Search for Gravitational Waves using Deep Neural Networks

Kuntal Pal
kpal002@ucr.edu
University of California, Riverside
California, USA

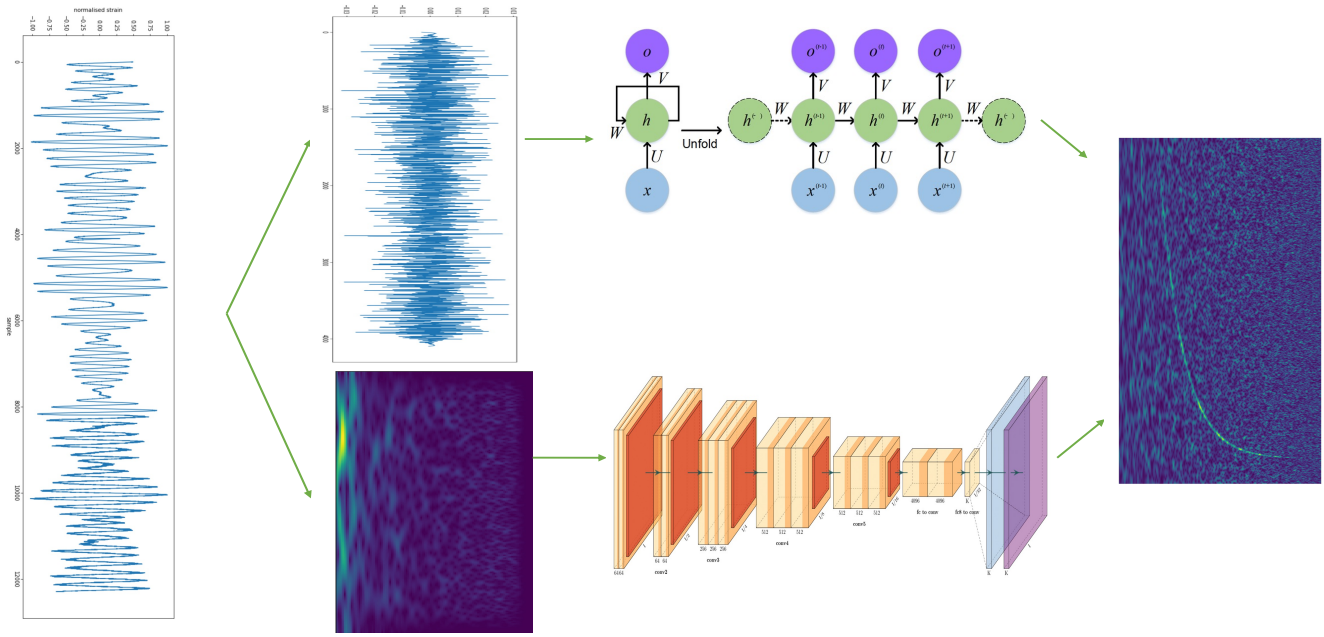


Figure 1: Project Workflow

ABSTRACT

Gravitational wave (GW) detection is one of the most landmark discoveries in physics and astronomy that opened new ways of understanding our universe. Detecting gravitational waves is an extremely difficult challenge due its unimaginably small amplitude buried under a plethora of background noise. In recent times, machine learning has played a crucial role in detecting and understanding gravitational waves. In this article, different deep learning models are trained and evaluated on artificially simulated GW signals from binary black hole merges and performance is evaluated in terms of successfully classifying signal data. Various preprocessing methods are also tried out to make the GW data trainable by neural networks.

,
© 2021 Association for Computing Machinery.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

KEYWORDS

Classification, Deep neural networks, Deep learning, CNN, LSTM, Encoder

ACM Reference Format:
Kuntal Pal. 2021. Search for Gravitational Waves using Deep Neural Networks. In ,. ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Gravitational waves(GW) were first predicted almost a century ago as a direct consequence of Einstein's general theory of relativity [7]. They are ripples in the fabric of spacetime that travel at the speed of light. Among the four fundamental forces of nature, gravity has the weakest coupling to matter (almost 10^{-35} times weaker than the electromagnetic force) and hence gravitational waves are so unimaginably tiny that Einstein himself doubted that they would ever be discovered. But contrary to his belief, the first gravitational wave was detected in 2015 by the LIGO collaboration [1]. The signal was generated from a binary black hole merger with a combined mass almost 60 times the mass of our sun and yet only recorded a peak magnitude strain of 10^{-21}. Since the breakthrough discovery, there have been numerous detections that have led researchers to observe a new population of massive, stellar-origin black holes,

to unlock the mysteries of neutron star mergers, and to measure the expansion of the Universe. Even with on of the most sensitive experiments on the planet, the signal is engulfed with detector noise of all kinds. New improved and more sensitive detectors are coming up in different parts of the world and with more data available, the challenges of noise separation and signal identification can be solved using data science.

