3D Detection of ALMA Sources through Deep Learning

Michele Delli Veneri[1,2][0000−0002−8178−2942], Lukasz Tychoniec[3][0000−0002−9470−2358], Fabrizia Guglielmetti[3][0000−0003−1201−2466], Eric Villard[3][0000−0003−4314−4947], and Giuseppe Longo[4][0000−0002−9182−8414]

[1] INFN Section of Naples, Napoli, via Cintia, 1, Italy, 80126
[2] Department of Electrical Engineering and Information Technology, University of Naples "Federico II", Via Claudio, 21, 80125 Naples NA, Italy
[3] ESO, Karl-Schwarzschild-Straße 2, 85748 Garching bei München
[4] Department of Physics "Ettore Pancini", University of Naples "Federico II", Via Cintia, 1, Italy, 80126

Abstract. We present a Deep Learning pipeline for the detection of astronomical sources within radiointerferometric simulated data cubes. Our pipeline is constituted by two Deep Learning models: a Convolutional Autoencoder for the detection of sources within the spatial domain of the cube, and a ResNet for the denoising and detection of emission peaks in the frequency domain. The combination of spatial and frequency information allows for higher completeness and helps to remove false positives. The pipeline has been tested on simulated ALMA observations achieving better performances and faster execution times with respect to traditional methods. The pipeline can detect 92% of sources with no false positives thus providing a reliable source detection solution for future astronomical radio surveys.

Keywords: Deep Learning · Object Detection · Radio Interferometry.

1 Introduction

In the last two decades, astronomical datasets underwent a rapid growth in size and complexity thus driving Astronomy in the big data regime [15, 3, 21, 29] and preventing, in most cases, the use of traditional interactive data reduction and analysis. Machine learning has therefore been extensively exploited by the community to solve a wide spectrum of problems spanning all aspects of the astronomical data cycle, from instrument monitoring to data acquisition and ingestion, to data analysis and interpretation [4, 17, 25, 14, 10, 7, 28, 1, 11, 19]. Particularly challenging are the problems posed by existent and future infrastructures for radio astronomy, such as the Atacama Large subMillimeter Array (ALMA), the Low Frequency Array (LOFAR) and the Square Kilometer Array (SKA) which are pushing astronomy in the exabyte and exascale computing. After the initial correlation and calibration of the raw signals in the Fourier space, these instruments provide data in the form of data cubes where two dimensions

are spatial (coordinates on the celestial sphere) and one is the frequency. In a very rough oversimplification, the extraction of the useful information from these data cubes requires the solution of a conceptually simple inverse problem which can be expressed in the form:

$$y = Ax + n \tag{1}$$

where y are the observations, x is the unknown signal, A is a degradation operator and n is the observational noise. A, which sometimes is also called *forward operator*, takes up a very complex underlying physical process and, when applied to the signal x, outputs the noiseless observation y. Many attempts have been made at solving this problem using Machine Learning (ML) based approaches[5, 30, 24, 22, 9].

In this paper, we present a deep-learning-based pipeline for the detection of sources within "uncleaned" data cubes, i.e. data which have not undergone any prior deconvolution (hereafter "dirty" datacubes). This choice was dictated by several shortcomings of the traditional cleaning approach , based on the tCLEAN algorithm [13], which works by iteratively performing deconvolutions of the individual frequency slices of the cube by assuming a parametric model for the source brightness distributions. Its main shortcomings are:

1. the iterative cleaning procedure works on an image basis assuming that each image (or slice) in the cube is independent from the others. Hence, each cube undergoes a time-consuming cleaning procedure which is unfeasible for current and future radio interferometers [6].
2. by working on each slice independently, tCLEAN completely ignores possible correlations between pixels along the frequency axis of the cube, and this leads to the introduction of biases and artifacts in the cleaned cube. For example, a noise peak would be deconvolved several time with the PSF of the instrument and then the recovered delta function would be convolved with the *clean beam* thus producing a structure morphologically similar to actual sources that underwent the same iterative deconvolution process.

