Final Project

Suryarao Bethapudi

August 2020

1 PART I

Why do we need a network of detectors to detect a GW source? Gravitational waves comprise of

1. Modeled waveforms
2. Unmodeled waveforms

.

Modeled waveforms are such which are fully parameterized. They are derived analytically. Search for such waveforms would involve performing a grid search or sophisticated searches (like Particle Swarm Optimization) for that set of parameters which closely fit the observed waveform.

Unmodeled waveforms, on the other hand, are not analytically defined. Hence, can't be searched for by finding the parameters. This idea of using correlation (network of detectors) is not limited to unmodeled waveforms, but also to the modeled waveforms. Hence, a network of detectors is much more desirous to detect GW sources.

What are the different frequency bands of GW and how are they detected? See Tab. 1. SMBH stands for Super Massive BlackHole. LISA ([1]), PTA ([2]), DECIBO ([3]).

What is discrete white noise? What does psd of discrete white noise look like? Discrete white noise is white noise which has been discreted. White noise is any random noise which has equal power at all frequencies. In other words, the power spectral density is constant for all frequencies ($(-\infty, \infty)$). Discretization which is digitization (or sampling) brings a small bandwidth of $[\nu_{\text{LOW}}, \nu_{\text{HIGH}}]$. PSD of discrete white noise looks like a constant for all frequencies in the band.

What do you understand by a digital filter? Why is digital filtering necessary in LIGO data analysis? A digital filter takes a discrete time sampled signal and transforms it into another discrete time sampled signal such that certain properties of the signal are either enhanced or attenuated.

Digital filtering is of paramount importance because LIGO is an extremely sensitive instrument and can pick up all kinds of noise and signals from spurious sources. Digital filtering enables to process and analyse the data in such a way which culls all the noises and makes the hidden signal visible.

[1] https://lisa.nasa.gov/
[2] https://link.springer.com/article/10.1007/s00159-019-0115-7
[3] https://iopscience.iop.org/article/10.1088/1742-6596/122/1/012006

Freq	Experiments	Sources
1-1000 nHz	Pulsar Timing Arrays	Inspiraling super massive binary blackhole system
0.1 - 1000 mHz	Laser Interferometer Space Antenna (LISA)	Ultra-compact binaries, extreme mass ratio inspirals
0.1 - 10 Hz	Deci-hertz Interferometer Gravitational wave Observatory (DECIGO)	Characterization of dark energy, formation mechanism of SMBH at center of galaxies, verification and characterization of inflation
10 - 1000 Hz	Light Interferometer Gravitational wave Observatory (LIGO)	Binary Black Hole mergers, Neutron star mergers

Table 1: Different frequency bands of GW and experiments targeting each

How do you define signal-to-noise ratio (SNR) in general? For a matched filter? SNR is defined as ratio of signal power to noise power. Usually, this is where the definition stop. To proceed further, we need to understand the kind of signals we are interested in detecting. For instance in case of modeled signals, the signal power is typically estimated to be the square of the amplitude. Noise power is typically estimated from the residuals left after subtracting the modeled signal from the data.

Matched filter is an optimal filter which given a signal maximizes the SNR (or its peak value) when sampled at the period of the signal and when the noise is purely additive and wide-sense-stationary.

Write down 5 reasons why it is important to analyze and mitigate glitches in GW data? Five reasons are:

False positives Detection of glitches can prevent any detection of glitches to be treated as real signal of astrophysical nature.[4] Although glitches at individual observatories are independent, given the time window of the coincident search, if any glitch is registered at both the observatories in that time window, it would be treated as a real signal. Hence, it can reduce false positives.[5]

Improving GW noise understanding[4] Glitch detection can improve GW noise modeling which can improve statistical limits on positive detection[6].

Debugging[6] Understanding these glitches can potentially help identify the source behind the glitch which can lead to further improvement in the sensitivity.

Reduce amount of usable data[5] Glitches renders the data unusable for any astrophysics search effort.

Reduce the significance of events[5] Glitches that occure around the same time as a real astrophysical event causes larger uncertainities in the parameters than in their absense.

Increase noise floor[5] Glitches increase the background noise of the data which reduces the significance of the detected events.

