Why using artificial intelligence in the search for gravitational waves?

Elena Cuoco, EGO and SNS
www.elenacuoco.com
Twitter: @elenacuoco
What are Gravitational Waves (GWs)?

General Relativity (1915)

\[ G = \frac{8 \pi G}{c^4} T \]

Gravitational Waves (1916)
How we detected GWs?
Astrophysical Gravitational Wave signals

An example signal from an inspiral gravitational wave source. [Image: A. Stuver/LIGO]

An artist's impression of two stars orbiting each other and progressing (from left to right) to merger with resulting gravitational waves. [Image: NASA/CXC/GSFC/T.Strohmayer]
International Collaboration
GW150914 and GW170817

First Detection of Gravitational Waves from 2 colliding Neutron Stars
~1.5–2 Solar mass each

First Detection of Gravitational Waves!
2 colliding Black Holes
~30Solar mass each

NGC 4993 GRB170817A
Hubble telescope

Artist’s illustration of the merger of two neutron stars, producing a short gamma-ray burst. [NSF/LIGO/Sonoma State University/A. Simionescu]
Why Machine Learning in Gravitational Wave research
Outline

Machine learning for Gravitational Wave Data analysis

- Glitches classification
  - Image-based
  - Wavelet-based

- New ideas and possible collaborations in COST action framework

- Noise Removal

- Real time analysis (ongoing work)
LIGO/Virgo data are time series sequences... noisy time series with low amplitude GW signal buried in
Known GW signals
Compact coalescing binaries has known theoretical waveforms
Optimal filter: Matched filter
Too many templates to test

Unknown GW signals
Core collapse supernovae
No Optimal filter
Parameters estimation

Noise
Moving lines
Broad band noise
Glitch noise
“Pattern recognition” by visual inspection
Example of GW signals in Time-Frequency plots
Example of Glitch signals

https://www.zooniverse.org/projects/zooniverse/gravity-spy
Example of other noise signals

I. Fiori courtesy
Numbers about data

Data Stream Flux
• 50MB/s

Data on disk
• 1-3PB

Number of events
• 1/week
• 1/day?

Number of glitches
• 1/sec
• 0.1/sec?

Should be analysed in less than 1min
How Machine Learning can help

Data conditioning
- Non linear noise coupling
- Use Neural Network to learn noise
- Use Neural Network to remove noise

Signal Detection/Classification/PE
- A lot of fake signals due to noise
- Fast alert system
- Manage parameter estimation
What is going in the ML LIGO/Virgo group

136 LIGO/Virgo members

30 active projects
Example of interesting works

- Labelling glitches: Gravity Spy
- Noise Removal
  
  Non-linear and non-stationary noise subtraction with Deep Learning

S. Coughlin courtesy

G. Vajente courtesy
- Deep learning procedure requiring only the raw data time series as input with minimal signal pre-processing.
- Performance similar to Optimal Wiener Filter
Glitches Classification Strategy

Training Process

Auxiliary channel -> Glitches detection & features extraction
Time series data -> Glitches detection & features extraction

No Label
Machine Learning Unsupervised Algorithm

Label
Machine Learning Supervised Algorithm

New data -> Trained pipeline

Predictions
Glitches classification efforts in LIGO/Virgo Community

- Gravity Spy (M. Zevin, S. Coughlin, J. R. Smith, A. Lundgren, D. Macleod, V. Kalogera)
- WDF-ML (E. Cuoco, A. Torres)
- WDFX (E. Cuoco, M. Razzano, A. Utina)
- PCAT (M. Cavaglià, D. Trifirò)
- Karoo GP (K. Staats, M. Cavaglià)
- Wavelet-DBNN (N. Mukund S. Abraham S. Mitra et al)
- ImageGlitch CNN (M. Razzano, E. Cuoco)
- Low latency transient detection and classification (I. Pinto, V. Pierro, L. Troiano, E. Mejuto-Villa, V. Matta, P. Addesso)
- Deep Transfer Learning (Daniel George, Hongyu Shen, E.A. Huerta)
- New ranking statistic for gstlal (K. Kim, T.G.F. Li, R.K.-L. Lo, S. Sachdev, R.S.H. Yuen)
Images-based glitch classification

Deep learning with CNN

Massimiliano Razzano and Elena Cuoco
2018 Class. Quantum Grav. 35 095016
Deep learning for Glitch Classification

