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76 entries, 0 to 75
Data columns (total 9 columns):
 #   Column                           Non-Null Count  Dtype  
---  ------                           --------------  -----  
 0   station_number                   76 non-null     object 
 1   significant_wave_height_meters   76 non-null     float64
 2   energy_period_seconds            76 non-null     float64
 3   omnidirectional_wave_energy_flux 76 non-null     float64
 4   path                             76 non-null     object 
 5   hm0_bin                          76 non-null     float64
 6   te_bin                           76 non-null     float64
 7   hm0_bin_center                   76 non-null     float64
 8   te_bin_center                    76 non-null     float64
dtypes: float64(7), object(2)
memory usage: 5.5+ KB
```

	station_number	significant_wave_height_meters	energy_period_seconds	omnidirectional_wave_energy_flux	path	hm0_bin	te_bin	hm0_bin_center	te_bin_center
0	243	0.969098	6.869854	3163.139256	/Users/asmacbook/Desktop/Programming/msds/dtsa...	0.5	6.0	0.75	6.5
1	243	0.751133	5.046030	1395.787017	/Users/asmacbook/Desktop/Programming/msds/dtsa...	0.5	5.0	0.75	5.5
2	243	0.736608	4.954708	1318.034155	/Users/asmacbook/Desktop/Programming/msds/dtsa...	0.5	4.0	0.75	4.5
3	243	0.877664	7.485963	2827.089576	/Users/asmacbook/Desktop/	0.5	7.0	0.75	7.5

station_number	significant_wave_height_meters	energy_period_seconds	omnidirectional_wave_energy_flux	path	hm0_bin	te_bin	hm0_bin_center	te_bin_center
243	0.676855	8.722353	1959.116238	/Users/asmacbook/Desktop/Programming/msds/dtsa...	0.5	8.0	0.75	8.5
4				Programming/msds/dtsa...				

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 112 entries, 0 to 111
Data columns (total 9 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   station_number                  112 non-null    object
 1   significant_wave_height_meters  112 non-null    float64
 2   energy_period_seconds           112 non-null    float64
 3   omnidirectional_wave_energy_flux  112 non-null    float64
 4   path                            112 non-null    object
 5   hm0_bin                         112 non-null    float64
 6   te_bin                          112 non-null    float64
 7   hm0_bin_center                  112 non-null    float64
 8   te_bin_center                   112 non-null    float64
dtypes: float64(7), object(2)
memory usage: 8.0+ KB
```

Initial sampling with one sample per bin yielded 76 half-hour segments for WETS (CDIP 225) and 112 segments for Nags Head (CDIP 243), providing a manageable dataset for initial model development. To explore the impact of dataset size on model performance, we also created expanded datasets with five samples per bin, resulting in 380 segments for WETS and 560 segments for Nags

Head. While these larger datasets offer more comprehensive coverage of wave conditions, they require significantly more computational resources for training.

This sampling approach ensures our training data captures the diverse wave conditions present at each site while maintaining computational feasibility. The structured nature of the sampling helps prevent bias toward more common wave conditions, potentially improving model robustness across different sea states.

```
353
```

```
493
```

Additional datasets with five samples per bin were also created (353 segments for WETS and 493 for Nags Head) and archived for future research, though this project focuses on the more computationally manageable single-sample datasets.

4 Deep Learning Models

Building on the architectural overview presented in Section 1.3, we implemented three neural network models using PyTorch J. Ansel, E. Yang, H. He, N. Gimelshein, A. Jain, M. Voznesensky, B. Bao, P. Bell, D. Berard, E. Burovski, G. Chauhan, A. Chourdia, W. Constable, A. Desmaison, Z. DeVito, E. Ellison, W. Feng, J. Gong, M. Gschwind, B. Hirsh, S. Huang, K. Kalambarkar, L. Kirsch, M. Lazos, M. Lezcano, Y. Liang, J. Liang, Y. Lu, C. Luk, B. Maher, Y. Pan, C. Puhrsch, M. Reso, M. Saroufim, M. Y. Siraichi, H. Suk, M. Suo, P. Tillet, E. Wang, X. Wang, W. Wen, S. Zhang, X. Zhao, K. Zhou, R. Zou, A. Mathews, G. Chanan, P. Wu, and S. Chintala [10] to predict ocean wave displacements. Each model architecture was chosen and designed to capture different aspects of the temporal patterns present in wave motion.

4.1 LSTM Model

Our base LSTM model, shown in Listing 3, provides a straightforward approach to sequence prediction. The model processes three-dimensional displacement inputs through stacked LSTM layers with dropout regularization. This architecture enables the model to learn wave patterns at different time scales. The final linear layer maps the LSTM outputs back to displacement predictions, creating a direct sequence-to-sequence prediction framework.

Listing 3: Long Short-Term Memory PyTorch Model

```
class LSTMModel(WavePredictionModel):
    def __init__(
        self,
        input_dim: int,
        hidden_dim: int = 128,
        num_layers: int = 2,
        dropout: float = 0.2,
        learning_rate: float = 1e-3,
    ):
        super().__init__(input_dim, learning_rate)
        self.save_hyperparameters()  # Save all init parameters

        self.hidden_dim = hidden_dim
        self.num_layers = num_layers

        self.lstm = nn.LSTM(
           input_dim, hidden_dim, num_layers, batch_first=True, dropout=dropout
        )
        self.fc = nn.Linear(hidden_dim, input_dim)

    def forward(self, x: torch.Tensor) -> torch.Tensor:
        lstm_out, _ = self.lstm(x)
        predictions = self.fc(lstm_out)
        return predictions
```

4.2 Enhanced LSTM Model

The enhanced LSTM implementation, detailed in Listing 4, extends the base model with several architectural improvements. Bidirectional processing allows the model to consider both past and future context when making predictions. The addition of skip connections helps maintain gradient flow through the deep network, while layer normalization stabilizes training.

Listing 4: Enhanced Long Short-Term Memory PyTorch Model

```
class EnhancedLSTMModel(WavePredictionModel):
    def __init__(
        self,
        input_dim: int,
        hidden_dim: int = 128,
        num_layers: int = 2,
        dropout: float = 0.2,
        learning_rate: float = 1e-3,
        bidirectional: bool = True,
    ):
        super().__init__(input_dim, learning_rate)
        self.save_hyperparameters()

        self.hidden_dim = hidden_dim
        self.num_layers = num_layers
        self.bidirectional = bidirectional

        # Input processing
        self.input_layer = nn.Sequential(
            nn.Linear(input_dim, hidden_dim),
            nn.LayerNorm(hidden_dim),
            nn.ReLU(),
            nn.Dropout(dropout / 2),
        )

        # Main LSTM layers with skip connections
        self.lstm_layers = nn.ModuleList()
        self.layer_norms = nn.ModuleList()

        lstm_input_dim = hidden_dim
        lstm_output_dim = hidden_dim * 2 if bidirectional else hidden_dim

        for _ in range(num_layers):
            self.lstm_layers.append(
                nn.LSTM(
                    lstm_input_dim,
                    hidden_dim,
                    num_layers=1,
                    batch_first=True,
                    bidirectional=bidirectional,
                    dropout=0,
                )
            )
            self.layer_norms.append(nn.LayerNorm(lstm_output_dim))
            lstm_input_dim = lstm_output_dim

        # Output processing
        self.output_layers = nn.ModuleList(
            [
                nn.Linear(lstm_output_dim, hidden_dim),
                nn.Linear(hidden_dim, hidden_dim // 2),
                nn.Linear(hidden_dim // 2, input_dim),
            ]
        )
```

4.3 Transformer Model

Our Transformer implementation, shown in Listing 5, takes a fundamentally different approach to sequence modeling. Rather than processing the wave motion sequentially, the model uses self-attention mechanisms to directly capture relationships between any two points in the input sequence. This architecture, combined with multi-head attention and position-wise feed-forward networks, should enable the model to identify both short-term wave patterns and longer-range dependencies in the displacement data.

