remote sensing

MDPI

Article

Hybrid CNN-LSTM Deep Learning for Track-Wise GNSS-R Ocean Wind Speed Retrieval

Sima Arabi [1,*], Milad Asgarimehr [1,2], Martin Kada [1] and Jens Wickert [1,2]

[1] Institute of Geodesy and Geoinformation Science, Technische Universität Berlin, 10553 Berlin, Germany; milad.asgarimehr@gfz-potsdam.de (M.A.); martin.kada@tu-berlin.de (M.K.); wickert@gfz-potsdam.de (J.W.)
[2] German Research Centre for Geosciences GFZ, 14473 Potsdam, Germany
* Correspondence: sima.arabi@campus.tu-berlin.de

Abstract: The NASA Cyclone GNSS (CYGNSS) mission provides one Delay Doppler Map (DDM) per second along observational tracks. To account for spatiotemporal correlations within adjacent DDMs in a track, a deep hybrid CNN-LSTM model is proposed for wind speed prediction. The model combines convolutional and pooling layers to extract features from DDMs in one track, which are then processed by LSTM as a sequence of data. This method leads to a test RMSE of 1.84 m/s. The track-wise processing approach outperforms the architectures that process the DMMs individually, namely based on Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN), and a network based solely on fully connected (FC) layers, as well as the official retrieval algorithm of the CYGNSS mission with RMSEs of 1.92 m/s, 1.92 m/s, 1.93 m/s, and 1.90 m/s respectively. Expanding on the CNN-LSTM model, the CNN-LSTM+ model is proposed with additional FC layers parallel with convolutional and pooling layers to process ancillary data. It achieves a notable reduction in test RMSE to 1.49 m/s, demonstrating successful implementation. This highlights the significant potential of track-wise processing of GNSS-R data, capturing spatiotemporal correlations between DDMs along a track. The hybrid deep learning model processing the data sequentially in one track learns these dependencies effectively, improving the accuracy of wind speed predictions.

Keywords: deep learning; CNN-LSTM; GNSS reflectometry; track-wise retrieval; ocean wind speed; CYGNSS

Citation: Arabi, S.; Asgarimehr, M.; Kada, M.; Wickert, J. Hybrid CNN-LSTM Deep Learning for Track-Wise GNSS-R Ocean Wind Speed Retrieval. *Remote Sens.* **2023**, *15*, 4169. https://doi.org/10.3390/rs15174169

Academic Editor: Chung-yen Kuo

Received: 21 June 2023
Revised: 8 August 2023
Accepted: 21 August 2023
Published: 24 August 2023

1. Introduction

Ocean surface wind plays a vital role in the Earth's system, serving as an indicator of ocean–atmosphere interaction. It influences ocean currents, facilitates heat and water vapor transport, impacts marine transportation, generates swells and waves, and influences the ocean ecosystem. However, it can also become a destructive force, leading to hurricanes that pose a threat to residential areas and cause a loss of life and extensive damage. Therefore, understanding and modeling this parameter are crucial for predicting the atmosphere–ocean system and establishing effective warning systems. This task has become increasingly important due to the amplified frequency and severity of tropical storms caused by climate change [1,2].

Global navigation satellite system reflectometry (GNSS-R) is a remote sensing technique that utilizes GNSS signals reflected from the Earth's surface. It enables the acquisition of various Earth parameters through instruments located at ground-based stations, onboard aircraft or airships, and different space platforms [3,4]. The GNSS-R technique offers distinct advantages for ocean wind monitoring, including its cost-effectiveness, lightweight instrumentation, and low power requirements. Furthermore, GNSS-R also provides data during rainfall with higher quality than conventional sensors [5,6]. These advantages allow for the deployment of multi-satellite constellations with high sampling rates, surpassing those of existing ocean surface wind data. The NASA Cyclone GNSS (CYGNSS) Earth Venture mission, launched in December 2016, represents the first mission fully dedicated to

GNSS-R [7]. CYGNSS consists of eight microsatellites and aims to monitor ocean surface winds, particularly in tropical cyclones and hurricanes. By complementing existing ocean wind data, CYGNSS overcomes challenges in hurricane forecasting.

Traditional wind speed retrieval methods primarily use parametric fitting approaches [5,8–10]. Typically, these methods involve conducting regression analysis between the quantities extracted from measured DDMs and a match-up dataset obtained from other instruments or reanalysis model estimates. For CYGNSS, the quantities of interest extracted from the DDMs are usually the normalized bistatic radar cross-section (NBRCS) or the leading edge slope (LES) of a three-delay x five-Doppler bin box, which includes the specular point bin [8]. Subsequently, the derived NBRCS/LES is then fitted to wind speed data using the minimum variance estimator (MVE). However, a limitation of these methods is that incorporating information from the entire DDM, instead of just a limited box, or introducing additional parameters for bias correction can be challenging.

Spaceborne GNSS-R is a novel technique that provides large datasets surpassing those obtained through traditional instruments. Deep learning holds potential as an effective processing tool in this domain. Recent studies have explored the potential of machine learning for GNSS-R ocean wind speed prediction, but many have relied on simple methodologies and architectures. Feedforward neural networks have been used to derive wind speed from TDS-1 and CYGNSS data, achieving root mean square errors (RMSE) of 2.76 m/s and 1.51 m/s, respectively [11,12].

Convolutional neural networks (CNNs) are commonly employed for tasks such as image and video recognition, object detection, classification, and segmentation. They have shown promise in predicting wind speed from CYGNSS delay-Doppler maps (DDMs). As DDMs can be viewed as image-like data, CNNs can extract features through conventional layers and process the hierarchical patterns within them. CNNs have recently been proposed to process DDMs directly and combine them with ancillary information. One study presented an ensemble average model, utilizing multiple trained models on supercomputing servers, achieving an RMSE of 1.36 m/s [13]. Another study reported a CNN-based model with an RMSE of 1.53 m/s [14]. Recently, CYGNSS DDMs have been processed based on transformer networks, resulting an RMSE of 1.43 m/s, and the models offered explainability through attention maps [15].

CYGNSS collects DDMs on board the spacecraft at one-second intervals along one-dimensional observation "tracks". The DDMs within a track exhibit spatial correlation, indicating that processing them on a track-wise basis could potentially enhance prediction accuracy compared to conventional methods that evaluate DDMs individually. It has been shown that the wind speed estimation bias of conventional methods can be reduced by a careful assessment of the DDM-extracted quality on a per-track basis [16]. Consequently, the implementation of a track-wise machine learning processing approach can potentially enhance the accuracy of wind speed prediction using CYGNSS DDMs.

Previous studies have demonstrated CNNs' ability to extract wind speed from individual DDMs [13,14]. However, CNNs can be combined with architectures such as long short-term memory (LSTM) to process CYGNSS data sequence-wise. LSTM benefits from its forget gates and artificial memory, enabling it to learn the spatial dependencies of DDMs within a track. On the other hand, CNNs excel at extracting hierarchical patterns from DDMs and combining complex features obtained through filter operations. In a recent study, wind speed estimations were conducted using a CNN-LSTM network [17]. However, the LSTM component of the network in this study was specifically designed to capture the temporal correlation of wind speed in a particular area, rather than considering DDMs in a track as sequential data. The current study introduced a hybrid CNN-LSTM deep learning model to investigate how the fusion of these two architectures enhances wind speed predictions based on a track-wise processing approach.

To address this question, the study developed a hybrid CNN-LSTM model and evaluated its performance against conventionally used architectures. Section 2 introduces the measurements and datasets. Section 3 provides an overview of the methodology and describes the

models' architecture. In Section 4, the results are presented, and the models' performance is evaluated and discussed. Finally, the concluding remarks are given in Section 5.

2. Dataset

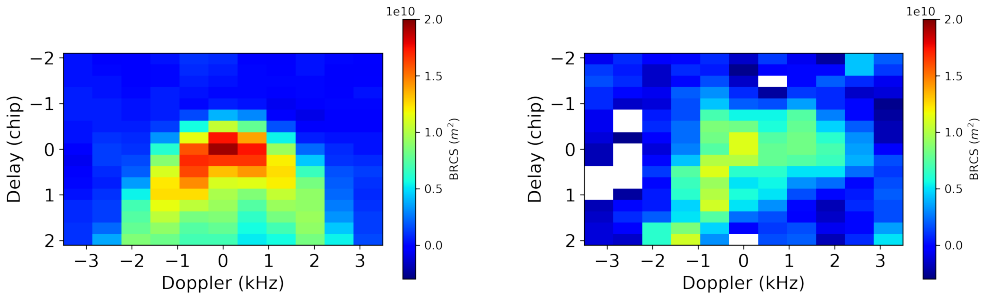
CYGNSS data are provided in three levels. Level 1 data consist of geolocated DDMs. Level 2 data provide time-tagged and geolocated average wind speeds, and Level 3 data offer science data records with average wind speeds gridded at a resolution of 0.2×0.2 degrees. In this study, we used Level 1 data (DDMs) with the objective of converting the data into wind speeds (Level 2).

The dataset was originally available in NetCDF format from NASA's Physical Oceanography Distributed Active Archive Center (PODAAC) (https://podaac.jpl.nasa.gov/dataset/CYGNSS_L1_V2.1 (accessed on 20 March 2023)). The dataset covers a six-month period from January 2018 to June 2018.

