Deep Learning for Physics Problems: A Case Study in Continuous Gravitational Waves Detection

Different propositions for the usage of deep learning algorithms in detecting continuous gravitational waves in both Time & Frequency domains

Essam Mohamed Farouq El-Tobgi Electronics and Telecommunications Engineer

***Abstract*—Deep learning has become a powerful tool for solving a wide variety of problems, including those in physics. In this paper, we explore the use of deep learning for the detection of continuous gravitational waves. We propose two different approaches: one based on time-domain analysis and the other based on frequency-domain analysis. Both approaches achieve nearly the same performance, suggesting that deep learning is a promising technique for this task. The main purpose of this paper is to provide an overview of the potential of deep learning for physics problems. We do not provide a performance-measured solution, as this is beyond the scope of this paper. However, we believe that the results presented here are encouraging and suggest that deep learning is a valuable tool for physicists.**

***Index Terms*—deep learning, physics, continuous gravitational waves, time-domain analysis, frequency-domain analysis.**

I. Introduction

GRAVITATIONAL waves are disturbances or ripples in the curvature of spacetime, generated by accelerated masses, that propagate as waves outward from their source at the speed of light. They were first proposed by Oliver Heaviside in 1893 and then later by Henri Poincaré in 1905 and subsequently predicted in 1916 by Albert Einstein on the basis of his general theory of relativity. Later he refused to accept gravitational waves. Gravitational waves transport energy as gravitational radiation, a form of radiant energy similar to electromagnetic radiation.Newton's law of universal gravitation, part of classical mechanics, does not provide for their existence, since that law is predicated on the assumption that physical interactions propagate instantaneously (at infinite speed) – showing one of the ways the methods of Newtonian physics are unable to explain phenomena associated with relativity.

The first indirect evidence for the existence of gravitational waves came in 1974 from the observed orbital decay of the Hulse–Taylor binary pulsar, which matched the decay predicted by general relativity as energy is lost to gravitational radiation. In 1993, Russell A. Hulse and Joseph Hooton Taylor Jr. received the Nobel Prize in Physics for this discovery. The first direct observation of gravitational waves was not made until 2015, when a signal generated by the merger of two black holes was received by the LIGO gravitational wave detectors in Livingston, Louisiana, and in Hanford, Washington. The 2017 Nobel Prize in Physics was awarded to Rainer Weiss, Kip Thorne and Barry Barish for their role in the direct detection of gravitational waves.

When scientists detected the first class of gravitational waves in 2015, they expected the discoveries to continue. There are four classes, yet at present only signals from merging black holes and neutron stars have been detected. Among those remaining are continuous gravitational-wave signals. These are weak yet long-lasting signals emitted by rapidly-spinning neutron stars. Imagine the mass of our Sun but condensed into a ball the size of a city and spinning over 1,000 times a second. The extreme compactness of these stars, composed of the densest material in the universe, could allow continuous waves to be emitted and then detected on Earth. There are potentially many continuous signals from neutron stars in our own galaxy and the current challenge for scientists is to make the first detection, and hopefully data science can help with this mission.

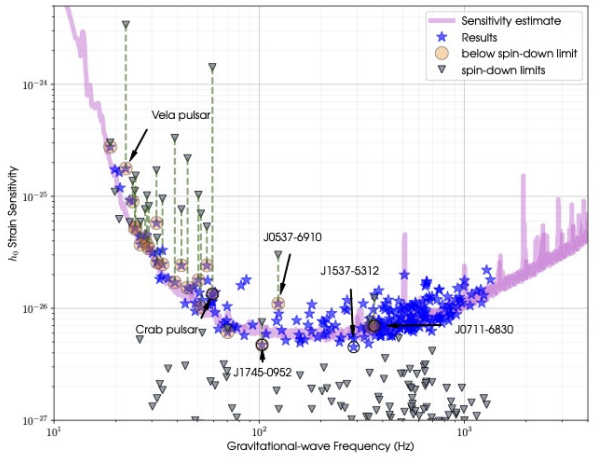

Fig. 1. Gravitational wave frequency vs. h0 strain sensitivity

This image, taken from a 2021 paper by the LIGO-Virgo-KAGRA collaboration, shows the maximum amplitude of a continuous wave any of these neutron stars could emit without being found by the search analyses. Circled stars show results constraining the physical properties of specific neutron stars. Traditional approaches to detecting these weak and hard-to-find continuous signals are based on matched-filtering variants. Scientists create a bank of possible signal waveform templates and ask how correlated each waveform is with the measured noisy data. High correlation is consistent with the presence of a signal similar to that waveform. Due to the long duration of these signals, banks could easily contain hundreds of quintillions of templates; yet, with so many

possible waveforms, scientists don't have the computational power to use the approach without making approximations that weaken the sensitivity to the signals.

II. Dataset analysis & Exploration

Our dataset is free and available on Kaggle from the European Gravitational Observatory(EGO) and comes in Hierarchical Data Format (HDF) which is a set of file formats (HDF4, HDF5) designed to store and organize large amounts of data. Originally developed at the U.S. National Center for Supercomputing Applications, it is supported by The HDF Group, a non-profit corporation whose mission is to ensure continued development of HDF5 technologies and the continued accessibility of data stored in HDF. In keeping with this goal, the HDF libraries and associated tools are available under a liberal, BSD-like license for general use. HDF is supported by many commercial and non-commercial software platforms and programming languages. The freely available HDF distribution consists of the library, command-line utilities, test suite source, Java interface, and the Java-based HDF Viewer (HDFView). The current version, HDF5, differs significantly in design and API from the major legacy version HDF4.

