Publications of the Astronomical Society of the Pacific, 134:024504 (16pp), 2022 February https://doi.org/10.1088/1538-3873/ac5159

Event Detection and Reconstruction Using Neural Networks in TES Devices: a Case Study for Athena/X-IFU

J. Vega-Ferrero[1,2], M. T. Ceballos[1], B. Cobo[1], F. J. Carrera[1], P. García[2], and J. Puyol-Gruart[2]
[1] IFCA, Instituto de Física de Cantabria (CSIC-UC), Av. de Los Castros s/n, E-39005 Santander, Spain; astrovega@gmail.com
[2] Artificial Intelligence Research Institute (IIIA), Campus UAB, E-08193 Bellaterra, Spain
Received 2021 October 11; accepted 2022 February 2; published 2022 February 25

Abstract

Transition Edge Sensors detector devices, like the core of the X-IFU instrument that will be on-board the Athena X-ray Observatory, produce current pulses as a response to the incident X-ray photons. The reconstruction of these pulses has been traditionally performed by means of a triggering algorithm based on the derivative signal overcoming a threshold (detection) followed by an optimal filtering (to retrieve the energy of each event). However, when the arrival of the photons is very close in time, the triggering algorithm is incapable of detecting all the individual pulses which are thus piled-up. In order to improve the efficiency of the detection and energy-retrieval process, we study here an alternative approach based on Machine Learning techniques to process the pulses. For this purpose, we construct and train a series of Neural Networks (NNs) not only for the detection but also for the recovering of the arrival time and the energy of simulated X-ray pulses. The data set used to train the NNs consists of simulations performed with the SIXTE/XIFUSIM software package, the Athena/X-IFU official simulator. The performance of our NN classification clearly surpasses the detection performance of the classical triggering approach for the full range of photon energy combinations, showing excellent metrics and very competitive computing efficiency. However, the precision obtained for the recovery of the energy of the photons cannot currently compete with the standard optimal filtering algorithm, despite its much better computing efficiency.

Unified Astronomy Thesaurus concepts: X-ray detectors (1815); Neural networks (1933); Telescopes (1689); X-ray astronomy (1810)

Online material: color figure

1. Introduction

The Athena X-ray Observatory (Nandra et al. 2013), selected in 2014 by ESA as the second Large-class mission in its Cosmic Vision science program, is designed to implement the Hot and Energetic universe science theme. Athena will be equipped with two X-ray detectors, one of which, the X-ray Integral Field Unit (X-IFU, Barret et al. 2018), is a cryogenic imaging spectrometer that offers spatially resolved high-spectral resolution X-ray spectroscopy over a 5′ equivalent diameter field of view. This microcalorimeter is based on superconducting Transition Edge Sensor (TES) technology.

In TES devices (see Irwin & Hilton 2005, and Ullom & Bennett (2015) for a review), the X-ray photons, which come from astronomical sources and are absorbed by the detector, produce rapid temperature increases that induce increments of the resistance in the superconductor element. As a response to this abrupt resistance increment, the device gives rise to electrical pulses whose area and height are related to the energy deposited by the photon. Due to telemetry constraints, the pulses need to be processed on-board to retrieve their energies, positions and arrival times.

The on-board processing is a sequential procedure that starts with the real time initial triggering on the full raw data coming from the X-IFU pixels. The purpose of this step is decreasing the amount of data passed to the second stage, the Event Processor, where the pulses are processed to retrieve the energy of the photons. The *record* of data selected in this way is aimed at containing isolated pulses.

The software package SIRENA[3] (see Ceballos et al. 2017, and references therein for a complete description) is integrated in the SIXTE (Dauser et al. 2019) package (containing the Athena official software for simulations), and it has been specifically developed as a test-bench to evaluate the performance of different algorithms that can be designed to detect the pulses, to reconstruct their energy content and to get their arrival time. As it can be run in software computers on the ground, it facilitates the testing of very different approaches and the comparison of their performance, both in terms of energy resolution and of

[3] https://sirena.readthedocs.io/

computational cost. The algorithm which best suits the requirements will then be selected to be on-board.

Several pulse processing techniques have been developed for TES detectors over almost three decades. Some of them are focused on the improvement of the energy and time resolution achieved while other techniques are aimed at identifying the piled-up pulses (nearly coincident events) for their later removal from the analysis. Among the former, the optimal filtering (Szymkowiak et al. 1993), the running sum algorithm (Tan et al. 2011), the interpolated covariant analysis (Peille et al. 2016 and references therein), the principal component analysis (Busch et al. 2016; Yan et al. 2016), the multi-pulse fitting (Fowler et al. 2015) or the fitting of the rising and falling part of the pulse shape (Ripoche & Heyl 2021). SIRENA implements several of these methods for testing but only those with a low computational cost will be suitable for on board implementation. In particular, the optimal filtering technique is the standard/baseline implementation as it shows a good compromise between resolution values and computational cost.

For the purpose of pulse detection to perform the initial triggering, SIRENA includes two algorithms. The *Threshold* method is the most basic, being based on the derivative of the signal exceeding a given threshold. On the other hand, the Adjusted derivative method (Boyce et al. 1999) requires the subtraction of a model pulse (from a calibration database) every time a pulse has been detected using the threshold; the detection is iteratively performed over the residual signal until no more pulses are found.

A performance test carried out using SIRENA over baseline X-IFU pixel simulations (Cobo et al. 2018) showed that, although this second method is more precise in general, it has two main caveats: the subtraction process always yields residuals, that may be miss-interpreted as additional pulses, and the method requires a great computational effort resulting in a non-viable option to be implemented on-board.

On the other hand the simpler *Threshold* approach is more affordable in terms of computational resources, but it fails when two (or more) photons arrive to the detector close in time (within 0.2 ms for the current pixels under study; LPA2.5a-style, see Miniussi et al. (2018) for a description of the pixel) because the derivative of the pulse signal does not have enough time to go below the detection threshold before the arrival of the following pulse.

The imperfect detection of these piled-up pulses (the pulse falling in the tail of a previous pulse is not detected) causes a wrong determination of their energy content in the event processor, leading in principle to a worse spectral resolution. The reason for this in the case for example of the optimal filtering, is that the piled-up pulses do not match the average shape of the optimal filter. To avoid as much as possible the inclusion of these pulses in the scientific analysis an additional step can be implemented: the reconstructed energy of the pulse will be compared on ground with a low resolution estimator (like the reconstruction performed using just a few samples of the pulse record) to unveil possible inconsistencies which will point to the presence of multiple pulses. This way these piled-up pulses can be ignored in the spectral analysis. With all this, the level of spectral spoiling of this effect using the *Threshold* method is, for the case of the X-IFU instrument, within the requirements.

The pile-up rejection is a goal shared by other processing techniques, like those based on the Singular Value Decomposition (Alpert et al. 2016) and Singular Vectors Projections (Borghesi et al. 2021), developed in the framework of the HOLMES experiment aimed at measuring the neutrino mass.

With the study presented here, we intend to explore an additional technique that could improve the initial step of pulse detection and pile-up rejection, not only for this type of detectors but also for those used under larger count rates than the X-IFU. In particular we have studied the contribution of Machine Learning (ML) techniques to the improvement of the pulse-finding and as a second step, the application of similar ML techniques in the reconstruction process to retrieve the energy of the pulses. We use the X-IFU instrument as a framework.

ML algorithms (in particular, Neural Networks, NNs) have been demonstrated to be an indispensable tool in many fields of research, not only in Astronomy and Astrophysics. The list of ML applications to the fields of Astronomy and Astrophysics is already vast: object detection and segmentation (Hausen & Robertson 2020), galaxy morphological classification (Huertas-Company et al. 2015; Domínguez Sánchez et al. 2018; Vega-Ferrero et al. 2021), galaxy mergers and tidal streams (Bottrell et al. 2019; Walmsley et al. 2019), galaxy–galaxy lensing identification (Metcalf et al. 2019), photometric redshift estimation (Pasquet et al. 2019; Campagne 2020) and Globular Clusters mass estimation (Ho et al. 2019; Su et al. 2020), just to mention a few examples and references. Moreover, it is likely to find applications of ML techniques to problem solving in other fields of research that are similar to what is our scope in this paper. For instance, X-ray pulse properties from free-electron laser (Sanchez-Gonzalez et al. 2017), neutron-gamma pulse shape discrimination (Griffiths et al. 2018), energy and position neutrino reconstruction in EXO-200 experiment (Delaquis et al. 2018), digital pulse shape analysis in solid state detectors (Mardiyanto et al. 2001; Flores et al. 2016) and cardiac pulse detection and classification (Elola et al. 2019).

In this paper and by means of realistic simulations, we explore the applications of ML algorithms to the detection and characterization of X-ray pulses in a TES detector, using the X-IFU instrument as a case study. In particular, we explore the potential of using Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs) with the aim of providing the number of pulses per record, their pseudo-energies (before final calibration) and their arrival times. We estimate also the computational cost of these techniques, as this

is the main driver for their possible application on-board X-ray satellites.

This paper is structured as follows: in Section 2, we describe the X-ray pulse simulations carried out with the SIXTE/XIFUSIM software package; in Section 3, we present the methodology applied to identify events detected in TES devices using ML algorithms; in Section 4, we show the ability of DNNs to estimate the differences in the arrival times of double pulses stored in the same record; in Section 5, we present the results on the energy reconstruction of isolated pulses using DNNs; finally, in Section 6, we summarize our results and main conclusions.

2. XIFUSIM Simulations of Athena/X-IFU events

Each pulse (event) generated by X-IFU corresponds to a photon of a given energy hitting the TES device. The shape of the pulse slightly varies with the photon energy due to the nonlinearity of the detector. Depending on the intrinsic brightness of the astronomical source (count rate), photons may arrive to the detector at close times.

As described in Section 1, neither the *Threshold* nor the Adjusted derivative methods are sensitive to pulses generated by X-ray photons with arrival times closer together than 0.2–0.3 ms for the X-IFU baseline pixels (i.e., pulses separated less than ∼30–45 samples given the sampling rate of 156.25 kHz assumed throughout the paper). To prove the limits of detection of our NNs (that will be described in the following section) we have created simulated data for the train and test data sets using the SIXTE/XIFUSIM simulator (Lorenz et al. 2020; Kirsch et al. 2021) for Athena/X-IFU, a software tool aimed at reproducing all the blocks of the entire instrument (from the output current of the TES sensor through the entire readout chain, including multiplexing, amplification and the digital readout) and that accurately reproduces the lab data of X-IFU TES devices. Using simulated data let us control the exact arrival time and energy of the photons and have enough statistics for the different combination of values, specially important for the training sets.

