O'TRAIN: A robust and flexible 'real or bogus' classifier for the study of the optical transient sky

K. Makhlouf, D. Turpin, D. Corre, S. Karpov, D. A. Kann, A. Klotz

A&A 664, A81 (2022)
https://doi.org/10.1051/0004-6361/202142952
© K. Makhlouf et al. 2022

Astronomy & Astrophysics

O'TRAIN: A robust and flexible 'real or bogus' classifier for the study of the optical transient sky[⋆]

K. Makhlouf[1,2], D. Turpin[1], D. Corre[3], S. Karpov[4], D. A. Kann[5], and A. Klotz[6,7]

[1] Université Paris-Saclay, CNRS, CEA, Astrophysique, Instrumentation et Modélisation de Paris-Saclay, 91191, Gif-sur-Yvette, France
e-mail: damien.turpin@cea.fr
[2] Ecole Centrale Lille, Cité scientifique CS 20048, 59651 Villeneuve d'Ascq cedex, France
[3] Sorbonne Université, CNRS, UMR 7095, Institut d'Astrophysique de Paris, 98 bis boulevard Arago, 75014 Paris, France
[4] CEICO, Institute of Physics of the Czech Academy of Sciences, Prague, Czech Republic
[5] Instituto de Astrofísica de Andalucía (IAA-CSIC), Glorieta de la Astronomía s/n, 18008 Granada, Spain
[6] IRAP, Université de Toulouse, CNRS, UPS, 14 Avenue Edouard Belin, 31400 Toulouse, France
[7] Université Paul Sabatier Toulouse III, Université de Toulouse, 118 route de Narbonne, 31400 Toulouse, France

Received 19 December 2021 / Accepted 20 April 2022

ABSTRACT

Context. Scientific interest in studying high-energy transient phenomena in the Universe has risen sharply over the last decade. At present, multiple ground-based survey projects have emerged to continuously monitor the optical (and multi-messenger) transient sky at higher image cadences and covering ever larger portions of the sky every night. These novel approaches are leading to a substantial increase in global alert rates, which need to be handled with care, especially with regard to keeping the level of false alarms as low as possible. Therefore, the standard transient detection pipelines previously designed for narrow field-of-view instruments must now integrate more sophisticated tools to deal with the growing number and diversity of alerts and false alarms.

Aims. Deep machine learning algorithms have now proven their efficiency in recognising patterns in images. These methods are now used in astrophysics to perform different classification tasks such as identifying bogus from real transient point-like sources. We explore this method to provide a robust and flexible algorithm that could be included in any kind of transient detection pipeline.

Methods. We built a convolutional neural network (CNN) algorithm in order to perform a 'real or bogus' classification task on transient candidate cutouts (subtraction residuals) provided by different kinds of optical telescopes. The training involved human-supervised labelling of the cutouts, which are split into two balanced data sets with 'true' and 'false' point-like source candidates. We tested our CNN model on the candidates produced by two different transient detection pipelines. In addition, we made use of several diagnostic tools to evaluate the classification performance of our CNN models.

Results. We show that our CNN algorithm can be successfully trained on a large and diverse array of images on very different pixel scales. In this training process, we did not detect any strong over- or underfitting with the requirement of providing cutouts with a limited size no larger than 50×50 pixels. Tested on optical images from four different telescopes and utilising two different transient detection pipelines, our CNN model provides a robust 'real or bogus' classification performance accuracy from 93% up to 98% for well-classified candidates.

Key words. methods: numerical – techniques: image processing

1. Introduction

Time-domain astronomy consists of studying variable sources in the Universe that periodically change in brightness or transient sources coming from the sudden outburst of known flaring objects or single cataclysmic events. Over the past few years, our way of observing and monitoring the optical transient sky has significantly evolved both with the arrival of new optical synoptic survey projects and the advent of multi-messenger astronomy. It is now a matter of observing consistently greater portions of the sky at higher sensitivities and with a high image cadence. New optical synoptic surveys such as ATLAS (Tonry et al. 2018), GOTO (Dyer et al. 2020), MeerLICHT and BlackGEM (Groot et al. 2019), SVOM/GWAC (Han et al. 2021), ZTF (Bellm et al. 2019; Graham et al. 2019), or the upcoming Vera Rubin/LSST (Ivezić et al. 2019) are now able to observe their entire observable sky over a timespan of just a few nights. Thanks to their large fields of view (FoV) and high image cadences at moderate and high sensitivities, a fraction of their observation time can also be dedicated to the follow-up of poorly localised multi-messenger alerts sent by gravitational wave detectors or high-energy neutrino and gamma-ray telescopes. The current optical surveys have already led to the discovery of tens of thousands of new optical transients. Those transients in part belong to known astrophysical classes such as supernovae or galactic flaring stars, while new classes of transients have been recently discovered. For example, since 2018, the ATLAS and ZTF surveys have already confirmed the existence of a new type of transient called fast blue optical transients, (FBOTs), for which only four events have been robustly identified thus far: ATcow18 (Smartt et al. 2018; Prentice et al. 2018), ZTF18abvkwla (Ho et al. 2020), CSS161010 (Coppejans et al. 2020), and AT2020xnd/ZTF20acigmel (Bright et al. 2022; Perley et al. 2021; Ho et al. 2021). Down to minute timescales, the SVOM/GWAC fast cadence survey (Han et al. 2021) has also detected new powerful outburts from nearby M dwarf stars (Xin et al. 2021; Wang et al. 2021). Finally, a new class of luminous supernova explosions, namely, the so-called super luminous

[⋆] The codes and diagnostic tools presented in this paper are available at https://github.com/dcorre/otrain supernovae (SLSNe), have also been identified as recently as a decade ago, with about 100 candidates currently reported and around 20 under extensive study (Quimby et al. 2011; Gal-Yam 2012, 2019).

In fact, we are just starting to extensively explore the plethora of optical transients, especially towards the shortest timescales (from minutes to hours). Hence, there is room for new discoveries by adapting the different survey or follow-up observational strategies (image cadence, filter strategy, frequency of revisits, limiting magnitude depth, etc.) as well as the algorithm used for identifying the different types of transients. Behind the scenes, there is actually an unprecedented amount of images and data available to process as well as transient candidates to characterise every night. As an example, a survey such as ZTF can typically produce up to 2×10^6 raw alerts at 5σ confidence level every clear night, among which 10^3–10^5 are likely real sources (Masci et al. 2019). This is nothing in comparison to the millions of real alerts (corresponding to 20 terabytes of data every night) the *Vera Rubin* LSST survey will produce daily during its 10 yr of operations. As a consequence, new detection pipelines, used for detecting and identifying in near real-time new sources in optical images, have to be designed by taking in consideration these new alert flow constraints. To this aim, many of them now make use of (deep) machine learning (ML) methods inspired by big data solutions. Deep ML algorithms allow for a quick processing of astrophysical images or different sets of complex information to give classification probabilities in near real-time. These probabilities can then be easily interpreted by astronomers to trigger further targeted follow-up observations. Usually, in time-domain astronomy, ML algorithms are used to perform the two following classification tasks: (1) ‘real or bogus’ classification to reduce the false alarm rate due to artefacts that may be falsely identified as new real sources (Gieseke et al. 2017; Duev et al. 2019; Turpin et al. 2020; Killestein et al. 2021; Hosenie et al. 2021) and (2) astrophysical classification to identify the types of transients the detection pipelines have detected based on several pieces of information about some key parameters evolving with time, the flux-colour evolution of the transient or its spectral shape and associated features, etc. (Carrasco-Davis et al. 2019, 2021; Möller & de Boissière 2020; Burhanudin et al. 2021).

These fast classifiers are now a standard in the software architecture of any detection pipeline to avoid the need for human intervention as much as possible. This is particularly relevant when the goal is to perform systematic manual visual inspections of the transient candidate cutouts, since this is a heavily time-consuming and sometimes complex task. In addition, when dealing with the processing of big data sets, the ML algorithms generally turn out to be more reliable and stable than human decision-taking, yielding better scientific perspectives for each promising transient candidate.

In this paper, we propose a robust machine learning algorithm called Optical TRAnsient Identification Network (O’TRAIN) to filter out any type of bogus from a list of optical transient (OT) candidates a detection pipeline may output. Our ‘real or bogus’ (RB) classifier is based on a convolutional neural network (CNN) algorithm, a method that has already proved its efficacy in such a classification task (Gieseke et al. 2017; Turpin et al. 2020). We developed a store of pedagogical tools to easily launch a training procedure and diagnose the classification performances. Therefore, we can provide a generic and flexible classifier that can be embedded in any detection pipeline and applied to a broad range of image characteristics. In Sect. 2, we briefly describe the general task a detection pipeline is supposed to perform and the expected outputs that will be used by our CNN model. We then detail, in Sect. 3, the architecture of our CNN and the implementation setup to perform the training presented in Sect. 4. The different data sets and the diagnosis tools used to evaluate the performances of our CNN architecture are described in Sects. 5 and 6, respectively. We show our results in Sect. 7 and we discuss some important future prospects for our work in Sect. 8. Our conclusions are given in Sect. 9.

2. Transient detection pipeline: Inputs and outputs

To find new transient sources in optical images, a detection pipeline usually acts on the so-called ‘science-ready’ images or pre-processed images. These images are actually the raw images corrected from the dark, bias, offset, and flat field master images and for which the astrometry has been calibrated with the WCS coordinates system using dedicated software such as SCAMP[1] (Bertin 2006) or ASTROMETRY.NET[2]. A second pre-processing step is usually needed to obtain a photometric calibration of the scientific images. This is done in several steps: (1) first, we extract the position of the point-like sources using, for example, the SEXTRACTOR software (Bertin & Arnouts 1996) or the python library SEP (Barbary 2016); (2) we estimate the point spread function (PSF) model of the image or in different regions of the images (PSF varying model) using such software as PSFEX (Bertin 2011, 2013); (3) we subtract the image background and mask the brightest (saturated) stars that can lead to errors when measuring both their centroid positions and total flux in analog digital units (ADU). This tasks can make use of the PHOTUTILS[3] Astropy package (Bradley et al. 2021) and PSFEX; (4) we cross-match the detected sources in the image with known star catalogues and build the photometric model for their instrumental magnitudes. This is usually done using the Xmatch (Boch et al. 2012; Pineau et al. 2020) and Vizier services of the CDS in Strasbourg, France.

Following these steps, the science-ready images can be exploited to search for new transient sources. Some detection pipelines might also add few other steps by already removing most of the cosmic-ray induced-artefacts or identifying moving objects (asteroids and solar system objects). Two methods are traditionally employed to find OT candidates: catalog cross-matching and the difference image analysis.

2.1. Catalog cross-matching method

This method consists in searching for uncatalogued sources in the scientific images. The list of sources and their positions extracted by SEXTRACTOR are compared with the positions of known stars by associating both sources within a given cross-match radius (typically a few arcseconds). To be able to compare all the extracted sources with a given catalog, it is necessary to choose the catalogs that have deeper image sensitivities than the science images. This method is easy to set up and to apply to a large amount of images but it suffers from two main important limitations. First, depending on the image pixel scale and the accuracy of the astrometric calibration of the science images, this method may hardly distinguish blended sources, leading either to false positive cross matchings or wrong mismatches. Secondly, for flaring or variable unknown sources, this method is limited to the detection of only large flux amplitude variations

[1] https://www.astromatic.net/software/scamp/
[2] https://astrometry.net/
[3] https://github.com/astropy/photutils

between the science and the reference images. For these reasons, the catalog cross-matching method would generally yield an incomplete list of transient candidates. This list must be completed by a difference image analysis to collect the transient sources close to catalogued sources or having faint variations in brightness, or both.

2.2. Difference image analysis

By definition, optical transients are new sources that suddenly imprint their patterns in the images with their fluxes strongly evolving with time. Performing the subtraction of the science images with some reference images allows to reveal the presence of these new sources as significant excesses in the residual images. While being a powerful tool to identify transient and flux-varying sources, it implies to perform several important steps, which require fine tuning prior to the subtraction. The reference images have to be carefully chosen in order to not contain the transient sources. Depending on the timescale of the OT flux evolution, the reference images are usually taken days to months prior to the science images. A reference image can originate either from the same telescope that acquired the science image or from all-sky surveys providing public image databases such as Pan-STARRS (Chambers et al. 2016) or 2MASS (Skrutskie et al. 2006). These catalogs of reference images can be directly downloaded from a catalog server or extracted by using the hips2fits[4] service at the CDS, Strasbourg. For such a subtraction technique, it is preferable to obtain reference images of deeper limiting magnitudes and better seeing than the science images as well as taken with the same or at least close filter system. A bad pixel map for the reference image can also be produced to already remove bad pixel values or saturated stars in the subtraction process. The science and reference images must be well aligned and the PSF resampled if the reference images originate from an all-sky survey have different pixel scales. A flux normalisation of the two sets of images is also performed to ensure the good quality of residual images. These steps and the image subtraction can be done by software such as MONTAGE[5] (image realignment) and HOTPANTS[6] (Becker 2015).

