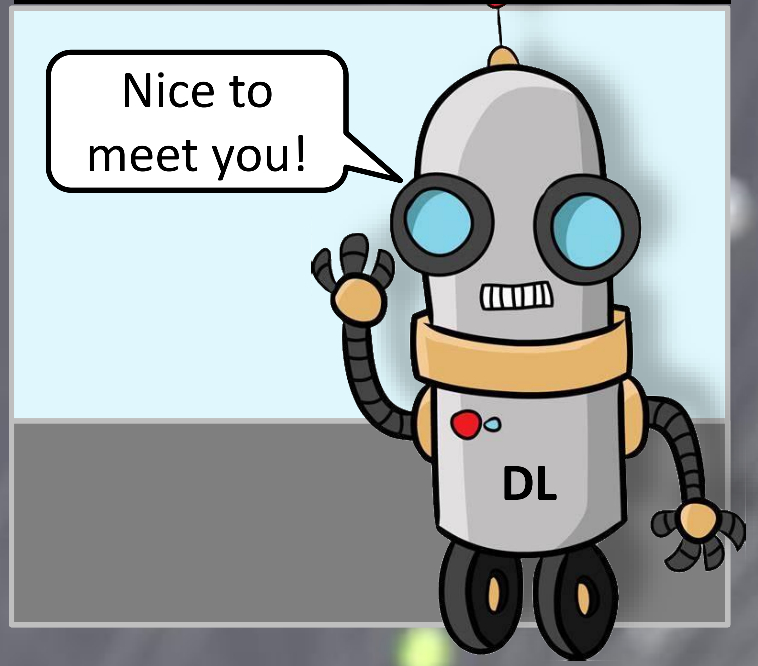
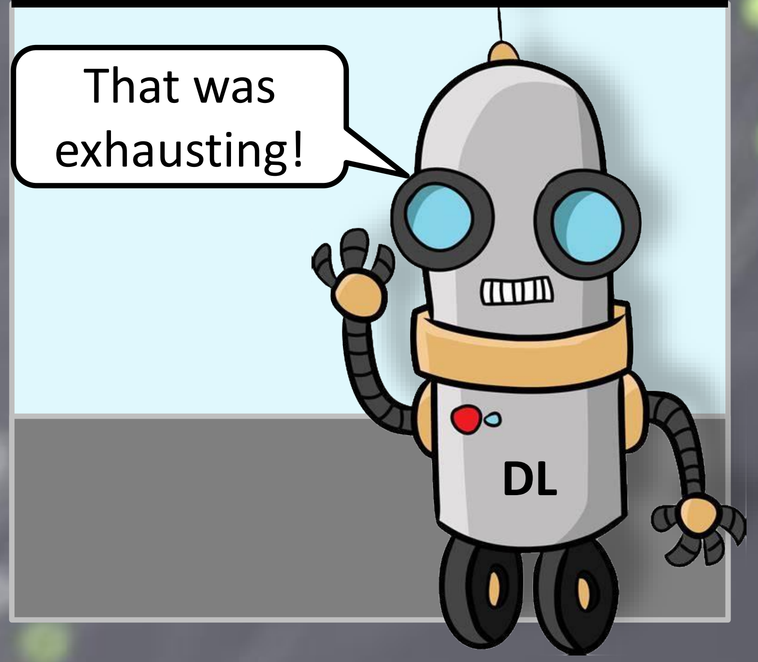
WHAT WE HAVE DONE YESTERDAY

12:00-12:45 ET
INTRO TO DL, PART 1

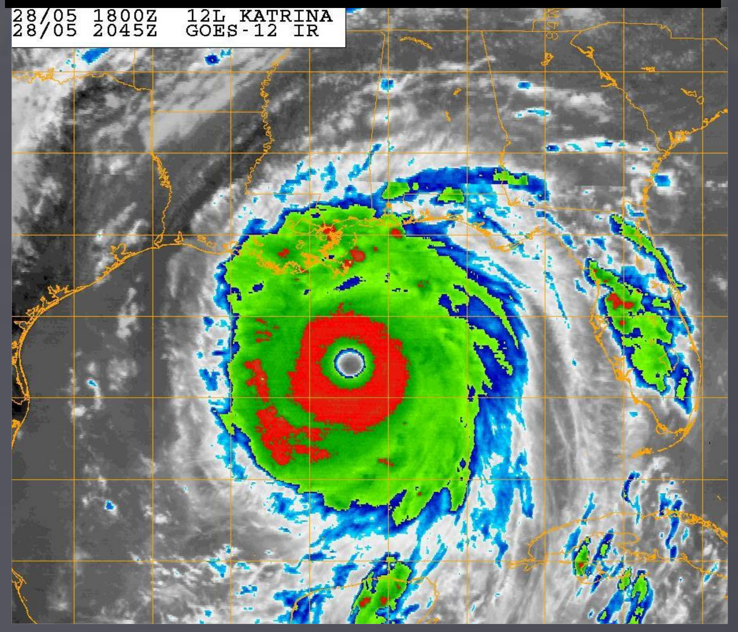


12:55-1:25 ET
INTRO TO DL, PART 2

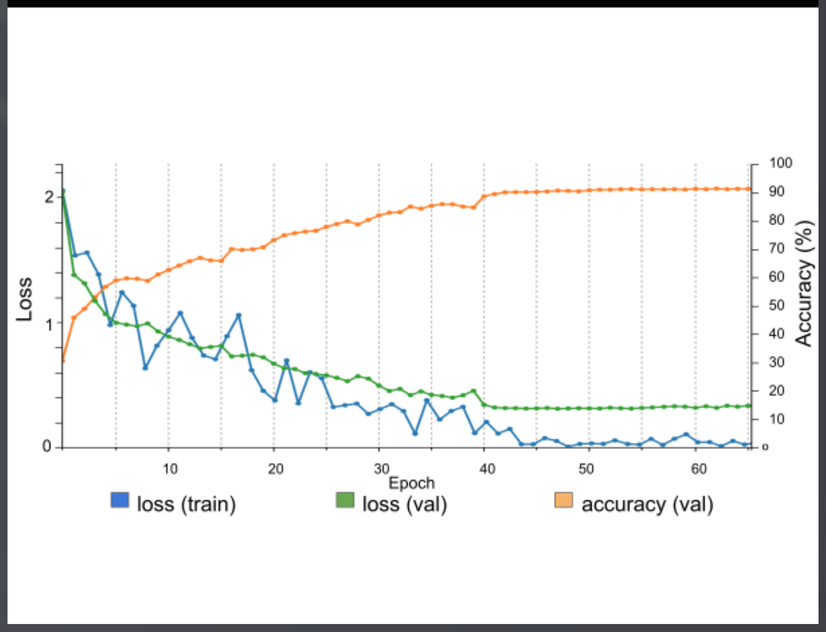


LAB 1: CNNs AND KERAS 101

Tomorrow
LAB 2: TROPICAL CYCLONES



Tomorrow
LAB 3: TC CHALLENGE



LAB PART 1
CNN AND KERAS 101

MNIST

The standard ‘hello world’ problem for deep learning

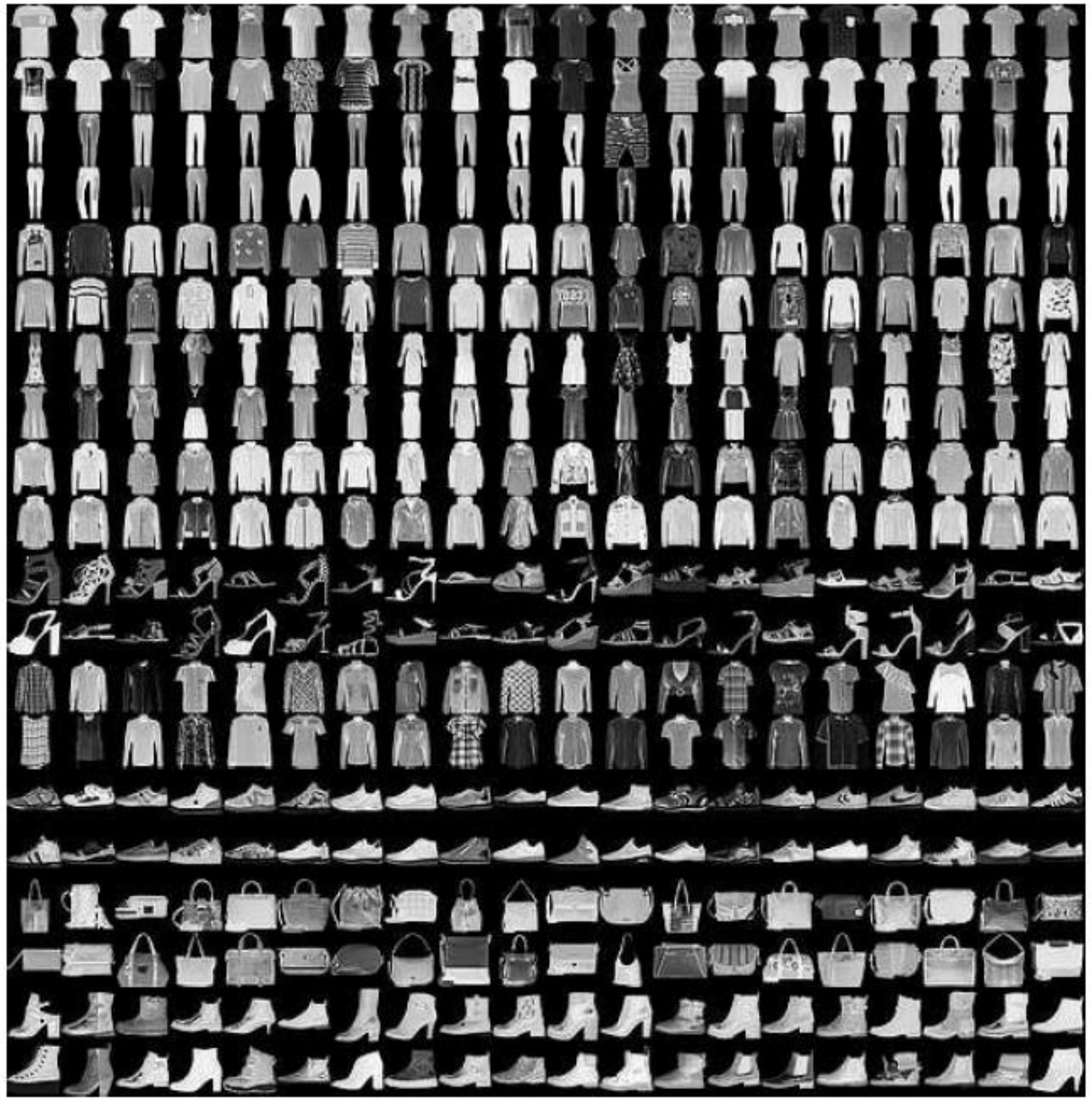
MNIST
Keras implementation

```python
from tensorflow import keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Dropout,Flatten,Conv2D,MaxPooling2D
from tensorflow.keras import backend as K

num_classes = 10
img_rows, img_cols = 28,28

# DATA
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.reshape(x_train.shape[0], img_rows, img_cols, 1)
x_test  = x_test.reshape (x_test.shape[0] , img_rows, img_cols, 1)
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test  = keras.utils.to_categorical(y_test,  num_classes)

# MODEL
input_shape = (img_rows, img_cols, 1)
model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
model.compile(loss      = keras.losses.categorical_crossentropy,
              optimizer= keras.optimizers.Adadelta(),
              metrics   = ['accuracy'])
# TRAIN
model.fit(x_train, y_train,batch_size=128,epochs=12,
          verbose=1, validation_data=(x_test, y_test))

# TEST
score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:',     score[0])
print('Test accuracy:', score[1])
```

