Report of master 2 research intership in computer science

Fused deep learning for hurricane track forecast from reanalysis data

Author :
Mo Yang

Internship supervisor :
Claire Monteleoni
Guillaume Charpiat

Host organisation :
Laboratoire de l'Accelerateur Lineaire
Inria

Secretariat - tel : 01 69 15 81 58
e-mail : alexandre.verrecchia@u-psud.fr

Contents

Abstract

Hurricanes is one of the most severe natural disaster that cost tremendous damage and death in each year. The forecast of incoming hurricanes trajectory can be very important for protecting people who live in shoreline areas and reducing their economic loss. In this report, we propose deep learning method for hurricanes trajectory forecasting. We introduce a fusion architecture. The proposed fusion architecture of neural network allows for processing different sources of data in parallel, including past hurricanes movements and reanalysis atmospheric wind fields and geopotential fields images. This fused network is trained to estimate the longitude and latitude 24h-forecast of hurricanes and depressions from a large database from both hemispheres (more than 3000 storms since 1979). It is demonstrated that the fused network which integrate all sources of data have significantly lower error distance than network that use any single source of data. (add compare with state-of-art)

Mots clefs

Hurricanes, Trajectory forecasting, Deep learning, Fusion architecture, Neural networks.

Chapter 1

Introduction

The research is carried out in a mixed team with members from two laboratories: Laboratoire de l'Accelerateur Lineaire and Inria Saclay. The team leaders combine Monteleoni's expertise in climate informatics and algorithms for learning in the presence of spatial and temporal non-stationarity, with Charpiat's expertise in deep learning architectures and applications. This chapter outline the related organizations, teams and the internship subject.

1.1 Laboratory of the Linear Accelerator

The Laboratory of the Linear Accelerator (LAL) is under the joint supervision of the Universite Paris-Sud and the Institut National de Physique Nucleaire et de Physique des Particules (IN2P3) of the CNRS. There are 309 agents on January 1st, 2017, including 135 researchers and 174 engineers and technicians. As the name suggested, the research activity of the LAL is centered on particles physics, supplemented by a strong component in cosmology and astrophysics. Most of the researchers have physicist background. Recently, LAL is enforcing its connection with nearby CNRS institutes and University Paris-Saclay to establish a coherent unit and share competencies and resources.

1.2 Inria research center Saclay

The French Institute for Research in Computer Science and Automation (Inria) is a French national research institution focusing on computer science and applied mathematics. Created in 1967, Inria now has 8 research centers located throughout France, over 2400 employees from 102 countries. 184 project-teams lead research in Inria and 4400 scientific publications were published per year (statistics in 2017)[6]. Inria promotes "scientific excellence for technology transfer and society". The duty of Inria is to respond to the challenges of digital transformation in multidisciplinary applications in fields as diverse as health, energy, security and privacy protection, environment, climate, transportation, economy, finance, agriculture...

The Inria Saclay-Île-de-France Research Centre was created in 2008 and is located at the heart of the main national research and higher education cluster. It is also a member of the Université Paris Saclay, and a major actor in the French Investments for the Future Programme (Idex, LabEx, IRT, Equipex). 450 researchers and engineers from Inria and its partners work in the center's 31 teams. The center is particularly active in three major areas: data and knowledge; safety, security and reliability; modeling, simulation and optimization (with priority given to energy)[1].

1.3 Related teams

Applied Statistics and Machine Learning (AppStat) is a group inside LAL with an initiative focusing on interdisciplinary projects between physics and computer science. Now the group acts in a variety of domains including image and music processing, bioinformatics, climate informatics, software engineering, grid control, experimental physics... The head of AppStat is Balazs Kegl. Members of the groups come from physics or computer science background. Each person carried on his or her research, but they can exchange ideas and share the experience with other research intuitions. Balazs Kegl and the team are also acting as coordinators of Paris-Saclay Center for Data Science (PSCDS), which forms a larger machine learning community. PSCDS allow researchers in the frame of Paris Saclay to create data challenge from their research and let others compete and collaborate. The goal is to link the people in the different background (both data scientists and domain experts) to cooperate in a complementary way[9].

TAO is the mixed INRIA Saclay - CNRS - LRI, University Paris-Sud research group interested in the interplay of Machine Learning (A for Apprentissage) and Optimization (O). Since 2016 team TAO in Inria is stopped and is replaced by two new teams: RANDOPT(RANDomized OPTimization) and TAU(TAckling the Underspecified). RANDOPT team locates at the Applied Maths Center at Ecole Polytechnique, works on all types of blackbox optimization especially CMA-ES type. TAU team is situated at LRI(laboratoire de recherche en informatique) in University Paris-Sud, aims to tackle the vagueness of the Big Data purposes[3] in three respects: providing machine with **commen sense**, steering a Big Data system in a dynamic environment, and trading-off between efficiency and protecting individual freedom and privacy[10]. The main research dimensions in TAU involve Causal Modeling (required to support prescriptive Big Data), Deep Learning (related to constructive representations, and their compositionality), Optimization and Meta-Optimization (including sequential decision making and categorization of problems), and Big-Data Driven Design.

1.4 Research team regarded to internship subject

The research is carried out in a mixed team with members from the two groups above. Claire Monteleoni is a visiting professor to AppStat in LAL, she works on machine learning algorithms and recently focuses on climate informatics. Guillaume Charpiat is a deep learning researcher in TAO/TAU team in the INRIA Saclay. His research interest covers neural network theory, computer vision, and optimization. Balazs Kegl is a researcher in the field of computer science in LAL and head of AppStat. His has a broad research interest in machine learning while largely focusing on standardizing workflow for machine learning production and facilitating the use of machine learning methods for the non-expert. Sophie Giffard-Roisin is a post-doctoral researcher at AppStat team in LAL. Her research focuses on applied maths and computer science, and more particularly machine learning techniques for computational sustainability and climate informatics.

1.5 Internship Subject

In recent years, a sequence of Atlantic hurricanes surprised and crippled several highly inhabited areas. Cuba had very little warning of the updated track of hurricane Irma before it was hit[8], and Puerto Rico was just reeling from hurricane Irma when it was devastated by hurricane Maria two weeks later. Better understanding and predicting the development and movement of such severe storms (hurricanes, cyclones, typhoons: names differ by region) is critical to protecting communities and ecosystems. This research focuses on exploring the capacity of machine learning approaches to improve predictions of severe storms. The study addresses the following questions:

- RQ1 . Can machine learning contribute to understanding and predicting the tracks and intensity of severe storms (hurricanes, cyclones, typhoons)?
- RQ2 . is deep learning an effective approach to such problems?