- Many approaches to data: we choose image classification of time frequency images.
- The architecture is based on Convolutional deep Neural Networks (CNNs).
- CNNs are more complex than simple NNs but are optimized to catch features in images, so they are the best choice for image classification.
Pipeline structure

Input GW data

- Image processing
- Time series whitening
- Image creation from time series (FFT spectrograms)
- Image equalization & contrast enhancement

Classification

- A probability for each class, take the max
- Add a NOISE class to crosscheck glitch detection

Network layout

- Tested various networks, including a 4-block layers

Run on GPU Nvidia GeForce GTX 780

- 2.8k cores, 3 Gb RAM
- Developed in Python + CUDA-optimized libraries
To test the pipeline, we prepared ad-hoc simulations. Add 6 different classes of glitch shapes.

Simulate colored noise using public H1 sensitivity curve.
Simulated signal families

Waveform
- Gaussian
- Sine-Gaussian
- Ring-Down
- Chirp-like
- Scattered-like
- Whistle-like
- NOISE (random)

To show the glitch time-series here we don’t show the noise contribution

Razzano M., Cuoco E. CQG-104381.R3
Simulated time series with 8kHz sampling rate

Glitches distributed with Poisson statistics $m=0.5$ Hz

2000 glitches per each family

Glitch parameters are varied randomly to achieve various shapes and Signal-To-Noise ratio
Building the images

Spectrogram for each image

2-seconds time window to highlight features in long glitches

Data is whitened

Optional contrast stretch
Datasets of 14000 images
Training/validation/test → 70/15/15
Image size 241px x 513px
Reduced the images by a factor 0.55 due to memory constraints
Use validation set to tune hyperparameters
On our hardware, training time ~8 hrs for ~100 epochs
When training is done, classification requires ~1 ms/image (on our configuration)
We compared classification performances with simpler architectures:

<table>
<thead>
<tr>
<th>Metric</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1 score</th>
<th>Log loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM</td>
<td>0.971</td>
<td>0.972</td>
<td>0.971</td>
<td>0.971</td>
<td>0.08</td>
</tr>
<tr>
<td>Shallow CNN</td>
<td>0.986</td>
<td>0.986</td>
<td>0.986</td>
<td>0.986</td>
<td>0.04</td>
</tr>
<tr>
<td>1 CNN block</td>
<td>0.991</td>
<td>0.991</td>
<td>0.991</td>
<td>0.991</td>
<td>0.02</td>
</tr>
<tr>
<td>3 CNN blocks</td>
<td>0.998</td>
<td>0.998</td>
<td>0.998</td>
<td>0.998</td>
<td>0.008</td>
</tr>
</tbody>
</table>

- Linear Support Vector Machine
- CNN with 1 hidden layer
- CNN with one block (2 CNNs+Pooling&Dropout)
- Deep 4-blocks CNNs
Deep CNN better at distinguishing similar morphologies
Example of classification results

Some cases of more glitches in the time window, always identify the right class

100% Sine-Gaussian
Wavelet Detection Filter (WDF) workflow

\[ x_i = h_i + n_i, \quad i = 0, 1, \ldots, N - 1, \]

\[ W f(a, b) = \langle f, \psi_{a,b} \rangle = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{b}} \psi^* \left( \frac{t-a}{b} \right) \, dt. \]

\[ \hat{h}_i = W^{-1}(t[W(x_i)]). \]
Wavelet Detection Filter

- Wavelet transform in the selected window size
- Retain only coefficients above a fixed threshold (Donoho–Johnston denoise method)
- Create a metric for the energy using the selected coefficients and give back the trigger with all the wavelet coefficients.
- In the wavelet plane, select the highest values and closest coefficients to build the event
- Put to zero all the other coefficients
- Inverse wavelet transform
- Estimate mean and max frequency and SNR max of the cleaned event

Gps, duration, SNR, SNR@max, freq_mean, freq@max, wavelet type triggered + corresponding wavelets coefficients.
eXtreme Gradient Boosting

- [https://github.com/dmlc/xgboost](https://github.com/dmlc/xgboost)
- Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016
- XGBoost originates from research project at University of Washington, see also the Project Page at UW.

