Astronomy & Astrophysics manuscript no. yolo_sdc2_paper
September 16, 2025

YOLO-CIANNA: Galaxy detection with deep learning in radio data

II. Winning the SKA SDC2 using a generalized 3D-YOLO network

D. Cornu[1], B. Semelin[1], P. Salomé[1], X. Lu[6], S. Aicardi[5], J. Freundlich[4], F. Mertens[1], A. Marchal[2,3], G. Sainton[1], F. Combes[1,7], C. Tasse[1,9]

[1]LUX, Observatoire de Paris, Université PSL, Sorbonne Université, CNRS, 75014, Paris, France
[2]Canadian Institute for Theoretical Astrophysics, University of Toronto, 60 St. George Street, Toronto, ON M5S 3H8
[3]Research School of Astronomy & Astrophysics, Australian National University, Canberra ACT 2610 Australia
[4]Université de Strasbourg, CNRS UMR 7550, Observatoire astronomique de Strasbourg, 67000 Strasbourg, France
[5]DIO, Observatoire de Paris, CNRS, PSL, 75014, Paris, France
[6]IDRIS, CNRS, F-91403 Orsay, France
[7]Collège de France, 11 Place Marcelin Berthelot, 75005, Paris, France
[9]Department of Physics & Electronics, Rhodes University, PO Box 94, Grahamstown, 6140, South Africa

Received February 8, 2024; accepted August 19, 2024

ABSTRACT

Context. As the scientific exploitation of the Square Kilometre Array (SKA) approaches, there is a need for new advanced data analysis and visualization tools capable of processing large high-dimensional datasets.
Aims. In this study, we aim to generalize the YOLO-CIANNA deep learning source detection and characterization method for 3D hyperspectral HI emission cubes.
Methods. We present the adaptations we made to the regression-based detection formalism and the construction of an end-to-end 3D convolutional neural network (CNN) backbone. We then describe a processing pipeline for applying the method to simulated 3D HI cubes from the SKA Observatory Science Data Challenge 2 (SDC2) dataset.
Results. The YOLO-CIANNA method was originally developed and used by the MINERVA team that won the official SDC2 competition. Despite the public release of the full SDC2 dataset, no published result has yet surpassed MINERVA's top score. In this paper, we present an updated version of our method that improves our challenge score by 9.5%. The resulting catalog exhibits a high detection purity of 92.3%, best-in-class characterization accuracy, and contains 45% more confirmed sources than concurrent classical detection tools. The method is also computationally efficient, processing the full ∼1TB SDC2 data cube in 30 min on a single GPU.
Conclusions. These state-of-the-art results highlight the effectiveness of 3D CNN-based detectors for processing large hyperspectral data cubes and represent a promising step toward applying YOLO-CIANNA to observational data from SKA and its precursors.

Key words. Methods: numerical – Methods: statistical – Methods: data analysis – Galaxies: statistics – Radio lines: galaxies

1. Introduction

New observing facilities generate ever larger datasets, which are often high-dimensional and have a wide dynamic range. The combined volume and complexity of these datasets put stress on classical data processing pipelines, which often fail to scale to the required data rate. To overcome these difficulties, statistical approaches such as machine learning (ML) methods are increasingly employed, as they are known to scale well with data size and dimensionality. The Square Kilometer Array (SKA, Braun et al. 2015) is a prominent example of next-generation telescopes in the radio domain. This instrument is expected to transform our understanding of the Universe, with a wide range of applications from setting constraints on the Cosmic Dawn to the detection of new exoplanets in our Galaxy. The unprecedented data rate foreseen for SKA is already prompting the international community to develop new strategies for data processing, storage, and distribution (Scaife 2020). In this context, the SKA Observatory (SKAO) organizes recurrent Science Data Challenges (SDCs) over simulated SKA-like data products (Bonaldi et al. 2021; Hartley et al. 2023; Bonaldi et al. 2025). These challenges aim to stimulate the development of highly efficient methods and to enable the community to prepare for handling SKA data.

