Machine Learning for transient noise event classification in LIGO and Virgo

Elena Cuoco[1,2,3,⋆] on behalf of *LIGO Virgo* Collaboration

[1] *European Gravitational Observatory, V. Amaldi, Cascina (Pisa)*
[2] *Scuola Normale Superiore, P.zza dei Cavalieri, 7, Pisa*
[3] *INFN, Largo Bruno Pontecorvo, 3, Pisa*

Abstract.
Noise of non-astrophysical origin contaminates science data taken by the Advanced Laser Interferometer Gravitational-wave Observatory (LIGO) and Advanced Virgo Gravitational-wave detectors. Characterization of instrumental and environmental noise transients has proven critical in identifying false positives in the first observing runs, for this reason we are investigating methods which can help with the characterization and classification of transient signals. Machine-Learning techniques have, in recent years, become more and more reliable and can be efficiently applied to our problems. In this paper, we review the machine learning methods used in transient-signal classification, showing examples of applications on simulated data from previous published papers and on real data taken by the LIGO detectors during the first observing run.

Introduction

The detection of gravitational waves (GWs) with the event GW150914[1] has inaugurated the era of gravitational astronomy and opened new challenges for the multi-messenger study of cosmic sources. Due to their sensitivity, the Advanced LIGO [2] and Advanced Virgo [3] interferometers probe a much larger volume of space and expand the capability of discovering new gravitational wave emitters. Most of the data collected by GW interferometers are essentially background noise: at low frequencies (<10 Hz) the main contribution to the noise is due to seismic ground motion and to the gradient of the local gravitational field, at higher frequencies the thermal noise of the subsystems, e.g. the suspensions of the mirrors, plays a dominant role. Above ∼200 Hz the most important limitation to the detector sensitivity is the shot noise of the laser light circulating in the detector cavities [4]. GW data are recorded as a time series, where, in a featured background noise, we have to catch and identify GW signals from astrophysical sources. To this aim, the understanding and characterization of the background noise is fundamental for the data conditioning for the detection pipelines, and for accurate astrophysical parameter estimation of the emitting GW source. GW data are characterized by the presence of non-stationary and non-Gaussian noise and contain several noise artifacts, called glitches. We have to consider that a single GW detector typically produces data with a rate of 7-8 Tb per day with a flux of 40Mb/s. The data must be analysed in the fastest and most efficient way

⋆e-mail: elena.cuoco@ego-gw.it

to increase the detection confidence and to obtain information in real time about likely noise sources, and to help the fast alert system for telescopes which will look for the electromagnetic counterparts. This is the main reason for the investigation of Machine Learning approaches for GW detector noise issues and the classification of the noise transients. Among the possible approaches to the automatic analysis and classification of glitches, machine learning looks very promising.

Deep learning is an area of machine learning that combines the architecture of artificial neural networks with the power of machine learning algorithms (see, e.g. [5]). By stacking many layers of artificial neurons in a neural network architecture, it is possible to build very efficient tools that perform classification or regression tasks. Deep learning algorithms have been suggested for applications in GW physics [6], highlighting the potential of convolutional neural networks (CNNs).

Different methods have been developed to automatically classify glitches in advanced GW detectors. For instance, in [7], three algorithms were tested on Advanced LIGO Engineering Run data : 1) PCAT, based on Principal Component Analysis [8], 2) LALInferenceBurst, based on Bayesian parameter estimation [9], and 3) WDF-ML, that couples a Wavelet Detection Filter [10] with an unsupervised machine learning clustering based on a Gaussian Mixture Model algorithm [11]. In [12], a supervised Bayesian classifier called Difference Boosting Neural Network (DBNN) has been tested on LIGO S6 data, also showing a very good performance. In the following sections, we will report some examples of the application of the machine learning and deep learning techniques used for the classification of GW transient signals using both simulated data and real data from the first observing run (O1) of the LIGO detectors. This article will report only a few examples of applications from previous works [13], [14] [15]. It is not an exhaustive report of the many transient signal classification works developed by different groups within the LIGO/Virgo collaboration, such as [6, 12, 15, 16]

1 Applications on simulated data

We will show two applications of classification tasks for transient signals using two different approaches, based on a machine learning pipeline and on a deep learning pipeline. To evaluate algorithm performance, we simulate a synthetic data set for GW LIGO-like detectors, where we inject some categories of glitches with different shapes and intensities. The simulated glitches have been added to Gaussian noise generated from the sensitivity curve of real interferometers. Here we used the sensitivity of LIGO Hanford detector (H1) available online[1].

