FH JOANNEUM
Data and Information Science

MASTER'S THESIS

Multivariate forecasting of tropical cyclones using combined neural networks.

Submitted at:
FH JOANNEUM Gesellschaft mbH
Master's Degree Programme
"Data and Information Science"

Author
Yegor HUDOZHNIK
52100061
Jahrgang DAT 2021

Supervisor
Ing. Mag. Dr. Andreas WINDISCH

Graz, 31. 12. 2023 Signature

Signed Declaration

I hereby declare that the present Master's thesis was composed by myself and that the work contained herein is my own. I also confirm that I have only used the specified resources. I also confirme that I have complied to the guideline of FH JOANNEUM for ensuring good scientific practice and for avoidance of misconduct. All formulations and concepts taken verbatim or in substance from printed or unprinted material or from the Internet have been cited according to the rules of good scientific practice and indicated by footnotes or other exact references to the original source.

The present thesis has not been submitted to another university for the award of an academic degree in this form. This thesis has been submitted in printed and electronic form. I hereby confirm that the content of the digital version is the same as in the printed version.

I understand that the provision of incorrect information may have legal consequences.

(Signature) (Place, Date)

Abstract:

Tropical cyclones (TCs) are extremely dangerous and destructive events that pose a threat to human life every year. Since the beginning of the meteorological observation era, predicting the behavior of cyclones has always been an issue. It has been proven that with climate change due to global warming, the proportion of stronger TCs increases, increasing the danger and potential harm of TCs.

Numerous techniques have been developed over the years and are used in ensembles to detect, predict and classify TCs. Nevertheless, the tasks in the field of TC prediction are considered challenging because the development of TC systems exhibits nonlinear behavior and depends on many environmental factors.

Traditional DL prediction methods are computationally intensive and require a relatively large amount of energy and time. Due to ongoing global warming, the behavior of TCs may constantly change and therefore requires the use of modern, environmentally friendly and more flexible learning methods for estimating and predicting the future behavior of TCs.

In recent years, the study of the application of deep learning (DL) methods in this area has proven to be very effective. These methods are designed to facilitate the forecasting process and automatically detect and adapt to possible trends that may arise over time. DL methods are able to strengthen the ability to predict based on actual behavior and take into account the evolution of recent trends in the movement and intensification of TCs.

DL methods are not intended to replace conventional modeling and prediction methods and cannot replace them because conventional methods deal with the actual formulation of physics, while DL methods provide the most modern statistical analysis and can thus exert their influence on research.

In our research, we have applied a novel approach of incorporating two-dimensional meteorological data to forecast the track and intensity of TCs. We have built and tested numerous sequence-to-sequence forecasting models based on ConvLSTM2D neural network layers and tested two-dimensional data compression using autoencoders as a data preparation technique. Our experiments have shown that the multivariate forecast yields perspective results. However, further research is required to address the nature of rare events, such as rapid intensification and sudden track change. We have also succeeded in detecting the influence of recent trends in TC behaviour changes in recent years and proved the ability of neural networks to fit themselves to those trends.

Keywords: Tropical Cyclones · Deep Learning · ConvLSTM2D · Sequence-to-sequence · Autoencoderes

Kurzfassung:

Tropische Wirbelstürme (TCs) sind extrem gefährliche und zerstörerische Ereignisse, die jedes Jahr eine Gefahr für Menschenleben darstellen. Seit dem Beginn der meteorologischen Beobachtungsära war die Vorhersage des Verhaltens von Wirbelstürmen immer ein Thema. Es ist erwiesen, dass im Klimawandel aufgrund der globalen Erwärmung der Anteil der stärkeren TC, zunimmt, wodurch die Gefahr und der potenzielle Schaden von TCs steigen.

Zahlreiche Techniken wurden im Laufe der Jahre entwickelt und werden in Ensembles eingesetzt, um TCs zu erkennen, vorherzusagen und zu klassifizieren. Dennoch werden die Aufgaben im Bereich der TC-Vorhersage als herausfordernd angesehen, da die Entwicklung von TC-Systemen ein nichtlineares Verhalten aufweist und von vielen Umweltfaktoren abhängt.

Herkömmliche DL-Vorhersagemethoden sind rechenintensiv und erfordern einen relativ hohen Energie- und Zeitaufwand. Aufgrund der fortschreitenden globalen Erwärmung kann sich das Verhalten von TCs ständig ändern und erfordert daher den Einsatz moderner, umweltfreundlicher und flexiblerer Lernmethoden für die Abschätzung und Vorhersage des zukünftigen Verhaltens von TCs.

In den letzten Jahren hat sich die Untersuchung der Anwendung von Deep Learning (DL)-Methoden in diesem Bereich als sehr effektiv erwiesen. Diese Methoden wurden entwickelt, um den Vorhersageprozess zu erleichtern und mögliche Trends, die im Laufe der Zeit auftreten können, automatisch zu erkennen und sich an sie anzupassen. DL-Methoden sind in der Lage, die Fähigkeit zur Vorhersage auf der Grundlage des tatsächlichen Verhaltens zu verstärken und die Entwicklung der jüngsten Trends bei der Bewegung und Intensivierung von TCs zu berücksichtigen.

DL-Methoden sind nicht als Ersatz für konventionelle Modellierungs- und Vorhersagemethoden gedacht und können diese auch nicht ersetzen, denn die konventionellen Methoden befassen sich mit der eigentlichen Formulierung der Physik, während die DL-Methoden die modernste statistische Analyse bieten und so ihren Einfluss auf die Forschung ausüben können.

In unserer Forschung haben wir einen neuartigen Ansatz angewandt, der zweidimensionale meteorologische Daten zur Vorhersage der Zugbahn und Intensität von TCs einbezieht. Wir haben zahlreiche Sequence-to-sequence Vorhersagemodelle auf der Grundlage von ConvLSTM2D-Netzschichten entwickelt und getestet und die zweidimensionale Datenkomprimierung mit Hilfe von Autoencodern als Datenaufbereitungstechnik erprobt. Unsere Experimente haben gezeigt, dass die multivariate Vorhersage aussichtsreiche Ergebnisse liefert. Es sind jedoch weitere Forschungen erforderlich, um die Natur seltener Ereignisse, wie z. B. eine schnelle Intensivierung und plötzliche Zugveränderungen, zu berücksichtigen. Es ist uns auch gelungen, den Einfluss der Trends bei den Veränderungen des TC-Verhaltens in den letzten Jahren zu erkennen und die Fähigkeit der neuronalen Netze zu beweisen, sich an diese Trends anzupassen.

Schlüsselwörter: Tropical Cyclones · Deep Learning · ConvLSTM2D · Sequence-to-sequence · Autoencoders

Contents

List of Tables

List of Figures

List of Acronyms

DL	Deep Learning	5
NOAA	National Oceanic Atmospheric Administration	2
NN	Neural Network	5
RNN	Recurrent Neural Networks	5
LSTM	Long Short-Term Memory	5
GRU	Gated Recurrent Units	5
GAN	Generative Adversarial Networks	5
CNN	Convolutional Neural Network	5
ADT	Advanced Dvorak Technique	8
LSR	Latent Space Representation	17
RI	Rapid Intensification	1
TF	Tensorflow	13
CEC	Constant Error Carousel	9

FC-LSTM Fully Connected LSTM	10
Rainfall-MSE Rainfall Mean Squared Error	10
NWP Numerical Weather Prediction	10
ReLu Rectified Linear Unit	18
MSE Mean Squared Error	19

Chapter 1

Introduction

1.1 Problem statement

The process of weather forecasting includes many technical processes and human-taken decisions. It is believed that the atmosphere has a high degree of randomness, and understanding atmospheric processes needs to be improved. This initially makes the whole weather forecast process very complex and challenging. It starts with collecting observation data from different sources, like surface observations, satellite sensors and imagery, aircraft missions, radars, radiosondes, offshore buoys, ships at sea and others. Then, the collected data is used to run weather simulations, construct reanalysis and update climate simulations. In the end, the various results are combined and examined by professional forecasters who, in turn, apply a pattern recognition process to it. The whole process consists of many other complex sub-processes. It depends on the ability to collect data, interpolate the missing gaps, model the behavior of the atmosphere in the correct way and derive decisions based on human expertise.

The tasks in the field of TC forecasting are perceived as especially challenging, because the development of TC systems features non-linear behavior [12] and depends on many environmental factors. Numerous techniques were developed throughout the years and are used in ensembles with weather simulations to detect, forecast and classify TCs. Conventional TC forecasting methods are computationally intensive and require a relatively large amount of energy and time.

Existing studies show, that due to the ongoing global warming, certain changes in TC behavior take place. For example, in the North Atlantic and North Pacific basins, TCs move to the north [22]. There is also an increase in the proportion of strong TCs of category 4 and 5 at a rate of between 25% and 30% per °C of global warming, balanced by a decrease in proportion of category 1 and 2 TCs among all ocean basins [7]. Moreover, it has been shown that there is a recent upward trend in Rapid Intensification (RI)[1] rates[10]. The observation of this trend is sometimes attributed to the improvement of the observation technology[19] and not related to the local atmospheric environment, but the warming ocean[20] and global warming still might be the reason. [1] show that the upward trend is positively correlated with anthropogenic forcing in multiple TC basins. The behavior of TCs may keep changing

[1]RI is defined as the 95th percentile of over-water 24-h intensity changes[9], which corresponds to an increase of the wind speed of at least 30 knots or 55 km/h.

and even increase, therefore require the use of modern, environment-friendly, and more flexible learning methods for estimation and forecasting.

DL methods are empowered to enforce the ability to forecast, based on actual observed behavior, and consider the development of the most recent trends in changes of tracks and intensification of TCs if the data is up to date and abundant. They are intended to be different from conventional modelling and forecasting methods because conventional methods deal with the formulation of physics and precise understanding of relationships of different atmospheric factors. However, DL Methods have the potential to have a significant impact on climate and weather research because they provide the most modern statistical analysis. Ultimately, these two approaches can component themselves by being used in conjunction as a hybrid model that takes physics and data analytics into account.

1.2 Nature of Tropical Cyclones

Basic characteristics of the TC formation process are playing a central role in the explanation of the chosen data and methods in our work. There are several weather conditions which create favourable, but not sufficient conditions for the formation of a Tropical Cyclogenesis [2]:

- Warm oceans with sea surface temperature higher than 26.5°C up to a depth of at least 45 meters and a cool atmosphere above it, so the difference in temperatures creates thunderstorm conditions.
- High humidity in the mid-troposphere (700 hPa[3]) is needed to preserve the circulation.
- Location of the storm of at least 500 km away from the equator, where the Coriolis force[4] is strong enough to deflect the movement of a TC, so the rotation is strong enough to develop a low pressure center.
- Low vertical wind shear. Wind shear is a difference in wind speed or direction, in our case between the surface and the upper troposphere. Low vertical wind shear allows convection of the air between different altitudes. High wind shear will disturb the convection.

When these conditions occur, it is likely that a TC system will form and evolve. Basic characteristics of this process is what we are going to forecast in our work. The following steps in formation of a TC[5] are conventionally being referenced to:

1. Tropical disturbance:
   Water evaporates from the warm ocean and raises as a column of clouds. When the air gains altitude it cools down and falls, starting to rotate.

[2]National Oceanic Atmospheric Administration (NOAA). (2023, June 1) Hurricanes FAQ, How do Tropical cyclones form? https://www.aoml.noaa.gov/hrd-faq/#tc-formation

[3]The standard air pressure at sea-level is 1013.25 hPa. The pressure of 700 hPa usualy corresponds to the altitude of 3012 meter above the sea level.

[4]Britannica: The effect of the Coriolis force is an apparent deflection of the path of an object that moves within a rotating coordinate system. It is equal 0 at the equator and is maximal at the poles.

[5]NOAA. (2023, July 3) How Does a Hurricane Form? https://scijinks.gov/hurricane/

2. Tropical depression (Wind speed 40-61 km/h):
   Warm air at high altitude causes high pressure that causes the winds to move outward. That makes the pressure at the surface drop. Air at the surface moves to the low pressure area, rises and intensifies the system.

