Neural networks for gravitational-wave trigger selection in single-detector periods

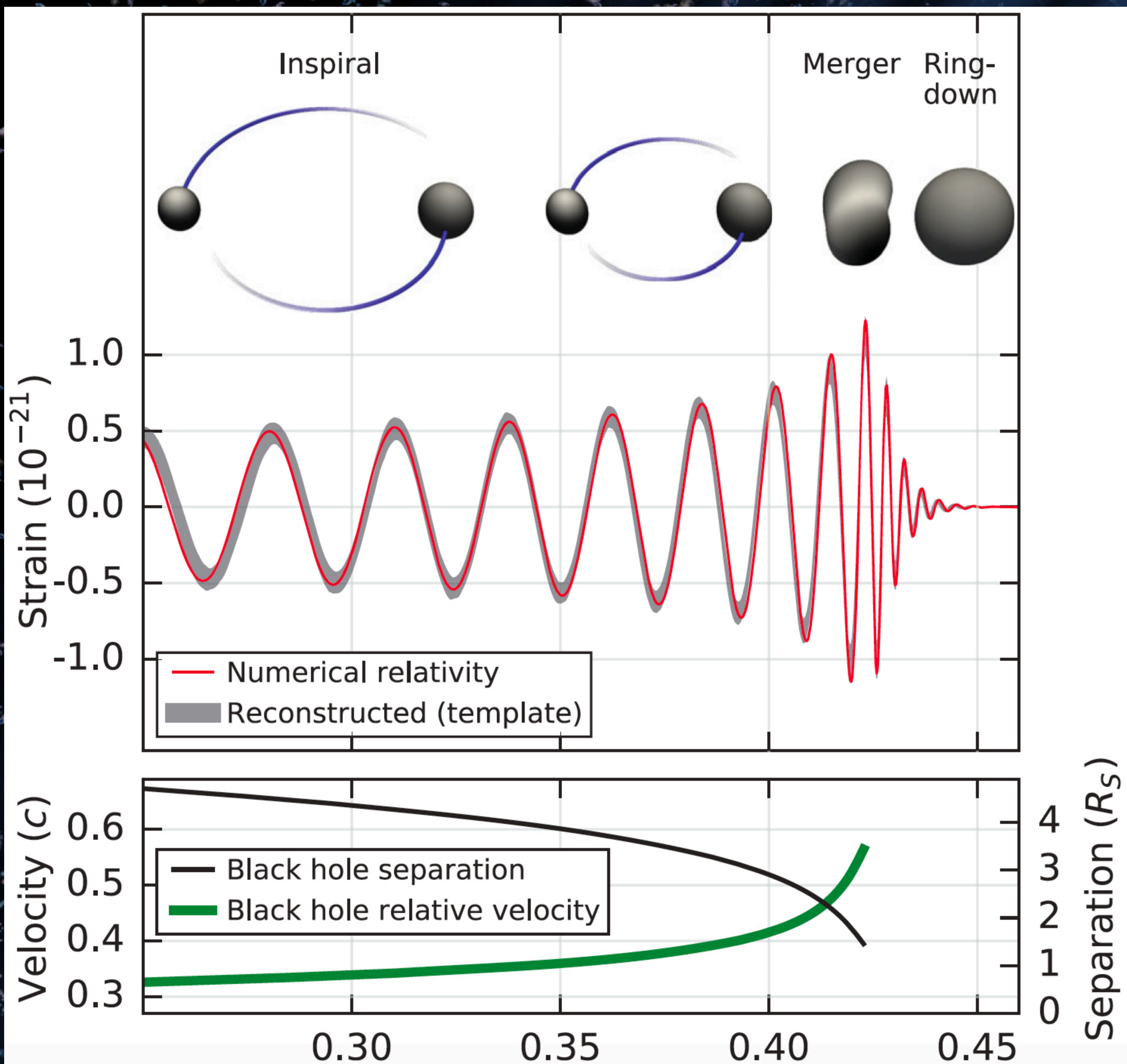
A. Trovato* with M. Bejger and E. Chassande-Mottin,
*APC, CNRS/IN2P3, Université de Paris

