Trigger proposal

Igor Pains
with Rafael Nóbrega

03/10/2024
Analysis & reconstruction meeting

1.
Introduction

Introduction

- Motivation: reduce data to manageable levels by selecting only events of interest, saving storage and processing resources.
  - Each run containing 400 images need ~1.36 Gb to be stored.
  - ~266 Gb of data was produced on September 26th.

Proposal

- Develop algorithms to be tested as online trigger to decide whether to save or not images taken by the detector.
  - Convolution of the image with several kernels: look for high correlation points. presentation
  - Explore Machine Learning methods (CNN). presentation

Improvements

- The CNN was improved by using bayesian optimization during the training.
  - Four different CNNs (with 6, 7, 8 or 9 convolutional layers) were achieved.

- CPU (cloud) and GPU (google colab) time analysis was performed.

2.
Dataset

Simulation

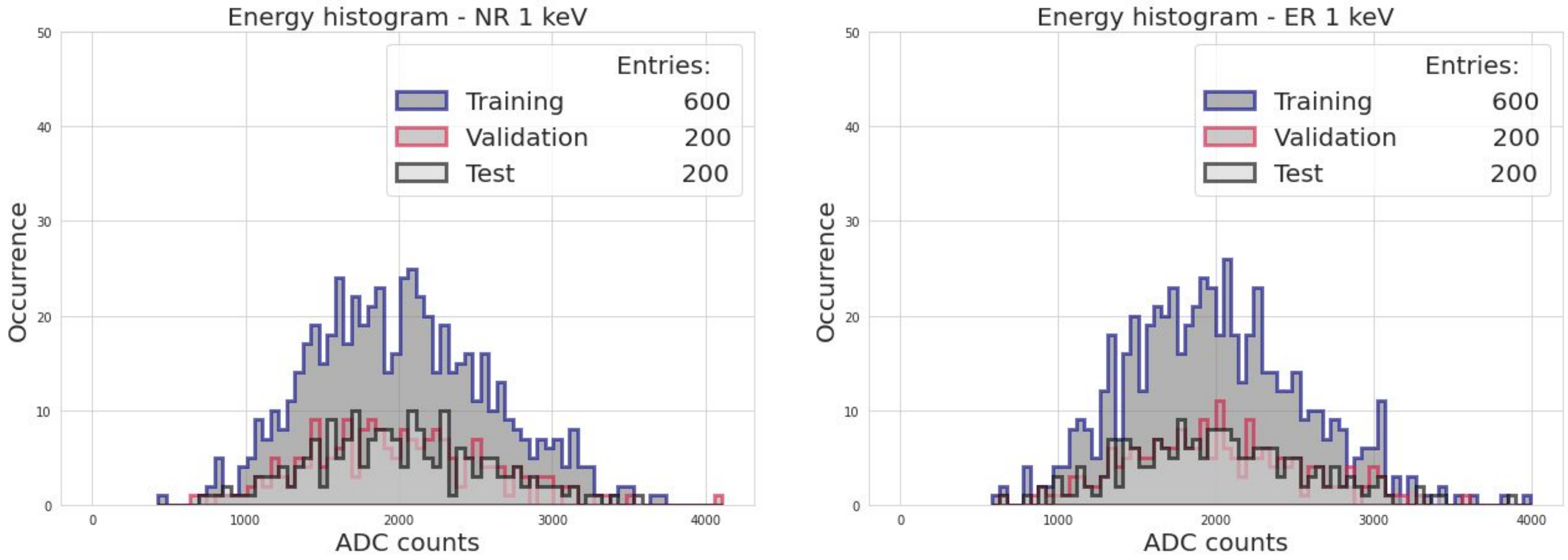
- We started using Pietro’s simulation, which contains:
  - ER events with 1, 3, 6, 10, 30 and 60 keV (1k each)
  - NR events with 1, 3, 6, 10, 30 and 60 keV (1k each)

- The 1 keV simulation was used to create smaller energies simulations (0.25 and 0.5 keV).

Datasets

- **Datasets**
  - **Training:**
    - Noise dataset: 600 images from pedestal runs (Run 4 underground).
    - ER and NR signal simulation: 600 images each containing 0.25-1 keV signals added to pedestal runs (different from noise dataset).
  - **Validation:**
    - Noise dataset: 200 images from pedestal runs.
    - ER and NR signal simulation: 200 images each containing 0.25-1 keV signals.
  - **Test:**
    - Same configuration as validation.