3. Tropical storm (Wind speed 62-119 km/h):
   The wind begins to twist around the eye center due to the Coriolis force.

4. Hurricane (Wind speed > 119 km/h):
   The storm is at least 15 km high and around 200 kilometers across. The eye is around 9 to 55 km wide. Trade winds which blow from east to west at the North Atlantic Basin push TCs to the Gulf of Mexico. The winds and the low pressure cause the water level in the eye to rise, which makes the landfall more dangerous because of the amount of water.

5. Decay:
   Hurricanes weaken once they have reached a landfall, because they don't have the support of warm waters to feed energy and the friction with the ground increases. They also weaken when they reach middle latitudes with cool sea surface temperature and strong wind shear that breaks the circulation.

One of the common ways to categorize Hurricanes and their potential damage is the Saffir-Simpson Hurricane Wind Scale. The scale ranks hurricanes into five categories based on their sustained wind speeds:

- Category 1: Wind speeds of 119-153 km/h. Potential damage to weak buildings, signs, trees and some damage to power infrastructure.
- Category 2: Wind speeds of 154-177 km/h. Damage to building windows and power outages for up to several days.
- Category 3: Wind speeds of 178-208 km/h. Severe damage to weak constructions and uprooted trees. Power outages for up to several weeks.
- Category 4: Wind speeds of 209-251 km/h. Some roof failures and severe damage to windows and doors.
- Category 5: Wind speeds of 252 km/h or higher. Complete roof and building failures. Total damage to power infrastructure with outages lasting for several months.

Category 4 and 5 hurricanes are considered major hurricanes due to their destructive potential and the widespread damage they can cause.

Chapter 2

Literature Overview

This section reviews previous work on closely related Deep Learning (DL) tasks in the field of TCs and investigates the advantages or disadvantages of different methods. It also introduces literature related to the Neural Network (NN) techniques, which we are going to use in our work.

The research of application of DL methods in the field already uses a variety of NN models and supplementing techniques. The uses are mainly divided into two tasks. Classification of data snapshots for the presence or absence of TC, which is mainly handled with Computer Vision NNs: Convolutional Neural Network (CNN) [2], YOLOv3 [14], DeepLabv3+ [15], and a track prediction task, with the application of Recurrent Neural Networks (RNN) [13], Long Short-Term Memory (LSTM) [25], Gated Recurrent Units (GRU) [25], Generative Adversarial Networks (GAN) [17], sometimes combined with CNN for feature extraction [25] [16].

In the reviewed literature for solutions to classification tasks, authors use combinations of data sources such as meteorological reanalysis, satellite imagery, and historical track records. Meanwhile, for track prediction, the use of meteorological or imagery data is rare, and in most cases, they use only historical track records.

In our work, we would like to combine the knowledge from both DL tasks and use technology and the advantages of computer vision to build a strong forecasting method. Therefore, we conduct our final methodology from different DL tasks.

2.1 Choice of the DL task and data

2.1.1 Gardoll and Boucher [2]

The first article considers the performance and sensitivity of a CNN binary classifier to the learning dataset. The authors accurately document their data preparation process, including producing a Python library to help organize datasets in order to present an easily repeatable way of handling the problem in order to help future research. The training labels are images centred on the cyclone positions, combining HURDAT2[1] hurricane tracks

[1]US National Oceanic & Atmospheric Administration Reanalysis Hurricane database V.2.

dataset with measures from ERA5[2] and MERRA2[3] meteorological reanalysis. Background images sample locations and times similar to the TC-containing images and do not include any TC by construction.

Their model showed very accurate results in using a very short time to train. Their research question was to compare the influence of the choice of training dataset vs. prediction dataset. This question is out of our scope. So, we will not consider investigating it. However, we find their choice of datasets very reasonable and modern because the meteorological reanalysis databases provide consistent and complete data on multiple weather variables. Datapoints are distributed on a high-resolution grid and provide a map with no gaps. One of the purposes of these datasets is to train weather models, with which numerical forecasting methods are initialized[4]. The use case corresponds to the main idea of our work, which is to use various data and implement and improve an NN pipeline that predicts an informative forecast.

2.1.2 Ren et al. [16]

The second article deals with the track prediction of typhoons formed in the South China Sea. A C-LSTM model is implemented as a combination of CNN and LSTM. The data used was the best track set of typhoons IBTrACS [5] in the northwest Pacific Ocean, which includes positional and meteorological variables.

First, the authors describe precisely the data preparation and cleansing process and apply the Granger Causality Test to select the variables with more significant influence on typhoon tracks to reduce the dimensionality and clean irrelevant data. The authors group the data of longitude, latitude, and 7 chosen meteorological variables into time series sequences of a fixed length of 20 samples per typhoon. For each typhoon sequence, they group every five data samples into a matrix and determine the subsequent matrices using the sliding window method. In the NN architecture, convolutional layers extract feature vectors from the input data, and LSTM layers learn temporal features and output the location of the TC system for the next time step.

This work concludes that CNN layers help drastically to improve the performance of LSTMs.

2.1.3 Wang et al. [25]

Wang et al. present a brief history of the application of DL methods in the field of TC forecasting and propose a new method of TC track forecast based on a combination of CNN and GRU. They use combined data from the IBTrACS track dataset for the Northwest Pacific Ocean and ERA5. The authors bring to a test 19 movement characteristics extracted from IBTrACS to discover which are the most meaningful. Based on the field knowledge,

[2]European Centre for Medium-Range Weather Forecasts, Reanalysis V.5.

[3]US National Aeronautics and Space Administration, Modern-Era Retrospective analysis for Research and Applications V.2.

[4]ECMWF. (2019, October). Use of ERA5 reanalysis to initialise re-forecasts proves beneficial. https://www.ecmwf.int/en/newsletter/161/meteorology/use-era5-reanalysis-initialise-re-forecasts-proves-beneficial

[5]US National Oceanic & Atmospheric Administration International Best Track Archive for Climate Stewardship data.

they choose from ERA5 the following variables: U and V components[6] of the wind at four pressure altitudes[7], Sea Surface Temperature, and Geopotential Height[8] at three pressure altitudes.

The authors perform sequence-to-sequence forecasting, taking information from the first 24h of TC development and predicting the next 6h-72h. Train (1979-2014), validation (2015-2018) and test (2019-2021) sets are divided by years, without shuffling and TCs that exist for more than 96h are removed from the data.

First, the authors implement basic RNN, LSTM, and GRU models and compare the 11 most important movement features. The GRU model shows the best result with a slight advance over the LSTM.

Then, a combined network, GRU_CNN, was presented. The architecture features four separate data streams for the variables: the meteorological variables go through CNN networks with alternating max-pool and convolutional layers, and IBTrACS scalar data goes through a sequence of two stacked GRU layers. The outputs of each data stream go to separate dense layers, get concatenated, and are merged through an additional dense layer. All layers are normalized with batch normalization, which, according to [8], accelerates the training of NNs and helps with the vanishing or exploding gradient problem. The authors justify using different data branches, because of the chosen different dimensionality of the ERA5 variables and the different nature of the variables requiring different learning rates.

The GRU_CNN network enhances the performance compared to their basic and competitive models. It also outperforms the Central Meteorological Observatory 6h-126 short-term forecasts. Separate training shows that the U and V components of the wind were the most significant 2D variables and played a dominant role in the <24h forecast. The results in this time range are similar to that of a model trained on all meteorological variables. Adding Sea Surface Temperature and Geopotential Height improved gradually 48h-72h mid-term forecasts.

Further variables such as cyclone intensity, rainfall, and wind speed are proposed to be used for prediction in future research.

The paper concludes that it is essential to enlarge the number of prediction variables to make the approach applicable in real life and suggests to consider further variables such as cyclone intensity, rainfall, and wind speed. Therefore a set of variables, similar to the work of Gardoll and Boucher [2], could indeed be meaningful and bring an improvement.

2.1.4 Rüttgers et al. [17]

The work of Rüttgers et al. caught our interest due to the use of an advanced neural network and pure satellite photos, especially the ability to predict cloud formation, which gives a better natural picture for real-life forecasting. The work uses satellite imagery (photos) and GAN to implement a track and cloud formation prediction.

[6]Positive U component of the wind comes from the west, negative from the east. V component positive comes from the south and negative from the north.

[7]Pressure altitude is the height above a theoretical level of Earth's surface, where the air pressure is 1,013.2 bar.

[8]Geopotential Height is a vertical coordinate referenced to Earth's mean sea level that represents the work involved in lifting one kilogram of mass over one meter of height with constant acceleration of gravity.

The results of this work propose that satellite imagery is insufficient to predict the tracks in an applicable manner. This reinforces our choice of reanalysis data.

The work inspires us to face the ability to produce a combined forecast that features additional characteristics of the TC system, like intensity or size, which might be necessary to understand how large the affected territories and the danger are, thus being essential for a real-life forecast. The HURDAT2 dataset possesses information on the size of TCs since 2014, provided by variables: wind radii maximum extent and radius of maximum wind. Intensity is provided by the variables: maximum sustained wind (in knots), and minimum pressure (in millibars). We are interested in predicting a subset of those values to approximate TC severity.

2.1.5 Olander et al. [13]

The authors are using Advanced Dvorak Technique (ADT) combined with DL methods to predict TC intensity by providing maximum sustained surface wind speed. They use ADT history imagery and satellite imagery and estimate the performance of a classification NN model. Through the work, a comparison to the conventional ADT algorithm and its error estimation is performed.

This work utilizes the Dvorak technique, a proven field knowledge technique based on a classification decision tree. ADT brings a pretty good estimate of TC intensity and is still used, for example, by NOAA[9], which utilizes satellite visible and infrared imagery. The technique was first introduced in 1969, performed by human analysts, and applied only in the northwest Pacific Ocean. Later, in 1998, an objective method run by computers was developed.

The work trains the NN on five TC basins, which is essential because the ADT classification is different between basins, and the method should have been reinitialized in the past to be used in new regions. Several architectures are compared: RNN with a single neuron output layer, multi-classification NN with multiple neuron output layer containing a probability distribution. Both variants provided similar accuracy characteristics.

The NNs in the work are relatively simple as to the current day and possess up to 128 neurons. Nevertheless, it still provides a significant improvement in success rates in comparison to the conventional ADT method.

2.1.6 Comparison of the reviewed articles, by DL Task.

This section summarises the literature overview and outlines its exploratory character. Table 2.1 summarizes the basic information about it.

The first article [2] introduces a well-structured approach to data preparation and an introduction to the task of identification of the TC system using a binary classifier. The classification task in the field has appeared as already well explored using DL methods, so we focused on the more complicated task of track prediction discussed in the following three articles [16] [25] [17], from which we learn about the high impact of a combination of feature extraction and memory-based NNs. In the last two articles [17] [13], we are presented with

[9]Hurricane FAQ. (2022, August 17). NOAA's Atlantic Oceanographic and Meteorological Laboratory. https://www.aoml.noaa.gov/hrd-faq/

Article	**DL Task**	**Datasets**	**Architecture**
Gardoll and Boucher [2]	Compare performance, sensitivity of a CNN to learning dataset	HURDAT2, MERRA-2, ERA5	CNN binary classifier
Ren et al. [16]	Typhoon track prediction	IBTrACS	C-LSTM
Wang et al. [25]	TC track prediction	IBTrACS, ERA5	RNN, LSTM, GRU, GRU_CNN
Rüttgers et al. [17]	Typhoon track and cloud structures prediction	Satellite images (VISSR)	GAN
Olander et al. [13]	TC maximum sustained surface wind speed prediction	Satellite images (VISSR)	RNN

Table 2.1
DL task and architecture across the literature.

the combined task of track and intensity prediction, which seems to be the most advanced task today.

2.2 NN architectures.

2.2.1 Hochreiter and Schmidhuber [6]

The authors of [6] introduced the LSTM as a new type of RNN network in 1997. It serves the target of remembering recent input events using feedback connections, passing the information to the next execution step of the layer. They define this process as Short-Term Memory, as opposed to the learning process of NN in which neuron weights are gradually changing, which they refer to as Long-Term Memory. LSTM has solved the problem of vanished or exploding gradient, which existed at other implementations of the Short-Term memory method using RNNs. They have achieved the advantage due to advanced architecture that enforces constant error in the Hidden States by Constant Error Carousel (CEC).