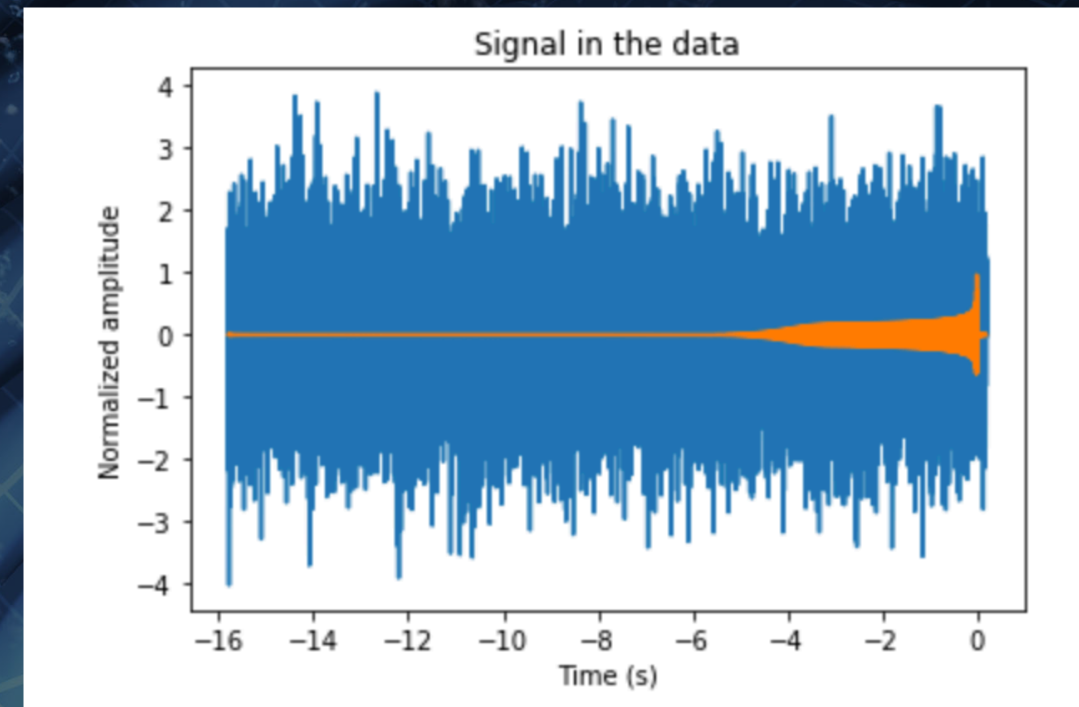
APC, CNRS, FRANCE

G2NET

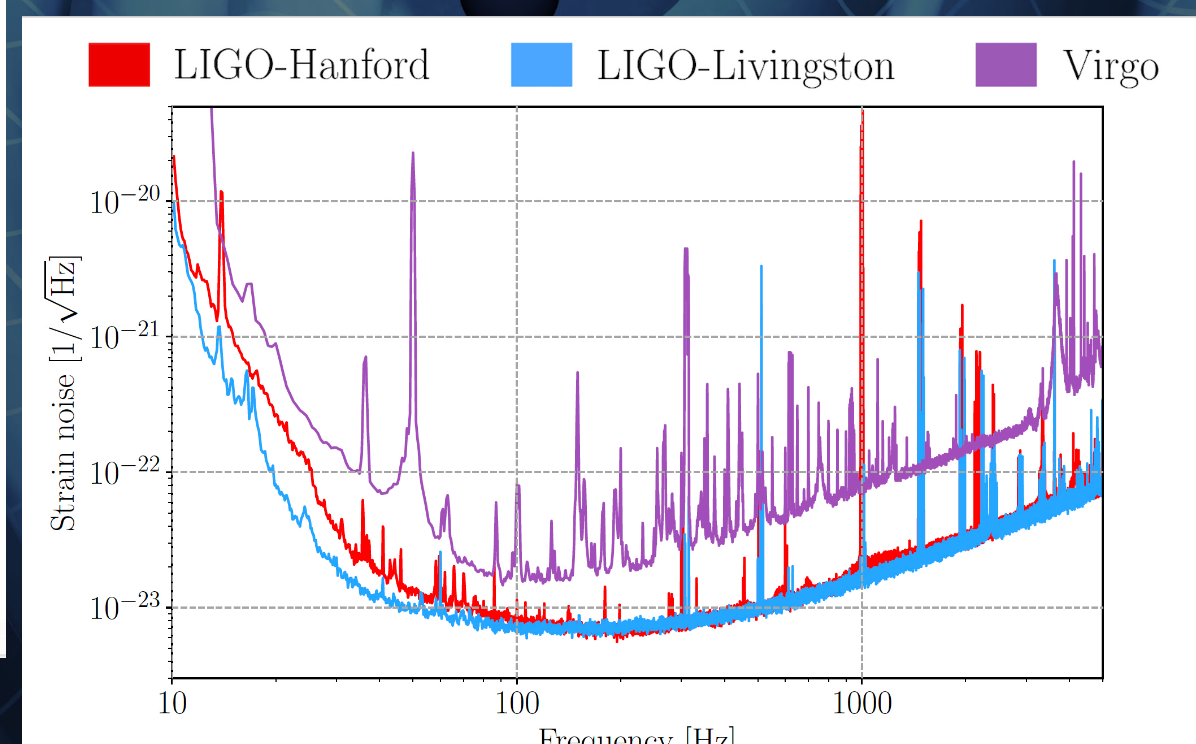
Université de Paris

Gravitational waves detection problem

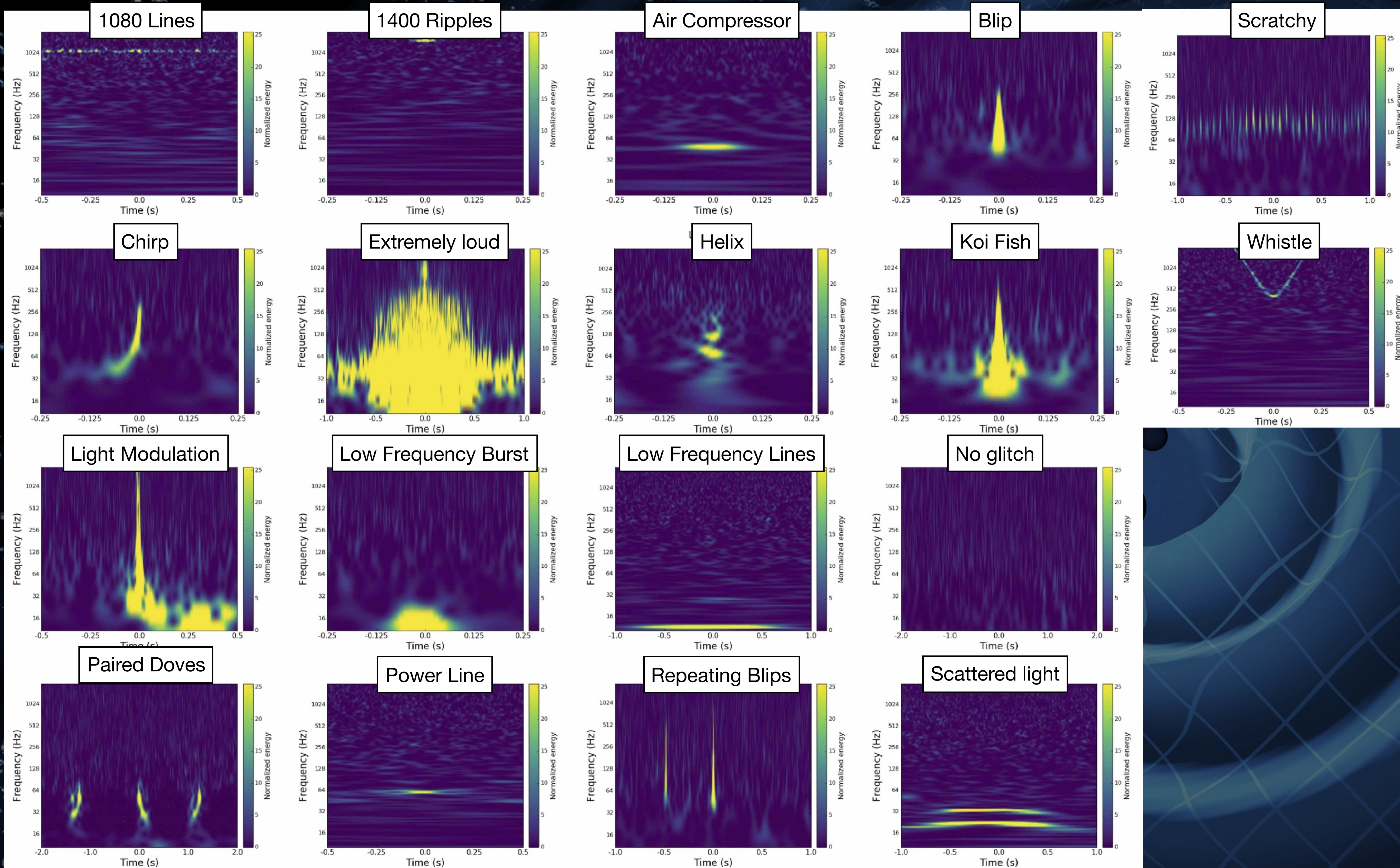




Rare and weak signals in complex background: non-Gaussian non-stationary



Glitches zoo



- **Credits: Gravity Spy dataset**

GW data representation for ML

- Spectrograms representation [e.g. CQG 35 (2018) 095016, Information Sciences 444 (2018) 172]
  - ✓ Deep-learning performs well on images (reuse standard solutions)
  - ✓ Disadvantages:
    - Volume of data (big images)
    - Spectrogram parameters/choice dependent
    - Risk of loosing information due to manipulation

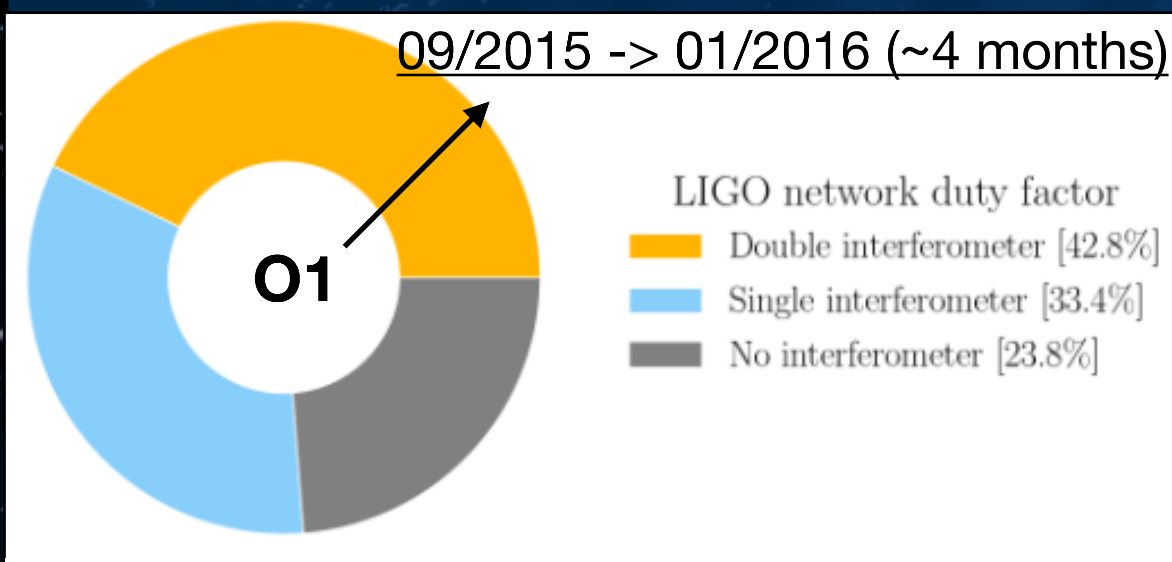
- Time series representation [e.g. Phys. Lett. B 778 (2018) 64, Phys. Rev. D100 (2019) 063015]
  - ✓ full information & reduced volume of data
  - ✓ Multi-detector searches, attempt to make high-confidence detection