The present paper is the second in a series introducing YOLO-CIANNA, a deep learning regression-based method for source detection and characterization. The series aims to evaluate the method's performance on simulated data from the SDCs, in preparation for its applications to observational data from SKA precursors and pathfinders. The first paper, Cornu et al. (2024) hereafter Paper I, covers the full method description and its application to simulated continuum images from the SDC1 dataset (Bonaldi et al. 2021). It also includes a concise review of ML object detection methods along with some examples of astronomical applications. The method description emphasizes specific design choices tailored to astronomical datasets. This first paper concludes with a discussion of the biases and limitations associated with the SDC1 challenge definition, and presents possible improvements to the method itself.

In the present work, we extend the results of Paper I by generalizing YOLO-CIANNA to 3D source detection in hyperspectral data and apply it to simulated HI emission cubes from the SDC2. The method was initially developed and used by team

MINERVA (Machine Learning for Radioastronomy at Observatoire de Paris) for its participation in the SDC2, in which the team secured first place. Therefore, this paper provides a detailed description of an updated version of MINERVA's method.

2. SDC2 description

2.1. Data cubes and source catalogs

The SDC2 is a galaxy detection and characterization challenge in simulated 3D HI cubes representative of future SKA-mid observations. A complete description of the challenge and of the associated data can be found in the SDC2 summary paper Hartley et al. (2023), which details the simulation of the source catalog using TRECS (Bonaldi et al. 2019), the construction of the sky model, and the addition of instrumental effects. The primary data product is a MAIN cube covering a 20 square degree region with a synthesized beam of 7 arcsec and a pixel resolution of 2.8 arcsec. It covers a frequency bandwidth of 950-1150 MHz (z = 0.235-0.495) sampled at 30 kHz. This results in a massive cube (851 GB) of size $5851 \times 5851 \times 6668$ [pixels, pixels, channels], corresponding to 228×10^9 voxels, a voxel being defined as the volume of a square pixel over one frequency channel. Continuum images of the same field are also provided with an identical spatial resolution and bandwidth, but sampled at a 10 MHz resolution. Thermal noise consistent with 2000h of SKA integration time was added, yielding 26 to 31 μJy beam^{-1} per channel. The cube also features continuum subtraction residues, instrumental artifacts, and radio frequency interferences (RFI) subtraction residues, improving its realism.

The MAIN cube contains 233 245 simulated HI sources. Each is characterized by its sky coordinates (RA, Dec), integrated line flux f, HI major axis diameter S at 1 M$_\odot$ pc^{-2} (hereafter HI size), line width w_{20} measured at 20% of the peak, major axis position angle PA, and inclination angle i (between the line-of-sight and the galaxy plane). These parameters are known for each source and stored in a truth catalog that was kept secret during the original challenge and released afterward. Instead, participating teams were provided a smaller LDEV cube (40 GB), obtained through the same data generation procedure, but with a reduced field of view of 1 square degree. It shares the same spatial and spectral resolution and covers the same bandwidth, resulting in a size of $1286 \times 1286 \times 6668$. It is also accompanied by continuum images and an LDEV truth catalog of 11 091 sources that can be used for calibrating or training detection methods. In this paper, we reproduce the challenge conditions by training our detector on the LDEV cube and catalog. The now-released MAIN truth catalog is solely used for analyzing our results and comparison with the SDC2 summary paper.

Figure 1 shows a cutout around one of the brightest sources from the truth catalog. To improve contrast, voxel values are rescaled following $p'_i = \tanh\left(3p_i/\max_p\right)$, where p_i and p'_i are the original and rescaled voxel values, respectively, and max_p is the maximum voxel value in the cutout. To better visualize the inner 3D structure, we set the voxel transparency as inversely proportional to their rescaled value. This figure illustrates the typical 3D structure of a galaxy observed in HI, where the line width correlates with its rotation. With the SDC2 observational setup, sources are compact in the plane of the sky but extended over tens of channels, with less signal at the central coordinates. The signal is mostly aligned in a plane, which defines the source angles, PA and i.