2.3. Minimum output required for a transient detection pipeline

Both of the methods mentioned above will output two lists of OT candidates that will then be merged into a single one to avoid redundancies. This final list basically describes the properties of each OT candidate, including: (1) Their celestial and physical coordinates in the image; (2) their Full Width at Half Maximum or FWHM; (3) their measured magnitude with the associated error; (4) their detection time; (5) additional information and flags that may help in classifying them such as edge position and extended object flags, signal-to-noise ratio (S/N), light curves, and so on.

In addition, small (typically few tens of pixels) cutouts centered at the position of the OT candidates are usually cropped from the original, reference and residual images for a manual visual inspection by a scientific expert. After the visual inspection, the OT candidates are kept either as promising sources to be followed-up or discarded. This selection task is exactly what a RB classifier must efficiently do. These cutouts and primarily the residuals will be the input for our RB classifier to deliver its decision.

2.4. Sources of artefacts

The various artefacts that we can find in astronomical images are usually produced via three main steps. First, during the acquisition, the images can be contaminated by cosmic ray tracks or blooming or crosstalk effects from the saturated and bright stars, artificial sources or tracks left by human-made flying objects (satellites, planes, etc.), or hot or bad groups of pixels. The so called 'ghost' sources (diffuse extended sources) are also observed in optical images due to multiple light reflections from the optical system back to the CCD chip. Secondly, at the CCD reading step, some issues can lead to some columns of pixels that are no longer exploitable. Finally, some artefacts can appear when subtracting two images either due to bad image alignment or a lack of optimisation of the HOTPANTS parameters especially for the subtraction of survey catalog images which have a different pixel scales compared to the science image. At the end, all these processes create a very large and diverse collection of artefacts that may lead to many false detections. While some artefacts are easily recognisable, others can hardly be distinguished from point-like sources by eye or with standard filtering systems such as PSF-matching methods.

3. O'TRAIN: A 'real or bogus' CNN classifier

3.1. Purpose of O'TRAIN

Artificial intelligence models are built to lighten the workload of the astronomers on duty and to help them in the decision-making process. They take on annotated data and try to understand the logic behind them by creating connections between their properties. Our work involves cutouts centered at the position of the OT candidates that we want to classify into two folders: real transients and bogus ones. The most dominant model in this sort of computer vision tasks is the convolutional neural network since they use every information in the input image (the pixels) without being computationally expensive. By applying filters, the CNN highlights regions of the image according to their relevance when classifying the image, hence the name convolutional (Yamashita et al. 2018). These regions are commonly called 'features'. They would distinguish, in our case, the real sources from the bogus ones. In this section, we go into more details about our model architecture, its configuration settings and how it learns to decide the relevant features of the source, and their spatial hierarchies.

3.2. Model architecture

We start off by extracting the features. At each convolutional layer, multiple filters are applied to the input of the layer, the results of these convolutions are called a feature maps. This output goes through a pooling layer that enhances the features, while maintaining the spatial information. The final output feature maps are then fed to fully connected layers (i.e. the 'dense' layers) thus connecting the features to the corresponding class. The number of filters in each convolutional layer, their sizes, the number of cells in the dense layers, are all configurations that should be decided to optimise the model's performance on our task. Since our cutouts are small (roughly few tens of pixels),

[4] https://alasky.u-strasbg.fr/hips-image-services/hips2fits
[5] http://montage.ipac.caltech.edu/docs/montagePy-UG.html
[6] https://github.com/acbecker/hotpants

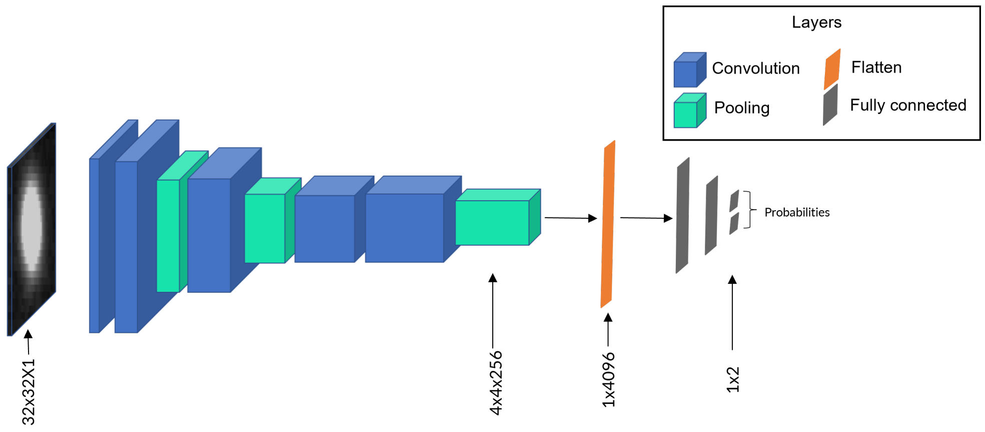


Fig. 1. Illustration of the O'TRAIN model architecture for performing the binary classification task: real or bogus transients.

we chose to apply 3×3 filters in our convolutional layers. The number of filters is increasing throughout the network. The more filters we use the more features we extract. As we did not want to extract features from the noise next to the source, we limited the number of filters in the first layers. In Fig. 1, we illustrate the different layers of our CNN among which the input images have been analysed step by step.

3.3. Training procedure

After building the CNN model, we launched the training procedure, where the model updates the parameters connecting the layers (also called trainable parameters) in order to minimise the difference between its prediction and the ground truth binary label. The data set is first ruffled and divided into training and validation sets. Each of these data sets are then split into batches (or samples). The model starts the training process by taking the first batch of images. They go through one by one and propose an update to the trainable parameters. We average these proposals and update the parameters accordingly, and move on to the next batch. After the last batch of the training set, the updated model is applied to the validation data set, so as to see if it can perform well on images it has never seen before. This is the generalisation aspect we look for in a deep learning model. This procedure, illustrated in Fig. 2, is done within one training step also called an epoch. The more epochs, the better the model's performance will be, provided that the performance on the validation data set does not deteriorate (cf. Sect. 6.3).

This procedure calls for a fine tuning of key parameters to optimise the training environment of the model, as follows: (1) The fraction of the original data set used to make the training and validation sets: this fraction is usually 70% over 30% to 90% over 10% for the training and validation data sets respectively. It is strongly unbalanced in order to enable the model to train on the maximum number of images while keeping enough images for the validation purpose. In this work, we set this value at 85% over 15%; (2) The number of epochs: as mentioned above, this parameters influence significantly the performance of the model. We trained our model with 30 epochs, and implemented a method to trace any classification performance deterioration on the validation data set; (3) The batch size: this parameter represents the number of images per sample in the training procedure. The bigger the size (more than 512), the more accurate our updates will be, with a trade-off involving the training speed. Since we used a GPU for our training, we set this value to 1024;

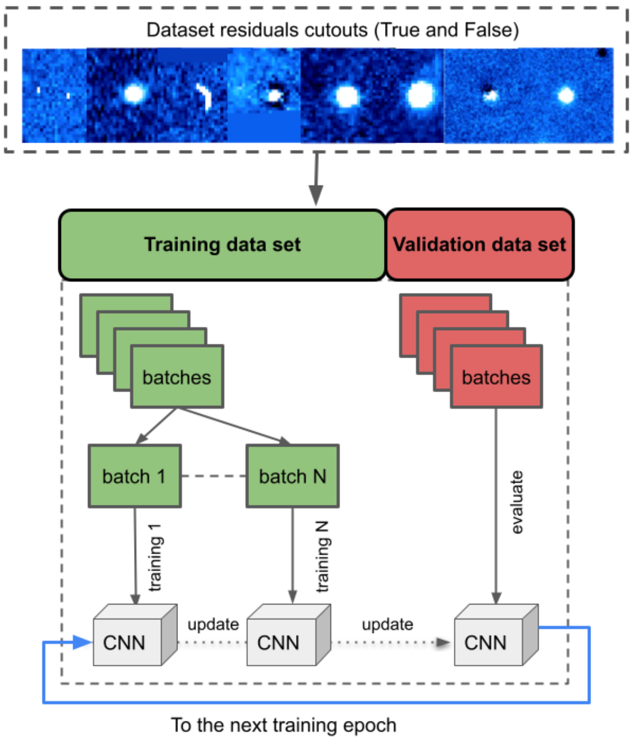
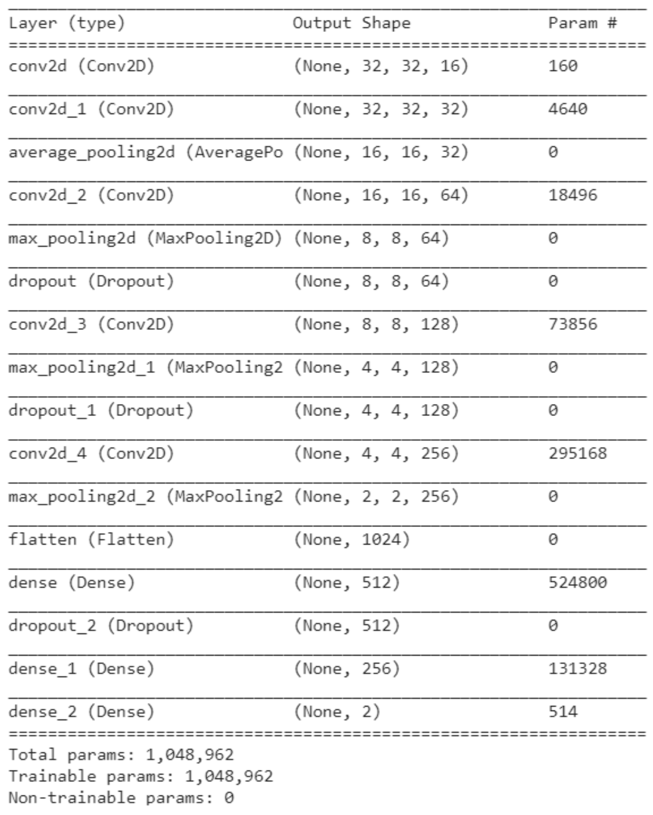


Fig. 2. Schema of the training procedure in O'TRAIN: In one training epoch, the original data set is split into two sub sets to train/update the CNN model and validate the classification results on a new batch of images. By repeating N times these epochs, the model converges towards the desired classification performances.

(4) The optimiser: the optimiser interferes in the way the trainable parameters get updated. The dominant optimiser for CNN applications is the Adam; (5) The learning rate: this configuration concerns how much the parameters get updated. We set this value to 0.001 in order to allow the model to converge to the minimum steadily.

