Scientific Research Publishing

Atmospheric and Climate Sciences, 2023, 13, 587-606
https://www.scirp.org/journal/acs
ISSN Online: 2160-0422
ISSN Print: 2160-0414

A Wavelet-Based Deep Learning Framework for Predicting Peak Intensity of Hurricanes in the Atlantic Ocean

Jiahe Liu, Xiaodi Wang

The Mathematics Department, Western Connecticut State University, Danbury, USA
Email: jackliupenny@gmail.com

How to cite this paper: Liu, J.H. and Wang, X.D. (2023) A Wavelet-Based Deep Learning Framework for Predicting Peak Intensity of Hurricanes in the Atlantic Ocean. *Atmospheric and Climate Sciences*, **13**, 587-606.
https://doi.org/10.4236/acs.2023.134033

Received: July 10, 2023
Accepted: October 27, 2023
Published: October 30, 2023

Abstract

Every year, hurricanes pose a serious threat to coastal communities, and forecasting their maximum intensities has been a crucial task for scientists. Computational methods have been used to forecast the intensities of hurricanes across varying time horizons. However, as climate change has increased the volatility of the intensities of recent hurricanes, newer and adaptable methods must be devised. In this study, a framework is proposed to estimate the maximum intensity of tropical cyclones (TCs) in the Atlantic Ocean using a multi-input convolutional neural network (CNN). From the Atlantic hurricane seasons of 2000 through 2021, over 100 TCs that reached hurricane-level wind speeds are used. Novel algorithms are used to collect and preprocess both satellite image data and non-image data for these TCs. Namely, Discrete Wavelet Transforms (DWTs) are used to decompose individual bands of satellite image data, eliminating noise and extracting hidden frequency details before training. Validation tests indicate that this framework can estimate the maximum wind speed of TCs with a root mean square error of 15 knots. This framework provides preliminary predictions that can supplement current computational methods that would otherwise not be able to account for climate change. Future work can be done by forecasting with time constraints, and to provide estimations for more metrics such as pressure and precipitation.[1]

Keywords

Tropical Cyclone (TC), Hurricane Intensity, Convolutional Neural Network (CNN), Discrete Wavelet Transform (DWT)

[1]A GitHub repository containing the code developed in this study can be found at https://github.com/jliu2006/hurricane_intensity.

1. Introduction

In the U.S., hurricanes in the last 30 years have caused over $1.1 trillion in damages and killed almost 7000 people, making them both the economically costliest and the deadliest type of disaster in the U.S. [1]. Furthermore, hurricanes have intensified significantly in recent years to cause enormous amounts of damage [2]. 2020 saw a record-breaking 30 named tropical storms in the Atlantic Ocean, exhausting the designated list of 21 storm names for only the second time on record [3]. That list was used up again in 2021, as another 21 named storms totaled $95 billion in damages [1] [4]. Hurricane Ida alone accounted for more than half of that cost, being responsible for at least $75 billion in damages [5].

The Atlantic hurricane season makes it evident that climate change has caused hurricanes to worsen in severity. However, coastal cities in the U.S. are only one part of the world's population and economic output. Hurricanes pose the same threat to coastal communities around the world [6], and therefore a method to reliably forecast the maximum intensities of impending hurricanes would be essential to prepare accordingly. Most importantly, these forecasts should be made while these hurricanes are still in early formation to provide as much time to prepare as possible.

Currently, there are a number of computational frameworks that estimate the peak intensities of TCs. One such method is Dvorak Analysis (DA), which utilizes four distinct geophysical properties in its calculation of TC intensity: vorticity, vertical wind shear, convection, and core temperature [7]. Each of these properties helps to relate pattern recognition in cloud formations to a maximum wind speed. When monitoring the physical properties of cloud formations, DA also tracks four primary types of patterns that are assigned to varying ranges of TC intensity: curved band patterns, shear patterns, central dense overcast, and eye tracking. For decades, DA has remained one of the most accurate and internally consistent methods of TC intensity estimation in the world. The most recent study of its accuracy, performed by Brown and Franklin in 2004, tested DA's performance on TCs in the Atlantic Ocean from 1997 to 2003. They found that the Dvorak-estimated maximum wind had an error of 5 knots (kts) or less in 50% of TCs. 75% of errors were within 12 kts, and 90% of errors were within 18 kts [8]. For this study, DA serves as a useful computational benchmark, although attention must also be drawn to more recent studies of estimating TC intensity.

An improvement of DA known as early-stage Dvorak analysis (EDA) was created by the Japan Meteorological Agency to estimate TC intensity based on satellite infrared imagery. EDA has since been used by the National Hurricane Center (NHC) and Central Pacific Hurricane Center (CPHC) to predict TC generation 48 hours in advance with accuracies of 15% - 57%, respectively [9]. A study combining EDA with multi-model ensemble forecasts found that the accuracies of forecasting TC generation could be improved to 35% - 79% [10].

With EDA as a benchmark, Matsuoka *et al.* [11] developed a deep-learning

approach to identify TCs and their precursor cloud formations based on outgoing longwave radiation (OLR) data using convolutional neural networks. In the western North Pacific, their framework identified TCs from July to November with a probability of detection of 79.9% - 89.1% and a false alarm ratio (FAR) of 32.8% - 53.4%. The framework may be applied to other ocean basins and can maintain a POD above 70% and a FAR below 50%. The accepted FAR for TC identification was relatively high, suggesting that because of climate change, providing reliable forecasts of the intensity of TCs has become markedly more challenging as well. Most notably, the high FAR indicates that their framework had a high tendency to falsely categorize cloud formations as TCs. This study proposes a framework designed to estimate the peak wind speed of TCs with a comparatively lower level of bias.

DeMaria *et al.* [12] proposed a Monte Carlo probability (MCP) method for estimating wind speed probabilities of TCs. MCP first generates path and intensity realizations for each TC, taking into account randomly sampled historical errors from the last 5 years. For each set of realizations, MCP then calculates wind structure realizations to reveal key information about the radii of wind thresholds in each TC. 1000 realizations are made which extend out to 120 hours in advance, and MCP uses them to calculate radii probabilities for 34-kt, 50-kt, and 64-kt winds. When applied to TCs in the Atlantic Ocean from 2003 to 2007, MCP produced 48-hour forecasts with a root mean square error of about 20 knots. For this study, MCP serves as a second computational benchmark, although MCP is a probabilistic approach that estimates wind speeds within ranges, and it can be expected that attempting to forecast exact wind speeds will incur larger errors.

Herrera *et al.* [13] used several types of non-linear regression models to forecast the movement and intensities of TCs. While this study did not utilize deep learning methods such as convolutional neural networks (CNNs), it applied wavelet analysis to find and forecast oscillating patterns of Atlantic hurricanes from categories 2 to 5. This demonstrates that wavelet analysis methods are viable methods of spatial data preprocessing when applied to TCs. In this study, CNN is enhanced through the use of discrete M-band wavelet transforms (DWTs) a method of wavelet analysis that this study will use to decompose spatial data into extracted detail channels.

Su *et al.* [14] used satellite imagery from NASA's Tropical Rainfall Measuring Mission (TRMM) to classify rapidly intensifying TCs by category. They found that the surface precipitation rate (mm/hr) within 100 km of a storm center had a strong correlation with the category of a TC. These findings are a crucial step in TC intensity forecasting, since spatial precipitation data is a feature that can be readily extracted and utilized in future intensity forecasting models. Similar to Herrera et al, however, this study did not experiment with CNNs. In this study, using spatial data on precipitation rate within a TC allows the CNN framework to forecast TC intensities with greater accuracy.

