



HAL
open science

MaxiMask and MaxiTrack: Two new tools for identifying contaminants in astronomical images using convolutional neural networks

M. Paillassa, E. Bertin, H. Bouy

► **To cite this version:**

M. Paillassa, E. Bertin, H. Bouy. MaxiMask and MaxiTrack: Two new tools for identifying contaminants in astronomical images using convolutional neural networks. *Astronomy and Astrophysics - A&A*, 2020, 634, pp.A48. 10.1051/0004-6361/201936345 . hal-02492851

HAL Id: hal-02492851

<https://hal.sorbonne-universite.fr/hal-02492851v1>

Submitted on 27 Feb 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

MAXIMASK and MAXITRACK: Two new tools for identifying contaminants in astronomical images using convolutional neural networks

M. Paillassa¹, E. Bertin², and H. Bouy¹

¹ Laboratoire d'astrophysique de Bordeaux, Univ. Bordeaux, CNRS, B18N, allée Geoffrey Saint-Hilaire, 33615 Pessac, France
e-mail: maxime.paillassa@u-bordeaux.fr

² Sorbonne Université, CNRS, UMR 7095, Institut d'Astrophysique de Paris, 98 bis bd Arago, 75014 Paris, France

Received 18 July 2019 / Accepted 9 December 2019

ABSTRACT

In this work, we propose two convolutional neural network classifiers for detecting contaminants in astronomical images. Once trained, our classifiers are able to identify various contaminants, such as cosmic rays, hot and bad pixels, persistence effects, satellite or plane trails, residual fringe patterns, nebulous features, saturated pixels, diffraction spikes, and tracking errors in images. They encompass a broad range of ambient conditions, such as seeing, image sampling, detector type, optics, and stellar density. The first classifier, MAXIMASK, performs semantic segmentation and generates bad pixel maps for each contaminant, based on the probability that each pixel belongs to a given contaminant class. The second classifier, MAXITRACK, classifies entire images and mosaics, by computing the probability for the focal plane to be affected by tracking errors. We gathered training and testing data from real data originating from various modern charged-coupled devices and near-infrared cameras, that are augmented with image simulations. We quantified the performance of both classifiers and show that MAXIMASK achieves state-of-the-art performance for the identification of cosmic ray hits. Thanks to a built-in Bayesian update mechanism, both classifiers can be tuned to meet specific science goals in various observational contexts.

Key words. methods: data analysis – techniques: image processing – surveys

1. Introduction

Catalogs extracted from astronomical images are at the heart of modern observational astrophysics. Minimizing the number of spurious detections in these catalogs has become increasingly important because the noise added by such contaminants can, in many cases, compromise the scientific objectives of a survey. Properly identifying and flagging spurious detections yields substantial scientific gains, but it is complicated by the numerous types of contaminants that pollute images. Some of them stem from the detector electronics (e.g., dead or hot pixels, persistence, saturation), from the optics (diffraction along the optical path, scattered and stray light), from post-processing (e.g., residual fringes), while others are the results of external events (cosmic rays, satellites, tracking errors). The amount of data produced by modern astronomical surveys makes visual inspection impossible in most cases. For this reason, developing fully automated methods to separate contaminants from true astrophysical sources is a critical issue in modern astronomical survey pipelines.

Most current pipelines rely on a fine prior knowledge of their instruments to detect and mask electronic contaminants (e.g., [Bosch et al. 2018](#); [Morganson et al. 2018](#)) and to some extent optical contaminants (e.g., [Kawanomoto et al. 2016a,b](#)). Cosmic ray hits can be identified by rejecting outliers in the timeline, provided that multiple consecutive exposures are available, by using algorithms sensitive to their peculiar shapes, such as Laplacian edge detection (e.g., LA Cosmic, [van Dokkum 2001](#)) or wavelets (e.g., [Ordénovic et al. 2008](#)). The Radon transform or the Hough transform have often been used to detect streaks caused by artificial satellites or planes in images (e.g. [Vandame 2002](#); [Nir et al. 2018](#)).

In this work, we want to overcome some of the drawbacks of the above mentioned methods. First, the typical data volume produced by modern surveys requires that the software is largely unsupervised and as efficient as possible. Second, we aim to develop a robust and versatile tool for the community at large and therefore want to avoid the pitfall inherent in software that is tailored to a single or a handful of instruments, without compromising on performance. Third, we would like to have a unified tool able to detect many contaminants at once. Finally, we want to assign to each pixel a probability of belonging to a given contaminant class rather than Boolean flags. These constraints lead us to choose machine learning techniques and in particular supervised learning and convolutional neural networks (CNNs).

Supervised learning is a field of machine learning dealing with models that can learn regression or classification tasks based on a data set containing the inputs and the expected outputs. During the learning process, model parameters are adjusted iteratively to improve the predictions made from the input data. The learning procedure itself consists of minimizing a loss function that measures the discrepancy between model predictions and the expected values. Minimization is achieved through stochastic gradient descent. We recommend [Ruder \(2016\)](#) for an overview of gradient descent based optimization algorithms.

Convolutional neural networks ([LeCun & Bengio 1995](#)) are particularly well-suited for identifying patterns in images. Unlike previous approaches that would involve hand crafted feature detectors, such as SIFT descriptors ([Lowe 1999](#)), CNN models operate directly on pixel data. This is made possible by the use of trainable convolution kernels to detect features in images. Convolution is shift-equivariant, which allows the same features to be detectable at any image location.

CNNs are now widely used in various computer vision tasks, including image classification, that is assigning a label to a whole image (Krizhevsky et al. 2012; Simonyan & Zisserman 2014; Szegedy et al. 2015), and semantic segmentation, that is assigning a label to each pixel (Long et al. 2015; Badrinarayanan et al. 2017; Garcia-Garcia et al. 2017).

In this work, we propose to identify contaminants using both image classification and semantic segmentation.

In the following, we first describe the images that we used and how we built our data sets. Then, we focus on the neural network architecture that we used. Finally, we evaluate the models performance on test sets and on real data.

2. Data

In this section we describe the data used to train our two neural networks. We distinguish between two types of contaminants: On the one hand, local contaminants, that affects only a fraction of the image at specific locations. This includes cosmic rays, hot columns and lines, dead columns and lines, dead clustered pixels, hot pixels, dead pixels, persistence, satellite trails, residual fringe patterns, “nebulosity”, saturated pixels, diffraction spikes, and over scanned pixels. These add up to 12 classes. On the other hand, global contaminants, that affects the whole image, such as tracking errors.

2.1. Local contaminant data

For local contaminants, we choose to build training samples by adding defects to uncontaminated images in order to have a ground truth for each contaminant. In this section we first describe the library of astronomical images used for our analysis, then focus on the selection of uncontaminated images, and finally describe the way each contaminant is added.

2.1.1. Library of real astronomical images

In an effort to have the most realistic dataset, we choose to use real data as much as possible and take advantage of the private archive of wide-field images gathered for the COSMIC-DANCE survey (Bouy et al. 2013). The COSMIC-DANCE library offers several advantages. First, it includes images from many past and present optical and near-infrared wide-field cameras. Images cover a broad range of detector types and ground-based observing sites, ensuring that our dataset is representative of most modern astronomical wide-field instruments. Table 1 gives an overview of the properties of the cameras used to build the image database. Second, most problematic exposures featuring tracking/guiding loss, defocusing or strong fringing were already identified by the COSMIC-DANCE pipeline, providing an invaluable sample of real problematic images.

In all cases except for Megacam, DECam, UKIRT and HSC exposures, the raw data and associated calibration frames were downloaded and processed using standard procedures with an updated version of *Alambic* (Vandame 2002), a software suite developed and optimized for the processing of large multi-chip imagers. In the case of Megacam, the exposures processed and calibrated with the *Elixir* pipeline were retrieved from the CADC archive (Magnier & Cuillandre 2004). In the case of DECam, the exposures processed with the community pipeline were retrieved from the NOAO public archive (Valdes et al. 2014). UKIRT exposures processed by the Cambridge Astronomical Survey Unit were retrieved from the WFCAM Science Archive. Finally, the HSC raw images were processed using the official HSC pipeline (Bosch et al. 2018). In all cases, a bad pixel map is

Table 1. Instruments used in this study.

Telescope	Instrument	Type	Platescale [pixel ⁻¹]	Ref.
CTIO Blanco	DECam	CCD	0′′26	(1)
CTIO Blanco	MOSAIC2	CCD	0′′26	(2)
KPNO Mayall	MOSAIC1	CCD	0′′26	(2)
KPNO Mayall	NEWFIRM	IR	0′′4	(3)
CFHT	MegaCam	CCD	0′′18	(4)
CFHT	CFH12K	CCD	0′′21	(5)
CFHT	UH8K	CCD	0′′21	(6)
INT	WFC	CCD	0′′33	(7)
UKIRT	WFCAM	IR	0′′4	(8)
LCO Swope	Direct CCD	CCD	0′′43	(9)
VST	OmegaCam	CCD	0′′21	(10)
Subaru	HSC	CCD	0′′17	(11)
VISTA	VIRCAM	IR	0′′34	(12)

References. (1) Flaugher et al. (2010); (2) Wolfe et al. (2000); (3) Autry et al. (2003); (4) Boulade et al. (2003); (5) Cuillandre et al. (2000); (6) Metzger et al. (1995); (7) Ives (1998); (8) Casali et al. (2007); (9) Rheault et al. (2014); (10) Kuijken et al. (2002); (11) Miyazaki et al. (2018); (12) Dalton et al. (2006).

associated to every individual image. In the case of DECam and HSC, a data quality mask is also associated to each individual image and provides integer-value codes for pixels which are not scientifically useful or suspect, including in particular bad pixels, saturated pixels, cosmic ray hits, satellite tracks, etc. All the images in the following consist of individual exposures and not co-added exposures.

2.1.2. Non-contaminated images

None of the exposures in our library are defect-free. The first step to create the non-contaminated dataset to be used as “reference” images consists in identifying the cleanest possible subset of exposures. CFHT-Megacam (u, r, i, z bands), CTIO-DECam (g, r, i, z, Y bands) and Subaru-HSC (g, r, i, z, y bands) exposures are found to have the best cosmetics and are selected to create the non-contaminated dataset. The defects inevitably present in these images are handled as follows.

First, dead pixels and columns are identified from flat-field images and inpainted using Gaussian interpolation (e.g., Williams et al. 1998). Then, the vast majority of cosmic rays are detected using the Astro-SCRAPPY Python implementation (McCully et al. 2018) of LA Cosmic (van Dokkum 2001) and also inpainted using Gaussian interpolation. Finally, given the high performance of the DECam and HSC pipelines, the corresponding images are perfect candidates for our non-contaminated datasets. These two pipelines not only efficiently detect but also interpolate problematic pixels (in particular saturated pixels, hot and bad pixels, cosmic ray hits). Such interpolations being a feature of several modern pipelines (e.g., various NOAO pipelines, but also the LSST pipeline), we choose to treat these pixels as regular pixels so that the networks are able to work with images originating from such pipelines.

Patches of size 400×400 pixels are randomly extracted from the cleaned images. 75% of them are used to generate training data and the remaining 25% for test data.

The final non-contaminated dataset includes 50 000 individual images, ensuring that we have a sufficiently diverse and large amount of training data for our experiment.

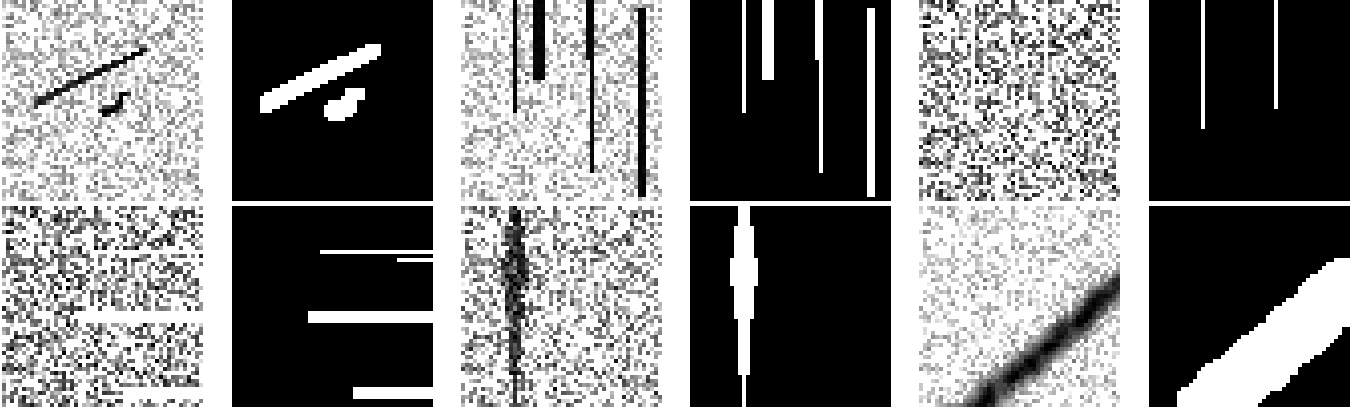


Fig. 1. Examples of contaminants and their ground truth. *Top row:* cosmic ray hits, hot columns, bad columns. *Bottom row:* bad lines, persistence, satellite trails.

A non-representative training set can severely impact the performance of a CNN and result in significant biases in the classification task. To prevent this, we measure a number of basic properties describing prototypical aspects of ground-based astronomical images to verify that their distributions in the uncontaminated dataset are wide enough and reasonably well sampled.

The measured properties include, for example, the average full-width at half-maximum (FWHM) of point-sources is estimated in each image using PSFEX (Bertin 2013). This allows us to ensure that the training set covers a broad range of ambient (seeing) conditions and point spread functions (PSFs) sampling. Also, the source density (number of sources in the image divided by the physical size of the image) is measured to make sure that our training set encompasses a broad range of source crowding, from sparse cosmological fields to dense, low-galactic latitude stellar fields.

Additionally, the background is modeled in all the images following the method used by SEXTRACTOR (Bertin & Arnouts 1996), i.e. using a combination of $\kappa\sigma$ -clipping and mode estimation. The background model provides important parameters such as the standard deviation of the background which is required in most of the data-processing operations that follow.

2.1.3. Cosmic rays (CR)

“Cosmic ray” hits are produced by particles hitting the detector or by the photons resulting from the decay of radioactive atoms near the detector. They appear as bright and sharp patterns with shapes ranging from dots affecting one or two pixels to long wandering tracks commonly referred to as “worm”, depending on incidence angle and detector thickness.

We create a library of real CRs using dark frames with long exposure times from the CFH12K, HSC, MegaCam, MOSAIC, and OmegaCam cameras. These cameras comprise both “thick”, red-sensitive, deep depletion charged-couple devices (CCDs), more prone to long worms, and thinner, blue-sensitive devices, more prone to unresolved hits. Dark frames are exposures taken with the shutter closed, so that the only contributors to the content of undamaged pixels are the offset, dark current, and CR hits (plus Poisson and readout noise). A mask M of the pixels affected by CR hits in a given dark frame D can therefore easily be generated by applying a simple detection threshold. We conservatively set this threshold to $3\sigma_D$ above the median value m_D of D :

$$\forall p, M_p = \begin{cases} 1 & \text{if } D_p > m_D + 3\sigma_D \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

Among all the dark images used, a bit more than 900 million cosmic ray pixels are detected after thresholding. Considering that the average footprint area of a cosmic ray hit is 15 pixels, this represents a richly diversified population of about 60 million cosmic ray “objects”.

Next we dilate M with a 3×3 pixel kernel to create the final $M^{(D)}$ mask. This mask is used both as ground truth for the classifier, and also to generate the final “contaminated” image C by adding CR pixels with rescaled values to the uncontaminated image U :

$$C = U + k_C \frac{\sigma_U}{\sigma_D} D \odot M^{(D)}, \quad (2)$$

where σ_U is the estimated standard deviation of the uncontaminated image background, \odot denotes the element-wise product and k_C is a scaling factor empirically set to $1/8$. D has been background-subtracted before this operation, using a SEXTRACTOR-like background estimation.

A typical CR hit added to an image and its ground truth mask are shown in Fig. 1.