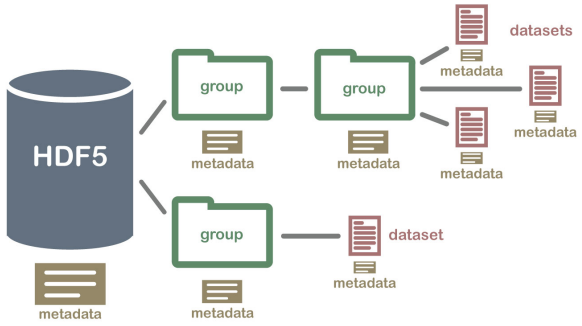

Fig. 2. HDF5 data files structure

The HDF5 format is designed to address some of the limitations of the HDF4 library, and to address current and anticipated requirements of modern systems and applications. In 2002 it won an R&D 100 Award. HDF5 simplifies the file structure to include only two major types of object: HDF Structure Example

- Datasets, which are typed multidimensional arrays
- Groups, which are container structures that can hold datasets and other groups

As when it comes to out dataset structure, it comes as follows:

- ID is the top group of the HDF5 file and links the datapoint to it's label in the train_labels csv (group)
- frequency_Hz contains the range frequencies measured by the dectors (dataset)
- H1 contains the data for the LIGO Hanford dececter (group)
- L1 contains the data for the LIGO Livingston dececter (group)
  - SFTs is the Short-time Fourier Transforms amplitudes for each timestamp at each frequency (dataset)
  - timestamps contains the timestamps for the measurement (dataset)

As when it comes to the data labels frequency it shows that the dataset is imbalanced which indicated the usage of a more convenient performance measurement metric like F-Measure:

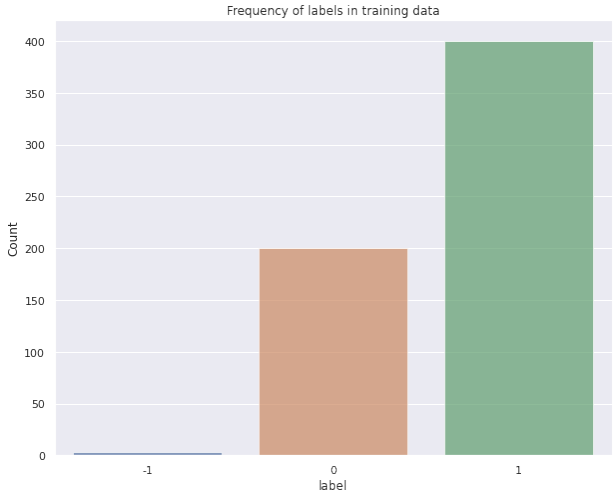

Fig. 3. Dataset labels frequency

The target labels; 1 if the data contains the presence of a gravitational wave, 0 otherwise. (Please note the presence of a small number of files labeled -1. Physicists are currently unable to determine the status of these files.)

Now we get to the spectrogram analysis which is a visual representation of the spectrum of frequencies of a signal as it varies with time. When applied to an audio signal, spectrograms are sometimes called sonographs, voiceprints, or voicegrams. When the data are represented in a 3D plot they may be called waterfall displays.

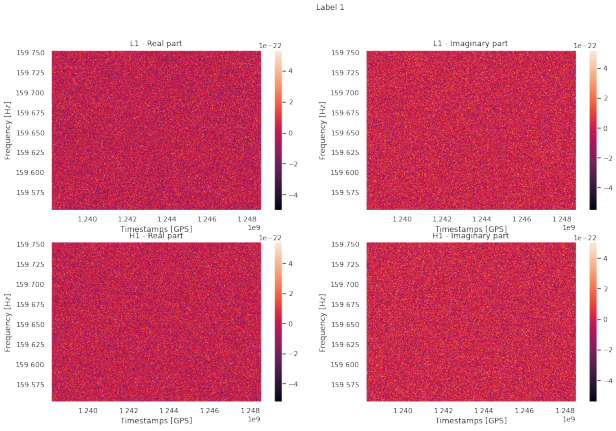

Fig. 4. H and L detectors real and imaginary components visualization

As when it comes to frequencies and timestamps distributions, they were as follows:

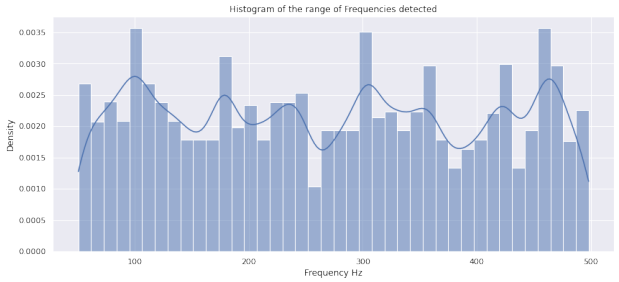

Fig. 5. Frequency values distribution

We notice the frequency variation is not that critical that is why during the time domain conversion process we would use the mean value of all the set of frequencies.

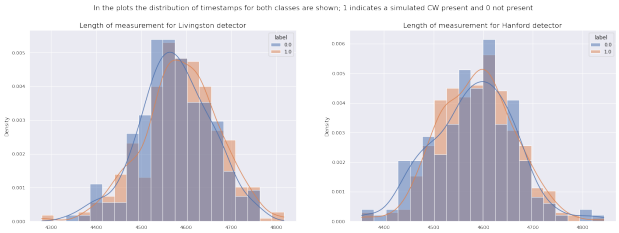

Fig. 6. The distribution for timestamps of both classes

III. Time Domain Dataset Conversion

The STFT is invertible, that is, the original signal can be recovered from the transform by the inverse STFT. The most widely accepted way of inverting the STFT is by using the overlap-add (OLA) method, which also allows for modifications to the STFT complex spectrum. This makes for a versatile signal processing method, referred to as the overlap and add with modifications method.

The inverse Fourier transform of X(t,w) for t fixed:

$$x(t) = \frac{1}{w(t-\tau)} \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\tau, \omega) e^{+i\omega t} \, d\omega.$$

Fig. 7. The inverse Fourier transform equation

The inverse Fourier transform output was as follows:

- Signal size in frequency domain is 4655
- Signal size in Time domain is 1670786
- Signal size in Time domain(resamples) is 16707

As it seems the original time domain sequence size is way too large that's why we will be resampling the signal in order to be able to deal with it during modeling. This will reduce tremendous amount of data informativity but will help the correct architecture to converge and also will be friendly when it comes to computational resources.