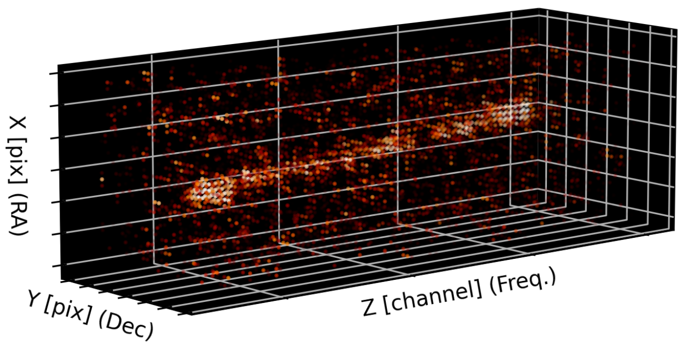

Fig. 1: 3D view of a $30 \times 30 \times 70$ subcube centered on a bright source from the MAIN catalog (id. 15 770) using a rescaled intensity. Transparency is inversely proportional to voxel intensity.

2.2. SDC2 scoring metric

The challenge task is to produce a catalog of detected sources in the MAIN cube along with all their properties. To evaluate the quality of a submission, the SDC2 relies on a dedicated scoring system. We provide a summary of the score computation as implemented in the scorer code[1]. The SDC2 scorer adopts the SDC1 matching system, which adds size and flux accuracy criteria to the classical sky separation metric. It is generalized to hyperspectral data by adding line width and central frequency accuracy criteria. The scorer defines a multiparameter error as

$$D_{\rm tot} = \sqrt{D^2_{\rm pos} + D^2_{\rm freq} + D^2_{\rm flux} + D^2_{\rm size} + D^2_{\rm width}}, \text{ with} \quad (1)$$

$$D_{\rm pos} = \sqrt{(x-\hat{x})^2 + (y-\hat{y})^2}/\hat{S}', \quad (2)$$

$$D_{\rm freq} = |\nu - \hat{\nu}|\,/\hat{w}_{20,Hz}, \quad (3)$$

$$D_{\rm flux} = \left|f - \hat{f}\right|/\hat{f}, \quad (4)$$

$$D_{\rm size} = \left|S - \hat{S}\right|/\hat{S}', \text{ and} \quad (5)$$

$$D_{\rm width} = |w_{20} - \hat{w}_{20}|\,/\hat{w}_{20}, \quad (6)$$

where truth values are indicated with a hat, (x,y) are the central pixel coordinates (converted from RA, Dec), ν the central frequency, S' the HI size convolved with the synthesized beam, f the line flux integral, S the HI size, and w_{20} the line width. Unlike the SDC1, all terms are equally weighted, and the global rejection threshold is set to $D_{\rm tot} \geq 5$. However, there are additional conditions on both the position and frequency that reject sources if $D_{\rm pos} \geq 1$ or $D_{\rm freq} \geq 1$. If multiple predictions match the same truth source, all are retained in the match catalog. However, their contribution to the final score is weighted down by the number of matches. In the rare case of a single prediction matching multiple truth sources, it is associated with the lowest $D_{\rm tot}$.

Matches are scored individually by comparing the predicted and true characteristics. The scorer defines a subscore for each source property to predict. Their average gives the match score. These are expressed as

$$s^j_k = \min\left(1, \frac{T^j}{E^j_k}\right), \quad \text{and} \quad s_k = \frac{1}{7}\sum_j^7 s^j_k, \quad (7)$$

where j represents a subscore, k identifies one source in the detection catalog, T^j is a threshold specific to each subscore, E^j_k is the error term specific to each subscore (possibly different from

[1] https://gitlab.com/ska-telescope/sdc/ska-sdc

Table 1: Error functions and thresholds for all subscores.

Subscore	Error function E^j	T^j
Sky position	$E^{\text{pos}} = \sqrt{(x-\hat{x})^2 + (y-\hat{y})^2}/\hat{S}''$	0.3
Central frequency	$E^{\text{freq}} = \lvert\nu - \hat{\nu}\rvert/\hat{w}_{20,Hz}$	0.3
Line flux	$E^{\text{flux}} = \lvert f - \hat{f}\rvert/\hat{f}$	0.1
HI size	$E^{\text{size}} = \lvert S - \hat{S}\rvert/\hat{S}''$	0.3
Line width	$E^{\text{width}} = \lvert w_{20} - \hat{w}_{20}\rvert/\hat{w}_{20}$	0.3
Position angle	$E^{PA} = \lvert PA - \hat{PA}\rvert$	10.0
Inclination angle	$E^{\text{inc}} = \lvert i - \hat{i}\rvert$	10.0