In particular, we simulated two types of records: single and double-pulse records with one and two pulses per record, respectively. Single-pulse records are characterized by the pulse energy (E_1, in keV). Double-pulse records are described by the first and second pulse energies (E_1 and E_2, in keV), and their separation (d_{12}, in sample units). We simulate pulses with energies in the range of [0.2, 12] keV according to the X-IFU requirements. Concerning the separation between pulses, we simulated them uniformly in the range [1, 100] in sample units. We do not explicitly present results for simulated triple-pulse records (containing three pulses per record) since they are equivalent to double-pulse records and very unlikely to occur in real observational data for the X-IFU science.

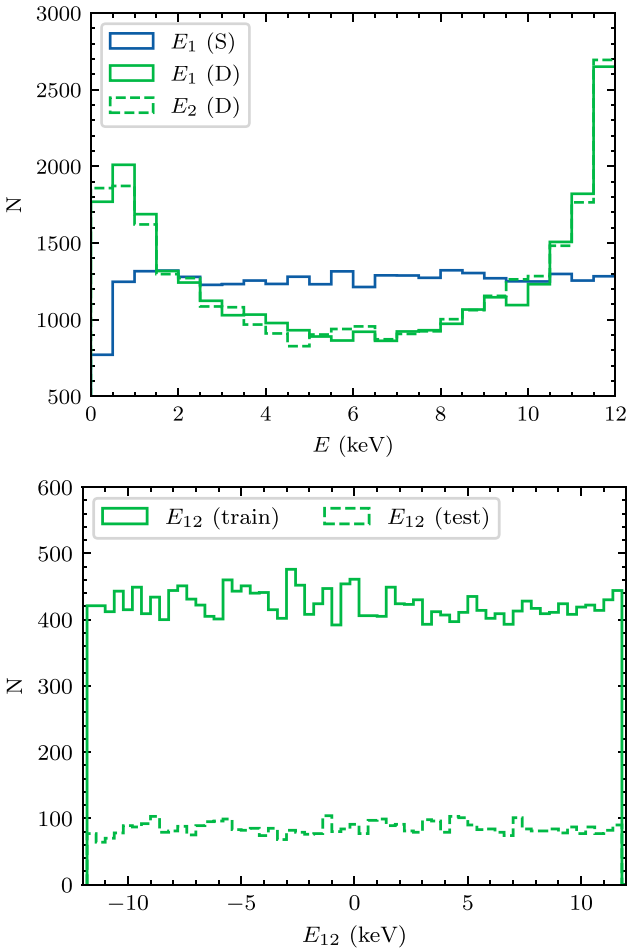

Figure 1. Top panel: distributions of the number of simulated pulses with a given energy (in keV). Blue histograms correspond to the energy distribution in single-pulse records, denoted as E_1 (S), while green histograms indicate the energy distributions of the double-pulse records for the first and the second pulses, denoted as E_1 (D) and E_2 (D), respectively. Bottom panel: distributions of the number of pulses with a given difference in energy between the first and the second pulse (E_{12} in keV, only for double-pulse records) for the training (solid histogram) and the test sample (dashed histogram).

The priority in the study will be the challenging cases for traditional methods (i.e., those with large and small differences of energies and similar arrival times), so we especially focus on generating those ("more complex") examples. Therefore, we simulate double-pulse records with energy differences between the first and the second pulse (i.e., $E_{12} = E_1 - E_2$) and separations (d_{12}) drawn from a continuous uniform distribution as follows: $E_{12} \in \mathrm{Unif}(-11.8,\ 11.8)$keV (given that E_1, $E_2 \in [0.2,\ 12]$) and $d_{12} \in \mathrm{Unif}(1,\ 100)$. For the single-pulse records, the energy of the single pulse (E_1) is obtained from a continuous uniform distribution within the defined energy range, i.e., $E_1 \in \mathrm{Unif}(0.2,\ 12)$. In total, we simulated 30,000 records of each type. In Figure 1, we show the distributions of

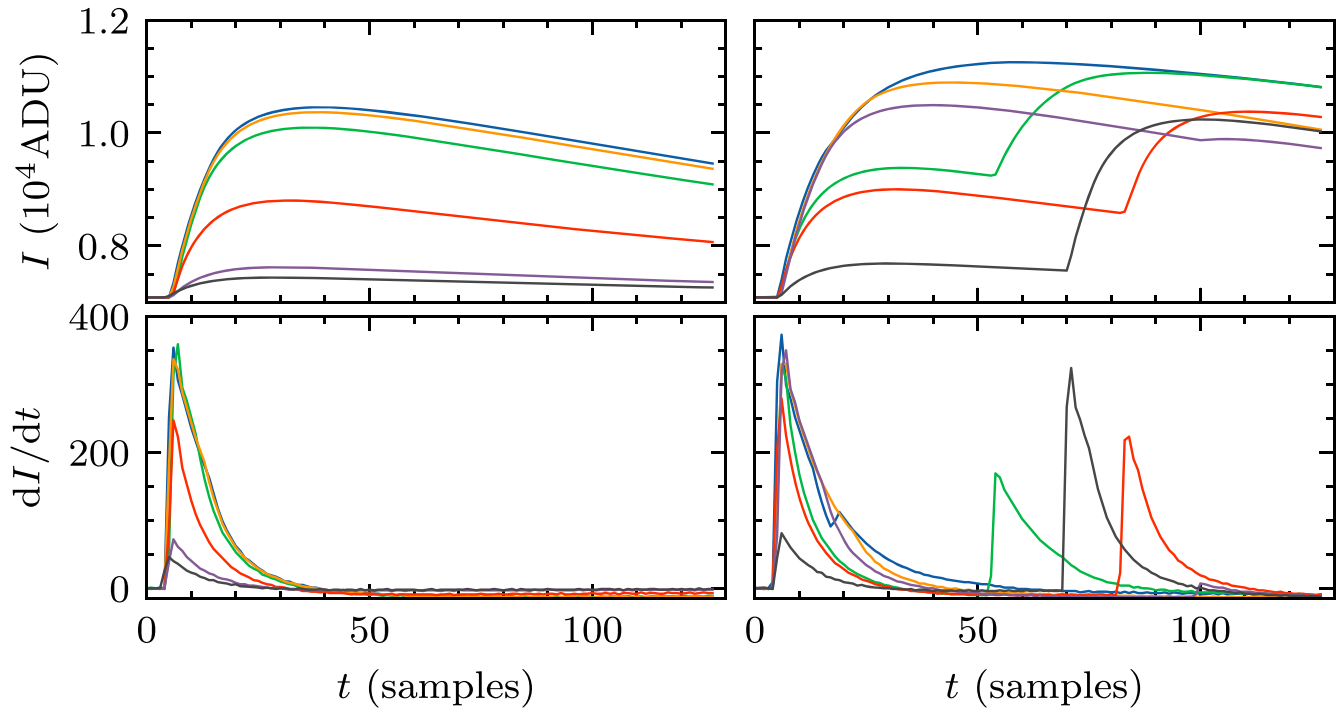

Figure 2. Top panel: current stream curves (in ADU, Analog-to-Digital Units) for six records of each type (single-pulse, left-hand panel; double-pulse, right-hand panel) as a function of the arrival time (in sample units). Bottom panel: first derivative of the intensity curves in the top panels. Each color line corresponds to a different record and thus, to different pulse energies and/or separations.

energies in our simulations of single-pulse and double-pulse records along with the distribution of the differences in energy between pulses (only for double-pulse records).

In the triggering phase, pulses arriving at separations larger than ∼100 samples (well above the pile-up limit of the above mentioned detection methods) can be detected individually so they would be stored in separated records. This is the main reason to generate two pulses in the same record with separations (d_{12}) below that particular value of approximately 100 samples. This criteria ensures that there is a sufficiently large number of interesting cases with small, similar and large differences of energy between consecutive pulses, and with close arrival times.

In Figure 2, we show the intensity (I) and its first derivative ($\mathrm{d}I/\mathrm{d}t$) as a function of the arrival time (t, in samples) for six examples of records of each type. The original records are simulated with record lengths of around 10,000 samples (large enough for the pulse to return to the baseline values) including a buffer of 1000 samples at the beginning of the records. For the analysis, however, we use record lengths of 128 samples (a section of the original record, starting with 7 samples before the pulse rise) which, given the range of distances between pulses, contain all the necessary information of the peaks of the two pulses in double-pulse records (the values of $\mathrm{d}I/\mathrm{d}t$ are already very close to zero for $t \gtrsim 120$ samples). Therefore, the arrival of the first pulse (in both single-pulse and double-pulse records) is preceded by approximately 6 samples in the first derivative of the intensity curve. In double-pulse records, the arrival of the secondary pulse with respect to the arrival of the primary pulse is determined by the parameter d_{12}. Since in the real life the pulse arrival time does no necessarily coincide with a sampling time, in the simulations it is randomized around ±0.5 samples. The effect of the randomization may be seen in the curves shown in Figure 2 and, more clearly, in Figure 5.

3. Identification of Events

In this section, we describe the methodology applied to identify events detected in TES devices using ML algorithms. The problem of identifying/classifying X-ray records according to the number of pulses per record may be faced by designing a NN in binary mode to distinguish between single-pulse and double-pulse records. In particular, we examine two different NN architectures: one is purely a DNN (consisting of a set of only dense layers), while the second one is a CNN (combining both convolutional and dense layers). The NN architectures are chosen based on a hyper-parameter optimization of the selected metric (e.g., the $F1$ score, see Equation (3)). Then, we compare their performance based on the metrics ($F1$ score, for instance) obtained for a test sample and the computational cost by means of the number of floating

point operations needed to process one record. Note that we do not refer to the number of floating point operations per second (FLOPS). Hereafter, the number of operations per records is computed using the *keras-flops* package in PYTHON.

3.1. Input Dataset

The input data used for the training and test phases of our NNs in binary mode is created from the pulse simulations described in Section 2. After a random shuffling we split the 60,000 records (half single-pulse records and half double-pulse records) into a training sample of 50,000 records and a test sample of 10,000 records (4979 are single-pulse records and 5021 are double-pulse records). As explained above, we work with records with a length of 128 samples, starting with typically 6($\pm$1) samples before the arrival of the first pulse (as shown in Figure 2).