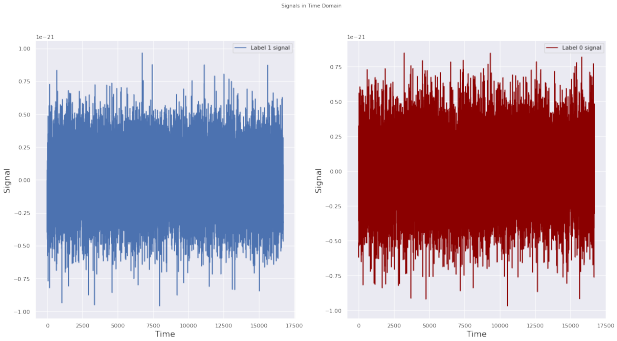

Fig. 8. The signal visualization in time domain

IV. Simulating Gravitational Waves

The gravitational waveform simulation of Rirориro is based upon the methods of Buskirk and Babiuc-Hamilton (2019), a paper which describes a computational implementation of an earlier theoretical gravitational waveform model by Huerta et al. (2017), using post-Newtonian expansions and an approximation called the implicit rotating source to simplify the Einstein field equations and simulate gravitational waves. Rirориro's calculation of signal-to-noise ratios (SNR) of gravitational wave events is based on the methods of Barrett et al. (2018), with the simpler gravitational wave model Findchirp (Allen et al. (2012)) being used for comparison and calibration in these calculations.

The implemented Python function returns two waves that represent orthogonal/diagonal waves. The output timescale that is returned is non-linear, so to convert these signals into uniform sampled signals as in the dataset, we need to resample. The function below will resample the gravitational wave signals to 2048Hz. It is crude though, based on nearest sample, but good enough for studying spectrums. Interpolation would be more proper.

When a gravitational wave passes by Earth, it squeezes and stretches space. LIGO can detect this squeezing and stretching. Each LIGO observatory has two "arms" that are each more than 2 miles (4 kilometers) long. A passing gravitational wave causes the length of the arms to change slightly. The observatory uses lasers, mirrors, and extremely sensitive instruments to detect these tiny changes.

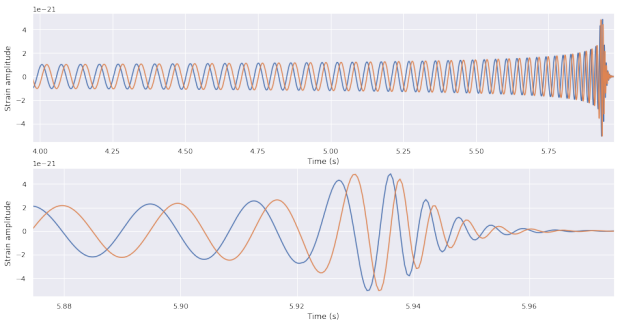

Fig. 9. The signal strain amplitude graph

Now after resampling the signal to 2048Hz (only the orthogonal part)

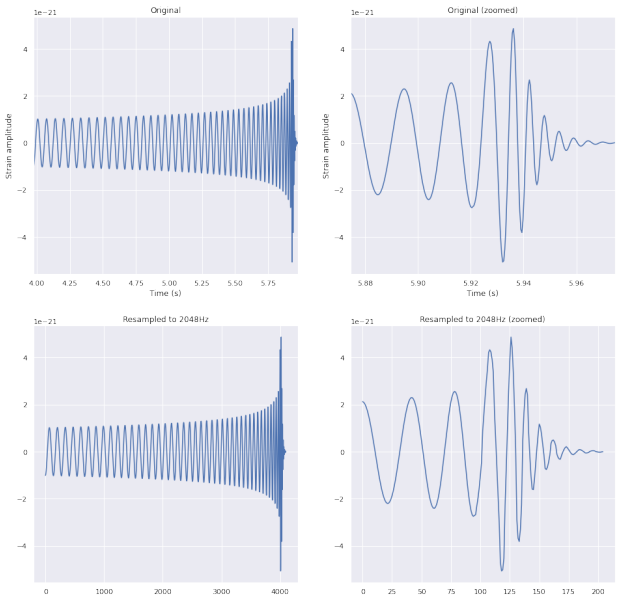

Fig. 10. The signal strain amplitude resampled to 2048Hz

Now we explore the amplitude vs. distance graph (inverse square law verification):

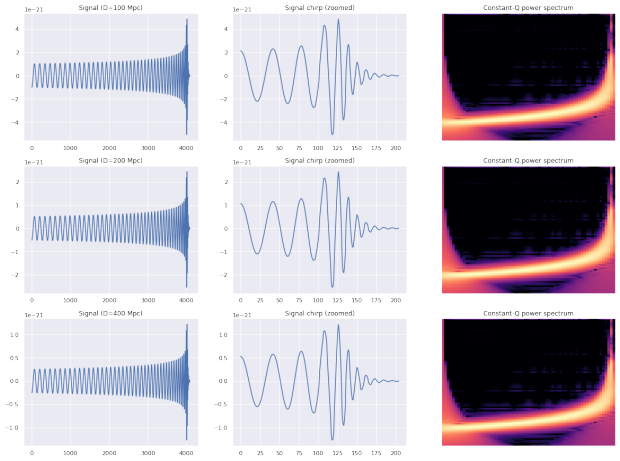

Fig. 11. The signal strain amplitude vs. distance

Surprisingly we can notice that gravitational waves amplitude doesn't follow the inverse square law but why?

In order to answer this question we have to explain the difference between monopolic, diapolic and quadrapolic signals. First off, there are fundamental ways that light and gravitational waves are the same. They both:

- do carry energy,
- do reach infinite distances,
- do spread out over space (in roughly a sphere) as you move farther away,
- and will be detectable, at a certain distance, in proportion to the magnitude of the signal.

Because the geometry of space is the same for both light and gravitation, the difference between these two behaviors must lie in the nature of the signal that we can detect.

To understand that, we need to understand how gravity is a fundamentally different kind of force than electromagnetism. This will lead us to better understand how gravitational radiation (our gravitational waves) behave differently than electromagnetic radiation (light) when we allow it to propagate across the vast distances of intergalactic space.

If you want to create electromagnetic or gravitational radiation. The simplest way you could imagine — which (spoiler) doesn't work — would be to spontaneously create or destroy charge in a region of space. Having a charge pop into (or out of) existence would create radiation of a very specific type: monopole radiation. Monopole radiation is what happens when you have a change in the amount of charge that's present.

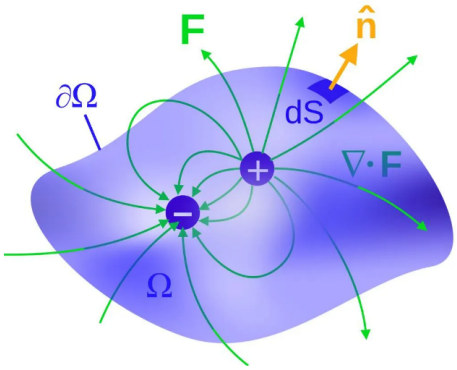

Fig. 12. Charge preservance in electromagnetism

We cannot do this for either electromagnetism or gravitation, however. In electromagnetism, electric charge is conserved; in gravitation, mass/energy is conserved. The fact that we don't get monopole radiation is important for the stability of our Universe. If charge or mass could spontaneously be created or destroyed, existence would be extremely different!