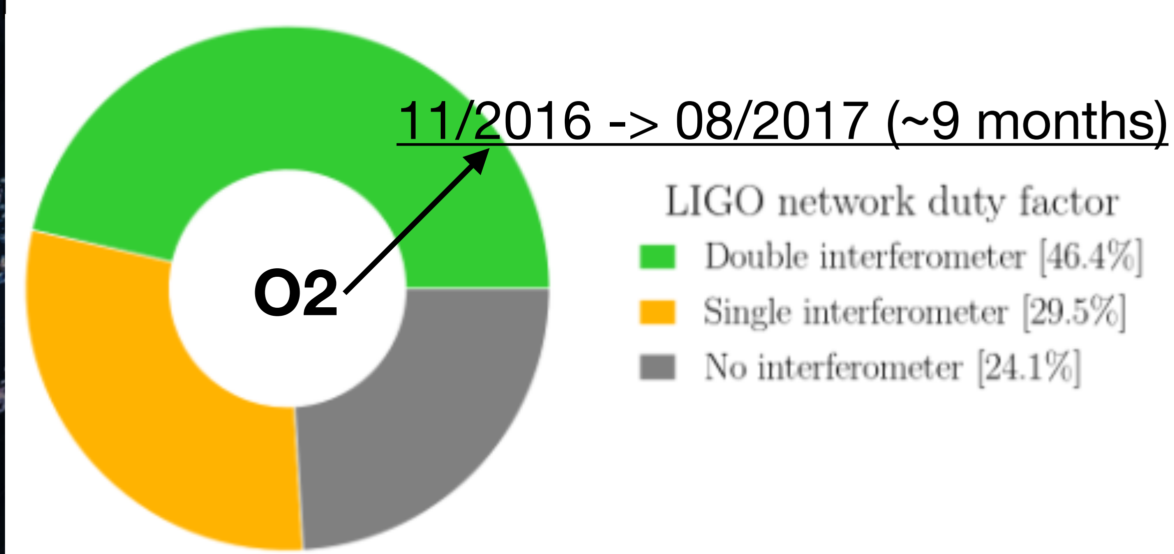
- This work:
  - ✓ time-series representation, single detector, trigger pre-selection

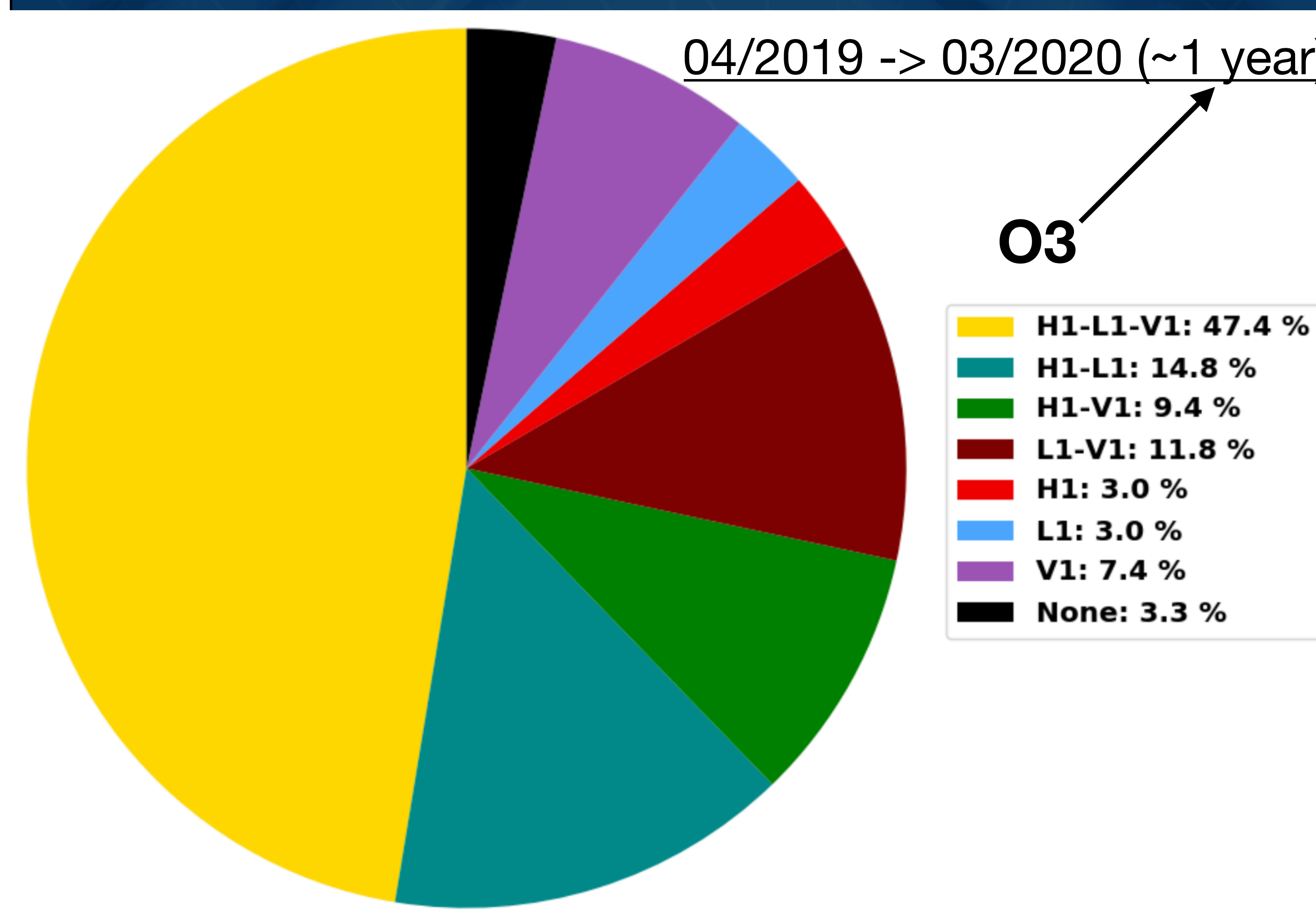
Single-detector time

- Glitch impact on sensitivity is larger during single-detector periods as coincidence with additional detector is impossible. Can machine learning help?