Listing 5: Transformer PyTorch Model

```
class TransformerModel(WavePredictionModel):

    def __init__(
        self,
        input_dim: int,
        d_model: int = 128,
        nhead: int = 8,
        num_layers: int = 4,
        dropout: float = 0.2,
        learning_rate: float = 1e-3,
    ):
        super().__init__(input_dim, learning_rate)
        self.save_hyperparameters()  # Save all init parameters

        self.input_projection = nn.Linear(input_dim, d_model)

        encoder_layer = nn.TransformerEncoderLayer(
            d_model=d_model,
            nhead=nhead,
            dim_feedforward=d_model * 4,
            dropout=dropout,
            batch_first=True,
        )
        self.transformer_encoder = nn.TransformerEncoder(
            encoder_layer, num_layers=num_layers
        )

        self.output_projection = nn.Linear(d_model, input_dim)

    def forward(self, x: torch.Tensor) -> torch.Tensor:
        x = self.input_projection(x)
        x = self.transformer_encoder(x)
        return self.output_projection(x)
```

5 Training

We implemented the model training process using PyTorch Lightning W. Falcon and The PyTorch Lightning team [15] to standardize the training, validation, and testing procedures. The Mean

Absolute Error was selected as the loss function for evaluating prediction accuracy. Data was split using an 80-20 train-test split, with the training portion further divided 80-20 for validation.

All models were trained with consistent hyperparameters: learning rate of 0.001, Adam optimizer, 128-sample input and prediction windows, and 128 hidden dimensions. Training data consisted of one 30-minute dataset per sea state bin for each CDIP location. We tested several model configurations, including baseline LSTM (25 epochs), extended LSTM (100 epochs), varying LSTM layer depths (4 and 6 layers), and both basic and enhanced Transformer architectures. Model specifications are shown in Table 3.

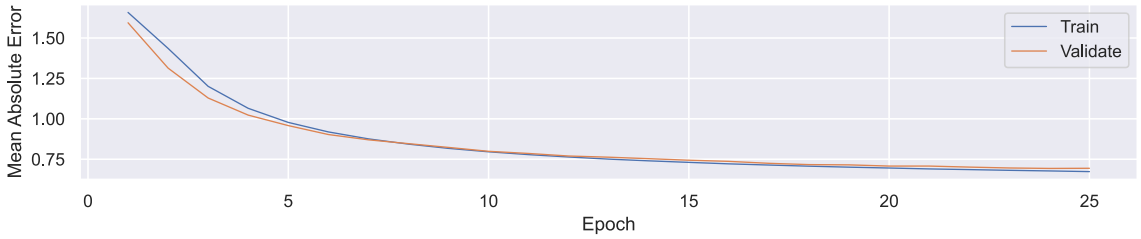
Table 3: Model Architecture Specifications

Model Type	Layers	Epochs	Special Features
Baseline LSTM	2	25	Basic implementation
Extended LSTM	2	100	Same as baseline with longer training
Deep LSTM	4	25	Additional LSTM layers
Deeper LSTM	6	25	Maximum layer depth tested
Enhanced LSTM	2	25	Bidirectional processing, skip connections, layer normalization
Basic Transformer	2	25	Multi-head attention mechanism
Enhanced Transformer	2	25	Additional positional encoding, enhanced feed-forward network

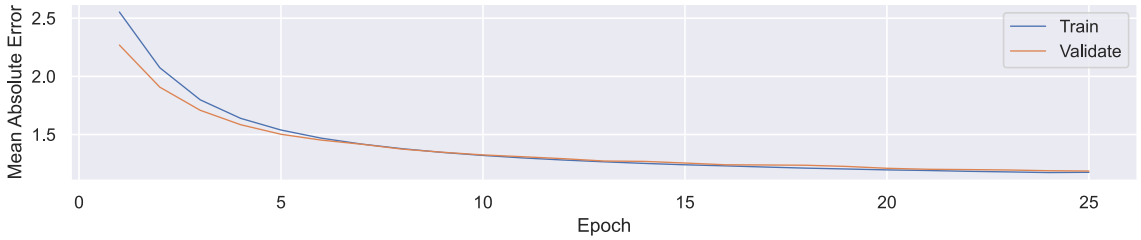
The full training code can be found in this projects GitHub Repository in the file `./train_window_from_spec.py`.

