Super-resolution in astronomical images: enhancing object detection

Irene Pintos Castro
USC / CEFCA
ipintoscastro@gmail.com

Manuel Fernández Delgado
USC
(Supervisor)

Andrés del Pino
CEFCA
(Supervisor)

***Abstract*—Using paired images of JAST/T80 Telescope in OAJ and of Hubble Space Telescope we aim at increasing the resolution of JAST/T80 images with the objective of enhancing object detection. Higher resolution versions of JAST/T80 images in crowded areas of the sky will significantly increase the number of stars and/or galaxies detected, enabling a better measurement of their photometry. We pre-processed both low-resolution and high-resolution images to build a dataset of paired crops, which is used to train the Real-ESRGAN neural network. We are able to get super-resolved images with a 4× scale and a FSIM=0.8903, showing an improvement in the source detection in crowded areas. However, visual inspection of bright stars and nearby galaxies reveals that there is room for learning a better model that can reproduce the PSF and the details of the extended objects.**

I. Introduction

Telescopes are the instruments we use to observe our Universe. The cameras installed on them provide us with the data needed for its study: images and spectra. Astronomical observations from Earth are always limited by the presence of the atmosphere. While space telescopes avoid atmospheric aberrations, they are an expensive solution and will still have technical limitations, as the point spread function (PSF) consequence of the finite aperture and light dispersion within the telescope.

An image shows the brightness of an object in the spatial domain, i.e., how many photons are coming from a specific location in space. Three properties of an image – size, brightness, and resolution – are the most important from a scientific point of view. From size, we learn about astronomical scales. From brightness, we learn the amount of energy that an object is producing, in order to investigate how it is producing that energy. The ability to distinguish one location from a nearby location is called spatial resolution. Higher resolution lets us know things like whether or not two galaxies are merging or if there are two stars close to each other instead of one star by itself. Astronomers aim at having the largest possible images with the highest possible resolution.

In addition to their large size, astronomical images have special characteristics: they are commonly stored in Flexible Image Transport System (FITS) digital file format; they have high range of grey values (32-bit float); and they are usually quite noisy. Though images are grey-scale, they contain some colour information because they are taken through a colour filter. Telescopes, instruments and detectors have different sensitivities to different wavelengths and usually work with a well-defined set of filters. These filters can have a variety of widths, from broad-band to narrow-band ($\sim$1000Å to few 10Å). Finally, one particularly important aspect of astronomical image processing is the choice of the stretch function. The representation of the pixel values that one selects has a high impact on the number and shape of the objects that we see in the image. For instance, a logarithmic representation tends to suppress the bright parts of the image and to enhance the fainter ones. This can be desirable though it will reduce the contrast, producing a lower dynamic range.

Super-resolution describes a class of methods that can upscale video or images from lower resolutions to higher ones. Such methods have been successfully demonstrated on astronomical imaging (e.g. [1]–[4]). Traditionally, interpolation methods such as bilinear and nearest neighbour interpolation are used for upscaling. However, these methods often introduce side effects such as noise amplification and blurring. If the PSF can be well determined, deconvolution techniques can be applied to removing the effect of the instrumentation and increasing the image resolution [3], [5]. However, obtaining the PSF is a very difficult task [6].

Over the past decade, deep learning approaches have been introduced to solve a variety of astronomical problems: denoising [7], segmentation [8], galaxy classification [9], cosmic web simulations [10], etc. Different neural networks dedicated to super-resolution tasks have shown promising results in natural image restoration, e.g., [11], [12] . These techniques, applied to ground-based astronomical images, can help to overcome instrumental limitations (e.g., large pixel scale) and the presence of the atmosphere of observations from observatories like the Observatorio Astrofísico de Javalambre (OAJ).

In this work, we tackle the problem of increasing the resolution of large pixel-scale ground-based astronomical images. We aim at generating super-resolved versions of images taken with the 83 cm Javalambre Auxiliary Survey Telescope (JAST80) and its T80Cam camera, with the focus on enhancing source detection. Higher resolution images should allow us to better disentangle close sources in crowded regions of the sky (e.g., globular clusters or the core of distant galaxy clusters), thus increasing the number of objects detected and enabling a better measurement of their brightness (photometry).

II. Data

Most works we find in the literature addressing astronomical image restoration tasks use simulated data instead of real data to train their models, e.g. [2], [4], and usually, the reason is the lack of target images. However, after decades of space astronomy, there are loads of publicly available high-quality images that we expect can be used as targets. Below we describe in more detail the data selected to achieve our objectives.

A. J-PLUS data

J-PLUS [13] is a survey being conducted at the Observatorio Astrofísico de Javalambre (OAJ) using the 83 cm Javalambre Auxiliary Survey Telescope (JAST80) and T80Cam, a panoramic camera of $9.2k \times 9.2k$ pixels that provides a 2 deg^2 field of view, with a pixel scale of 0.55 arsec pix^{-1}. The J-PLUS filter system is composed of seven mediumband and five broadband filters spanning the full optical range (3500 - 10000 Å). This work started making use of the second data release (DR2) of J-PLUS and was extended to the internal DR3[1]. It covers more than 3000 square degrees, with 1642 (co-added) individual images of 2 deg^2. To create the paired image sample we exploit in this work, we run queries using the Astronomical Data Query Language (ADQL) interface and the Image Search services of the J-PLUS database [14].

J-PLUSsurvey strategy uses the rSDSS filter as the reference to perform source detection, therefore rSDSS images are what we aim at super resolve to improve object detection as the initial objective.

B. ACS/HST data

The Advanced Camera for Surveys (ACS) is a third-generation Hubble Space Telescope (HST) instrument. It employs two fundamentally different types of detectors: Charge-coupled device (CCD) for use from the near-UV to the near-IR, and a Multi-Anode Microchannel Array detector (MAMA) for use in the UV. ACS/HST has three channels: the Wide Field Channel (WFC), the High Resolution Channel (HRC), and the Solar Blind Channel (SBC). Installed in March 2002, its CCD Electronics Box and Low Voltage Power Supply incrementally failed in June 2006 and January 2007. The replacement components installed on May 2009, successfully restored the function of the WFC. This WFC has a field of view of 0.056×0.056 deg^2 and a pixel scale of $\sim$0.05 arcsec pix^{-1}. The format of the WFC images is $2 \times 2048 \times 4096$ pixels, which means two rectangles with a gap between them. Its filter system consist of nine broadband and three narrowband filters.

To obtain ACS/HST images we use the Astroquery python package, that has a module to query the Barbara A. Mikulski Archive for Space Telescopes (MAST) [15]. From all the data products available in MAST, we get the `_drc` and `_drz` images: calibrated, geometrically-corrected, dither-combined images created by AstroDrizzle.

[1]the early internal data release took place on 10th June 2022

From the nine broadband filters of ACS/HST, F625W is the most similar, i.e. covers a comparable wavelength range, to the rSDSS filter from J-PLUS. The F606W filter has a broader coverage and fully includes the rSDSS filter, but goes further into the blue. In Fig. 1 we show the transmission profiles of these three filters.

C. Paired images

J-PLUS and ACS/HST archives have a huge amount of data, including both images and catalogues. However, the number of overlapping fields in the sky is limited, for example, compared to J-PLUS, ACS/HST has only surveyed a small fraction of the sky. In addition, not all ACS/HST fields have been observed with all filters, specifically, our target filter F625W is not the most common one and has limited amount of data. The methodology we follow to gather the maximum number of paired J-PLUS and ACS/HST images is:

- We download the J-PLUS catalogue that includes the information about all the rSDSS *tiles* (co-added images of 2 deg^2).
- From the central position (RA,DEC) of the J-PLUS tile and with a search box of width 0.8 deg, we look in the ACS/HST MAST archive for the science F625W `_drc` and `_drz` images available.
- When a J-PLUS tile has matching ACS/HST images we download both from their respective archives.
- We visually inspect paired images for quality control. We discard paired images that are not well aligned (usually due to ACS/HST bad quality astrometry), ACS/HST images that are too noisy or that the cosmic ray correction was not (correctly) applied, and ACS/HST images that are not fully covered in the J-PLUS tile. As a result 24 ACS/HST images overlapping with 17 J-PLUS tiles are selected, while ACS/HST images from other 30 J-PLUS tiles are rejected.