FASHION MNIST

A slightly more interesting version of MNIST

Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	



6 STEPS APPROACH

Data → Task → Model

↓

Evaluation ← Learning ← Loss

LEVELS OF AI ENGAGEMENT

LEVEL 1	LEVEL 2	LEVEL 3	LEVEL 4	LEVEL 5
AI in a supporting role, but decoupled from main production system	AI and main production system influence each other but are largely stand-alone	AI takes over parts of the main production system	AI replaces significant parts of the main system, classical parts play supporting role	The system is designed with AI in mind from the start; classical algorithms generate training data

Data Analytics

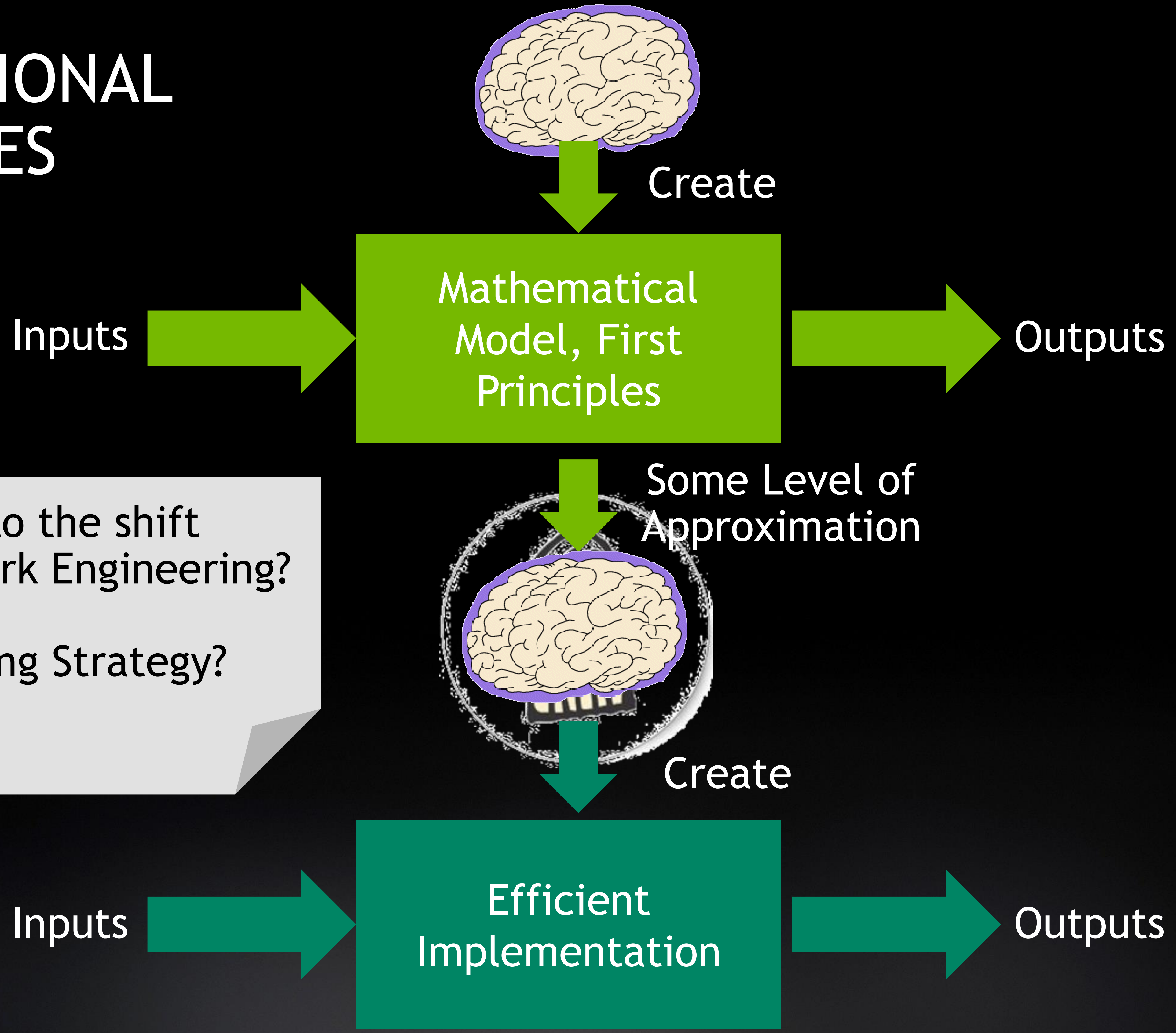
Numerical Simulation

Signal Processing

Visualization

…

COMPUTATIONAL SCIENCES



Similarities to the shift
Feature → Network Engineering?

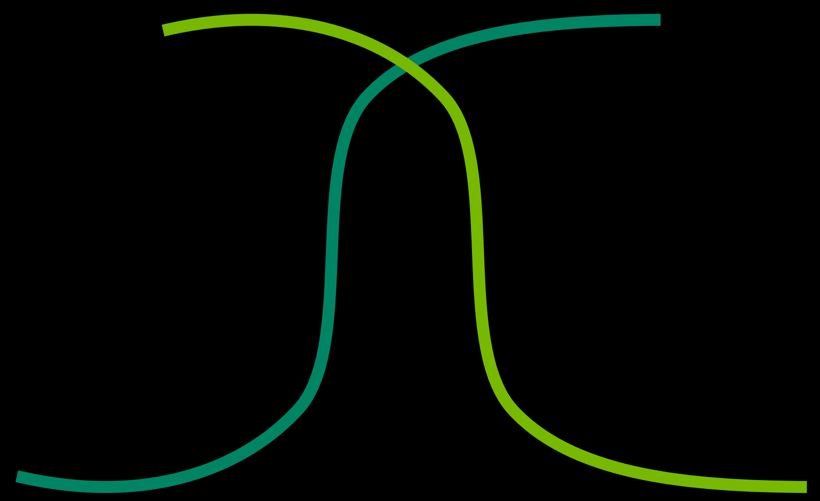
NNs as a Porting Strategy?

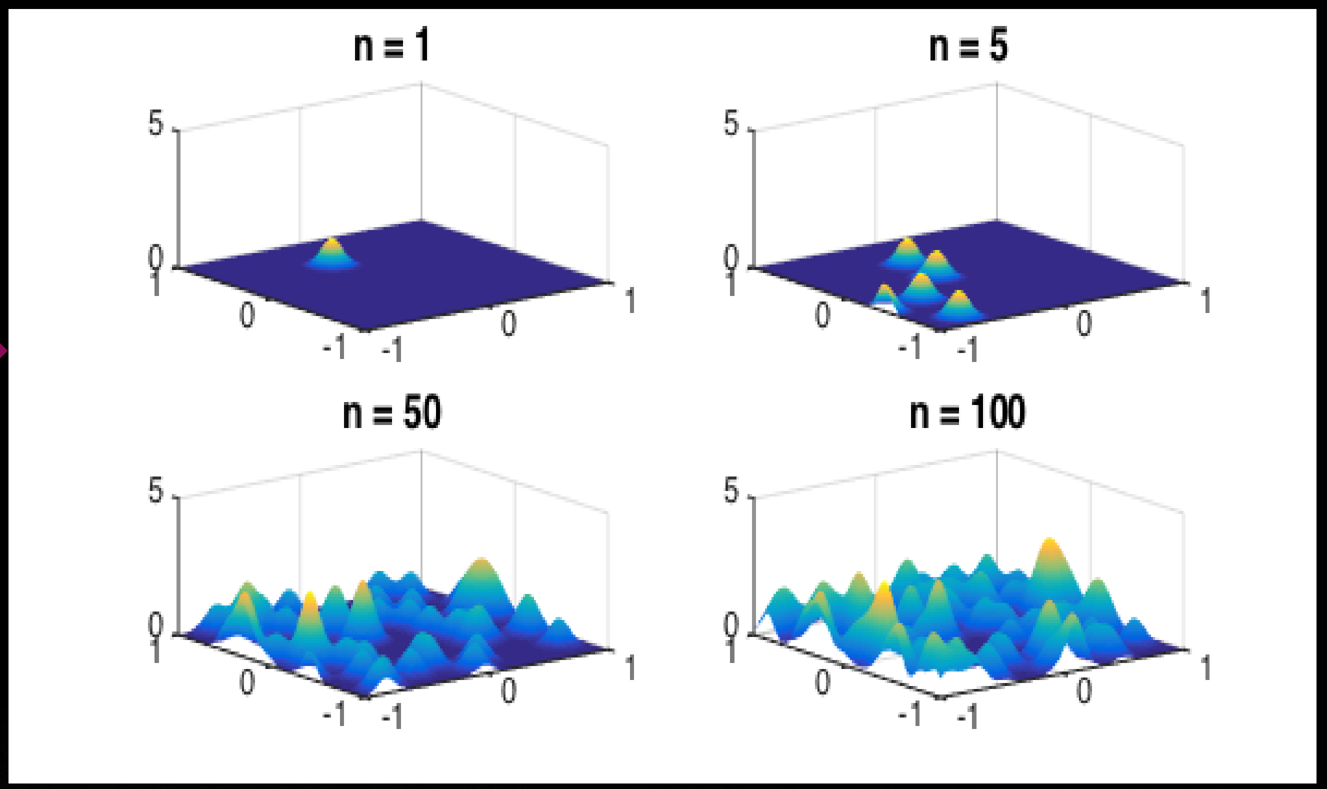
CAN THIS WORK ∀? ABOLUTELY, YES!