5.1 Training Metrics

5.1.1 Baseline Model



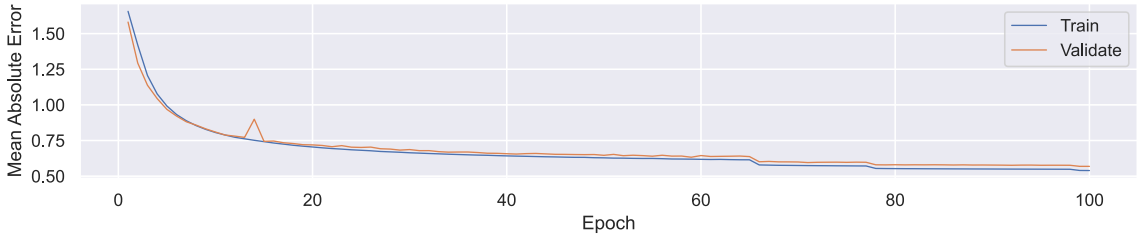
(a) CDIP 225



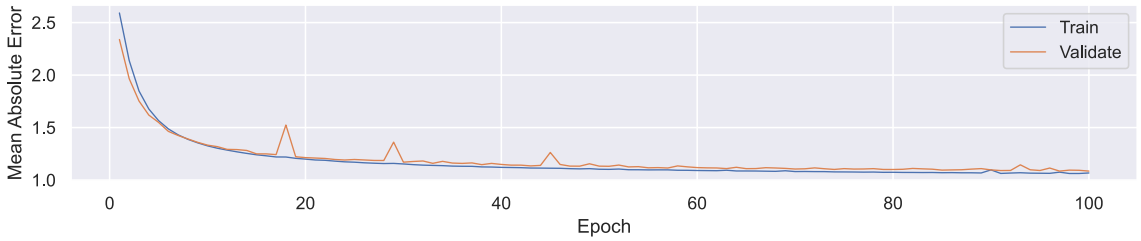
(b) CDIP 243

Figure 21: Baseline Model: Training vs. Validation Mean Absolute Error

5.1.2 100 Epoch LSTM Model



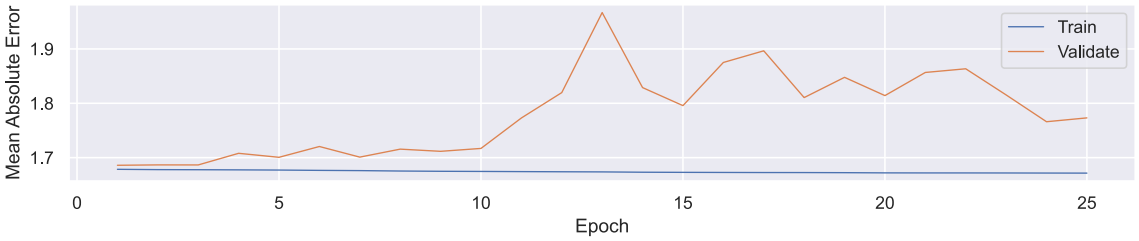
(a) CDIP 225



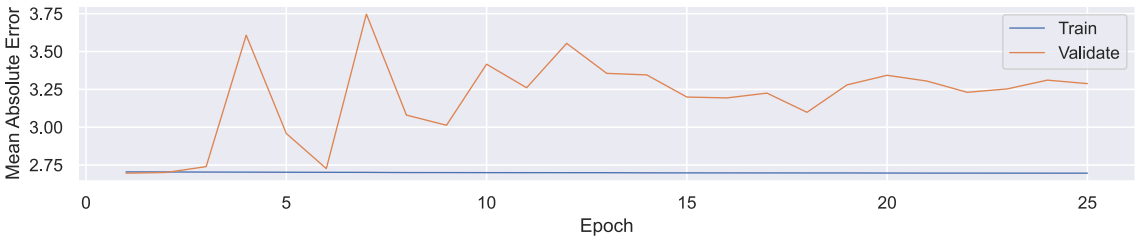
(b) CDIP 243

Figure 22: 100 Epoch LSTM Model: Training vs. Validation Mean Absolute Error

5.1.3 Transformer Model



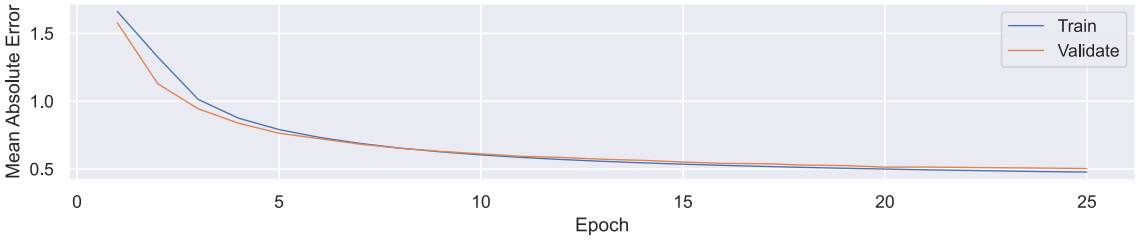
(a) CDIP 225



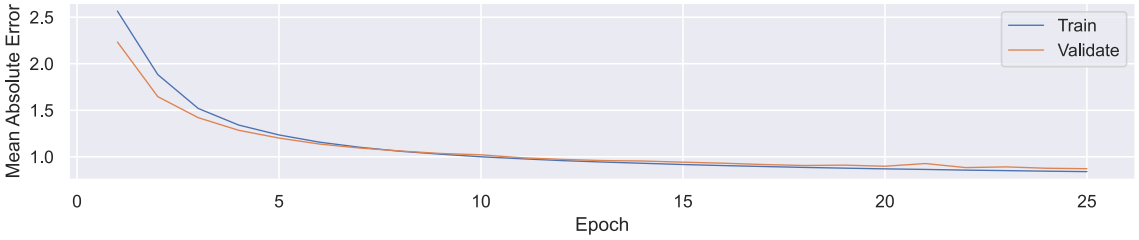
(b) CDIP 243

Figure 23: Transformer Model: Training vs. Validation Mean Absolute Error

5.1.4 4 Layer LSTM Model



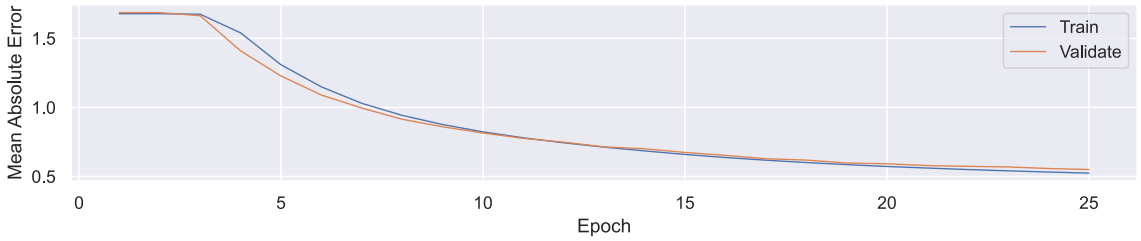
(a) CDIP 225



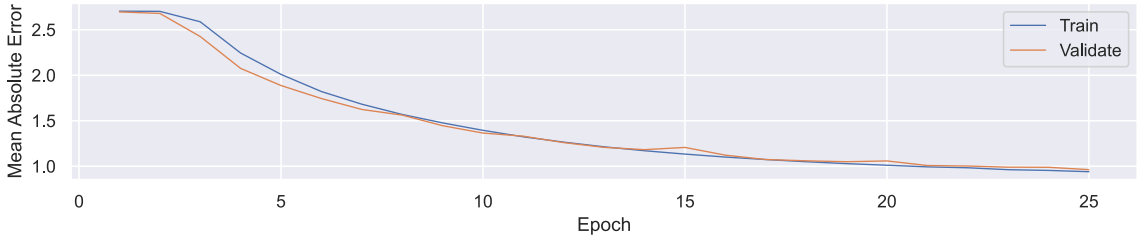
(b) CDIP 243

Figure 24: 4 Layer LSTM Model: Training vs. Validation Mean Absolute Error

5.1.5 6 Layer LSTM Model



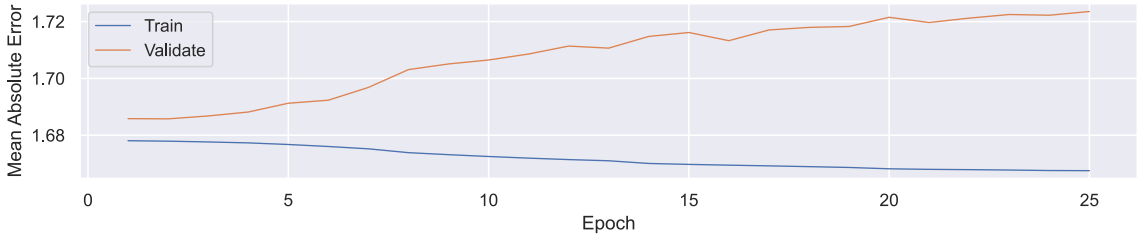
(a) CDIP 225



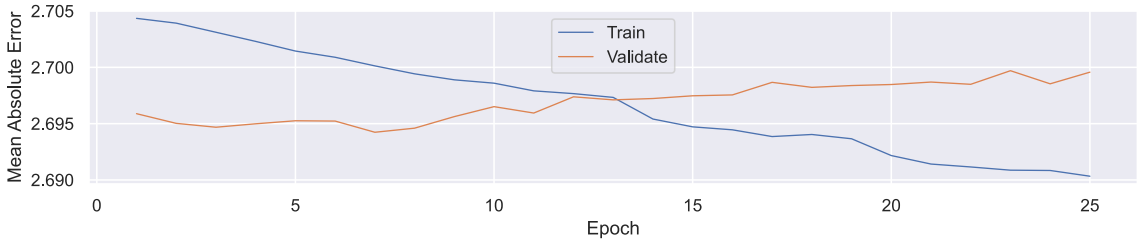
(b) CDIP 243

Figure 25: 6 Layer LSTM Model: Training vs. Validation Mean Absolute Error

5.1.6 Enhanced Transformer Model



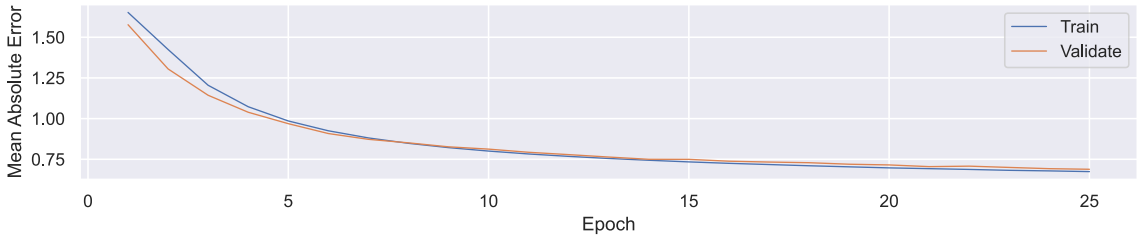
(a) CDIP 225



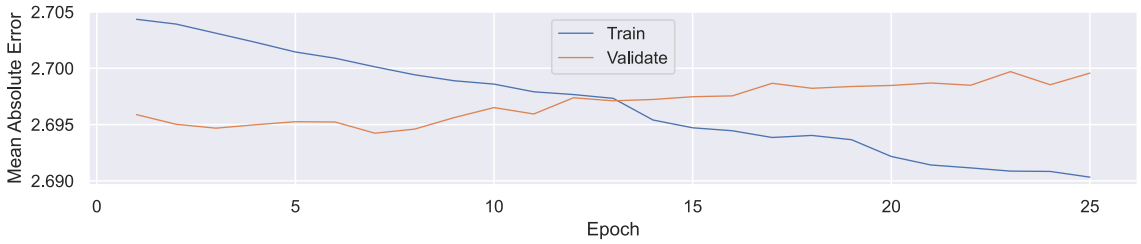
(b) CDIP 243

Figure 26: Enhanced Transformer Model: Training vs. Validation Mean Absolute Error

5.1.7 Enhanced LSTM Model



a(a) CDIP 225



a(b) CDIP 243

Figure 27: Enhanced LSTM Model: Training vs. Validation Mean Absolute Error

5.2 Training Results Summary

The training results, shown in Figure 21 through Figure 27, indicate superior performance from the LSTM-based models. All LSTM variants demonstrated stable learning curves without significant overfitting. Notably, the 100-epoch LSTM model showed continued improvement in both training and validation loss, suggesting potential benefits from extended training periods. The Transformer models, while marginally functional, generally showed poor learning patterns compared to their LSTM counterparts.

6 Results

Model performance was evaluated using a test set comprising 20% of the sampled data. Each model generated predictions using 128-sample input windows, with results mapped back to the original time series for comparison with measured displacements.

6.1 Select Visual Analysis

To examine model performance across different wave conditions, we present a series of representative time series comparisons. Each figure shows a 128-sample window of predictions alongside the corresponding measured wave displacements, with varying combinations of significant wave height (H_{m_0}) and energy period (T_e). These comparisons provide insight into how the models handle different wave states and reveal characteristic prediction patterns.