D. Data pre-processing

As mentioned before, the pixel value representation plays a fundamental rule in how the information contained in the image is displayed (e.g., reduce noise or boost faint objects). In astronomical images most of the information is concentrated on a small range of counts, i.e., a field may have a few very bright stars with thousands of counts per pixel while the rest of the sources in that area of the sky would be in the range of tens to hundreds of counts. Thus, the pre-processing tasks on the image are a sensitive step. The pre-processing we apply is as follows:

- Only for ACS/HST images, we resize them to match $4\times$ the pixel scale of J-PLUS images.
- We scale the image using a linear, a logarithmic and a square root function. See Sect. IV-B for more details about the final scaling function.
- We clip the highest and lowest values of the image. Based on our observations of crowded fields, particularly the centre of a globular cluster and a nearby elliptical galaxy,

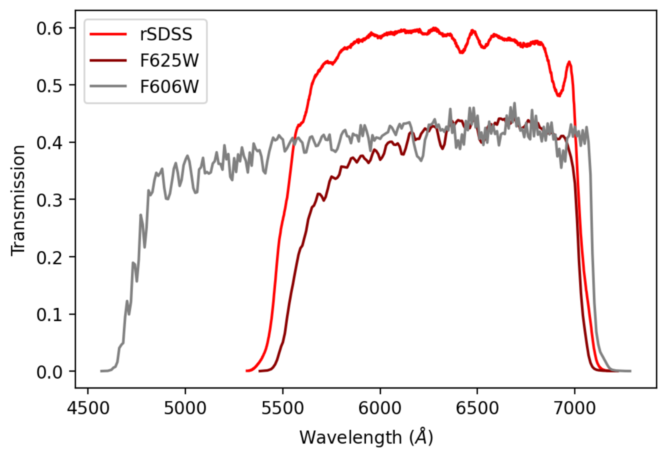

Fig. 1. Transmission profiles of the filters of J-PLUS and ACS/HSTimages used in our sample: rSDSS and F625W, respectively. The first includes the transmission through the entire system formed by the filter, the instrument and the atmosphere, while the second is in space and therefore does not include the atmosphere. We also show in grey the filter F606W for ACS/HST that is compatible with J-PLUS rSDSS filter.

we set the clipping of J-PLUS images in the 0.01-99.99% range and of ACS/HST images in the 0.1-99.9% range.

- We normalize the image to the [0, 1] range. This should facilitate the training process and the validation with different metrics.
- We generate small paired crops of the ACS/HST and J-PLUS images. The 24 ACS/HST images are two times 2048×4096 pixels, too large to be fed as input to any neural network. Thus, a trimming process is required to make small images. Our approach is to scan the entire image from left to right and top to bottom by shifting half the target crop size. This means that half of the crop overlaps with the next one. Once we have an ACS/HST crop, we use its central sky position (RA,DEC) to trim from J-PLUS image a matched crop. At this point, the astrometric alignment between ACS/HST and J-PLUS is crucial to make sure that the paired pixels contain the same information. For the cropping process we use the astcrop function of the GNU Astronomy Utilities (Gnuastro) package [16]. To avoid numerical issues during the learning process we remove all crops that have NaN values in any pixel.

We run several test with paired crops following this pre-processing getting unsatisfactory results. The main problem was the appearance of many faint sources in the super-resolved images, even when the J-PLUS image showed a rather dark uniform background. We conclude that one important factor causing this effect was the fact that ACS/HST images are much deeper than the J-PLUS counterparts, making an unfair comparison. To mitigate this effect we added the following steps after the scaling and before the normalization for the ACS/HST images:

- We detect sources above 2σ after an iterative 3σ clipping process.
- We mask circular apertures around the detected sources.
- We estimate the background as the median and the noise as the standard deviation over the masked image with a 3σ clipping estimation.
- We remove the background level from the scaled ACS/HST image.
- We clip all pixels below 10 times the noise estimate. This means that we are keeping only those sources with a 10σ level of detection in ACS/HST.
- We add random noise equivalent to 10 times the noise. This means that the original 10σ detections turn into 1σ detections.

TABLE I
TRAIN, VALIDATION, AND TEST SAMPLES.

Sample	J-PLUS tiles	ACS/HST images	Paired crops
Train	11	14	2842
Validation	3	5	1039
Test	3	5	-

Figure 2 shows a zoom in of a J-PLUS crop and the same region in the ACS/HST original image and the ACS/HST crop after pre-processing (including resizing, background subtraction and noise addition). We observe how the original ACS/HST image has more faint sources and less noise, which can confuse the neural network. To process astronomical FITS images we utilize several functions of Astropy[2] a community-developed core Python package for Astronomy [17], [18], along with Photutils [19] an affiliated package that provides tools for detecting and performing photometry of astronomical sources.

E. Train, Validation and Test samples

The set of crops is split into train, validation and test sets where only the training sample is used to update the weights of the network; the validation sample is used to monitor the performance of the network and to optimize the hyper-parameters; and the test sample is not seen by the network during training and reserved for final evaluation.

The overlapping of crops causes that the information in two contiguous crops is not independent, as part of the image is repeated. This means that crops from the same image can not be used for training, validation and testing. Thus, we split the samples based on the J-PLUS tiles: 11 for training, 3 for validation and 3 for testing. The selection of validation and test images is not random, we select in both cases one tile including a nearby galaxy and other tile including a globular cluster. In Table I we summarize the total number of tiles/images and crops that form each subsample. Note that for the final evaluation of our model we do not use small crops but a large squared trim of the ACS/HST image, as we want to compare the number of (reliable) objects detected.

[2]http://www.astropy.org

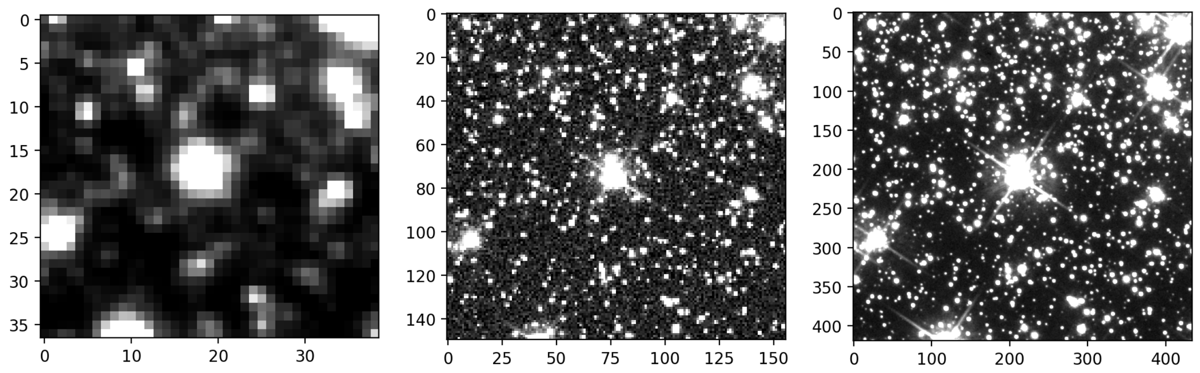

Fig. 2. Zoom into one crop to show the difference between the J-PLUS (left), ACS/HST pre-processed (centre), and ACS/HST original (right) images.

III. Methodology

A. Model Architecture

The input to our super-resolution model is the J-PLUS crop and for the target image we use the ACS/HST resized crop with 4× the resolution of J-PLUS. For our super-resolution problem, we take an approach based on a GAN architecture [20]. GANs use a generator network (see Fig. 3) to generate realistic images and a discriminator network (see Fig. 4) to ensure that the generated images are visually indistinguishable from the high-resolution target images. We follow the approach of Real-ESRGAN [11] because of its proven success to super-resolve real-world images, especially in relation to background textures. In addition, all the codes and models are publicly available to use [21], [22].

Real-ESRGAN model builds upon ESRGAN [23], which is an enhancement of the seminal SRGAN [24]. The generator is the same used in ESRGAN, i.e. a deep network with residual-in-residual dense blocks (RRDB, see Fig. 3). This block is inspired by the DenseNet architecture [25] and connects all layers within the residual block with each other. It consists of three Dense Blocks, each containing four consecutive convolution layers followed by Leaky ReLU activations and an additional convolutional layer. The concatenated output of every previous layer is fed into the next convolution layer. The upsample layers consist of a nearest-neighbour interpolation.

The discriminator in Real-ESRGAN aimed at producing accurate gradient feedback for local textures, instead of discriminating global scales. Such requirement of a greater discriminative power for complex training outputs led the authors to use as discriminator a U-Net [26] with spectral normalization [27]. The U-Net outputs realness values for each pixel, as shown in Figure 4, and can provide detailed per-pixel feedback to the generator.

B. Loss functions

We follow the prescription from [11] for the loss functions. These authors first trained a model with the L1 loss. Then, with this model - named Real-ESRNet - as the initialization of the generator, the Real-ESRGAN is trained with a combination of L1 loss, perceptual loss [28] and GAN loss [20]. The perceptual loss is a feature reconstruction loss that encourages the pixels to have similar feature representations as computed by a loss network ϕ. The feature reconstruction loss is the squared and normalized Euclidean distance between feature representations:

$$\ell_{feat}^{\phi,j}(\hat{y}, y) = \frac{1}{H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y)\|_2^2 \tag{1}$$

Following [11] our perceptual loss includes the {conv1, ...conv5} feature maps (with weights {0.1, 0.1, 1, 1, 1}) before activation in the pre-trained VGG19 network [29]. For the GAN loss we use its standard form (called *vanilla* in the basicsr [22] package) through the binary cross entropy loss:

$$\begin{aligned} \ell_G &= \ell_{BCE}(1, D(G(z))) \\ \ell_D &= \ell_{BCE}(1, D(x)) - \ell_{BCE}(0, D(G(z))) \end{aligned} \tag{2}$$

Thus, the combined loss used to train the generator of the GAN is as follows:

$$\ell_{G-GAN} = \alpha \ell_{\ell_G} + \psi \ell percep + \nu \ell_1 \tag{3}$$

where $\alpha = 0.1$ and $\psi = \nu = 1$.