Deep learning models have the ability to forecast the intensity of TCs. This study aimed to fill in the research gap left by Herrera *et al.* and Hu *et al.* by using a CNN-based model framework to forecast the peak intensity of a TC during early stages of formation. The algorithm created by Hu *et al.* was adapted by including data collected on the rate of precipitation in each TC, whose high correlation with TC intensity could be significant when applied to early-stage forecasting. This study also incorporated the methodology of Herrera *et al.* by using DWTs to preprocess our satellite imagery, which can reveal hidden details and features that would be beneficial for our CNN model framework.

2. Data and Methods

The objective of this paper was to introduce a novel deep learning-based approach for estimating maximum TC intensity. Computational approaches such as DA and MCP have been historically consistent, but as climate change makes TCs more volatile and unpredictable, these methods will be unable to adapt. Computational methods are rigid by nature, and modifying or adding new parameters to adapt to changes in TCs can be extremely costly and time-consuming. Deep learning serves as an extremely flexible and cost-effective approach that can quickly provide preliminary intensity estimates. Existing methodologies for estimating TC intensity rely on maximum wind speed as the primary indicator of its strength, which is commonly measured in nautical miles per hour, or knots (kts). Therefore, our approach also includes estimations in kts.

2.1. Study Area

This study focused on the Atlantic hurricane season, which is tracked in detail by national environmental agencies in the United States. Additionally, much of the information recorded by satellites and weather observation stations is made publicly available. The Atlantic Ocean basin is the second largest basin, covering an area of approximately 106,400,000 square kilometers, or 41,100,000 square miles [15]. From 2000 through 2021, the Atlantic Ocean basin saw the formation of nearly 400 TCs, including tropical depressions and subtropical storms [16] [17].

2.2. Data Collection

The first step was to gather spatial data representing three crucial features of TC development: precipitation data, ocean temperature data, and satellite image data. Global 6-hourly precipitation data were collected from two NASA missions, the Tropical Rainfall Measuring Mission (TRMM) and the Global Precipitation Measurement Mission (GPM), which are collectively referred to as Integrated Multi-satellitE Retrievals for GPM (IMERG). Global daily ocean surface temperature data was collected from the National Oceanic and Atmospheric Administration's (NOAA) Optimum Interpolation Sea Surface Temperature record (OISST). Daily global satellite imagery was taken from NASA's Moderate Reso-

lution Imaging Spectroradiometer (MODIS). Our data collection process also required the use of NOAA's Northeast and North Central Pacific Hurricane Database (HURDAT2). Table 1 provides a summary of the data that was collected in this step.

For each TC that formed, HURDAT2 provided a profile that tracked the location of its center as well as wind speeds at varying distances from its center. These profiles contained one such record every 6 hours: at 12:00 AM (0000), 6:00 AM (0600), 12:00 PM (1200), and 6:00 PM (1800). A script was developed to automatically download these profiles as JSON files, which were then used to download spatial data. Because the objective of this study was to create an early forecasting system, only data of each TC's path up until landfall was collected. The exception to this process was if a TC made landfall over any of the Caribbean islands, since TCs that pass over this region sometimes do not die out and instead continue to grow, often making contact over land multiple times. Figure 1 shows the path of Hurricane Sandy in 2012, which made landfall three times and serves as an example of why this exception is necessary. Two bounding boxes were created to cover as much of the Caribbean islands as possible: one between 16˚N 85˚W and 25˚N 65˚W, and one between 23˚N 79˚W and 26˚N

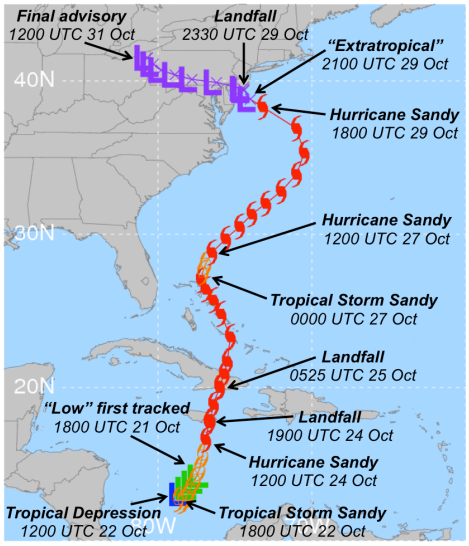

Figure 1. Sandy_track.

Table 1. The table should consist of the following data.

	Data Source	Features Used	Temporal Resolution	Spatial Resolution
MODIS	MOD09CMG	Satellite Imagery	24 hours	0.05˚
IMERG	TRMM/GPM	Hourly precipitation	6 hours	0.1˚
NOAA	OISST	Skin surface temperature	24 hours	0.25˚
NOAA	HURDAT2	TC path, wind speed	6 hours	0.1˚

74˚W. If a TC made contact with land anywhere within this area, the script would ignore the landing and continue to download records until the TC made landfall over a point not in either of the bounding boxes. Figure 2 uses the HURDAT2 profile of Hurricane Ida in 2021 as an example of how this downloading process worked.

A second script was then created to download data from MODIS, IMERG, and NOAA using the correct days and locations from the HURDAT2 TC profiles. For each day in a TC's HURDAT2 profile, the earliest recorded coordinates were directly used to specify identical bounding boxes for IMERG and OISST, which created subimages that were downloaded as 200 × 200 spatial arrays. Note that, in order to obtain daily precipitation values for each TC, it was necessary to use each 6-hourly record from HURDAT2 to average the 6-hourly records from IMERG.

For the daily satellite imagery from MODIS, getting precise and accurate images of each TC was crucial, since the presence of a clearly-defined formation of clouds into spirals was likely a strong indicator of TC strength. Therefore, we modified our approach for processing MODIS data into three steps to ensure that each TC's eye was centered in its image and that the model could learn as many features from this imagery as possible.

2.3. Data Preprocessing

2.3.1. Hurricane Center Identification in MODIS Satellite Imagery

The first step in MODIS preprocessing was to center the eye of every TC to maximize the efficiency of the deep learning model in its feature extraction. This step also ensured that each image was as consistent as possible in terms of extracted area surrounding each TC. MOD09CMG is one satellite that provides continuous imagery of the entire globe as it orbits, so it often covers the Atlantic Ocean at times that differed from the 6-hour intervals provided by HURDAT2. This difference in time would allow a TC to move so that the coordinates of its eye provided by HURDAT2 would no longer represent the location of its eye in the MODIS satellite imagery.

```
AL092021,                IDA,     40,
20210826, 1200,  , TD, 16.5N,  78.9W,  30, 1006,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,   60
20210826, 1800,  , TS, 17.4N,  79.5W,  35, 1006,   60,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,   50
20210827, 0000,  , TS, 18.3N,  80.2W,  40, 1004,   60,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,   50
20210827, 0600,  , TS, 19.4N,  80.9W,  45, 1002,   70,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,   40
20210827, 1200,  , TS, 20.4N,  81.7W,  55,  996,   80,   60,    0,   60,   30,    0,    0,    0,    0,    0,    0,    0,   30
20210827, 1800, L, HU, 21.5N,  82.6W,  70,  987,   80,   60,   40,   60,   40,   30,    0,   20,   20,    0,    0,    0,   20
20210827, 2325, L, HU, 22.4N,  83.2W,  70,  988,   80,   60,   40,   60,   40,   30,    0,   20,   20,    0,    0,    0,   20
20210828, 0000,  , HU, 22.6N,  83.5W,  70,  989,  100,   60,   40,   70,   50,   30,    0,   30,   20,    0,    0,    0,   20
20210828, 0600,  , HU, 23.5N,  84.7W,  70,  987,  100,   60,   40,   70,   50,   30,    0,   30,   20,    0,    0,    0,   20
20210828, 1200,  , HU, 24.4N,  85.7W,  70,  986,  110,   80,   60,  100,   50,   40,   20,   30,   25,   20,    0,    0,   20
20210828, 1800,  , HU, 25.6N,  86.6W,  80,  976,  110,  100,   70,  100,   50,   40,   20,   40,   25,   20,   10,   20,   20
20210829, 0000,  , HU, 26.7N,  87.6W,  90,  967,  120,  100,   80,  110,   70,   60,   40,   60,   35,   30,   20,   30,   20
20210829, 0600,  , HU, 27.6N,  88.7W, 115,  950,  120,  100,   80,  110,   70,   60,   40,   60,   35,   30,   20,   30,   15
20210829, 1200,  , HU, 28.5N,  89.6W, 130,  929,  130,  110,   80,  110,   70,   60,   40,   60,   45,   35,   20,   30,   10
20210829, 1655, L, HU, 29.1N,  90.2W, 130,  931,  130,  110,   80,  110,   70,   60,   40,   60,   45,   35,   20,   30,   10
20210829, 1800,  , HU, 29.2N,  90.4W, 125,  932,  130,  120,   80,   80,   70,   60,   40,   40,   45,   35,   20,   25,   10
20210830, 0000,  , HU, 29.9N,  90.6W, 105,  944,   80,  120,   80,   70,   50,   60,   40,   40,   30,   30,   20,   20,   10
20210830, 0600,  , HU, 30.6N,  90.8W,  65,  978,   80,  130,   80,   40,   50,   50,    0,    0,   30,   30,    0,    0,   30
20210830, 1200,  , TS, 31.5N,  90.9W,  40,  992,   50,  160,   60,   30,    0,    0,    0,    0,    0,    0,    0,    0,   40
20210830, 1800,  , TS, 32.2N,  90.5W,  35,  996,    0,  160,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,   80
20210831, 0000,  , TD, 33.0N,  90.0W,  30,  996,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,  250
20210831, 0600,  , TD, 33.8N,  89.4W,  25,  996,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,  210
20210831, 1200,  , TD, 34.4N,  88.4W,  25,  996,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,  250
```