3.4. Python implementation

We implemented the model using the Python libraries TENSORFLOW and KERAS compatible with Python version 3.7 and above. In Fig. 3, we show the output of KERAS displaying our CNN architecture and the number of trainable parameters after



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 32, 32, 16)        160
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 32, 32, 32)        4640
_________________________________________________________________
average_pooling2d (AveragePo (None, 16, 16, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 64)        18496
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 8, 8, 64)          0
_________________________________________________________________
dropout (Dropout)            (None, 8, 8, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 128)         73856
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 4, 128)         0
_________________________________________________________________
dropout_1 (Dropout)          (None, 4, 4, 128)         0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 4, 4, 256)         295168
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 2, 2, 256)         0
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 512)               524800
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               131328
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 514
=================================================================
Total params: 1,048,962
Trainable params: 1,048,962
Non-trainable params: 0
```

Fig. 3. Layer architecture and the list of the trainable parameters of O'TRAIN displayed by KERAS before launching a training.

launching a training procedure. All our codes as well as the diagnostic tools we present in this paper are publicly available in a documented git project[7].

4. Training procedure of O'TRAIN

4.1. Building the training data set

To build a training data set for our O'TRAIN CNN, we have to start by creating two folders labelled 'True' and 'False' , in which we store the small cutouts of the subtraction residuals centered at the positions of the real transient sources and the bogus, respectively. Each folder must contain sources that will be as close as possible to the ones the imaging instruments and the detection pipelines will output in real conditions. In addition, we require some non-standard keywords in the fits header of the 'True' and 'False' cutouts that will be used to quickly identify and characterise the candidates afterwards. We list them here: (1) CANDID (mandatory) : unique ID or name of the candidate; (2) MAG (optional): magnitude of the source; (3) MAGERR (optional): error on the magnitude of the source; (4) FILTER (optional) : filter used for the observation; (5) FWHM (optional): estimation of the Full-Width at Half Maximum of the source; (6) EDGE (mandatory): 'True' if the source is close an image edge (depends on the detection pipeline setup) and 'False' otherwise. The label of all the candidates stored in the 'True' folder is set to "1" while it is set to "0" for the candidates in the 'False' folder. The label of all the candidates stored in the 'True' folder is set to '1', whereas it is set to '0' for the candidates in the 'False' folder. For a supervised ML algorithm as O'TRAIN, it is required to provide a large amount of labelled candidates for the training, typically several thousands. Depending on the telescope discovery potential, OT sources are sometimes too rare in the science images to sufficiently populate the 'True' folder. To overcome this issue, point-like sources can be simulated in the original science images in order to artificially increase the number of real OT candidates (data augmentation techniques, see Gieseke et al. 2017). Artefacts can also be simulated (if needed) using software like SKYMAKER (Bertin 2009). We note that in all our analysis, we did not make use of such a technique for simulating artefacts as we had obtained enough of them in the science and residual images after performing the image subtractions. However, we had to simulate most of our OT candidates in the original images. In the following section, we describe how we have simulated additional point-like sources in our science images and the building of the final data cube that will then be used for training the CNN model.

[7] https://github.com/dcorre/otrain

Table 1. The numpy arrays stored in the training data cube python dictionary (.npz).

Numpy array	Content
CUBE	The cutouts centered at the positions of the candidates
CANDIDS	The unique IDs or names of the candidates
MAGS	The value of the MAG keywords
MAGERRS	The value of the MAGERR keywords
FILTERS	The value of the FILTER keywords
FWHMS	The value of the FWHM keywords
LABELS	The value of the labels ('1' or '0')

4.2. Point-like source simulation

To simulate point-like sources in an optical image, a model of the PSF response has to be determined to give the adequate shapes of the simulated star-like sources. We used PSFEX to estimate the PSF response function of the science images into which we wanted to inject sources. To be more realistic, we estimated a spatially varying PSF response for each science image in order to take into account the possible distortion of the PSF in different regions of the images and, in particular, close to the edges. Hence, we divided the science images into grids of 9×9 regions. In each grid, we simulated N point-like sources at random positions by convolving a polynomial function with the local PSF response function. On top of these new simulated sources, we finally added a shot noise signal. We note that in our simulations, we did not take into account the positions of pre-existing sources in the science images when we inserted our simulated sources. This choice was motivated by the fact that in real conditions, an OT source could lie close to a catalogued source, be blended with it, or even be detected very close to the image edges. As a consequence, this may lead to possible failures when trying to detect these simulated sources depending on the detection pipeline setup. The injected sources are simulated in a wide range of magnitudes in order to test our CNN classification performances in the context of different conditions – from bright stars up to the faintest ones close to the detection limit.

4.3. Training data cube

Once the 'True' and 'False' folders are adequately filled by enough candidate cutouts, we processed all of them to build

Table 2. Telescope and image properties of the four telescopes used to test our CNN model.

Tel. name	D (cm)	FoV radius (arcmin)	Pixel scale (”/pixel)	Im. size (pixels)
JAST	83	60.42	0.58	9216×9232
FRAM-N	25	18.72	1.52	1056×1024
TACA	16	51.66	2.81	1832×1224
TCA	25	79.81	3.31	2048×2048

a final data cube that is to be given as a single input to train our CNN model. The final data cube contains several python numpy arrays (.npy format) zipped in a dictionary-like archive as a .npz format. The different objects of the .npz dictionary are described in Table 1. We note that if some candidates are flagged as ‘edge’ sources, they are automatically discarded from the final data cube. We note also that the content of this data cube is not set in stone as additional relevant parameters could be added if needed. In the data cubes we have simulated, the ‘True/False’ ratio is kept balanced (50%/50%), however, we have implemented the possibility to produced unbalanced ’True/False’ data cube if needed.

5. Testing O’TRAIN on different data sets

One of our main goals is to propose a CNN model that can robustly classify real sources and bogus coming from a wide range of optical instruments (i.e. covering a wide range of pixel scale values, PSF response functions, depth of the limiting magnitudes, etc.) and detection pipelines. Therefore, we decided to test our O’TRAIN CNN on four different telescope images processed by two different detection pipelines. Many of the science images we have used were kindly provided by the telescope teams of the GRANDMA Collaboration[8] (Agayeva et al. 2021), which operate a worldwide robotic telescope network. In particular, during the third acquisition run of the GW LIGO/Virgo detectors, GRANDMA took a large amount of images covering different sky regions (Antier et al. 2020a,b). The diversity of the weather and seeing conditions found in those images allowed us to build unbiased training data sets. We choose to use the images produced by the following telescopes: (1) the Javalambre Auxiliary Survey Telescope (JAST/T80) located at the Observatorio Astrofísico de Javalambre[9] (OAJ); (2) the FRAM-CTA-N telescope located at the Observatorio del Roque de los Muchachos[10]; (3) the TAROT Calern (TCA) telescope located at the Calern French Plateau (Observatoire de la Côte d’Azur, OCA); (4) the TACA telescope located at Guitalens, France. Those telescopes produce images with different pixel scales from good spatial resolution with the JAST telescope ($0\farcs58$/pix) up to poorly sampled images like for the TCA telescope ($3\farcs31$/pix). Hence, our CNN has to be robust and flexible enough to keep decent classification performances along these image feature differences. In Table 2, we summarise some important properties of the images produced by these four telescopes.

