Explainable Artificial Intelligence
Application to Tropical Cyclone CNN

Eric Wendoloski
Connor Sprague
Ingrid Guch
Kurt Roettiger
Phil Slingerland
Sue Vogel

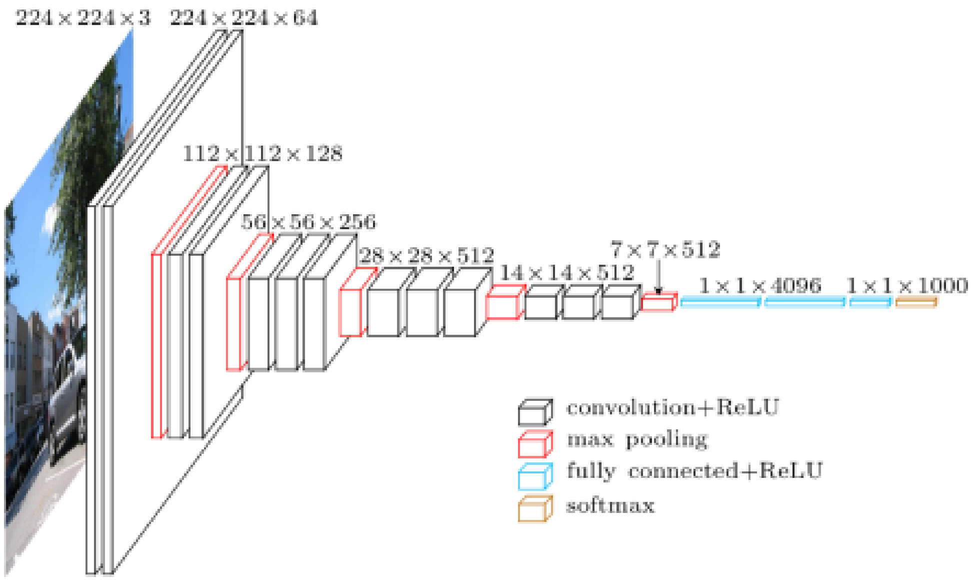
17 DEC 18

Approved For Public Release. OTR2019-00184.

Overview

- Difficulty explaining origins of AI-based prediction a key barrier to adoption

- Machine-learning (ML) models (basis for AI) often perceived as black boxes
  - Notion prevalent in deep-learning (DL; e.g., Convolutional Neural Networks – CNNs)
  - CNNs composed of interconnected layers; O(10M-100M+) trainable parameters
  - Filters within layers trained to distill relevant features that map to some output



VGG16 CNN bottleneck architecture - 138M trainable parameters
Figure adapted from Cord et al. 2016

Overview

- Operational users require explainable decision processes / knowledge of model vulnerabilities
  - Predictions informing life and death decisions must be traceable
  - Predictions of physical processes should depend on physically relevant features
  - Vulnerabilities may be exploited to alter predictions

- Explainable AI techniques capable of conveying DL model behavior/vulnerabilities
  - Analysis rooted in the visual information distillation process
  - Demonstrate physical features/concepts that the model relates to its predictions
  - Provide understanding of a vulnerability to adversarial inputs

All topics presented represent areas of open research in the community.

Tropical Cyclone (TC) Classification CNN

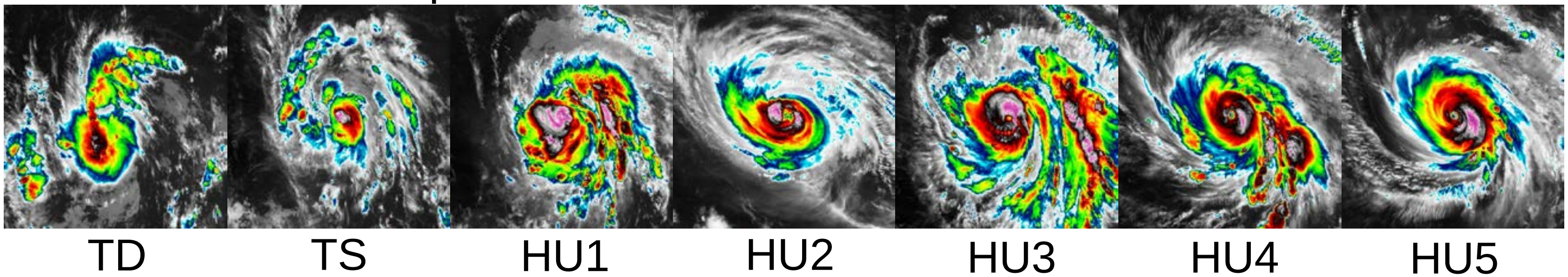
- Model trained to categorize TCs
  - Tropical depression (TD), Tropical Storm (TS), Cat 1 – Cat 5 Hurricanes
  - Null class included of randomly pulled regions away from TCs

- **Model only intended to highlight application of methods**
  - Methods extensible to numerous environmental problems

- TC Data:
  - 2017/2018 Atlantic & Eastern Pacific TCs – interpolated to hourly
    - 2017 – NOAA Best Track Data
    - 2018 – CIRA RAMMB Archive

- CNN Input
  - GOES-East satellite images - 11.2 um band
  - Image chips centered on TC center of circulation
  - ~14000 images from 71 TCs

Maria TD-HU5 examples

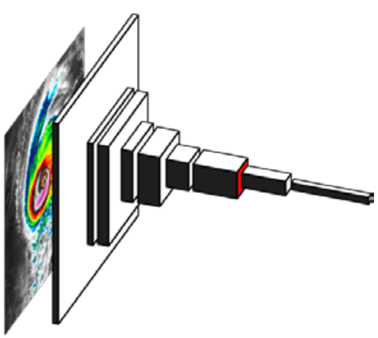


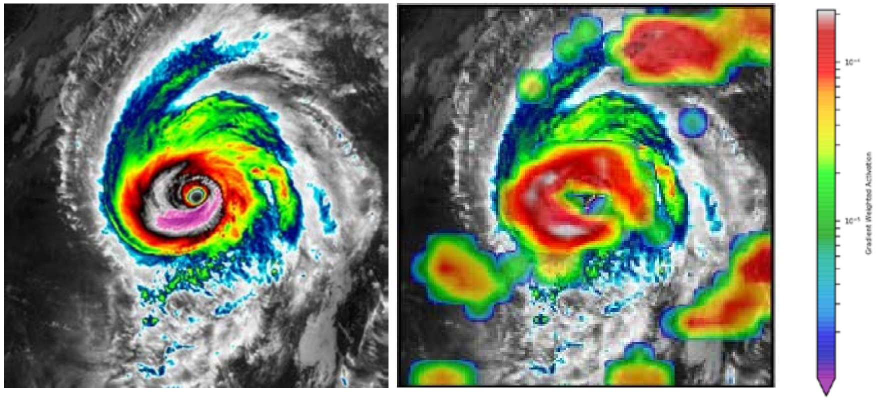
Gradient Weighted Class Activation Mapping (Grad-CAM)

