KLE Society's

KLE Technological University



**DMA COURSE PROJECT**

**G2Net Gravitational Wave Detection**

**Bachelor of Engineering**

**In**

**Computer Science and Engineering**

**Team No: 05**

Team Members:

Pradeep Chegur – 01FE19BCS294

Praveen Thakkannavar – 01FE19BCS298

Swagat Ingalagaon – 01FE19BCS299

Naveen Doddamani – 01FE19BCS305

Faculty In charge

**Prof. Shankar G**

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

**TABLES OF CONTENTS**

1. Introduction
2. Problem Statement
3. Related Work

3.1. Basic EDA and a Baseline Kera’s Model:

3.2. Simple CNN Model Demonstration.

1. Methodology

4.1. Understanding the data

4.2. Exploratory Data Analysis

4.3. Data Preprocessing

4.4. Model Building

1. Results and Discussion
2. Conclusion
3. References

**1. Introduction:**

Gravitational waves are 'ripples' in space-time caused by some of the most violent and energetic processes in the Universe.

* Challenge: G2Net Gravitational Wave Detection

* About: To find gravitational wave signals from binary black hole collisions

* Duration: 22 Sep 2021- Entry Deadline

                29 Sep 2021- Final Submission

In this competition, our aim is to detect GW signals from the mergers of binary black holes. Specifically, we have to build a model to analyze simulated GW time-series data from a network of Earth-based detectors.

**2. Problem Statement:**

**To detect GW signals from the mergers of binary black holes. Specifically, to build a model to analyze simulated GW time-series data from a network of Earth-based Detectors.**

**3. Related Work:**

# **3.1.** **Basic EDA and a Baseline Kera’s Model:**

# **Note on Kera’s Sequential model**

There are two ways to build Kera’s models: sequential and functional.

**The sequential API** allows you to create models layer-by-layer for most problems. It is limited in that it does not allow you to create models that share layers or have multiple inputs or outputs. In short, you create a sequential model where you can easily add layers, and each layer can have convolution, max pooling, activation, drop-­out, and batch normalization.

Alternatively, **the functional API** allows you to create models that have a lot more flexibility as you can easily define models where layers connect to more than just the previous and next layers. In fact, you can connect layers to (literally) any other layer. As a result, creating complex networks such as siamese networks and residual networks become possible.

From the definition of Keras documentation the Sequential model is a linear stack of layers. You can create a Sequential model by passing a list of layer instances to the constructor. The common architecture of ConvNets is a sequential architecture. However, some architectures are not linear stacks. For example, Siamese networks are two parallel neural networks with some shared layers.

# **3.2.** **Simple CNN Model Demonstration.**

<https://www.analyticsvidhya.com/blog/2021/01/image-classification-using-convolutional-neural-networks-a-step-by-step-guide/>

In this notebook author Instantiate the Sequential model and then Added the first Convoluted2D layer input\_shape & MaxPooling2D layer followed by that Second pair of Conv1D and MaxPooling1D layers and finally Third pair of Conv1D and MaxPooling1D layers. Along with this they added flatten and dense layers to the final out put file.

**4. Methodology:**

# **4.1. Understanding the Dataset:**

**Test Data:**

* Number of rows: 560000
* Number of Columns: 2
* Attributes: ID, Target.

**Train Data:**

* Number of rows: 226000
* Number of Columns: 2
* Attributes: ID, Target.

**Data Import and Reading a CSV file:**

Here **id** is unique value for every wave and **target** tells us about that the given wave contains gravitational wave or not, if target is 0 indicates gravitational wave is not present and if target is 1 gravitational wave is present.

**4.2. Exploratory Data Analysis:**

* **Plotting Bar Graph to count target values:**

****

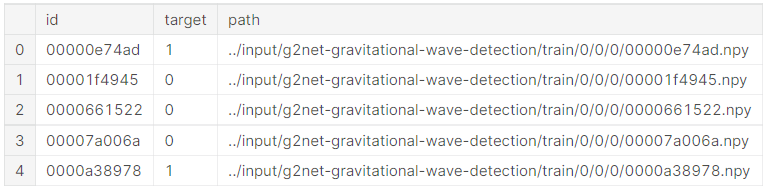
**Fig.1**

Classifying the data into 2 classes of target 1 and target 0

* 0 – 280070
* 1 – 279930
* We can infer from the above data that we have almost same probabilities of getting gravitational wave signal in the time series containing detector noise
* **Adding attribute Path:**