1.6 Outlines of the rest of the report

The rest of the report is organized as follows:

Chapter 2 describes the context of hurricane track forecasting. Then Chapter 3 sets up the problem and presents the related prior knowledge to the research. Our proposed method is described in Chapter 4. Chapter 5 will be dedicated for the experiments and analysis. Chapter 6, as a concluding chapter, summarizes the main findings and identified future research opportunities as well as critiques of the current research.

Chapter 2

Background

2.1 Hurricane Trajectory Forecasting

Cyclones, hurricanes or typhoons are words for the same phenomena: a rare and complex event characterized by strong winds surrounding a low-pressure area. Hurricane is one of the most severe natural disasters that cost tremendous damage and death each year. During its life cycle, a hurricane can expend as much as 10000 nuclear bombs [5]. Since 2000, over 45000 people were killed by hurricanes. In 2005, hurricane Katrina had cost as much as 125 billion US Dollars worth of damage. The explosion of the population also raises the risk of potential damage made by hurricanes[33]. In 2010, it was estimated that 1.53 billion people lived in hurricanes prone areas in 81 different countries and territories. 133.7 millions of people were exposed to hurricanes. The number has increased three times compared to the year 1970. Hurricane's huge threat to human society makes the forecast of hurricane trajectory crucial, in order for people living in that area to be alarmed and get some time to protect their goods and evacuate. Improving the prediction of hurricane movements has a significant impact on society.

Formation of Hurricanes

The evolution and path of hurricanes depend on many factors at different scales and altitudes. All hurricanes are formed over warm ocean water near the equator. Warm and moist air over the ocean rises upward, which cause low-pressure area below. The air from surrounding high-pressure areas flows over into low-pressure areas, and new air is heated and rise upward again. As warm and moist air rise and cools off, the water in the air forms clouds. Figure **??** shows the wind flows inside storm. The storm grows up and rotates as the cycle continues to be fed by ocean's heat and water evaporating from the surface. When the winds in the rotating storm reach 39mph(miles per hour, equivalent to 63km/h), the storm is called a "tropical storm". When the wind speeds reach 74 mph, the storm is officially called a "tropical cyclone", or hurricane. The Saffir–Simpson hurricane scale (SSHS) classifies hurricanes into five categories distinguished by the intensities of their sustained winds. The highest classification in the scale, category 5, consists of storms with sustained wind speeds higher than 156mph, See Figure2.1. To avoid ambiguity, the term 'storm' will be used for

Saffir-Simpson Hurricane Scale		
Category	Wind Speed	
	mph	knots
5	>=156	>=135
4	131-155	114-134
3	111-130	96-113
2	96-110	84-95
1	74-95	65-83
Non-Hurricane Scale		
Tropical Storm	39-73	34-64
Tropical Depression	0-38	0-33

Table 2.1: The Saffir-Simpson Hurricane wind scale and Non-Hurricane Scale

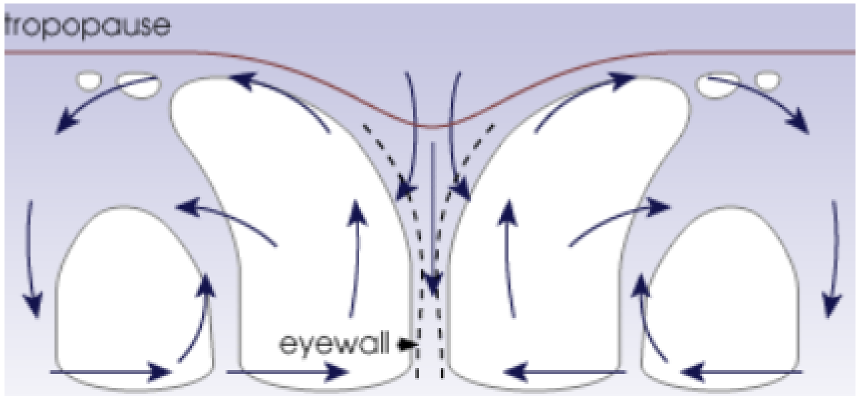

Figure 2.1: Graphic of wind flow inside storm [5]

all kinds of categories in SSHS, and the term 'hurricane' will only be used for those in category 5 in the report.
A hurricane can be very unpredictable. The hurricane movement can perform loops, hairpin turns, and sharp curves. Its intensity can also change dramatically in a very short time. In 2005, hurricane Katrina was only classified as a tropical storm when it made its first landfall in Florida. After emerged into the Gulf of Mexico, it rapidly strengthened into a Category 5 hurricane in two days. Its second landfall finally caused the largest damage in history. The track and evolution of Hurricane Katrina are shown in Figure 2.2 Global warming is also considered to influence hurricane activities. It is estimated that global warming will cause more intense hurricanes globally in the coming century and have higher rainfall rates[26], which may lead to both more representative and more consistent error statistics for forecasting.

2.2 Forecast Method for Meteorologists

Tracks and intensity are the two main goals of the forecast. Today, the forecasts (track and intensity) are provided by a numerous number of guidance models [7]. In the 70's and 80's the models were statistical models based on historical relationships between storm behavior and various other parameters[15]. Today statistical models are mostly used for testing and comparing new models. With the increase of computing capacity and data assimilation techniques, dynamic models gradually replaced statistical models. Dynamical models solve the physical equations governing the motions in the atmosphere, but they are very complex and computationally demanding. These models usually run on high-speed supercomputers.

Guidance models are characterized as either early or late, depending on whether or not they are available to the Hurricane Specialist during the forecast cycle. [7] The late model can take hours to run. For example, if the NWS/Global Forecast System (GFS) runs at 12UTC, the result will not be available until 16UTC. The Dynamic models, in general, are late models. Although they can have a very precise forecast, the bottleneck is the computing speed. If the computation lasts for days, it doesn't make sense to make a prediction since the prediction will already be outdated. But fortunately, there is a technique that allows to take the latest available run of a late model and adjust its forecast to apply to the current synoptic time and initial conditions[7]. Current national forecasts are typically driven by consensus methods able to combine different dynamical models. The forecasts are usually calculated to have from 12h-predictions, 24h-predictions, 36h-predictions