As the approach employed was supervised learning, the CYGNSS DDMs needed to be labeled with wind speed information. To label the data, we used ERA5 weather estimates provided by the European Centre for Medium-Range Weather Forecasts (ECMWF) [18]. ERA5 offers hourly estimates of atmospheric, land, and oceanic parameters at a spatial resolution of 30 km. Each CYGNSS DDM corresponds to a specific spatial grid cell, and the closest available ERA5 estimate within the same time interval was considered as the collocated wind speed label. The nominal width of the CYGNSS swath path is recognized to extend up to 25 km, as indicated by the data publisher. However, the spatial resolution of CYGNSS measurements is significantly influenced by ocean roughness conditions. CYGNSS collects one DDM per second, employing a coherent integration time of 1 s. The potential existed for the inclusion of more than one DDM within individual ERA5 spatial segments. We randomly selected one sample from each segment.

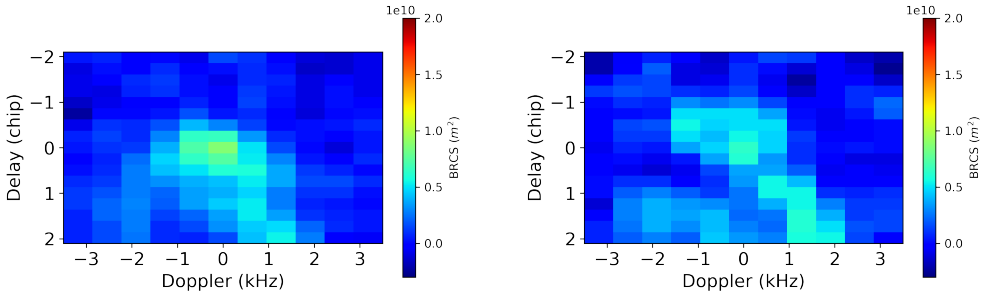
The dataset included the DDMs, which were obtained by cross-correlating the reflected signal with a replica of the signal. The resulting cross-correlation power was then mapped based on delays and Doppler frequency shifts. The power mapped in the DDMs was dependent on the ocean surface state (i.e., surface roughness caused by winds) along with a set of technical parameters, such as the antenna gain of the GPS transmitter and CYGNSS receiver. The DDMs were further processed into DDM BRCS, where technical effects were corrected to a major extent, resulting in a dependency primarily on the ocean surface state. Figure 1 displays examples of CYGNSS DDM BRCS at different wind speeds, illustrating that the BRCS at each DDM bin varied inversely with wind speed. This correlation allowed for the estimation of wind speed using the BRCS DDMs.

Figure 2a depicts the relationship between NBRCS and the collocated wind speeds obtained from ERA5 estimates. The figure showcases the expected trend in NBRCS as a function of wind speed, thereby confirming the accuracy of the collocation process.



(**a**) (**b**)

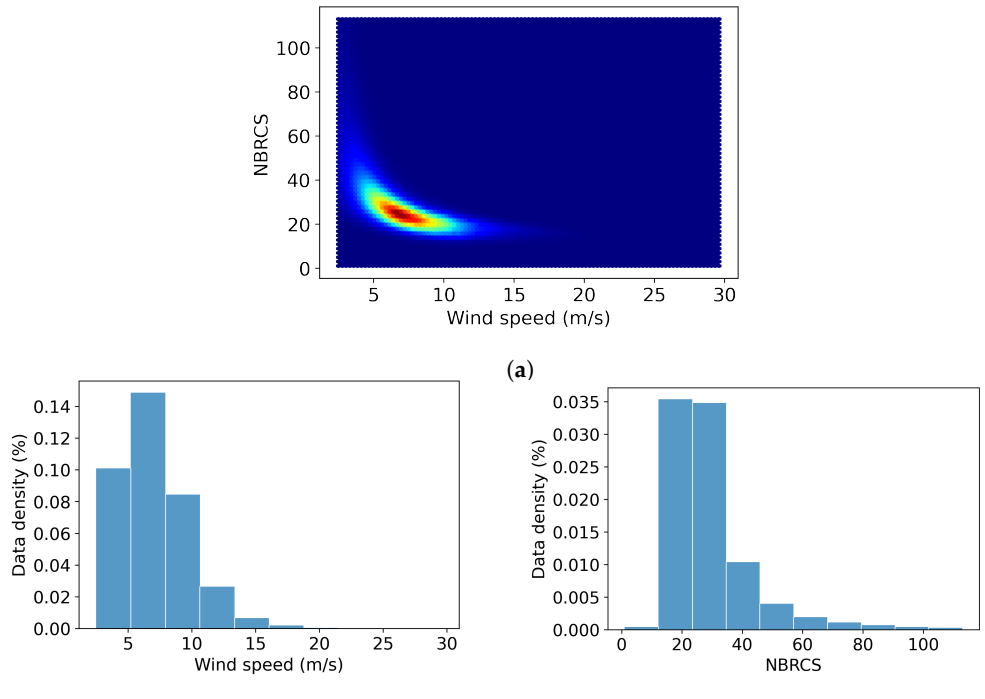
Figure 1. *Cont.*



(**c**) (**d**)

Figure 1. Exemplary BRCS DDMs of CYGNSS at wind speeds below 3 m/s (**a**) and between 5 and 7 m/s (**b**), 11 and 13 m/s, (**c**), and 14 and 16 m/s (**d**).

The data measurements were filtered by a set of quality conditions, as reported in [13]. The outlier measurements were also filtered out at a confidence level of 95%. To be more specific, the samples lying outside the 95% confidence interval of NBRCS were already removed. Figure 2b,c show the histograms of samples.



(**a**)

(**b**) (**c**)

Figure 2. NBRCS versus wind speed (**a**) and histogram of the dataset used in this study, wind speed (**b**) and NBRCS (**c**).

3. Model Architectures

To explore the effectiveness of different architectures and determine whether the sophisticated LSTM-CNN networks outperformed others, several architectures were developed. These architectures included: (i) a simple network comprising fully connected layers, (ii) a CNN-based network, (iii) an LSTM-based network, (iv) an LSTM-CNN network that conducted track-wise processing, and (v) an upgraded version of this network, referred to as LSTM-CNN+ in this study. LSTM-CNN+ incorporated ancillary data through parallel fully connected layers (illustrated in the following). All these architectures were developed and evaluated in this study using the same training and test data to ensure a fair comparison. It must be noted that the specific hyperparameters in each network, including the number of hidden layers and neurons in these layers, the unit size, and the dropout rate,

were determined through trial and error, based on achieving the best performance on the validation dataset. This was the only viable approach and is also common in deep learning model development studies, including those for GNSS-R wind speed retrievals [11–15].

3.1. Fully Connected Layers

The first architecture employed in this study was based on fully connected (FC) layers, which constituted a feed-forward neural network (FFNN). In an FFNN, information flows only in a forward direction, with connections between neurons transmitting outputs solely to subsequent layers, rather than back to previous layers or the same layer. Essentially, the information processed through multiple hidden layers is sequentially passed forward, without any backward flow. In contrast, a recurrent neural network (RNN) allows for interconnections that can form cycles or loops, thus creating an artificial memory [19]. More details on RNNs can be found in Section 3.2.

The network based on fully connected layers consisted of three layers. The DDMs were reshaped from an 11×17 input array into a 187×1 vector for input into the network. Thus, to ensure that the input layer had the correct number of input features, the number of neurons in the first layer was set to 187. The hidden layer of the network comprised 12 neurons, while the last layer output the wind speed with a single neuron. Additionally, a dropout layer with a rate of 0.2 was included between the hidden and output layers. A dropout layer was also used within the following architectures to prevent overfitting. This dropout layer randomly deactivated a portion of the neurons during training, reducing their co-dependency and promoting better generalization [20].

3.2. Long Short-Term Memory

Here, the problem was reformulated such that the delay-Doppler map (DDM) bins were treated as sequential data rather than a feature vector, enabling the RNN to model the correlations between the bins. However, although RNNs have the potential to excel with this processing strategy, they may encounter difficulties when trained on long sequences [19,21]. To address this issue, long short-term memory (LSTM) was proposed. LSTM is a specialized type of RNN with an enhanced recurrent hidden unit to capture long-term dependencies [22–25].

Figure 3 illustrates the architecture of the proposed LSTM for processing the DDMs. The input of the network consisted of the DDMs with bins denoted as x from an 17×11 array of pixels. In this approach, the network treated every DDM as a sequence of 17 vectors, each comprising 11 elements. Consequently, the network could learn the inter-correlation among the pixels of the 17 input sequences of data, with each sequence having a size of 11 elements. Thus, x^k represents the kth input (row of the DDM) in that sequence. The output of the network was a continuous value indicating the wind speed in m/s for the entire data sequence, i.e., an individual DDM. The processing strategy of the LSTM followed a many-to-one approach.

3.3. Convolutional Neural Network

A CNN is a type of deep learning model that is commonly used in tasks such as speech and image classification, as well as video analysis. CNNs are effective in automatically extracting visual features from data. Typically, a CNN is composed of multiple convolutional and pooling layers, and it may also include FC layers. Pooling layers reduce the dimensionality of the data by taking small blocks from the output of the convolutional layers and producing a single output value for each block. Among the different types of pooling, max-pooling, where the maximum value within a block is passed to the next layer, is particularly popular.

