NIST Technical Note 2167

Forecasting the Evolution of North Atlantic Hurricanes: A Deep Learning Approach

Rikhi Bose
Adam L. Pintar
Emil Simiu

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2167

National Institute of
Standards and Technology
U.S. Department of Commerce

NIST Technical Note 2167

Forecasting the Evolution of North Atlantic Hurricanes: A Deep Learning Approach

Rikhi Bose
Materials and Structural Systems Division
Engineering Laboratory

Adam L. Pintar
Statistical Engineering Division
Engineering Laboratory

Emil Simiu
Materials and Structural Systems Division
Engineering Laboratory

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2167

June 2021

U.S. Department of Commerce
Gina M. Raimondo, Secretary

National Institute of Standards and Technology
James K. Olthoff, Performing the Non-Exclusive Functions and Duties of the Under Secretary of Commerce for Standards and Technology & Director, National Institute of Standards and Technology

National Institute of Standards and Technology Technical Note 2167
Natl. Inst. Stand. Technol. Tech. Note 2167, 36 pages (June 2021)
CODEN: NTNOEF

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2167

Abstract

Accurate prediction of storm evolution from genesis onwards may be of great importance considering that billions of dollars worth of property damage and numerous casualties are inflicted each year all over the globe. In the present work, two classes of Recurrent Neural Network (RNN) models for predicting storm-eye trajectory have been developed. These models are trained on input features available in or derived from the North Atlantic hurricane database maintained by the National Hurricane Center (NHC). The models utilize historical data to estimate probabilities of storms passing through any location. Furthermore, inputs to the models are such so that the model development methodology is applicable to any oceanic basin in any part of the globe. Model forecasting errors have been analyzed in detail. The error analysis shows that the Many-To-Many class of models are appropriate for long-term forecasting while Many-To-One prediction models perform comparably well for $6-hr$ predictions. Application of these models to predicting more than 40 test storms in the North Atlantic basin shows that, at least for forecasts of up to 12 hours, they outperform all data-based storm trajectory prediction models cited in the paper. Apart from providing very fast predictions, the RNN models require less information than models used in current practice. Because the models presented herein emulate the trajectory trends of historical storms, they are currently being used for the simulation of synthetic storm tracks and features and the subsequent estimation of extreme wind speeds with various mean recurrence intervals.

Key words

Hurricane forecasting; HURDAT2; Long Short-Term Memory (LSTM); North Atlantic hurricanes; Recurrent Neural Networks (RNN); Time series forecasting.

Table of Contents

List of Figures

1. Introduction

Hurricanes and tropical storms are rotating storms originating in the Atlantic basin. The maximum sustained wind speeds exceed 74 mph (33 ms^{-1}) for hurricanes, and are comprised between 39 mph (17 ms^{-1}) and 74 mph for tropical storms. The storms form as warm moist air rises from the sea surface and is replaced by cold dry air. This results in large cloud systems that rotate about their low-pressure core region owing to the Coriolis effect. In the northern hemisphere the rotation is counterclockwise. The probability of a storm formation is maximum during the months of highest sea surface temperature. The purpose of this paper is to present a Deep Learning (DL) based methodology for forecasting North Atlantic basin storm trajectories starting from their genesis.

The National Hurricane Center (NHC) uses several models for forecasting storm tracks and intensity. These models may be characterized as statistical, dynamical and statistical-dynamical. Statistical models (e.g., the Climatology and Persistence (CLIPER) model [1]) are typically less accurate than state-of-art dynamical models and are used as baseline models. Dynamical models, such as National Oceanic and Atmospheric Administration's (NOAA) Geophysical Fluid Dynamics Laboratory (GFDL) hurricane prediction model [2] numerically solve thermodynamics and fluid dynamics equations that govern the atmospheric motions and may require hours of modern supercomputer time for providing even a 6–hr forecast. Other models combine the skills of the statistical and dynamical models. To take advantage of the skills of available statistical and dynamical models NHC uses ensemble prediction systems (EPSs) (such as the ensemble model of the European Centre for Medium-Range Weather Forecasts (ECMWF) [3, 4]) that can be more accurate and reliable than their components.

DL methods have recently taken giant strides in using large amounts of data to uncover relations that are impossible to obtain by conventional techniques through the creation of a functional form between the input feature space and the output target variables. We propose a DL approach for efficient prediction of hurricane trajectories over several $6-$ hr time intervals. The target variables in the DL hurricane track prediction problem are the coordinates of a storm's center. The functional dependence includes as inputs storm features available in or derived from the hurricane database maintained by NHC.

The data-based approach has been recently applied to several aspects of weather forecasting [5]. Hurricane forecasting using data-based approaches have mainly been formulated as image processing problems in [6–9]. A Convolutional Long short-term memory (ConvLSTM), a mixed neural network model, was used to extract spatio temporal information from a large database of instantaneous atmospheric conditions recorded as a pixel-level history of storm tracks. The ConvLSTM comprises Convolutional Neural Network (CNN) layers and layers of a class of Recurrent Neural Networks (RNN), called Long short-term memory (LSTMs) [10]. A tensor-based Convolutional Neural Network (TCNN) was used in [9] to improve forecasting hurricane intensity model performance in coupled TCNN (C-TCNN) and Tucker TCNN (T-TCNN). Hurricane track forecasting using satellite images in [8] used the Generative Adversarial Network (GAN). They used image time-series of

typhoons in the Korean peninsula for model training. The GAN model was tested on 10 test storms previously unseen by the model during training. Average prediction error for $6-hr$ forecast for the test storms was 95.6 km.

In [11], a flexible sparse RNN architecture was used to model the evolution of North Atlantic hurricane tracks. Available time records of the target storm were initially compared with those of other storms, and a Dynamic Time Warping technique was used to assign similarity scores to each storm available in a historical storm database. Only storms with high similarity scores were chosen for model training. Storm evolution prediction was thus improved by incorporating historical trends. However, for obtaining appropriate storms for the model to train on, significant numbers of time records from the target storm are required to be available. In the present work, we account for translation trends of historical storms by using storm displacement probabilities as input features. The storm displacement probabilities take into consideration $6-hr$ displacement of all storms from a given computational cell to adjacent cells in a coordinate-transformed domain.

In another attempt to use RNN models, authors in [12] used a grid-based Neural Network approach for hurricane trajectory forecasting. An Long short-term memory (LSTM) RNN model predicted storm-wise scaled grid numbers in the $2-D$ latitude-longitude domain. The grid-based prediction scheme was based on the fact that, for a historical storm, the distance being traveled is proportional to the the number of time records available for it. The model forecasts a storm's location six hours in advance, and is claimed to improve upon the forecasting performance of the models developed in [11]. However, no results were provided for long-term forecast. The compounded error accumulation, a most pressing issue related to the application of NNs for storm track prediction, was not considered in their work. In the present effort, we attempt to use LSTM-RNNs to provide long-term forecasts for storm evolution up to thirty hours in advance. We demonstrate the compounded error accumulation problem and its remedy by using Many-To-Many prediction-type RNNs.

In a recent work, CNNs coupled with Gated recurrent unit (GRU) RNN was employed [13]. The use of a feature selection layer before the NN layers augmented the model's learning capability from the underlying spatial and temporal structures inherent in trajectories of tropical cyclones. The model's performance was compared with the performance of other numerical and statistical weather forecasting models. They reported a $12-hr$ forecast error of ≈ 100 km (62 mi) between the predicted and true cyclone-eye locations, which is comparable or marginally less than the statistical model used in [14] and a numerical model used in [15]. However, the NN model used in [13] was more accurate than the aforementioned statistical and dynamical models for long-term forecasts. In particular, the $72-hr$ track forecast error was was less than half the errors reported for traditional track forecasting models. The RNN models developed herein were tested for hundreds of validation and test storms and the prediction errors were extensively analyzed. Six- and twelve-hour forecast error for the prediction models reported herein were ≈ 30 km and 66 km, respectively. To our knowledge, the LSTM-RNN models developed in the present work provides the best performance of all data-based NN models developed so far for short-term storm trajectory forecasting at least up to 12 hours.

The paper is structured as follows. In Section 2, the database used for model development, the HURDAT2 database has been discussed from the statistical, the feature engineering and the model formulation points of view. The methodology used for the calculation of $6-hr$ storm displacement probabilities has also been described in this section. Section 3 discusses the model type, its architecture and implementation, training strategies, and hyperparameter tuning. Section 4 contains the prediction results of the trained models and an extensive analysis of forecasting error, compares the predicted trajectories with trajectories of historical test storms, and discusses limitations of the models developed herein and the scope for future improvements. Conclusions are drawn in Section 5.

2. Database: HURDAT2

The National Hurricane Center conducts a post-storm analysis of each storm and updates a database that contains a six-hour best track for each storm analyzed. Features of the Atlantic basin storms were originally tabulated in the HURDAT (HURricane DATabase) database [16, 17]. An updated version of the database, named the HURDAT2 (Hurricane Data 2^{nd} generation) was developed in 2013 and is maintained by NHC. HURDAT2 lists the storm numbers and names, the time of the record (year, month, date and time), record identifier (e.g., landfall, change of systemstatus), system status (e.g., tropical storm of hurricane intensity) of a storm, storm location (latitude and longitude), maximum 1-min wind speed at 10 m elevation in knots, central pressure in millibars, and radius in nautical miles corresponding to 34, 50 and 64 knot wind speeds in all four quadrants. However, the database is incomplete. For example, the central pressure is tabulated for each storm only since 1979, and the wind radii since 2004. Moreover, although storms have been tabulated since 1851, data tabulated before the use of satellites in 1970s was based on sparse observations, and is therefore less reliable. So, only part of the database is useable for DL purposes. This section presents a statistical analysis, on the basis of which data used for DL model training, validation and testing was compiled.

2.1 Feature space

There are 1893 storms listed in the HURDAT2. We intend to use a time series for each storm. The maximum number of records for each storm used in this work to predict the records for the next $6-hr$ timestep is five. Upon retaining storms with seven or more time records (we have used average storm translation speed over $6-hr$ intervals as an input feature), 1800 storms may be used. Only 560 of these 1800 storms have both the central pressure and maximum wind speed included for all records. Among these 560 storms, only 254 storms have radius corresponding to 34 knot wind speed assigned for all quadrants at all times. Even then, for many of these storms, the assigned radius is zero for 34 knot wind speed for a few quadrants. If we relax the criterion for storm selection by only keeping storms with maximum $1-min$ wind speed at 10 m elevation included in all records, 1736 storms can be used. Therefore, the central pressure and radii corresponding to the three wind speeds in four quadrants were not included in the input feature space. We used the latitude, longitude and maximum $1-min$ wind speed at 10 m elevation to construct the feature space for model training.