- Objective
  - Determine if the model focusing on the most important features for prediction

- Visual explanation for CNN decisions
  - Indicates input image pixels most important to prediction of a class
    - These pixels positively influence prediction of a given class

- Method (Selvaraju et al. 2017)
  - Run image through CNN
  - Gather **activations** – output of a CNN layer
  - Compute gradient of the predicted score for class of interest
  - Global average pool the gradients – one avg. gradient for each filter in the layer
  - Multiply activations by respective gradients, apply RelU
    - Weights activations by how important they are to predicting the class of interest
  - Results may be aggregated as layer mean or individual filter results can be examined
  - Results viewed as heatmap of pixel importance on the input image

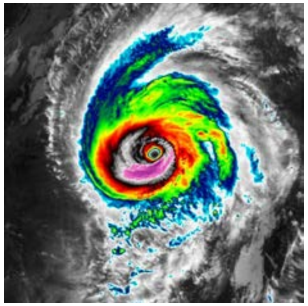




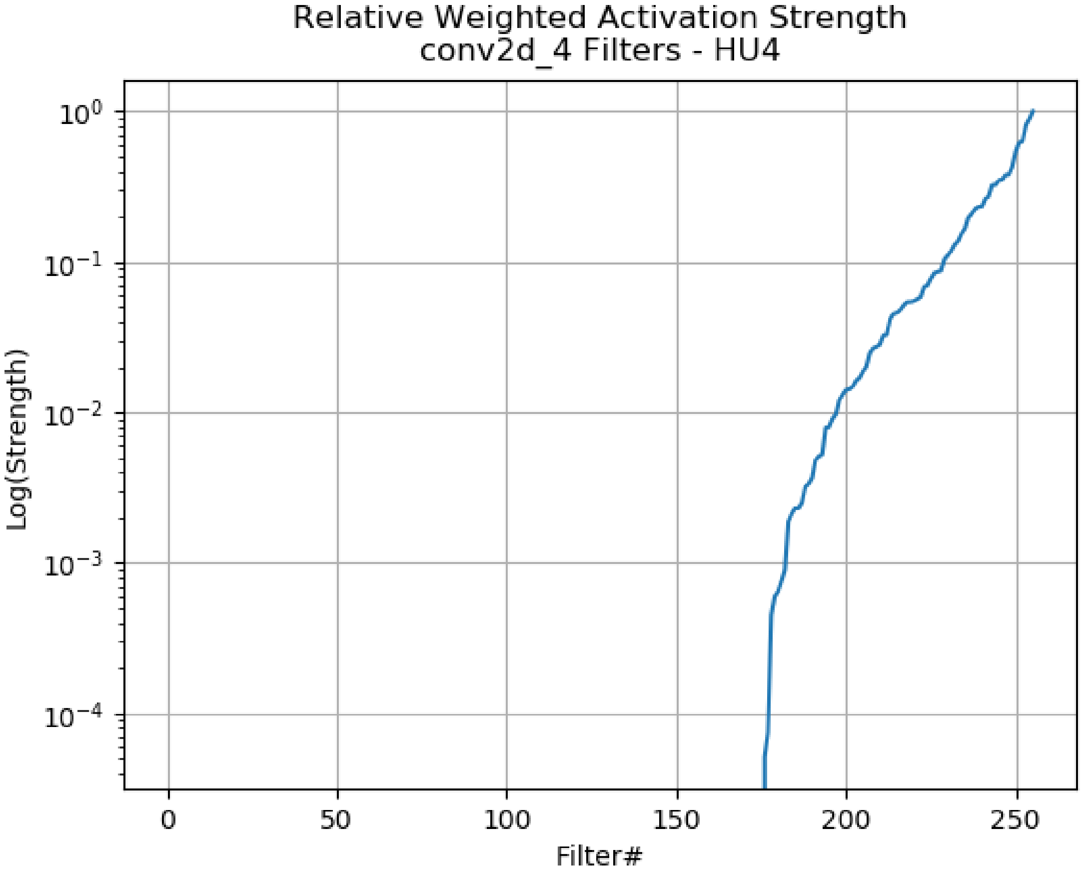
Michael as Cat. 4 (left), Grad-CAM heatmap (right; final conv. layer, 3rd strongest)

Grad-CAM-based filter influence in TC CNN Final Conv Layer

- Gathered weighted activation for all 256 filters in last layer of TC CNN
  - Summed weighted activations for each filter
  - Allows sorting of filters that produced most influential output to Cat. 4 prediction