Figure 2. Ida_hurdat.

To accurately identify the location of each TC's eye, it was necessary to develop an algorithm to linearly interpolate the coordinates provided by HURDAT2 based on its temporal difference from those same coordinates provided by MOD09CMG, which was called Hurricane Center Identification (HCI). MOD09CMG provides a time array so that for every pixel of imagery taken in a given day, there is a corresponding time pixel. Given the profile of a TC from HURDAT2 and a certain day of MODIS imagery, HCI first finds the earliest record of that day (ex. 12:00 AM) in the TC's profile. It then uses the coordinates of its eye to get the timestamp that corresponds to that location in the given MODIS imagery. Next, HCI searches the TC's HURDAT2 profile for two 6-hour intervals: one whose timestamp is closest to and before the time obtained from MODIS, and one whose timestamp is closest to and after that time. HCI then stores the times and coordinates of both HURDAT2 records. With the location and time of a TC from HURDAT2 and the time from MODIS, HCI is able to utilize linear interpolation to calculate the new coordinates of the TC's eye in MODIS (Figure 3).

A few factors negatively impacted the accuracy of HCI, the first being the varying velocities at which a TC could travel. Because HCI linearly interpolated the new coordinates of each TC only by its temporal differences from HURDAT2 records, it did not account for any fluctuations in the TC's velocity, such as an acceleration or change in direction. Second, because of the path that MOD09CMG took as it orbited, it produced several evenly spaced and identical areas with no satellite data, known as swath gaps. If HCI found that a set of coordinates from HURDAT2 corresponded to a time of 0000 in the MODIS time matrix, this meant that it had encountered one such swath gap, and it instead had to take the average time at which the area surrounding the swath gap was taken. Despite these three factors having an adverse effect on the potential accuracy of HCI, manual assessment of sample images revealed that, in practice, these factors did not translate to any significant error.

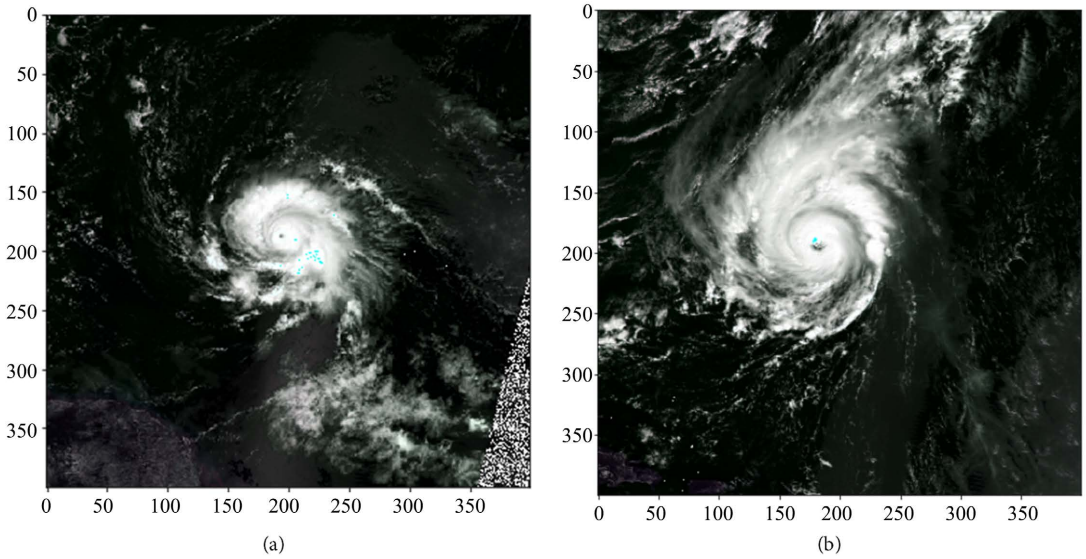

Figure 3. (a) SAM 9-25-2021; (b) SAM 10-01-2021.

2.3.2. Dynamic Swath Gap Filling in MODIS Satellite Imagery

Due to the Earth's rotation and the path that MOD09CMG takes as it orbits the Earth, there are regular swath gaps in each day's MODIS satellite image. At the equator, these swath gaps are 388 kilometers in width, and their widths gradually reduce to zero at ±30˚ latitude. One such swath gap covers part of the Caribbean Sea and therefore appears in some MODIS satellite subimages. This swath gap poses as a substantial problem for a deep learning model, which would be trained to detect the presence of a swath gap as a significant feature instead of the appearance of each TC. Therefore, these swath gaps had to be filled in with pseudo-random values so that there would be both no substantial pattern in the swath gaps and no significant difference between the colors of the filled gap and the colors of the rest of the satellite image. This way, our deep learning model would be less likely to extract features from and train on the swath gap itself.

To address this issue, modifications were made to a previously developed algorithm called Neighbor RGB to fill in swath gaps using randomly selected pixel values within a certain radius from the gap [18]. The resulting algorithm, called Dynamic Swath Gap Filling (DSGF), was able to fill in swath gaps given a processed multi-channel satellite image by randomly picking neighboring pixel values in a dynamically-changing area surrounding the swath gap. First, DSGF started from the bottom left corner of each channel, checking the values of each row from left to right. If DSGF found that a certain value in a row equals −28,672, the defined value for any area representing a swath gap, it saved the index of that value in the row and began counting upwards by 1 to track the length of the swath in that row. DSGF then continued checking every single value until it reached either the end of the swath gap or the end of the row, at which point it calculated the area around the swath to randomly select values from during filling. DSGF did this by dividing the length of the swath gap by 2, obtaining the length of the area to the left of the swath gap. The remaining length represented the length of the area to the right of the swath gap. A check was then made so that if the selected length to either the left or right of the swath gap exceeded the length of its corresponding part of the row (*i.e.*, the selected length would go past the borders of the row), DSGF took the maximum length possible from that side and add the remaining length onto the area covered by the other side. With the dynamic length of the left and right areas of filled values defined, DSGF then stored the values in those lengths from the 9 rows above the current row into an array and randomly selected values out of that array to fill the swath gap in that row (**Figure 4**).

