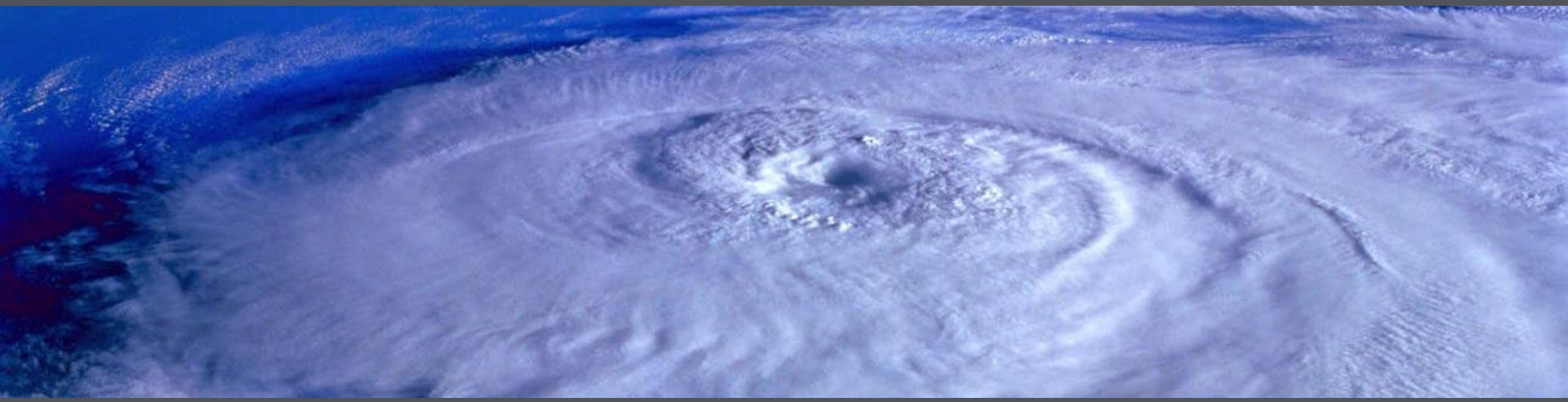
Department of Computer Science

University of Reading

USING DEEP LEARNING TO IDENTIFY ATMOSPHERIC FEATURES



Julian Kunkel[1], Bryan Lawrence[1], Daniel Galea[1] and Jeffrey Adie[2]

[1]Uni. Reading [2]NVIDIA AI Tech Center


NVIDIA
THE NVIDIA AI TECHNOLOGY CENTER

NVIDIA AI TECHNOLOGY CENTER (NVAITC)