2.1.4. Hot columns and lines, dead columns, lines, and clustered pixels, hot pixels, and dead pixels (HCL, DCL, HP, DP)

These contaminants mainly come from electronic defects and the way the detectors are read. They correspond to pixels having a response very different from that of neighbors, either much lower (bad pixels, traps) or much noisier (hot pixels). These blemishes can be found as single pixels, in small clusters, or affecting a large fraction of a column or row. We treat single pixels and clumps, columns, and lines separately, although they may often share a common origin.

All these hot or dead pixels added to the uncontaminated images are simulated. The number of these pixels is set as follow.

For columns and lines, a random number of columns and lines is chosen with a uniform distribution over $[1,4]$. Each column or line has a uniform length picked between 30 and the whole image height or width. It has a uniform thickness in $[1,3]$. For punctual pixels, a random fraction of pixels is chosen with a uniform distribution between 0.0002 and 0.0005. Pixels are uniformly distributed over the image. Clustered pixels are given a rectangular or a random convex polygonal shape. The random convex shapes are constrained to have 5 or 6 edges and to fit in 20×20 bounding boxes.

The values of these pixels are computed as follows. For hot values, a uniformly distributed random base value ν is chosen in the interval $[15\sigma_U, 100\sigma_U]$. Then hot values are generated according to the normal law $\mathcal{N}(\nu, (0.02\nu)^2)$ so that hot values are randomly distributed over $[0.9\nu, 1.1\nu]$. For dead values, one of the following three equiprobable recipes is chosen at random to generate bad pixel values. Either all values are exactly 0. Either values are generated according to the normal law $\mathcal{N}(0, (0.02\sigma_U)^2)$ so that these are close to 0 values but not exactly 0. Either a random base value ν is chosen with a uniform distribution in the interval $[0.1m_U, 0.7m_U]$, where m_U is the median of the uncontaminated image sky background. In this case, dead pixel values are generated using the normal law $\mathcal{N}(\nu, (0.02\nu)^2)$, so that values fall in the interval $[0.9\nu, 1.1\nu]$.

Example of such column and line defaults are shown in Fig. 1.

2.1.5. Persistence (P)

Persistence occurs when overly bright pixels in a previous exposure leave a remnant image in the following exposures.

To simulate this effect in an uncontaminated image, we applied the so-called ‘‘Fermi model’’ described in Long et al. (2015). Persistence, in units of $e^- \cdot s^{-1}$, is modeled as a function of the initial pixel level x_p and time t :

$$f(x_p, t) = A_p \left(\frac{1}{\exp(-\frac{x_p - x_0}{\delta x}) + 1} \right) \left(\frac{x_p}{x_0} \right)^\alpha \left(\frac{t}{1000} \right)^{-\gamma}. \quad (3)$$

The goal of Long et al. (2015) was to fit the model parameters $x_0, \delta x, \alpha, \gamma$ using observations to later predict persistence for their detector. In our simulations, parameter values are randomized to represent various types and amounts of persistence (see Table 2). To compute the pixel value of the persistence effect, we derive the number of electrons emitted by the persistence effect during the exposure. In the following, we note T the duration of the exposure in which the persistence effect occurs, and Δt the delay between that exposure and the previous one. We obtain the number of ADUs collected at pixel p during the interval $[\Delta t, \Delta t + T]$ by integrating Eq. (3) and dividing by the gain G :

$$P_p = \frac{1}{G} \int_{\Delta t}^{\Delta t + T} f(x_p, t) dt \quad (4)$$

$$= \frac{A_p}{G} \left(\frac{1000^\gamma}{\exp(-\frac{x_p - x_0}{\delta x}) + 1} \right) \left(\frac{x_p}{x_0} \right)^\alpha \left(\frac{(\Delta t + T)^{1-\gamma} - \Delta t^{1-\gamma}}{1 - \gamma} \right). \quad (5)$$

These pixel values are then added to the uncontaminated image:

$$C = U + k_P \sigma_U \frac{P - P_{\min}}{(P_{\max} - P_{\min})}, \quad (6)$$

where P are the persistence values computed in Eq. (5), P_{\min} and P_{\max} are the minimum and maximum of these values, and k_P is a scaling factor empirically set to 5.

Images of saturated stars are simulated using SKYMAKER (Bertin 2009) and binarized to generate masks of saturated pixels. The masks define the footprints of persistence artifacts, within which the x_p ’s are computed (Table 2). An example is shown in Fig. 1.

Table 2. Parameters used for the generation of persistence.

A_p	1
$x_p (e^-)$	Poisson(x_m) with $x_m \sim \mathcal{N}(15.10^5, (0.02 \times 15.10^5)^2)$
$x_0 (e^-)$	$\mathcal{N}(9.10^4, (0.02 \times 9.10^4)^2)$
$\delta_x (e^-)$	$\mathcal{N}(18.10^3, (0.02 \times 18.10^3)^2)$
α	0.178
γ	1.078
$G (e^- \cdot s^{-1})$	$\mathcal{N}(10, 1)$

2.1.6. Trails (TRL)

Satellites or meteors, and even planes crossing the field of view generate long trails across the frame that are quasi-rectilinear. We simulate these motion-blurred artifacts by generating close star images with identical magnitudes along a linear path using once again SKYMAKER. We also generate a second population of trails with magnitude changes to account for satellite ‘‘flares’’. A random, Gaussian-distributed component with a ≈ 1 pixel standard deviation is added to every stellar coordinate to simulate jittering from atmospheric turbulence, so that the stars are not aligned along a perfect straight line. For meteors, defocusing must be taken into account (Beketešević et al. 2018). The amount of defocusing θ , expressed as the apparent width of the pupil pattern in arc-seconds, is:

$$\theta = \frac{180}{\pi} \times 3600 \times \frac{D}{d}, \quad (7)$$

where D is the diameter of the primary mirror, and d the meteor distance, both in meters. D and d are randomly drawn from flat distributions in the intervals $[2, 8]$ and $[80\,000, 120\,000]$, respectively.

The ground truth mask is obtained by binarizing the satellite image at a small and arbitrary threshold above the simulated background. This mask is then dilated using a 7×7 pixel structuring element.

To avoid any visible truncation, we add the whole simulated satellite image multiplied by a dilated version $M^{(S)}$ of the ground truth mask to the uncontaminated image:

$$C = U + k_T \frac{\sigma_U}{\sigma_T} T \odot M^{(T)}, \quad (8)$$

where σ_S is the standard deviation of the satellite image background, σ_U the standard deviation of the uncontaminated image background, and k_T is a scaling factor empirically set to 6. An example of a satellite trail is shown in Fig. 1.

2.1.7. Fringes (FR)

Fringes are thin-film interference patterns occurring in the detectors. The irregular shape of fringes is caused by thickness variations within the thin layers. To add fringing to images, we use real fringe maps produced at the pre-processing level by ALAMBIC for all the optical CCD cameras of Table 1. These reconstructed fringe maps are often affected by white noise, which we mitigate by smoothed using a top-hat kernel with diameter 7 pixels. The fringe pattern F can affect large areas in an image but not necessarily all the image. To reproduce this effect, a random 3rd-degree 2D polynomial envelope E that covers the whole image is generated. The final fringe envelope $E^{(F)}$

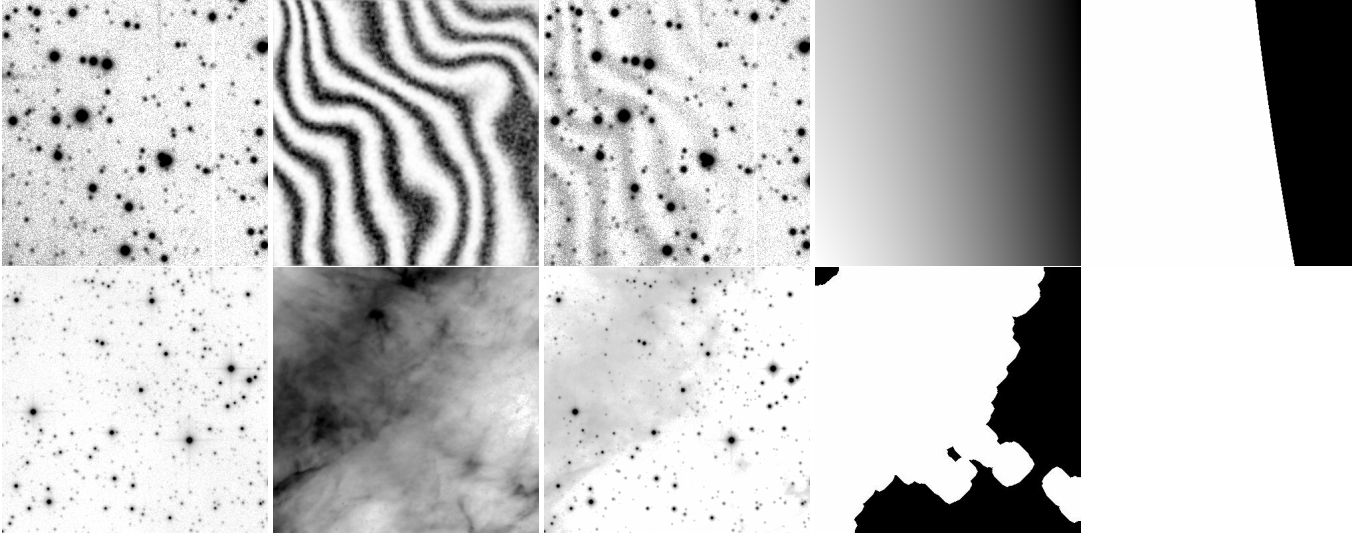


Fig. 2. Examples of added fringes and nebulosities. *Top*: fringes; uncontaminated input exposure, smoothed fringe pattern, contaminated image, ground truth mask, polynomial envelope. *Bottom*: nebulosities; uncontaminated input exposure, *Herschel* 250 μm molecular cloud image, contaminated image, ground truth mask.

is computed by normalizing E over the interval $[-5, 5]$ and flattening the result using the sigmoid function:

$$E_p^{(F)} = \left(1 + \exp \left(-5 \frac{2E_p - E_{\max} - E_{\min}}{E_{\max} - E_{\min}} \right) \right)^{-1}, \quad (9)$$

where E_{\min} and E_{\max} are the minimum and maximum values of E_p , respectively.

The fringe pattern, modulated by its envelope, is then added to the uncontaminated image:

$$C = U + k_F \frac{\sigma_U}{\sigma_F} F \odot E^{(F)}, \quad (10)$$

where σ_F is the standard deviation of the fringe pattern and k_F is an empirical scaling factor set to 0.6. The ground truth mask is computed by thresholding the 2D polynomial envelope to -0.20 .

An example of a simulated contamination by a fringe pattern can be found in Fig. 2.

2.1.8. Nebulosity (NEB)

Extended emission originating from dust clouds illuminated by star light or photo-dissociation regions can be present in astronomical images. These “nebulosities” are not artifacts but they make the detection and measurement of overlapping stars or galaxies more difficult; they may also trigger the fringe detector. Hence, it is useful to have them identified and properly flagged. Because thermal distribution of dust closely matches that of reflection nebulae at shorter wavelength (e.g., [Ienaka et al. 2013](#)), we use far-infrared images of molecular clouds around star-forming regions as a source of nebulous contaminants. We choose pipeline-processed 250 μm images obtained with the SPIRE instrument ([Griffin et al. 2010](#)) on-board the *Herschel* Space Observatory ([Pilbratt et al. 2010](#)), which we retrieve from the *Herschel* Science Archive. The 250 μm channel offers the best compromise between signal-to-noise ratio and spatial resolution. Moreover, at wavelengths of 250 μm and above, low galactic latitude fields contain mostly extended emission from the cold gas and almost no point sources (apart from a few proto-stars and proto-stellar cores). Therefore, they are perfectly

suited to being added to our optical and near-infrared wide-field exposures. We do not resize or reconvolve the SPIRE images, taking advantage of the scale-invariance of dust emission observed down to the arcsecond level in molecular clouds ([Miville-Deschênes et al. 2016](#)).

We add the nebulous contaminant data to our uncontaminated images in the same way we do for fringes, except that there is no 2D polynomial envelope. The whole nebulosity image is background-subtracted (using a SEXTRACTOR-like background estimation) to form the final nebulosity pattern N which is then added to the uncontaminated image:

$$C = U + k_N \frac{\sigma_U}{\sigma_N} N, \quad (11)$$

where k_N is an empirical scaling factor set to 1.3. The ground truth mask is computed by thresholding N at one sigma above 0. This mask is then eroded with a 6 disk diameter structuring element to remove spurious individual pixels, and dilated with a 22 disk diameter structuring element. An example of added nebulosity is shown in Fig. 2. The light from line-emission nebulae may not necessarily exhibit the same statistical properties as the reflection nebulae targeted for training. However line-emission nebulae are generally brighter and in practice the classifier has no problem detecting them.

2.1.9. Saturation and bleeding (SAT)

Each detector pixel can accumulate only a limited number of electrons. Once the full well limit is reached, the pixel becomes saturated. In CCDs, charges may even *overflow*, leaving saturation trails (a.k.a bleeding trails) along the transfer direction. Such pixels are easily identified in clean images knowing for each instrument the saturation level.

2.1.10. Diffraction spikes

Diffraction spikes are patterns appearing around bright stars and caused by light diffracting around the spider supporting the secondary mirror. Given the typical cross-shape of spiders, the pattern is usually relatively easy to identify. In some

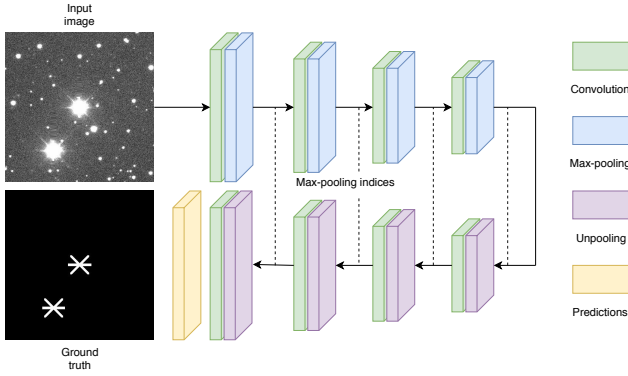


Fig. 3. Neural network used specifically for spike detection.

cases, the pattern can deviate significantly from a simple cross because it is affected by various effects, such as distortions, telescope attitude, the truss structure of spider arms, rough edges, or cables around the secondary mirror support, reflections on other telescope structures, ... A specific strategy was put in place to build a spikes library to be used to train the CNN.

On the one hand, MegaCam and DECam are mounted on equatorial telescopes and the orientation of spikes is usually (under standard northeast orientation) a “+” for Megacam and an “x” for DECam¹. On the other hand, HSC is mounted on the alt-az Subaru telescope, and spikes do not display any preferred orientation, making their automated identification more complicated. For this reason, we define a two-step strategy, in which, first, samples of “+”- and “x”-shaped spikes are extracted from DECam and Megacam images, and randomly rotated to generate a library of diffraction spikes with various orientations. The library is then used to train a new CNN that for identifying spikes in HSC images.

MegaCam and DECam analysis. We first identify the brightest stars using SEXTRACTOR and extract 300×300 pixel image cutouts around them. The cutouts are thresholded at three sigma above the background and binarized. Element-wise products are computed between these binary images and large “+”-shaped (Megacam) or “x”-shaped (DECam) synthetic masks to isolate the central stars. Each point-wise product is then matched-filtered with a thinner version of the same pattern and binarized using an arbitrary threshold set to 15 ADUs. The empirical size of the spike components is estimated in these masks by measuring the maximum extent of the resulting footprint along any of the two relevant spike directions (horizontal and vertical or diagonals). Finally, the maximum size of the two directions is kept and empirically rescaled to obtain the final spike length and width. If the resulting size is too small, we consider that there is no spike in order to avoid false positives (e.g., a star bright enough to be detected by SEXTRACTOR but without obvious spikes). Figure 5 gives an overview of the whole process.

HSC analysis. We train a new neural network to identify spikes in all directions. For that purpose, we build a new training set using the spikes identified in MegaCam and DECam images as described above and apply a random rotation between 0° and 360° to ensure rotational invariance. The neural network has a simple SegNet-like convolutional-deconvolutional architecture (Badrinarayanan et al. 2015), but it is not based on VGG hyper-parameters (Simonyan & Zisserman 2014). It uses 21×21 , 11×11 , 7×7 and 5×5 convolutional kernels in 8, 16, 32 and 32 feature maps, respectively. The model architecture is shown

¹ DECam images sometimes also exhibit a horizontal spike of unknown origin (Melchior et al. 2016).