To construct a valid feature space and to reduce the effect of areal distortion associated with the spherical coordinate system at higher latitudes, a transformed set of coordinates was used instead of the latitude (ϕ) and longitude (λ) coordinates. We used Lambert's conic conformal projection (LCC) [18] to obtain the transformed pair of $x-$ and $y-$coordinates. LCC projection is widely used for geographical representation of North America because of its superior projection properties in mid latitudes. A cone is placed on top of the sphere to be projected and the points on the sphere are then projected onto the surface of the cone. The $2-D$ unrolled cone-surface coordinates are the projected coordinates. By this method,

the shape is retained, however, the area is distorted. Two standard parallels associated with the conic projection (latitudes where the cone passes through the sphere) used in the current transformation are $33^{\circ}N$ and $45^{\circ}N$. Most of the U.S. mainland is contained within these standard parallels. No distortion is obtained along the standard parallels. Distortion increases farther away from these coordinates.

In addition to the LCC projected coordinates, we use the storm translation velocity as inputs. Both translation direction and speed were used. These are specifically included for the purpose of storm intensity modeling [19, 20]. It was assumed that the storm translation is linear between two time instants six hours apart. Translation direction (θ) is obtained at l^{th} time instant as,

$$\theta_l = \tan^{-1}\left(\frac{\phi_l - \phi_{l-1}}{\lambda_l - \lambda_{l-1}}\right) \tag{1}$$

Similarly, six-hour-averaged translation speed (V) is calculated going backward in time. The distance (d) between storm locations at two time instants has been calculated using the Haversine formula.

$$V_l = \frac{d(\phi_{l-1}, \phi_l, \lambda_{l-1}, \lambda_l)}{\Delta t \equiv 6hrs}. \tag{2}$$

The maximum $1-min$ sustained wind speed (w_m) at 10 m elevation is also used from the HURDAT2 database. In HURDAT2, w_m is approximated to the nearest 10 kt (5.14 ms^{-1}) between 1851 and 1885 and to the nearest 5 kt (2.57 ms^{-1}) thereafter. Both w_m and V were used in ms^{-1} unit in this work.

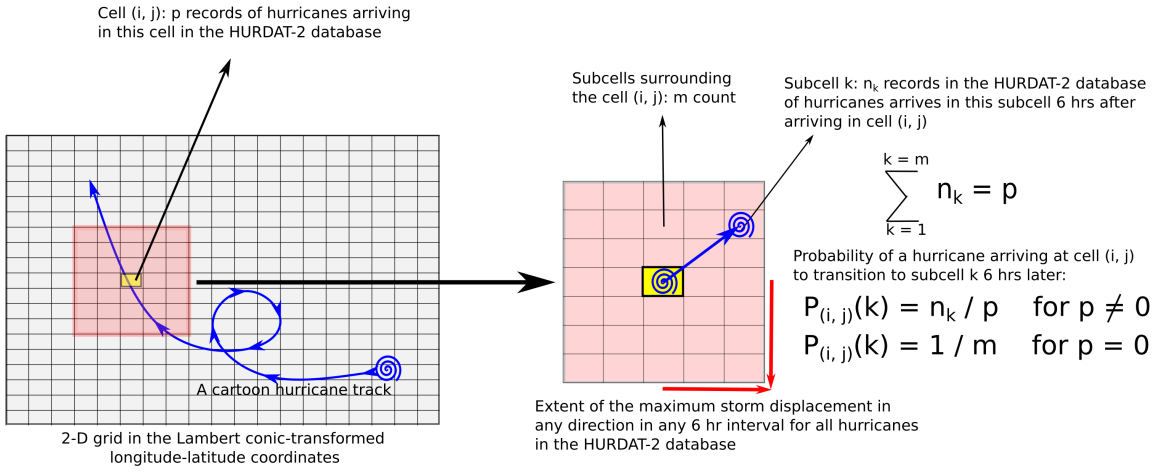

Fig. 1. Schematic depicting the calculation of displacement probabilities associated with each cell in the grid domain.

It is desirable for the input features to contain information associated with trends of past

storm motion [11, 21]. Once such information is provided by appropriate feature engineering, DL methods are well-equipped to excavate complex patterns or relations among input and output features that are otherwise intractable. So, storm displacement probabilities computed from the historical storms were also included as input features for the current model. A schematic depicting the calculation of displacement probabilities is shown in Fig. 1. At first, the $2-D$ domain bounded by the extents of the $x-$ and $y-$ coordinates (the extents are the maximum and minimum of each coordinate corresponding to any storm record in the considered database) was decomposed into rectangular computational cells. Consider all storms for which a record is contained in any one of those computational cells. For those storms, the maximum numbers of cells traversed by any storm in any $6-hr$ interval, in the $x-$ and $y-$ directions, are m_x and m_y, respectively. Therefore, associated with any given cell (i, j) (colored yellow in Fig. 1) is a set of $m = (2m_x + 1)(2m_y + 1)$ cells within which all $6-hr$ displacements of any historical storm is contained. Such a set is colored red in Fig. 1. Assume there are p records of hurricanes that arrived at the $(i, j)-th$ cell and then transitioned to the k^{th} associated cell in the next 6 hours. If this leads to n_k records being sampled at k^{th} associated cell, $\sum_1^m n_k = p$. Therefore, the displacement probability of a storm arriving at the $(i, j)-th$ cell transitioning to the k^{th} associated cell in the next 6 hours is,

$$p_{(i,j)}(k) = \begin{cases} \frac{n_k}{p}, & \text{if } p \neq 0 \\ \frac{1}{m}, & \text{if } p = 0 \end{cases} \tag{3}$$

It is clear that for a fine grid, the number of cells with $p = 0$ would be larger than for a coarse grid. On the other hand, if the grid is too coarse, storm motion trends reflected by the displacement probabilities could be obscured. However, for a finer grid (and/ or for a very thin upper tail of the distribution of V), m would increase, and n_k would decrease. In addition to increased number of features at each instant, displacement probabilities from a cell to adjacent cells could be more biased on a specific historical storm's displacement, which is not desirable for the prediction of new storms.

2.2 Statistics

The computational domain containing the 1736 storms with at least seven records, including w_m, covers intervals of $\phi \in [7.4^\circ N, 81^\circ N]$ and $\lambda \in [109.5^\circ W, 63^\circ E]$. The storm translation speeds, which does not exceed $V = 40.5\ ms^{-1}$ for any storm in the database, determines the number m of subcells associated with a given computational cell, as shown in Fig. 1.

Histograms of ϕ, λ and V have been plotted in Fig. 2. It is evident that for all these quantities, the right tail of the distribution is very thin. Only for 33 of the 47880 time records was $\phi > 70^\circ N$, and only for 26 time records was $\lambda > 10^\circ E$ (to calculate V and include it as an input feature, the first time record of the 1736 storms was excluded from the dataset). Only 93 records of storm motion satisfied the condition, $V > 25\ ms^{-1}$. Therefore, it is reasonable to exclude the 71 storms for which these limits were exceeded. Thus, 1665 storms remain in the final database used for model training, validation and testing.

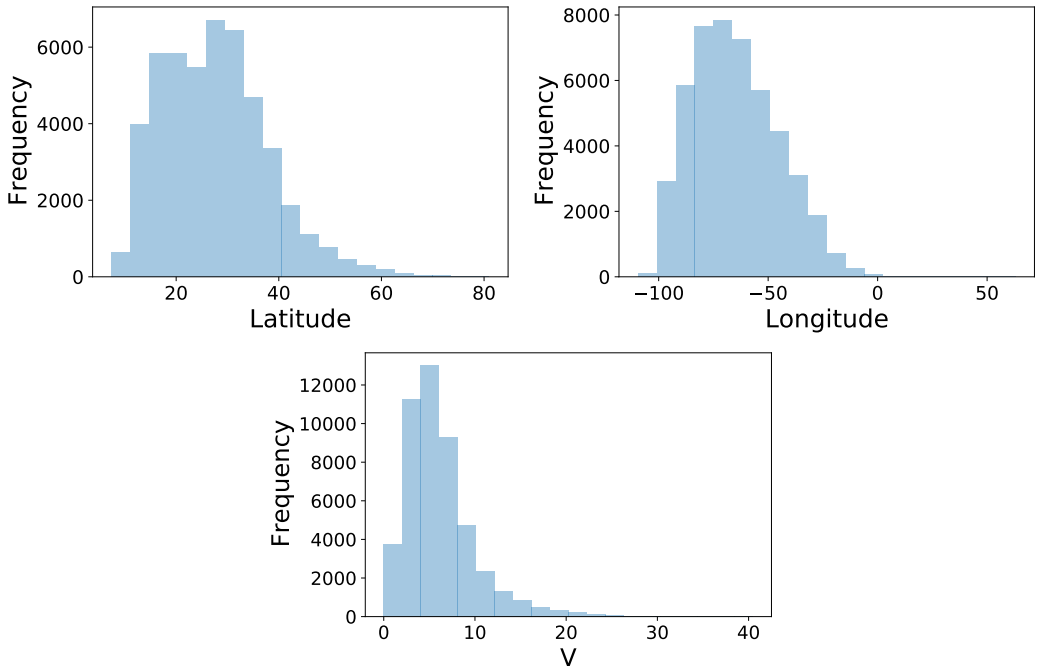

Fig. 2. Histograms of latitude (ϕ) and longitude (λ) in degrees and storm translation speed (V) in ms^{-1}. Positive values of ϕ and λ represent the northern and eastern hemispheres, respectively. 1736 storms from the *HURDAT*2 database are used (see text).

Storm displacement probabilities were calculated based on the evolution of those 1665 storms. 61 grid points were used in both x and y directions of the LCC projected coordinates, resulting in the whole 2-D domain being decomposed into 3600 computational cells. Despite application of the three above mentioned criteria, 1687 computational cells had no time records of an eye of a storm passing through those (i.e., for 1687 out of the 3600 computational cells, sampled records $p = 0$). The maximum $6-hr$ storm displacement in the 1665 storm-database could be captured within $m_x = m_y = 4$. So, the displacement probability was calculated for $m = 9 \times 9 = 81$ associated cells for each computational cell. Therefore, the number of input features for each time record was 86 (LCC transformed x and y coordinates, storm translation speed V and direction θ, maximum $1-min$ wind speed at 10 m elevation w_m, and historical $6-hr$ displacement probabilities calculated at 81 cells associated to the cell containing x and y).