LSTM is a type of RNN network and features a bi-directional learning flow. That means that output from some nodes influences subsequent input to the same nodes. The memory cell of LSTM is a more complex network than RNN and features the Input, Forget and Output Gates, the Hidden State, Cell State and the output node. Hidden State keeps the Short-Term Memory information; it is updated at every step and propagates its information to the next step. The Cell State interacts with the gates, regulates their influence on the information that flows through the memory cell and retains long-term dependencies in the sequence data. The input gate uses input data and the Hidden State and applies an activation function in order to decide how much information from the previous Hidden State will be remembered in the memory cell. Forget Gate controls how much of previous memory should be forgotten and is the modern implementation of the CEC. The Output

Gate controls how much the Hidden State should influence the next Hidden State and how much it should influence the current time step's output, which is the prediction of the cell.

The LSTM network was shown in the [6] to be able to retain the Short-Term Memory effect for over 1000 steps; lThere seems to be a common believe that later later applications show that LSTMS can solve tasks that require Short-Term Memory even for more than millions of execution steps. The mean number of TC observations in the HURDAT2 dataset in 1970-2021 is 27, and the maximum is 118. So, we expect that LSTM will be able to retain Short-Term Memory easily during the training or prediction process. The use of LSTM can help significantly in achieving our goal.

2.2.2 Shi et al. [18]

An alternative to common LSTM, which utilizes 3-dimensional data of time, rows, and columns, was introduced by [18]. The paper deals with the problem of precipitation nowcasting in a local region. Nowcasting is a type of weather forecasting that operates in a very short time range from 0 to 6 hours. This task is a novel approach because conventional Numerical Weather Prediction (NWP) methods produce longer-term forecasts. Authors compare the results of their method with the current state-of-the-art NWP nowcasting method ROVER[26].

The paper formulated the problem as a spatiotemporal sequence to sequence prediction and solved it. The proposed version of the LSTM memory cell features 3D tensors as the inputs, the outputs, the hidden states, and all the memory cell gates. It uses the Hadamard product[10] on Cell Outputs and Convolutional Operator on the input-to-state and state-to-state transitions instead of matrix addition and multiplication, compared to regular LSTM. By doing so, ConvLSTM determines the next state of the Hidden State not only by the inputs and the sequence of previous Hidden States but also by local neighbours of the Hidden States. The authors use zero padding in convolutional operations to keep the initial dimension of the data and avoid making assumptions about out-of-the-scope space. The trained NN architecture is based on temporally connected LSTM encoder and decoder layers, which was introduced for sequence-to-sequence prediction by [23]. Several LSTM layers are stacked together, following [21], and the Hidden States and outputs of the decoder network are initialized with the values of the Hidden States and outputs from the encoder network. The final forecast is given by a 1×1 convolutional layer receiving all the states of the decoder network. In this architecture, the encoder network does not pass a sequence to the decoder network, and the decoder network does not produce the output sequence by itself.

The authors evaluate their NN on a synthetic Moving-MNIST dataset [21] against Fully Connected LSTM (FC-LSTM)[3] and outperform it by an improvement of 24% of cross-entropy metric. The authors also evaluate their model on real-life radar echo dataset and outperform the state-of-the-art operational ROVER algorithm by an improvement of 17% and outperform FC-LSTM by 24% of Rainfall Mean Squared Error (Rainfall-MSE).

[10]Wikipedia: Hadamard product is a binary operation that takes in two matrices of the same dimensions and returns a matrix of the multiplied corresponding elements.

The use case described in the paper, as well as the type of data, is very similar to our DL task. Therefore, we would like to utilize the proposed NN layer and investigate the performance of different model architectures.

Chapter 3

Research Questions

In this study, we construct a pipeline that receives time and location series data from meteorological reanalysis to produce a forecast for TC system development, including the trajectory and intensity changes over time. The goal is to build and output a visualization of the future evolution of the TC system and predict a point for a landfall location.

While track prediction has already been proven to work well with DL approaches, the prediction of intensity is still rarely implemented using DL methods. Conventional intensity prediction is considered a hard task [24] in the field of TC forecasting. It is so because of the natural complexity of the meteorological conditions as well as an increase in the proportion of intense TCs and RIs, as described earlier. The occurrence of RI events is by itself a reason for error in forecasts; according to [24], rapid changes in intensity in both directions, explain about 20% of the variance of forecast errors in the Atlantic and 30% in the east Pacific.

The combination of meteorological 2D data for trajectory forecast of TCs is a relatively novel approach with only a few occurrences in the literature and from only a recent period. For example, [25] have proved the positive effect of using Sea Surface Temperature and Geopotential Height meteorological variables to gradually improve the TC trajectory prediction in mid-term forecasts of 24h-72h.

We test the ability to predict compound sets of variables of different dimensions. We use scalar data from HURDAT2 and 2-dimensional meteorological data from ERA5 datasets and inspect the possible improvement compared to partial sets of those variables. For improvement of the forecast, we bring to a test four different NN architectures based on ConvLSTM2D Tensorflow (TF) layers.

A further research question that this work addresses is to detect the influence of global warming and changes in the behavior of TCs on the training process and the forecasting abilities of our model. We inspect the changes in forecasting error of location and intensity, depending on the choice of train and test sets. We expect that if we take more recent years as test data and omit the 'year' variable, we will face an increase in error in the prediction of intensity and an increase in error in the prediction of latitude towards northern observations.

Summarizing the above mentioned topics, the research questions for the study are the following.

1. Construction of a pipeline for simultaneous sequence-to-sequence forecasting of the track and the intensity of TCs.
    (a) Use field knowledge for improved data preparation.
    (b) Bring larger set of variables to test, in order to fit the compound character of the task.
2. Detect influence of the climate change on the training process of the model.

Chapter 4

Methodology and Research Design

4.1 Data sets and variables

A TC system development is a complex physical system dependent on many environmental features. Though the application of DL methods has shown to be effective in research, mid (24h - 72h) and long-term (>72h) predictions are considered hard and regularly have a large error. The forecast of intensity development features non-linear behavior [12], and its prediction is still perceived as a hard DL task. Common forecasting methods involve information from meteorological models and climate simulations and take into account various variables in order to produce a good quality forecast. Hence, there is a need for DL methods to involve a variety of meteorological variables that are considered to be relevant as well.

A meteorological reanalysis is a combination of observations with a computer model which reconstructs past weather conditions using previous short-range weather forecasts in order to reconstruct weather conditions accurately in the past. The result of the rerun of the model fills the gaps in observations in order to produce evenly distributed maps of values. This data reconstructs past weather and climate; it is used to initialize numerical weather prediction methods and builds models needed to calibrate forecast products [1].

In this work, a set of meteorological variables from the ERA5 reanalysis data set (see Table 1) will be used to predict the development of a TC. This set was first suggested, based on previous research, and proved to be efficient by [11] in the application of a CNN network at a task of detection of extreme events, like TCs, Atmospheric Rivers and Weather Fronts. It also was proved to be effective by [2], but these works did not include feature importance tests, so exclusion tests may be relevant. Those sets of values are considered images in the work in the sense that they represent a continuous map of values distributed over a spatial map. Those values do not represent colours but can be represented as visual images using suitable software. We will acquire TC tracks data from the HURDAT2 dataset, which holds the location and intensity information in 6-hour steps.Our training and prediction processes encompass data from the years 1974 to 2021. To ensure consistency in coordinate handling across hemispheres, we negate the latitude by multiplying it by -1 for the south-

[1]ECMWF. (2019, October). Use of ERA5 reanalysis to initialise re-forecasts proves beneficial. https://www.ecmwf.int/en/newsletter/161/meteorology/use-era5-reanalysis-initialise-re-forecasts-proves-beneficial

ern hemisphere and similarly negate the longitude by multiplying it by -1 for the western hemisphere. More detailed information on the data sets is available in the appendix.

ERA5 Dataset	**HURDAT2 Datset**
Mean sea level pressure (Pa)	Latitude (WGS84)
Total column water vapour (kg / m^2)	Longtitude (WGS84)
Northward wind at 10 meters (m / s)	Maximum sustained wind (in knots)
Eastward wind at 10 meters (m / s)	Year
Eastward wind at 850 hPa (m / s)	Month
Northward wind at 850 hPa (m / s)	
Temperature at 500 hPa (K)	
Temperature at 200 hPa (K)	

Table 4.1
List of variables used in the forecasting process.

4.2 Forecasting method

The training data is a sequence of multi-layer images of meteorological variables centred on the location of a TC, accompanied by the location coordinates, year, month, and wind intensity data. The size of the images is 64x64 data points, which corresponds to 16x16 degrees of longitude and latitude. The data is organized in sequences of past and future observations in time steps of 6 hours for a supervised learning process, where future observations serve as a label for appropriate past observations. Thus, the forecast is a sequence of the same set of variables. We will use four past observations, which correspond to an observation of 24h and predict 4, 8, or 12 observations, which correspond to forecasts of up to 24h, 48h, or 72h. Training data samples are cut from the original sequences of TC observations; each subsequent sample shifted in advance of one 6-hour step from the previous one until there were not enough observations left to fill the required number of future observations. We use all of the TCs that fit the minimum requirement of minimal length. When we try to predict a longer term, we lose all the TCs that are shorter than the minimal length, and they are not included in the training. This drawback is natural because if we concentrate on a certain time period, shorter TCs will not reach it either way. However, at the same time, the same explanation holds for making it meaningful to perform separate training for different time ranges.

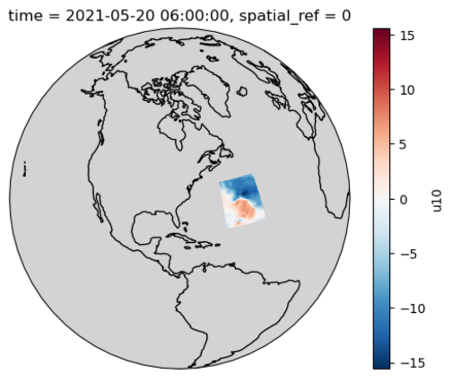


Figure 4.1. Eastward wind on 10 meter. Visualisation of chosen size window for ERA5 data on the globe, centered on a TC.

4.3 Meteorological data dimensionality reduction

The meteorological data is considered in a 64x64 dimension 2D format, which may result in heavy memory loads and long training because of a large number of neurons. For example E1D1 model would have 79,328,737 neurons, with input dimension suitable for the original data and filters multiplied by 8 in order to keep the same ratio of the filters. So, a dimensionality reduction is relevant in order to reduce computational complexity. We will also check which method of dimensionality reduction gives us the most useful result in terms of the training of our forecasting model.

Since we deal with image-like data, using CNNs is possible, which is commonly used for vision-related tasks. Common CNN kernel filters, max pooling, average pooling, convolutional edges and others diminish the image and assist in understanding the data. Afterwards, dense NN layers are usually added and are trained for specific tasks on the modified data. The efficiency of Dense layers depends on the choice of the first stack of modifying layers. The layers have a high number of neurons in the input level and include a high number of hidden neurons in order to detect correlations in the data. To reduce the size of the network, we can continue to shrink the input further with filters, but this may cause severe information loss. This technique was successfully implemented by [25].

Feature detection techniques also could be used to detect a common pattern of a formation of TC, like in [13], where the authors implemented Advanced Dvorak technique[2] to classify TCs using satellite photo images. This approach uses feature detection layers in order to improve the vision ability. It helps to divide the larger task into smaller, simpler ones, also reducing the dimension in a smart manner with a lower tendency to lose information. Such an approach would require the preparation of labelled data and additional supervised training processes and would serve the purpose of detecting or classifying a TC. Therefore, this method is out of the scope of this work.

