Convolutional neural network for directed searches of continuous gravitational waves

Takahiro S. Yamamoto (RESCEU, University of Tokyo)
Collaborating with A. L. Miller (Nikhef), M. Sieniawska, and T. Tanaka (Kyoto U.)

Continuous gravitational waves and neutron stars workshop, June 17-20, 2024 @ Hannover, Germany

Continuous gravitational waves

- Persistent, quasi-monochromatic signals
- Amplitude is very weak
- We need to accumulate O(yr) long data to detect CGWs
- The sensitivity of CGW search is limited by two difficulties.
    - Computational cost.
    - Instrumental lines (narrow spectral artifacts).

Topic of this talk

- This talk is based on two papers:
  - TSY & Tanaka, PRD103, 084049 (2020)
  - TSY, Miller, Sieniawska, and Tanaka, PRD106, 024025 (2022)

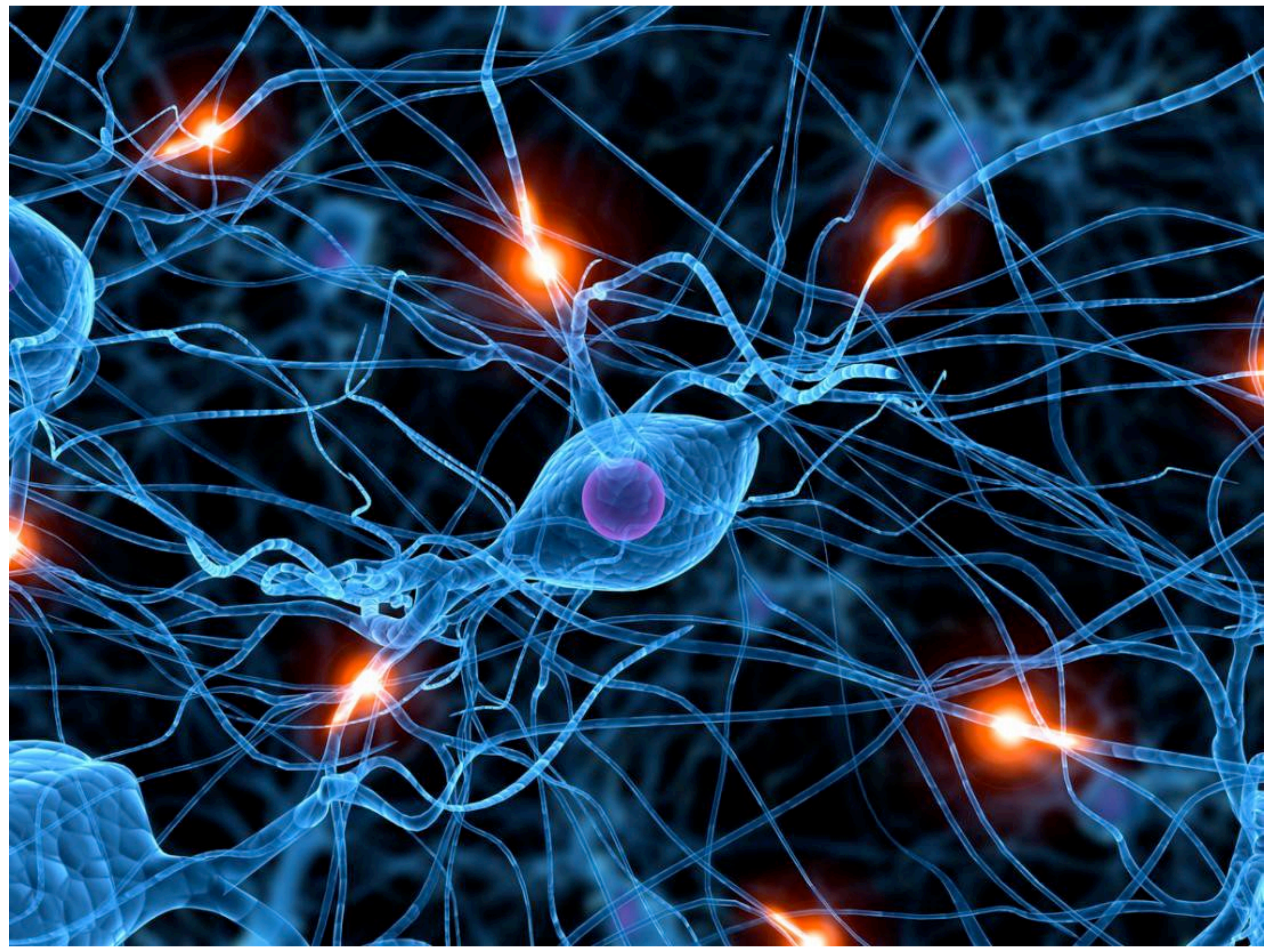
  in which we proposed new pipeline for **all-sky searches** (no information about source) with a convolutional neural network.

- I also talk about our ongoing work, in which we extend our work **for directed searches** (the source direction is known) with modified preprocessing algorithm.

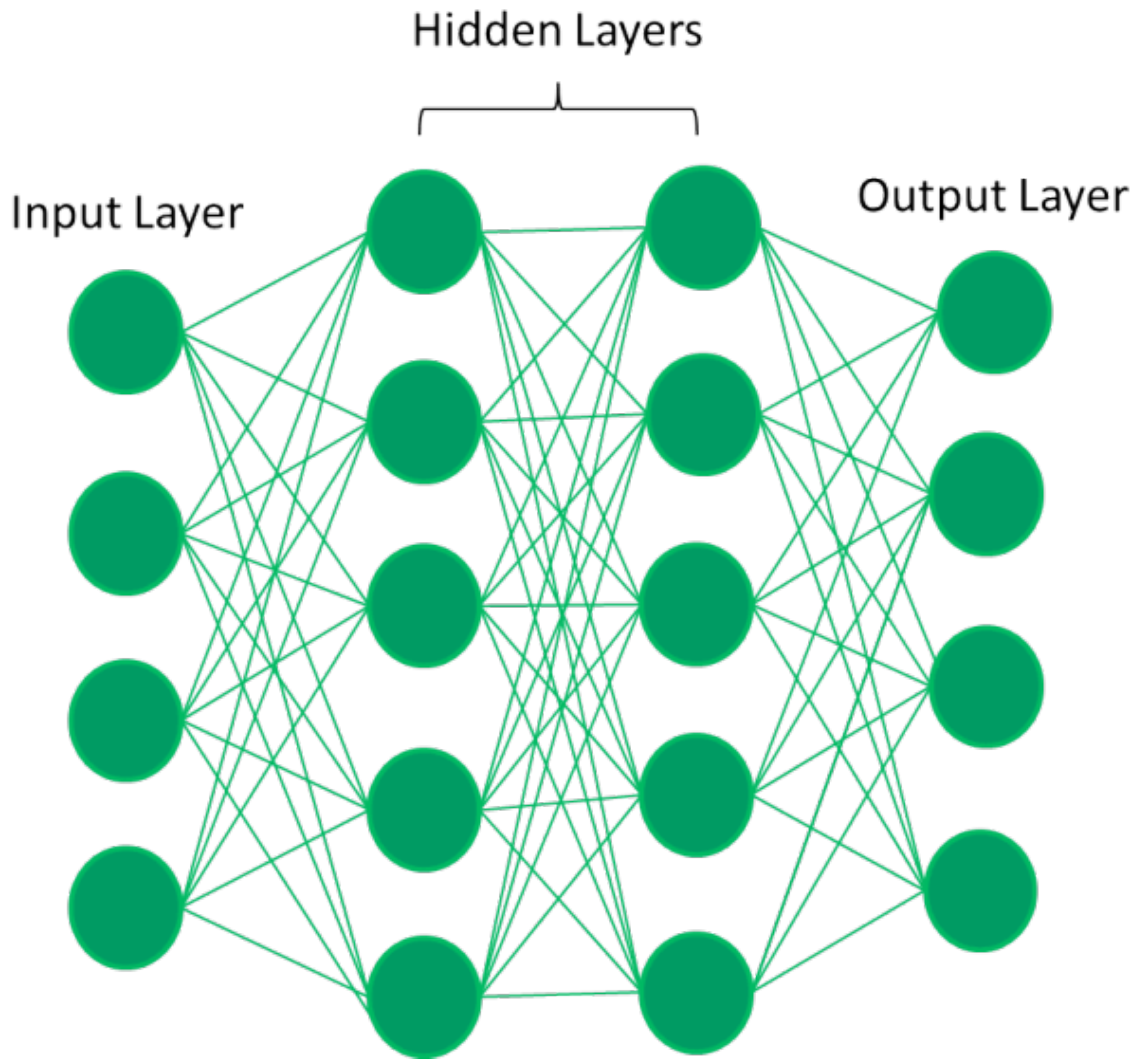
Deep learning

e.g. Goodfellow *et al.*, “Deep learning” as a textbook

- Neural network (NN) is inspired by the structure of a human brain, mimicking the way that biological neurons signal to one another.
- Highly non-linear function controlled by many parameters.
- NN’s parameters are optimized using a training dataset before we apply NN to test data or a real event.