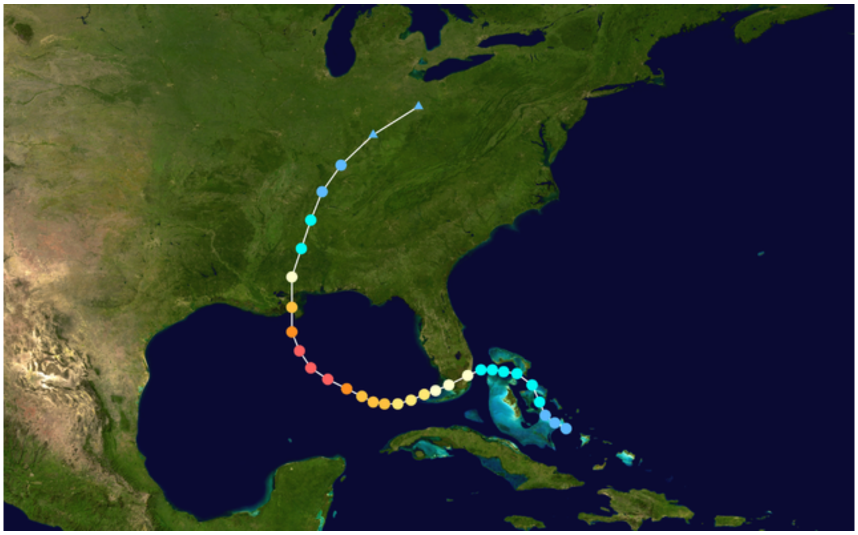

Figure 2.2: Hurricane Katrina track. Uses the color scheme from the Saffir-Simpson Hurricane Scale. The points show the location of each storm at six-hour intervals. The color represents the storm's maximum sustained wind speeds as classified in the Saffir-Simpson Hurricane Scale (see 2.1), and the shape of the data points represent the nature of the storm [4]

and up to 120h-predictions.

2.3 Machine Learning in Climate Science

The hurricane statistical forecasting models perform poorly with respect to the dynamical models, even if the database made of past hurricane is constantly growing. The machine learning methods, able for example to capture non-linearity and complex relations, have only been scarcely tested. Artificial Neural Network have recently shown their efficacy in a various number of forecasting tasks. Particularly, convolutional neural networks (CNN) have raised attention as they are suited for large imaging data. A convolutional LSTM model has been used for precipitation forecast in a promising study [40]. Another recent study predicts the evolution of sea surface temperature maps by combining CNN with physical knowledge [13]. CNNs have also been used for the detection of extreme weather like hurricanes from meteorological variables patches[34]. These studies show the strong potential on various climate problems.

To our knowledge, only two preliminary studies have tackled the hurricane forecast tracking using machine learning: the first one uses random forests on local reanalysis histograms [31], however, it was tested only on tropical storms in 2015 in North Atlantic, and the mean error of 6h-forecasts seem to indicate poor results (more than 60km). The second uses a sparse recurrent neural network from trajectory data [32]. It was tested on only four hurricanes and also seem to give large distance errors (mean 6h-forecast error is 72km).

CHAPTER 3

Prelimiaries

3.1 Problem Setting

The goal of the research is to predict the trajectory of Hurricane using the information from the past storms since 1979. The task is shown in Figure 3.1. We aim at building an end-to-end model using two types of data (reanalysis and hand-crafted features), for each time step of each storm, we want to independently predict its displacement in the future(depending on forecast cycle).

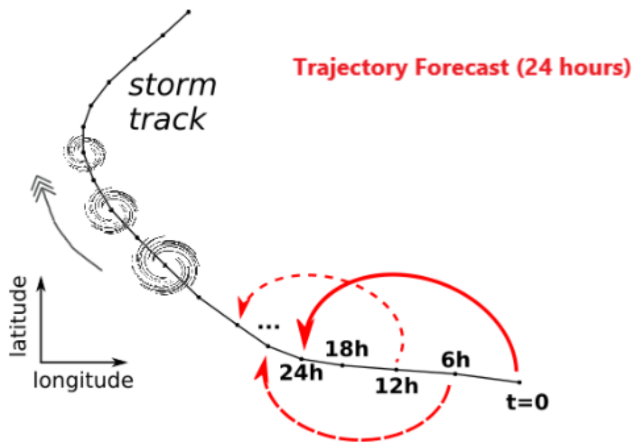

Figure 3.1: General architecture: the two types of data are feeding two neural networks trained separately. The final fused network is re-trained before predicting the forecast track.

The objective of the internship is to study for using deep learning as an alternative methodology for hurricane trajectory forecasting. Our research is based on a large body of work on recent advances in convolutional neural networks. We will briefly describe the related literature to our approach in the following part of this chapter.

3.2 Neural Network for sequence modeling

From the machine learning perspective, hurricane track forecasting can be essentially taken as a spatiotemporal sequence forecasting problem, where the input can be spacial atmospheric fields evolving with time, and the output can be a fixed number (1 or more) of hurricane's future displacement. The problem is difficult to solve in the first place due to the high dimensionality of the spatiotemporal sequences. It is important to build a prediction model that could effectively learn patterns from both its spatial and temporal structure.

From the machine learning perspective, hurricane track forecasting can be essentially taken as a spatiotemporal sequence forecasting problem, Recent advances in deep learning have shown promising results on enormous pattern recognition tasks, such as ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [35] [29] [38] and natural language processing [18] [37]. Weather deep learning could open a new track for hurricane track forecasting is a key problem we want to tackle in this research. The artificial neural network has also been largely used for sequence modeling. In general, Convolutional Neural Network (CNN) and Recurrent Neural Network(RNN) are the two main architectures that are dedicated to sequence modeling.

For most deep learning practitioners, Recurrent Neural Network(RNN) is the default choice for sequence modeling. RNN is a type of neural network that maintains a hidden vector backpropagated through time. Basic RNN is known difficult to train. For long sequences, backpropagation causes gradient exploding or vanishing problem. More advanced architectures are used instead in practice such as LSTM[22] and GRU [12], which are proven to be robust for having long-range dependencies, and doesn't suffer from gradient exploding or vanishing. The family of RNN has been widely used in language modeling[19] and machine translation[37]. While for our problem, standard LSTM or GRU are not suitable models for end-to-end learning because they cannot tackle with high-dimensional input tensor data.

CNN has been applied to sequences for a long time. In the earlier days Time Delay Neural Network (TDNN) was used for speech recognition [39] that inspired largely to CNN. Later CNN has been widely applied to NLP tasks and achieved excellent results in sentence modeling [23], classification [24], prediction [12]... Compared to RNNs, CNNs are faster and use less memory bandwidth on the same sequence modeling tasks. In particular, a recent application of CNN to machine translation [16] achieved state-of-art accuracy at nine times the speed of recurrent neural systems. Another study has carried out a comprehensive comparison between a generic convolutional model and canonical recurrent models such as LSTMs and GRUs on typical sequence modeling tasks that are commonly used to benchmark RNN themselves, and concluded "a simple convolutional architecture outperforms canonical recurrent networks across a diverse range of tasks and datasets, while demonstrating longer effective memory" [11].