Signal split



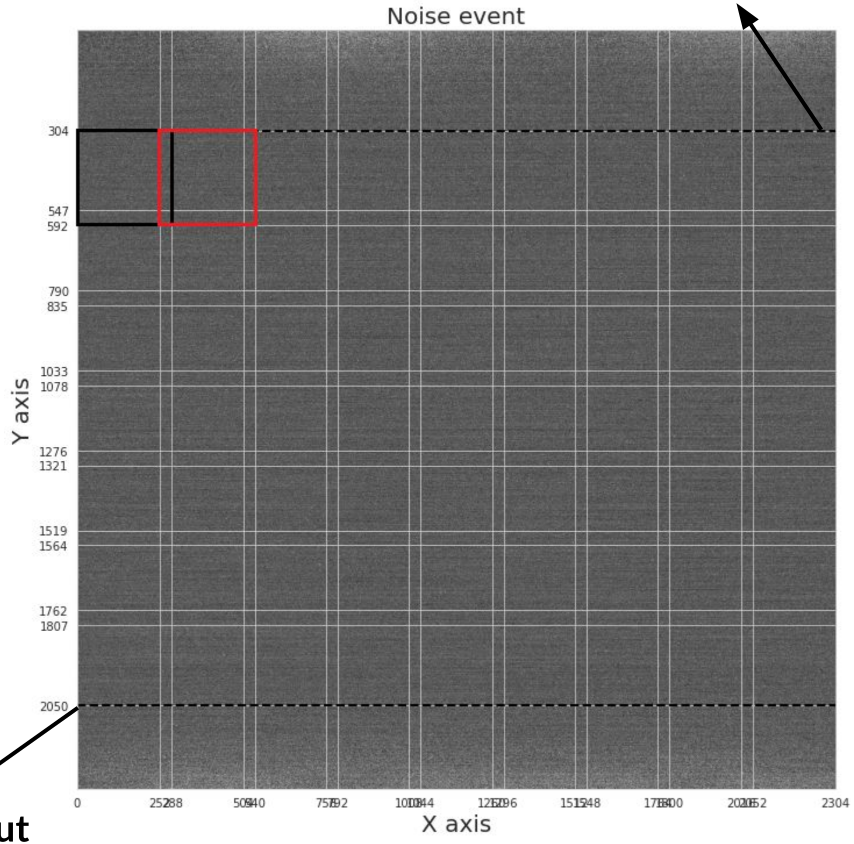
- The signal split was done in a way to maintain the three distributions as similar as possible.

Noise

- A 2304x2304 image is too big for a CNN to handle, meaning that a size reduction is needed.

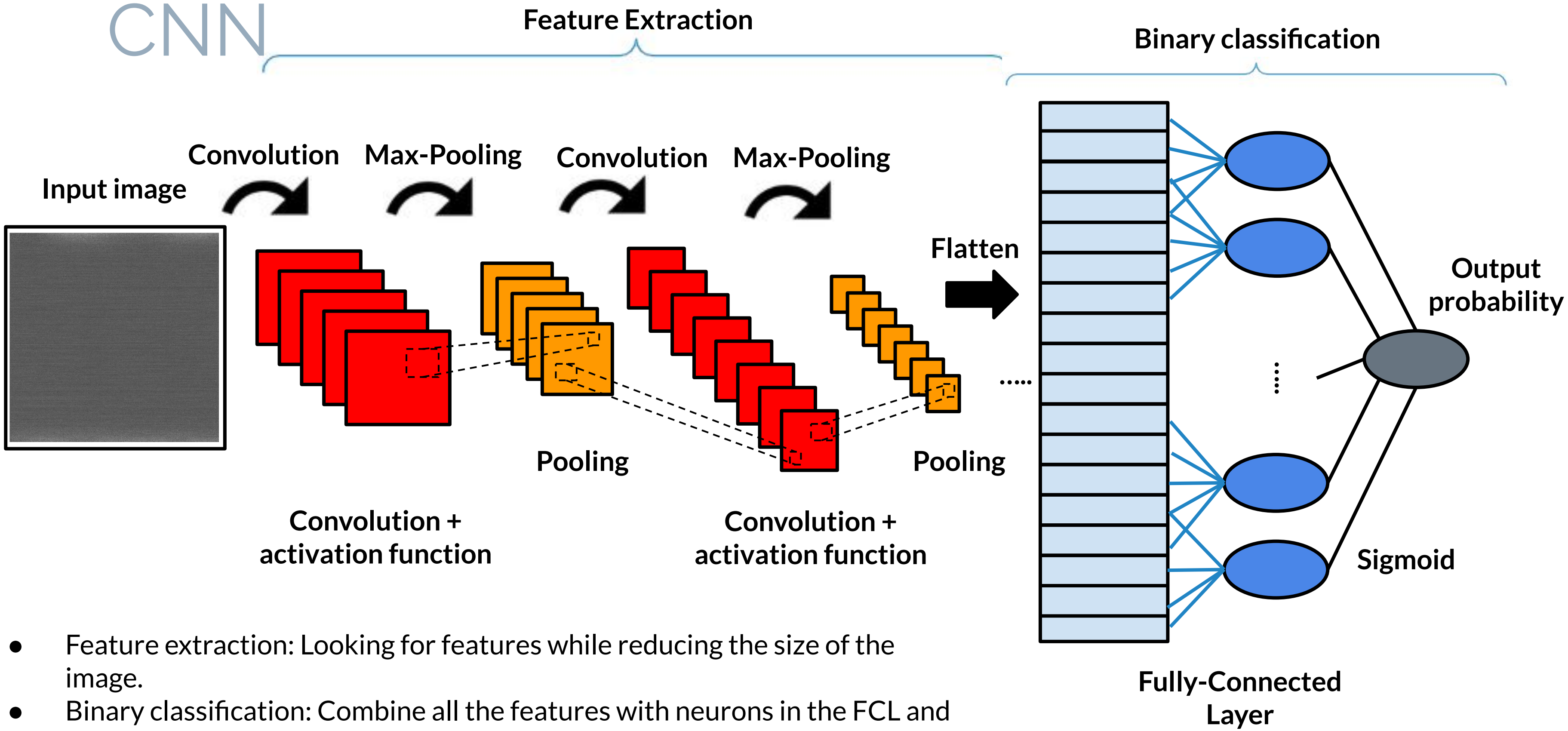
- A possible approach is to send fractionated patches from the original image to the CNN.
    - Tensorflow extract patches function was used.