The main novelty of our proposed pipeline with respect to other architectures [26, 16, 9, 24, 22] is to include frequency information (which is usually discarded) to help detect the sources and remove false detections. As we shall demonstrate, frequency information can help both in deblending spatially blended (overlapping) sources and in the detection of faint sources.

Our paper is structured as it follows: In Section 2 we describe the architectures of the deep learning models used in our pipeline, the complete data flow within the pipeline in order to explain its inner workings, and the training strategies for all the models. In Section 3 we present the simulation algorithm used to generate the realistic ALMA observations needed to train and test our pipeline, and we analyse the pipeline performances in detecting sources within the test set. A comparison is also made with the performances of *blobcat*: a classical source detection algorithm widely used within the community [12]. Finally, in Section 4 we draw our concluding remarks, and lay the prospect for future

work. We wish to emphasize that while this paper focuses mainly on the analysis of ALMA data cubes, the methodology is general and can easily be exported to the processing of similar data (e.g. LOFAR and SKA) as well as to other fields (such as radiology) requiring an accurate analysis of data cubes.

2 Methodologies

The pipeline we present in this work can be described as a decision graph interconnecting two deep learning models, each one taking a specialised role in the detection process. The architectures were chosen on the basis of their strengths: a convolutional architecture (we shall call it "Blobs Finder") to process spatial information and a Recurrent Neural Network (Deep GRU) to process frequency information. Before describing the flow of data within the pipeline, we hereby shortly describe the DL models.

2.1 The Blobs Finder

Blobs Finder is a 2D Deep Convolutional Autoencoder trained to solve the image deconvolution problem:

$$D[x, y] = P[x, y] \times M[x, y] + N[x, y] \tag{2}$$

where $D[x, y]$ is the dirty (stacked) image produced integrating along the frequency the dirty cube, $P[x, y]$ is the dirty PSF and $N[x, y]$ is the combination of all noise patterns in the data. Blobs Finder is trained with the dirty images as inputs, and the sky model images as targets. Both input and target images are normalized to the $[0, 1]$ range, which helps with the training process and allows to make a probabilistic interpretation of the autoencoder output. Standard data augmentation techniques are used to improve the model generalization capabilities. The Blobs Finder architecture is constituted by an Encoder Network and a Decoder Network. The Encoder consists of four convolutional blocks and a final fully connected layer. Each block contains a 2D Convolution layer with stride 2 and a kernel size of 3, a Leaky ReLU (Rectified Linear Unit) activation function and a 2D Batch Normalization layer. Each block halves the spatial extent of its input and doubles the number of channels. The Decoder is constituted by a fully connected layer, followed by four deconvolutional blocks and a final layer. Each deconvolutional block contains a bilinear interpolation function with a stride of 2 which up-samples spatially the input, a 2D Transposed Convolution with a kernel size of 3 and a stride of 1 which reduces the number of channels while preserving the spatial dimensions, a leaky ReLU activation function and a 2D Batch Normalization layer. The final layer (used to normalize the input to the $[0, 1]$ range) is a 2D Convolution with a kernel size of 1 and a stride of 1 followed by a Sigmoid activation function. To train Blobs Finder, we use as a loss function the weighted combination of two well-known losses in the DL image reconstruction and denoising framework: the l_1 loss and the Structural dissimilarity loss $DSSIM$ which is based on Structural Similarity Index measurement.

2.2 The Deep GRU

Deep Gated Recurrent Unit (GRU) is a Recurrent Neural Network (RNN) [23] constructed by combining together two layers of GRUs and a fully connected layer. GRU [8] tries to solve the main shortcoming of RNNs, a type of neural network designed to capture correlation in sequences of data (usually 1D signals), which is the exploding or vanishing of gradients, by introducing gating of its hidden states. Gating is introduced through the *reset* and *update* gates which, respectively, control how much of the previous hidden state must be remembered and the degree at which the current hidden state is similar to the previous one at each frequency iteration. The first helps to capture short-scale correlations in the data, while the second captures the long-scale correlations. In our implementation, each layer of GRUs outputs 32 hidden states (or channels to keep the nomenclature homogeneous) which are then concatenated to form a latent vector of size $[b, 64 \times 128]$ before being fed to a fully connected layer. The layer transforms its input in a vector with the same size as the input signal and then a Sigmoid activation function is applied to normalize it to the $[0, 1]$ range. As loss function, we use the l_1 loss.