C. Evaluation Metrics

To evaluate our training we monitor the losses describe in III-B along with two evaluation metrics: the Peak Signal to Noise Ratio (PSNR) and the Structural Similarity Index (SSIM). The PSNR is a measure of the ratio between the maximum signal and the corrupting noise. It is commonly used to quantify reconstruction quality in denoising models. A higher PSNR value equates to better, though it is only conclusively valid when it is used to compare results from the same content. The SSIM is a perceptual metric that incorporates the idea that spatially close pixels have strong inter-dependencies. These two metrics are included in the BasicSR [22] open-source image and video restoration toolbox distribution based on Pytorch, thus we can directly compute them in the Real-ESRGAN pipeline.

For a quantitative evaluation during the optimisation of the hyper-parameters and the final performance of the model, we also compute the Multi-scale Structural Similarity Index

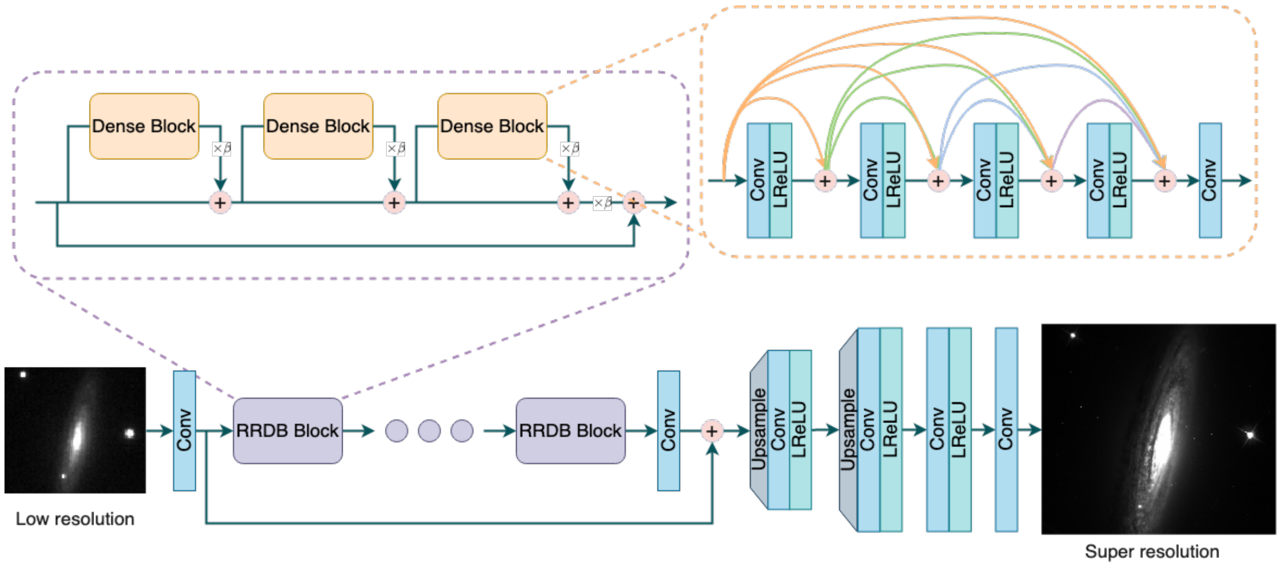

Fig. 3. Architecture of the generator (RRDBNet)

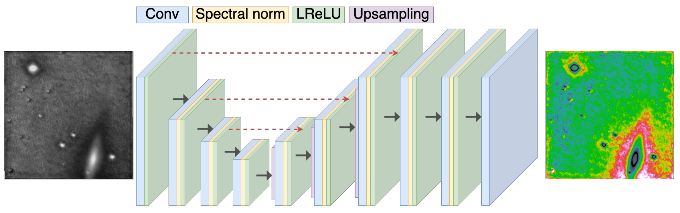

Fig. 4. Architecture of the discriminator (UNet)

(MS-SSIM), the Feature Similarity Index (FSIM [30]) and the Haar wavelet-based Perceptual Similarity Index (HaarPSI, [31]) metrics. MS-SSIM measures the similarity of structure in images on a combination of different scales. FSIM maps the features and measures the similarities between two images: it uses phase congruency as the primary feature to measure the significance of a local structure and the gradient magnitude as the secondary feature to obtain the contrast information. HaarPSI utilizes the coefficients obtained from a Haar wavelet decomposition, followed by an additional logistic function mapping to the local similarities obtained from high-frequency Haar wavelet filter responses, to construct local similarity maps. To compute all these metrics we use the PyTorch Image Quality (PIQ [32]) library, which contains a collection of measures and metrics for image quality assessment.

IV. Experimental Setup

A. Training details

We train with a scaling factor of $\times 4$ between the J-PLUS (low resolution) and the ACS/HST resized (high resolution) crops. These input images are only 1-channel, meaning that when training from the pre-trained ESRGAN model, which uses 3-channel input images, the weights and bias from the first and last convolutional layers are ignored. Our implementation uses training codes and architecture algorithms from BasicSR [22] and Real-ESRGAN [11] open-source repositories. Some codes were modified to take into account the special format of our astronomical images (FITS) and other requirements for our training procedure (e.g. reading the LR/HR pairs from a meta-file, change of the learning rate milestones on-the-fly).

Following ESRGAN and Real-ESRGAN prescriptions, we divide the training process into two stages. First, we train the model with the L1 loss. The weights are initialized with the pre-trained ESRGAN model. The learning rate starts at 2×10^{-4} and is divided by 2 at [100k, 200k] iterations. Then, this trained L1 model is used as the initialization for the generator. The generator is trained using the loss function in Eq. 3, with the learning rate set to 10^{-4} and halved at [100k, 200k] iterations.

In both stages the training HR patch size is set to 256 and the batch size to 4 (see Sect. IV-B for more details). The Real-ESRGAN training applies a second-order degradation model (it includes blurring with different types of kernels, adding Gaussian and Poisson noise, and a JPEG compression quality factor). We made several tests with this degradation approach and found that it produces undesired results, like deep depressions around bright sources. Thus, we only apply resizing (with a percentage of 80 to 120 and a probability of 0.25, 0.25, and 0.5 for up, down, and keep, respectively) and the last $sinc$ function with a 0.8 probability (use to model ringing and overshoot artifacts). The training pairs are synthesize on the fly. At each iteration, the training samples are randomly selected from a training pair pool to form a training batch with higher diversity. The size of the pool is set to 100.

We employ Adam optimizer, with typical values β_1 and β_2 of 0.9 and 0.99, and alternately update the generator and discriminator network until the model converges. We train our models within the Google Colaboratory environment, so the GPUs assigned vary from run to run. The GPUs available and that we used are NVIDIA Tesla P100, T4 and K80.

B. Optimization of the hyper-parameters

1) Model

There is a set of parameters that define the model or how it learns that are subject to tune to improve the final result. In our model, the number of RRDB blocks and the number of convolutional filters have to be the same as in the architecture of ERSGAN that we are using as pretrained model: 23 and 64, respectively.

We determine the learning hyper-parameters by running a random hyper-parameter search over a set of learning rate (between 2×10^{-3} and 10^{-4}) and batch size (1, 2, 4) values. We train a subsample of 1739 images in linear scale in 7 random configurations for 500 iterations. Based on the SSIM tendency of the validation subsample and the trend of the loss of the training subsample we select a learning rate of 2×10^{-3} and a batch size of 4. The learning rate is the higher value that shows an increasing SSIM and reasonable low values for the loss function. Our strategy for training includes a multi-step learning rate decrease, so this value is successively divided in half after [100k, 200k] iterations.

2) Data

As described in Sect. II-D, the pre-processing of the image modifies how the information is weighted and this could have a huge impact on the performance of the model. We test different scaling options, namely, linear, square root, and logarithmic scales. In Figure 5, we show the effect of these three scales in two different images. Finally, we select the linear scale as the one with the best performance: highest values is most metrics and best visual super-resolution solution. After scaling, the values for the clipping are set by a qualitative analysis of target objects (as bright globular clusters) and we keep this process outside of the hyper-parameter optimization.

The last pre-processing step we take into account is the size of the crops we use for training. As we are aiming at a 4× super-resolution, we are limited on the bigger side by the hardware capacity and on the smaller side by the PSF of J-PLUS. We test two different pairs of image sizes: 32/128 and 64/256 (J-PLUS/ACS/HST). We note that the size of the training set is going to be different depending on the size of the crop due to the methodology for cropping we follow. The results for both sizes are quite similar (e.g. SSIM=0.541 vs 0.542 or FSIM=0.648 vs 0.649). We select the ACS/HST 256 size to facilitate the generalization of the model as the smaller size will not include big sources and many of the J-PLUS crops might display only background noise. As observed by [23], training a deeper network benefits from a larger patch size, since an enlarged receptive field helps to capture more semantic information. However, it costs more training time and consumes more computing resources. We run a test with even larger crops of 128/512 that showed good convergence. The problem is that such a large size reduces significantly the training sample as the ACS/HST images are small, rotated, and have a gap of NaN values in the diagonal, what causes that we cannot generate crops of size 512 from many ACS/HST images.

