UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

Departamento de Obras Civiles

Predicción de línea de costa por medio de modelos de Deep Learning
The Coastal Parrot Effect

como requisito parcial para optar al titulo de

Ingeniero Civil

y el grado de

Magíster en Ciencias de la Ingeniería Civil

Tesis de Grado y Memoria de Título presentada por

Braulio Alejandro Guajardo Gutiérrez

Profesor Guía
Patricio Catalán

JUNIO 2025

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☐ Memoria o trabajo de título; ☒ Tesis de Postgrado;

Título del trabajo: Prediccion de linea de costa por medio de modelos de deep learning (The Coastal Parrot Effect)

Nombre del candidato(a): Braulio Alejandro Guajardo Gutierrez

Carrera / Grado: Ingenieria civil / Magister en ciencias de la ingenieria civil

Campus: Casa Central Valparaiso **;** **Departamento**: Obras civiles

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Patricio Catalan, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 18/06/2025 ; **Firma:** Patricio Catalán

Estudiante o Candidato(a):

Fecha: 18/06/2025 ; **Firma:**

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

UNIVERSIDAD TECNICA
FEDERICO SANTA MARIA

TÍTULO DE LA TESIS:

Predicción de línea de costa por medio de modelos de Deep Learning The Coastal Parrot Effect

AUTOR:

Braulio Alejandro Guajardo Gutiérrez

TRABAJO DE TESIS, presentado en cumplimiento parcial de los requisitos para el **Grado de Magíster en Ciencias de la Ingeniería Civil** de la Universidad Técnica Federico Santa María.

	Nombre	Firma
Profesor Guía	Patricio A. Catalán	
Miembro 1 Comisión	Rodrigo Cienfuegos Carrasco	
Miembro 2 Comisión	Ricardo Ñanculef Alegría	

(Valparaíso/Santiago), Chile, 06/06/2025

Contents

Predicción de línea de costa por medio de modelos de Deep Learning
The Coastal Parrot Effect

Braulio Alejandro Guajardo Gutiérrez[1], Patricio Catalán[1]

[1] Universidad Técnica Federico Santa María

Abstract

Beaches are dynamic natural buffers and vital engines of economic and social development, yet they are increasingly threatened by rising sea levels and intensifying extreme weather events. Accurately forecasting shoreline evolution is essential for sustainable coastal management and climate adaptation. While classical models often fail to capture the nonlinear and abrupt nature of coastal dynamics, recent deep learning (DL) approaches—particularly CNN and CNN-LSTM—have shown promising results. However, Transformer-based models remain largely unexplored. More critically, the balance between physical understanding and neural network flexibility is still an open question in coastal modeling.

This study evaluates the Transformer-based architecture PatchTST against CNN and CNN-LSTM under diverse input configurations that vary in the inclusion of hydrodynamic variables, shoreline data, and the target variable itself. The model is designed to avoid direct use of the target variable, instead relying on shoreline statistics to center predictions, forcing the extraction of meaningful patterns from exogenous inputs. A Channel Attention mechanism is also introduced to dynamically assess input relevance. Results show that PatchTST outperforms baseline models when shoreline information is included, but also reveal the trade-offs faced when limiting physical priors. These findings highlight the importance of hybrid strategies that balance data-driven modeling with domain knowledge, offering new insights into the design of robust and explainable forecasting tools for coastal environments.

keywords: Shoreline prediction, Coastal Morphodynamics, Deep Learning, Transformer Models, CNN, CNN-LSTM

1 Introduction

Since the beginning of civilization, coastal areas have been vital centers for the economic, cultural, and social development of human communities, acting as meeting points, trade, and progress (Gillis, 2012). Currently, nearly 40% of the world's population resides in coastal regions, underscoring their importance as drivers of human and economic development (United Nations, 2017). In addition to their socio-economic role, beaches act as natural barriers that dissipate ocean energy, protecting coastal infrastructure and populations from extreme events (Temmerman et al., 2013).

However, this natural balance is increasingly threatened. Rising sea levels and intensified storms are projected to cause up to 67% of beaches in regions like Southern California to erode by 2100 (Vitousek et al., 2017). In Europe, projections indicate average shoreline retreats nearing 100 meters on sandy coasts, with substantial land loss expected under high-emission scenarios (Athanasiou et al., 2020). Globally, nearly 50% of sandy coastlines could vanish if adequate mitigation measures are not implemented (Vousdoukas et al., 2020), raising urgent concerns about the future of coastal systems (Watkiss et al., 2019).

Climate change is intensifying these threats primarily through sea level rise and more frequent extreme storms, significantly increasing erosion rates and the vulnerability of coastal populations (Behera, 2024; Paprotny et al., 2024). In the United States, floods that currently occur once every 50 years could become annual events in many coastal zones by 2050 (Barnard et al., 2020). Furthermore, accelerated geomorphological transformations threaten not only natural ecosystems but also critical infrastructure and the livelihoods of millions (Barnard et al., 2019), with disproportionate impacts expected in densely populated and resource-limited areas (Masson-Delmotte et al., 2021).

These rapid changes highlight the urgent need for advanced modeling tools capable of capturing complex coastal dynamics. Nevertheless, coastal monitoring and modeling remain challenging due to the limited availability of high-resolution spatial and temporal data (Turner et al., 2016). The scarcity of long-term records restricts the analysis of shoreline trends and variability (Castelle et al., 2020), and factors such as seasonal changes and extreme weather affect both the quality and continuity of observations (Ludka et al., 2019). These limitations have often led researchers to focus on isolated sites, reducing the generalizability of findings.

The rise of remote sensing technologies has partially mitigated these constraints. Satellite imagery provides large-scale, continuous spatial coverage through platforms like Google Earth Engine (Gorelick et al., 2017), enabling global shoreline analysis with high temporal consistency (Luijendijk et al., 2018). Additionally, tools such as CoastSat facilitate the extraction of time series of coastline positions with sub-decadal precision (Vos et al., 2019). Beach-installed camera systems have also proven essential for obtaining high-frequency coastal data, which, when combined with machine learning, allow for real-time detection of wave and shoreline features (Stringari et al., 2019). Recent applications in Chile demonstrate the effectiveness of these techniques for monitoring dynamic beach environments (Cienfuegos et al., 2023).

Despite these technological advances, classical models still struggle to forecast extreme shoreline behaviors. Equilibrium-based models, while effective in capturing long-term trends, fail to reproduce rapid morphological changes due to their reliance on stationary assumptions (Davidson et al., 2013). As demonstrated by Montaño et al. (2020), even a suite of 19 numerical models showed limited predictive capacity for short-term erosion events, particularly when operating outside their calibration periods. These challenges are amplified by rising wave energy linked to ocean warming (Reguero, 2019), suggesting that models based on fixed dynamics may no longer be sufficient to address future coastal risks.

In response to these limitations, data-driven methods such as Machine Learning (ML) and Deep

Learning (DL) have emerged as promising alternatives (Goldstein et al., 2019). These approaches address the high complexity and uncertainty of coastal systems more flexibly than traditional physically based models (Chang & Lai, 2014; Larson & Kraus, 1995; Stive & de Vriend, 2002). Neural networks, particularly when integrated with satellite and geospatial data, have improved erosion risk assessments even under low-resolution or noisy data conditions (Ahmed et al., 2021; Chen et al., 2022), contributing to more effective planning and management.

Initial DL applications employed time series forecasting techniques like NARNET and NARXNET in beaches such as Narrabeen, Australia (Zeinali et al., 2021). These were soon outperformed by more robust architectures, such as LSTM networks (Hochreiter & Schmidhuber, 1997b) and SARIMA models, which showed superior performance in high-variability settings (Yial et al., 2021). More recently, CNN and CNN-LSTM architectures have become the standard for modeling coastal dynamics, achieving strong performance by capturing multi-scale spatial and temporal patterns from high-dimensional inputs (Adusumilli, Cirrito, Engeman, et al., 2024a; Gomez-de la Pena et al., 2023). However, these models still rely heavily on explicit combinations of input variables and have shown limited capability in capturing sudden morphological shifts.

Although DL has made important contributions to shoreline forecasting, more flexible and powerful architectures—such as Transformers—remain underexplored. Originally designed for sequence modeling, Transformers provide advantages such as parallel processing and superior handling of long-range dependencies, making them suitable candidates for time series applications in coastal contexts (Schepper & Van Oordt, 2021).

This study evaluates the performance of the Transformer-based model PatchTST (Nie et al., 2023), comparing it to established CNN and CNN-LSTM architectures developed for coastal forecasting (Gomez-de la Pena et al., 2023). The evaluation is performed under different input configurations, including combinations of hydrodynamic variables and shoreline data.

To evaluate not only predictive performance but also generalization capacity, the study includes two distinct test scenarios. While the first test serves as a standard benchmark for comparing model accuracy, the second test is designed to assess model robustness under altered shoreline conditions. In this case, a morphological change in the coastline occurs, affecting the target variable while the forcing hydrodynamic conditions remain similar. This setup allows for examining how well the models adapt to shifts in coastal behavior within the same geographical context.

A central focus of this work is to assess whether increasing model complexity and versatility leads to meaningful gains in predictive capability, particularly in light of findings by Senechal and Coco (2024a), who observed that increasing the depth of simple networks does not necessarily result in better performance. This evaluation aims to clarify whether more sophisticated architectures like Transformers can outperform shallower models and overcome limitations observed in prior DL implementations.

Additionally, this study questions the prevailing assumption that neural networks must emulate physically based models by relying solely on exogenous variables. While physically inspired inputs are valuable, they may constrain the learning capacity of DL models, which are inherently designed to extract complex, nonlinear relationships from data. Thus, this work explores whether hybrid strategies that incorporate both exogenous (hydrodynamic) variables and the target variable (shoreline position) can improve predictions without fully discarding the physical interpretability of the problem.

This dual perspective—centered on both architectural complexity and input configuration—seeks to clarify whether more versatile deep learning models can offer substantial improvements in shoreline prediction. At the same time, it explores how combining data-driven learning with physically meaningful inputs may enhance model generalization without fully detaching from coastal system

dynamics. These questions define the analytical framework of this study and reinforce its broader aim: to contribute to the development of robust, interpretable, and context-aware DL models for coastal morphodynamic forecasting.

2 Background

2.1 Network Training

Neural networks are a class of parameterized function compositions, commonly referred to as layers, that transform an input representation into a new feature space (Hinton, 2007). A neural network consists of L sequential layers, each denoted as l_i, where $i \in \{1, \cdots, L\}$. Each layer contains a set of computational units known as "neurons", which process and transform the incoming data. The output of layer l_i serves as the input to the subsequent layer l_{i+1}, undergoing a non-linear transformation controlled by a set of parameters θ_i, commonly referred to as "weights". These weights define how information is propagated through the network, ultimately mapping the input to an output space. Mathematically, given an input x, the transformation performed by a neural network can be expressed as follows:

$$f_L(\theta_L, x) = f_{L-1}(\theta_{L-1}, f_{L-2}(\theta_{L-2}, \ldots, f_1(\theta_1, x))) \tag{1}$$

where each function f_i represents a non-linear transformation applied at layer i. This process, known as forward propagation or the forward pass, defines how input data is processed to generate an output. For simplicity, the dependency on parameters θ is often omitted, leading to the more compact notation $f(x)$ instead of $f(x, \theta)$.

2.1.1 Network Training

The training process of a neural network involves optimizing its weights based on a given dataset. Initially, the weights are randomly initialized (Y. A. LeCun et al., 2012), and forward propagation is applied to obtain model predictions. The difference between these predictions and the ground truth labels is quantified using a loss function, which provides a measure of error. To iteratively minimize this error, gradient-based optimization techniques such as gradient descent are employed (Y. A. LeCun et al., 2012). The backpropagation algorithm (Rojas & Rojas, 1996) is used to compute gradients of the loss function with respect to the network parameters, propagating the error backward through the layers and updating the weights accordingly.

2.1.2 Model Evaluation

Once training is complete, the network's performance is assessed using a separate test set. During this phase, the model processes previously unseen data by applying forward propagation to generate predictions. The generalization ability of the model is then evaluated using appropriate performance metrics, which depend on the specific task and dataset (Goyo et al., 2024). These metrics provide insights into the model's predictive accuracy and its ability to perform effectively on new data.

2.2 Neural Network Architectures

This subsection presents some of the most commonly used neural network architectures, which are employed throughout this study.

2.2.1 Multi-Layer Perceptron

The Multi-Layer Perceptron (MLP) is one of the simplest and most widely used neural network architectures (Gardner & Dorling, 1998). It is also referred to as a fully connected (FC) network, as each neuron in a given layer l_i is connected to every neuron in the preceding layer l_{i-1}, where $i \in \{1, \cdots, L\}$.

Given a time series input $x \in D$, where D is our dataset, the transformation applied by an MLP can be expressed as:

$$a_{l_i} = f(\omega_{l_i} \cdot x + b) \tag{2}$$

where ω_{l_i} represents the weight parameters, b is the bias term, and a_{l_i} denotes the activation output of the non-linear activation function in layer l_i.

In forecasting tasks, the final layer produces a continuous-valued output representing the predicted value(s) for future time steps. Unlike classification tasks, where the output represents probability distributions over discrete categories, regression models aim to minimize the difference between predicted and actual values.

The weights in Equation (2) are optimized through an iterative learning process that minimizes a suitable loss function. In time series forecasting, a commonly used loss function is the Mean Squared Error (MSE), defined as:

$$\mathscr{L}_s(x, y) = \frac{1}{T} \sum_{t=1}^{T} (y_t - \hat{y}_t)^2 \tag{3}$$

where $\mathscr{L}_s(x, y)$ measures the squared difference between the predicted values $\hat{y}_t$ and the actual target values y_t over T forecasted time steps. The overall cost function for a dataset with N samples is given by:

$$J(\Theta) = \frac{1}{N} \sum_{n=1}^{N} \mathscr{L}_s(x_n, y_n) \tag{4}$$

where Θ represents the set of all trainable parameters. The optimization process follows an iterative update rule based on gradient descent:

$$\theta = \theta - \alpha \frac{\partial J}{\partial \theta}, \quad \forall \theta \in \Theta \tag{5}$$

where α is the learning rate, controlling the step size of parameter updates. By iteratively adjusting the weights in the direction of the negative gradient, the network progressively minimizes the forecasting error, improving its ability to predict future values accurately.

2.2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of deep neural networks designed to efficiently process sequential and spatially structured data, making them well-suited for time series forecasting (Krizhevsky et al., 2012; Y. LeCun et al., 2015). Their primary advantage lies in their ability to automatically extract hierarchical features through convolution operations, which involve applying learnable filters to the input data. These filters can be one-dimensional (1D), two-dimensional (2D), or higher-dimensional, depending on the nature of the data and the forecasting task.

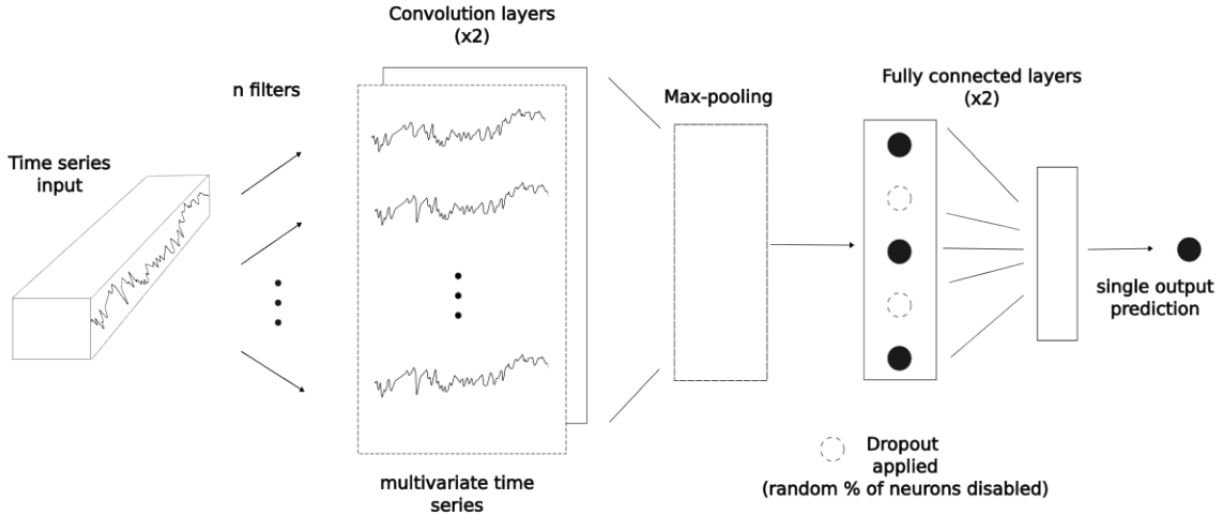


Figure 1: Internal operations performed by a CNN network. Source: Adapted from Gomez-de la Pena et al. (2023)

The convolutional layers in CNNs serve as feature extractors by applying local transformations, capturing temporal dependencies in time series without requiring explicit handcrafted features. Unlike traditional fully connected networks, CNNs leverage weight-sharing mechanisms, making them computationally efficient and less prone to overfitting. Furthermore, they can model time-invariant patterns, allowing them to generalize well even in the presence of noise or missing values.