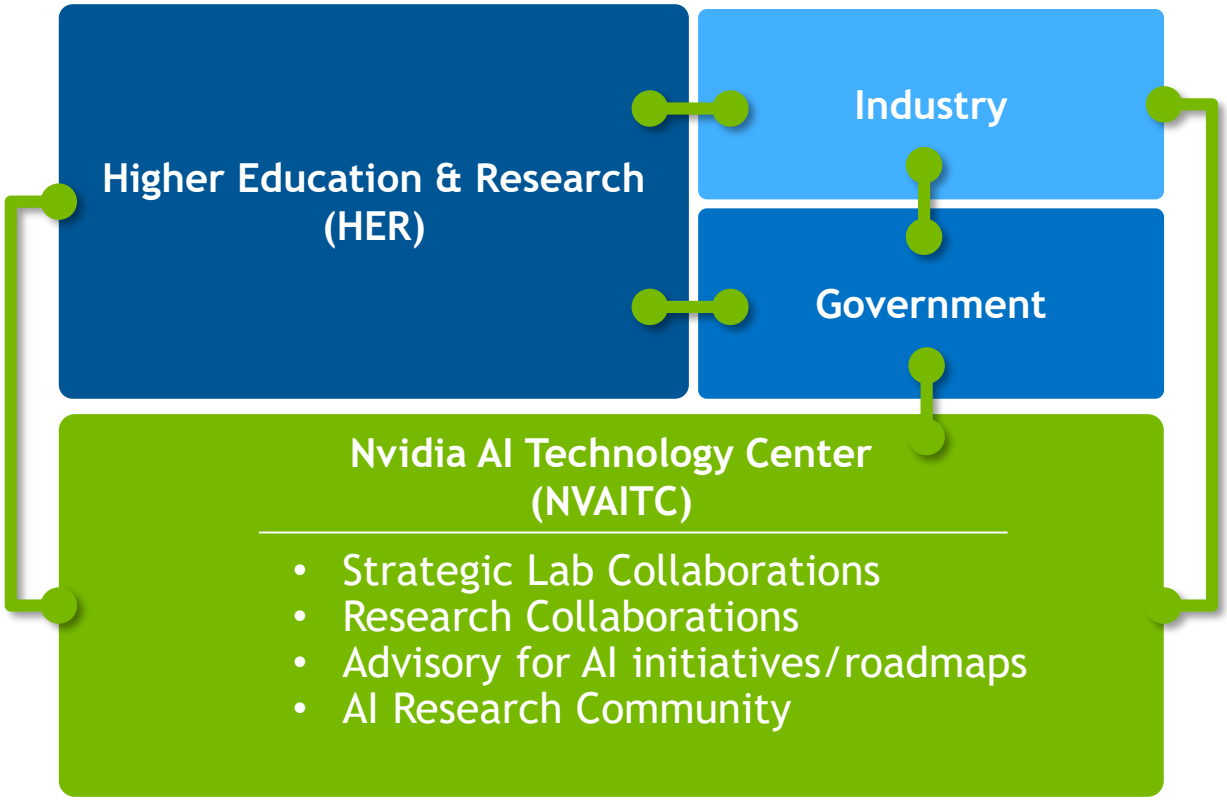
Catalyse AI transformation through research-centric integrated engagements

Established Aug 2015 in Singapore

Collaboration Footprint: Singapore. ASEAN. Taiwan. China. Hong Kong. Australia. Europe.

NVAITC STRATEGIC INVOLVEMENTS

Positioned alongside collaborators to catalyse and enable AI transformation



NVAITC COLLABORATORS

Singapore

- Nanyang Technological University Singapore
- Singapore University of Technology and Design
- SGInnovate
- Infocomm Media Development Authority
- National Research Foundation Prime Minister's Office Singapore
- Temasek Polytechnic
- NUS National University of Singapore
- EDB Singapore

Taiwan

- National Taiwan University
- Tsing Hua University
- 南科AI.ROBOT自造基地
- NCHC

China

- Chinese Academy of Sciences
- Xian Jiaotong University
- Peking University
- Fudan University
- Shanghai Jiao Tong University
- The University of Nottingham — United Kingdom · China · Malaysia

Hong Kong

- 香港科技大學 The Hong Kong University of Science and Technology
- HKSTP 香港科技園
- ASTRI Hong Kong Applied Science and Technology Research Institute Company Limited 香港應用科技研究院有限公司
- LSCM

Europe

- uni.lu Université du Luxembourg
- CSC
- epcc The University of Edinburgh
- University of Reading
- cini consorzio interuniversitario nazionale per l'informatica
- Hartree Centre Science & Technology Facilities Council

Thailand

- จุฬาลงกรณ์มหาวิทยาลัย

Australia

- Monash University

Indonesia

- epsindo PT. Epsindo Prima Sinergi

AI Acceleration Advisory Board

India

- IIT Hyderabad

ONGOING PROJECT INVOLVING THE NVAITC ATMOS

KiSTi 한국과학기술정보연구원 Korea Institute of Science and Technology Information

UNIST ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

연세대학교 YONSEI UNIVERSITY

NVIDIA

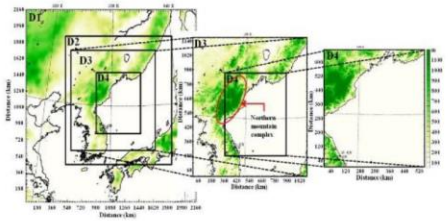
See their paper: DeepTC: ConvLSTM Network for Trajectory Prediction of Tropical Cyclone using Spatiotemporal Atmospheric Simulation Data, 2018