If charge and mass/energy are conserved, then the next step is to either move your charges (or masses) rapidly back-and-forth, or to take charges of opposite signs and change the distance between them. This would create what we call dipole radiation, which changes the distribution of charge without changing the total amount of charge.

In electromagnetism, this creates radiation, because moving an electric charge back-and-forth changes the electric and magnetic fields together. This matters, because changing electric and magnetic fields that are mutually perpendicular to each other and in-phase if wis what an electromagnetic wave actually is. This is the simplest way to make light, and it radiates just like you're familiar with. The light carries energy, and the energy is what we detect, which is why objects appear dimmer as 1/r2 the farther away they are.

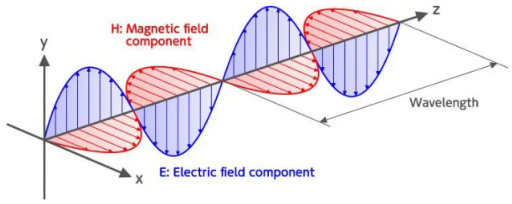

Fig. 13. Electromagnetic wave propagation

In gravity, however, freely moving a mass doesn't make gravitational radiation, because there's a conservation rule about masses in motion: the conservation of momentum. Similarly, separating masses doesn't make gravitational radiation either, because the center of mass remains constant. There's also a conservation rule about masses moving at a certain distance from the center of mass: the conservation of angular momentum.

Because energy, momentum, and angular momentum are conserved, you have to go past both monopole and dipole moments; you need a specific change in how the masses are distributed around their mutual center of mass. The simplest way to imagine this is to take two masses and have them mutually rotate around their center of mass, which results in what we call quadrupole radiation.

The amplitude of gravitational quadrupolar radiation falls off as 1/r, meaning the total energy falls off as 1/r2, just as it did for electromagnetic radiation. But this is where the fundamental difference between gravitation and electromagnetism comes in. There's a big difference between what you can physically detect for quadrupole and dipole radiation.

For electromagnetic (dipole) radiation, when the photons hit your detectors, they get absorbed, causing a change in the energy levels, and that change in energy — which remember, falls off as 1/r2 — is the signal you observe. That's why objects appear to dim according to an inverse square law.

For gravitational (quadrupole) radiation, however, it doesn't get directly absorbed in a detector. Rather, it causes objects to move towards or apart from one another in proportion to the amplitude of the wave. Even though the energy falls off as 1/r2, the amplitude only falls off as 1/r. That's why gravitational waves fall off according to a different law than electromagnetic waves.

But the amplitude, as we received it, compressed and expanded the entire Earth by about the diameter of three protons. The energy is huge and falls off as 1/r2, but we cannot detect energy for gravitational waves. We can only detect amplitude, which (thankfully) only falls off as 1/r, which is a very good thing. The amplitudes may be tiny, but if we can detect any signal at all, it's only a small step forward to detecting that same magnitude signal at any distance.

V. Gravitational Wave Signal Generation

The unusual dataset division which is normally splits as 80:20 but in our case it is 1:16 indicates that the competition creators are encouraging participants to generate their own data.

Standard CW signals can be parameterised in terms of two sets of parameters: the Doppler-modulation parameters gamma and the amplitude parameters A.

The former encode how the frequency of a signal modulates due to its intrinsic frequency evolution and the movement of the Earth in the Solar system, while the latter describes the overall amplitude of a CW depending on the parameters of the source.

For a CW emitted by a rapidly-spinning and isolated neutron star (NS), Doppler-modulation parameters include the frequency F0 and the linear spindown parameter F1, both taken at a reference time tref, and the sky position in terms of the right ascension Alpha and declination Delta angles of equatorial coordinates. Amplitude parameters, on the other hand, include the average amplitude of a CW signal h0, the initial phase of the signal phi, the polarization angle psi and (the cosine of) the inclination angle of the source cosi, which gives us the relative orientation of the NS with respect to the detector.

As described before, the amplitude of a CW signal is usually expressed in terms of the noises's amplitude using depth D or signal-to-noise ratio (SNR) p. For our purposes, the former is essentially a quotient

$$\mathcal{D} = \frac{\sqrt{S_{\mathrm{n}}}}{h_0}$$

Fig. 14. The noises's amplitude using depth expression

while the latter is a more involved expression which also depends on the duration of the dataset at hand and the detector's response function. It is important to note, however, that p and D scale reciprocally: "weak" signals have a low SNR and a high depth (since they are "buried deeper into the noise" than a strong signal).

As mentioned before from the split of train and test datasets the challenge creators are encouraging participants to generate their own data but in out case we will keep it at the point of generating samples for explanations and the imbalanced classes problem will be solved with the class weight parameter while training.

A specific sample requires of background noise and optionally a signal. In order to generate noise, one needs to specify a set of detectors (H1 or L1 in this case), the duration of the sample and the Amplitude Spectral Density of the noise sqrtSX. CW analyses are simple in this front, as sqrtSX is proportional to the (stationary) standard deviation of an underlying zero-mean Gaussian process.

Sample duration can be specified in two ways. If the sample contains contiguous data (i.e. the detector was taking science-quality data uninterrupted), one can simply specify the starting time and duration of the sample using tstart and duration. Data with gaps, on the other hand, can be generated by specifying a specific set of timestamps using the timestamps option.

Data is saved as a list of Short Fourier Transforms (SFTs). The duration and windowing of these SFTs can also be

modified using Tsft, SFTWindowType and SFTWindowBeta. Most analyses tune Tsft around 1800 seconds order to ensure the power of a putative CW signal stays within a bin.