2.3 The Pipeline

The pipeline can be divided into tree logical blocks: 2D source detection, frequency denoising and emission detection, and source focusing (these blocks are marked in Fig. 1). To ease the logical flow of the pipeline, we assume that all DL models have been trained to act as simple, functional map between their inputs and outputs.

1. **2D Sources Detection** (**1 - 4**). The image cube is normalized to the $[0, 1]$ range and then it is integrated along frequency (1) to create a 2D image. We refer to this image as the "dirty image". The dirty image is then cropped to a size of $[256, 256]$ pixels (which is large enough to contain the whole source and removes the edge of the images which are characterized by low SNR), normalized to the $[0, 1]$ range (2) and then fed to the first DL model **Blobs Finder**. The autoencoder processes the image and predicts a 2D probabilistic map (normalized to the $[0, 1]$ range) of source detection (3). A hard thresholding value of 0.1 is used to binarize the probabilistic map and then the *scikit-learn* [20] *label* and *regionprops* algorithms are used to extract bounding boxes around all blobs of pixels (4). The thresholding value is chosen to be 0.1 in order to peak all the signal detected by Blobs Finder, while excluding small fluctuation in the background. We refer to these blobs as source candidates. Figure 2 show, respectively, an example of an input dirty image containing 6 simulated sources (outlined by green bounding boxes and of which two are spatially blended), the target sky model image (with in green the target bounding boxes and in red the predicted bounding boxes extracted through thresholding of the predicted 2D probabilistic map), and the 2D prediction map with in red the predicted bounding boxes.

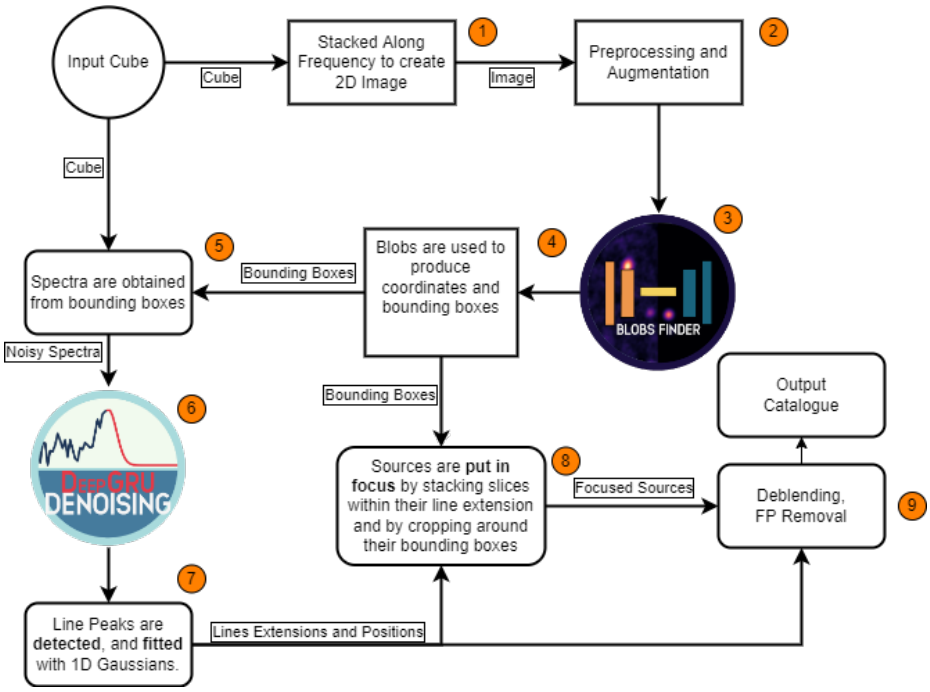

Fig. 1: The full pipeline schema, the numbers show the logic flow of the data within the pipeline (see the text for explanations).