A typical CNN for time series forecasting consists of multiple convolutional layers that extract relevant temporal patterns, followed by pooling layers that reduce dimensionality while preserving key information. Fully connected layers then aggregate these features to generate the final prediction (see Fig. 1). To enhance generalization, techniques such as dropout (Srivastava et al., 2014) and batch normalization (Ioffe & Szegedy, 2015) are often employed.

While originally designed for image recognition, CNNs have been successfully adapted for forecasting tasks across various domains, including hydrology (Van et al., 2020) and financial time series analysis. Their ability to learn time-dependent patterns without requiring explicit assumptions about the underlying data structure makes them powerful tools for predictive modeling.

In the context of time series forecasting, a convolution operation can be visualized as sliding a one-dimensional filter across the sequence. Each filter detects local temporal patterns and transforms the input time series into a feature representation that is more suitable for prediction. Following definition from Ismail Fawaz (2019), a convolution centered at a given time step t is defined as:

$$C_t = f(\omega * X_{t-l/2:t+l/2} + b) \quad \forall t \in [1, T] \tag{6}$$

where a univariate time series X of length T is convolved (denoted by $*$) with a filter ω of length l. A bias term b is then added before applying a non-linear activation function f (e.g., Rectified Linear Unit, ReLU). The resulting time series C represents a filtered version of the original input.

Applying multiple filters to the input sequence results in a multivariate representation where each dimension corresponds to a learned temporal pattern. This allows CNNs to capture diverse features, such as short-term fluctuations and long-term trends. Because the same filter is applied across all

time steps, CNNs learn time-invariant representations, making them well-suited for generalizing across different forecasting horizons.

By stacking multiple convolutional layers, the model progressively captures higher-order temporal dependencies, enabling accurate and robust forecasting across a variety of time series applications.

2.2.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) networks, introduced by Hochreiter and Schmidhuber (1997a), are a type of recurrent neural network (RNN) designed to process sequential data and capture long-range dependencies in time series. Unlike traditional RNNs, LSTMs incorporate memory cells that allow them to retain information over extended time periods, mitigating issues such as vanishing gradients. This capability makes them particularly well-suited for time series forecasting, where capturing both short- and long-term dependencies is crucial for accurate predictions.

LSTMs achieve this through a gating mechanism that controls the flow of information within the network. These gates determine which information should be retained, updated, or discarded at each time step, allowing efficient state management. The forget gate discards irrelevant past information, the input gate updates the cell state with new data, and the output gate determines which part of the cell state contributes to the final output.

By maintaining an internal state over time, LSTMs can effectively model temporal patterns in time series, making them powerful tools for forecasting applications (see Fig.2). The mathematical formulation of an LSTM cell is as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{7}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{8}$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{9}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{10}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{11}$$

$$h_t = o_t * \tanh(c_t) \tag{12}$$

where σ denotes the sigmoid activation function, $*$ represents the Hadamard product (element-wise multiplication), f_t is the forget gate, i_t is the input gate, $\tilde{c}_t$ is the candidate cell state, c_t is the cell state at time t, o_t is the output gate, and h_t is the hidden state of the LSTM. The model parameters (W and b) are learned during training.

For forecasting, the hidden states of the LSTM are used to generate predictions for future time steps. A common approach is to stack multiple LSTM layers to capture complex temporal dependencies, followed by a fully connected output layer that maps the final hidden state to the predicted value(s). The model is trained by minimizing a loss function, such as the mean squared error (MSE), which quantifies the difference between the predicted and actual values:

$$\mathscr{L} = \frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2 \tag{13}$$

where y_n is the true value and $\hat{y}_n$ is the predicted value at time step n. By iteratively updating its parameters via backpropagation through time (BPTT), the LSTM learns to minimize forecasting errors and improve predictive accuracy.

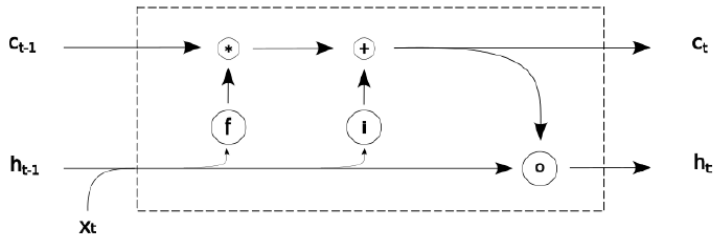


Figure 2: Information flow in an LSTM block, where f is the forget gate (Eq. (7)), i is the input gate (Eq. (8)), o_t the output gate (Eq (11)), c_t and h_t are the cell state and the hidden state, respectively, at time t, this image comes from the work of Gomez-de la Pena et al. (2023)

Due to their ability to model long-range dependencies, LSTMs have been widely applied in time series forecasting across various domains, including energy demand prediction, financial market forecasting, and climate modeling. Their adaptability and effectiveness make them a fundamental tool for capturing complex temporal patterns in sequential data.

2.2.4 CNN-LSTM Neural Networks

A CNN-LSTM is a hybrid neural network architecture that combines the strengths of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. In this model, the CNN component first extracts relevant temporal patterns from the input time series, capturing local dependencies and feature representations. The extracted features are then passed to the LSTM unit, which analyzes how these patterns evolve over time to produce accurate forecasts (see Fig. 3).

This architecture is particularly well-suited for time series forecasting problems where spatial or local dependencies exist within the input sequences. CNN layers efficiently capture short-term patterns through convolutional filters, while LSTMs handle long-range temporal dependencies by maintaining an internal state over multiple time steps. This combination allows CNN-LSTM networks to model both short- and long-term relationships in time series data effectively.

Mathematically, given an input time series $X \in \mathbb{R}^{T \times d}$, where T is the number of time steps and d is the number of input features, the CNN applies a set of filters to generate a feature map representation C:

$$C_t = f(\omega * X_{t-l/2:t+l/2} + b), \quad \forall t \in [1, T] \tag{14}$$

where ω represents the filter weights, $*$ denotes the convolution operation, b is a bias term, and f is a nonlinear activation function. The resulting feature map C is then passed to the LSTM, which updates its hidden state as follows:

$$h_t = \text{LSTM}(C_t, h_{t-1}) \tag{15}$$

where h_t represents the hidden state at time t. The final forecast $\hat{y}$ is obtained through a fully connected output layer:

$$\hat{y}_t = W_o \cdot h_t + b_o \tag{16}$$

CNN-LSTM architectures have been successfully applied in time series forecasting tasks across various domains, such as weather prediction, energy consumption forecasting, and financial time

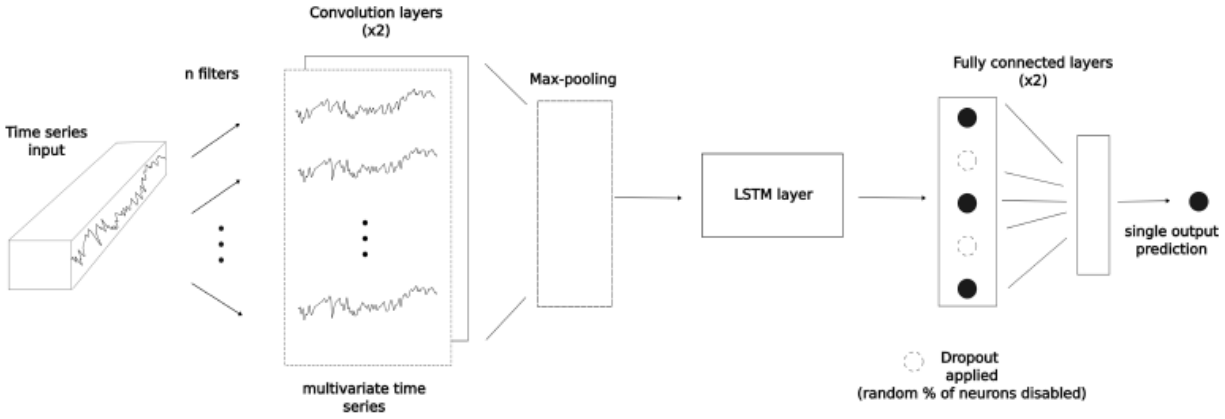


Figure 3: Internal operations of a CNN-LSTM network. The LSTM layer is stacked after the convolutional blocks to capture long-term dependencies in the extracted features. Image adapted from Gomez-de la Pena et al. (2023).

series analysis. Their ability to extract informative features and model complex temporal dependencies makes them a powerful approach for improving forecast accuracy in dynamic environments.

2.2.5 Transformers

Transformers, initially introduced by Vaswani et al. (2017), have revolutionized sequence modeling by replacing recurrence and convolutions with self-attention mechanisms. This architectural shift enables greater parallelization and efficiency in capturing long-range dependencies, making Transformers well-suited for time series forecasting.

Unlike RNNs and LSTMs, which process sequences sequentially, Transformers handle all time steps simultaneously through self-attention. This significantly reduces training time while improving the ability to model long-term dependencies, which is crucial in forecasting applications such as climate modeling and financial predictions.

Transformers employ a scaled dot-product attention mechanism, which computes attention weights as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{17}$$

where Q, K, and V are learned projections of the input sequence. Multi-head attention further enhances this by computing multiple attention outputs in parallel:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{18}$$

These mechanisms allow Transformers to efficiently model both local and global dependencies, addressing the vanishing gradient limitations of recurrent models.

To illustrate how Transformers work, a schematic is included in Fig. 4.

A recent advancement in time series forecasting is PatchTST (Nie et al., 2023), which enhances Transformer efficiency by segmenting time series into patches and enforcing channel independence. Given an input sequence of length L, PatchTST splits it into smaller patches of length P with

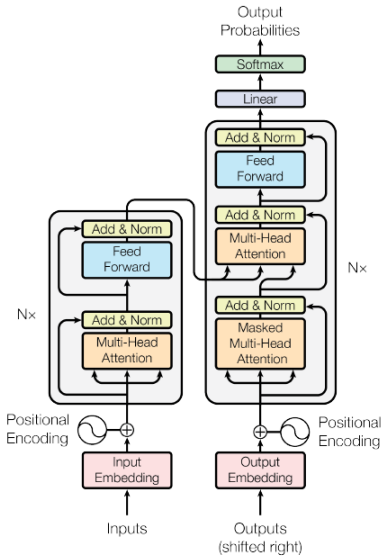


Figure 4: Diagram of how Transformers work.

stride S, reducing computational complexity and enabling longer historical context utilization. The architecture of PatchTST is illustrated in Fig. 5.

For a multivariate time series with M features, each univariate series $x^{(i)}$ is independently fed into the Transformer model. The segmentation process yields a sequence of patches:

$$N = \left\lfloor \frac{L-P}{S} \right\rfloor + 2 \tag{19}$$

where N is the number of patches. The term $+2$ accounts for the inclusion of the initial patch (as in the standard patching formula) and an additional patch that results from padding applied at the end of the sequence when the series length is not exactly divisible by the stride. This reduces the input token length from L to approximately L/S, significantly lowering memory requirements.

The Transformer encoder maps each patch to a latent space of dimension D using a learnable linear projection and positional encoding:

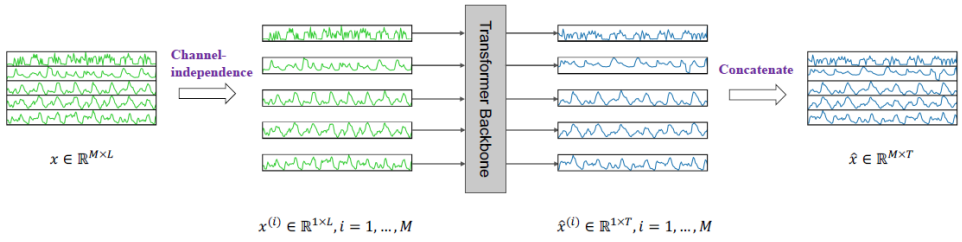
$$x_d^{(i)} = W_p x_p^{(i)} + W_{pos} \tag{20}$$

Multi-head self-attention then processes these representations, followed by residual connections and normalization layers. The final forecast $\hat{x}^{(i)}$ is obtained via a linear output layer:

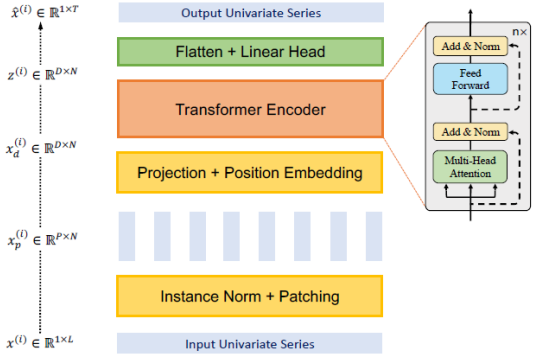
$$\hat{x}^{(i)} = (\hat{x}_{L+1}^{(i)}, ..., \hat{x}_{L+T}^{(i)}) \in \mathbb{R}^{1\times T} \tag{21}$$

PatchTST mitigates distribution shift by applying instance normalization, ensuring consistent feature scaling across different time series samples. The model is trained using mean squared error (MSE) loss:

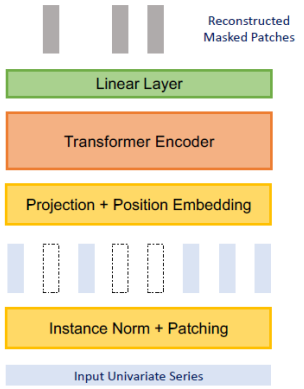
$$L = \mathbb{E}\left[\frac{1}{M}\sum_{i=1}^{M} \|\hat{x}_{L+1:L+T}^{(i)} - x_{L+1:L+T}^{(i)}\|_2^2\right] \tag{22}$$



(a) PatchTST Model Overview



(b) Transformer Backbone (Supervised)



(c) Transformer Backbone (Self-supervised)

Figure 5: PatchTST architecture. (a) Multivariate time series data is split into different channels. They share the same Transformer structure, but the feedforward processes are independent. (b) Each channel univariate time series is passed through an instance normalization operator and segmented into patches. These patches are used as input tokens for the Transformer. (c) Self-supervised representation learning with PatchTST, where patches are randomly selected and set to zero. The masked patches will be reconstructed by the model. Image taken from the work of Nie et al. (2023).

By leveraging patching and channel-wise modeling, PatchTST improves long-range forecasting performance while significantly reducing computational overhead.

3 Related Works

Early efforts to model shoreline evolution were grounded in conceptual frameworks. Wright and Short (1984) introduced a morphodynamic classification of beach states, ranging from reflective to dissipative, based on the dimensionless parameter Ω, which relates wave height to sediment grain size. Building on this, Wright et al. (1985) proposed that beaches tend toward a dynamic equilibrium state influenced by the history of wave energy, suggesting that past conditions play a crucial role in determining present shoreline configurations.

In the mid-2000s, quantitative models emphasizing dynamic equilibrium gained traction. Miller and Dean (2004) proposed a model wherein shorelines evolve toward an equilibrium position under given wave climates. Yates et al. (2009) expanded on this by incorporating feedback mechanisms informed by antecedent wave conditions. A notable advancement was the ShoreFor model developed by Davidson et al. (2013), which introduced a memory parameter (ϕ) to capture hysteresis effects

in shoreline response. Splinter et al. (2014) later demonstrated the transferability of ShoreFor with minimal calibration across multiple sites. Complementarily, Vitousek et al. (2017) integrated both longshore and cross-shore sediment transport to simulate shoreline change under climate change scenarios, while Montaño et al. (2021) introduced the SPADS model, capable of modeling shoreline evolution over storm-scale to decadal timescales.

