CS-235 Midterm Report

Kuntal Pal

November 8, 2021

1 Playing with data

The dataset to be used for the project was released as part of the kaggle competition [1] for identifying gravitational wave signals from binary black hole mergers. The training consist of time series data containing simulated gravitational wave measurements from a network of 3 gravitational wave interferometers (LIGO Hanford, LIGO Livingston, and Virgo).

Each data sample (npy file) contains 3 time series (1 for each detector) and each spans 2 sec and is sampled at 2,048 Hz. There are in total 560000 training samples and 226000 test samples. The training samples are labeled 0 (background) and 1 (signal+noise). The test samples are not labeled as competition. All the npy files are combinned to a single .h5 file which reduces the storage space by half and also loads faster.

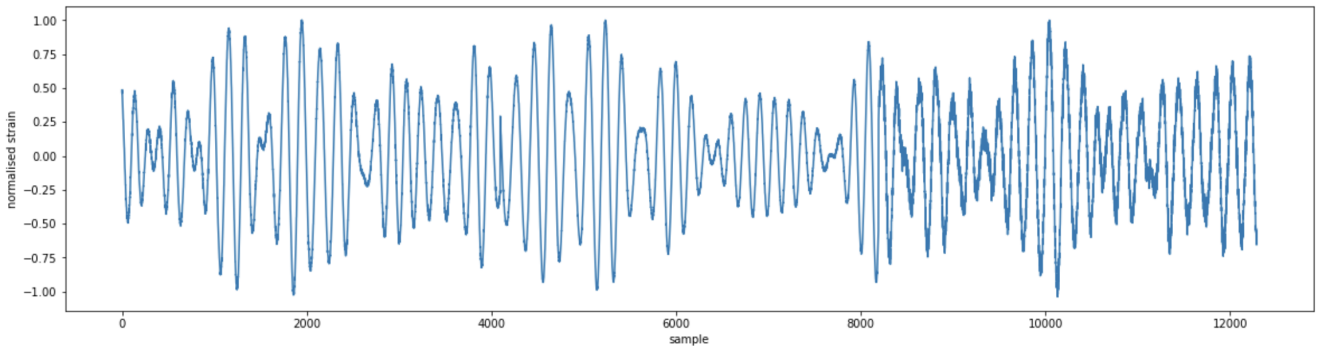

Figure 1: Combined time series from 3 detectors with label y=1.

The data sample, in total pretty huge (close to 80 gb). So the idea is to run each of the algorithms on a fraction of the dataset on my local computer and then run with the full dataset on the UCR cluster (most likely on the GPU).

Given that gravitational wave signals are notoriously difficult to detect, the first intuition was that preprocessing the data may be a significant part of analysing the data.

1.1 Bandpass filter

First, each time series from three detectors are normalized with respect to the maximum value in the series and then joined together to form a single univariate time series as shown in Fig.(1). The the signal is passed through an order 8 butterworth filter in the range [50,500] Hz. The choice of the filter and the range is decided after going through filtering techniques for gravitational waves in the literature (see for example [8, 9])

1.2 Gramian Angular Field and Markov Transition Field

Encoding time series to images is a pretty common technique used for time series classification. Following [3], I tried to implement two such techniques namely, Gramian Angular Field, which is a polar coordinate

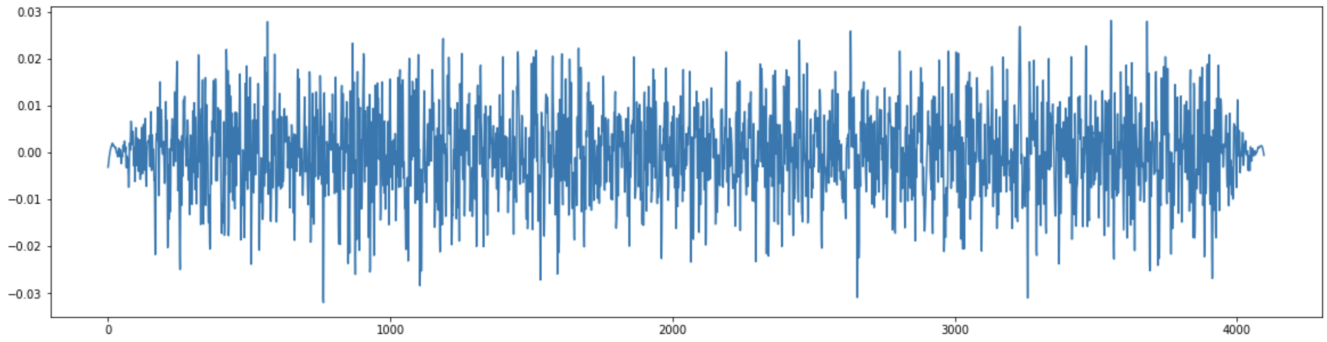

Figure 2: Signal after filtering with label y=1.

representation of the time series and Markov Transition Field, where the idea is to build the Markov matrix of quantile bins after discretization and encode the dynamic transition probability in a quasi-Gramian matrix.