Can you name 7 different sources of noise that show up in a gravitational wave data from terrestrial detectors? The seven noise sources are:

- Instrumental noise:
  1. Shot noise
  2. Suspension thermal noise
  3. Quantum sensing noise
  4. Mirror coating noise
- Environmental noise:
  5. Seismic noise
  6. Gravity Gradient noise
  7. Anthropogenic noise
  8. Weather
  9. Electrical and magnetic disturbances

[4]https://dl4physicalsciences.github.io/files/nips_dlps_2017_25.pdf
[5]https://arxiv.org/abs/1611.04596
[6]https://iopscience.iop.org/article/10.1088/1742-6596/243/1/012006/pdf

2 PART II

Project II

2.1 Dataset

Each waveform is a 1-second long data sampled at f_s sampling rate which is set to 2048 Hz. There are three classes in this dataset and each are described below. All distributions which are sampled from are Uniform distributions.

Mixed sine There are 500 elements in this class. Each element is a sum of 100 different sine waves with frequencies taken randomly from $[300, 600)$. Amplitudes are taken randomly from $[0.5, 3.0)$.

Chirp There are also 500 elements in this class. Each element is a chirp generated by `scipy.signal.chirp`. Start frequency (frequency of the chirp at the $T = 0$) is chosen randomly from $[200, 250)$ Hz. End frequency (frequency of the chirp at highest T possible) is chosen randomly from $[900, 1000)$ Hz. The amplitude is sampled randomly from $[0.5, 3.0)$. Lastly, the frequency sweep of the chirp is also randomly selected from 'linear', 'logarithmic', 'hyperbolic', and 'quadratic'.

Core collapse Supernovae(CCSN) CCSN GW waveforms which are provided by the instructor are used. It is noticed that magnitude of minimum is much much more than magnitude of maximum. This fact is used to find the 2048 samples of region where the signal's features are contained. From the least point of the signal, 0.2×2048 towards the left and 0.8×2048 towards the right mark the endpoints of the signal to be sliced. This way, 2048 samples can be sliced which contain all the features of the signal.

Machine Learnings has to be done on spectrogram (T-F planes), hence, these waveforms have to be processed to generate spectrograms. This method is described here. Spectrogram is performed using `scipy.signal.spectrogram`. The following optional arguments are used.

1. Sampling frequency = f_s
2. Number of samples per segment = $f_s/9$
3. Number of samples per FFT = $f_s/9$
4. Number of samples to be overlapped between segments = $f_s/10$

This takes in 2048 samples waveform and generates a TF-plane with dimensions 118×90. This is treated as the input to the CNN.

For generating noisy samples, Gaussian noise of zero mean and variance unity is added to the waveform and then corresponding TF-planes are generated.

One element of each class for both noiseless and noisy datasets are plotted in Fig. 1.

2.2 CNN

CNN is expected to take in an image plane of $1 \times 118 \times 90$ and expected to return out three probabilities for each class. The architecture of the CNN is described in Fig. 2. Each convolutional (`conv2d`) layer is followed by a Leaky-Rectifier[7] as the non-linear activation function and Dropout[8,9] to prevent overfitting.

2.3 Training

Training is done in mini-batches of 10. Adam[10] is the optimizer used. Since this is a multi-class classification problem, the loss function used is CrossEntropyLoss.

2.4 Code

The following files are submitted with this report:

`create_ms.py` Python script which generates mixed-sine spectrograms and time series data

`create_chirp.py` Python script which generates chirp spectrograms and time series data

`create_ccsn.py` Python script which generates CCSN spectrograms

`runner.py` Python script which performs the training

. The Deep Learning framework used is PyTorch. Training is done entirely on CPU.

The code and the trained models can be found in https://github.com/shiningsurya/gwa_final.

[7] https://ai.stanford.edu/ amaas/papers/relu_hybrid_icml2013_final.pdf
[8] https://arxiv.org/abs/1207.0580
[9] https://arxiv.org/abs/1411.4280
[10] https://arxiv.org/abs/1412.6980