2.3 Zonal data

Inception locations of the 1665 storms considered for compilation of the database for model formulation have been plotted in Fig. 3. The Atlantic basin is generally split into five zones or sub-basins, namely, the Tropical Atlantic, the Caribbean sea, Gulf of Mexico, East Coast, and Sub-tropical Atlantic. Total storm inceptions in each of these sub-basins

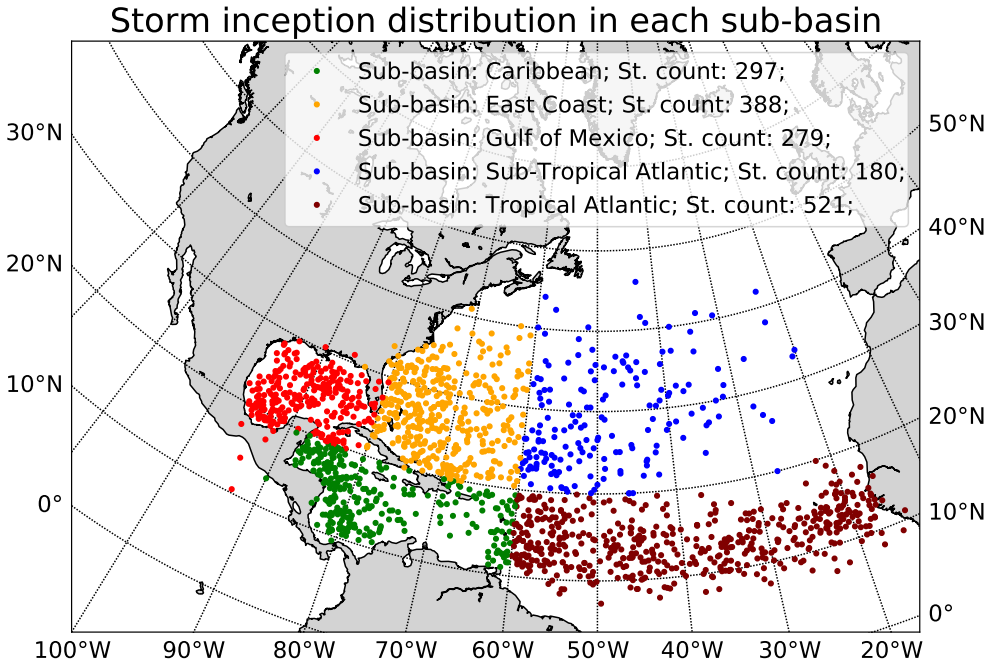

Fig. 3. Distribution of storm inception in each sub-basin of the 1665 (out of the 1893) storms from the HURDAT2 database used for modeling. Number of storms generated in each sub-basin are included.

are also listed. Figure 4 shows the zonal probability for inception of the historical storms considered in Fig. 3. Most of the storm inceptions take place in the tropical region where sea-surface termperature is higher. The smallest number of storms are generated farther north of latitude $20^{\circ}N$ in the Sub-Tropical Atlantic.

Each historical storm's path-line may be generated by plotting all records from its lifespan. Each $6-hr$ motion of a storm may be approximated as a straight line connecting two consecutive records in a storm's lifespan. In Fig. 5, path-lines of 50 randomly chosen storms emerging from each sub-basin are shown. The path-lines are a reasonable representation of the underlying trend of storm motion. One might surmise that since the storms emerging from a particular sub-basin would be subjected to similar input conditions (e.g., sea surface temperatures), the characterization of their evolution trajectories could be based on their inception zones. However, plots in Fig. 5 indicate that a storm's evolution trend is characterized by its instantaneous location. Indeed, the storm tracks show a clear evolution trend. For example, the overall trend of storms generated in the tropical Atlantic sub-basin tend to initially move westward. Then these storms travel northward. Irrespective of a storm's inception zone, north of $\approx 20^{\circ}N$, the direction of a storm's motion slowly

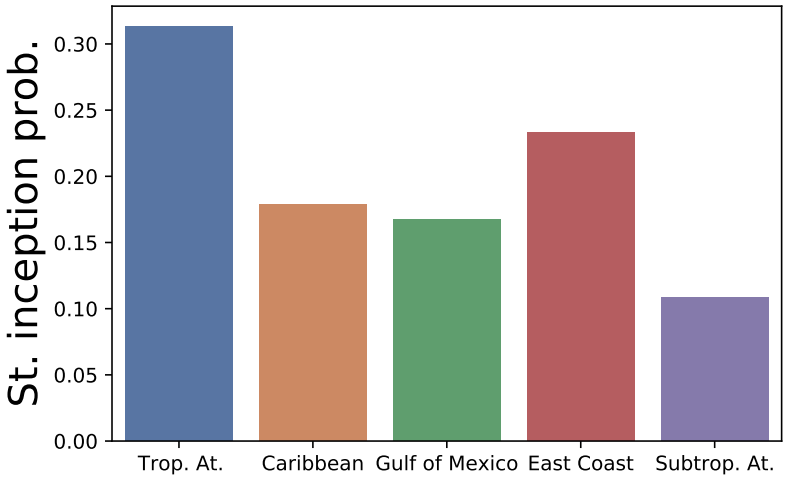

Fig. 4. Zonal storm (St.) inception probability distribution of the 1665 (out of the 1893) storms from the HURDAT2 database used for modeling.

turns eastward. Also, a storm generally moves northward towards regions with colder sea-surface temperature. Therefore, training models based on inception sub-basin may not be fruitful. As suggested by the plots depicting the storm tracks, an instantaneous location-based evolution model that also learns historical storm trajectory trend might work well. This work has used DL models that feed on input features such as local coordinates and historical displacement probabilities.

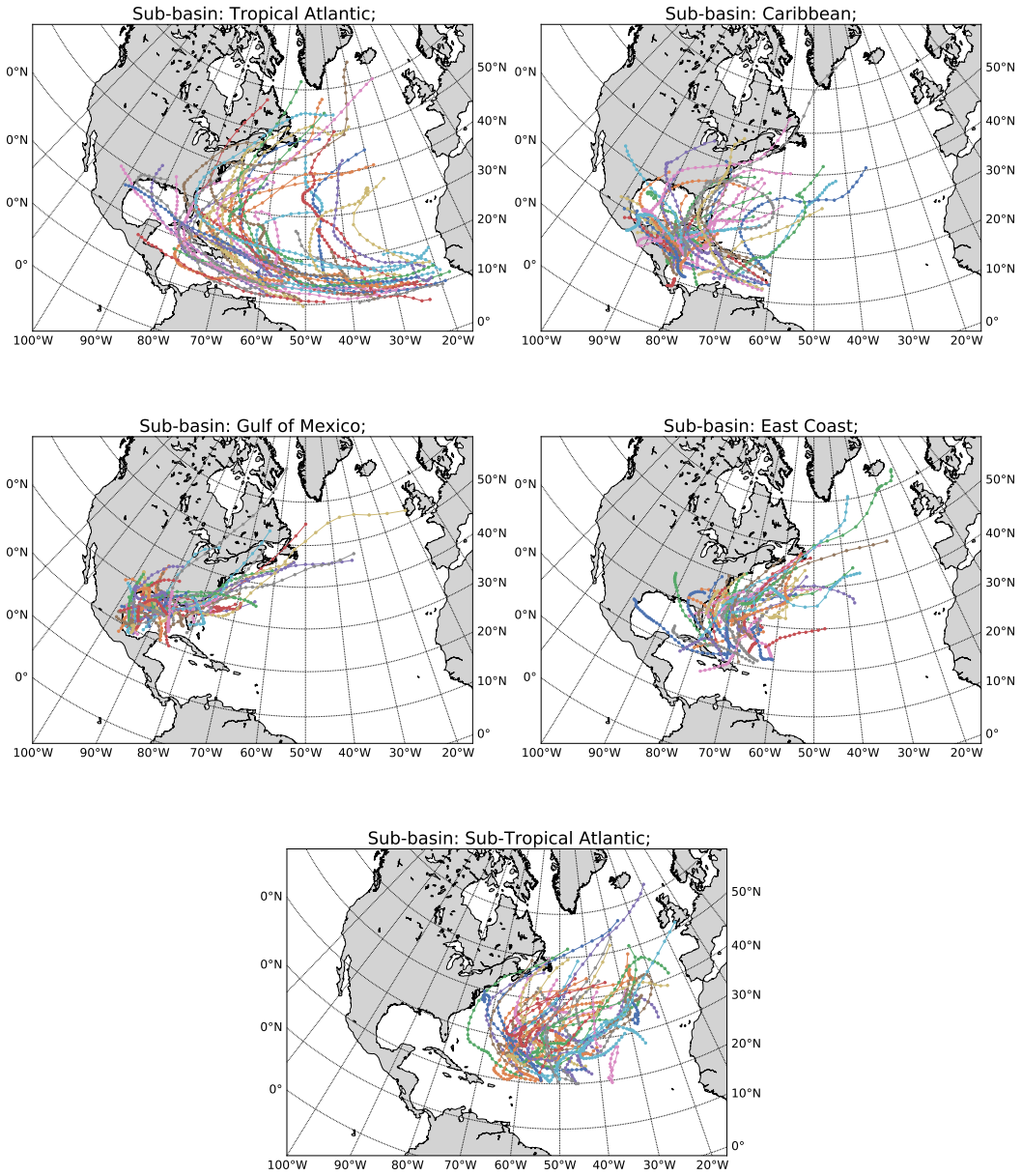

Fig. 5. 50 randomly selected historical storm tracks from HURDAT2 plotted based on their inception sub-basins. Trajectory between two consecutive records in $6-hr$ intervals in a storm's lifetime has been approximated as a straight line.

3. Model development

Once a database was obtained, we proceeded to the model development. Both classification and regression problems may be formulated. In the current work, for either of these problems, we used the Long short-term memory (LSTM) [10] Recurrent Neural Networks (RNN). This section considers the formulation of the current problem with the chosen RNN model and its implementation, including model training and model parameter tuning.