In the following sections, we briefly describe the transient detection pipelines we used to produce the inputs for O’TRAIN and then, we detail the training data set we built for each telescope.

[8] https://grandma.lal.in2p3.fr
[9] https://www.cefca.es/observatory/description
[10] https://www.iac.es/es/observatorios-de-canarias/observatorio-del-roque-de-los-muchachos

5.1. GMADET pipeline

The GMADET pipeline is publicly available on a dedicated git repository[11]. It takes science-ready images (dark, flat, bias calibrated) as inputs. First, several preprocessing steps are performed: cosmic rays are removed from this science-ready image using either LACOSMIC (van Dokkum 2001) or ASTROSCRAPPY (McCully et al. 2018) python package, the background is estimated using PHOTUTILS (Bradley et al. 2021) routines, the point spread function is estimated using PSFEx (Bertin 2013), finally, the astrometric calibration is computed with SCAMP using the *Gaia* catalog (Bertin 2006). The image is then subtracted using the HOTPANTS code (Becker 2015). Following the procedure described in section 4.2, we simulated N point-like sources for which we stored in an ascii file their positions in the different images. We then ran the GMADET transient detection pipeline on those images. By cross matching the positions of the transient candidates output by GMADET with those of the initial list of simulated point-like sources, we retrieved a vast majority of them. For these ‘True’ candidates, we produced cutouts of 32×32 pixel sizes. The rest of the candidates with a failed cross match in position are directly put in the ‘False’ folder with the same cutout size (32×32 pixels). With GMADET, we usually got much more ‘False’ candidates with respect to the ‘False’ ones. To keep our final training data cube balanced, we randomly picked-up the same number of ‘False’ cutouts than in the ‘True’ folder.

5.2. STDPIPE pipeline

Science-ready images are passed through a custom STDPIPE pipeline (Karpov 2021). The steps implemented in the pipeline includes extraction of objects in the image using SEXTRACTOR, cross-matching them with the Pan-STARRS DR1 catalogue, determining the photometric solution for each frame using Pan-STARRS r magnitude and the $g - r$ colour-for-colour correction, and carrying out an image subtraction using Pan-STARRS r band images acquired through the HIPS2FITS service (Boch et al. 2020) as a template with the HOTPANTS code (Becker 2015). Then the difference images are weighted with the image noise model and transient detection is performed on these weighted images, taking into account the masks from both original images and templates to identify possible difference image artefacts. We simulated a set of artificial sources and injected them into the images before the image subtraction using the position-varying PSF model obtained with the PSFEx code (Bertin 2013). For the spatially coincident (within $1''$) transient candidate output by STDPIPE with our simulated stars, we drew some cutouts (63×63 pixels) centered at the transient candidate position and stored them in the ‘True’ folder. The rest of the transients non spatially coincident with the simulated sources were then pushed into a ‘False’ folder. They mostly consist of either a variety of CCD defects or cases where the image alignment was not good enough to properly eliminate the transients (e.g. due to an imprecise HOTPANTS kernel). The STDPIPE pipeline is publicly available on a dedicated git repository[12].

5.3. Training data sets

Below, we describe the original images and the procedure used to build the data cubes from the four selected telescopes. The final data cube configurations for each training data set are summarised in Table 3.

[11] https://github.com/dcorre/gmadet
[12] https://github.com/karpov-sv/stdpipe

Table 3. Summary of the different data cube configurations used to train our O'TRAIN CNN.

Tel. name	Detection pipeline	Number of candidates	True/False %	Cutouts size (pixels)
JAST	GMADET	13 558	50/50	32 × 32
JAST	STDPIPE	18 328	50/50	32 × 32
FRAM	GMADET	8906	50/50	32 × 32
TACA	GMADET	18 450	50/50	32 × 32
TCA	GMADET	11 414	50/50	32 × 32

Notes. The number of candidates reported is the sum of the labelled 'True' and 'False' candidates stored in the respective data cubes.

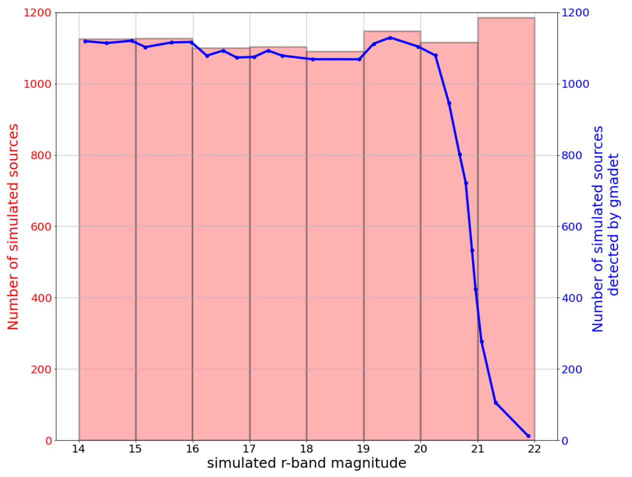


Fig. 4. Numbers of simulated point-like sources injected in the JAST images (red histogram) and retrieved by the GMADET transient detection pipeline (blue curve) as function of their simulated magnitudes. The source fluxes are computed from an arbitrarily chosen zero point magnitude value. The drop observed at magnitude $r \sim 20.5$ in the blue curve shows the point at which the sources are close to the detection threshold used in the GMADET pipeline.

5.3.1. JAST/T80 telescope

We used images taken during the follow-up observations of the O3 GW event S200213t on February 2020 (Blazek et al. 2020; Antier et al. 2020b). After injecting artificial point-like sources in the images using both the GMADET and the STDPIPE transient detection pipelines, we performed searches for transient candidates with the two pipelines in order to populate the 'True' and 'False' folders. Our simulated sources span a wide range of magnitudes that are drawn from an arbitrary zero-point magnitude in order to cover both faint and bright transient source cases. As an example, in Fig. 4, we show the magnitude distribution of the simulate sources retrieved by the GMADET pipeline.

In Fig. 5, we show some examples of the residual cutouts produced by both the GMADET and the STDPIPE pipelines and then stored in the 'True' and 'False' folders. We note that for the training process, when using STDPIPE, we finally cropped the cutouts down to the sizes 32 × 32 pixels in order to compare the classification results with the GMADET cutouts directly. We elaborate this choice in Sect. 8.

5.3.2. FRAM-CTA-N telescope

As a part of its nightly operation routine, the FRAM-CTA-N telesceope (a 25 cm $f/6.3$ telescope located at Observatorio del Roque de los Muchachos, La Palma, Spain, equipped with B, V, R, and z filters and having a $26' \times 26'$ field of view with $1''.52$/pix pixel scale) performs sky survey observations of random locations of the sky. We used the images acquired during these observations as an input for evaluating our transient classifier. To do so, the images from the telescope were pre-processed by the telescope data acquisition and archiving system that handles initial astrometric calibration using the locally installed ASTROMETRY.NET code, as well as basic steps such as dark subtraction, flat-fielding, and masking of cosmic rays. We used the GMADET pipeline to retrieve thousands of artificial point-like sources we have simulated in each image. We then populated the dedicated 'True' and 'False' folders following the method described in Sect. 5.1. In Fig. 6, we show some examples of the cutouts stored in the 'True' and 'False' folders.

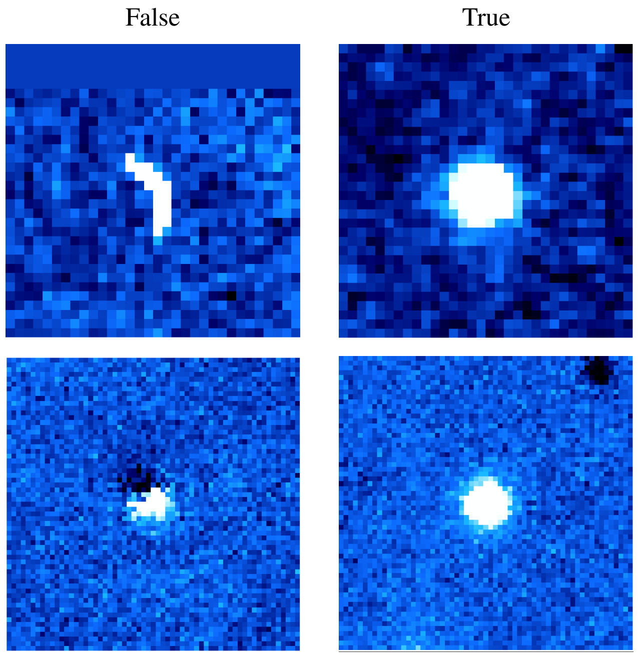


Fig. 5. Some JAST cutouts of bogus (*left column*) and point-like sources candidates (*right column*). The cutouts at the top are produced by the GMADET pipeline (32 × 32 pixels) while the ones at the bottom are produced by the STDPIPE pipeline (63 × 63 pixels).

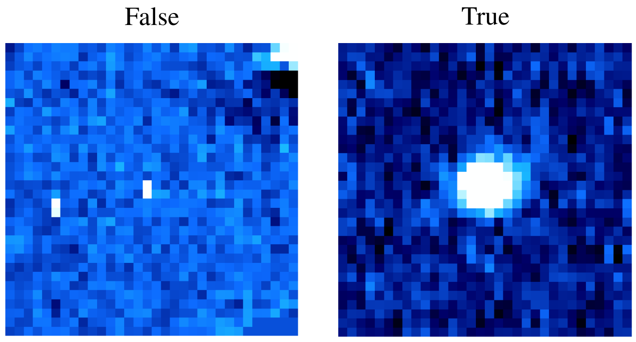


Fig. 6. Some FRAM-CTA-N cutouts of bogus (*left*) and point-like sources candidates (*right*). The cutouts at the top are produced by the GMADET pipeline (32 × 32 pixels).

5.3.3. TACA

We used the TACA images produced during the optical follow-up of the O3 GW event S200114f. Two nights of observations were collected on the January 15–16, 2020 to obtain 36 images

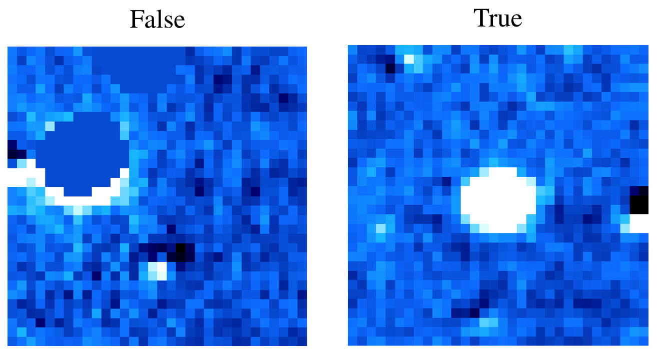


Fig. 7. Some TACA cutouts of bogus (*left*) and point-like sources candidates (*right*). The cutouts are produced by the GMADET pipeline (32 × 32 pixels).

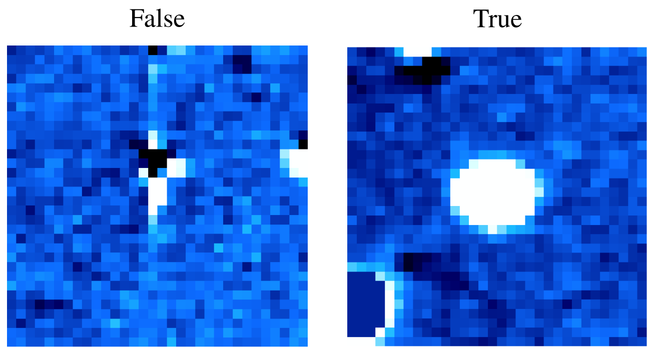


Fig. 8. Some TCA cutouts of bogus (*left*) and point-like sources candidates (*right*). The cutouts are produced by the GMADET pipeline (32 × 32 pixels).

of the sky. Using the GMADET pipeline, we simulated four hundred sources per image from which we built a list of candidates. After cross-matching these candidates with the position of the simulated sources, we were able to build the 'True' and 'False' folder. In Fig. 7, we show some examples of the cutouts stored in these folders.

5.3.4. TCA

The TCA telescope took a significant number of follow-up observations during the O3 LVC campaign for the GRANDMA Collaboration (Antier et al. 2020a,b). We used TCA images from various regions of the sky taken in the first half of the O3 run. We then applied the method described in Sect. 5.1 to simulate thousands of artificial sources in these images using the GMADET pipeline. Finally, we obtained several thousand candidates to populate the data cube of 'True' and 'False' sources as shown in Table 3. In Fig. 8, we show some examples of the cutouts stored in the 'True' and 'False' folders.