There are also works that aim at combining RNN and CNN architectures. Convolutional LSTM replaces fully-connected layers in LSTM with convolutional layers to capture the spatiotemporal structure of the data in a weather nowcasting problem [40]. This work shows promise in combining different architectures of neural networks. But it needs sufficient training data and is difficult to train. In our research, we consider using the convolutional network as our starting point.

3.3 Convolutional Neural Network

An example of CNN is presented in Figure 3.2 It is usually made of several convolutional layers followed by some fully connected layers. The convolutional layer is inspired by the animal's cortex visual system, where each neuron only processes data for its receptive field. Between two successive convolutional layers, there is usually also sub-sampling layers also known as pooling layers. The Advantage of CNN is that it learns both spatial and temporal differences in different scales and thus could extract features automatically without learning a massive number of parameters. The CNNs are widely adopted as a very effective model for analyzing images or images-like data for pattern recognition.

Modern CNNs tend to be deep with a large number of hidden layers. AlexNet is seen as a breakthrough when it won the 2012 ImageNet LSVRC-2012 competition by a large margin (reached 15.3% compared to the previous best 26.2% in error rates on top-5 classification problem). AlexNet has five convolutional layers followed by three fully-connected layers. Researchers continually introduce new architectures of CNNs that are deeper, more complex and have better precision on ImageNet. The CNNs, as deep learning in general, is an empirical construction of a learning algorithm. Some works in CNNs show that CNNs should have more layers for the hierarchical representation of visual data to work[21]. VGG Net[36] is one of the most influential works that gives a guideline to design a CNN architecture. A VGG Net applies alternatively convolutional layers and max-pooling layers through the whole network and put several fully connected layers at the end of the network to generate output. Rather than using relatively large receptive fields in

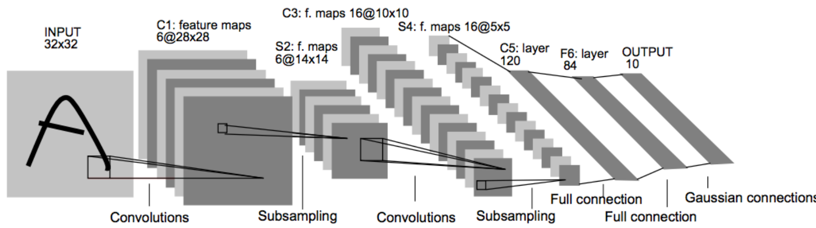

Figure 3.2: Architecture of LeNet, shows a series of layers in CNN that learn hierarchical features representation [30]

the first convolutional layers (e.g., 11x11 with stride 4 in AlexNet[29]), the author suggests using very small 3x3 receptive fields throughout the whole network. The reason is that there could be more non-linear layers and a smaller number of parameters, and effectively to have scopes from small to large receptive fields. It turns out that the more important for VGG Net becomes tunning the depth of the network without worrying about other hyper-parameters like kernel size. It is the simplicity and effectiveness of VGG Net that makes it among the most popular CNN architectures.

CNN is known to be tedious to train. To build an excellent CNN, researchers have proposed different methods. Some of them are crucial to successfully train a CNN. Training a CNN with a huge number of learnable parameters usually require a considerable amount of examples. The data Augmentation methods such as flipping or randomly cropping can effectively increase the network's performance when there is only a limited number of examples. Proper input data normalization and weights initialization are essential steps to accelerate convergence during the training. To realize it, the input data can be zero-centered and normalized at each channel to have a similar distribution. Weights are usually initialized by random weights drawn by Gaussian distributions[29], but it can lead to a very slow convergence when training very deep CNN. A paper[17] shows that the reason is that the norm of the outputs in each layer does not initially sum to 1. If the model has too many layers, the norm will be exploded. The author proposes a method called 'Xavier' initialization to ensure the outputs in each layer are initially standard normally distributed. 'Xavier' initialization is based on the assumption that activations are linear, which is not valid on rectifier nonlinearities(ReLU). Another paper[20] suggests a more robust initialization that deals with the rectifier nonlinearities. Batch Normalization(BN) layers are also recommended to be used to maintain the distribution of the outputs in each layer during training and thus to accelerate convergence[29].

Chapter 4

Method

4.1 Data Description

Storm track data

The raw storm track data used in this research is composed of more than 3000 extra-tropical and tropical storm tracks since 1979 extracted from the NOAA database IBTrACS[25], see Figure 4.1. The tracks are defined by the 6-hourly center locations (latitude and longitude) for the entire lives of the storms. They come from both hemispheres, and the number of records per storm varies from 2 to 120 time-steps. In total, the database counts more than 90 000 time steps.

Reanalysis

Reanalysis is a systematic approach to produce datasets for climate monitoring and research[2]. Reanalyses are created via an unchanging ("frozen") data assimilation scheme and model(s) which take all available observations as inputs every 6-12 hours over the period being analyzed. The raw input data can include but not limited to radiosonde, satellite, buoy, aircraft and ship reports. The framework generates dynamically consistent estimate of the climate atmospheric fields or state at each time point. Reanalyses can cover the entire globe from the Earth's surface to well above the stratosphere and have hundreds of available variables. The data is structured and easy to handle from a processing standpoint. In general, the model resolution and biases have been steadily improved over time. While Reanalyses also have limitation. The reliability of the data can vary depending on the location, period, and variables considered due to observational constraints.

The ERA-Interim is one of the reanalysis datasets that cover the data-rich period since 1979 to date. The ERA-Interim is the latest global atmospheric reanalysis produced by The European Centre for Medium-Range Weather Forecasts (ECMWF) and is continued in real time. The spectral

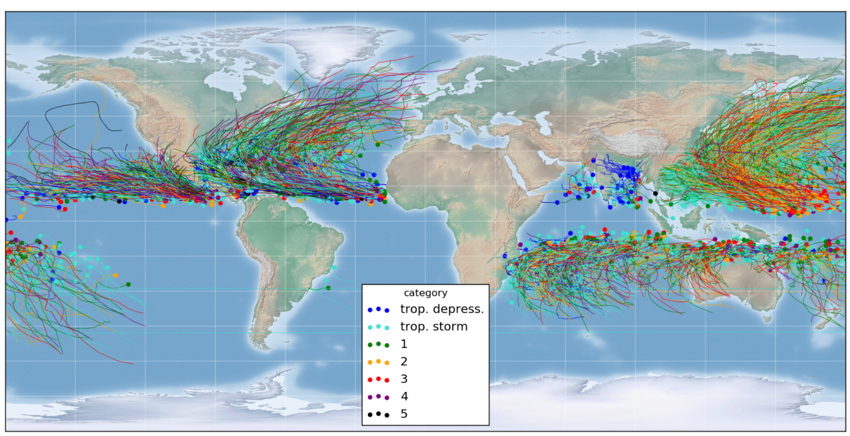

Figure 4.1: Database: more than 3000 tropical/extra-tropical storm tracks since 1979. Dots = initial position, colors = maximal storm strength according to the Saffir-Simpson scale.

resolution is T255 (about 80 km), and there are 60 vertical pressure levels, with the model top at 0.1 hPa (about 64 km).