- The right images were divided into 63 patches with 288x288 pixels each with an overlap region between them.



3.
CNN

CNN



- Feature extraction: Looking for features while reducing the size of the image.
- Binary classification: Combine all the features with neurons in the FCL and classify the input image.

CNN architecture

- The input shape of the CNN limits the number of convolutional and max-pooling layers that can be used.
  - An image with 288 ($2^5 . 3^2$) pixels may use up to **7 (5+2)** layers with regular max-pooling.
  - Custom max-pooling layers may be used to increase the number of layers up to 9.

- Four CNN architectures were selected (number of layers from 6 to 9).
  - The bayesian optimization was used during training.
  - The approach is to select a range of possible hyperparameters (number of filters in each conv layer, neurons on dense layer, etc) and the method will find the optimal values.

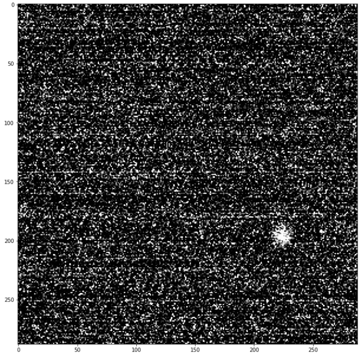
CNN training

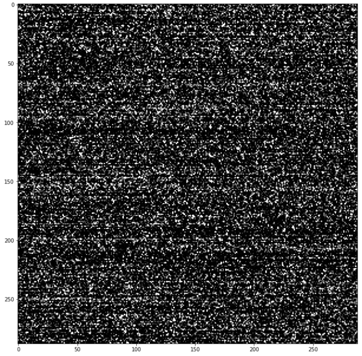
- Both ER and NR were used together during the CNN training.
  - The signal was randomly rotated and placed in a position among the noise.

- 4800 images with 288x288 pixels were used on CNN training and 1600 on validation.
  - Every signal from the split was used twice.
  - The noise patch used was always different.

- The best result was achieved by using 0.5 keV signals on training.
  - 0.25 keVs signals generally led to overfitting.