Machine learning has played a crucial role in trying to solve the challenges in gravitational wave data analysis that include improving data quality, searches for binary black holes, neutron stars and unmodelled gravitational wave bursts, as well as trying to gain a better understanding of the astrophysics of gravitational wave sources [25]. In this project, the aim is to detect GW signals from the mergers of binary black holes using deep learning architectures to analyze simulated GW time-series data from a network of Earth-based detectors. The idea is to first implement a variety of preprocessing methods that help achieve state of the art results for time series data in general as well as try out some of the recent ideas suggested in literature. The data will then be fed to a diverse class of neural network architectures to train and evaluate a binary classification model to predict if the given set of signals has gravitational waves from binary black hole mergers in them or not. Some of the architectures and methods used in the project have only been proposed recently and others, though extensively used in other domains to achieve state of the art results, have not yet been used in the context of gravitational waves signals.

2 RELATED WORK

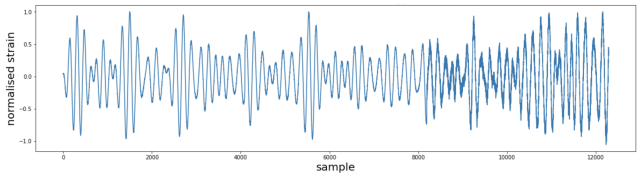
Machine learning has been used to try to solve a wide variety of challenges related GW physics [25]. In the context of binary black hole signals, matched filtering technique [4] remains a popular choice for detection. Deep learning, though only being used recently achieves similar performance compared to matched-filtering while being several orders of magnitude faster [2, 8–12, 15, 18, 20, 24]. In the context of recent times, machine learning is an exponentially growing field and new ideas and methods are being introduced on a daily basis. Thus, the aim of the project is to apply some of the newly proposed methods in the context of GW search optimization.

3 DATASET

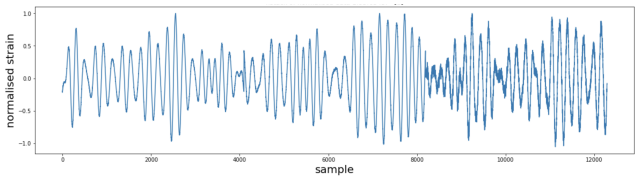
For this project, the dataset used was released by European Gravitational Observatory (EGO) as part of a kaggle competition [5]. The dataset consists of There are in total 560000 training samples and 226000 test samples. The training samples are labeled 0 (background) and 1 (signal+noise). The test samples are not labeled as it is part of the kaggle competition. Each data sample consists of a set of three time series data containing simulated gravitational wave measurements from a network of 3 gravitational wave interferometers (LIGO Hanford, LIGO Livingston, and Virgo) and each spans 2 sec and is sampled at 2,048 Hz. Each time series contains either detector noise or detector noise plus a simulated gravitational wave signal.

4 PREPROCESSING IS THE KEY

Given the tiny magnitude of GW signals and the amount of noise present, data preprocessing becomes an essential ingredient GW data analysis. Data visualization for samples labeled 0 and 1 revealed no obvious features to the naked eye as can be seen below.



(a) label = 0 (noise)



(b) label = 1 (signal+noise)

Figure 2: Normalized combined time series data for (a) label 0 (noise) and (b) label 1 (signal+noise)

However, as seen in both the sample plots, the strain data is a combination of many frequencies and analysing the signals in frequency domain, instead of the time domain, might provide better insights. It was decided to follow different approaches for preprocessing before feeding them to different neural network architectures.First, raw signal data is processed to enhance the signal, remove noise using whitening and/or filtering which is standard workflow in GW analysis[7]. Second, the time series data is converted into images either in the form of spectrograms in the frequency domain or some other techniques for transformation (for eg. Gramian Angular Field or Markov Transition field.)

4.1 Whitening

Whitening is one of the first steps in astrophysical data analysis as it requires no prior knowledge of the spectrum of the data. The process of whitening is basically trying to make the signal more uniform by suppressing the extra noise at low frequencies and at the spectral lines, to better see the weak signals in the most sensitive band.

If $\mathbf{X}$ represents a single time series data with covariance matrix Σ, then the whitened signal $\mathbf{Y}$ = W X with the whitening matrix W satisfying

$$W^T W = \Sigma^{-1} \tag{1}$$

The new whitened signal has identity as the new covariance matrix. Whitened data have the same power at all frequencies.

As suggested by the TA, the time series data provided are too short for whitening to have any impact and hence not used after a certain point in the analysis.

4.2 Bandpass filter

Filtering is simply removing unwanted frequencies from a signal. A bandpass filter selects a frequency within a certain range and rejects frequencies outside that range. The data available is sampled at 2048 Hz which implies the Nyquist frequency is 1024 Hz. Then the

filter range can be anywhere between 0 and 1024 Hz as sampling beyond the Nyquist frequency leads to aliasing. For the data from merger binary black holes, the frequency is in the lower range of 35 ∼ 350 Hz [1]. Keeping a safer estimate, the choice for the bandpass filter range is kept at 20 ∼ 500 Hz which is also a common choice in literature [27].