6.1.1 Baseline LSTM Timeseries

6.1.2 CDIP 225 - Kaneohe Bay, HI

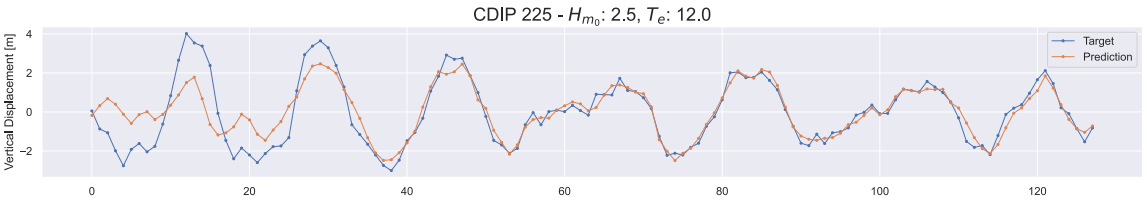


Figure 28: Baseline LSTM - CDIP 225 - Test Section 1

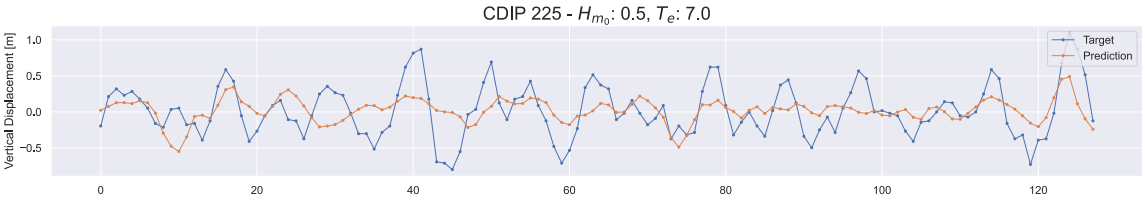


Figure 29: Baseline LSTM - CDIP 225 - Test Section 2

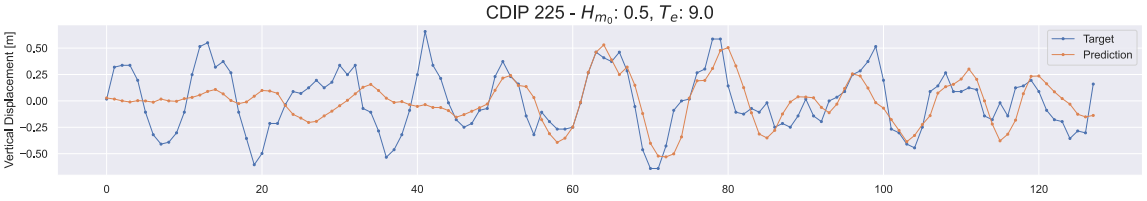


Figure 30: Baseline LSTM - CDIP 225 - Test Section 3

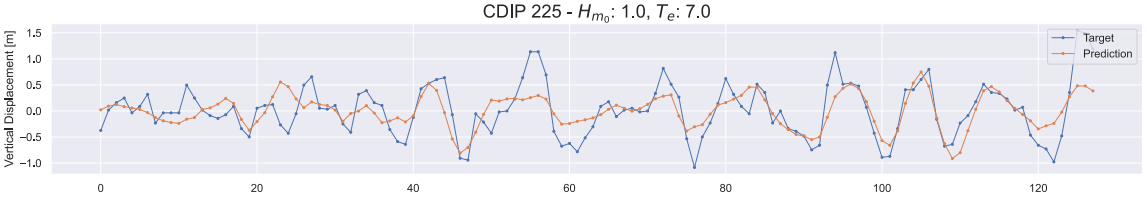


Figure 31: Baseline LSTM - CDIP 225 - Test Section 4

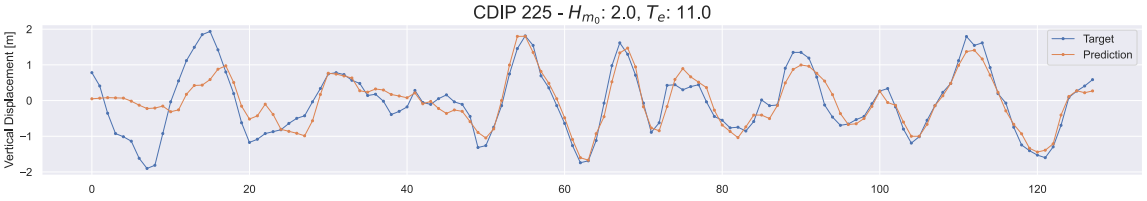


Figure 32: Baseline LSTM - CDIP 225 - Test Section 5

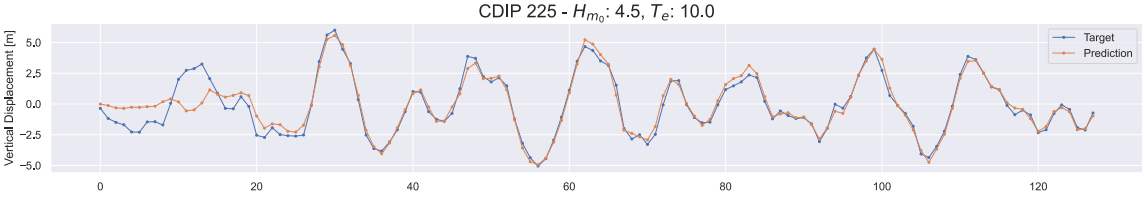


Figure 33: Baseline LSTM - CDIP 225 - Test Section 6

6.1.3 CDIP 243 - Nags Head, NC

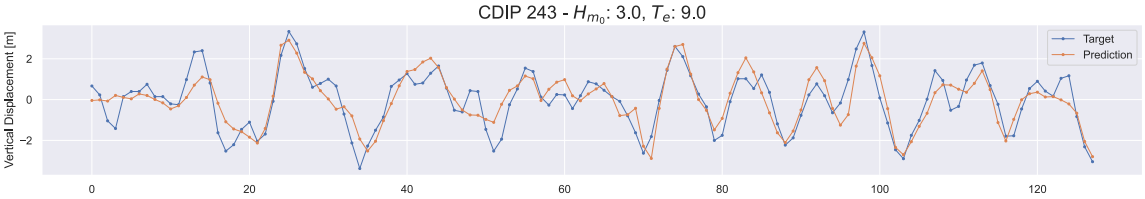


Figure 34: Baseline LSTM - CDIP 243 - Test Section 1

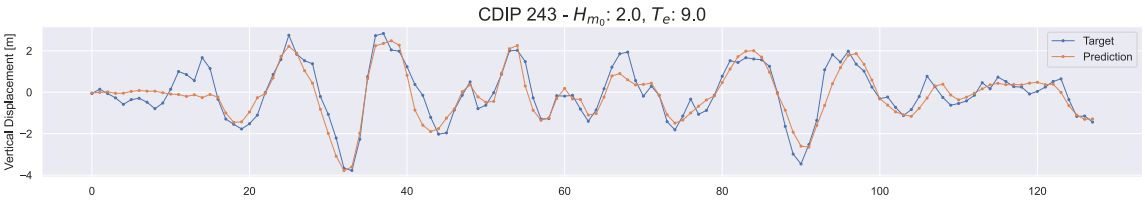


Figure 35: Baseline LSTM - CDIP 243 - Test Section 2

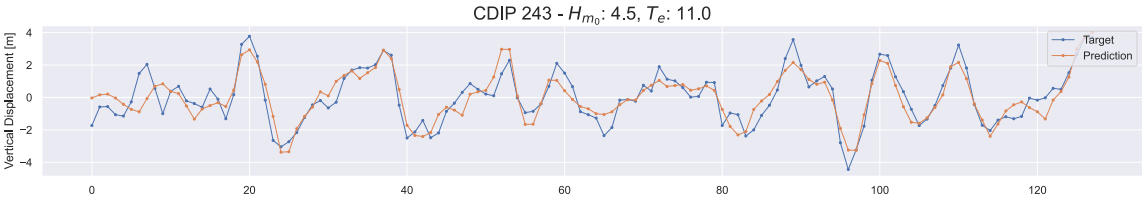


Figure 36: Baseline LSTM - CDIP 243 - Test Section 3

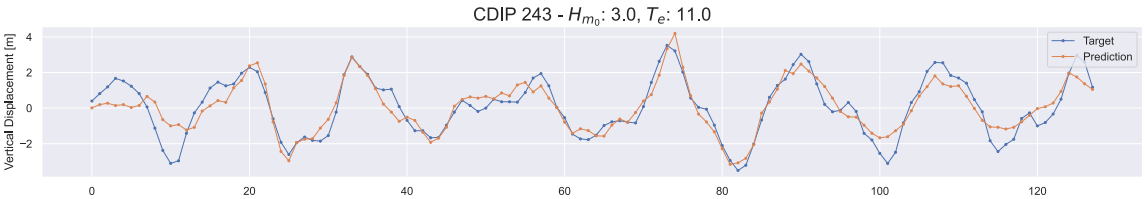


Figure 37: Baseline LSTM - CDIP 243 - Test Section 4

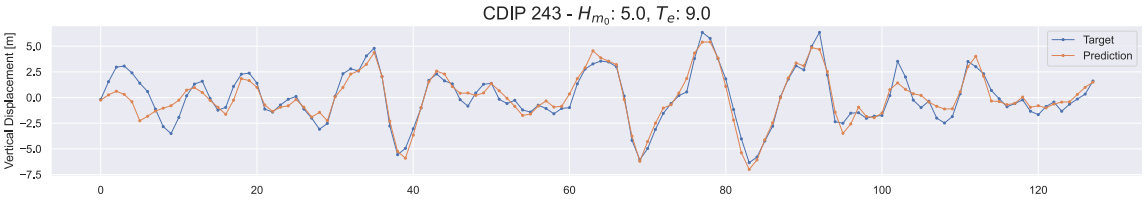


Figure 38: Baseline LSTM - CDIP 243 - Test Section 5

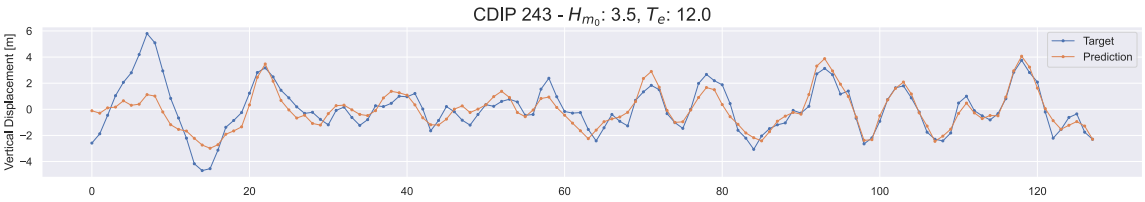
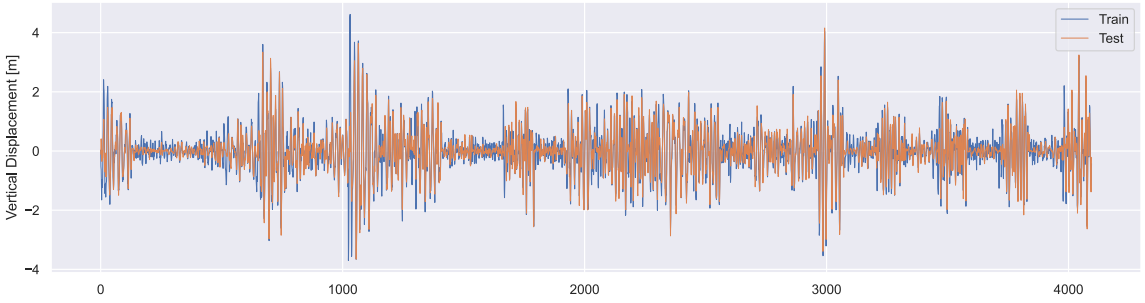


Figure 39: Baseline LSTM - CDIP 243 - Test Section 6

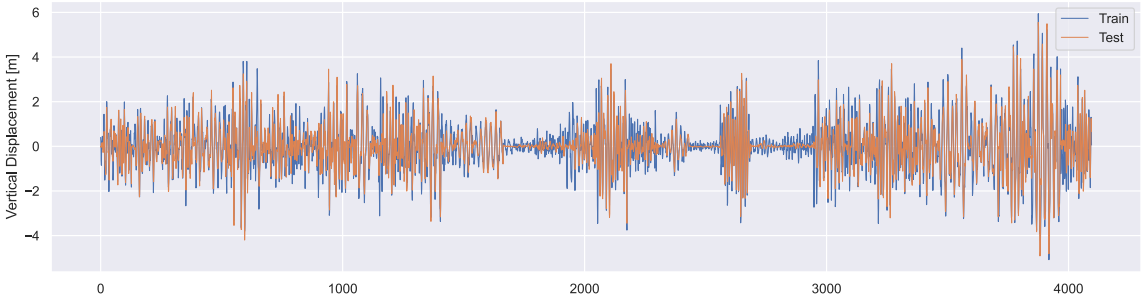
The time series comparisons, shown in Figure 28 through Figure 39, reveal varying prediction quality across different wave conditions. A common pattern emerges where predictions show larger errors at the start of each sequence but improve towards the end. This behavior suggests the models require several time steps to establish the wave state context before making accurate predictions. The LSTM models in particular demonstrate this adaptation, likely due to their ability to build and refine internal state representations as they process the sequence. Overall, the baseline model demonstrates capacity to capture underlying wave patterns, though prediction accuracy varies with wave state conditions.