Despite their contributions, equilibrium-based models face limitations in representing non-linear interactions and multi-scale dynamics. This has spurred interest in data-driven approaches. Calkoen et al. (2021) evaluated eight statistical and machine learning models across over 37,000 global transects using satellite-derived shoreline data. Their results showed that both traditional and machine learning methods outperformed ordinary least squares regression, with ML models such as DeepAR, MQCNN, and DeepSSM offering improvements in computational efficiency and multi-step forecasting capabilities.

At a local scale, Zeinali et al. (2021) used recurrent neural networks (RNNs) to predict monthly shoreline positions at Narrabeen Beach, Australia, achieving reliable performance based solely on historical shoreline data. Senechal and Coco (2024b) explored a different approach, applying a simple feedforward neural network (FFN) using sparse morphological and hydrodynamic parameters extracted from video imagery at Biscarrosse Beach in France. The study showed that even with limited data availability, their ANN model achieved an average prediction error of 6.7 meters for one-year forecasts, effectively capturing both storm-driven and seasonal shoreline changes.

More recently, Gomez-de la Pena et al. (2023) applied convolutional neural networks (CNNs) and hybrid CNN-LSTM models to forecast shoreline positions at Coromandel Peninsula, New Zealand. These models outperformed traditional equilibrium-based frameworks in terms of RMSE and the ability to replicate the statistical distribution of observed shoreline variability. The study also highlighted that CNNs, despite lacking explicit memory mechanisms, performed comparably to CNN-LSTMs, suggesting an implicit capacity to capture temporal dependencies. Furthermore, incorporating the Mielke index (Duveiller et al., 2016) as a loss function improved model performance in terms of variability representation.

Calcraft et al. (2025) investigated whether LSTM models could learn physically meaningful dynamics similar to those used in ShoreFor. Their results showed strong correlations between the internal memory states of the LSTM and the equilibrium parameter Ω_{eq}, indicating that deep learning models are capable of capturing physically interpretable features through data-driven training alone.

Most recently, Repina et al. (2025) conducted a comprehensive evaluation of five reduced-complexity shoreline models using 40 years of monthly survey data from Narrabeen Beach, Australia. Their findings underscored the challenges of simulating both cross-shore and longshore shoreline dynamics at embayed beaches and emphasized the importance of rigorous long-term validation. Hybrid models were identified as promising tools that balance process-based fidelity with computational efficiency, although site-specific calibration remained essential.

Together, these studies highlight the evolving role of machine learning and reduced-complexity models in coastal forecasting. While traditional statistical approaches remain competitive in some settings, modern deep learning and hybrid frameworks increasingly demonstrate superior capabilities in capturing the complexity of shoreline dynamics across multiple timescales. The integration of data-driven methods with domain-specific physical understanding appears essential for improving prediction accuracy, ensuring robustness across coastal settings, and supporting informed coastal management decisions in the face of accelerating climate change.

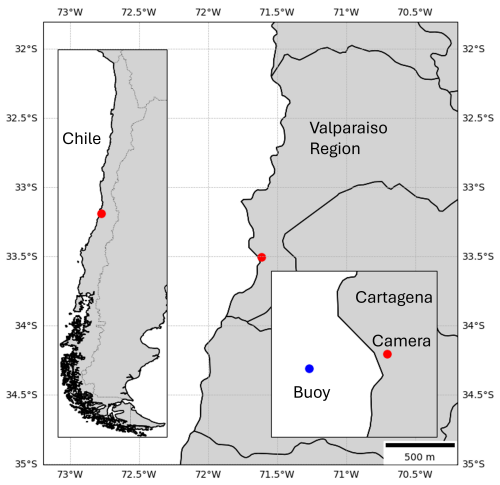


Figure 6: The image shows the location of the beach under study. The red dot represents the camera, and the blue one is the buoy that records hydrodynamic parameters.

4 Methodology

4.1 Data

This study is carried out at Playa Grande de Las Cruces, located in Cartagena, Valparaíso Region, Chile (Fig. 6). It is a sandy beach with an approximate length of 5 kilometers. The data used in this work come from the SIMONA Costa project (Cienfuegos et al., 2023), an initiative aimed at the continuous monitoring and anticipation of the response of coastal systems in Chile. This system has been implemented in various pilot beaches, including Las Cruces in the Valparaíso Region, Punta de Lobos, and the mouth of the Nilahue Estuary in the O'Higgins Region. SIMONA Costa integrates advanced monitoring technologies to generate high spatial and temporal resolution information, contributing to the understanding of coastal dynamics. This provides a dataset that spans from 09/16/2019 to 05/31/2022.

The first main dataset consists of shoreline positions obtained by a camera installed on the beach. This camera produces three 20-minute videos per hour. To represent each hour in the time series, an average of the shoreline position from the first video is used, while the other two serve as backups in case of data loss. This allows the identification of shoreline evolution and results in a time series with hourly resolution.

In this study, four transects were defined on Playa Grande de Las Cruces, referred to as POS1, POS2, POS3, and POS4. These represent the shoreline position at different locations along the beach (Fig. 7). The objective is to develop models capable of forecasting the time series corresponding to transect POS1, using various combinations of input data including hydrodynamic variables and the positions of adjacent transects.

The second dataset corresponds to hydrodynamic variables continuously recorded by a buoy placed offshore (Fig. 6). These variables are obtained through the WaveWatch III wave forecasting model (Tolman, 2009), which has been validated at several points along the Chilean coast (Luarte, 2017). The model is complemented by data from meteorological stations and tide gauges, accessed through the IOC–Sea Level Monitoring database and provided by the Hydrographic and Oceanographic Service of the Chilean Navy (SHOA). Hourly data is recorded 24 hours a day, allowing detailed capture of oceanic and atmospheric conditions. The hydrodynamic variables and those

Figure 7: Image of the beach showing the four transects from which shoreline positions are obtained.

derived from them are listed below:

- Significant wave height (Hm0 [m])
- Energy period (Te [s])
- Mean periods (Tm1 [s], Tm2 [s])
- Peak period (Tp [s])
- Mean wave direction (Dm [°])
- Peak wave direction (Dp [°])
- Angular spreading (Spr [°])
- Wave power (P [kW/m])
- Deepwater wave power (P0 [kW/m])
- Wind speed in U direction (Uwind [m/s])
- Wind speed in V direction (Vwind [m/s])
- Total wind speed (WindSpeed [m/s])
- Wind direction (WindDirection [°])
- Wave steepness (Wave steepness [-])

Coastline monitoring is restricted to daylight hours due to limitations of the camera system. Additionally, occasional data losses occur due to adverse weather or technical failures. A significant interruption was recorded during the period from 03/16/2020 to 09/19/2020 (Fig. 8), when the camera was out of service.

Additionally, the beach underwent a morphological transformation due to the deposition of exogenous sediments (Fig. 9). This process led to coastal accretion and a shift in the equilibrium shoreline position across the analyzed transects, as observed by the upward trend in the shoreline time series between 01/01/2021 and 31/03/2021.

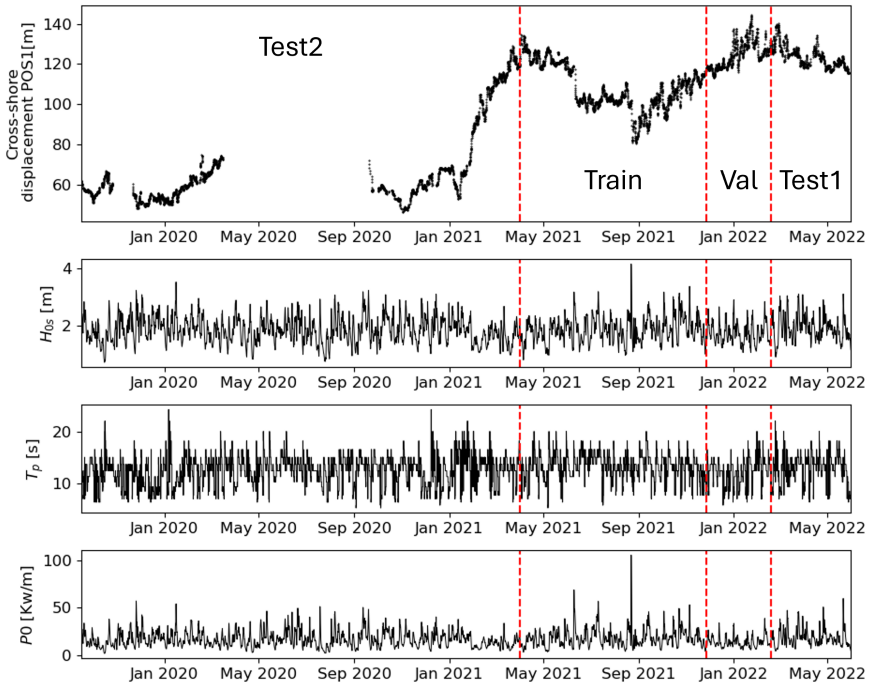


Figure 8: Time series of transect POS1, whose behavior is the prediction target, alongside relevant hydrodynamic variables such as significant wave height, peak period, and deepwater wave power.

(a) Beach morphology before the change.

(b) Beach morphology after the change.

Figure 9: Morphological transformation due to exogenous sediment input, leading to coastal accretion.

4.2 Experimental Design

In this study, initial efforts focused on predicting the shoreline rate of change (dx/dt) over a time interval (Δt). However, as noted by Calcraft et al. (2025) in their work on shoreline modeling using LSTM networks, this approach posed significant challenges. In particular, transforming the data into a differential form appeared to distort the inherent relationships among the input variables, limiting the model's ability to identify meaningful correlations and make accurate predictions. Consequently, the decision was made to predict the absolute shoreline position instead. This adjustment aligns

with the findings of Calcraft et al. (2025), who concluded that the loss of intrinsic connections in transformed data leads to suboptimal model performance. By focusing on absolute positions, the models were better able to preserve and exploit the contextual dependencies present in shoreline behavior.

The morphological change observed on the beach marked a distinct shift in coastal dynamics. To avoid contaminating the training process with patterns associated with previous morphological states, model training was restricted to data collected after the transformation. This ensures that the learned relationships correspond to the current equilibrium conditions of the beach.

Seven distinct combinations of input variables were evaluated to assess how different types of information and model architectures affect predictive performance. These case studies are as follows:

1. Hydrodynamic variables only.
2. Adjacent shoreline positions only (POS2, POS3, POS4).
3. Hydrodynamic variables + adjacent positions.
4. Hydrodynamic variables + the target position (POS1).
5. Adjacent positions + the target position (POS1).
6. Hydrodynamic variables + adjacent positions + the target position (POS1).
7. Target position (POS1) only.

The first three scenarios exclude the target variable (POS1) from the inputs, whereas the last four explicitly incorporate it. This design aims to evaluate the impact of including POS1 on model performance and to determine whether the models can infer relationships between hydrodynamic conditions, adjacent shoreline positions, and the target position. In all experiments, models operate using a fixed-length input segment—referred to here as the context window—to predict future shoreline positions. All models are strictly autoregressive and are not given access to future information beyond the prediction horizon. The architectures proposed by Gomez-de la Pena et al. (2023) use a fixed context window of 40, while PatchTST treats it as a tunable hyperparameter. This distinction allows for an investigation into how varying the length of past data influences forecasting skill.

Following the segmentation of the dataset after the morphological transformation, the data was further split chronologically into 60% for training, 20% for validation, and 20% for testing. Additionally, an extra set of pre-transformation data was retained for a second round of testing (Fig. 8). This additional test aims to assess the model's generalization ability across distinct morphological conditions and to examine model behavior throughout the transition phase.

The dataset division respects chronological order, as beach response is known to be influenced by previous states (D'Anna et al., 2022). Moreover, prior analysis suggests that whether data is split randomly or sequentially has negligible effect on predictive accuracy, indicating that shuffling the data is not a critical factor in improving model performance (Adusumilli, Cirrito, Engeman, et al., 2024b).

4.3 Data Preprocessing

To standardize the input data and facilitate model training, a MinMaxScaler was applied. This method scales the values of each variable to fall within the range [0, 1], preserving their relative

distributions. The scaler was fitted using only the training data and subsequently applied to the validation and test sets to ensure consistency. The transformation applied is defined as:

$$X_{\text{scaled}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{23}$$

where X is the original value, and $X_{\min}$ and $X_{\max}$ correspond to the minimum and maximum values computed from the training set.

Once the data is scaled, it is used as input for the models. After prediction, the outputs—still in normalized scale—are transformed back to the original scale using the inverse of the MinMax transformation:

$$X = X_{\text{scaled}} \cdot (X_{\max} - X_{\min}) + X_{\min} \tag{24}$$

This inverse transformation allows the predicted shoreline positions to be directly compared with the original observations in physical units (e.g., meters), enabling the calculation of evaluation metrics such as RMSE and MAE.

4.4 Baselines

4.4.1 CNN Architecture

The baseline Convolutional Neural Network (CNN) architecture implemented in this study builds upon the work of Gomez-de la Pena et al. (2023), which demonstrated strong performance in shoreline forecasting tasks, particularly in terms of root mean square error (RMSE) and shoreline variability representation. The original model was adapted to support the seven distinct input configurations defined in this study.

Consistent with the referenced work, all CNN models were trained using the Adam optimizer and employed the Mielke index as the loss function, due to its robustness in capturing the oscillatory behavior characteristic of coastal time series.

Ten hyperparameter configurations were tested, corresponding to those proposed in the original work by Gomez-de la Pena et al. (2023) (see Table 1). Each model was trained for a maximum of 30 epochs, with early stopping based on validation loss improvements using a patience of 10 epochs. Upon early stopping, the model restored the weights associated with the best validation performance. When historical shoreline position data (POS1) was part of the input, it was included as an additional input channel using the same context window length.

4.4.2 CNN-LSTM Architecture

The hybrid CNN-LSTM model implemented in this study also follows the methodology proposed by Gomez-de la Pena et al. (2023), combining CNN layers for short-term pattern extraction with Long Short-Term Memory (LSTM) layers to capture long-range temporal dependencies. The architecture was adapted and validated using the seven different input datasets defined for the experiments.

As with the CNN baseline, the Mielke index was used exclusively as the loss function, due to its demonstrated ability to effectively represent oscillatory coastal dynamics.

For hyperparameter tuning, ten parameter configurations were evaluated, based on those from Gomez-de la Pena et al. (2023) (see Table 2). The same training protocol was used, including the Adam optimizer, early stopping with a patience of 10 epochs, and a maximum training duration of 30 epochs. When early stopping was triggered, the model restored the best weights. In scenarios

Table 1: Hyperparameter combinations evaluated for the CNN architecture. Each configuration was tested with a maximum of 30 training epochs and early stopping.

Configurations	Filters	Kernel Size	Dropout	Epochs	Batch Size
1	57	72	0.4	30	16
2	75	88	0.3	30	60
3	79	88	0.4	30	60
4	49	72	0.0	30	16
5	2	64	0.0	30	32
6	27	64	0.3	30	60
7	71	88	0.2	30	60
8	74	88	0.3	30	32
9	78	88	0.4	30	32
10	58	72	0.3	30	32

including historical shoreline data (POS1), this input was integrated as an additional channel with the same context window length as other input features.

Table 2: Hyperparameter combinations evaluated for the CNN-LSTM architecture. The LSTM memory size was fixed in most cases to 200 units.

Configurations	Filters	Kernel Size	LSTM Memory	Dropout	Epochs	Batch Size
1	19	64	200	0.2	30	60
2	119	88	200	0.4	30	60
3	23	64	200	0.4	30	60
4	66	72	200	0.2	30	32
5	174	72	200	0.3	30	32
6	114	88	200	0.2	30	32
7	43	64	200	0.2	30	60
8	167	72	200	0.3	30	60
9	147	64	100	0.3	30	60
10	150	64	200	0.3	30	32

4.4.3 PatchTST Architecture

In this study, the PatchTST model (Nie et al., 2023), a Transformer-based architecture specifically optimized for multivariate time series forecasting, was implemented. PatchTST was selected for its proven ability to capture long-range temporal dependencies and its superior predictive performance compared to other Transformer-based methods.