NVIDIA AI Tech Centre

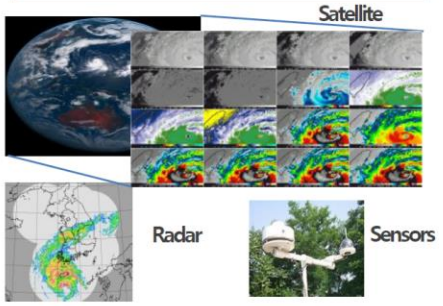
KISTI'S DL MODELS

Input Data → Output Data

Numerical Model Results

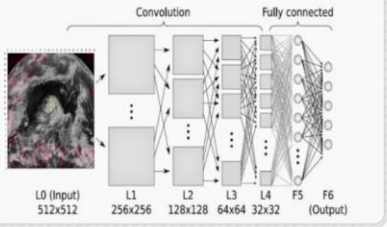


Image/Sensor Data

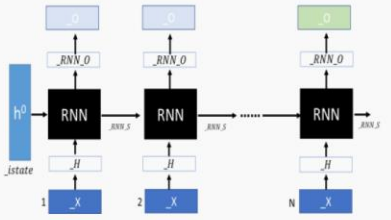


Prediction Model based on DL

Convolutional Neural Networks



Recurrent Neural Networks



Disaster Prediction

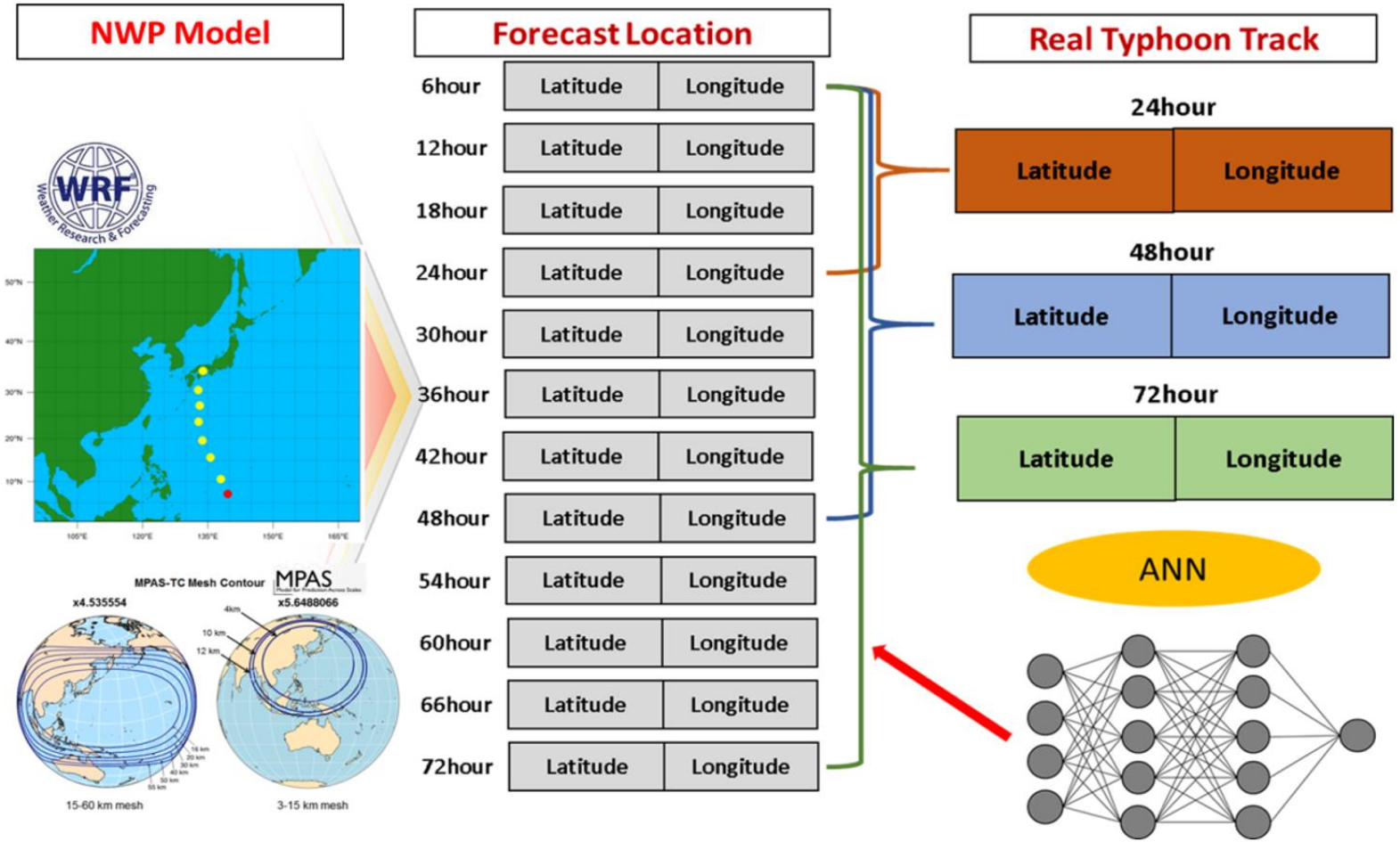
Typhoon/Precipitatio

Flood/Surge

DL Models

- Observation Data → DeepRain, GlobeNet
- Observation Data + Simulation Data → DeepTC

NWP + ML MODEL


NWP Model
WRF
Weather Research & Forecasting
MPAS-TC Mesh Contour
MPAS
x4.535554
x5.6488066
15-60 km mesh
3-15 km mesh
Forecast Location
6hour Latitude Longitude
12hour Latitude Longitude
18hour Latitude Longitude
24hour Latitude Longitude
30hour Latitude Longitude
36hour Latitude Longitude
42hour Latitude Longitude
48hour Latitude Longitude
54hour Latitude Longitude
60hour Latitude Longitude
66hour Latitude Longitude
72hour Latitude Longitude
Real Typhoon Track
24hour
Latitude Longitude
48hour
Latitude Longitude
72hour
Latitude Longitude
ANN

PROJECT: DEEP LEARNING ATMOSPHERIC FEATURES

University of Reading

- Collaboration between University of Reading and NVAITC
  - PhD student: Daniel GALEA d.galea@pgr.reading.ac.uk
- Supervisors
  - Primary: Bryan Lawrence
  - Secondary: Julian Kunkel
  - Supported by NVAITC (Jeffrey Adie)

MOTIVATION

University of Reading

- Numerical Weather Prediction (NWP) models and climate model runs
  - produce TB of data every day
- Manual inspection of data to make accurate forecasts or work on improving climate science is infeasible
- Storing large amount of data leads to ever-increasing costs

- Hence, a method to reduce the data would be desirable
  - Can we store data only when we observe interesting events?