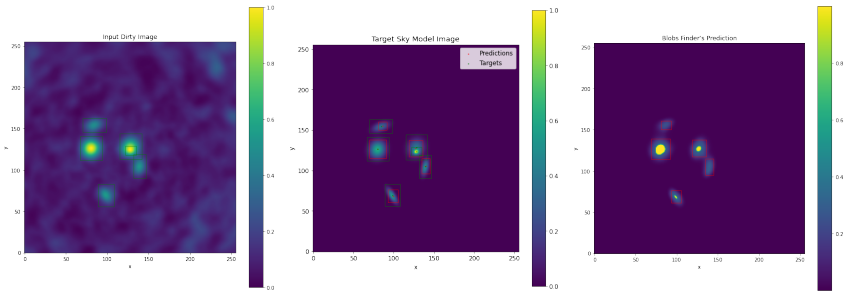

Fig. 2: Left: Input dirty image; center: Target Sky Model Image; right: Blobs Finder Prediction Image (green = true bounding boxes, red = predicted).

2. **Frequency Denoising** (**5 - 7**): bounding boxes around source candidates (blobs) are used to extract dirty spectra from the input cube. For each source candidate, its spectrum is extracted by adding, for each of the 128 frequency slices of the cube, the pixels inside its bounding box. The spectra are standardized, i.e. rescaled to have 0 mean and standard deviation = 1, and then fed to **Deep GRU**. Then the Deep Gated Recurrent Unit denoises the standardized spectra and outputs 1D probabilistic maps of source emission lines (hereafter, cleaned spectra (6)). In order to detect emission peaks, the cleaned spectra are then analysed with the *scipy* [27] *find_peaks* algorithm (threshold value of 0.1). Each peak is then fitted with a Gaussian [2]. As fitting algorithm, we employ the *LevMarLSQFitter* algorithm, and as initial guesses for the mean and amplitude of the Gaussian model, respectively, the previously detected peak position and its amplitude. All detected peaks are then recorded alongside their FWHMs (7). At this stage, to account for possible false positives produced by Blobs Finder, all potential candidates that show no meaningful peak in their spectra are removed. If more than one peak is found inside a given spectrum, the candidate likely is the superimposition of two or more blended sources and thus is flagged for deblending.
3. **Source Spectral Focusing** (**8 - 9**): this phase has two main objectives; deblend sources and remove false positives. The first is tackled via *spectral focusing* aimed at increasing the Signal to Noise Ratio (SNR) of the source (by cropping a $[64, 64]$ pixel box around its bounding box and integrating only within the peak FWHM, see Fig. 3). In order to measure sources SNRs, we introduce two diverse SNR metrics defined as follows:
   - **Global SNR**:
   $$SNR = \frac{median(x_s(r))}{var(x_n(R-r))} \tag{3}$$
   where $x_s(r)$ are the values of the source pixels contained within the circumference of radius r that inscribes the source bounding box, and $x_n(R-r)$ are the pixel values within an annulus of internal radius r and external radius R which has the same area of the inscribed circumference;
   - **Pixel SNR**:
   $$snr = \frac{x_i}{var(X)} \tag{4}$$
   where x_i is the value of the given pixel, and $var(X)$ is the variance computed on the full image.

   These two SNR estimators are used respectively to disentangle false positives from true sources and to deblend possible multiple sources within a given blob.
   First, the source is focused on the highest flux peak (primary peak) and the global SNR calculation is made to understand if the potential source must be discarded. Also, the pixel SNR measurement is used to find the highest SNR pixel in the image, which will act as a reference for the next phase of the deblending process. The candidate source is then focused around the secondary peaks. If the secondary peaks are outside the emission range of