Note: $\hat{S}''$ is the target HI size convolved with twice the synthesized beam size, which is different from $\hat{S}'$.

the D terms used as matching criteria), and s^j is the resulting subscore. Error functions for each parameter are listed in Table 1. The final individual score is bound to the [0, 1] interval. The total score is the sum of all individual matched scores minus the number of false detections, following

$$M_s = \sum_{k}^{N_{\text{match}}} s_k - N_{\text{false}}. \tag{8}$$

In addition, we can also estimate mean subscores and a global average characterisation score as

$$\bar{s}^j = \frac{1}{N_{\text{match}}} \sum_{k}^{N_{\text{match}}} s_k^j \quad \text{and} \quad \bar{s} = \frac{1}{N_{\text{match}}} \sum_{k}^{N_{\text{match}}} s_k. \tag{9}$$

The scorer can be used for either the MAIN or LDEV cube. Hereafter, the term score refers to a scoring over the MAIN cube, unless stated otherwise.

2.3. Detectability selection function

YOLO-CIANNA is a supervised ML approach and therefore requires a set of labeled examples. For this, we rely on the provided LDEV cube and truth source catalog. However, as with the SDC1, the vast majority of the listed sources are too faint to be detectable ($\sim 85\%$). Including undetectable sources in the target catalog may be debatable from a challenge perspective. Still, the presence of faint sources in the simulation enhances background noise realism. It also ensures a realistic source signal-to-noise ratio (S/N) distribution in the target catalog without imposing an arbitrary detectability cut.

As with most supervised ML approaches, our method is sensitive to mislabeling in the training examples. It involves objects that should be detectable but not labeled as targets, or labeled targets that are too faint to be distinguishable from the background (Sects. 2.2 and 3.4 of Paper I). Therefore, we define a detectability selection function to curate the LDEV truth catalog. This function combines the integrated line flux f with an estimate of the source volume brightness V_b. We approximate the source volume by a cylinder with a radius equal to its beam-convolved HI size, expressed in pixels, and a length equal to its line width, in channels. The volume brightness is obtained as

$$V_b = \frac{\hat{f}}{\pi S'^2 D}, \text{ with} \tag{10}$$

$$S' = \sqrt{\hat{S}_{deg}^2 + B_s^2}\,\frac{1}{P_s}, \quad \text{and} \quad D = \frac{\hat{w}_{20}}{c}\,\frac{\hat{\nu}^2}{\nu_{HI_0}}\,\frac{1}{C_s}, \tag{11}$$

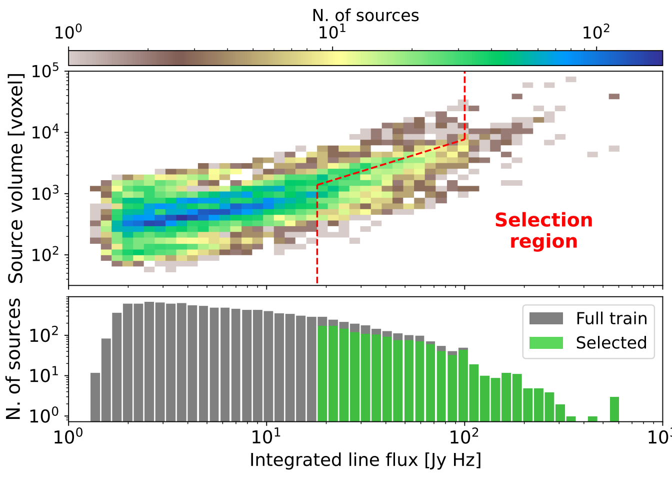

Fig. 2: *Top:* Two-dimensional histogram of LDEV truth source volume as a function of their integrated line flux. The red dashed line represents our selection function, with sources to the right of the line being selected. *Bottom:* line flux distribution for the full LDEV catalog and the selected sample.

where $\hat{S}_{deg}$ is the target source size in degree, B_s the synthesized beam size in degree, P_s the pixel size in degree, $\hat{w}_{20}$ the line width in km s^{-1}, c the speed of light in km s^{-1}, $\hat{\nu}$ the target central frequency in Hz, ν_{HI_0} the rest frequency of HI in Hz, and C_s the frequency channel size in Hz. The resulting volume brightness is in Jy Hz voxel^{-1}. The selection function is then defined as

$$(f \geq 100) \quad \text{or} \quad (f > 18 \text{ and } V_b > 0.013). \tag{12}$$