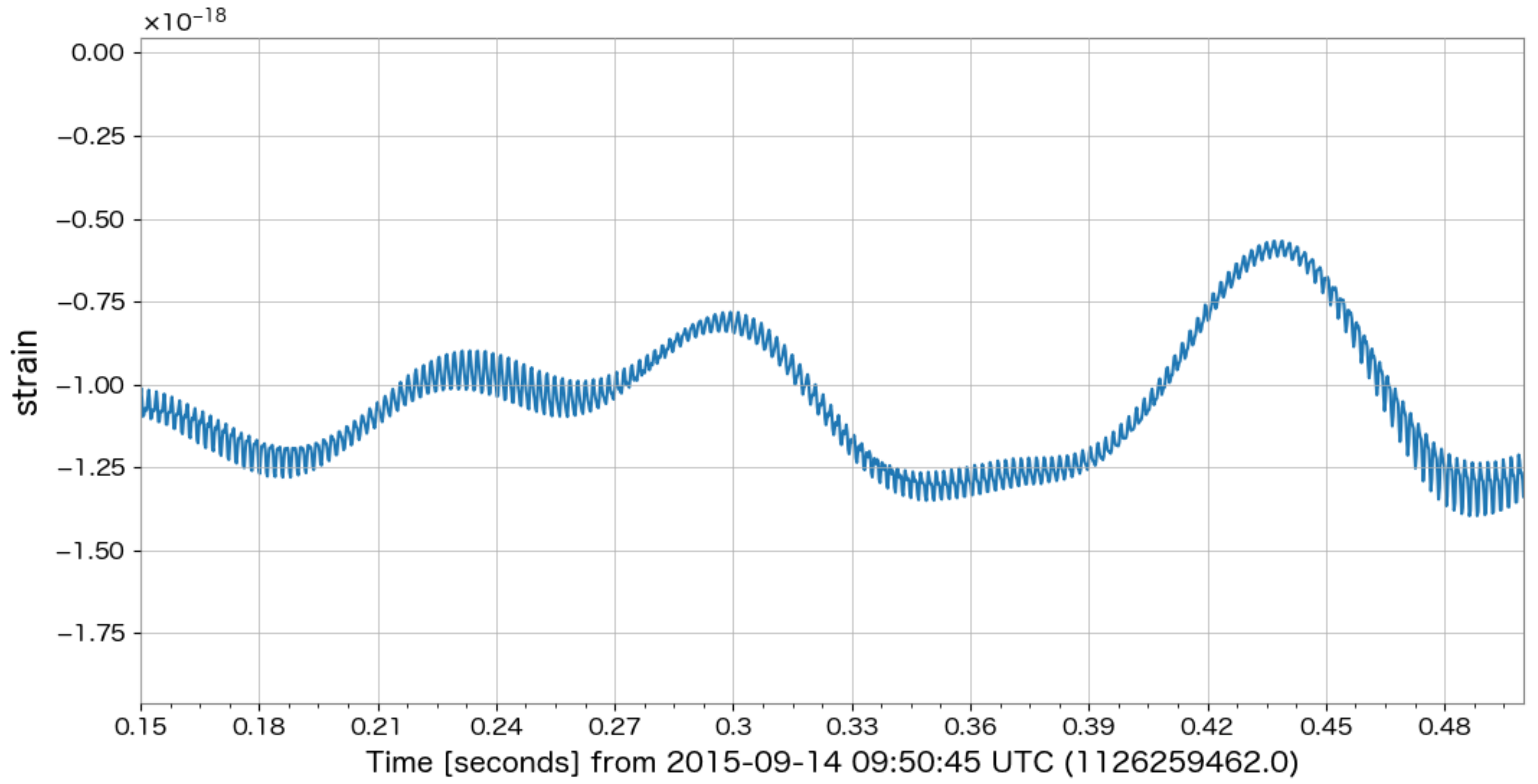
https://medium.com/predict/artificial-neural-networks-mapping-the-human-brain-2e0bd4a93160

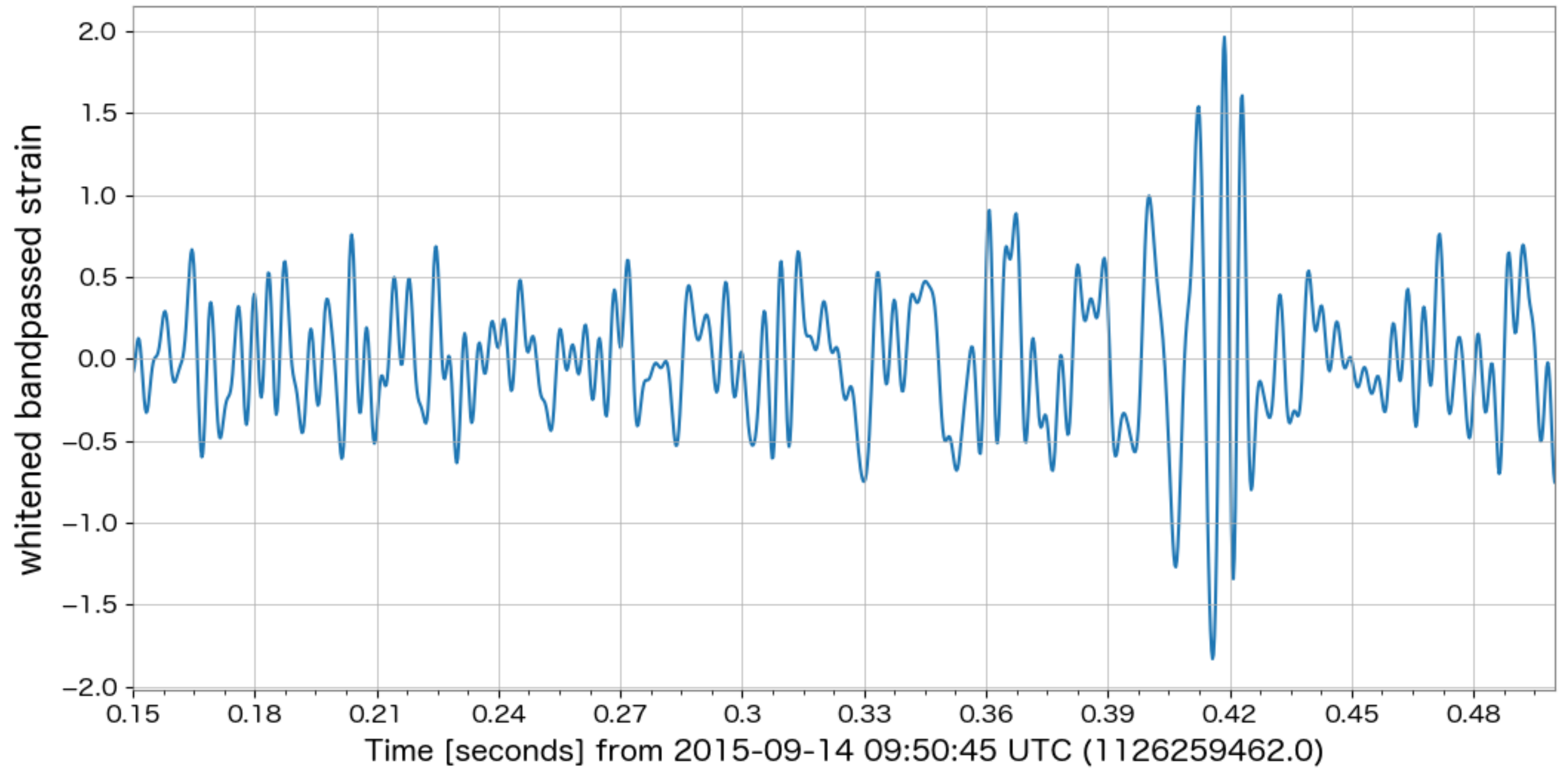


https://developer.nvidia.com/blog/digits-deep-learning-gpu-training-system/

Preprocess

- The preprocess transforms data into another type of data so that neural networks will be able to treat the data more easily.
- Example: whitening and bandpass for CBC signals
  - A raw strain data is dominated by detector noise. CBC signals can not be found by eye. After whitening and bandpass, CBC signals can be easily found by eye.





Proposed algorithm

```
Place the coarse grid points on the sky
For n_grid:
    Remove doppler modulation
    Make spectrogram
    For frequency bin:
        Perform Fourier transform over all times
        Give transformed data to neural network and get prediction
        If prediction = "CGW exists":
            Store {n_grid, frequency bin} as a candidate
```

Preprocess

Neural network

Waveform model of CGW & line

CGW

$$h_{\mathrm{obs}}(t) = h_0 \left[F_+(t) \frac{1+\cos^2\iota}{2} \cos \underline{\Phi(t)} + F_\times(t) \cos\iota \sin \underline{\Phi(t)} \right]$$

Phase

$$\underline{\Phi(t)} = 2\pi f_{\mathrm{gw}} \left(t + \underline{\frac{\boldsymbol{r}(t)\cdot\boldsymbol{n}}{c}} \right) + \pi \dot{f} \left(t + \underline{\frac{\boldsymbol{r}(t)\cdot\boldsymbol{n}}{c}} \right)^2 + \phi_0$$

Doppler modulation

$\mathbf{r}(t)$: detector position w.r.t. SSB, $\mathbf{n}$: source direction

Line

$$n_{\mathrm{line}}(t) = n_0 \cos(2\pi f_{\mathrm{line}} t + \phi_0)$$

(We assume a line has stable frequency and exists for entire observation period.)

Amplitude is measured by $$\hat{h}_0 = \frac{h_0}{\sqrt{S_n}}, \quad \hat{n}_0 = \frac{n_0}{\sqrt{S_n}}$$

Remove Doppler modulation

We use a coarse grid points on the sky. It should be enough dense to remove the Earth rotation (daily).

$$\Phi(t) \sim 2\pi f_{\mathrm{gw}} t + \Phi_{\oplus}(t) + \Phi_{\odot}(t)$$

On the frequency bin being closest to the GW frequency, SFT only has the residual phase originated from the Earth's orbital motion (annual).

$$\Phi(\tau) \sim 2\pi f_{\mathrm{gw}} \tau + \delta\Phi_{\odot}(\tau)$$

SSB time $\tau := t + \dfrac{\boldsymbol{r}(t) \cdot \boldsymbol{n}_{\mathrm{grid}}}{c}$

Residual phase $\delta\Phi_{\odot} := 2\pi f_{\mathrm{gw}} \dfrac{\boldsymbol{r}_{\odot}(\tau) \cdot \Delta\boldsymbol{n}}{c}$

$\Delta\boldsymbol{n} := \boldsymbol{n}_{\mathrm{source}} - \boldsymbol{n}_{\mathrm{grid}}$

* For explanation, we neglect the antenna pattern functions, the window function.
* Also, we neglect d*f*/d*t*.
* Lines can be distinguished from CGWs because of this process.

Perform Fourier transform after SFT

We pick up the frequency bin closest to f_{gw} .

$$h_{j,k}^{\mathrm{SFT}} \sim \exp[i\boldsymbol{r}(t)\cdot\Delta\boldsymbol{n}] \sim \exp[iR_{\mathrm{ES}}\Delta\theta\cos(\Omega_{\odot}t)]$$