CNNs have the ability to automatically extract features from images and learn multiple layers of increasingly complex features. The initial layers of a CNN extract simple features, while subsequent layers extract and detect higher-level features. These features can then be passed to fully connected layers for further processing and generating the final output.

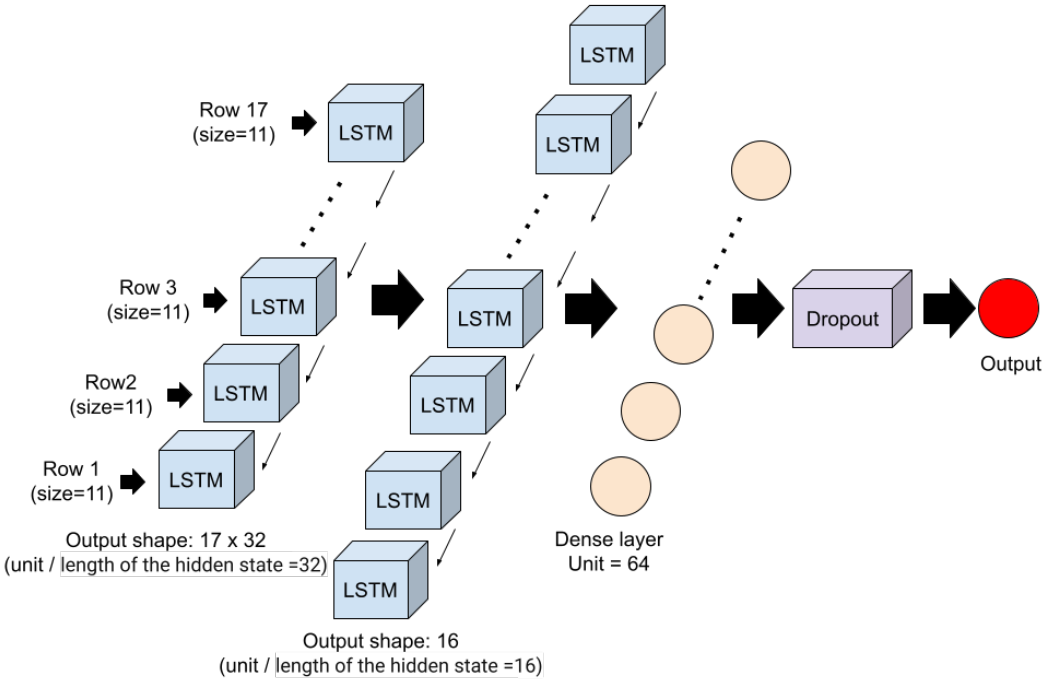


Figure 3. The architecture of the LSTM network used for extracting wind speed from CYGNSS BRCS DDMs consisting of several layers. The first two layers comprised 32 and 16 recurrent units, respectively. These were then followed by a dense layer with 64 units; a dropout layer with a rate of 0.2; and, finally, the output dense layer with a single neuron.

Although CNNs are commonly used for tasks like image and video recognition, object detection, and segmentation, in this case, they were applied to a regression task: predicting the wind speed from CYGNSS DMMs. The DDMs, being two-dimensional arrays, could be treated as image-like data, and their features could be extracted and processed using convolutional layers. The hierarchical patterns present in the DMMs could be leveraged by the network. The overall architecture is depicted in Figure 4.

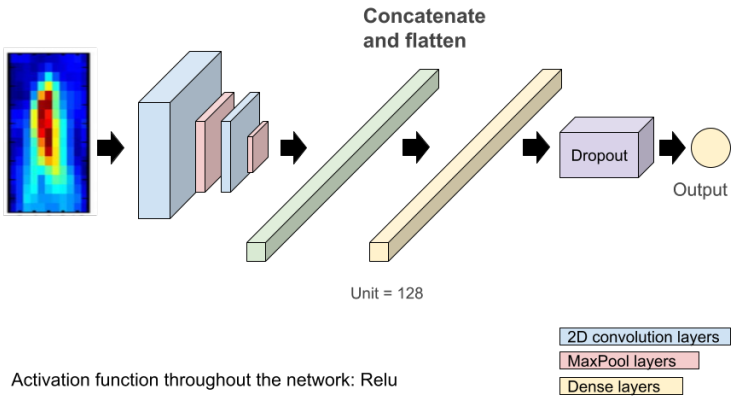


Figure 4. The architecture of the CNN used for extracting wind speed from CYGNSS BRCS DDMs. The kernel size of the max-pooling layer was (2, 2), and the stride was 1.

3.4. Hybrid CNN-LSTM

By combining the unique capabilities of the deep neural networks discussed in the previous subsections, we could leverage the strengths of each model. LSTM benefits from its forget gates and artificial memory, enabling it to learn spatial dependencies within the DDMs in one track. On the other hand, CNNs excel at learning hierarchical patterns and assembling complex features by leveraging simpler patterns identified by the filters. With the goal of improving wind speed predictions, we proposed a hybrid model that combined the LSTM and CNN architectures.

CYGNSS DDMs are collected in one-second intervals along one-dimensional observation tracks onboard the spacecraft. The adjacent DDMs within a track exhibit spatial correlation. Therefore, by processing the DDMs in a "track-wise" manner, we could potentially enhance prediction accuracy compared to traditional methods that evaluate DDMs individually. To achieve this, we proposed the CNN-LSTM architecture, which processed the DDMs as described in the following.

Firstly, the CYGNSS DDM tracks were divided into subtracks consisting of 10 DDMs. Convolutional layers were applied in parallel to extract features from each DDM. These features were then flattened and evaluated in a sequence-wise manner by an LSTM. After passing through two LSTM layers with 63 and 32 recurrent units, respectively, the higher-level features were fed into FC layers. The final FC layer, containing ten neurons, output ten wind speeds corresponding to each DDM within the shortened track. The architecture is visualized in Figure 5. Initially, a 2D convolutional network was defined, comprising Conv2D and MaxPooling2D layers. The Conv2D layer extracted features from DDM snapshots, treating them as images with a single channel (similar to black and white images) and reading a 17 $\times$ 11 pixel DDM. The output of Conv2D was an array with a size of 17 $\times$ 11. Subsequently, the MaxPooling2D layer with a block size of 2 $\times$ 2 consolidated the output, resulting in an 8 $\times$ 5 matrix. This matrix was then flattened, transforming the 8 $\times$ 5 map into a 40-element vector.

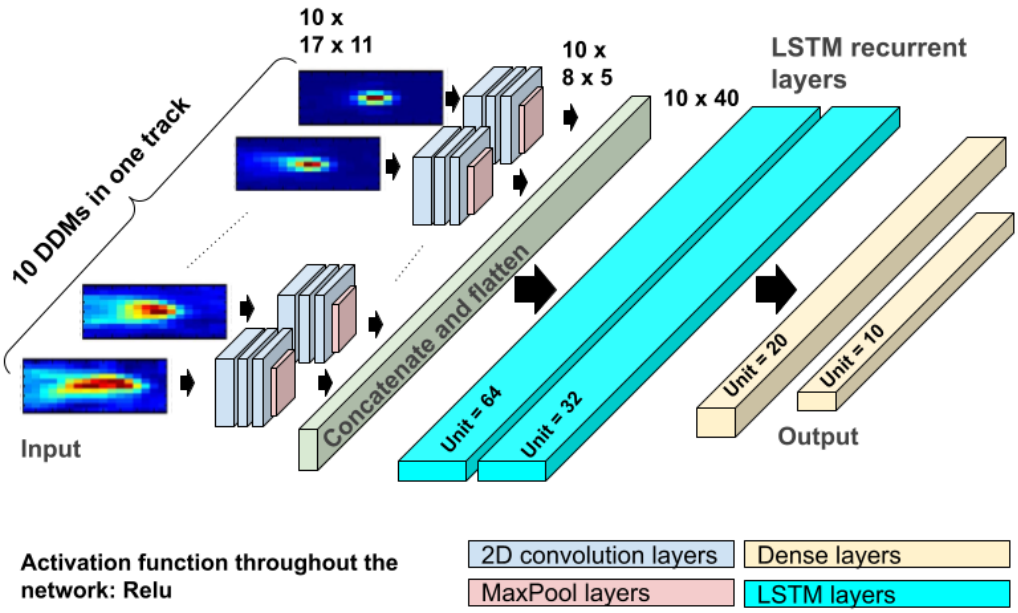


Figure 5. The architecture of the CNN-LSTM for the extraction of wind speed from CYGNSS BRCS DDMs. The kernel size of the max-pooling layer was (2, 2), and the stride was 1.

This process was repeated for each of the ten DDMs within a shortened track, generating a 10 $\times$ 40 array as the output of the CNN segment. The CNN segment was applied to each input DDM, and the output of each DDM was transferred to the LSTM as a single time

step. Thus, the LSTM's task was categorized as "many-to-many", involving the prediction of ten wind speeds from ten DDMs, each of which was initially preprocessed by the CNN.

It is important to note that in this approach, the CNN and LSTM were treated as independent steps, as implemented in [26]. This was distinct from the CNN-LSTM proposed in [27], where matrix multiplication was replaced with convolution operations at each gate within the LSTM cell.