TABLE II
RESULTS OF OPTIMIZATION OF THE HYPER-PARAMETERS ON VARIOUS METRICS.

Scale	Size	PSNR	SSIM	MS-SSIM	FSIM	HaarPsy
Linear	64/256	23.5255	0.5197	0.7707	0.7391	0.2875
Sqrt	”	15.7723	0.4204	0.5187	0.6550	0.2849
Log	”	21.1608	0.5414	0.7235	0.6483	0.4648
Log	32/128	22.5977	0.5420	0.7280	0.6492	0.4644

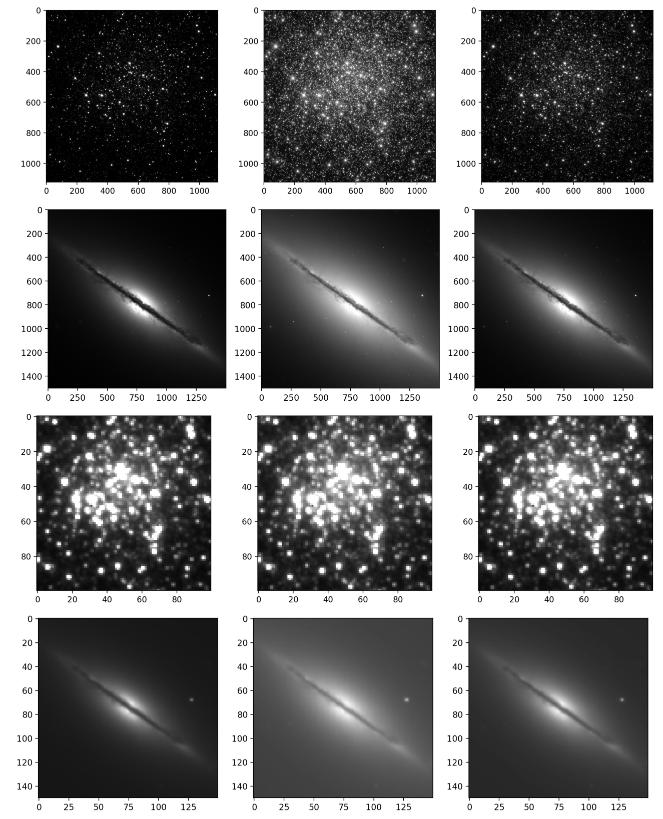

Fig. 5. Images of the centre of the globular cluster NGC6341 (top) and of the galaxy NGC5866 (bottom) in ACS/HST (first and third rows) and J-PLUS (second and fourth rows) for the three scales tested: linear (left), logarithmic (centre), and square root (right). After scaling we apply clipping and normalize to the (0, 1) range (see the text for more details). The gray-scale is the same in all panels with black for 0 and white for 1.

V. RESULTS

A. Training and evaluation

We are limited in time, in part due to the use of GPUs in Colab for computing, so the number of iterations in the training process has yet to be increased, probably in the two training stages (see Sect. IV-A). We show in Figure 6 the loss functions obtained in the two training stages. We can see how the L1 loss of the upper panel follows the expected behaviour: during the first iterations the decline is fast and

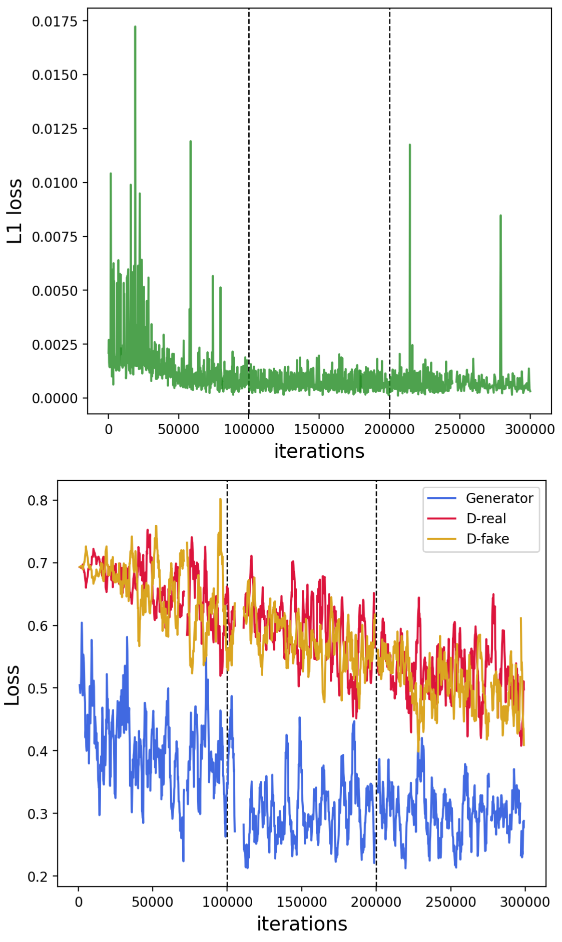

Fig. 6. Loss functions from our training process in two stages: L1 loss of the first stage (top) and the generator total loss and discriminator real and fake losses of the second stage (bottom). Vertical dashed black lines indicate where the learning rate is halved.

bouncy and then it flattens and decrease very slowly. We train for 300k iteration before going into the second training stage, but the result may improve continuing the training for a few more 100k iterations. For example the original Real-ESRGAN was trained for 1000k iterations. The training with the GAN is more complicated and each iteration consumes more time. We can see from the bottom panel that the losses from the generator and the discriminator are stable. In this case, we run 300k iterations, so we expect the final model to improve with more training time.

To have a better idea of the performance of the models as we train, we measure at selected number of iterations the evaluation metrics described in Sect. III-C. For computing these values we used the test sample described in Sect. II-E. In Table III we report the metric estimates for initial, intermediate, and final training, while in Figure 7 we show the complete trend. Regarding the Real-ESRNet model from the first training stage, most metrics show a steep increase during the first ∼100k iterations. SSIM and MS-SSIM seem to reach almost their maximum value (above 0.98) just after 100k, not

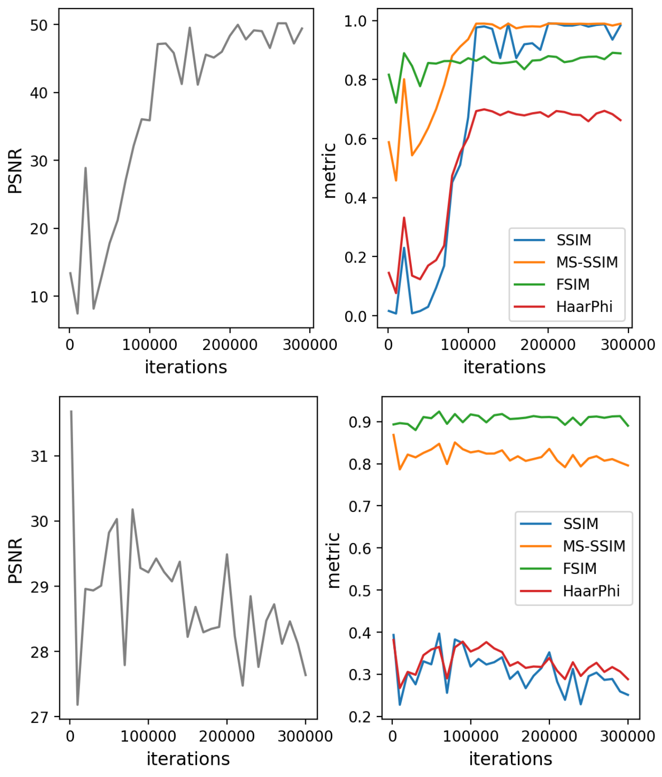

Fig. 7. Evaluation metrics computed over the test sample as a function of the iterations for the first (top) and second (bottom) training stages.

leaving much room for quantifying improvement, while we know via visual inspection that this model is far from giving a satisfactory super-resolved image. Thus, SSIM and MS-SSIM are probably not appropriate metrics to evaluate super-resolution tasks for astronomical images. PSNR and FSIM keep increasing more slowly after the first 100k iterations, indicating, as we already mentioned, that more training time could still improve our model. HaarPhi reports a rather low value (around 0.6), which would agree what our visual inspection (see Sect. V-B). However, it also flattens after ∼100k iterations, while we do observe that the model gets better between about 150k and 300k.

Regarding the Real-ESRGAN model from the second training stage, metrics do not show a good trend. All but FSIM start at the highest value, rapidly drop in a few iterations, then show a moderate increase up to ∼100-150k, continuing later with the descent. FSIM starts at a high value, close to 0.9 and similar to the 0.89 value of the Real-ESRNet 300k, that grows slightly to 0.91 during the first 50-100k and flattens here for the rest of the training. This behaviour is rather compatible with the super-resolved images we obtain (see Figure 9) where there is no significant improvement between 100 and 300k.

B. *Qualitative results*

In Figure 8 we show examples of the super-resolved versions of a test image after 1k, 100k, 200k, and 300k iterations

TABLE III
RESULTS OF THE TWO STAGE TRAINING.