6. Classification performance diagnosis tools

6.1. Metrics

Once a residual cutout is analysed by our O'TRAIN CNN, it will deliver, as output, a probability, $\mathrm{p_{class}}$, that the source at the center is a bogus or a real source. Typically, when $\mathrm{p_{class}} = 0$, the CNN model has unambiguously identified a bogus and reciprocally when $\mathrm{p_{class}} = 1$, the CNN model has determined the OT candidate is real. A random guess would lead to $\mathrm{p_{class}} = 0.5$. Actually, a probability threshold, $\mathrm{p_t}$, needs to be defined to make

Table 4. Components of the confusion matrix.

	Predicted Positive (1)	Predicted Negative (0)
Actual positive (1)	True positive (TP)	False negative (FN)
Actual negative (0)	False positive[(a)] (FP)	True negative (TN)

Notes. [(a)]Also known as a 'false alarm'.

a final decision, namely, whether a candidate is tagged as a 'True' and 'False' OT source. As a consequence, we can determine that an OT candidate is well classified if it satisfies the two following configurations:

1. $\mathrm{p_{class}} \geq \mathrm{p_t}$ and the OT candidate was initially labelled as 'True'. We call this candidate a 'True Positive'.
2. $\mathrm{p_{class}} < \mathrm{p_t}$ and the OT candidate was initially labelled as 'False'. We call this candidate a 'True Negative'.

All other possible configurations lead to misclassifications (false positives or false negatives). In order to have a global and the most realistic perspectives of our model's performance, we implemented multiple evaluation metrics and curves. Since the threshold, $\mathrm{p_t}$, affects our metrics, we plotted their values on a 2D curve with a varying threshold. This enables us to see which one works best for each telescope. Firstly, for a binary classification, we may generate a 'confusion matrix' to display the frequency of every combination of predicted classes and actual classes, as shown in Table 4.

The confusion matrix allows to quickly identify pathological classification behaviours of our model especially if the fraction of false positives (FP) or false negatives (FN) is high. We typically do not want to exceed 5% of the total candidates misclassified as FP, while keeping the FN as low as possible. The other implemented metrics help to summarise the confusion matrix and emphasise different aspects of the classification performance: (1) the Accuracy (Acc) corresponding to the percentage of candidates that are accurately classified; (2) the Loss function. In our model, we use a cross entropy loss function with the following formula:

$$\mathrm{loss} = -(y \times \log(\mathrm{p}) + (1 - y) \times \log(1 - \mathrm{p})), \tag{1}$$

where y is a binary indicator of a class and p is the probability given to said class; (3) the Precision (Prec) that calculates the number of real point-like sources well classified by the model amongst the candidates classified as real by the model. A good precision score (near 1) shows that the model is usually right in its predictions of the positive class, marked as 'real sources'; (4) the Recall which calculates how many real transients were well classified in the true transient data set, so a good recall score indicates that the model was able to detect many positive candidates; (5) the $F1$-score that estimates the harmonic mean of recall and precision:

$$\frac{2 \times \mathrm{precision} \times \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}. \tag{2}$$

The F1-score ranges from 0 to 1, the better the performance is, the higher the value of this score; (6) the Matthews correlation coefficient (MCC, Matthews 1975) that takes into account the four parts of the confusion matrix, with the following formula:

$$\frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}}. \quad (3)$$

The MCC score is in essence a measure of correlation between the observed and the predicted binary classification. They would be considered positively associated if most data falls along the diagonal cells in the confusion matrix. In other words, the MCC score is an application of the Pearson correlation coefficient on the confusion matrix. Hence, a MCC score greater than 0.7 corresponds to strong agreement between predictions and observations (Boslaugh 2012; Nettleton 2014; Schober et al. 2018); (7) the ROC curve that displays the true positive rate (TPR) versus the false positive rate (FPR). An ideal model would have a vertical line on $x = \mathrm{FPR} = 0$ and horizontal line for $y = \mathrm{TPR} = 1$. This metric enables us to visualise the global performance; (8) the Recall-Precision curve as with the $F1$-score, it displays the model performance on the positive class (real sources).

The evaluation of the confusion matrix, displayed by the ROC and the recall-precision curves, is clear and easily interpretable, however, it might not be realistic. While the recall-precision curve helps us to compare the model with an always-positive classifier, it fails to include the evaluation on the negative class. On the other hand, the ROC curve leverages the four values in the confusion matrix, but its analysis could be misleading for unbalanced data sets. Even if our global data sets are balanced, we lose this property when, for example, we split the candidates into ranges of magnitude and uncertainty in magnitude.

A more sophisticated solution was proposed by Yamashita et al. (2018) to assess the performance of the model over a varying threshold, based on the MCC and the $F1$-score. The $F1$-MCC curve displays the MCC score versus the $F1$-score for a varying threshold. This allows us to determine the confidence score of the model for every telescope, by looking at the threshold that optimises both scores.

6.2. Optimising the classification threshold

A threshold of $\mathrm{p_t} = 0.5$ is usually the default value and the most intuitive one. But it is not always the most suitable to provide the best classification performances. Therefore, for each telescope data set, we studied the impact of varying thresholds of probability, $\mathrm{p_t}$, on the classification results. We did so in order to choose the $\mathrm{p_t}$ that will finally maximise the classification performances. We based our choice of the best $\mathrm{p_t}$ on the ROC, the precision-recall and the $F1$-MCC curves for which we plotted TPR versus FPR, the precision versus the recall and the $F1$-score versus the MCC, respectively, for a varying threshold, $\mathrm{p_t}$. Therefore, the best $\mathrm{p_t}$ is the one that will jointly maximise the score of these three chosen metrics. With regard to the ROC curve, we selected a value that maximises the g-mean score:

$$g_{\mathrm{mean}} = \sqrt{\mathrm{TPR} \times (1 - \mathrm{FPR})}. \quad (4)$$

For the precision-recall curve, we try to maximise the $F1$-score. The $F1$-MCC curve shows the threshold, $\mathrm{p_t}$, at which we maximise both the $F1$-Score and the MCC by maximising the sum of these two metrics. The value of the best threshold differs from one telescope to another and helps us to get a better perspective of the CNN model's performance (as explained in further detail in Sect. 7).

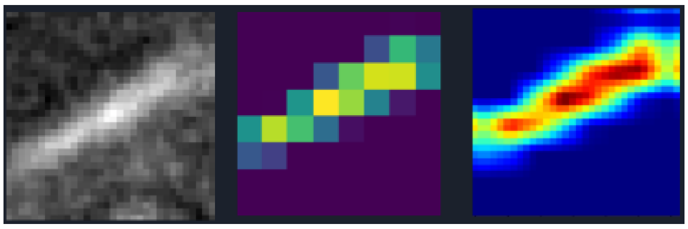


Fig. 9. Illustration the Grad-CAM function that helps identifying the regions of interest in the image that triggered the O'TRAIN classification. *Left*: residual cutout (32×32 pixels) of a bogus identified in the JAST telescope images. *Center*: the GRAD-CAM Heatmap (8×8 pixels) showing the final feature maps $\times$ their importance where our O'TRAIN CNN focused on to take its decision. *Right*: rescaled Grad-CAM heat map (32×32 pixels) for a direct comparison with the cutout.

6.3. Issues of overfitting or underfitting

Providing the model with just enough data to reasonably train it is crucial. With a small data set, the model does not have enough information to make the necessary distinction between a true transient and a bogus one. The ideal case would be to have a large data set of several tens of thousands of candidates with no contamination, along with the computational resources to be able to train a model on a data set of this size. The model should be at least complex enough to create good non-linear connection between these characteristics, but not too complex, so that we may avoid creating connections that are not relevant to the classification and that are only present in the training data set. When the model is not trained enough, we call it an underfitting, and the performance is inefficient in terms of both training and validation data sets. If the model, on the other hand, is learning too much and getting trivial information, we call it an overfitting; in this case, the performance in the validation data set is considerably worse than it is on the training data set. We can estimate such an over(under)fitting behaviour of our CNN thanks to two metrics: loss and accuracy. We can track these two values throughout the training process (i.e. the epochs) to see if at some point we notice the beginning of a divergence between these values on the training and the validation data sets. If so, we should configure the model so that it stops the training at that moment.