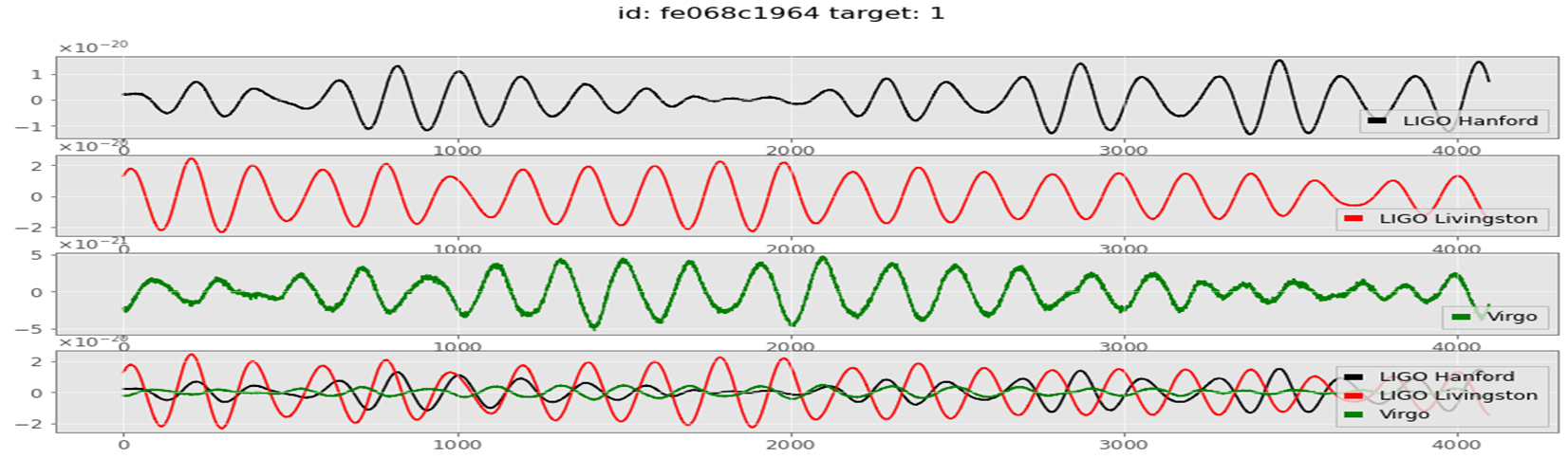
Each file is in .npy form and each time series instance spans over 2 seconds and is sampled at 2048 Hz.

All the files are saved in nested four layers to avoide data discrepancy and Data loss.