Signal image



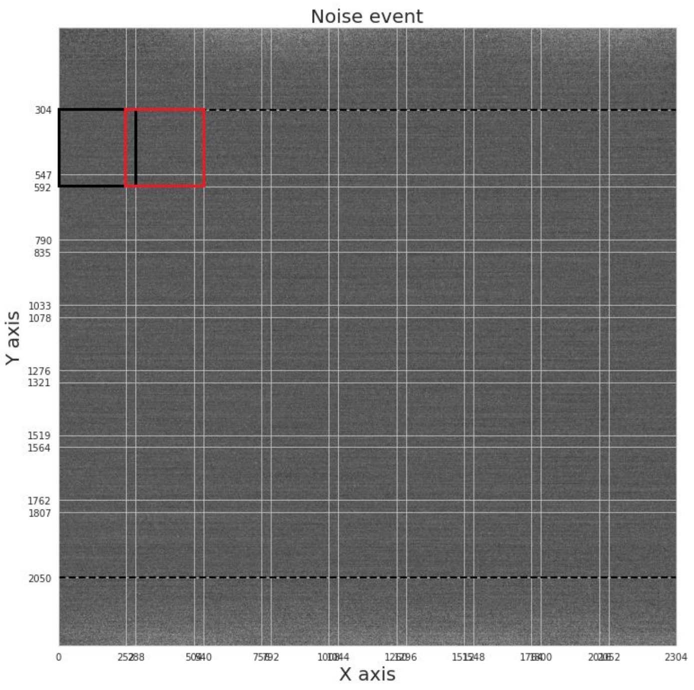


Noise image

CNN test

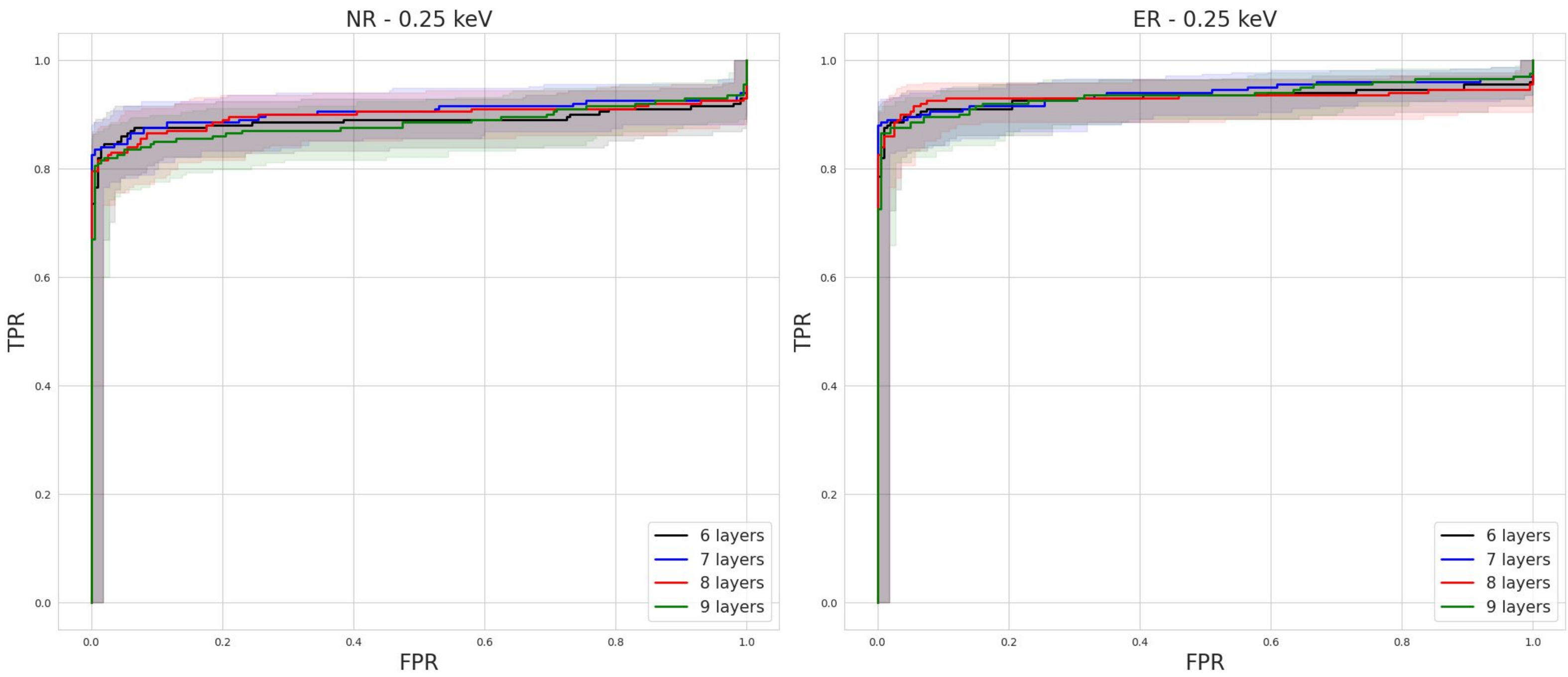
- Since the actual image has 2304x2304 pixels, the test should be performed in way to use all that information.
  - The highest prediction of the CNN on each one of the 63 patches from noise images is stored.
  - The highest prediction on the CNN on each one of the patches that contain an information of the signal is stored.

- This procedure was used on the 400 images separated for test.
  - ER and NR were tested separately to see the CNN performance.
  - 0.25 and 0.5 keV signals were used for test.