1.1 Data simulations

We simulate six families of short time signals, that approximate the time-frequency evolution of main noise transients observed in the real GW detector[17]. Each glitch is described by the strain time series h(t), and its injected SNR is the SNR obtained by the Wiener filer [18] with that given waveform. The simulated families are detailed in [13] and are Gaussian glitches (GAUSS), sine-gaussian (SG), ringdown (RD), whistle-like and scattered light-like. We also included a family of chirp-like transients (CHIRPLIKE), in order to show the potential of the machine learning pipeline in distinguishing transients of astrophysical origin and of known waveform, such as those produced by coalescing binaries. Figure 1 shows a sample gallery of glitches belonging to these families.

The times t_0 at which the glitches occur are drawn from a Poisson distribution with mean rate of 0.5 Hz. With this choice it might be that two glitches are nearby in time, thus offering the opportunity to test the classification performance and robustness against nearby glitches. The sampling rate of the

[1]Data publicly available at the LIGO Open Science Center https://losc.ligo.org/s/events/GW150914/P150914/fig1-observed-H.txt

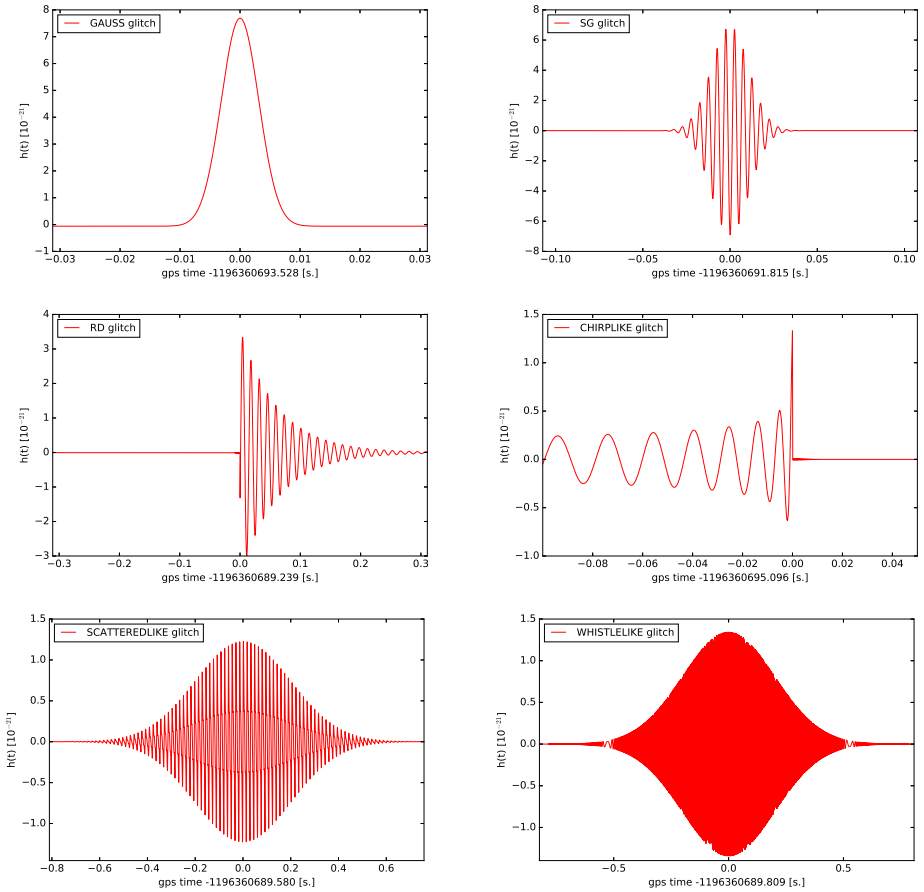

Figure 1: Gallery of simulated glitches used for this study. We show only high-amplitude glitches without the contribution of the Gaussian noise in order to highlight the temporal evolution of the transients, from [13].

simulated time series is 8192 Hz, which is half the sampling rate used for LIGO and Virgo [19]. We included in this set 2000 glitches for each family. In order to evaluate the accuracy of the algorithm as a function of the intensity of the transients, for each glitch we computed the optimal matched filter SNR. Figure 2 shows respectively the distributions in SNR and a time-frequency plot of the occurrence of simulated glitches.

1.2 Decision Trees pipeline: WDFX

The Wavelet Detection Filter (WDF) was used for the first time in the analysis of the association between gamma-ray bursts and a GW signal for the GRB 050915a detected by the Swift satellite in 2005, when the Virgo detector was engaged in one of its science runs, namely the C7 run [20]. WDF was also used in the first approaches for unsupervised classification for glitches [7, 21]. In this application, we set up a new version of the pipeline based on a first stage where the WDF algorithm