Model	Iter	PSNR	SSIM	MS-SSIM	FSIM	HaarPsy
Real-ESRNet	1k	13.4043	0.0161	0.5878	0.8164	0.1451
	100k	35.9025	0.6718	0.9365	0.8722	0.6042
	300k	49.4180	0.9833	0.9891	0.8886	0.6624
Real-ESRGAN	2k	31.6790	0.3933	0.8684	0.8933	0.3814
	100k	29.2125	0.3182	0.8268	0.9171	0.3540
	300k	27.6371	0.2511	0.7959	0.8903	0.2883

for the Real-ESRNet model. The 1k model entirely corrupts the image: the galaxy disappears, and the big star is grainy. As we train, the model progressively recovers the details of the image: the galaxy shows the bulge, the disk, and the extended low surface brightness parts; the stars become more point-like sources, with most of the brightness concentrated, and the brightest star starts to show the spikes of the PSF from ACS/HST. However, we are still far from recovering the fainter objects: we cannot see the more extended part of the galaxy or the faintest stars that we discern in the original J-PLUS image.

In Figure 9 we see examples of the super-resolved images of the same test image after 2k, 100k, 200k, and 300k iterations for the Real-ESRGAN model. Compared to the previous Real-ESRNet models, we can notice three main differences: (i) the background shows a stronger textured pattern; (ii) the extended, low brightness parts of the galaxy are better distinguished from the background; and (iii) fainter sources are present. Though the final result is unsatisfactory in reproducing visually a real astronomical image, these three differences represent an improvement with respect to the Real-ESRNet 300k model in three key properties of our images. Thus, we can support the need for the second training stage including the perceptual loss, and the GAN loss with the discriminator. The test with other discriminator architecture or even without discriminator remains as future work.

C. Source detection

We can roughly check the effectiveness of our super-resolution model (though knowing it can be improve) by performing source detection over the J-PLUS image and its super-resolved version, comparing the result of the latter also to the ACS/HST image to confirm that the sources detected are real and not just artifacts produced by the GAN. We use GNUAstro [16] dedicated functions for detection, segmentation, and catalogue generation. For a fair comparison, we use the same parameter values in these functions for both images. We find a positive result in the crowded region at the centre of the globular cluster NGC6341 (one of the validation tiles): more sources are detected in the super-resolved image version and these sources are also found in the ACS/HST image. In Figure 10 we show this result. For example, focusing on the centre of the image we see that only two sources are detected in the original J-PLUS image, while a total of 5 are extracted in the super-resolved version, all having counterparts in ACS/HSTŚlightly to the upper left from the centre, we have another clear detection that in the original J-PLUS was blended between two brighter sources. In the middle of the bottom half we have another source that in the original J-PLUS is too faint to be detected but that becomes a clear detection in the super-resolved image. We are aware that many more sources appear in the ACS/HST image, but we are limited not only by the power of the super-resolution model but also by the depth on the original J-PLUS image. Finally, we note that there is a small shift between the detections in both images, being the J-PLUS original detection more consistent with the ACS/HST image, suggesting that is our model the one that introduces such displacement. We estimate that the shift is of the order of 1px of the J-PLUS original image. Further analysis is required to find the origin of this issue and correct it.

D. Caveats and future work

1) Training sample: Filters

We are limited by the amount of data that we can retrieve from both ACS/HST and J-PLUS covering the same portion of the sky at similar wavelengths. We select the broadband rSDSS filter from J-PLUS because it is the one used for object detection, which is the main task that we want to improve. The closest filter from ACS/HST in matching the rSDSS transmission curve is the F625W filter. However, the broader F606W filter also contains rSDSS in its wavelength coverage. Figure 1 shows the transmission curves of these three filters. The number of images using F606W in the HST archive is much higher, meaning that we can greatly increase our training set in number by the cost of pairing images that represent slightly different parts of the electromagnetic spectrum. We begin the training with the most similar filter, i.e. F625W with a limited training sample, getting good results. However the model can be greatly refined and the use of a much larger sample using F606W data is a strong candidate for improving the model.

We follow the steps described in Sect. II-C and II-D to augment the LR/HR paired sample. So far we are testing whether the model improves by increasing the original F625W training sample of 2842 crops to an F625W+F606W training sample of 11348 crops. This sample is generated from 15 new J-PLUS tiles that have 36 overlapping ACS/HST F606W images in total. The initial results up to date are included in Appendix A. Note that we can still greatly augment this sample as we are still downloading ACS/HST F606W data, for instance, after 3 days of querying the MAST archive we

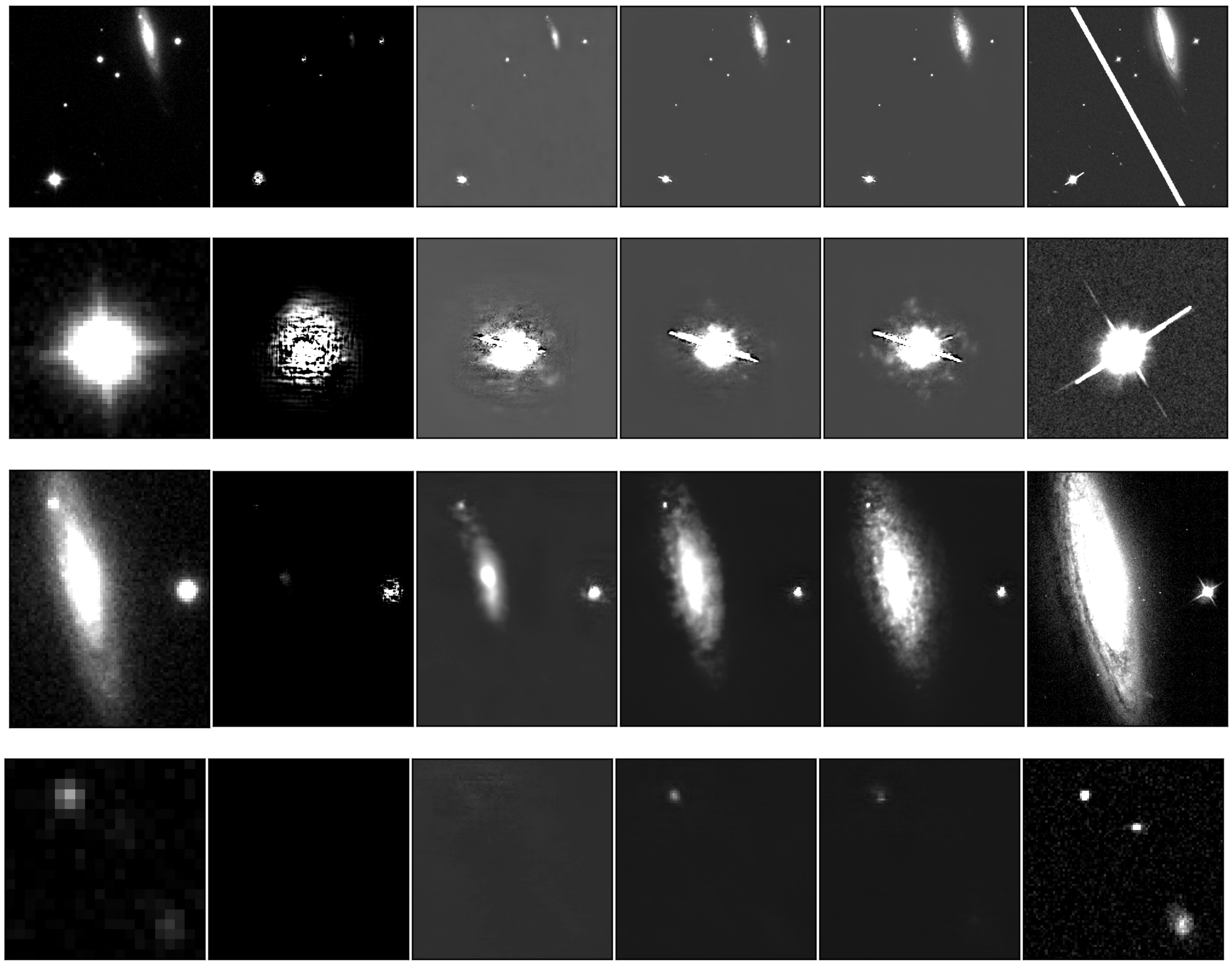

Fig. 8. Original J-PLUS test image (left) and its super-resolved version after 1k, 100k, 200k, and 300k iterations of the first training stage, i.e. the Real-ESRNet model, along with the corresponding ACS/HST pre-processed image. We show the entire crop of the ACS/HST area (top), a zoom in into the biggest star (second row), a zoom in into the galaxy (third row), and a zoom in into a couple of faint objects (bottom).

have new 350 ACS/HST images overlapping 99 J-PLUS tiles pending quality control.

2) Pair matching: astrometry solutions

The astrometry solution, meaning the precise measurement of celestial objects' positions in the sky, is recorded in the FITS header of the images. We use directly this information as downloaded from the respective archives. During the quality control check we discard those pairs of ACS/HST/J-PLUS images which astrometry visually disagrees. However, we are aware that the current match is limited by the J-PLUS pixel size, which is much larger than the ACS/HST one. This means that still certain misalignment between images could be present and could affect the learning process. We expect this mismatch to be random, i.e. that there is none preferential direction, and the neural network seems to be able to handle it since we do not observe any trend in the trained models that we can associate to this issue.