4.
Results

CNN ROC 0.25 keV



CNN ROC 0.25 keV

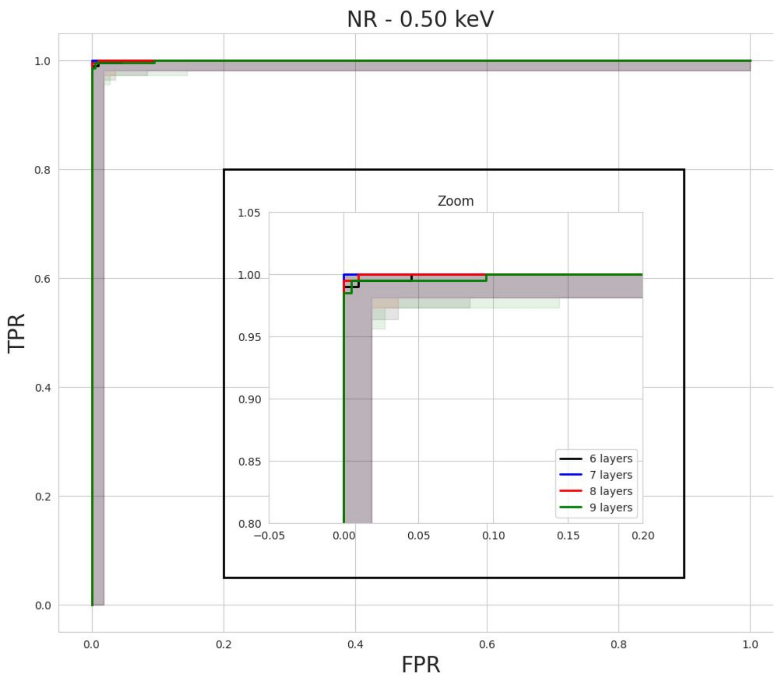
- All four architectures show close results based on ROC curves.
  - The CNN of last presentation had AUC of 0.866 and 0.912 for NR and ER respectively.

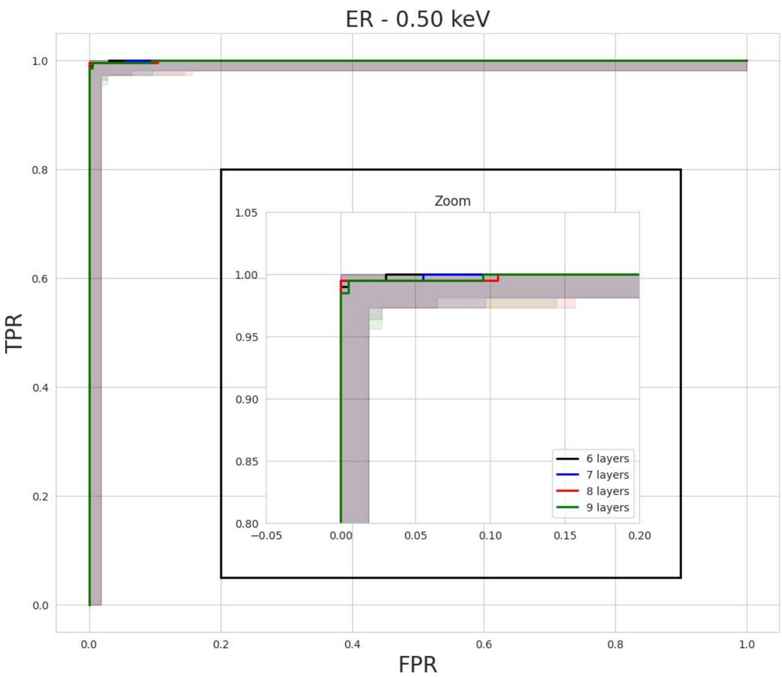
- ER test dataset was slightly easier for all filters and CNNs used (probably due to dataset split).

Area Under Curve for 0.25 keV ROCs

Architecture	NR	ER
6 layers	$0.8888\pm^{0.0423}_{0.0714}$	$0.9308\pm^{0.0322}_{0.0650}$
7 layers	$0.9036\pm^{0.0393}_{0.0691}$	$0.9389\pm^{0.0298}_{0.0631}$
8 layers	$0.8974\pm^{0.0405}_{0.0708}$	$0.9312\pm^{0.0315}_{0.0637}$
9 layers	$0.8836\pm^{0.0444}_{0.0742}$	$0.9348\pm^{0.0317}_{0.0654}$