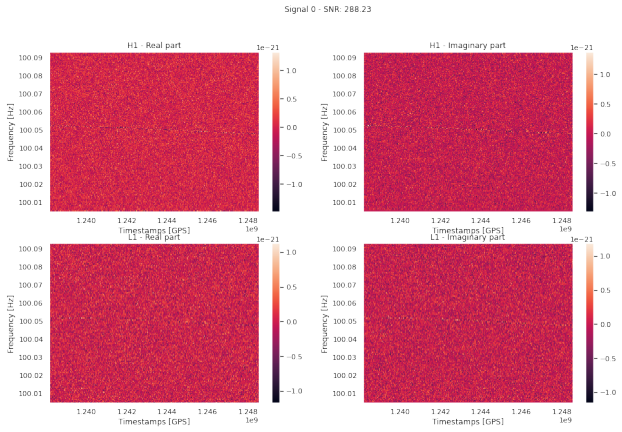

Fig. 15. Generated data sample spectrogram for real and imaginary parts of the signal from H1 and L1 detectors

VI. Modeling In Time Domain Using LSTM Based Network

Long short-term memory (LSTM) is an artificial neural network used in the fields of artificial intelligence and deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. Such a recurrent neural network (RNN) can process not only single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition, machine translation, robot control, video games, and healthcare. LSTM has become the most cited neural network of the 20th century. LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the vanishing gradient problem that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

The compact forms of the equations for the forward pass of an LSTM cell with a forget gate are:

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
\tilde{c}_t &= \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \sigma_h(c_t)
\end{aligned}
$$

Fig. 16. LSTM gates equations

When it comes to the proposed LSTM architecture, it comes as follows:

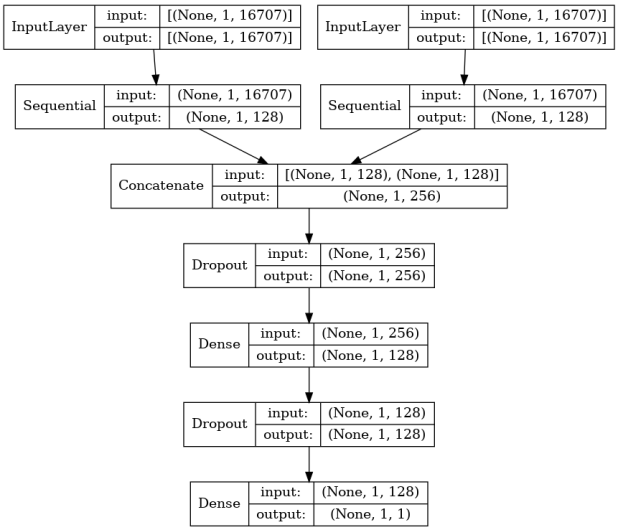

Fig. 17. LSTM architecture with 17,931,265 trainable parameters

VII. Modeling In Time Domain Using transformer Based Network

The paper 'Attention Is All You Need' introduces a novel architecture called Transformer consisting mainly of an encoder and a decoder. Both encoder and decoder are composed of modules that can be stacked on top of each other multiple times, which is described by Nx in the figure. We see that the modules consist mainly of Multi-Head Attention and Feed Forward layers. The inputs and outputs (target sentences) are first embedded into an n-dimensional space since we cannot use strings directly. Recurrent Networks were, until now, one of the best ways to capture the timely dependencies in sequences. However, the team presenting the paper proved that an architecture with only attention-mechanisms without any RNN (Recurrent Neural Networks) can improve on the results in translation task.

One slight but important part of the model is the positional encoding of the different words. Since we have no recurrent networks that can remember how sequences are fed into a model, we need to somehow give every word/part in our sequence a relative position since a sequence depends on the order of its elements. These positions are added to the embedded representation (n-dimensional vector) of each word.

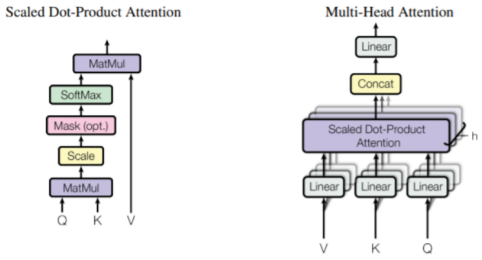

Fig. 18. Attention mechanism illustration

Let's start with the left description of the attention-mechanism. It's not very complicated and can be described by the following equation:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$$

$$\text{where} \quad Q = YW^Q,\ K = XW^K,\ V = XW^V$$

Fig. 19.

Q is a matrix that contains the query (vector representation of one word in the sequence), K are all the keys (vector representations of all the words in the sequence) and V are the values, which are again the vector representations of all the words in the sequence. For the encoder and decoder, multi-head attention modules, V consists of the same word sequence than Q. However, for the attention module that is taking into account the encoder and the decoder sequences, V is different from the sequence represented by Q.

This means that the weights are defined by how each word of the sequence (represented by Q) is influenced by all the other words in the sequence (represented by K). Additionally, the SoftMax function is applied to the weights a to have a distribution between 0 and 1. Those weights are then applied to all the words in the sequence that are introduced in V (same vectors than Q for encoder and decoder but different for the module that has encoder and decoder inputs).

The attention-mechanism can be parallelized into multiple mechanisms that can be used side by side. The attention mechanism is repeated multiple times with linear projections of Q, K and V. This allows the system to learn from different representations of Q, K and V, which is beneficial to the model. These linear representations are done by multiplying Q, K and V by weight matrices W that are learned during the training.

Those matrices Q, K and V are different for each position of the attention modules in the structure depending on whether they are in the encoder, decoder or in-between encoder and decoder. The reason is that we want to attend on either the whole encoder input sequence or a part of the decoder input sequence. The multi-head attention module that connects the encoder and decoder will make sure that the encoder input-sequence is taken into account together with the decoder input-sequence up to a given position.

After the multi-attention heads in both the encoder and decoder, we have a pointwise feed-forward layer. This little feed-forward network has identical parameters for each position, which can be described as a separate, identical linear transformation of each element from the given sequence.

After the L1 and H1 time domain(obtained after performing inverse short time Fourier transform on the original data) parts are passed to the transformer network which will work as an auto encoder to output feature vector that will be passed then to an MLP layer then processed to the output layer with the sigmoid activation function to output the final classification probability.