4.3 Short-time Fourier Transform

STFT transforms signal in the time-frequency domain by computing discrete Fourier transforms (DFT) over short overlapping windows. It is defined as

$$X[m,k) = \sum_{n=0}^{n=N-1} = x[n+mH]w[n]e^{\frac{-2\pi ikn}{N}} \tag{2}$$

where H is referred to as the hop length. The hop length parameter is specified in samples and determines the step size in which the window is to be shifted across the signal. X[m,k) is complex and denotes the k^{th} Fourier coefficient for the m^{th} time domain.

The spectrogram is given by

$$Y[m,k) = |X[m,k)|^2 \tag{3}$$

It can be visualized by means of a two-dimensional image, where the horizontal axis represents time and the vertical axis represents frequency. In this image, the spectrogram value Y[m,k) is represented by the intensity or color in the image at the coordinate (m,k).

4.4 Constant Q- Transform (CQT)

The idea of transforming time series data to frequency domain spectrograms was first suggested by the instructor which turned out to be quite useful. The constant Q transform as introduced in [3] is very close related to the Fourier transform. Like the Fourier transform a constant Q transform is a bank of filters, but in contrast to the former it has geometrically spaced center frequencies

$$f_k = f_0 \cdot 2^{\frac{k}{b}} (k = 0, ...) \tag{4}$$

where b is the number of filters per octave. The bandwidth of the k-th filter is cchosen as

$$\Delta_k^{cq} = f_{k+1} - f_k = f_k(2^{\frac{1}{b}} - 1) \tag{5}$$

This provides a constant ratio of frequency to resolution

$$Q = \frac{f_k}{\Delta_k^{cq}} = \frac{1}{(2^{\frac{1}{b}} - 1)} \tag{6}$$

The major advantage of CQT over STFT is that it increases the window size for lower frequencies providing more detail which might be beneficial for the data involved as most black hole mergers occur in the low frequency range.

4.5 Gramian Angular Field (GAF)

Gramian Angular Field [26] images are represented as a Gramian matrix where each element is the trigonometric sum (i.e., superposition of directions) between different time intervals. The transformation proceeds as follows:

- Normalize the time series so that all values fall in the interval [-1,1]

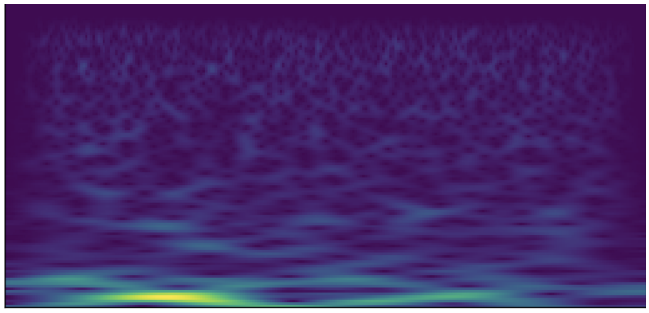


Figure 3: CQT transformed data for label = 1 (signal+noise)

- Represent the time series in the polar coordinates by encoding the amplitude as the angular cosine and time stamp as the radius
- Construct the gram matrix by considering the trigonometric sum between each point to identify the temporal correlation within different time intervals. The GAF is defined as

$$G = \begin{bmatrix} \cos(\phi_1 + \phi_1) & \dots & \cos(\phi_1 + \phi_n) \\ \cos(\phi_2 + \phi_1) & \dots & \cos(\phi_2 + \phi_n) \\ \vdots & \ddots & \vdots \\ \cos(\phi_n + \phi_1) & \dots & \cos(\phi_n + \phi_n) \end{bmatrix} \tag{7}$$

Gramian Angular Fields preserve temporal dependency and also contain temporal correlations as $G_{(i,j||i-j|=k)}$ represents the relative correlation by superposition of directions with respect to time interval k.

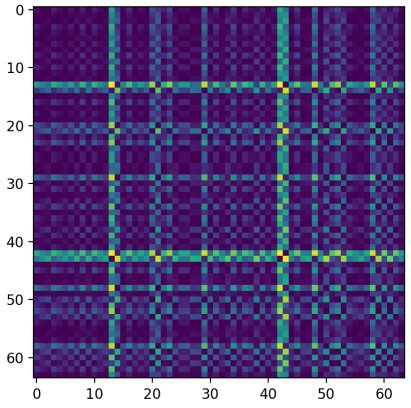


Figure 4: GAF transformed images for label = 1 (signal+noise)

4.6 Markov Transition Fields (MTF)

The main idea behind Markov Transition field [26] is to use Markov transition probabilities to to preserve information in the time domain. The transformation is carried out in the following way.