2.3.3. Discrete M-Band Wavelet Transform

The third algorithm in our data preprocessing utilized Discrete M-Band Wavelet Transforms (DWTs) to decompose each spatial channel into M^2 different frequency channels, or components. DWTs are a crucial step in image preprocessing because they can emphasize hidden details by separating low-frequency components from high-frequency counter parts of the image. These details allow

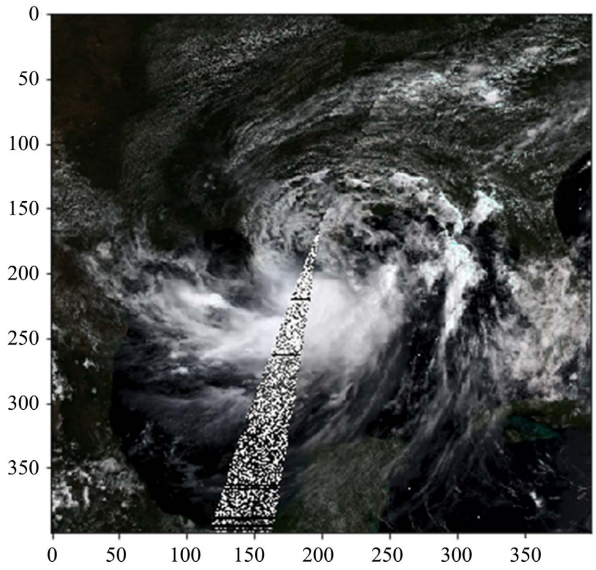

Figure 4. BARRY dynamic swath gap filling 7-12-2019.

our deep learning model to learn relationships with fewer TCs. An orthogonal M-band DWT is determined by a filter bank consisting of M filters ($M \geq 2$), including a low-pass filter α and $M-1$ high-pass filters $\beta^{(j)}$ for $j=1,\cdots,M-1$. An orthogonal M–band wavelet filter bank is said to have N vanishing moments if its filters satisfy the following conditions [19]:

$$\sum_{i=1}^{n} \alpha_i = \sqrt{M} \tag{1}$$

$$\sum_{i=1}^{n} i^k \beta_i^{(j)} = 0 \quad \text{for} \quad k=0,\cdots,N-1, \quad j=1,\cdots,M-1 \tag{2}$$

$$\alpha = \beta^{(j)} = 1 \quad \text{for} \quad j=1,\cdots,M-1 \tag{3}$$

$$\alpha\beta^{(j)} = 0 \quad \text{for} \quad j=1,\cdots,M-1 \tag{4}$$

$$\beta^{(i)}\beta^{(j)} = 0 \quad \text{for} \quad i,j=1,\cdots,M-1, \text{ and } i \neq j \tag{5}$$

Figure 5 illustrates the four frequently employed wavelet functions that were considered in this study: Biorthogonal-1.3, Daubechies-3, Symlets and Discrete Meyer, as well as their associated decomposed image components when applied on a satellite image of Hurricane Ida.

For this study, the bi-orthogonal DWT was chosen because of its ability to detect and filter out white Gaussian noise, or high contrast of neighboring pixel intensity values [20]. This helped to filter out cloud formations that were small and unrelated to the TC itself. Bi-orthogonal wavelets create associated wavelet transforms that are invertible but not necessarily orthogonal. Furthermore, bi-orthogonal wavelet transforms are symmetrical, while orthogonal ones are not. This allows bi-orthogonal wavelets to have more freedom than orthogonal wavelets [21]. For each bi-orthogonal wavelet, its filter banks must each satisfy the bi-orthogonality condition:

$$\sum_{n\in Z} a_n \overline{a}_{n+2m} = 2\delta_{m,0} \tag{6}$$

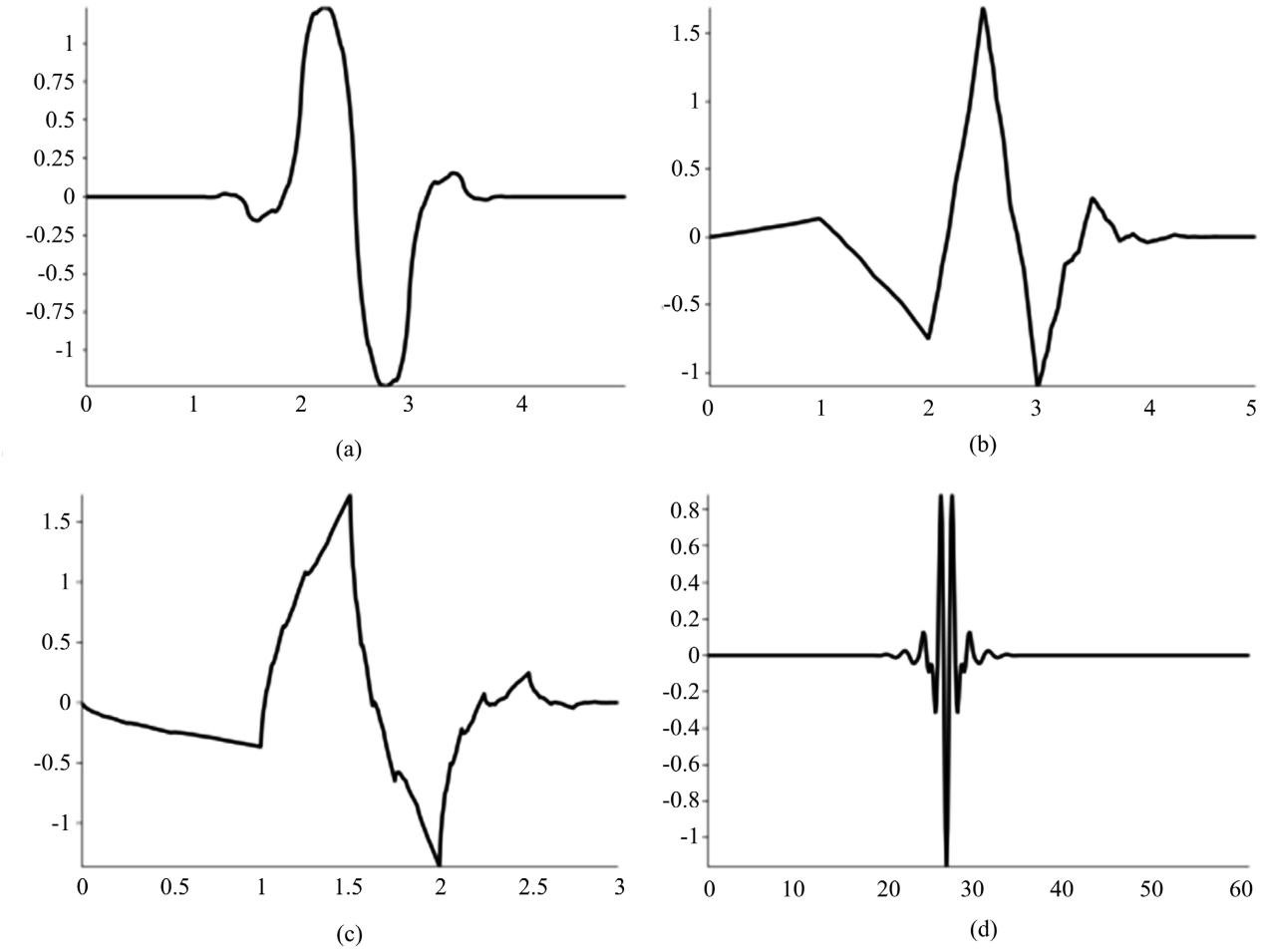

Figure 5. (a) Biorthogonal 1.3 wavelet; (b) Daubechies 3 wavelet; (c) Symlets wavelet (1); (d) Discrete meyer wavelet.