- Single-detector time:
  - ✓ 2.7 months in O1+O2; 1.6 month in O3

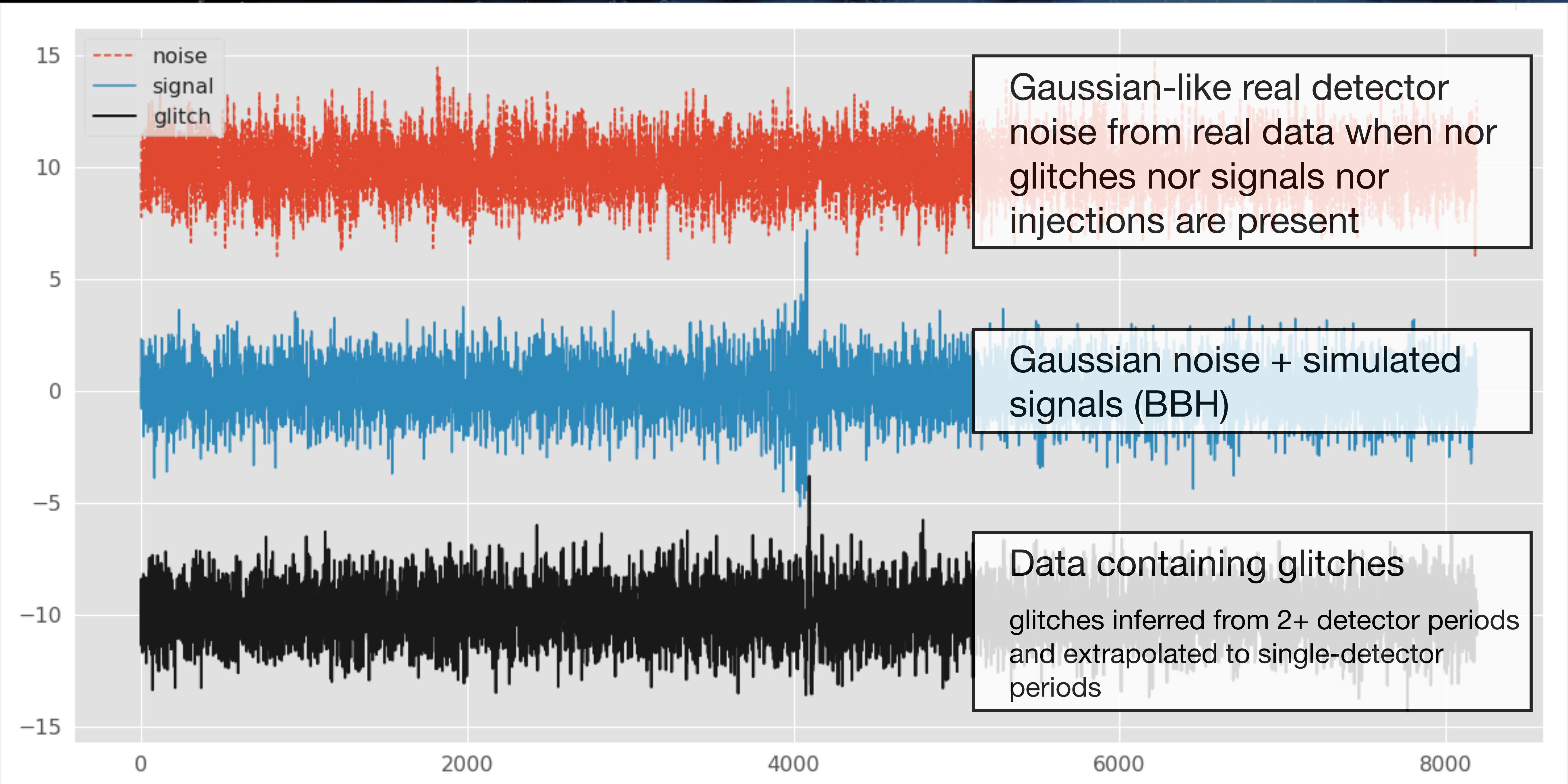






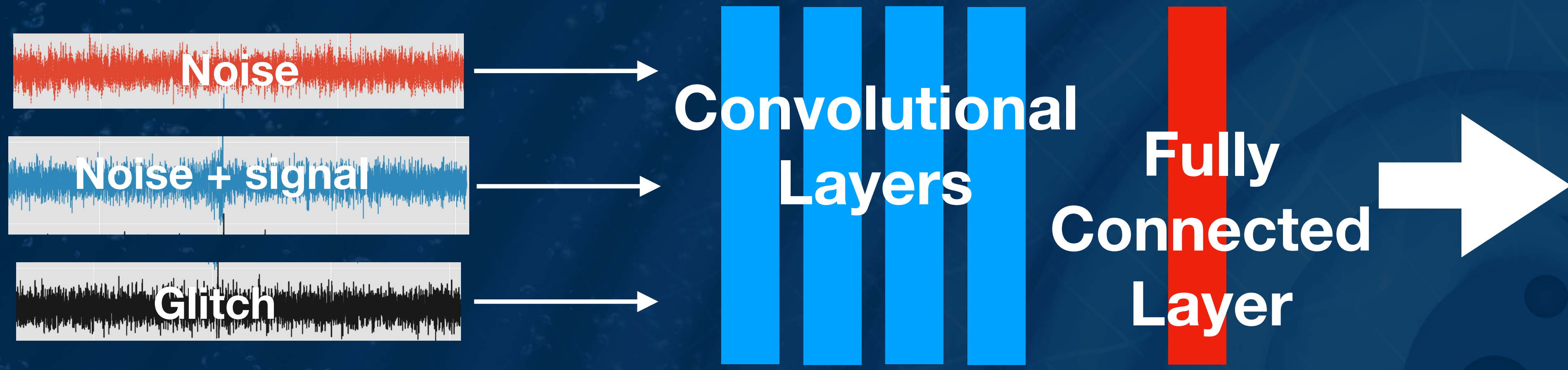
Training data: 3 classes

Segments of glitches and “clean” noise data samples from the one month of LIGO O1 run (downsampled to 2048 Hz), whitened by the amplitude spectral density of the noise.



Network used

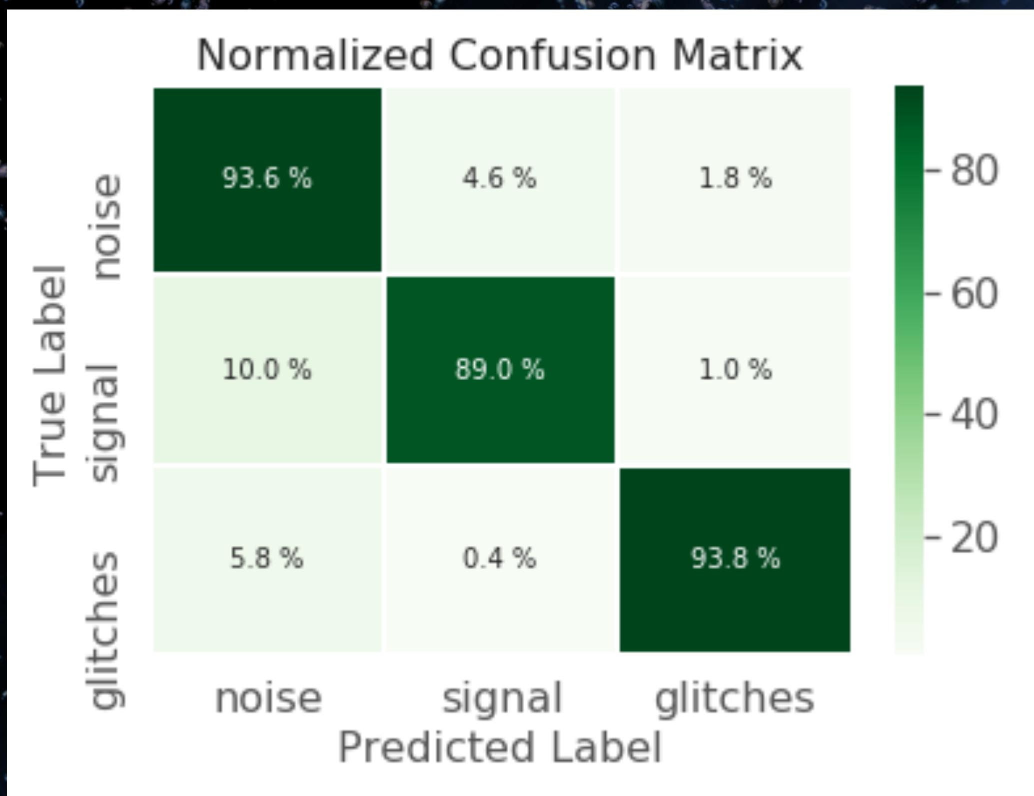
- CNN used: small network with 4 convolution layers (with dropouts and pooling) used as classifier to distinguish the 3 classes: noise, noise+signal, glitches



Output: probability of belonging to each class ➜ class chosen according to the higher probability

Layer #	1	2	3	4	5
Type	Conv	Conv	Conv	Conv	Dense
Filters	64	32	16	8	-
Kernel Size	16	8	8	4	-
Strides	4	2	2	1	-
Activation	relu	relu	relu	relu	softmax
Dropout	0.5	0.5	0.25	0.25	-
Max Pool	4	2	2	2	-

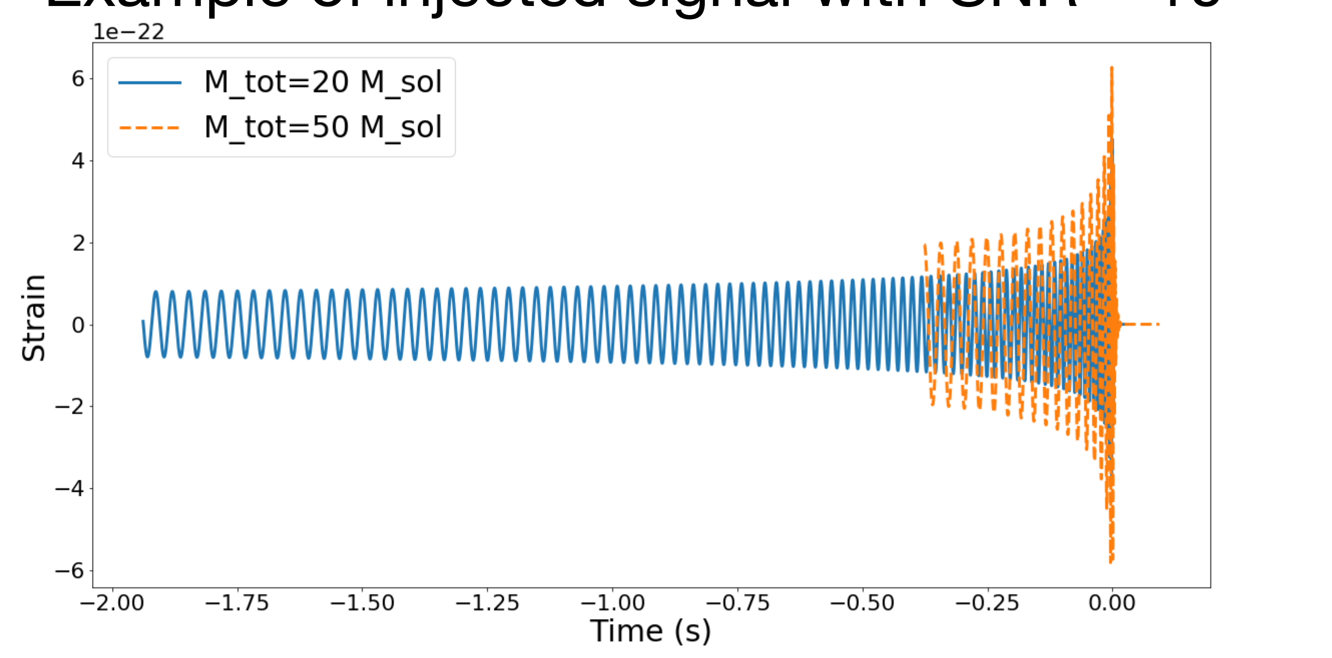
Confusion matrix and dataset details



Additional dataset details

- **Segments length: 1 second**
- **Injected signals (BBH)**
  - $m_1+m_2 \in (33,60)\ M_\odot$
  - SNR $\in$ (8,20)
- **Selected glitches**
  - SNR > 10