3.5. Enhanced Hybrid CNN-LSTM (CNN-LSTM+)

The results presented in the following section demonstrate that the CNN-LSTM architecture outperformed other architectures in wind speed prediction. To further enhance the performance, an additional pipeline was introduced, which combined ancillary information and corrected associated biases. One example of such a bias was the effect of a higher incidence angle, which led to a longer signal path and consequently a lower signal power. The received power was also influenced by the power of the signals transmitted from the GPS satellite, known as equivalent isotropically radiated power (EIRP).

While the BRCS DMMs used in this study were expected to account for these effects, it has been shown that incorporating these input variables could enhance wind speed predictions using machine learning techniques [11]. One possible explanation is that the theoretical models employed for computing the BRCS may not completely remove technical effects, such as those associated with the instrumental properties, as they rely on simplifying assumptions that may not be valid for all real-world scenarios. Additionally, traditional models lack validation under different conditions, given that spaceborne GNSS-R is a relatively new remote sensing technique.

Therefore, leveraging deep learning to model and mitigate these effects could be a viable solution. By incorporating ancillary information and correcting associated biases through the additional pipeline, the accuracy of wind speed predictions could be further improved. It has been shown that incorporating manually extracted features into CNNs can improve prediction accuracy. Handcrafted features are often added to networks for tasks like image classification [28]. It has also been shown that adding DDM-extracted features to CNNs can enhance the accuracy of wind speed predictions [13].

Figure 6 presents the correlogram, illustrating the relationships between each pair of the four additional variables (NBRCS, LES, incidence angle, and GPS EIRP) and wind speed. The diagonal of the correlogram represents the distribution of each variable, visualized through histogram plots. The ancillary data depicted in this figure were chosen based on theoretical knowledge. These data, rooted in physical understanding, either exhibited strong correlations with wind speed, aiding the network in achieving improved predictions (NBRCS and LES), or represent technical parameters known to introduce biases (incidence angle and GPS EIRP). Although the figure illustrates some dependencies clearly, the impact of others may not be entirely evident. Nevertheless, upon incorporating these parameters into the network, they led to a significant improvement in performance.

The architecture of CNN-LSTM+ is shown in Figure 7. In this architecture, FC layers were added to process the four ancillary parameters simultaneously for each of the 10 DDMs. These features were then concatenated with the flattened features obtained from the convolutional and max-pooling layers, which processed the BRCS DDMs. The concatenated features from both the main and ancillary data were treated as a sequence and processed by LSTM layers. Finally, the FC layers output 10 wind speed variables, corresponding to each DDM within a track. By incorporating both the main and ancillary data, CNN-LSTM+ aimed to improve the accuracy of wind speed predictions.

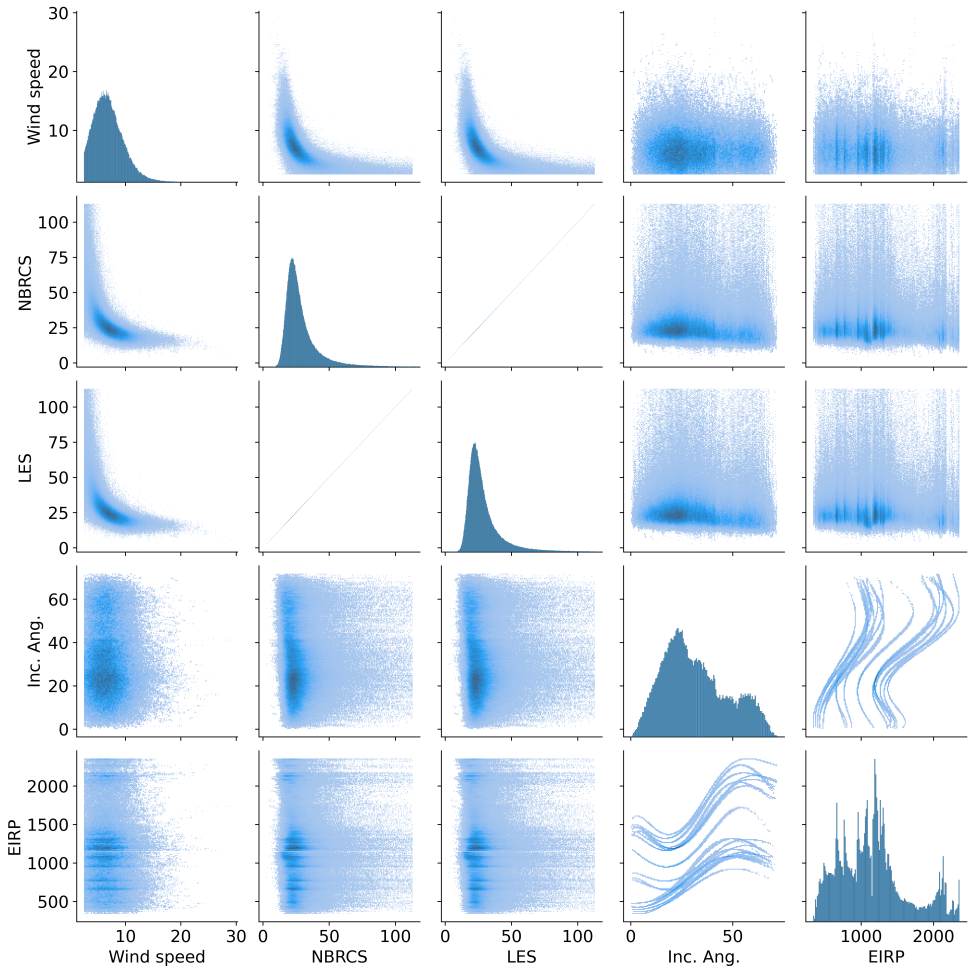


Figure 6. Correlogram of the ancillary data, NBRCS (unitless), LES (unitless), incidence angle (degrees), and EIRP (Watts), considered as additional input variables for CNN-LSTM+, and wind speed (m/s).

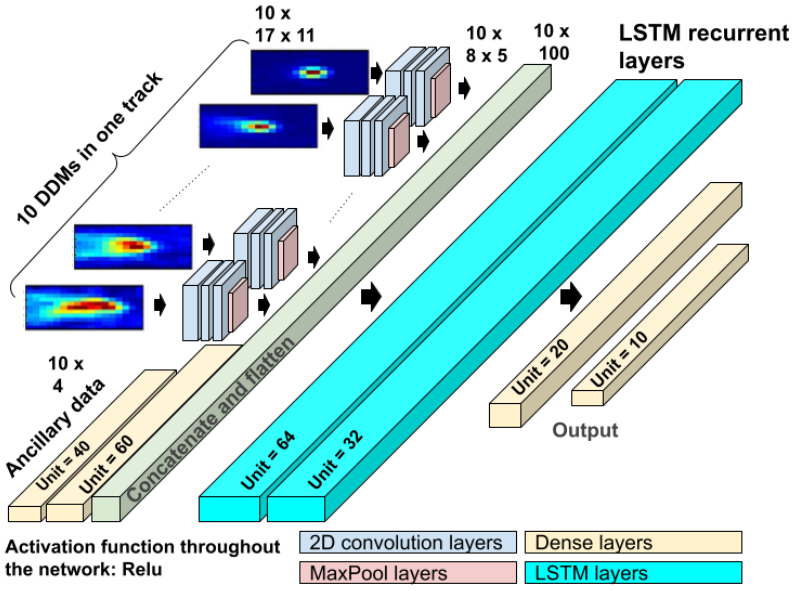


Figure 7. The architecture of CNN-LSTM+ for the extraction of wind speed from CYGNSS BRCS DDMs. The kernel size of the max-pooling layer was (2, 2), and the stride was 1.

4. Implementation and Results

4.1. Training

For the implementation of deep learning in CYGNSS data processing, the Python environment and the Keras library were used. Keras is a well-known open-source Python library that enables the creation and validation of deep learning models.

The rectified linear unit (ReLU) activation function was used throughout all the proposed networks. The ReLU function is defined as $f(x) = \max(0, x)$, where x represents the input to a neuron. This choice was made to overcome the vanishing gradient problem associated with sigmoid and hyperbolic tangent activation functions, leading to faster learning and improved performance. For weight optimization, the Adam algorithm [29] was employed. Adam is an optimization algorithm that serves as an alternative to classical stochastic gradient descent, enabling the iterative updating of weights towards their optimal values.

The dataset was divided into three groups: training (55%), validation (20%), and test (25%). The datasets used for training, validation, and testing were collected from different time periods. This approach was adopted to ensure that the test data remained independent and unbiased, allowing for a fair evaluation of the trained networks. The network was trained using the training dataset, with careful consideration given to the number of iterations for weight updates. Overfitting to the training dataset was prevented by implementing "early stopping". This technique automatically halted the training process when no improvement in model performance on the validation dataset was observed. Specifically, the loss value on the validation dataset was monitored, and training was stopped if it did not decrease after a certain number of iterations, referred to as the "patience". In this implementation, a patience of 10 iterations was set.

Before training, the dataset underwent normalization. This process scaled each input value to a range of 0 to 1. The MinMaxScaler preprocessing class from the scikit-learn library was used for dataset normalization. The BRCS DDMs served as the input attributes, while their corresponding labels, ERA5 wind speed, represented the output attributes. The input for each of the five networks was aligned to conform to the specified architectures.