Another option is to apply an autoencoder and use its Latent Space Representation (LSR). In the training process of autoencoders, the input data is diminished by encoding layers to LSR, and decoding layers try to reconstruct it back to the original dimensionality. It makes the LSR, by its construction, sufficient for the representation of the original data. The efficiency of this method might be measured by the reconstruction loss.

We test two different types of autoencoders to produce the LSR. Deep Autoencoder, which consists of Dense layers and Convolutional Autoencoder with Conv2D layers followed by MaxPooling layers in the encoder network and Conv2D layers followed by UpSampling2D layers in the decoder network.

We would like to test an assumption, whether the data encoded by the encoder will be easier to process for the subsequent forecasting models. The basis for this assumption is that by construction, the LSR contains the most relevant information for the representation of the original data. And the smaller number of data points will also make it easier for the network to learn.

Since the training process is unique for each type of data, we train a separate copy of each autoencoder model for each variable and then save the suitable LSR frame for each original frame of the variable.

[2]Wikipedia: The Dvorak technique (developed between 1969 and 1984 by Vernon Dvorak) is a widely used system to estimate tropical cyclone intensity based solely on visible and infrared satellite images.

We will compare the above methods with data resized by bicubic interpolation from OpenCv[3] given by 'resize' function, using interpolation=INTER_CUBIC. When resizing, this method looks at a 4x4 grid of neighbours of the pixel and uses a weighted average of the neighbours to interpolate the new pixel when the closer neighbours have a higher weight. This is a common resizing method in many image processing applications, which gives relatively clear picture quality[5]. Since the transformation keeps the overall image very close to the original, we will consider this type of compression as the ground-truth image and compare other compressions to it.

4.4 Development and choice of autoencoders

The development process of autoencoders includes many decisions and manual fixes, tests of different numbers of layers, types of activation functions and kernel sizes. The problem is that every NN model has specific hyperparameters for specific data. In the case of autoencoders, different sizes and numbers of layers can vary a lot. We want to bring our data through a bottleneck that concentrates the information flow at the LSR between the encoder and the decoder. However, the question of how much to cut at each layer could be answered only empirically after training the proposed network.

During the development process, we discovered that a deep autoencoder network grows very fast in the number of neurons and can easily lead to long training times. Moreover, the network gave worse results when we added more than 2-3 layers. We have concluded that the network is is too big for the amount and type of data and cannot train all the weights properly. During the training, when we used Rectified Linear Unit (ReLu) activation function we also encountered a problem: the network might tend to output very dull images with a very simplified picture, as in the 4.2.

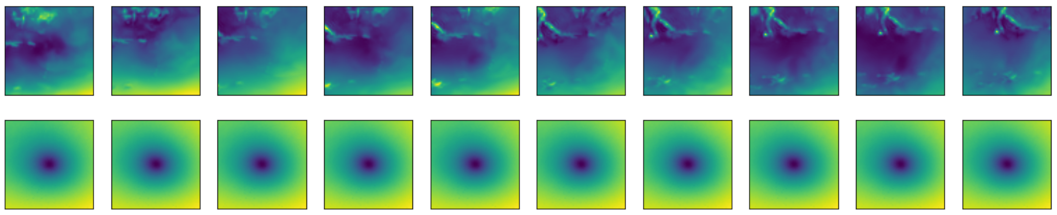


Figure 4.2. Dying ReLu example.
Original vs. restored images by Deep autoencoder of V wind component on 10 meters height.

This phenomenon is known as the 'dying ReLU' and occurs in standard ReLU activation functions. When a neuron receives a negative input, it outputs zero. This tendency can lead neurons to become 'stuck' during training, consistently outputting zeros, rendering them inactive and resulting in the loss of crucial information. Leaky ReLU addresses this by enabling a slight gradient or information flow for negative inputs. Unlike the standard ReLU, Leaky ReLU does not clamp negative inputs to zero; instead, it allows a small, non-zero output, usually controlled by a slope parameter (alpha). This parameter determines the extent of the leak for negative values, thereby preventing neurons from becoming entirely inactive and aiding in preserving information flow during training.

[3]OpenCv Python package: https://pypi.org/project/opencv-python/

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha \cdot x & \text{if } x \leq 0 \end{cases}$$

Figure 4.3. Leaky ReLu formula.

The selected architectures are as follows:

Deep Autoencoder with two dense layers: The initial dense layer accepts an input size of 4096 and outputs a dimensionality of 600, while the subsequent Dense layer reduces this to a dimensionality of 64. The entire model encompasses 4,997,360 trainable parameters. The visualization of the architecture schema of the Deep Autoencoder can be seen in Figure 4.4, the model summary is presented in the Table 4.2:

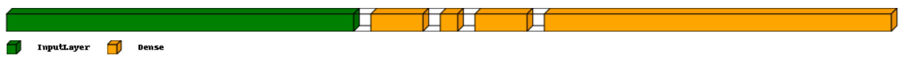


Figure 4.4. Architecture schema of Deep Autoencoder.

Layer (type)	**Output Shape**	**Param #**
input_15 (InputLayer)	(None, 4096)	0
dense_28 (Dense)	(None, 600)	2458200
dense_29 (Dense)	(None, 64)	38464
dense_30 (Dense)	(None, 600)	39000
dense_31 (Dense)	(None, 4096)	2461696

Table 4.2
Deep Autoencoder model Summary

A Convolutional Autoencoder with encoder network, which includes three Conv2D layers succeeded by MaxPooling layers employing a 2x2 pooling window. The decoder network has three Conv2D layers followed by UpSampling2D layers with a 2x2 size, culminating in one more Conv2D layer utilizing a sigmoid activation function. All Conv2D layers, excluding the final one, utilize the LeakyReLU activation function, set with alpha=0.15, and maintain a 3x3 filter window. This configuration results in a model comprising a total of 2,796 trainable parameters. The vizualization of the architecture schema of the Convolutional Autoencoder can be seen in Figure 4.5, the model summary is presented in the Table 4.3.

The stark contrast in network sizes is evident here, where a seemingly straightforward Dense network possesses 1787 times more neurons than a comparatively intricate Convolutional network.

The chosen Deep Autoencoder architecture produces blurry yet generally exact images which maintain the right centres of low or high values and correctly represent asymmetric images. We can see an example in Figure 4.4. In the training process, the loss function has reached the Mean Squared Error (MSE) value of 0.00057305; the validation loss function has reached the MSE value of 0.00061997. The best value for the validation loss was reached at the 51st epoch, reaching the early stopping with no improvement of the validation loss function for 20 epochs.

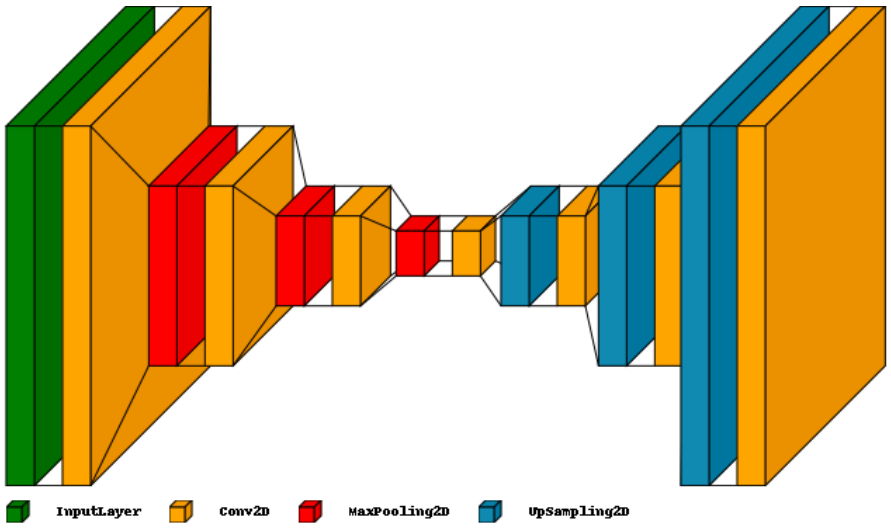


Figure 4.5. Architecture schema of Convolutional Autoencoder.

The chosen Convolutional Autoencoder architecture produces sharper images, maintains the right centres of low or high values, and correctly represents asymmetric images. The main advantage of the Convolutional Autoencoder is that it can produce sharper edges on the images in comparison to our Deep Autoencoder architecture. We can see an example in Figure 4.5. In the training process, the loss function has reached the MSE value of 0.00019757; the validation loss function has reached the MSE value of 0.0002031. The best value for the validation loss was reached at the 200th epoch, and no early stopping happened.

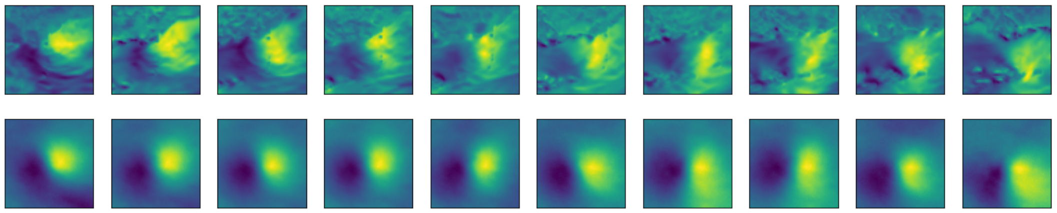


Figure 4.6. Original vs. decoded images by Deep autoencoder, V wind component on 10 meters height.

We visualized the LSR for educational purposes to see how far autoencoders take the representation of the image away from the original image while compressing it. We can see that the visualizations of the Deep Autoencoder LSR have no meaningful structure for the human eye; see Figure 4.8. While the LSR of the Convolutional Autoencoder does remind us of the original image, the directions on the image do not seem to correspond to the final reconstructed image4.9.

Finally, during the development process the architecture with convolutional layers has shown more stable results than the architecture with dense layers. The training time

Layer (type)	**Output Shape**	**Param #**
input_1 (InputLayer)	[(None, 64, 64, 1)]	0
conv2d (Conv2D)	(None, 64, 64, 16)	160
max_pooling2d (MaxPooling2D)	(None, 32, 32, 16)	0
conv2d_1 (Conv2D)	(None, 32, 32, 8)	1160
max_pooling2d_1 (MaxPooling2D)	(None, 16, 16, 8)	0
conv2d_2 (Conv2D)	(None, 16, 16, 1)	73
max_pooling2d_2 (MaxPooling2D)	(None, 8, 8, 1)	0
conv2d_3 (Conv2D)	(None, 8, 8, 1)	10
up_sampling2d (UpSampling2D)	(None, 16, 16, 1)	0
conv2d_4 (Conv2D)	(None, 16, 16, 8)	80
up_sampling2d_1 (UpSampling2D)	(None, 32, 32, 8)	0
conv2d_5 (Conv2D)	(None, 32, 32, 16)	1168
up_sampling2d_2 (UpSampling2D)	(None, 64, 64, 16)	0
conv2d_6 (Conv2D)	(None, 64, 64, 1)	145

Table 4.3
Convolutional Autoencoder model Summary.

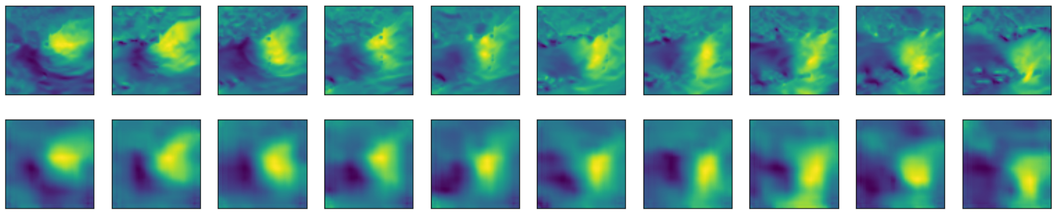


Figure 4.7. Original vs. decoded images by Convolutional autoencoder of V wind component on 10 meters height.

of deeper architectures with dense layers were time consuming. Moreover, the resulting reconstructions of the convolutional architectures look more informative.