Each wavelet filter's elements can then be determined by:

$$b_n = (-1)^n \bar{a}_{N-1-n} \quad (n = 0, \cdots, N-1) \tag{7}$$

$$\bar{b}_n = (-1)^n a_{M-1-n} \quad (n = 0, \cdots, M-1) \tag{8}$$

We apply the biorthogonal DWT to our image dataset. Because it is a 2-band transform, it produced 4 components for every image channel: an approximation (a) component and horizontal (h), vertical (v), and diagonal (d) detail components (**Figure 6** and **Figure 7**).

2.4. Multi-Input Parallel Convolutional Neural Network

2.4.1. Artificial Neural Network

An Artificial Neural Network (ANN) consists of three types of layers: input layers, hidden layers, and output layers. Each layer is then composed of neurons. Every neuron in one of these layers takes input values x from the neurons in the preceding layer. The neuron then performs a weighted sum with trainable weights w and adds a trainable bias value b_i. Finally, the neuron applies a specified activation function f to produce an output value z_n, which it will feed as an

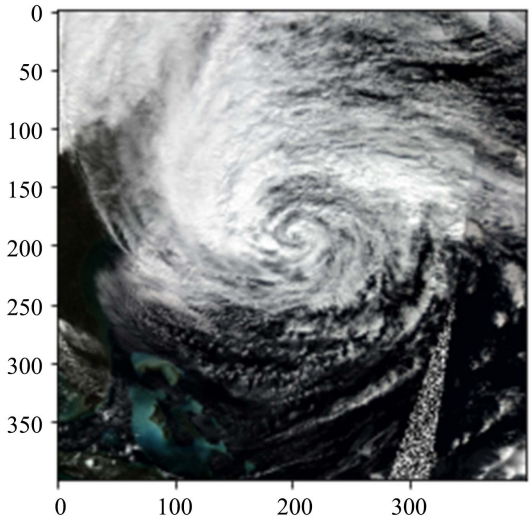

Figure 6. Sandy-10-28-2012_original.

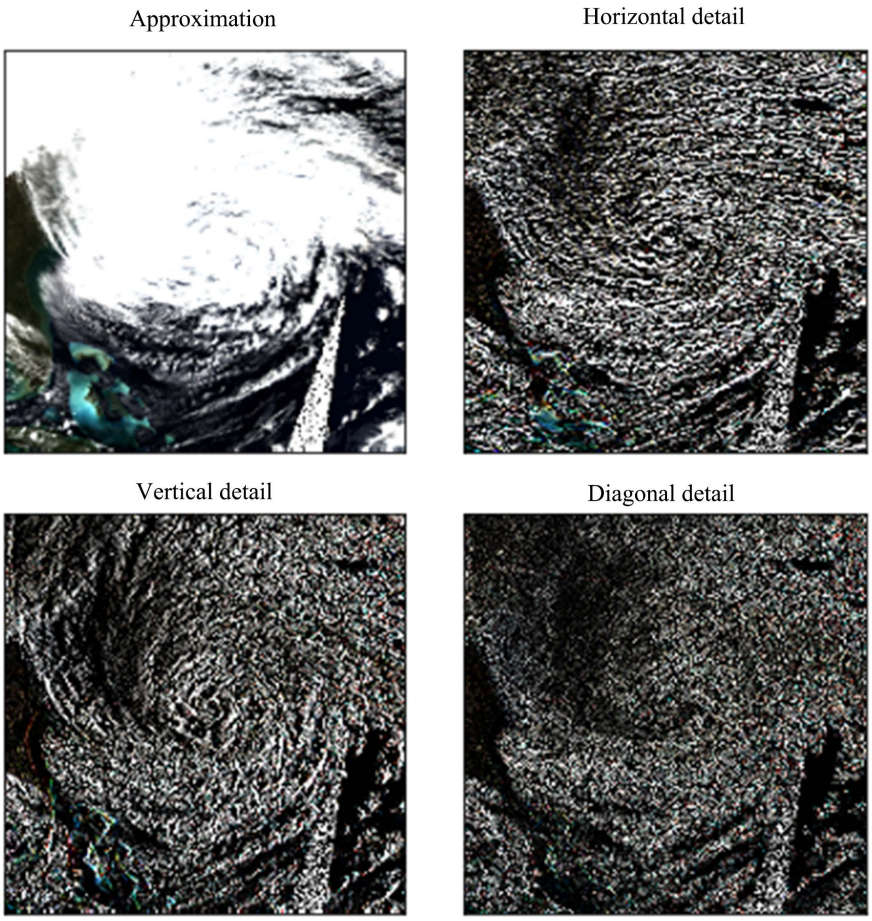

Figure 7. Sandy-10-28-2012_wavelet.

input value into every neuron in the following layer. Because hidden and output layers can consist of neurons that take input values from every neuron in the preceding layer, they can also be defined as fully connected layers, or Dense lay-

ers. Given an ANN with n neurons in each layer, any neuron in the fully connected layers of the model will produce output values that can be determined by the following two equations:

$$y_i = b_i + \sum_{j=1}^{n} w_j x_j \tag{9}$$

$$z_i = f\left(y_i\right) \tag{10}$$

2.4.2. Convolutional Neural Network

A Convolutional Neural Network (CNN) is a type of ANN that is generally used when features need to be learned from input data [22]. In addition to the fully connected layers of an ANN, a CNN makes use of a convolutional layer and a pooling layer, which make up the feature learning portion of the neural network. The input variable x into this CNN consists of c 2D spatial channels. One such channel x_b, consisting of $P \times P$ pixels, is filtered through a convolutional layer which uses N $Q \times Q$ convolution windows, or filters. Starting from the top-left corner, each filter takes an identically shaped array (*i.e.*, $Q \times Q$) of input values from the channel and multiplies it with the weighted values of the filter. This process is repeated for the entire spatial channel as the window is shifted. The elements in each input channel may be expressed as $x_{i,j}\left(0 \le i \le P, 0 \le j \le P\right)$, and the weights in each filter can be expressed as $w_{s,t,n}\left(0 \le s \le Q-1, 0 \le t \le Q-1, 0 \le n \le N\right)$. The associated bias unit for each filter can be expressed as b_n. Therefore, elements in a filtered channel (the output of a given convolutional layer) can be written as $y_{i,j,n}$ and calculated using the following equation:

$$y_{i,j,n} = \sum_{s=0}^{Q-1} \sum_{t=0}^{Q-1} w_{s,t,n} \cdot \hat{x}_{i+s,j+l} + b_n \tag{11}$$

The first step in developing the model framework was to create a convolutional block to learn features from my image dataset, which was done with two convolutional layers, each followed by an average pooling and batch normalization layer.

12 filters were used in each convolutional layer and a large filter shape of (11, 11) to allow our model to prioritize the detection of large TC features such as a clearly-defined center or tail. Because there are twelve channels and therefore twelve 11 × 11 convolutional windows stacked on top of each other, the convolutional windows together form a 3-dimensional array.