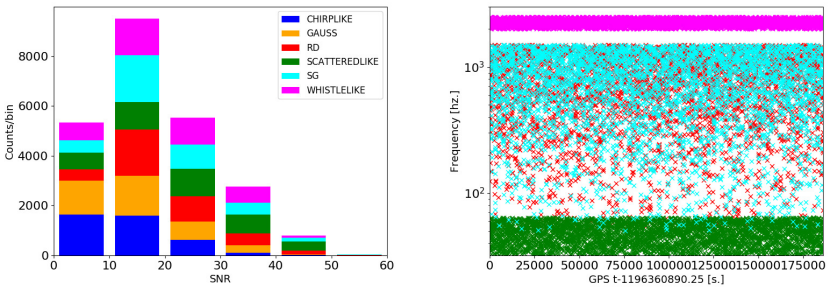

Figure 2: Distribution of simulated glitches. Top: Distribution of Signal-to-noise ratios. Bottom: Time-frequency distribution. The SNR distribution is the result of our choice to have an uniform spacing in the parameter space of the simulated glitches. This results in a larger number of low-SNR glitches, which is good since we want to have a large statistics of faint glitches, more difficult to detect and classify, from [13].

is applied and a second stage where the XGBoost [22] supervised classifier was used to identify the main types of signals. Wavelet-based algorithms are well tuned for the identification of transients signals; as different wavelet types could better match different waveform morphologies, WDF performs wavelet decomposition using different types of wavelet basis, including the Daubechies, Haar and spline wavelets [23–25]. The wavelet transform of a signal $f(t)$ is defined as the projection of f on the wavelet basis

$$Wf(a,b) = \langle f, \psi_{a,b} \rangle = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{b}} \psi^* \left(\frac{t-a}{b} \right) \, dt, \tag{1}$$

where ψ^* is the complex conjugate of the mother wavelet. The wavelet transform has a time frequency resolution that depends on the scale b. The time spread is proportional to b, and the frequency spread is proportional to the inverse of b. The factor a represents a scale, which dilates or compress the signal.

Before applying any detection and classification pipeline, the data should be conditioned. The whitening procedure [26] can remove the contribution of Gaussian colored noise. Many pipelines use whitening in the frequency domain, while the whitening applied here is based on a time-domain procedure, using an Auto Regressive (AR) fit to the data as described in [26, 27]. The whitening procedure is based on a Linear Predictor Filter, whose parameters are estimated through a parametric Auto Regressive (AR) model fit to the noise Power Spectral Density (PSD), as described in [26]. In this work we used a model with 4000 AR parameters to fit the noise and whiten the data. The goal of the supervised learning, and in this case supervised classification, is to reach a high accuracy in predicting the output variable on a given set of input data. EXtreme Gradient Boosting (XGBoost) implements the gradient boosting framework. The gradient boosting decision tree algorithm is a decision tree implementation in which the objects or classes are classified by moving them down the tree from the root, to some leaf node where the classification is generated. At each node of the tree there is a test placed on that object and, based on this the object can take different classifications corresponding to each branch starting at the node. Boosting in this framework consists of having a large number of weak learners or trees and train them multiple times in order to minimize the loss function of the mapping. As a first step we have created our training data by running WDF on the simulated data sets. The second step is finding the signals coincident within 0.1 secs with the injected

ones. WDF, with a threshold of $SNR = 2$, was able to detect 97% of the injected signals, considering all the signals injected at different SNR, which is a very high efficiency, considering that also signals with $SNR = 1$ were injected in the data. The average of accidental coincidences, regardless the values of injected SNR, is around 10%, with a time-shift of 1 sec.

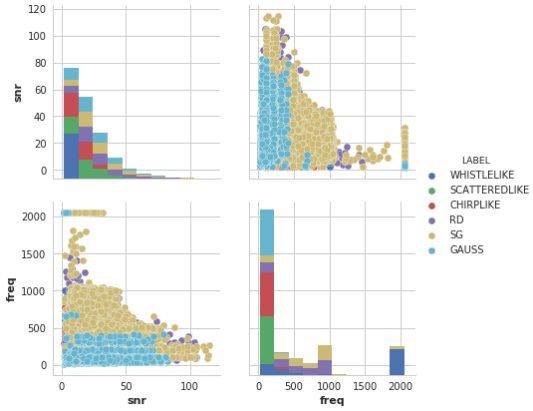

Figure 3: The SNR and frequency distribution of the detected triggers are reported in the diagonal of the plot. Upper right the scatter plot of SNR versus frequency, bottom left the scatter plot of frequency versus SNR, from [14]

.

In Fig. 3 we reported the meta-parameter values of the signals recovered by WDF. We used a hyper-parameters selection method for the XGBoost classifier using a grid search cross validation [28]. The parameter selection could be further tuned, using also some features selection [29], which have not been applied in this work. We performed 2 main classification tasks:

- Binary classification: Chirp-like signal versus the rest,
- Multi-label classification.

To prepare our training data we selected only the coincident triggers, adding almost 2000 not coincident glitches labeled as 'NOISE', for a total of 25000 triggers. We then create train/validation/test sets in the ratio 70/15/15, by random shuffling the input data set.