3) Include extra-info: seeing, exposure time, PSF

In addition to the pixel brightness that the images provide, we can gather other helpful information related to its the meta-data. For instance, in the case of the J-PLUS tiles the estimate of the seeing is provided in the archive, and it would give to the network extra information about how good were the observing conditions and how punctual the point-like sources are expected to be. From the ACS/HST archive we could retrieve the total exposure time of the image, which would be a proxy for how deep is the image, i.e. how faint are the faintest sources we find. The PSF provides essential information regarding the shape of the point-like sources. As can be seen in the second row of Figures 8 and 9, the PSF of J-PLUS and ACS/HST are very different and, so far, our model is failing at converting the J-PLUS PSF into the ACS/HST one. We expect that modifying the architecture to receive this extra information could boost the learning process and improve the

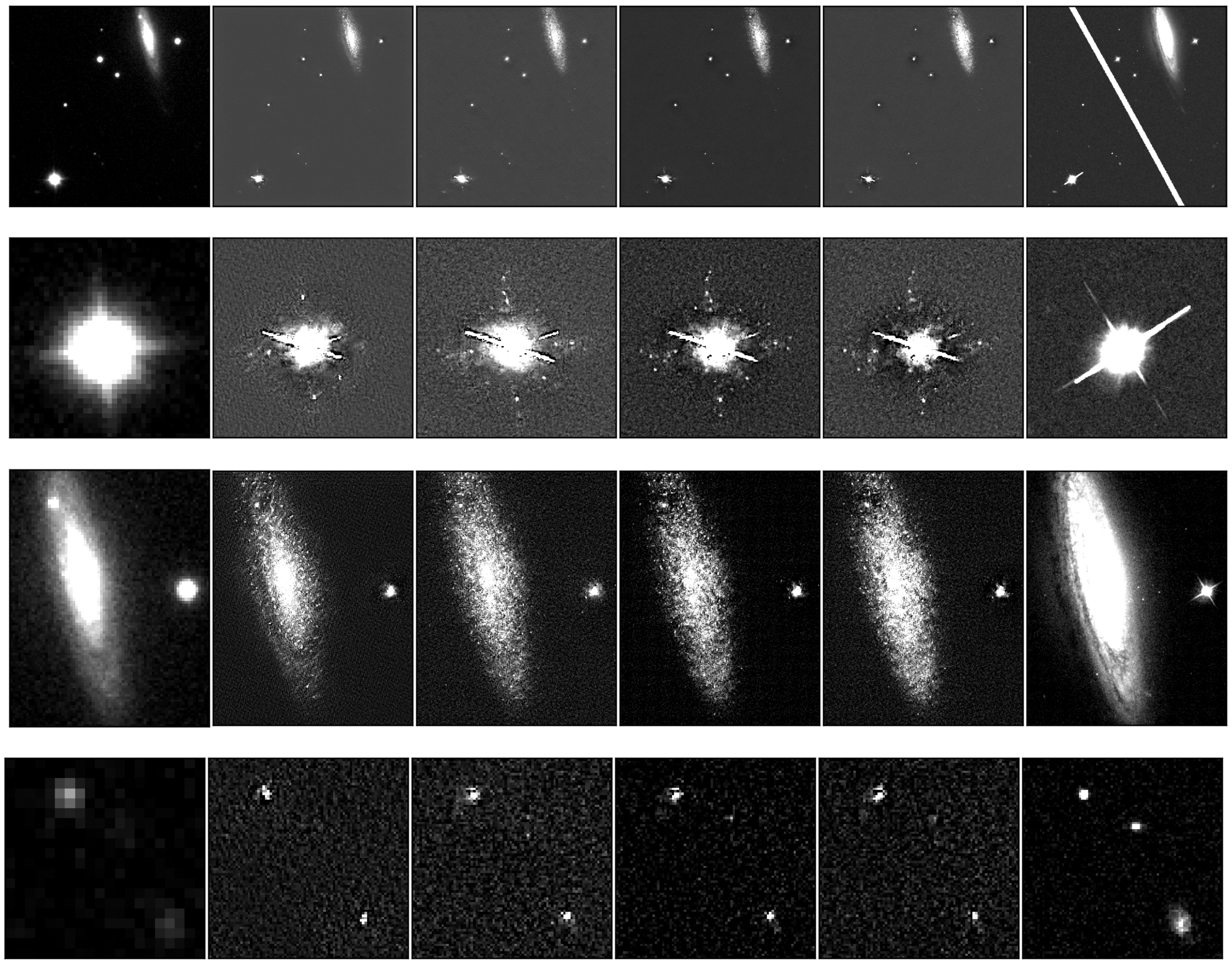

Fig. 9. Original J-PLUS test image (left) and its super-resolved version after 2k, 100k, 200k, and 300k iterations of the second training stage, i.e. the Real-ESRGAN model, along with the corresponding ACS/HST pre-processed image. We show the entire crop of the ACS/HST area (top), a zoom in into the biggest star (second row), a zoom in into the galaxy (third row), and a zoom in into a couple of faint objects (bottom).

performance of the final model. However, this is out of the scope of the present work and will remain as future work.

4) Other SR architectures

We selected the Real-ESRGAN model because a direct inference of the official trained version on two test images of a globular cluster and a nearby galaxy provided a good result, both visually and on a quick source detection. The generator of this architecture has recently showed promising results on super-resolution and denoising of simulated X-ray astronomical images [4]. However, there are other models that we aim at testing and, if the initial results are satisfactory, training. For instance, we are interested in Normalizing Flow models like SRFlow [33], that outputs many different images for a single input. These type of models have fewer hyperparameters than GAN approaches and converge monotic and stable. They also show higher consistency between the input and the super-resolved outputs, which can help in minimizing the artifacts. Foreseeing the offer of super-resolution within the J-PLUS archive, we would also like to test lighter and faster models, like the efficient PAN model [12].

E. Computational resources

This work implies a heavy use of computational resources. On the one hand, the original images and their crops require lots of storage space. A single J-PLUS tile image is about 50Mb, but the size of a single ACS/HST image can vary from 250Mb to almost 600Mb. This means that only for the storage of the samples listed in Table I we need $\sim$20Gb, but for the augmented dataset described in Sect. V-D1 we require of the order of 220Gb. To train our models in Colab, we have to upload at least the crops to Google Drive, which for the sample of F625W crops only takes up more than 1Gb. But not only the images take a large amount of storage, both the models and the training states - which we have to save every 1-2k iterations - take up 120 to 160Mb, meaning that have to be

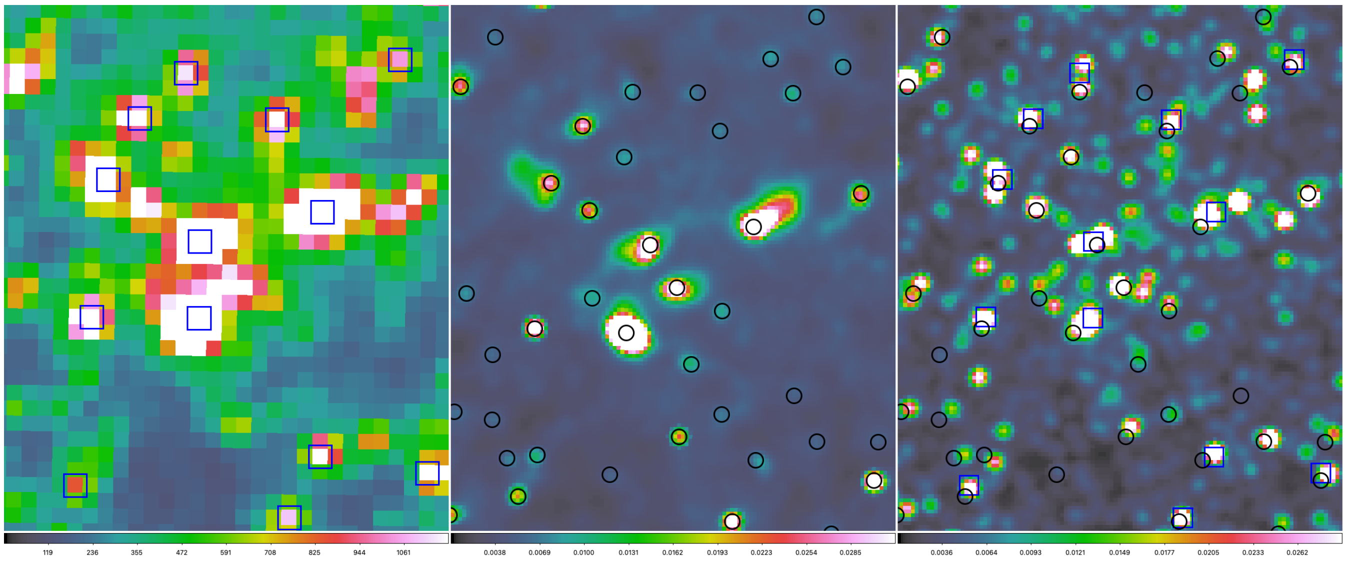
Fig. 10. Central region of the NGC6341 globular cluster in J-PLUS (left), its Real-ESRGAN 300k super-resolved version (centre), and the ACS/HST pre-processed (right). The super-resolution and the ACS/HST image are smoothed for clarity purposes. We overlay the sources detected in J-PLUS (blue squared) and the ones detected in its super-resolved version (black circles).

careful and download and remove them from the Drive every few 10k iterations to avoid a full storage on Drive.