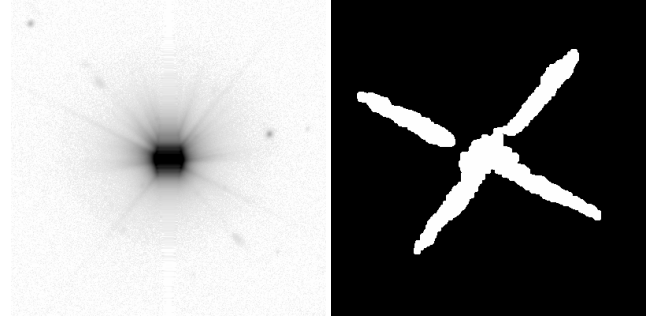


Fig. 4. Example of a spike mask obtained by inference of the separate neural network.

in Fig. 3. Activation functions are all ELU except on the last layer where it is softmax. It is trained to minimize the softmax cross entropy loss with the Adam optimizer (Kingma & Ba 2014). Each pixel cost is weighted to balance the disproportion between spike and background pixels. If p_s is the spike pixel proportion in the training set, then spike pixels are weighted with $1 - p_s$, while background pixels are weighted with p_s (this is the two-class equivalent of the basic weighting scheme described in Sect. 3.1). Once trained we run inferences on all the brightest stars detected with SEXTRACTOR in the HSC images. Output probabilities are binarized based on the MCC (see Eq. (22)) and the resulting mask is empirically eroded and dilated to obtain a clean mask. An example is given in Fig. 4.

2.1.11. Overscan (OV)

Overscan regions are common in CCD exposures, showing up as strips of pixels with very low values at the borders of the frame. To avoid triggering false predictions on real data, overscans must be included in our training set. Doing so, and although these are not truly contaminants, we find it useful to include an “overscan” class in the list of identified features. Overscan regions are simulated by including random strips on the sides of images. Pixel values in the strips are generated in the same way as bad pixel values.

2.1.12. Bright background (BBG) and background (BG)

The objects of interest in this study are the contaminants. Hence, following standard computer vision terminology, all the other types of pixels, including both astronomical objects and empty sky areas, belong to the “background”.

We find that defining a distinct class for each of these types of background pixels helps with the training procedure. We thus define the “bright background” (BBG) pixels as pixels belonging to astronomical objects² (except nebulosity) present in the uncontaminated images, and background pixels (BG) as pixels covering an empty sky area.

Ground truth masks for bright background pixels are obtained by binarizing the image before adding the contaminants to $10\sigma_U$. The remaining pixels are sky background pixels, which are not affected by any labeled feature.

2.2. Global contaminants

We now describe the data used to identify global contaminants.

² Including astrophysical sources in the “background” class can seem somewhat counter-intuitive in a purely astronomical context, but for consistency we choose to follow the computer vision terminology and meaning.

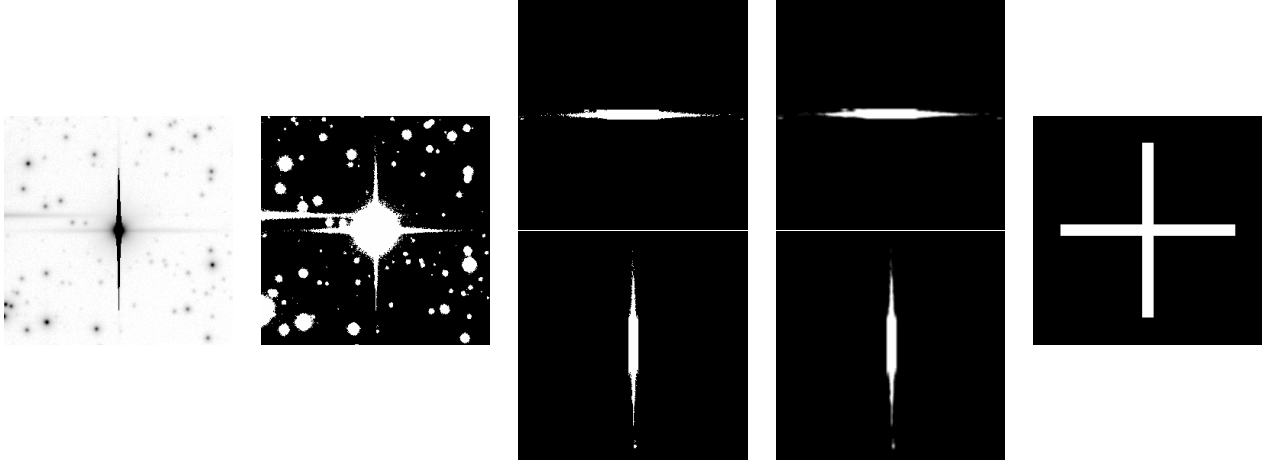


Fig. 5. Empirical spike flagging process. *From left to right:* source image centered on a bright star candidate, the same image thresholded, the two pointwise products, the matched filtered pointwise products, the final mask drawn from the empirical size computed with the two previous masks.

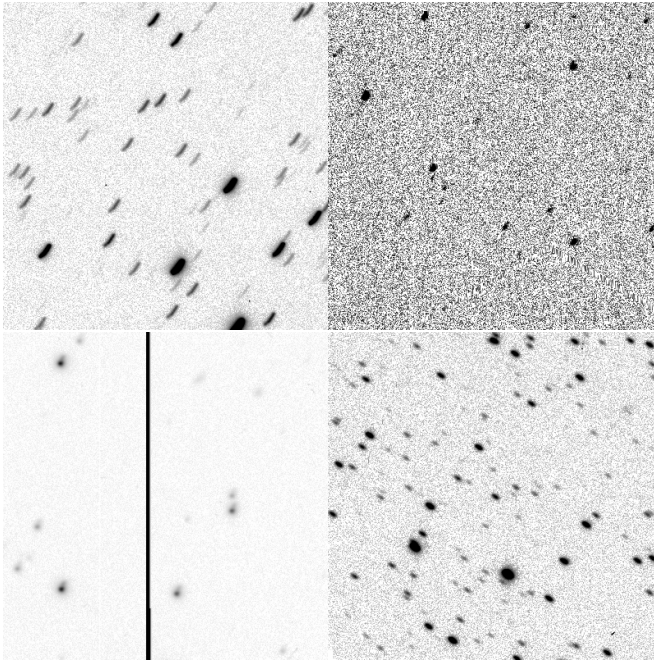


Fig. 6. Examples of images affected by tracking errors.

Tracking errors happen when the telescope moves during an exposure due to, for instance, telescope guiding or tracking failures, wind gusts, or earthquakes. As illustrated in Fig. 6, this causes all the sources to be blurred along a path on the celestial sphere generated by the motion of the telescope. Because tracking errors affect the entire focal plane, the analysis is performed globally on the whole image. The library of real images affected by TR events is a compilation of exposures identified in the COSMIC-DANCE survey for the cameras of Table 1, and images that were gathered over the years at the UKIRT telescope, kindly provided to us by Mike Read.

2.3. Generating training samples

Both types of contaminants – global and local contaminants – must be handled separately: they require different neural network architectures, and different training data sets as well.

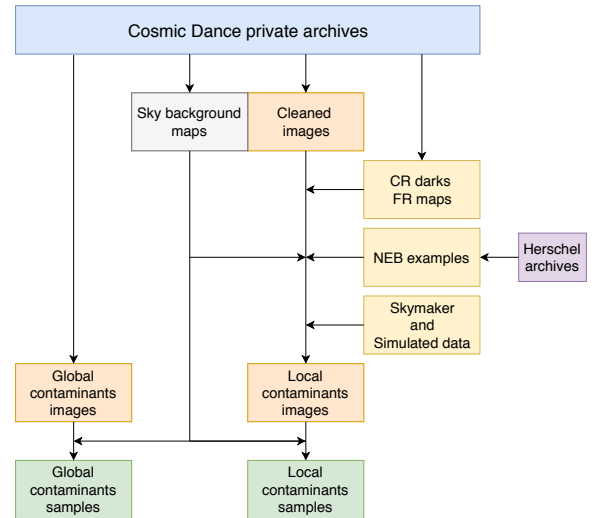


Fig. 7. Schematic view of the sample production pipeline. All COSMIC-DANCE archive images have their background map computed. Clean images are built from the COSMIC-DANCE archives. Contaminants from diverse sources (COSMIC-DANCE archives, *Herschel* archives or simulations) are added to clean images; this step uses the background maps. The resulting local contaminant images are dynamically compressed (see Sect. 2.3.3) and ready to be fetched into the neural network. Global contaminant samples are directly obtained from the COSMIC DANCE archives and dynamically compressed.

Figure 7 gives a synthetic view of the sample production pipeline and the various data sources.

The breakdown per imaging instrument of the COSMIC DANCE dataset is listed Table 3.

The following subsections treat about some special features of the sample generation.

2.3.1. Local contaminants

The order in which local contaminants are added is important. Bad columns, lines, and pixels are added last because they are static defaults defining the final value of a pixel, no matter how many photons hit them.

In our neural network architecture contaminant classes do not need to be mutually exclusive. Each pixel can be assigned several classes as several defaults can affect a given pixel

Table 3. COSMIC-DANCE archive usage per imaging instrument.

Instrument	Clean	CR	No TR	TR
DECam	✓		✓	
MOSAIC2		✓		
MOSAIC1		✓		
NEWFIRM			✓	✓
Megacam	✓	✓	✓	✓
CFH12K		✓	✓	✓
CFH8K				✓
WFC			✓	✓
WFCAM			✓	✓
Direct CCD (LCO Swope)			✓	✓
VST		✓	✓	✓
HSC	✓	✓	✓	✓
VIRCAM			✓	✓

Notes. Clean is for uncontaminated images, CR for dark images used for cosmic ray identification, No TR is for images not affected by tracking errors, and TR for images affected by tracking errors.

Table 4. All the contaminants and their abbreviated names.

Contaminant	Abbreviation
Cosmic rays	CR
Hot columns/lines	HCL
Dead columns/lines/clusters	DCL
Hot pixels	HP
Dead pixels	DP
Persistence	P
Trails	TRL
Fringes	FR
Nebulosities	NEB
Saturated pixels	SAT
Diffraction spikes	SP
Overscanned pixels	OV
Bright background	BBG
Background	BG

(e.g., fringes and cosmic ray hit). On the other hand, the faint background class that defines pixels not affected by any default excludes all other classes. A list of all the contaminants included in this study are presented in Table 4.

Figure 8 shows examples of local contaminant sample input images, each with its color-coded ground truth.

2.3.2. Global contaminants

The global contaminant dataset contains images that have been hand labeled as affected by tracking errors or not. The images, taken from the COSMIC DANCe archives, are not cleaned, hence they are potentially affected by preexisting local contaminants. This is because the global contaminant detector is intended to be operated before the local one.

2.3.3. Dynamic compression

All images are dynamically compressed before being fed to the neural networks using the following procedure:

$$\tilde{C} = \text{arsinh} \left(\frac{C - B + \mathcal{N}(0, \sigma_U^2)}{\sigma_U} \right).$$

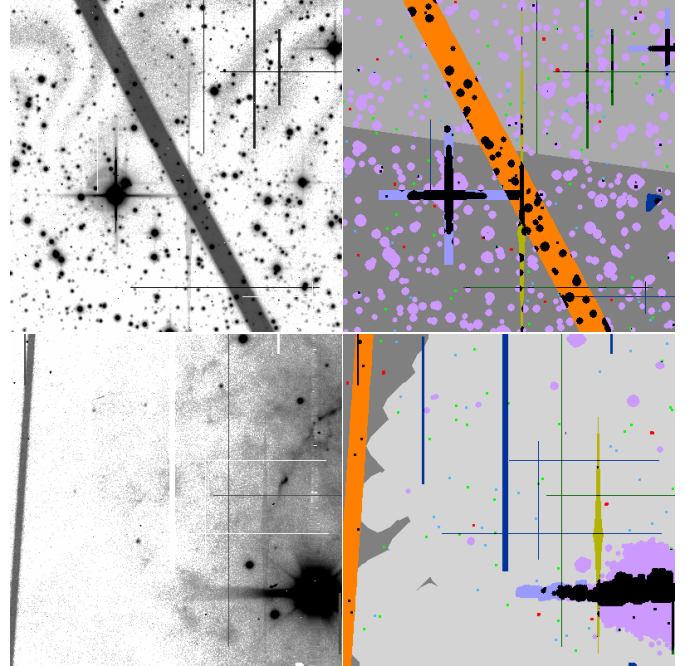


Fig. 8. Examples of input (*left*) and their ground truth (*right*). Each class is assigned a color so that the ground truth can be represented as a single image (red: CR, dark green: HCL, dark blue: BCL, green: HP, blue: BP, yellow: P, orange: TRL, gray: FR, light gray: NEB, purple: SAT, light purple: SP, brown: OV, pink: BBG, dark gray: BG). Pixels that belong to several classes are represented in black. In the interest of visualization, hot and dead pixel masks have been morphologically dilated so that they appear as 3×3 pixel areas in this representation.

The aim of dynamic compression is to reduce the dynamic range of pixel values, which is found to help neural network convergence. The image is first background subtracted. Then, a small random offset is added to increase robustness regarding background subtraction residuals. The resulting image is normalized by the standard deviation of the background noise and finally compressed through the arsinh function, which has the property to behave linearly around zero and logarithmically for large (positive or negative) values.

2.3.4. Data augmentation

We deploy data augmentation techniques to use our data to the maximum of its information potential. The two following data augmentation procedures are applied to the set of local contaminant training samples. First, random rotations, using as angles multiples of 90° , are applied to cosmic ray, fringe patterns, and nebosity patterns. Secondly, some images are rebinned. When picking up a clean image, we check if the image can be 2×2 rebinned with the constraint that the FWHM remains greater than 2 pixels – the FWHM of the image was previously estimated using SEXTRACTOR (Bertin & Arnouts 1996). This value is chosen on the basis of the plate sampling offered by current ground-based imagers. If the image can be 2×2 rebinned while meeting the condition above, it has a 50% probability to be rebinned.

3. Convolutional neural networks

In this section, we describe the convolutional neural networks used for our analysis. The first one, MAXIMASK, classifies pixels (“local contaminants”) while the second one, MAXITRACK, classifies images (“global contaminants”).

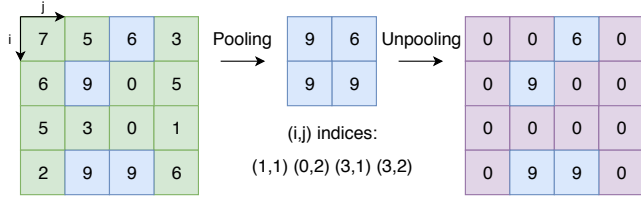


Fig. 9. Example of an unpooling process. Indices of max-pooling are kept up and reused to upsample the feature maps.

3.1. Local contaminant neural network

3.1.1. Architecture

The model used for the semantic segmentation of the local contaminants, MAXIMASK, is based on [Badrinarayanan et al. \(2015\)](#) and [Yang et al. \(2018\)](#), which both rely on a VGG-like architecture ([Simonyan & Zisserman 2014](#)). It consists of three parts.

The first part contains single and double convolutional layers followed by max-pooling downsampling. This enables the network to compute relevant feature maps at different scales. During this step, max-pooling pixel indices are kept up for later reuse.

The second part also incorporates convolutional layers and recovers spatial resolution by upsampling feature maps using the max-pooling indices. An example of unpooling is given in [Fig. 9](#). At each resolution level, the feature maps of the first part are summed with the corresponding upsampled feature maps to make use of the maximum of information.

The third part is made of extra unpool-convolution paths (UCPs) that recover the highest image resolution from each feature map resolution so that the network can exploit the maximum of information of each resolution. Thus, it results 5 pre-predictions, one for each resolution.