We obtained an overall accuracy for binary classification of more than 95%, with a detailed confusion matrix reported in fig. 4. The noise events set was made by 'glitches' simulated events different by chirp-like and some random noise glitches. The chirp-like events have been correctly identified for 81% of the cases. While noise events for 98% ones. In the multi-class case, the overall accuracy obtained was more than 82%, detailed in the confusion matrix of fig. 5. The classification of longer signals has a higher accuracy with respect to the short ones, in particular for the Sine Gaussian waveforms. This could be due to the selection of a small overlap between two consecutive windows, in the event a signal falls between them.

It has been shown that the WDF extracted features and the XGBoost algorithm can classify different datasets of simulated waveforms with an accuracy > 90%. The power of the methods relies on the possibility to have one in-time classifier for the data acquired by GW detectors.

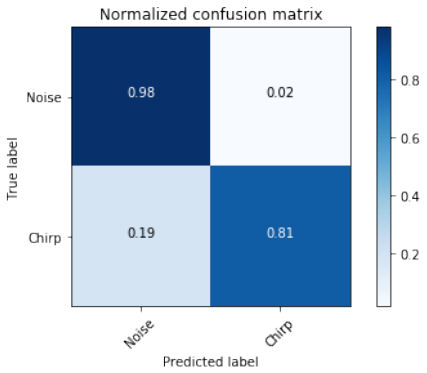

Figure 4: XGBoost binary classification confusion matrix, from [14].

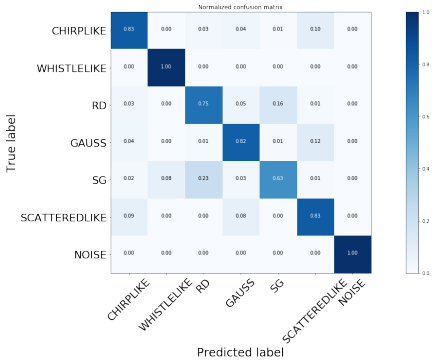

Figure 5: XGBoost multi label classification confusion matrix, from [14].

1.3 Deep Learning pipeline: Convolutional Neural Network

The Deep Learning pipeline will take as input features the images representing the time-frequency mapping of the transient signals. We started from the same synthetic data set and in order to build our training, validation and test sample we applied a whitening procedure to the simulated time series and then we built the image spectrograms over a time window of 2 seconds centered on the glitch time. The size of the output images are 241 pixels along the time axis and 513 pixels along the frequency axis. We then applied clipping and contrast stretching. In order to test the capability of the CNN to discriminate glitches from noise, we selected 2000 random windows of 2 seconds with no glitches and we built a seventh class, called NOISE and added to our training, validation and test samples. We also scaled the sizes of images by a factor of 0.55 on both axis. This scaling reduces the memory required without impacting significantly on the performance of the classification. In Figure 6 we show a gallery of spectrograms of whitened simulated glitches without any processing. If the images of the glitches are not available, we introduce an image preparation stage, that converts the whitened [30] time series to images showing the time-frequency evolution of the glitches.