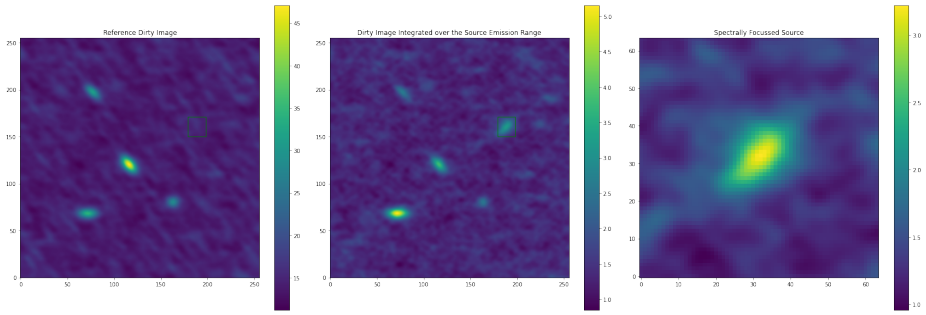

Fig. 3: An example of source focusing of a potential source

the brightest source, then the latter should disappear from the focused image (because the image is produced by integrating the cube outside the source emission range), and the pixel SNR measurement is used to find the highest SNR pixel in the image. If this pixel is different from the previously found reference pixel, then the neighboring pixels around this pixel are linked with a friend of friends algorithm in an iterative manner. At each iteration, the Global SNR is measured. Pixels are added in this fashion until a plateau in the global SNR is reached. If the highest SNR pixel and the reference pixel are within 1 pixel and the global SNR measured in the integrated cube image is higher than the measured SNR in the spectrally focussed image, then the source is discarded as a False Positive. A bounding box is finally created in order to encompass all the selected pixels, and a $[64, 64]$ pixel image is cropped around the bounding box.

3 Experiments

To train and test the proposed pipeline, we need a statistically significant sample of ALMA model and dirty cube pairs. To this effect we generated our own realistic simulations of ALMA data cubes by combining python and bash scripting with the Common Astronomy Software Application (CASA) v. 6.5.0.15 [18] python libraries. Each model cube was created by first generating a central source and then by adding between 2 and 5 additional sources such that each source emission flux is lower or equal to that of the central source. In order to simulate 3D sources, we combined 2D Gaussian Components in the spatial plane with 1D Gaussian components (emission lines) in frequency space. The source morphological parameters are sampled from the intervals reported in Table 1.

We then feed the sky models to the CASA *simobserve* taks which simulates interferometric measurements sets through a series of observing parameters. We use the ALMA Cycle 9 configuration C-3 for the antenna configuration, simulating 43 antennas in the 12-m Array with a maximum baseline of 0.50 km (for trchnical details see [18]). The dirty data cubes are then obtained by performing the fast Fourier transform of the visibility data and gridding. We generate

Table 1: Sampling intervals of the model source parameters. Sources are generated by randomly sampling from the outlined uniform distributions. The first column shows the parameter name, the second the range from which the parameter values are sampled, and the third the unit.

Parameter Name	Range	Unit
Number of components	[2 – 5]	–
Amplitude of 2D Gaussian compoment	[1 – 5]	arbitrary
FWHM of the 2D Gaussian component	[2 – 8]	pixel
Spectral index	[-2 – 2]	–
Position in the xy plane	[100 – 250]	pixel
Position angle	[0 – 90]	deg
Line amplitude	[1 – 5]	arbitrary
Line center	[10 – 110]	chan
Line FWHM	[3 – 10]	chan

$5,000$ simulated cube pairs containing a total of $22,532$ simulated sources and divide them into train, validation, and test sets (60%, 20%, 20%). The three sets contain respectively $13,512$, $4,465$, and $4,556$ simulated sources. While the two models are trained indipendently in parallel on the same training data, the predictions are made sequentially given that the input of each model is the output of the previous one.