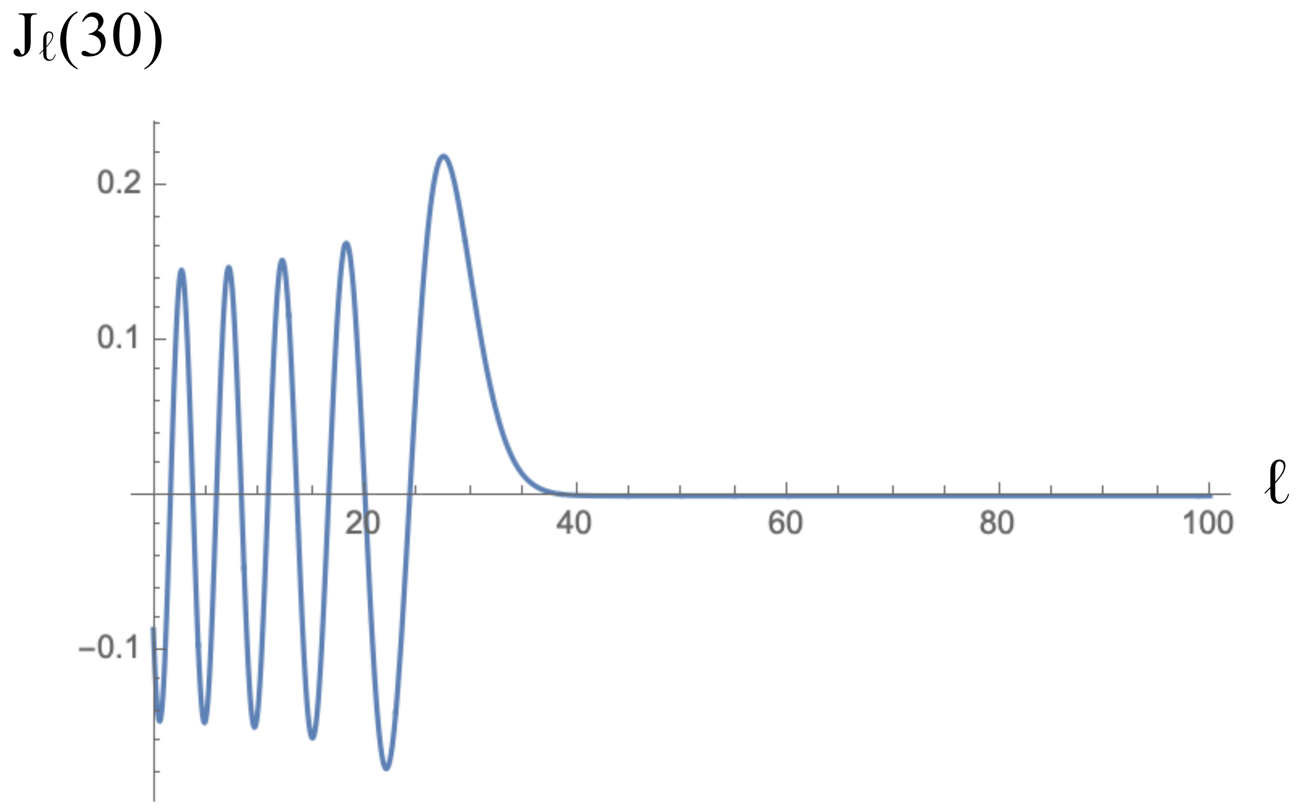
SFT data contains residual phase which can be modeled by cos function.

It can be rewritten by the Fourier transform of Bessel function by Jacobi-Anger expansion.

$$\exp[iR_{\mathrm{ES}}\Delta\theta\cos(\Omega_{\odot}t)] = \sum_{\ell=-\infty}^{\infty} i^{\ell} J_{\ell}(R_{\mathrm{ES}}\Delta\theta)e^{i\ell\Omega_{\odot}t}$$

By performing another Fourier transform, you can accumulate the signal power into small number of bins.

$$\mathrm{H}_{\ell,k} := \frac{1}{N}\sum_{j=0}^{N-1} h_{j,k}^{\mathrm{SFT}} e^{-2\pi i j\ell/N} \simeq J_{\ell}(R_{\mathrm{ES}}\Delta\theta)$$



Transformed data

TSY & Tanaka, PRD103, 084049 (2020)
TSY *et al.*, PRD106, 024025 (2022)

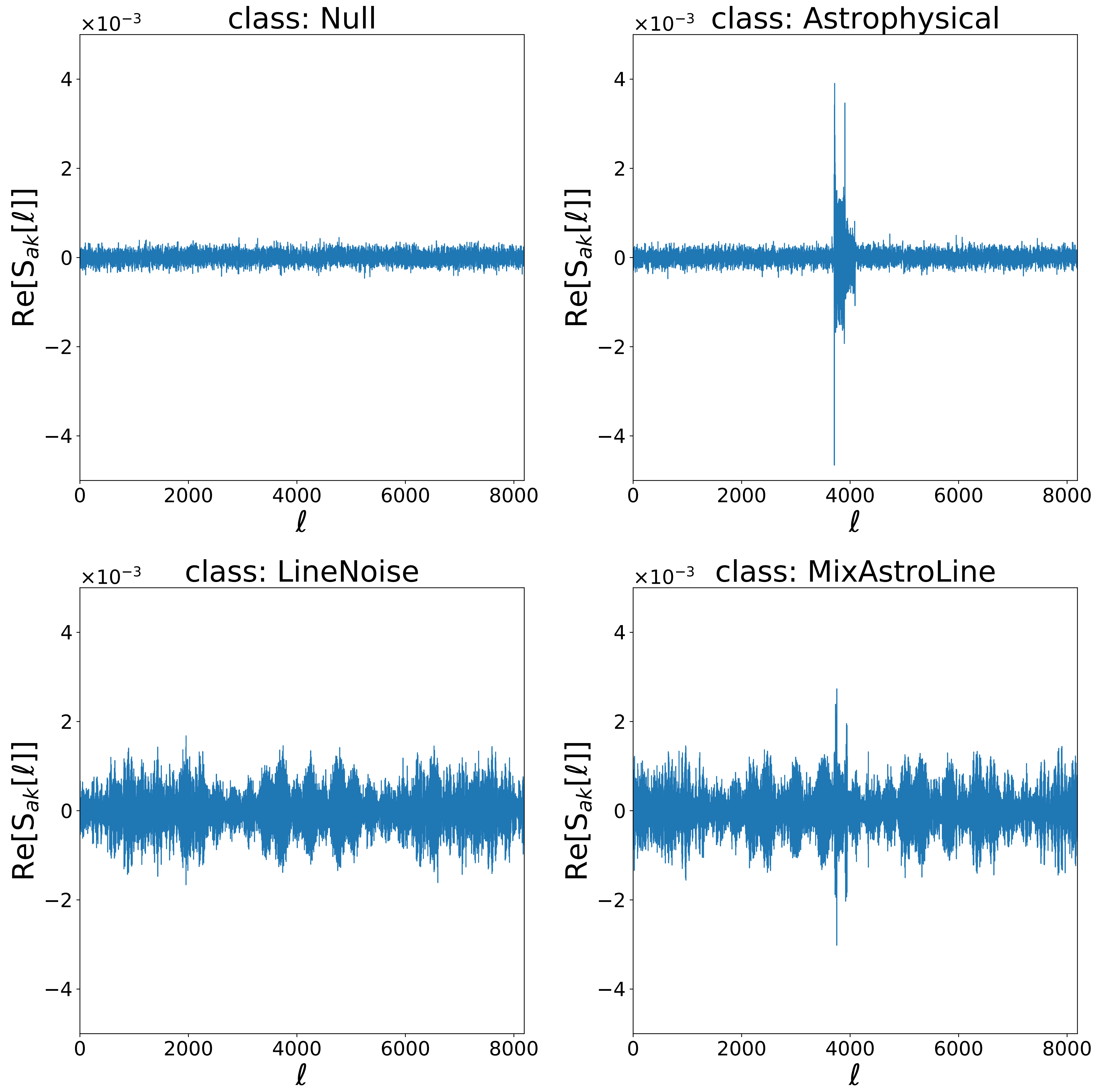
We assume four classes:

1. Null
2. Astrophysical (GW signal)
3. LineNoise
4. MixAstroLine (Line noise + GW signal)

(All classes contain Gaussian noises.)

CGW and lines can be distinguished by the differences in the response to the removal of Doppler modulation.

* In these figures, we inject a strong CGW signal for visualization.



Dataset settings

- $f_{\mathrm{gw}} = f_k + \beta T_{\mathrm{seg}}^{-1}$, where $f_k = 100$ Hz and $\beta \in [-0.5, 0.5]$
- Fix df/dt = 0 [Hz/sec] (but we tested CNN also for nonzero df/dt cases.)
- T_{seg} = 2048 sec
- T_{obs} = 16777216 sec (~ 0.5yr)
- Random inclination angle, polarization, initial phase
- Sky positions are also randomly chosen. For each waveform, we pick up the closest grid point to the source location.
- Amplitude of signal : 0.01 ~ 10.0
- Amplitude of line noise : 1.0 ~ 10.0
- Line noise is a sinusoidal waveform. $f_{\mathrm{line}} = f_k + \beta T_{\mathrm{seg}}^{-1}$, $\beta \in [-0.5, 0.5]$
- # of training data: Null = Astrophysical = LineNoise = Mix = 20,000

Neural network & training settings

- input size = 8192 (= T_{obs} / T_{seg})
- channels = 2 (real and imaginary parts of transformed signal)
- output size = 4, we interpret each of them as the probability of each class
- loss function: cross entropy loss
- batch size: 512
- total epoch: 300
- learning rate: 1.0e-4
- optimizer: Adam
- implemented with PyTorch, trained with a GPU GeForce 1080Ti

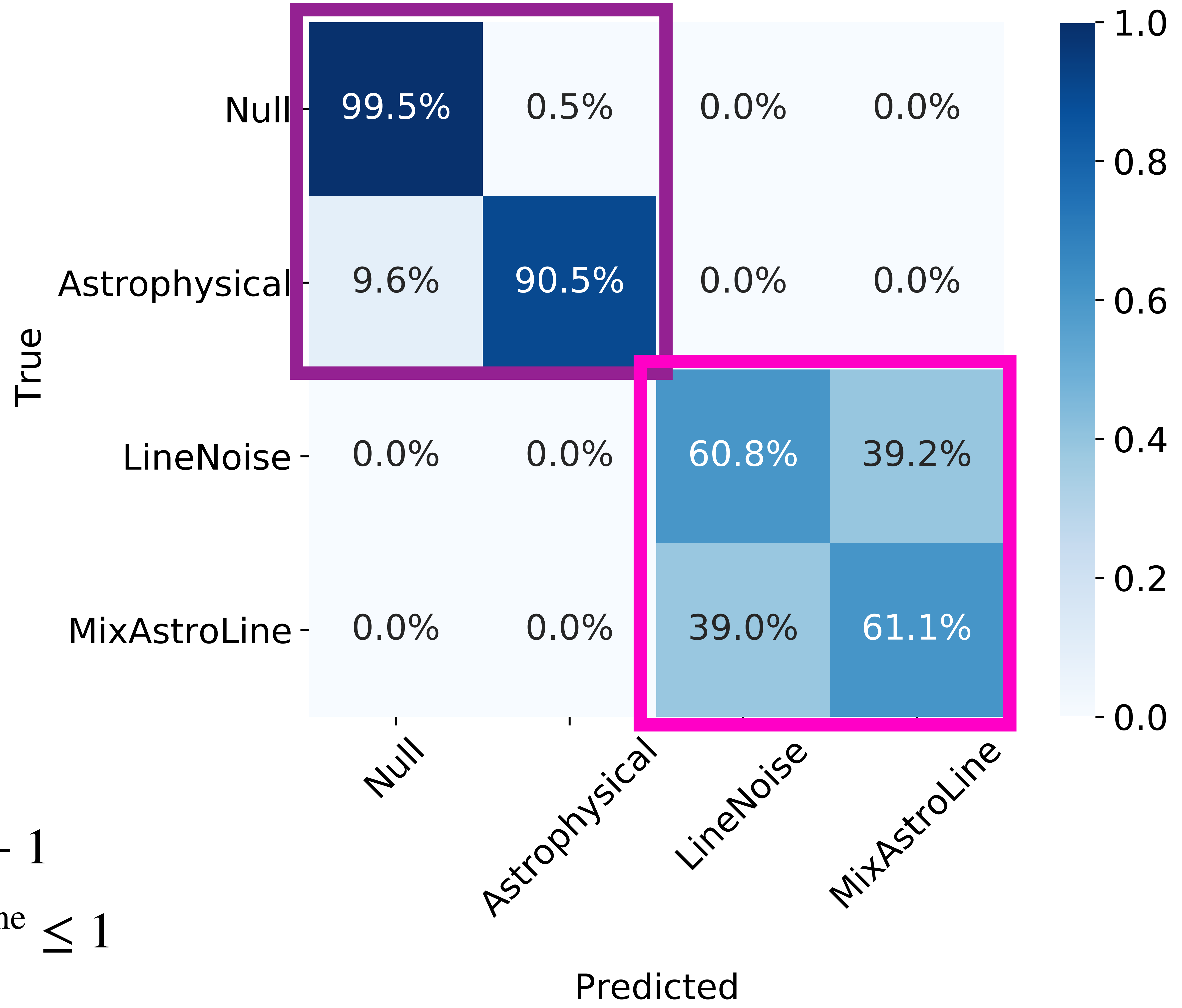
Layer	Output size	# of parameters
1D convolutional	(16, 8177)	528
ReLU	(16, 8177)	-
1D convolutional	(16, 8162)	4112
ReLU	(16, 8162)	-
Max pooling	(16, 2040)	-
1D convolutional	(32, 2033)	4128
ReLU	(32, 2033)	-
1D convolutional	(32, 2026)	8224
ReLU	(32, 2026)	-
Max pooling	(32, 506)	-
1D convolutional	(64, 503)	8256
ReLU	(64, 503)	-
1D convolutional	(64, 500)	16448
ReLU	(64, 500)	-
Max pooling	(64, 125)	-
Flattening	(8000,)	-
Fully-connected	(512,)	4096512
ReLU	(512,)	-
Fully-connected	(64,)	32832
ReLU	(64,)	-
Fully-connected	(4,)	130
Softmax	(4,)	-