Proof: Universal Approximation Theorem

$\alpha *$

$\beta *$





Take many non-linearities

Combine to form peaks
(one hidden layer is enough!)

And assemble your arbitrary function with arbitrary ε

Problem: this is an essentially useless theorem for practical purposes

WILL THIS WORK ∀?

Considering pesky practical constraints, like memory and performance

- Anecdotal Evidence: ∃ scientific cases where NNs seem to do work extremely well
- Save bet: it will not work for ∀
- Therefore, by induction (sort of):
  - There exists ∃ a subspace in ∀ HPC applications, for which AI works well
  - Need to explore the **size** and **shape** of this subspace
  - Currently I think it is fair to say we don’t understand this domain very well
  - **But:** Each individual case promising 10x, 100x,1000x performance improvement is probably worth exploring; those can be groundbreaking!

But Intuition is Misleading

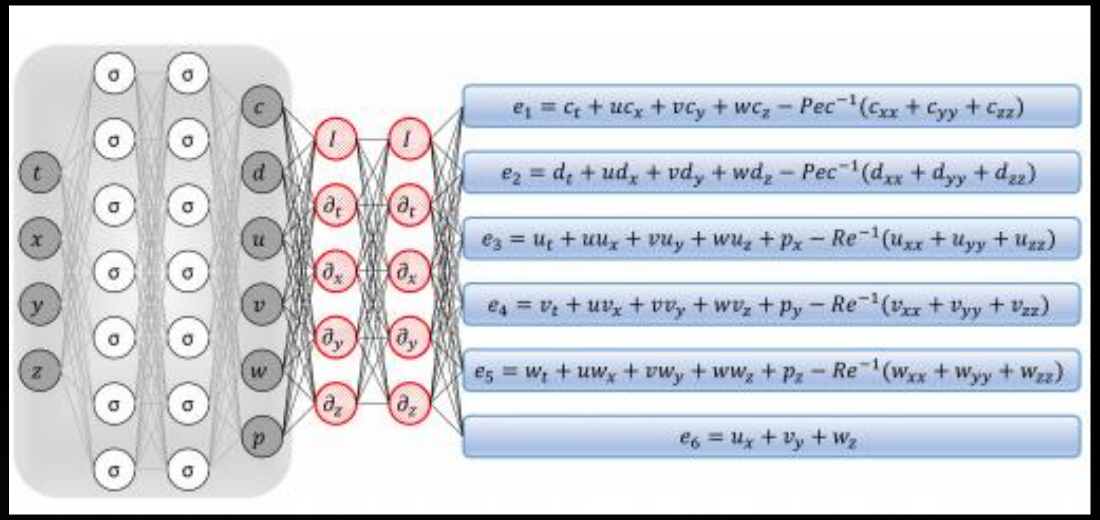
HOW TO FILL IN THE 

Experience,
Intuition, and Art

+ Tools Support

E.g. Adversarial Fuzzing

E.g. Declarative Building
Blocks to NN Translation



E.g. Physics Informed
Networks?[1],
ODE Networks?[2]

1) Hidden Fluid Mechanics: A Navier-Stokes Informed Deep Learning Framework, M. Raissi et al.
2) Neural Ordinary Differential Equations, R.T.Q. Chen et al.

SCIENTIFIC CHALLENGES

Barriers to acceptance of deep learning as a tool for science

- **Interpretability:** Can I understand what the neural-net is doing?
- **Robustness:** Will it always give me the right answer?
- **Coverage:** How much training data do I need?
- **Convergence:** How can I ensure that training will converge?
- **Uncertainty:** How certain can I be of the answers?

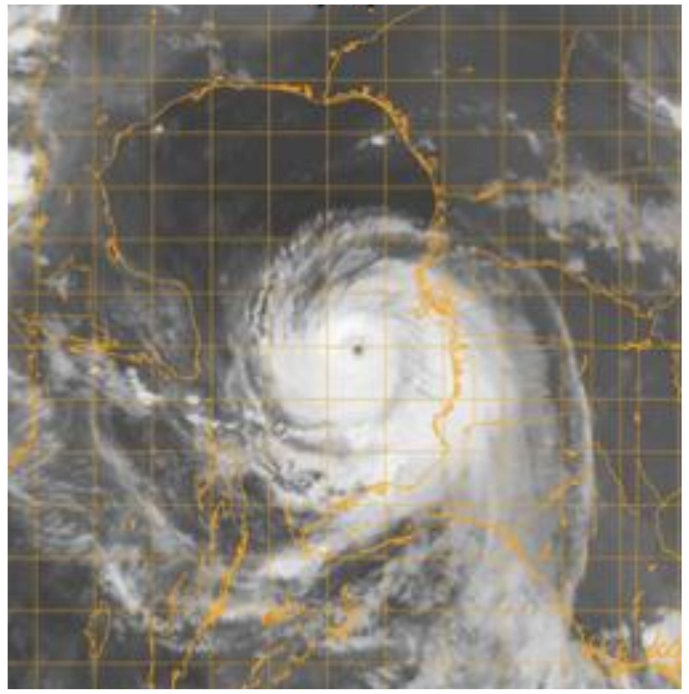
ESTIMATING TROPICAL CYCLONE INTENSITY

Paper Overview

Tropical Cyclone Intensity Estimation Using a Deep Convolutional Neural Network

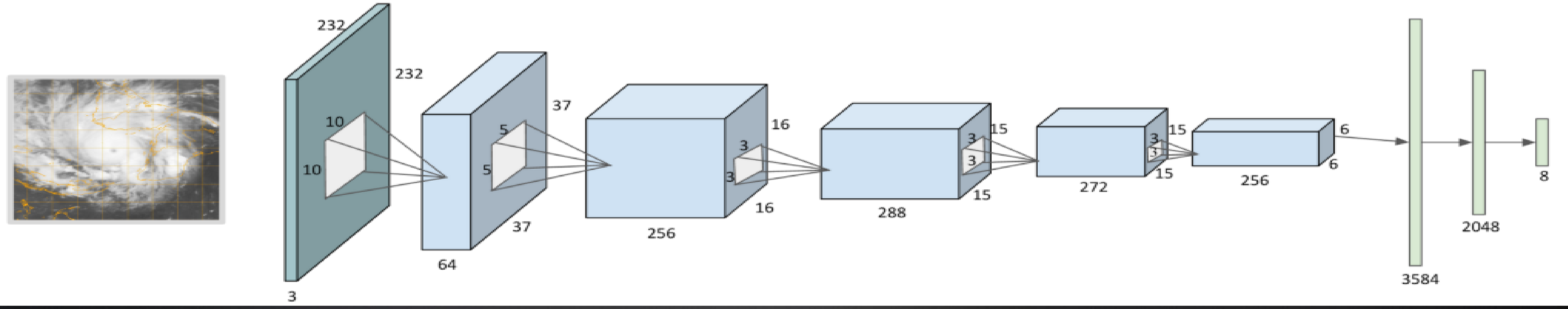
Ritesh Pradhan, Ramazan Aygun, *Senior Member, IEEE,* Manil Maskey, *Member, IEEE,* Rahul Ramachandran, *Senior Member, IEEE,* and Daniel Cecil