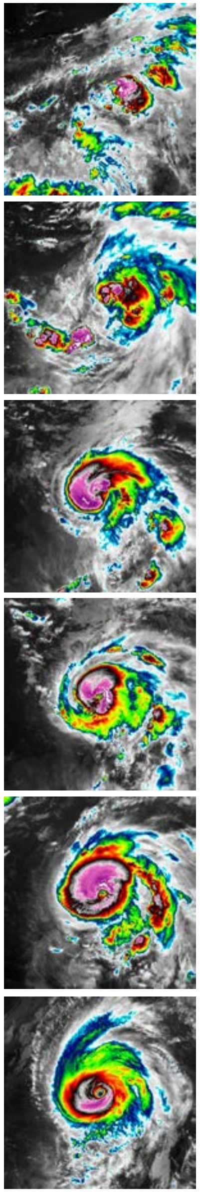
Input: Michael as HU4

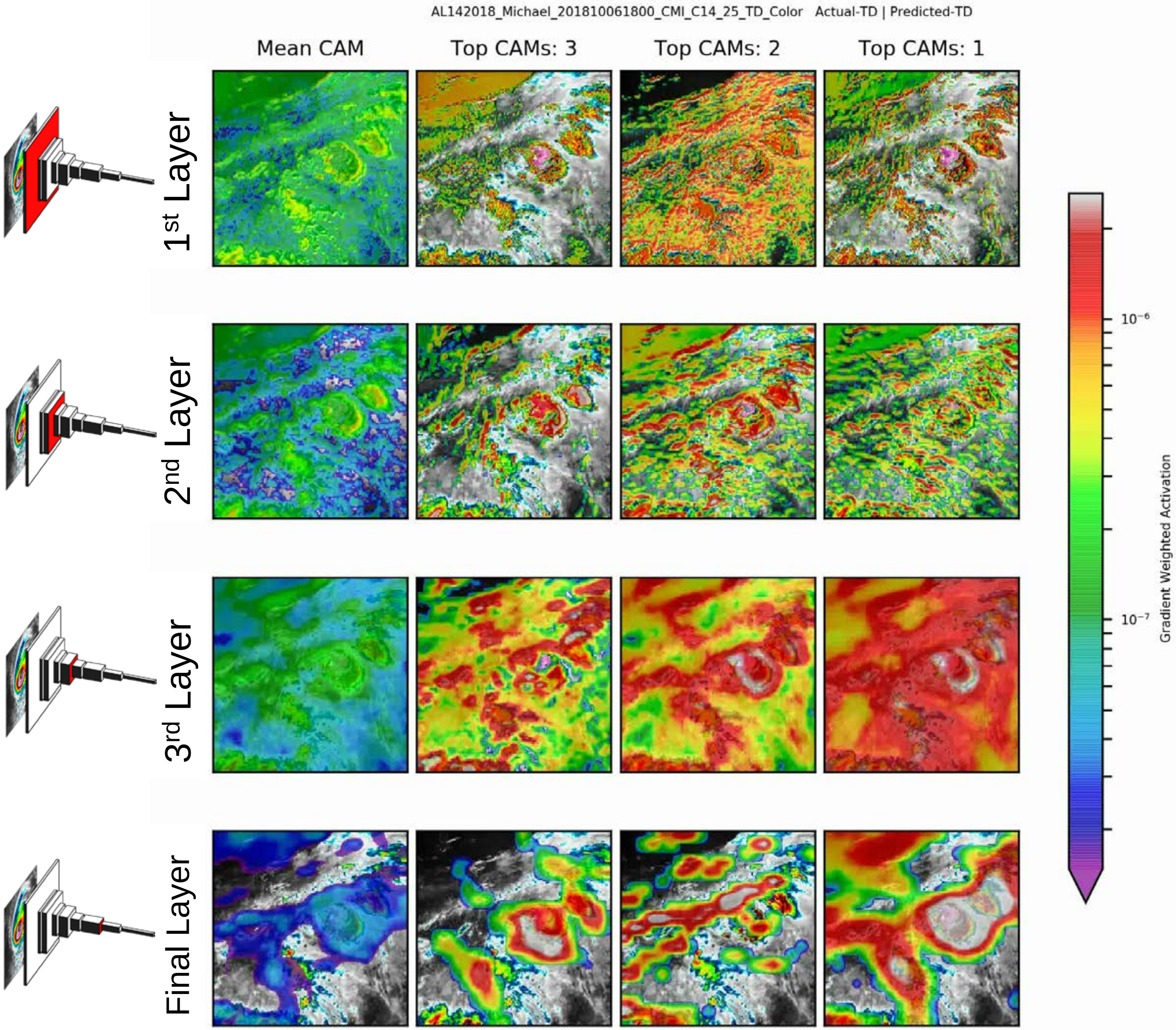


- Only 80/256 filters in final layer produced output with meaningful positive influence on the HU4 prediction
- The top filters may be 1+ orders of magnitude more influential to class prediction than remaining filters
- Behavior of active filters has implications for model-capacity
  - *Are some of the filters inactive or non-influential over all classes? Model may be deeper than necessary*
  - *Are most filters active over all classes? Model may be too shallow*

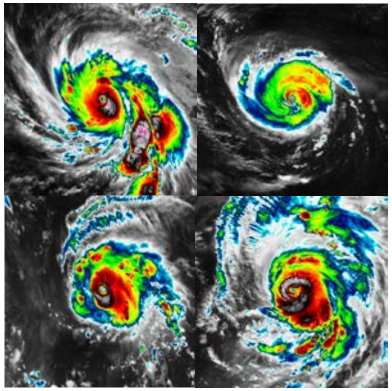
CAM Heatmaps – Michael as TD – Cat. 4 by Conv. Layer

Inputs





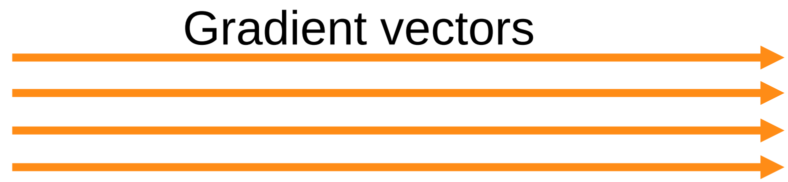
Testing with Concept Activation Vectors (TCAV; Kim et al. 2018) - Importance of Eye structure to HU4 predictions



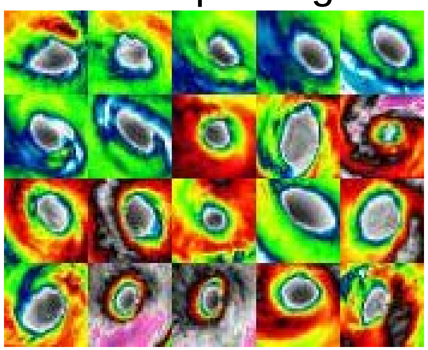
- Gather known HU4 class images.
- A few hundred here.
- Taken from training set.

Calculate gradient of model loss function for HU4 class w.r.t. activations from final layer

Gradient vectors point in direction of decreasing probability of correct class identification

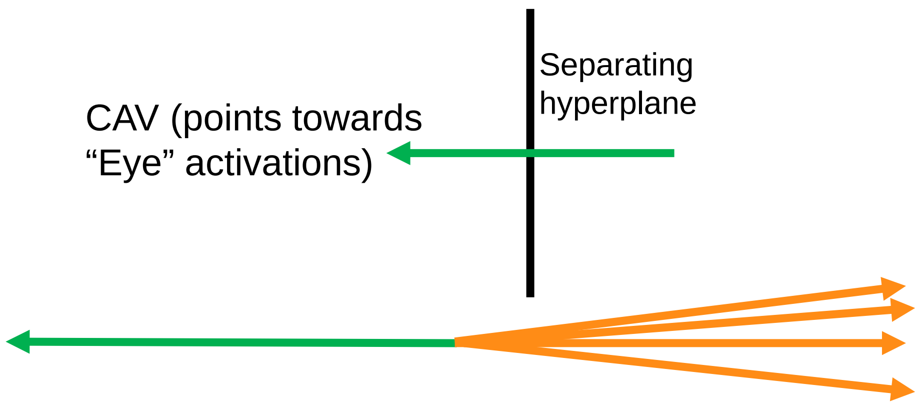


Concept images

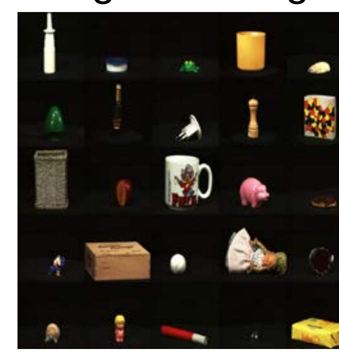


Determine Concept Activation Vector

1. Gather concept images (eye; N>50)
2. Gather negative images
3. Gather layer activations for 1,2
4. Train linear classifier on activations to obtain CAV