The FC layer model was trained for 40 epochs, achieving a mean squared error (MSE) loss value of 3.77 $(m/s)^2$. The validation loss was slightly lower at 3.75 $(m/s)^2$. This LSTM model was trained after 87 epochs, with comparable loss values of 3.77 and 3.75 $(m/s)^2$ for the training and test data, respectively. The LSTM model required more training time compared to the FC layer model due to its more complex architecture.

The CNN model was trained after 94 epochs. The final MSE loss values for the training and validation data were 3.86 and 3.78 $(m/s)^2$, respectively. The CNN-LSTM model was trained after 54 epochs, achieving a training loss of 3.03 $(m/s)^2$ and a validation loss of 2.29 $(m/s)^2$. The CNN-LSTM model demonstrated a significant improvement in terms of training and validation loss compared to the FC layer, LSTM, and CNN architectures.

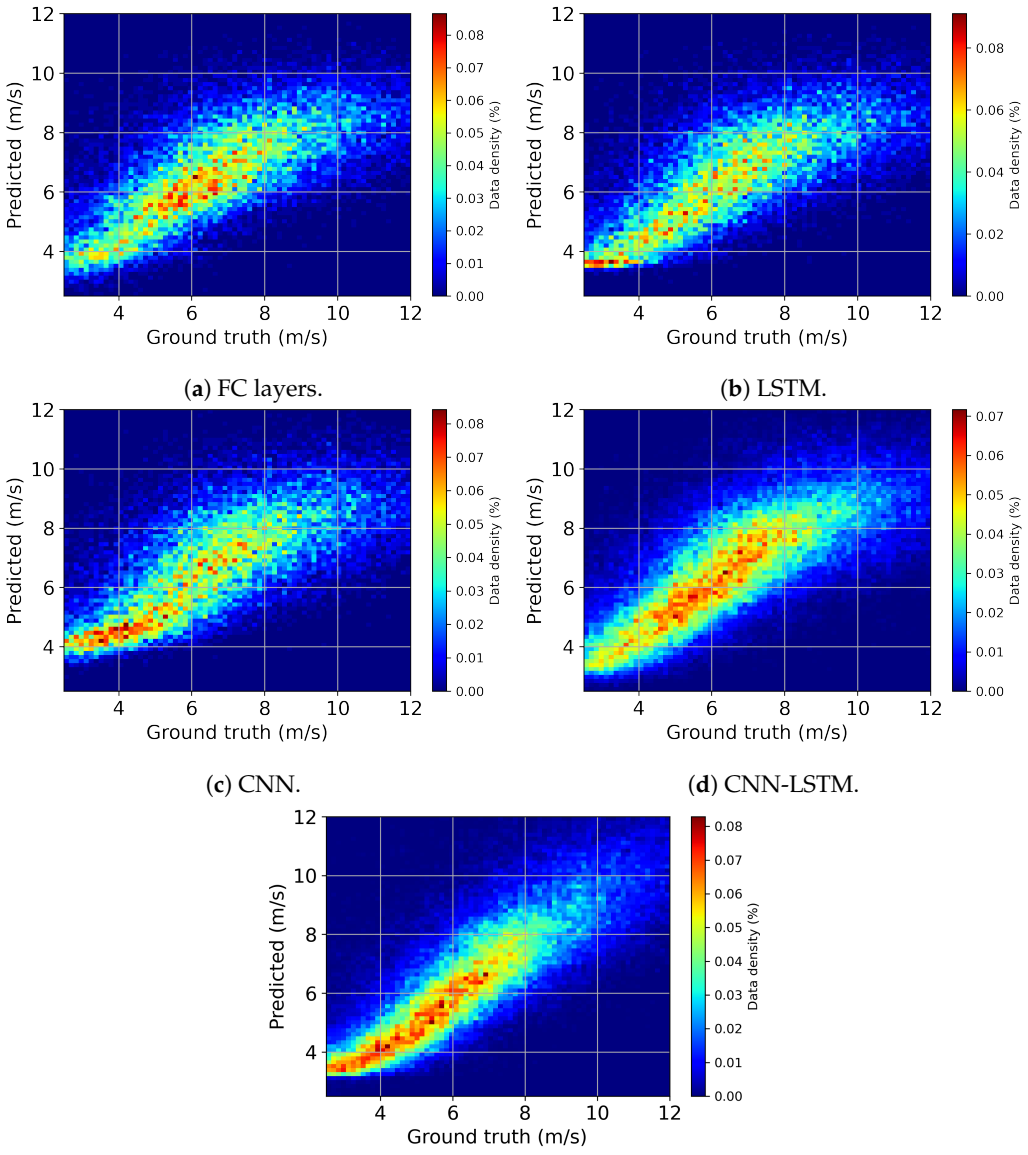
Finally, the CNN-LSTM+ model was trained after 50 epochs, resulting in a training loss of 2.22 $(m/s)^2$ and a validation loss of 2.26 $(m/s)^2$. These values indicated a substantial improvement in performance over the training and validation datasets compared to the previous models.

4.2. Models' Performance

To assess the performance of the models, wind speed datasets were generated using each of the five deep learning networks over the test data. The test data were not used during the model development, either directly (similar to the training data) or indirectly (similar to the validation data). This allowed the evaluation of the networks' performance on unseen data and their generalization capability.

Figure 8 illustrates the results of the deep learning methodologies on a global scale. The log-density plots depict the winds predicted by each of the five models compared to the ground truth, which comprised the ERA5 winds, for the test dataset. The figure also provides the RMSE and bias values for each network. The wind retrievals from all networks

exhibited a nearly symmetric distribution centered around the 1:1 line within the range of 4 to 8 m/s. However, at low wind speeds, the networks tended to overestimate the winds. This behavior was not specific to the retrieval methods but rather a common issue across different methodologies, which could be explained by the characteristics of the observables. GNSS-R measurements, for example, show limited sensitivity to winds below 2.5 m/s, and simulations have confirmed this limitation in predicting very low wind speeds [30]. Nonetheless, this overestimation of low winds is not a significant disadvantage of GNSS-R measurements and is also observed in other radar data. Similar overestimations of low winds have been reported in other studies regardless of the methodology employed [5,13,31].



(**a**) FC layers. (**b**) LSTM.

(**c**) CNN. (**d**) CNN-LSTM.

(**e**) CNN-LSTM+.

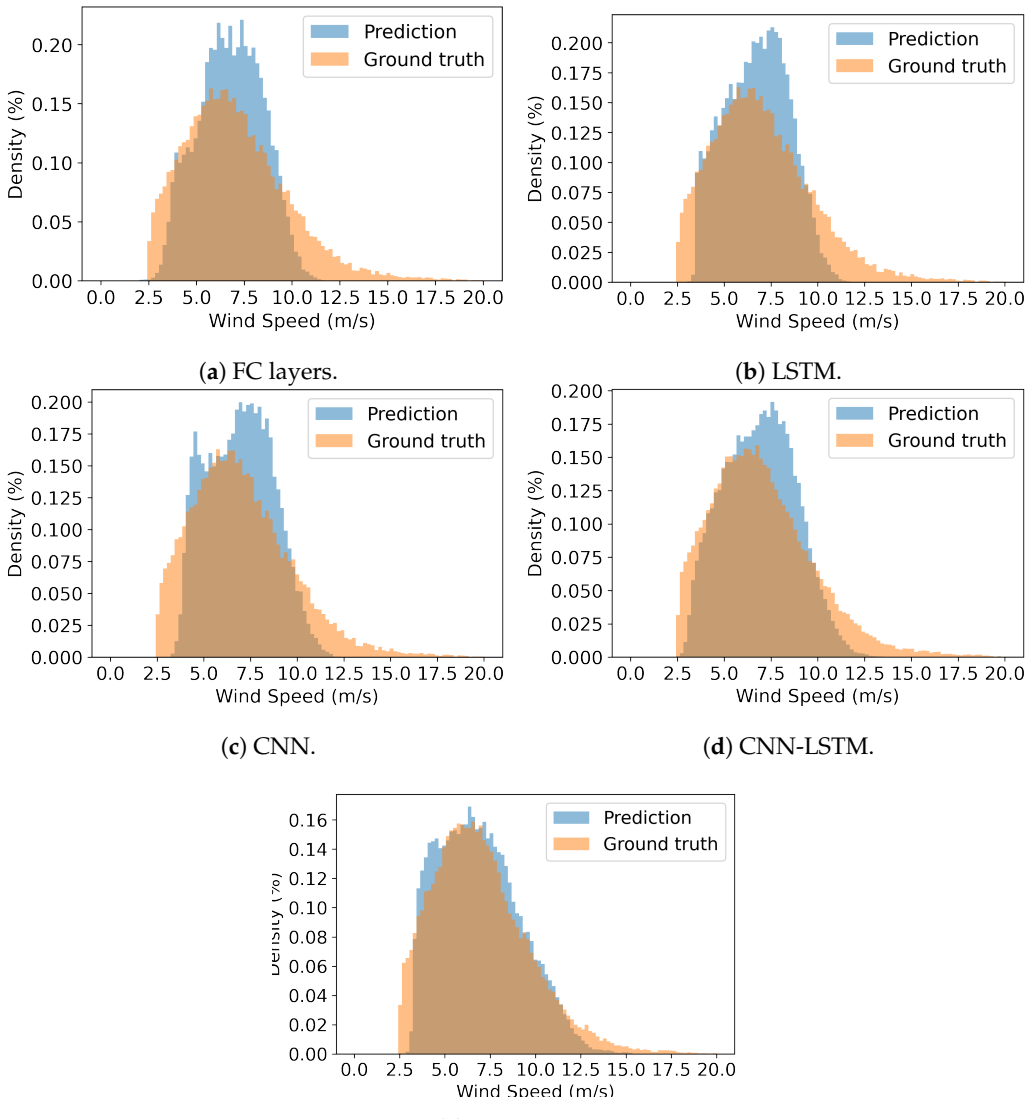
Figure 8. Predicted wind speeds over the test data versus ERA5 estimates. (**a**) FC layers: RMSE = 1.93 m/s, bias = −0.24 m/s. (**b**) LSTM: RMSE = 1.92 m/s, bias = −0.28 m/s. (**c**) CNN: RMSE = 1.92 m/s, bias = −0.08 m/s. (**d**) CNN-LSTM: RMSE = 1.84 m/s, bias = −0.12 m/s. (**e**) CNN-LSTM+: RMSE = 1.49 m/s, bias = −0.08 m/s.