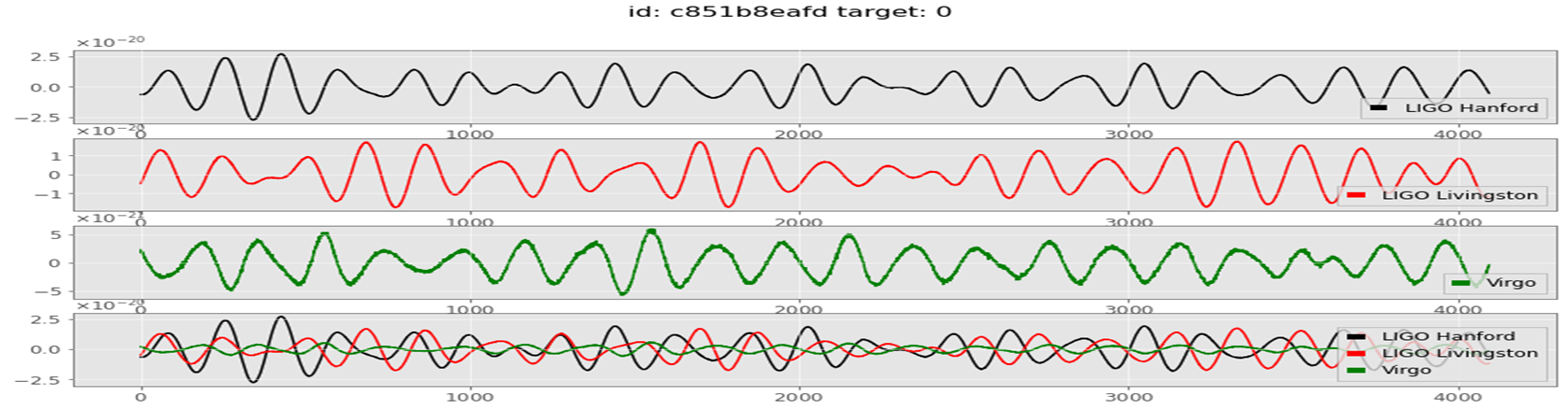


**Fig.2**

* **Data Visualization**
* Plotting time domain graph for selected wave which has target value 0 and 1.



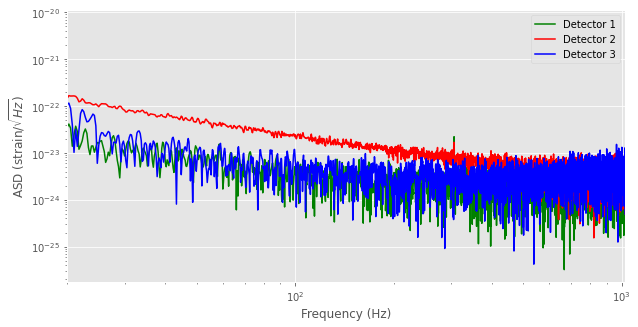
**Fig.3**



**Fig.4**

From the above two graphs we can not say which wave has simulated gravitational wave and which has noise. Apart from the target values both the waves look similar it is hard to differentiate between both the waves.

**Plotting the Data in frequency domain:**

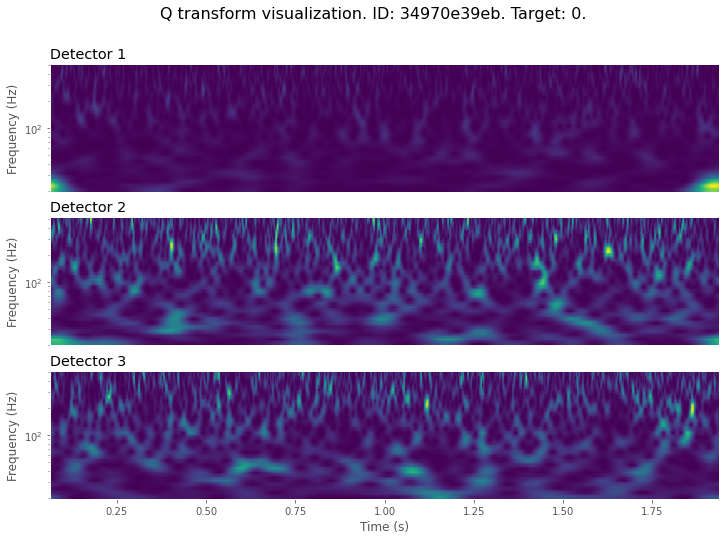


**Fig.5**

A time-domain graph shows how a signal changes over time, whereas a frequency-domain graph shows how much of the signal lies within each given frequency band over a range of frequencies. A frequency-domain representation can also include information on the phase shift that must be applied to each sinusoid in order to be able to recombine the frequency components to recover the original time signal.

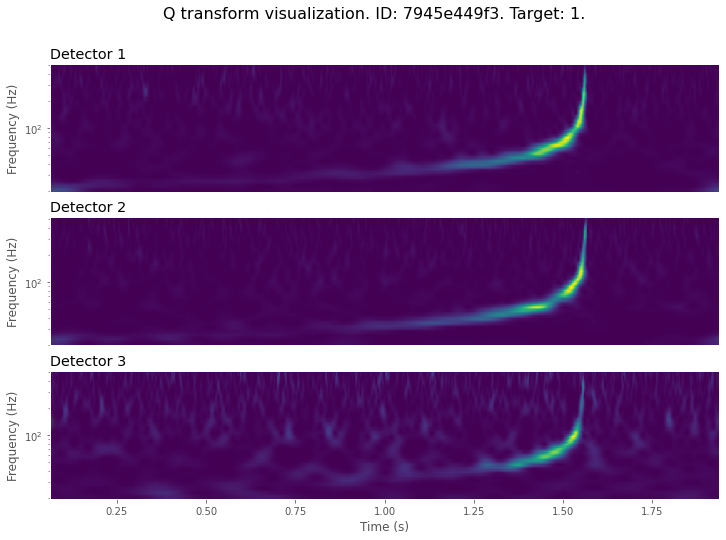
* **Performing the Constant Q-transform to visualize the selected data:**

Another very common way to visualize a GW signal is to perform a constant Q-transform (or CQT). This is a time-frequency representation widely used in processing musical data.