4.2 Feature selection

In the research, we use storm track data and reanalysis to predict hurricane's track in the future. To capture the movement of a storm, we have classified four sources of information:

1. **The wind fields.**The trajectory of a storm depends on large-scale atmospheric flows. Wind fields are the direct observation of the atmospheric flows. Thus, we extracted the wind fields of the neighborhood of the storm at every time step from the ERA-interim reanalysis database [14]. Specifically, we extracted the u-wind and v-wind fields on a $25x25$ degree grid centered on the current storm location, at three atmospheric pressure levels (700 hPa, 500 hPa, and 225 hPa). The choice of the three pressure levels was driven by statistical forecast models [15].

2. **The geopotential height fields.** Atmospheric flows are directly due to the existence of **pressure gradients**, because particles in a fluid manner naturally flow from areas of higher pressure to areas of lower pressure. Geopotential height is a vertical coordinate referenced to Earth's mean sea level which is an adjustment of geometric height using a variation of gravity with latitude and evaluation (gravity changes with different latitude). Geopotential height has a positive correlation with pressure at a certain **pressure level**. For example, if somewhere has a higher geopotential height at a certain pressure level, it means that at the same geometric level that place has a higher pressure. In meteorology, scientists often use geopotential height as a function of pressure to facilitate calculation. Similar to wind fields, we extracted the geopotential height fields of the neighborhood of the storm at every time step on a $25x25$ degree grid centered on the current storm location, at three atmospheric pressure levels (700 hPa, 500 hPa, and 225 hPa).

3. **Displacement in history** A storm's future displacement can be predicted from his historical displacement in a statistical approach.

4. **Other hand-crafted features.** Other useful features we extracted from a storm are: **current latitude and longitude**, **current windspeed** at the center of the storm, **Jday predictor**(Gaussian function of "Julian day of storm init - peak day of the hurricane season"[15]), and **current distance to land**. We call them meta data.

Another reason why we focused on the wind and geopotential parameters is that we applied a sparse feature selection technique (Automatic Relevance Determination, based on linear regression) over all available reanalysis fields, which highlighted the usefulness of wind and pressure.

4.3 Overview of Proposed Method

Because of the different nature of the data sources, it is not straightforward to mix all the data as a common input to a bigger network. We propose a fusion neural network architecture taking into account past trajectory data and reanalysis atmospheric physical fields images. An overview of the architecture is shown in Fig 4.2. We devise our fusion architecture accordingly, dividing into 3 streams: **Wind CNN**, **Pressure CNN** and **Past tracks + meta NN**. Wind CNN and Pressure CNN are convolutional neural networks that take atmospheric fields as input, Past tracks + meta NN is a small neural network which takes 0D features as input. Each stream network is supposed to make prediction independently. We fine-tune the parameters of each individual stream network for the same task of predicting the forecast track. Then we integrate the three networks into a fusion network and retrain the parameters. The different steps will be outlined in the following sections.

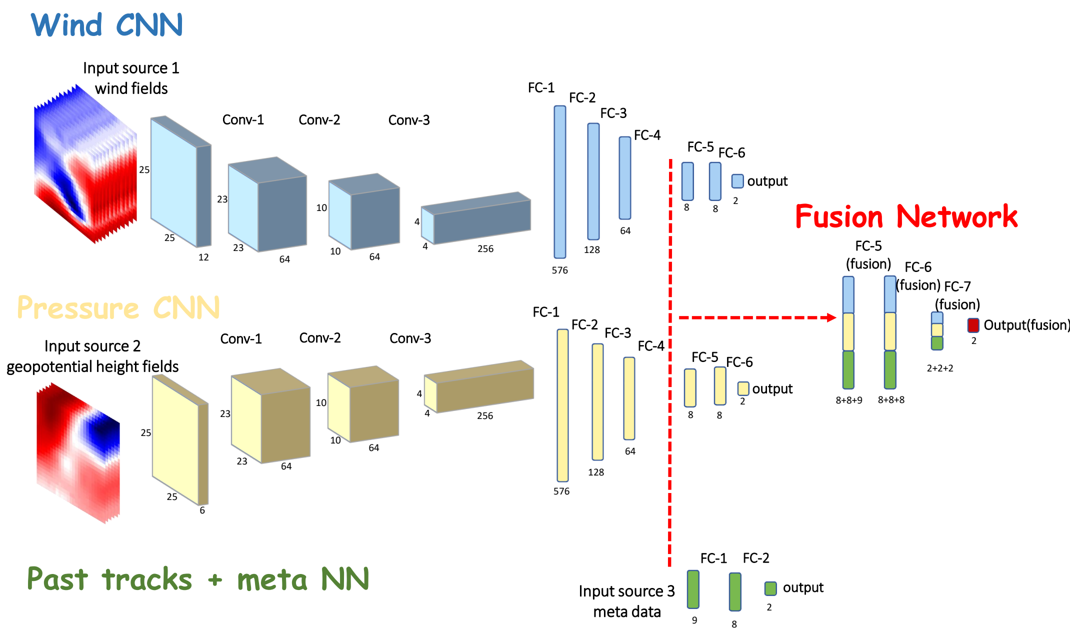

Figure 4.2: General architecture: the three types of data are feeding three neural networks trained separately. The final fused network is re-trained before predicting the forecast track

4.4 Wind CNN and Pressure CNN

In this section, we discuss the architectures of Wind CNN and Pressure CNN(the blue and yellow streams in Fig. 4.2). The two networks are described in the same section because they are very similar in which they both extract information from atmospheric fields. The input data for Wind CNN and Pressure CNN is the centered atmospheric fields at different pressure levels at the current time step as well as its consecutive time steps in history. An essential problem is how many time steps should be taken at each time. We started by taking wind fields measured at t and $t - 6h$ at the same locations. We use a simple data structure: The input of Wind CNN can be seen as 12 images of size 25x25, including u, v components of wind fields at three atmospheric pressure levels (700 hPa, 500 hPa, and 225 hPa) at the two time-steps. And the input of Pressure CNN has six images of size 25x25, including geopotential height fields at the three pressure levels at the two time steps.