Refs:
Kingma and Ba, arXiv: 1412.6980 (2014)
Paszke *et al.*, arXiv: 1912.01703 (2019)

Validity against to line noises

TSY & Tanaka, PRD103, 084049 (2020)
TSY *et al.*, PRD106, 024025 (2022)

- CNN can distinguish the **presence** and the **absence** of line noise.
- Null (only Gaussian noise) is misclassified as Astrophysical class with the probability of 0.5%.
- In the presence of line noise, it is hard to detect the signal.



- 2,000 data for each class
- GW amplitudes are uniformly sampled from $-2 \leq \log_{10} \hat{h}_0 \leq -1$
- Line noise amplitudes are uniformly sampled from $0 \leq \log_{10} \hat{h}_0^{\mathrm{line}} \leq 1$

Rough comparison

TSY & Tanaka, PRD103, 084049 (2020)
TSY *et al.*, PRD106, 024025 (2022)

Sensitivity

$$\mathcal{D}^{95\%} = \sqrt{S_n}/h_0^{95\%}$$

Method	Frequency band	$\mathcal{D}^{95\%}$
FrequencyHough	at 100 Hz	42$\sim$ 43
SkyHough	at 116.5 Hz	47.2
Time-domain $\mathcal{F}$-statistic	at 100 Hz	26$\sim$52
SOAP	on 40$\sim$500 Hz	9.9
Our method	$\lesssim$ 100 Hz	43.9

Computational cost (CPU)

Method	core-hour
FrequencyHough	9×10^6
SkyHough	2.5×10^6
Time-domain $\mathcal{F}$-statistic	2.4×10^7
SOAP	$1 - 2 \times 10^2$
Our method	1.4×10^5

Comparable or better sensitivity w/ O(10-100) speed up

- ✓ Intel E5-2670 8 operations/clock, 2.6GHz -> 20.8GFlops/core
- ✓ Simulated data for our method, observational result for other methods.
- ✓ The parameter region and the data duration are different depending on the method

Roughly estimated computational time for GPU $T_{\mathrm{CNN}} \simeq 1.02 \times 10^8$ [sec]

It can be expected to be decreased by

(i) the improvement of hardware, (ii) the use of multiple GPUs, (iii) optimize params.

Results for non-zero d*f*/d*t*

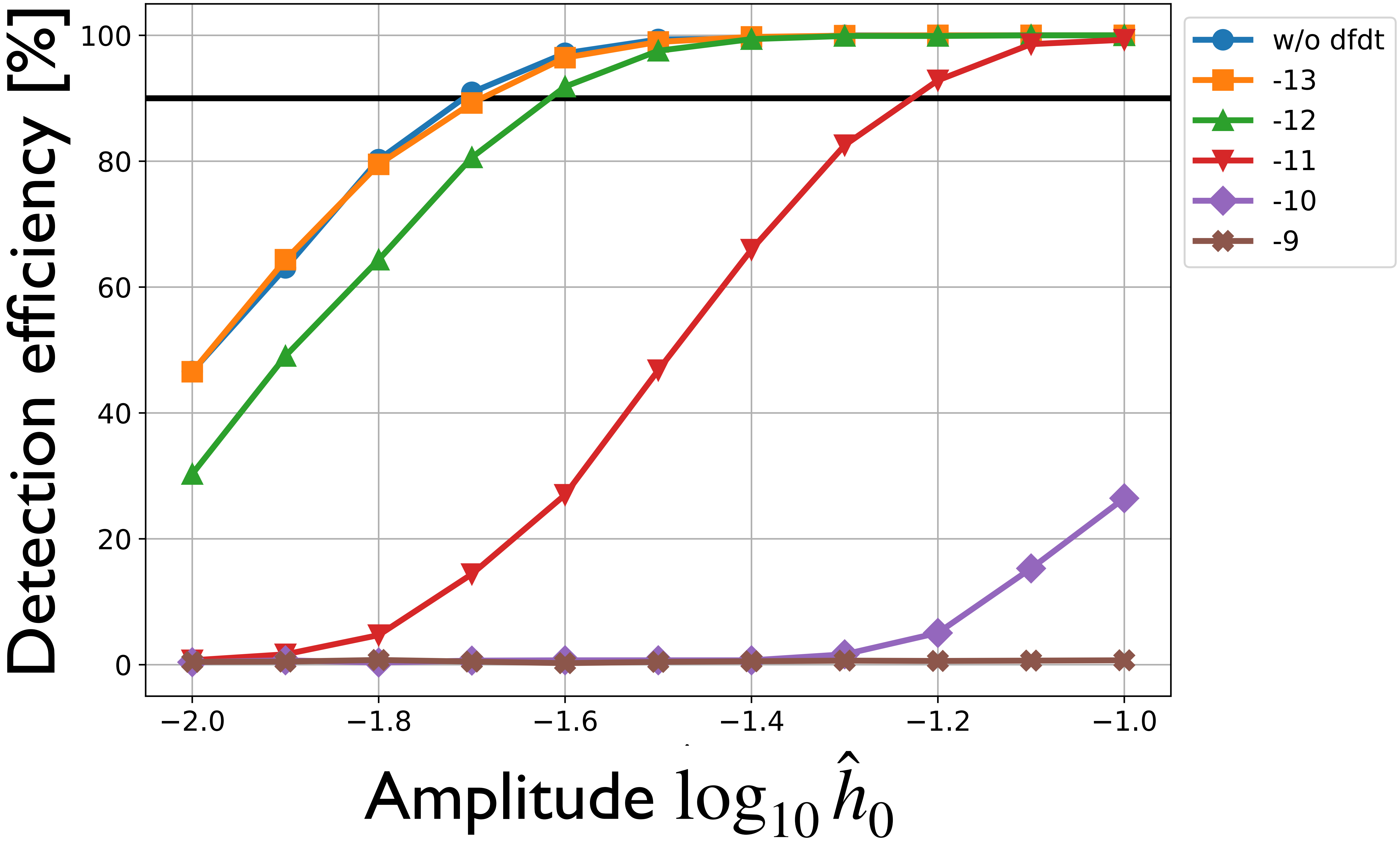
TSY & Tanaka, PRD103, 084049 (2020)
TSY *et al.*, PRD106, 024025 (2022)

We tested our neural network for the data with non-zero d*f*/d*t*, although the training data have no d*f*/d*t*.

The CNN seems good up to 10^{-12}Hz/sec, where

$(df/dt)(T_{\mathrm{obs}}) \sim (T_{\mathrm{seg}})^{-1}$

For $|df/dt| > 10^{-12}$Hz/sec, the signal cannot be contained into one frequency bin even after the preprocessing.



Extension to directed search

- We assume the grid point exactly matches with the source position.
  - In directed search, we can use the information about the source position.
  - However, the grid still can slightly deviate from the true source position.

- We use grids on df/dt to remove the effect of df/dt from the frequency evolution.
  - The grid width will be determined from the computational cost.

- We enable CNN to take multiple frequency bins as an input.
  - Input data become 2-dimensional image.
  - It may allow incomplete subtraction of the Doppler effects and/or the effect of df/dt.

Proposed algorithm

Place the coarse grid points on the sky and on df/dt

For $\{\boldsymbol{n}_{\mathrm{grid}}, (df/dt)_{\mathrm{grid}}\}$:

  Remove doppler modulation and frequency evolution by df/dt

  Make spectrogram

  For frequency band:

    Perform Fourier transform over all time for each frequency bin

    Give transformed data to neural network and get prediction

    If prediction = "CGW exists":

      Store $\{\boldsymbol{n}_{\mathrm{grid}}, (df/dt)_{\mathrm{grid}}$, frequency bin$\}$ as a candidate

Preprocess

Neural network

Dataset settings (diff)