AIMS

University of Reading

- Automatic detection of meteorological features in data using DL
  - Focus on tropical cyclones
- Integration into the existing workflow of a NWP or climate model
  - For in-situ data analysis
  - For triggering I/O upon event detection
- Constraints:
  - Detection of features in a distributed/parallel application
  - Domain decomposition must be taken into account

PROGRESS

University of Reading

1. Develop a first DL model to detect TCs
   - Learning about DL and machine learning
     - Testing parallel training ...
   - Verification of the accuracy: can TCs be detected correctly?
   - Exploring hyperparameters
     - Benefits/drawbacks of different network architectures
     - Analysis isn't complete, but many aspects tested
   - Development of appropriate training/validation sets

2. *Ongoing activity:* Integration into the model

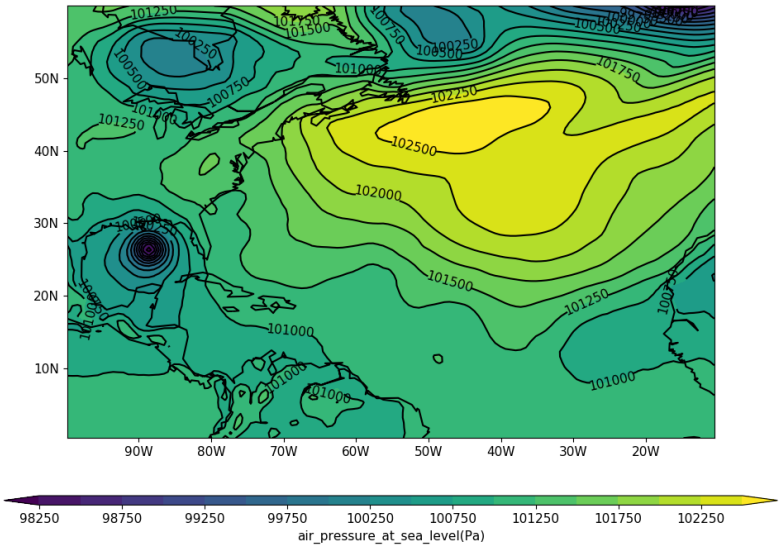
Let's have a look at the best model we optained so far

ACCURACY - DATA

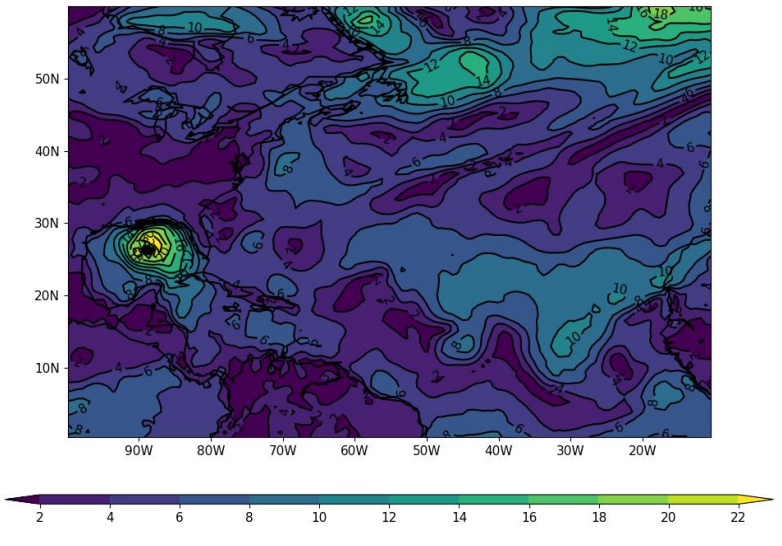
University of Reading

- ECMWF's ERA-Interim data set from 1st January 1979 to the 30th June 2017
- Resolution of ~75km
- Five fields were used to train the Deep Learning model
  - Mean Sea Level Pressure (MSLP)
  - 10m Wind Speed
  - Vorticity at 850hPa
  - Vorticity at 700hPa
  - Vorticity at 600hPa