The exact flux and volume brightness thresholds are the result of an extensive manual search aimed at maximizing the score of a detector trained on the corresponding training sample. When applied to the LDEV truth catalog, this function flags 1383 of the 11 091 sources as detectable and thus usable as training targets. We represent our selection cuts over a 2D histogram of the source volume against the line flux integral for the LDEV truth catalog in Fig. 2, along with a comparison of the line flux histograms for the truth catalog with our selected training catalog.

2.4. Data cube preprocessing

We perform a series of preprocessing steps to enhance source detectability and prepare the data cube for use with our detector. We start by identifying the sky positions for which the mean signal over all continuum images is above 3×10^{-3} Jy beam^{-1} and set all the corresponding cube voxels to zero, masking continuum-subtraction residuals. We then normalize every frequency channel by its standard deviation. This reduces the impact of RFI subtraction residuals and makes the channel noise homogeneous across the whole bandwidth. It also allows us to consider the detection task as invariant with the frequency position. This way, a detector can be trained and applied over subcubes along the frequency axis.

Due to the high-frequency resolution, the source signal is diluted over multiple channels. This could be mitigated by smoothing the cube along the frequency axis or averaging every few channels, which would increase apparent S/N for some sources. However, it would destroy part of the fine-grained 3D structure that the detector could exploit. We observed that such preprocessing allows for faster training convergence but reduces the final score. We therefore retain the original frequency resolution to maximise detection performance.

The last step is to renormalize the cube. As for the SDC1, we apply a scaled hyperbolic tangent transformation to the voxels following $p'_i = \tanh(\alpha p_i)$ where p_i is the cube voxel value after the previous preprocessing steps, and p'_i is the final normalized voxel value. The parameter α controls the steepness of the curve and thus impacts the final signal contrast and saturation. After some exploration, we adopted $\alpha = 0.3$ for the SDC2. We normalize both the MAIN and LDEV cubes once and store them in a reduced unsigned 16-bit integer format for faster dynamic loading. We observed no impact of this reduction on the final score, provided the reduction is done after the tanh normalization. To further reduce the loading time, we split the MAIN cube into chunks of $512\times512\times6668$ (3.5 GB), cutting along the two sky axes. The cuts are made based on our prediction pipeline setup (Sect. 3.5), ensuring partial overlap and padding when necessary. The LDEV cube is left as a single file and fully loaded into system memory for the training phase (Sect. 3.3.1).

3. Method

3.1. YOLO-CIANNA generalization for 3D detection

In Paper I (Cornu et al. 2024), we provide a complete description of the YOLO-CIANNA method for detecting, classifying, and characterizing objects in 2D images. Here, we present adaptations of the formalism for detecting 3D objects. As explained in Paper I, YOLO-CIANNA is a regression-based detection method implemented as part of the broader `CIANNA` development framework (Cornu 2024)[2]. The YOLO-CIANNA method constrains a fully convolutional neural network (CNN) backbone to create a mapping from an input data space to a regular grid of detection units. Each unit is associated with an independent subregion of the input space and is tasked with detecting all objects for which the central coordinate lies within this region. Objects are represented as bounding boxes. During the training phase, the network is fed with labeled images. Each detection unit produces a set of candidate boxes that are matched and compared with the known list of target boxes. This comparison is then used to adjust network weights so future predictions will be closer to the target. Additionally, the model learns to predict an objectness score for each detection, representing its confidence that a given predicted box corresponds to a real object.