When it comes to the proposed transformer architecture, it comes as follows:

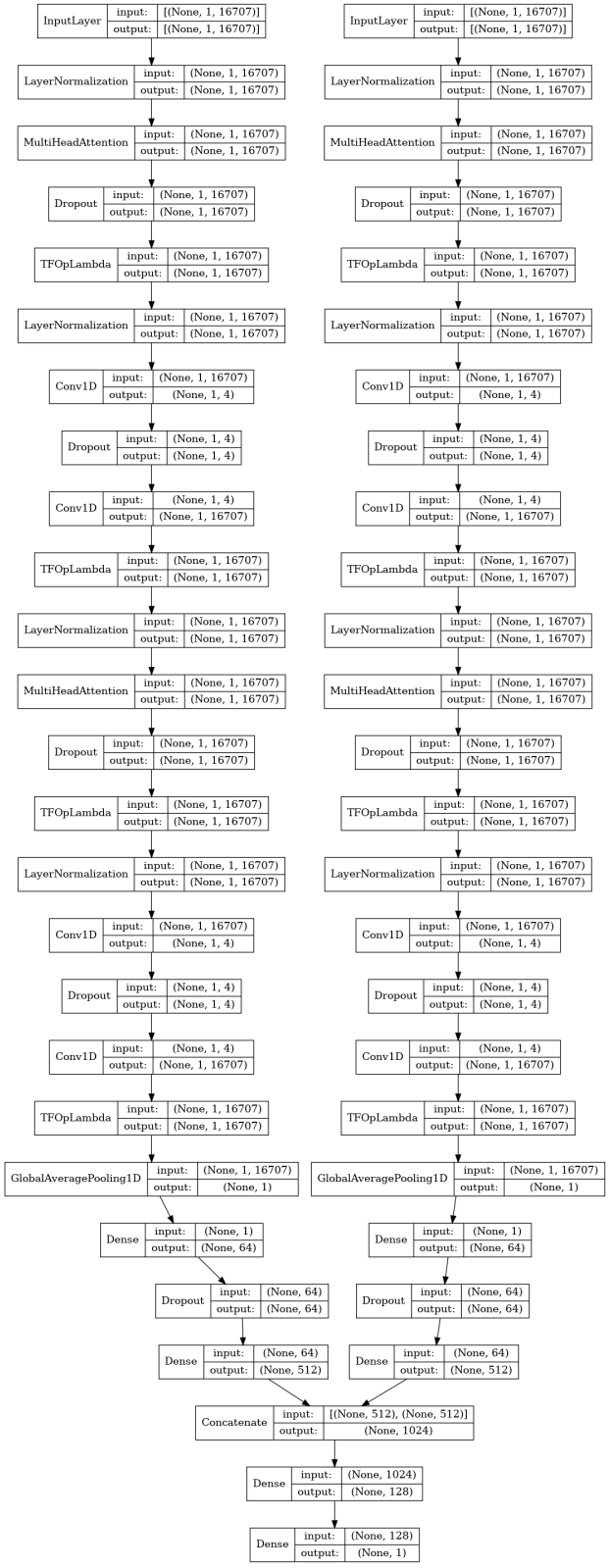

Fig. 20. Transformer architecture with 138,003,641 trainable parameters

VIII. Modeling In Frequency Domain Using CNN Based Network And Spectrogram As Input

Convolutional neural network (CNN, or ConvNet) is a class of artificial neural network (ANN), most commonly applied to analyze visual imagery. CNNs are also known as Shift Invariant or Space Invariant Artificial Neural Networks (SIANN), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation-equivariant responses known as feature maps. Counter-intuitively, most convolutional neural networks are not invariant to translation, due to the downsampling operation they apply to the input. They have applications in image and video recognition, recommender systems, image classification, image segmentation, medical image analysis, natural language processing, brain–computer interfaces, and financial time series.

When it comes to the proposed CNN architecture, it comes as follows:

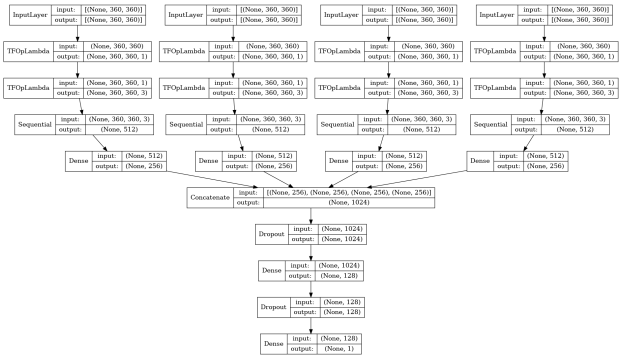

Fig. 21. CNN architecture with 1,097,985 trainable parameters

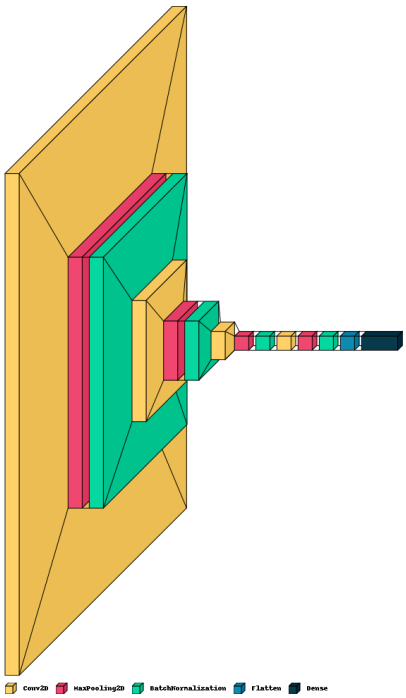

Fig. 22. CNN base architecture visualization

IX. Modeling In Frequency Domain Using ViT Based Network And Spectrogram As Input

The concept of Vision Transformer (ViT) is an extension of the original concept of Transformer. It is only the application of Transformer in the image domain with slight modification in the implementation in order to handle the different data modality. More specifically, a ViT uses different methods for tokenization and embedding. However, the generic architecture remains the same. An input image is split into a set of image patches, called visual tokens. The visual tokens are embedded into a set of encoded vectors of fixed dimension. The position of a patch in the image is embedded along with the encoded vector and fed into the transformer encoder network which is essentially the same as the one responsible for processing the text input. There are multiple blocks in the ViT encoder and each block consists of three major processing elements: Layer Norm, Multi-head Attention Network (MSP) and Multi-Layer Perceptrons (MLP). Layer Norm keeps the training process on track and let model adapt to the variations among the training images. MSP is a network responsible for generation of attention maps from the given embedded visual tokens. These attention maps help network focus on most important regions in the image such as object(s).