CNN ROC 0.5 keV



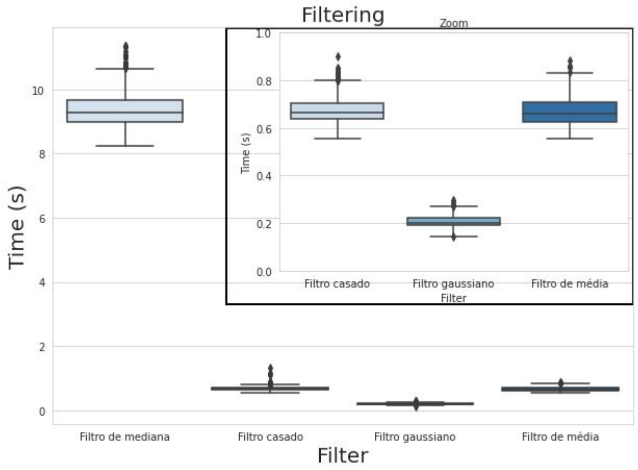


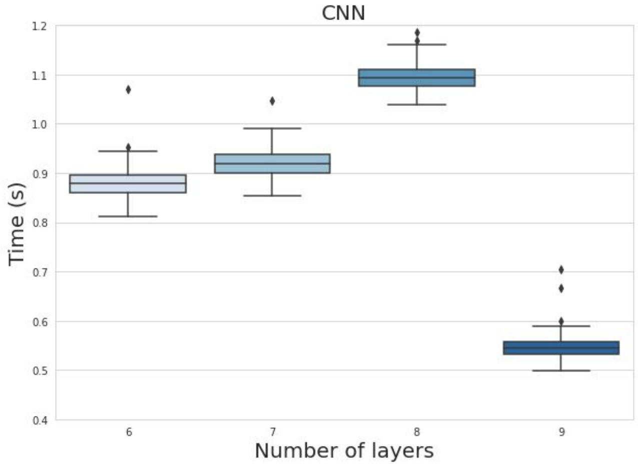
Almost perfect ROC curves for 0.5 keV

Time analysis CPU

- Gaussian filter is by far the fastest using cloud CPU (0.2 seconds per image).

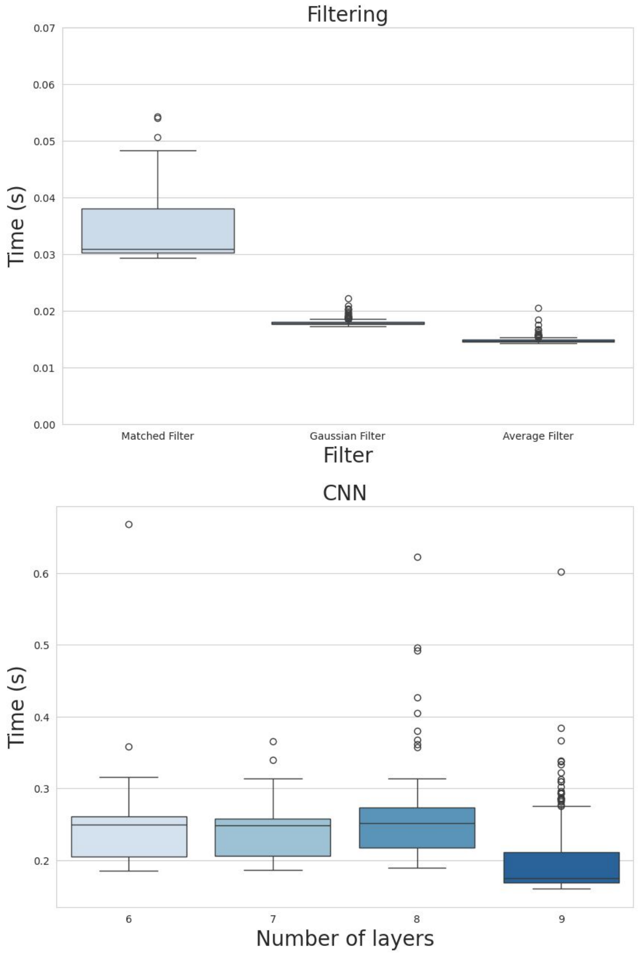
- The fastest CNN on CPU is the one with 9 convolutional layers (0.55 seconds per image).
    - Overall number of operations is smaller than others.