On the other hand, at high wind speeds, some of the networks tended to underestimate the winds. This behavior was expected as the DDM observables, such as NBRCS, exhibited reduced sensitivity to changes in wind speed. Underestimation at high wind speeds has also been observed in winds retrieved by the official CYGNSS algorithm [31]. This issue is

common in traditional radar measurements, which also experience degraded performance at high wind speeds. However, the magnitude of this bias at high wind speeds differed among the deep learning models. As shown, the CNN-LSTM+ model maintained the symmetry of the log-density plot even at higher wind speeds, centered around the 1:1 line. The superiority of this network may stem from its ability to better learn trends at high wind speeds and overcome the "tendency to the mean" issue discussed in Section 4.3.

The log-density plots of the networks over the test data demonstrate how the performance improved with more advanced architectures. The predicted winds exhibited greater concentration around the 1:1 line for the CNN-LSTM and CNN-LSTM+ models. Among them, the CNN-LSTM+ model stood out as the best model for obtaining wind speed, with an RMSE of 1.49 m/s and a bias of −0.08 m/s. Overall, all the networks showed acceptable accuracy, with an RMSE below 2 m/s.

The distributions of the wind speeds predicted by the five networks, compared to the ground-truth ERA5 wind estimates, are depicted in Figure 9. Overall, the wind speed distributions exhibited a distortion characterized by an increased concentration of predictions around the mean wind speed value. However, the extent of this distortion varied among the different networks. The networks based on FC layers, LSTM, and CNN (Figure 9a, Figure 9b and Figure 9c, respectively) displayed a more pronounced accumulation of wind speed predictions around the mean.



(**a**) FC layers.

(**b**) LSTM.

(**c**) CNN.

(**d**) CNN-LSTM.

(**e**) CNN-LSTM+.

Figure 9. Distributions of predicted wind speeds versus that of ERA5 estimates.

Similarly, the CNN-LSTM network (Figure 9d) also exhibited a distortion, although it showed a slight improvement compared to the previous models. In Figure 9e, the distribution of wind speeds predicted by the CNN-LSTM+ network is compared to the true wind speeds. It demonstrated a much closer fit to the true wind speed distribution, indicating a significant improvement compared to the previous deep learning architectures. Further discussion on this improvement is provided in Section 4.3.

The CNN-LSTM+ architecture, being the best-performing model in this study, was further evaluated and visualized in terms of its predicted wind speeds. Figure 10 presents the predicted wind speeds along two exemplary tracks, as well as the corresponding BRCS DDMs. Generally, consistency was observed between the predicted wind speeds and the ERA5 wind speeds.

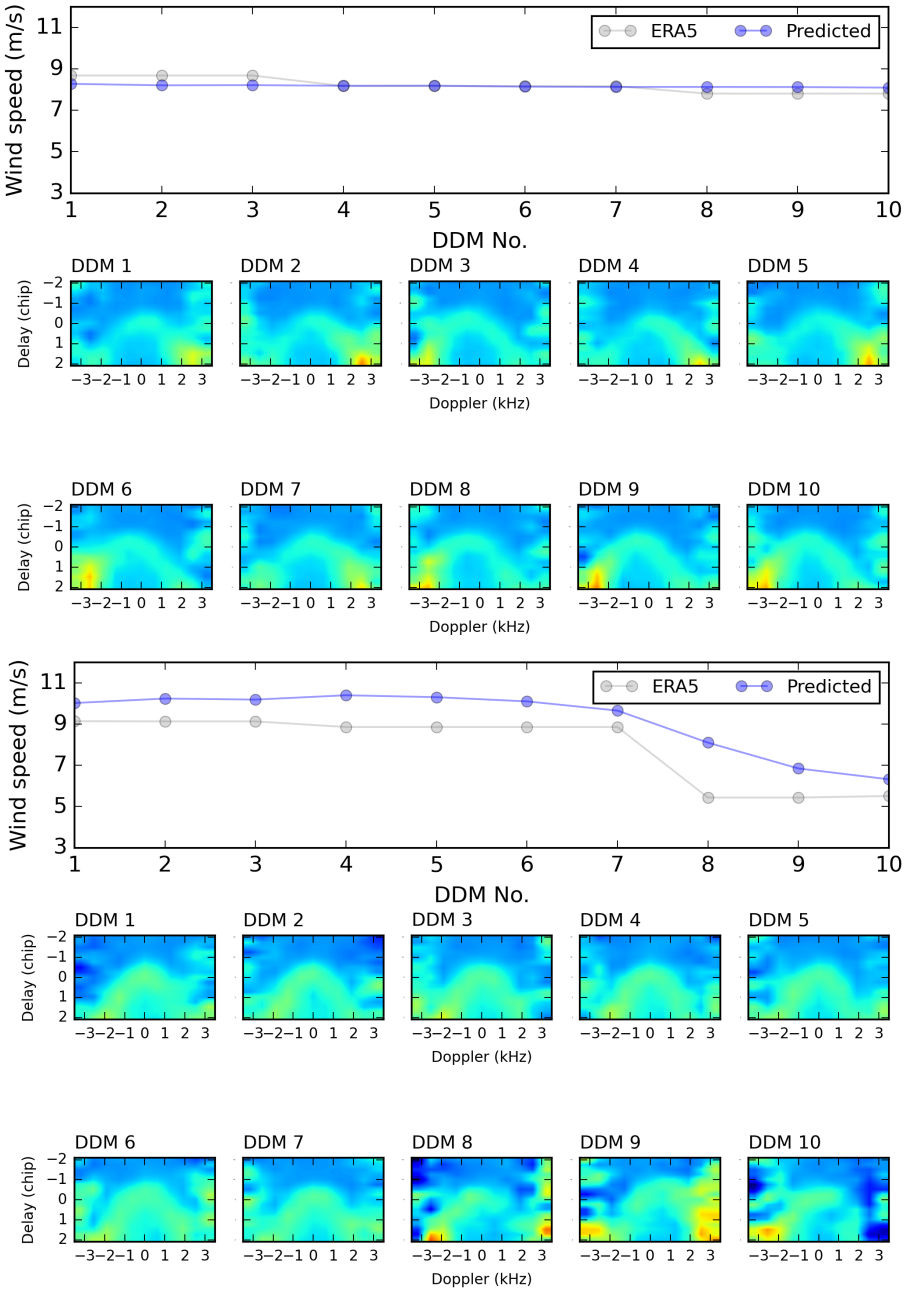


Figure 10. Wind speed profiles predicted by CNN-LSTM+ in two exemplary tracks, along with the corresponding BRCS DDMs.

To assess spatial trends and performance variability, Figure 11 displays the global map of the average wind speed, as well as the RMSE and bias of the CNN-LSTM+. There was notable agreement between the global average maps of ERA5 and the wind speeds predicted by the CNN-LSTM+. The RMSE and bias values remained within acceptable ranges worldwide, although the RMSE tended to be slightly larger in regions with higher latitudes where the average wind speed is higher. These findings are discussed in more detail in Section 4.3.