INPUT: 232 x 232 pixels



OUTPUT: 8 CLASSES

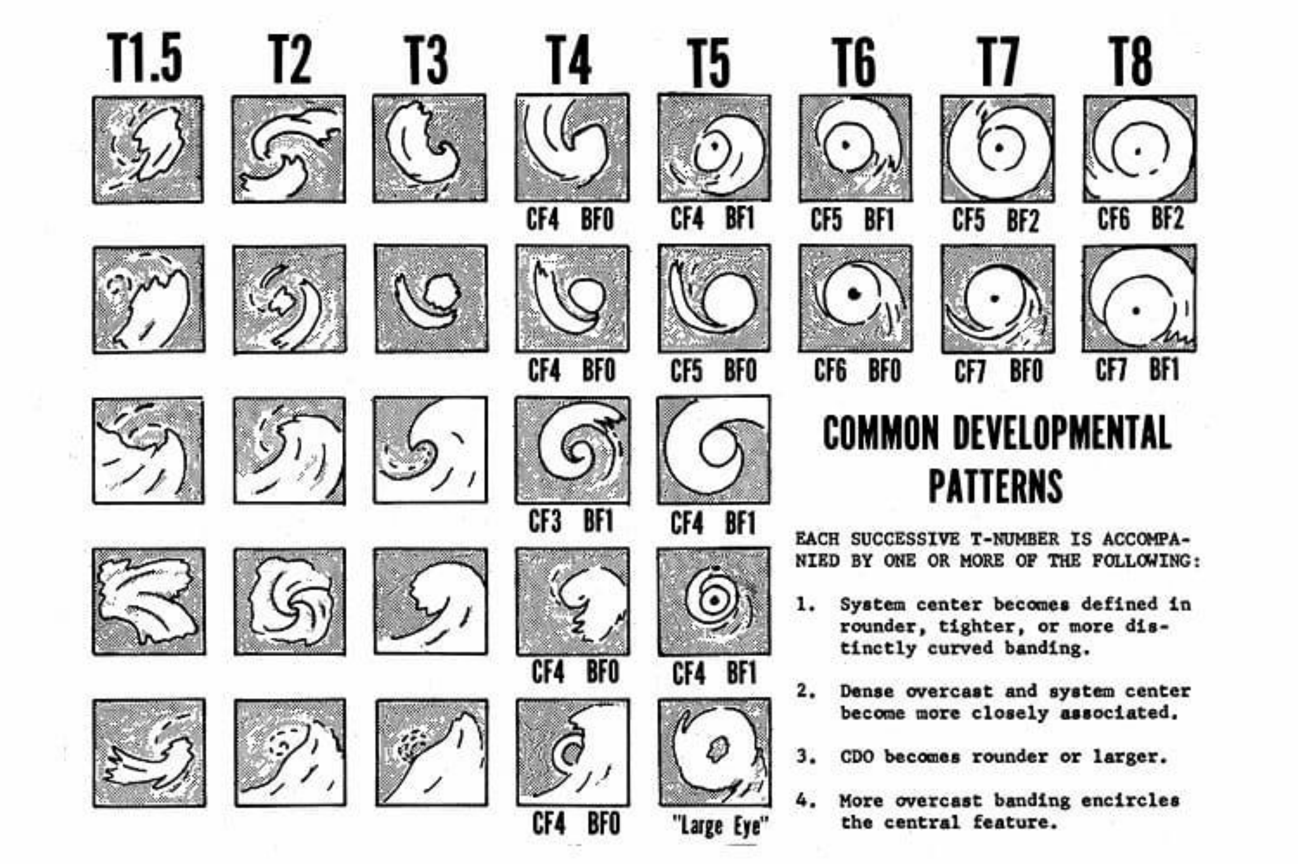
Category	Symbol	Wind speeds	Damage
Five	H5	≥ 137 knots	Catastrophic
Four	H4	113-136 knots	Catastrophic
Three	H3	96-112 knots	Devastating
Two	H2	83-95 knots	Extensive
One	H1	64-82 knots	Significant
Tropical storm	TS	34-63 knots	Significant
Tropical depression	TD	20-33 knots	Small
No Category	NC	≤ 20 knots	-



ESTIMATING TROPICAL CYCLONE INTENSITY

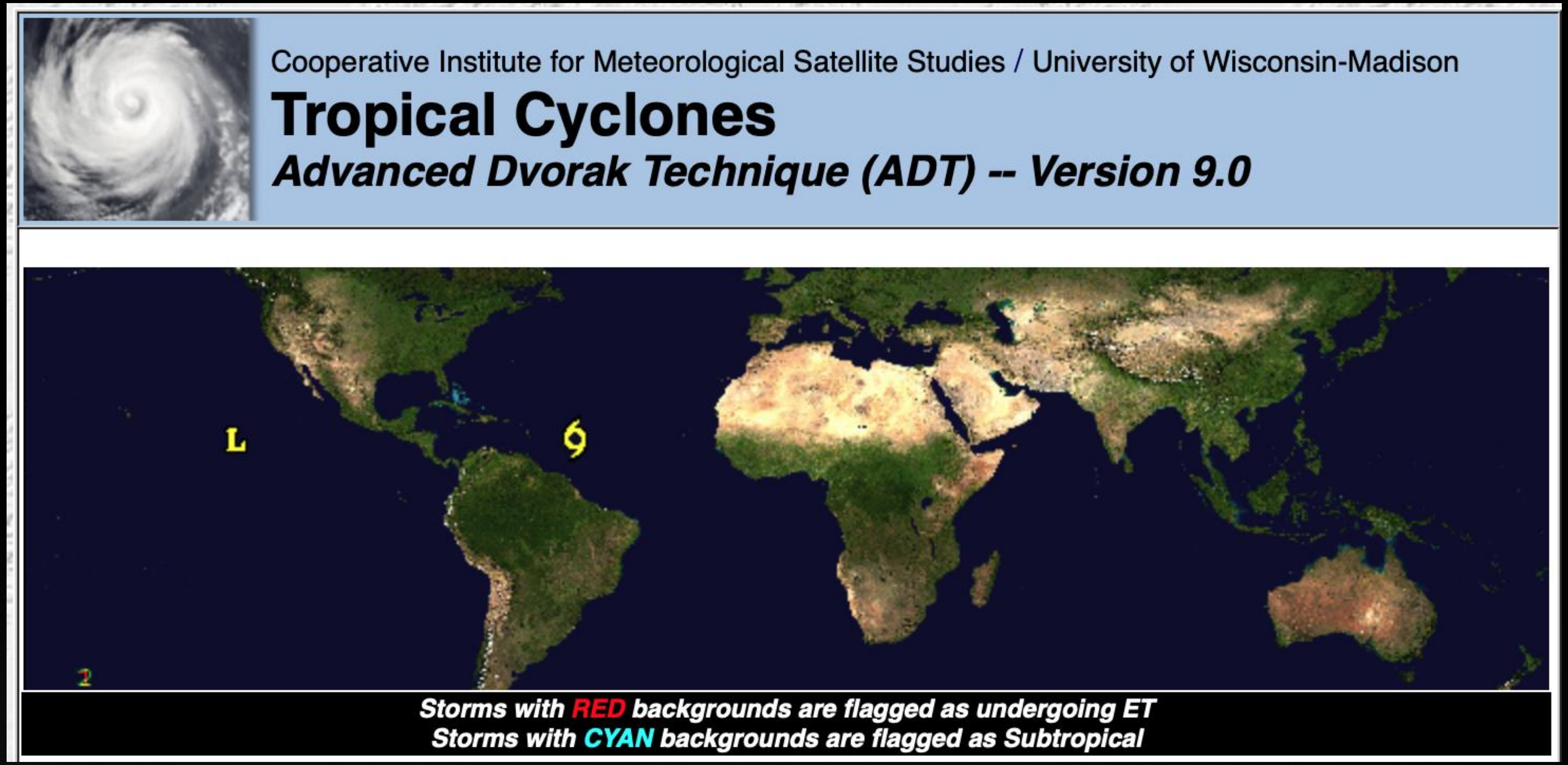
Background: Dvorak technique

Dvorak Technique (1974)



https://doi.org/10.1175/1520-0493(1975)103%3C0420:TCIAAF%3E2.0.CO;2

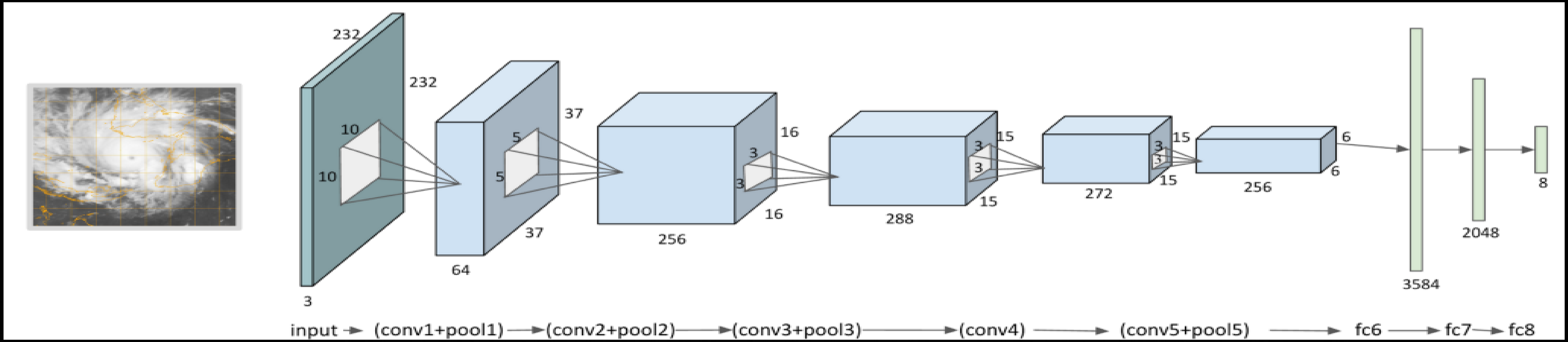
Advanced Dvorak Technique- version 9 (2019)