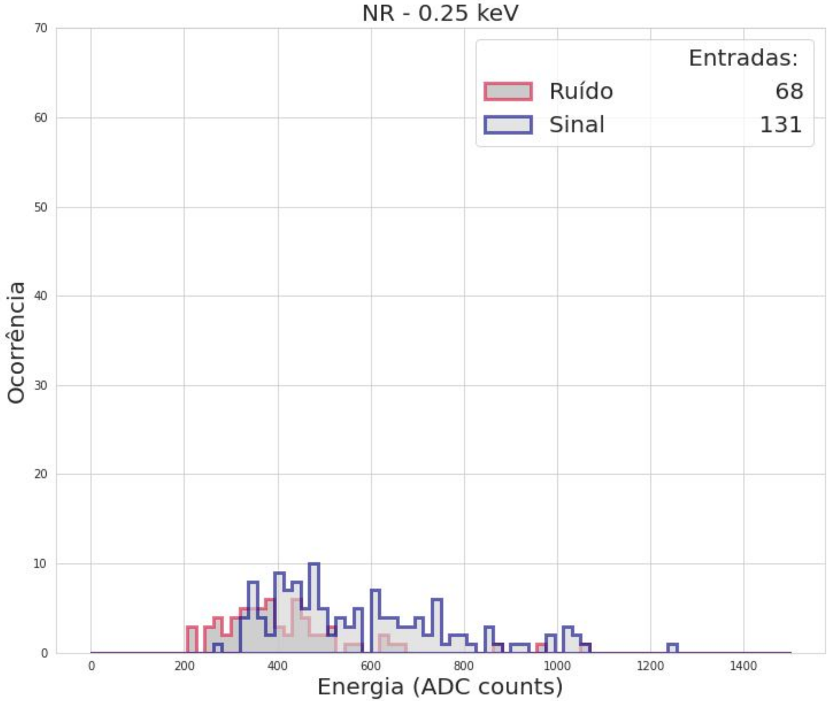


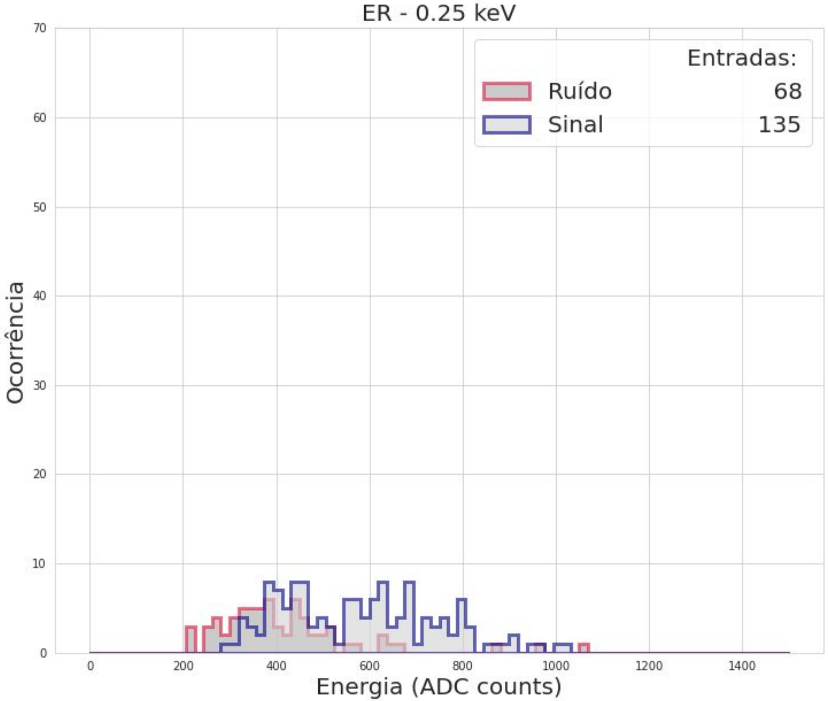
Time analysis GPU

- Gaussian filter needs 0.02 seconds per image with Tesla T4 GPU from google colab.
- All CNNs have similar performance with GPU (0.2 seconds per image).