We choose to provide our NNs with the first derivative of the pulse intensity as input. This choice is based on the accuracy obtained after training several NNs with both the intensity and the first derivative curves as input data. Given that it is a recommended practise in ML, we also normalized the input data. Once the first derivative of the pulse intensity is computed from the records of 128 samples, we normalize individually the first derivative curve by its maximum value. Although negligible in comparison with the number of operations due to the NN architecture (as we describe in the following sections), the operations made out of the computation of the derivative of the pulse intensities are performed by hardware during the *Threshold* phase. Therefore, we do not include the computation cost of the derivative of the pulse intensities in the total number of operations. After the normalization, the dynamical range of the input data is ($\lesssim$0, 1), with a preference toward positives values since the slope of the pulse intensity is higher before the peak (i.e., when the pulse is ascending) than after it. Since we are not inferring here the pulse energy (Section 5), for which the integrated area below the intensity curve of a pulse is fundamental, normalizing the records individually does not have an impact on the classification task, but it helps keeping a smaller dynamical range compared to the case without normalization or with a global normalization (i.e., normalizing by the maximum value of all the records). Given that the first derivative may take negative values, we also checked a normalization with non-negative values but with slightly less accurate results compared to the individual normalization described above. Summarizing, we provide the NNs with input vectors of length 128 samples (therefore, 1D NNs) containing the information of the first derivative of the pulse intensity normalized by its maximum value.

3.2. Binary Classification

For the binary classification, we label as Positives (P) the single-pulse records, while the Negatives (N) labels are assigned to double-pulse records.

The architectures of the NNs used in this work for the binary classification task are of two types:

1. a pure DNN consisting of $n = (2, 3, 4)$ dense layers of variable sizes (M_i, where $i \in [1, n]$ denotes the layer number) and linear activation functions (the so-called, *relu*) followed by a dense layer of size 1 with a sigmoid activation function.
2. a CNN composed of a combination of $n = (2, 3, 4)$ convolutional blocks followed by a global max pooling layer and a dense layer of size 1 with a sigmoid activation function. Each convolutional block consists of: a convolutional filter with M_i channels of sizes L_i and a *relu* activation function; a max pooling layer with variable pool size (K_i); and a dropout layer with a variable frequency rate (d). To avoid reducing excessively the dimensions of the output array after the set of convolutions, we force the pool size of the deepest convolutional block to be equal to one, i.e., $K_n = 1$ (no max pooling is applied). The *padding* in the convolutional filters is set to "same". Note that the frequency rate of the dropout layer (d) is the same for all the convolutional blocks.

Another key hyper-parameter to set in order to train a neural network is the learning rate for gradient descent (an optimization technique commonly used in training ML algorithms) This parameter scales the magnitude of our weight updates in order to minimize the network's loss function. Hereafter, we use an adaptive optimization algorithm (*adam*) with a variable learning rate (denoted as l_r).

The chosen design for the CNN as a set of convolutional blocks followed by a global max pooling layer is justified after a comparison with other architectures. For instance, we also checked the performance of a CNN consisting of convolutional blocks followed by a flatten layer, but we found that it had a lower performance. We did not try a combination of CNN followed by a recurrent layer because, for analogous results, this design leads to a larger number of parameters (and therefore of the number of operations) compared to a pure CNN, as demonstrated by Elola et al. (2019) on a similar pulse classification task.

As we describe in the next section, the set of hyper-parameters is found by performing an hyper-parameter optimization on a validation set in order to maximize the algorithm performance.

Given the configuration in binary mode of the previously described NNs, the output value represents the likelihood, p (in the range [0, 1]), of being a Positive (P) case (i.e., a single-

pulse record). Consequently, the value $1-p$ represents the likelihood of being a Negative (N) case (i.e., a double-pulse record). A way to check the accuracy of NN models is based on the area under the ROC curve (ROC AUC) for the different predicted probabilities (see Powers 2011, for instance). The ROC curve is a representation of the false positive rate (FPR = FP/N, i.e., the ratio of the number of false positives to negative cases) versus the true positive rate (TPR = TP/P, i.e., the ratio of the number of true positives to positive cases) for different probability thresholds. Note that by definition the sum of TN and FP correspond to the number of negative cases (N = TN + FP), while the sum of TP and FN yields to the number of positive cases (P = TP + FN). An optimal classifier is the one that maximizes the ROC AUC and, therefore, for that classifier it is possible to derive the probability threshold ($p_{\rm thr}$) that optimizes the ROC curve (i.e., maximizes TPR and minimizes FPR). Nevertheless, depending on the user purpose, one can vary the $p_{\rm thr}$ to obtain a more complete or less contaminated sample. According to this definition, records with predicted values $p < p_{\rm thr}$ are classified as Negatives, while records with predicted values $p \geqslant p_{\rm thr}$ are classified as Positives.

A complementary way to test the model performance is the precision and recall scores, which can be defined as follows:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

The precision is intuitively the ability of the classifier to not label as positive a sample that is negative (or a purity/contamination indicator), while the recall represents the ability of the classifier to find all the positive samples (i.e., a completeness indicator). The best value for both the precision and recall is 1 and the worst value is 0. Additionally, the $F1$ score, expressed as follows,

$$F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \tag{3}$$

can be interpreted as a weighted average of the precision and recall, where the relative contribution of precision and recall to the $F1$ score are equal. The $F1$ score reaches its best value at 1 and the worst value at 0.

3.2.1. Hyper-parameter Optimization

The aim of hyper-parameter optimization in ML is to find the combination of hyper-parameters of a given ML algorithm that returns the best performance as measured on a validation set. In this section, we describe how we optimize the hyper-parameters of the DNN and CNN architectures proposed in the previous section.

The hyper-parameters of every model were optimized using a Bayesian optimization approach (Snoek et al. 2012). Briefly, the Bayesian optimization method builds a probability model of the objective function (or surrogate function) and use it to select the most promising hyper-parameters to evaluate in the true objective function. Bayesian approaches, in contrast to random or grid search, keep track of past evaluation results which they use to form a probabilistic model mapping the hyper-parameters to a probability of a score on the objective function. Recent studies report that Bayesian optimization is more efficient than manual tuning, grid or random search since it requires less computation time and the overall performance on a valuation set is higher (Bergstra et al. 2012). Variants of the Bayesian optimization differ on how the surrogate function is computed. In this study, we consider the Tree-structured Parzen Estimators (TPE). By applying Bayes rule, the TPE approach is able to find better hyper-parameters (i.e., better performance) than random search in the same number of trials.

We choose the $F1$ score (Equation (3)) as the objective function for the hyper-parameter optimization. We run the optimization process by maximizing the $F1$ score for a total of 50 trials and 50 epochs per trial and store the $F1$ score and the list of hyper-parameters for each trial. We implement the optimization pipeline for the proposed DNNs and the CNNs independently for each architecture in terms of the number of layers (n). In other words, we obtain a different set of optimized hyper-parameters for the DNNs and the CNNs with 2, 3 and 4 layers separately (followed in all cases by the output dense layer).

The hyper-parameters subject to the optimization in the case of the DNN are the learning rate for the *adam* optimizer, l_r, and the sizes of the filters in each layer, L_i. For the CNN, the hyper-parameters to be optimized are:

1. the learning rate for the *adam* optimizer with allowed values of $l_r = (10^{-1}, 10^{-2}, 10^{-3}, 10^{-4})$;
2. the sizes and the number of channels of the convolutional filters in each convolutional block may vary as $L_i = (4, 8, 16, 32, 64, 128)$ and $M_i = (3, 5, 7, 9, 11, 13, 15)$, respectively;
3. the size of the max pooling layer in each convolutional block may take values of $K_i = (1, 2, 3)$, where $K = 1$ means that no max pooling is applied;
4. the frequency rate of the dropouts (equal for all the convolutional blocks) may vary as $d = (0.1, 0.2, 0.3, 0.4, 0.5)$;
5. and the batch size is allowed to take the values $b = (150, 200, 250, 300, 350, 400, 450, 500, 550, 600)$.

Concerning the max pooling layers and given the dimensions of the CNN: for the configuration with $n = 2$, only the pool size K_1 may vary while $K_2 = 1$; for the configuration with $n = 3$, the pool sizes K_1 and K_2 may vary while $K_3 = 1$; and for the

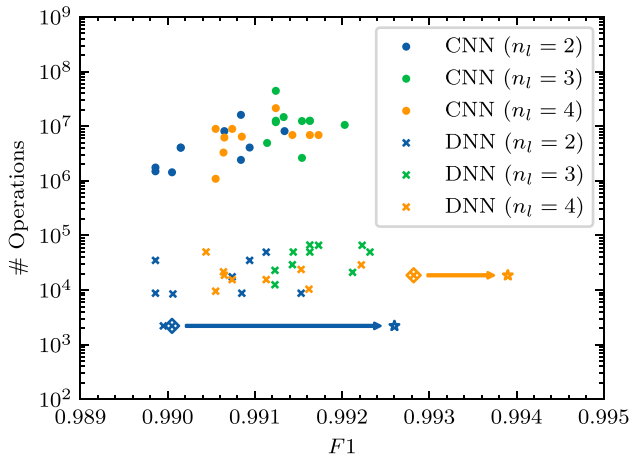

Figure 3. Hyper-parameter optimization of the CNNs (filled dots) and the DNNs (crosses) for the binary classification. The number of operations is shown as a function of the $F1$ score for the 10 trials with the best $F1$ score for each NN configuration (2 layers in blue, 3 layers in green and 4 layers in orange) after 50 epochs of training. The two crosses embedded in empty diamonds highlight the selected configurations according to the number of operations (for $n = 2$ in blue) and to the $F1$ value (for $n = 4$ in orange). The stars (in blue for the DNN with $n = 2$ and in orange for the DNN with $n = 4$) correspond to the results for the best model (among the $k = 10$ realizations) of the highlighted configurations after 300 epochs of training. The selected configurations along with the metrics are described in Tables 1 and 2.

configuration with $n = 4$, the pool sizes K_1, K_2 and K_3 may vary but $K_4 = 1$.