On the other hand, this work has a high computational time consumption, which is even larger because the use of Colab is discontinuous. From the ETA provided by the training pipeline we estimate that the non-stop training time of 300k iterations for first stage is about 1.1 days and for the second, about 3.3 days. Another task that consumes much time is downloading images from the archives. While downloading a J-PLUS tile is the order of seconds, an ACS/HST image takes 4 to 10min. The pre-processing of the images is done locally, and it takes ∼20min to do the J-PLUS and ACS/HST crops for the training sample in Table I.

VI. Conclusions

We develop a deep-learning based super-resolution model to enhance source detection in ground-based low-resolution astronomical images. We use paired images of J-PLUS and ACS/HST to train a Real-ESRGAN model in two different stages: first with the L1 loss and, then, the trained Real-ESRNet is used as initialization for the generator during the training of the GAN. Due to the special characteristics of the astronomical data a careful pre-processing of the images is essential to the success of the learning process.

The resulting Real-ESRGAN after 300k iterations, our final model up to today, shows problems at reproducing the PSF of ACS/HST images and the extended details of nearby galaxies. However, it is successful at highlighting the fainter sources, a key attribute to enhance source detection in astronomical images. A quick test to prove the effectiveness of our model, to reinforce the usefulness of super-resolution techniques in improving source detection, produce satisfactory results: more sources are detected in the centre of a crowded field in the super-resolved version of a J-PLUS image compared to the original, and all these sources are real since they appear in the corresponding ACS/HST image.

We note that there is room for improvement, but our project has demonstrated the feasibility of enhancing source detection in J-PLUS images is possible through super-resolution techniques.

References

[1] Zhan Li, Qingyu Peng, Bir Bhanu, Qingfeng Zhang, and Haifeng He. Super resolution for astronomical observations. , 363(5):92, May 2018.

[2] C. J. Díaz Baso and A. Asensio Ramos. Enhancing SDO/HMI images using deep learning. , 614:A5, June 2018.

[3] Takatoshi Shibuya, Noriaki Miura, Kenji Iwadate, Seiji Fujimoto, Yuichi Harikane, Yoshiki Toba, Takuya Umayahara, and Yohito Ito. Galaxy morphologies revealed with Subaru HSC and super-resolution techniques. I. Major merger fractions of L_{UV} 3-15 $L*_{UV}$ dropout galaxies at z 4-7. , 74(1):73–91, February 2022.

[4] Sam F. Sweere, Ivan Valtchanov, Maggie Lieu, Antonia Vojtekova, Eva Verdugo, Maria Santos-Lleo, Florian Pacaud, Alexia Briassouli, and Daniel Cámpora Pérez. Deep Learning-Based Super-Resolution and De-Noising for XMM-Newton Images. In *SciOps 2022: Artificial Intelligence for Science and Operations in Astronomy (SCIOPS). Proceedings of the ESA/ESO SCOPS Workshop held 16-20 May*, page 19, May 2022.

[5] William Hadley Richardson. Bayesian-based iterative method of image restoration∗. *J. Opt. Soc. Am.*, 62(1):55–59, Jan 1972.

[6] K. L. Yeo, A. Feller, S. K. Solanki, S. Couvidat, S. Danilovic, and N. A. Krivova. Point spread function of SDO/HMI and the effects of stray light correction on the apparent properties of solar surface phenomena. *Astronomy and Astrophysics*, 561:A22, January 2014.

[7] Antonia Vojtekova, Maggie Lieu, Ivan Valtchanov, Bruno Altieri, Lyndsay Old, Qifeng Chen, and Filip Hroch. Learning to denoise astronomical images with U-nets. , 503(3):3204–3215, May 2021.

[8] Ryan Hausen and Brant E. Robertson. Morpheus: A Deep Learning Framework for the Pixel-level Analysis of Astronomical Image Data. , 248(1):20, May 2020.

[9] H. Domínguez Sánchez, M. Huertas-Company, M. Bernardi, D. Tuccillo, and J. L. Fischer. Improving galaxy morphologies for SDSS with Deep Learning. , 476(3):3661–3676, February 2018.

[10] Andres C. Rodríguez, Tomasz Kacprzak, Aurelien Lucchi, Adam Amara, Raphaël Sgier, Janis Fluri, Thomas Hofmann, and Alexandre Réfrégier. Fast cosmic web simulations with generative adversarial networks. *Computational Astrophysics and Cosmology*, 5(1), Nov 2018.

[11] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *International Conference on Computer Vision Workshops (ICCVW)*, 2021.

[12] Hengyuan Zhao, Xiangtao Kong, Jingwen He, Yu Qiao, and Chao Dong. Efficient image super-resolution using pixel attention. In *European Conference on Computer Vision*, pages 56–72. Springer, 2020.

[13] A. J. Cenarro, M. Moles, D. Cristóbal-Hornillos, A. Marín-Franch, A. Ederoclite, J. Varela, C. López-Sanjuan, C. Hernández-Monteagudo, R. E. Angulo, H. Vázquez Ramió, K. Viironen, S. Bonoli, A. A. Orsi, G. Hurier, I. San Roman, N. Greisel, G. Vilella-Rojo, L. A. Díaz-García, R. Logroño-García, S. Gurung-López, D. Spinoso, D. Izquierdo-Villalba, J. A. L. Aguerri, C. Allende Prieto, C. Bonatto, J. M. Carvano, A. L. Chies-Santos, S. Daflon, R. A. Dupke, J. Falcón-Barroso, D. R. Gonçalves, Y. Jiménez-Teja, A. Molino, V. M. Placco, E. Solano, D. D. Whitten, J. Abril, J. L. Antón, R. Bello, S. Bielsa de Toledo, J. Castillo-Ramírez, S. Chueca, T. Civera, M. C. Díaz-Martín, M. Domínguez-Martínez, J. Garzarán-Calderaro, J. Hernández-Fuertes, R. Iglesias-Marzoa, C. Iñiguez, J. M. Jiménez Ruiz, K. Kruuse, J. L. Lamadrid, N. Lasso-Cabrera, G. López-Alegre, A. López-Sainz, N. Maícas, A. Moreno-Signes, D. J. Muniesa, S. Rodríguez-Llano, F. Rueda-Teruel, S. Rueda-Teruel, I. Soriano-Laguía, V. Tilve, L. Valdivielso, A. Yanes-Díaz, J. S. Alcaniz, C. Mendes de Oliveira, L. Sodré, P. Coelho, R. Lopes de Oliveira, A. Tamm, H. S. Xavier, L. R. Abramo, S. Akras, E. J. Alfaro, A. Alvarez-Candal, B. Ascaso, M. A. Beasley, T. C. Beers, M. Borges Fernandes, G. R. Bruzual, M. L. Buzzo, J. M. Carrasco, J. Cepa, A. Cortesi, M. V. Costa-Duarte, M. De Prá, G. Favole, A. Galarza, L. Galbany, K. Garcia, R. M. González Delgado, J. I. González-Serrano, L. A. Gutiérrez-Soto, J. A. Hernandez-Jimenez, A. Kanaan, H. Kuncarayakti, R. C. G. Landim, J. Laur, J. Licandro, G. B. Lima Neto, J. D. Lyman, J. Maíz Apellániz, J. Miralda-Escudé, D. Morate, J. P. Nogueira-Cavalcante, P. M. Novais, M. Oncins, I. Oteo, R. A. Overzier, C. B. Pereira, A. Rebassa-Mansergas, R. R. R. Reis, F. Roig, M. Sako, N. Salvador-Rusiñol, L. Sampedro, P. Sánchez-Blázquez, W. A. Santos, L. Schmidtobreick, B. B. Siffert, E. Telles, and J. M. Vilchez. J-PLUS: The Javalambre Photometric Local Universe Survey. , 622:A176, February 2019.

[14] J-plus-dr2 - data access services. https://archive.cefca.es/catalogues/jplus-dr2, 2022. Accessed: 2022-07-05.

[15] Mast queries (astroquery.mast). https://astroquery.readthedocs.io/en/latest/mast/mast.html, 2022. Accessed: 2022-07-05.

[16] M. Akhlaghi and T. Ichikawa. Noise-based Detection and Segmentation of Nebulous Objects. *ApJS*, 220:1, September 2015.

[17] Astropy Collaboration et al. Astropy: A community Python package for astronomy. , 558:A33, October 2013.

[18] Astropy Collaboration et al. The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package. , 156(3):123, September 2018.