3.1 Classification

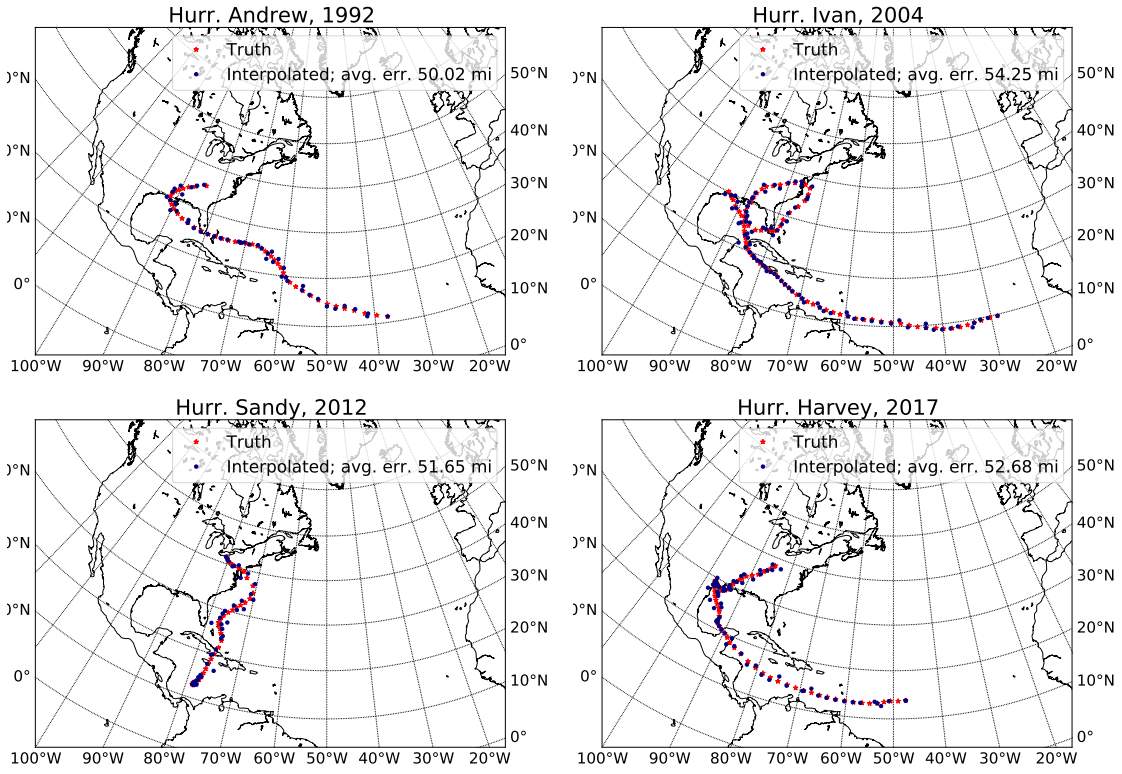

Fig. 6. Interpolation error of selected hurricane trajectories; actual trajectories are colored red, and the blue symbols represent the actual coordinates interpolated to the center of the $2-D$ computational cell containing it. Error in miles averaged over the whole trajectory is given.

A classification problem may be envisaged utilizing the structured computational grid for the calculation of $6-hr$ storm displacement probabilities. Given a storm's current location, translation velocity and $6-hr$ displacement probabilities, the model could be trained on predicting the subcell number k in Fig. 1. In that case, the training and testing data targets would be the One-Hot encoded subcell number k, to which a storm eye located within a given computational cell at a given instant transitions in the next time instant.

Generally, DL algorithms are trained to assign a probability to each class in a classification problem. In the current problem, probabilities will be assigned to each of the $m = 81$ subcells associated with a computational cell. From a computational cell (i, j), a storm moves to the subcell for which the assigned probability is maximum. The storm location at the next time instant is calculated from the $k-th$ associated cell's location relative to the computational cell (i, j). The storm eye is interpolated to the center of the $k-th$ associated cell containing it.

Selected hurricane trajectories are shown in Fig. 6. In addition to the actual $6-hr$ hurricane eye positions in the database (marked by red symbols), the positions obtained by interpolation to the center of the computational cells containing these are also shown. Error in miles averaged over the whole trajectory of each hurricane have been reported. The computational grid (as in Fig. 1) used for this purpose contains 60 cells in each of longitude and latitude directions. The number of subcells associated with each computational cell is 143 (13×11) and is used for displacement probability calculations as well as for prediction of classes for the classification problem on storm trajectory evolution. The averaged interpolation error in miles is $\approx$ 80.5 km (50 mi) for each location for all hurricanes. Due to the interpolation error alone, the predictions are likely to veer off the actual trajectories very quickly. As this error is more than the average forecasting error obtained from the regression problem, the classification approach was not pursued.

3.2 Regression

The supervised regression problem under consideration is relatively straightforward. The inputs to the model are the two LCC projected x– and y–coordinates, storm translation velocity (speed V and direction θ), max. 1–min windspeed at a 10 m elevation w_m, and 6–hr storm displacement probabilities associated with the storm's position in the computational grid at the current and few previous timesteps (86 input features at each timestep $\times$ the number of timesteps). The model outputs are the storm position(s) x and y at $6-hr$ intervals for the chosen number of output timesteps. The model performance, measured in the classification problem in terms of accuracy, is measured by the loss function defined as the mean squared error ($m.s.e.$) between the model predicted and the true position(s) of N samples.

$$m.s.e. = \frac{\sum_{i=1}^{N} (x_{predicted,i} - x_{true,i})^2 + (y_{predicted,i} - y_{true,i})^2}{N}$$

Gradients of the $m.s.e.$ are computed w.r.t. changes in parameters/ weights of the model. The model weights are updated so that the $m.s.e.$ is minimized. V, θ and $6-hr$ displacement probabilities associated with each timestep are computed from the storm's position at the current and previous timesteps. These features are used as inputs in the next timestep. For the current purpose of predicting the storm trajectories, true values of w_m are used at each timestep for model testing.

3.3 Model: LSTM RNN

Recurrent Neural Networks (RNN) were innovated to extract pattern and context from sequences. RNNs are applied in a wide range of sequence related problems, including modeling and prediction of languages and sentiment, video tagging, a sequence prediction in time. HURDAT2 is a sequence of time records. RNNs may therefore be expected to be useful for predicting evolving dynamical systems that depend on events of the past, such as hurricanes [7, 12]. Among all RNN algorithms, Long Short-Term Memory (RNN) algorithm was prescribed in [10] to tackle the vanishing gradient problem. Over long sequences, relevant past information may get lost or, equivalently, gradients may vanish while training a model using back propagation. In an LSTM unit, past information may be retained via a cell state that passes through all LSTM layers. Early use of LSTMs in weather forecasting is reported in [22].

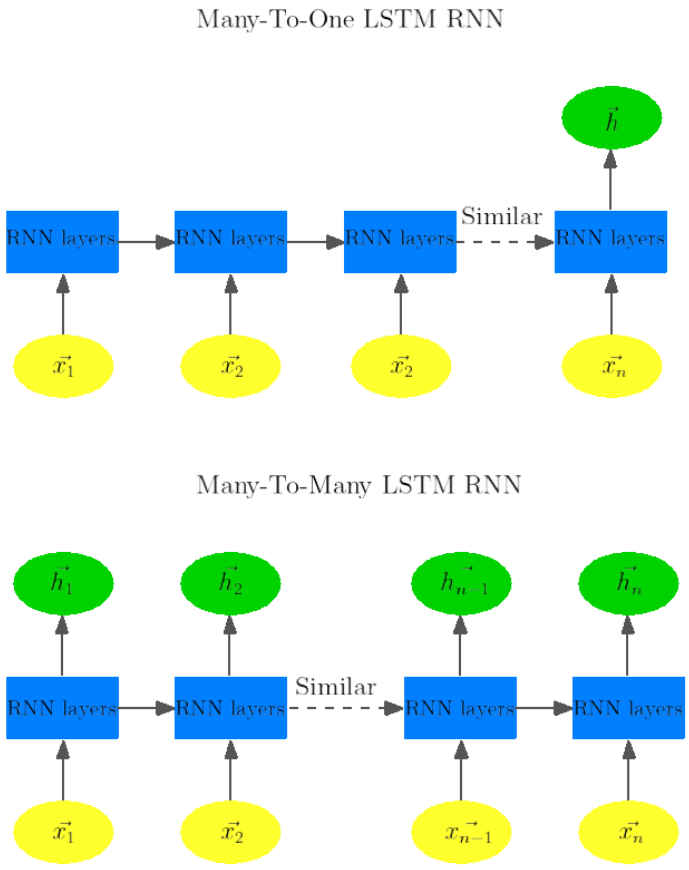

Fig. 7. Different *LSTM* RNN architectures used in the present study.

Schematics of the LSTM RNN models used in the present work have been shown in

Fig. 7. In the figures, the vector represented by $\vec{x}_i$ contains the input features at the i^{th} time instant. The input layer is colored yellow. For Many-To-One algorithms ($M2O$) n–time records are fed to the model to obtain the output vector $\vec{h}$ in the output layer (green). One or several layers of LSTM-RNN units (blue) may be used. Increased depth of the networks augments their ability to learn complex patterns. The $M2O$ model results presented here used 3 layers of LSTM units. Many-To-Many ($M2M$) prediction models used here output the same number of time records ($\vec{h}_i$'s) as the number of input time records (n in the diagram). Bi-Directional LSTM layers were used for the $M2M$ LSTM model. Each bi-directional layer comprises two layers of LSTMs receiving the inputs separately in an ascending and a descending order in time, respectively. Some information from the future may thus be used to predict an earlier timestep.

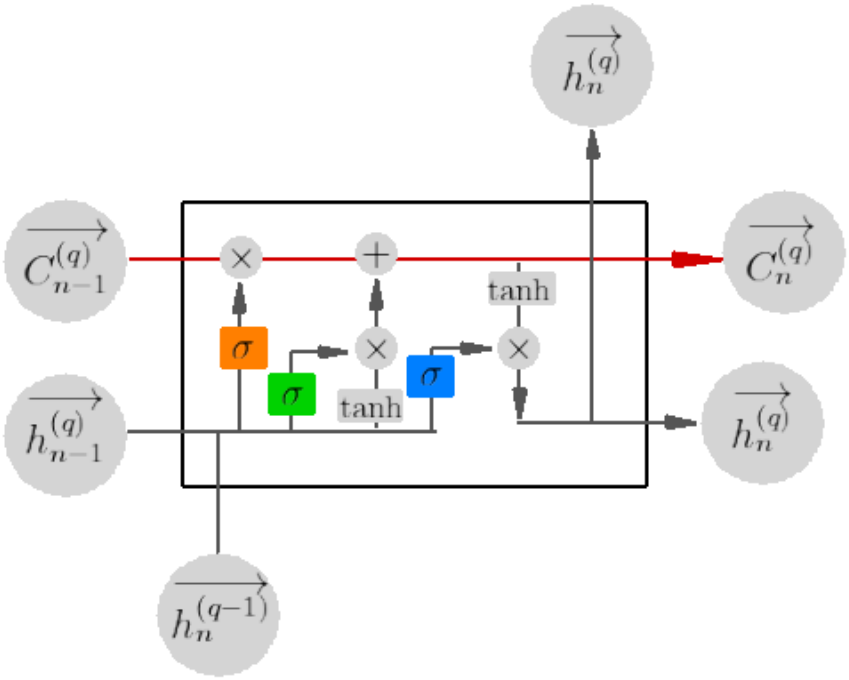

Fig. 8. A representative LSTM unit; the unit belongs to the q^{th} layer of the model. Corresponding input time step is n, i.e., at the input layer ($q = 1$), $\vec{h}_n^0 = \vec{x}_n$ input features at the n^{th} timestep. The symbols $\times$ and $+$ represent pointwise operation.