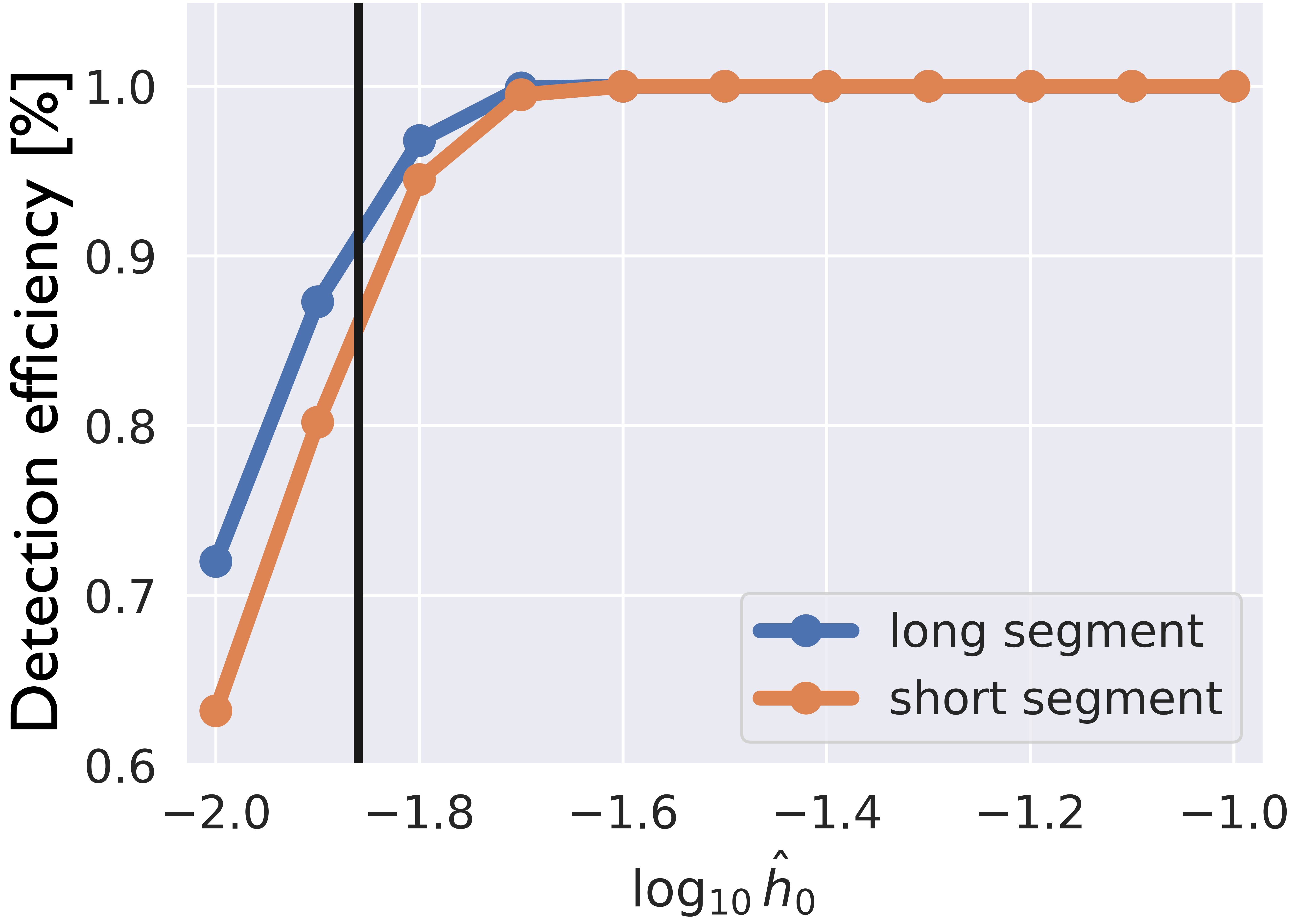
- f_{gw} is sampled from uniform distribution on [10, 1000] Hz.
- df/dt is sampled from log-uniform distribution on [-10^{-8}, 10^{-8}] Hz/sec.
- The source direction and the grid direction coincide (CasA).
- T_{seg} = 65536 sec or 262144 sec
- Dataset consists of 2 classes, {Null, Astrophysical}
- Input is two dimensional image: size (height, width) = (200, 512) or (1200, 128)

- Random inclination angle, polarization, initial phase
- $\log_{10} \hat{h}_0 \in [-2.0, 1.0]$ (→ amplitude of signal : 0.01 ~ 10.0)
- # of data, training : validate = 10000 : 1000. The dataset is augmented by generating random Gaussian noise for every iteration.

Detection efficiency

- Grid width on df/dt = 10^{-11} Hz/sec
- Orange line: T_{seg} = 65536 sec
- Blue line: T_{seg} = 262144 sec
- Black line: $\log_{10} \hat{h}_0$ = -1.86=95% upper limit of CasA at 500Hz with early O3 data (ref: LIGO&Virgo, PRD105, 082005 [2022])

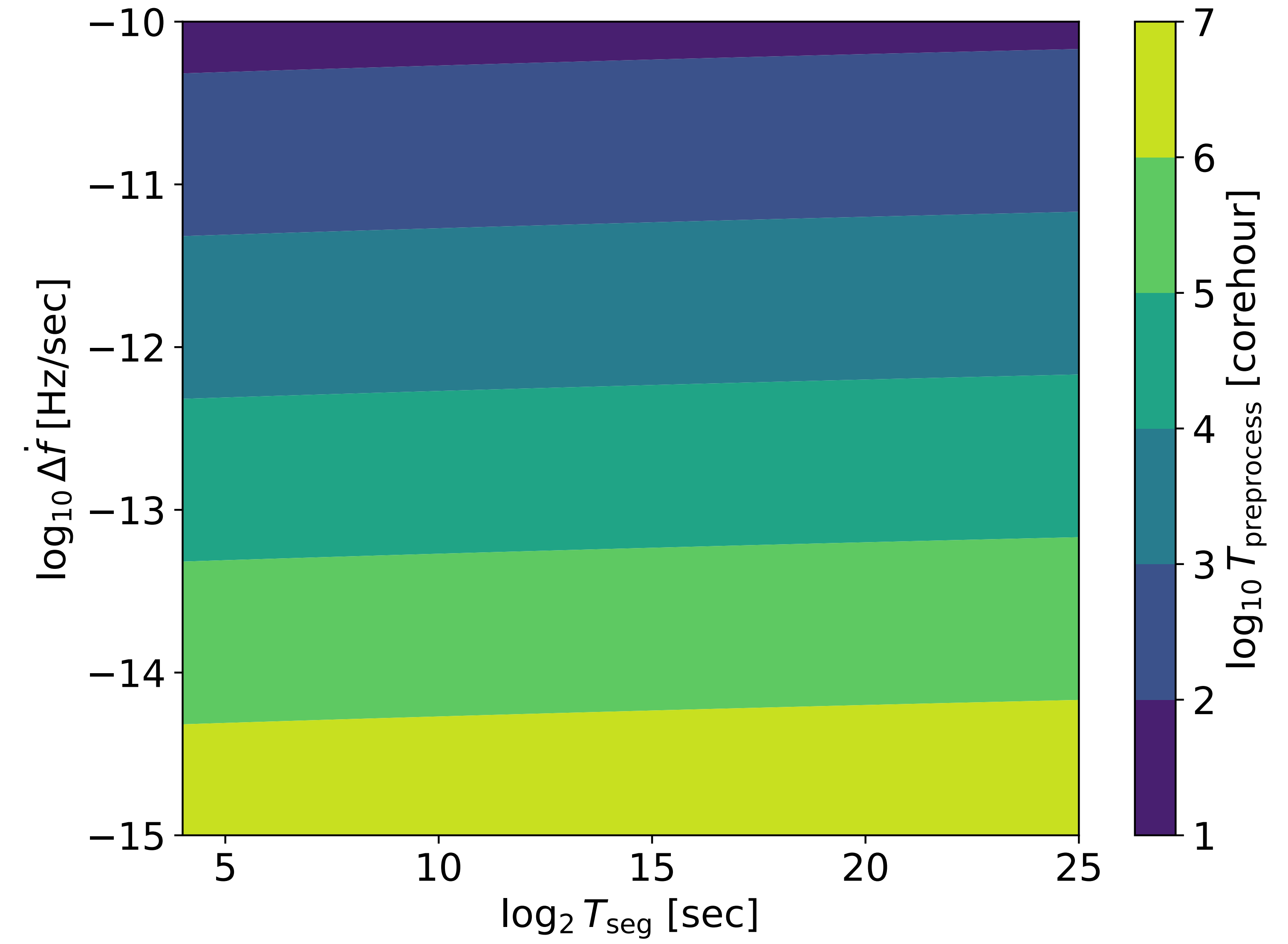
Preliminary



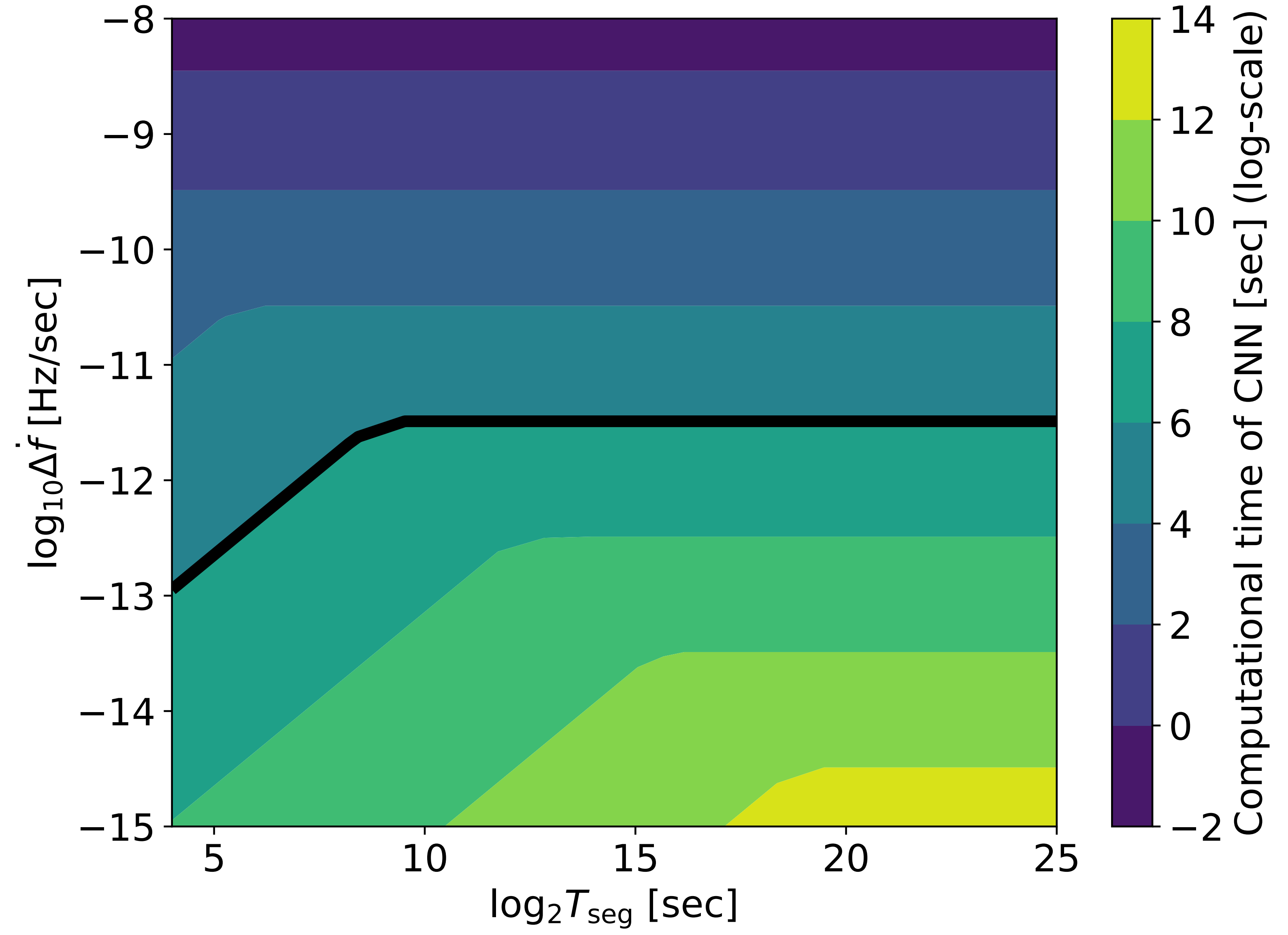
Computational cost

Preliminary

The algorithm is controlled by two parameters: SFT segment duration and grid width on d*f*/d*t*.