A single crucial observation can be that no signal "jumps out" to the naked eye, by considering various time-domain / frequency-domain plots. This highlights the non-trivial nature of this data science problem. Any further suggestions for time series data processing that I may have overlooked and should be considered are welcome.

2 Project workflow so far

Following the timeline of the project proposal, the first two algorithms to implement were Support Vector Machines and XGboost. I have decided not to implement either of the two. The reason being that, for SVM in general, the training complexity grows with the size of the dataset and and hence may not work for our case. I tried kernel SVM with a small subset of the data (10k samples) but the performance was poor and almost identical to simple random guessing. XGBoost on the other hand, works very good on tabular data but not so much on with time series unless there is a clear way of feature extraction from the time series data, which may be possible but given the time constraints, I decided to not get into it for now.

So the goal for the project, is to entirely focus on diverse neural network architectures for classification. The idea is to implement atleast the following three methods.

- A Resnet/CNN architecture.
- A type of RNN (most likely [4] as mentioned in the proposal)
- A transformer architecture (For example [5] which was also mentioned in the proposal.)

So far, with the aim to build a Residual network, I started with simple 1D-CNN architecture (shown below) to get a hang of how CNN works in general as I have almost no experience with it before. Next the plan is to use 1D-Resnet to train directly with the time series data and a 2D-Resnet to train with the transformed images and then compare the performance.

References

[1] G2Net Gravitational Wave Detection, Find gravitational wave signals from binary black hole collisions, https://www.kaggle.com/c/g2net-gravitational-wave-detection/data

[2] Gravitational Wave Data Analysis with Machine Learning, https://iphysresearch.github.io/Survey4GWML/

[3] Wang, Zhiguang and Tim Oates. "Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks." (2014).

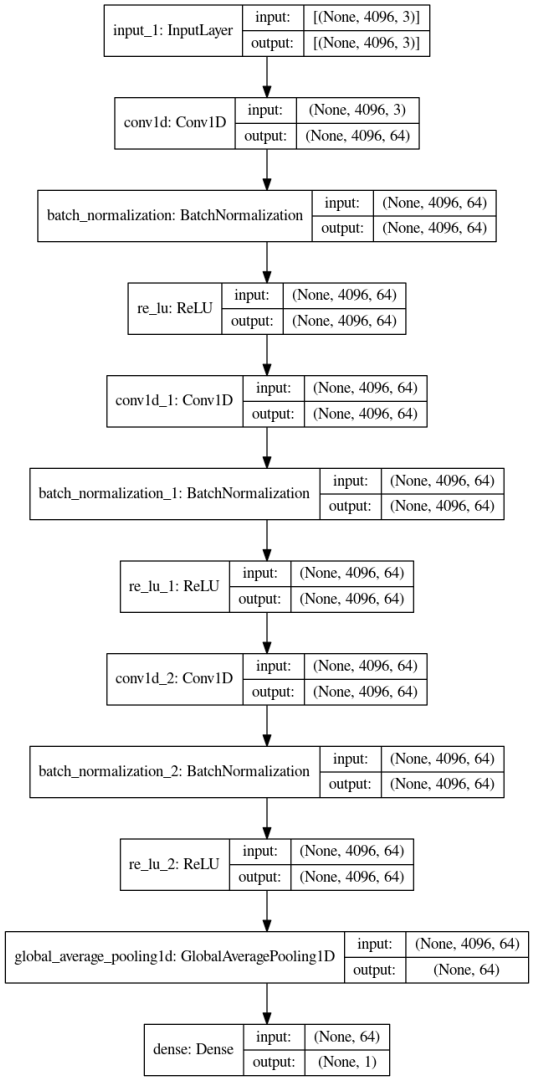

Figure 3: The CNN architecture

[4] Chang, S., Zhang, Y., Han, W., Yu, M., Guo, X., Tan, W., ... & Huang, T. S. (2017). Dilated recurrent neural networks. arXiv preprint arXiv:1710.02224.

[5] Liu, M., Ren, S., Ma, S., Jiao, J., Chen, Y., Wang, Z., & Song, W. (2021). Gated Transformer Networks for Multivariate Time Series Classification. arXiv preprint arXiv:2103.14438.

[6] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

[7] Gravitational Wave Data Analysis with Machine Learning, https://qiskit.org/textbook/ch-machine-learning/machine-learning-qiskit-pytorch.html

[8] Convenient filtering techniques for LIGO strain of the GW150914 event, http://dx.doi.org/10.1088/1475-7516/2019/04/032, journal=Journal of Cosmology and Astroparticle Physics

[9] A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals, https://doi.org/10.1088/1361-6382/ab685e, publisher = IOP Publishing