6.2 Quantitive Visualization

6.2.1 Baseline Model



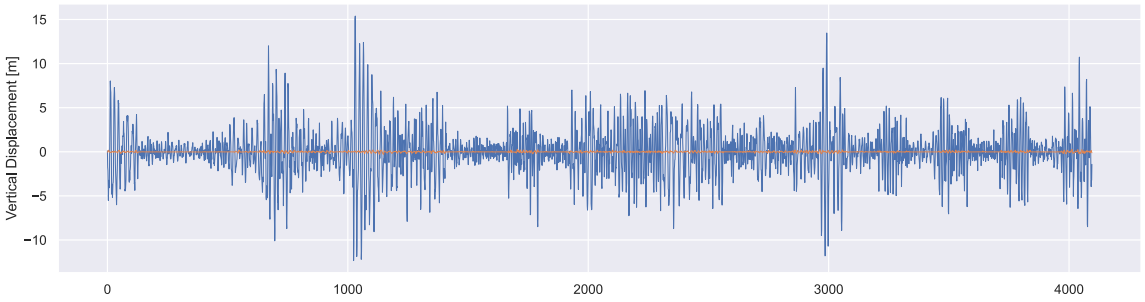
n(a) CDIP 225



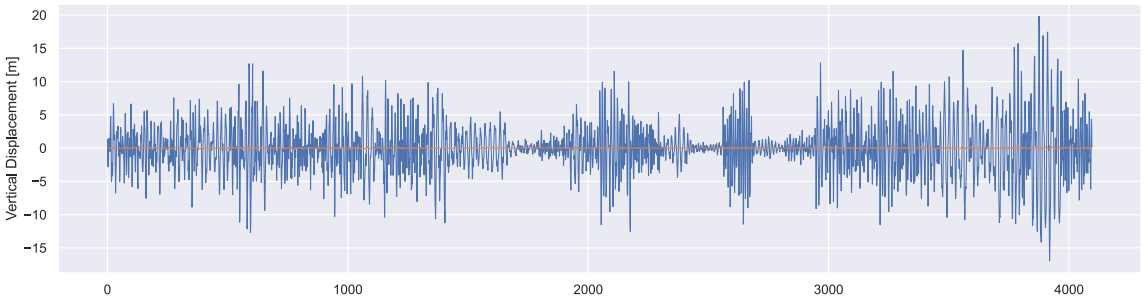
n(b) CDIP 243

Figure 40: Baseline LSTM - All Test Sections

6.2.2 Transformer Model



o(a) CDIP 225

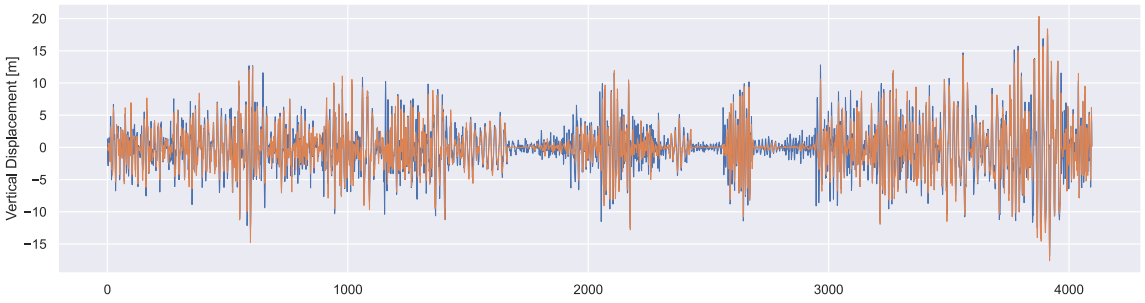


o(b) CDIP 243

Figure 41: Transformer - All Test Sections

6.2.3 LSTM 100 Epoch Model

p(a) CDIP 225

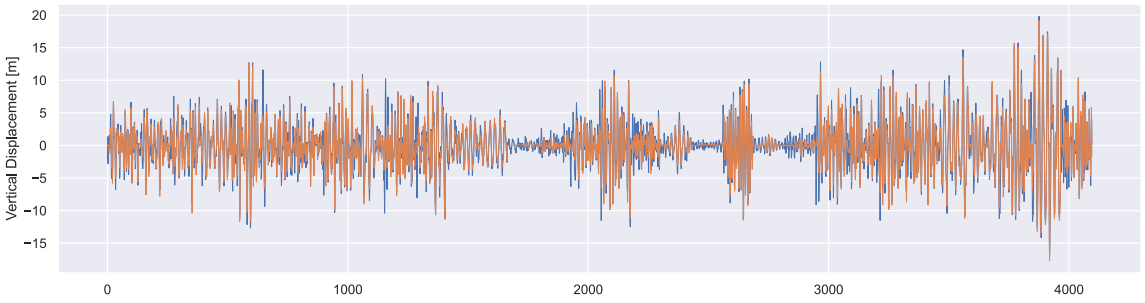


p(b) CDIP 243

Figure 42: LSTM 100 Epoch - All Test Sections

6.2.4 LSTM 4 layers

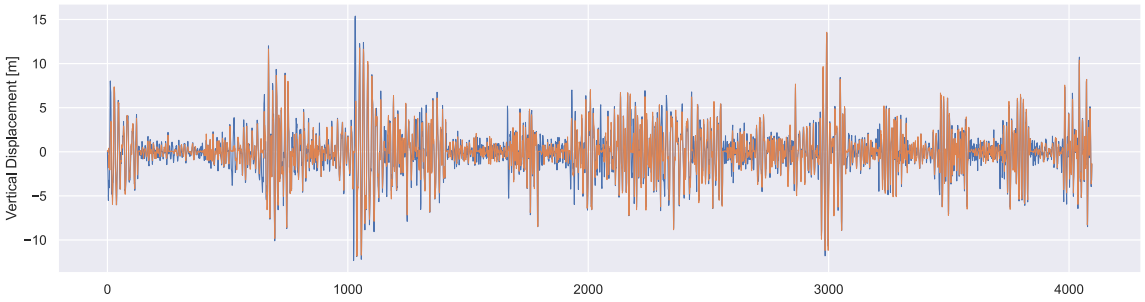
q(a) CDIP 225



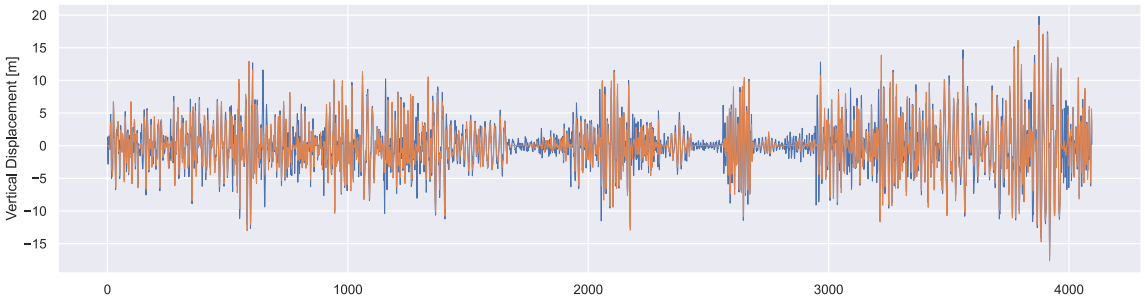
q(b) CDIP 243

Figure 43: LSTM 4 Layer - All Test Sections

6.2.5 LSTM 6 Layer



r(a) CDIP 225

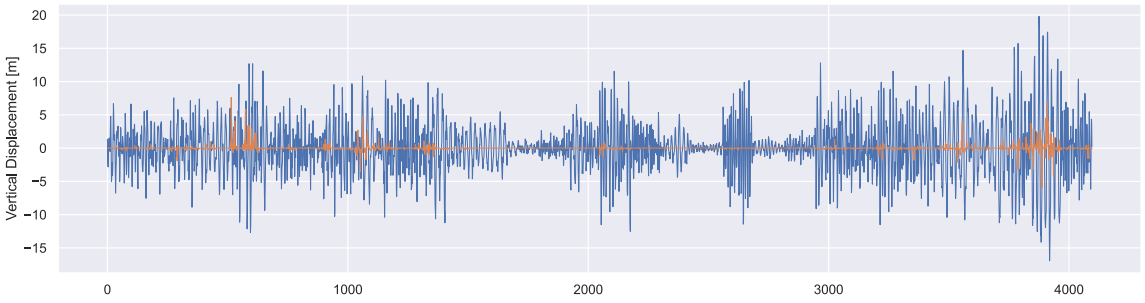


r(b) CDIP 243

Figure 44: LSTM 6 Layer - All Test Sections

6.2.6 Enhanced Transformer

s(a) CDIP 225

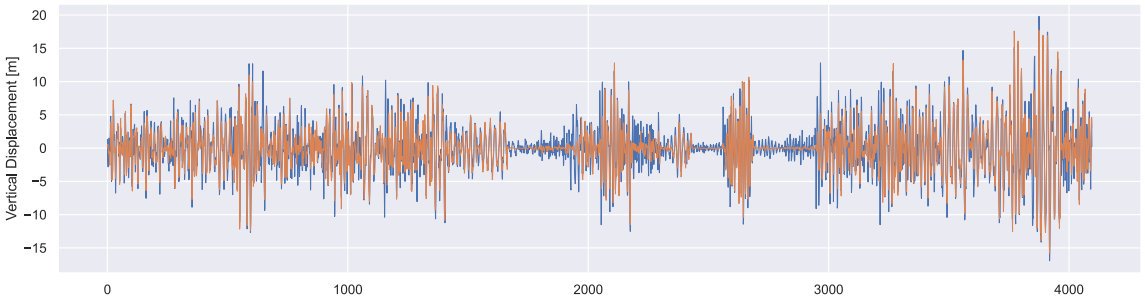


s(b) CDIP 243

Figure 45: Enhanced Transformer - All Test Sections

6.2.7 Enhanced LSTM

t(a) CDIP 225



t(b) CDIP 243

Figure 46: Enhanced LSTM - All Test Sections

Figure 40 through Figure 46 present the complete test set predictions against measured displacements, offering a comprehensive view of model performance across all wave conditions. The LSTM-based architectures demonstrate strong predictive capabilities, closely tracking the measured wave displacements across varying sea states. In particular, the baseline, enhanced, 4-layer, and 6-layer LSTM models show consistent prediction accuracy, suggesting successful learning of the underlying wave dynamics across diverse conditions.

The visual alignment between predictions and measurements for the LSTM models indicates their ability to capture both the frequency and amplitude characteristics of the wave motion. This comprehensive fit across different wave conditions supports the quantitative metrics and demonstrates the models' generalization capabilities. In contrast, the Transformer models show notable deviation from the measured displacements, confirming their limited effectiveness for this prediction task.

6.3 Results Comparison

We evaluated model performance using three metrics: Mean Absolute Error (MAE), coefficient of determination (R^2), and Pearson's correlation coefficient (ρ).

6.3.1 Mean Absolute Error

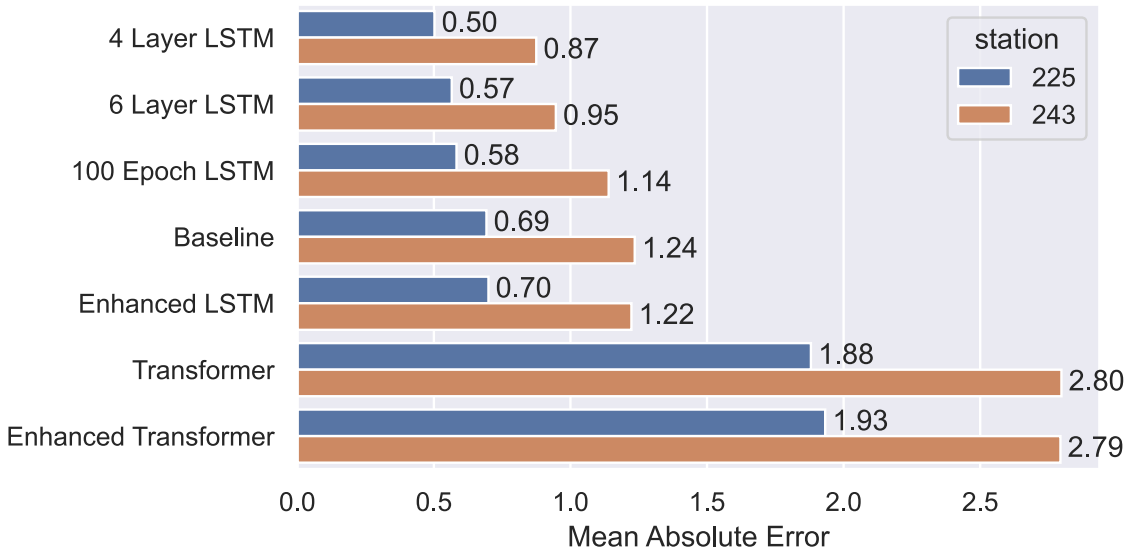


Figure 47: Mean Absolute Error Comparison by Model (Lower is better)