Note that the model utilized a zero-padding technique along with a convolutional stride size of (1, 1) to ensure that the output size of the filtered spatial data was identical to that of the input data. The Rectified Linear Unit (ReLU) function was used as the activation function of each convolutional layer. ReLU filters out negative input values as shown in the equation below:

$$f\left(y_{i,j,n}\right) = \max\left(y_{i,j,n}, 0\right) \tag{12}$$

2.4.3. Average Pooling

After each convolutional layer, the spatial data was downsampled using average

pooling layers, which take the average (mean) value of a given spatial input over a specified window size (pool size) for each channel of the spatial input. The window then shifts by a specified number of cells (strides). Given a spatial input with shape (x, y), a certain pool size (j, k), and a certain number of strides s, the shape of the downsampled output data can be found with the expression below:

$$\left(\left\lfloor \frac{x-j}{s} \right\rfloor + 1, \left\lfloor \frac{y-k}{s} \right\rfloor + 1\right) \tag{13}$$

2.4.4. Batch Normalization

Following each average pooling layer, the downsampled spatial data was normalized by passing them through batch normalization layers, which transform the input data for each channel b to have a mean output value m_b close to 0 and an output standard deviation close to 1. During the model's training process, each batch normalization layer calculates a moving mean $m'_{p,b}$ and moving variance $v'_{p,b}$ based on the mean and variance of the current epoch p and the current channel b ($m_{p,b}, v_{p,b}$). Given a momentum constant c within the interval $(0,1)$, the batch normalization layer updates the moving mean and variance as follows:

$$m'_{p+1,b} = m'_{p,b} \cdot c + m_{p,b} \cdot (1-c) \tag{14}$$

$$v'_{p+1,b} = v'_{p,b} \cdot c + v_{p,b} \cdot (1-c) \tag{15}$$

Note that \$m\$ and \$v\$ are non-trainable and initialized to the mean and variance of the first epoch, respectively.

Each batch normalization layer also learns a scaling factor γ_b (initialized to 1) and a bias parameter β_b (initialized to 0) for each channel. During model testing, each layer uses these parameters to transform the validation data so that for each spatial input value $i_{x,y,b}$ being normalized, the model would return a normalized output $j_{x,y,b}$ using the following equation:

$$\gamma_b \cdot \frac{i_{x,y,b} - m'_b}{\sqrt{v'_b + \epsilon}} + \beta_b \tag{16}$$

Note that ϵ is a small configurable constant that applies to all channels during batch normalization.

2.4.5. Parallel Feature Learning

Once the convolutional block was created, the next step was to place three of these blocks in parallel so that the model framework would be able to perform feature learning on three sets of spatial input data for each TC. This allowed for the creation of a dataset with three stages such that each stage of input data could represent every TC at a specified stage of early development. This turned the model framework into a three-stage input-based CNN, enabling it to utilize more data and get a better generalization of each TC's strength.

2.4.6. Fully Connected Layers

The final step in developing the model framework was to flatten the learned features from the CNN layers and feed them into two fully connected layers and

one output layer, each with a DropOut regularization rate of 0.5 and a ReLU activation function. With the completed model framework, the next step was to begin experimentation.

2.4.7. System Flowchart

Figure 8 is a flowchart of our system framework, including all data preprocessing algorithms and CNN layers. Each step of the framework on the right is fully automated and color coded based on which one of four main processes it falls into, which are shown on the left.

3. Experiments and Results

Once the three-stage model framework was developed, experimentation was done by using four different sets of three days of each TC's formation to simulate various stages of early forecasting for all TCs (ex. days one through three, two through four, …). Once each scenario was configured and the appropriate data was processed, we randomly split the datasets into training and testing samples. 80% of the data was used for training and the remaining 20% was used to evaluate model performance. The root mean square error (RMSE) and mean absolute error (MAE) were recorded for the training and validation results of each scenario. Figures 9-11 compare the predictions of our trained model to each TC's actual maximum wind speed.

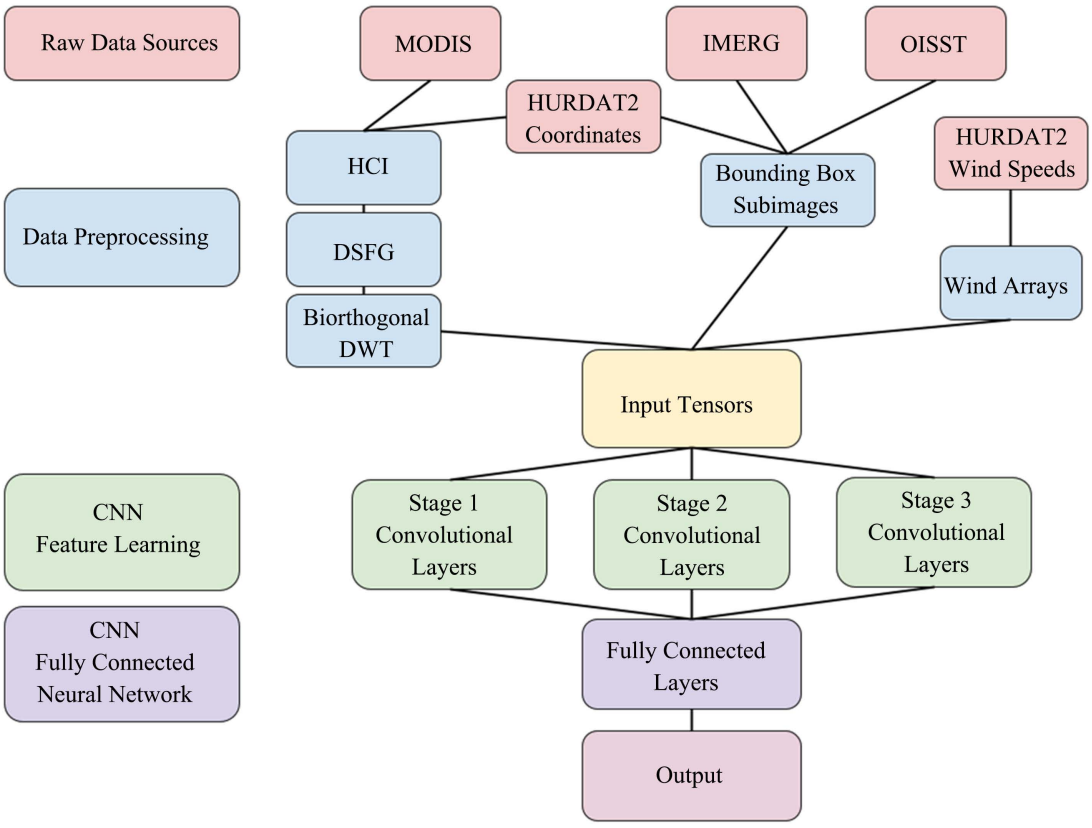

Figure 8. Framework_flowchart.png.

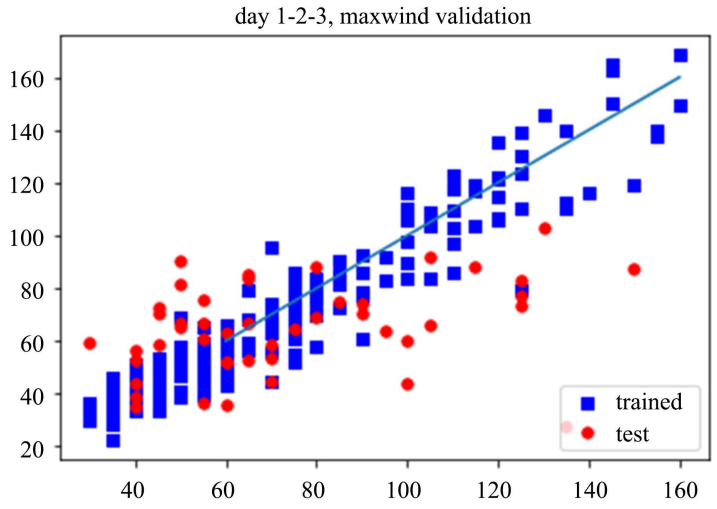

Figure 9. Days 1, 2, 3.