We used as a guideline a typical CNN architecture alternating convolutional layers and max-pooling layers and added several fully connected layers at the end of the network [36]. All hidden layers are equipped with the rectification (ReLU) non-linearity. Batch normalization is used after each Conv layer and FC layer except the last layer. To measure the difference in performance brought by increasing number of convolutional layers, we have evaluated several configurations of Wind CNN and Pressure CNN.

The configurations we have evaluated for Wind CNN and Pressure CNN are outlined in Table 4.1, one per column. All configurations follow the generic design described above and differ only in depth. In Table 4.2 we have shown the number of learnable parameters in each of those configurations. We note that those configurations have almost the same number of learnable parameters despite

Table 4.1: **ConvNet configurations**(shown in columns). The depth of the configurations increases from left(A) to the right(D), as more layers are added. The convolutional layer parametres are denoted as "conv(kernel size)-(number of output channels)". The ReLU activation layer and Batch normalization layers are not shown in figure

<table>
<tr><td colspan="4">ConvNet Configurations</td></tr>
<tr><td>A</td><td>B</td><td>C</td><td>D</td></tr>
<tr><td>7 layers</td><td>8 layers</td><td>9 layers</td><td>10 layers</td></tr>
<tr><td colspan="4">input (25*25, 12 channels image)</td></tr>
<tr><td>conv3-32
maxpool</td><td>conv3-32
conv3-32
maxpool</td><td>conv3-64
conv3-64
maxpool
conv3-256</td><td>conv3-64
conv3-64
maxpool
conv3-128
conv3-256
maxpool</td></tr>
<tr><td colspan="4">FC-576</td></tr>
<tr><td colspan="4">FC-128</td></tr>
<tr><td colspan="4">FC-64</td></tr>
<tr><td colspan="4">FC-8</td></tr>
<tr><td colspan="4">FC-8</td></tr>
<tr><td colspan="4">FC-2</td></tr>
</table>

Table 4.2: Number of parameters (in millions)

Network	A	B	C	D
Number of parameters	2.27	2.33	2.75	2.67

their different depths. Training details and results can be found in the next chapter where all experiments are described.

4.5 Past tracks + meta NN

Another important source of information is the previous displacements and the other handcrafted features that we mentioned in the prior part of this chapter. They can be treated as 0D vector. Temporal patterns can be extracted from a storm's historical tracks. Other handcrafted features can also provide additional (and important) clue for prediction. We designed a small neural network (two small fully connected layers, the green stream in Fig.4.2) able to learn the future track from this past track and other handcrafted data. The past displacements of a storm are defined as the storm's displacement every 6 hours in history. An important question is how long should we take from a storm's past tracks. We have evaluated the network for the task of predicting storm displacement in 24 hours using different length of past tracks. We finally decided to use two past tracks (the displacement from $t-12h$ to $t-6h$ and $t-6h$ to t) because we noted that using longer past tracks doesn't improve the performance significantly.

4.6 Fusion Network

It is demonstrated that the three network: Wind CNN, Pressure CNN, Past tracks + Meta NN can independently predict a storm's future displacement. We then consider improving the performance

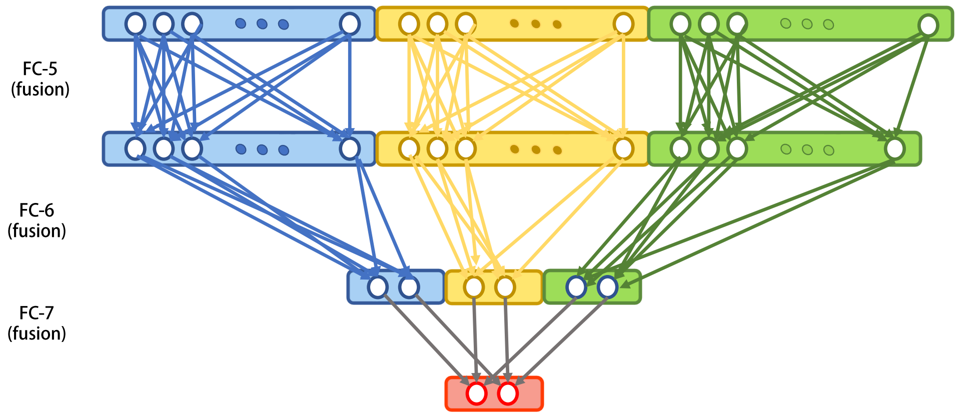

Figure 4.3: Connection of neurons in fusion layers before training fusion network.

by integrating the three networks into a fusion network. Once the three individual stream networks are trained, we concatenate their certain number of last layers and add a layer at the end of the network as the fused output layer. An example of fusion network is shown in Fig.4.2 where the last two layers of each stream networks are concatenated and merge into the fusion network.

Figure 4.3 zooms in the fusion layers and explains how neurons are connected before training the fusion network. After fusing, the connections in previous stream networks(blue, yellow and green) are retained, no connection exists between different streams. The fusion output (red in Figure 4.3) is the average of the three stream networks' outputs.

We are also interested in different possible configurations of the fusion network. We have carried out a throughout evaluations on the performance of different configurations. Details and results will be shown in the next chapter.

4.7 Network Training Framework

In the previous sections of the chapter, we presented the details of the network configurations. In this section, we describe the algorithmic details of network training.

Input preprocessing

Following the machine learning standard, we divided the entire dataset into train (60%) / validation (20%) / test (20%). To avoid data leakage, we first separate all the storms into the three bins. Then, within each set, all time instants from the storms were treated independently. It ensures that training examples, validation examples, and test examples are extracted from different storms. Before being fed into the models, the input data is first standardized by subtracting mean value then dividing the standard variation of each layer, computed on the training set, from each pixel.

Training

Let $D = \{(x_w^1, x_p^1, x_{0d}^1, loc_t^1, loc_{t+\delta t}^1), ..., (x_w^n, x_p^n, x_{0d}^n, loc_t^n, loc_{t+\delta t}^n)\}$ be the labeled data for training our fusion network; with x_w^i, x_p^i, x_{0d}^i denoting wind fields, pressure fields, 0d features respectively and loc_t^i, $loc_{t+\delta t}^i$ corresponding to the ground truth location at current time and after δt time. δt can be 6 hours, 12 hours, 18 hours, 24 hours... depending on the specific forecasting task. We train our model using a two-stage approach.