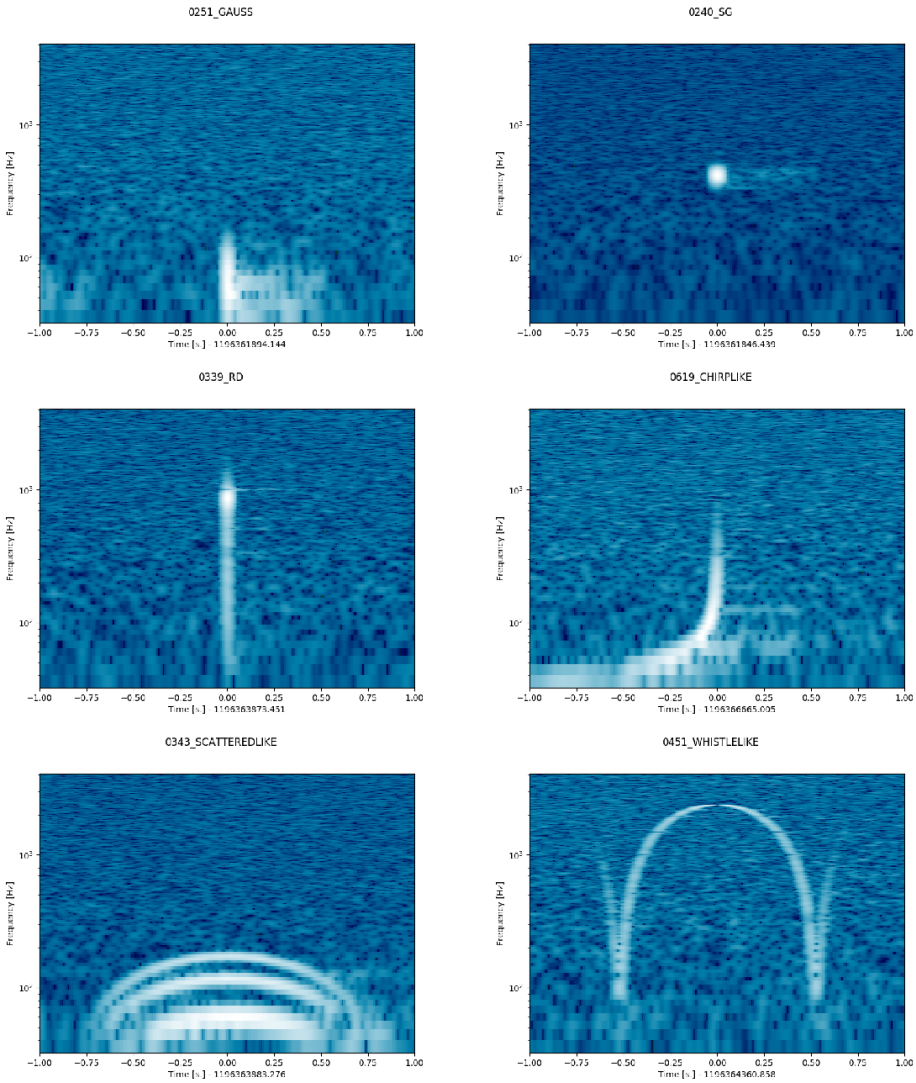

Figure 6: Gallery of sample spectrograms obtained using simulated glitches from simulated sets 1 and 2. (a): Gaussian (SNR∼7) (b): Sine Gaussian (SNR∼22) (c) Ringdown (SNR=12) (d) Chirp-like (∼8) (e) Scattered light-like (SNR∼24) (f) Whistle-like (SNR∼10), from [13].

In our framework of supervised CNNs, a simple way to implement the anomaly detection is using a "signal vs noise" binary classification approach[2]. To this purpose, the images containing glitches can be grouped in a SIGNAL class, and a set of time-frequency images of data with no glitches can be used to produce a set of images for a NOISE class, since they contain only the non transient component of the detector noise.

[2]From now on, by *signal* we mean a transient glitch signal, and for *noise* we mean the stationary, non-transient, detector noise

Since CNNs work smoothly on predicting many classes, we implemented in our pipeline a multi-class classification. For each glitch image, the pipeline computes the probability of belonging to one of the N_c glitch class or to the NOISE class.

For the training phase, we split our glitch set into training, validation, and test sets with the ratio 70/15/15. In a realistic situation, it is possible that the number of images is different for each class [15]. To fix this class imbalance problem, we implemented a balancing step based on image augmentation procedures [31]. Given the appearance of the time-frequency images of glitches, we perform the augmentation by simply shifting the images by a small amount, typically of ∼10% of the size of the image. We have found that rotating and flipping the images is not very important on glitch images, that show always the same orientation and direction in time.

Once the training phase has been completed, the model and the weights computed by the CNN are saved on disk. Using this model, the pipeline can perform classification starting from the time series or the image of the new glitches. The output of this step is the class of the glitch, with the associated probability of belonging to that class. The CNN architecture has been built using the Python-based libraries Theano [32] and Keras [33], which provide high performance and easy implementation of prototypes. The code has been developed and tested on a GPU NVIDIA GeForce GTX 780 running CUDA 8.0 and the accelerated libraries cuDNN developed for deep learning applications [34]. We

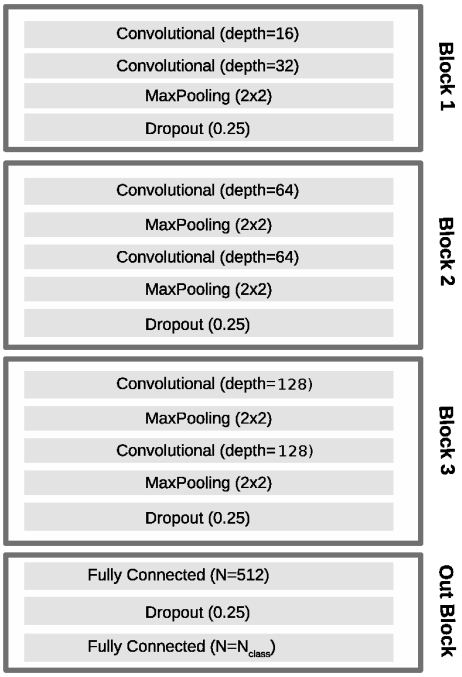

Figure 7: Diagram showing the structure of the CNN used in this work The size of the convolutional kernels is always 3x3. Here the output size is the number of classes N_c, but becomes N_c+1 if the NOISE class is used, from [13].

have tested difference CNN configurations and we selected an architecture based on four main blocks. The first block is made of a 16 and a 32-deep layer, while the second and the third block contains 2 layers with depth of 64 and 128 respectively. Each convolutional kernel has a 3×3 and a *ReLU* ac-

tivation function. Each block contains Maximum Pool layers (Keras MaxPooling2D) and is followed by a dropout layer (Keras Dropout). The last block is made of a fully connected layer (Keras Dense) with 512 neurons and by an output layer with a size equal to the number of classes N_c and with a *SoftMax* activation function. If the NOISE class is added, the output size is increased by one unit. Figure 7 shows the blocks in the CNN used for the pipeline.