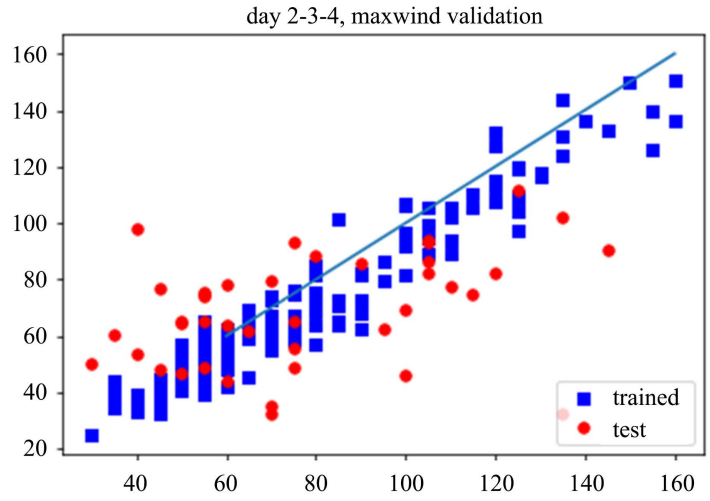

Figure 10. Days 2, 3, 4.

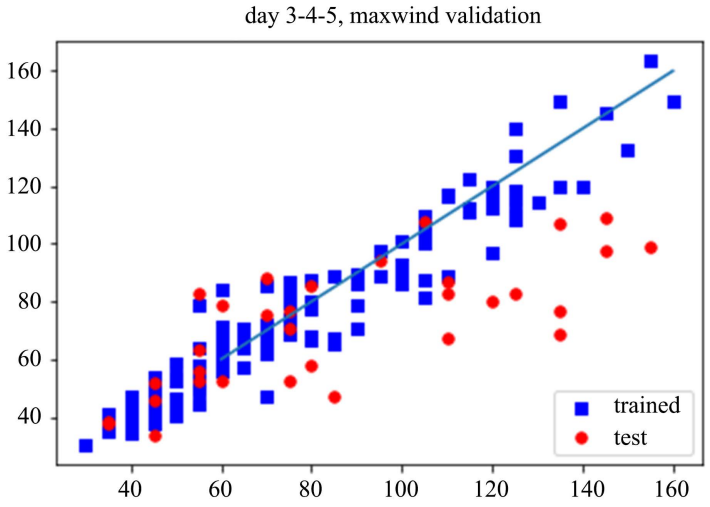

Figure 11. Days 3, 4, 5.

3.1. Estimating Peak Wind Speed Using 3 Consecutive Days

First, the three-stage model was trained on three consecutive days during each TC's development stage to ensure that the images in this dataset would capture patterns of growth and intensifying wind speeds over a constant time interval. This would allow the model to pick up on those patterns during feature learning so that it would attribute larger patterns of growth to a high maximum wind speed.

The training and validation results are shown in Figures 9-11. The x-axis of each graph represents the observed maximum wind speed in knots of each TC, while the y-axis of each graph represents the model's predicted maximum wind speed. Training points are represented by blue squares, and validation points are represented by red dots.

3.2. Estimating Peak Wind Speed Using 3 Consecutive Categories

After training on sets of consecutive days, further experimentation was done using data from the first three categories of each TC. This dataset was filtered so that the model would be trained on the days when each TC had reached its maximum wind speed in the first three categories of TC classification: tropical depression, tropical storm, and Category 1 hurricane. This increased both the time period analyzed and the change in TC intensity and cloud patterns that the model could train on. Because this experimental scenario was based on categories instead of days in TC formation, it fell into a different category. Figure 12 shows the training and validation results of the model when trained on three consecutive categories.

4. Analysis and Discussion

When trained on three consecutive days of each TC, the observed RMSE and MAE slightly improved as the set of consecutive days tested was shifted later in each TC's formation. This is an expected outcome because the selection of days 3

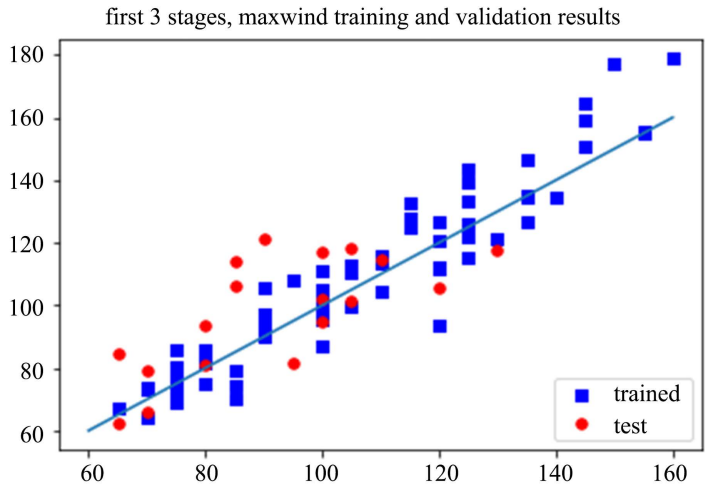

Figure 12. Three consecutive stages.

through 5 excludes TCs that last 4 days or less and don't show significant signs of growth before dying out. As a result, the selection of days effectively filters out insignificant TCs that would have polluted the datasets of the first two selections of days.

Our model performed the best when trained on images of the first three categories of each TC, producing a validation RMSE of 14.95 kts and a validation MAE of 12.08 kts. This is likely due to the fact that the first three categories of a TC show more significant signs of growth than the first three days, allowing the model to better distinguish between stronger and weaker TCs. Additionally, the selection of the strongest day in each of the first three categories means that the resulting dataset would have included satellite imagery of the overall strongest day of any TC that only reached wind speeds of a Category 1 hurricane. This reduced our model's effectiveness as an early warning forecasting system, as it would no longer be using images of TCs during early stages of development to predict their maximum wind speeds at a later point in time.

Our model framework is able to estimate a peak wind speed, but it does not predict a time frame in which that wind speed would be achieved or for how long that wind speed would be sustained. However, compared with other methodologies such as MCP, which was found to have an RMSE of around 20 knots, and DA, which produced an RMSE of approximately 10 knots, this means that our framework remains comparable in terms of error. Furthermore, there are a few aspects in the data sources and framework used that, when improved on, may produce significantly better results.

5. Conclusions

In this study, we developed a fully-automated model framework based on deep learning to both categorize and estimate the peak wind speed of TCs in the Atlantic Ocean. Our framework downloaded spatial data of TCs from 2000 through 2021 with multiple scripts that required no human intervention. Satellite imagery was taken from MOD09CMG, spatial precipitation data from IMERG, sea temperature from OISST, and TC path information and wind speeds from HURDAT2. From over 400 TCs that formed during this time period, our framework selected about 100 TCs for each of the four experimental scenarios we tested. Then, our framework's HCI algorithm calculated the coordinates of a given cyclone for very accurate centering of bounding boxes. To fill in swath gaps left by the MODIS satellite's orbit patterns, our DSGF algorithm randomly replaced swath gap pixels with neighboring pixel values within a certain area. Our framework then applied M^*-Band DWTs to all of the collected spatial channels to reduce noise and extract feature-dense arrays.

This framework estimates the maximum wind speeds of TCs in the Atlantic Ocean with an MAE error of 12.08 knots and has multiple advantages over current computational models. First, it utilizes deep learning during model training, meaning that adapting this framework to climate change requires very little time and resources compared to methods such as DA and MCP. Second, it is ex-

tremely flexible and easy to experiment on. The specific days and number of days used during training can easily be modified to quickly produce a large range of results.

In its current state, this framework may serve as an early warning system to supplement DA and MCP. Because of the flexibility of deep learning, this model may also be trained to provide early warnings in ocean basins across the world, given that the available data on TCs in these regions remains the same.