1. **Training the stream networks:**
   We first proceed by training the individual stream networks. For training Wind CNN, Let F_{wind} be the mapping function of Wind CNN. Learning F_{wind} requires the estimation of the network parameters θ_w which is carried out by minimizing the loss between the forecast storm location and the corresponding ground truth location. The forecast value and ground truth are in latitude and longitude. We first transform their differences in latitude and longitude into the distance in kilometers. Let $F_{transform}$ be the transformation function from differences in latitude and longitude to distance. Then we use Mean Squared Error (MSE) in kilometers as the loss function:

$$L(\theta_w) = \frac{1}{n}\sum_{i=1}^{n} \left\| F_{transform}((F_{wind}(x_w^i;\theta_w) + loc_{i\ t}) - loc_{i\ t+\delta t}) \right\|^2 \tag{4.1}$$

   Where n is the total number of instances. Using MSE as the loss function can effectively penalize on large errors. Training the Pressure CNN and Past tracks + meta NN can be performed analogously. After training, the three stream networks can be used to perform separate prediction of each modality.

2. **Training the fusion network:**
   The training of the fusion network also follows two stages. We divide the layers of the fusion network into two parts: fusion layers (layers that are fused) and stream layers (layers that are retained from stream networks). The division is clearly shown in Fig.4.2. In the first stage, we optimize only the weights in the fusion layers (keeping the weights of the stream layers intact). In the second stage, we let the weights in the whole network be equally optimized. Let F_f be the function of fusion network, θ_f, θ_s be parameters of fusion layers and of stream layers. Analogous to Eq.4.1, the first stage's loss function would be:

$$L(\theta_f) = \frac{1}{n}\sum_{i=1}^{n} \left\| F_{transform}((F_f(x_w^i, x_p^i, x_{0d}^i;\theta_f) + loc_{i\ t}) - loc_{i\ t+\delta t}) \right\|^2 \tag{4.2}$$

   And second stage's loss function would be:

$$L(\theta_f, \theta_s) = \frac{1}{n}\sum_{i=1}^{n} \left\| F_{transform}((F_f(x_w^i, x_p^i, x_{0d}^i;\theta_f, \theta_s) + loc_{i\ t}) - loc_{i\ t+\delta t}) \right\|^2 \tag{4.3}$$

   We found that having the two stage of optimization augment the performance compared to optimize everything from the beginning.

Implementation Details

The training was performed by mini-batch gradient descent with Adam optimizer. The batch size was set to 256. The training was regularized by weight decay (the L_2 penalty multiplier set to

0.01). Initial learning rates were fine-tuned individually in each model. The weights of the model were initialized followed by He initialization[20]. Each model converged within 200 epochs. All sample were permuted after each epoch. Each evaluation was repeated by three times with different randomization and then compute an average score. Our implementation used PyTorch 4.0. All training and evaluation were processed on 2 TitanX GPUs with data parallelism[27].

CHAPTER 5

Experiments

In this section, we describe important experiments in the research. We first determine the best-performing fusion network setting followed by a two-step manner on the validation set and evaluate it on the test set. Then we compare our fusion network with the existing forecast models under the same criterion.

5.1 Configurations comparison

Selecting the fusion network's configuration can be seen as a special hyper-parameter tunning. In the previous chapter, we explained the method of designing the fusion network configuration. In this section, we will describe the related experiments and results. Evaluations about selecting network configurations are processed on validation set since it is part of hyper-parameter tunning. The process can be identified into two steps. Our fusion approach aims at combining the stream networks. In the first step, the stream networks' configurations are determined. In the second step, we explore different fusion strategies and select the fusion network configuration with the best performance. The two steps are described as follows:

Selecting Stream Network Configuration

In the previous chapter, we have outlined the four candidate architectures (shown in Table. 5.1) for Wind CNN and Pressure CNN. We evaluated their performance on 24 hours of storm track prediction. The result of the evaluation on validation set (Wind CNN) is shown in Table 5.1. We give two scores: Mean Square Error(MSE) and Mean Absolute Error(MAE). Specifically, the MAE denotes the model's mean absolute prediction error in kilometers. With the increase of depth, we observed very unobvious improvement on the result. A possible reason could be that climate patterns are so simple that don't need more convolutional layers. Another possible reason is that it is underfitting due to lack of labeled training data. Nevertheless, more convolutional layers mean that the network can learn features at more levels of abstraction. Finally, we choose Network C after compromising between depth and computational efficiency.

Another experiment is to evaluate how adding more temporal parts to the input data structure can improve the performance. We then added storm fields data at the same location at more consecutive time steps to input data. We process the input data on the same architecture and don't observe noticeable improvement. Then we decided to use only information at t and $t-6h$.

Table 5.1: Performance of candidate configurations (Wind CNN) on 24 hours storm track prediction on validation set using wind fields

Model(in Table. 4.1)	Mean Sqaure Error (km$\hat{2}$)	Mean Abosolute Error(km)
A	31430.08	145.43
B	31761.95	146.62
C	31552.91	145.5997
D	31772.62	146.73

Table 5.2: Comparison of fusion models performance on 24 hours storm track prediction on validation set

Model	Mean Sqaure Error(km^2)	Mean Abosolute Error(km)
Fus network, 3 streams, fuse 2 FC	25453.48	130.04
Fus network, 3 streams, fuse 4 FC	25628.22	130.27
Fus network, 3 streams, fuse 6 FC	25846.74	130.95
Fus network, 3 streams, fuse 6 FC + 1 Conv layers	25928.81	131.17

Table 5.3: Comparison of fusion models performance on 24 hours storm track prediction on validation set

Model	Mean Sqaure Error(km^2)	Mean Abosolute Error(km)
0D NN	51367.42	180.00
Pres. CNN,	39387.76	164.91
Wind CNN	30426.02	142.71
Fus network, 0D + Wind, fuse 2 FC	28723.56	136.64
Fus network, Pres. + Wind, fuse 2 FC	28045.30	137.26
Fus network, Three streams, fuse 2 FC	25453.48	130.04

Selecting Fusion Network Configuration

After confirming the stream networks' configuration, we then proceed to explore different fusion strategies. Therefore two scenarios are considered: 1) how many layers should be fused? 2) Does fusing three streams outperforms fusing two streams or using single stream?

To answer the first question, we have evaluated four fusion networks that are based on the same three pre-trained stream networks and only differ in the number of fused layers. The result is shown in Table 5.2. The fusion network that fuses 2 FC performs slightly better than others, but generally, the four networks have the same level of performance.

To answer the second question, we have compared the fusion network fusing all three streams with networks fusing two streams and single stream networks. The result is shown in Table 5.3. We can see the improvement of fusing all three stream networks concerning fusing two stream networks or using single stream networks.

Figure 5.1 shows the 24h-forecast results on the test set in absolute distance error. It is verified that the fusion network fusing all three stream networks outperform other networks in Table 5.3.

5.2 Comparison with existing forecasting models

In this section, we compare our fusion model CNN on 24 hours forecast with the existing forecasting models: OFCL and BCD5. BCD5 is a statistical model which is often used to benchmark other hurricane track forecasting methods. OFCL is the National Hurricane Center (NHC) official forecast. In general, the performance of these models varies per year. With increased computational power, the performance of OFCL is constantly improving. In addition, we define the baseline prediction as equal to four times the last displacement (from $t - 6h$ to t).