In Figure 3, we show the results of the hyper-parameter optimization of the CNNs and the DNNs configurations described previously. The optimization is obtained on the test sample of 10,000 records after training the NNs with a sample of 50,000 records. We only show the 10 trials (from the 50 trials generated) with the highest $F1$ scores for each configuration (i.e., 2, 3 and 4 layers). Both the CNNs and the DNNs achieve excellent results, with values $F1 \gtrsim 0.99$. For all the CNN configurations, the performances based on the $F1$ score are comparable with those obtained with the pure DNNs, however the number of operations is between 2–3 orders of magnitude larger. The CNN with the best performance, which achieves a $F1 = 0.9920$, with a number of operations of 1.1×10^7, is the one with $n = 3$. For the DNN architectures, on one hand, the maximum $F1$ score for the DNNs is reached by the configuration with $n = 4$ (labeled as bin-best) and a number of operations of $\approx 1.9 \times 10^4$ and $F1 = 0.9928$. On the other hand, the DNN with $n = 2$ (labeled as bin-ops) has a performance of $F1 \approx 0.9901$ with a considerably smaller number of operations of $\approx 2.2 \times 10^3$.

For this reason, hereafter we will focus our efforts on the use of DNNs for the classification task presented in this section and for the characterization tasks (i.e., arrival time and energy estimation) described in the following sections. The hyper-parameters that described the selected DNNs architecture are shown in Table 1.

Table 1
Summary of the Hyper-parameter Optimization of the DNNs with the Best Performance

Model	Metric	# Ops.	n	l_r	b	L_i
bin-best	0.9928	1.9×10^4	4	0.001	250	(32, 32, 64, 32)
bin-ops	0.9901	2.2×10^3	2	0.001	150	(8, 8)
time-best	0.62	7.0×10^4	4	0.001	300	(64, 128, 128, 16)
time-ops	0.72	1.5×10^4	4	0.001	150	(32, 64, 16, 4)
enrg-best	3.3	3.3×10^4	3	0.0001	150	(32, 128, 64)
enrg-ops	4.6	2.6×10^3	2	0.001	200	(8, 32)

Note. Column 1 indicates the model name. Column 2 shows the metric used for the optimization: $F1$ score (Equation (3)) for the binary classification (first two rows), MAE (Equation (4)) in sample units for the arrival time estimation (third and fourth rows) and MAE in eV for the energy estimation (last two rows) of the architecture with the best performance. Column 3 indicates the number of operations required to process one record. Columns 4, 5 and 6 correspond to the number of layers (n), the learning rate (l_r) of the *adam* optimizer and the batch size (b), respectively. Column 7 indicates the size (L_i) of the dense layers. Note that the n dense layers in the DNNs are followed by an additional dense layer of size 1 (or the output layer).

3.2.2. Results

From the hyper-parameter optimization process described in the previous section, we choose the two DNNs with the best performance in terms of the $F1$ score (denoted as bin-best with $n = 4$) and according to number of operations (labeled as bin-ops with $n = 2$) as shown in Table 1. Then, we apply a k-fold cross-validation where the training sample of 50,000 pulses is split into k sets. One by one, a set is selected as the validation set and the $k - 1$ remaining sets are combined into the corresponding training set. Since this is repeated for each of the k sets, we finally obtain a total of k model realizations with an independent random initialization and trained with a slightly different training set. We fix $k = 10$ and, therefore, we derive 10 model predictions trained with 45,000 pulses, validated with 5000 pulses and tested on a test set of 10,000 pulses. We denote the single-pulse records as S records and the double-pulse records as D records. We increase the number of epochs with respect to the optimization phase to 300 epochs. We store the model weights at the epoch with the lowest loss (not necessarily the model of the last epoch) for each k model and apply it to the test sample.

The performance among the different realizations for the selected DNNs is quite consistent. For the $k = 10$ realizations of the bin-best model, we obtain a median $\tilde{F}1 = 0.9935$ and a $F1 = 0.9939$ for the best model. For the $k = 10$ realizations of the bin-ops model, we derive a median $\tilde{F}1 = 0.9904$ and a $F1 = 0.9926$ for the best model. We summarize these metrics in Table 2. Hereafter, we show the results for the best model

Table 2
Summary of the Metrics for the $k = 10$ Realizations of the DNN Models described in Table 1

Model	Median	Min	Max
bin-best	0.9935	0.9924	0.9939
bin-ops	0.9904	0.9892	0.9926
time-best	0.55	0.53	0.57
time-ops	0.62	0.59	0.65
enrg-best	3.0	2.8	3.9
enrg-ops	4.5	3.3	4.7

Note. Column 1 indicates the model name. Column 2 shows the median of the metric for the $k = 10$ realizations: $F1$ score (Equation (3)) for the binary classifications (first two rows), the MAE (Equation (4)) in sample units for the arrival time estimation (third and fourth rows) and MAE in eV for the energy estimation (last two rows). Column 3 and 4 indicate the minimum and the maximum value of the corresponding metric among the $k = 10$ realizations, respectively. Each model is trained up to 300 epochs.

among the $k = 10$ realizations of the bin-best and the bin-ops models.

In Figure 4, we show the confusion matrices for the bin-best and the bin-ops models. As expected from the corresponding $F1$ scores, the bin-ops classifier has a slightly better performance than the bin-ops one. The number of FN records (i.e., S records classified as D records) is almost negligible in both cases (2 FN of total of 4979 S records). Interestingly, the FN records for the bin-ops model are both S records with very energetic pulses (11.90 keV and 11.99 keV), while for the bin-ops model the 2 FN are S records with low energy pulses (0.26 keV and 0.41 keV). In Figure 5, we show the normalized first derivative intensity curves (i.e., the input vector of the DNNs) of the four FN records found in the bin-best and the bin-ops models. Note that the first derivative intensity curves are normalized individually by their maximum value. From this figure, it is clearly visible a "knee" after the maximum of the first derivative (at around $t = 20$ samples) but before the maximum of the pulse signal (i.e., the first derivative still has a positive sign, so the signal is still increasing power) for the FN records in the bin-best model. We interpret that this variation in the first derivative is confusing the DNNs making these S records look like D records. For the FN records detected by the bin-ops model, given the low energy of these two pulses, the noise level (baseline signal) is comparable with the signal right after the peak of the first derivative intensity curve. We conclude that the baseline signal is interpreted by the DNNs as a secondary peak in the first derivative and, therefore, the corresponding records are not classified as a S records, but wrongly as D ones.

The number of FP records (i.e., D records classified as S records) shown in Figure 4 is 59 for the bin-best model and 72 for the bin-ops model. These values corresponds to $\sim$1% of the 5021 D records included in the test sample. In particular, these FP records are mainly D records with large differences in energy and small separations (i.e., arrival times) between the first and the second pulses in each record.

In Figure 6, we show the fraction of FP records in the E_{12}–d_{12} plane for the bin-best and the bin-ops models. The fraction of FP records is computed as the number of D records classified as S records divided by the number of D records, i.e., $\mathrm{FP}/(\mathrm{TN} + \mathrm{FP})$. Note that the number of D records is roughly constant per hexagonal bin since we simulate both the E_{12} and d_{12} from a uniform distribution. The majority of the FP records are located in the top-left corner of the E_{12} - d_{12} plane with $d_{12} \lesssim 15$ samples and $E_{12} \gtrsim 3$ keV. However, even for the most extreme pulses, those with $d_{12} \lesssim 5$ samples and $E_{12} \gtrsim 7$ keV, the fraction FP records is below 50%. Apart from the top-left corner, the fraction of FP records in the rest of the E_{12} - d_{12} plane is negligible. Another implication that may be extracted from Figure 6 is that the DNNs here examined have a better performance when applied to D records with $E_{12} < 0$ keV (i.e., the second pulse is more energetic than the first one). Therefore, these DNNs are better at distinguishing the less energetic pulse during the rise of the most energetic one that the other way around. This is probably due to the shape of the pulses as detected by X-IFU (i.e., the sharp rise of the pulse is much more pronounced than the fall after it peaks). Overall, this result provides a significant improvement with respect to the classical method that fail to distinguish records with multiple pulses with arrival times closer together than $\sim$30–45 samples, independently of the energies of the pulses (Cobo et al. 2018).

4. Estimation of the Arrival Time of Double Events

The estimation of the arrival time of a second pulse in the record (and not only the identification of the record as "double-pulse") would permit the recovery of the individual pulses for the analysis, since they could be reconstructed afterwards with the information of the uncontaminated pulse length.

In this section, we explore the ability of NNs to estimate the differences in the arrival times of X-ray pulses stored in the same record. First, we start by training and testing a series NNs (analogous to the ones described in Section 3.2) to estimate the separation of pulses in double-pulse records. After some preliminary checks we select the DNNs as a better approach, mainly in terms of the number of operations (but also the accuracy) to the estimation of the event separation, similarly to the findings of Sanchez-Gonzalez et al. (2017). Therefore, we focus on a pure DNN to retrieve the differences in the arrival time estimation (d_{12}) of two consecutive X-ray pulses stored in the same record (i.e., D records).