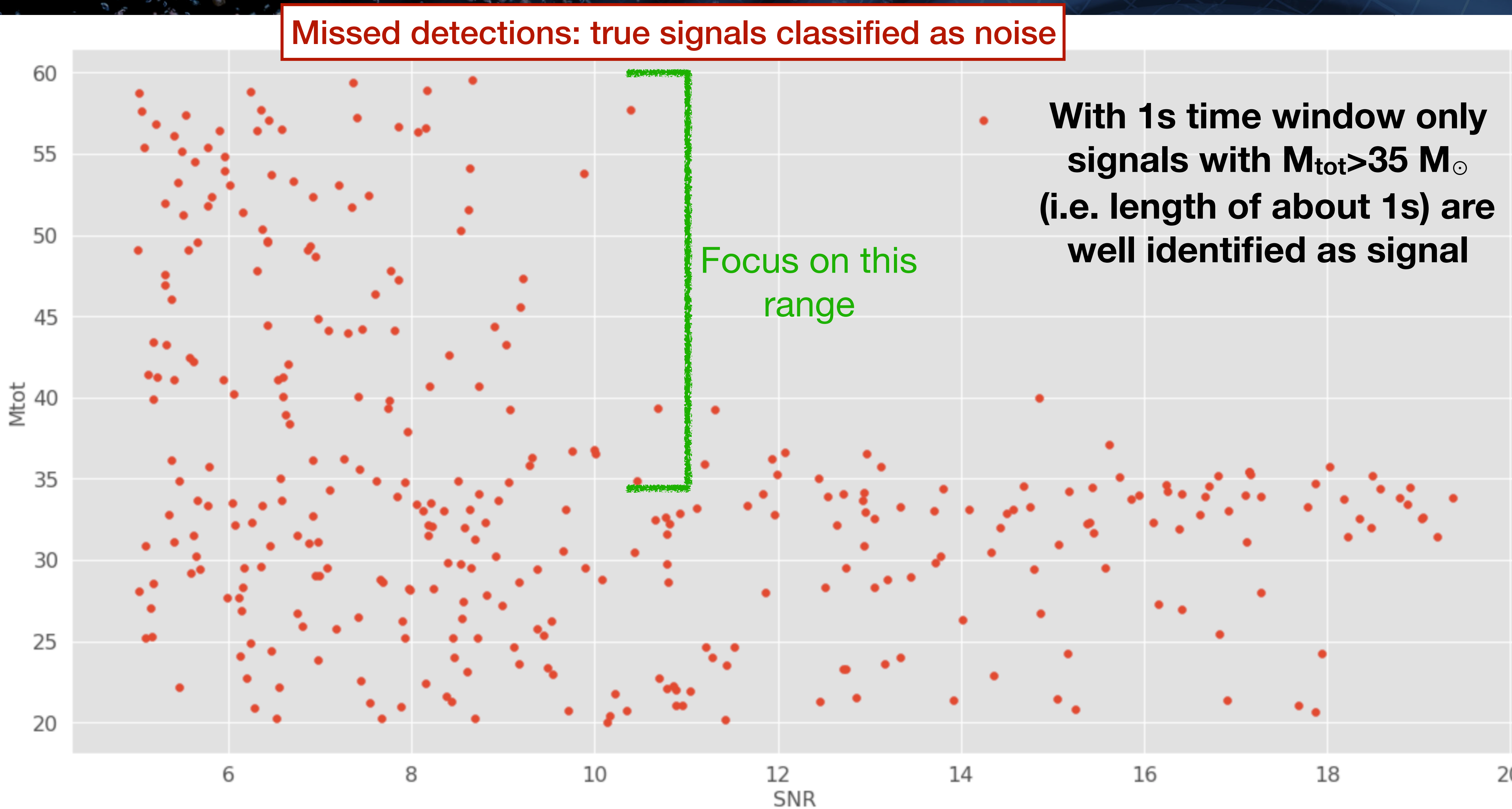
Example of injected signal with SNR = 10



Detectability across the parameter space

1 s time window

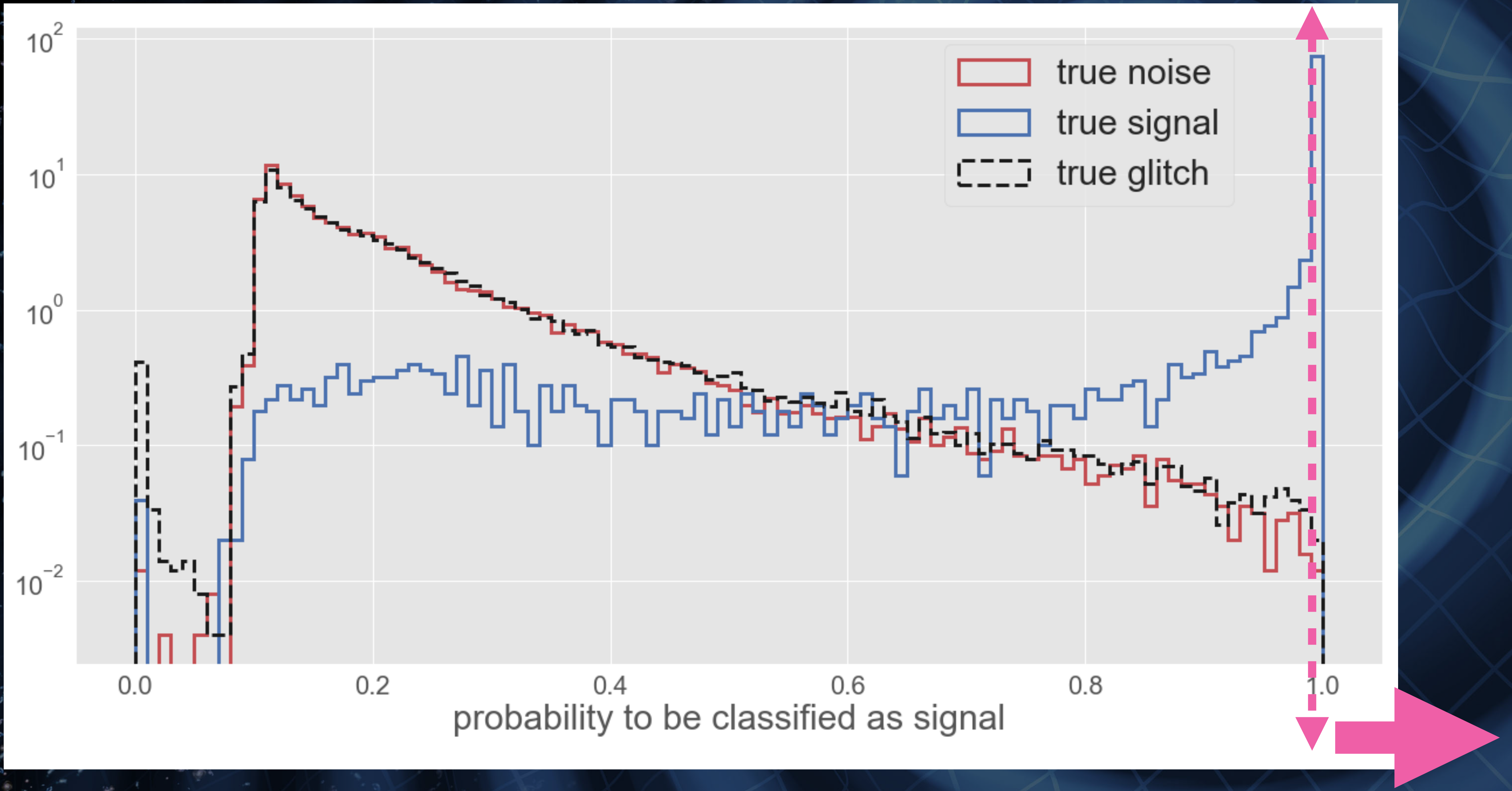
Signal with $m_{tot} \in (20, 60)\ M_\odot$