https://doi.org/10.1175/WAF-D-19-0007.1

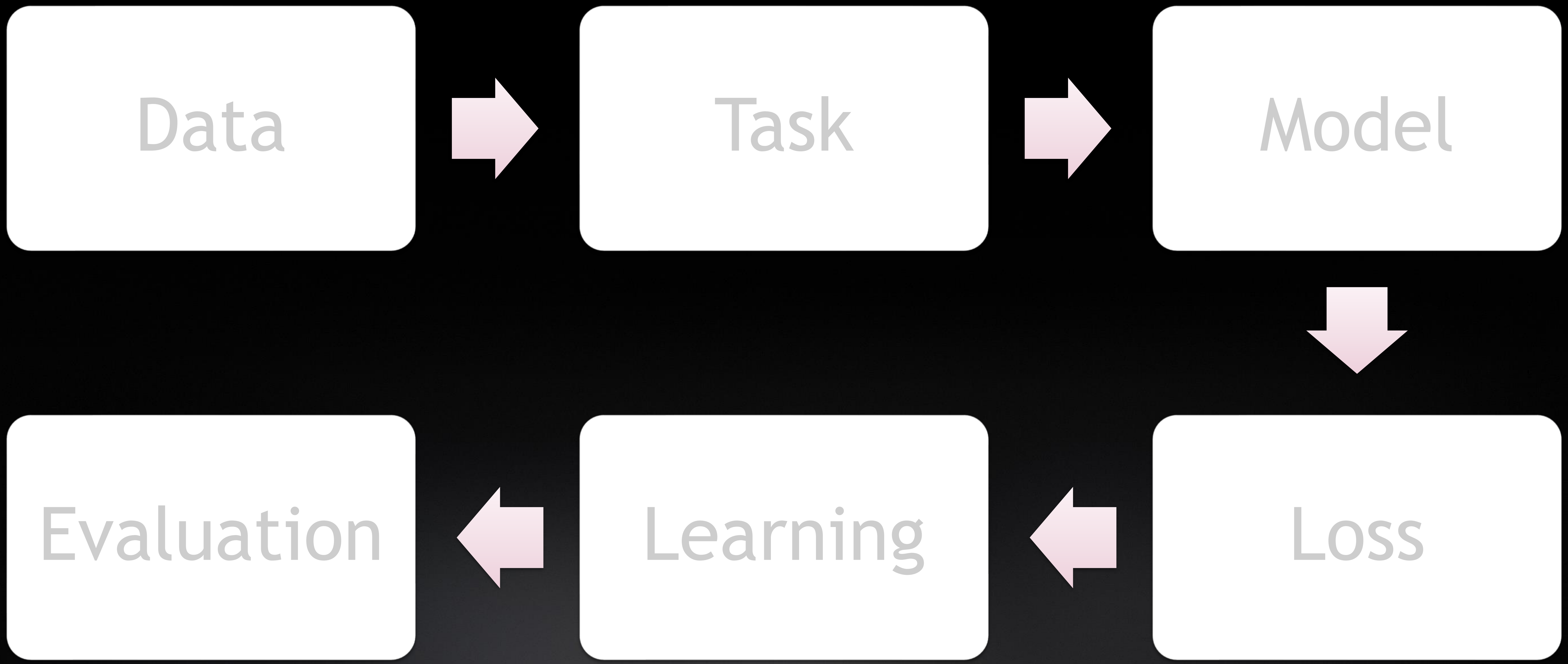
ESTIMATING TROPICAL CYCLONE INTENSITY

CNN Model

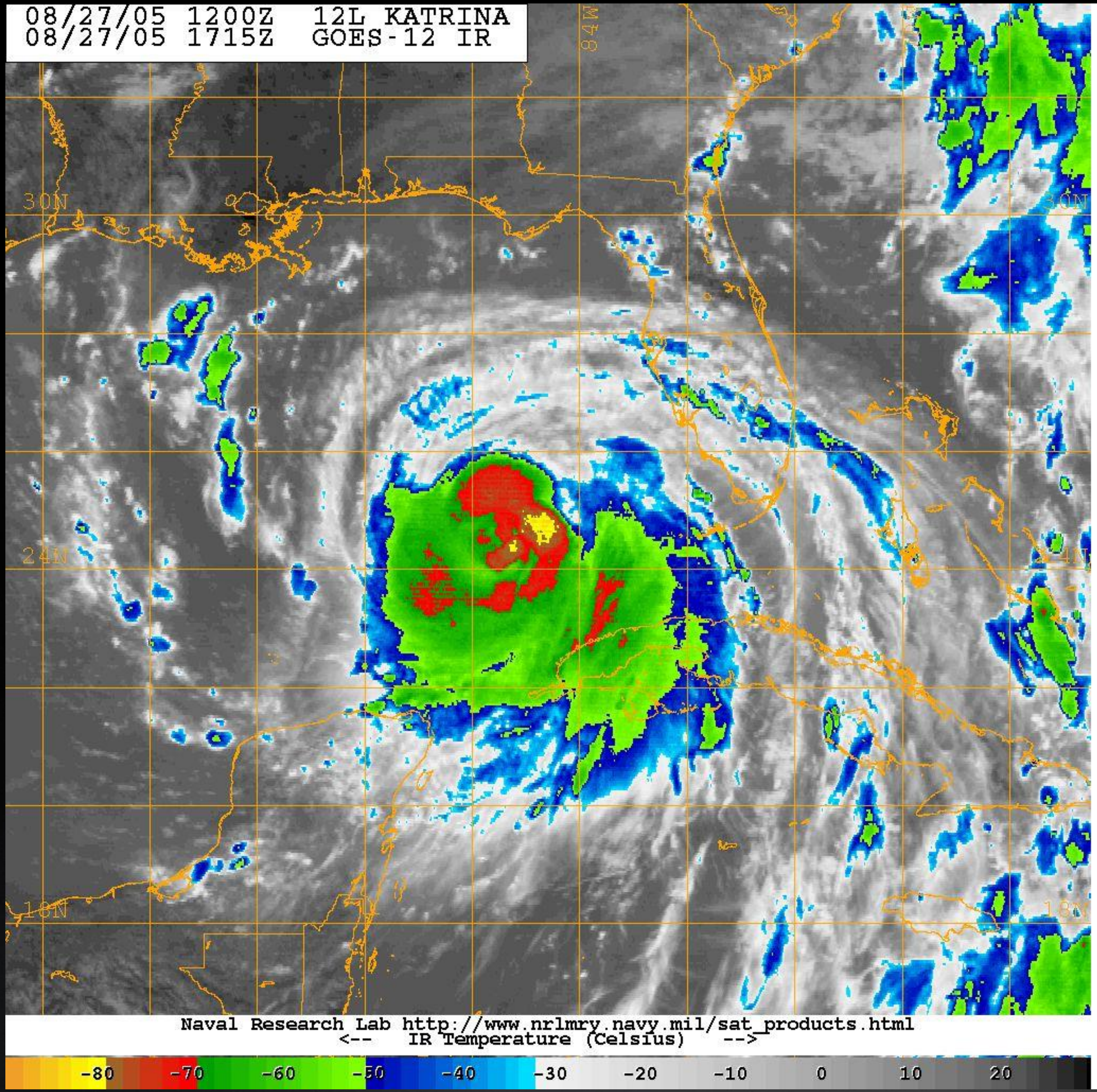


6 STEPS APPROACH

Steps to follow while solving a Machine Learning problem



DATA



SAFFIR-SIMPSON HURRICANE WIND SCALE
AND RELATED CLASSIFICATIONS

Category	Symbol	Wind speeds	Damage
Five	H5	$\geq$ 137 knots	Catastrophic
Four	H4	113–136 knots	Catastrophic
Three	H3	96–112 knots	Devastating
Two	H2	83–95 knots	Extensive
One	H1	64–82 knots	Significant
Tropical storm	TS	34–63 knots	Significant
Tropical depression	TD	20–33 knots	Small
No Category	NC	$\leq$ 20 knots	-

TASK

Multi-class Classification.

NC (No Category , ≤ 20 knots)

TD (Tropical Depression , $20-33$ knots)

TS (Topical Storm , $34-63$ knots)

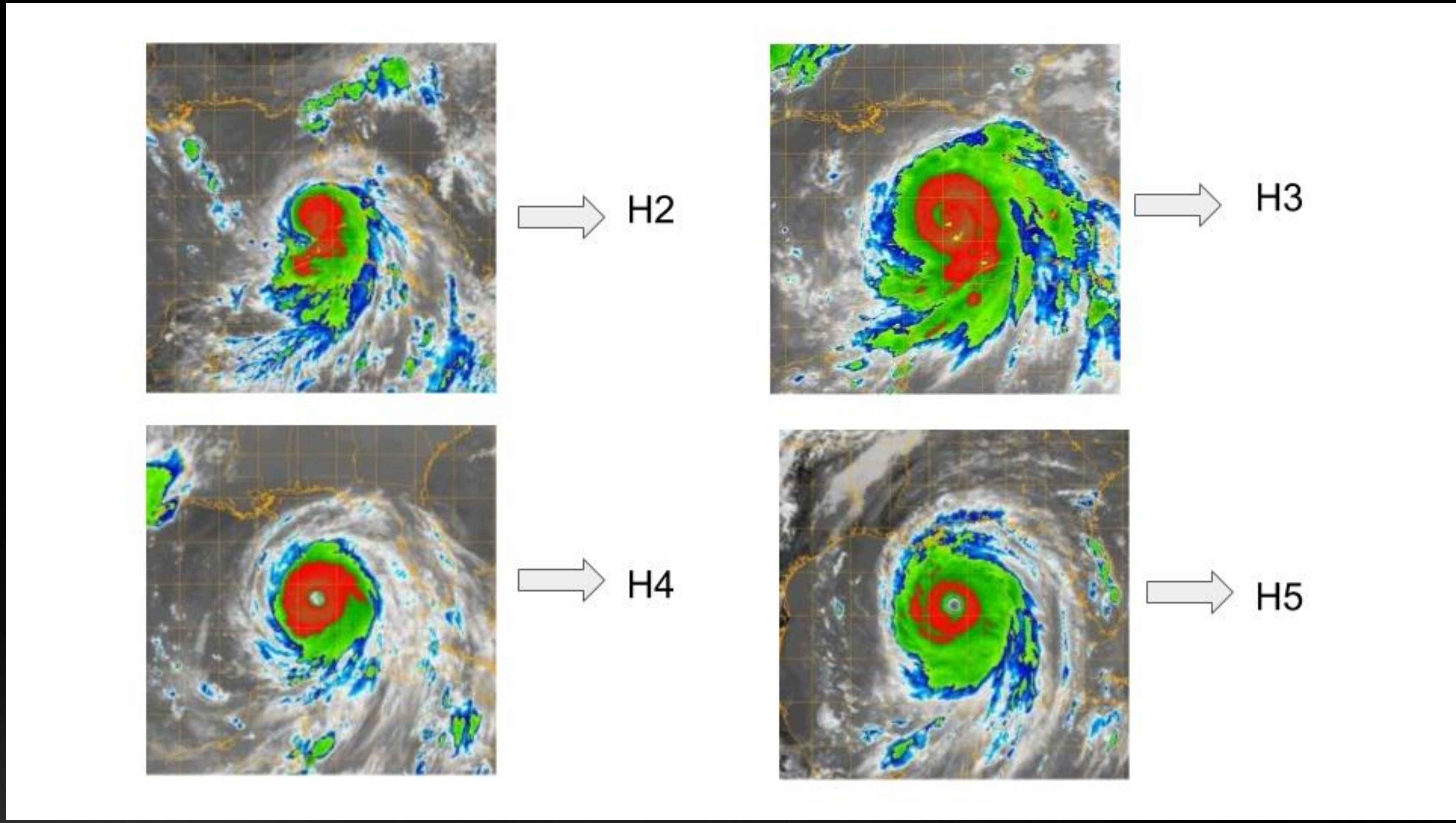
H1 (Category One , $64-82$ knots)