As described in Section 2, the range of the separations between the first and the second pulses in D records are uniformly distributed as $d_{12} \in \mathrm{Unif}(1, 100)$. Consequently, our DNNs need to be configured in regression mode in order for

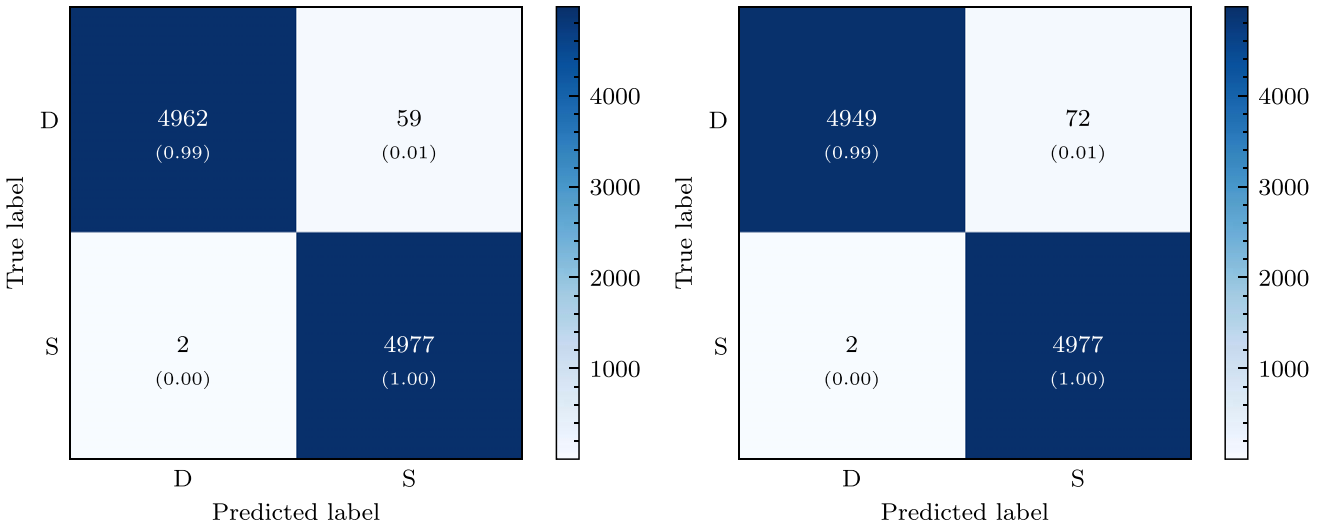

Figure 4. Confusion matrices for the binary classification obtained with the DNN with the best $F1$ score (bin-best, left-hand panel) and the DNN with the lowest number of operations (bin-ops, right-hand panel) in the hyper-parameter optimization. The number of pulses in the test sample shown here is equal to 10,000 consisting on 4979 single-pulse (S) records and 5021 double-pulse (D) records. In each box, we show both the number of pulses and the fraction in the corresponding "true" category.

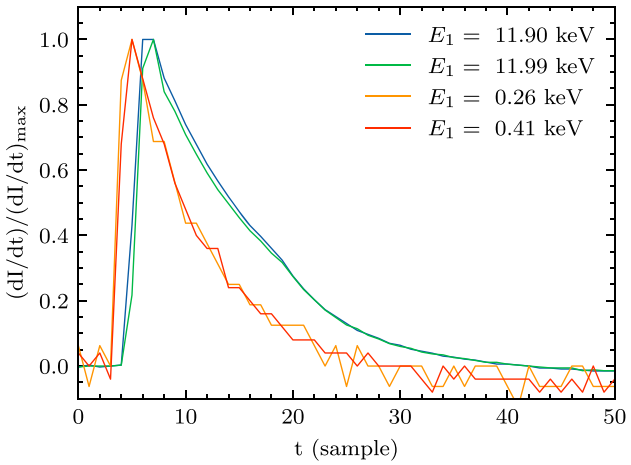

Figure 5. Normalized first derivative of the current stream curves of the FN records for the bin-best and bin-ops models as a function of the arrival time (in sample units).

them to derive a continuous value within the range of d_{12}. The architecture of the DNNs consists of 2, 3 or 4 dense layers of variable sizes (M_i) and a *relu* activation function followed by a fully connected layer of one neuron with a linear activation function.

As in the previous classification tasks, the input data to the DNNs are vectors of length 128 containing the first derivative of the D pulse intensities normalized by its maximum. Therefore, the input signals are within the range [$\lesssim$0, 1]. Although a *global* normalization (i.e., normalizing by the maximum value of all the pulses) might be used, it is not crucial since the pulse separation is not sensitive to the shape of the pulse, just to the beginning of it. In any case, we tried a *global* normalization, but we found it to have a slightly worse performance compared with the *individual* normalization.

The input values for the D records are the separation between the consecutive pulses, d_{12} in sample units. These quantities are previously normalized by the length of the input vector, i.e., $\bar{d}_{12} = d_{12}/128$. Once the predictions of the model are derived we undo the normalization to retrieve the separation between pulses and the corresponding metrics.

4.1. Hyper-parameter Optimization

In this section, we present the results of the hyper-parameter optimization of the DNN architectures designed to estimate the separation of pulses in D records. We choose the mean absolute error (MAE) as our objective function,

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\tilde{y}_i - y_i|, \quad (4)$$

which is a typical objective function for a NN in regression mode. By y and $\tilde{y}$ we refer to the input separation (i.e., d_{12}) and the predicted separation (denoted as d_{p}) between consecutive pulses, respectively. The lower the MAE, the better the performance of the NN architecture. Then, we run the optimization pipeline for a total of 50 trials and 100 epochs per trial, and store the MAE and the list of hyper-parameter for each trial. The hyper-parameter space is the same as the ones

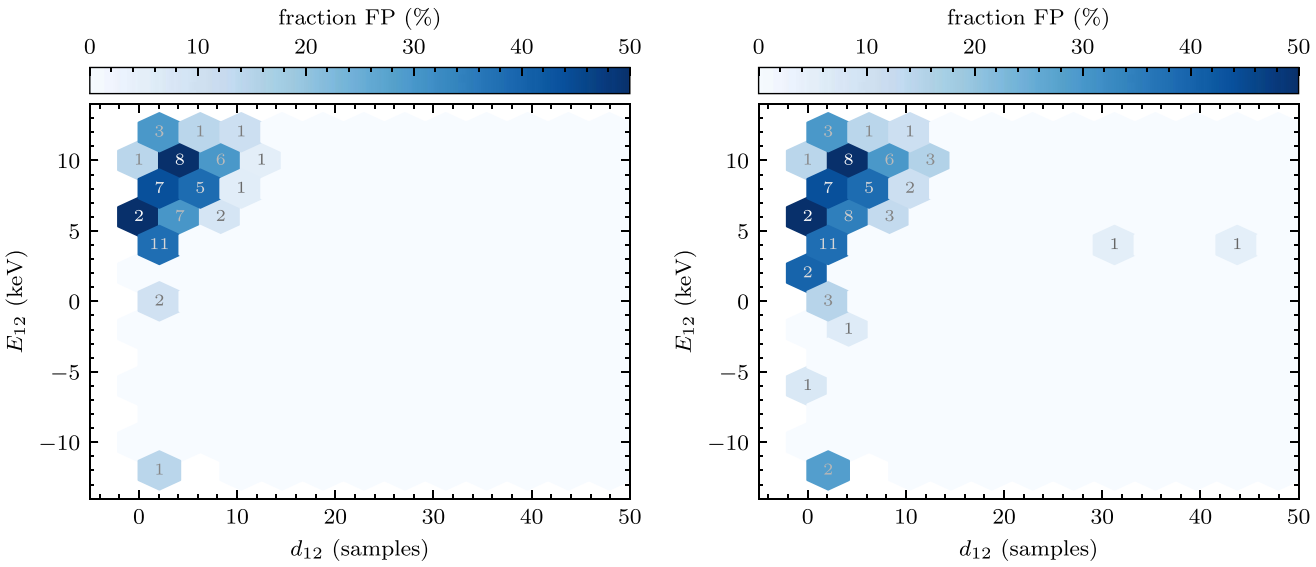

Figure 6. Fraction of false positive (FP) records in the E_{12} - d_{12} plane for the bin-best (left-hand panel) and the bin-ops (right-hand panel) models. The color of the hexagonal bins corresponds to the fraction (in %) of FP records (computed as the number of D records classified as S records divided by the number of D records) as coded in the color bar. The number in each of the hexagonal bins indicates the number of FP records in that particular bin.

defined for the DNNs in the binary classification optimization (see Section 3.2.1). In Figure 7, we show the results of the hyper-parameter optimization of the DNNs with $n = (2, 3, 4)$ for the separation of pulses in D records. We show the number of operations versus the MAE (in sample units) for the 10 trials with the smallest MAE for each DNN configuration. On one hand, the DNN with the best performance is the DNN with $n = 4$ (labeled as time-best), MAE = 0.62 samples and a number of operations equal to 7.0×10^4. On the other hand, we also select the DNN with $n = 4$ (labeled as time-ops), the lowest number of operations (equal to 1.5×10^4) and a reasonably good MAE value of 0.72 (although the DNN with $n = 3$ has a slightly lower number of operations, its MAE is almost 0.1 larger). Both selected architectures are described in Table 1.

Note that the input separation d_{12} is given by an integer number within the range [0, 100] samples, while the (output) predicted separation (d_p) is a real number in the same range. Therefore, values of MAE below 1 sample denote that the DNNs are able to accurately determine (with less than 1 sample error) the arrival time of a secondary pulse in D records with respect to the primary pulse in the same record. However, this does not mean that the predictions made by the DNNs are below the measurement error (which is beyond the scope of this section), but that the DNNs are able to predict the position of a secondary pulse in a D record with an error below ± 1 sample.

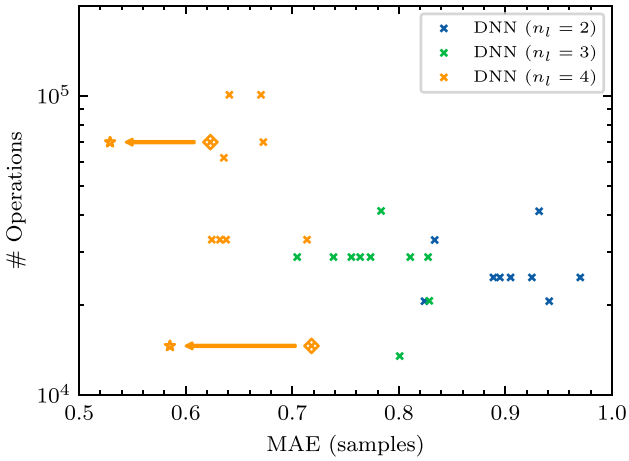

Figure 7. Hyper-parameter optimization of the the DNNs (crosses) for the arrival time estimation of D records. The number of operations is shown as a function of the MAE (Equation (4)) for the 10 trials with the lowest MAE for each DNN configuration (2 layers in blue, 3 layers in green and 4 layers in orange) after 100 epochs of training. The two crosses embedded in empty diamonds highlight the selected configurations according to the number of operations and to the MAE values. The stars correspond to the results for the best model (among the $k = 10$ realizations) of the highlighted configurations after 300 epochs of training. The selected configurations along with the metrics are described in Tables 1 and 2.