The 5 pre-predictions are finally concatenated and a last convolution layer builds the final predictions. The sigmoid activation functions in this last layer are not *softmax*-normalized, to allow non-mutually exclusive classes to be assigned jointly to pixels. All convolutional layers use 3×3 kernels and apply ReLU activations. The architecture is represented in [Fig. 10](#) and hyperparameters are described more precisely in [Table 5](#). The neural network is implemented using the TensorFlow library ([Abadi et al. 2016](#)) on a TITAN X Nvidia GPU.

3.1.2. Training and loss function

Training is done for 30 epochs on 50 000 images, with mini-batches shuffled at every epoch. The batch size is kept small (10) to maintain a reasonable memory footprint. The model is trained end-to-end using the Adam optimizer ([Kingma & Ba 2014](#)). The loss function L is the sigmoid cross-entropy ([Rubinstein 1999](#)) summed over all classes and pixels, and averaged across batch images:

$$L = -\frac{1}{\text{card}(\mathcal{B})} \sum_{b \in \mathcal{B}} \sum_{p \in \mathcal{P}} w'_{p,b} \sum_{\omega_c \in \mathcal{C}} \left(y_{b,p,c} \log \hat{y}_{b,p,c} + (1 - y_{b,p,c}) \log(1 - \hat{y}_{b,p,c}) \right), \quad (12)$$

where \mathcal{B} is the set of batch images, \mathcal{P} is the set of all image pixels, \mathcal{C} is the set of all contaminant classes, $w'_{p,b}$ is a weight applied to pixel p of image b in the batch (see below), $\hat{y}_{b,p,c}$ is the sigmoid prediction for class ω_c of pixel p of image b in the

batch, and $y_{b,p,c}$ is the ground truth label for class ω_c of pixel p of image b defined as:

$$y_{b,p,c} = \begin{cases} 1 & \text{if } \omega_c \in C_{p,b} \\ 0 & \text{otherwise} \end{cases}, \quad (13)$$

where $C_{p,b} \subset \mathcal{C}$ is the set of contaminant classes labeling pixel p of image b in the batch. In order to improve the back-propagation of error gradients down to the deepest layers, several losses are combined. In addition to the main sigmoid cross-entropy loss L computed on the final predictions, we can compute a sigmoid cross-entropy for each of the 5 pre-predictions. There are several ways to associate all of these losses. Like [Yang et al. \(2018\)](#), we find that adding respectively 33% or 50% of each of the 3 or 2 smallest resolution losses to the main loss works best. The two main rules here are that the additional loss weights should sum to 1 and that higher resolution pre-predictions become less informative as they get closer to the one at full resolution.

Basic training procedures are vulnerable to strong class imbalance, which makes it more likely for the neural network to converge to a state where rare contaminants are not properly detected. Contaminant classes are so statistically insignificant (down to one part in 10^6 with real data, typically) that the classifier may be tricked into assigning all pixels to the background class. To prevent this, we start by applying a basic weighting scheme to each pixel according to its class representation in the training set, that is each pixel p of batch image b belonging to classes in $C_{p,b}$ is weighted by $w_{p,b}$ defined as

$$w_{p,b} = \sum_{\omega_c \in C_{p,b}} w_c, \quad (14)$$

with

$$w_c = \left(P(\omega_c|T) \sum_i \frac{1}{P(\omega_i|T)} \right)^{-1}, \quad (15)$$

where $P(\omega_c|T)$ is the fraction of pixels labeled with class ω_c in the training dataset T . The $P(\omega_c|T)$'s do not sum to one as many pixels belong to several classes and are thus counted several times. We find that the weighting scheme brings slightly better results and less variability in the training if weights are computed at once from the class proportions of the whole set, instead of being recomputed for each image. From [Eq. \(15\)](#) we have:

$$\forall i \in \mathcal{C}, \forall j \in \mathcal{C}, \frac{w_i}{w_j} = \frac{P(\omega_j|T)}{P(\omega_i|T)} \text{ and } \sum_{\omega_c \in \mathcal{C}} w_c = 1. \quad (16)$$

However, with this simple weighting scheme, background class pixels that are close to rare features are given very low weights, although they are decisive for classification. To circumvent this, weight maps are smoothed with a 3×3 Gaussian kernel with unit standard deviation so that highly weighted regions spread over larger areas. Other kernel sizes and standard deviations were tested but we find 3 and 1 to give the best results. The resulting weights of this smoothing are the $w'_{p,b}$ presented in the loss function of [Eq. \(12\)](#).

Finally, the solution is regularized by the l2 norm of all the N network weights, by adding the following term to the total loss:

$$L2_{\text{reg}} = \lambda \sum_i^N \|\mathbf{k}_i\|_2, \quad (17)$$

where the \mathbf{k}_i 's are the convolution kernel vectors. λ sets the regularization strength. We find $\lambda = 1$ to provide the best results.

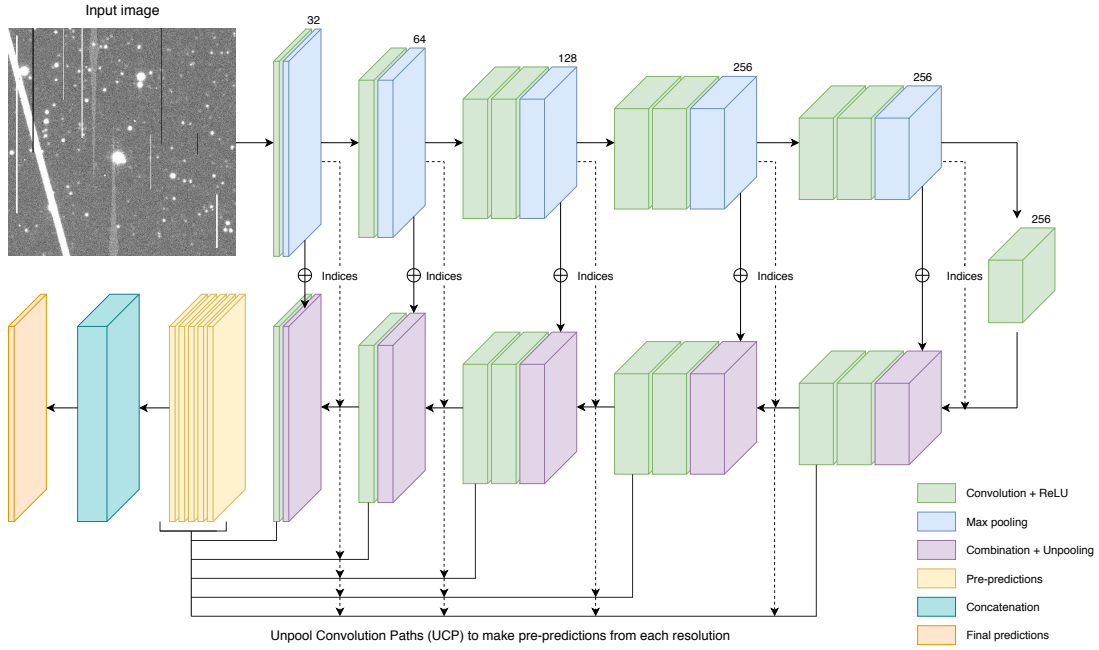


Fig. 10. Scheme representation of the local contaminants neural network architecture.

Table 5. Description of the local contaminants neural network architecture, including map dimensions.

Layer	Size	UCP from each resolution			
Input	$400 \times 400 \times 1$				
Conv	$400 \times 400 \times 32$				
Maxpool	$200 \times 200 \times 32$				
Conv	$200 \times 200 \times 64$				
Maxpool	$100 \times 100 \times 64$				
Conv	$100 \times 100 \times 128$				
Conv	$100 \times 100 \times 128$				
Maxpool	$50 \times 50 \times 128$				
Conv	$50 \times 50 \times 256$				
Conv	$50 \times 50 \times 256$				
Maxpool	$25 \times 25 \times 256$				
Conv	$25 \times 25 \times 256$				
Conv	$25 \times 25 \times 256$				
Maxpool	$13 \times 13 \times 256$				
Conv	$13 \times 13 \times 256$				
Unpooling	$25 \times 25 \times 256$				
Conv	$25 \times 25 \times 256$				
Conv	$25 \times 25 \times 256$	UCP			
Unpooling	$50 \times 50 \times 256$	Idem			
Conv	$50 \times 50 \times 256$	None			
Conv	$50 \times 50 \times 128$	Idem	UCP		
Unpooling	$100 \times 100 \times 128$	Idem	Idem		
Conv	$100 \times 100 \times 128$	None	None		
Conv	$100 \times 100 \times 64$	Idem	Idem	UCP	
Unpooling	$200 \times 200 \times 64$	Idem	Idem	Idem	
Conv	$200 \times 200 \times 32$	Idem	Idem	Idem	UCP
Unpooling	$400 \times 400 \times 32$	Idem	Idem	Idem	Idem
Conv	$400 \times 400 \times 14$	Idem	Idem	Idem	Idem
Concat		$400 \times 400 \times 70$			
Conv		$400 \times 400 \times 14$			

Notes. All convolution kernels are 3×3 and max-pooling kernels are 2×2 . All activation functions (not shown for brevity) are ReLU, except in the output layer where the sigmoid is used.

3.2. Global contaminant neural network architecture

The convolutional neural network that detects global contaminants (tracking errors), MAXITRACK, is a simple network made of convolutional layers followed by max-pooling and fully connected layers. The architecture of the network is schematized in Fig. 11 and detailed in Table 6. Because the two classes are mutually exclusive (affected by tracking errors or not), we adopt for the output layer a softmax activation function and a softmax cross-entropy loss function (Rubinstein 1999). Training is done for 48 epochs on 50 000 images with a mini-batch size of 64 samples, using the Adam optimizer.

4. Results with test data and quality assessment

4.1. Local contaminants neural network

We evaluate the quality of the results in several ways. First, we estimate the performance of the network on test data, both quantitatively through various metrics, and qualitatively. We verify that there is no over-fitting by checking that performance on the test set is comparable to that on the training set. Next, we show that performance is immune to the presence or absence of other contaminants in a given image. We finally compare the performance of the cosmic ray detector to that of a classical algorithm.

4.1.1. Performance metrics

We first estimate classification performance on a benchmark test set comprising 5000 images. Because the network is a binary classifier for every class, we can compute a Receiver Operating Characteristic (ROC) curve for each of them. ROC curves represent the True Positive Rate (TPR) vs. the False Positive Rate (FPR):

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}, \quad (18)$$

$$FPR = \frac{FP}{N} = \frac{FP}{TN + FP}, \quad (19)$$

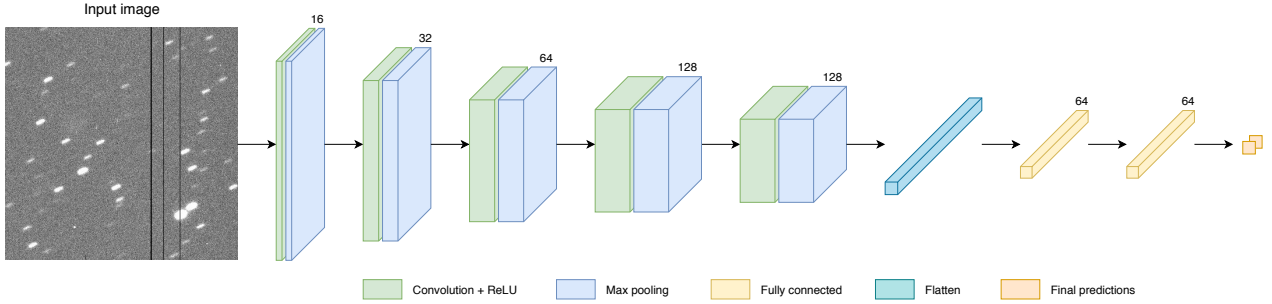


Fig. 11. Scheme representation of the global contaminants neural network architecture.

Table 6. Description of the global contaminant neural network architecture, including map dimensions.

Layer	Size
Input	$400 \times 400 \times 1$
Conv	$400 \times 400 \times 16$
Maxpool	$200 \times 200 \times 16$
Conv	$200 \times 200 \times 32$
Maxpool	$100 \times 100 \times 32$
Conv	$100 \times 100 \times 64$
Maxpool	$50 \times 50 \times 64$
Conv	$50 \times 50 \times 128$
Maxpool	$25 \times 25 \times 128$
Conv	$25 \times 25 \times 128$
Maxpool	$13 \times 13 \times 128$
Flatten	21 632
Fully connected	64
Fully connected	64
Fully connected	2

Notes. All convolution kernels are 9×9 and max-pooling kernels are 2×2 . All activation functions (not shown for brevity) are ReLU, except in the output layer where predictions are done using softmax.

where P is the number of contaminated pixels, TP is the number of true positives (contaminated pixels successfully recovered as contaminated), FN is the number of false negatives (contaminated pixels wrongly classified as non-contaminated), N is the number of non-contaminated pixels, FP is the number of false positives (non-contaminated pixels wrongly classified as contaminated), and TN is the number of true negatives (non-contaminated pixels successfully recovered as non-contaminated).

The accuracy (ACC) is subsequently defined as

$$ACC = \frac{TP + TN}{P + N}. \quad (20)$$

The more the ROC curve bends toward the upper left part of the graph, the better the classifier. However with strongly imbalanced datasets, such as our pixel data, one must be very cautious with the TPR, FPR and ACC values for assessing the quality of the results. For example, if one assumes that there are 1000 pixels of the contaminant class (P) and 159 000 pixels of the background class (N) in a 400×400 pixel sub-image, a TPR of 99% and a FPR of 1%, corresponding to an accuracy of 99%, would actually represent a poor performance, as it would imply 990 true positives, 10 false negatives, 157 410 true negatives, and 1590 false positives. In the end, there would be more false positives FP (pixels wrongly classified as contaminated) than true positives TP.

For this reason the ROC curves in Fig. A.1 are displayed with a logarithmic scale on the FPR axis. We require the FPR to be very low (e.g smaller than 10^{-3}) to consider that the network performs properly.

On the other hand, recovering the exact footprint of large, fuzzy defects is almost impossible at the level of individual pixels, which makes the classification performance for persistence, satellite trails, fringes, nebulosities, spikes and background classes look worse in Fig. A.1 than it really is in practice.

Also, two ROC curves are drawn for cosmic rays and trails. The second one (in green) is computed using only the instances of the class that are above a specific level of the sky background. These instances were defined by retaining those which had more than a half of their pixels above 3σ . These second curves shows that the network performs better on more obvious cases.

In addition to the FPR, TPR, ACC and AUC, we use two other metrics helpful for assessing the network performance: the purity (or precision), representing the fraction of correct predictions among the positively classified samples, and the Matthews correlation coefficient (MCC, Matthews 1975), which is an accuracy measure that takes into account the strong imbalance between classes.

$$PUR = \frac{TP}{TP + FP} = \text{Purity or Precision}, \quad (21)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \quad (22)$$

In the above example, the purity would reach only 38% and the MCC only 61%, highlighting the classifier poor positive class discrimination.

Figure A.3 shows the true positive rate against the purity. Again, the purple curve represents how a random classifier would perform. In these curves the best classifier would sit in the top right ($TPR = 1$ and $PUR = 1$). The darkest points also represent lowest thresholds while the lighter are the highest ones.

Some qualitative results are presented in Fig. 12. A given pixel is assigned a given class if its probability to belong to this class is higher than the best threshold in the sense of the MCC.

Finally, MCCs are represented in Fig. A.2, as a function of the output threshold. In each curve, the threshold giving the best MCC is annotated around the best MCC point. It is important to note that the best threshold depends on the modification of the prior that has been applied to the raw output probabilities. This update of the prior is explained in Sect. 5.