To generalize the formalism presented in Paper I toward 3D detection, we add two new dimensions to the detection units' output vector to predict both the central position z and the size d of a box along the third dimension. We express a 3D bounding box as a function of output vector elements, following

$$x = o^x + g_x\,, \qquad y = o^y + g_y\,, \qquad z = o^z + g_z, \tag{13}$$

$$w = p_w \mathrm{e}^{(o^w)}\,, \qquad h = p_h \mathrm{e}^{(o^h)}\,, \qquad d = p_d \mathrm{e}^{(o^d)}, \tag{14}$$

where g_x, g_y, and g_z are the output grid coordinates of the unit making the prediction, o^x, o^y, and o^z are predicted sigmoid-activated values refining the position inside the subregion, p_w, p_h, and p_d are predefined size priors, and o^w, o^h and o^d are preditect linear-activated values adjusting the size of the predicted box. The corresponding $\mathcal{L}_{\rm pos}$ and $\mathcal{L}_{\rm size}$ regression sublosses from Eq. 14 of Paper I are trivially adapted with a third dimension to obtain our full YOLO-CIANNA 3D loss.

We also update the box proximity and similarity metric used in many parts of the method. For this, we rely on the Intersection over Union (IoU) measurement generalized to 3D boxes. In this paper, we use its distance-aware variant, the DIoU (Zheng et al. 2020), bounded within $[-1, 1]$, and defined as

$$\mathrm{DIoU} = \mathrm{IoU} - \frac{\rho^2}{d^2}, \tag{15}$$

where ρ is the Euclidean distance between the two box centers and d is the diagonal length (in 3D, the space diagonal) of the smallest box that encloses the two boxes to compare. This proximity criterion is used both to define the target objectness and in the association function, as described in Paper I.

3.2. Network backbone architecture

Using the updated method, a classical 2D CNN backbone could be used for 3D detection by treating the third dimension as a stack of independent 2D input channels. However, taking advantage of spatial invariance through fine-grained gridding across all positional dimensions usually improves detection accuracy. Thus, we grid the third dimension into independent detection units as well. For this, we rely on `CIANNA`'s ability to build 3D spatial convolutional layers and design a fully 3D network backbone. The output layer is therefore directly in the form of a 3D output grid (indexed by g^x, g^y, g^z), and detection units are structurally centered on their associated input subvolume. This structure also reduces the number of learnable weights required in the model compared to a 2D backbone approach. More details about the interplay between our method and the network backbone can be found in Sect. 3.6 and Appendix A of Paper I.

Even though a fully convolutional architecture is not bound to a specific input size, we chose to use a fixed subcube input dimension of $64 \times 64 \times 256$. The ratio between the spatial and spectral dimensions has been defined regarding the typical source shape (Fig. 1) and constraints in both the training and prediction pipelines (Sects. 3.3.1 and 3.5). In the SDC2 data, and likely for any hyperspectral data as well, the signal distribution along the frequency dimension is very different from that of the two sky dimensions. Although all three axes are considered spatial in terms of the box definition and matching criteria, we use anisotropic filters and stride configurations to account for the intrinsic differences in signal distribution across dimensions. Our best-performing SDC2 backbone is presented in Fig. 3, which was obtained through an extensive search over the architecture and hyperparameter space using our SDC1 backbone as a starting point. The detailed configuration of our backbone architecture can be found in the example scripts hosted on `CIANNA`'s GitHub repository and archived at xx.xxxx/zenodo.xxxxxxxx[3]. As we discussed in Appendix B of Paper I, we observed that classical backbones constrained from generic image datasets underperformed on the SDC2 task, and even more if used pre-trained.

Despite being deeper with 23 convolutional layers, this architecture only contains 3.35 million learnable weights, which is a quarter of what we had in our 17-layer SDC1 backbone. This results from a reduction in the number of filters, which is motivated by the reduced number of sources in the training catalog (1383 sources, Sect. 2.3). Still, the lesser morphological diversity in simulated HI sources compared to simulated continuum sources allows the network to achieve high detection and characterization accuracy with fewer parameters. We note that the first layer convolves the frequency axis with filters spanning height

[2] CIANNA is open-source and freely accessible through GitHub https://github.com/Deyht/CIANNA. The version used in this paper corresponds to the V-1.0 release 10.5281/zenodo.12806325.