4.1.1. Results

We present in this section the results of the estimation of the arrival time of D records for the DNNs with the best

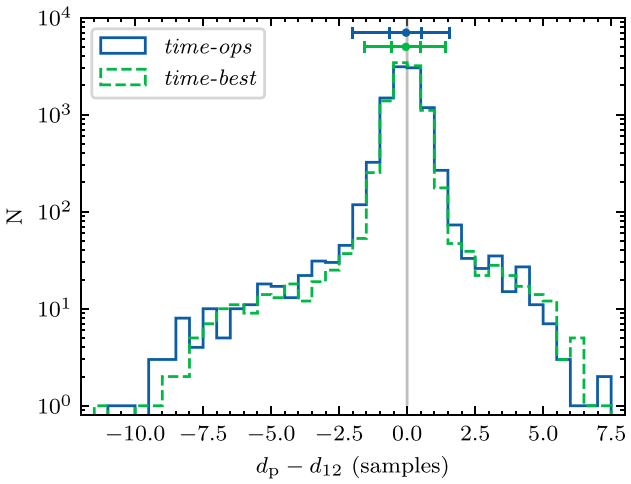

Figure 8. Distribution of the differences between the predicted (d_p) and the true separation (d_{12}) of pulses in D records in the test sample (10,000 records) for the time-best model (dashed green histogram) and the time-ops model (solid blue histogram). For each model color dots indicate the median value of the distribution, while the error bars enclose 68% (1σ) and 95% (2σ) of the test sample. Note the logarithmic scale in the vertical axis. I suggest instead to use a stepped empty histogram for one of them, I think that it would be clearer and friendlier to see in b/w.

performance in terms of the MAE (time-best model) and the number of operations (time-ops model) among the architectures described previously. The optimized DNNs for the estimation of the separation of D records are trained and validated with a sample of 30,000 D records (described in Section 2). The sample is split into training sample (20,000 records) and test sample (10,000 records). As for the precedent classification task, we perform a k-fold cross-validation with $k = 10$. Therefore, we obtain 10 different realizations with independent weights initialization, trained and validated with slightly different combination of pulses. Then, the models are tested over the test sample, which has been never shown to the DNN during the training process. The number of epochs is fixed to 300 epochs. We store the model weights at the epoch with the lowest MAE for each realization and apply it to the test sample.

For the separation of D records we obtain a median MAE = 0.55 samples for the $k = 10$ realizations of the time-best model, while the best realization has a MAE = 0.53 samples. For the time-ops model, the median MAE = 0.62 samples for the $k = 10$ realizations and the best realization has a MAE = 0.59 samples. These metrics are summarized in Table 2. Hereafter, we show the results for the best model among the $k = 10$ realizations of the time-best and the time-ops models.

The distribution of the differences between the predicted and the true separation ($d_p - d_{12}$) of pulses in D records is shown in Figure 8. The median values are 0.0 ± 0.6 and 0.0 ± 0.5 (errors correspond to 68% of the test sample) for the time-best and the time-ops models, respectively.

To better illustrate the results obtained on the determination of the separation of pulses in D records, we show in Figure 9 two boxplots for the ($d_p - d_{12}$) derived by the time-best and the time-ops models as a function of the E_{12} and d_{12}. The median values are very close to zero in both cases with small deviations for high values of E_{12} or low values of d_{12}. However, the D records with high values of E_{12} or low values of d_{12} are also those with larger interquartile range (IQR) and number of outliers. The IQR is defined as the width between the third and first quantiles (i.e., IQR = Q3–Q1), where the Q1 and Q3 correspond to the 25th and 75th percentiles, respectively. It is clear also from the errors bars that, for the extreme cases where $d_{12} \lesssim 20$, the separation of the pulses, d_{12}, has a stronger impact in its determination than the difference in energy of the pulses, E_{12}. In fact, most of the outliers with $|d_p - d_{12}| > 3$ samples are due to records with $d_{12} \lesssim 20$.

Once a record is classified by the DNNs described in Section 3.2 as a D record, in this section, we demonstrate that the DNNs are able to determine the arrival time of the two consecutive pulses with enough precision to help the reconstruction method afterwards. The individual pulses can be then recoverable for the analysis, since the estimation of its energy is possible using the uncontaminated pulse length. In particular, for D records that are separated less than $d_{12} \sim 30$–45 samples, the DNN constraints on d_{12} will improve significantly the characterization and energy determination of pulses in D records where the classical triggering fails. The reconstruction would be then performed over the recovered individual pulses not needing to throw them away by a labeling method and avoiding a possible unnoticed reconstruction of piled-up pulses (worst case scenario).

5. Estimation of the Energy of Single Events

As a final step on the reconstruction of the X-ray pulses detected by an instrument like Athena/X-IFU, in this section, we describe the methodology used for inferring their energies using DNNs. We focus our efforts on retrieving the energies of pulses in S records, studying whether they are compatible with the current estimates by the standard method which uses the well-tested optimal filtering technique (Szymkowiak et al. 1993). Applied over the pulses simulated by XIFUSIM with the characteristics of the X-IFU baseline pixel (∼2.5 eV FWHM energy resolution at 6keV) the optimal filtering technique is able to recover ∼eV input resolution for the reference energy, while keeping a modest number of operations. For instance, the optimal filtering technique requires roughly 49,000 operations (8192 products + 8192 summations in 3 positions) to reconstruct the energy of just one record. Then, our objective is to test whether the NN methodology reaches this eV precision (not bursting the computational cost), or perhaps is

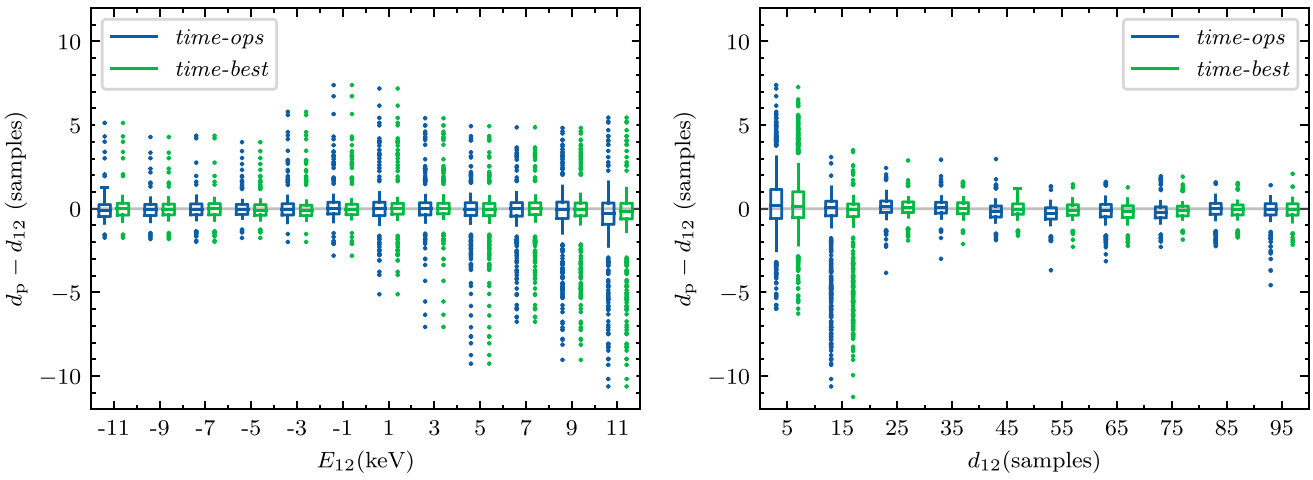

Figure 9. Boxplots for the $(d_p - d_{12})$ as a function of the E_{12} (left-hand panel) and d_{12} (right-hand panel). The results are shown for the time-ops (in blue) and the time-best models (in green) separately. The test sample of 10,000 D records is split in bins of E_{12} of 2 keV within the range $[-12, 12]$ keV, and in bins of d_{12} of 10 samples within the range [0, 100] samples. Therefore, the boxes are centered in the middle value of each corresponding bin. The box encloses the IQR. The line within each box indicates the median of the distribution (50th percentile). The minimum and maximum values in the data set (represented with the error bars) excluding outliers correspond to Q1 $-$ 1.5 $\times$ IQR and Q3 + 1.5 $\times$ IQR, respectively. A value outside that range is considered an outlier (dots) with $\gtrsim 2.7\sigma$.

even able to improve the resolution at higher energies, where the assumption of signal linearity with energy of the optimal filtering technique produces worse resolution values.

As done in previous sections, since we try to minimize the number of operations, we train and test only a series of DNNs (not CNNs). The input data are vectors of length 128 containing the first derivative of the S pulse intensities. Given that the energy of the pulse is proportional to the area under the pulse intensity curve (also the shape, see Figure 2), in this case, we normalize the input vectors to the maximum value of all the S records. In other words, the maximum value of the input data is 1 for the record with the largest value of the first derivative of the pulse intensity and below 1 for the rest of the S records. In this way, we maintain the information contained on the shape of the pulse intensity curve about the energy of the pulse. The input values of the S records are the pulse energies within $E_1 \in [0.2, 12.0]$ keV. Therefore, the DNNs are set in regression mode to retrieve a continuous value of E_1 within the given energy range. The architecture of the DNNs consists of 2,3 or 4 layers of variable sizes (M_i) and a *relu* activation function followed by a fully connected layer of one neuron with a *linear* activation function.

5.1. Hyper-parameter Optimization

We perform a hyper-parameters optimization of the DNN architectures mentioned above to estimate the energies of S records with the MAE (Equation (4)) as the objective function of the TPE Bayesian optimization. In this case, y and $\tilde{y}$ correspond to the energies E_1 and their predicted values (E_p), respectively. Then, we run the optimization pipeline for 50 trials and 100 epochs per trials, and store the MAE values, the number of operations and the list of hyper-parameters for each trial.

The hyper-parameter space is defined in the same way as in the previous classification tasks. In Figure 10, we show the results of the hyper-parameter optimization of the DNNs with $n = (2,\ 3,\ 4)$ for the energy determination of pulses in S records. The DNN with the best performance (labeled as enrg-best) is the one with $n = 3$ and has a MAE 3.3 eV and the number of operations is 33,000 (see Table 1). Additionally, the DNN with $n = 2$ (labeled as enrg-ops) has a similar performance (MAE 4.6 eV) with 2600 operations (one order of magnitude less operations with respect to the enrg-best model). Given these numbers, the enrg-ops model is clearly the cheapest choice regarding the computational cost while retrieving satisfactory estimates of the pulse energies in S records. Contrarily, the enrg-best model has a larger number of operations, but still below the number required by the high resolution reconstruction using the optimal filtering technique, for which 49,000 operations per pulse are required.