H2 (Category Two , $83-95$ knots)

H3 (Category Three , $96-112$ knots)

H4 (Category Four , $113-136$ knots)

H5 (Category Five , ≥ 137 knots)



MODEL 1

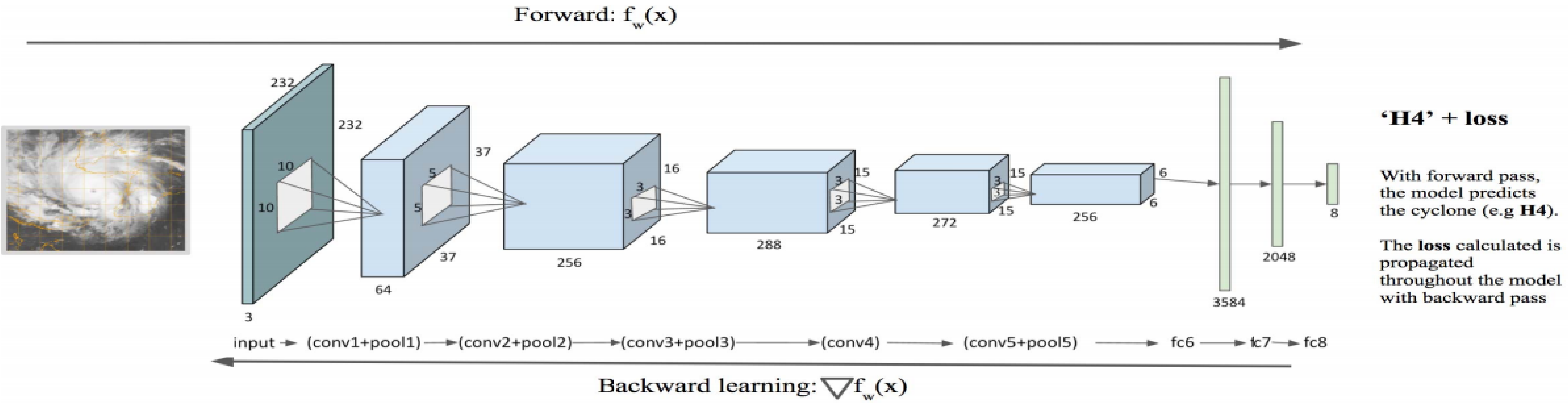


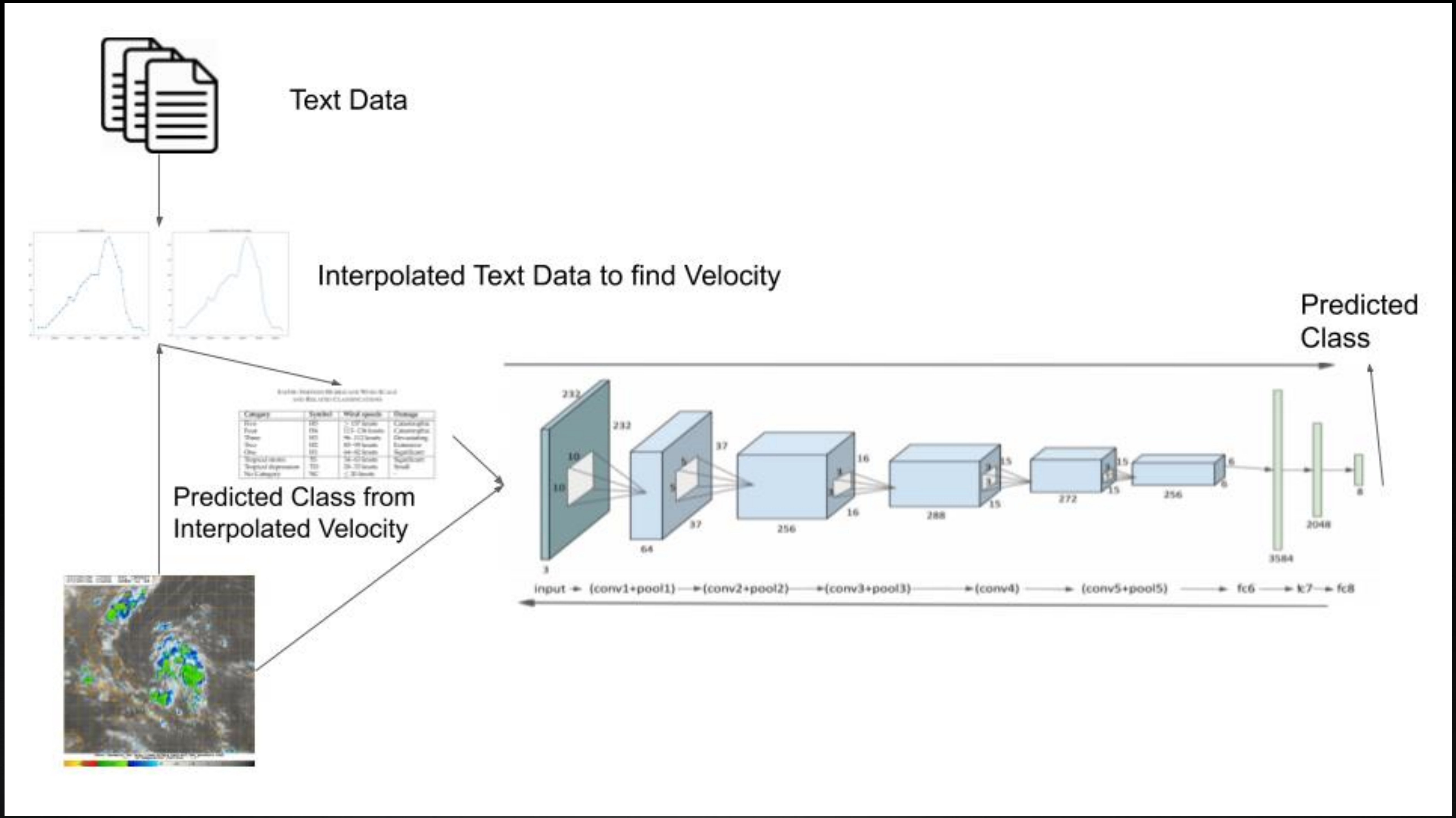
Fig. 2. Network architecture for hurricane intensity estimation showing different steps of convolution and pooling.

Loss Function: Multi-class Cross-Entropy loss functions

Optimizer SGD (Stochastic Gradient Descent)

Training and Evaluation: Training Set 72 % , Test Set, 8 %, Validation Set 10%

SUMMARY OF APPROACH



PREPROCESSING DATA

Pre-Processing Data:

Step 1 : Resize Image from (1024, 1024 ,3) to (256 , 256 ,3)

Step 2 : Choose a random (232 , 232 , 3) patch from the (256 , 256 , 3) and feed into our model.

There are different types of Resizing:

- cv2.INTER_AREA (Preferable for Shrinking)
- cv2.INTER_CUBIC (Preferable for Zooming but slow)
- cv2.INTER_LINEAR (Preferable for Zooming and the default option)

localhost:9090/pun/sys/dashboard/batch_connect/sys/ai-science-climate/session_contexts/new

NVIDIA DeepOps Cluster | Files | Jobs | Clusters | Interactive Apps | My Interactive Sessions | Help | Logged in as gunterr | Log Out

Home / My Interactive Sessions / AI For Science Climate Lab

Interactive Apps

Desktops

- NVIDIA DeepOps Desktop

Servers

- AI For Science CFD Lab
- AI For Science Climate Lab
- VS Code Server

AI For Science Climate Lab

This app will launch a Jupyter Notebook server on one or more nodes.

Launch

* The AI For Science Climate Lab session data for this session can be accessed under the data root directory.