Quantitative

Qualitative

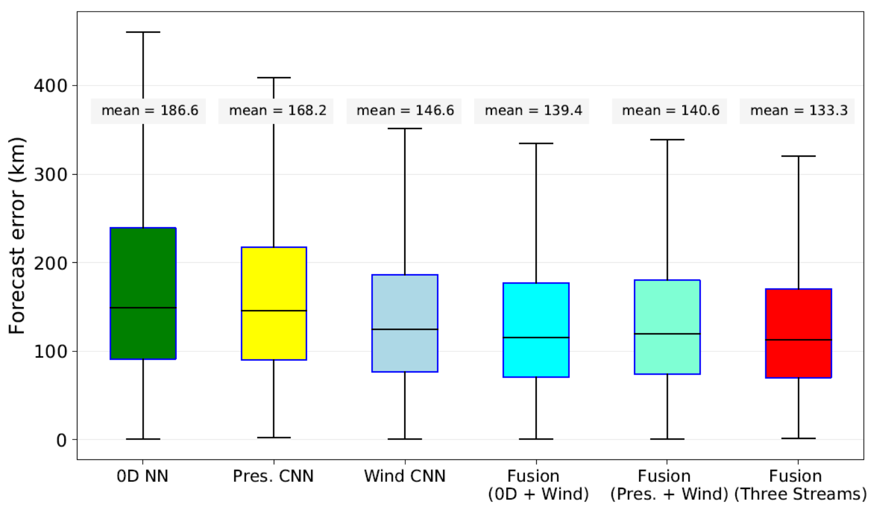

Figure 5.1: 24h-forecast results on the test set (storms coming from all oceanic basins), in distance between predicted and real location. Baseline = previous displacement (going straight).

Chapter 6

Conclusion and Discussion

We showed a promising deep learning framework for storm track forecasting. We demonstrated the benefit of coupling four types of data (wind fields, pressure fields, past tracks and other hand crafted data) in an efficient fusion model.

...

We think that the use of such deep learning methods can help the current forecast modellers by providing a complementary prediction that could be integrated in some consensus methods.

Acknowledgment

many thanks to ...

Bibliography

[1] At the heart of the main national research and higher education cluster. https://www.inria.fr/en/centre/saclay/overview. Accessed: 2018-07-04.

[2] Atmospheric reanalysis: Overview and comparison tables. https://climatedataguide.ucar.edu/climate-data/atmospheric-reanalysis-overview-comparison-tables. Accessed: 2018-07-04.

[3] History of tao. https://tao.lri.fr/tiki-index.php. Accessed: 2018-07-04.

[4] Hurricane katrina track. https://en.wikipedia.org/wiki/Meteorological_history_of_Hurricane_Katrina#/media/File:Katrina_2005_track.png. Accessed: 2018-07-31.

[5] Hurricanes: The greatest storms on earth. https://earthobservatory.nasa.gov/Features/Hurricanes. Accessed: 2018-07-31.

[6] key figures about the inria. https://www.inria.fr/en/institute/inria-in-brief/key-figures. Accessed: 2018-07-04.

[7] Nhc track and intensity models. https://www.nhc.noaa.gov/modelsummary.shtml. Accessed: 2018-07-04.

[8] Nyt sept 9, 2017. https://www.nytimes.com/2017/09/09/us/hurricane-irma-florida.html. Accessed: 2018-07-04.

[9] Portrait of balazs kegl. https://www.universite-paris-saclay.fr/en/recherche/portraits-de-chercheurs/balazs-kegl. Accessed: 2018-07-04.

[10] Tackling the underspecified. https://www.inria.fr/equipes/tau. Accessed: 2018-07-04.

[11] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[12] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[13] Manu de Bezenac, Arthur Pajot, and Patrick Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. *arXiv preprint arXiv:1711.07970*, 2017.

[14] Dick P Dee, S M Uppala, AJ Simmons, Paul Berrisford, P Poli, S Kobayashi, U Andrae, MA Balmaseda, G Balsamo, d P Bauer, et al. The era-interim reanalysis: Configuration and performance of the data assimilation system. *Quarterly Journal of the royal meteorological society*, 137(656):553–597, 2011.

[15] Mark DeMaria, Michelle Mainelli, Lynn K Shay, John A Knaff, and John Kaplan. Further improvements to the statistical hurricane intensity prediction scheme (ships). *Weather and Forecasting*, 20(4):531–543, 2005.

[16] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*, 2017.

[17] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[18] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[19] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[23] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[24] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[25] Kenneth R Knapp, Michael C Kruk, David H Levinson, Howard J Diamond, and Charles J Neumann. The international best track archive for climate stewardship (ibtracs) unifying tropical cyclone data. *Bulletin of the American Meteorological Society*, 91(3):363–376, 2010.

[26] Thomas R Knutson, Joseph J Sirutis, Gabriel A Vecchi, Stephen Garner, Ming Zhao, Hyeong-Seog Kim, Morris Bender, Robert E Tuleya, Isaac M Held, and Gabriele Villarini. Dynamical downscaling projections of twenty-first-century atlantic hurricane activity: Cmip3 and cmip5 model-based scenarios. *Journal of Climate*, 26(17):6591–6617, 2013.

[27] Alex KRIZHEVSKY. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[28] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[30] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[31] Sterenn Marie Liberge, Sileye Ba, Philippe Lenca, and Ronan Fablet. Prévision de trajectoires de cyclones à l'aide de forêts aléatoires avec arbres de régression. In *Conférence internationale francophone sur l'extraction et la gestion des connaissances*, pages 623–634. Hermann, 2011.

[32] Mina Moradi Kordmahalleh, Mohammad Gorji Sefidmazgi, and Abdollah Homaifar. A sparse recurrent neural network for trajectory prediction of atlantic hurricanes. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, pages 957–964. ACM, 2016.

[33] Pascal Peduzzi, Bruno Chatenoux, H Dao, Andréa De Bono, Christian Herold, James Kossin, Frédéric Mouton, and Ola Nordbeck. Global trends in tropical cyclone risk. *Nature climate change*, 2(4):289, 2012.

[34] Evan Racah, Christopher Beckham, Tegan Maharaj, Samira Ebrahimi Kahou, Mr Prabhat, and Chris Pal. Extremeweather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. In *Advances in Neural Information Processing Systems*, pages 3402–3413, 2017.

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[37] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[39] Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. In *Readings in speech recognition*, pages 393–404. Elsevier, 1990.

[40] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.