Hereafter, we perform the estimation of the pulse energies of S records with both architectures, the DNN model labeled as enrg-best and the one denoted as enrg-ops, which are described in Table 1.

5.1.1. Results

We present the results of the energy estimation of pulses in S records for the DNN with $n = 2$ and $n = 3$ described in the previous section. Since this is a more complex task, we increased the number of S records in the training sample to help

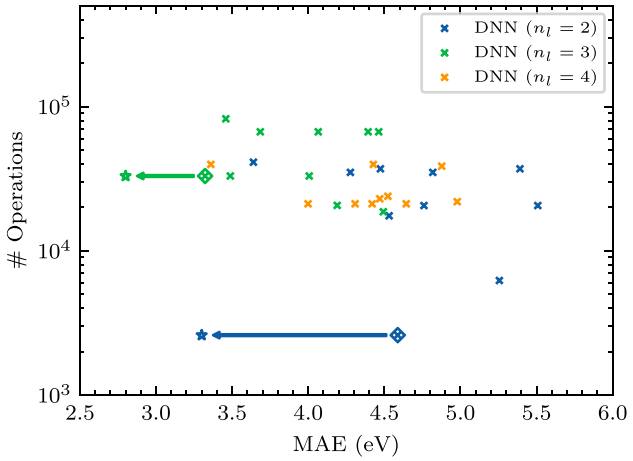

Figure 10. Hyper-parameter optimization of the DNNs for the estimation of the energy of pulses in S records. The number of operations is shown as a function of the MAE (in eV) for the 10 trials with the smallest MAE at each DNN configuration (2 layers in blue, 3 layers in green and 4 layers in orange) after 100 epochs of training. The two crosses embedded in empty diamonds highlight the selected configurations according to the number of operations (for $n = 2$ in blue) and to the $F1$ value (for $n = 4$ in orange) . The stars (in blue for the DNN with $n = 2$ and in orange for the DNN with $n = 4$) correspond to the results for the best model (among the $k = 10$ realizations) of the highlighted configurations after 300 epochs of training. The selected configurations along with the metrics are described in Table 1 and in Table 2.

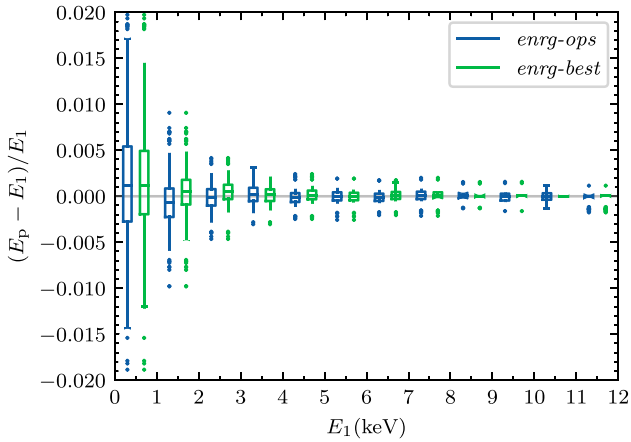

Figure 11. Boxplot for the estimated relative error, $(E_p - E_1)/E_1$, of the pulse energies in S records as a function of the E_1. The results are shown for the enrg-ops (in blue) and the enrg-best models (in green) separately. The test sample of 10,000 S records is split in bins of E_1 of 1 keV within the range [0, 12] keV. Therefore, the boxes are centered in the middle value of each corresponding bin.

the DNNs to retrieve the energies of the pulses in S records more accurately. The optimized DNNs are trained and validated with a sample of 60,000 S records (50,000 for training and 10,000 for testing). We also perform a k-fold cross validation with $k = 10$. The number of epochs is now extended to 300 with respect to the optimization procedure. After the training phase, we store the model weights at the epoch with the lowest MAE for each realization and apply it to the test sample.

We obtain median values of MAE = 4.5 eV and MAE = 3.0 eV for the 10 realizations trained for estimating the energy of the pulses in S records with the enrg-ops and enrg-best models. The realization (among the $k = 10$) with the smallest MAE for the enrg-ops model has a MAE = 3.3 eV, while for the enrg-best this value decreases to MAE = 2.8 eV. These results are quoted in Table 2. Hereafter, we show the results for the best model (i.e., smallest MAE) among the $k = 10$ realizations of the enrg-best and enrg-ops models.

In Figure 11, we show the estimated relative error of the predicted value (denoted as E_p) and the true energy of the pulses (E_1) in the 10,000 S records that conform the test sample. Despite some modest differences, the enrg-ops and enrg-best models lead to similar results. The relative errors of the predicted energies are constrained within less than ∼0.5% for pulses with $E_1 > 2$ keV. Weaker pulses (i.e., $E_1 < 2$ keV) are more difficult to characterize in terms of their energies, leading to larger values of $(E_p - E_1)/E_1$. In the bin with the less energetic pulses ($0 < E_1 < 1$ keV), the 2.7σ range extends beyond the 1% relative error with some outliers above the 1.5% error. It is also interesting to note the increasing number of outliers in the last energy bin, $11 < E_1 < 12$ keV, toward the negative values of $(E_p - E_1)/E_1$. Consequently, we conclude that for some of these powerful pulses (in the saturation limit of the detector) the DNNs are underestimating the true energy. Nevertheless, the relative errors for the majority of these particular cases are constrained within less than ∼0.1% (or within less than ∼0.2% considering the outliers). In any case, the predicted energies are in excellent agreement with the true energies of pulses in S records, showing median values of $(E_p - E_1)/E_1$ (as well as IQR) very close to zero for the whole energy range.

In order to compare our results for the energy estimation of pulses in S records with the classical optimal filtering method, in Figure 12, we show the FWHM values of the energy difference $E_p - E_1$ computed in energy intervals of 1 keV. The FHWM is approximated by 2.35 times the standard deviation of the energy difference $E_p - E_1$. The FHWM values are systematically larger for the enrg-ops than for the enrg-best model, ranging from ∼7 keV at the lowest energies to ∼14 keV for the largest energies. The overall FWHM values of $E_p - E_1$ are 8.3 eV and 10.0 eV for the enrg-best and the enrg-ops models, respectively. The FWHM values in the energy range 5–7 keV are ∼9 eV and ∼7 eV for the enrg-ops and the enrg-best models, respectively, which are more than twice the FWHM value of 2.5 eV derived with the optimal filtering method.

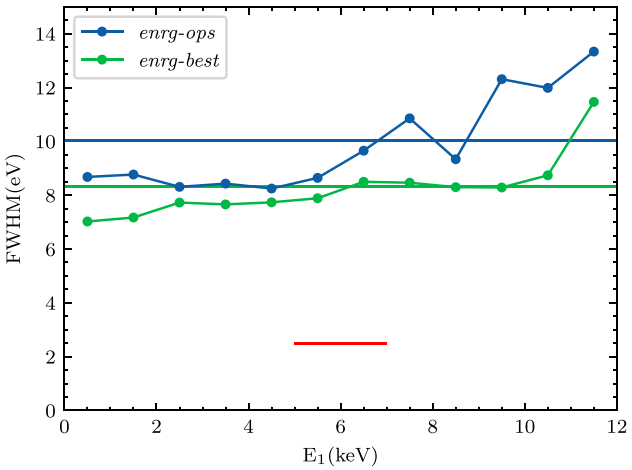

Figure 12. FWHM of the energy difference $E_p - E_1$ as a function of the true energy of the S pulses (E_1). Blue and green dots correspond to the FWHM values in energy intervals of 1 keV for the enrg-ops and the enrg-best models, respectively. The blue and green horizontal lines indicate the overall FWHM for each model. The red horizontal segment corresponds to the FWHM of 2.5 eV obtained with the classical optimal filtering method for the energy interval 5–7 keV. Note that units of FWHM are in eV.

As before mentioned, the classical optimal filtering method reconstructs accurately the energies of pulses in S records with roughly 49,000 operations. Therefore, although with a larger value of the FWHM for $E_p - E_1$, the number of operations needed for retrieving the pulse energies (with the selected number of samples in this study) is comparable or significantly smaller whether the enrg-best (3.3×10^4 operations per record) model or the enrg-ops (2.6×10^3 operations per record) are used, respectively. Note that before reconstructing the energy of a S records, this has to be previously classified as S record and, therefore, summing up 2.2×10^3 operations per record in the case of the bin-ops model, for instance.

6. Conclusions

We presented a Machine Learning approach to detect, identify and characterize current pulses produced as a response of incident X-ray photons in TES detector devices. Alternatively to the classical methods (threshold detection and optimal filtering), we employ a series of NNs (both DNNs and CNNs): a) to identify records containing single-pulse (S) or double-pulse (D) records; b) to recover the arrival time of pulses in D records; and c) to estimate the energy of the simulated X-ray pulses in S records.

The data set used to train and test the NNs consists of simulations performed with SIXTE/XIFUSIM, the Athena/X-IFU official simulator. This simulator framework constitutes a quite reliable representation of the X-IFU-like TES arrays behavior as demonstrated by the test of several reconstruction algorithms (initially studied on simulations) on TES laboratory Mn Kα X-ray data from GSFC and NIST laboratories (Ceballos, M.T. et al. 2022, in preparation). On the other hand, simulated data have the unique advantage of controlling the conditions (arrival time and energy) of the X-ray events, impossible for laboratory data where nearly coincident events could pass inadvertently. However, as some simplifications have been used in the simulations (use of non-stationary noise, for example) it will be a line for future work the study of these techniques on laboratory data in the situations where the close arrival of the events are obvious or can be spotted by other detection methods. Nonetheless, we do not foresee major discrepancies since the identification of the pulses is mainly governed by their rise time, their shape and the noise level, parameters that are quite accurately described by the SIXTE/XIFUSIM framework. It should be stressed though that the results for the NN architectures presented here are optimal for the specific X-IFU-like TES devices, probably being necessary a further adjustment for other configuration of TES detectors with different rise time, pulse shapes or noise models.

As they are initially thought for an on board application, our NN architectures are designed to maximize the performance of the NNs, while minimizing their complexity (i.e., reducing the computational cost in terms of the number of operations).

For the identification of the different types of records we optimize and train a series of NNs configured in binary mode to distinguish between S and D records. The data set used for training and testing consist of 60,000 records, 30,000 of each type (S and D records). The input vectors of 128 samples for the NNs contain the values of the first derivative of the pulse intensity. For the binary classifier we choose the DNNs with $n = 2$ and $n = 4$ (labeled as bin-ops and bin-best, respectively) described in Table 1 for showing an excellent performance ($F1 \gtrsim 0.99$ in both cases, see Equation (3)).