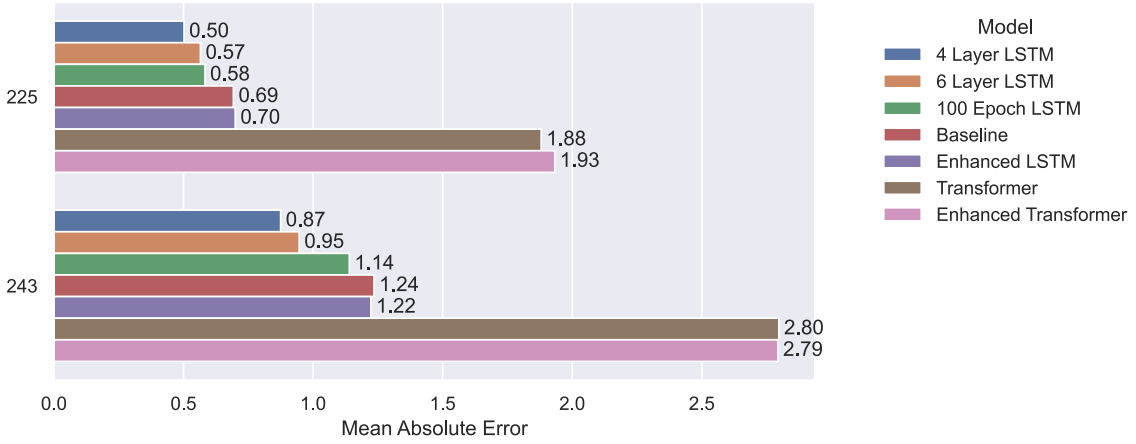


Figure 48: Mean Absolute Error Comparison by Station (Lower is better)