With 1s time window only signals with M_{tot}>35 $M_\odot$ (i.e. length of about 1s) are well identified as signal

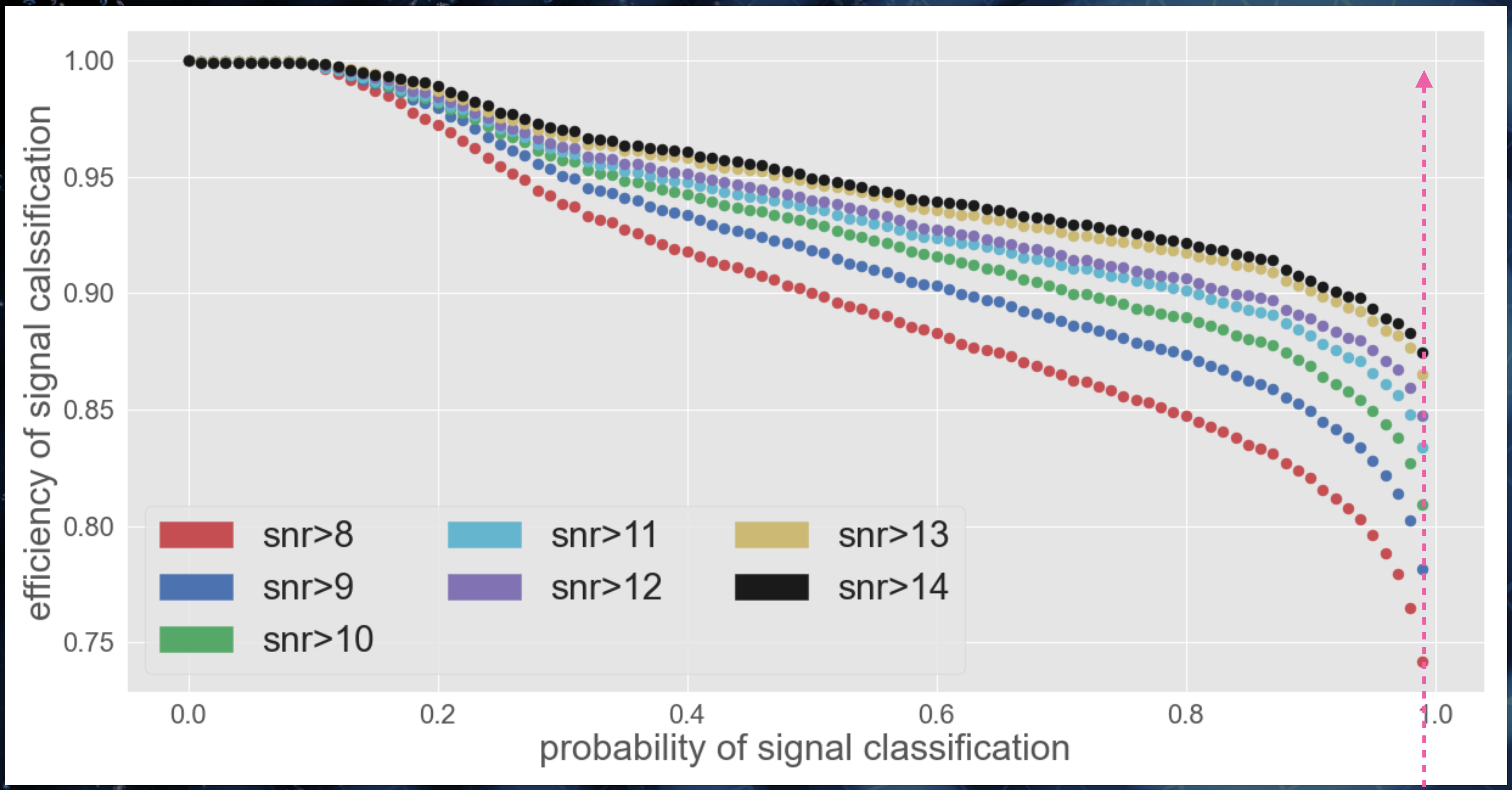
Probabilities of classification

Test: use the probability of the signal classification as statistic to distinguish signal vs noise+glitches



Efficiency

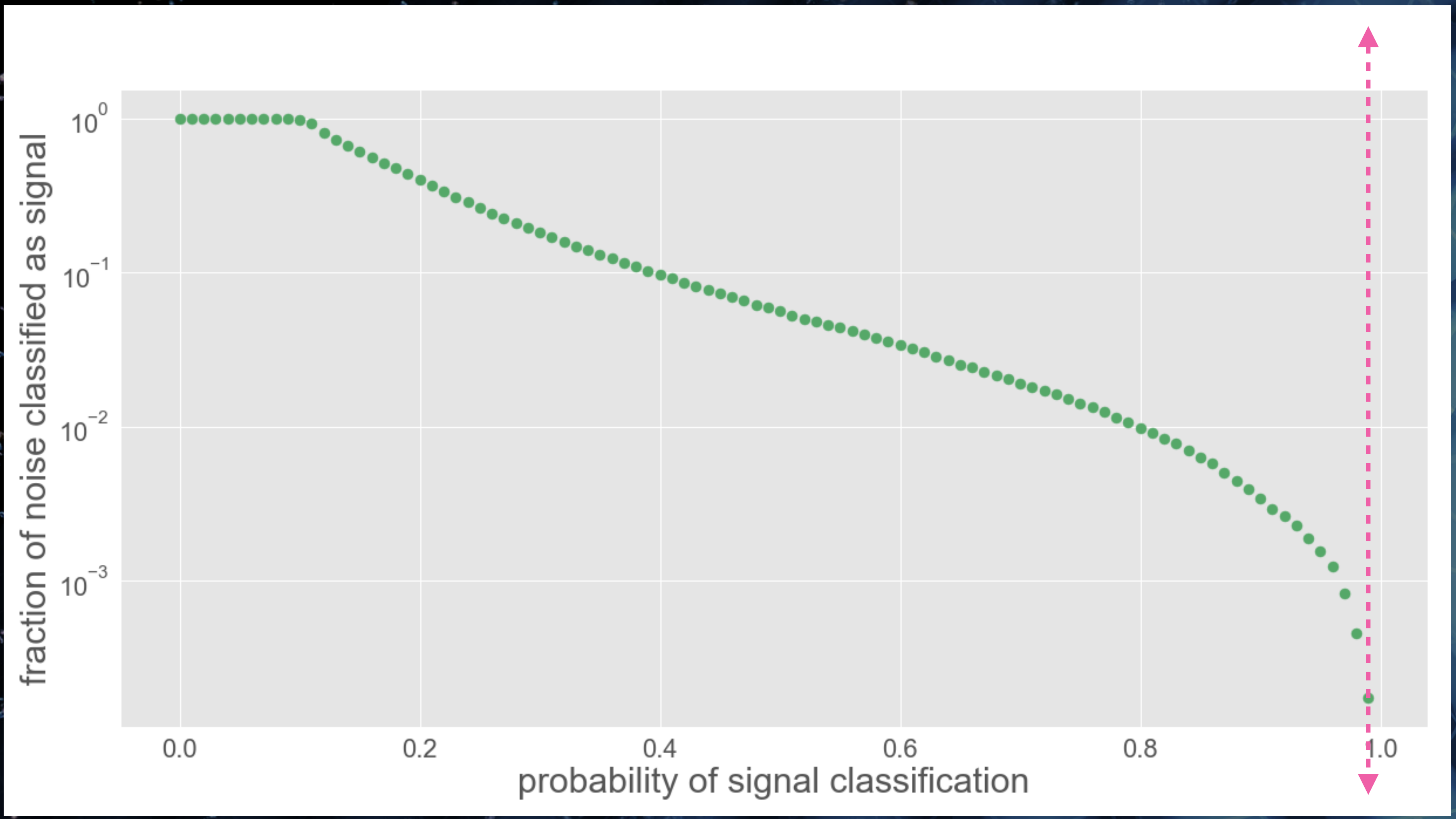
Efficiency = Fraction of signal well classified w.r.t. all the signals present in the dataset



With a stringent cut on probab_signal>0.99, reasonable efficiency around 80-90% for signals with SNR>10

False Alarm Rate

FAR = Fraction of noise+glitches classified as signals w.r.t. all the noise+glitches present in the dataset



With a stringent cut on probab_signal>0.99, FAR ≈ 1/83 min) ➜ this means about 2000 false alarms in O1!