4.1.2. Robustness regarding the context

The MaxiMask neural network is trained using mostly images that include all contaminant classes. Hence, we must check if the network performs equally well independently of the context,

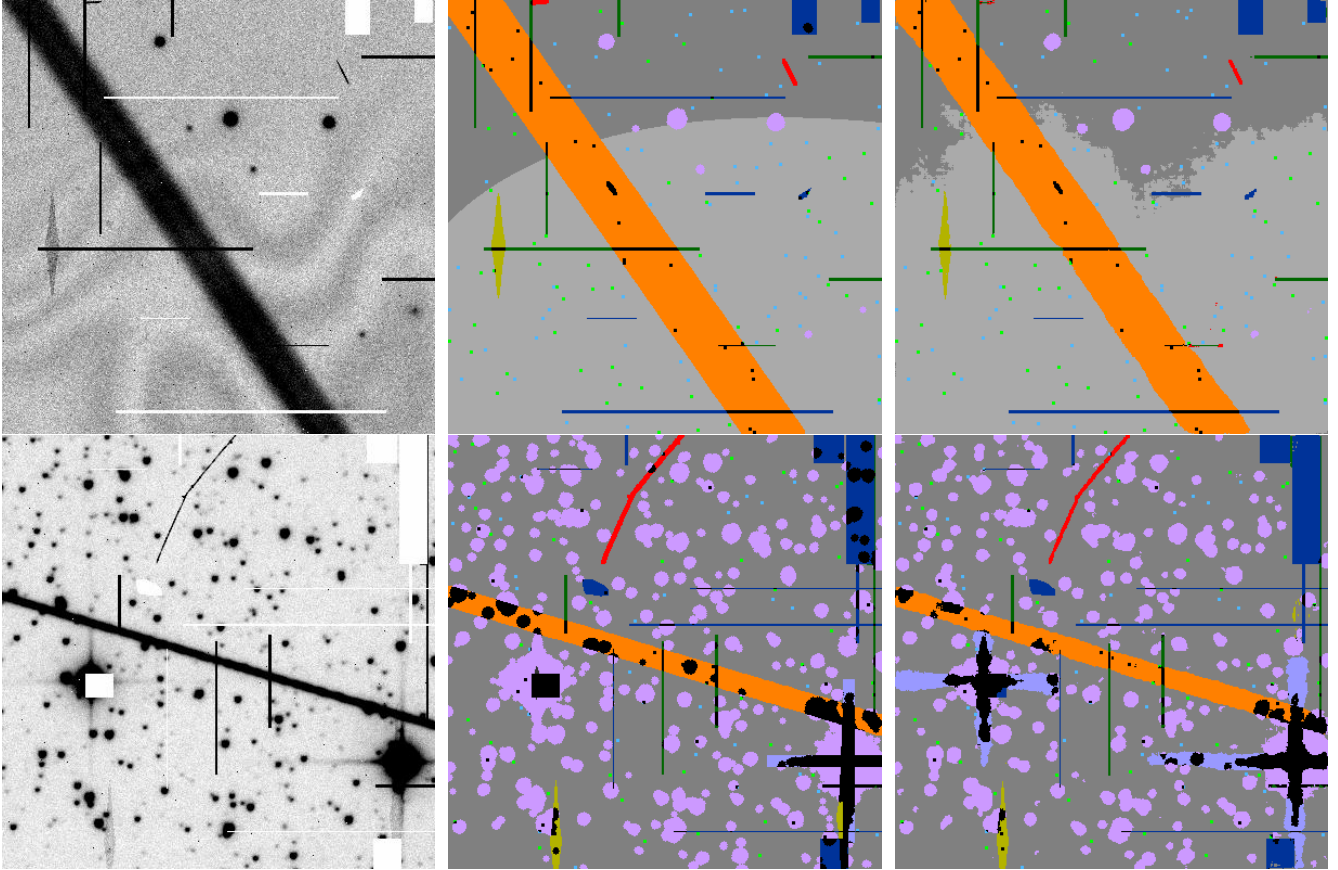


Fig. 12. Examples of qualitative results on test data. *Left:* input; *middle:* ground truth; *right:* predictions. Each class is assigned a color so that the ground truth can be represented in one single image. Class predictions are done according to the threshold giving the highest MC coefficient. The color coding is identical to that of Fig. 8.

that is if it delivers equally good results for images containing, for example, a single class of contaminant.

To this aim, for every contaminant class, we generate a dataset of 1000 images affected only by this type of contaminant (except saturated and background pixels), and another dataset of 1000 images containing only saturated and background pixels. We then compare the performance of MaxiMask for each class with the that obtain on the corresponding dataset. We find that performance (AUC) is similar or even slightly higher for the majority of the classes. This shows that the network is not conditioned to work only in the exact context of the training. The results are presented in Table 7.

As it can be seen, for all classes but fringes and nebulosity, performance improves when a single type of contaminant is present. The slight improvement may come from the fact that ambiguous cases (when pixels are affected by more than one contaminant class, e.g., a cosmic ray or a hot pixel over a satellite trail) are not present in the single contaminant test set.

4.1.3. Cosmic rays: effect of PSF undersampling and comparison with LA Cosmic

Undersampling makes cosmic ray hits harder to distinguish from point-sources. To solve this issue, [van Dokkum \(2001\)](#) has developed LA Cosmic, a method based on a variation of Laplacian edge detection. It is largely insensitive to cosmic ray morphology and PSF sampling. LA Cosmic thus offers an excellent opportunity to test the performance of MaxiMask on undersampled exposures.

Table 7. AUC of each class depending on the test set context.

Class	All contaminant set AUC	Single contaminant set AUC
CR	0.96927	0.98314
HCL	0.99763	0.99957
DCL	0.99872	0.99976
HP	0.99741	0.99965
DP	0.99739	0.99975
P	0.99352	0.99951
TRL	0.99511	0.99813
FR	0.98057	0.93326
NEB	0.97895	0.84575
SAT	0.99965	0.99974
SP	0.96125	0.98061
OV	0.99997	1.00000
BBG	0.98484	0.99165
BG	0.96895	0.98371

To do so, we generate two datasets containing only the cosmic ray contaminant class (plus object and background). A well sampled set of images with FWHMs larger than 2.5 pixels, and an undersampled image set with FWHMs smaller than 2.5 pixels. We run MaxiMask and the Astro-SCRAPPY Python implementation LA Cosmic. To make a fair comparison, LA Cosmic masks are dilated in the same way as the ground truth cosmic ray masks of MaxiMask. However, while MaxiMask

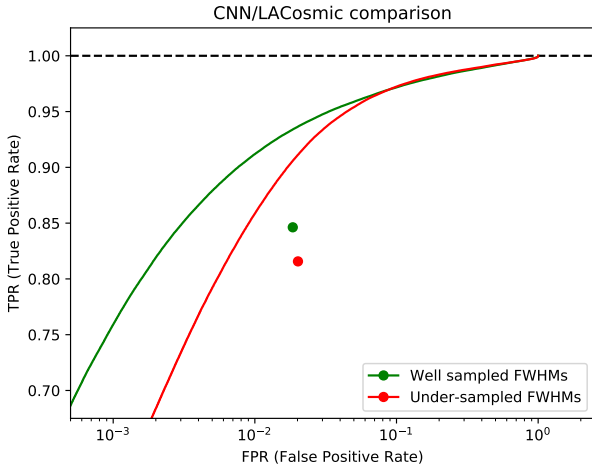


Fig. 13. CR detection performance comparison with LA Cosmic.

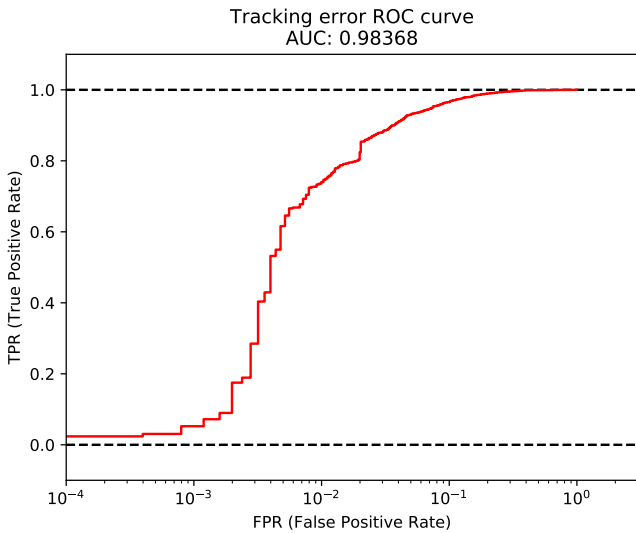


Fig. 14. Global contaminant neural network ROC curve; the steps are a consequence of limited statistics.

generates probability maps that can be thresholded at different levels, LA Cosmic only outputs a binary mask. To compare the results we therefore build ROC curves for the neural network and over-plot a single point representing the result obtained with LA Cosmic.

Figure 13 shows that the neural network performs better than LA Cosmic in both regimes with our data.

4.2. Global contaminants neural network

The ROC curve for the global contaminant neural network is shown in Fig. 14. It is computed from a test set of 5000 images.

5. Modifying priors

If one knows what class proportions are expected in the observation data, output probabilities can be updated to better match these priors (e.g., Saerens et al. 2002; Bailer-Jones et al. 2008).

The outputs of a *perfectly trained* neural network classifier with a cross-entropy loss function can be interpreted as Bayesian posterior probabilities (e.g., Richard & Lippmann 1991; Hampshire & Pearlmutter 1991; Rojas 1996). Under this

assumption and using Bayes' rule, the output for the class ω_c of the trained neural network model defined by a training set T writes:

$$P(\omega_c|\mathbf{x}, T) = \frac{p(\mathbf{x}|\omega_c, T)P(\omega_c|T)}{\sum_{\omega \in \{\omega_c, \bar{\omega}_c\}} p(\mathbf{x}|\omega, T)P(\omega|T)}, \quad (23)$$

where \mathbf{x} is the input image data around the pixel of interest, $p(\mathbf{x}|\omega_c, T)$ is the distribution of \mathbf{x} conditional to class ω_c in the training set T , and $P(\omega_c|T)$ is the prior probability of a pixel to belong to the class ω_c in the trained model.

As each output acts as a binary classifier, the sum is done on the class ω_c (contaminant) and its complementary $\bar{\omega}_c$ (“not the contaminant”).

With the observation data set O we may similarly write:

$$P(\omega_c|\mathbf{x}, O) = \frac{p(\mathbf{x}|\omega_c, O)P(\omega_c|O)}{\sum_{\omega \in \{\omega_c, \bar{\omega}_c\}} p(\mathbf{x}|\omega, O)P(\omega|O)}, \quad (24)$$

where $P(\omega_c|O)$ is the expected fraction of pixels with class ω_c in O .

Now, if the appearance of defects in O matches that in the training set T , we have $p(\mathbf{x}|\omega_c, T) = p(\mathbf{x}|\omega_c, O)$, and we can rewrite (24) as:

$$P(\omega_c|\mathbf{x}, O) = \frac{P(\omega_c|\mathbf{x}, T) \frac{P(\omega_c|O)}{P(\omega_c|T)}}{\sum_{\omega \in \{\omega_c, \bar{\omega}_c\}} P(\omega|\mathbf{x}, T) \frac{P(\omega|O)}{P(\omega|T)}} \quad (25)$$

$$= \frac{1}{1 + \left(\frac{1}{P(\omega_c|\mathbf{x}, T)} - 1 \right) \frac{P(\omega_c|T)}{P(\omega_c|O)} \frac{1 - P(\omega_c|O)}{1 - P(\omega_c|T)}}. \quad (26)$$

If pixels were all weighted equally, the training priors $P(\omega_c|T)$ would simply be the class proportions in the training set. However, this is not the case here, and pixel weights have to be taken into account. To do so, we follow Bailer-Jones et al. (2008)'s approach, by using as an estimator of $P(\omega_c|T)$ the posterior mean on the test set T' (which by construction is distributed identically to the training set):

$$\hat{P}(\omega_c|T) = \frac{1}{\text{card}(T')} \sum_{\mathbf{x} \in T'} P(\omega_c|\mathbf{x}, T'). \quad (27)$$

These corrected probabilities are used to compute the MC coefficient curves in Fig. A.2 (whereas the prior correction does not affect the ROC and purity curves).

MAXIMASK comes with the $P(\omega_c|T)$ values already set, therefore one only needs to specify the expected class proportions in the data, that is the $P(\omega_c|O)$'s.

6. Application to other data

As a sanity check, we apply MAXIMASK to data obtained from different instruments not part of the training set. Examples of the resulting contaminant maps are shown in appendix.

Our first external check is with ZTF (Bellm et al. 2019) data. The MAXIMASK output for a science image featuring a prominent trail with variable amplitude is shown in Fig. A.4. We can note the ability of MAXIMASK to properly flag both the trail and overlapping sources.

Our second external check is with the ACS instrument onboard the *Hubble* Space Telescope (Fig. A.5 and A.6). This test illustrates MAXIMASK's ability to distinguish cosmic rays from poorly sampled, diffraction-limited point source images.

Given the seemingly good performance of MAXIMASK on images from instruments not part of the training set, one question that may arise is whether MAXIMASK can readily be used on production for such instruments, without any retraining or transfer learning. Our limited experience with MAXIMASK seems to indicate that this is indeed the case, although retraining may be beneficial for specific instrumental features. As shown here, excellent performance can be reached by training with 50 000 400×400 images taken from three different instruments. We think that a minimum of 10 000 400×400 would be a good start to train on a single instrument. Assuming CCDs of approximately 2000×2000 pixels, thus containing 25 400×400 images, it would just need 400 CCDs, equivalent to 10 fields for a 40 CCD camera.

Our last series of tests is conducted on digital images of natural scenes (landscape, cat, human face), to check for possible inconsistencies on data that are totally unlike those from the training set. Reassuringly, the maps produced by MAXIMASK are consistent with the expected patterns. For instance, the cat's whiskers are identified as cosmic ray impacts, and pixels with the lowest values as bad pixels.

7. Using MaxiMask and MaxiTrack

MAXIMASK and MAXITRACK are available online³. MAXIMASK is a Python module that infers probability maps from FITS images. It can process a whole mosaic, a specific FITS image extension, or all the FITS files from a directory or a file list. For every FITS file being processed a new FITS image is generated with the same HDU (Header Data Unit) structure as the input. Every input image HDU has a matching contaminant map HDU in output, with one image plane per requested contaminant. The header contains metadata related to the contaminant, including the prior and threshold used. An option can be set to generate a single image plane for all contaminants, using a binary code for each contaminant. Such composite contaminant maps can easily be used as flag maps, for example, in SEXTRACTOR. Based on command line arguments and configuration parameters, one can select specific classes, apply updates to the priors and thresholds to the probability maps. The code relies on the TensorFlow library and can work on both CPUs or GPUs, although the CPU version is expected to be much slower: MAXIMASK processes about 1.2 megapixel per second with an NVidia Titan X GPU, and about 60 times less on a 2.7 GHz Intel i7 dual-core CPU. Yet, there is probably room for improvement in processing efficiency for both the CPU and GPU versions.

MAXITRACK is used the same way as MAXIMASK, except that the output is a text file indicating the probability for the input image(s) to be affected by tracking errors (one probability per extension if the image contains several HDUs). It can also apply an update to the prior. It runs at 60 megapixels s^{-1} with an NVidia Titan X GPU and is 9 times slower on a 2.7 GHz Intel i7 dual-core CPU.

8. Summary and perspectives

We have built a data set and trained convolutional neural network classifiers named MAXIMASK and MAXITRACK to identify contaminants in astronomical images. We have shown that they achieve good performance on test data, both real and simulated. By delivering posterior probabilities, MAXIMASK and MAXITRACK give the user the flexibility to set appropriate

threshold levels and achieve the desired TPR/FPR trade-offs depending on the scientific objectives and requirements. Both classifiers require no input parameters or knowledge of the camera properties.

Even though the mix of contaminants in the training set is unrealistic, being dictated by training requirements, we have checked that this does not impact performance. Output probabilities can be corrected to adapt the behavior of MAXIMASK to any mix of contaminants in the data.

We are aware that several types of contaminants and images are missing from the current version and may be added in the future.

Local contaminants include two particularly prominent classes of contaminants: optical and electronics ghosts. Unwanted reflections within the optics result in stray light in exposures. These reflections can produce spurious images from bright sources commonly referred to as "optical ghosts". Sometimes, reflections from very bright stars outside of the field may also be seen. Detectors read through multiple ports also suffer from a form of electronic ghost known as cross-talk. Electronic cross-talk causes bright sources in one of the CCD quadrants to generate a ghost pattern in other quadrants. The ghosts may be negative or positive and are typically at the level of $1:10^4$. Both effects are a significant source of nuisance in wide field exposures, especially in crowded fields and deep images, where they generate false, transient sources, and can affect high precision astrometric and photometric measurements.

