Deep Residual Networks for GW Astronomy

Aristotle University of Thessaloniki

Paraskevi Nousi

Alexandra E. Koloniari, Nikolaos Passalis, Panagiotis Iosif,

Nikolaos Stergioulas, Anastasios Tefas

Overview

Motivation and Goal

- Participation in the first Machine Learning Gravitational-Wave Search Mock Data Challenge
- Four datasets of increasing difficulty: from Gaussian noise to real noise from the O3a observing run

Our contribution

- ✗ Gaussian noise is not realistic
- → We focus on the real noise scenario
- ✓ Using deep residual networks, adaptive input normalization to account for non-stationary inputs, batched whitening process, and input augmentations
- → Leading algorithm among other ML-based submissions in real noise

MLGWSC

MLGWSC-1: The first Machine Learning Gravitational-Wave Search Mock Data Challenge

Marlin B. Schäfer[1,2], Ondřej Zelenka[3,4], Alexander H. Nitz[1,2], He Wang[5], Shichao Wu[1,2], Zong-Kuan Guo[5], Zhoujian Cao[6], Zhixiang Ren[7], Paraskevi Nousi[8], Nikolaos Stergioulas[9], Panagiotis Iosif[10,9], Alexandra E. Koloniari[9], Anastasios Tefas[8], Nikolaos Passalis[8], Francesco Salemi[11,12], Gabriele Vedovato[13], Sergey Klimenko[14], Tanmaya Mishra[14], Bernd Brügmann[3,4], Elena Cuoco[15,16,17], E. A. Huerta[18,19], Chris Messenger[20], Frank Ohme[1,2]

[1] *Max-Planck-Institut für Gravitationsphysik, Albert-Einstein-Institut, D-30167 Hannover, Germany*
[2] *Leibniz Universität Hannover, D-30167 Hannover, Germany*
[3] *Friedrich-Schiller-Universität Jena, D-07743 Jena, Germany*
[4] *Michael Stifel Center Jena, D-07743 Jena, Germany*
[5] *CAS Key Laboratory of Theoretical Physics, Institute of Theoretical Physics, Chinese Academy of Sciences, Beijing 100190, China*
[6] *Department of Astronomy, Beijing Normal University, Beijing 100875, China*
[7] *Peng Cheng Laboratory, Shenzhen, 518055, China*
[8] *Department of Informatics, Aristotle University of Thessaloniki, GR-54124 Thesssaloniki, Greece*
[9] *Department of Physics, Aristotle University of Thessaloniki, GR-54124 Thessaloniki, Greece*
[10] *GSI Helmholtz Center for Heavy Ion Research, Planckstraße 1, 64291 Darmstadt, Germany*
[11] *Università di Trento, Dipartimento di Fisica, I-38123 Povo, Trento, Italy*
[12] *INFN, Trento Institute for Fundamental Physics and Applications, I-38123 Povo, Trento, Italy*
[13] *INFN, Sezione di Padova, I-35131 Padova, Italy*
[14] *Department of Physics, University of Florida, PO Box 118440, Gainesville, FL 32611-8440, USA*
[15] *European Gravitational Observatory (EGO), I-56021 Cascina, Pisa, Italy*
[16] *Scuola Normale Superiore, Piazza dei Cavalieri 7, I-56126 Pisa, Italy*
[17] *INFN, Sezione di Pisa, Largo Bruno Pontecorvo, 3, I-56127 Pisa, Italy*
[18] *Data Science and Learning Division, Argonne National Laboratory, Lemont, Illinois 60439, USA*
[19] *Department of Computer Science, University of Chicago, Chicago, Illinois 60637, USA and*
[20] *SUPA, School of Physics and Astronomy, University of Glasgow, Glasgow G12 8QQ, United Kingdom*

https://arxiv.org/abs/2209.11146

MLGWSC-1

- `github.com/gwastro/ml-mock-data-challenge-1`
- objective characterization of ML GW detection algorithms
- allow for easy comparison between different search algorithms
- 4 datasets of increasing complexity
- common code to generate data:
  - background (Gaussian or O3a noise)
  - foreground (same noise + injected waveforms)
  - parameters of injected signals (IMRPhenomXPHM model: non-aligned, spinning BBH waveforms)
- evaluation set generated with the same code, with random seed withheld from participants algorithms