PatchTST introduces two core innovations: segmentation of the input time series into subsequences, known as *patching*, and the concept of *channel independence*, wherein each input variable (or channel) is processed independently by the Transformer backbone. While the original PatchTST architecture is designed for multi-input, multi-output scenarios, the present study required an adaptation to predict a single target variable (shoreline position) from multiple inputs (Figure 10). To accommodate this, a novel architectural component, referred to here as the *Channel Attention* block, was designed and integrated into the PatchTST framework. This module employs multi-head self-attention to dynamically weigh the relative contribution of each input channel, producing a unified prediction. After the Transformer backbone generates channel-specific representations, the Channel Attention block aggregates them, emphasizing the most informative features for shoreline predic-

tion. This addition enhances both predictive performance and interpretability by identifying which variables are most influential.

A critical component of this modified implementation is the clear distinction between two predictive scenarios, depending on whether historical data for the target variable is available:

- **Scenario without Target Variable:** When no historical shoreline data is available, the model relies entirely on input variables for prediction. A context window containing prior time steps is used to infer the next shoreline position. In this configuration, the model must learn both the scale and variability of shoreline movement from the input data alone.
- **Scenario with Target Variable Available:** When historical shoreline data is available, the model processes input variables as before, but additionally incorporates a separate *normalization window* containing past observations of the target variable. This window may be shorter than the context window and is not fed directly into the model. Instead, its statistical descriptors—specifically, the most recent shoreline value and its standard deviation—are used post-prediction to denormalize the model output. This approach allows the model to focus on predicting relative deviations from recent shoreline states, while the normalization window provides the necessary statistical context for recovering absolute values.

The introduction of a distinct normalization window, rather than incorporating shoreline history directly into the model input, addresses a critical issue observed during preliminary experiments. When the target variable was included as an input channel, hyperparameter optimization consistently selected a minimal context window (often just one time step). This behavior was due to the resulting low standard deviation, which led the model to produce trivial predictions that mirrored the most recent value, rather than learning true dynamics. By isolating shoreline history within a separate normalization window, used exclusively for output rescaling, the model is explicitly guided to learn meaningful relationships from the inputs alone. Additionally, the Channel Attention block reinforces this approach by identifying and prioritizing the most relevant variables, thus improving both performance and interpretability of the model's reasoning process regarding coastal dynamics.

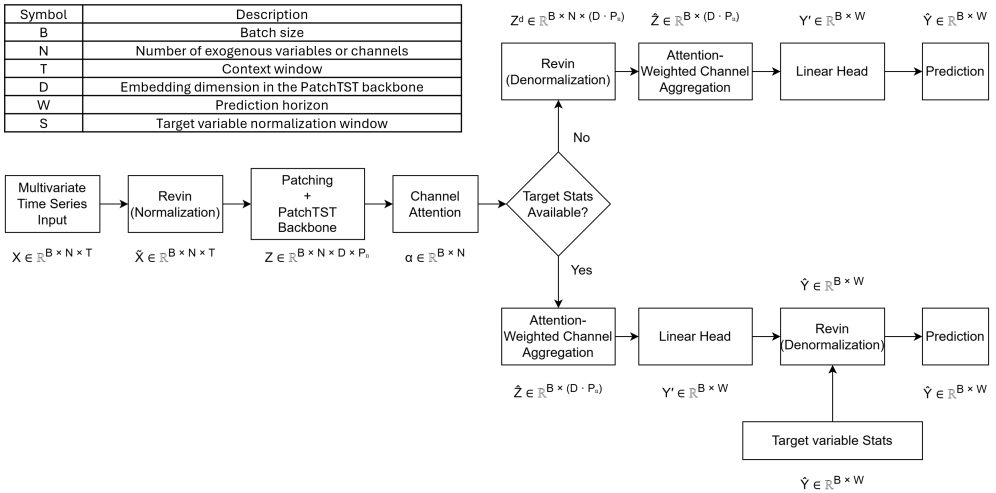


Figure 10: The figure illustrates the complete processing pipeline of the adapted PatchTST architecture. It includes the added *Channel Attention* block and outlines both operational modes: with and without access to target variable statistics. The flow shows how input tensors are processed, with tensor dimensions annotated for clarity (refer to the legend for symbol definitions).

To optimize model performance, hyperparameter tuning was conducted using Optuna (Akiba et al., 2019), a Bayesian optimization framework. A uniform hyperparameter search space (Table 3) was applied across all input scenarios, executing 2000 optimization trials for each configuration. Each trial trained the model for 15 epochs, using validation performance as the selection criterion. The objective was to minimize root mean square error (RMSE) while maximizing Mielke index, Pearson correlation, and R^2 score.

Optimal configurations were selected by ranking the trials based on metric-specific thresholds. For RMSE and MAPE, the 20th percentile (lowest values) was used, while for Mielke, Pearson, and R^2, the 80th percentile (highest values) served as the cutoff. A scoring system was applied, with configurations earning a point for each metric in which they exceeded the corresponding threshold. The top-ranked configurations—those scoring across all five metrics—were further sorted by the Mielke index due to its relevance in capturing coastal oscillatory behavior (Gomez-de la Pena et al., 2023). The ten best-performing models were then retrained for 100 epochs and benchmarked against CNN and CNN-LSTM architectures using the test datasets.

Table 3: Hyperparameter ranges explored during tuning. Each parameter was optimized using 2000 trials per input configuration.

Hyperparameter	**Exploration Range**
Normalization Window	2 to 300
Context Window	1 to 300
Patch Length	1 to 300
Stride	1 to 150

4.5 Evaluation Metrics

To compare model performance, a set of widely recognized evaluation metrics is employed. These metrics, described below, allow for both quantitative and graphical assessment of each model's accuracy and consistency across diverse scenarios.

4.5.1 RMSE (Root Mean Squared Error)

RMSE measures the average magnitude of the error and is especially useful in contexts where large deviations are undesirable. It allows direct comparison of prediction errors across different models when using the same data scale.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2} \tag{25}$$

where x_i are the observed values, $\hat{x}_i$ are the predicted values, and n is the number of observations.

4.5.2 MAPE (Mean Absolute Percentage Error)

MAPE expresses prediction error as a percentage, making it particularly suitable for applications where relative error is more interpretable, such as financial or energy demand forecasting.

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{x_i - \hat{x}_i}{x_i}\right| \tag{26}$$

where x_i, $\hat{x}_i$, and n are defined as above.

4.5.3 R^2 Score

The coefficient of determination (R^2) measures the proportion of variance in the observed data that is explained by the model. An R^2 value close to 1 indicates strong explanatory power.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{27}$$

where $\bar{x}$ is the mean of the observed values.

4.5.4 Pearson Correlation

The Pearson correlation coefficient quantifies the strength and direction of a linear relationship between two variables. It ranges from -1 (perfect negative correlation) to 1 (perfect positive correlation).

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(\hat{x}_i - \bar{\hat{x}})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(\hat{x}_i - \bar{\hat{x}})^2}} \tag{28}$$

where $\bar{x}$ and $\bar{\hat{x}}$ are the means of the observed and predicted values, respectively.

4.5.5 Mielke Index

The Mielke index (Duveiller et al., 2016) is a dimensionless, symmetric metric used to evaluate the agreement between observed and predicted time series. It is particularly suitable for assessing models that aim to capture variability and oscillatory behavior.

$$d = 1 - \frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{n}(|x_i - \bar{x}| + |\hat{x}_i - \bar{x}|)^2} \tag{29}$$

where $\bar{x}$ is the mean of the observed values.

4.5.6 Density Heat Scatter Plot

A density heat scatter plot is a graphical tool that compares observed and predicted values, enhancing standard scatter plots with color-coded density information. In this representation, observed values are placed on the horizontal axis and predictions on the vertical axis. A 1:1 reference line indicates perfect agreement.

Unlike basic scatter plots, this version includes a heatmap layer that highlights areas of high prediction density. Color intensity indicates the local concentration of points, enabling the detection of systematic biases and prediction clustering. Dense concentration along the diagonal suggests strong predictive accuracy and consistency.

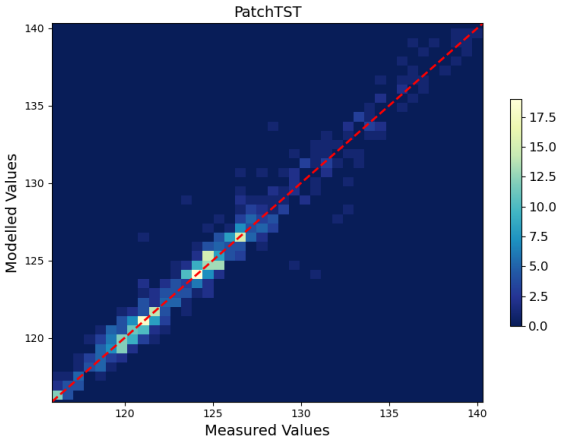


Figure 11: Example of a density heat scatter plot comparing observed and predicted values.

4.5.7 Taylor Diagram

The Taylor diagram (Taylor, 2001) is a compact graphical summary that simultaneously represents three aspects of model performance: the Pearson correlation, the centered root mean square difference (CRMSD), and the standard deviation. Each model is represented as a point in a polar coordinate system, with the radial distance corresponding to its standard deviation, the angle to its correlation with the reference, and the distance from the reference point representing the CRMSD.

This diagram enables the comparison of multiple models against the same reference data, offering both a quantitative and visual understanding of performance. Models that closely match the observed data appear near the reference point, indicating high agreement and low error.

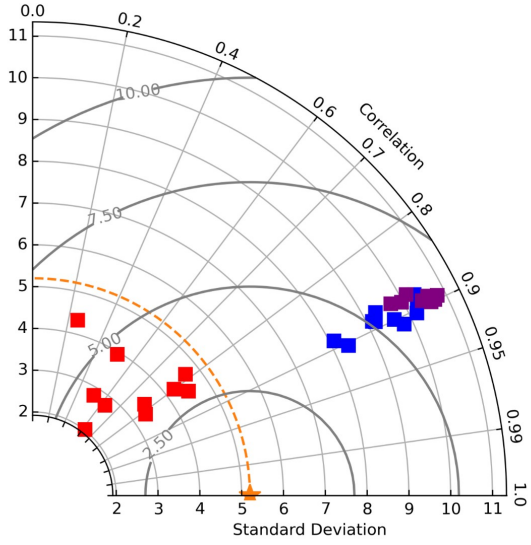


Figure 12: Example of a Taylor diagram summarizing model performance.

5 Results

This section begins with an in-depth analysis of the hyperparameter space explored for the PatchTST model. A total of 2000 trials were executed for each of the seven input configurations, leading to a comprehensive exploration comprising 14,000 configurations. The baseline CNN and CNN-LSTM models relied on the top-10 configurations previously proposed in Gomez-de la Pena et al. (2023), with no further hyperparameter optimization conducted in the present study.

To visualize this high-dimensional search space, two sets of parallel axis plots were generated. Figure 13 displays the results for the cases without the inclusion of the target variable (Inputs 1–3), while Figure 14 corresponds to the scenarios that include the target variable (Inputs 4–7). In these plots, each gray line represents a single trial, and each axis corresponds to a performance metric (RMSE, MAPE, R^2, Pearson correlation, and Mielke index). The pink vertical lines indicate the top 20% performance threshold for each metric. Configurations surpassing all five thresholds simultaneously are highlighted in red. This visualization enables efficient identification of high-performing regions in the hyperparameter space, as well as interactions between parameters.

Although no single configuration achieved universal optimality across all inputs, distinct patterns emerged for different scenarios. For example, shorter patch lengths (typically below 100 time steps) were consistently associated with high-performing configurations in Inputs 3 through 7. In contrast, Inputs 1 and 2 favored longer patches, while Input 2 showed a clear preference for patch lengths around 140 time steps, the best results in Input 1 were achieved with a broader range of values, from as low as 14 to over 270 time steps (Table 4).

The context window length also revealed input-specific behavior. For Inputs 1 and 2, the best trials were concentrated near the upper bounds of the search space, around 286 and 182 time steps respectively. Meanwhile, Inputs 3 to 7 exhibited a more heterogeneous distribution of optimal context lengths. Notably, Input 5 achieved top performance with short windows often below 50 time steps.

For inputs involving target normalization (Inputs 4 through 6), the normalization window length showed a marked tendency toward values around 100 time steps (Figure 14). The exception was Input 5, where the top-performing models relied on substantially shorter normalization windows (around 12 steps). Regarding stride, no globally dominant pattern was observed, however, Inputs 5 and 6 revealed clustering around specific stride values (25 and 80 time steps, respectively), hinting at localized stability in those configurations.

Table 4: Top-10 hyperparameter configurations identified for each input. Each subtable lists the best-performing combinations of parameters based on validation performance.
Abbreviations: C = Context window, P = Patch length, S = Stride, N = Normalization window.

Configurations	**Input 1**			**Input 2**			**Input 3**			**Input 4**				**Input 5**				**Input 6**				**Input 7**		
Rank	C	P	S	C	P	S	C	P	S	N	C	P	S	N	C	P	S	N	C	P	S	C	P	S
1	286	127	6	182	137	73	89	61	9	85	113	14	51	12	30	2	25	85	185	72	80	79	76	31
2	276	275	6	182	137	131	89	34	27	85	114	14	99	12	30	2	24	49	177	14	80	80	76	69
3	125	30	6	175	137	73	89	2	27	80	166	20	78	12	30	2	26	79	177	14	80	79	76	36
4	286	30	47	182	157	73	153	61	40	67	113	14	51	12	30	13	24	79	226	98	80	79	76	35
5	286	30	49	182	137	90	89	61	11	86	166	79	45	12	75	2	24	136	177	14	80	80	76	73
6	128	35	18	284	72	136	161	1	73	81	115	14	51	20	30	2	24	85	244	72	80	80	76	24
7	269	183	149	182	70	73	83	61	9	66	113	14	51	12	51	2	24	94	177	72	80	80	76	62
8	282	97	6	182	155	84	254	22	95	85	113	14	99	12	16	10	2	79	177	93	39	80	76	54
9	282	104	6	94	44	84	254	2	95	13	166	14	78	12	42	10	32	162	177	53	80	79	76	24
10	151	14	6	35	13	84	102	51	13	84	113	14	51	12	30	30	35	86	185	72	80	80	75	24

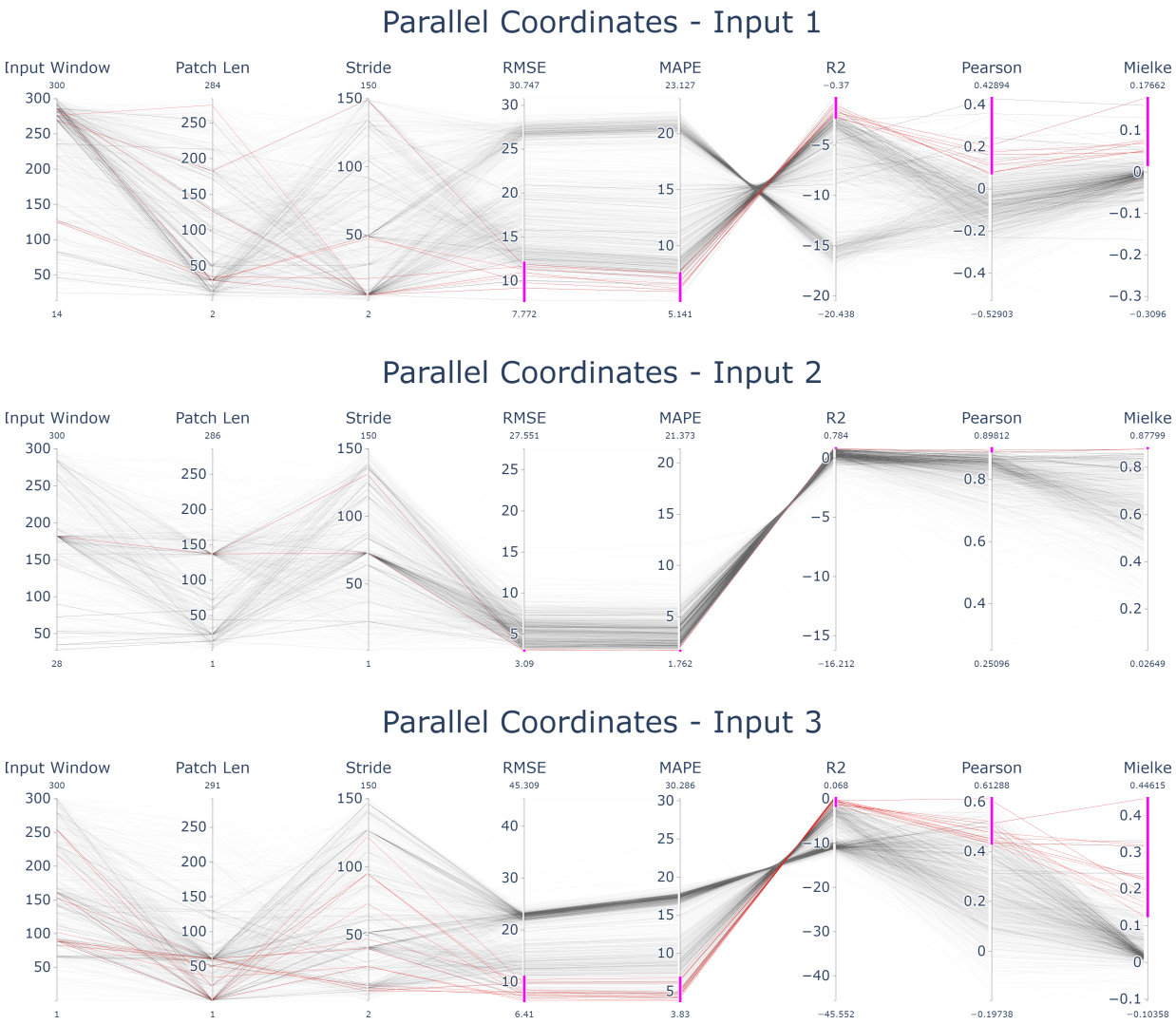


Figure 13: Parallel axis plot illustrating the optimal configurations selected during PatchTST training for the cases without the inclusion of the target variable (Input 1 to 3). Configurations achieving superior results (top 20%) are highlighted in red. Performance metrics are evaluated on the validation set.