Another category of common issues is defocused or excessively aberrated exposures, as well as trails caused by charge transfer inefficiency, all of which which could easily be implemented in MAXITRACK.

Also, the training set used in the current version of MAXIMASK and MAXITRACK does not include images from space-born telescopes nor, more generally, diffraction-limited imagers. Therefore, they are unlikely to perform optimally with such data, although limited testing indicates that they may remain usable for most features, an example of prediction on HST data is shown in Figs. A.5 and A.6.

Finally, MAXIMASK could be extended to not only detect contaminants, but also to generate an inpainted (i.e., "corrected") version of the damaged image areas wherever possible.

Acknowledgements. M. P. acknowledges financial support from the Centre National d'Etudes Spatiales (CNES) fellowship program. We are grateful to Mike Read, of the Royal Observatory, Edinburgh, for providing us with data from the UKIRT telescope, and to Vincent Lepetit for providing comments and suggestions that helped improve the paper. This research has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 682903, P.I. H. Bouy), and from the French State in the framework of the "Investments for the future" Program, IdEx Bordeaux, reference ANR-10-IDEX-03-02. We gratefully acknowledge the support of NVIDIA Corporation with the donation of one of the Titan Xp GPUs used for this research. This research draws upon data distributed by the NOAO Science Archive. NOAO is operated by the Association of Universities for Research in Astronomy (AURA) under cooperative agreement with the National Science Foundation. Based on observations made with the *Isaac Newton* Telescope operated on the island of La Palma by the Isaac Newton Group in the Spanish Observatorio del Roque de los Muchachos of the Instituto de Astrofísica de Canarias. The *Isaac Newton* Telescope is operated on the island of La Palma by the Isaac Newton Group in the Spanish Observatorio del Roque de los Muchachos of the Instituto de Astrofísica de Canarias. This paper makes use of data obtained from the Isaac Newton Group Archive which is maintained as part of the CASU Astronomical Data Centre at the Institute of Astronomy, Cambridge. Based on data obtained from the ESO Science Archive Facility. This research used the facilities of the Canadian Astronomy Data Centre operated by the National Research Council of Canada with the support of the Canadian Space Agency. Based in part on data collected at Subaru Telescope which is operated by the National Astronomical Observatory of Japan and obtained from the SMOKA,

³ <https://www.github.com/mpaillassa/MaxiMask>

which is operated by the Astronomy Data Center, National Astronomical Observatory of Japan. The Hyper Suprime-Cam (HSC) collaboration includes the astronomical communities of Japan and Taiwan, and Princeton University. The HSC instrumentation and software were developed by the National Astronomical Observatory of Japan (NAOJ), the Kavli Institute for the Physics and Mathematics of the Universe (Kavli IPMU), the University of Tokyo, the High Energy Accelerator Research Organization (KEK), the Academia Sinica Institute for Astronomy and Astrophysics in Taiwan (ASIAA), and Princeton University. Funding was contributed by the FIRST program from Japanese Cabinet Office, the Ministry of Education, Culture, Sports, Science and Technology (MEXT), the Japan Society for the Promotion of Science (JSPS), Japan Science and Technology Agency (JST), the Toray Science Foundation, NAOJ, Kavli IPMU, KEK, ASIAA, and Princeton University. This paper makes use of software developed for the Large Synoptic Survey Telescope. We thank the LSST Project for making their code available as free software at <http://dm.lsst.org>. The Pan-STARRS1 Surveys (PS1) have been made possible through contributions of the Institute for Astronomy, the University of Hawaii, the Pan-STARRS Project Office, the Max-Planck Society and its participating institutes, the Max Planck Institute for Astronomy, Heidelberg and the Max Planck Institute for Extraterrestrial Physics, Garching, The Johns Hopkins University, Durham University, the University of Edinburgh, Queen's University Belfast, the Harvard-Smithsonian Center for Astrophysics, the Las Cumbres Observatory Global Telescope Network Incorporated, the National Central University of Taiwan, the Space Telescope Science Institute, the National Aeronautics and Space Administration under Grant No. NNX08AR22G issued through the Planetary Science Division of the NASA Science Mission Directorate, the National Science Foundation under Grant No. AST-1238877, the University of Maryland, and Eotvos Lorand University (ELTE) and the Los Alamos National Laboratory. Based on data collected at the Subaru Telescope and retrieved from the HSC data archive system, which is operated by Subaru Telescope and Astronomy Data Center at National Astronomical Observatory of Japan. This paper includes data gathered with the Swope telescope located at Las Campanas Observatory, Chile. Based on observations obtained with MegaPrime/MegaCam, a joint project of CFHT and CEA/DAPNIA, at the Canada-France-Hawaii Telescope (CFHT) which is operated by the National Research Council (NRC) of Canada, the Institut National des Sciences de l'Univers of the Centre National de la Recherche Scientifique (CNRS) of France, and the University of Hawaii. This research has made use of NASA's Astrophysics Data System Bibliographic Services. This research made use of Astropy, a community-developed core Python package for Astronomy (Astropy Collaboration, 2013, <http://dx.doi.org/10.1051/0004-6361/201322068>). The *Herschel* spacecraft was designed, built, tested, and launched under a contract to ESA managed by the *Herschel*/Planck Project team by an industrial consortium under the overall responsibility of the prime contractor Thales Alenia Space (Cannes), and including Astrium (Friedrichshafen) responsible for the payload module and for system testing at spacecraft level, Thales Alenia Space (Turin) responsible for the service module, and Astrium (Toulouse) responsible for the telescope, with in excess of a hundred subcontractors

References

- Abadi, M., Barham, P., Chen, J., et al. 2016, in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*, 16, 265
- Autry, R. G., Probst, R. G., Starr, B. M., et al. 2003, in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, eds. M. Iye, & A. F. M. Moorwood, 4841, 525
- Badrinarayanan, V., Kendall, A., & Cipolla, R. 2015, ArXiv e-prints [arXiv:1511.00561]
- Badrinarayanan, V., Kendall, A., & Cipolla, R. 2017, *IEEE Trans. Pattern Anal. Mach. Intell.*, 39, 2481
- Bailer-Jones, C. A., Smith, K., Tiede, C., Sordo, R., & Vallenari, A. 2008, *MNRAS*, 391, 1838
- Beketešević, D., Vinković, D., Rasmussen, A., & Ivezić, Ž. 2018, *MNRAS*, 474, 4837
- Bellm, E. C., Kulkarni, S. R., Graham, M. J., et al. 2019, *PASP*, 131, 018002
- Bertin, E. 2009, *Mem. Soc. Astron. It.*, 80, 422
- Bertin, E. 2013, *Astrophysics Source Code Library* [record ascl:1301.001]
- Bertin, E., & Arnouts, S. 1996, *A&AS*, 117, 393
- Bosch, J., Armstrong, R., Bickerton, S., et al. 2018, *PASJ*, 70, S5
- Boulade, O., Charlot, X., Abbon, P., et al. 2003, in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, eds. M. Iye, & A. F. M. Moorwood, 4841, 72
- Bouy, H., Bertin, E., Moraux, E., et al. 2013, *A&A*, 554, A101
- Casali, M., Adamson, A., Alves de Oliveira, C., et al. 2007, *A&A*, 467, 777
- Cuillandre, J. C., Luppino, G. A., Starr, B. M., & Isani, S. 2000, in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, eds. M. Iye, & A. F. Moorwood, 4008, 1010
- Dalton, G. B., Caldwell, M., Ward, A. K., et al. 2006, *Proc. SPIE*, 6269, 62690X
- Flaugher, B. L., Abbott, T. M. C., Annis, J., et al. 2010, in *Ground-based and Airborne Instrumentation for Astronomy III*, Proc. SPIE, 7735, 77350D
- García-García, A., Orts-Escobedo, S., Oprea, S., Villena-Martínez, V., & García-Rodríguez, J. 2017, ArXiv e-prints [arXiv:1704.06857]
- Griffin, M. J., Abergel, A., Abreu, A., et al. 2010, *A&A*, 518, L3
- Hampshire, II, J. B., & Pearlmutter, B. 1991, *Connectionist Models* (Elsevier), 159
- Ienaka, N., Kawara, K., Matsuoka, Y., et al. 2013, *ApJ*, 767, 80
- Ives, D. 1998, *IEEE Spectrum*, 16, 20
- Kawanomoto, S., Komiyama, Y., & Yagi, M. 2016a, in *Subaru Users' Meeting FY2016*
- Kawanomoto, Y., Yagi, M., & Kawanomoto, S. 2016b, in *Subaru Users' Meeting FY2016*
- Kingma, D. P., & Ba, J. 2014, ArXiv e-prints [arXiv:1412.6980]
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. 2012, in *Advances in Neural Information Processing Systems*, 1097
- Kuijken, K., Bender, R., Cappellaro, E., et al. 2002, *The Messenger*, 110, 15
- LeCun, Y., & Bengio, Y. 1995, *The Handbook of Brain Theory and Neural Networks* (Cambridge: MIT Press), 3361
- Long, J., Shelhamer, E., & Darrell, T. 2015, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3431
- Long, K. S., Baggett, S. M., & MacKenty, J. W. 2015, *Persistence in the WFC3 IR Detector: an Improved Model Incorporating the Effects of Exposure Time*, Tech. rep.
- Lowe, D. G. 1999, *ICCV'99: Proceedings of the International Conference on Computer Vision*, 1150
- Magnier, E. A., & Cuillandre, J.-C. 2004, *PASP*, 116, 449
- Mathews, B. W. 1975, *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405, 442
- McCully, C., Crawford, S., Kovacs, G., et al. 2018, <https://doi.org/10.5281/zenodo.1482019>
- Melchior, P., Sheldon, E., Drlica-Wagner, A., et al. 2016, *Astron. Comput.*, 16, 99
- Metzger, M. R., Luppino, G. A., & Miyazaki, S. 1995, *Bull. Am. Astron. Soc.*, 27, 1389
- Miville-Deschênes, M.-A., Duc, P.-A., Marleau, F., et al. 2016, *A&A*, 593, A4
- Miyazaki, S., Komiyama, Y., Kawanomoto, S., et al. 2018, *PASJ*, 70, S1
- Morganson, E., Gruendl, R. A., Menanteau, F., et al. 2018, *PASP*, 130, 074501
- Nir, G., Zackay, B., & Ofek, E. O. 2018, *AJ*, 156, 229
- Ordénovic, C., Surace, C., Torrèani, B., & Llèbaria, A. 2008, *Stat. Methodol.*, 5, 373
- Pilbratt, G. L., Riedinger, J. R., Passvogel, T., et al. 2010, *A&A*, 518, L1
- Rheault, J. P., Mondrik, N. P., DePoy, D. L., Marshall, J. L., & Suntzeff, N. B. 2014, *Spectrophotometric Calibration of the Swope and duPont Telescopes for the Carnegie Supernova Project 2*
- Richard, M. D., & Lippmann, R. P. 1991, *Neural Comput.*, 3, 461
- Rojas, R. 1996, *Neural Comput.*, 8, 41
- Rubinstein, R. 1999, *Methodol. Comput. Appl. Probab.*, 1, 127
- Ruder, S. 2016, ArXiv e-prints [arXiv:1609.04747]
- Saerens, M., Latinne, P., & Decaestecker, C. 2002, *Neural Comput.*, 14, 21
- Simonyan, K., & Zisserman, A. 2014, ArXiv e-prints [arXiv:1409.1556]
- Szegedy, C., Liu, W., Jia, Y., et al. 2015, ArXiv e-prints [arXiv:1409.4842]
- Valdes, F., Gruendl, R., & DES Project 2014, in *Astronomical Data Analysis Software and Systems XXIII*, eds. N. Manset, & P. Forshay, *ASP Conf. Ser.*, 485, 379
- van Dokkum, P. G. 2001, *PASP*, 113, 1420
- Vandame, B. 2002, in *Astronomical Data Analysis II*, eds. J. L. Starck, & F. D. Murtagh, *SPIE Conf. Ser.*, 4847, 123
- Williams, C. K. I. 1998, in *Prediction with Gaussian Processes: From Linear Regression to Linear Prediction and Beyond*, ed. M. I. Jordan (Dordrecht: Springer), 599
- Wolfe, T., Armandroff, T., Blouke, M. M., et al. 2000, in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, eds. M. M. Blouke, N. Sampat, G. M. Williams, & T. Yeh, 3965, 80
- Yang, T., Wu, Y., Zhao, J., & Guan, L. 2018, *Cognit. Syst. Res.*, 53, 20

Appendix A: Performance metric curves and qualitative tests

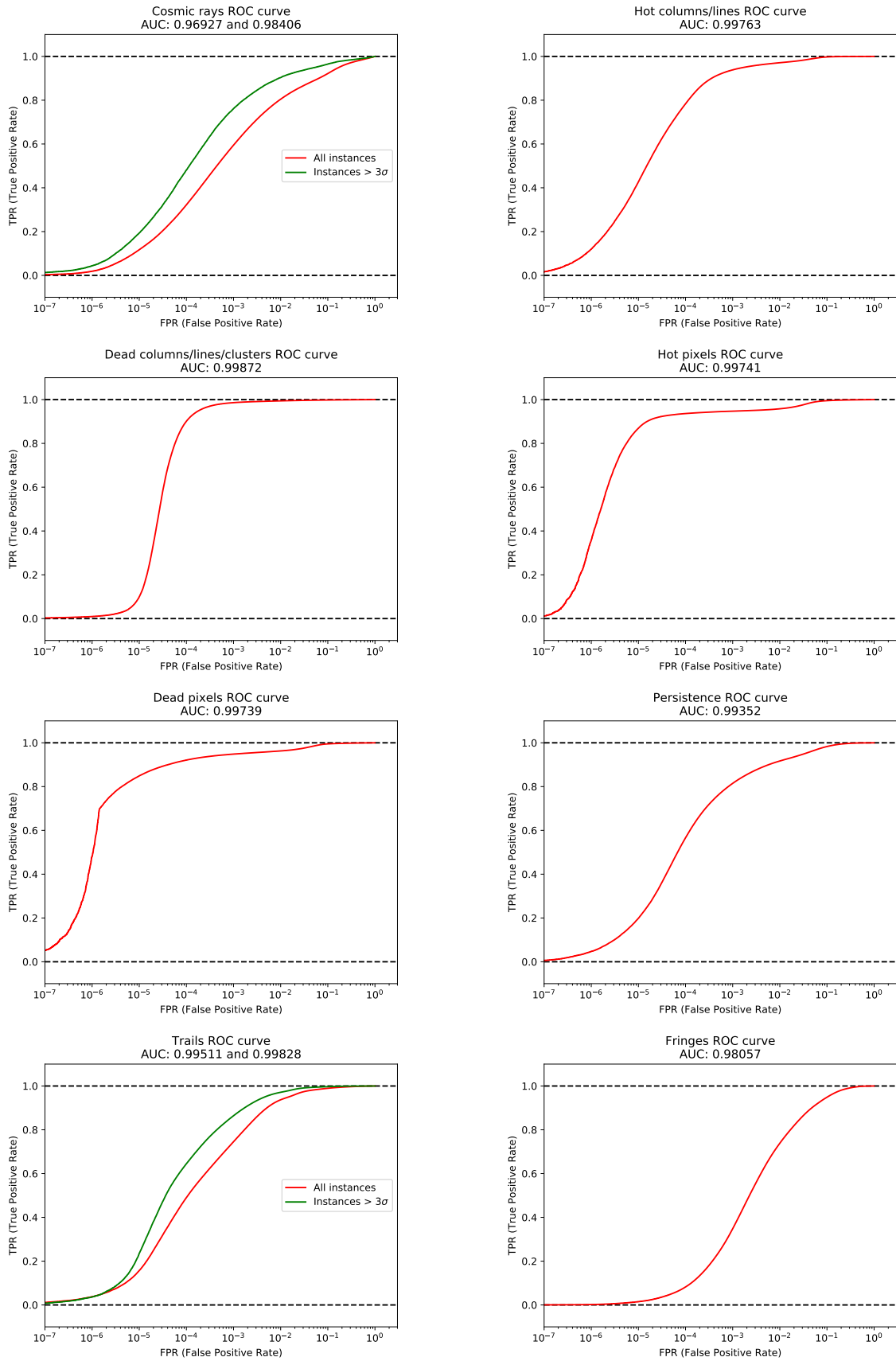


Fig. A.1. ROC curves: TPR vs. FPR. The FPR axis is in logarithmic scale so that very low FPR are best visualized. The ROC curve and the AUC are provided for each class.