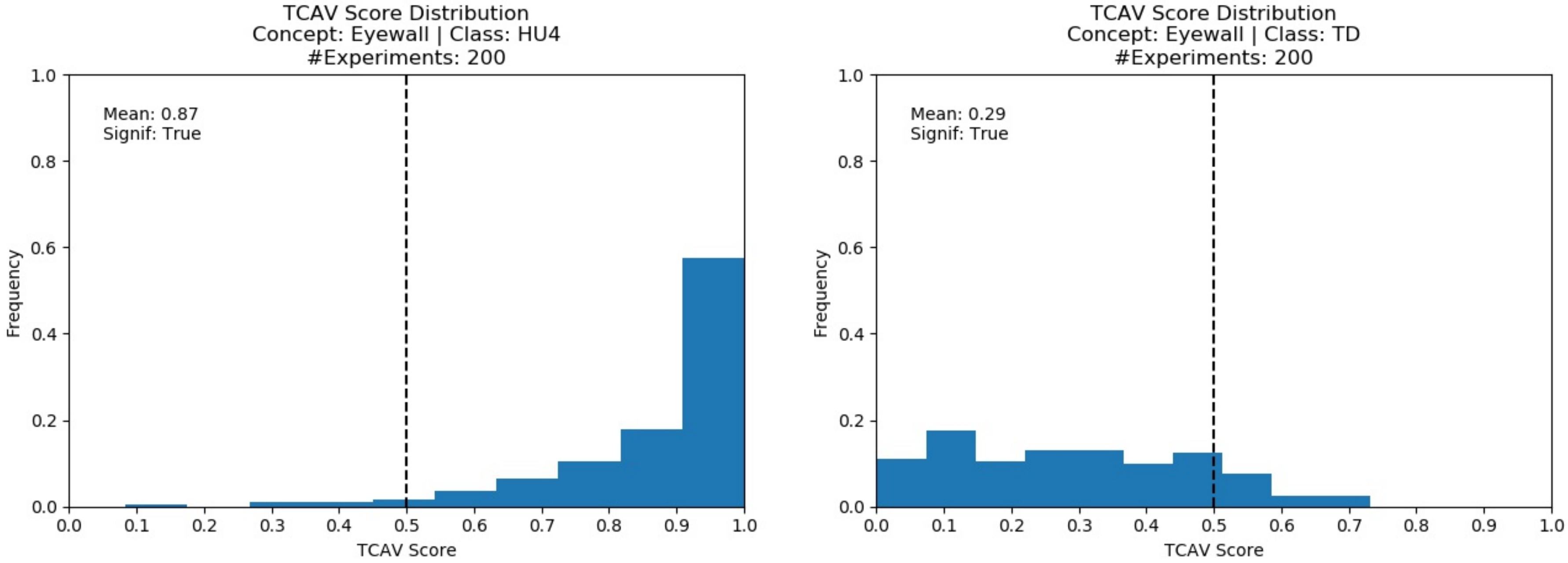
Negative images



Negative images from ALOI (Geusebroek et al. 2005)

Does CAV tend to point in opposite direction of gradient vectors?
If so, CAV points in direction of increasing probability of correct class identification

“Eye” Concept Activation Vectors



- Kim et al. 2018 suggests a concept is important if CAVs point opposite of >50% of gradient vectors

- 87% of HU4 gradient vectors point in opposite direction of “Eye” CAV
    - Eye important to prediction of Category 4 TCs

- 29% of TD gradient vectors point in opposite direction of “Eye” CAV
    - Eye not important to prediction of TD class

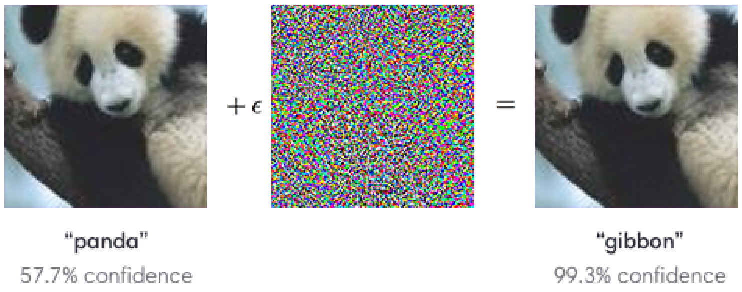
- Results obvious for this test case, but:
    - Such methods extensible to physical features deemed most important by the user for a given model and class.

Adversarial Attacks

- Purpose of most adversarial attacks to cause erroneous model inference
  - Occurs at testing or deployment stage
  - Targeted – misguide to specific class
  - Untargeted – misguide to arbitrary class

- Adversarial generation noise to images causing misclassification
  - Noise often imperceptible to humans
  - Adversarial examples expose and exploit flaws in decision function of model
  - Model-to-model transferability possible (Papernot et al. 2016)
  - Implies model security risk even when attacker has no access to victim model

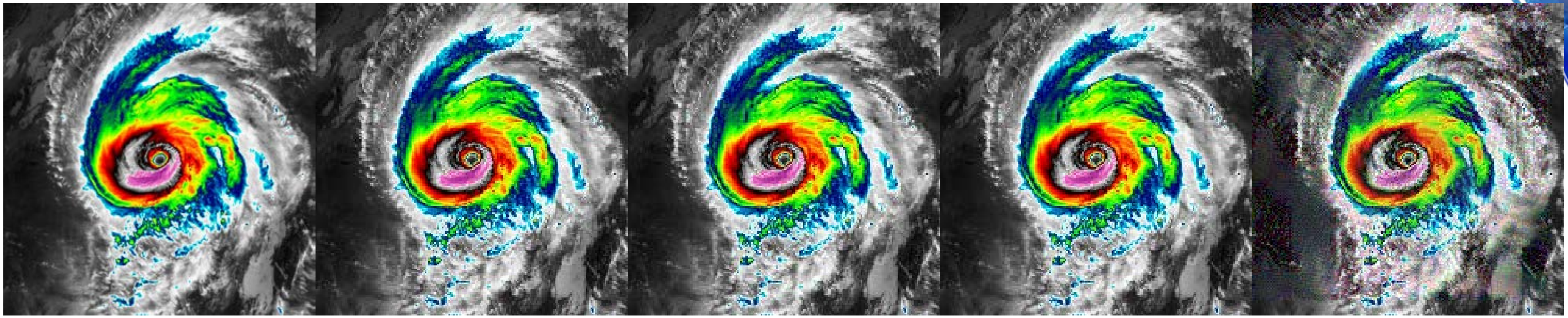
- Operational use of CNNs requires knowledge of vulnerabilities/robustness
- May require ability to detect, screen, and remediate adversarial inputs