Tree Ensemble

\[ y_n = \sum_{k=1}^{K} f_k (x_n) \]
Wavelet Detection Filter and XGBoost (WDFX)

**Step 1: WDF**
- Data
- Whitening in Time Domain
- Wavelet Transform
- Denoising
  - SNR Estimation
  - SNR > Threshold
  - Whitening Parameter

**Step 2: Machine Learning**
- Supervised classification
  - Glitches Families
  - Unsupervised Classification
  - Feature Selection (PCA, ...)
  - Wavelets Coefficients
  - Save Event and Parameters at Peak
WDF results

- Detected 97% of injected signals (some with SNR=1)
- False alarm rate: 10% for a time window shift of 1 sec
- Good parameters estimation
Parameters estimation

Time difference distribution

SNR difference distribution

Frequency difference distribution
Machine learning

\[ L = -\frac{1}{N} \sum_{i=1}^{N} ((y_i \log(p_i) + (1 - y_i)(\log(1 - p_i))) + \Omega \]

Train/validation/test set: 70/15/15

<table>
<thead>
<tr>
<th>task</th>
<th>Classes</th>
<th>Learning-rate</th>
<th>Max_depth</th>
<th>estimators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>2</td>
<td>0.01</td>
<td>7</td>
<td>5000</td>
</tr>
<tr>
<td>Multi-label</td>
<td>7</td>
<td>0.01</td>
<td>10</td>
<td>6000</td>
</tr>
</tbody>
</table>
WDFX: Binary Classification Results

Overall accuracy >90%

Normalized confusion matrix

- Noise: 0.98, 0.02
- Chirp: 0.19, 0.81

Injected  Recovered
WDFX Results: Multi-Label Classification

Overall accuracy >80%
release an end to end framework for the glitches identification, classification and archiving ML classification schemes for GW glitches.

To evaluate possible HPC solutions for DL pipelines for online glitch classification.

LAPP, Trust-IT Services company, EGO
Noise removal trough Deep learning

Gabriele Vajente¹, Michael Coughlin¹, Rich Ormiston²
¹LIGO Laboratory Caltech
²University of Minnesota Twin Cities

Same work for Virgo.
A. Iess et al. with the help of Gabriele
Recurrent Neural Networks for noise cancellation

A. Iess (PhD student), G. Vajente, E. Cuoco, V. Fafone

Gaussian background (from Ad Virgo sensitivity curve)

Beam Jitter Noise modulated by suspension transfer function (simulated)

A. Iess courtesy
3 WITNESS CHANNELS (INPUTS)

1. Beam Jitter
2. Suspension motion
3. Seismic modulation

RECURRENT NEURAL NETWORK

h PREDICTION (OUTPUT)
3 WITNESS CHANNELS (INPUTS)

1. Beam Jitter
2. Suspension motion
3. Seismic modulation

INPUTS

\[ X_1(t) \]
\[ X_2(t) \]
\[ X_3(t) \]

RECURRENT LAYERS

BILINEAR LAYER

FULLY CONNECTED LAYERS

\[ Y_P(t) \]

• **PyTorch 0.4.0**
• Number of Epochs: 100
• Optimizer: **ADAM** (ADaptive Moment estimation)
• Initial learning rate: **0.004**
• Learning rate decay: factor 1.0 every 15 epochs
• Loss function: **Squared Error**
• 3 Recurrent layers (Elman/GRU/LSTM) + 1 Bilinear + 4 Fully connected layers
• Batch size: 1 second
- RNNs good for time-series prediction, retain memory through *context units*
- Bilinear layer to model non-linear noise coupling
- Computational load concentrated in training step
- Wiener filters bad for removing non-linear noise
G2net: A network for Gravitational Waves, Geophysics and Machine Learning

Action Chair: E. Cuoco, EGO and SNS
Vice Chair: C. Messenger, Glasgow University
G2net: goals of the ACTION

Facilitate conceiving innovative solutions for the analysis of the data of Gravitational Wave (GW) detectors.

Investigate possible solutions to monitor the low-frequency Newtonian noise through the use of adaptive robots.

Investigate new strategies for the handling/suppression of instrumental and environmental noise using Machine Learning techniques.

Train a new generation of young scientists with broad skills in Machine Learning, GW, Control and Robotics.

Bridge the gap between the disciplines of GW physics, geophysics, computer science and robotics.
G2net more info

https://www.cost.eu/actions/CA17137
Thanks!