4.5 Combining data of different dimensions

In the literature it is seen as a standard practice when implementing architectures involving CNNs to incorporate a flatten layer after CNN filters and then put the 1-dimensional data into dense layers, like in the work of [25]. The flattened 1-dimensional data can also be directly fed into the next sequence to sequence NN architecture using conventional LSTM, or GRUs. However, the flattening approach has some basic disadvantages. First of all, in the case of dense layers, they have many more neurons than a convolutional structure and do not suit modern sequence-to-sequence prediction. In the case of direct feed to LSTM or GRU, we would also have a very heavy network because all of the grid data points are fed into the network as separate variables.

An alternative to common LSTM, in this case, would be a 2-dimensional LSTM. We are using ConvLSTM2D implementation by Keras, which was developed based on the research of [18]. This implementation features convolutional input and recurrent transfor-

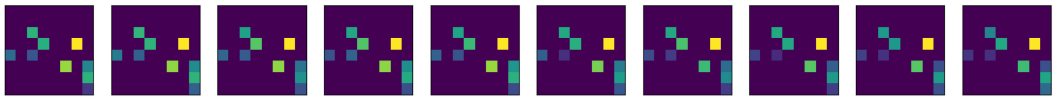


Figure 4.8. LSR of Deep autoencoder of V wind component on 10 meters height.

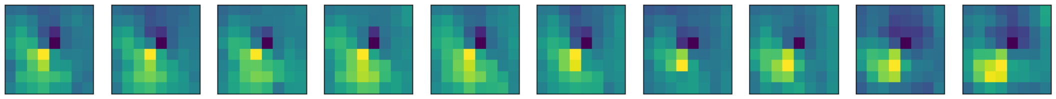


Figure 4.9. LSR of Convolutional autoencoder of V wind component on 10 meters height.

mations instead of matrix addition and multiplication compared to regular LSTM. The possible input of ConvLSTM2D is a 5D tensor of data in the shape of (samples, time, rows, cols, channels), and the possible output is a 5D tensor with shape: (samples, timesteps, new_rows, new_cols, filters). Each variable is a separate 2D channel, so we expand our scalar variables from HURDAT2 to be the same dimensionality as ERA5 variables by replicating the same value in all rows and columns.

4.6 Architecture of the model

In our architecture, we choose to handle all the variables in a single information stream because we believe that all the variables are naturally connected since they represent the same meteorological system. We want to check whether meteorological variables assist in predicting the scalar variables, which is the fundamental question of our study, and whether the scalar variables assist in training on the meteorological variables.

In order to produce sequence-to-sequence forecasting, we will use encoding and decoding LSTM layers according to the architecture proposed by [23]. While the encoding layer is responsible for learning the input data of past observations, the decoding layer generates the sequence of future observations using the predicted representation from the encoding layer.

Here, we also test the option of stacking two encoding and two decoding LSTM layers. The purpose of this technique is to receive an additional level of abstraction of input observations over time. The second LSTM layer should improve long-term predictions. This technique was introduced and shown to be effective in speech recognition tasks by [4] and is now a common technique in sequence prediction tasks. The authors of [4] showed that stacking several LSTMs is more efficient than increasing the number of memory cells in a layer.

For our first type of architecture, we are using three stacked ConvLSTM2D layers with batch normalization layers between them as the encoder network. This network will utilize only the option of sequence output of the ConvLSTM2D layers. Each layer passes to the next one in the sequence of hidden states for each temporal time step. The decoder network will be a single Conv3D layer that receives the last sequence from the last ConvLSTM2D layer in the encoding network. We carry out this implementation only in order to highlight the advantage of the encoding-decoding architecture. We will name the model 'Stacked' across the document; the model has 3,876,685 trainable parameters.

For our central architecture, we follow [18] [21] for the general architecture design. Stacked ConvLSTM2D layers inside the encoder network and stacked ConvLSTM2D layers inside the decoder network pass the sequence of hidden states to each other. The connection between the encoder and decoder networks is only by passing the last states and not the sequence. The input of the decoder network is the last hidden state, repeated as the desired number of future predictions. We initialize the initial states of the decoder network with the last sidden state and last cell state. So, the i-th encoding ConvLSTM2D layer initializes the i-th ConvLSTM2D decoding layer. For the final output, we apply a dense layer with a number of units equal to the number of channels using the TimeDistributed layer to every temporal slice of the output. We apply a batch normalization layer between each pair of layers that passes sequences between them to accelerate the training and help preserve the gradient from vanishing or exploding, as suggested by [8]. The decoder part ends with two time distributed layers. The time distributed Layers apply a dense layer to every temporal slice of an input tensor independently. It consistently applies the same layer to every temporal slice, maintaining the same weights and biases for each time step. This process facilitates learning patterns or features across the entire sequence. During the development process, we discovered that an additional second dense layer applied by time distributed layer accelerates the training process and preserves the same quality of predictions.

We will test 3 versions of this architecture: with one, two or three stacked ConvLSTM2D layers in the encoder part and correspondingly in the decoder part; we name the models 'E1D1', 'E2D2', 'E3D3' correspondingly. E1D1 architecture has 1,314,733 trainable parameters, E2D2 has 6,723,757, and E3D3 has 8,362,669 . We can see the visualization of the E2D2 model at Figure 4.10

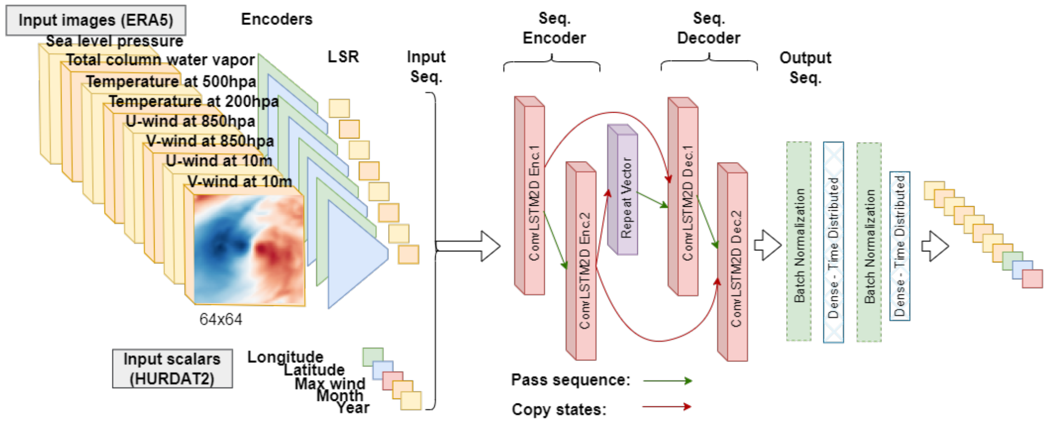


Figure 4.10. Architecture of the E2D2 model.

4.7 Extracting the result

Since we have decided to artificially create a two-dimensional channel for scalar variables from the HURDAT2 dataset by multiplying the appearances of the same value in all rows and columns of a frame in the channel, and our prediction sequence corresponds to the same shape and dimensions of the input sequence, we have to perform extraction actions in order to receive a final value of the desired variable. Our model architecture has a single data stream for all the variables, and all channels undergo the same manipulations together, though treated internally as different channels. It means that every channel in our final result is affected by all other channels; the scalar variables receive a projection of meteorological variables on them. We can visualize the predicted array can be visualized as an image; see Figure 4.11 where we observe some patterns similar to meteorological data.

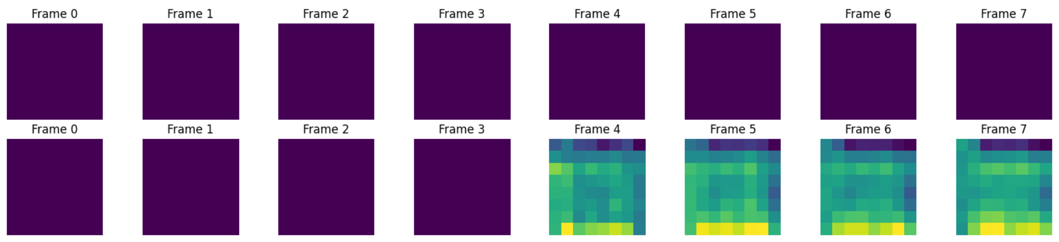


Figure 4.11. Visualization of predicted Longitude channel.
First line - original values. Second line - 4 original seed values followed by the predicted values.

Because we have chosen MSE as our loss function for the training, our model predicts the values with weights optimized to minimize the overall MSE of all single image values together. For the extraction method of the scalar variables, it would be mostly natural to take the mean value between all values inside one frame and consider it the predicted value.

4.8 Use of batch normalization layer

We have investigated the significance of the batch normalization layer by performing separate training of the E1D1 model. The batch normalization layer was incorporated atop the sequence derived from the decoding ConvLSTM2D layer. With batch normalization implemented, the E1D1 model required 645 seconds for training, reaching early stopping at 30 epochs. Notably, by the 20th epoch, it achieved a validation loss of 0.00238. In contrast, the same model without Batch Normalization consumed 1510 seconds, reaching early stopping at 68 epochs and attaining a slightly higher validation loss of 0.00268 on the 58 epoch. So even in our most basic and lightweight architecture, the application of batch normalization saves us 63% of training time.

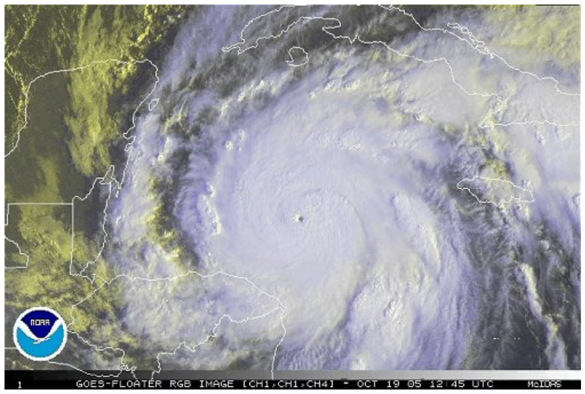
However, when we implemented batch normalization in the E2D2 network across all sequence passes between the layers (See Figure 4.10), it significantly reduced the performance. Our analysis revealed that the mixed usage of ConvLSTM2D and batch normalization layers carries drawbacks. Batch normalization's normalization across the batch dimension disrupted temporal dependencies, impacting the network's performance.

Given that the E1D1 network passes a batch normalized sequence only once—from the decoding layer to the time distributed layer employing dense layers—the temporal dependencies remained unaffected. Consequently, we opted to utilize batch normalization solely between LSTMs and dense layers to preserve temporal dependencies.

4.9 Information on the chosen TC for visualization

We will bring in our study as an example for visualization of TC Wilma 2005, which was classified as category five on the Saffir-Simpson Hurricane Wind Scale. TC Wilma was the most powerful TC ever recorded in the Atlantic basin in terms of minimum central pressure of 882 hPa. The central pressure of a storm serves as a critical indicator of its intensity. The pressure gradient drives stronger winds, drawing air towards the low-pressure center at higher velocities. Additionally, as the pressure decreases, the storm's core becomes more compact and concentrated, consolidating, and organising its energy.

In terms of maximal sustained wind, Wilma is among other TCs who reached the maximal recorded speed of 95 knots. Its rapid intensification phase occurred between 18:00 UTC on October 18 and 06:00 UTC on October 19, 2005, when Wilma traversed the area of high ocean heat content. During its RI event, the TC system had a 29-hPa drop in central pressure in the first 6 hours and a 54-hPa drop in the second 6 hours and reached the maximum speed of 295 kilometres per hour.


1
GOES-FLOATER RGB IMAGE [CH1,CH1,CH4] - OCT 19 05 12:45 UTC McIDAS
Figure 4.12. Hurricane Wilma after RI, 12:45 UTC, October 19, 2005 – About 340 miles southeast of Cozumel, Mexico.

We have included the TC Wilma in our test dataset, and we will use it to present the performance and the generalization capabilities of our models with the help of a meaningful example.