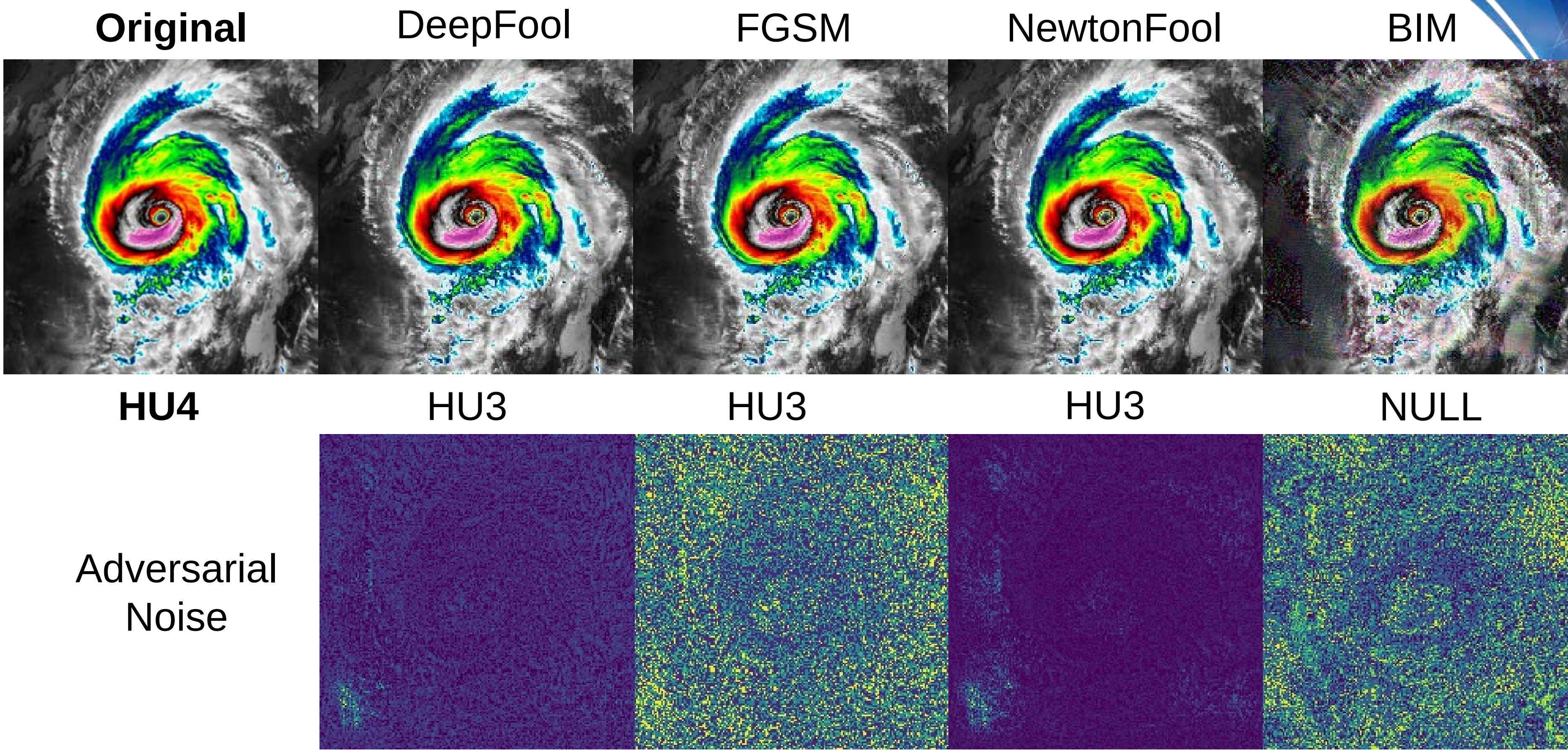
Fast Gradient Sign Method (FGSM) - Goodfellow et al. 2015

Adversarial Examples

Pick the original image. Four images are adversarial.



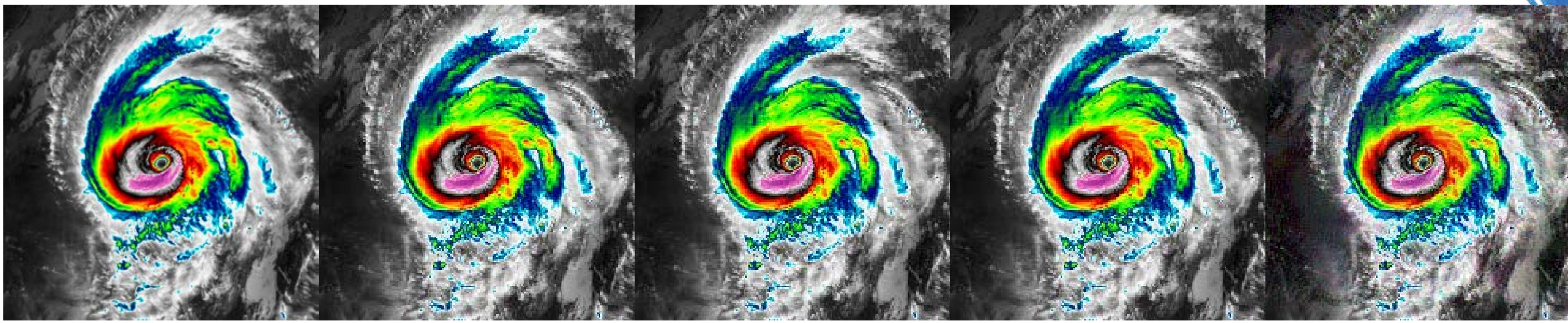
Adversarial Examples



- Adversarial methods used here
  - Fast Gradient Sign Method (FGSM) – Goodfellow et al. (2015)
  - NewtonFool – Jang et al. (2017)
  - DeepFool – Moosavi-Dezfooli et al. (2016)
  - Basic Iterative Method (BIM) - Kurakin et al. (2016)