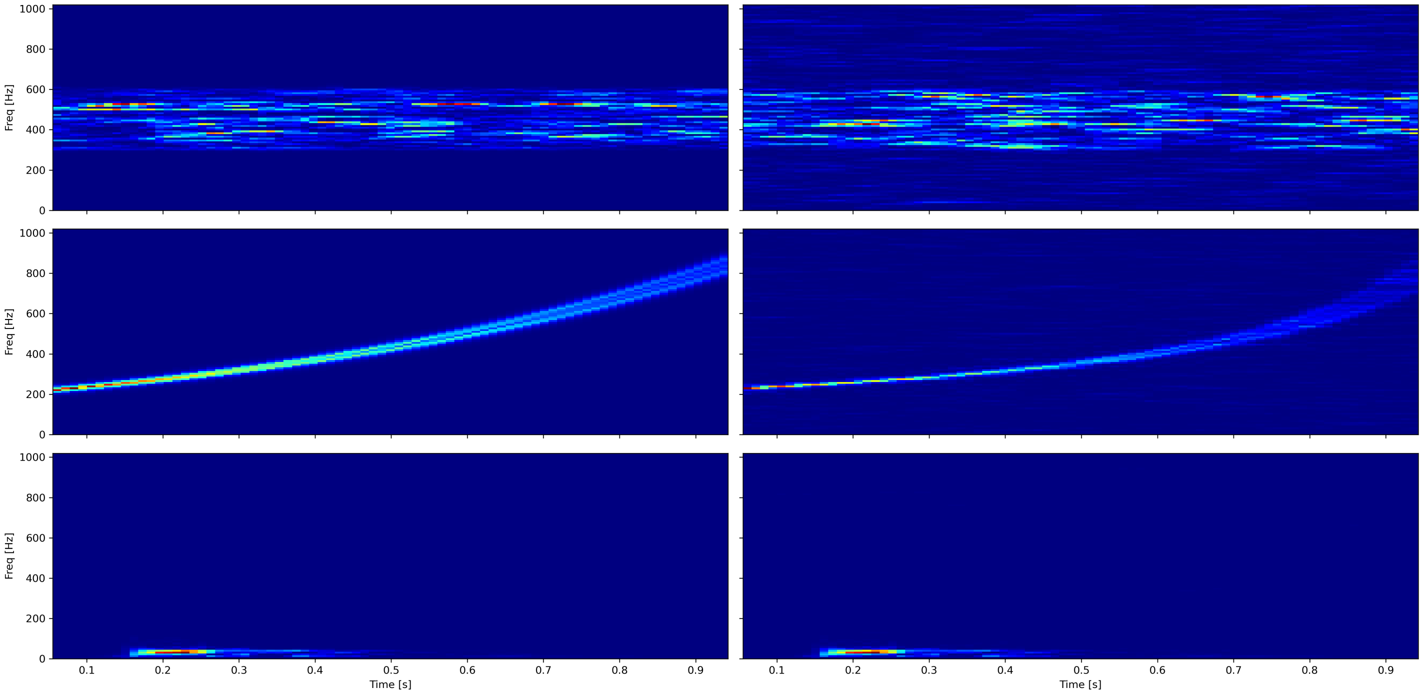

Figure 1: A sample set of TF-planes used as input for the CNN. *Top row:* Mixed sine signals. *Middle row:* Chirp signals. *Bottom row:*CCSN signals. *Left:* Noiseless elements. *Right:* Noisy elements.