The output of the CNN is a vector of probabilities of glitches belonging to each of the 7 classes (6 glitch families and NOISE). In order to assign the class to each glitch, we take the class with the highest probability. This choice can be modified in order to have hints on glitch belonging to new, unknown, classes, e.g. by assigning the class only if the probability is above a certain threshold.

We then computed the confusion matrix for the classification, and the results are shown in Figure 8. We see that the CNN reached an accuracy for each class of greater than 99%.

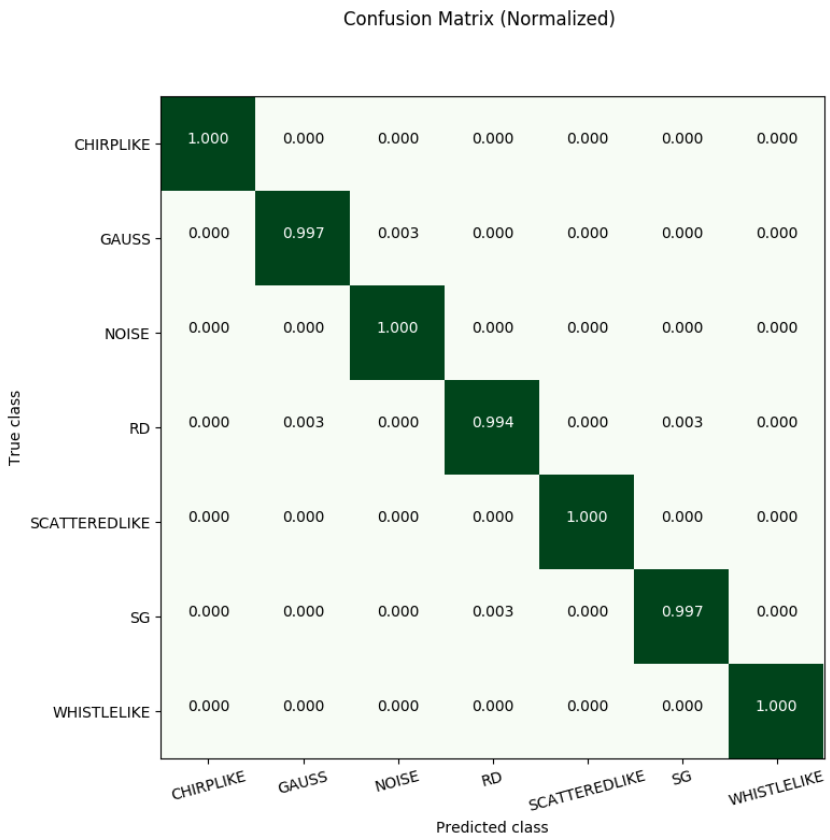

Figure 8: Confusion matrix for the multi-label classification based on images, evaluated on the test set, from [13]

The results show that image-based deep learning algorithms are very powerful in discriminating glitch transients from noise in GW interferometers. Furthermore, the CNNs use the time-frequency dependence of the glitches to provide an accuracy of > 99% in classification of glitches among different classes [13].

2 Application on real data

We will show one example application of deep learning, with the use of the CNN architecture described previously on real data taken during O1 [2]. The O1 run spanned September 2015 through January 2016. It produced two detections that were reported by the LIGO-Virgo collaboration. To apply such a kind of supervised approach, we need a labelled data set of glitch spectrograms for the O1 data. The labelling process is fundamental for any supervised machine learning pipeline, but it is also time-consuming. That's why the unsupervised solution was investigated in the past [7, 21] either on simulated data or on real data from LIGO engineering runs. LIGO launched the project GravitySpy [15] for citizen scientists to classify detector transients and potential GW signals. This ambitious project seeks to leverage the advantages of citizen science and machine learning to analyze and characterize transients in GW data, which will improve the effectiveness of GW searches. The idea is that as the citizen scientists make identifications, these classifications are provided to the GW data analysis pipelines in order to make use of them. In addition, as the citizen scientists identify new categories of transients, the machine learning algorithms are automatically updated to use these in their classifications. In figure 9 we reported an examples of some glitch categories identified and classified by the Gravity Spy project.