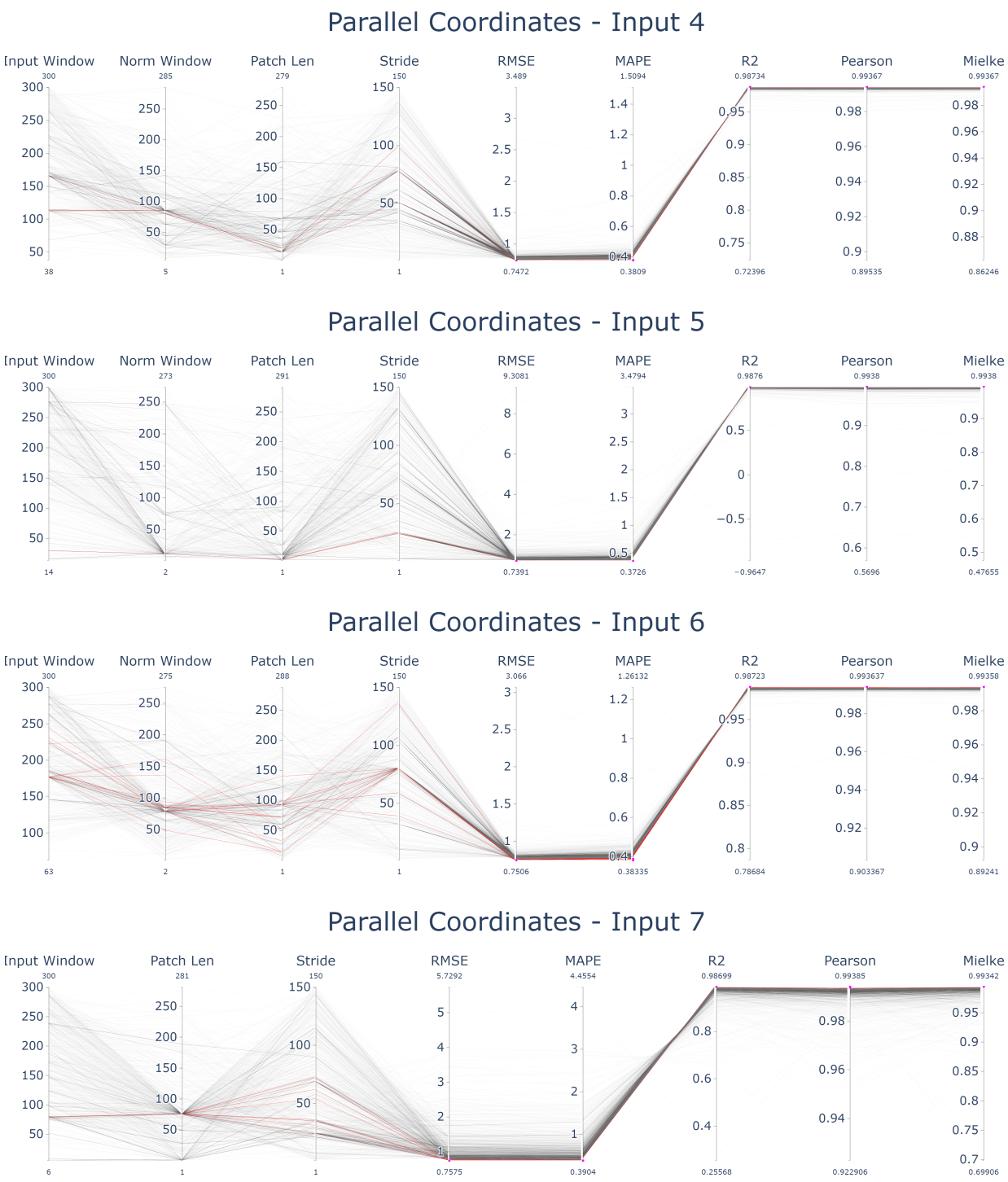


Figure 14: Parallel axis plot illustrating the optimal configurations selected during PatchTST training for the cases with the inclusion of the target variable (Input 4 to 7). Configurations achieving superior results (top 20%) are highlighted in red. Performance metrics are evaluated on the validation set.

To complement this analysis with a high-level perspective, Figure 15 presents Taylor diagrams comparing the predictive behavior of the top 10 models per architecture, input, and test scenario. Each input is represented by a distinct marker shape, and each architecture by color. Proximity to the orange star, which denotes perfect agreement with ground truth (Pearson correlation equals one, RMSE equals Zero and standard deviation equal to the real values), indicates superior predictive accuracy.

The diagram clearly illustrates that for Inputs 1 through 3, PatchTST models exhibited lower correlation and higher RMSE compared to their CNN and CNN-LSTM counterparts. However, beginning with Input 4, where the target coastline position is incorporated, PatchTST architecture consistently approaches the reference point, outperforming the baselines across all metrics. Where PatchTST exhibit near-perfect agreement with observed values, both in Test 1 and the generalization scenario of Test 2.

These findings not only validate the effectiveness of the hyperparameter search strategy but also highlight the architecture's capacity to leverage target-related input structures for significantly enhanced predictive performance.

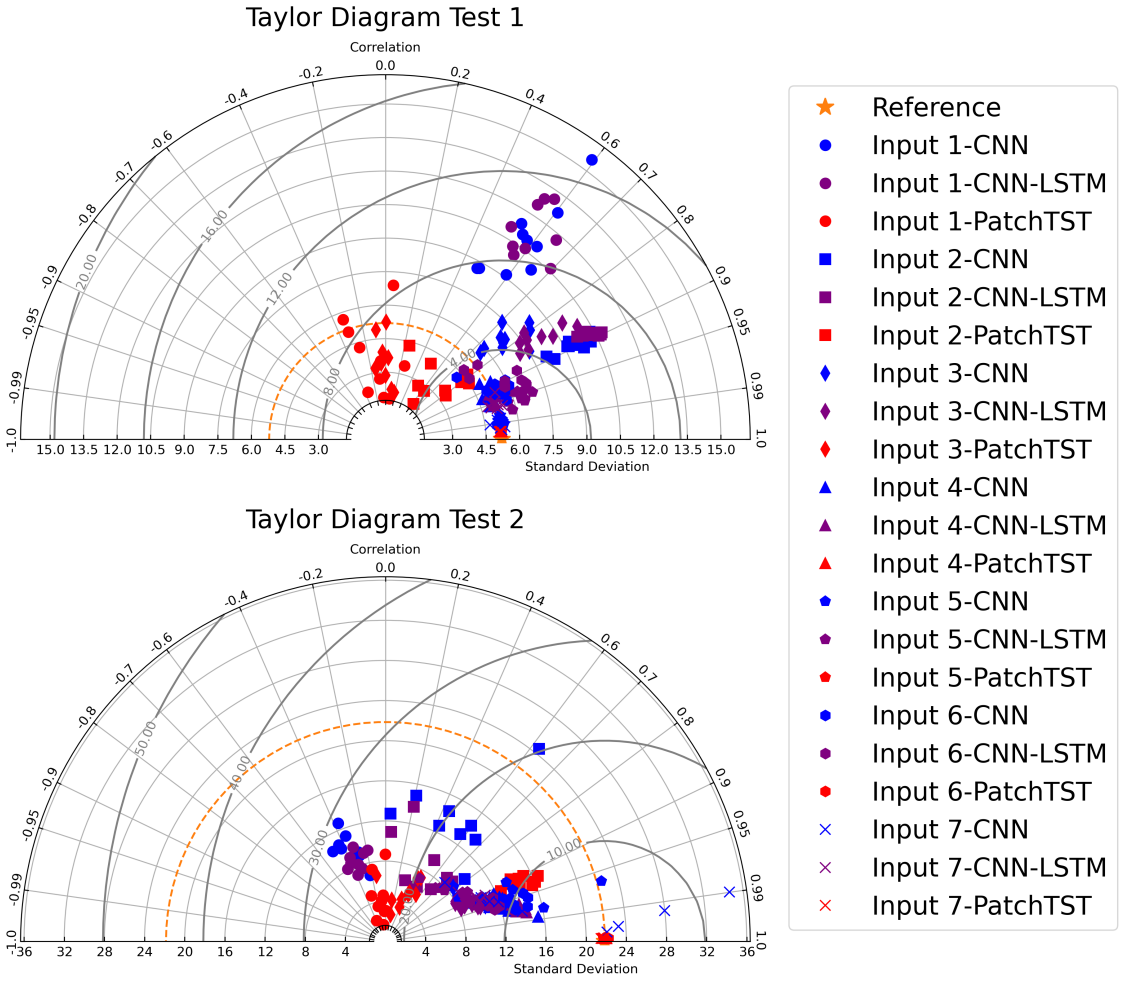


Figure 15: Taylor diagrams illustrating the predictive performance of the best 10 models for each architecture across Inputs 1–7, evaluated on Test 1 and Test 2. Models closer to the reference star indicate higher performance.

To enhance the interpretability of model behavior across varying input scenarios, ensemble prediction figures have been systematically grouped into three categories. Figures 16 present results for Inputs 1 to 3, where the target variable is excluded from the input features. Figures 17 correspond to Inputs 4 to 6, which incorporate the target coastline position alongside the original variables. Finally, Figure 18 displays results for Input 7, where the target variable is the sole input.

This organization facilitates a clearer assessment of how the inclusion of the target variable affects predictive performance. Each figure displays the ensemble mean prediction from the top 10 models per architecture, along with a shaded envelope representing the minimum and maximum values across the ensemble. Vertical dashed lines denote the boundaries between test, training, and validation phases. Specifically, the first split at 31/03/2021 marks the end of Test 2 and the start of the training phase, the second split at 27/11/2021 indicates the beginning of validation, which ends on 18/02/2022 with the start of Test 1. This temporal segmentation enables a detailed examination of model behavior across different phases and enhances the interpretability of generalization performance.

In Input 1, CNN and CNN-LSTM architectures captured the overall trend of the coastline but exhibited considerable variability among ensemble members. In contrast, PatchTST produced erratic and unstable outputs throughout all phases, particularly during Test 2, where its prediction envelope expanded and its ensemble mean diverged substantially from the observed shoreline. As shown in Table 5, PatchTST registered the highest RMSE in Test 1 (20.97 m) and correlation-based metrics close to zero or negative, reflecting poor temporal alignment. Although PatchTST yielded better numerical scores in Test 2, its predictions lacked clear structure or coherence, as visually confirmed in Figure 16.

In the case of Input 2, PatchTST improved substantially, reproducing the overall coastline trajectory with greater stability. The prediction envelope narrowed, and the ensemble mean became more consistent with the observed values. As reported in Table 5, PatchTST outperformed the baseline models across most metrics in both Test 1 and Test 2. In particular, the architecture reached a Pearson correlation of 0.91 and an RMSE of 12.89 m in Test 2. However, in Test 1, CNN still obtained the best results in shape-based metrics, with a Pearson coefficient of 0.89 and a Mielke index of 0.59.

For Input 3, CNN and CNN-LSTM models maintained reasonable agreement with the observed coastline and slightly improved over their Input 2 performance. PatchTST, however, returned to unstable behavior similar to that observed in Input 1. Its prediction envelope remained wide, and the mean forecast diverged from the ground truth, particularly during the test segment. Table 5 shows that PatchTST again recorded the lowest scores across nearly all metrics. By contrast, CNN achieved the best performance, especially in Test 2, where it reached an RMSE of 19.05 m and a Pearson coefficient of 0.91.

Overall, the results for Group 1 indicate that PatchTST underperforms when the target position is excluded from the input. While CNN-based models do not achieve high accuracy, they produce more coherent and consistent predictions, particularly in Test 1. In Test 2, all models face difficulties, though CNN retains relatively better generalization in the absence of target-specific information.

To evaluate the effect of incorporating the target variable into the input features, predictions for Inputs 4 to 6 are presented in Figure 17. These configurations include the coastline position as an additional input, allowing the models to leverage direct information on the target series. As in the previous figures, the average prediction of the top 10 models per architecture is shown along with the minimum–maximum envelope, and vertical dashed lines indicate the division between training, validation, and testing phases.

The inclusion of the target variable had a substantial positive effect on PatchTST's performance

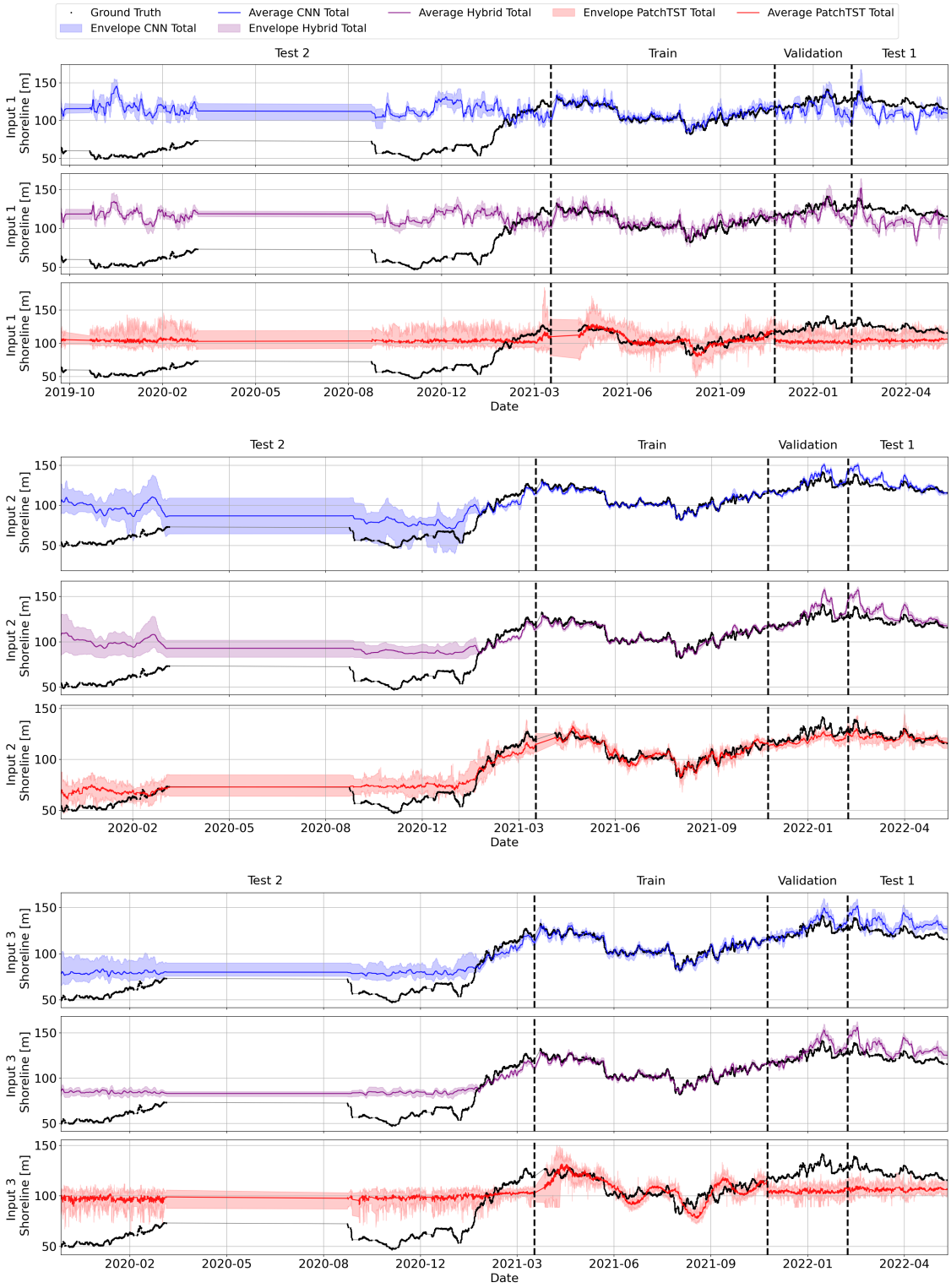


Figure 16: Ensemble predictions of the top 10 models for Inputs 1–3 (without including the target variable). The shaded region represents the range (minimum–maximum) of predictions across the models, while the solid line shows their average. Vertical dashed lines indicate the split between test 2, training, validation, and test 1.

across all three inputs. For Input 4, predictions by PatchTST closely aligned with the observed coastline throughout the entire time span. The ensemble envelope was extremely narrow, indicating high agreement among the top models. This is supported by Table 5, where PatchTST achieved a RMSE of 0.37 m in Test 1 and 0.32 m in Test 2, with R^2, Pearson, and Mielke values all reaching 0.99 or higher. In contrast, the baseline models showed greater variability and slightly lower accuracy, particularly during the test phase.