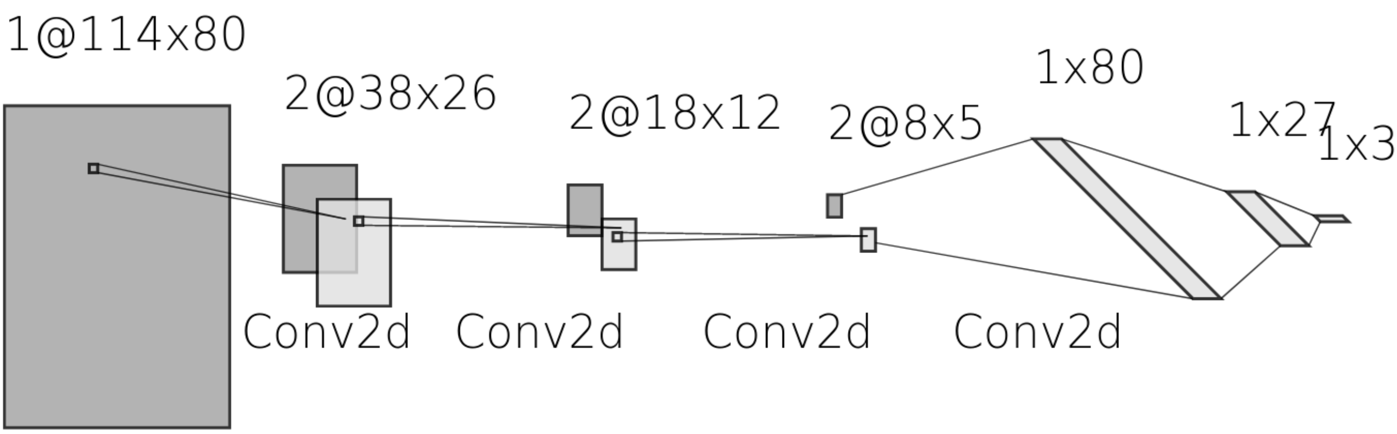

Figure 2: Convolutional Neural Network used here. Each layer is followed by Leaky ReLU as activation and Dropout is used against overfitting.

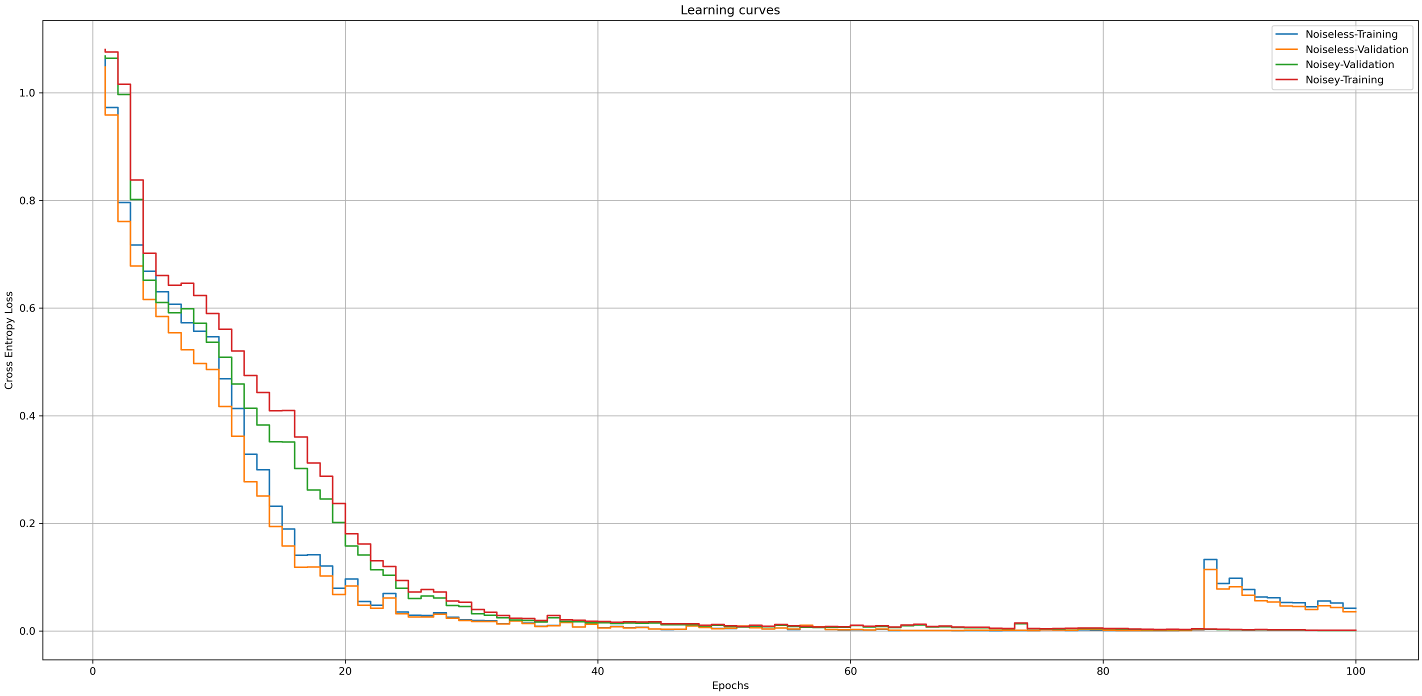

Figure 3: Learning curves. X-axis is epochs and Y-axis is cross entropy loss. Noiseless curves converge quicker than the noise-y curves. Noiseless curves produce less loss early on.

2.5 Results

The learning curves are plotted in Fig. 3. The accuracy curves are plotted in Fig. 4. CNN achieves 100% recall and precision as well.