- Given a times X, the first step is to identify its Q quantile bins and assign each time step x_i to the corresponding bins q_j where j ∈ [1,Q].
- Next a Q × Q weighted adjacency matrix W is constructed by counting transitions among quantile bins in the manner of a first-order Markov chain along the time axis. Each entry to the matrix w_{ij} is calculated by the frequency with which a point in the quantile q_j is fol- lowed by a point in the quantile q_i
- After normalization, the adjacency matrix becomes the Markov transition matrix

$$M = \begin{bmatrix} w_{ij|x_1 \in q_i, x_1 \in q_j} & \cdots & w_{ij|x_1 \in q_i, x_n \in q_j} \\ w_{ij|x_2 \in q_i, x_1 \in q_i} & \cdots & w_{ij|x_2 \in q_i, x_n \in q_i} \\ \vdots & \ddots & \vdots \\ w_{ij|x_n \in q_i, x_1 \in q_i} & \cdots & w_{ij|x_n \in q_i, x_n \in q_i} \end{bmatrix} \tag{8}$$

By assigning the probability from the quantile at time step i to the quantile at time step j at each pixel M_{ij} , the MTF actually encodes the multi-span transition probabilities of the time series.

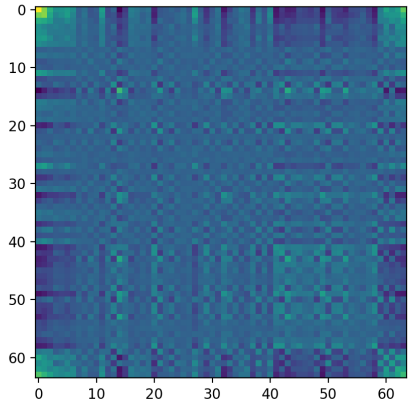


Figure 5: MTF transformed image for label = 1 (signal+noise)

5 NEURAL NETS FOR CLASSIFICATION

Once preprocessing is completed for the time series data, deep learning can help with classifying GW signals from the background. In the past decade, deep learning [22] has emerged as the biggest name in machine learning. It has solved problems that a few years ago was thought to be intractable, for example, the automatic recognition of patterns in spatial and temporal data with an accuracy superior to that of humans. It has solved problems beyond the realm of traditional, hand-crafted machine learning algorithms and captured the imagination of practitioners who are inundated with all types of data. Even for gravitational wave searches, Convolutional Neural networks (CNN) were the first neural networks to match the performance of matched filtering technique at a much faster rate [12]

For this project, two different types of deep neural architectures will be used

- Convolutional Neural Networks (CNN) for imaged based time series data
- Residual Neural networks (RNN) for raw time series data.

The ACM Reference Format text is required for all articles over one page in length, and is optional for one-page articles (abstracts).

5.1 Simple 2D-CNN

CNN architectures are most popular for image based classification problems. Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

The first model consists of 3 blocks of Convolution+MaxPool layer with a fully connected network in the end that takes flattened output from the Convolution layer. Dropout is added after the fully connected layer to reduce overfitting. The output layer uses a sigmoid activation function for binary output. All the other layers use ReLU activation function.

```
=================================================================
input_1 (InputLayer)         [(None, 65, 64, 3)]       0

conv2d (Conv2D)              (None, 64, 64, 3)         21

conv2d_1 (Conv2D)            (None, 64, 64, 32)        896

conv2d_2 (Conv2D)            (None, 64, 64, 32)        9248

batch_normalization (BatchN  (None, 64, 64, 32)        128
ormalization)

max_pooling2d (MaxPooling2D  (None, 32, 32, 32)        0
)

conv2d_3 (Conv2D)            (None, 32, 32, 64)        18496

conv2d_4 (Conv2D)            (None, 32, 32, 64)        36928

batch_normalization_1 (Batc  (None, 32, 32, 64)        256
hNormalization)

max_pooling2d_1 (MaxPooling  (None, 16, 16, 64)        0
2D)

conv2d_5 (Conv2D)            (None, 16, 16, 256)       147712

conv2d_6 (Conv2D)            (None, 16, 16, 256)       590080

batch_normalization_2 (Batc  (None, 16, 16, 256)       1024
hNormalization)

max_pooling2d_2 (MaxPooling  (None, 8, 8, 256)         0
2D)

dropout (Dropout)            (None, 8, 8, 256)         0

flatten (Flatten)            (None, 16384)             0

dense (Dense)                (None, 512)               8389120

dropout_1 (Dropout)          (None, 512)               0

dense_1 (Dense)              (None, 1)                 513

=================================================================
Total params: 9,194,422
Trainable params: 9,193,718
Non-trainable params: 704
```

Figure 6: Summary of the CNN model

5.2 Residual Networks

Residual networks [13], first introduced in 2015, provided an effective way for neural networks to get deeper by introducing residual mapping between the layers. If $\mathcal{H}(x)$ denote the initial underlying mapping between nonlinear stacked layers, then residual mapping introduces a new mapping

$$\mathcal{F}(x) := \mathcal{H}(x) - \mathbf{x}$$
$$\implies \mathcal{H}(x) := \mathcal{F}(x) + \mathbf{x}$$

Basically, it introduces shortcut connections in the form of identity mapping skipping two or more layers. These skip connections allow gradient information to pass through the layers to the deeper parts of the network, helping maintain signal propagation even in deeper networks. This avoids the problem of vanishing gradients in deep networks during back propagation where small gradient values can hinder the process of learning. Hence, the residual mapping makes it is easier to train than the previous unreferenced mapping without adding any additional parameters to the model. It does so by solving the problem of vanishing gradient