Adversarial Examples

Original	DeepFool	FGSM	NewtonFool	BIM

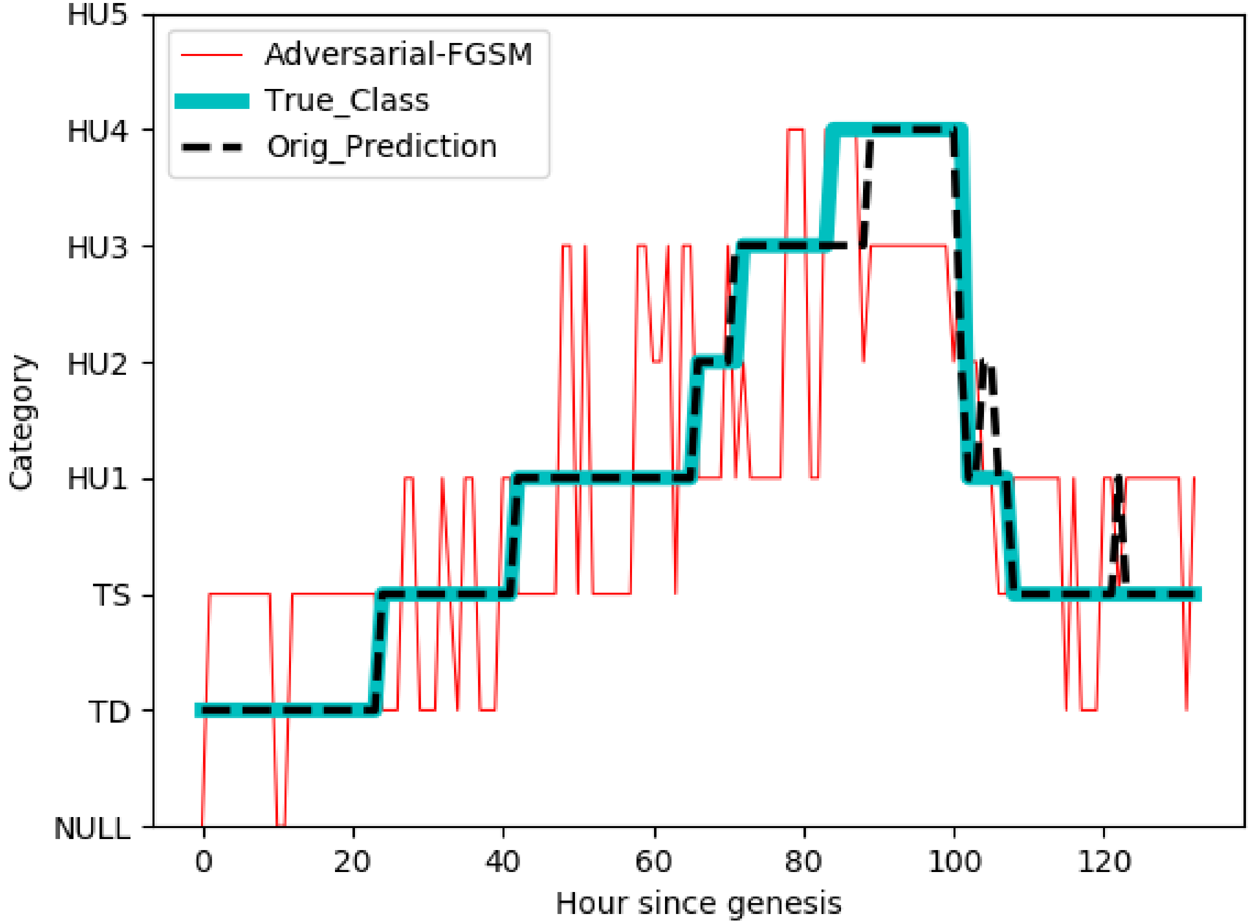


HU4	HU3	HU3	HU3	NULL

Varying levels of success in altering Michael Top-1 accuracies

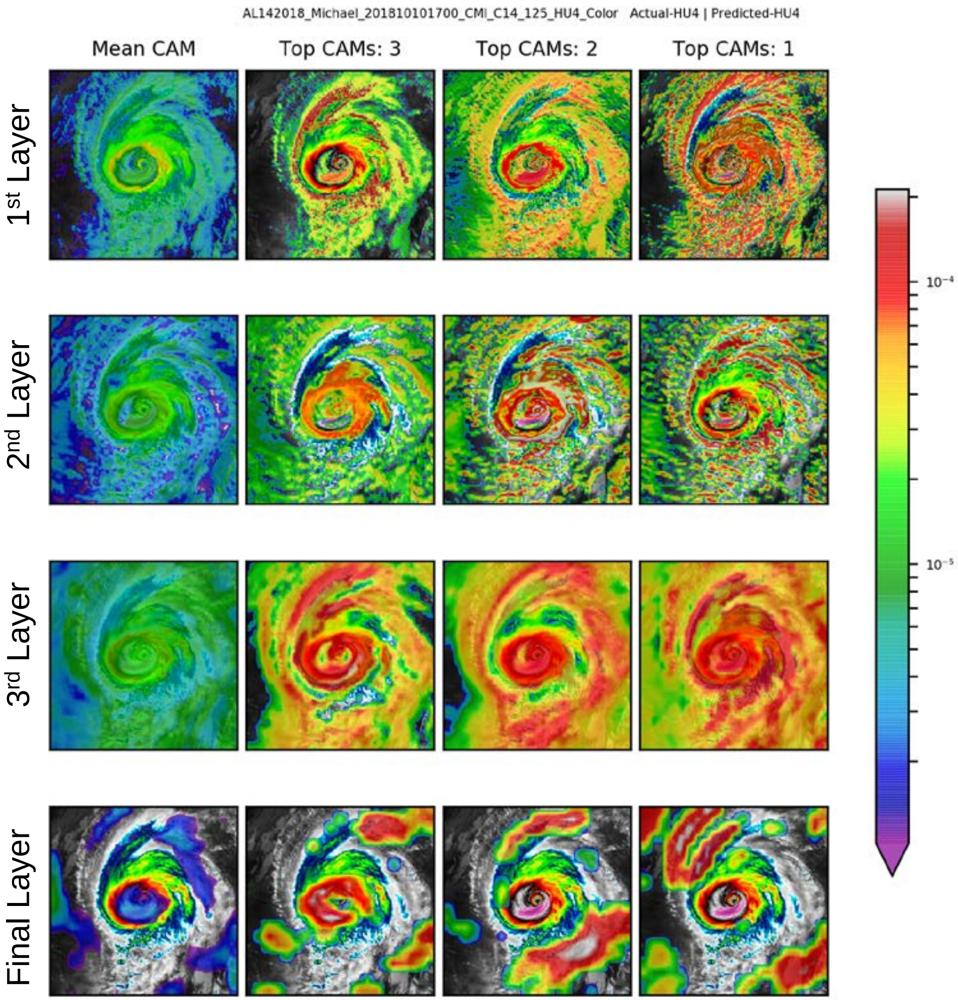
Attack	**Top-1 Accuracy (%)** **N=123**
Original (clean)	100
DeepFool	40.7
FGSM (min ϵ [0.01,0.3])	0.81
NewtonFool	0.00
BIM	38.2