Noise rejection is too limited

Trigger pre-selection (rather than high-confidence detection)

Results similar to other works on the subject

e.g. arXiv:1904.08693*, 1701.00008**, arXiv:1711.03121

*FAR of 1/40 minutes with detection ratio of 86%

**FAR of 0.6% and 100% sensitivity for SNR>10

Conclusion and perspectives

- GW signal classifier from single-detector time-series
  - ✓ Able to reach correct classification to the percent scale
  - ✓ However, not sufficient for high-confidence detection (too many false alarms)
  - ✓ Due to large class imbalance in the observations (signal very rare, noise very common)
- Can noise rejection be improved? Can we optimize the CNN with this objective specifically?
  - ✓ Focus on the imbalance between classes
  - ✓ Explore different metrics: e.g. max Recall at fixed Precision
  - ✓ Suggestions are welcome


Backup slides

Precision and recall

Choose a relevant class: e.g. signal

$$\text{Precision} = \frac{tp}{tp + fp}$$

Fraction of signal well classified w.r.t. those classified as signal

$$\text{Recall} = \frac{tp}{tp + fn}$$

Fraction of signal well classified w.r.t. all the signals present in the sample

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

can be a misleading metric for imbalanced data sets

CNN with different settings

- CNN used: small network with 4 convolution layers (with dropouts and pooling) used as classifier to distinguish the 3 classes: noise, noise+signal, glitches

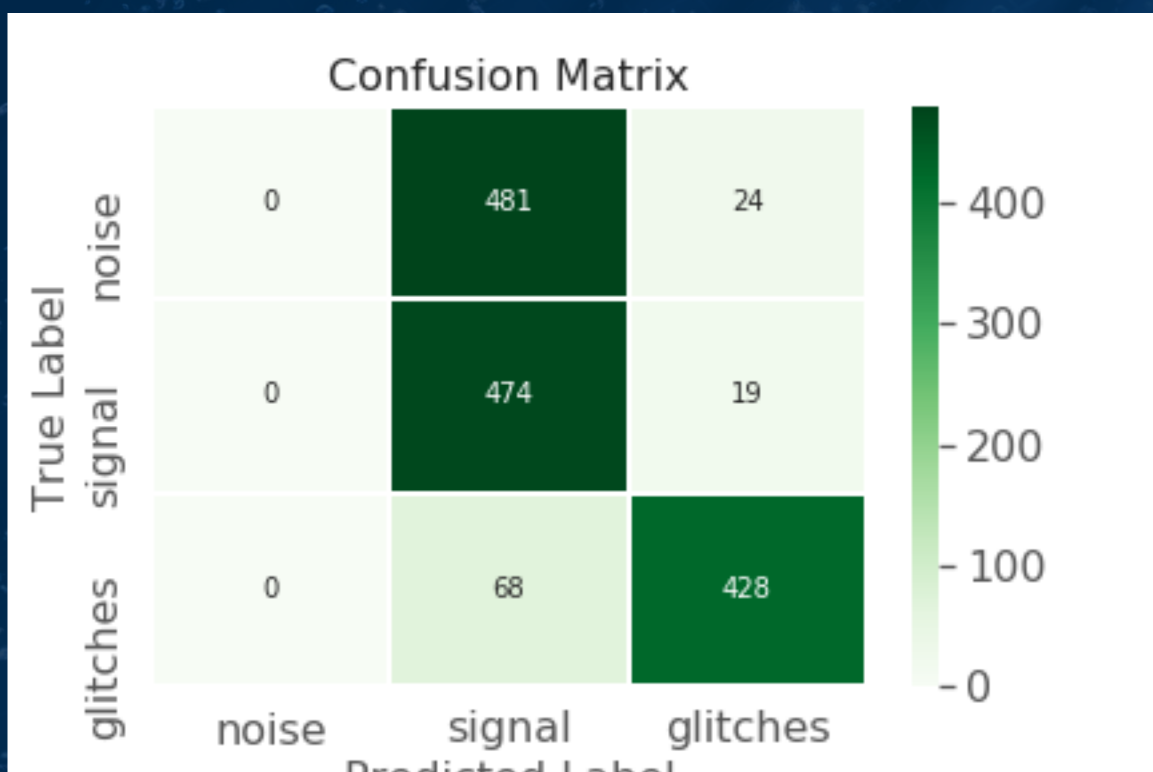
- We tested the CNN changing:
  - ✓ Length of the input time window: 4 s, 2 s, 1 s
  - ✓ Parameters of the injected BBH signal: mainly masses and SNR
  - ✓ Cuts of the SNR of the glitches
  - ✓ Parameters of the network itself (# filters, kernel size, stride, dropout, pooling, …)
  - ✓ Tested various decision statistics from which an "answer" is obtained

I'll show the effect of some of these tests

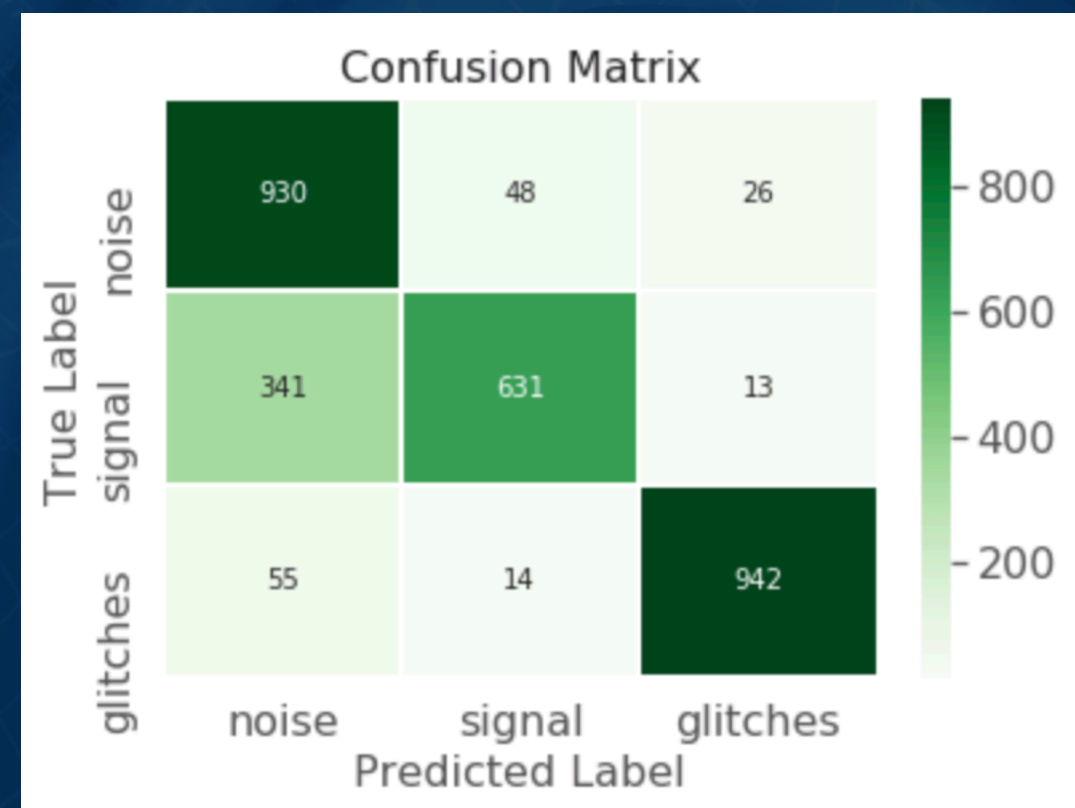
Length of the time window (I)

- Length of the time window (= size of the input data segment) coupled with the masses of the simulated signals
  - ✓ Signals with m_1, $m_2 \in (10, 30)\ M_\odot$, $5<\mathrm{SNR}<20$; glitches with SNR>10