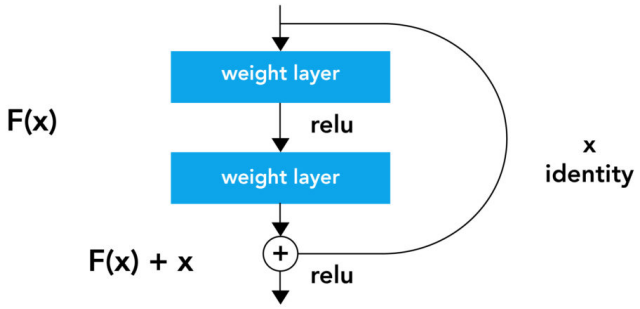


Figure 7: Summary of the CNN model

Both Resnet-34 and Resnet-50 are built from scratch and trained on the GW data to get an idea about the strength of a network and its depth.

5.3 LSTM/GRU + Fully Convolutional Networks (FCN)

Traditional neural networks assume that inputs they receive and the outputs generated are independent of each other, Recurrent neural networks (RNNs), on the other hand, have a concept of 'memory' as they take information from prior inputs to influence the current input and output. The connection between nodes form a directed or undirected graph that along a temporal sequence. Hence RNNs are naturally suited to processing time-series data and other sequential data.

In order to directly train on the time series data (after normalization and filtering), a combination of RNN and CNN architectures as suggested in the literature [16, 17] is implemented. For the FCN part, there are three convolutional blocks with filters 128, 256 and 128. Each convolutional block consists of a 1D-temporal convolution followed by batch normalization followed by ReLU activation. Simultaneously, the time series data is passed through a dimensional shuffling. The idea is to perceive the given time series data in two different views. The FCN block considers the time series data as a univariate series with N time steps. On the other hand, the RNN block receives the times as a multivariate time series with a single time step. Both Gated Recurrent Units (GRU) and Long-Short Term Memory (LSTM) are considered for the RNN block even though the original work only considered LSTM architecture. Both GRU and LSTM are improvements over the traditional RNN architecture in an attempt to deal with vanishing gradient problem bu introducing gating mechanism that controls the memorizing process.

In addition, at the end of each convolution block, there is a squeeze-and-excite block that adaptively recalibrates channel-wise feature responses by explicitly modelling interdependencies between channels. The details of the operation is not included and can be found here [14].

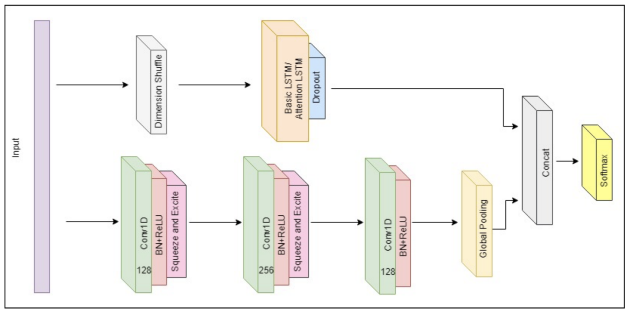


Figure 8: The LSTM/GRU + FCN architecture. Source: [23]

5.4 Encoder

The final architecture studied for the project is an encoder network [23]. The model considered as encoder is a standard convolutional network, with a convolutional attention mechanism to summarize the time axis, and a final fully-connected layer to set the desired output. The convolutional network is formed by three convolutional blocks with two 2-factor max-pooling layers between them. Each convolutional block is formed by a 1-dimensional convolution, followed by an instance normalization layer, a parametric rectified linear unit (PReLU) activation, and a dropout layer . After the first part of the network, half of the filters are input to a time-wise softmax activation, which acts as an attention mechanism for the other half of the filters. So for a single filter,

$$h = \mathbf{h} \cdot \mathbf{a} \tag{9}$$

where $\mathbf{h}$ is the result of a single 1-dimensional convolutional filter over a time-wise signal, and $\mathbf{a}$ is the time-wise attention vector. For the three convolutional layers 128, 256, and 512 filters are used respectively, with kernels of size 5, 11, and 21, and same-length padding of 3, 5, and 10.

6 RESULTS AND ANALYSIS

All the deep learning models are implemented on Tensorflow and Keras. Each time series data is stored as a numpy file which is converted and stored as a hierarchical Data format (HD5). Then the data is normalized using min-max normalization to be in the range

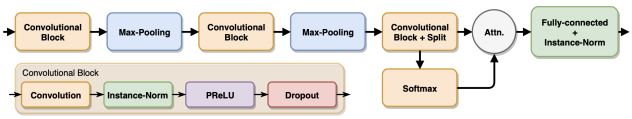


Figure 9: Architecture diagram of the proposed encoder (top) and the convolutional block (bottom left). Source: [23]

Table 1: Fine tuned Adam parameters

Parameters	Values
learning rate :	5×10^{-5}
β_1	0.9
β_2	0.999
Decay parameter	0.0

[-1,1] and bandpass filtering is applied. Finally, any transformations required for converting the time series data to images, as outlined previously, is carried out. Training and validation data is split in the ratio 9:1.