With Input 5, PatchTST maintained this high level of accuracy. The predictions were nearly indistinguishable from the observed data, and the ensemble spread remained minimal. RMSE values were again below 0.35 m in both tests, and the correlation metrics indicated near-perfect predictions. CNN and CNN-LSTM models improved compared to previous inputs but still lagged behind in precision and stability. Their envelopes showed more pronounced oscillations, especially in Test 2.

For Input 6, the trend continued. PatchTST demonstrated consistently high accuracy and low variability, while the baseline models struggled to maintain a close fit with the ground truth. As seen in the summary table, PatchTST outperformed both CNN and CNN-LSTM across all metrics in both test phases. In Test 2, it achieved an RMSE of 0.32 m and nearly perfect agreement across all shape-based metrics.

In summary, the results for Group 2 confirm the effectiveness of incorporating the target variable into the model input. PatchTST achieved not only the lowest error rates but also the highest consistency across ensemble members. The predictions were smooth, stable, and closely matched the observed coastline throughout all phases. Although CNN and CNN-LSTM also benefited from the inclusion of the target variable, they exhibited more variability and lower alignment, especially during generalization in Test 2.

The final input configuration, Input 7, considers only the target coastline position as the input feature. Figure 18 presents the ensemble predictions for this case. This minimalistic setup allows evaluating how well each architecture can extrapolate temporal dependencies based solely on the target's past evolution, without additional explanatory variables.

PatchTST once again delivered highly accurate and stable forecasts. The ensemble envelope was narrow and closely followed the ground truth across all segments, including the test phases. In both Test 1 and Test 2, the model achieved RMSE values below 0.55 m and shape metrics (R^2, Pearson, and Mielke) above 0.99 (Table 5). This level of agreement demonstrates the model's ability to exploit the internal structure of the target series, even in isolation from other input variables.

The CNN architecture also produced relatively good results for Input 7, especially in Test 1, with RMSE = 0.98 m and strong correlation metrics (Pearson = 0.99). However, performance deteriorated in Test 2, where the ensemble became wider and predictions deviated more noticeably from the true shoreline. The CNN-LSTM model, meanwhile, showed lower accuracy in both phases and maintained a broader envelope throughout, indicating persistent instability when relying exclusively on the target signal.

Overall, Input 7 confirms the superior capacity of PatchTST to model temporal dynamics even under reduced input scenarios. Unlike the CNN and CNN-LSTM models, which displayed degradation when no auxiliary variables were available, PatchTST preserved its predictive robustness, showing strong generalization and coherence across both testing domains.

Finally, Figure 19 provides a visual comparison of the best-performing model from each architecture, selected based on the Mielke index. These density scatter plots complement the previous ensemble analyses by focusing on the highest-quality individual predictions. From Input 4 onwards—where the target coastline position is included as part of the input—all architectures improved in accuracy. However, PatchTST stands out by producing predictions that are nearly

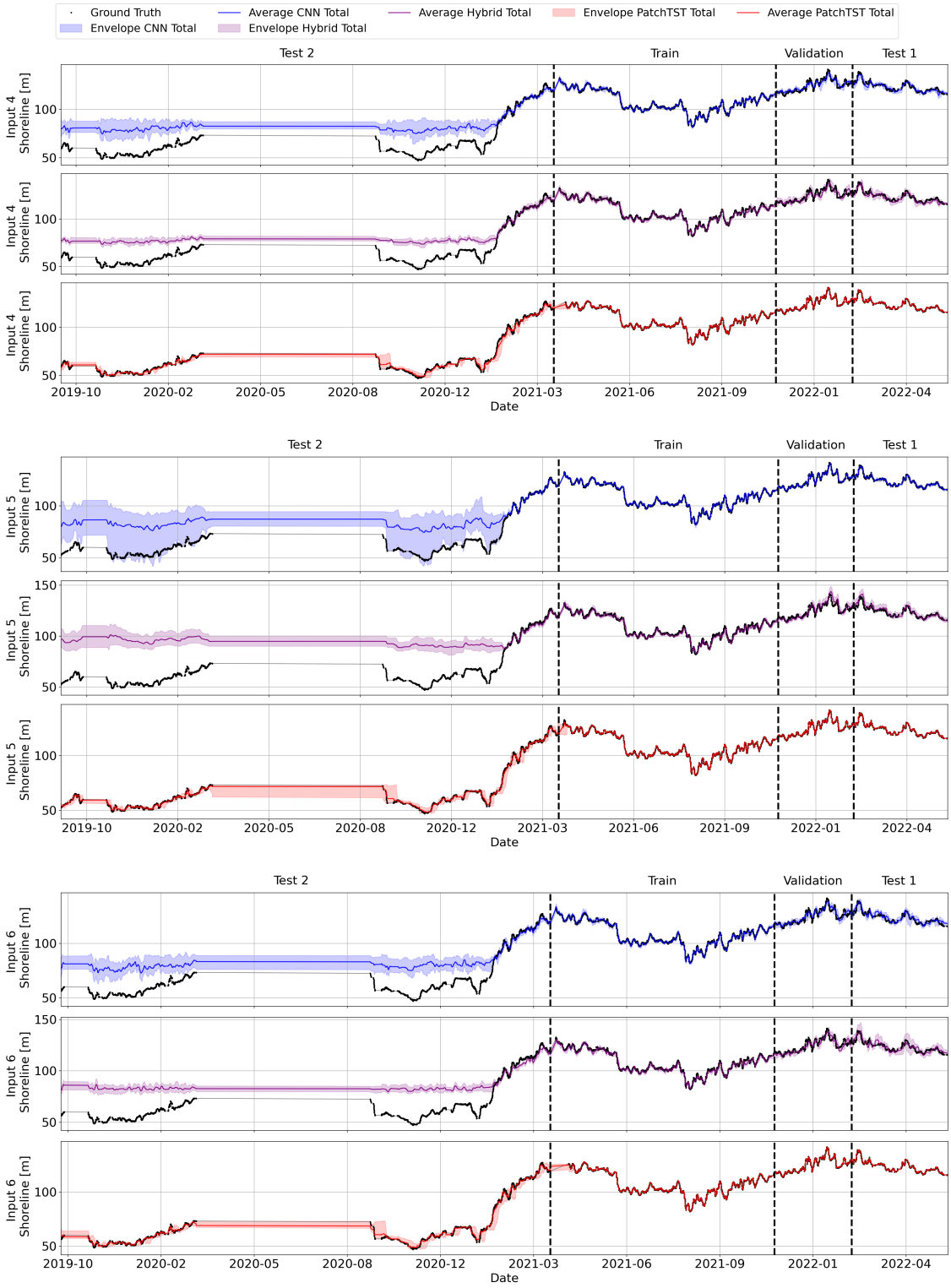


Figure 17: Ensemble predictions of the top 10 models for Inputs 4–6 (including the target variable). The shaded region represents the range (minimum–maximum) of predictions across the models, while the solid line shows their average. Vertical dashed lines indicate the split between test 2, training, validation, and test 1.

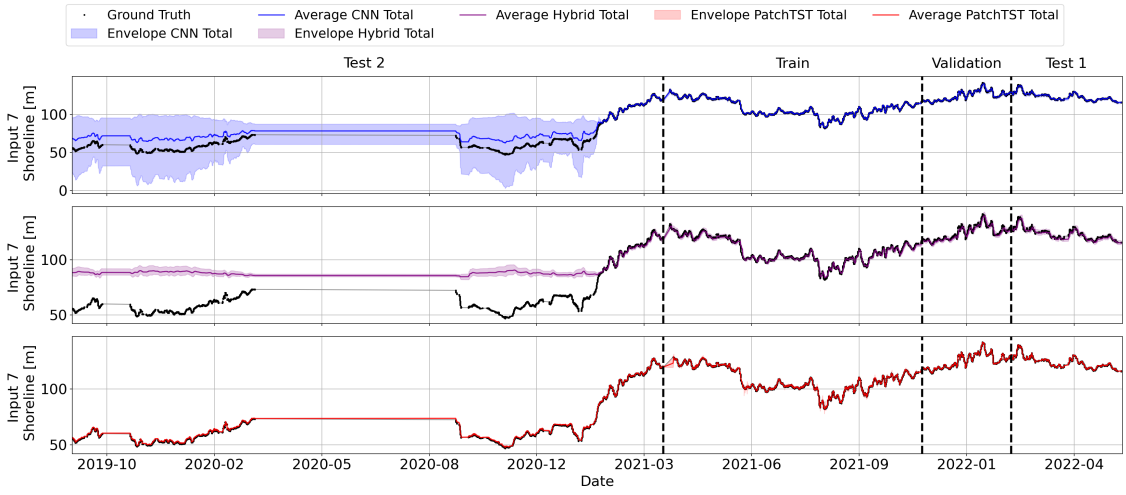


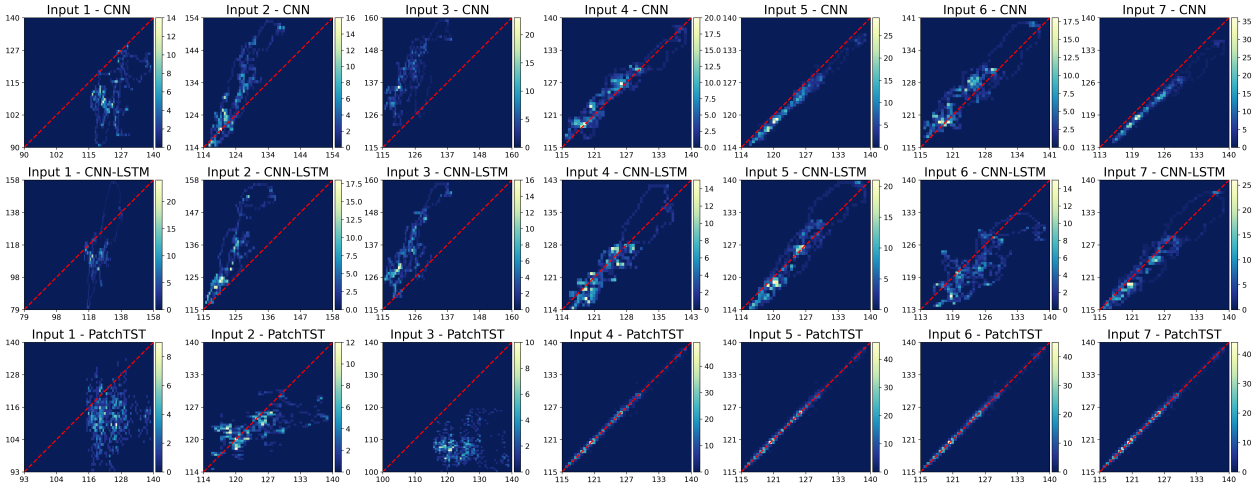
Figure 18: Ensemble predictions of the top 10 models for Input 7 (only the target variable). The shaded region illustrates the range of predictions (min–max), and the solid line represents the average. Vertical dashed lines indicate the split between test 2, training, validation, and test 1.

Table 5: Results from Test 1 and Test 2 for the different input sets tested. The table presents the average metric values from both tests, and highlights in bold the best-performing model for each input.

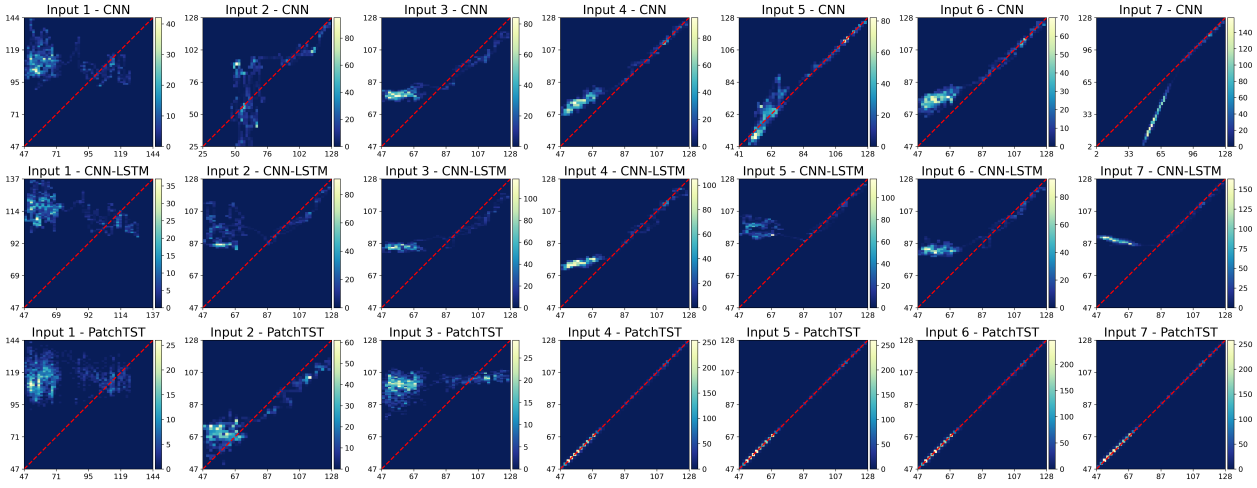
Input	**Model**	**Test 1**					**Test 2**				
		RMSE [m]	MAPE [%]	R^2 [-]	Pearson [-]	Mielke [-]	RMSE [m]	MAPE [%]	R^2 [-]	Pearson [-]	Mielke [-]
Input 1	CNN	**15.4348**	**10.7038**	**-7.8734**	**0.5680**	**0.2152**	52.4234	80.1377	-4.8150	-0.3614	-0.0598
	CNN-LSTM	15.7791	11.0153	-8.3204	0.5778	0.2223	52.2339	80.1797	-4.7546	-0.3497	-0.0502
	PatchTST	20.9717	15.7937	-16.9666	-0.1074	-0.0212	**40.2648**	**61.5340**	**-2.3361**	**-0.1374**	**-0.0156**
Input 2	CNN	7.7603	4.7458	-1.2602	**0.8945**	**0.5948**	28.8181	41.3910	-0.8288	0.5091	0.2864
	CNN-LSTM	10.6701	6.8630	-3.2753	0.8922	0.4634	32.3704	48.6318	-1.2596	0.5092	0.1704
	PatchTST	**4.9990**	**3.1649**	**-0.0181**	0.6530	0.5009	**12.8851**	**17.8954**	**0.6429**	**0.9144**	**0.7769**
Input 3	CNN	**11.4726**	**8.6175**	**-4.1435**	0.7687	**0.3193**	**19.0522**	**28.8489**	**0.1932**	**0.9164**	**0.5563**
	CNN-LSTM	12.0879	8.8767	-4.4836	**0.8337**	0.3352	23.2155	35.5666	-0.1419	0.9091	0.3955
	PatchTST	18.5144	13.8588	-11.8377	-0.0073	-0.0034	34.9045	53.2980	-1.5270	0.3314	0.0624
Input 4	CNN	2.2642	1.4280	0.7988	0.9161	0.8990	18.9601	28.6945	0.2213	0.9619	0.5980
	CNN-LSTM	2.7788	1.8471	0.7083	0.9221	0.8719	16.2782	24.5564	0.4388	0.9713	0.6874
	PatchTST	**0.3659**	**0.2027**	**0.9950**	**0.9975**	**0.9975**	**0.3153**	**0.2980**	**0.9998**	**0.9999**	**0.9999**
Input 5	CNN	1.0047	0.6544	0.9594	0.9878	0.9799	21.1102	31.3813	-0.0438	0.9142	0.5360
	CNN-LSTM	2.6844	1.7242	0.6944	0.9533	0.8937	31.4753	47.5777	-1.1129	0.7550	0.2240
	PatchTST	**0.3478**	**0.1903**	**0.9955**	**0.9978**	**0.9978**	**0.2919**	**0.2762**	**0.9998**	**0.9999**	**0.9999**
Input 6	CNN	2.5986	1.6895	0.7388	0.9005	0.8746	18.3556	27.5962	0.2694	0.9514	0.6162
	CNN-LSTM	3.7822	2.4710	0.4398	0.8536	0.7697	21.4840	32.5865	0.0210	0.9363	0.4925
	PatchTST	**0.3775**	**0.2096**	**0.9947**	**0.9974**	**0.9973**	**0.3242**	**0.3122**	**0.9998**	**0.9999**	**0.9999**
Input 7	CNN	0.9773	0.6283	0.9554	0.9906	0.9778	18.6546	27.9150	0.0644	0.9375	0.6209
	CNN-LSTM	2.0438	1.3173	0.8373	0.9522	0.9219	26.2456	39.5671	-0.4555	0.8875	0.3533
	PatchTST	**0.3690**	**0.2076**	**0.9950**	**0.9977**	**0.9975**	**0.5449**	**0.7504**	**0.9993**	**0.9999**	**0.9997**

indistinguishable from the ground truth, especially in Test 2, where generalization to unseen data poses the greatest challenge.

