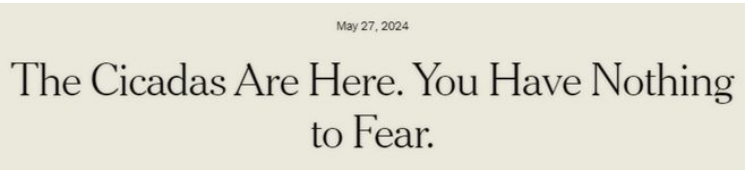
CICADA -like?

PRINCETON UNIVERSITY

# Tutorial 1: Anomaly Detection on CMS Open Data

Andrew Loeliger, Elliott Kauffman


May 27, 2024

The Cicadas Are Here. You Have Nothing to Fear.

New York Times

I love these headlines from last year too much to not include them in anything I make…


Kindness in the Time of Cicadapocalypse

New York Times

# A Quick Bit of Intro

PRINCETON UNIVERSITY

- I am Andrew Loeliger, a post-doc at Princeton, and part of the CICADA Team. I am being helped out by Elliott Kauffman, our Ph.D. student with CICADA

- In the exercises/tutorials I have prepared, I have attempted to design a spread across a pretty broad range going from neural-network-less basic methods to some advanced stuff with graph autoencoders, and skill levels going from “first time with machine learning” to “experienced”.

- There is probably too much stuff to do all at once. If you really want to try a bunch of stuff out, I do recommend going through this stuff offline tonight or some other time.
    - I would guess someone could get through about 2 or so of these, but if I have drastically underestimated, or the exercises are too basic for the large amount of gathered expertise here, I am throwing down the gauntlet to challenge the room to surprise me with the most unique and effective anomaly detection idea you can make in the next hour on the open data I have provided. I also appreciate delightfully (or deceptively) simple but effective methods.
- Depending on preferences in the room, I can focus on 1 or more of the tutorials in front of the room
    - Elliott and myself will be available for assistance around the room hopefully

# The Exercises/Tutorials

1. Anomaly detection *without* neural networks with SciKit-Learn outlier detection
   a. “Basic”
   b. Looks at covariance estimation and isolation forests
   c. Using Object Multiplicity and MET
2. Basic flat autoencoders, “time”-distributed autoencoders, and knowledge distillation with some set-invariant (adjacent) ideas
   a. “Intermediate”
   b. Basic neural network techniques for anomaly detection, and some compression ideas (used by the the AD triggers and FPGA methods)
   c. Using Jet information
3. 2D-Convolutional auto-enccoders, and exploring some tuning and esoteric losses
   a. “More Advanced”
   b. Using Object 4-vectors in a grid
4. Graph Autoencoders
   a. “Advanced”
   b. Using Particle Flow Candidates arrayed as k-nearest neighbors set of graphs.

# Link, and a How-To

PRINCETON UNIVERSITY

- The tutorials I have prepared are here: https://github.com/aloeliger/CMS_OpenData_Exercises

- It is designed to be run on linux, and the only real prerequisites should python3, tar, and gzip
  - It would be good to run this on a machine with a GPU… but it should be possible to do without
- There is a setup script in there, “setup.sh” that should be run *once*.
  - Installs a python virtual environment and quite a few packages
  - Decompresses a bunch of data
  - If you need to get out of the environment use “deactivate” and if you need to get back into the environment, don’t rerun the script, instead source the environment activate script
- The environment and data is spacious!
  - Environment takes about 7.5 Gb (contains both torch and tensorflow)
  - Data takes about 5.6 Gb decompressed
  - I would recommend running somewhere you have about 15 Gb of space, 20 would be ideal
- The exercises themselves are Jupyter notebooks. If you are ssh’d into a machine, I recommend port forwarding! I have a note in the README about how to do that.
  - There are 4 Exercises marked by difficulty I expect them to have
  - There are notebooks with “_tutorial” and notebooks without. The “_tutorial” notebooks are the exercises to work through, the ones without are “answer key” ones, with the code I wrote to do all these things. You can take a look at those if you are stuck

# CERN/CMS Open Data

- CERN maintains open data from many experiments, including the 4 LHC experiments here: https://opendata.cern.ch/

The Link:
https://github.com/aloeliger/CMS_OpenData_Exercises