An LSTM unit/ cell is shown in Fig. 8. The superscript in the vector variables indicates the layer number (this unit belongs to layer q of the model); the subscript denotes the corresponding input step n as in Fig. 7. A cell comprises four main components, the cell state (passing through the units, colored red), the forget gate (colored orange), the input gate (colored green) and the output gate (colored blue). The three gates basically apply the three activation functions (in the schematic, σ and tanh represent sigmoid and hyperbolic tangent activation functions, respectively), each of which has a specific role in informa-

tion propagation through the model. The cell state ($\overrightarrow{C_n^{(q)}}$) is the unique component of an LSTM RNN. The cell state passes through all timesteps $n = 1, 2, ...$ of a given layer, and is therefore able to preserve information from the past and also accumulate new information with increasing n [23]. The cell shown in the diagram receives an input from the previous layer belonging to the same time step $\overrightarrow{h_n^{(q-1)}}$, and also from the previous time step in the same layer, $\overrightarrow{h_{n-1}^{(q)}}$. Based on these inputs to the cell, the forget gate dictates the part of the cell state to be discarded at the current unit. On the basis of these same inputs, input gate dictates the information from the present inputs to be added (marked by $+$) to the cell state. Consequently, after these operations, the cell state gets modified in the current LSTM unit ($\overrightarrow{C_{n-1}^{(q)}} \rightarrow \overrightarrow{C_n^{(q)}}$), which is the cell state received by the LSTM cell to the right, i.e., the next timestep in the same layer. The updated cell state also participates in obtaining the output from the current cell ($\overrightarrow{h_n^{(q)}}$) after the sigmoid activation is applied at the output gate.

3.4 Model: Architecture & Implementation

Models trained on varying numbers of input time records (n) can predict one or several time steps at once based on their architecture. Both Many-To-One ($M2O$) and Many-To-Many ($M2M$) type prediction algorithms have been used. As their nomenclature suggests, upon processing a time sequence with n time records, the $M2O$ prediction models forecast the storm locations at only one time instant, while the $M2M$ models used here output n number of time records. From here onwards, models are named as $M2On$ or $M2Mn$ to reflect their architecture. As each timestep in the training database $\equiv 6-hr$, the $M2On$ and $M2Mn$ models forecast 6 and $6n$ hours at once, respectively.

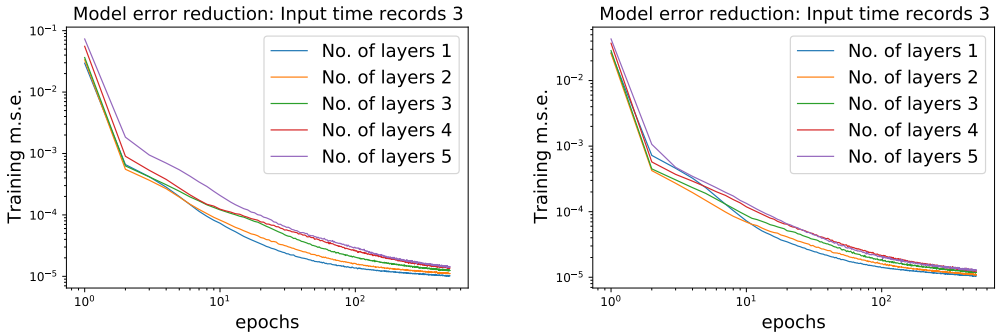

Fig. 9. Training loss plotted against epochs for different number of hidden LSTM-RNN layers in the $M2O$ prediction models with similar (different) number of trainable parmaters in each model shown in the left (right) frame.

Hidden layers: The number of hidden layers is an important model parameter. Theoretically, a model is able to capture more complex patterns in the underlying data with increase

in number of hidden layers/ model depth. However, increasing the number of layers will increase the number of trainable parameters, possibly resulting in data overfitting. In a set of simulations with up to 5 hidden layers for the *M*2*O* LSTM-RNNs, the layer-to-layer output dimension was reduced with increasing depth in the model, so that the number of trainable parameters did not change significantly among the models with additional hidden layers. In another set of simulations, the layer-to-layer output dimension was kept constant in the hidden LSTM layers, so that the number of trainable parameters was proportional to the number of hidden layers. The number of trainable parameters represents the degrees-of-freedom for a model. The number of input time records was also varied. In Fig. 9, reduction of the loss function defined as the mean-squared-error (*m.s.e.*) between the predicted and true scaled LCC coordinates, is plotted against the number of epochs. The left frame shows the error reduction for models with similar numbers of trainable parameters; the right frame shows the same plot for models with various numbers of trainable parameters. For both sets of models the error level saturates at around the same value after about 200 epochs. The performance of models after convergence does not improve with additional LSTM layers in either plot. Similar performance was also obtained for models (not shown here) that feed on more or less than 3 input time records.

Increasing hidden RNN layers may result in an increase in the number of trainable parameters. To avoid overfitting the training data, approximately similar numbers of trainable parameters were used when checking for optimal numbers of layers for both *M*2*O* and *M*2*M* models. In addition to the input and output layers, three layers of LSTM-RNN cells were used for the final version of the *M*2*O* LSTM model. It was found that increasing the number of hidden LSTM layers while keeping the number of trainable parameters approximately the same did not improve the results significantly. The numbers of neurons in the three hidden LSTM layers between the input and output layers were 128, 32 and 8, respectively.

For the *M*2*M* models, two bi-directional LSTM layers were used at either side of a repeat-vector layer. This layer is required for the multi-step data transfers between two layers and does not contain any trainable parameter. Before the output layer, a time-distribution layer was required to output multiple time steps. Each of the LSTM cells in bi-directional layers had an input and output dimension of 64; a total of 128 neurons were used in each of these layers. The number of trainable parameters for all the tested models were between 3 to 6 times the numbers of data sequences/ samples obtained from the database.

Dropout: To increase the robustness of NN models, a regularization parameter called dropout is used. This value indicates the number of randomly chosen neurons to be switched off in each layer. Dropping out neurons increases variance in model prediction while reducing the model's bias towards the training data. For this reason the important dropout parameter is widely used to avoid overfitting. We used a dropout value of 0.1 for each hidden layer, meaning that 10 % of the neurons were randomly dropped at each hidden layer when feeding the data forward from layer to layer. Special care was taken in considering the number of trainable parameters for each model. Tuning the dropout value to increase

model robustness was deemed unnecessary.

Optimization: The models were trained with the Adam optimization algorithm [24]. The algorithm updates the weights of the NNs via backpropagation based on the loss functions calculated in the current epoch or the optimization iteration loop. Finally, the models were implemented using the Keras API. Keras is a popular high-level NN framework written in Python. It can use several lower-level APIs as chosen by the user. We used the Keras API with TensorFlow as the lower level backend library.

3.5 Model: Training strategies

Number of input time records (n): Several aspects of model training and validation warrant discussion due to the complexity of the problem under consideration. An important consideration is the number of input time instants, n that could minimize the model prediction error. Intuitively, increase in n should make the model prediction more accurate. However, a high value of n implies that the predictions may only be obtained when the storm has significantly evolved. Consequently less time is available for preparation of a possible landfall, which is undesirable. This choice also dictates model training strategy. Number of input time records used in each input sequence implicitly determines the number of data sequences that may be generated from the storm database for model training/ validation/ testing. Additional preprocessing such as zero padding may be used at early stages of a storm's life span (available time instants $< n$). To mitigate this issue, both $M2On$ and $M2Mn$ LSTM models were developed for a range of n, between 1 and up to 5 time instants ($n_{max} = 5$ in Fig. 7).

Data scaling: Scaling of the data fed to the neural networks is an important aspect of the present problem. NNs perform better when the input data is contained in the interval $[0,1]$. Note that each storm in the database is an individual entity that may be totally uncorrelated with some or most other storms. Also, the total distance traveled by any storm in the database should be proportional to the number of available records for that storm. This has led other researchers [12] to use storm-based scaling, so that the features for any storm contains both upper (1) and lower (0) bounds. In this work data normalization for the whole database under consideration was performed using the Min-Max scaler. Each feature f is scaled as, $f_s = \frac{f - f_{min}}{f_{max} - f_{min}}$, so that scaled feature f_s is contained in the interval $[0,1]$. This scaling is preferred over standardization, because the displacement probabilities also belong in this interval. Furthermore, to predict a new storm's evolution using the trained model, scaling of the features is well defined as the maximum and minimum of a feature are taken from the database on which the model is trained and validated. The storm-wise data scaling method renders prediction of a new storm impossible, because the relevant scaling parameters such as the minimum, maximum or the mean and standard deviation of a feature for a new storm are unknown a priori.

Sequence generation: Although whole database scaling was preferred over storm-wise data scaling, storms were segregated for the purpose of training, validation and testing. A few important historical storms were chosen for testing. The rest of the storms were

chosen randomly from the database without replacement. Of the 1665 storms considered, 1332 (i.e., $\approx 80\%$) were used for training the model while 15% storms were used for validation. Data sequences for training, validation and testing were generated separately from the segregated lists of storms. The number of sequences used for training, validation and testing varied between specific training instances because of random sampling of storms and also because lifespans of storms vary. The number of training data sequences was always $\sim 30,000$ or more. In some earlier works [11, 12], a portion of a given storm's data was used for training. The model was tested on the remaining portion of the storm's records. In our view, this provides an unfair test of the model performance, owing to bias associated with prediction on a storm, some portion of which has been already seen by the model. A trained model would not have this advantage for real-time forecasting of an entirely new storm absent in the database.

3.6 Hyperparameter tuning

While training a NN, several user defined parameters must be tuned to obtain best performance. These are often chosen manually and tuned by trial and error. The important hyperparameters chosen via tuning, the learning rate, the number of full optimization iterations/ epochs the model is subjected to and the batch size are discussed here.

Learning rate & Epochs: The learning rate is a hyperparameter of the optimizer algorithm which indicates the rate of updating of weights w.r.t. the computed deviation of the loss function for small changes in weights. A high learning rate converges to the optimal weights faster while a small learning rate may require a large number of epochs to converge. However, a large learning rate may result in missing the optimal point. In Keras, the default learning rate for the Adam optimizer is set as 0.001. At this default value, the validation loss was prone to sudden jumps in and around the optimal valley's minimum. In our calculations, to smoothly converge to the optimal weights we used an initial learning rate of 0.0001 for the first 250 epochs. Each epoch represents a full model-weight optimization loop including a forward pass of calculating the predictions based on all the input sequences from the training data and a backpropagation step, in which the model weigths are updated based on the calculated loss function in the forward step. In the subsequent training iterations, models already trained were further trained by reducing the learning rate by an order of magnitude every 250 epochs (for example, for the second set of 250 epochs the learning rate was 0.00001). This is done until the models stop improving.

Batch size: Another important hyperparameter for efficient model training is the batch size. In each epoch, all training time sequences are subjected to the forward model prediction pass once. However, for quicker model convergence, a smaller number of sequences may be used at a time for an optimization loop and model parameters/ weights may be updated. This can be done several times within an epoch. The smaller number of sequences used for updating the model weights one time in each of these sub-epochs is called the batch size. Too large batch size results in smooth gradients computed and averaged over all input sequences. On the other hand, small batch size results in chaotic gradient calculations as-

sociated with properties of small chunks of training data sequences resulting in an irregular path to converged solutions. In our calculations, we used a batch size of 32 which significantly quickened the training process, especially in the Graphic Processing Unit (GPU) clusters. A checking criterion/ checkpoint for model performance was used to check model performance after completion of each epoch. The model weigths were stored in case the validation loss obtained with the updated model weights was lower than the validation loss obtained with the previously stored model weights. At the end of the entire training process, model weights were thus obtained that yielded the lowest loss function value defined by the *m.s.e.* between the predicted and the true LCC coordinates for the validation data sequences.