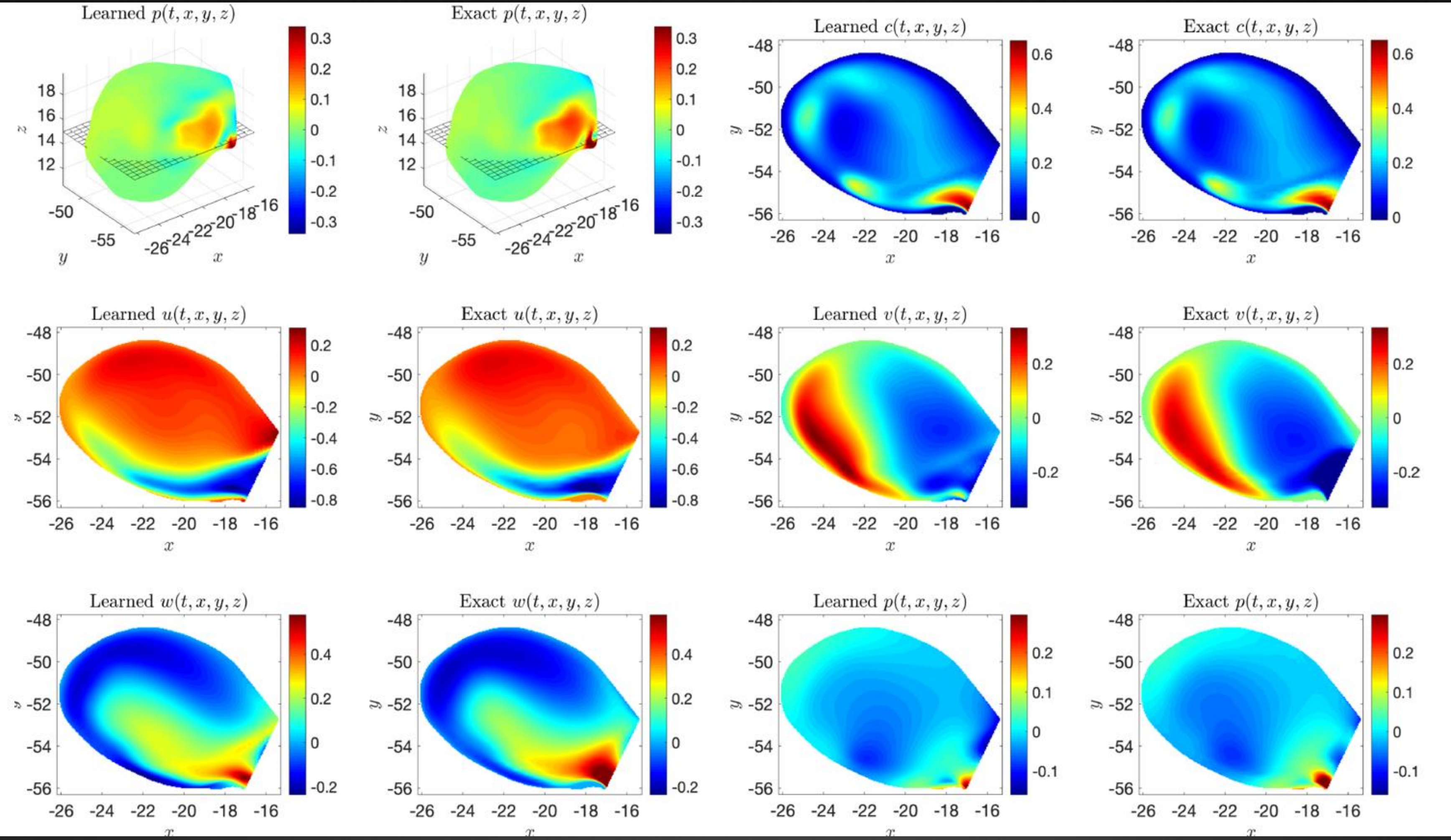
powered by OPEN OnDemand

OnDemand version: v1.8.18

65°F Partly cloudy ENG INTL 08:23 29/07/2021

STEADY STATE FLOW WITH NEURAL NETWORKS

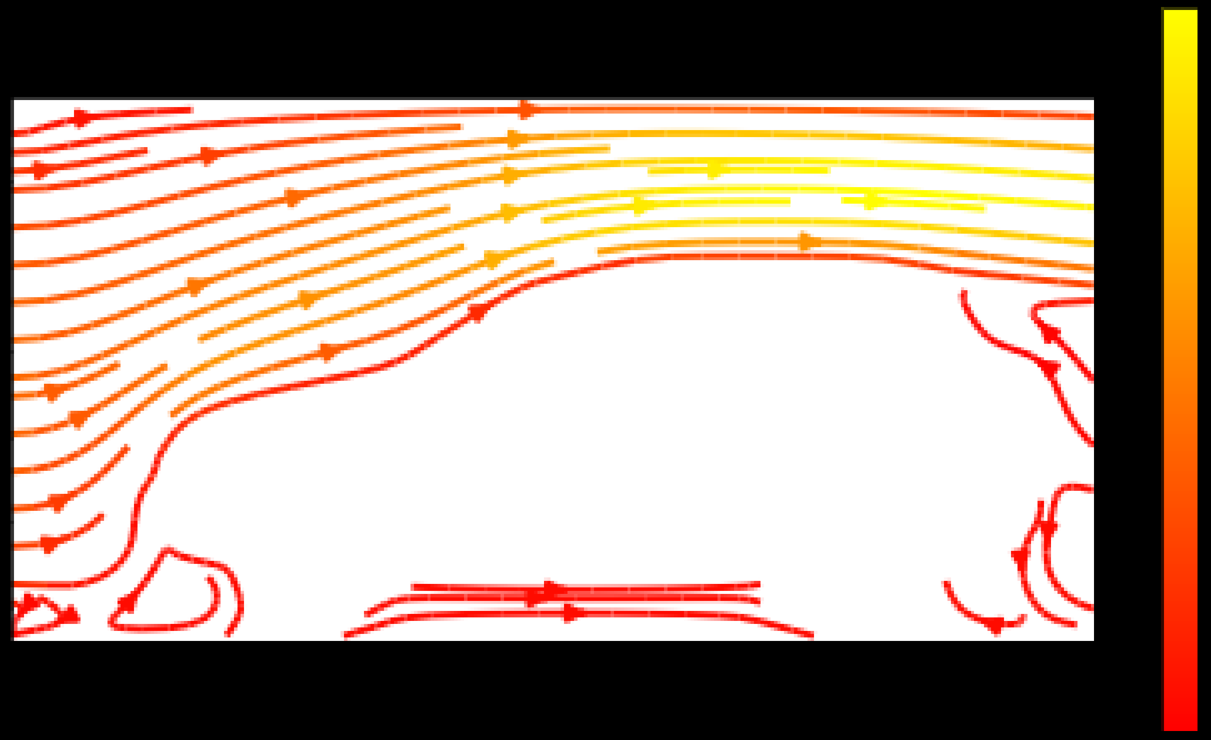
Flow fields are simulated using computational fluid dynamics (CFD) solvers



- CFD simulation is usually a computationally expensive, memory demanding and time-consuming iterative process
- *CFD limit opportunities for design space exploration and forbid interactive design*

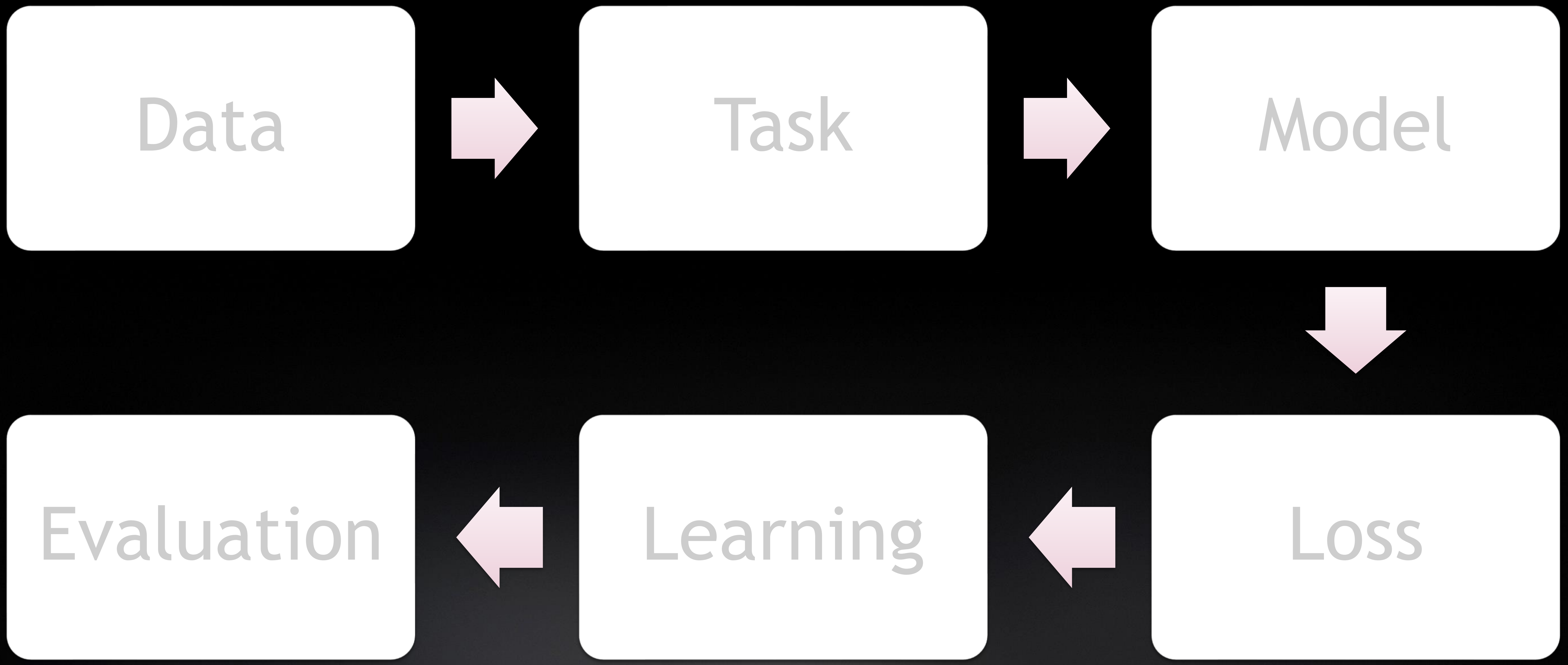
STEADY STATE FLOW WITH NEURAL NETWORKS

Our aim is to predict 2D flow around objects. The input is the boundary around which we want to calculate the flow. Here is an example of input data and the corresponding flow that was calculated using the Lattice Boltzmann method. (Mechsys).