[3] Pre-review archive at https://share.obspm.fr/s/swyCT7BgEGjtZK3

channels with a stride of two, which searches for large spectral structures and reduces the frequency axis dimension, compensating for the absence of frequency smoothing in the data preparation. Like our SDC1 backbone, the first few layers progressively reduce both the spatial dimensionality and the number of filters, acting primarily as a small denoising subnetwork tailored to this detection task. We also include the typical darknet sequence of alternating 3×3 convolutions with 1×1 convolutions over a smaller number of filters (Redmon & Farhadi 2017). This forces the network to find more compressed representations between spatial pattern extractions steps, which reduces the number of learnable weights compared to stacking identically configured spatial convolutions. Here, these compression steps have a secondary purpose, which is to capture spectral patterns with 3D filters of size $1 \times 1 \times 3$. This increases the receptive field over the spectral dimension faster than for the two sky dimensions, which helps extract extended patterns along the frequency axis. All layers except the last one use a leaky-ReLU activation with a 0.1 leakage factor. Like our SDC1 backbone, we have a group normalization layer between C20 and C21 with a group size of two (Wu & He 2018), and a 25% dropout rate on the C21 layer (Srivastava et al. 2014), both for regularization purposes.

Even though we could reduce the spectral dimension to have a regular output grid in all dimensions, we achieve better spectral localisation by sampling it over a finer output detection grid. The total reduction factors for each dimension at the end of the network are then 8 : 8 : 16, and each detection unit is responsible for an $8 \times 8 \times 16$ input subvolume. The receptive field at the last layer is $66 \times 66 \times 294$, which corresponds to the input subvolume accessible to each detection unit to make a decision. Removing layers significantly reduces the achievable score, indicating that network depth compensates for fewer learnable weights in achieving the required expressivity for the task.

3.3. YOLO-CIANNA SDC2 configuration

The backbone architecture sets the output grid resolution, but the detection units themselves must be configured manually through a set of hyperparameters. In this section, all YOLO-CIANNA-related parameters follow the notations and definitions from Paper I. Some parameter values are provided for reproducibility and not necessarily discussed or commented on. The final configuration is the result of a thorough exploration of the corresponding hyperparameter space.

The typical volume source density in the SDC2 data is very low. Because our output grid already has a high resolution compared to the average source size, we observed that a single unit per grid element is enough to achieve the best score. If trained with more units, the detector often converges toward a solution where a single unit makes all the detections. This is likely due to the small training sample and to the absence of clear and balanced subgroups in the source parameter space (Sect. A). The background detection scaling factor is set to $\lambda_{\mathrm{void}} = 0.01$.

As stated in Sect. 3.1, our detection units represent objects as bounding boxes, which are essential to the training association function. As thoroughly discussed in Paper I, astronomical sources rarely exhibit the sharp and well-defined edges required to define bounding boxes. We therefore rely on a box proxy of size S' in both sky axes and D in the frequency axis for each source (Eq. 11). We rotate this box based on PA and search for the smallest box with edges aligned with the survey axis and that contains all rotated vertices, thereby defining our final target box. The impact of the inclination angle i is already included in the definition of the source HI size. However, the