**Fig.6**

The above QT graph is the graph of wave which has the target value 0 this means that there is no simulatd gravitational wave present, So we can clearly see that it in the graph that there is no sudden spike in the graph.



**Fig.7**

The above QT graph is the graph of wave which has the target value 1 this means that there is simulatd gravitational wave present in it, So we can clearly see that it in the graph that there is sudden spike in the graph.

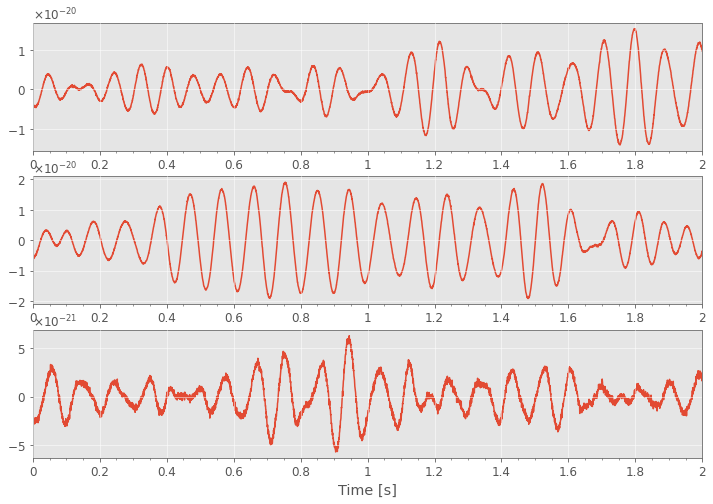
**4.3. Data Pre-processing:**

1. Applying **Tukey window** function.
2. Applying **whitening** the data.
3. Applying **Bandpass** for data.

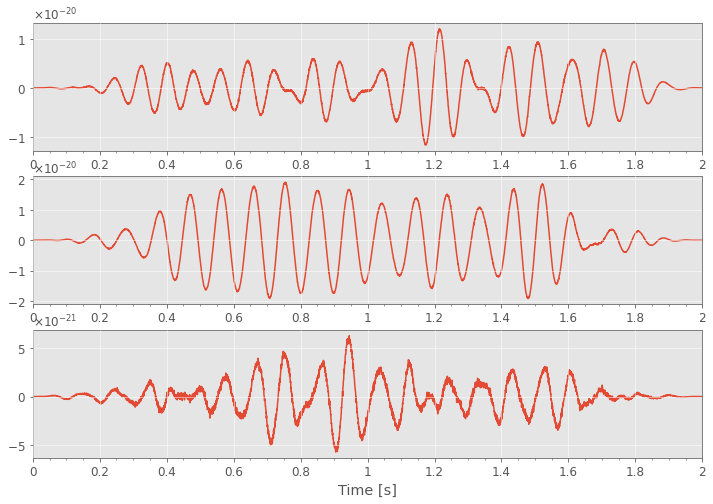
**4.3.1. Applying Tukey window function.**

WHY: To suppress spectral leakage (leaking the energy from peak value t other samples).

IF NOT DONE**:** Leads to spectral leakage, Spurious correlation in phase between bins.