For Inputs 1 to 3, where the target variable is absent, all models struggled to replicate the true coastline trajectory. This effect was particularly noticeable in Test 2, where predictions became noisier and more dispersed. Still, for Input 2, PatchTST showed better alignment than its counterparts, suggesting a partial ability to infer the underlying dynamics even without the target signal. In contrast, CNN and CNN-LSTM models showed limited generalization, with performance dropping substantially when transitioning from validation to test phases. From Input 4 onwards, the advantage of PatchTST became more pronounced. Despite using the best hyperparameter configurations for each architecture, the CNN-based models consistently underperformed relative to PatchTST, reaffirming its robustness across input conditions.



(a) Test 1: Best models comparison



(b) Test 2: Best models comparison

Figure 19: Density heat scatter plot comparison of coastline predictions for the best-performing models according to the Mielke metric.

6 Discussion

This study aimed to evaluate whether increasing architectural flexibility through the use of Transformer-based models, particularly PatchTST, can improve predictive skill in shoreline forecasting tasks when compared to more traditional deep learning models such as CNN and CNN-LSTM (see Table 5). A key aspect of this evaluation involved examining the role of input configuration, especially the presence or absence of the target shoreline position (POS1). The current trend in shoreline modeling has focused on mimicking physically-based approaches by introducing exogenous variables that act as external forcings. While this strategy may be intuitive from a physical perspective, it might not fully leverage the capabilities of deep learning architectures. For this reason, the study also investigated how different combinations of input variables affect model performance and interpretability, including cases with and without direct shoreline observations.

The hyperparameter analysis showed that PatchTST generally favored intermediate to long context windows, particularly in scenarios where the target shoreline position (POS1) was excluded (Figure 13). This behavior suggests that, in the absence of a direct link between inputs and outputs, the model attempts to compensate by extending its temporal range to extract broader patterns from the data.

In contrast, Input 5, which included both POS1 and adjacent shoreline positions, showed a consistent preference for shorter context windows, normalization periods, and patch lengths. This indicates that when the input includes variables closely related in space and time to the target, the model relies on local information and requires less temporal depth to make accurate predictions. However, such configurations also risk producing trivial outputs by learning to repeat recent values rather than capturing dynamic processes. This highlights the importance of evaluating hyperparameter selection not only in terms of performance, but also in terms of the behavioral biases the model may develop.

Across most input types, optimal patch lengths remained below 100 time steps. Shorter patches likely enhanced the model's ability to detect rapid fluctuations and short-term variability, which are more relevant in high-resolution hourly data. In this context, longer patches may obscure high-frequency signals and introduce unnecessary smoothing. For the configurations that included POS1, particularly Inputs 4 and 6, the most effective models converged around a normalization window of 100 time steps (Figure 14). This suggests that the selected normalization window length provides the model with a statistical reference that preserves enough variance, allowing it to capture abrupt changes in the shoreline position without underestimating their magnitude during denormalization.

When the target variable POS1 was excluded (Inputs 1 to 3), PatchTST showed significant sensitivity to the nature and scale of the input data. For Input 1, which included only hydrodynamic variables, the model produced noisy outputs lacking a coherent trend (Figure 16). This behavior is likely due to a mismatch between the temporal and magnitude dynamics of the inputs and the shoreline response. As the model processes each input channel independently, it struggled to extract meaningful relationships from variables that do not exhibit strong direct correlations with the target. In this context, the attention mechanism failed to identify a dominant signal, resulting in erratic predictions.

A notable improvement was observed with Input 2, where adjacent shoreline positions were included but POS1 was still excluded. The spatial coherence and physical relevance among the input channels allowed the model to infer general shoreline behavior, even in the absence of the direct target. This configuration provided enough structured information for the model to identify local dependencies and spatial trends. However, performance dropped again in Input 3 when hydrodynamic variables were reintroduced. The combined input—while richer in quantity—was

more heterogeneous in nature and scale, which likely confused the attention mechanism. This is reflected in the Taylor diagram (Figure 15), where the Input 3 predictions are clearly farther from the reference point compared to Input 2 in both test sets. These results reinforce the idea that simply increasing the number of input variables does not guarantee better performance. Instead, consistency in scale and physical meaning across channels plays a more critical role in enabling the model to learn effective temporal dependencies.

From Input 4 onward, where POS1 was included, all architectures showed a clear improvement in performance (Figure 17). PatchTST benefited the most, largely due to its approach to normalization during both training and inference. Although POS1 is not directly fed into the model as an input feature, its statistics—specifically the most recent value and the standard deviation calculated over the normalization window—are used to denormalize the model's output. This strategy offers two key advantages: it centers the prediction around the last known shoreline position, and it provides a dynamic scaling factor that reflects the recent variability of the target. As a result, the model is not required to infer the absolute position or scale of the shoreline from unrelated inputs, allowing it to focus entirely on capturing the internal structure and relationships among the available variables. This leads to more stable and accurate predictions, particularly in scenarios with rapid shoreline fluctuations, and enhances the model's ability to detect interactions between shoreline evolution and external forcings.

CNN and CNN-LSTM also exhibited improved results with the inclusion of POS1, although to a lesser extent. These models extract features jointly from all input channels, which may dilute the specific influence of the target variable when present. This effect is especially noticeable in Input 7 (Figure 18), where all architectures achieved strong results, but PatchTST continued to outperform the others. In this configuration, shape-based metrics such as R^2, Pearson correlation, and Mielke index consistently exceeded 0.90, with values reaching up to 0.99 in several cases (Table 5).

The comparison across architectures revealed distinct behavioral patterns depending on the input configuration. In the absence of the target variable (Inputs 1 to 3), CNN and CNN-LSTM were still able to capture the general shoreline trend, though with limited capacity to reproduce high-frequency fluctuations. PatchTST, by contrast, delivered inconsistent results under these conditions, particularly in Inputs 1 and 3, where the input data lacked either spatial coherence or direct relevance to the shoreline position. The only exception was Input 2, where adjacent shoreline positions were provided. In this case, PatchTST showed a clear improvement, suggesting that its attention mechanism was able to leverage spatial relationships to infer the underlying shoreline dynamics.

Once POS1 was included (Inputs 4 to 7), PatchTST consistently outperformed both CNN and CNN-LSTM across all configurations. The model's architecture, which emphasizes individual channel processing and attention-based temporal learning, appears better suited to capture complex interactions when the input contains high-quality, directly relevant signals. While CNN and CNN-LSTM also improved with POS1, their performance gains were less pronounced. These models rely on shared convolutional feature extraction, which may limit their ability to isolate and exploit the contribution of the target variable. The performance gap across architectures became especially evident in Input 4 and Input 7, where PatchTST achieved the highest scores across all metrics.

The evaluation on Test 2, which featured unseen morphological configurations, provided insight into each model's ability to generalize beyond the training distribution. PatchTST consistently delivered robust and stable predictions across the most relevant configurations, particularly in Inputs 2 and 4 through 7 (Figures 16, 17, and 18). These results suggest that the model's attention-based mechanism is effective at capturing long-range dependencies and structural patterns that remain valid even under shifting environmental conditions.

CNN and CNN-LSTM, on the other hand, showed a notable drop in accuracy and produced broader and more biased prediction envelopes. In some cases, CNN was able to approximate the general shoreline trend, but its mean predictions often deviated from the ground truth. CNN-LSTM performed the worst under Test 2, likely due to overfitting to the training distribution. This limitation is clearly illustrated in the heatmaps (Figure 19), where CNN-LSTM's predicted values are poorly aligned with the observed shoreline positions in Test 2.

To quantify these differences, percentage improvements were calculated using the results from Test 2, which represented the most challenging evaluation scenario due to its previously unseen morphological conditions. Metrics from the summary table (Table 5) were used to compare PatchTST against both CNN and CNN-LSTM across key configurations.

In Input 2, which included spatially coherent shoreline positions but excluded POS1, PatchTST achieved RMSE reductions of up to 60%, along with gains exceeding 80% in R^2 and Pearson correlation. The Mielke index also improved by more than 70%, confirming the model's ability to infer shoreline behavior from spatial patterns alone.

For Input 4, where both POS1 and hydrodynamic variables were present, PatchTST reached near-perfect performance, reducing RMSE by 98% and reaching close to 100% in R^2, Pearson correlation, and Mielke index relative to both baseline models. Even in Input 7, which relied exclusively on POS1, the model maintained a 98% RMSE gain and perfect scores in correlation-based metrics, despite the limited physical context.

These results highlight PatchTST's ability to generalize under adverse conditions, especially when inputs provide structured or target-centered information. The improvements observed in Test 2 underscore the robustness of its attention-based architecture and its suitability for real-world shoreline forecasting tasks.

Although Input 7 achieved strong predictive results, it relied exclusively on POS1 and did not incorporate external variables. Its main advantage lies in its low computational cost, as it requires fewer input channels and a simpler data structure, making it suitable for fast or lightweight applications. However, the absence of contextual information limits the interpretability of its outputs. While the predictions closely match the observed shoreline positions, they provide no insight into the external drivers influencing shoreline dynamics.

The highest overall performance was obtained with Input 5, which included POS1 along with adjacent shoreline positions. This configuration allowed the model to exploit both spatial continuity and historical context, resulting in the most accurate predictions across all metrics. Nevertheless, like Input 7, it offered limited physical interpretability, as it did not include any hydrodynamic forcings.

In contrast, Input 4 presented a more balanced alternative by combining POS1 with relevant hydrodynamic variables. This setup preserved high predictive accuracy while embedding physical meaning into the model inputs, enabling interpretations that link shoreline behavior to environmental conditions. CNN and CNN-LSTM remained more sensitive to input structure, especially in simplified configurations, showing broader prediction envelopes and less stable performance.

These results indicate that while minimal inputs can be appealing due to their simplicity and efficiency, they may not be suitable for applications requiring interpretability or process understanding. Hybrid input configurations like Input 4 represent a practical trade-off, offering strong and generalizable predictions while maintaining alignment with physical processes. The findings underscore the importance of both architectural choice and input design in the development of shoreline forecasting models, particularly when aiming for operational or decision-support applications.

7 Conclusions

This study confirmed that Transformer-based architectures, particularly PatchTST, deliver superior predictive performance compared to classical deep learning models such as CNN and CNN-LSTM in shoreline forecasting tasks. The attention-based design of PatchTST enabled it to consistently outperform baseline models, especially when supplied with input configurations that included either direct or indirect information about the target shoreline position. Unlike CNN-based models, which showed moderate success across all inputs, PatchTST demonstrated a significant performance advantage when the input data was appropriately structured, consistently achieving lower error metrics and higher correlation scores under both tests conditions. These findings provide a clear affirmative response to the central research question, highlighting the impact of architectural versatility in enhancing predictive accuracy for dynamic coastal environments.

The architectural features of PatchTST played a central role in achieving its superior performance. Its channel-wise attention mechanism allowed the model to process each input variable independently over time, facilitating the detection of temporal dependencies that would be otherwise diluted in architectures based on shared convolutional encoding. In addition, PatchTST's use of the target shoreline position for output denormalization—rather than as a direct input—proved particularly effective in configurations such as Inputs 4, 5, and 6. This approach forced the model to rely exclusively on external variables to generate predictions, centering the output using only the most recent shoreline observation and its variability. Such a mechanism allowed PatchTST to isolate the forecasting task from direct shoreline history and instead focus on learning relationships between the input drivers.

The configuration of input variables emerged as a key factor influencing model performance. When the target shoreline position (POS1) was included directly in the input, as in Input 7, the model achieved strong predictive accuracy, albeit with limited physical interpretability. Conversely, in Inputs 4, 5, and 6, POS1 was used exclusively for denormalization purposes, allowing the model to center its predictions without relying on the shoreline history as an input. This distinction proved essential for evaluating the model's ability to learn from external variables alone. Input 4, in particular, demonstrated that by combining POS1 (used for denormalization) with relevant hydrodynamic drivers, it was possible to achieve both high accuracy and improved interpretability. These results highlight the importance of not only selecting informative variables, but also structuring them in a way that aligns with the intended forecasting objectives.

PatchTST also demonstrated strong generalization capabilities, maintaining high predictive performance under unseen morphological conditions, as evaluated in Test 2. Its attention-based architecture enabled the model to capture long-range dependencies and adjust effectively to shifts in the input distribution. Unlike CNN and CNN-LSTM, which frequently exhibited systematic overestimations and reduced stability in this scenario, PatchTST consistently produced accurate and coherent predictions. This robustness highlights the suitability of attention-driven models for real-world shoreline forecasting applications, where environmental variability and limited data consistency are common challenges.

The results also revealed a fundamental trade-off between predictive accuracy and physical interpretability, referred to in this work as the *Coastal Parrot Effect*. Similar to how a parrot can replicate speech without understanding its meaning, certain neural network configurations—particularly Input 7, which used only the target shoreline position as input—achieved high predictive performance while ignoring the underlying physical drivers of coastal change. In contrast, hybrid configurations like Input 4 compelled the model to learn interactions between shoreline evolution and environmental forcings, offering a more interpretable framework. Additionally, the integration of a channel

attention block within the PatchTST architecture provided insights into the relative importance of each input variable during prediction. Although the interpretability of attention weights remains limited in physical terms and warrants further methodological development, this feature represents a promising step toward more transparent and explainable deep learning models in coastal applications.

In conclusion, the effective application of Transformer-based models such as PatchTST to shoreline forecasting depends not only on architectural design, but also on the thoughtful selection and structuring of input variables. This study showed that it is possible to achieve high predictive accuracy while preserving a degree of physical interpretability, particularly through hybrid input configurations. Balancing these two aspects is essential for advancing the use of deep learning in coastal science. As attention-based models continue to evolve, their potential for both accurate forecasting and process understanding positions them as valuable tools for real-world coastal monitoring and decision-making. Achieving greater transparency in how these models operate will be key to building trust and unlocking their full potential in the face of increasingly dynamic coastal challenges.