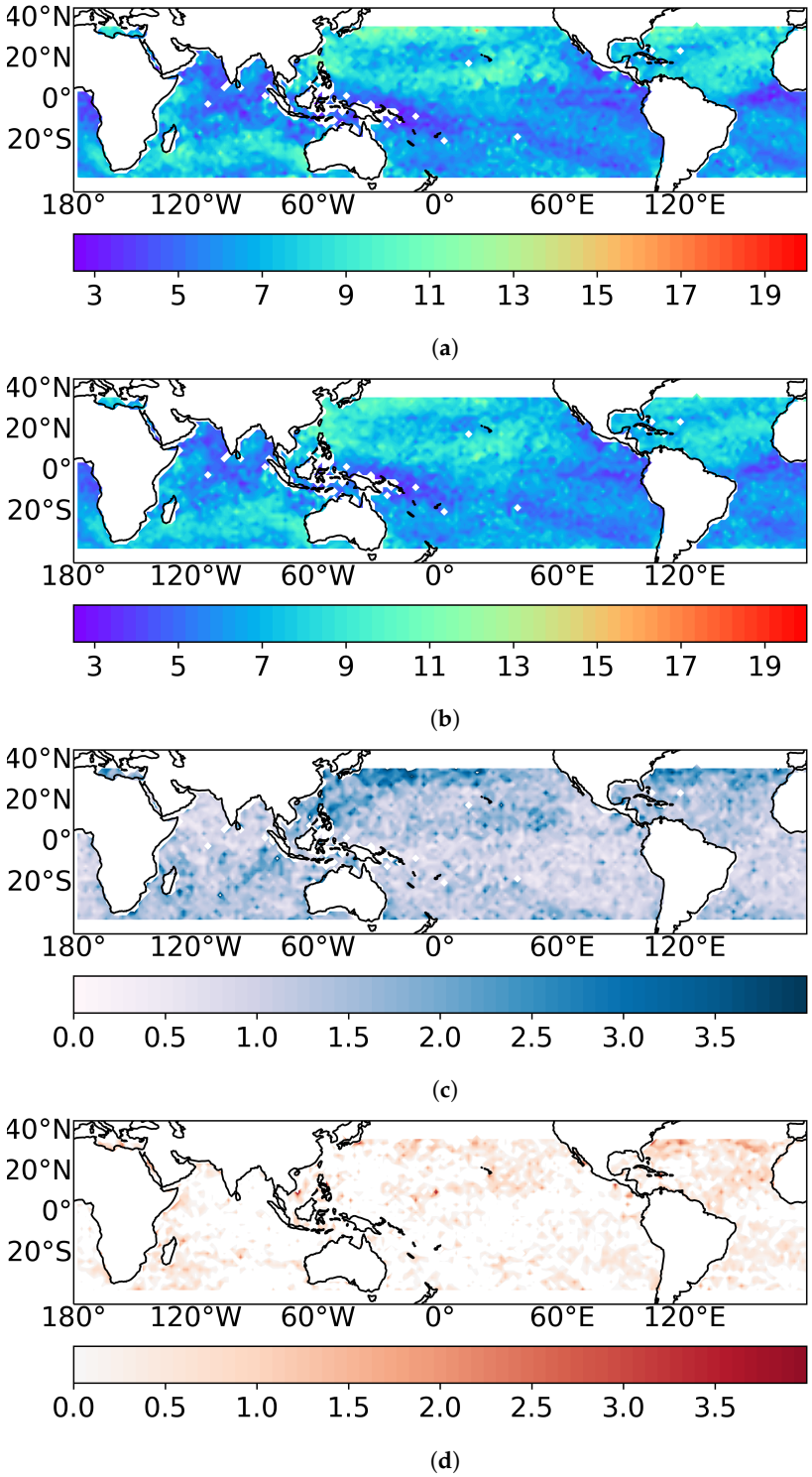


Figure 11. The global map of average wind speed from ERA5 (**a**), and that predicted by CNN-LSTM+ (**b**), as well as the RMSE (**c**) and bias (**d**). The units in all the maps are m/s.

As a benchmark for evaluating the proposed deep learning models, we compared the results to wind speeds retrieved using the official operational MVE method [8,32], which is used to generate Level 3 CYGNSS data products. The traditional retrieval algorithm follows a time-averaging procedure [33] to smooth out the wind speeds, thereby improving their accuracy but sacrificing spatial resolution. It is important to note that no averaging or smoothing was applied to the results of the deep learning models in this study, which should be taken into consideration when comparing the outcomes.

The RMSE and bias of the CYGNSS baseline algorithm, evaluated over a dataset with the same population and temporal coverage as the test dataset used for the deep learning architecture evaluation, were 1.90 m/s and 0.20 m/s, respectively. The RMSEs of the FC layers, LSTM, and CNN models were slightly higher compared to the MVE, whereas the CNN-LSTM and CNN-LSTM+ models outperformed the baseline algorithm. CNN-LSTM+ achieved a 22% reduction in the RMSE compared to the MVE. The bias of CNN-LSTM+ was also smaller than that of the MVE, with values of -0.08 m/s and 0.20 m/s, respectively. Finally, Table 1 summarizes the performance of the deep learning architectures and the MVE algorithm.

Table 1. The performance of the trained deep learning models and the operational retrieval algorithm, MVE.

Architecture	RMSE (m/s)	Bias (m/s)	Training Epochs	Training MSE $(m/s)^2$	Validation MSE $(m/s)^2$
FC layers	1.93	−0.24	40	3.89	3.79
LSTM	1.92	−0.28	87	3.77	3.75
CNN	1.92	−0.08	94	3.86	3.78
CNN-LSTM	1.84	−0.12	54	3.03	3.29
CNN-LSTM+	1.49	−0.08	50	2.22	2.26
MVE	1.90	0.20	-	-	-

4.3. Summary and Discussion

The FC layers model was the simplest architecture that could be used. Due to this simplicity, the model needed fewer epochs compared to other architectures; however, the training and validation loss, as well as the test RMSE, were the highest among the models. Comparing the training time of the FC layers and LSTM models, it was seen that LSTM required a longer training time, as shown by the higher number of training epochs. However, LSTM resulted in slightly improved training loss, validation loss, and test RMSE values. LSTM, with its artificial memory, was able to capture the dependencies of the DMM bins on each other and, as a result, predict the wind speed based not only on the absolute values of the DDM bins, but also on their states relative to each other (the patterns in the DDMs).

The CNNs were able to extract the visual patterns through their convolutional and pooling layers. However, it was shown that they required 94 epochs for training and were the slowest of the architectures. Although the training and validation loss were slightly better than those of LSTM, no improvement in the test RMSE was shown. The bias was smaller, at −0.08 m/s.

The hybrid CNN-LSTM model showed an improvement in training and validation loss and the test RMSE, and it converged to the optimal weights more quickly compared to CNN and LSTM after 54 epochs, although the network's architecture was more complicated. CNN-LSTM was able to better learn the patterns in DDMs and output a more accurate wind speed.

Within CNN-LSTM+, the introduction of NBRCS and LES could significantly improve the performance of the network. Additionally, the technical parameters of the incidence angle and GPS EIRP were considered as inputs so that the network could correct the associated effects. Therefore, CNN-LSTM+ resulted in significantly lower values

of training and validation loss and the best test RMSE and bias values of 1.49 m/s and −0.8 m/s, respectively.

The distribution of the winds predicted by FC layers, LSTM, and CNN-LSTM showed distortion with respect to that of the true winds. This issue could have been mainly due to the "regression to the mean" issue, which is a common problem in deep learning using unbalanced data. The loss values used in regression, such as the MSE, are normally values averaged over all predicted values. When most of the data population was concentrated at the mean value, i.e., 6–7 m/s, the network tended to a conservative prediction around this value if it was not able to learn the trends at extreme ranges. With this strategy, the loss value, i.e., the MSE in this study, could still be close to its optimal value, although the extreme predictions were smoothed closer to the mean. This issue was resolved by CNN-LSTM+ to an acceptable level, showing a much better fit between the distribution of the predicted and true wind speeds, see Figure 9. Under high winds, the DDM observables, NBRCS, and LES were less responsive to wind speed, see Figure 2a, and errors could cause large biases in this range. CNN-LSTM+, processing all the DDMs in addition to NBRC and LES, and learning the effects of the incidence angle and GPS EIRP could improve the predictions under higher winds. As a result, the predicted distribution fit the true distribution better.

The wind speeds derived from CNN-LSTM+ represented the best data and are further evaluated in Figure 11. The maps show consistency between the spatial patterns of the predicted and ERA5 average wind speeds, which further proves the accuracy of the product. As expected, the RMSE was larger when the average wind speed was higher, and a larger bias was also seen.

The best architecture in this study, CNN-LSTM+, with a test RMSE of 1.49 m/s, outperformed the currently operational CYGNSS retrieval algorithm, MVE, with an RMSE of 1.90 m/s. The operational methods for wind speed retrievals were based on the NBRCS and LES extracted from a limited window centered at the DDM bin with the maximum value. These parameters were fitted to matching data based on the MVE, whereas CNN-LSTM+ processed all the DMMs throughout all delay and Doppler frequency shift bins in addition to NBRCS and LES. CNN-LSTM+, which also took into account ancillary data, was able to learn their associated effects, which are not easily described using theoretical models. As a result, CNN-LSTM+ offered significantly better performance.

5. Conclusions

Spaceborne GNSS-R exploiting reflected GNSS signals for remote sensing applications offers a wide range of possibilities for monitoring Earth parameters. One prominent application is the monitoring of ocean wind speeds, which can complement existing datasets and improve hurricane modeling and forecasts.

With the availability of large GNSS-R datasets from spaceborne missions, similar to other remote sensing techniques like optical Earth observations, machine learning methods can be employed to process GNSS-R data and generate higher-quality data products. This study focused on implementing deep learning for spaceborne GNSS-R to retrieve global ocean wind speeds. Five different deep learning architectures were developed and evaluated, and the results demonstrated that these models could achieve acceptable accuracy with RMSE values below 2 m/s. More advanced architectures, such as LSTM and CNN, outperformed a network based solely on FC layers, with RMSEs of 1.92 m/s, 1.92 m/s, and 1.93 m/s, respectively.

The CYGNSS mission, which generates one DDM every second along its swath, or track, provided the dataset for this study. To account for the spatiotemporal correlations in the adjacent DDMs in one track, a deep hybrid CNN-LSTM model was developed. It combined convolutional and pooling layers to extract features from a sequence of ten DDMs on one track, which were then processed by LSTM. The CNN-LSTM model, with a test RMSE of 1.84 m/s, achieved an improvement compared to the other networks and the current official retrieval algorithm of the CYGNSS mission, with an RMSE of 1.90 m/s.