In the next few sections, the performance of the individual architectures are discussed. A batch size of 64 is maintained throughout all the models and each of them are trained between 7-10 epochs. Three different optimization algorithms namely SGD, Adam and AdamW were tried out which Adam provided the best results when fine tuned to the following parameters.

ROC-AUC scores are used as the major criteria for evaluation as the kaggle competition from which the data set is used, uses the same scores as the metric of performance. Along with that training and validation accuracy and loss are also monitored.

6.1 Performance on data transformed to images

A simple 2D-CNN, ResNet-34 and ResNet-50 architectures are trained using images transformed data that includes Constant Q transform and Short-time Fourier Transform spectrograms as well as Gramian and Markov Transition Fields.

The Test scores are obtained after submission to the kaggle competition. Only the best models based on the AUC scores are saved and then used on the test set. Even though both the ResNet architectures are deeper, they achieved similar scores compared to the 2D-CNN model. This might hint that the deeper CNN models are unable to extract any more features from the available dataset and more data is required for boost in performance. In general, a deeper network implies more trainable parameters and hence more data almost always help the cause or else it starts overfitting which also seems to be the case here. Out of all the transformations, Constant-Q transformed spectrograms performed the best whereas the models almost failed to learn anything from the Gramian Angular fields. Markov transition fields performed worse and hence have not been included.

6.2 Performance on filtered time series data

Time series data, after appropiate normalization and filtering, is directly fed into LSTM/GRU + FCN and Encoder networks. The encoder performs the best out of the three and achieves almost similar accuracy compared to CNN classifiers on CQT spectrogram images. The replacement of LSTM block with GRU does not lead to any improvement in performance. In these cases, the learning rate for Adam optimizer can be inreased upto 1×10^{-4} and still achieve similar accuracy but leads to greater fluctuations in loss and accuracy. The squeeze-and-excite block adapted [14] for the FCN network does not lead to significant improvement in accuracy for both LSTM and GRU.

7 CONCLUSION AND FUTURE WORK

The highest AUC score on the kaggle competition from which the dataset has been used for this project was 0.88538. The top score obtained in this project is 0.85844 by the ResNet34 architecture on CQT transformed data. So the performance of the models in several instances, are comparable to the best representations used during the competition. Also, both raw time series data and image transformed data performed equally good given that the model is able to maximize feature extraction. Preprocessing the data, seems to be crucial than the choice of models, specially going deep with the architecture, in order to achieve better accuracy. In this direction, other transformations like the variable Q-transform [19], may be used to generate spectrogram images with better feature mapping. Since increasing the depth of the architecture did not lead to significant boost in performance, a good idea might be to generate either by data augmentation or by generating new data. Data augmentation can include swapping detector 1 (Hanford) and 2 (Livingston) signals as they almost appear to be identical compared to detector 3 (Virgo). Also some additional noise can be added to the data to create more samples. As suggested by the TA, generating new data samples identical to the data provided requires a bit of reverse engineering [6]. First the parameters used to generate the samples needs to be estimated which again can be done using neural networks and then use them to generate samples close to the available data [21]. Parameter estimation can be tricky as the number of parameters is large and getting close to the actual parameters used is essential as eyeballed parameters' ranges and distributions will not be really beneficial.

8 ACKNOWLEDGMENTS

The author would like to thank Jose Wudka, instructor Vagelis Papalexakis and TA Rutuja Gurav for useful discussions and comments during the course of the project. This section has a special environment:

REFERENCES

[1] B.P. Abbott, R. Abbott, T.D. Abbott, M.R. Abernathy, F. Acernese, K. Ackley, C. Adams, T. Adams, P. Addesso, R.X. Adhikari, and et al. 2016. Observation of Gravitational Waves from a Binary Black Hole Merger. *Physical Review Letters* 116, 6 (Feb 2016). https://doi.org/10.1103/physrevlett.116.061102

[2] Christopher Bresten and Jae-Hun Jung. 2019. Detection of gravitational waves using topological data analysis and convolutional neural network: An improved approach. arXiv:1910.08245 [astro-ph.IM]

[3] Judith C. Brown. 1991. Calculation of a constant Q spectral transform. *Journal of the Acoustical Society of America* 89 (1991), 425–434.