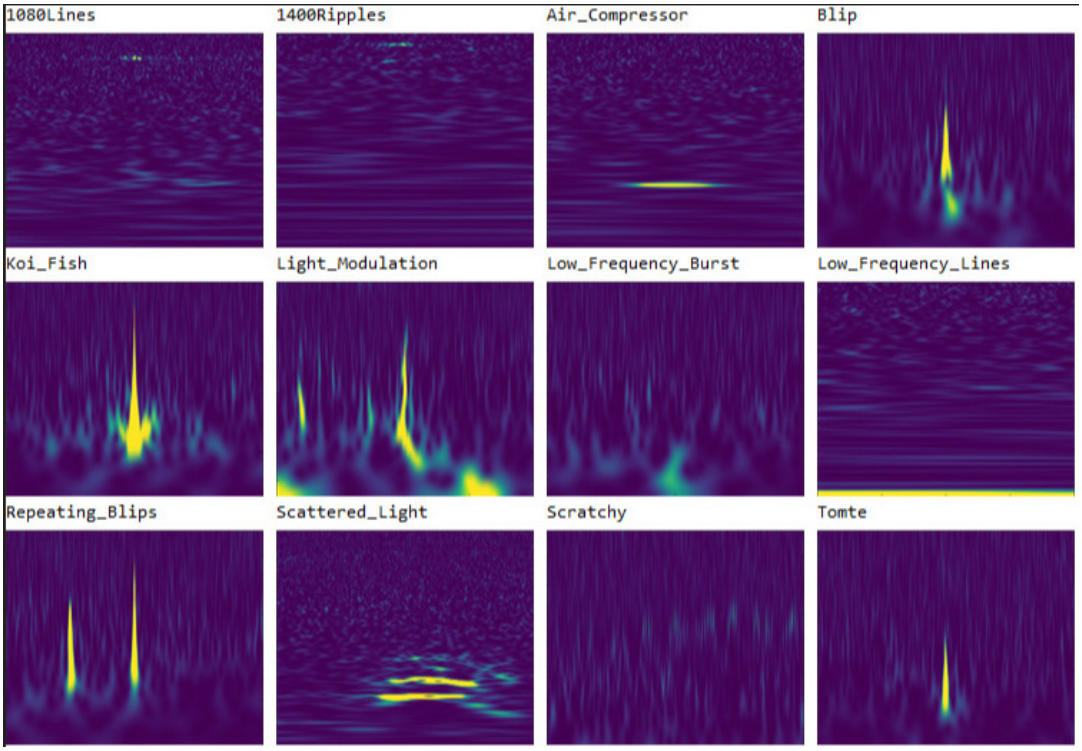

Figure 9: Time-frequency representation of labelled glitches due exclusively to noise in the GW mainstream data for LIGO detectors[15].

In Table 1, we report the different glitch categories classified by the Gravity Spy project[15]. We have 20 different classes, which are not balanced. We cured this problem with image duplication

Table 1: Main glitch categories in the GW mainstream data for LIGO detectors [15]

Glitch name	# H1	# L1
Air compressor	56	3
Blip	1495	374
Chirp	34	32
Extremely loud	266	188
Helix	3	276
Koi fish	580	250
Light modulation	568	5
Low_frequency burst	184	473
Low_frequency lines	82	371
No_glitch	117	64
None_of_the_above	57	31
Paired doves	27	-
Power_line	274	179
Repeating blips	249	36
Scattered_light	393	66
Scratchy	95	259
Tomte	70	46
Violin_mode	179	-
Wandering_lines	44	-
Whistle	2	303

and image augmentation to have a balanced data set. We used the CNN architecture described in the previous section to estimate the accuracy we can reach on real data, considering the presence of 20 different categories of glitches. Even on real data, we obtained an accuracy of $> 98\%$. Most of not well classified glitches are due to the poor quality of the images, with low contrast.

3 Conclusion

In this paper, we have reviewed some of the classification efforts, either based on boosting tree methods or on deep learning, for the automatic classification of glitches and other noise transients in advanced GW detectors. The input features of the pipelines can be extracted by the signal or through a wavelet decomposition or are images representing the time-frequency evolution of the glitches, that could be provided for instance by an external detection pipeline. We showed that we can reach accuracy in distinguishing chirp-like signals from generic noise transient signals of higher than 90%, as is found by many teams in the LIGO Virgo collaboration. This encourages the use of machine learning based techniques to set up automatic tools to classify noise sources.