Our submission

- Four basic components:
  1. Large training set
  2. Whitening process implemented as neural layer
  3. Adaptive input normalization
  4. Deep Residual networks

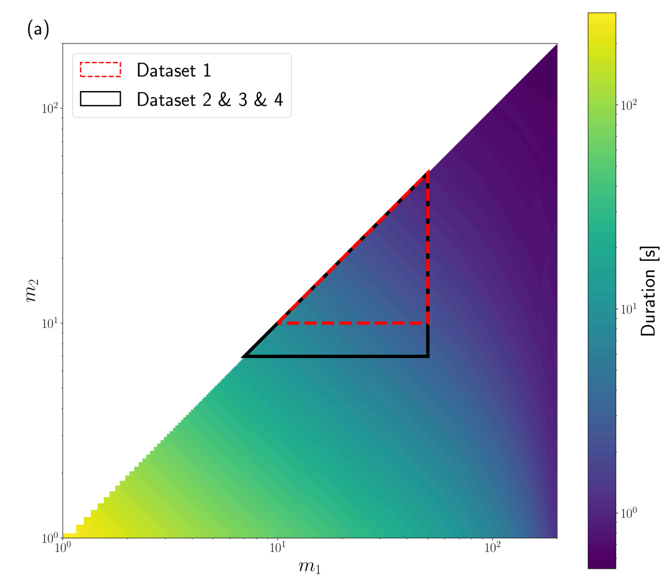
Training set

- Dataset 4 only
- We start by generating about 300k noise segments of 1.25s duration each, over the span of one week
- We generate a large number (about 20k) of waveforms with parameters within the given ranges
- Our training set then consists of about 600k samples, half noise only, half noise + waveform
- Noise and waveforms are combined once, before training

→ A validation set with about 86k samples is generated in a similar way, from 1 day of data

→ Our test set is generated using the provided code (1 month)

Training Set

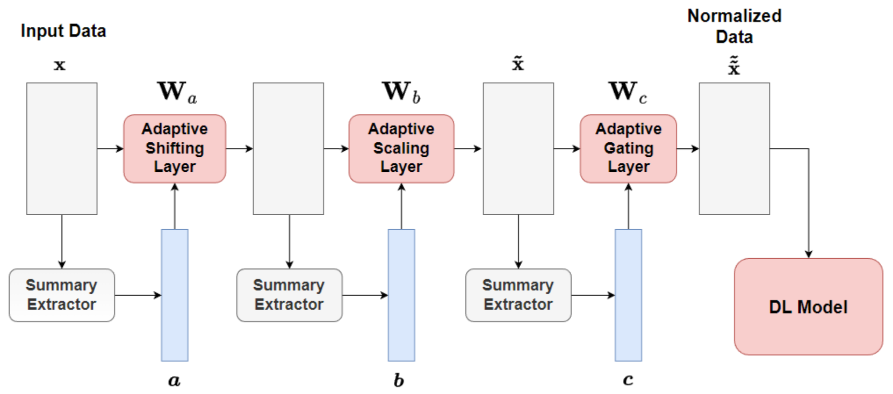


Whitening

- We found whitening to be an important preprocessing step
- But PyCBC's method is CPU-based, and processes each sample on its own

→ So we implement whitening using PyTorch functions
  - Welch method, following PyCBC's methods including inverse spectrum truncation
  - Operations performed on GPU directly, no need for CPU utilization or CPU to GPU data transfer
  - Some operations can be performed as batched

Deep Adaptive Input Normalization



Passalis et al. https://arxiv.org/abs/1902.07892

Deep Residual Networks

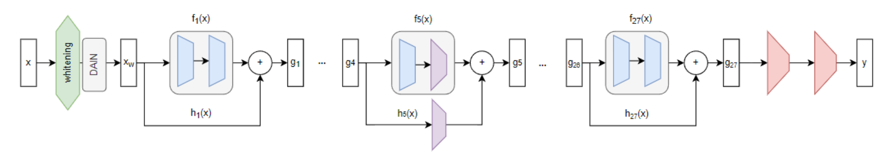
- Originally proposed for 2D image analysis (recognition, detection, etc.) `https://arxiv.org/abs/1512.03385`
- The residual connections allow for effectively training deeper networks, alleviating the gradient vanishing problem
- Let $\mathbf{x} \in \mathcal{R}^{2\times 2048}$ be the input of a residual block, then the output is given as:

$$\mathbf{g} = f(\mathbf{x}) + h(\mathbf{x})$$

  where f is a block of two convolutional layers followed by ReLU activation functions and BN layers, and h is the *transfer* function
- h can be either a convolutional layer or an identity mapping (i.e., $h(\mathbf{x}) = \mathbf{x}$), depending on whether or not f changes the dimensionality of its input