6.3.2 Coefficient of Determination, R^2

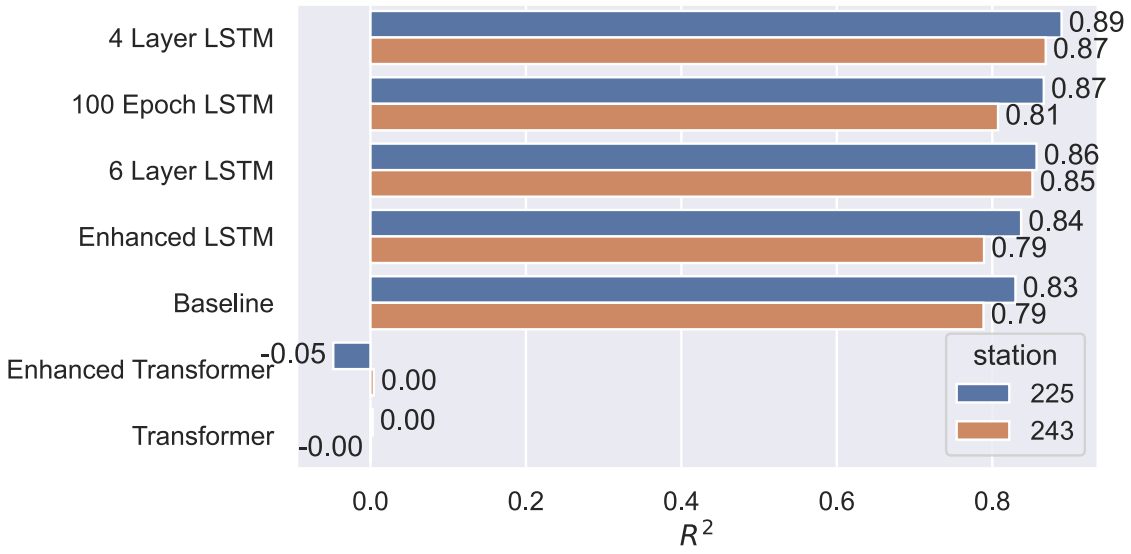


Figure 49: R^2 Comparison by Model

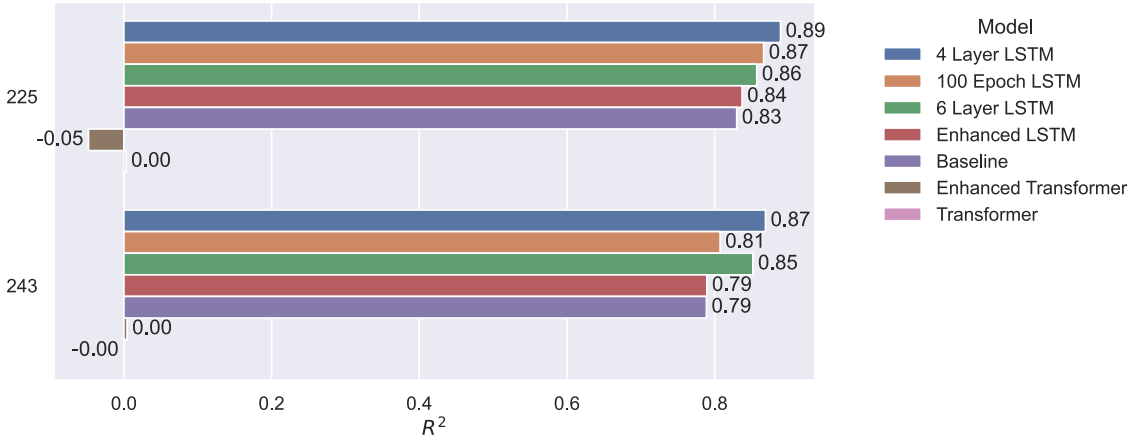


Figure 50: R^2 Comparison by Station

6.3.3 Pearson's Correlation [ρ]

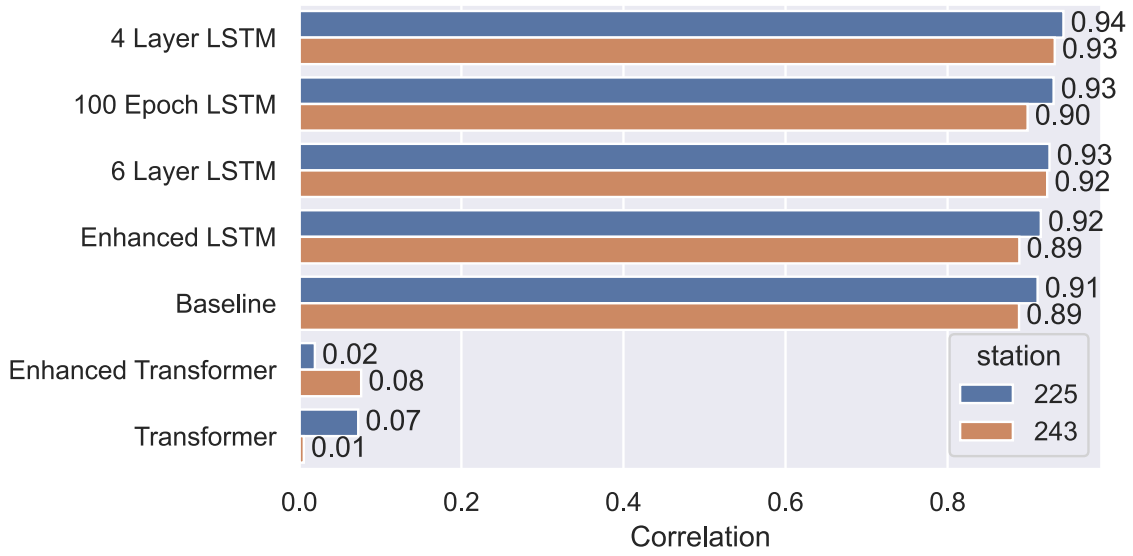


Figure 51: Pearson's Correlation Comparison by Model

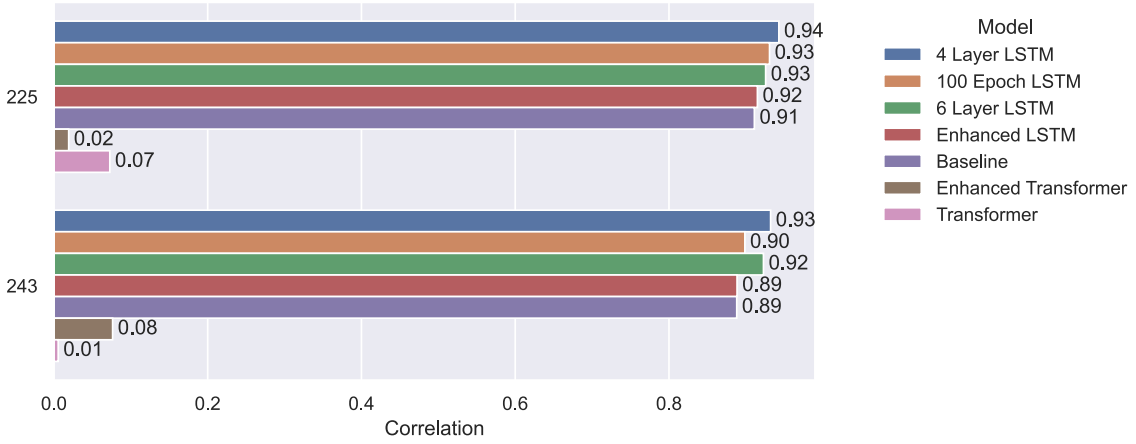


Figure 52: Pearson's Correlation Comparison by Station

Table 4: Results Summary

	label	station	mae	r2	correlation
6	4 Layer LSTM	225	0.502155	0.889149	0.942949
7	4 Layer LSTM	243	0.874835	0.868795	0.932219
4	100 Epoch LSTM	225	0.582889	0.866286	0.930816
8	6 Layer LSTM	225	0.565404	0.856992	0.925749
9	6 Layer LSTM	243	0.946516	0.851715	0.922910
12	Enhanced LSTM	225	0.699822	0.837119	0.915022
0	Baseline	225	0.692493	0.829845	0.911056
5	100 Epoch LSTM	243	1.139891	0.807409	0.898738
13	Enhanced LSTM	243	1.223194	0.789425	0.888552
1	Baseline	243	1.235250	0.788793	0.888162
11	Enhanced Transformer	243	2.793854	0.004303	0.076359
2	Transformer	225	1.880350	0.002838	0.072718
10	Enhanced Transformer	225	1.932608	−0.047880	0.019221
3	Transformer	243	2.797447	−0.000079	0.005579

6.4 Results Summary

We evaluated model performance using three metrics: Mean Absolute Error (MAE), coefficient of determination (R^2), and Pearson's correlation coefficient (ρ). As shown in Figure 47 through Figure 51, the LSTM-based models consistently outperformed the Transformer architectures across all metrics. The 4-layer LSTM achieved the best performance with MAE of $0.50\,\mathrm{m}$ and $0.87\,\mathrm{m}$ for WETS and Nags Head respectively, along with the highest R^2 values (0.89, 0.87) and correlation coefficients (0.94, 0.93). Increasing training epochs from 25 to 100 improved the baseline LSTM's performance, particularly for WETS where the MAE decreased from $0.69\,\mathrm{m}$ to $0.58\,\mathrm{m}$. However, further increasing model depth to 6 layers showed no additional benefit.

Both the basic and enhanced Transformer models performed poorly, with R^2 values near zero and correlation coefficients below 0.1, suggesting they failed to capture meaningful wave patterns. The enhanced LSTM showed no improvement over the baseline model, indicating that the additional architectural complexity did not benefit the wave prediction task.

Across all models, prediction accuracy was consistently better for WETS compared to Nags Head, as shown in Figure 48. This aligns with our earlier observation that WETS exhibits more regular wave patterns, while Nags Head experiences more variable conditions.

7 Conclusion

This project developed deep learning models for wave surface elevation prediction using data from two contrasting locations: the Wave Energy Test Site (WETS) in Hawaii and Nags Head, North Carolina. Through systematic data processing and structured sampling across sea states, we created representative training datasets capturing the range of wave conditions at each site. Multiple neural network architectures were implemented, trained, and tested using PyTorch, with performance evaluated through mean absolute error, coefficient of determination (R^2), and correlation metrics. Comprehensive testing across different sea states and visualization of predicted wave patterns provided quantitative and qualitative assessment of model capabilities. The results demonstrate the viability of neural networks for wave prediction, though with varying degrees of success across different architectures.

7.1 Key Findings

The LSTM-based models consistently outperformed Transformer architectures, with the 4-layer LSTM achieving the best results ($\mathrm{MAE} = 0.50\,\mathrm{m}$ at WETS). Model performance showed sensitivity to architecture choices, where increasing complexity beyond four layers provided diminishing returns. Prediction accuracy was generally better at WETS than Nags Head, reflecting the more regular wave patterns at the Hawaiian site. While our models showed good prediction capabilities, some evidence of overfitting suggests room for improvement in model regularization.