Given the limitations arisen from being on-board of a space mission (such as the CPU, RAM and storage capacities), the best suited option would be the DNN architecture labeled as bin-ops (described in Table 1) for the binary classification of S and D records. In terms of the lowest computational cost, the bin-ops model has a number of operations of 2200. In other words, the bin-ops model keeps a reasonable computational cost with an excellent performance in terms of its $F1$ score. For this DNN architecture, the number of S records miss-classified as D records is roughly 2 in $\sim$5000, while the number of D records miss-classified as S records is 72 in $\sim$5000 ($\sim$1%). Consequently, the small fraction of false negative cases translates into a high completeness sample of S pulses, recall $= 4977/(4977 + 2) \approx 1.0$, while keeping a high purity, precision $= 4977/(4977 + 72) \approx 0.99$). More importantly, as shown in Figure 5, our DNNs in binary mode only fail for very extreme cases of D records with both large E_{12} and small d_{12} parameters. Despite these cases, our DNN clearly surpasses the detection performance of the classical triggering approach for the full range of photon energy and arrival time combinations.

In a second phase, we optimize and train a series of NNs designed to estimate the arrival time of pulses in D records. The arrival times of the secondary pulses in D records are drawn from a uniform distribution and made of integer values within the range of [1, 100] samples. Therefore, the NNs for this task are created in regression mode to retrieve the difference in the arrival time of two consecutive X-ray pulses stored in the same record (i.e., d_{12}). In particular, we propose to use DNNs since they constitute a better approach than the CNNs not only in terms of performance, but mainly regarding the computational cost. After the hyper-optimization process, we find that two DNNs with $n = 4$ (see Table 1 for a detailed description of their architectures) are the ones with the best performances and reasonable computational costs for the arrival time estimation of D records. In particular, the model with the best performance and a modest computational cost (15,000 operations), denoted as time-ops, shows a MAE = 0.62 samples (see Equation (4)). Again, the largest differences between the predicted and the true arrival times of pulses in D records (i.e., $d_p - d_{12}$) are mostly due to records with values of $d_{12} \lesssim 20$ samples and values of $E_{12} \gtrsim 5$ keV. In contrast to the classical method, the results for the DNN presented here allow a "sub-sample" estimation (i.e., MAE $\lesssim 1$ sample) of the pulse separation in D records, even for those in which pulses have very close arrival times.

By knowing when a D record is detected using the binary classifier denoted as bin-ops, and what is the separation between the two consecutive pulses (d_{12}) applying the time-ops model for the arrival time estimation, we are obtaining a extremely valuable information to help the classical method to recovering for the analysis the pulses in double-pulse records being able to retrieve a more accurate estimate of the (primary and secondary) pulse energies.

In a final stage, we describe the methodology used to infer the energies of pulses in S records and compare the results we obtained with the well-tested optimal filtering technique. For this task we optimize and train a DNN in regression mode to infer the input pulse energy within the energy range [0.2–12] keV. We select two DNNs with different architectures: the first one with $n = 3$ denoted as enrg-best and the second with $n = 2$ labeled as enrg-ops. Although both architectures obtain similar performances (MAE ≈ 3 eV), the enrg-ops model has reasonably low computational cost (of 2600 operations) compared to the one with enrg-best model (with more than 10 times more operations). After the training and testing phases, we obtain a MAE ≈ 3.3 eV for the difference between the estimated and the true pulse energies, $E_p - E_1$. The median values of the estimate relative error of $E_p - E_1$ are close to zero in the whole energy range with larger IQR toward the low-energy end of E_1. However, the FWHM of $E_p - E_1$ for energies within 5–7 keV is $\sim$9 eV for the enrg-ops model ($\sim$7 eV for the enrg-best model). Summarizing, although keeping a low number of operations (almost one order of magnitude below the optimal filtering technique), the FWHM values obtained with our enrg-ops model are more than $\sim$3 times larger than the optimal filtering estimates for low and intermediate pulse energies, and well above the requirements of the X-IFU instrument.

The failure to determine with enough accuracy the energy of the single pulses is what ultimately leads us to conclude that the machine learning methods developed here are not a viable alternative to the current baseline optimal filtering technique to estimate the energy of the pulses, despite their higher computing efficiency. However, they can be a very valuable help to recover pulses considered as piled-up by the classical triggering mechanisms, that otherwise should be excluded from the analysis by an additional identification procedure, or (in the worst case scenario) would be incorporated to the analysis as single pulses, thus contributing to worsen the spectral resolution.

This paper is supported by European Union's Horizon 2020 research and innovation program under grant agreement No 871158, project AHEAD2020. JP-G and PG acknowledge the project "Machine Learning for the adaptation and improvement of applications" (MALGAMA) under the CSIC Intramural 20152170 program. The authors gratefully acknowledge the computer resources at Artemisa, funded by the European Union ERDF and Comunitat Valenciana as well as the technical support provided by the Instituto de Física Corpuscular, IFIC (CSIC-UV). The authors also acknowledge the computer resources provided by the Clúster d'Altes Prestacions per Inteligència Artificial at the Instituto de Investigación en Inteligencia Artificial (IIIA-CSIC) and of the Grupo de Astrofisica y Cosmologia computacional at the Universidad Autónoma de Madrid (UAM).

ORCID iDs

J. Vega-Ferrero, https://orcid.org/0000-0003-2338-5567
B. Cobo https://orcid.org/0000-0002-7480-91900000-0002-7480-91900000-0002-7480-9190
F. J. Carrera https://orcid.org/0000-0003-2135-9023

References

Alpert, B., Ferri, E., Bennett, D., et al. 2016, JLTP, 184, 263
Barret, D., Lam Trong, T., den Herder, J.-W., et al. 2018, Proc. SPIE, 10699, 106991G
Bergstra, J., Yamins, D., & Cox, D. D. 2012, arXiv:1209.5111
Borghesi, M., De Gerone, M., Faverzani, M., et al. 2021, EPJC, 81, 5
Bottrell, C., Hani, M. H., Teimoorinia, H., et al. 2019, MNRAS, 490, 5390
Boyce, K. R., Audley, M. D., Baker, R. G., et al. 1999, Proc. SPIE, 3765, 741
Busch, S., Adams, J., Bandler, S., et al. 2016, JLTP, 184, 382–8
Campagne, J.-E. 2020, arXiv:2002.10154
Ceballos, M. T., Cobo, B., Peille, P., et al. 2017, in Astronomical Data Analysis Software and Systems XXV 512 of ASP Conf. Ser., ed. N. P. F. Lorente, K. Shortridge, & R. Wayth (San Francisco, CA: ASP), 605

Cobo, B., Ceballos, M. T., Peille, P., et al. 2018, Proc. SPIE, 10699, 106994S
Dauser, T., Falkner, S., Lorenz, M., et al. 2019, A&A, 630, A66
Delaquis, S., Jewell, M. J., Ostrovskiy, I., et al. 2018, JInst, 13, P08023
Domínguez Sánchez, H., Huertas-Company, M., Bernardi, M., Tuccillo, D., & Fischer, J. L. 2018, MNRAS, 476, 3661
Elola, A., Aramendi, E., Irusta, U., et al. 2019, Entropy, 21, 305
Flores, J. L., Martel, I., Jiménez, R., Galán, J., & Salmerón, P. 2016, NIMPA, 830, 287
Fowler, J. W., Alpert, B. K., Doriese, W. B., et al. 2015, ApJS, 219, 35
Griffiths, J., Kleinegesse, S., Saunders, D., Taylor, R., & Vacheret, A. 2018, arXiv:1807.06853
Hausen, R., & Robertson, B. E. 2020, ApJS, 248, 20
Ho, M., Rau, M. M., Ntampaka, M., et al. 2019, ApJ, 887, 25
Huertas-Company, M., Gravet, R., Cabrera-Vives, G., et al. 2015, ApJS, 221, 8
Irwin, K., & Hilton, G. 2005, Transition-Edge Sensors, 99 (Berlin: Springer), 81
Kirsch, C., Lorenz, M., Peille, P., et al. 2021, J. Low Temp. Phys., submitted
Lorenz, M., Kirsch, C., Merino-Alonso, P. E., et al. 2020, JLTP, 277
Mardiyanto, M. P., Uritani, A., Sakai, H., Kawarabayashi, J., & Iguchi, T. 2001, NIMPA, 462, 405
Metcalf, R. B., Meneghetti, M., Avestruz, C., et al. 2019, A&A, 625, A119
Miniussi, A. R., Adams, J. S., Bandler, S. R., et al. 2018, Superconductor Science and Technology, 193, 337
Nandra, K., Barret, D., Barcons, X., et al. 2013, arXiv:1306.2307
Pasquet, J., Bertin, E., Treyer, M., Arnouts, S., & Fouchez, D. 2019, A&A, 621, A26
Peille, P., Ceballos, M. T., Cobo, B., et al. 2016, Proc. SPIE, 9905, 99055W
Powers, D. A. 2011, J. Mach. Learn. Technol, 2, 2229
Ripoche, P., & Heyl, J. 2021, Journal of Astronomical Telescopes, Instruments, and Systems, 7, 1
Sanchez-Gonzalez, A., Micaelli, P., Olivier, C., et al. 2017, NatCo, 8, 15461
Snoek, J., Larochelle, H., & Adams, R. P. 2012, arXiv:1206.2944
Su, Y., Zhang, Y., Liang, G., et al. 2020, MNRAS, 498, 5620–8
Szymkowiak, A. E., Kelley, R. L., Moseley, S. H., & Stahle, C. K. 1993, JLTP, 93, 281
Tan, H., Hennig, W., Warburton, W. K., Doriese, W. B., & Kilbourne, C. A. 2011, ITAS, 21, 276
Ullom, J. N., & Bennett, D. A. 2015, Superconductor Science and Technology, 28, 084003
Vega-Ferrero, J., Sánchez, H. D., Bernardi, M., et al. 2021, MNRAS, 506, 1927–43
Walmsley, M., Ferguson, A. M. N., Mann, R. G., & Lintott, C. J. 2019, MNRAS, 483, 2968
Yan, D., Cecil, T. W., Gades, L. M., et al. 2016, JLTP, 184, 397