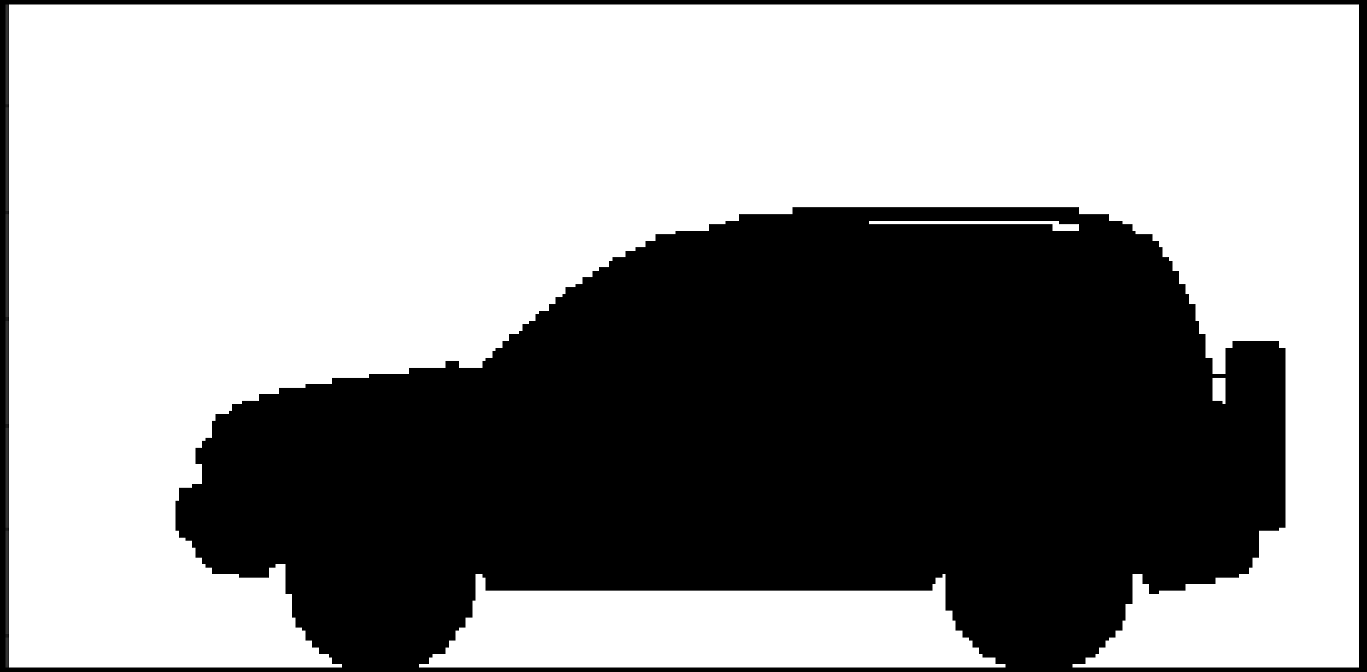


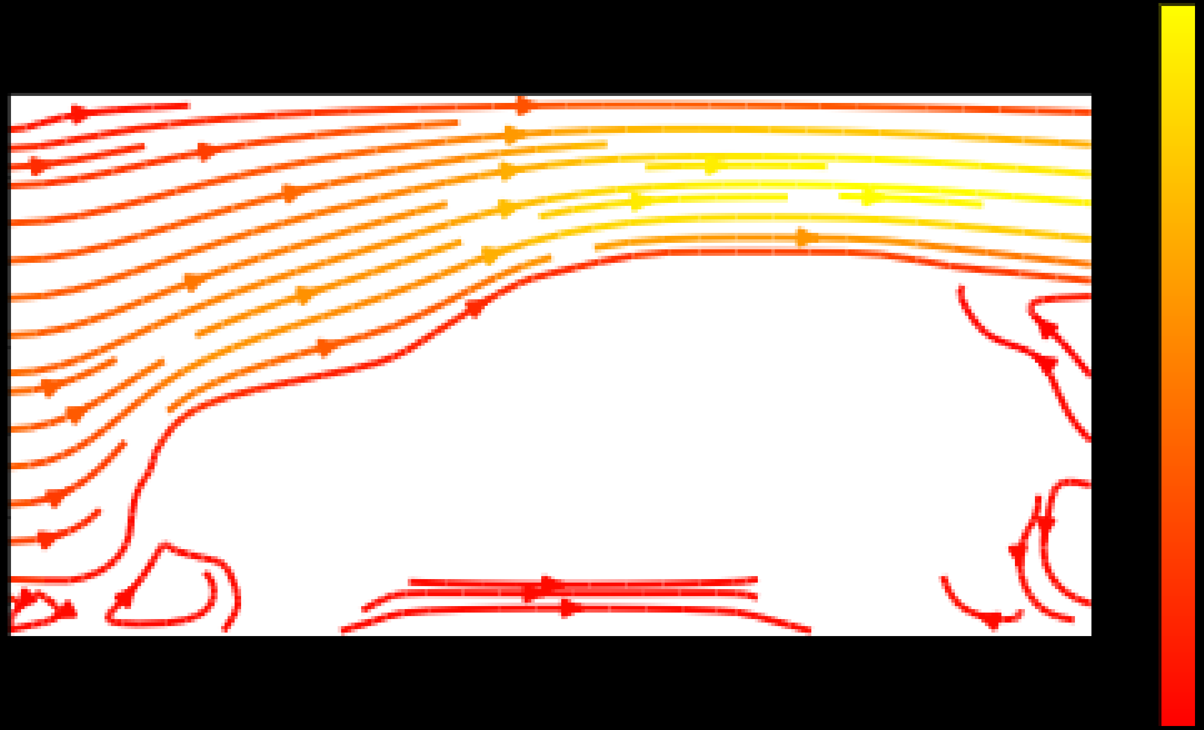
6 STEPS APPROACH

Steps to follow while solving a Machine Learning problem



DATA AND TASK



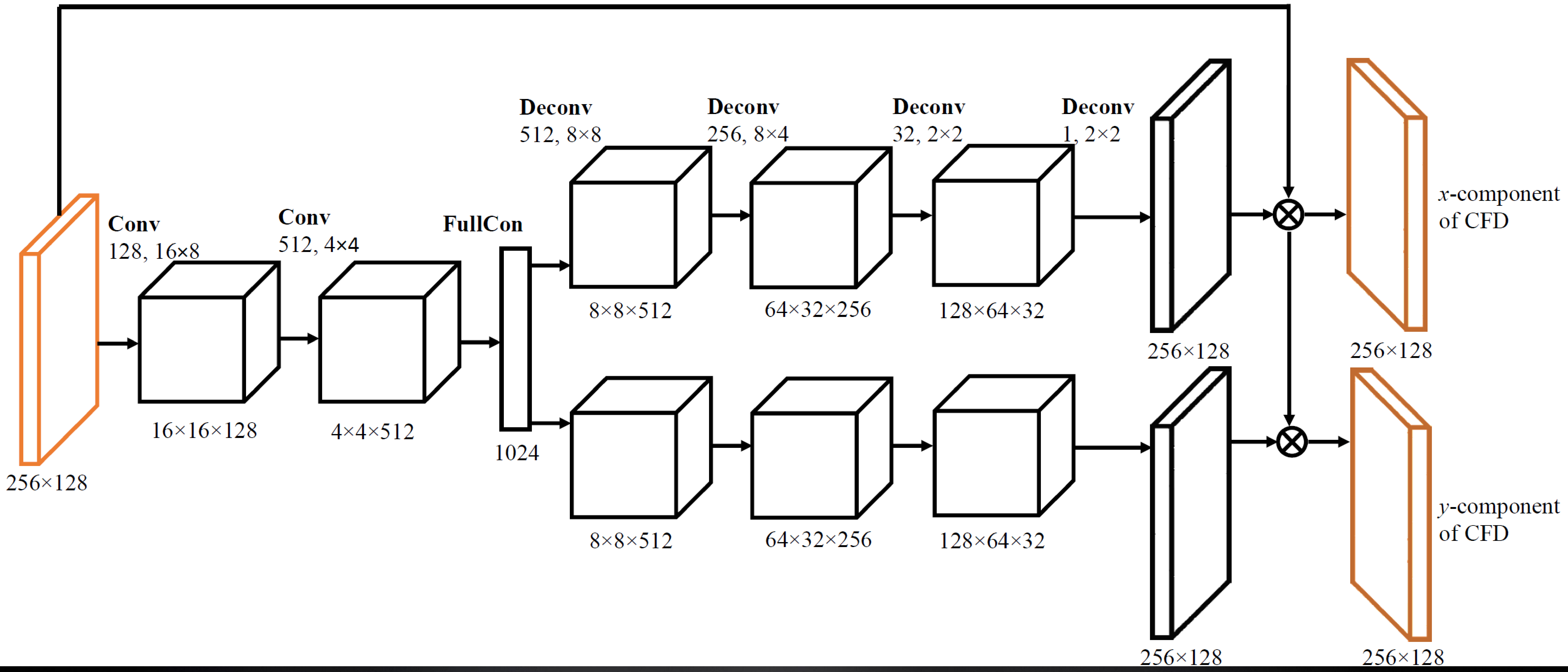


Predict the velocity vectors of both the x and y channels from our model.

MODEL

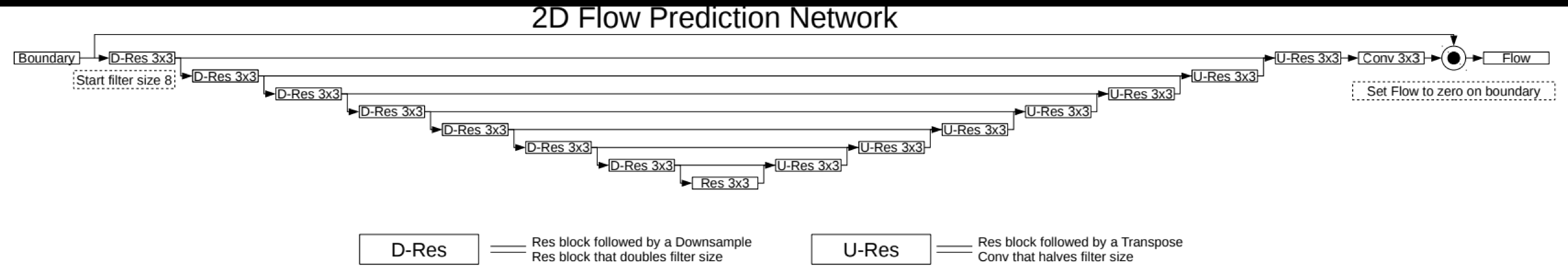
We will be building the following Models and benchmarking them as we proceed :

- Simple Fully Connected Networks
  - *3 Layer Network*
  - *5 Layer Network*
- Convolution Neural Networks
  - *Binary Boundary*
  - *Signed Distance Function*
- Advanced Networks
  - *Gated Residual Network*
  - *Non-Gated Residual Network*



Conv
128, 16×8
Conv
512, 4×4
FullCon
Deconv
512, 8×8
Deconv
256, 8×4
Deconv
32, 2×2
Deconv
1, 2×2
256×128
16×16×128
4×4×512
1024
8×8×512
64×32×256
128×64×32
256×128
256×128
x-component
of CFD
8×8×512
64×32×256
128×64×32
256×128
256×128
y-component
of CFD

U GATED NETWORK



Thanks!

gunterr@NVIDIA.com