3.1 Source Detection

Hereafter, we present the performances of Blobs Finder and Deep GRU in detecting sources within the $1,000$ cubes in the Test set. To check if a source has been detected by Blobs Finder, we measure the 2D Intersection over Union (2D IoU) between the true 2D bounding box and the predicted one, while for Deep GRU we measure the 1D IoU between the true emission ranges and the detected ones. In both cases, we use a threshold of 0.8. To ensure that the central part of the source emission of a True Positive (TP) is always detected, we require the distance between the centres of the true and predicted bounding boxes is smaller then 3 pixels. Blobs Finder predicts 4056 (89%) sources (TP). Matching them with the true $4,556$ sources in the Test set, $4,205$ (92.3%) pass the 2D IoU criterion, meaning that an additional 149 sources are detected by Blobs Finder but are spatially blended with another source. Blobs Finder misses 354 sources (FN) and detects no False Positives (FP). The 4056 bounding boxes are used to extract a corresponding number of dirty spectra from the dirty cubes. The Deep GRU detects $4,202$ emission peaks out of the $4,205$ present in the extracted spectra but also produces 62 false positives. Sources are then "spectrally focused" within the predicted frequency emission ranges and SNR checks are made to detect false positives and, in the case, deblend multiple sources. At the end of this phase all 62 false positives are correctly identified and eliminated.

In order to compare with the traditionally used algorithm *blobcat*, we run it on the 1000 dirty images in the test set. Blobcat requires two parameters: a

detection (T_d) and cut (T_f) SNR threshold to decide which peaks in the image are candidates for blobs and where to cut the blobs boundaries around them (in other words: pixels with a SNR higher than T_f are selected to form islands and island boundaries are defined by T_d). To make the fairest comparison possible, we measured *blobcat* performances with different choices of T_d and T_f through a grid-search strategy ($T_d \in [2, 15]$, $T_f \in [1, 10]$) and, in this paper, we report the best results ($T_d = 8\sigma$, $T_f = 4\sigma$). The same criterions used for Blobs Finder were then used to measure the performances. *blobcat* successfully detects $2,779$ (61%) sources, produces $2,429$ false detections and misses 1777 sources. The majority of sources missed by *blobcat* are spatially blended with brighter sources, or present a $SNR <= 5.0$, or are located at the edges of the images. Table 2 summarises the source detection performances. Regarding the time of execution, our pipeline can process an ALMA cube, on average, in 3.2 milliseconds , while *blobcat* takes on average 32 seconds. This is a crucial factor for ALMA (which is transitioning to a TB data regime [6]) and for other future radiointerferometers such as SKA.

Algorithm	**TP /**	**FP**	**FN**	**Precision**	**Recall**	**Mean IoU**
Pipeline	4202 (92.3%)	0	354 (7.7%)	1.0	0.923	0.84
blobcat	2779 (61%)	2429	1777 (39%)	0.53	0.609	0.71

Table 2: Comparison between the sequential proposed source finding pipeline composed by Blobs Finder, DeepGRU and Spectral Focusing, and *blobscat*. Columns show true positives (TP), false positives (FP), false negatives (FN), precision, recall and mean intersection over union (Mean IoU) between true bounding boxes and predicted ones. TP and FN are also expressed as fractions over the total number of sources.

4 Conclusions

In this paper, we present a novel pipeline for source detection in radiointerferometric data cubes. The pipeline is constituted by two DL models: Blobs Finder (Deep Convolutional Autoencoder) and Deep GRU (RNN). Blobs Finder detect sources within the integrated data cubes (2D images produced by integrating the cubes along the frequency axis) and the found candidate sources are used to extract spectra which are then fed to the Deep GRU. Deep GRU takes care of denoising the spectra in order to detect emission lines. Spatial and Spectral information is then combined to remove false positives and spatially deblend sources. To test the pipeline capabilities, we produce our own realistic simulations of ALMA observations, $5,000$ data cubes containing $22,532$ simulated sources. We also compare the pipeline performances with *blobcat*, a standard source finding algorithm extensively used within the community, showing that our proposed pipeline achieves better performances and faster execution times.

References

[1] A Akhazhanov et al. "Finding quadruply imaged quasars with machine learning – I. Methods". In: *Monthly Notices of the Royal Astronomical Society* 513.2 (Apr. 2022), pp. 2407–2421. DOI: 10.1093/mnras/stac925. URL: https://doi.org/10.1093/mnras/stac925.