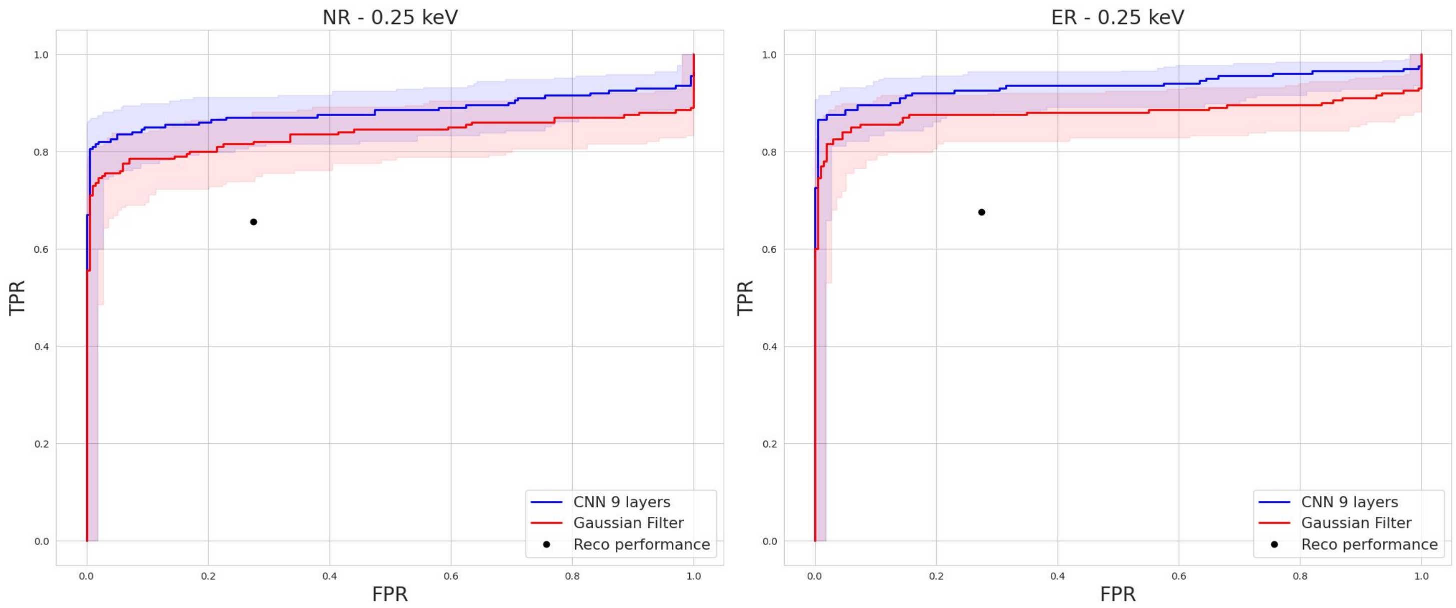
Reconstruction 0.25 keV





The reconstruction found noise clusters on 55 events (27.5% false alarm) and detected ~135 signals (67.5% signal detection).

CNN ROC 0.25 keV vs Reco



- Both methods (best filter and best CNN) detect all events clustered by the reconstruction.
- A signal detection performance of 80% would represent a false alarm close to 0% for the CNN.

4.
Conclusion

Conclusion

- It was possible to improve even more the CNN performance (~0.03 AUC increase) compared to the last one.

- It was clear that an improvement in signal detection has to be compensated with processing time (CNN vs filtering comparison).

- The GPU was able to speed up the CNNs and gaussian filter to 0.2 and 0.02 s per image respectively.

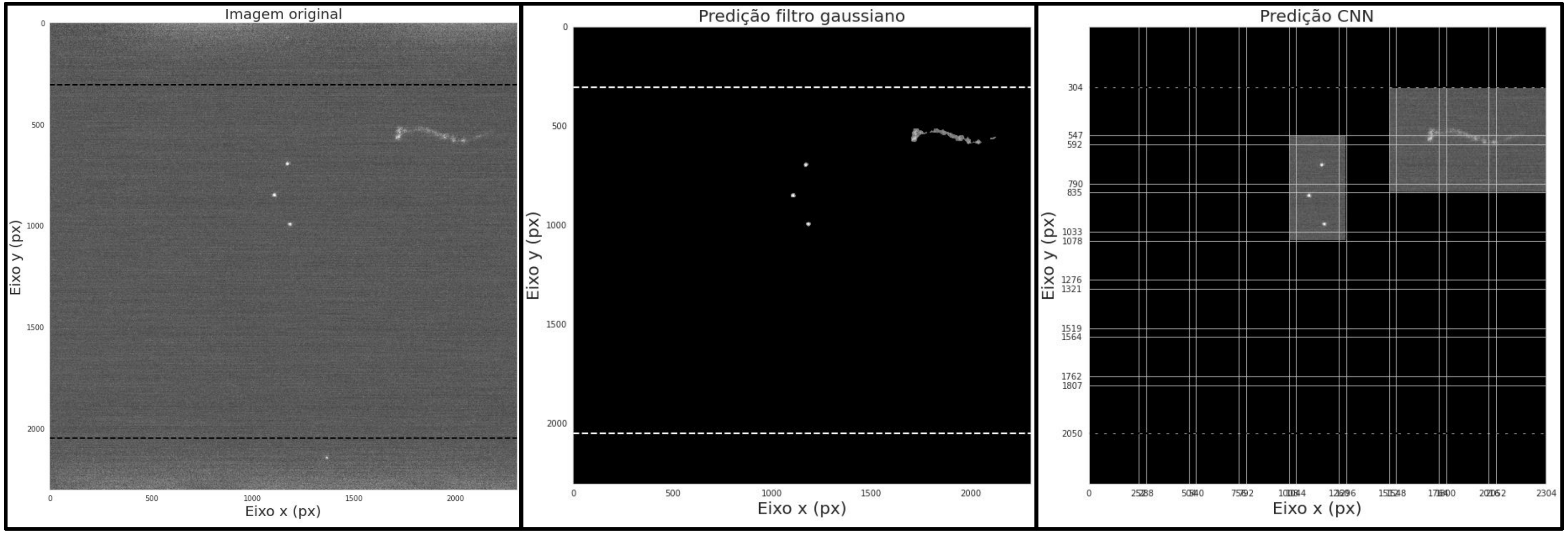
Next steps

- Study methods to simplify a trained CNN model: Bit reduction, weight combination, pruning and vectorization.
  - Reduce processing time without harming performance.

- Write a paper about these results.
  - Trigger proposal based on filters and CNN.

Thanks!

Trigger on real data NRAD (extra)



As regiões onde há alguma informação de sinal foram corretamente identificadas pelos algoritmos de trigger.