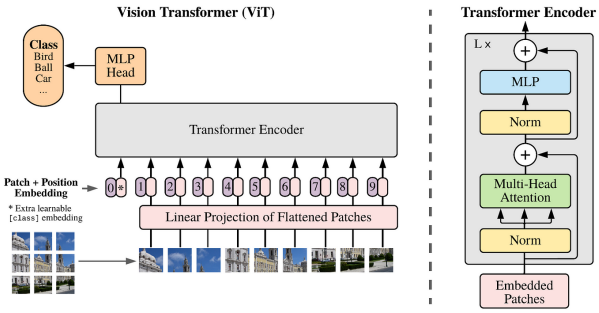

Fig. 23. Vision transformer illustration

The differences between CNNs and Vision Transformers are many and lie mainly in their architectural differences. In fact, CNNs achieve excellent results even with training based on data volumes that are not as large as those required by Vision Transformers. This different behaviour seems to derive from the presence in the CNNs of some inductive biases that can be somehow exploited by these networks to grasp more quickly the particularities of the analysed images even if, on the other hand, they end up limiting them making it more complex to grasp global relations.

On the other hand, the Vision Transformers are free from these biases which leads them to be able to capture also global and wider range relations but at the cost of a more onerous training in terms of data. Vision Transformers also proved to be much more robust to input image distortions such as adversarial patches or permutations. However, choosing one architecture over another is not always the wisest choice, and excellent results have been obtained in several Computer Vision tasks through hybrid architectures combining convolutional layers with Vision Transformers.

Implementing patches generation as a layer with the following characteristics:

- Image size: 360 X 360
- Patch size: 40 X 40
- Patches per image: 81
- Elements per patch: 1600

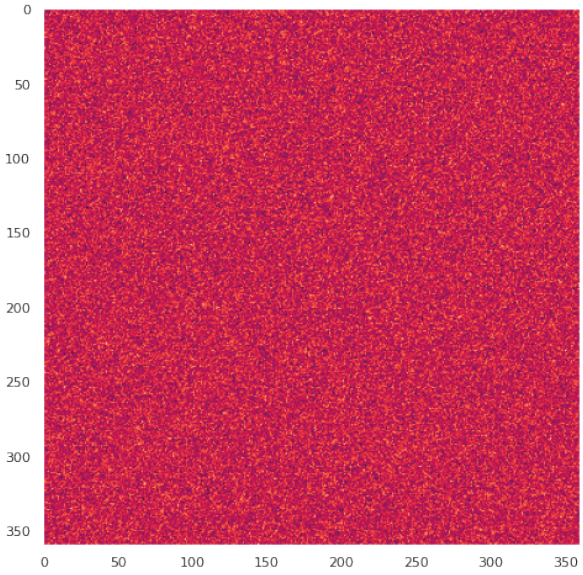

Fig. 24. Input image before being divided into patches

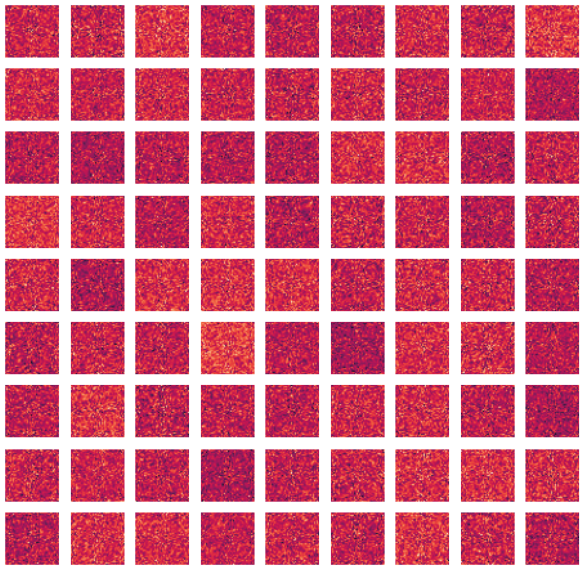

Fig. 25. Input image divided into patches

The PatchEncoder layer will linearly transform a patch by projecting it into a vector of size projection dim. In addition, it adds a learnable position embedding to the projected vector. The ViT model consists of multiple Transformer blocks, which use the layers.MultiHeadAttention layer as a self-attention mechanism applied to the sequence of patches. The Transformer blocks produce a [batchsize, numpatches, projectiondim] tensor, which is processed via a Dense head to produce the final output. So to sum this up we are using the vision transformer here to work as an image embedding layer which will extract the required features from the real and imaginary spectrogram images of the frequency component and then pass the feature vector to the following fully connected layer to further processing until we reach the final layer with the sigmoid activation function to output the prediction probability.

When it comes to the proposed ViT architecture, it comes as follows:

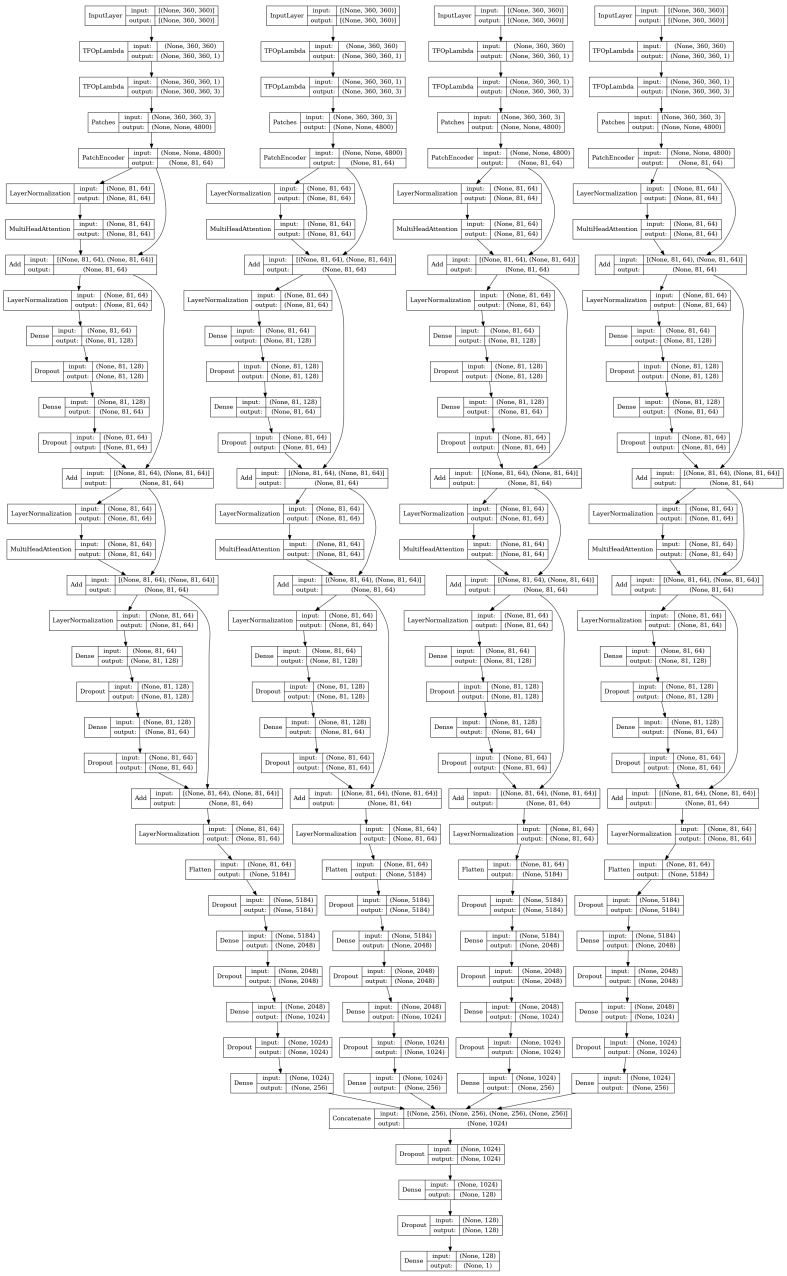

Fig. 26. ViT architecture with 53,965,057 trainable parameters

X. Results Interpretation

As expected the CNN model could converge faster 3x than the ViT model as the available dataset size isn't the ideal for such architecture(we're talking about 600 instances here) which is based on vision transformers which in turn needs more data than CNN to work properly. The models loss curves are descending but it doesn't mean that the models are learning well, that's why further INVESTIGATION and IMPROVEMENTS shall be done on this work to find out whether the imbalanced classes is the main issue here or if we need to try feature extraction techniques(noise cancellation filters, etc.) other than using the SFT's available. As when it comes to the F Measure value which is a metric used to evaluate the performance of a model. It combines precision and recall into a single score. The F-score is calculated as the harmonic mean of precision and recall. The harmonic mean gives more weight to lower values, so it puts more emphasis on high recall than high precision. The models achieved an average of 80.59% on training and an unchangeable average of 77.50% on validation meaning that more data is required to be generated to compensate the imbalance in the dataset.

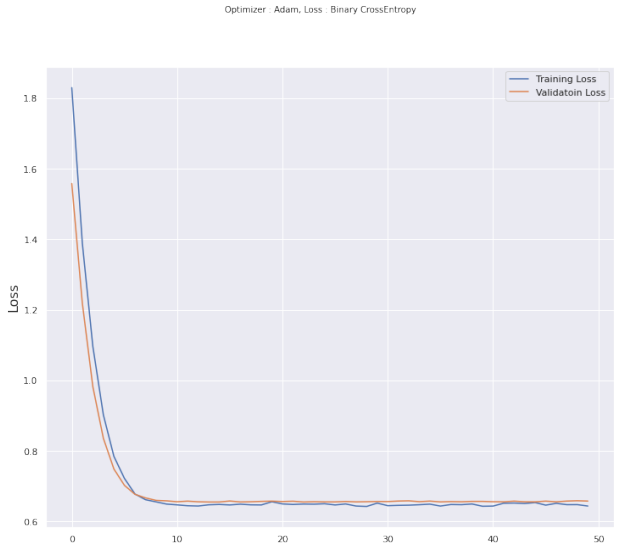

Fig. 27. LSTM based model learning curve

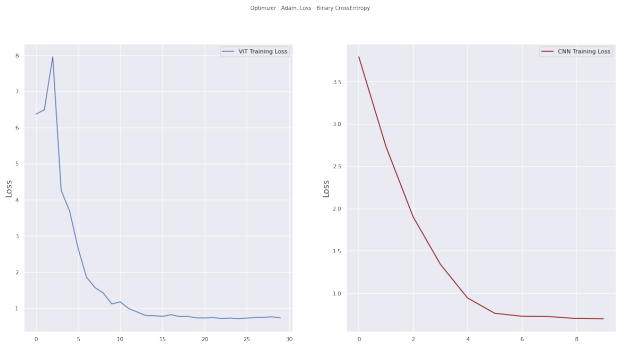

Fig. 28. ViT vs. CNN learning curve

Acknowledgment

The author would like to thank the European Gravitational Observatory for their great effort in organizing the G2Net competition and for providing the dataset to the scientific community. This dataset will be invaluable to anyone who is interested in unveiling the secrets of the universe.

XI. References

van Zeist, W.G.J., Stevance, H.F., and Eldridge, J.J. (2021). Rironiro: Simulating gravitational waves and evaluating their detectability in Python. Journal of Open Source Software, 6(59), 2968. DOI: 10.21105/joss.02968.

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. (pp. 5998-6008).

O'Shea, K., & Nash, R. (2023). An introduction to convolutional neural networks. IEEE Signal Processing Magazine, 30(1), 118-136.

Dosovitskiy, A., Beyer, H., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). Vision transformer. arXiv preprint arXiv:2002.04088.

NASA Space Place. (2023, March 8). Gravitational waves. Retrieved from https://spaceplace.nasa.gov/gravitational-waves/en/

ScienceDirect Topics. (2023, May 31). Short-time Fourier transform. Retrieved from https://www.sciencedirect.com/topics/engineering/short-time-fourier-transform

Berkeley Python Numerical Methods Group. (2023, June 1). HDF5 Files. Retrieved from https://pythonnumericalmethods.berkeley.edu/notebooks/chapter11.05-HDF5-Files.html

Abbott, R., Abbott, T. D., Abbott, F. R., Acernese, F., Ackley, K., Adams, C., ... & Zweizig, J. (2022). Searches for gravitational waves from known pulsars at two harmonics in the second and third LIGO-Virgo observing runs. The Astrophysical Journal, 935(1), 1. doi:10.3847/1538-4357/ac663c.