4 s time window



1 s time window



CNN: Conv1D (500, 5) | MaxPooling1D (3) | Conv1D (250, 5) | Conv1D (500, 5) | MaxPooling1D (3) | Conv1D (150, 5) | MaxPooling1D (3) | Dropout (0.5)

Tuning the network

Network that seems to work better

```
Layer (type)                 Output Shape              Param #
=================================================================
reshape (Reshape)            (None, 2048, 1)           0
_________________________________________________________________
conv1d (Conv1D)              (None, 509, 64)           1088
_________________________________________________________________
dropout (Dropout)            (None, 509, 64)           0
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 127, 64)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 60, 32)            16416
_________________________________________________________________
dropout_1 (Dropout)          (None, 60, 32)            0
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 30, 32)            0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 12, 16)            4112
_________________________________________________________________
dropout_2 (Dropout)          (None, 12, 16)            0
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 6, 16)             0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 3, 8)              520
_________________________________________________________________
dropout_3 (Dropout)          (None, 3, 8)              0
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 1, 8)              0
_________________________________________________________________
global_average_pooling1d (Gl (None, 8)                 0
_________________________________________________________________
dropout_4 (Dropout)          (None, 8)                 0
_________________________________________________________________
dense (Dense)                (None, 3)                 27
=================================================================
Total params: 22,163
Trainable params: 22,163
Non-trainable params: 0
_________________________________________________________________
None
```

Tuning class weights

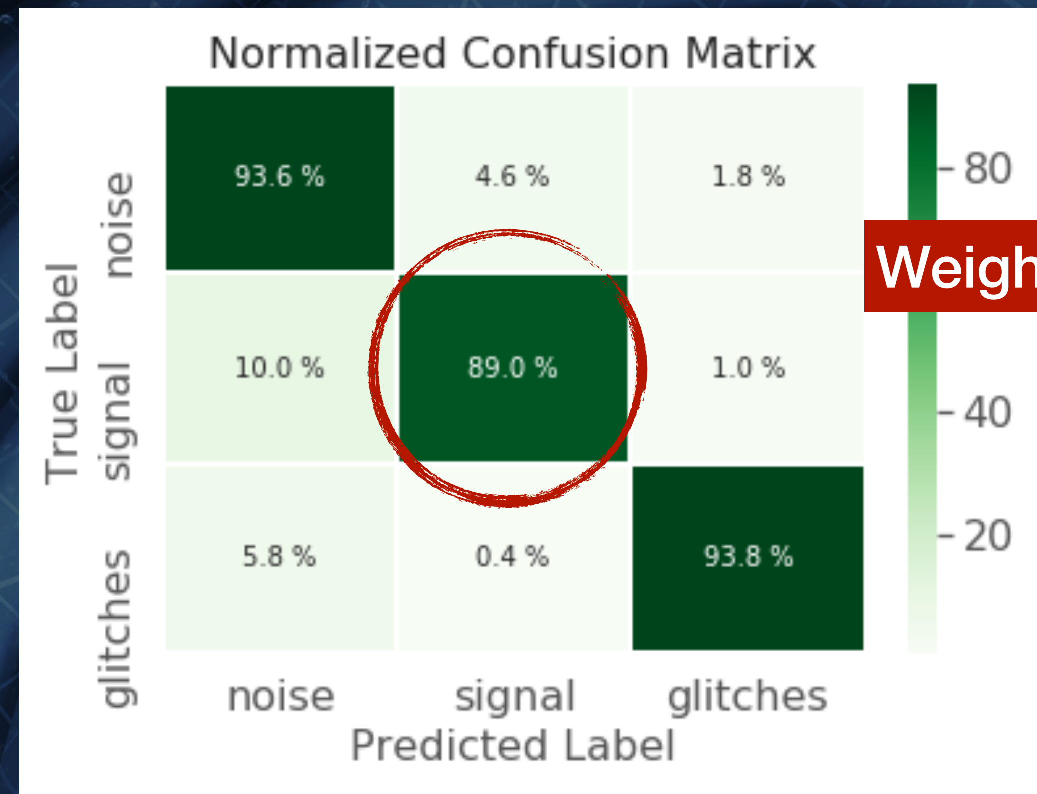
- ★ **Unbalanced datasets**: identification of the signal which is a rare event
- ★ **class_weights**: the loss function assign higher value to the classes with higher weight, i.e. the loss becomes a weighted average, where the weight of each sample is specified by class_weight and its corresponding class.

Tuning the network

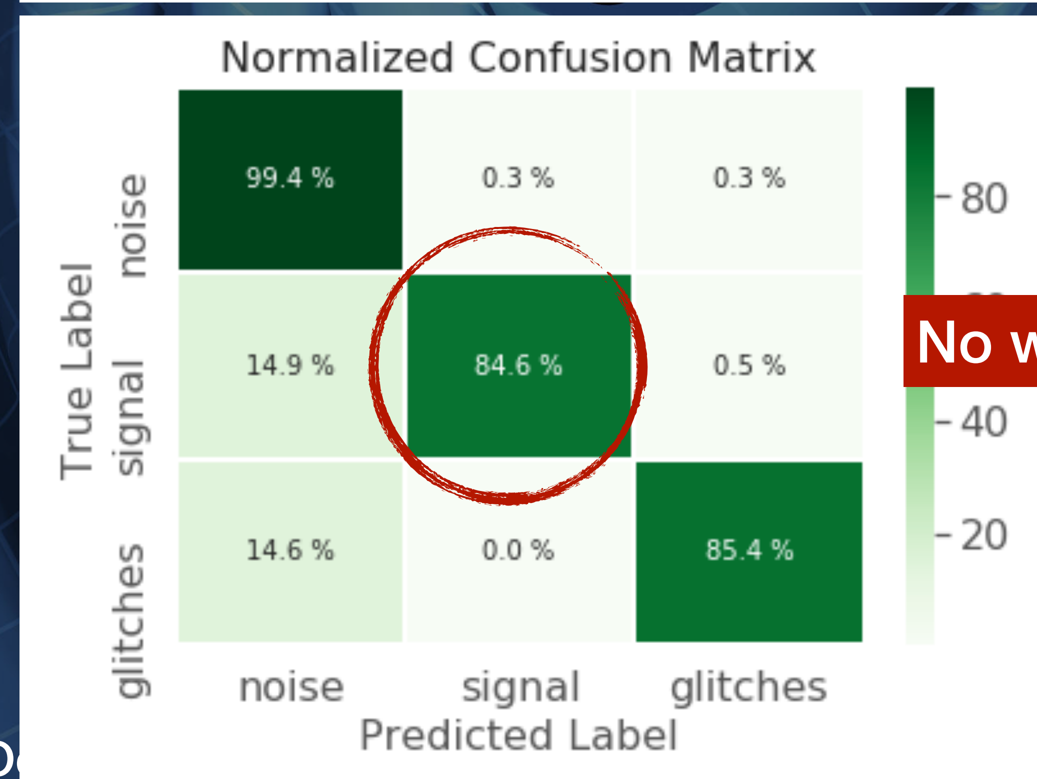
Network that seems to work better

```
Layer (type)                 Output Shape              Param #
=================================================================
reshape (Reshape)            (None, 2048, 1)           0
_________________________________________________________________
conv1d (Conv1D)              (None, 509, 64)           1088
_________________________________________________________________
dropout (Dropout)            (None, 509, 64)           0
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 127, 64)           0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 60, 32)            16416
_________________________________________________________________
dropout_1 (Dropout)          (None, 60, 32)            0
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 30, 32)            0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 12, 16)            4112
_________________________________________________________________
dropout_2 (Dropout)          (None, 12, 16)            0
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 6, 16)             0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 3, 8)              520
_________________________________________________________________
dropout_3 (Dropout)          (None, 3, 8)              0
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 1, 8)              0
_________________________________________________________________
global_average_pooling1d (Gl (None, 8)                 0
_________________________________________________________________
dropout_4 (Dropout)          (None, 8)                 0
_________________________________________________________________
dense (Dense)                (None, 3)                 27
=================================================================
Total params: 22,163
Trainable params: 22,163
Non-trainable params: 0
_________________________________________________________________
None
```

Tuning class weights



Weights: 1,2,1



No weights

Length of the time window (II)

A network working with windows of 1s could be combined with another one with 2 s windows, each optimised for different ranges in masses

True signals classified as noise (prob to be noise higher that the other prob)