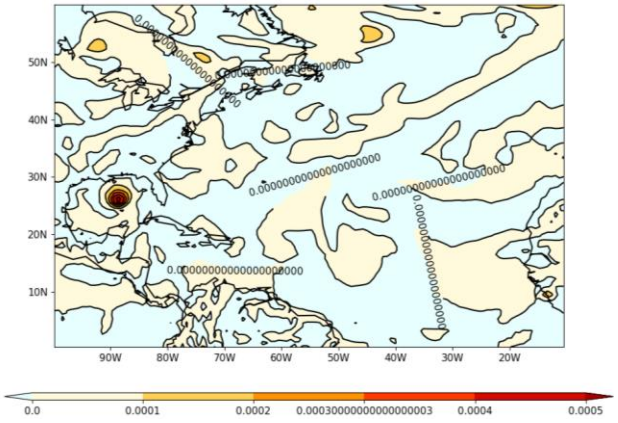
Data for Hurricane Katrina at 25/08/2005 at 18Z



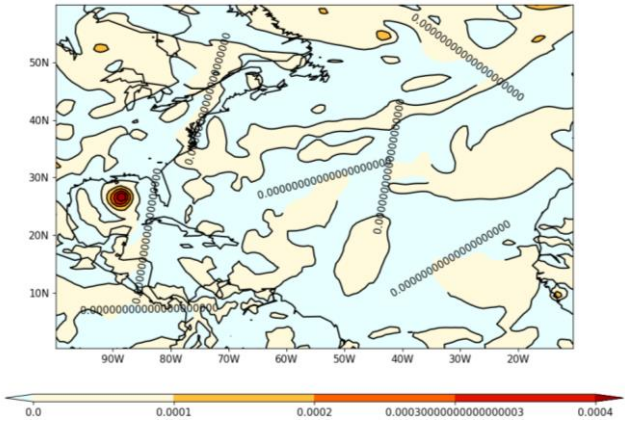
MSLP



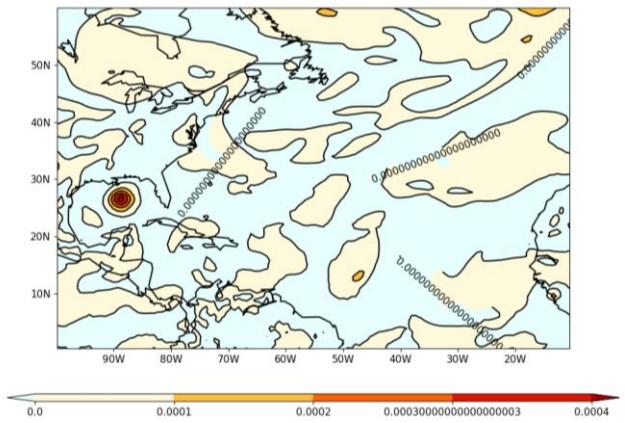
10m Wind Speed



Vorticity at 850hPa



Vorticity at 700hPa

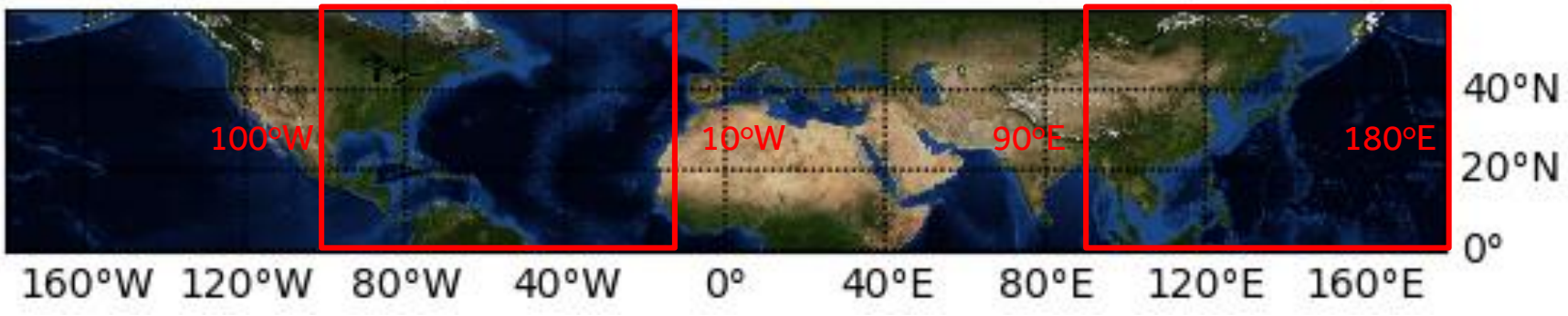


Vorticity at 600hPa

PROGRESS - DATA

University of Reading

- Data cropped to cover Western Atlantic and Western Pacific basins



- The final dataset had 51931 cases
  - ~80% having no TC present
  - ~20% having TCs present
- Data was normalized by:
  - First applying a log scale to the wind speed
  - Min/Max normalization to obtain values in the range of [0, 1]
- Resolution: 225km (1/3rd of original data)
- Data split into
  - training set (60%)
  - validation set (20%)
  - test set (20%)

DEEP LEARNING MODEL

University of Reading

- DL model containing convolutional layers and fully connected layers
- Manual hyperparamter tuning was performed:
  - Method of weight initialization (Glorot)
  - Number of convolutional layers and fully connected layers
  - Number of nodes in the layers
  - Normalization method (L2 normalization penalty factor 0.0001)
  - Activation functions (Leaky ReLU)
  - Batch size (2000)
  - Methods to balance the dataset

- The final model obtained 94% accuracy for detecting Tropical Cyclones in the Western Atlantic and Western Pacific basins

DEEP LEANING MODEL

- Convolution layer 10x10 window, 8 kernels, 1 stride
  - Dropout probability 0.3
- MaxPool 2x2 window, 1 stride
- Convolution layer 5x5 window, 16 kernels, 1 stride
  - Dropout probability 0.3
- MaxPool 2x2 window, 1 stride
- Convolution layer 10x10 window, 32 kernels, 1 stride
- MaxPool 2x2 window, 1 stride
- Flatten
- Dense (Fully-Connected) layer 128 nodes
  - Dropout probability 0.3