4. Results

In the following discussion, models are named to reflect their prediction architecture. These models have been named $M2On$ and $M2Mn$, where, $n = 1,2,3,4$ and 5 represents the number of records a model takes in as input. As each timestep in the training database $\equiv 6-hr$, the $M2On$ and $M2Mn$ models forecast 6 and $6n$ hours at once, respectively.

4.1 Mean forecast error

A set of five historical storms were always included in the test dataset. These five storms were chosen on account of their destructiveness upon landfall and of the complexity of their trajectories. The other storms were randomly chosen without replacement for validation and testing. 15% of the storms from the eventual 1665–storm database were chosen for validation purposes and 5% as test storms. Once the models were trained, these were tested on both the validation and test storms. The average error in distance between the predicted and the true positions of a storm's eye was computed at each prediction step.

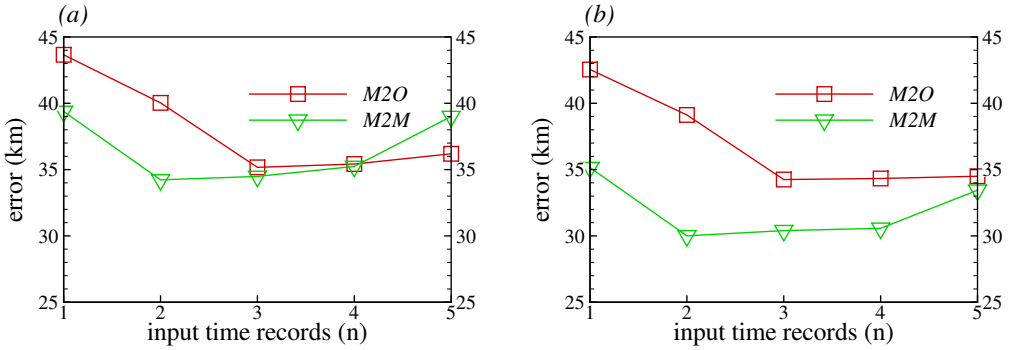

Fig. 10. Average 6–hr forecasting error in distance for the $M2O$ and $M2M$ models computed on (*a*) validation and (*b*) test storms.

Average $6-hr$ forecast error in distance for all models is shown in Fig. 10. The plots in Fig. 10(*a*) and 10(*b*) show the average error computed on the validation and test sequences, respectively. Figure 10 demonstrates that the models have been optimally trained as the validation and test errors are very similar for all models. Overall, the $M2M$ models are more accurate for all n. The $M2M2$ model performs best of all models for $6-hr$ forecasting; mean error computed for the validation and test storms are 34.2 and 30 km, respectively. This is not surprising: authors in [19] noted a near linear statistical relationship between time-rate changes in translation variables at current and previous timesteps, meaning that storm velocity information from two previous instants is likely to produce a more accurate prediction. Although the number of trainable parameters for all models belong in a similar range, the $M2On$ models improve with increase in n, and the $M2Mn$ model predictions worsen slightly with increase in $n > 2$. This is because the $M2M$ models predict several

time instants for an input sequence. During model training for higher values of n, more error is incurred for predictions at later time instants (error monotonically increases with prediction steps). Consequently, while updating model weights, the optimization algorithm emphasizes in reducing the larger error incurred at later prediction steps. However, for $n > 2$, the difference in $6-hr$ forecast error between the $M2On$ and $M2Mn$ models is within 4 km.

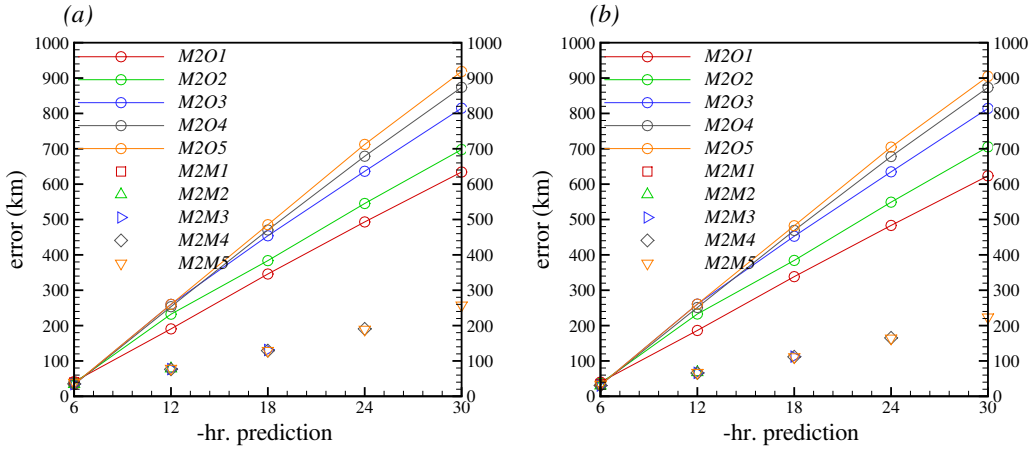

Fig. 11. Average 6, 12, 18, 24 and 30 $-hr$ forecast errors in distance computed on (a) validation and (b) test storms for $M2O$ and $M2M$ models trained on different number of input time records. Errors from the $M2O$ models are represented by circular symbols and the $M2M$ models are represented by other types of symbols. Models trained on a given number of time records are represented by symbols of same color.

For a given number of available time records for a storm, which model predicts a storm's future trajectory more accurately? Figures 11(a) and 11(b) show the mean forecasting errors computed on validation and test sequences, respectively. The $M2Mn$ model can predict n time records at once, i.e., $6n$ hours in advance. For the $M2O$ models however, the predicted record may be used as an input for the next time instant and so on. The forecasting errors for up to 5 time steps in advance obtained in this manner are also reported for all $M2O$ models. For example, for the $M2O5$ model, for a given storm's features of the first 5 time instants were used to predict the storm location at 6^{th} time instant. While predicting the 7^{th} time record of the storm (i.e., the 2^{nd} predicted location), records 2–5 from the original storm history and the record predicted at the 1^{st} prediction step were used as the 5 input time records. Although the LCC projected x– and y– coordinates, V and θ and the $6-hr$ storm displacement probabilities are updated at each time step, the true value of w_m is used from the database. This process is continued until the 10^{th} time record of the storm is predicted to make the model predict 5 time records in advance. This provides a fair comparison between $M2O$ and $M2M$ model accuracies in predicting several time steps in the future. This prediction scenario is similar to a real-time storm trajectory forecasting.

The similarity of the reported mean validation and test errors for each model implies that the models have converged to a global optimum. Due to the aforementioned time record updating procedure for the $M2O$ models, the 12, 18, 24 and 30–hr predictions are prone to compounded error growth. Error incurred at each prediction step worsens the forecasting accuracy at future steps as the predicted coordinates which deviate from the truth are used as input for predicting the future storm locations. Due to the error accumulation, the $M2O$ model forecasts beyond the 1^{st} prediction step worsen for increasing n, as for a higher n more erroneous input time records are fed to the model for predicting the future. All $M2M$ models have similar accuracy over each $6-hr$ interval. Therefore, given an initial time sequence to predict a storm's evolution, especially to predict it several hours in advance, $M2M$ models are more reliable. The average $12-hr$ ($24-hr$) forecasting errors for test sequences are 65.9 km (165.2 km) for the $M2M4$ model and 66.85 km (163.9 km) for the $M2M5$ model, respectively.

4.2 Storm trajectory prediction

Figures 10 and 11 show forecasting errors of the LSTM-RNN models computed over hundreds of validation and test storms. How do these models perform for individual test storms? This section reports trajectory predictions by these models for four extraordinarily powerful test storms: Andrew (1992), Ivan (2004), Sandy (2012) and Harvey (2017). Andrew, Ivan and Harvey were generated in the tropical Atlantic sub-basin; their trajectories exhibit the trend of trajectories of most storms generated in that sub-basin (see Fig. 5). Hurricane Ivan was chosen because of the loopy nature of the trajectory near the end of its lifespan (see Fig. 6). The trajectory of hurricane Harvey has a sharp change toward southeast opposite to the general trend. Hurricane Sandy was generated in the Caribbean sea. Its overall motion is consistent with the general trajectory trend of storms generated in the Caribbean (Fig. 5). However, at the very beginning of its lifespan, it moves southward before turning sharply to travel northward. The aforementioned trajectory trends are complex compared to the general trends of storm motion.

Figure 12 shows the $6-hr$ forecasts by the $M2O$ models for the four hurricanes. Forecasting error was computed at all prediction instants and the mean forecasting error (per prediction step) was computed over the whole trajectories which are reported for all $M2O$ models. Exact features were used as inputs to the model. $M2O1$ performs the worst of all models; error computed for Andrew and Harvey is less than 47 km and 34 km; for Sandy less than 50 km;and for Ivan less than 52 km) for the $M2O1$ model. Model performance significantly improves for $n > 2$. Among these three models, despite their complex trajectories, maximum of mean $6-hr$ forecasting error computed for each of these storms were $\approx$ 28 km, 40 km, 29 km and 39 km for the Andrew, Ivan, Harvey and Sandy.