To improve the performance of this framework, the quality of spatial data can be improved. This study aimed to create a fully-automated framework, and as a result, there was no manual intervention to filter out any "noise" images of TCs that would have impaired this model's ability to learn features. For example, MODIS imagery of a TC that is almost completely covered by a swath gap would be irreparable by DSGF, since it needed enough significant neighboring pixels to reasonably fill in the swath gap. As a result, this type of image would effectively be unusable after preprocessing, and our model would not be able to learn any significant features from it. With the elimination of swath gaps, the performance of our model would improve substantially. Additionally, a satellite image source that had more frequent and more precise global images would improve performance and make it possible to utilize time series CNNs (*i.e.*, TempCNN) to estimate how the maximum intensity of a TC might change over time. In the future, we would also like to include more sources of spatial data in our dataset such as the vertical height of each TC's cloud formations.

Our model framework's estimations can also be expanded to estimate more TC features, such as pressure, which has an inverse relationship with TC strength. Additionally, our framework will also be modified to include peak intensity classifications using the Saffir-Simpson scale in its forecasts. Estimating the peak categories of TCs will allow our framework to provide more accurate predictions at the cost of precision, and it will allow our framework to produce more types of results. With these improvements made, our framework has the potential to replace current computational methods. Furthermore, we could also create an algorithm using M-Band wavelet-based functional mixed model with time constraints and provide estimations for more metrics such as pressure and precipitation.

Acknowledgements

The authors thank the National Aeronautics and Space Administration (NASA) and the National Oceanic and Atmospheric Administration (NOAA) for making the satellite-based data freely available. The first author wishes to thank Talia Dardis for her professional support and advice during the research process, and acknowledges the resources and facilities provided by Western Connecticut State University (WCSU).

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] NOAA (2022) Hurricane Costs. https://coast.noaa.gov/states/fast-facts/hurricane-costs.html

[2] Klotzbach, P.J., Wood, K.M., Schreck III, C.J., Bowen, S.G., Patricola, C.M. and Bell, M.M. (2022) Trends in Global Tropical Cyclone Activity: 1990-2021. *AGU Journal of Geophysical Research*, **49**, e2021GL095774. https://doi.org/10.1029/2021GL095774

[3] NOAA (2021) Record-Breaking Atlantic Hurricane Season Draws to an End. https://www.noaa.gov/news/record-breaking-atlantic-hurricane-season-draws-to-end

[4] Minott, O. (2021) Looking Back at the 2021 Atlantic Hurricane Season. Bipartisan Policy Center. https://bipartisanpolicy.org/blog/looking-back-at-the-2021-atlantic-hurricane-season

[5] NOAA (2022) Hurricane Ida. https://www.nhc.noaa.gov/data/tcr/AL092021_Ida.pdf

[6] Toepke, S. (2021) Exploring Bot Pervasiveness in Global Cities Using Publicly Available Volunteered Geographic Information. *The 5th International Conference on Geographical Information Systems Theory, Applications and Management*, 23-25 April 2021, 143-153.

[7] Velden, C., Harper, B., Wells, F., Beven II, J.L., Zehr, R., Olander, T., Mayfield, M., Guard, C., Lander, M., Edson, R., Avila, L., Burton, A., Turk, M., Kikuchi, A., Christian, A., Caroff, P. and McCrone, P. (2006) The Dvorak Tropical Cyclone Intensity Estimation Technique—A Satellite-Based Method That Has Endured for over 30 Years. *Bulletin of the American Meteorological Society*, **87**, 1195-1210. https://doi.org/10.1175/BAMS-87-9-1195

[8] Brown, D. and Franklin, J. (2004) Dvorak Tropical Cyclone Wind Speed Biases Determined from Reconnaissance-Based "Best Track" Data (1997-2003). *26th Conference on Hurricanes and Tropical Meteorology*, Miami, 3-7 May 2004, 86-87.

[9] Cossuth, J.H., Knabb, R.D., Brown, D.P. and Hart, R.E. (2013) Tropical Cyclone Formation Guidance Using Pregenesis Dvorak Climatology. Part I: Operational Forecasting and Predictive Potential. *AMS Weather and Forecasting*, **28**, 100-118. https://doi.org/10.1175/WAF-D-12-00073.1

[10] Yamaguchi, M. and Koide, N. (2017) Tropical Cyclone Genesis Guidance Using the Early Stage Dvorak Analysis and Global Ensembles. *AMS Weather and Forecasting*, **32**, 2133-2141. https://doi.org/10.1175/WAF-D-17-0056.1

[11] Matsuoka, D., Nakano, M., Sugiyama, D. and Uchida, S. (2018) Deep Learning Approach for Detecting Tropical Cyclones and Their Precursors in the Simulation by a Cloud-Resolving Global Nonhydrostatic Atmospheric Model. *Progress in Earth and Planetary Science*, **5**, Article No. 80.

[12] DeMaria, M., Knaff, J.A., Knabb, R., Lauer, C., Sampson, C.R. and DeMaria, R.T. (2009) A New Method for Estimating Tropical Cyclone Wind Speed Probabilities. *AMS Weather and Forecasting*, **24**, 1573-1591. https://doi.org/10.1175/2009WAF2222286.1

[13] Herrera, V.M.V., Martell-Dubois, R., Soon, W., Herrera, G.V., Cerdeira-Estrada, S., Zuniga, E. and Rosique-de la Cruz, L. (2022) Predicting Atlantic Hurricanes Using Machine Learning. *Atmosphere*, **13**, Article No. 707. https://doi.org/10.3390/atmos13050707

[14] Su, H., Wu, L., Jiang, J.H., Pai, R., Liu, A., Zhai, A.J., Tavallali, P. and DeMaria, M.

(2020) Applying Satellite Observations of Tropical Cyclone Internal Structures to Rapid Intensification Forecast with Machine Learning. *Geophysical Research Letters*, **47**, e2020GL089102. https://doi.org/10.1029/2020GL089102

[15] Fava, M. (2022) Atlantic Ocean Basin: A Detailed Map. UNESCO, Paris. https://oceanliteracy.unesco.org/atlantic-ocean/

[16] NCEI (2021) Annual 2021 Tropical Cyclones Report. https://www.ncei.noaa.gov/access/monitoring/monthly-report/tropical-cyclones/202113

[17] NHC (2021) 2021 Atlantic Hurricane Season. https://www.nhc.noaa.gov/data/tcr/index.php?season=2021&basin=atl

[18] Chen, S., Cao, E., Koul, A., Ganju, S., Praveen, S. and Kasam, M.A. (2021) Reducing Effects of Swath Gaps in Unsupervised Machine Learning. *Committee on Space Research Machine Learning for Space Sciences Workshop*, *Cross-Disciplinary Workshop on Cloud Computing*.

[19] Liu, Z., Liu, T., Sun, W., Zhao, Y. and Wang, X. (2022) MBand Wavelet-Based Imputation of scRNA-seq Matrix and Multi-View Clustering of Cells. *The FASEB Journal*, **36**. https://doi.org/10.1096/fasebj.2022.36.S1.R5102

[20] Sanjeev, P. and Jayanthi, S. (2008) Image Denoising Using Matched Biorthogonal Wavelets. 6*th Indian Conference on Computer Vision*, *Graphics and Image*, Bhubaneswar, 16-19 December 2008, 25-32.

[21] Stéphane, M. (1999) A Wavelet Tour of Signal Processing. Academic Press, Cambridge. https://doi.org/10.1016/B978-012466606-1/50008-8 https://www.sciencedirect.com/book/9780123743701/a-wavelet-tour-of-signal-processing

[22] Martinez, J.C. (2020) M-Band Wavelet-Based Imputation of scRNA-seq Matrix and Multi-View Clustering of Cells. AIgents. https://aigents.co/data-science-blog/