Building upon CNN-LSTM, the CNN-LSTM+ model was proposed, incorporating parallel FC layers with convolutional and pooling layers to process additional ancillary data, including NBRCS, LES, incidence angle, and GPS EIRP. This improved network effectively learned the patterns dependent on wind speed by considering higher-level features and potentially correcting biases associated with the incidence angle and GPS EIRP. The CNN-LSTM+ model achieved a notable reduction in the test RMSE to 1.49 m/s, demonstrating successful implementation. The distribution of predicted wind speeds aligned better with the true wind speeds, and the global average wind speed generated by CNN-LSTM+ was consistent with the ERA5 estimates.

This study highlighted the significant potential of the track-wise processing of GNSS-R data as an alternative to conventional methodologies that rely on the individual processing of DDMs. By considering the spatiotemporal correlations between DDMs along a track, the CNN-LSTM model developed in this study effectively captured these dependencies and improved the accuracy of wind speed predictions. By taking into account the inherent correlations between DDMs along a track, the model was able to extract valuable information from the sequence of DDMs and better capture the dynamics of the ocean surface, resulting in enhanced wind speed predictions.

The ConvLSTM+ model proposed in this study has versatile applications in GNSS-R. Its track-wise processing capability enables it to learn the spatiotemporal dependencies of DDMs in one track, making it advantageous for ocean ice studies using data from polar missions.

Deep-learning-based data products, like that presented in this study, have the potential for integration into Earth system modeling and weather forecasting through assimilation. However, deep learning can be used for predictions, formulating end-to-end trainable models to forecast future ocean states based on available GNSS-R and other instrument data.

Although applied to CYGNSS data, the methodology can be extended to process data from upcoming GNSS-R missions like ESA HydroGNSS and PRETTY, planned for launch in 2024 and 2023, respectively.

Author Contributions: Conceptualization, M.A.; methodology, S.A., M.A. and M.K.; software, S.A.; validation, S.A.; formal analysis, S.A.; investigation, S.A.; data curation, S.A. and M.A.; writing—original draft preparation, S.A.; writing—review and editing, M.A.; visualization, S.A.; supervision, M.K., J.W. and M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was financially supported by Technische Universität Berlin and German Research Centre for Geosicences and received no external funding.

Data Availability Statement: All the data used in this study are publicly available as detailed in Section 2.

Acknowledgments: We express our gratitude to the scientific teams affiliated with the CYGNSS mission at NASA and the University of Michigan, as well as the European Centre for Medium-Range Weather Forecasts (ECMWF) for providing public free access to the datasets used in this study.

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

References

1. Woodward, A.J.; Samet, J.M. Climate change, hurricanes, and health. *Am. J. Public Health* **2018**, *108*, 33–35. [CrossRef]
2. Pachauri, R.; Meyer, L.; Plattner, G.; Stocker, T. *Synthesis Report. Contribution of Working Groups I, II and III to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*; Intergovernmental Panel on Climate Change: Geneva, Switzerland, 2014.
3. Zavorotny, V.U.; Gleason, S.; Cardellach, E.; Camps, A. Tutorial on remote sensing using GNSS bistatic radar of opportunity. *IEEE Geosci. Remote Sens. Mag.* **2014**, *2*, 8–45. [CrossRef]
4. Carreno-Luengo, H.; Camps, A.; Ruf, C.; Floury, N.; Martin-Neira, M.; Wang, T.; Khalsa, S.J.; Clarizia, M.P.; Reynolds, J.; Johnson, J.; et al. The IEEE-SA working group on spaceborne GNSS-R: Scene study. *IEEE Access* **2021**, *9*, 89906–89933. [CrossRef]
5. Asgarimehr, M.; Wickert, J.; Reich, S. TDS-1 GNSS reflectometry: Development and validation of forward scattering winds. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2018**, *11*, 4534–4541. [CrossRef]

6. Asgarimehr, M.; Wickert, J.; Reich, S. Evaluating impact of rain attenuation on space-borne GNSS reflectometry wind speeds. *Remote Sens.* **2019**, *11*, 1048. [CrossRef]
7. Ruf, C.S.; Chew, C.; Lang, T.; Morris, M.G.; Nave, K.; Ridley, A.; Balasubramaniam, R. A new paradigm in earth environmental monitoring with the cygnss small satellite constellation. *Sci. Rep.* **2018**, *8*, 8782. [CrossRef]
8. Clarizia, M.P.; Ruf, C.S.; Jales, P.; Gommenginger, C. Spaceborne GNSS-R Minimum Variance Wind Speed Estimator. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 6829–6843. [CrossRef]
9. Garrison, J.; Komjathy, A.; Zavorotny, V.; Katzberg, S. Wind speed measurement using forward scattered GPS signals. *IEEE Trans. Geosci. Remote Sens.* **2002**, *40*, 50–65. [CrossRef]
10. Li, C.; Huang, W. An Algorithm for Sea-Surface Wind Field Retrieval From GNSS-R Delay-Doppler Map. *IEEE Geosci. Remote Sens. Lett.* **2014**, *11*, 2110–2114. [CrossRef]
11. Asgarimehr, M.; Zhelavskaya, I.; Foti, G.; Reich, S.; Wickert, J. A GNSS-R geophysical model function: Machine learning for wind speed retrievals. *IEEE Geosci. Remote Sens. Lett.* **2019**, *17*, 1333–1337. [CrossRef]
12. Reynolds, J.; Clarizia, M.P.; Santi, E. Wind speed estimation from CYGNSS using artificial neural networks. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 708–716. [CrossRef]
13. Asgarimehr, M.; Arnold, C.; Weigel, T.; Ruf, C.; Wickert, J. GNSS reflectometry global ocean wind speed using deep learning: Development and assessment of CyGNSSnet. *Remote Sens. Environ.* **2022**, *269*, 112801. [CrossRef]
14. Guo, W.; Du, H.; Guo, C.; Southwell, B.J.; Cheong, J.W.; Dempster, A.G. Information fusion for GNSS-R wind speed retrieval using statistically modified convolutional neural network. *Remote Sens. Environ.* **2022**, *272*, 112934. [CrossRef]
15. Zhao, D.; Heidler, K.; Asgarimehr, M.; Arnold, C.; Xiao, T.; Wickert, J.; Zhu, X.X.; Mou, L. DDM-Former: Transformer networks for GNSS reflectometry global ocean wind speed estimation. *Remote Sens. Environ.* **2023**, *294*, 113629. [CrossRef]
16. Saïd, F.; Jelenak, Z.; Park, J.; Chang, P.S. The NOAA track-wise wind retrieval algorithm and product assessment for CyGNSS. *IEEE Trans. Geosci. Remote Sens.* **2021**, *60*, 1–24. [CrossRef]
17. Lu, C.; Wang, Z.; Wu, Z.; Zheng, Y.; Liu, Y. Global ocean wind speed retrieval from GNSS reflectometry using CNN-LSTM network. *IEEE Trans. Geosci. Remote Sens.* **2023**, *61*, 5801112. [CrossRef]
18. Hersbach, H.; Bell, B.; Berrisford, P.; Hirahara, S.; Horányi, A.; Muñoz-Sabater, J.; Nicolas, J.; Peubey, C.; Radu, R.; Schepers, D.; et al. The ERA5 global reanalysis. *Q. J. R. Meteorol. Soc.* **2020**, *146*, 1999–2049. [CrossRef]
19. Williams, R.J.; Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.* **1989**, *1*, 270–280. [CrossRef]
20. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
21. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training recurrent neural networks. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 17–19 June 2013; pp. 1310–1318.
22. Schmidhuber, J.; Hochreiter, S. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780.
23. Graves, A. Generating sequences with recurrent neural networks. *arXiv* **2013**, arXiv:1308.0850.
24. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef] [PubMed]
25. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys. D Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
26. Gope, P.; Sarkar, S.; Mitra, P. Prediction of extreme rainfall using hybrid convolutional-long short term memory networks. In Proceedings of the 6th International Workshop on Climate Informatics: CI, Boulder, CO, USA, 21–23 September 2016; Volume 2016.
27. Xingjian, S.; Chen, Z.; Wang, H.; Yeung, D.Y.; Wong, W.K.; Woo, W.c. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 802–810. [CrossRef]
28. Tianyu, Z.; Zhenjiang, M.; Jianhu, Z. Combining cnn with hand-crafted features for image classification. In Proceedings of the 2018 14th IEEE International Conference on Signal Processing (ICSP), Beijing, China, 12–16 August 2018; pp. 554–557. [CrossRef]
29. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
30. Asgarimehr, M.; Zavorotny, V.; Wickert, J.; Reich, S. Can GNSS reflectometry detect precipitation over oceans? *Geophys. Res. Lett.* **2018**, *45*, 12–585. [CrossRef]
31. Ruf, C.S.; Gleason, S.; McKague, D.S. Assessment of CYGNSS wind speed retrieval uncertainty. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2018**, *12*, 87–97. [CrossRef]
32. Ruf, C.S.; Balasubramaniam, R. Development of the CYGNSS geophysical model function for wind speed. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2018**, *12*, 66–77. [CrossRef]
33. Clarizia, M.P.; Zavorotny, V.; Ruf, C. CYGNSS algorithm theoretical basis document level 2 wind speed retrieval. *Revision* **2015**, *2*, 148–0138.