The same four storm trajectories predicted by the $M2Mn$ models have been shown in Fig. 13. To obtain predictions by a chosen $M2M$ model for a given storm, exact input features at n instants have been fed to the model to predict the storm locations at next n time instants at $6-hr$ intervals. So, the input time sequences are as $i = (0 \rightarrow n-1), (n \rightarrow 2n-$

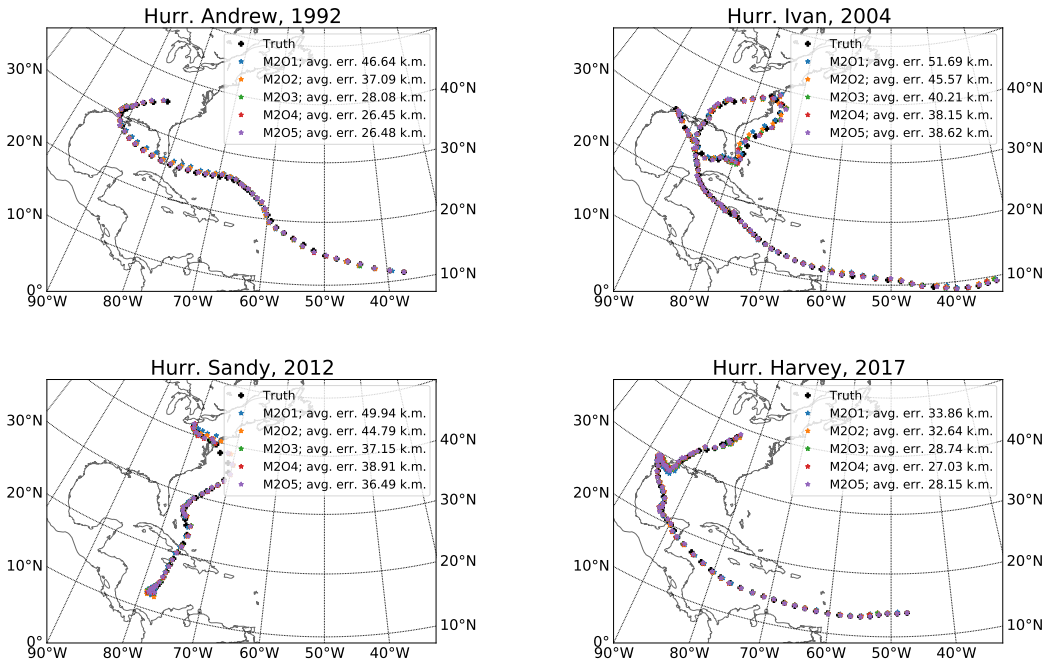

Fig. 12. 6–*hr* forecast of selected hurricane trajectories; Black '+' symbols represent the true locations; Colored '*' symbols represent the *M*2*On* model forecasts. Inputs to the model are the true feature data at each timestep. Average error computed over the whole trajectory is also given.

$1),(2n \to 3n-1),\ldots$. Corresponding output sequences are $i = (n \to 2n-1),(2n \to 3n-1),(3n \to 4n-1),\ldots$. Each model predicts a target time record only once. The mean error in each prediction step computed over the whole trajectories for each model is also reported. It should be noted that for models with $n > 1$, the computed error is the forecasting error per prediction which is different from the mean $6n-hr$ forecast errors in Fig. 11. Although the forecasting error for the *M*2*Mn* model is highest at prediction step n, the reported error takes into account the model's more accurate predictions at the earlier prediction steps. The computed mean error per prediction step is smallest for Harvey and largest for Sandy. Mean errors computed over 12 and 24 hour forecasts for the *M*2*O*2 and *M*2*O*4 models for hurricane Harvey are 38.76 and 76.2 km. For hurricane Sandy these are $\approx$ 63.8 and 128.2 km, respectively.

A hybrid approach is proposed for real-time storm evolution forecasting with the current models. When only one record is available for a storm, a model trained on features of a single time instant is used for prediction. When records of two time instants are available, a model trained on two time records may be used. Similarly, models trained on increasing numbers of input time records can be used for prediction, up to the point where 5 time records are available. A model trained on 5 input time records is used to predict the remaining life span of the storm's evolution.

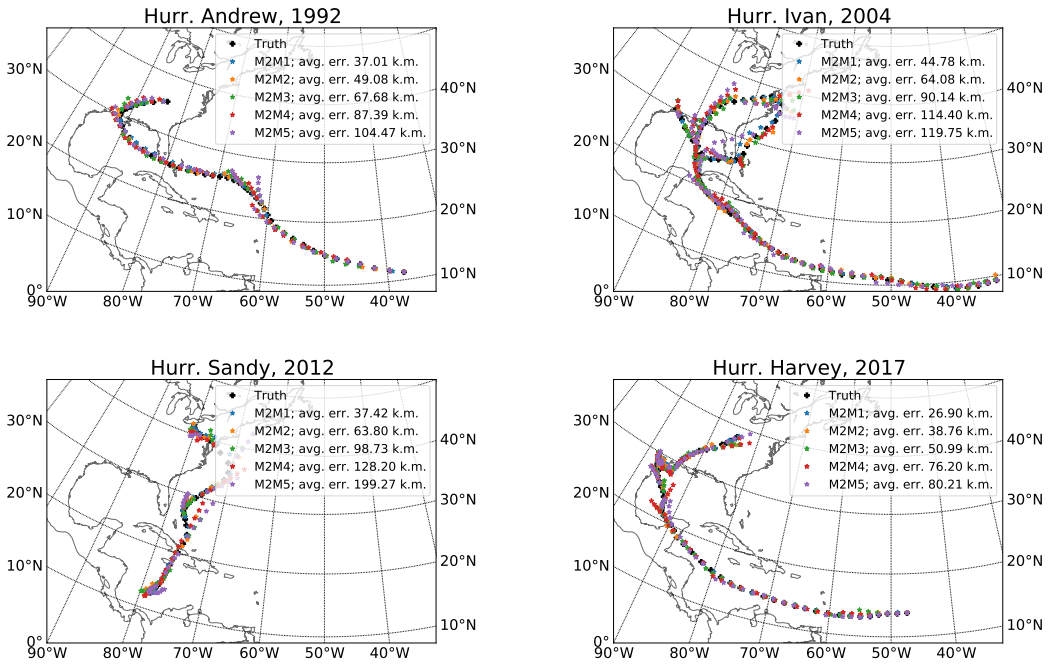

Fig. 13. $6n-hr$ forecasts of selected hurricane trajectories; Black '+' symbols represent the true locations; Colored '*' symbols represent the $M2Mn$ model forecasts. Inputs to the model are the true feature data at each timestep. Average error computed over the whole trajectory is also given.

4.3 Limitations

For real-time forecasting of storm trajectories, only the available time records from storm's current and past status may be used. Previous discussion on performance of the LSTM-RNN models is based on a Machine/ Deep Learning perspective on the best/ ideal performance of the models for the problem being considered. The models were only fed the true features and these only predict up to 30 hours in advance. However, longer forecast of storm trajectories may be necessary for disaster management purposes. To test these models' capability for this purpose, only the first input sequence of a storm's trajectory was fed to the model and the whole trajectory was predicted by the $M2On$ and $M2Mn$ models. Inputs to the models from the second prediction sequence onwards include the predicted storm locations and corresponding derived storm speed, direction and displacement probabilities in previous time instants (only the true value for w_m was used in all sequences). The trajectories for the four chosen test storms are shown in Figs. 14 and 15. This is equivalent to a $228-hr$ forecast by the $M2M1$ model for hurricane Sandy with the shortest lifespan among the four test storms. Deviations of the predicted storm trajectories from the true trajectories in these figures are basically illustrations of the compounded error growth previously discussed. Computed error in distance at each time step averaged over the whole trajectory is also reported for all the models.

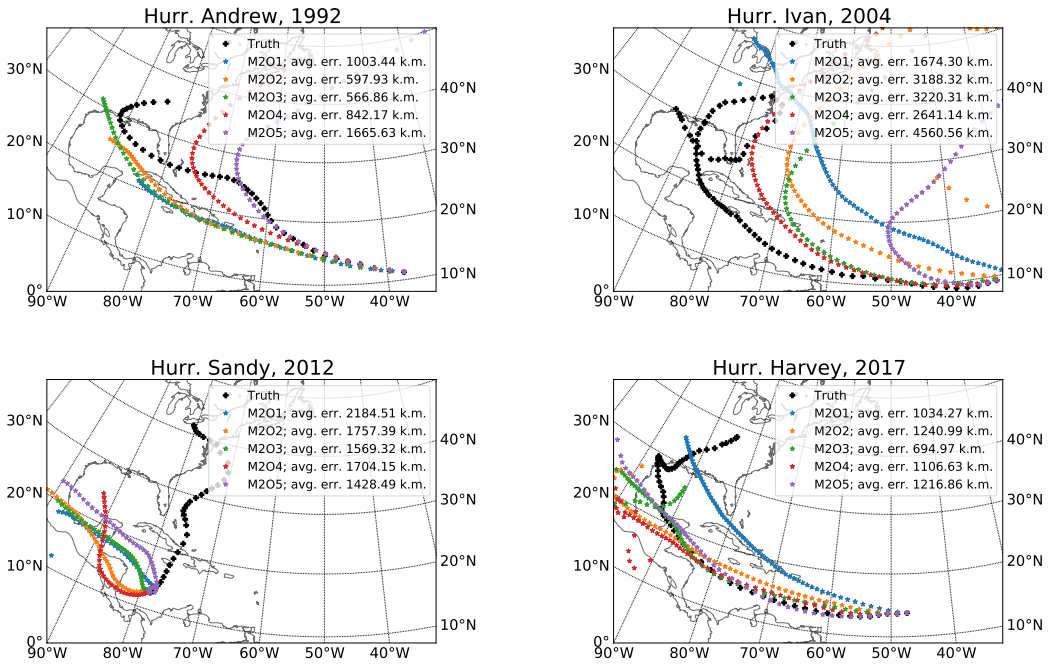

Fig. 14. Forecast of whole trajectories of pre-selected hurricanes given only the 1^{st} sequence; Black '+' symbols represent the true locations; Colored '*' symbols represent the *M2On* model forecasts. 2^{nd} prediction sequence onwards, inputs to the models are based on predictions at previous prediction steps. Average error calculated over the whole trajectory is reported.

The reported errors for both sets of models are in most instances in thousands of kilometres for each predicted time step. The predicted trajectories in the initial stages of hurricane evolution are significantly closer to the true trajectories compared to later on in a storm's lifespan. This is to be expected, however, as the compounded errors incurred at each prediction step worsen the forecast at the next prediction step. Consequently, the predictions veer off from the true trajectories. None of the models is able to predict the complex loop in Ivan's trajectory. Similar deviations are obtained for the trajectory of hurricane Sandy. This storm initially travels southward and then turns about 180° to travel northward. Although most models correctly predict the eventual northward motion of the storm at least qualitatively, the models are unable to predict the sudden change in direction of translation. The models perform better in predicting the relatively simpler trajectories of hurricanes Andrew and Harvey. However, the forecast errors are still very large.