8 Future Work

This study has demonstrated the strong predictive capability of the PatchTST model for shoreline forecasting. However, the current implementation is limited to single-step predictions. Future work should extend this approach to multi-step forecasting, allowing the prediction of shoreline evolution over longer time horizons. This would better leverage the sequential learning strengths of Transformer architectures and provide more comprehensive tools for coastal planning and early warning systems.

Another promising direction lies in improving model interpretability. While this study introduced a Channel Attention mechanism, further work could explore advanced techniques for interpreting Transformer-based models. These may include visualizing attention weights over time and inputs, or integrating model-agnostic interpretability tools such as SHAP or LIME. Such methods could provide insights into the model's decision-making process and clarify the role of individual input variables in shaping the forecasts.

Data imputation is another valuable avenue. Transformer models could be trained to reconstruct missing shoreline data during night-time periods—where camera-based monitoring is unavailable—or during extended sensor outages, as occurred in the Test 2 dataset. This could significantly enhance the temporal continuity of coastal datasets. Similarly, imputation of hydrodynamic variables could ensure the availability of complete input sequences, enabling more robust model operation in real-time applications.

Finally, future work could focus on extending the spatial scope of predictions. While this study concentrated on a single transect, a natural extension would involve forecasting shoreline evolution at multiple locations simultaneously. Developing spatially resolved models capable of generating 2D shoreline forecasts would mark a major advancement, bridging the gap between point-based time series prediction and fully distributed morphodynamic modeling. This could support more comprehensive coastal risk assessments and adaptive management strategies in the face of accelerating environmental change.

References

Adusumilli, S., Cirrito, N., Engeman, L., et al. (2024a). Predicting shoreline changes along the california coast using deep learning applied to satellite observations. *Journal of Geophysical Research: Machine Learning and Computation*, *1*, e2024JH000172. https://doi.org/10.1029/2024JH000172

Adusumilli, S., Cirrito, N., Engeman, L., Fiedler, J. W., Guza, R. T., Lange, A. M. Z., Merrifield, M. A., O'Reilly, W., & Young, A. P. (2024b). Predicting shoreline changes along the california coast using deep learning applied to satellite observations. *Journal of Geophysical Research: Machine Learning and Computation*, *1*, e2024JH000172. https://doi.org/10.1029/2024JH000172

Ahmed, N., Howlader, N., Hoque, M. A. A., & Pradhan, B. (2021). Coastal erosion vulnerability assessment along the eastern coast of bangladesh using geospatial techniques. *Ocean and Coastal Management*, *199*, 105408. https://doi.org/10.1016/j.ocecoaman.2020.105408

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, 2623–2631. https://doi.org/10.1145/3292500.3330701

Athanasiou, P., van Dongeren, A., Giardino, A., Vousdoukas, M., Ranasinghe, R., Koutroulis, A., Reimann, L., & Hinkel, J. (2020). Uncertainties in projections of sandy beach erosion due to sea level rise: An analysis at the european scale. *Scientific Reports*, *10*, 11895. https://doi.org/10.1038/s41598-020-68576-0

Barnard, P. L., O'Neill, A. C., Vitousek, S., van Ormondt, M., Eshleman, J., Erikson, L. H., Jones, B., Limber, P., Hapke, C., Sherwood, C. R., et al. (2019). Coastal vulnerability across the pacific dominated by el niño/southern oscillation. *Nature Geoscience*, *12*, 500–506. https://doi.org/10.1038/s41561-019-0352-9

Barnard, P. L., van Ormondt, M., Erikson, L. H., Eshleman, J., Hapke, C., Hart, J. A., Ruggiero, P., Warrick, J. A., Foxgrover, A. C., & Vitousek, S. (2020). Future coastal flooding and erosion risk due to sea-level rise along the california coast. *Climatic Change*, *163*, 1535–1561. https://doi.org/10.1007/s10584-020-02988-z

Behera, S. K. (2024). Understanding the impact of climate change on extreme events. *Frontiers in Science*, *2*, 1433766. https://doi.org/10.3389/fsci.2024.1433766

Calcraft, K., Splinter, K. D., Simmons, J. A., & Marshall, L. A. (2025). Do lstm memory states reflect the relationships in reduced-complexity sandy shoreline models. *Environmental Modelling & Software*, *183*, 106236.

Calkoen, F., Luijendijk, A., Rodriguez Rivero, C., Kras, E., & Baart, F. (2021). Traditional vs. machine-learning methods for forecasting sandy shoreline evolution using historic satellite-derived shorelines. *Remote Sensing*, *13*(5), 934. https://doi.org/10.3390/rs13050934

Castelle, B., Bujan, S., Marieu, V., & Ferreira, S. (2020). 16 years of topographic surveys of rip-channelled high-energy meso-macrotidal sandy beach. *Scientific Data*, *7*, 410.

Chang, C.-T. A., & Lai, J.-C. (2014). A hybrid model integrating wavelet and artificial neural networks for forecasting extreme hydrological events. *Journal of Hydrology*, *515*, 504–514.

Chen, C., Liang, J., Xie, F., Hu, Z., Sun, W., Yang, G., Yu, J., Chen, L., Wang, L., Chen, H., He, X., & Zhang, Z. (2022). Temporal and spatial variation of coastline using remote sensing images for zhoushan archipelago, china. *International Journal of Applied Earth Observation and Geoinformation*, *107*, 102711. https://doi.org/10.1016/j.jag.2022.102711

Cienfuegos, R., Lucero, F., Catalán, P., Martínez, C., Díaz, P., & Caravaca, L. (2023). Sistema de monitoreo y anticipación de la resiliencia costera – simona costa. *XXVI Congreso Chileno de Ingeniería Hidráulica*.

D'Anna, M., Idier, D., Castelle, B., Rohmer, J., Cagigal, L., & Mendez, F. J. (2022). Effects of stochastic wave forcing on probabilistic equilibrium shoreline response across the 21st century including sea-level rise. *Continental Shelf Research*, *241*, 104666. https://doi.org/10.1016/j.csr.2022.104666

Davidson, M. A., Splinter, K. D., & Turner, I. L. (2013). A simple equilibrium model for predicting shoreline change. *Coastal Engineering*, *73*, 191–202.

Duveiller, G., Fasbender, D., & Meroni, M. (2016). Revisiting the concept of a symmetric index of agreement for continuous datasets. *Scientific reports*, *6*, 1–14. https://doi.org/10.1038/srep19401

Gardner, M. W., & Dorling, S. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, *32*(14-15), 2627–2636.

Gillis, J. R. (2012). *The human shore: Seacoasts in history*. University of Chicago Press. https://books.google.com/books?hl=enlr=id=hi-ewiLTfrUC

Goldstein, E. B., Coco, G., & Plant, N. G. (2019). A review of machine learning applications to coastal sediment transport and morphodynamics. *Earth-Science Reviews*, *194*, 97–108. https://doi.org/10.1016/j.earscirev.2019.04.022

Gomez-de la Pena, E., Coco, G., Whittaker, C., & Montano, J. (2023). On the use of convolutional deep learning to predict shoreline change. *EGUsphere*, *2023*, 1–24.

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). Google earth engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*. https://doi.org/10.1016/j.rse.2017.06.031

Goyo, M. A., Ñanculef, R., & Valle, C. (2024). Inversetime: A self-supervised technique for semi-supervised classification of time series. *IEEE Access*, *12*, 165081–165093. https://doi.org/10.1109/ACCESS.2024.3486669

Hinton, G. E. (2007). Learning multiple layers of representation. *Trends in cognitive sciences*, *11*(10), 428–434.

Hochreiter, S., & Schmidhuber, J. (1997a). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hochreiter, S., & Schmidhuber, J. (1997b). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 448–456. https://proceedings.mlr.press/v37/ioffe15.html

Ismail Fawaz, H. y. o. (2019). Deep learning for time series classification: A review. *Data Mining and Knowledge Discovery*, *33*, 917–963. https://doi.org/10.1007/s10618-019-00619-1

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*.

Larson, M., & Kraus, N. C. (1995). Prediction of cross-shore sediment transport at different spatial and temporal scales. *Marine Geology*, *126*(1-4), 111–127.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*, 436–444. https://doi.org/10.1038/nature14539

LeCun, Y. A., Bottou, L., Orr, G. B., & Müller, K.-R. (2012). Efficient backprop. In *Neural networks: Tricks of the trade* (pp. 9–48). Springer.

Luarte, H. R. H. (2017). *Calibración y validación del modelo wavewatch iii v.4.18, para su aplicación en la generación de un reanálisis de oleaje en las costas de chile* [Master's thesis, Universidad de Valparaíso]. https://repositoriobibliotecas.uv.cl/handle/uvscl/10417

Ludka, B. C., Guza, R. T., O'Reilly, W. C., Merrifield, M. A., Flick, R. E., Bak, A. S., et al. (2019). Sixteen years of bathymetry and waves at san diego beaches. *Sci. Data*, *6*, 1–13. https://doi.org/10.1038/s41597-019-0167-6

Luijendijk, A., Hagenaars, G., Ranasinghe, R., Baart, F., Donchyts, G., & Aarninkhof, S. (2018). The state of the world's beaches. *Sci. Rep.*, *8*, 1–11. https://doi.org/10.1038/s41598-018-24630-6

Masson-Delmotte, V., Zhai, P., Pirani, S., Connors, C., Péan, S., Berger, N., Caud, Y., Chen, L., Goldfarb, M., & Scheel Monteiro, P. M. (2021). Ipcc, 2021: Summary for policymakers. in: Climate change 2021: The physical science basis. contribution of working group i to the sixth assessment report of the intergovernmental panel on climate change.

Miller, J. K., & Dean, R. G. (2004). A simple new shoreline change model. *Coastal Engineering Journal*, *51*(4), 385–407.

Montaño, J., Coco, G., Antolínez, J. A. A., Beuzen, T., Bryan, K. R., Cagigal, L., Castelle, B., Davidson, M. A., Goldstein, E. B., Ibaceta, R., Idier, D., Ludka, B. C., Masoud-Ansari, S., Méndez, F. J., Murray, A. B., Plant, N. G., Ratliff, K. M., Robinet, A., Rueda, A., & Vos, K. (2020). Blind testing of shoreline evolution models. *Scientific Reports*, *10*(1). https://doi.org/10.1038/s41598-020-59018-y

Montaño, J., Coco, G., Cagigal, L., Mendez, F., Rueda, A., Bryan, K. R., & Harley, M. D. (2021). A multiscale approach to shoreline prediction. *Geophysical Research Letters*. https://doi.org/10.1029/2020GL090587

Nie, S., Wen, Q., Zhou, Z., Yang, R., Song, S., & Liu, L. (2023). Time-series transformer with patch attention for long-term series forecasting. *Proceedings of the 39th International Conference on Machine Learning (ICML)*. https://proceedings.mlr.press/v162/nie22a.html

Paprotny, D., Rhein, B., Vousdoukas, M. I., Terefenko, P., Dottori, F., Treu, S., Śledziowski, J., Feyen, L., & Kreibich, H. (2024). Merging modelled and reported flood impacts in europe in a combined flood event catalogue for 1950–2020. *Hydrology and Earth System Sciences*, *28*, 3983–4010. https://doi.org/10.5194/hess-28-3983-2024

Reguero, B. G. y. o. (2019). A recent increase in global wave power as a consequence of oceanic warming. *Nat. Commun.*, *10*, 1–14. https://doi.org/10.1038/s41467-018-08066-0

Repina, O., Carvalho, R. C., Coco, G., Antolínez, J. A., de Santiago, I., Harley, M. D., Jaramillo, C., Splinter, K. D., Vitousek, S., & Woodroffe, C. D. (2025). Evaluating five shoreline change models against 40 years of field survey data at an embayed sandy beach. *Coastal Engineering*, *199*, 104325. https://doi.org/10.1016/j.coastaleng.2024.104325

Rojas, R., & Rojas, R. (1996). The backpropagation algorithm. *Neural networks: a systematic introduction*, 149–182.

Schepper, K., & Van Oordt, M. (2021). Evaluating transformers in time series forecasting for coastal dynamics. *Coastal Engineering Journal*, *63*, 24–37. https://doi.org/10.1142/S0578563421400037

Senechal, N., & Coco, G. (2024a). On the role of hydrodynamic and morphologic variables on neural network prediction of shoreline dynamics. *Geomorphology*, *451*, 109084.

Senechal, N., & Coco, G. (2024b). On the role of hydrodynamic and morphologic variables on neural network prediction of shoreline dynamics. *Geomorphology*, *451*, 109084.

Splinter, K. D., Turner, I. L., Davidson, M. A., Barnard, P. L., & Castelle, B. (2014). A generalized equilibrium model for predicting daily to interannual shoreline response. *Journal of Geophysical Research: Earth Surface*, *119*(9), 1936–1958. https://doi.org/10.1002/2014JF003106

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929–1958.

Stive, M. J., & de Vriend, H. J. (2002). Modeling sediment transport and morphodynamics of coastal environments. *Coastal Engineering*, *47*, 153–177.

Stringari, C., Harris, D., & Power, H. (2019). A novel machine learning algorithm for tracking remotely sensed waves in the surf zone. *Coastal Engineering*, *147*, 149–158. https://doi.org/10.1016/j.coastaleng.2019.02.002

Taylor, K. E. (2001). Summarizing multiple aspects of model performance in a single diagram. *Journal of Geophysical Research: Atmospheres*, *106*, 7183–7192. https://doi.org/10.1029/2000JD900719

Temmerman, S., Meire, P., Bouma, T. J., Herman, P. M., Ysebaert, T., & De Vriend, H. J. (2013). Ecosystem-based coastal defence in the face of global change. *Nature*, *504*, 79–83. https://doi.org/10.1038/nature12859

Tolman, H. L. (2009). *User manual and system documentation of wavewatch iii version 3.14* (Technical Note No. 276). NOAA / NWS / NCEP / MMAB.

Turner, I., Harley, M., Short, A., Simmons, J., Bracs, M., Phillips, M., et al. (2016). A multi-decade dataset of monthly beach profile surveys and inshore wave forcing at narrabeen, australia. *Sci. Data*, *3*, 160024. https://doi.org/10.1038/sdata.2016.24

United Nations. (2017). Factsheet: People and oceans. In: Proceedings of the Ocean Conference, pp. 1–7.

Van, S. P., Le, H. M., Thanh, D. V., Dang, T. D., Loc, H. H., & Anh, D. T. (2020). Deep learning convolutional neural network in rainfall–runoff modelling. *Journal of Hydroinformatics*, *22*, 541–561. https://doi.org/10.2166/hydro.2020.095

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, *30*, 6000–6010.

Vitousek, S., Barnard, P. L., & Limber, P. (2017). Can beaches survive climate change? *Journal of Geophysical Research: Earth Surface*, *122*, 1060–1067. https://doi.org/10.1002/2017JF004308

Vos, K., Splinter, K. D., Harley, M. D., Simmons, J. A., & Turner, I. L. (2019). Coastsat: A google earth engine-enabled python toolkit to extract shorelines from publicly available satellite imagery. *Environmental Modelling Software*, *122*, 104528. https://doi.org/10.1016/j.envsoft.2019.104528

Vousdoukas, M. I., et al. (2020). Sandy coastlines under threat of erosion. *Nature Climate Change*, *10*, 260–263. https://doi.org/10.1038/s41558-020-0697-0

Watkiss, P., Troeltzsch, J., McGlade, K., Watkiss, M., et al. (2019). The economic cost of climate change in europe: Synthesis report on coacch interim results. policy brief by the coacch project.

Wright, L., & Short, A. (1984). Morphodynamic variability of surf zones and beaches: A synthesis. *Marine Geology*, *56*, 93–118.

Wright, L., Short, A., & Green, M. (1985). Short-term changes in the morphodynamic states of beaches and surf zones: An empirical predictive model. *Marine Geology*, *62*, 339–364.

Yates, M. L., Guza, R. T., O'Reilly, W. C., & Seymour, R. J. (2009). Equilibrium shoreline response of southern california beaches to wave climate. *Marine Geology*, *267*(1–2), 1–14.

Yial, N., et al. (2021). Advanced deep learning models for forecasting shoreline changes under extreme weather conditions. *Journal of Coastal Research*, *38*, 1054–1065. https://doi.org/10.2112/JCOASTRES-D-20-00103.1

Zeinali, S., Dehghani, M., & Talebbeydokhti, N. (2021). Artificial neural network for the prediction of shoreline changes in narrabeen, australia. *Applied Ocean Research*, *107*, 102362. https://doi.org/10.1016/j.apor.2020.102362