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 et al., Physical Review Letters **116**, 061102 (2016), 1602.03837

[2] LIGO Scientific Collaboration, J. Aasi, B.P. Abbott, R. Abbott, T. Abbott, M.R. Abernathy, K. Ackley, C. Adams, T. Adams, P. Addesso et al., Classical and Quantum Gravity **32**, 074001 (2015), 1411.4547

[3] F. Acernese, M. Agathos, K. Agatsuma, D. Aisa, N. Allemandou, A. Allocca, J. Amarni, P. Astone, G. Balestri, G. Ballardin et al., Classical and Quantum Gravity **32**, 024001 (2015), 1408.3978

[4] G. Cella, A. Giazotto, Review of Scientific Instruments **82**, 101101 (2011), http://dx.doi.org/10.1063/1.3652857

[5] J. Schmidhuber, Neural Networks **61**, 85 (2015)

[6] D. George, E.A. Huerta, ArXiv e-prints (2017), 1701.00008

[7] J. Powell, D. Trifirò, E. Cuoco, I.S. Heng, M. Cavaglià, Classical and Quantum Gravity **32**, 215012 (2015), 1505.01299

[8] A.J. Calder, A. Burton, P. Miller, A.W. Young, S. Akamatsu, Vision Research **41**, 1179 (2001)

[9] R. Essick, S. Vitale, E. Katsavounidis, G. Vedovato, S. Klimenko, The Astrophysical Journal **800**, 81 (2015)

[10] F. Acernese, P. Amico, M. Al-Shourbagy, S. Aoudia, S. Avino, D. Babusci, G. Ballardin, R. Barillé, F. Barone, L. Barsotti et al., Classical and Quantum Gravity **22**, S1041 (2005)

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., Journal of Machine Learning Research **12**, 2825 (2011)

[12] N. Mukund, S. Abraham, S. Kandhasamy, S. Mitra, N.S. Philip, Physical Review D**95**, 104059 (2017), 1609.07259

[13] M. Razzano, E. Cuoco, Classical and Quantum Gravity **35**, 095016 (2018)

[14] Cuoco E., Razzano M. and Utina A., EUSIPCO 2018 proceeding (2018)

[15] M. Zevin, S. Coughlin, S. Bahaadini, E. Besler, N. Rohani, S. Allen, M. Cabero, K. Crowston, A.K. Katsaggelos, S.L. Larson et al., Classical and Quantum Gravity **34**, 064003 (2017), 1611.04596

[16] D. George, H. Shen, E.A. Huerta, ArXiv e-prints (2017), 1706.07446

[17] N. Christensen, LIGO Scientific Collaboration, Virgo Collaboration, Classical and Quantum Gravity **27**, 194010 (2010)

[18] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series* (Wiley, New York, 1949)

[19] B.P. Abbott, R. Abbott, T.D. Abbott, M.R. Abernathy, F. Acernese, K. Ackley, M. Adamo, C. Adams, T. Adams, P. Addesso et al., Classical and Quantum Gravity **33**, 134001 (2016), 1602.03844

[20] F. Acernese et al., Classical and Quantum Gravity **24**, S671 (2007)

[21] J. Powell, A. Torres-Forné, R. Lynch, D. Trifirò, E. Cuoco, M. Cavaglià, I.S. Heng, J.A. Font, Classical and Quantum Gravity **34**, 034002 (2017), 1609.06262

[22] C.G. Tianqi Chen, *XGBoost: A Scalable Tree Boosting System.*, in *KDD '16 Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), pp. 785–794

[23] I. Daubechies et al., *Ten lectures on wavelets*, Vol. 61 (SIAM, 1992)

[24] S. Mallat, *A wavelet tour of signal processing* (Academic Press, 1998)

[25] M. Unser, *Ten good reasons for using spline wavelets*, in *Proc. SPIE Vol. 3169, Wavelets Applications in Signal and Image Processing* (1997), p. 422–431

[26] E. Cuoco, G. Calamai, L. Fabbroni, G. Losurdo, M. Mazzoni, R. Stanga, F. Vetrano, Classical and Quantum Gravity **18**, 1727 (2001), gr-qc/0011041

[27] E. Cuoco, G. Losurdo, G. Calamai, L. Fabbroni, M. Mazzoni, R. Stanga, G. Guidi, F. Vetrano, Physical Review D**64**, 122002 (2001), gr-qc/0104071

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., Journal of Machine Learning Research **12**, 2825 (2011)

[29] J.F. T. Hastie, R. Tibshirani, *The Elements of Statistical Learning* (Springer, 2001)

[30] E. Cuoco, G. Calamai, L. Fabbroni, G. Losurdo, M. Mazzoni, R. Stanga, F. Vetrano, Classical and Quantum Gravity **18**, 1727 (2001)

[31] A. Krizhevsky, I. Sutskever, G.E. Hinton, pp. 1097–1105 (2012)

[32] Theano Development Team, arXiv e-prints (2016), /1605.02688

[33] F. Chollet et al., *Keras*, https://github.com/fchollet/keras (2015)

[34] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, E. Shelhamer, CoRR (2014), 1410.0759