The limitation of the LSTM-RNN models in predicting a whole storm trajectory from an initial condition is not surprising. Starting from it's genesis, a storm's evolution is a complex nonlinear phenomenon dependent on several fluid and thermodynamic effects. This remains one of the most challenging problems in Geophysical Fluid Dynamics. For example, in [25], testing several ensemble prediction systems (EPSs) for forecasting North

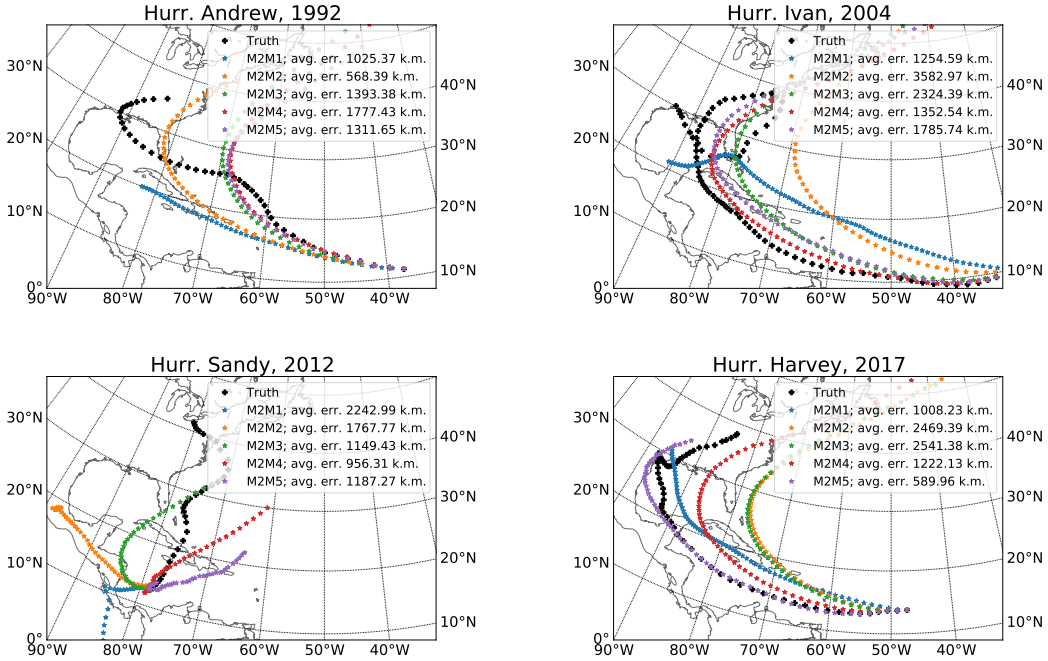

Fig. 15. Forecast of whole trajectories of pre-selected hurricanes given only the 1^{st} sequence; Black '+' symbols represent the true locations; Colored '*' symbols represent the *M*2*Mn* model forecasts. 2^{nd} prediction sequence onwards, inputs to the models are based on predictions at previous prediction steps. Average error calculated over the whole trajectory is reported.

Atlantic hurricane trajectories, the minimum 24–*hr* total track error (same as the error in distance used in the present work) increased by ≈ 47–77 *mi* (75.64 – 123.92 km) for every $24-hr$ prediction interval for a $120-hr$ forecast for all storms in the period 2008 – 2015. Mean $24-hr$ forecast error for the test storms were ≈ 165 and 164 km for our *M*2*M*4 and *M*2*M*5 models (Fig. 11). It should be noted that the EPSs are physics-based models that are guided by accurate fluid dynamic simulations, and therefore use a wealth of information unavailable to the current data-based models. Furthermore, following the current methodology, *M*2*Mn* models with $n > 5$ may be developed to further reduce long-term forecast errors due to compounded error accumulation associated with long-term trajectory forecasting.

5. Conclusions

A family of Long short-term memory (LSTM) Recurrent Neural Network (RNN) models has been developed for predicting hurricane trajectories in the North Atlantic basin. Both Many-To-One and Many-To-Many type prediction models were trained validated and tested. The input features are a storm's Lambert's conic conformal (LCC) projected x and y coordinates, storm translation speed and direction, the maximum 1–min wind speed at 10 m elevation and a storm's 6–hr displacement probabilities in a $2-D$ grid laid over the considered domain in the x–y plane. $6-hr$ storm displacement probabilities feed the models with historical storm trajectory trends at a given location. The database used for the model development is the HURDAT2 maintained by NOAA. A set of 1665 storms out of the 1893 available storm tracks maintained in the HURDAT2 were chosen to exclude statistical outliers and also facilitate feature engineering.

The proposed model architectures are able to forecast up to 30 hours in advance. Forecasting errors computed and averaged over three hundred of validation and test storms demonstrate the efficacy of the presented models (Figs. 10 and 11). The minimum mean $6-hr$ forecasting error of all models was ≈ 30 km. Overall, the $M2M$ models are more accurate. The $M2On$ models are of comparable accuracy for short-term forecasting (Fig. 10). Considering that the storm-eye radii may extend up to 80 km and storm radii may be as large as hundreds of kilometres [26], the model forecasts are reasonably accurate. The $M2M$ models are more accurate at earlier prediction steps; the prediction error increases linearly between prediction steps. $M2On$ models are more error-prone due to compounded error accumulation for long-term predictions. The compounded error accumulation may be avoided by using the $M2Mn$ models for prediction. Present work demonstrates that $M2Mn$ prediction models may be trained for higher $n > 5$ to forecast $6n$ hours with significantly more accuracy compared to $M2On$ models. Both sets of models were tested to predict evolution of four historically significant storms with complex trajectories. The minimum and maximum mean $6-hr$ forecast errors for the $M2O$ ($M2M$) models in predicting four chosen complex-trajectory test hurricanes were 26.45 and 51.7 km (27 and 37 km) (Figs. 12, and 13). Because $M2M5$ ($M2M1$) model forecasts were for 30 (6) hours, the maximum (minimum) mean error computed for each prediction step over whole trajectories was obtained for this model. For hurricanes Andrew, Ivan, Sandy and Harvey, the mean forecast error per prediction step for the $M2M5$ ($M2M1$) model were $\approx$ 104 km, 80 km, 120 km and 200 km (37 km, 27 km, 45 km and 37 km), respectively.

The most significant advantage of the models presented in this work is their fast forecasting capability using modest computational resources. Unlike previous attempts [11–13], once trained, these models require minimum available information for any test storm for making predictions. A maximum of $30-hr$ storm evolution data is required, however, these are capable of providing predictions for as little as only one time record for any storm. Other models based on statistics and fluid dynamics used by the NHC require relatively large computational resources and a wealth of information for making predictions. For example the dynamical models require using modern supercomputers to solve

the equations governing atmospheric flow conditions, and therefore, may require hours to provide even a $6-hr$ forecast. To our knowledge, the models presented in this paper are the most accurate of all data-based NN models developed so far, even compared to the image processing NN models for forecasting at least up to 12 hours. Comparison of storm trajectory forecasting errors in the ensemble based EPS models presently used by NHC (which uses various other models, including statistical, dynamical or trajectory models) with the LSTM-RNN models presented in this work shows similar prediction errors even for $30-hr$ forecasting. The models developed herein are currently being used for the simulation of hurricane tracks and features that statistically emulate the HURDAT2 database.

Acknowledgments

R.B. served as a NIST Director's Postdoctoral Researcher.

References

[1] Neumann CJ (1972) An alternate to the hurran (hurricane analog) tropical cyclone forecast system .

[2] Kurihara Y, Bender MA, Tuleya RE, Ross RJ (1995) Improvements in the gfdl hurricane prediction system. *Monthly Weather Review* 123(9):2791–2801.

[3] Buizza R, Milleer M, Palmer TN (1999) Stochastic representation of model uncertainties in the ecmwf ensemble prediction system. *Quarterly Journal of the Royal Meteorological Society* 125(560):2887–2908.

[4] Richardson DS (2000) Skill and relative economic value of the ecmwf ensemble prediction system. *Quarterly Journal of the Royal Meteorological Society* 126(563):649–667.

[5] Schultz M, Betancourt C, Gong B, Kleinert F, Langguth M, Leufen L, Mozaffari A, Stadtler S (2021) Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A* 379(2194):20200097.

[6] Giffard-Roisin S, Yang M, Charpiat G, Kégl B, Monteleoni C (2018) Deep learning for hurricane track forecasting from aligned spatio-temporal climate datasets .

[7] Kim S, Kim H, Lee J, Yoon S, Kahou SE, Kashinath K, Prabhat M (2019) Deephurricane-tracker: Tracking and forecasting extreme climate events. *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (IEEE), , pp 1761–1769.

[8] Rüttgers M, Lee S, Jeon S, You D (2019) Prediction of a typhoon track using a generative adversarial network and satellite images. *Scientific reports* 9(1):1–15.

[9] Chen Z, Yu X (2020) A novel tensor network for tropical cyclone intensity estimation. *IEEE Transactions on Geoscience and Remote Sensing* .

[10] Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural computation* 9(8):1735–1780.

[11] Moradi Kordmahalleh M, Gorji Sefidmazgi M, Homaifar A (2016) A sparse recurrent neural network for trajectory prediction of atlantic hurricanes. *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, , pp 957–964.

[12] Alemany S, Beltran J, Perez A, Ganzfried S (2019) Predicting hurricane trajectories using a recurrent neural network. *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33, pp 468–475.

[13] Lian J, Dong P, Zhang Y, Pan J, Liu K (2020) A novel data-driven tropical cyclone track prediction model based on cnn and gru with multi-dimensional feature selection. *IEEE Access* 8:97114–97128.

[14] Jeffries RA, Miller RJ (1993) Tropical cyclone forecasters reference guide. 3. tropical cyclone formation (NAVAL RESEARCH LAB MONTEREY CA),

[15] Demaria M, Aberson SD, Ooyama KV, Lord SJ (1992) A nested spectral model for hurricane track forecasting. *Monthly Weather Review* 120(8):1628–1643.

[16] Jarvinen BR, Neumann CJ, Davis MA (1984) A tropical cyclone data tape for the north atlantic basin, 1886-1983: Contents, limitations, and uses .

[17] Landsea CW, Franklin JL (2013) Atlantic hurricane database uncertainty and presen-

tation of a new database format. *Monthly Weather Review* 141(10):3576–3592.

[18] Lambert JH (1972) *Notes and Comments on the Composition of Terrestrial and Celestial Maps (1772).* 8 (Department of Geography, University of Michigan), .

[19] Emanuel K, Ravela S, Vivant E, Risi C (2006) A statistical deterministic approach to hurricane risk assessment. *Bulletin of the American Meteorological Society* 87(3):299–314.

[20] Schwerdt RW, Ho FP, Watkins RR (1979) Meteorological criteria for standard project hurricane and probable maximum hurricane windfields, Gulf and East Coasts of the United States .

[21] Parasuraman K (2020) Hurricane florence — building a simple storm track prediction model. Available at https://towardsdatascience.com/hurricane-florence-building-a-simple-storm-track-prediction-model-1e1c404eb045.

[22] Gómez P, Nebot A, Ribeiro S, Alquézar R, Mugica F, Wotawa F (2003) Local maximum ozone concentration prediction using soft computing methodologies. *Systems analysis modelling simulation* 43(8):1011–1031.

[23] Karpathy A, Johnson J, Fei-Fei L (2015) Visualizing and understanding recurrent networks. *arXiv preprint arXiv:150602078* .

[24] Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:14126980* .

[25] Leonardo NM, Colle BA (2017) Verification of multimodel ensemble forecasts of north atlantic tropical cyclones. *Weather and Forecasting* 32(6):2083–2101.

[26] Willoughby HE, Darling R, Rahn M (2006) Parametric representation of the primary hurricane vortex. part ii: A new family of sectionally continuous profiles. *Monthly weather review* 134(4):1102–1120.