Upper: Computational time for preprocess



Lower: Computational time for CNN

Assuming 8.8742 x 10^{-5} sec/image which is used in our previous work

Summary

- We proposed the deep learning method for all-sky search with double Fourier transform. It has the comparable sensitivity to other pipelines with cheaper computational cost though the comparison is very rough.
- We are going to extend our algorithm for the directed search. Based on the preliminary test in which a signal is injected into simulated Gaussian noise, our method can have a comparable sensitivity to the current pipelines.
- To be implemented: data gaps, realistic line models (e.g., fluctuating frequency), non-stationarity of PSDs, multiple detectors

Appendix

Maybe too technical

Classification problem

NN returns four-dimensional vectors in which each component takes value from 0 to 1 and their sum equals to unity.

$$\boldsymbol{p}_{\mathrm{pred}} = (p_1, p_2, p_3, p_4)\,, \qquad \sum_{i=1}^{4} p_i = 1$$

$$(\text{predicted class}) = \mathrm{argmax}_{i=1,2,3,4} p_i$$

1-of-K representation: standard way to label the data for classification problem

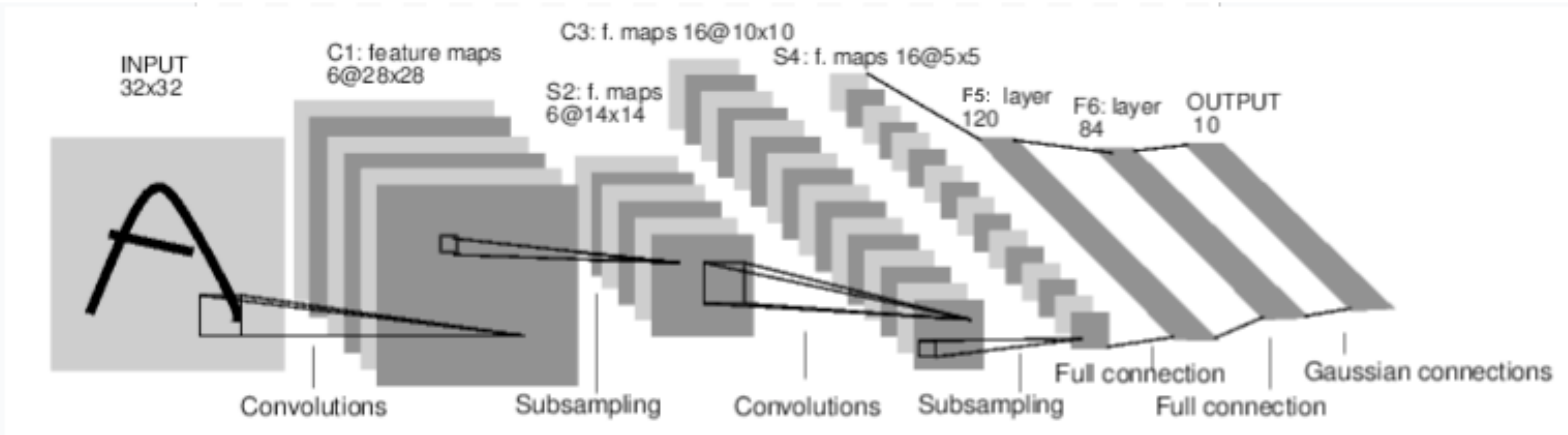
$$t_i = \begin{cases} 1 & \text{if } i\text{th class is true} \\ 0 & \text{otherwise} \end{cases}$$

Loss function: measure the difference between the answer and the NN prediction.

$$L(\boldsymbol{p}, \boldsymbol{t}) = -\sum_{i=1}^{4} t_i \ln p_i$$

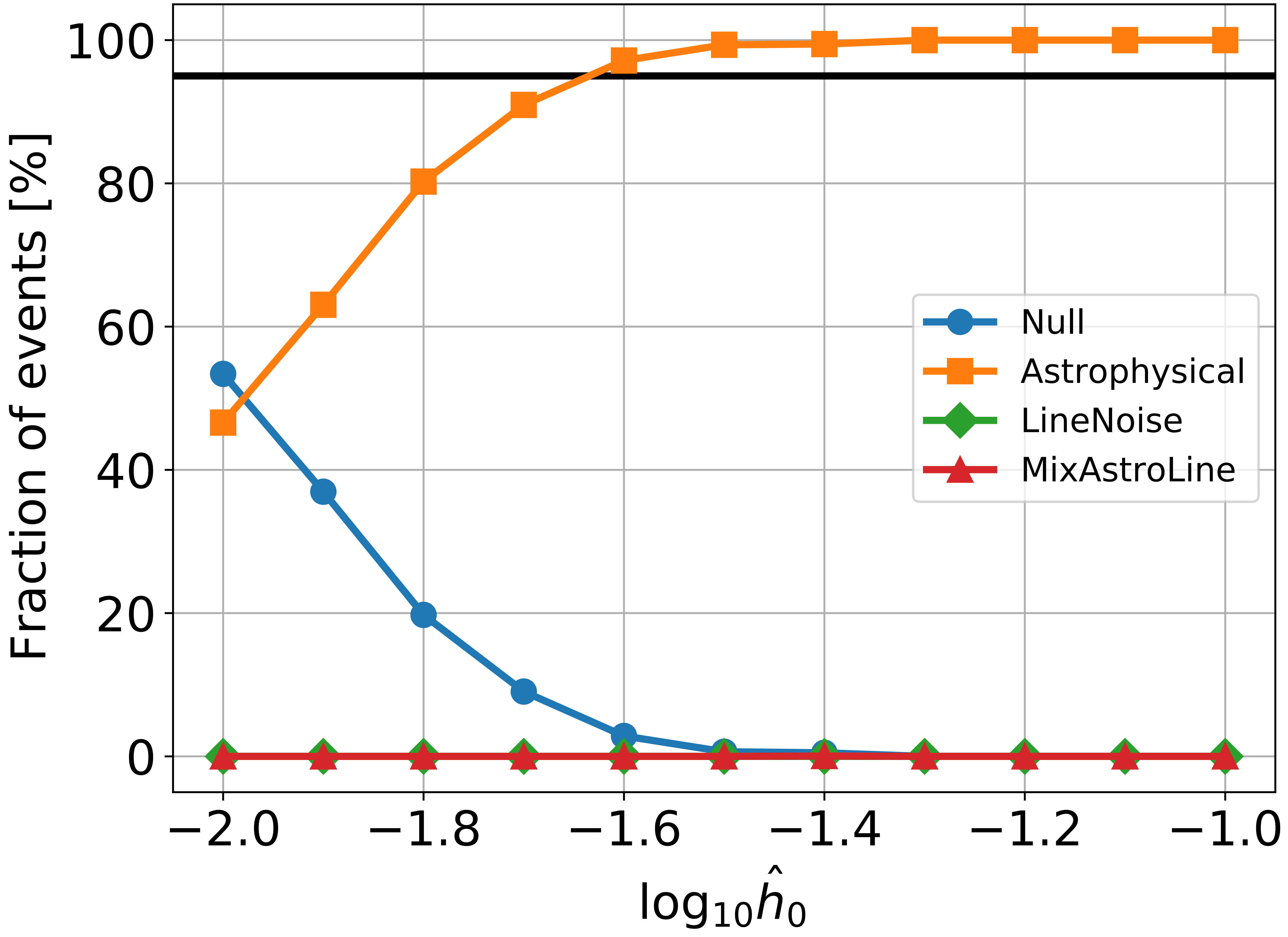
Convolutional neural network

- Convolutional neural network is advantageous for extracting local patterns.
- Pick up a small patch from an image and convolute it with filters.
- Repeating this process for many different patches, we get feature maps.
- e.g., colored image = 2-dimansional pixels with 3channels (RGB)



Results

For Astrophysical class



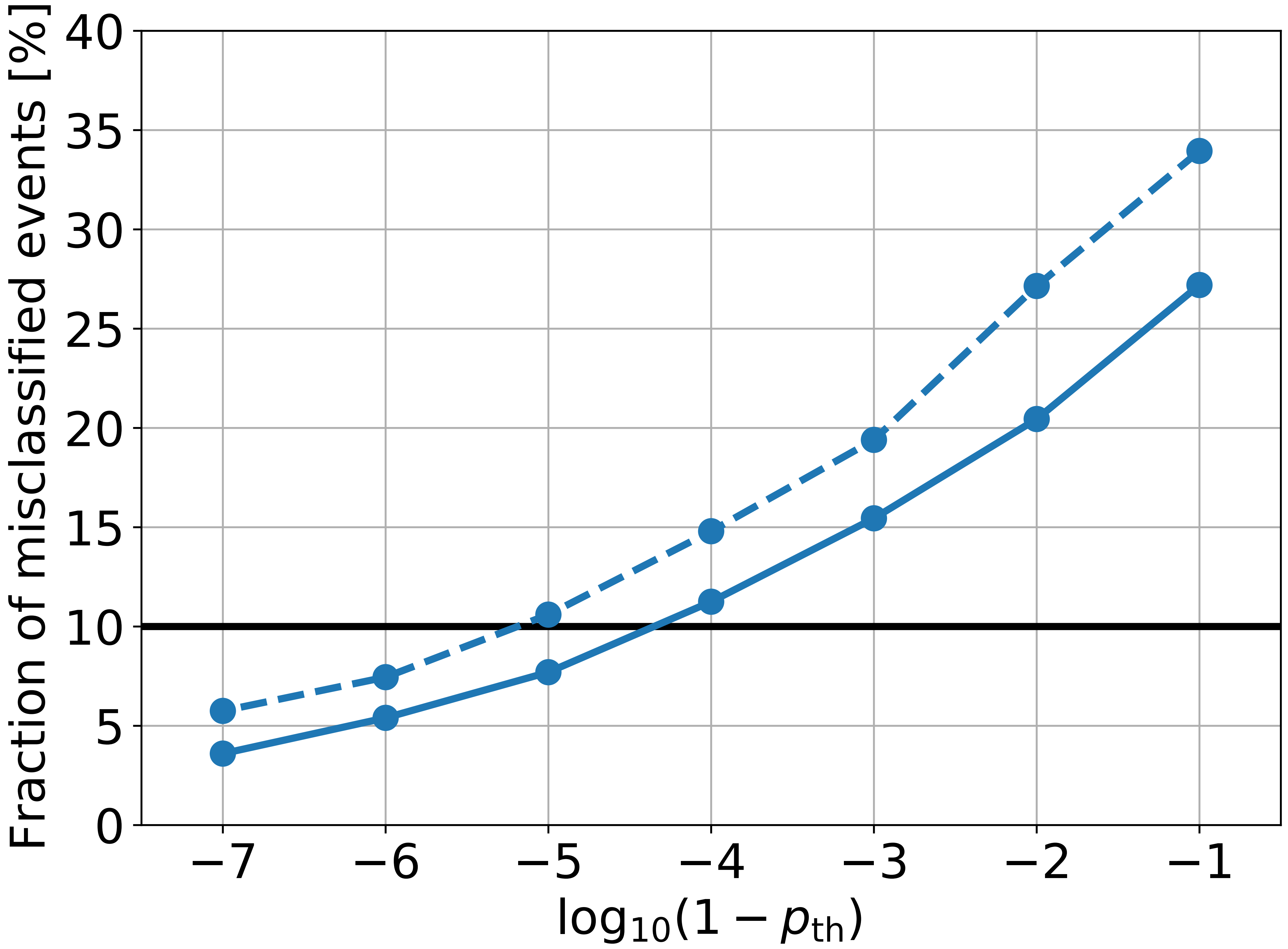
$$\text{Sensitivity depth} = \frac{1}{\hat{h}_0^{95\%}} \simeq 43.9$$

Results

Line noise class

The detection threshold can be changed.