[2] Astropy Collaboration et al. "The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package". In: *Astronomical Journal* 156.3, 123 (Sept. 2018), p. 123. DOI: 10.3847/1538-3881/aabc4f. arXiv: 1801.02634 [astro-ph.IM].

[3] Dalya Baron. *Machine Learning in Astronomy: a practical overview*. 2019. DOI: 10.48550/ARXIV.1904.07248. URL: https://arxiv.org/abs/1904.07248.

[4] Ignacio Becker et al. "Scalable End-to-end Recurrent Neural Network for Variable star classification". In: *arXiv* (2020). DOI: 10.48550/ARXIV.2002.00994. URL: https://arxiv.org/abs/2002.00994.

[5] Micah Bowles et al. "Attention-gating for improved radio galaxy classification". In: *Monthly Notices of the Royal Astronomical Society* 501.3 (Dec. 2020), pp. 4579–4595. DOI: 10.1093/mnras/staa3946. URL: https://doi.org/10.1093/mnras/staa3946.

[6] John Carpenter et al. "The ALMA Development Program: Roadmap to 2030". In: (2020). DOI: 10.48550/ARXIV.2001.11076. URL: https://arxiv.org/abs/2001.11076.

[7] Ting-Yun Cheng et al. "Optimizing automatic morphological classification of galaxies with machine learning and deep learning using Dark Energy Survey imaging". In: *Monthly Notices of the Royal Astronomical Society* 493.3 (Feb. 2020), pp. 4209–4228. DOI: 10.1093/mnras/staa501. URL: https://doi.org/10.1093/mnras/staa501.

[8] Junyoung Chung et al. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *CoRR* abs/1412.3555 (2014). arXiv: 1412.3555. URL: http://arxiv.org/abs/1412.3555.

[9] Liam Connor et al. "Deep radio-interferometric imaging with POLISH: DSA-2000 and weak lensing". In: *Monthly Notices of the Royal Astronomical Society* 514.2 (May 2022), pp. 2614–2626. DOI: 10.1093/mnras/stac1329. URL: https://doi.org/10.1093/mnras/stac1329.

[10] Roberta Duarte, Rodrigo Nemmen, and João Paulo Navarro. "Black hole weather forecasting with deep learning: a pilot study". In: *Monthly Notices of the Royal Astronomical Society* 512.4 (Mar. 2022), pp. 5848–5861. DOI: 10.1093/mnras/stac665. URL: https://doi.org/10.1093/mnras/stac665.

[11] Simon Goode et al. "Machine learning for fast transients for the Deeper, Wider, Faster programme with the Removal Of BOgus Transients (ROBOT) pipeline". In: *Monthly Notices of the Royal Astronomical Society* 513.2 (Apr. 2022), pp. 1742–1754. DOI: 10.1093/mnras/stac983. URL: https://doi.org/10.1093/mnras/stac983.

[12] C. A. Hales et al. "blobcat: software to catalogue flood-filled blobs in radio images of total intensity and linear polarization". In: *Monthly Notices of the Royal Astronomical Society* 425.2 (Aug. 2012), pp. 979–996. DOI: 10.1111/j.1365-2966.2012.21373.x. URL: https://doi.org/10.1111/j.1365-2966.2012.21373.x.

[13] J. A. Hogbom. "Aperture Synthesis with a Non-Regular Distribution of Interferometer Baselines". In: *Astronomy and Astrophysics* 15 (June 1974), p. 417.

[14] Sheng-Chieh Lin et al. "Estimating cluster masses from SDSS multiband images with transfer learning". In: *Monthly Notices of the Royal Astronomical Society* 512.3 (Mar. 2022), pp. 3885–3894. DOI: 10.1093/mnras/stac725. URL: https://doi.org/10.1093/mnras/stac725.

[15] Giuseppe Longo, Erzsébet Merényi, and Peter Tiňo. "Foreword to the Focus Issue on Machine Intelligence in Astronomy and Astrophysics". In: *Publications of the Astronomical Society of the Pacific* 131.1004 (2019), pp. 1–6. ISSN: 00046280, 15383873. URL: https://www.jstor.org/stable/26874447 (visited on 06/24/2022).