6.4. Understanding the model's decision with the gradient class activation map (Grad-CAM)

The output of a CNN and its classification decisions are sometimes not easy to understand or interpret. We used gradient-weighted class activation mapping (i.e. Grad-CAM, Selvaraju et al. 2020) to track the regions that were considered as the most important ones by the model to make its decision. The idea here is to take the last feature maps (right before flattening them) and multiply each one of them with its corresponding importance in the classification by back propagating the derivative of the loss with respect to it. We add them up and resize this output to the size of the input images in order to overlay them. The Grad-CAM is therefore a powerful diagnosis tool for the O'TRAIN users to understand what has triggered the CNN classification decision. In Fig. 9, we show an example of the Grad-CAM output tested on one bogus cutout from the JAST telescope. In these images, we can clearly see that the CNN model focuses on the correct region of the residual cutout to make the right decision, namely, classifying a non-point-like source as a bogus transient.

Table 5. Results of the different metrics used to evaluate the classification performances of our CNN models trained on 5 telescope data cubes.

Telescope name	Detection pipeline	p_t [0-1]	Acc [0-1]	Loss [0-1]	Prec [0-1]	Recall [0-1]	$F1$-score [0-1]	MCC [0-1]	Confusion Matrix
JAST	GMADET	0.35	0.98	0.06	0.98	0.99	0.99	0.97	$\begin{bmatrix} 0.49 & 0.01 \\ \leq 0.01 & 0.49 \end{bmatrix}$
JAST	STDPIPE	0.51	0.95	0.10	0.94	0.97	0.96	0.91	$\begin{bmatrix} 0.46 & 0.03 \\ 0.02 & 0.5 \end{bmatrix}$
FRAM-CTA-N	GMADET	0.55	0.93	0.30	0.89	0.97	0.93	0.85	$\begin{bmatrix} 0.43 & 0.06 \\ 0.02 & 0.49 \end{bmatrix}$
TACA	GMADET	0.52	0.93	0.20	0.91	0.96	0.95	0.86	$\begin{bmatrix} 0.43 & 0.06 \\ 0.01 & 0.50 \end{bmatrix}$
TCA	GMADET	0.34	0.94	0.2	0.92	0.97	0.95	0.89	$\begin{bmatrix} 0.45 & 0.04 \\ 0.01 & 0.49 \end{bmatrix}$

Notes. The values of the metrics have been computed based on the best probability threshold, p_t. p_t has been chosen to maximise the $F1$-MCC scores and minimise the false alarm rate (i.e. false positives).

7. Results and classification performance

Given the diagnosis tools described above, we were able to evaluate the classification performance of our CNN models trained on the five training data sets described in Table 3. We describe our results in the following sections.

7.1. O'TRAIN classification performance

We applied the aforementioned metrics and curves to track the performance of the model, in both the 'True' and 'False' classes and for a varying threshold and ranges of magnitude. This allows us to get a very general and global perspective of the performances. Following the method described in Sect. 6.2, we selected the best threshold, p_t, for each telescope training set. Then we calculated the accuracy and loss of the last epoch on the validation data set, the precision, recall, $F1$-score, and MCC, as well as the confusion matrix with percentages of each part among the validation set. The results are summarised in Table 5.

We paid particular attention to the MCC score since it gathers all parts of the classification (or the confusion matrix) and is more robust than other metrics. We want it to be greater than 0.7 to ensure we have a strong correlation between the O'TRAIN predicted classes and the labelled true ones, see Sect. 6.1. For all of our training data cubes, the MCC scores are greater than 0.84 and, therefore, they satisfied our scientific requirements. We also noticed that all the FP are kept below 2% which is again perfectly in agreement with our scientific requirements (FP $\leq$ 5%).

As an example, we show in Fig. 10, a collection of different diagnosis curves we produced after the training of O'TRAIN on the JAST cutouts created with the GMADET pipeline. First, we estimated the best probability threshold, p_t to clarify the CNN decision. Once this was fixed, we used this value (p_t = 0.35) to derive all other metrics values. On the one hand, we can see that the model unambiguously classifies the candidates, namely ~50% 'False' with p_{class} = 0 and ~50% 'True' with p_{class} = 1. Such a drastic classification behaviour shows that the model is well designed for making these binary classifications and the training data cube is correctly constructed with a high purity of both 'True' and 'False' candidates. The ROC curve is near the ideal case for this telescope and for all ranges of magnitude. These near ideal diagnosis curves associated with the metrics values listed in Table 5 allow us to validate the performance of our CNN model. By checking the accuracy and the loss metrics, we note that after ten epochs, the O'TRAIN model converges towards high accuracies (Acc > 0.98) and small losses (loss < 0.1). Despite a little deviation in the loss metric between the training and the validation data sets, we did not detect any significant over(under)fitting as we will show in the following section. In Appendix A, we show the same diagnosis curves for the other telescope cases we studied in this work. They all show very good classification performances in agreement with our scientific requirements.

7.2. Evaluating the over(under)fitting behaviour of the trained CNN model

As explained in Sect. 6.3, an important parameter that ensures the classification performance will be reproducible in real observational conditions (with a data sets never seen by our CNN model) is the over(under)fitting parameter. As we trained our CNN on relatively small data set sizes, we might be sensitive to over(under)fitting behaviours. To measure it from each of our five data cubes, we computed $\Delta Acc = Acc_{train} - Acc_{val}$ as function of different data set sizes by gradually changing the sizes of the 'True' and 'False' folders used for the training. As shown in Fig. 11, we found that with training data set sizes smaller than 5000 candidates including both 'True' and 'False' ones, the learning process leads to unavoidable over- or under-fittings with $|\Delta Acc| > 2\%$. For larger data sets, the architecture of our O'TRAIN CNN is finally well adapted for being trained on such a binary classification task as it converges towards $\Delta Acc \sim 0$. In a complementary way, we also show in Fig. 11 the evolution of the accuracy, the F1-score and the MCC metrics as functions of the data set size for the JAST-GMADET data cube only. We also see that the metrics converge towards high scores once the data set size used for the training is larger than 5000 cutouts.

These results allow us to conclude that our CNN model will have reproducible robust classification results while being trained on a relatively small amount of labelled cutouts (typically 10 k of cutouts).

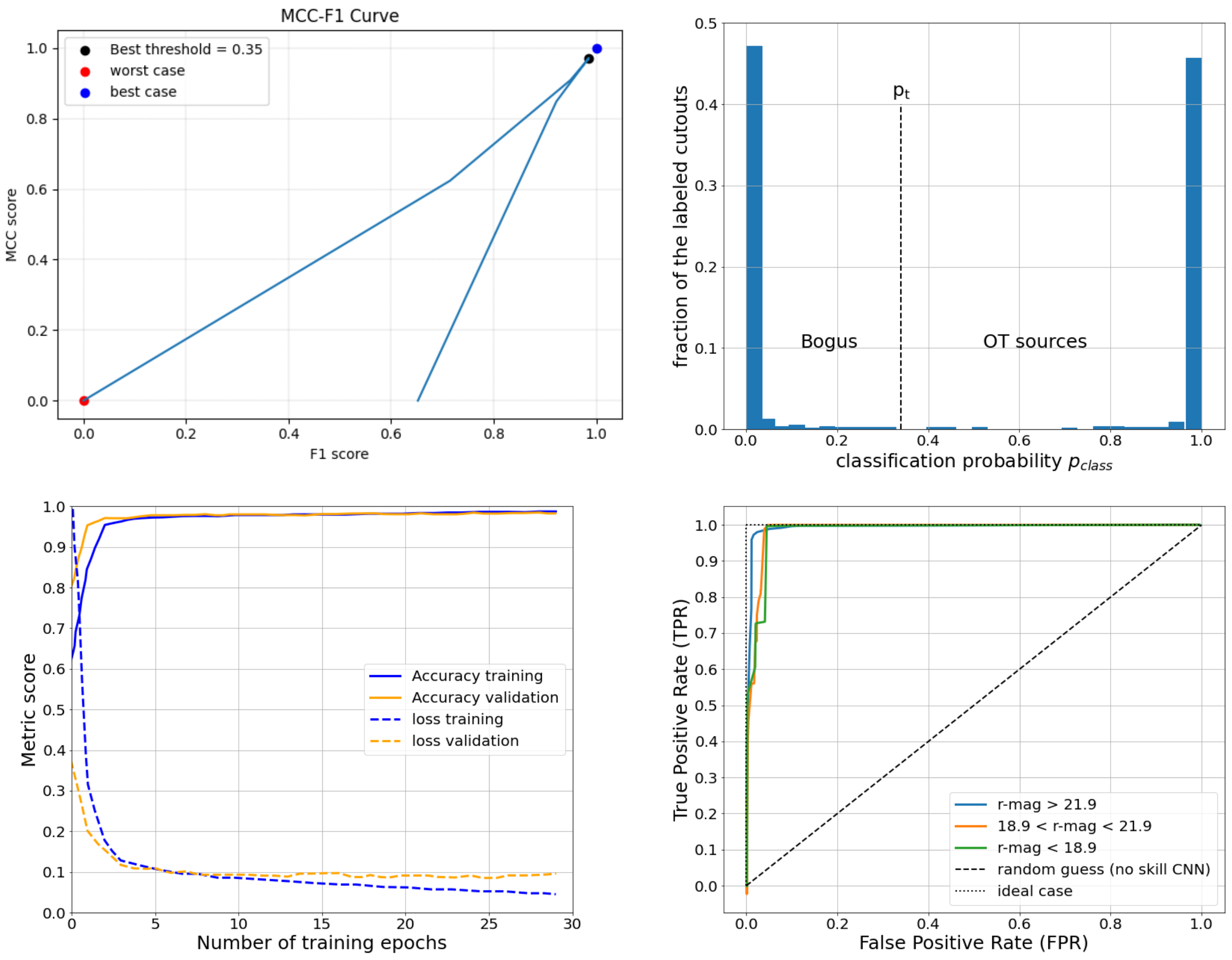


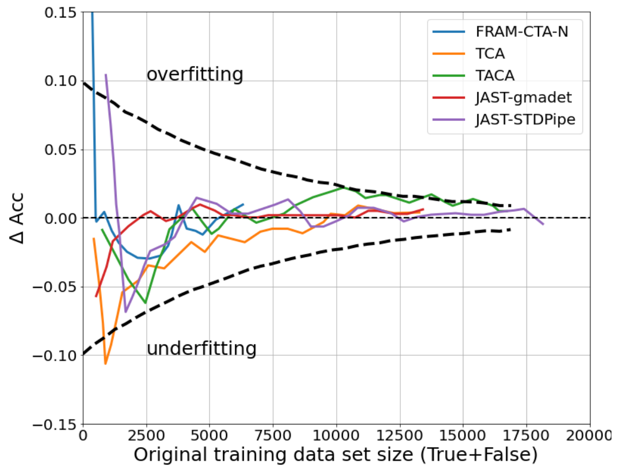
Fig. 10. Different visualisation of the metrics used to evaluate the performance of the O'TRAIN model trained on the JAST-GMADET data cube. *Top left*: evolution of the MCC and $F1$-score as a function of different values of p_t. The value $p_t = 0.35$ maximises both the MCC and $F1$-score, our two main diagnosis to evaluate the CNN classification performances. *Top right*: classification probability distribution of the validation data set at the last epoch of training. *Bottom left*: evolution of the accuracy and loss metrics as a function of the number of training epochs. *Bottom right*: ROC curve per bin of the OT candidate magnitudes in order to diagnose the behaviour of the O'TRAIN model for different source brightness.

8. Discussion and perspectives

In this work, we demonstrate that we are able to obtain very good RB classification results with our algorithm. However, there are still some limitations involved, which we discuss in this section.

8.1. Size of the cutouts used to train O'TRAIN

The cutouts contain the information about the OT candidate signal at the center of the image, the noise model, and (possibly) the signal from other sources located elsewhere in the image. As the CNN model analyses all those features in the cutouts, large cutouts may result in confusing the classification decision as more features have to be extracted and weighted. This is due to the fact that the size of the convolutional filter kernels may not be well adapted to extract numerous and large features in the cutouts. Therefore, we decided to study the impact of the cutout sizes in our CNN decisions. To do so, we used the cutouts ('True' and 'False') produced by the STDPIPE pipeline in 63×63 pixels from the JAST telescope original images. We built several data cubes with the cutout sizes varying from 15×15 pixels to 63×63 pixels while still keeping the OT candidates at the center. We show in Fig. 12 that the maximum acceptable size for these cutouts to maintain a high level of classification performance is 48×48 pixels. Above this cutout size, the model behaves randomly and ends up erroneously classifying the candidates. We attribute this to a structural limitation of our CNN algorithm which employs small kernel filters (3×3 pixels) in the convolutional layers. These kernels might therefore not be adapted to catch features in cutouts larger than 48 pixels. subsection Possible ways to improve the classification Currently, our CNN algorithm performs its classification process on the residual cutouts provided by the subtraction of science and reference images. While we obtained just a few false positives, we ended with a non-negligible number of false negatives. In other words, we optimised O'TRAIN to keep the false alarm rate as low as possible with the cost of losing real optical transient sources (conservative approach). However, additional informations on each candidate could be used to reduce the FN and, hence, improve the classification performances. Gieseke et al. (2017) explored this approach by considering three cases for their CNN training: (1) The CNN predictions are based on the analysis of the science, the reference and the residual cutouts; (2) the CNN predictions are based on the analysis of the science and the reference cutouts; 3) the CNN predictions are based on the analysis of the residual cutouts only.