If $p_{\mathrm{th}} \leq p_{\mathrm{Mix}}$, CNN classify the event as Mix class.



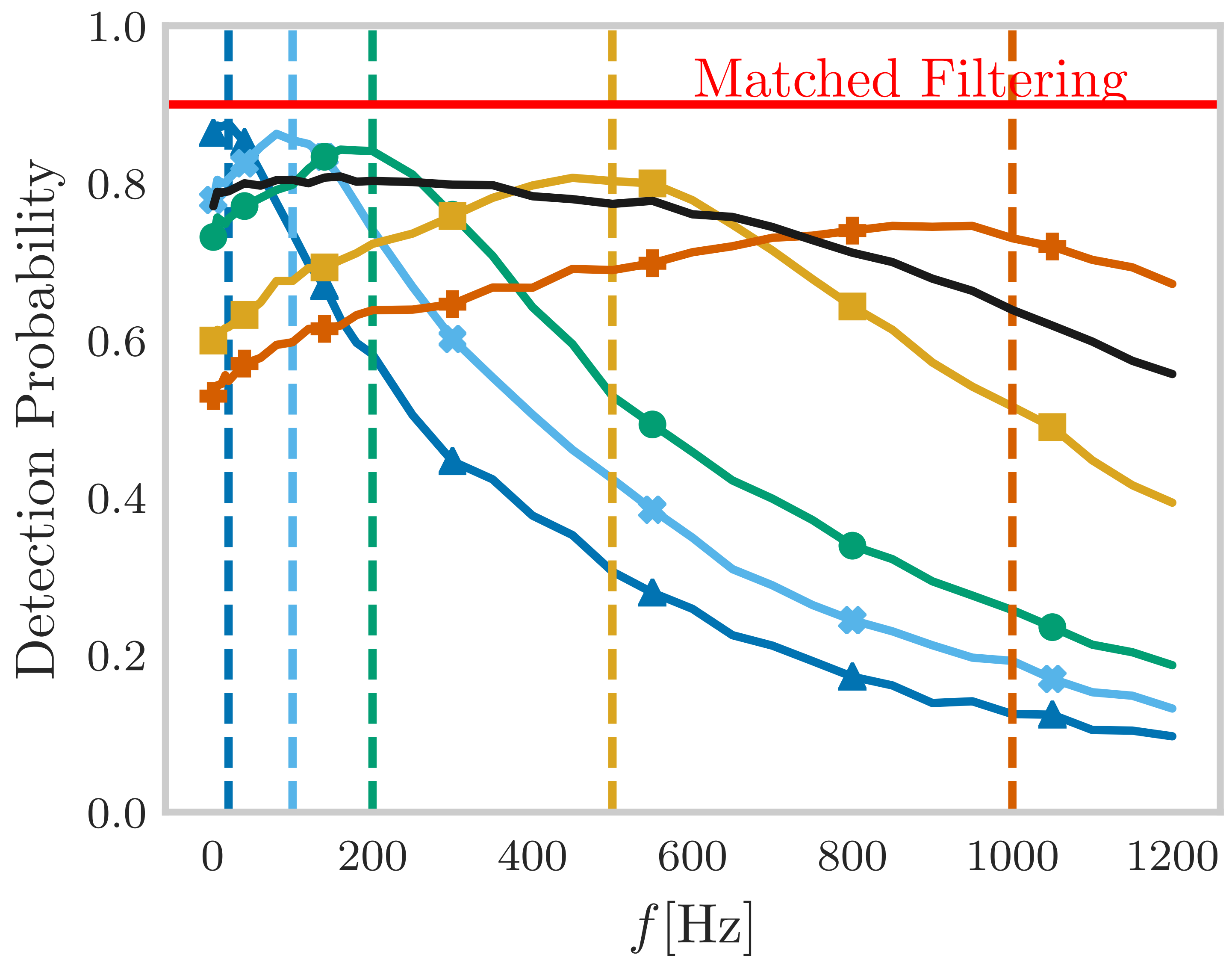
Solid : $\log_{10} \hat{h}_0^{\mathrm{line}} = 0.0$

Dashed : $\log_{10} \hat{h}_0^{\mathrm{line}} = 1.0$

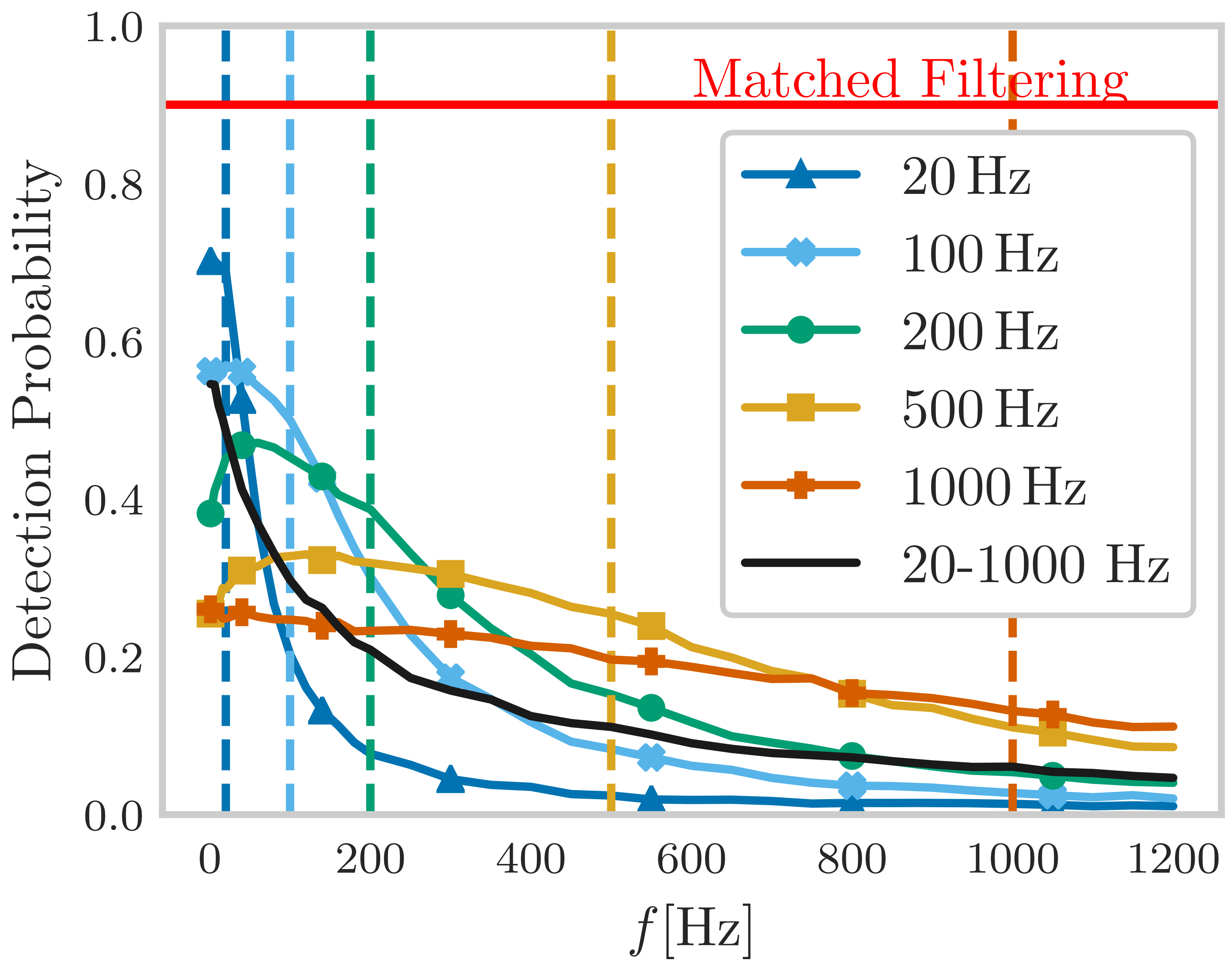
← FAP = 10%

Deep learning application

ref
Dreissigacker *et al.*, PRD100, 044009 (2019)
Dreissigacker & Prix, PRD102, 022005 (2020)



T=10⁵sec



T=10⁶sec

TABLE II. WEAVE parameters and characteristics for the two searches.