- Dense (Fully-Connected) layer 64 nodes
  - Dropout probability 0.3
- Dense (Fully-Connected) layer 1 node, Sigmoid activation

Resolution in km

- *2250 x 2250*
- 450 x 450
- *4500 x 4500*
- 900 x 900
- *9000 x 9000*
- 1800 x 1800

FORTRAN IMPLEMENTATION

University of Reading

- Integration of DL model into NWP/climate workflow
  - Fortran code base!
- Approach
  - Training in Python
  - Loading weigths in C++ using the frugally-deep package
  - Fortran/C++ interface
- Reproduction of the validation results loading the same data from Fortran

Frugally-Deep: https://github.com/Dobiasd/frugally-deep

FORTRAN IMPLEMENTATION DETAILS

University of Reading

- Save Deep Learning model from Python
  - model.save('keras_model.h5', include_optimizer=False)
- Use python script included in frugally-deep to save the model in an appropriate format:
  - python3 convert_model.py keras_model.h5 fdeep_model.json
- Load model in C++ using frugally-deep package:
  - const auto model = fdeep::load_model("fdeep_model.json");
- Load data from FORTRAN
  - pass it to function in C++
  - make inference
  - pass inference result back to FORTRAN

NEXT STEPS

University of Reading

- Expand the DL model to be able to handle data from across the globe
- Improve the Deep Learning model performance
  - How does the resolution impact on the model's performance?
  - Does more training data help improve final testing accuracy?
  - Can a better method for balancing the dataset be found?
  - Can better hyperparameters be found?
  - How do more/less fields impact on the final testing accuracy?
- Deal with the domain decomposition of the parallel model
  - Convolutional layer seems a good candidate
  - Need some data from neighbours?
- Finish the DL model in the workflow of an NWP or climate model