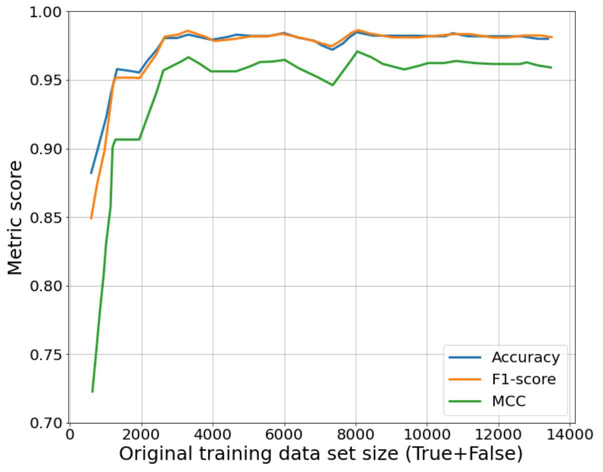


Fig. 11. *Top*: Evolution of ΔAcc between the batch and the validation data sets as function of the size of the initial training data set after 25 epochs. This visualisation allows to quickly identify if the CNN algorithm is in an over(under)fitting regime (outliers of the dashed lines). *Bottom*: Evolution of three metrics: accuracy, $F1$-score and MCC as a function of the training data set size (JAST-GMADET).

According to these authors, the training data sets based on the first and second configurations described above give similarly good classification performances. On the contrary, their CNN model gave slightly worse results when it only analysed the residual cutouts. An interesting perspective is highlighted here, namely, the possibility of discarding the image subtraction process (that can produce plenty of artefacts) for RB classification purposes. In our approach, we only used the standard residual images output from the GMADET and STDPIPE pipelines which already provides decent classification results. In order to give more flexibility and scientific perspectives to O'TRAIN, an update allowing the users to give different types of cutouts as a data cube input (science-ready, reference, residual, mask template images) is foreseen.

8.2. Towards more sophisticated methods: FASTER-R and MASK-R CNN

Thanks to the Grad-CAM tools included in O'TRAIN, we noticed that some OT candidates were misclassified due to the presence of additional sources in the cutouts (see Fig. 13).

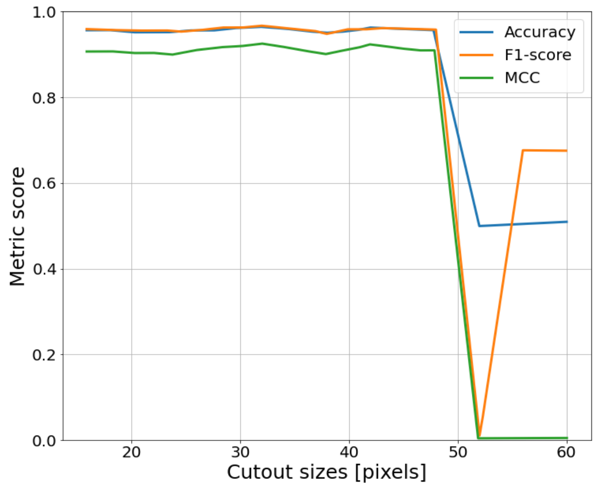


Fig. 12. Evolution of the accuracy, $F1$-score, and MCC metric scores as a function of the cutout sizes given as input to O'TRAIN. The architecture of our O'TRAIN model applied to the JAST-STDPIPE is well adapted for a RB classification if the cutouts sizes do not exceed ~50 pixels.

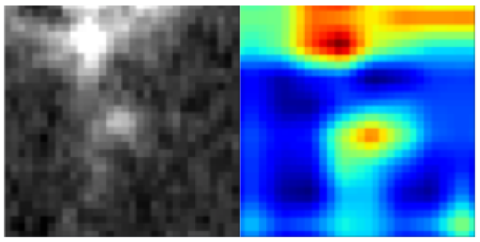


Fig. 13. Example of a misclassified OT (FN) by O'TRAIN. While the CNN focuses a fraction of its attention into the central part of the cutouts (where the OT is), most of its classification decision relies on the presence of a brighter source and a non-uniform background noise at the top of the cutouts.

This kind of misclassification might be solved by the addition of more information in the data cube as discussed above. It could also be completely avoided by using novel techniques to analyse the astronomical images by deep convolutional neural networks such as: FASTER-R (Ren et al. 2015) and/or MASK-R CNN (He et al. 2017) algorithms. Whilst both methods consists of finding regions in the image that contains the object (i.e. the OT) that we are looking for, the FASTER-R CNN will output bounding boxes containing these objects, whereas the MASK-R-CNN goes one step further and gives us the exact pixels of the targeted objects.

For the training, the FASTER-R-CNN, as explained by Ren et al. (2015), takes on a data set of images with information such as the coordinates of the bounding boxes containing sources of the image. It applies a CNN model to extract features, and passes them to another model called the 'region proposal network' (RPN), which performs a regression on the coordinates, and then on to a classifier to predict the class of the detected objects. Like all training processes, the model tries to align its predictions with the ground truth. In this case, it places a sliding window (anchors) that could vary in size and proportions and then calculates the score of Intersection over the Union (IoU) between its predicted boxes and the ones in the data set we had given it.

The RPN outputs several proposals, called regions of interest (ROI), for all objects detected in the image. Since these regions will not have the same size, a pooling (i.e. ROI Pool) will be applied before passing it through a classifier as explained in Girshick (2015). The coordinates in the data set, and therefore the predicted coordinates, are on the original image scale. However, the feature maps were decreased a few times from the original image. To adapt the coordinates, the ROI Pool divides them by the ratio of the size of the original image and that of the feature maps (in our case, this ratio would be $n = 8$) and takes the integer part:

$$new\ coordinates = [original/n]. \tag{5}$$

The ROI Pool also transforms the regions into a fixed sized output in order to give it to the classifier. Given a shape fixed by the user, the ROI Pool will divide each region of interest into a grid of this shape, each bin of the grid might contain just a few pixels. The ROI Pool will take the maximum value of each cell and produce a matrix of the desired shape.

The MASK-R CNN outperforms the FASTER-R-CNN as it adds a branch to this structure for a binary mask, showing whether the pixel is part of the object or not, and thus performing a pixel-level detection of objects. He et al. (2017) noticed a misalignment between the ROIs after the ROI Pool layer and the regions in the original image, which could cause a drop in the model's performance since the mask requires precision at a pixel level. These authors adjusted this part by using ROIAlign instead of ROIPool (Dai et al. 2015; He et al. 2017), where they did not take the integer part in Eq. (5); this outputs coordinates as float values, so we do not have the exact pixels. The region in ROI Align is still divided into a grid. In each bin of this grid, we regularly put four sample points and for each point, we kept the coordinates of the middle cell of the nearest four pixels and the values of these pixels, in its vicinity (i.e. if the sample point is on the top-left of the bin, we looked for the nearest pixels to this point on the top-left of the bin). Using the bilinear interpolation between these coordinates and the values, we were able to get a representative point. Finally, we obtained four representative points for the specific bin and we kept their maximum values.

This model could be well adapted to our RB problem, since some astronomical tools, such as SExtractor, exist to create a segmentation map of the optical sources present in the image and an offset of only two pixels can greatly affect the performances. Implementing these new classification methods is beyond the scope of this paper, however, further development will be employed to include them in the O'TRAIN framework in the future.

9. Conclusions

Real and bogus classifiers are now standards in the pipelines aiming at detecting a large number of transient phenomena at optical wavelengths. We have developed a robust and flexible convolutional neural network model called O'TRAIN. It aims at distinguishing real point-like sources from various types of bogus in optical images. We have shown that O'TRAIN reaches high classification performances for a wide range of telescope pixel scales and observational conditions. In addition, we demonstrated the capabilities of our CNN model to behave robustly against various types of inputs (cutout and data set sizes, residuals images from different subtraction methods, etc.) provided by two publicly available transient detection pipelines (gmadet and STDPipe). Indeed, on the five training data cubes, we obtained very good classification performances with MCC scores ranging in the interval [0.84 - 0.99]. Such performances can be obtained with a relatively small size of the training data sets, typically on the level of a few tens of thousands of labelled cutouts, without creating any over- or underfitting during the training epochs. To guide users, we have built a complete user-friendly and easy-to-use frame work to launch training as well as to check diagnostic tools monitoring the performance of the CNN model. This will greatly help observers who would like to use such a machine learning technique in their own pipeline even if they are not fully experts on the method. Overall, O'TRAIN is a publicly available code and aims to be upgradeable in the future, with new features that would enhance the flexibility of the code and the classification perspectives such as the simultaneous classification of multiple sources at once.