Chapter 5

Experiments

5.1 Training

Training of the models was performed on on Python 3 Google Compute Engine backend (GPU) with Tesla V100-SXM2-16GB System RAM 12.7 GB and 15.0 GB of GPU RAM. In order to overcome the overfitting effect we have used Early Stopping TF callback, conditioned on improvement of the validation loss function and Model Checkpoint TF callback. After the training finishes we reload the weights of the model from the last best Checkpoint. Deep Autoencoder was trained and performed early stopping on 196 Epoch and it took 304 seconds. Convolutional Autoencoder trained for 200 epochs and it took 264 seconds.

Training sequence-to-sequence models on bicubic compressed data for short-term forecasts resulted in varied training times. The Stacked model completed training in 851 seconds, reaching early stopping by the 16th epoch; E1D1 took 563 seconds, reaching early stopping by the 17th epoch; E2D2 required 740 seconds, performing early stopping at the 34th epoch; E3D3 took 1563 seconds to reach early stopping by the 16th epoch. The patience parameter was set to 10 epochs.

5.2 Architecture selection

First, we would like to choose the better configuration for the most efficient encoding of the meteorological data and get an insight into the efficiency of the different model architectures. We run training with all variables on the short-term forecast of 24 hours. Short-term forecast training has two main advantages. More data samples are available for the training; since RI events generally develop during 24h, running on a set of samples with 24h of past observations and 24h of prediction will, on average, give insight into more data samples on the upcoming RI event.

To precisely understand the training dynamics, we compare the result of the validation MSE of each variable separately. The values were generated automatically during the training using custom metrics. We chose MSE as the loss function for the training method. It means that obtaining the separate MSE values gives us an impression of how much each variable contributed to the overall MSE. It allows us to understand which variables train better. The values are presented in the Table 5.1.

We generate the values using models trained on the training dataset, using the TF predict function applied to all data samples of the predicted track of a TC from the validation dataset. We predict the track in steps of 4 observations. We predict the observations of the next 24 hours based on original observations of the last 24 hours each time; we do not include duplicate observation points. The values in Table 5.2 represent the mean absolute difference between all observations in all samples and their corresponding predictions. Distance in kilometres was first measured separately for each location based on longitude and latitude before the calculation of the MAE. For example, for short-term 24-hour prediction, we will have in the resulting track prediction each time four predicted observations of +6,+12,+18, and +24 hours each, followed by the subsequent four predicted observations of +6, +12, +18, +24 until the end of the track. We will compare each of those predictions with the ground truth observation. This procedure will remain the same throughout the chapter.

5.3 Result of Architecture selection

	Loss	**Lat**	**Lon**	**Wind**	**msl**	**u10**	**v10**	**tcwv**	**v850**	**u850**	**t200**	**t500**
Stacked+B	0.00155	0.00137	0.00108	0.00303	0.00131	0.00220	0.00234	0.00591	0.00174	0.00157	0.00289	0.00160
E1D1+B	0.00031	0.00043	0.00040	0.00156	0.00063	0.00138	0.00155	0.00410	0.00101	0.00124	0.00146	0.00074
E2D2+B	0.00042	0.00035	0.00037	0.00150	0.00052	0.00148	0.00152	0.00360	0.00097	0.00097	0.00135	0.00072
E3D3+B	0.00049	0.00091	0.00035	0.00116	0.00096	0.00183	0.00167	0.00402	0.00094	0.00098	0.00140	0.00088
Stacked+C	0.00132	0.00121	0.00088	0.00293	0.01083	0.00103	0.05416	0.00714	0.04012	0.00149	0.00179	0.00491
E1D1+C	0.00046	0.00044	0.00055	0.00141	0.00507	0.00058	0.00960	0.00455	0.00070	0.00093	0.00068	0.00294
E2D2+C	0.00042	0.00062	0.00052	0.00125	0.00481	0.00052	0.00956	0.00454	0.00069	0.00083	0.00069	0.00306
E3D3+C	0.00044	0.00052	0.00049	0.00185	0.00488	0.00051	0.00968	0.00460	0.00080	0.00093	0.00071	0.00292
Stacked+D	0.00119	0.00114	0.00131	0.00277	1.33531	4.95765	2.29369	2.34665	3.06112	1.22288	0.00081	1.67651
E1D1+D	0.00266	0.00096	0.00085	0.00233	0.00932	0.02256	0.02922	0.05668	0.01899	0.01526	0.00109	0.01680
E2D2+D	0.00120	0.00070	0.00055	0.00282	0.01003	0.02098	0.02885	0.05520	0.01670	0.01462	0.00096	0.01516
E3D3+D	0.00218	0.00124	0.00093	0.00175	0.00991	0.02194	0.02881	0.05828	0.02002	0.01579	0.00100	0.01948

Table 5.1
Comparison of partial MSE loss functions on the validation dataset.
B. - Bicubic Interpolation, C. - Convolutional Autoencoder, D. - Deep Autoencoder
Cyan color highlights the lowest value across the given compression type, and red color is across all compression types.

Based on the results from Table 5.1, we can see that the E2D2 model had the best validation loss function values for meteorological variables from ERA5 across all compression methods. E1D1 and E3D3 have performed very closely to E2D2. Generally, we can conclude that the Encoder-Decoder architecture successfully predicts the ERA5 variables. Notably, the use of convolutional compression has led to the lowest values of partial MSE validation loss function for the variables u10, u850, v850, and t200, with significant differences compared to other compression methods. Bicubic compression has led to the lowest values for the meteorological variables msl, v10, two, and t500, with significant differences compared to other compression methods. We can see an example in Figure 5.1. The figure represents a visualization of original values and predictions of the V-component of the wind at a height of 10 meters, for the full track of Wilma 2005 TC. The prediction was produced by the model E3D3 on convolutional compression and have reached value of validation MSE of 0.00051. The odd lines display the original value from 0 to 44. The even lines display four original seed values, numbered 0 to 3. In the prediction process, we predict each time frame of four observations of the next 24 hours based on original observations of the last 24 hours. The resulting predictions are concatenated and are displayed as frames numbered 4-44 in the second and fourth lines.

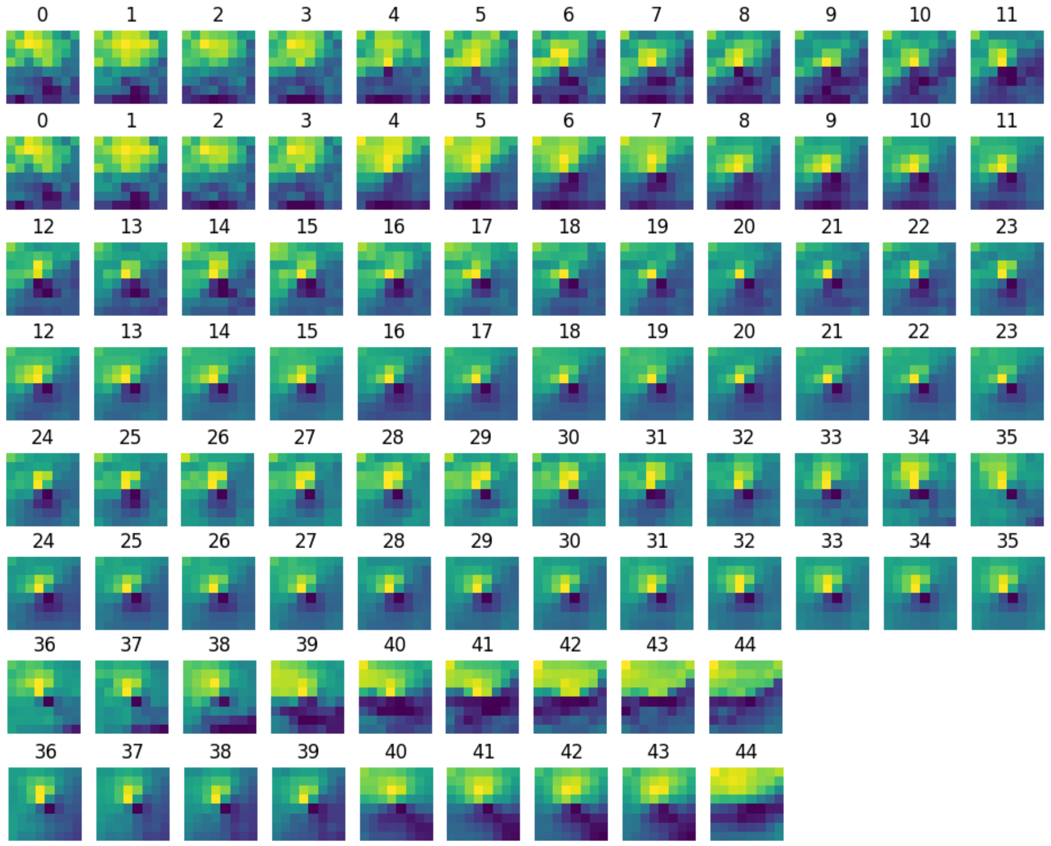


Figure 5.1. V10 variable, Wilma 2005. Model E3D3, convolutional compression, short-term forecast.
Odd lines - original values. Even lines - 4 original seed values (frames 0-3) followed by the predicted values (frames 4-44).

We can observe the MAE of the models on the test dataset in Table 5.2. The minimal values of MAE are shown bold across each compression method; red highlights the best value across all compression methods.

	Lat (deg.)	**Lon (deg.)**	**Wind (knots)**	**Distance (km)**
Stacked + B.	1.579	2.532	10.869	329.662
E1D1 + B	**0.689**	1.646	6.809	186.559
E2D2 + B	0.808	**1.188**	6.895	**156.763**
E3D3 + B.	0.765	1.710	**6.653**	201.698
Stacked + C.	1.510	2.378	9.886	312.839
E1D1 + C.	**0.714**	2.518	7.342	260.770
E2D2 + C.	0.885	1.949	**6.707**	228.117
E3D3 + C	0.938	**1.530**	7.107	**196.077**
Stacked + D.	3.752	5.814	12.579	716.729
E1D1 + D.	1.051	2.156	8.826	256.734
E2D2 + D.	**1.023**	**1.632**	10.631	**209.497**
E3D3 + D.	1.469	2.729	**7.062**	329.555

Table 5.2
Comparison of MAE, on test dataset.
B. - Bicubic Interpolation, C. - Convolutional Autoencoder, D. - Deep Autoencoder

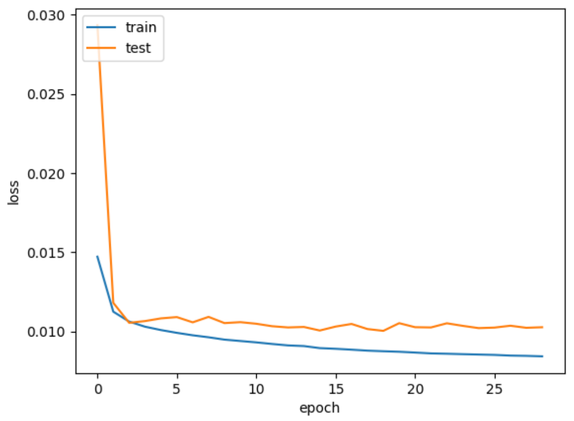


Figure 5.2. Overfitting effect of Stacked model during training.

Even though Stacked had minimal values of partial MSE test-loss function, it has shown the weakest performance on the validation and test datasets. It points to an overfitting effect. Figure 5.2 shows that the Loss and Validation Loss functions diverge from each other. The model has learned the training data so well that it loses the generalization effect and starts to perform poorly on new unseen data on the test.

In our case, the model encounters difficulties in learning long-term correlations in the data; it achieves low loss function values because it succeeds in fitting its weights very well for the specific training data. The reason for it is incorporated in the architecture of the model. The different layers do not share hidden and cell states between them, so no internal information specific to the LSTM architecture is passed between the different ConvLSTM2D layers of the model. It is also the same case regarding the decoding part of the network. It was replaced for simplicity by the Conv3D layer, which also does not receive any internal memory from the LSTM layers. Because of this, we decided not to investigate this architecture further in the following steps.