Hurricane Michael Predictions

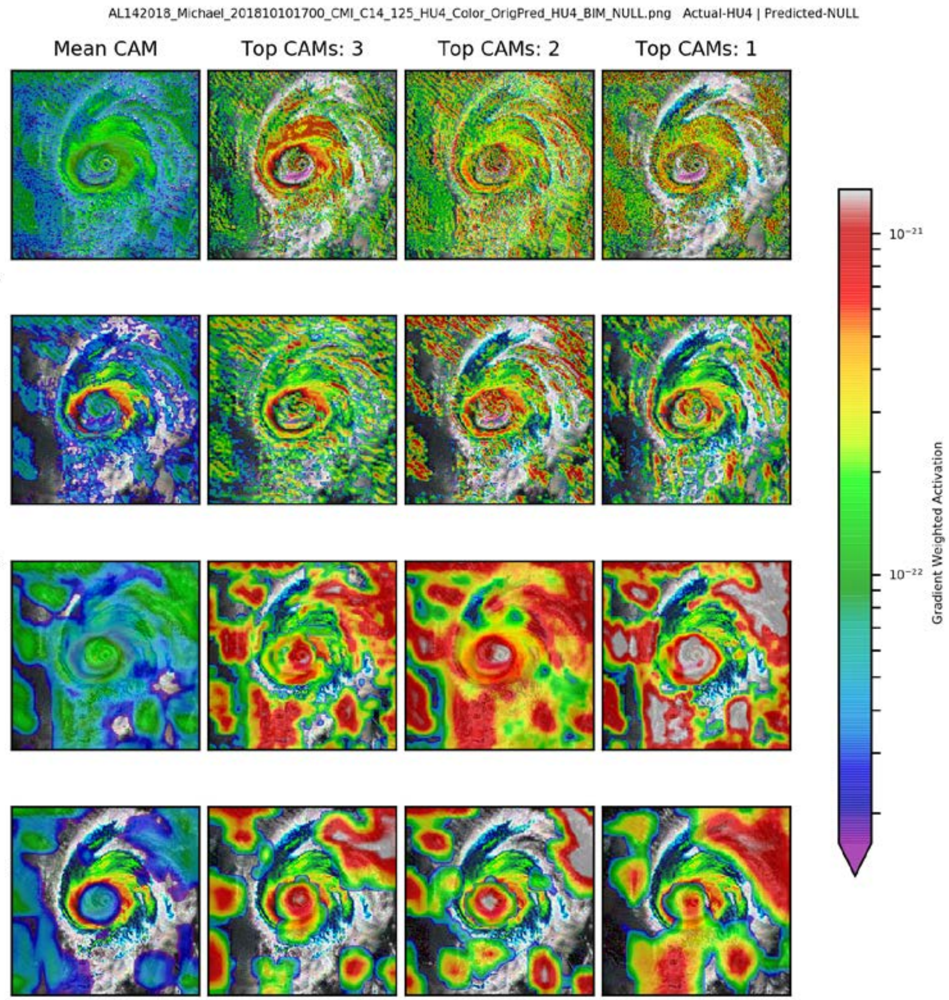


Grad-CAM Heatmaps – Michael – Cat. 4 + Adversarial Noise

Original Input Image CAMs



Adversarial Input Image CAMs



Original prediction: HU4
Adversarial prediction: NULL – note less focused, activations 17 orders of magnitude weaker

Quantifying Model Robustness

- Metrics for quantifying model robustness
  - Minimal perturbation required for misclassification (most common)
  - Sensitivity of loss function to input changes
  - Sensitivity of logits w.r.t. input changes

- Empirical robustness (ER – minimal input perturbation required for success of given attack)
  - Euclidean distance between successful adversarial & original input (Moosavi-Dezfooli et al. 2016)
  - Larger ER → for a given attack type, larger perturbations required

Class	ER
TD	0.013
TS	0.016
HU1	0.011
HU2	0.012
HU3	0.013
HU4	0.011
HU5	0.011
NULL	**0.133**

- NULL class order of magnitude more robust to FGSM
  - Random cloud fields and cloud free regions
  - Model built more robust decision boundary for NULL
  - Leaves less opportunity for adversaries to exploit deficiencies

- TC classes built on finite number of storms
  - Limited data causes deficiencies in space of storm structures
  - Similarities from class to class problematic
  - Easier to exploit deficient decision boundaries

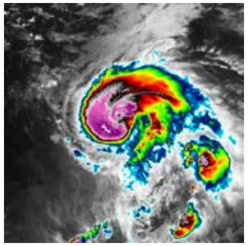
- Model hardening possible through fine-tuning on adv. examples
  - Hardening HU4 class for FGSM attack
  - **ER: 0.011 → 0.127**

Adversarial Defense Methods

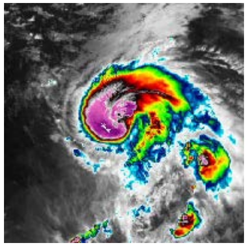
- Increase robustness with model hardening
  - Training on adversarial imagery (fine-tuning or from start)
    - Shown to offer regularization
  - Noise data augmentation during training

- Reduce adversarial noise through data preprocessing at test/deployment
  - E.g., filter noise by reducing bit depth, spatial smoothing, data compression

Original



Adversarial



	HU1	**HU2**	**HU3**	**HU4**	**HU5**	**NULL**	**TD**	**TS**
Original	**0.994**	0.002	0.002	0.000	0.000	0.000	0.000	0.002
FGSM	0.261	0.080	**0.601**	0.000	0.000	0.000	0.000	0.058
PNG	0.261	0.080	**0.601**	0.000	0.000	0.000	0.000	0.058
JPEG-Q100	**0.545**	0.058	0.333	0.000	0.000	0.000	0.000	0.063
JPEG-Q1	**0.712**	0.055	0.108	0.000	0.000	0.001	0.006	0.117

- Runtime detection
  - Additional classifier fit on adversarial and clean data (activations or inputs)

See Nicolae et al. 2018 for details on common defense methods.

Summary

- Grad-CAM
  - Capable of highlighting input pixels most important to class prediction
  - Helpful in diagnosing filter contributions to class prediction
  - Results local to input images
  - May be used to flag certain potential adversarial attacks

- TCAV
  - Offers general method to test if a concept deemed important by the user is significantly important to the prediction of a given class

- Adversarial
  - ML models can be highly vulnerable to adversarial attack
    - Extends to shallow and deep architectures
  - Attacks expose weaknesses in generalization of decision boundaries
  - Fine-tuning allows model to generalize such that decision-boundary flaws are remediated

- Future efforts
  - Extension of explainable AI approaches to regression rather than classification
  - Visualization of decision-boundary improvements
  - General methodology to increase attack-agnostic robustness

References

Carlini, N. and D. Wagner, 2017: Towards evaluating the robustness of neural networks. https://arxiv.org/abs/1608.04644v2

Cord, M., T. Durand, N. Thome, and P. Gallinari, 2016: Deep learning and weak supervision for image classification. Accessed 12 November 2018, http://webia.lip6.fr/~cord/pdfs/news/TalkDeepCordI3S.pdf

Geusebroek, J, G. Burghouts, and A. Smeulders, 2005: The Amsterdam library of object images, *International Journal of Computer Vision*, **61**, 103-112.

Goodfellow, I, J. Shlens, and C. Szegedy, 2015: Explaining and harnessing adversarial examples. https://arxiv.org/abs/1412.6572v3

Jang, U., X. Wu, and S. Jha, 2017: Objective metrics and gradient descent algorithms for adversarial examples in machine learning. https://doi.org/10.1145/3134600.3134635

Kim, B., M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, and R. Sayres, 2018:Interpretability beyond feature attribution: quantitative testing with concept activation vectors (TCAV). https://arxiv.org/abs/1611.11279v5

Kurakin, A., I. Goodfellow, and S. Bengio, 2016: Adversarial machine learning at scale. https://arxiv.org/abs/1807.012396

Landsea, C. W. and J. L. Franklin, 2013: Atlantic Hurricane Database Uncertainty and Presentation of a New Database Format. Mon. Wea. Rev., 141, 3576-3592.