Acknowledgements. D.T. is funded by the CNES Postdoctoral Fellowship program at the CEA-Saclay/AIM/Irfu laboratory. S.K. acknowledges support from the European Structural and Investment Fund and the Czech Ministry of Education, Youth and Sports (Project CoGraDS – CZ.02.1.01/0.0/0.0/15_003/0000437). FRAM-CTA-N operation is supported by the Czech Ministry of Education, Youth and Sports (projects LM2015046, LM2018105, LTT17006) and by European Structural and Investment Fund and the Czech Ministry of Education, Youth and Sports (projects CZ.02.1.01/0.0/0.0/16_013/0001403 and CZ.02.1.01/0.0/0.0/18_046/0016007). D.A.K. acknowledges support from Spanish National Research Project RTI2018-098104-J-I00 (GRBPhot). This research has made use of the VizieR catalogue access tool, CDS, Strasbourg, France. We also want to thank Pascal Yim, professor at Ecole Centrale Lille, for his inputs on Faster-R-CNN and Mask-R-CNN models. The authors warmly thank the GRANDMA Collaboration for providing images taken from their optical follow-up of gravitational wave sources during the O3 run.

References

Agayeva, S., Alishov, S., Antier, S., et al. 2021, in Revista Mexicana de Astronomia y Astrofisica Conference Series, 53, 198
Antier, S., Agayeva, S., Aivazyan, V., et al. 2020a, MNRAS, 492, 3904
Antier, S., Agayeva, S., Almualla, M., et al. 2020b, MNRAS, 497, 5518
Barbary, K. 2016, J. Open Source Softw., 1, 58
Becker, A. 2015, HOTPANTS: High Order Transform of PSF and Template Subtraction, Astrophysics Source Code Library, [record ascl:1504.004]
Bellm, E. C., Kulkarni, S. R., Barlow, T., et al. 2019, PASP, 131, 068003
Bertin, E. 2006, in Astronomical Society of the Pacific Conference Series, Astronomical Data Analysis Software and Systems XV, eds. C. Gabriel, C. Arviset, D. Ponz, & S. Enrique, 351, 112
Bertin, E. 2009, Mem. Soc. Astron. Italiana, 80, 422
Bertin, E. 2011, in Astronomical Society of the Pacific Conference Series, Astronomical Data Analysis Software and Systems XX, eds. I. N. Evans, A. Accomazzi, D. J. Mink, & A. H. Rots, 442, 435
Bertin, E. 2013, PSFEx: Point Spread Function Extractor
Bertin, E., & Arnouts, S. 1996, A&AS, 117, 393
Blazek, M., Agayeva, S., Baransky, A., et al. 2020, GRB Coordinates Network, 27116, 1
Boch, T., Pineau, F., & Derriere, S. 2012, in Astronomical Society of the Pacific Conference Series, Astronomical Data Analysis Software and Systems XXI, eds. P. Ballester, D. Egret, & N. P. F. Lorente, 461, 291
Boch, T., Fernique, P., Bonnarel, F., et al. 2020, in Astronomical Society of the Pacific Conference Series, eds. R. Pizzo, E. R. Deul, J. D. Mol, J. de Plaa, & H. Verkouter, 527, 121
Boslaugh, S. 2012, Statistics in a Nutshell (O'Reilly Media, Inc.)
Bradley, L., Sipőcz, B., Robitaille, T., et al. 2021, https://doi.org/10.5281/zenodo.5525286
Bright, J. S., Margutti, R., Matthews, D., et al. 2022, ApJ, 926, 112
Burhanudin, U. F., Maund, J. R., Killestein, T., et al. 2021, MNRAS, 505, 4345
Carrasco-Davis, R., Cabrera-Vives, G., Förster, F., et al. 2019, PASP, 131, 108006
Carrasco-Davis, R., Reyes, E., Valenzuela, C., et al. 2021, AJ, 162, 231
Chambers, K. C., Magnier, E. A., Metcalfe, N., et al. 2016, ArXiv e-prints, [arXiv:1612.05560]
Coppejans, D. L., Margutti, R., Terreran, G., et al. 2020, ApJ, 895, L23
Dai, J., He, K., & Sun, J. 2015, Instance-aware Semantic Segmentation via Multi-task Network Cascades
Duev, D. A., Mahabal, A., Masci, F. J., et al. 2019, MNRAS, 489, 3582
Dyer, M. J., Steeghs, D., Galloway, D. K., et al. 2020, SPIE Conf. Ser., 11445, 114457G
Gal-Yam, A. 2012, Science, 337, 927

Gal-Yam, A. 2019, ARA&A, 57, 305
Gieseke, F., Bloemen, S., van den Bogaard, C., et al. 2017, MNRAS, 472, 3101
Girshick, R. 2015, ArXiv e-prints, [arXiv:1504.08083]
Graham, M. J., Kulkarni, S. R., Bellm, E. C., et al. 2019, PASP, 131, 078001
Groot, P., Bloemen, S., & Jonker, P. 2019, in The La Silla Observatory - From the Inauguration to the Future, 33
Han, X., Xiao, Y., Zhang, P., et al. 2021, PASP, 133, 065001
He, K., Gkioxari, G., Dollár, P., & Girshick, R. 2017, ArXiv e-prints, [arXiv:1703.06870]
Ho, A. Y. Q., Perley, D. A., Kulkarni, S. R., et al. 2020, ApJ, 895, 49
Ho, A. Y. Q., Margalit, B., Bremer, M., et al. 2021, ApJ, 932, 2
Hosenie, Z., Bloemen, S., Groot, P., et al. 2021, Exp. Astron. 51, 319
Ivezić, Ž., Kahn, S. M., Tyson, J. A., et al. 2019, ApJ, 873, 111
Karpov, S. 2021, STDPipe: Simple Transient Detection Pipeline, Astrophysics Source Code Library, [record ascl:2112.006]
Killestein, T. L., Lyman, J., Steeghs, D., et al. 2021, MNRAS, 503, 4838
Masci, F. J., Laher, R. R., Rusholme, B., et al. 2019, PASP, 131, 018003
Matthews, B. 1975, Biochim. Biophys. Acta (BBA) - Protein Struct., 405, 442
McCully, C., Crawford, S., Kovacs, G., et al. 2018, https://doi.org/10.5281/zenodo.1482019
Möller, A., & de Boissière, T. 2020, MNRAS, 491, 4277
Nettleton, D. 2014, Bibliography (Boston: Morgan Kaufmann), 279
Perley, D. A., Ho, A. Y. Q., Yao, Y., et al. 2021, MNRAS, 508, 5138
Pineau, F.-X., Boch, T., Derrière, S., & Schaaff, A. 2020, in Astronomical Society of the Pacific Conference Series, Astronomical Data Analysis Software and Systems XXVII, eds. P. Ballester, J. Ibsen, M. Solar, & K. Shortridge, 522, 125
Prentice, S. J., Maguire, K., Smartt, S. J., et al. 2018, ApJ, 865, L3
Quimby, R. M., Kulkarni, S. R., Kasliwal, M. M., et al. 2011, Nature, 474, 487
Ren, S., He, K., Girshick, R., & Sun, J. 2015, ArXiv e-prints, [arXiv:1506.01497]
Schober, P., Boer, C., & Schwarte, L. 2018, Anesth. Analg., 126, 1
Selvaraju, R. R., Cogswell, M., Das, A., et al. 2020, Int J Comput Vis, 128, 336
Skrutskie, M. F., Cutri, R. M., Stiening, R., et al. 2006, AJ, 131, 1163
Smartt, S. J., Clark, P., Smith, K. W., et al. 2018, The Astronomer's Telegram, 11727, 1
Tonry, J. L., Denneau, L., Heinze, A. N., et al. 2018, PASP, 130, 064505
Turpin, D., Ganet, M., Antier, S., et al. 2020, MNRAS, 497, 2641
van Dokkum, P. G. 2001, PASP, 113, 1420
Wang, J., Xin, L. P., Li, H. L., et al. 2021, ApJ, 916, 92
Xin, L. P., Li, H. L., Wang, J., et al. 2021, ApJ, 909, 106
Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. 2018, Insights Imaging, 9, 611

Appendix A: O'TRAIN diagnosis curves for JAST-STDPIPE, TACA, FRAM, and TCA images

JAST-STDPipe post-training diagnosis curves

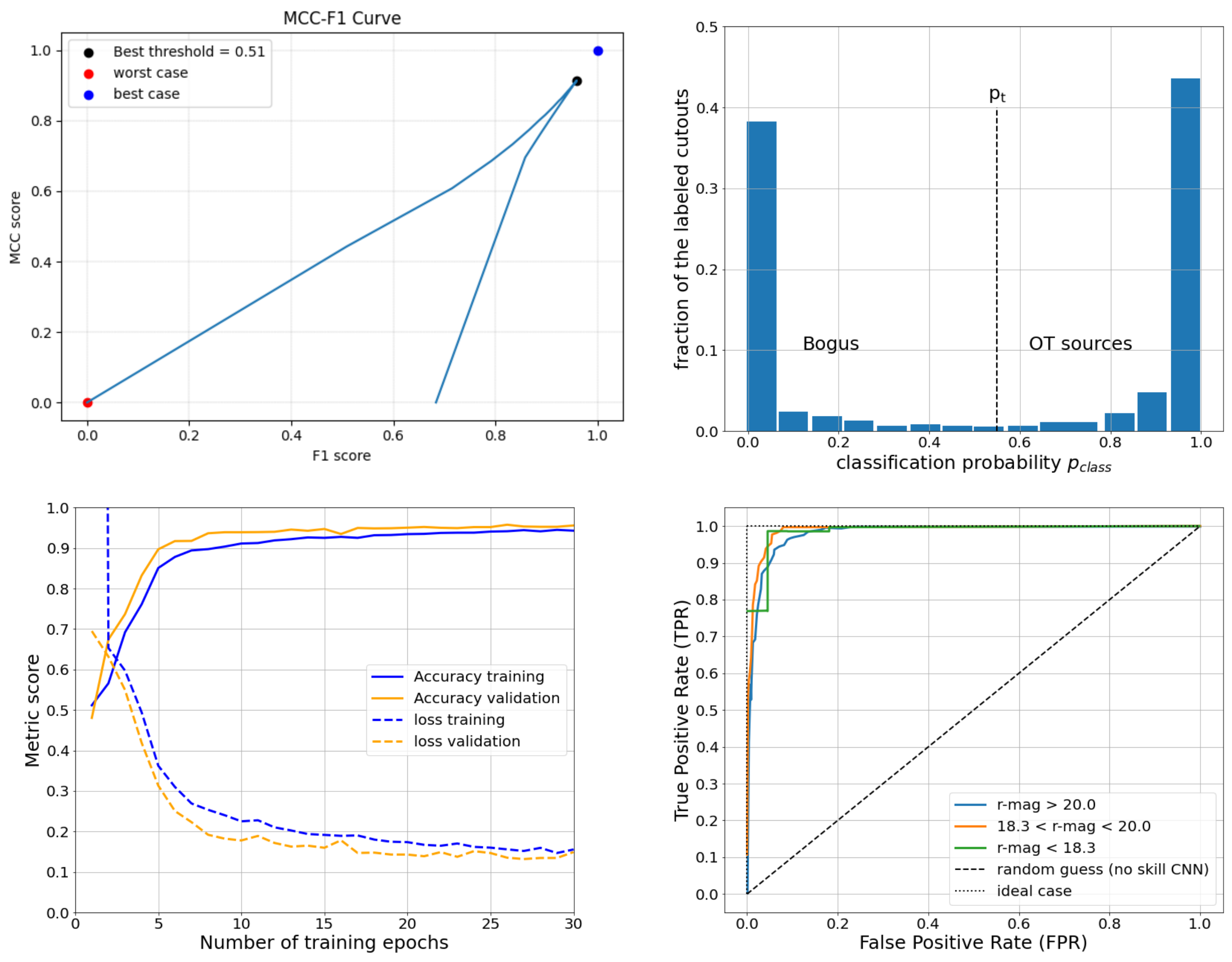


Fig. A.1. Different visualisations of the metrics used to evaluate the performance of the O'TRAIN model trained on the JAST-STDPIPE data cube. Top left: Evolution of the MCC and F1-score as a function of different values of p_t. The value $p_t = 0.51$ (best threshold) maximises both the MCC and F1-score. Top right: Classification probability distribution of the validation data set at the last epoch of the training. Bottom left: Evolution of the accuracy and loss metrics as a function of the number of training epochs. Bottom right:) ROC curve per bin of the OT candidate magnitudes in order to diagnose the behaviour of the O'TRAIN model for different source brightness.

TACA post-training diagnosis curves

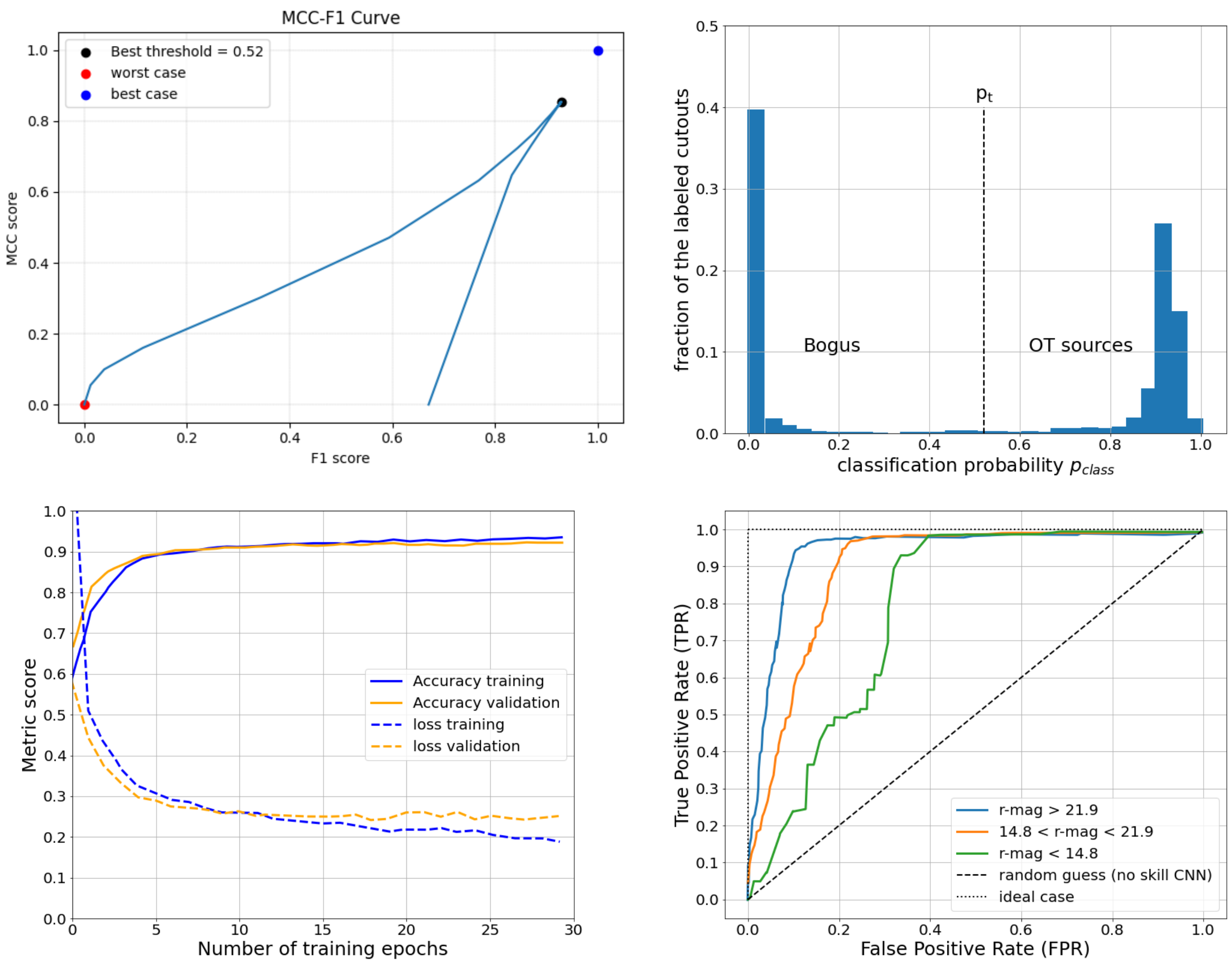


Fig. A.2. Different visualisations of the metrics used to evaluate the performance of the O'TRAIN model trained on the TACA-GMADET data cube. Top left: Evolution of the MCC and F1-score as a function of different values of p_t. The value $p_t = 0.52$ (best threshold) maximises both the MCC and F1-score. Top right: Classification probability distribution of the validation data set at the last epoch of the training. Bottom left: Evolution of the accuracy and loss metrics as a function of the number of training epochs. Bottom right: ROC curve per bin of the OT candidate magnitudes in order to diagnose the behaviour of the O'TRAIN model for different source brightness.

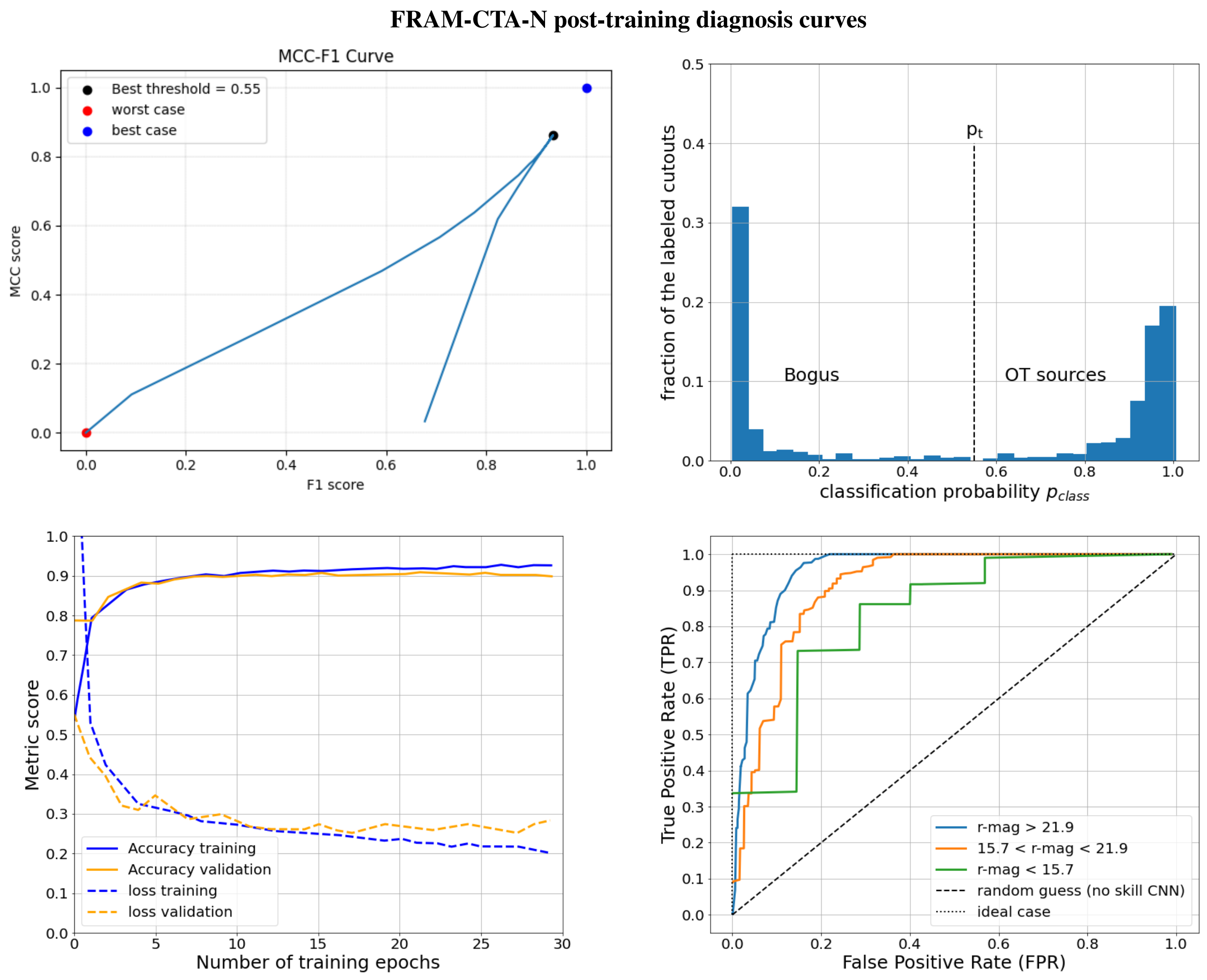


Fig. A.3. Different visualisations of the metrics used to evaluate the performance of the O'TRAIN model trained on the FRAM-CTA-N-GMADET data cube. Top left: Evolution of the MCC and F1-score as a function of different values of p_t. The value $p_t = 0.55$ (best threshold) maximises both the MCC and F1-score. Top right: Classification probability distribution of the validation data set at the last epoch of the training. Bottom left: Evolution of the accuracy and loss metrics as a function of the number of training epochs. Bottom right: ROC curve per bin of the OT candidate magnitudes in order to diagnose the behaviour of the O'TRAIN model for different source brightness.

TCA post-training diagnosis curves

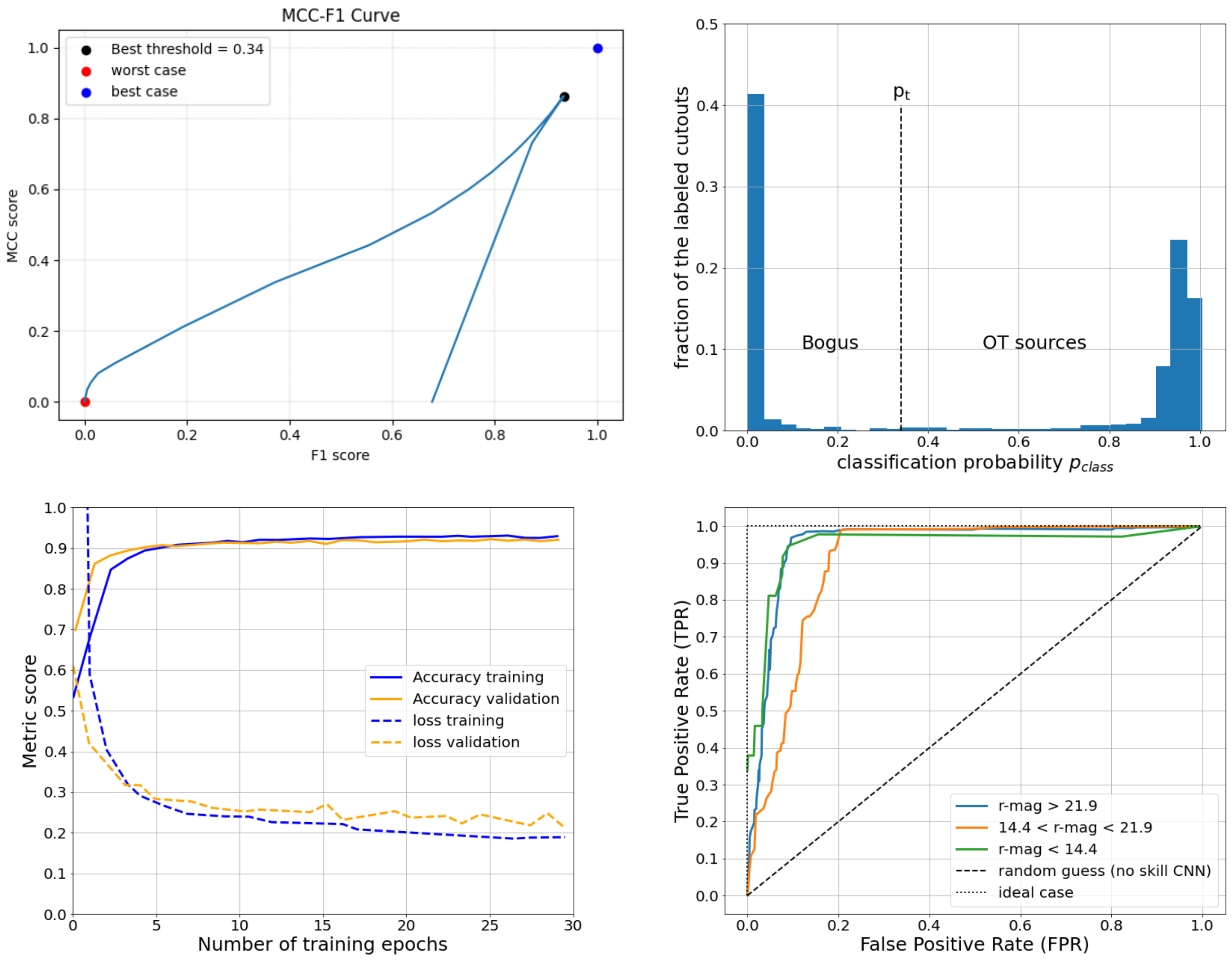


Fig. A.4. Different visualisations of the metrics used to evaluate the performance of the O'TRAIN model trained on the TCA-GMADET data cube. Top left: Evolution of the MCC and F1-score as a function of different values of p_t. The value $p_t = 0.34$ (best threshold) maximises both the MCC and F1-score. Top right: Classification probability distribution of the validation data set at the last epoch of the training. Bottom left: Evolution of the accuracy and loss metrics as a function of the number of training epochs. Bottom right: ROC curve per bin of the OT candidate magnitudes in order to diagnose the behaviour of the O'TRAIN model for different source brightness.