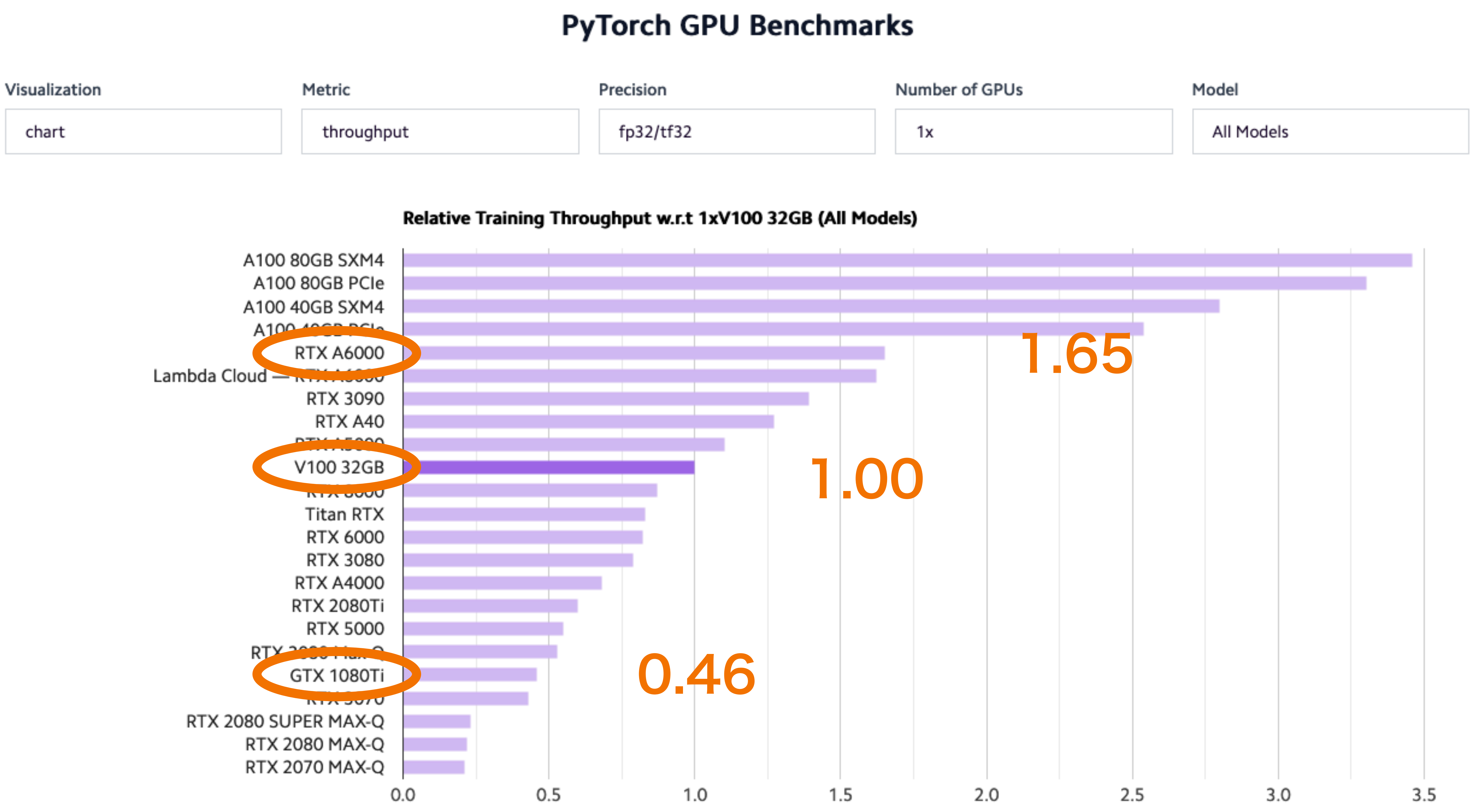
Name	$T = 10^5$ s	$T = 10^6$ s
Mismatch parameter m	0.1	0.2
Average SNR loss $\langle\mu\rangle$	5%	11%
Number of templates $\mathcal{N}$	4×10^{11}	3×10^{14}
Search time [single CPU core]	6.7×10^6 s	3.9×10^9 s

TABLE VII. DNN computing cost (in seconds) for training, search and follow-up (using matched filtering). The respective matched-filtering cost can be found in Table II.

Cost [s]	Training	Search	Follow-up	Total
$T = 10^5$ s	4.3×10^5	58.8	2.2×10^4	4.5×10^5
$T = 10^6$ s	4.3×10^6	196	6.5×10^7	6.9×10^7

Input : the Fourier transform of strain data.

GPU Benchmark



Speed up factor

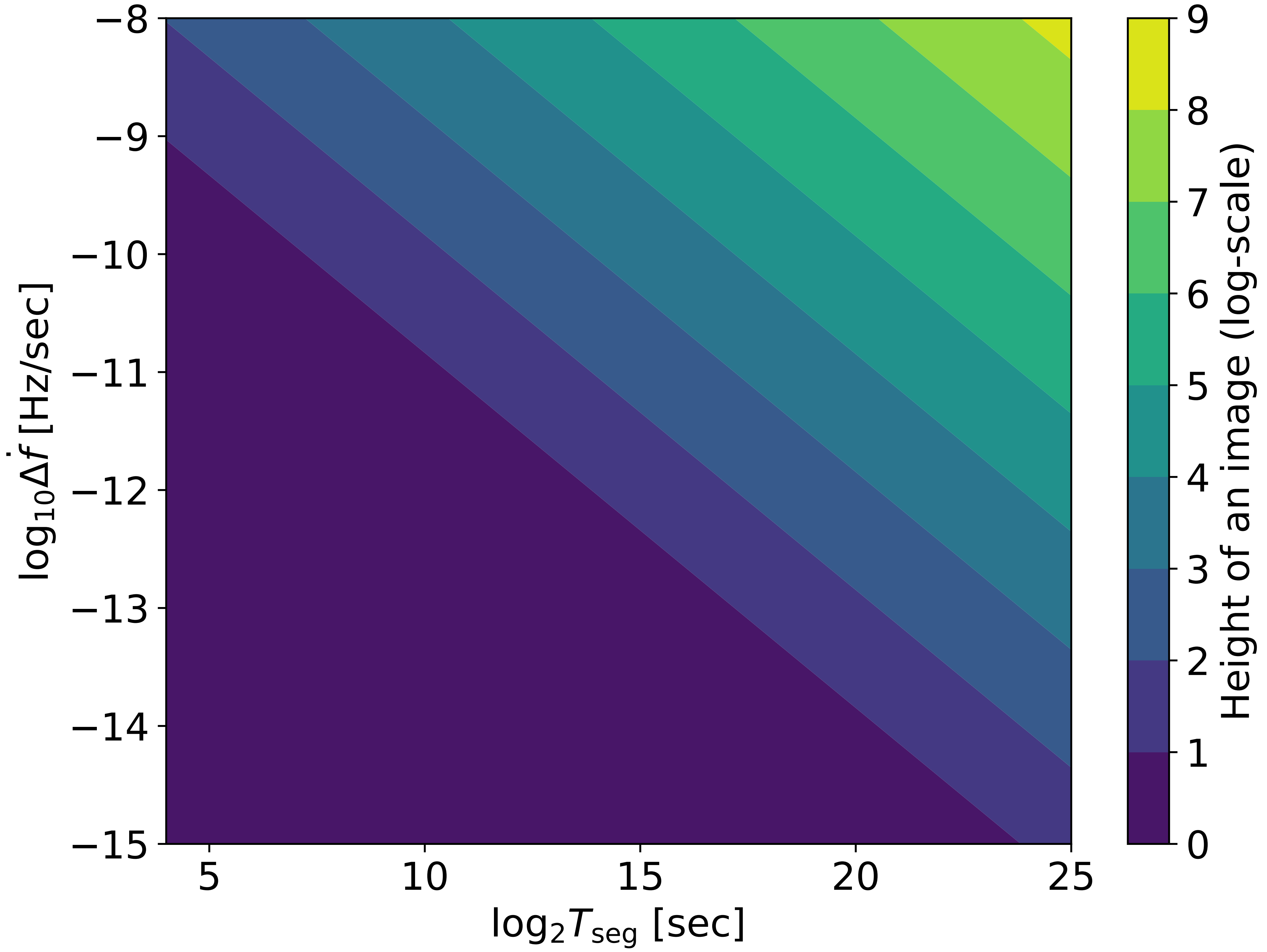
https://lambdalabs.com/gpu-benchmarks

Height of an image

Image size = (Height, Width)

Width = Nseg = Tobs / Tseg

Height = $20 \cdot \dfrac{[\Delta \dot{f}] T_{\mathrm{obs}}}{T_{\mathrm{seg}}^{-1}}$

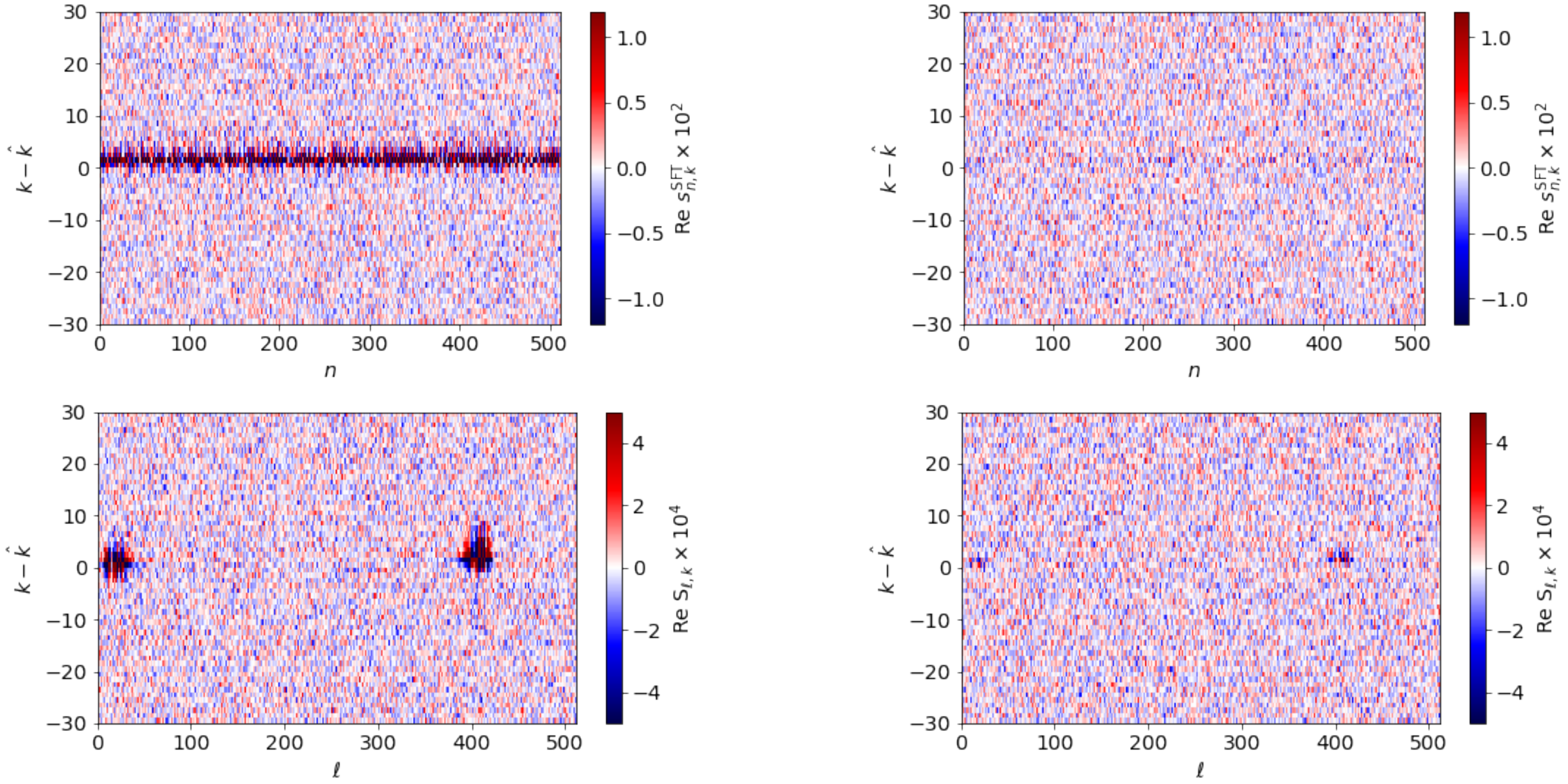


Example of an image

Fgw = 489.08Hz, residual df/dt = 0.0

SFT image

SFT + double FT image



$\log_{10}\hat{h}_0 = -1.0$

$\log_{10}\hat{h}_0 = -2.0$