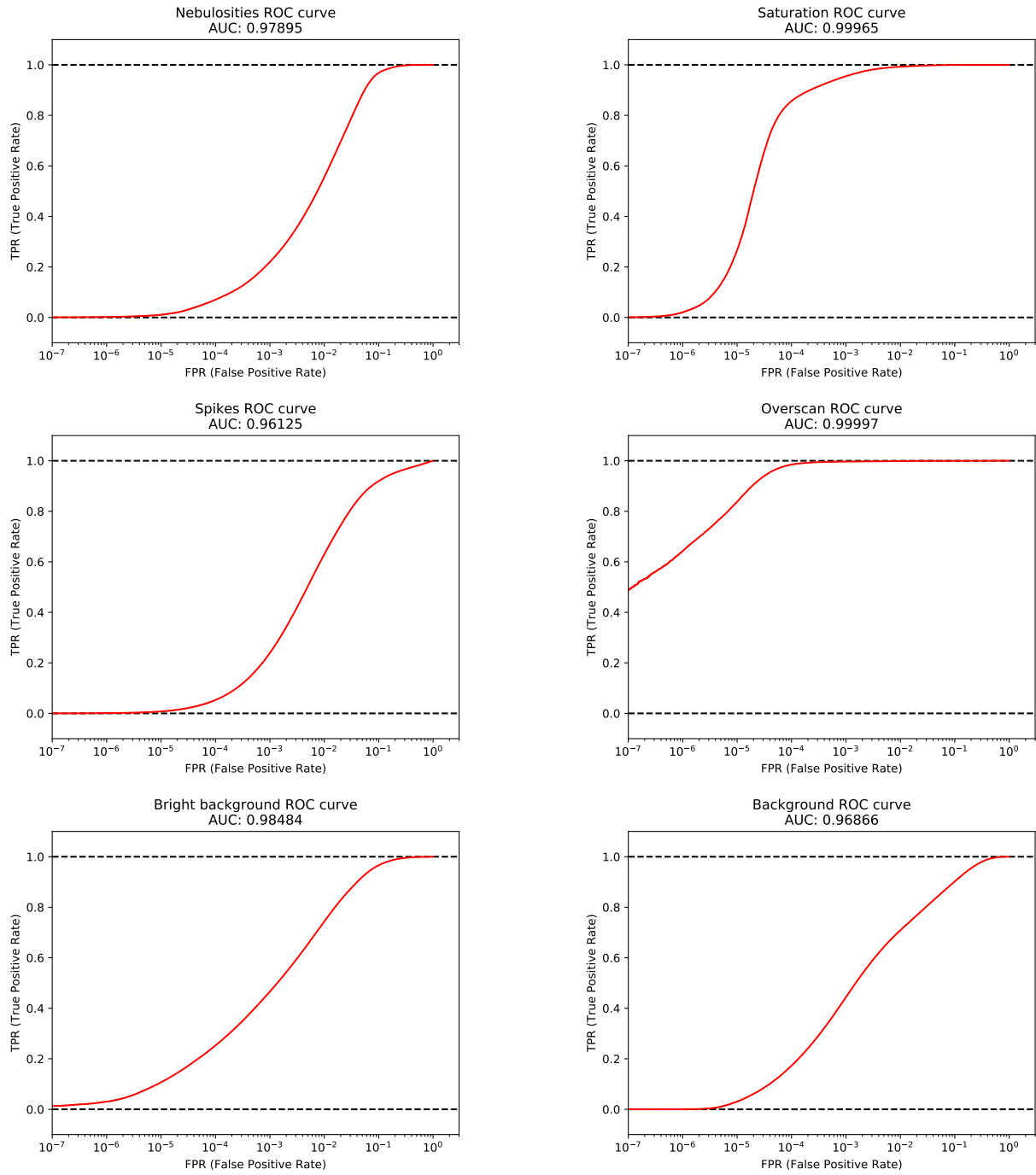


Fig. A.1. continued.

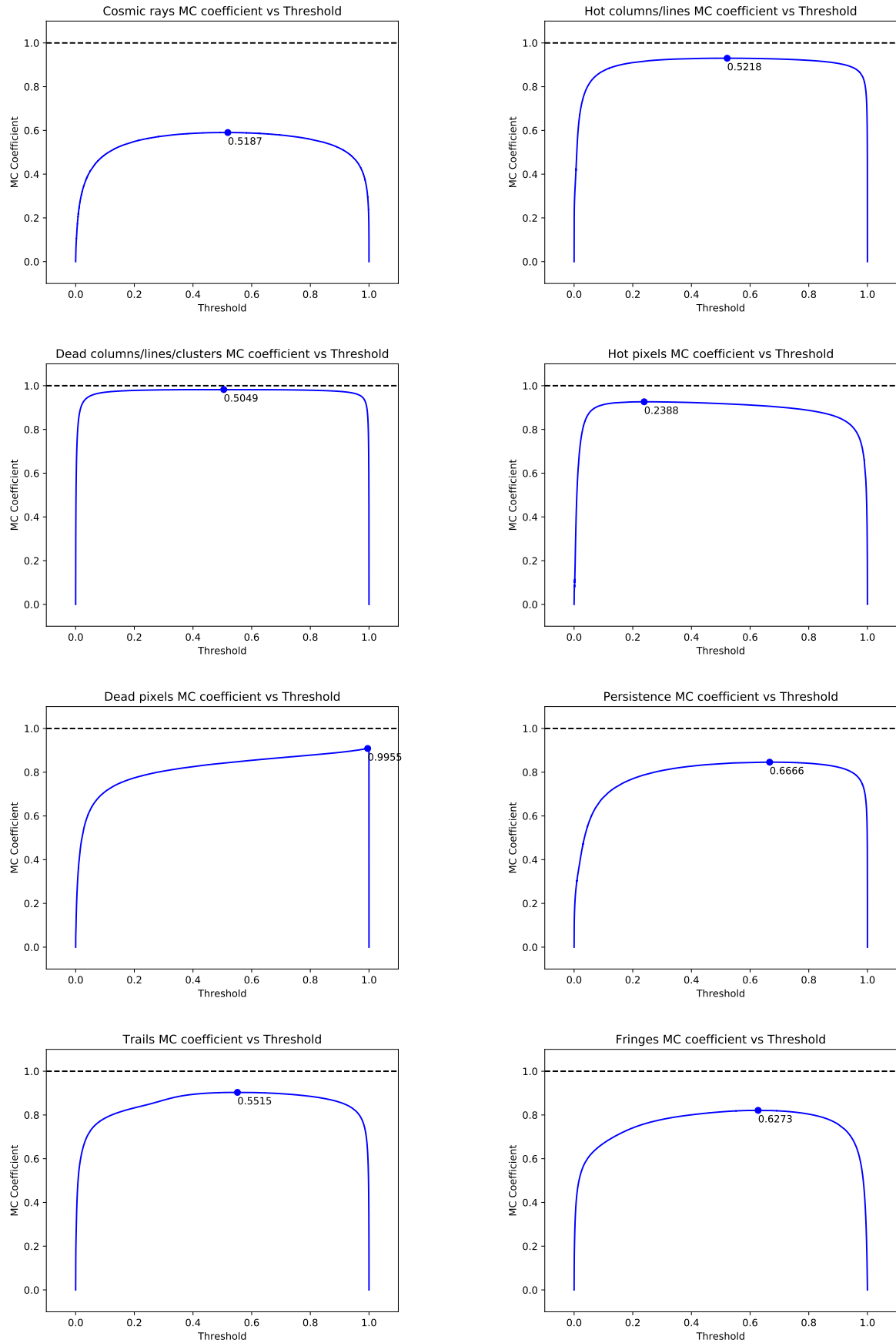


Fig. A.2. MC coefficient curves: MC coefficient vs. detection threshold. On each curve is annotated the threshold for which the MC coefficient is the highest. These curves were computed using the probabilities corrected from priors using empirical training priors.

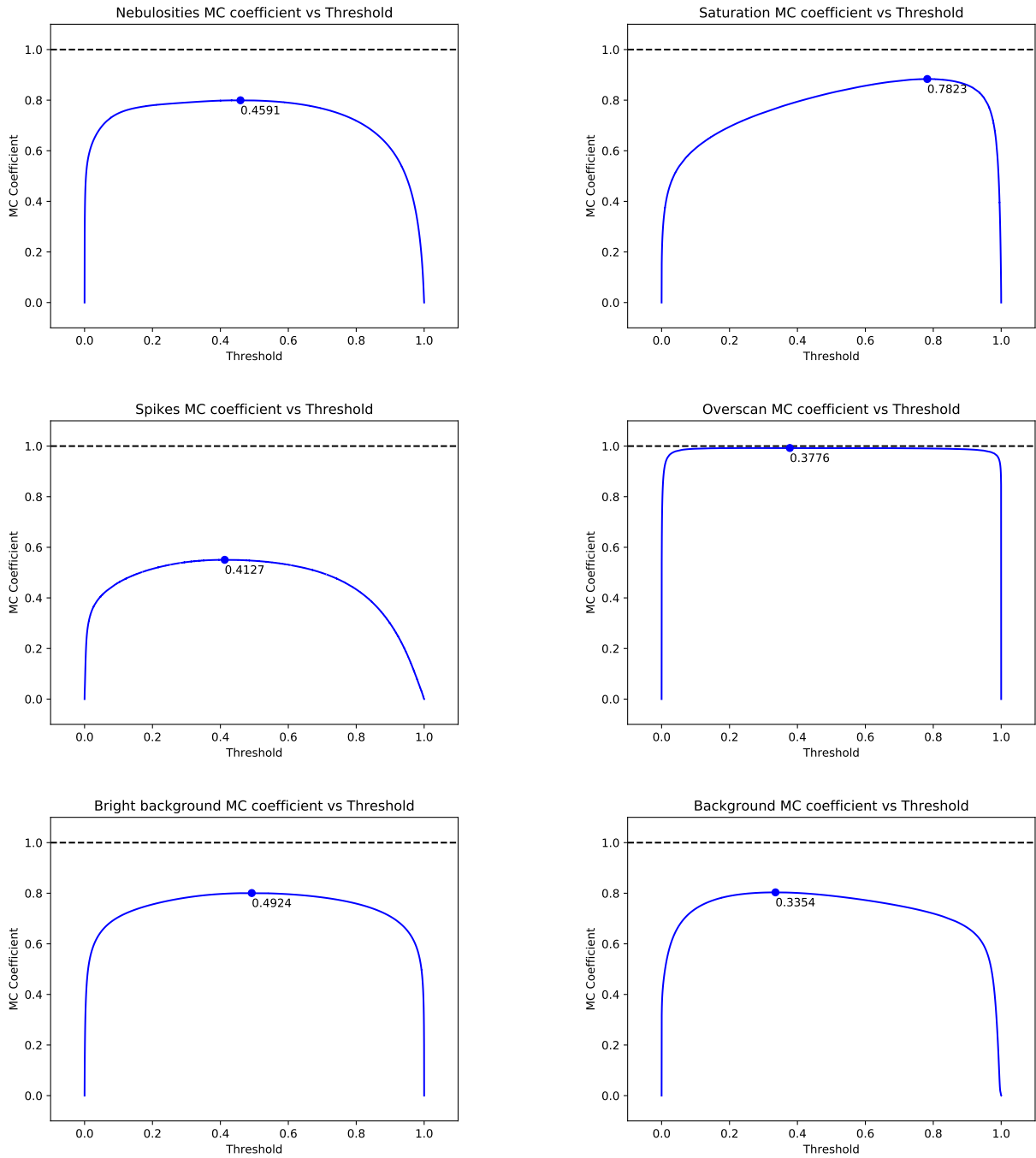


Fig. A.2. continued.

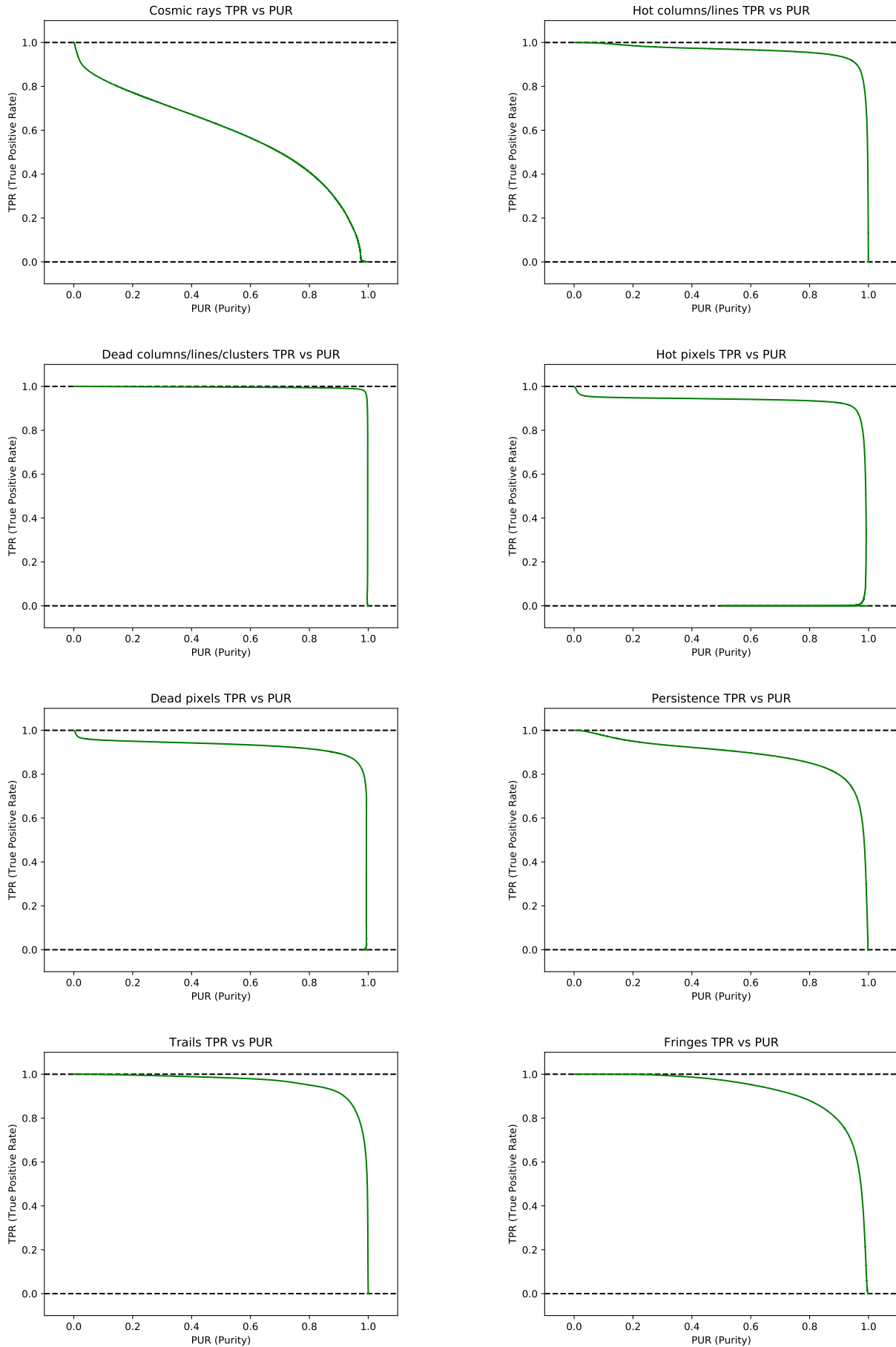


Fig. A.3. Purity curves: TPR vs. PUR.