Before applying Tukey function: 

**Fig.8**

After applying Tukey function: 

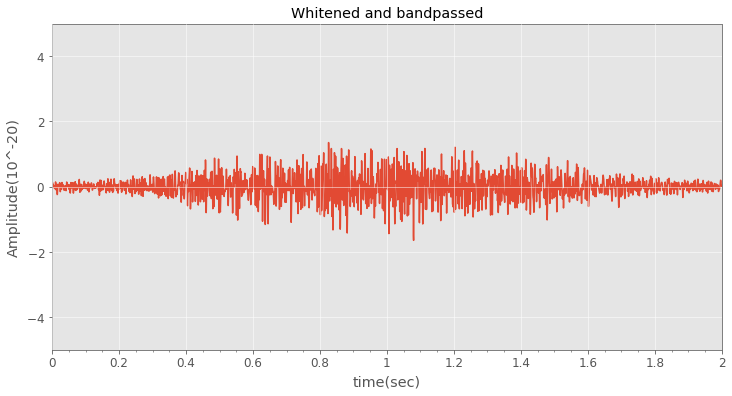
**Fig.9**

**4.3.2. Spectral whitening:**

WHY: To ensure data in each frequency bin has equal significance by down weighting where the noise is loud.

**4.3.3. Bandpass filtering:**

WHY: Used in order to concentrate on the main frequency band that might contain physically interesting signals.



**Fig.10**

# **4.4. Learning Model:**

**1. Kera’s Model:**

* Kera’s is a powerful and easy-to-use free open-source Python library for developing and evaluating deep learning models.
* Keras can be used to build a neural network to solve a classification problem.

There are two ways to build Keras models:

1. Sequential API
2. Functional API

* Sequential API allows to create models layer-by-layer for large data.
* Sequential API is easy to create multiple layers.
* In Sequential API, each layer has exactly one input and one output.
* Functional API is for multiple inputs and multiple outputs.
* Our model built by using Sequential API.

**Data Generator Class for Keras Sequential Model building:**

* Data Generator class is to handle large data with batching, so the RAM does not need to handle the full data at once.
* Functions used in this class are:

1. Initialization: To pass parameters.

2.Length: Returns the number of steps in an epoch.

3. Get item: To obtain a given batch of data.

4. Data Generation: To produce batches of data.

* In this model, we used

1. Used 1 Convoluted2D layer, 1 Flatten layer and 2 Dense layers.
2. Epoch used: 01

**2. Simple CNN:**

* The convolutional neural network models are used everywhere in the image data space.
* They work phenomenally well on computer vision tasks like image classification, object detection, image recognition, etc.
* In this model, we used

1. Epoch used: 03
2. Added 3 pairs of Convoluted2D layer and MaxPooling2D layer, 1 Flatten layer and 2 Dense layers.

**3. Efficient Net CNN Model:**

* In general, the Efficient-Net models achieve both **higher accuracy and better efficiency** over existing CNNs, reducing parameter size and FLOPS by an order of magnitude.
* The conventional practice for model scaling is to arbitrarily increase the CNN depth or width, or to use larger input image resolution for training and evaluation.
* In this model, we used

1. Epoch used: 04
2. Added 6 Convoluted2D layer, 3 Flatten layer.

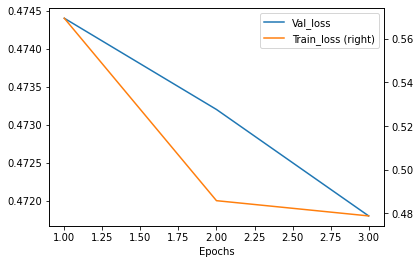
# **5. Results and Discission:**

**1. Result obtained for Kera’s model:**

* Validation Accuracy: 0.4983

**2. Result obtained for simple CNN model:**

* New Validation Accuracy: 0.8351
* Accuracy Improved: 0.3368
* **Validation loss and Train loss Graph obtained for simple CNN model:**



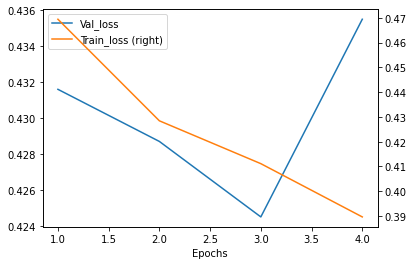
**Fig.11**

The above graph is validation loss and train loss over the number of epochs.

As we can clearly see that the validation loss and Train loss is decreasing as we increase the number of epochs this means that to attain high accuracy the number of epochs should be more.