Classification has not been affected by the noise. We still achieve 100% accuracy however we see that the learning takes a bit more epochs for the noisy dataset (see Fig. 3). This is the power of CNN.

The same is held true in accuracy (see Fig. 4). Noiseless accuracy is more than noisey until converge is achieved.

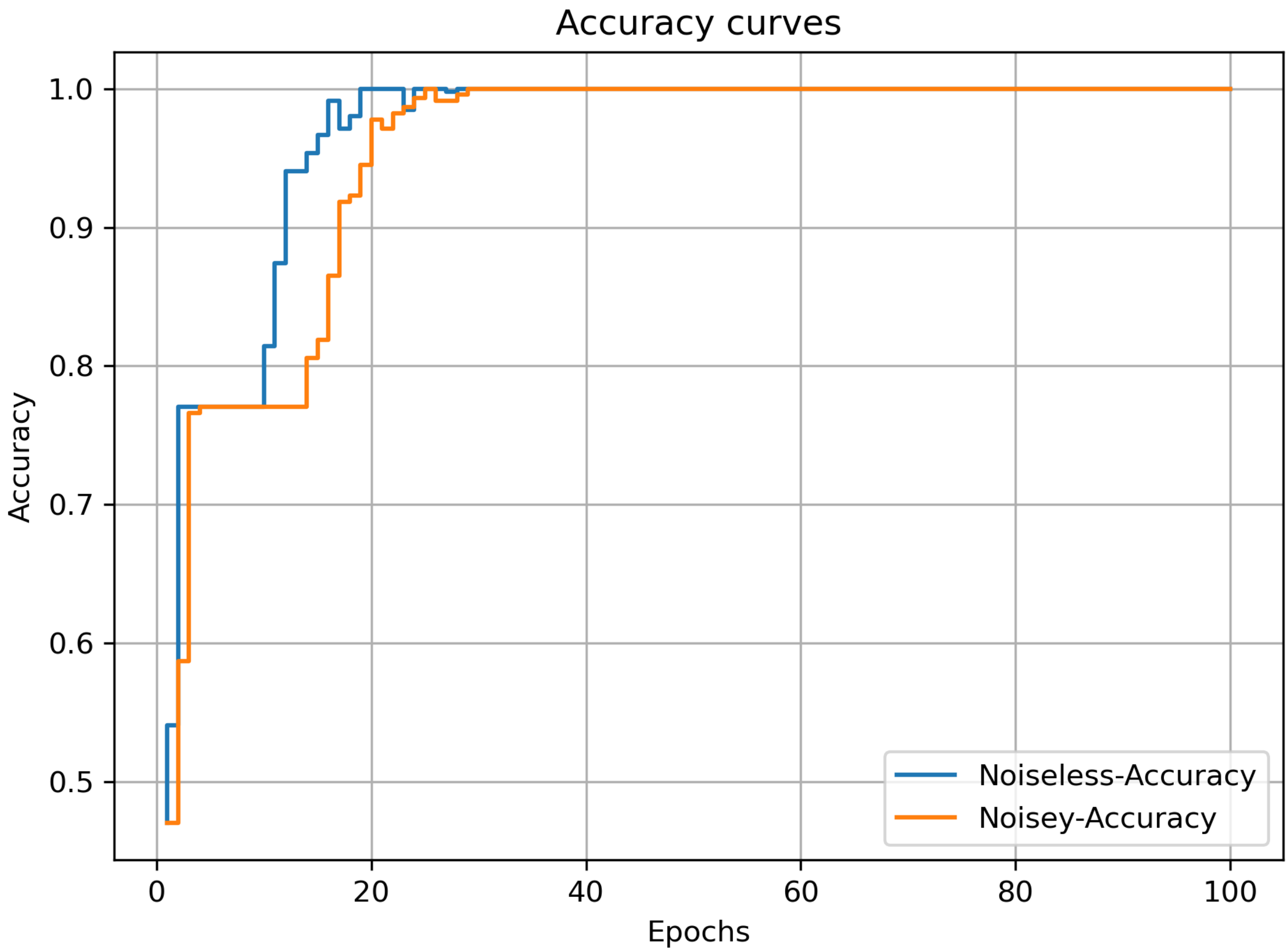

Figure 4: Accuracy curves. X-axis is epochs and Y-axis is accuracy on validation sets (25% of the data). We see that noiseless accuracy is more than noisey until they both converge.