Deep Residual Networks

- We tried various network depths, ranging from 10 to 54, the latter being our final model
- This network takes as input the whitened, normalized 1s samples and outputs 2 values corresponding to the two possible outcomes: noise only vs. noise + waveform



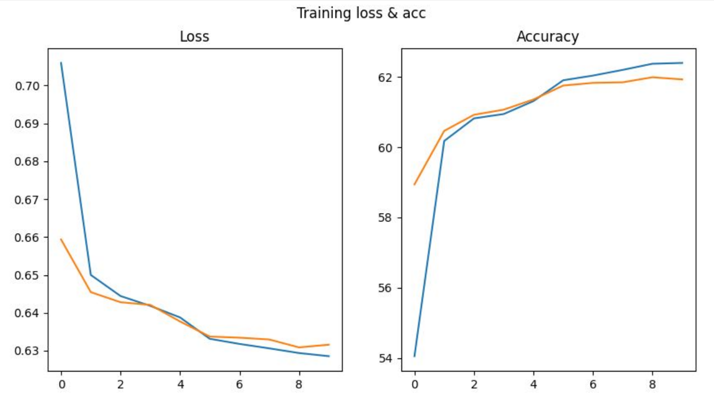
Deep Residual Networks

residual blocks	filters	strided	input D
4	8		2×2048
1	16	✓	8×2048
2	16		16×1024
1	32	✓	16×1024
2	32		32×512
1	64	✓	32×512
2	64		64×256
1	64	✓	64×256
2	64		64×128
1	64	✓	64×128
2	64		64×64
5	32		64×64
3	16		32×64

Training

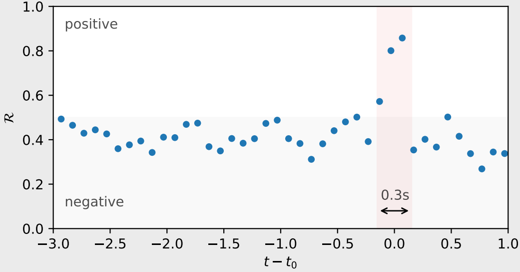
- 4.25s of noise are taken around each input sample to compute the PSDs for each channel/detector
- Each sample is whitened using its computed PSDs, then cropped to 1s
- If positive (i.e., noise + waveform), the sample is cropped around the reference time of coalescence, such that if falls within the 0.5s to 0.7s mark
- The entire network is optimized using a regularized cross entropy objective function
- Final validation accuracy is around 61%

Training


Training loss & acc
Loss
Accuracy
0.70
0.69
0.68
0.67
0.66
0.65
0.64
0.63
0
2
4
6
8
62
60
58
56
54
0
2
4
6
8

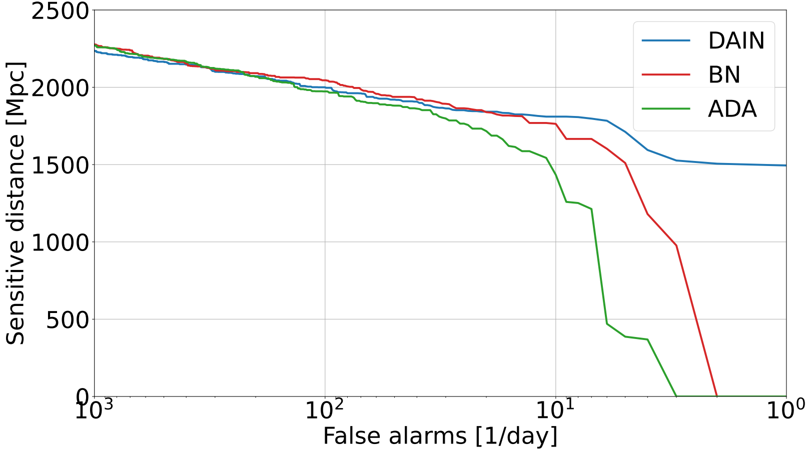
Deployment

- The test set is split in 4.25s long segments with a step size of 3.1s
- The PSDs for each segment are computed
- Each segment is split into 31 samples, using an internal step size of 0.1s, which are processed as a batch
- This results in triggers as shown below, which are then clustered in time