**3. Result obtained for Efficient-Net CNN model:**

* New Validation Accuracy : 0.8624
* Accuracy Improved : 0.0273
* **Validation loss and train loss Graph obtained for simple Efficient-Net CNN model:**

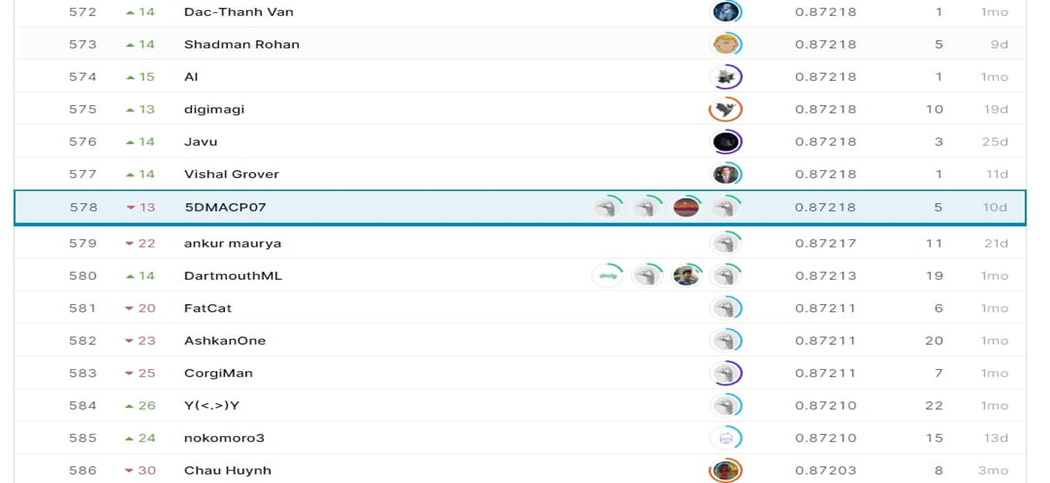


**Fig.12**

The above graph is validation loss and train loss over the number of epochs for Efficient-Net model.

As we can clearly see that the validation loss and Train loss is decreasing as we increase the number of epochs up till epoch value is three, this means that to attain high accuracy and to avoid the data loss the number of epochs should not exceed three so that the accuracy is maximum and loss is minimized.

* **Leader board Ranking:**



**Fig.13**

**Comparison of three models:**

|  |  |  |  |
| --- | --- | --- | --- |
| Model Name | Kera’s Model | Simple CNN Model | Efficient-Net CNN Model |
| **Epoch Used** | 01 | 03 | 04 |
| **Convoluted 2D Layer** | 01 | 03 | 06 |
| **Max-pooling 2D Layer** | 00 | 03 | 00 |
| **Flatten Layer** | 01 | 01 | 03 |
| **Dense Layer** | 02 | 03 | 00 |
| **Accuracy** | 0.4983 | 0.8351 | 0.8624 |
| **Justification** | * A first basic model * Epoch used is one * Layers used are less * No combination of convoluted with max-pooling layer * Accuracy is less | * Net to Kera’s model * Epoch used are three * At most 3 layers are used * Combination of convoluted with max-pooling layer * Accuracy is better than basic model | * Next to simple CNN Model * Epoch used are four * At most 6 layers are used * No combination of convoluted with max-pooling layer * Accuracy is maximum among three |
| Leader Board Ranking | 1165 | 1019 | 578 |

**Table.1**

**6. Conclusion:**

The understanding of dataset and working on data preprocessing is done to get an appropriate dataset which is used for building the models. The built models are tested with test dataset and the number of epochs is increased to improve the accuracy of the model. So, from the given model’s Efficient-Net CNN model results better accuracy with accuracy of 0.8624.

# **7. References:**

* A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals: <https://iopscience.iop.org/article/10.1088/1361-6382/ab685e>
* Challenge link: <https://www.kaggle.com/c/g2net-gravitational-wave-detection>
* Information about Gravitational Wave: <https://spaceplace.nasa.gov/gravitational-waves/en/>
* Information about Binary black hole: <https://www.frontiersin.org/articles/10.3389/fspas.2020.00028/full>
* A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals: <https://iopscience.iop.org/article/10.1088/1361-6382/ab685e>
* Reference video about for the project: <https://www.youtube.com/watch?v=UeI4-kyuAwI>
* Information about Kera’s model: <https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/>
* Information about CNN model: <https://www.analyticsvidhya.com/blog/2021/01/image-classification-using-convolutional-neural-networks-a-step-by-step-guide/>