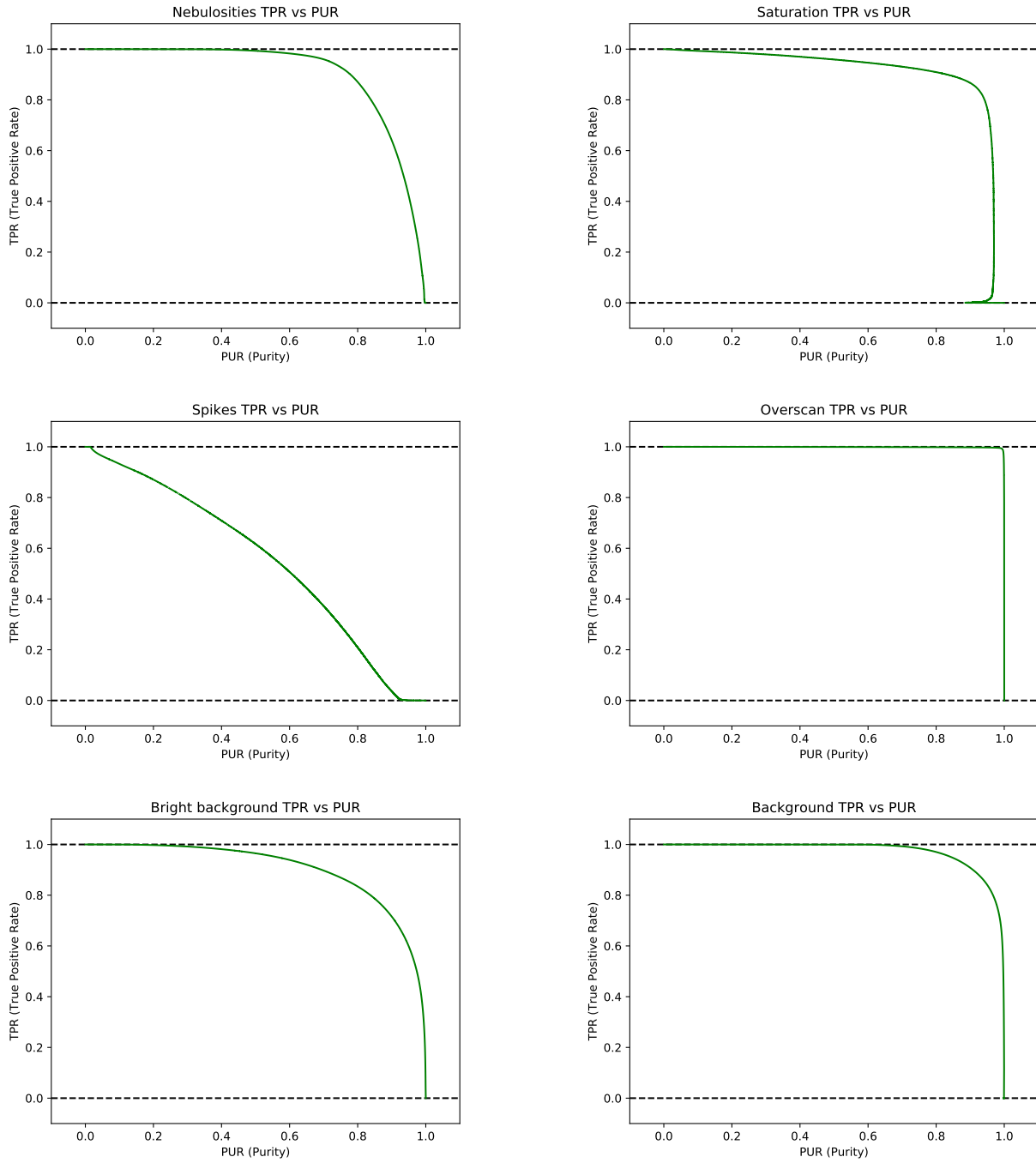


Fig. A.3. continued.

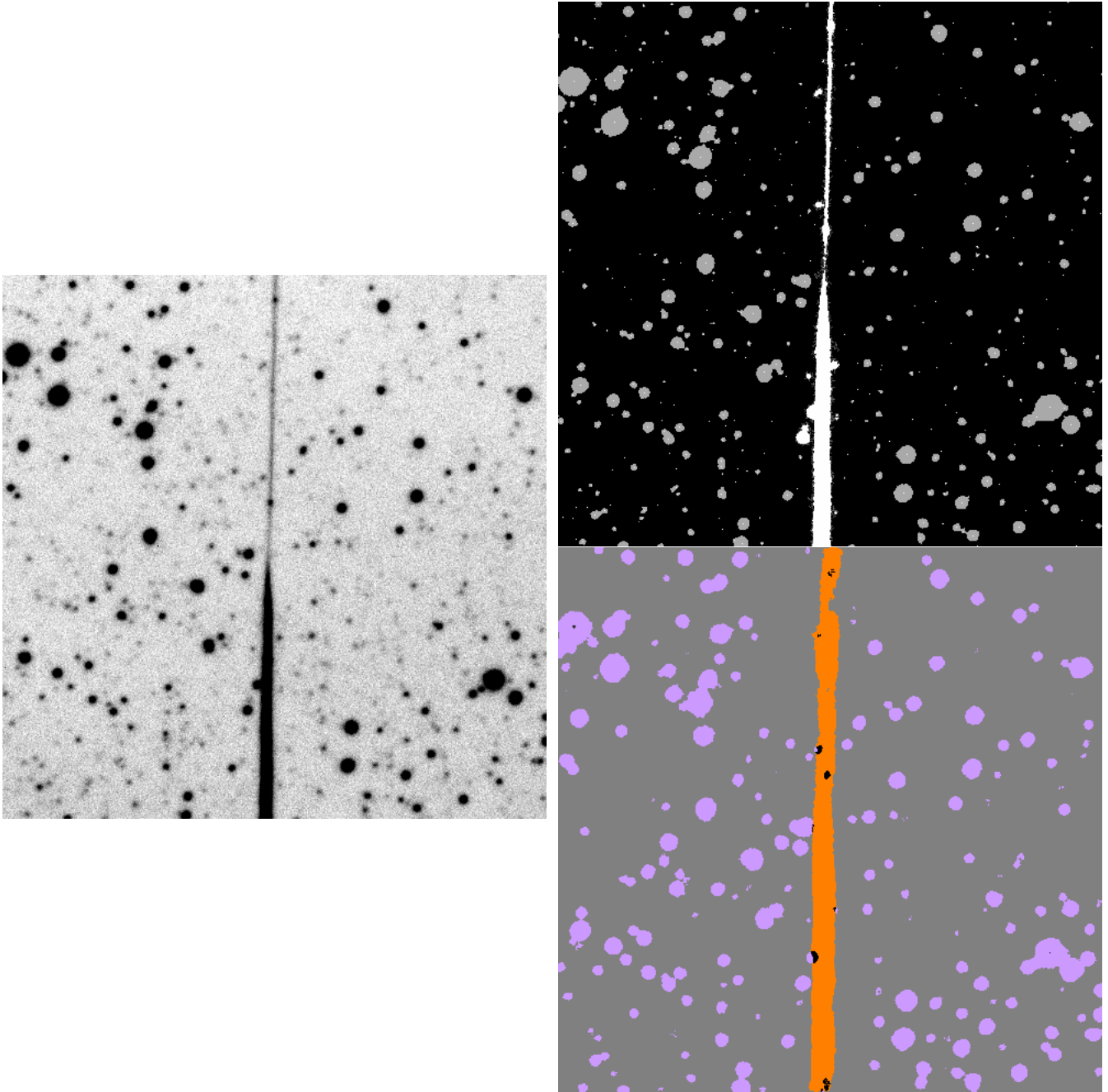


Fig. A.4. Prediction example for an instrument not used in training: ZTF (Bellm et al. 2019). *Left:* a science image exposure. *Top right:* mask from the ZTF pipeline. *Bottom right:* flagging by MAXIMASK; the trail is correctly recovered. Also, MAXIMASK CNN is able to correctly flag pixels where the trail overlaps sources whereas in the ZTF pipeline, all pixels (i.e., pixels belonging only to the trail, pixels belonging only to sources, and pixels belonging to both the trail and sources) are flagged as both trail and source.

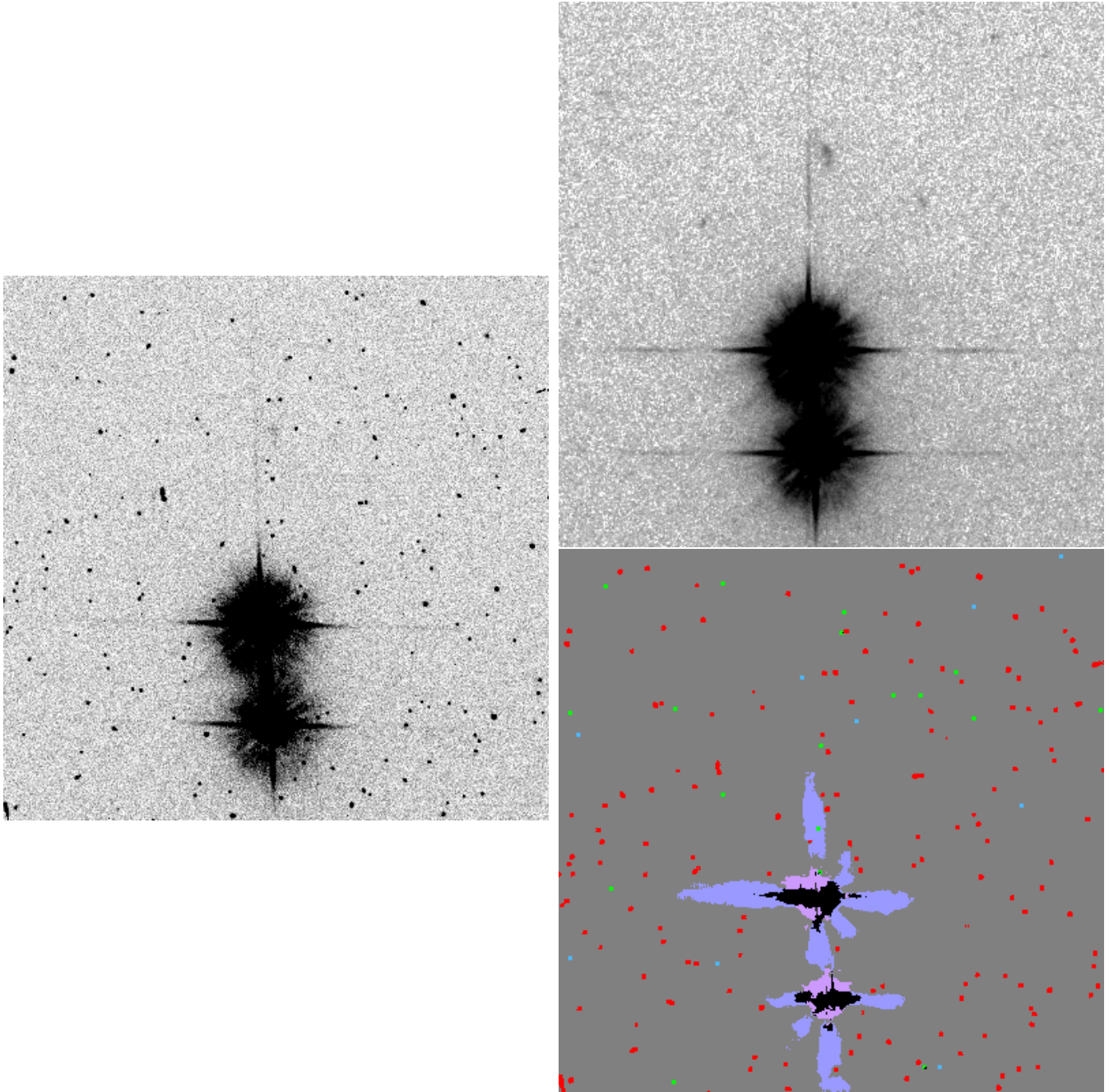


Fig. A.5. Example of a prediction for a space instrument (HST) not used in training (ACS exposure). *Left:* a calibrated (flat-fielded, CTE-corrected) individual exposure of a stellar field in the Pleiades. *Top right:* fully calibrated, geometrically-corrected, dither-combined image where cosmic rays and artifacts have been removed. *Bottom right:* MAXIMASK contaminant identification. Each class is assigned a color so that the ground truth can be represented as a single image (red: CR, dark green: HCL, dark blue: BCL, green: HP, blue: BP, yellow: P, orange: TRL, gray: FR, light gray: NEB, purple: SAT, light purple: SP, brown: OV, pink: BBG, dark gray: BG). Pixels that belong to several classes are represented in black. For the sake of visualization, hot and dead pixel masks have been morphologically dilated so that they appear as 3×3 pixel areas in this representation.

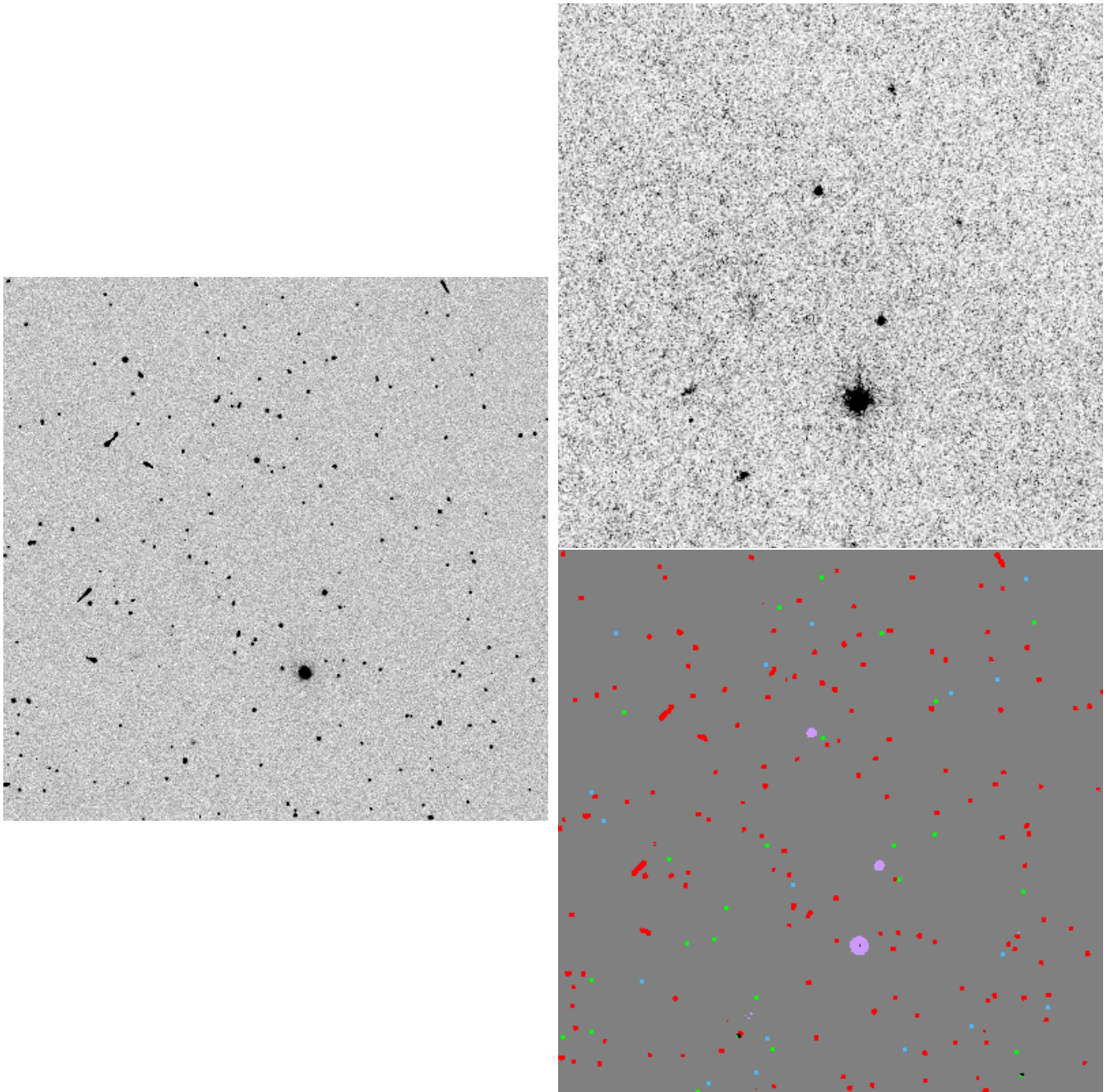


Fig. A.6. Same as Fig. A.5 at a different location in the field to illustrate the ability of MAXIMASK to differentiate poorly sampled stellar images from cosmic rays.