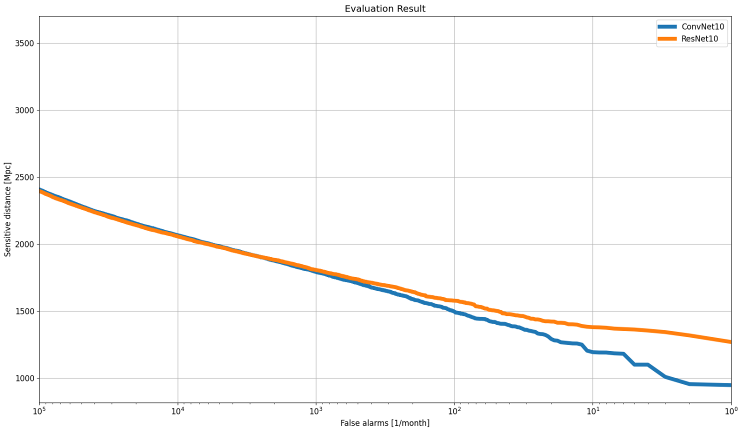
Experiments

Effect of DAIN



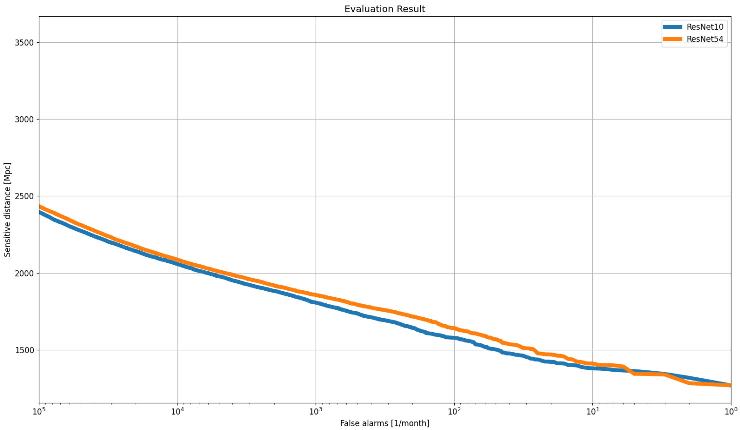
Experiments

Effect of residual connections

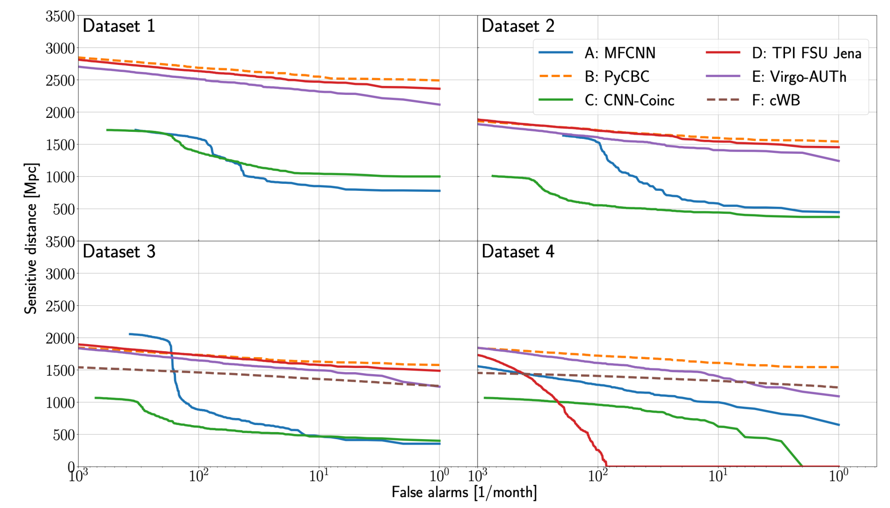


Experiments

Effect of depth



Final Results



Future Directions

We've been working on:

- Larger training set - denser parameter space
- Wider networks
- Curriculum Learning

Further improving our results!

Thank you!

Questions?