Based on the results of MAE, presented in Table 5.2, we conclude that the compression with the deep autoencoder was not helpful enough and lost too much information that was sufficient for the models to train effectively. According to the values of the partial validation loss function presented in Table 5.1, we conclude that, indeed, all of the models had significantly lower values in the measurement of the errors of the meteorological variables compared to other types of compression. So, we will not investigate the LSR of the deep autoencoder further.

In conclusion of this experiment, we can obtain from Table 5.2 across all compression methods that the E2D2 model has the best performance in terms of MAE of the total distance in kilometres, reaching a value of 156. E3D3 has the best MAE of the Maximum Sustained Wind Speed, reaching the value of 6.653 in knots. We can also observe some informative correlations. Across convolutional compression (lines 5-9), there is a trend of reducing MAE of Total Distance for more complex architecture. We conclude that the higher complexity of E3D3 architecture prevents the model from training more efficiently on the bicubic compression, which is close to the original data and possesses more rare information and finer lines or areas. Consequently, the relative simplicity of the data compressed by the convolutional autoencoder assists the more complex architecture type of E3D3 in more effective training. Nevertheless, the improvement is not sufficient to outperform the E1D1 and E2D2 models trained on the bicubic compressed data.

In figure 5.3, we visualize the result of the full track prediction by the models trained in the 5.2 experiment, and the method described in the subsection 5.2. The example of Wilma 2005 is a part of the test dataset and represents the most intense TC in the Atlantic basin in terms of the minimal center pressure. By obtaining this example, we can judge the generalization ability of the models. In the prediction process, we predict the values of four observations of the next 24 hours each time based on original observations of the last 24 hours. We concatenate and display the resulting predictions as the continuation of the plot. We can see the visualizations in the Figure 5.3. The red line and the four first observations on the green line represent the original values, and the green line after the fourth point represents the predicted values.

E1D1 Convolutional and E2D2 Bicubic show the most stable predicted tracks.

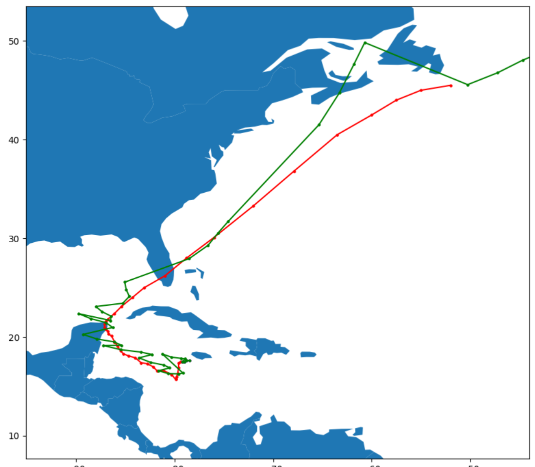
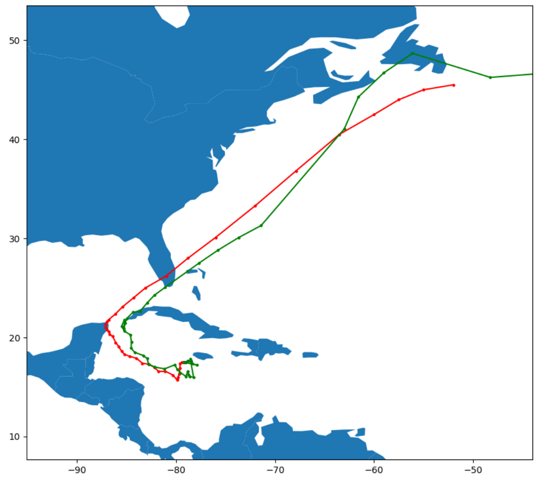
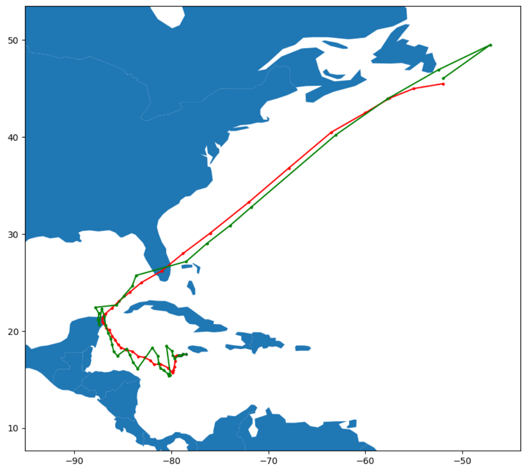
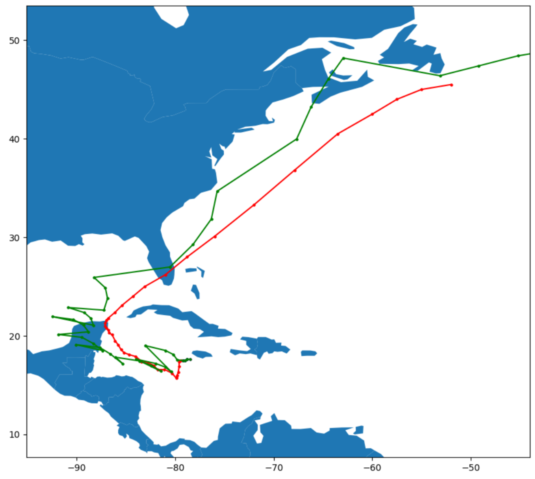
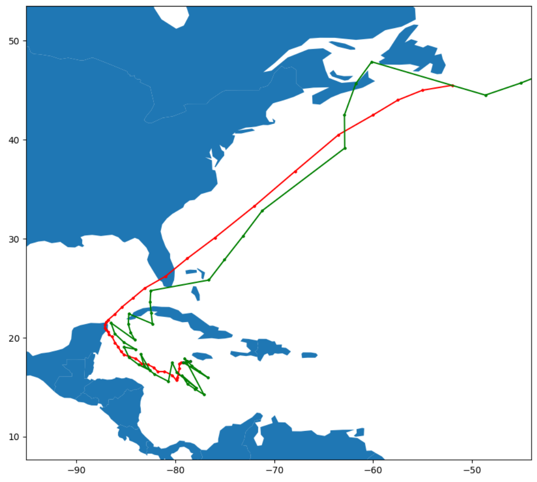
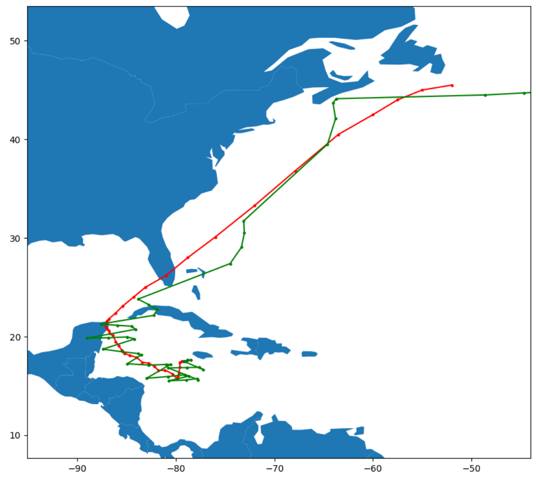
	Bicubic	Convolutional
E1D1		
E2D2		
E3D3		

Table 5.3
Visualisations of Short-Term predictions of Wilma 2005.

5.4 Exclusion test

In this experiment, we would like to analyze whether meteorological variables of ERA5 introduce an unneeded noise. The resulting values are presented in the table 5.6

	Lat (deg.)	**Lon (deg.)**	**Wind (knots)**	**Distance (km)**
Stacked	1.499	2.608	10.792	324.740
E1D1	0.996	**1.076**	7.394	**163.838**
E2D2	**0.854**	1.258	**7.138**	170.673
E3D3	1.445	1.294	7.362	223.451

Table 5.4
Comparison of MAE on the test dataset with limited training on HURDAT2 variables only.

We can obtain the MAE of the results in Table 5.6. In terms of the MAE of Distance, the Stacked and E3D3 models have performed slightly worse in comparison to their best runs on all variables. E1D1 have performed better without the meteorological variables reaching a value of MAE smaller in 12 km. However, there is a significant difference if we look at the results of E2D2. Without the ERA5 variables, it has an MAE value lower in 14 km. We conclude that the addition of Meteorological variables to the training set does bring an improvement, but only E2D2 succeeds in utilizing this advantage. It corresponds to the observation regarding the values of partial validation loss function MSE of the meteorological variables.

5.5 Test dependency on year

In this section, we test the dependency of the models on the variable Year. We train the models E1D1 and E2D2 with bicubic compression and E3D3 with convolutional compression as our best versions of the encoding-decoding architecture without the variable Year and compare the result to the previous runs. In all our experiments we divide the sample data across the years, leaving the most recent years in the validation and test datasets. By performing the consolidation, our models are not exposed to the most recent trends in movement and intensity of TCs, as expected from [22] [7] [1] . In this experiment we exclude the variable Year from the calculation in a custom loss function and perform the trainings for short-term forecast.

The resulting values are presented in the Table 5.5. The more complex models, E2D2 and E3D3, have succeeded in performing better when the Year variable is among the training set. Using the set of variables, a backward tendency appears; the more complex model has achieved poorer results. Notable the model E2D2 have reached an MAE value lower in 30 km better using the Year variable than the lowest MAE value in this experiment. We can also obtain that there is also an advantage in predicting the wind intensity.

	Lat (deg.)	**Lon (deg.)**	**Wind (knots)**	**Distance (km)**
E1D1 + B	1.088	**1.302**	6.982	**186.015**
E2D2 + B	**0.748**	2.082	7.128	230.930
E3D3 + C	1.286	2.128	**6.835**	268.783

Table 5.5
Comparison of MAE, on test dataset.
B. - Bicubic Interpolation, C. - Convolutional Autoencoder

5.6 Best run on 24h, 48h, 72h

In this section, we train the models E1D1 and E2D2 with bicubic compression and E3D3 with convolutional compression as our best versions of the encoding-decoding architecture. We extend our experiment to mid and long-term forecasts of 48 and 72 hours, correspondingly. The measurement method remains the same, as described in Section 5.2.

	Term	**Lat (deg.)**	**Lon (deg.)**	**Wind (knots)**	**Distance (km)**
E1D1 + B	24h	**0.689**	1.646	**6.809**	186.559
E2D2 + B	24h	0.808	**1.188**	6.895	**156.763**
E3D3 + C	24h	0.938	**1.530**	7.107	196.077
E1D1 + B	48h	2.180	2.643	**9.952**	376.790
E2D2 + B	48h	3.171	5.889	19.780	725.805
E3D3 + C	48h	**2.128**	**2.421**	13.081	**359.293**
E1D1 + B	72h	**1.895**	**3.240**	**11.530**	**403.252**
E2D2 + B	72h	2.205	4.095	11.994	490.421
E3D3 + C	72h	2.169	4.630	14.988	552.555

Table 5.6
Comparison of MAE on the test dataset compared on different forecasting periods.

Chapter 6

Conclusion and Future Work

In this study, we have shown that incorporating the meteorological variables improves the track and intensity forecast. Also, the architecture of the encoder-decoder forecast with two ConvLSTM2D layers in the encoding and the decoding network has shown the best result. Thus, the second layer has succeeded in bringing an improvement in learning long-term correlations. However, the meteorological variables may often make it harder for the NN model to predict the location and intensity compared to the basic example of running only on scalar data from HURDAT2. Consequently, thorough development and experiments are required to investigate further the simultaneous prediction of 2-dimensional and one-dimensional TC data.

Feature importance technique is required; for example, using such an approach, like measuring the partial loss functions presented in Table 5.1 can assist in classifying the variables into groups and later performing exclusion tests in the same manner, like in section 5.4.

Moreover, we have shown that convolutional compression has yielded significantly lower values of partial MSE validation loss function for some variables. It would be interesting to compress only those variables for which the compression brought better results than the original data.

It is also desirable to find additional variables like the values of Coriolis force in different locations or data manipulation techniques from modern TC research that will help train the model more efficiently and reach higher levels of precision. An example of such features is the division of the basin into logical clusters, in which the expected behaviour of TCs may differ. Another way to provide additional information could be by treating the coordinate system three-dimensionally rather than as a two-dimensional projection. Additional meteorological variables should also be considered, for example, geopotential height at different pressure levels.