[19] Larry Bradley, Brigitta Sipőcz, Thomas Robitaille, Erik Tollerud, Zè Vinícius, Christoph Deil, Kyle Barbary, Tom J Wilson, Ivo Busko, Hans Moritz Günther, Mihai Cara, Simon Conseil, Azalee Bostroem, Michael Droettboom, E. M. Bray, Lars Andersen Bratholm, P. L. Lim, Geert Barentsen, Matt Craig, Sergio Pascual, Gabriel Perren, Johnny Greco, Axel Donath, Miguel de Val-Borro, Wolfgang Kerzendorf, Yoonsoo P. Bach, Benjamin Alan Weaver, Francesco D'Eugenio, Harrison Souchereau, and Leonardo Ferreira. astropy/photutils: 1.5.0, July 2022.

[20] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. June 2014.

[21] Real-esrgan repository. https://github.com/xinntao/Real-ESRGAN, 2022. Accessed: 2022-07-15.

[22] Xintao Wang, Ke Yu, Kelvin C.K. Chan, Chao Dong, and Chen Change Loy. BasicSR: Open source image and video restoration toolbox. https://github.com/xinntao/BasicSR, 2018. Accessed: 2022-06-30.

[23] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Computer Vision – ECCV 2018 Workshops*, 2019.

[24] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[25] Forrest N. Iandola, Matthew W. Moskewicz, Sergey Karayev, Ross B. Girshick, Trevor Darrell, and Kurt Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *ArXiv*, abs/1404.1869, 2014.

[26] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]).

[27] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. 02 2018.

[28] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *ICLR*, 2015.

[30] Lin Zhang, Lei Zhang, Xuanqin Mou, and D. Zhang. Fsim: A feature similarity index for image quality assessment. *Trans. Img. Proc.*, 20(8):2378–2386, aug 2011.

[31] Rafael Reisenhofer, Sebastian Bosse, Gitta Kutyniok, and Thomas Wiegand. A haar wavelet-based perceptual similarity index for image quality assessment. *Signal Processing: Image Communication*, 61:33–43, 2018.

[32] Sergey Kastryulin, Dzhamil Zakirov, and Denis Prokopenko. PyTorch Image Quality: Metrics and measure for image quality assessment, 2019. Open-source software available at https://github.com/photosynthesis-team/piq.

[33] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflow: Learning the super-resolution space with normalizing flow. In *ECCV*, 2020.

Appendix

A. Preliminary results: Real-ESRNet trained with F625W+F606W

We select a sample of 11+15 J-PLUS tiles that have 60 overlapping ACS/HST images in total to generate a F625W+F606W training sample of 11348 crops and test if the augmentation of the training sample (even though with images covering a longer wavelength range to the blue) helps us improve the model. Figure 11 shows the L1 loss function during the training of the first stage up to 300k iterations, it is rather flat with spikes and declines slowly. In Figure 12 we show the metrics measured for our test sample during the 300k iterations. Compared to the results obtained with the F625W sample, the learning curves for this metrics grow monotonically (with noisy peaks) showing that no overfitting is happening and that there is still room for learning as none of them have reach their maximum. Figure 13 continues is this direction showing that the learning process is taking longer with this larger and more diverse training dataset. However, from the brightest star we can see how the recovery of the PSF seems to go in a better direction than the previous model (compare to Figure 8). We will continue training in this stage for a few more 100k iterations before passing to the second training stage.

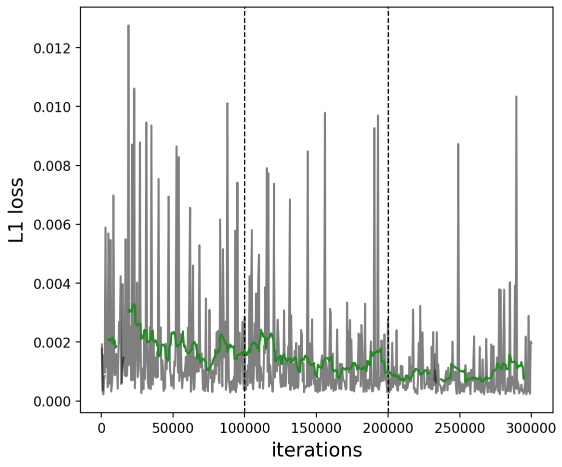

Fig. 11. Loss function from the first training stage: L1 loss. Vertical dashed black lines indicate where the learning rate is halved.

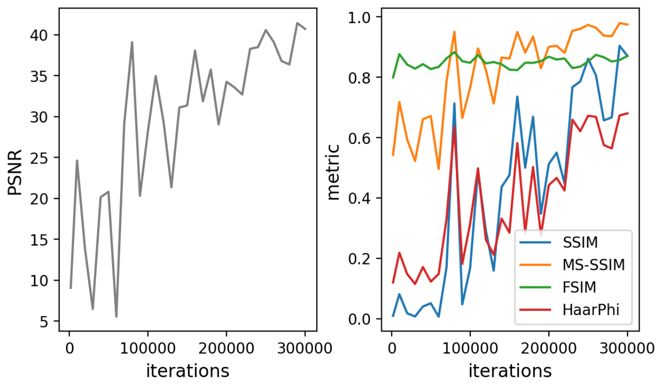

Fig. 12. Evaluation metrics computed over the test sample as a function of the iterations for the first training stage, i.e. Real-ESRNet.

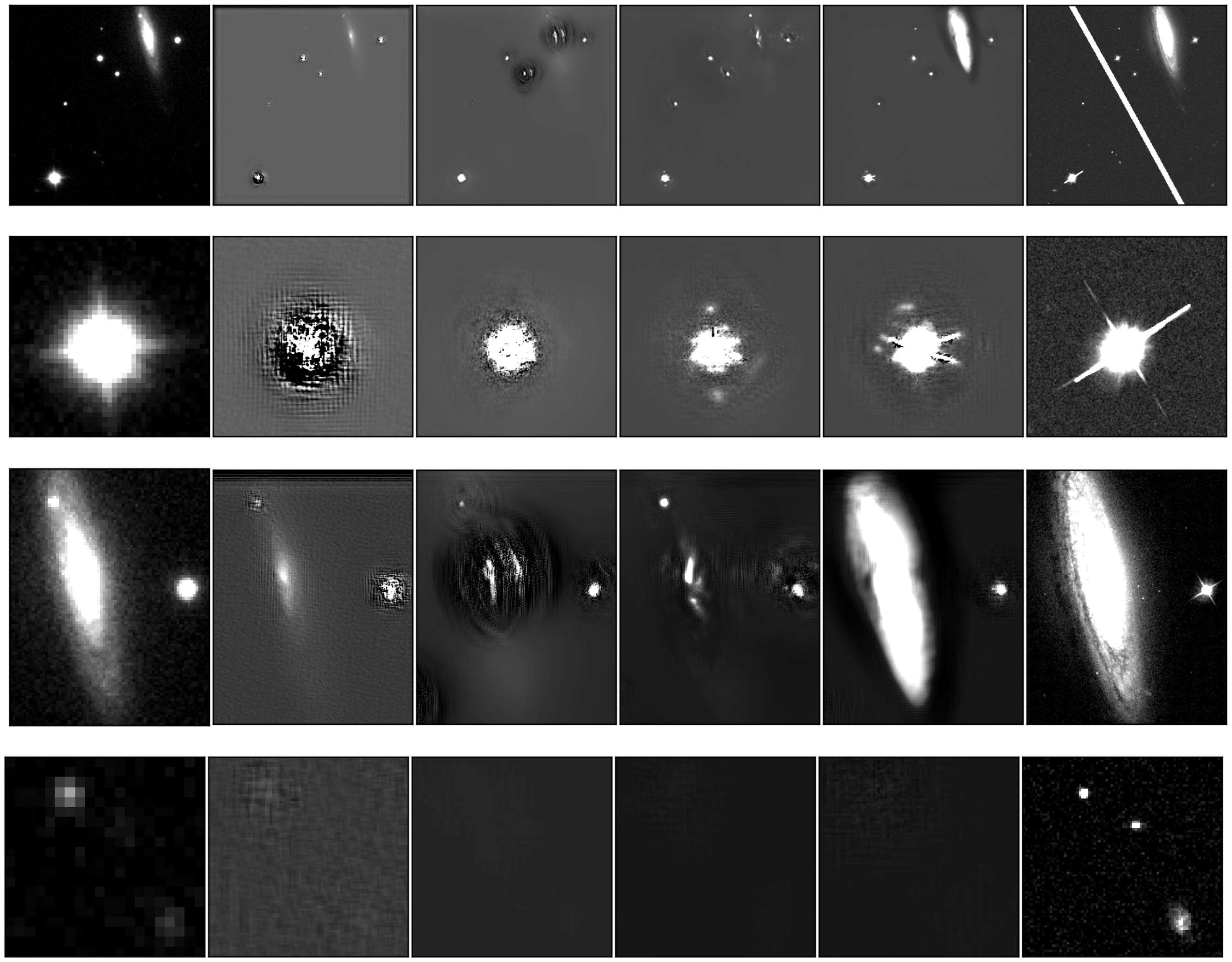

Fig. 13. Original J-PLUS test image (left) and its super-resolved version after 1k, 100k, 200k, and 300k iterations of the first training stage, i.e. the Real-ESRNet model, along with the corresponding ACS/HST pre-processed image. We show the entire crop of the ACS/HST area (top), a zoom in into the biggest star (second row), a zoom in into the galaxy (third row), and a zoom in into a couple of faint objects (bottom).