[4] LIGO Collaboration. 2015. *How we searched for binary black holes and found GW150914.* Retrieved December 2, 2021 from https://www.ligo.org/science/Publication-GW150914CBC/index.php

Table 2: Best performance on CQT spectrogram images

Model	Train ROC-AUC	Val. ROC-AUC	Test ROC-AUC
2D-CNN	0.8585	0.8506	0.85493
ResNet34	0.8619	0.8534	0.85844
ResNet50	0.8575	0.8477	0.85308

Table 3: Best performance on STFT spectrogram images

Model	Train ROC-AUC	Val. ROC-AUC	Test ROC-AUC
2D-CNN	0.8340	0.8291	0.83294
ResNet34	0.8303	0.8244	0.82986
ResNet50	0.8509	0.8430	0.84674

Table 4: Best performance on GAF images

Model	Train ROC-AUC	Val. ROC-AUC	Test ROC-AUC
2D-CNN	0.6035	0.5967	0.5912
ResNet34	0.6038	0.5358	0.52637

Table 5: Best performance on filtered time series data

Model	Train ROC-AUC	Val. ROC-AUC	Test ROC-AUC
LSTM+FCN	0.6844	0.6814	0.68467
GRU+FCN	0.6832	0.6814	0.68539
Encoder	0.8441	0.8345	0.83749

[5] European Gravitational Observatory EGO. 2021. *G2Net Gravitational Wave Detection.* Retrieved December 2, 2021 from https://www.kaggle.com/c/g2net-gravitational-wave-detection

[6] European Gravitational Observatory EGO. 2021. *Signal.* Retrieved December 2, 2021 from https://www.kaggle.com/c/g2net-gravitational-wave-detection/discussion/275507

[7] Albert Einstein. 1916. Näherungsweise Integration der Feldgleichungen der Gravitation. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften (Berlin* (Jan. 1916), 688–696.

[8] Hunter Gabbard, Michael Williams, Fergus Hayes, and Chris Messenger. 2018. Matching Matched Filtering with Deep Networks for Gravitational-Wave Astronomy. *Phys. Rev. Lett.* 120 (Apr 2018), 141103. Issue 14. https://doi.org/10.1103/PhysRevLett.120.141103

[9] Timothy D. Gebhard, Niki Kilbertus, Ian Harry, and Bernhard Schölkopf. 2019. Convolutional neural networks: A magic bullet for gravitational-wave detection? *Phys. Rev. D* 100 (Sep 2019), 063015. Issue 6. https://doi.org/10.1103/PhysRevD.100.063015

[10] Daniel George and E.A. Huerta. 2018. Deep Learning for real-time gravitational wave detection and parameter estimation: Results with Advanced LIGO data. *Physics Letters B* 778 (2018), 64–70. https://doi.org/10.1016/j.physletb.2017.12.053

[11] Daniel George and E. A. Huerta. 2017. Deep Learning for Real-time Gravitational Wave Detection and Parameter Estimation with LIGO Data. arXiv:1711.07966 [gr-qc]

[12] Daniel George and E. A. Huerta. 2018. Deep neural networks to enable real-time multimessenger astrophysics. *Phys. Rev. D* 97 (Feb 2018), 044039. Issue 4. https://doi.org/10.1103/PhysRevD.97.044039

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385 [cs.CV]

[14] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. 2019. Squeeze-and-Excitation Networks. arXiv:1709.01507 [cs.CV]

[15] S. J. Kapadia, T. Dent, and T. Dal Canton. 2017. Classifier for gravitational-wave inspiral signals in nonideal single-detector data. *Phys. Rev. D* 96 (Nov 2017), 104015. Issue 10. https://doi.org/10.1103/PhysRevD.96.104015

[16] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Shun Chen. 2018. LSTM Fully Convolutional Networks for Time Series Classification. *IEEE Access* 6 (2018), 1662–1669. https://doi.org/10.1109/access.2017.2779939

[17] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. 2019. Multivariate LSTM-FCNs for time series classification. *Neural Networks* 116 (Aug 2019), 237–245. https://doi.org/10.1016/j.neunet.2019.04.014

[18] Plamen G. Krastev. 2020. Real-Time Detection of Gravitational Waves from Binary Neutron Stars using Artificial Neural Networks. arXiv:1908.03151 [astro-ph.IM]

[19] Jialong Li, Hongxia Wang, Peisong He, Sani M. Abdullahi, and Bin Li. 2022. Long-term variable Q transform: A novel time-frequency transform algorithm for synthetic speech detection. *Digital Signal Processing* 120 (2022), 103256. https://doi.org/10.1016/j.dsp.2021.103256

[20] Antonis Mytidis, Athanasios Aris Panagopoulos, Orestis P. Panagopoulos, Andrew Miller, and Bernard Whiting. 2019. Sensitivity study using machine learning algorithms on simulated r-mode gravitational wave signals from newborn neutron stars. *Phys. Rev. D* 99 (Jan 2019), 024024. Issue 2. https://doi.org/10.1103/PhysRevD.99.024024

[21] PyCBC. 2021. *Generating waveforms.* Retrieved December 2, 2021 from https://pycbc.org/pycbc/latest/html/waveform.html