Moosavi-Dezfooli, S., A. Fawzi, and P. Frossard, 2016: DeepFool: a simple and accurate method to fool deep neural networks. https://arxiv.org/abs/1511.04599

Nicolae M.I., and Coauthors, 2018: Adversarial robustness toolbox v0.3.0. https://arxiv.org/abs/1807.01069

Papernot, N., P.D. McDaniel, and I.J. Goodfellow, 2016: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv:1605.07277

RAMMB, 2018: Real-time tropical cyclone products – 2018 Season. Accessed 12 November 2018, http://rammb.cira.colostate.edu/products/tc_realtime/season.asp?storm_season=2018

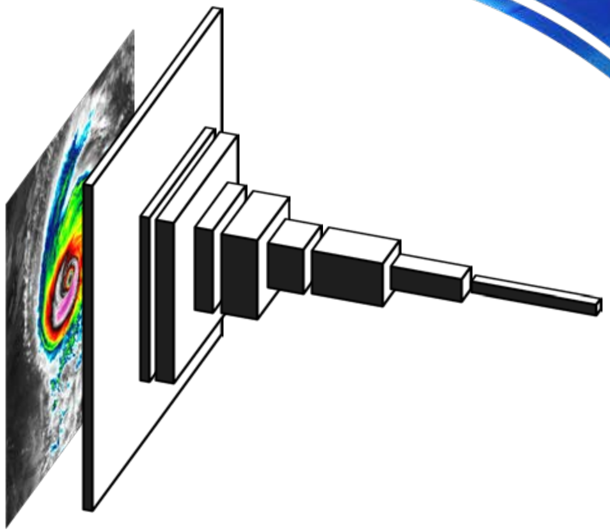
Selvaraju, R.R., M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, 2017: Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. https://arxiv.org/abs/1610.02391v3

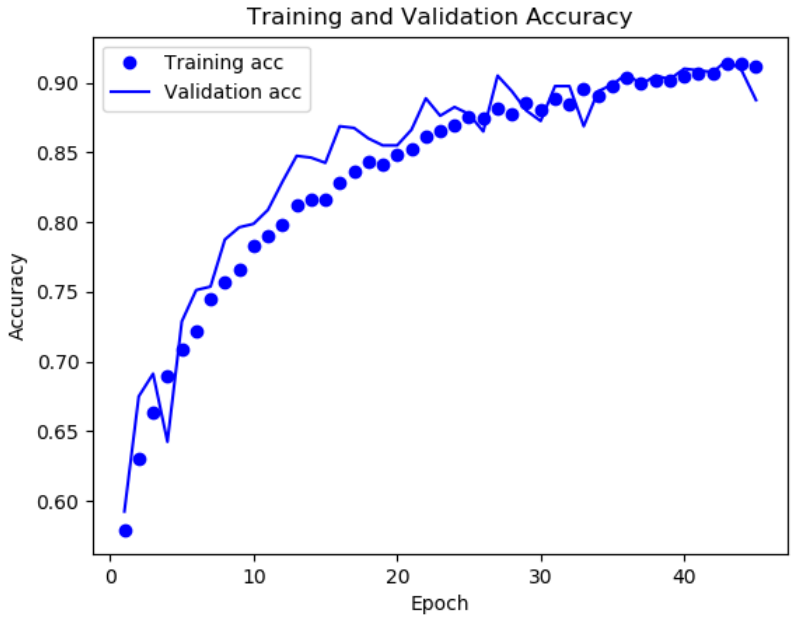
Useful Toolboxes

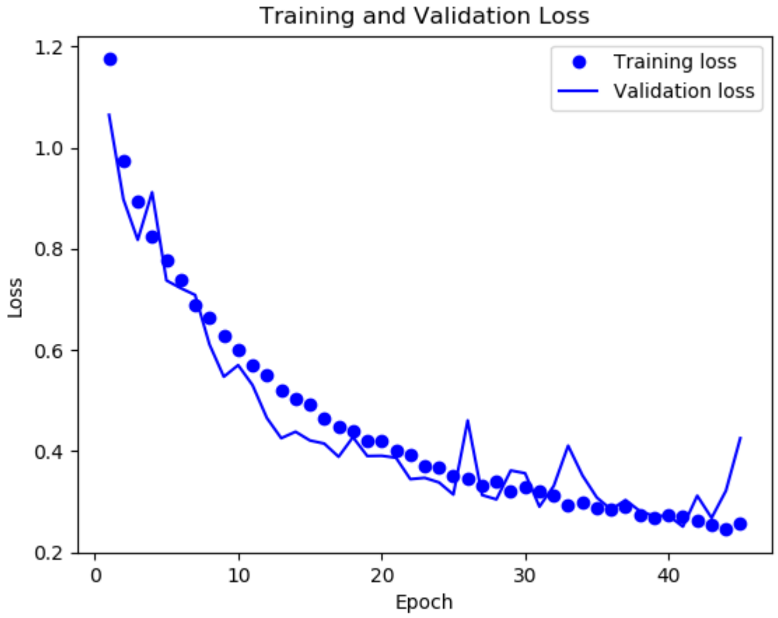
IBM Adversarial Robustness Toolbox - https://github.com/IBM/adversarial-robustness-toolbox
FoolBox - https://github.com/bethgelab/foolbox
CleverHans - https://github.com/tensorflow/cleverhans

EO Tropical Cyclone CNN

- Model Architecture
  - *4 convolutional layers*
  - *1 dense layer mapped to TC classifications*
  - *52M trainable parameters*
  - *Regularized via dropout and data augmentation*
    - Prevents overfitting to training data
  - *Test accuracy: ~90%*
  - *Weights saved at epoch 40*







EO Tropical Cyclone CNN

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 198, 198, 32)	896
max_pooling2d_1 (MaxPooling2	(None, 99, 99, 32)	0
conv2d_2 (Conv2D)	(None, 97, 97, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 48, 48, 64)	0
conv2d_3 (Conv2D)	(None, 46, 46, 128)	73856
max_pooling2d_3 (MaxPooling2	(None, 23, 23, 128)	0
conv2d_4 (Conv2D)	(None, 21, 21, 256)	295168
max_pooling2d_4 (MaxPooling2	(None, 10, 10, 256)	0
flatten_1 (Flatten)	(None, 25600)	0
dropout_1 (Dropout)	(None, 25600)	0
dense_1 (Dense)	(None, 2048)	52430848
dense_2 (Dense)	(None, 8)	16392

Total params: 52,835,656
Trainable params: 52,835,656
Non-trainable params: 0