7.2 Lessons Learned

The temporal nature of wave prediction requires careful consideration of model architecture and data preparation. LSTM networks proved particularly effective at capturing wave patterns, while Transformer models, despite their success in other sequential tasks, failed to capture meaningful wave dynamics. Increasing training epochs consistently improved performance, suggesting that extended training periods might yield further improvements. The structured sampling approach across sea states proved valuable for creating representative training datasets.

7.3 Future Work

Several promising directions for future research emerge from this work. Extension to north and east displacement predictions would provide a more complete wave motion model. Physics-informed neural networks could incorporate wave dynamics directly into the learning process. Alternative data sampling strategies and real-time validation would enhance model robustness. Additional coastal locations would test model generalization, while optimized scaling methods might improve prediction accuracy. These enhancements could lead to more reliable wave prediction systems for maritime applications.

Bibliography

[1] J. V. Ringwood, "Wave energy control: status and perspectives 2020 **This paper is based upon work supported by Science Foundation Ireland under Grant no. 13/IA/1886 and Grant No. 12/RC/2302 for the Marine Renewable Ireland (MaREI) centre.," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 12271–12282, 2020, doi: 10.1016/j.ifacol.2020.12.1162.

[2] O. Abdelkhalik *et al.*, "On the control design of wave energy converters with wave prediction," *Journal of Ocean Engineering and Marine Energy*, vol. 2, no. 4, pp. 473–483, doi: 10.1007/s40722-016-0048-4.

[3] Coastal Data Information Program, "Station 225: Kaneohe Bay, WETS, HI - Wave, Sea Surface Temperature, and Ocean Current Time-Series Data." UC San Diego Library Digital Collections, 2023. doi: 10.18437/C7WC72.

[4] Coastal Data Information Program, "Station 243: Nags Head, NC - Wave, Sea Surface Temperature, and Ocean Current Time-Series Data." UC San Diego Library Digital Collections, 2023. doi: 10.18437/C7WC72.

[5] Datawell BV, "Datawell Waverider Reference Manual." 2006.

[6] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[7] A. Vaswani *et al.*, "Attention Is All You Need." [Online]. Available: https://arxiv.org/abs/1706.03762

[8] S. Mandal and N. Prabaharan, "Ocean wave forecasting using recurrent neural networks," *Ocean Engineering*, vol. 33, no. 10, pp. 1401–1410, 2006, doi: 10.1016/j.oceaneng.2005.08.007.

[9] N. K. Kumar, R. Savitha, and A. A. Mamun, "Regional ocean wave height prediction using sequential learning neural networks," *Ocean Engineering*, vol. 129, pp. 605–612, 2017, doi: 10.1016/j.oceaneng.2016.10.033.

[10] J. Ansel *et al.*, "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation," in *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*, ACM, Apr. 2024. doi: 10.1145/3620665.3640366.

[11] H. Sak, A. Senior, and F. Beaufays, "Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition." [Online]. Available: https://arxiv.org/abs/1402.1128

[12] S. Hoyer and H. Joseph, "xarray: N-D labeled Arrays and Datasets in Python," *Journal of Open Research Software*, vol. 5, no. 1, Apr. 2017, doi: 10.5334/jors.148.

[13] The pandas development team, "pandas-dev/pandas: Pandas." [Online]. Available: https://github.com/pandas-dev/pandas

[14] R. Fao *et al.*, "MHKiT-Software/MHKiT-Python: v0.8.2." [Online]. Available: https://doi.org/10.5281/zenodo.13320100

[15] W. Falcon and The PyTorch Lightning team, "PyTorch Lightning." [Online]. Available: https://github.com/Lightning-AI/lightning