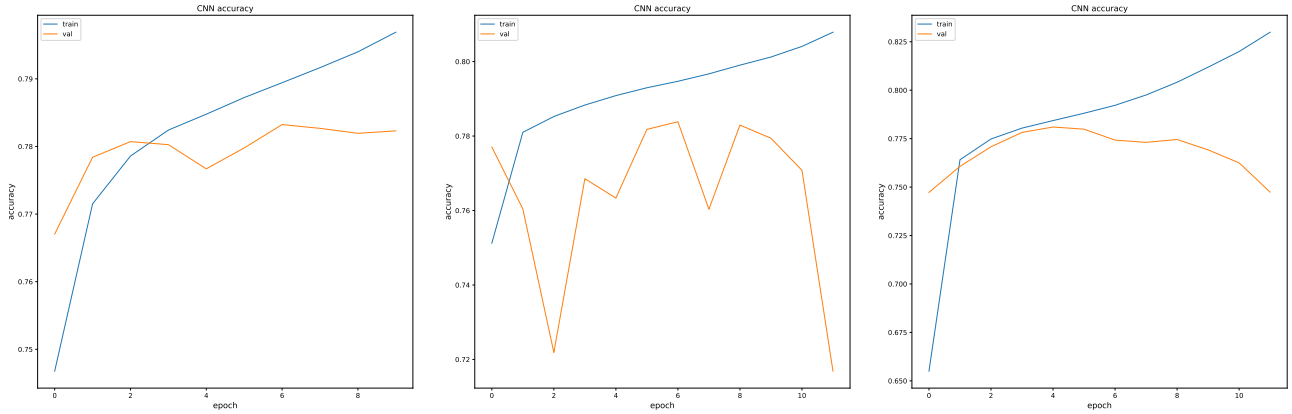
[22] Saptarshi Sengupta, Sanchita Basak, Pallabi Saikia, Sayak Paul, Vasilios Tsalavoutis, Frederick Atiah, Vadlamani Ravi, and Alan Peters. 2020. A review of deep learning with special emphasis on architectures, applications and recent trends. *Knowledge-Based Systems* 194 (2020), 105596. https://doi.org/10.1016/j.knosys.2020.105596

[23] Joan Serrà, Santiago Pascual, and Alexandros Karatzoglou. 2018. Towards a universal neural network encoder for time series. arXiv:1805.03908 [cs.LG]

[24] Hongyu Shen, E. A. Huerta, Eamonn O'Shea, Prayush Kumar, and Zhizhen Zhao. 2021. Statistically-informed deep learning for gravitational wave parameter estimation. arXiv:1903.01998 [gr-qc]

[25] He Wang. 2020. *Gravitational Wave Data Analysis with Machine Learning.* Retrieved December 2, 2021 from https://iphysresearch.github.io/Survey4GWML/

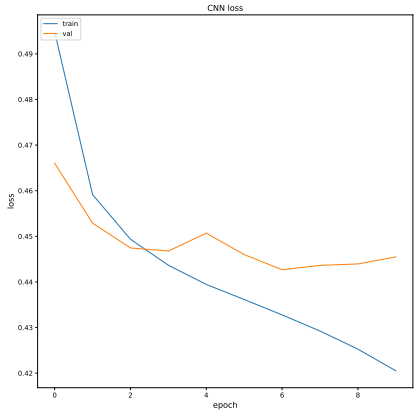
[26] Zhiguang Wang and Tim Oates. 2014. Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks.



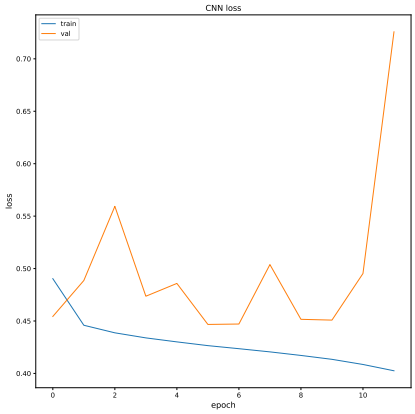
(a) 2D-CNN accuracy

(b) ResNet34 accuracy

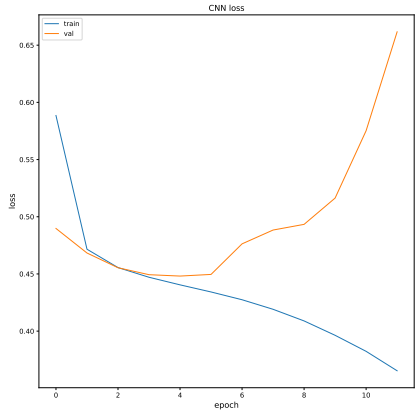
(c) ResNet50 accuracy



(d) 2D-CNN loss



(e) ResNet34 Loss



(f) ResNet50 loss

Figure 10: Training and Validation loss for CQT spectrograms

[27] Kenta Yanagisawa, Dongbao Jia, Shigeki Hirobayashi, Nami Uchikata, Tatsuya Narikawa, Koh Ueno, Hirotaka Takahashi, and Hideyuki Tagoshi. 2019. A time–frequency analysis of gravitational wave signals with non-harmonic analysis. *Progress of Theoretical and Experimental Physics* 2019, 6 (06 2019). https://doi.org/10.1093/ptep/ptz043 arXiv:https://academic.oup.com/ptep/article-pdf/2019/6/063F01/28788035/ptz043.pdf 063F01.