[16] Vesna Lukic, Francesco de Gasperin, and Marcus Brüggen. "ConvoSource: Radio-Astronomical Source-Finding with Convolutional Neural Networks". In: *Galaxies* 8.1 (Dec. 2019), p. 3. DOI: 10.3390/galaxies8010003. URL: https://doi.org/10.3390/galaxies8010003.

[17] Berta Margalef-Bentabol et al. "Detecting outliers in astronomical images with deep generative networks". In: *Monthly Notices of the Royal Astronomical Society* 496.2 (June 2020), pp. 2346–2361. DOI: 10.1093/mnras/staa1647. URL: https://doi.org/10.1093/mnras/staa1647.

[18] J. P McMullin et al. "CASA Architecture and Applications". In: *Astronomical Data Analysis Software and Systems XVI ASP Conference Series, Vol. 376, proceedings of the conference held 15-18 October 2006 in Tucson, Arizona, USA. Edited by Richard A. Shaw, Frank Hill and David J. Bell., p.127* 376 (Oct. 2007).

[19] W. J. Pearson et al. "Identifying galaxy mergers in observations and simulations with deep learning". In: *Astronomy & Astrophysics* 626 (June 2019), p. 49. DOI: 10.1051/0004-6361/201935355. URL: https://doi.org/10.1051/0004-6361/201935355.

[20] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[21] Meyer Z. Pesenson, Isaac Z. Pesenson, and Bruce McCollum. "The Data Big Bang and the Expanding Digital Universe: High-Dimensional, Complex and Massive Data Sets in an Inflationary Epoch". In: *Advances in Astronomy* 2010 (2010), pp. 1–16. DOI: 10.1155/2010/350891. URL: https://doi.org/10.1155/2010/350891.

[22] S Rezaei et al. "DECORAS: detection and characterization of radio-astronomical sources using deep learning". In: *Monthly Notices of the Royal Astronomical Society* 510.4 (Dec. 2021), pp. 5891–5907. DOI: 10.1093/mnras/stab3519. URL: https://doi.org/10.1093/mnras/stab3519.

[23] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*. Ed. by David E. Rumelhart and James L. Mcclelland. Cambridge, MA: MIT Press, 1986, pp. 318–362.

[24] K. Schmidt et al. "Deep learning-based imaging in radio interferometry". In: *Astronomy & Astrophysics* (Apr. 2022). DOI: 10.1051/0004-6361/202142113. URL: https://doi.org/10.1051/0004-6361/202142113.

[25] Sam F. Sweere et al. *Deep Learning-Based Super-Resolution and De-Noising for XMM-Newton Images*. 2022. DOI: 10.48550/ARXIV.2205.01152. URL: https://arxiv.org/abs/2205.01152.

[26] A Vafaei Sadr et al. "DeepSource: point source detection using deep learning". In: *Monthly Notices of the Royal Astronomical Society* 484.2 (Feb. 2019), pp. 2793–2806. DOI: 10.1093/mnras/stz131. URL: https://doi.org/10.1093/mnras/stz131.

[27] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[28] Zhenping Yi et al. "Automatic detection of low surface brightness galaxies from Sloan Digital Sky Survey images". In: *Monthly Notices of the Royal Astronomical Society* 513.3 (Mar. 2022), pp. 3972–3981. DOI: 10.1093/mnras/stac775. URL: https://doi.org/10.1093/mnras/stac775.

[29] Ivan Zelinka, Massimo Brescia, and Dalya Baron, eds. *Intelligent Astrophysics*. Springer International Publishing, 2021. DOI: 10.1007/978-3-030-65867-0. URL: https://doi.org/10.1007/978-3-030-65867-0.

[30] Qingguo Zeng, Xiangru Li, and Haitao Lin. "Concat Convolutional Neural Network for pulsar candidate selection". In: *Monthly Notices of the Royal Astronomical Society* 494.3 (Apr. 2020), pp. 3110–3119. DOI: 10.1093/mnras/staa916. URL: https://doi.org/10.1093/mnras/staa916.