Bibliography

References

[1] Kieran Bhatia et al. “A potential explanation for the global increase in tropical cyclone rapid intensification”. en. In: *Nature Communications* 13.1 (Nov. 2022). Number: 1 Publisher: Nature Publishing Group, p. 6626. ISSN: 2041-1723. DOI: 10.1038/s41467-022-34321-6.

[2] Sébastien Gardoll and Olivier Boucher. “Classification of tropical cyclone containing images using a convolutional neural network: performance and sensitivity to the learning dataset”. In: *Geoscientific Model Development* 15.18 (2022), pp. 7051–7073. ISSN: 1991-959X. DOI: 10.5194/gmd-15-7051-2022.

[3] Alex Graves. *Generating Sequences With Recurrent Neural Networks.* arXiv:1308.0850 [cs]. June 2014. DOI: 10.48550/arXiv.1308.0850.

[4] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. *Speech Recognition with Deep Recurrent Neural Networks.* arXiv:1303.5778 [cs]. Mar. 2013. DOI: 10.48550/arXiv.1303.5778.

[5] Dianyuan Han. “Comparison of Commonly Used Image Interpolation Methods”. en. In: ISSN: 1951-6851. Atlantis Press, Mar. 2013, pp. 1556–1559. ISBN: 978-90-78677-61-1. DOI: 10.2991/iccsee.2013.391.

[6] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-term Memory”. In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[7] Greg Holland and Cindy L. Bruyère. “Recent intense hurricane response to global climate change”. In: *Climate Dynamics* 42.3 (2014), pp. 617–627. ISSN: 1432-0894. DOI: 10.1007/s00382-013-1713-0.

[8] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.* arXiv:1502.03167 [cs]. Mar. 2015. DOI: 10.48550/arXiv.1502.03167.

[9] John Kaplan and Mark DeMaria. “Large-Scale Characteristics of Rapidly Intensifying Tropical Cyclones in the North Atlantic Basin”. EN. In: *Weather and Forecasting* 18.6 (Dec. 2003). Publisher: American Meteorological Society Section: Weather and Forecasting, pp. 1093–1108. ISSN: 1520-0434, 0882-8156. DOI: 10.1175/1520-0434(2003)018<1093:LCORIT>2.0.CO;2.

[10] Philip J. Klotzbach et al. “Trends in Global Tropical Cyclone Activity: 1990–2021”. en. In: *Geophysical Research Letters* 49.6 (2022). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2021GL095774, e2021GL095774. ISSN: 1944-8007. DOI: 10.1029/2021GL095774.

[11] Yunjie Liu et al. *Application of Deep Convolutional Neural Networks for Detecting Extreme Weather in Climate Datasets.* arXiv:1605.01156. 2016. DOI: 10.48550/arXiv.1605.01156.

[12] S. Neetu et al. “Global assessment of tropical cyclones intensity statistical-dynamical hindcasts”. In: (2017). Accepted: 2017-08-31T07:30:51Z Publisher: Royal Meteorological Society, London. DOI: 10.1002/qj.3073.

[13] Timothy Olander et al. “Investigation of Machine Learning Using Satellite-Based Advanced Dvorak Technique Analysis Parameters to Estimate Tropical Cyclone Intensity”. In: *Weather and Forecasting* 36.6 (2021), pp. 2161–2186. ISSN: 1520-0434, 0882-8156. DOI: 10.1175/WAF-D-20-0234.1.

[14] Shanchen Pang et al. “NDFTC: A New Detection Framework of Tropical Cyclones from Meteorological Satellite Images with Deep Transfer Learning”. In: *Remote Sensing* 13.9 (2021). Publisher: Multidisciplinary Digital Pub- lishing Institute, p. 1860. ISSN: 2072-4292. DOI: 10.3390/rs13091860.

[15] Prabhat et al. “ClimateNet: an expert-labeled open dataset and deep learning architecture for enabling high-precision analyses of extreme weather”. In: *Geoscientific Model Development* 14.1 (2021), pp. 107–124. ISSN: 1991-959X. DOI: 10.5194/gmd-14-107-2021.

[16] Jia Ren, Nan Xu, and Yani Cui. “Typhoon Track Prediction Based on Deep Learning”. In: *Applied Sciences* 12.16 (2022). Number: 16 Publisher: Multidisciplinary Digital Publishing Institute, p. 8028. ISSN: 2076-3417. DOI: 10.3390/app12168028.

[17] Mario Rüttgers et al. “Prediction of a typhoon track using a generative adversarial network and satellite images”. In: *Sci Rep* 9.1 (2019), p. 6057. ISSN: 2045-2322. DOI: https://doi.org/10.1038/s41598-019-42339-y.

[18] Xingjian Shi et al. *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting.* arXiv:1506.04214 [cs]. Sept. 2015. DOI: 10.48550/arXiv.1506.04214.

[19] Udai Shimada, Munehiko Yamaguchi, and Shuuji Nishimura. “Is the Number of Tropical Cyclone Rapid Intensification Events in the Western North Pacific Increasing?” In: *Sola* 16 (2020), pp. 1–5. DOI: 10.2151/sola.2020-001.

[20] Jinjie Song, Yihong Duan, and Philip J. Klotzbach. “Increasing trend in rapid intensification magnitude of tropical cyclones over the western North Pacific”. en. In: *Environmental Research Letters* 15.8 (Aug. 2020). Publisher: IOP Publishing, p. 084043. ISSN: 1748-9326. DOI: 10.1088/1748-9326/ab9140.

[21] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. *Unsupervised Learning of Video Representations using LSTMs.* arXiv:1502.04681 [cs]. Jan. 2016. DOI: 10.48550/arXiv.1502.04681.

[22] Joshua Studholme et al. "Poleward expansion of tropical cyclone latitudes in warming climates". In: *Nature Geoscience* 15.1 (2022), pp. 14–28. ISSN: 1752-0908. DOI: 10.1038/s41561-021-00859-1.

[23] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014.

[24] Benjamin C. Trabing and Michael M. Bell. "Understanding Error Distributions of Hurricane Intensity Forecasts during Rapid Intensity Changes". EN. In: *Weather and Forecasting* 35.6 (Oct. 2020). Publisher: American Meteorological Society Section: Weather and Forecasting, pp. 2219–2234. ISSN: 1520-0434, 0882-8156. DOI: 10.1175/WAF-D-19-0253.1.

[25] Liang Wang et al. "Forecasting tropical cyclone tracks in the northwestern Pacific based on a deep-learning model". English. In: *Geoscientific Model Development* 16.8 (Apr. 2023). Publisher: Copernicus GmbH, pp. 2167–2179. ISSN: 1991-959X. DOI: 10.5194/gmd-16-2167-2023.

[26] Wang-chun Woo and Wai-kin Wong. "Operational Application of Optical Flow Techniques to Radar-Based Rainfall Nowcasting". en. In: *Atmosphere* 8.3 (Mar. 2017). Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, p. 48. ISSN: 2073-4433. DOI: 10.3390/atmos8030048.

Appendix

6.1 Hurricane Database (HURDAT2)

HURDAT2 database is developed by National Hurricane Center (NHC) and Central Pacific Hurricane Center (CPHC) and is managed by Atlantic Oceanographic and Meteorological Laboratory (AOML) under National Oceanic Atmospheric Administration US (NOAA).

Copyright notice: As required by 17 U.S.C. 403, third parties producing copyrighted works consisting predominantly of the material produced by U.S. government agencies must provide notice with such work(s) identifying the U.S. Government material incorporated and stating that such material is not subject to copyright protection. The information on government web pages is in the public domain unless specifically annotated otherwise (copyright may be held elsewhere) and may therefore be used freely by the public.

The database is divided into two parts: Atlantic hurricane database 1851-2021, and Northeast and North Central Pacific hurricane database 1949-2021.

The database includes official assessment of the cyclone's history, provided by post-storm analysis of each tropical cyclone in the observation areas, which takes into account observations that are not available in real-time.

Format and variables:

Comma-delimited, text format with six-hourly (00:00, 06:00, 12:00, 18:00 UTC) information on the location, maximum winds, central pressure, and (beginning in 2004) size of all known tropical cyclones and subtropical cyclones. The intensity of the TC is divided into several categories, only the two strongest are usually taken into account for DL research tasks.

Format and variables

Header line format:

- Basin:
  - EP – Northeast Pacific
  - CP – North Central Pacific
  - AL - Atlantic
- ATCF cyclone number for that year
- Year
- Name if available, or else "UNNAMED"

- Number of best track entries – rows – to follow

  Data lines:

- Date
- Hours and minutes (UTC)
- Record identifier:
  - L – Landfall (center of system crossing a coastline)
  - P – Minimum in central pressure
  - I – An intensity peak in terms of both pressure and maximum wind
  - S – Change of status of the system
  - T – Provides additional detail on the track (position) of the cyclone
- Status of system:
  - TD – Tropical cyclone of tropical depression intensity ($<$ 34 knots)
  - TS – Tropical cyclone of tropical storm intensity (34-63 knots)
  - HU – Tropical cyclone of hurricane intensity ($>$ 64 knots)
  - EX – Extratropical cyclone (of any intensity)
  - SD – Subtropical cyclone of subtropical depression intensity ($<$ 34 knots)
  - SS – Subtropical cyclone of subtropical storm intensity ($>$ 34 knots)
  - LO – A low that is neither a tropical cyclone, a subtropical cyclone, nor an extratropical cyclone (of any intensity)
  - DB – Disturbance (of any intensity)

  Location:

- Latitude
- Hemisphere: North or South
- Longitude
- Hemisphere: West or East

  Strength:

- Maximum sustained wind (in knots)
- Minimum Pressure (in millibars)
- Wind radii maximum extent variables (in nautical miles), tracked since 2004. These values describe the size of the TC:
- 34 kt, northeastern quadrant

- 34 kt, southeastern quadrant
- 34 kt, southwestern quadrant
- 34 kt, northwestern quadrant
- 50 kt, northeastern quadrant
- 50 kt, southeastern quadrant
- 50 kt, southwestern quadrant
- 50 kt, northwestern quadrant
- 64 kt,northeastern quadrant
- 64 kt, southeastern quadrant
- 64 kt, southwestern quadrant
- 64 kt, northwestern quadrant
- Radius of Maximum Wind (in nautical miles)

6.2 ECMWF Reanalysis v.5 (ERA5)

ERA5 is produced by the Copernicus Climate Change Service (C3S) at European Centre for Medium-Range Weather Forecasts (ECMWF).

Copyright notice: Copyright ©[2022] European Centre for Medium-Range Weather Forecasts (ECMWF). Source www.ecmwf.int Licence Statement: This data is published under a Creative Commons Attribution 4.0 International (CC BY 4.0). https://creativecommons.org/licenses/by/4.0/ Disclaimer: ECMWF does not accept any liability whatsoever for any error or omission in the data, their availability, or for any loss or damage arising from their use. Where applicable, an indication if the material has been modified and an indication of previous modifications.

It is a global atmospheric reanalysis database, covering the period from January 1950 to the present. New data is available within 5 days in real-time.

Format and variables:

The data is available in GRIB format, which is standardized and commonly used in meteorology to store weather data. It is divided into two parts, single-level and pressure levels measurements.

The data is provided in hourly time rate and on a 30 km spatial grid. Atmospheric, land and oceanic climate variables are provided either as single-level measurements in meters above ground or as pressure level measurements on 137 different levels from the surface up to a height of 80km.

Format and variables

It is a partial list of all variables, including only the ones relevant to my work.

- msl - Mean sea level pressure (Pa)
- tcwv - Total column water vapour (kg/m^2)
- v10 - Northward component of the wind at 10 meters (m/s) 10 metre V wind component
- u10 - Eastward component of the wind at 10 meters (m/s) 10 metre U wind component
- v850 - Northward component of the wind at 850 hPa (m/s)
- u850 - Eastward component of the wind at 850 hPa (m/s)
- t200 - Temperature at 200 hPa (K)
- t500 - Temperature at 500 hPa (K)