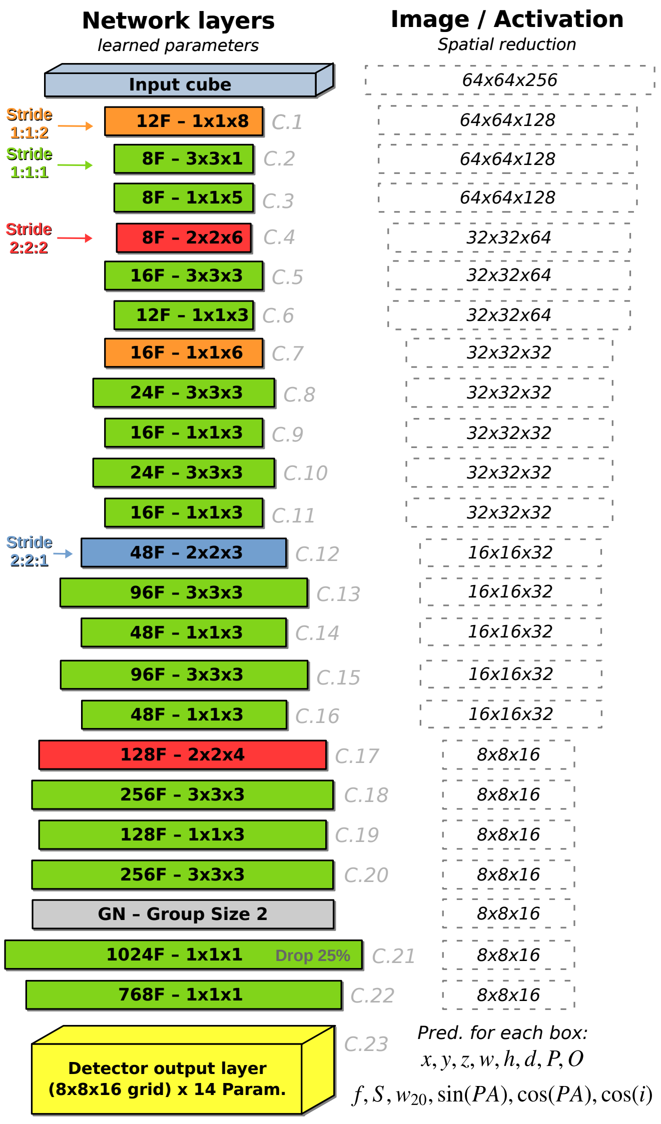

Fig. 3: Our SDC2 3D CNN backbone architecture. The *left* column provides layer structural properties, whereas the *right* column indicates the spatial output dimension starting from a $64 \times 64 \times 256$ input cube. Layers are stacked top to bottom. The layer width scales with the number of filters. Layer colors encode the stride setting: green preserves dimensions, orange reduces the frequency dimension by two, red reduces all dimensions by two, and blue reduces sky dimensions by two. The output grid size and the predicted parameters are shown next to the last layer.

resulting boxes remain small, which increases training difficulty as an unconstrained detector initially relies on fortuitous matches to identify the objects of interest. This problem would arise for every dataset, but it is more pronounced in 3D as positional errors across all dimensions are combined in the DIoU. To compensate, we increase the size of the target bounding boxes by 4 pixels in both sky axes and 10 channels in the frequency axis. We stress that this box definition has no impact on the HI size S and line width prediction w_{20} used in the scorer matching criterion, which are predicted as additional regression parameters. The corresponding target box is illustrated for a few sources in Fig B.1. From this, we set our detection unit box size priors to $10 \times 10 \times 36$. We observed that small changes in the target box

Table 2: Detection hyperparameters.

Param	γ^{f}	γ^{S}	$\gamma^{w_{20}}$	$\gamma^{\sin(PA)}$	$\gamma^{\cos(PA)}$	$\gamma^{\cos(i)}$
Value	3.0	2.0	2.0	1.0	1.0	2.0
Param	$L^{\mathrm{fIoU}}_{\mathrm{GBNB}}$	L^{fIoU}_{P}	L^{fIoU}_{O}	L^{fIoU}_{p}	$L^{\mathrm{fIoU}}_{\mathrm{diff}}$	$L^{\mathrm{obj}}_{\mathrm{diff}}$
Value	0.4	-0.1	-0.1	0.4	0.1	0.2

Table 3: Scaling factors and limits of output subparts.

	Pos.	Size	Prob.	Obj.	Param.
Activation	Sigm	Lin	Sigm	Sigm	Lin
λ	2.0	1.0	1.0	6.0	3.0
Pre-activ. scaling	0.5	0.5	0.2	0.5	0.5
Pre-activ. max	8.0	1.8	8.0	8.0	1.2
Pre-activ. min	-8.0	-1.4	-8.0	-8.0	-0.2

sizes can significantly affect the score, whereas changing the size of the prior only has a minimal effect.

Our single detection unit is configured to predict the box center coordinates (x, y, z), the box size (w, h, d), a detection probability P, and an objectness score O. To these mandatory elements, we add f, S, w_{20}, PA, and i, all expressed as individual linearly-activated output vector elements constrained by a regression loss subpart. All parameters are normalized in a similar range to balance their loss contributions. f is clipped in between [10,300] Jy Hz, log-scaled, and rescaled to [0,1]. S is clipped in between [4,25] arcsec and linearly scaled to [0,1]. w_{20} is clipped in between [100,700] $\mathrm{km\,s^{-1}}$, log-scaled, and also rescaled to [0,1]. To avoid tasking the network with impossible predictions, we set $PA = 180°$ for all small sources with $S < 6$ arcsec (beam size is 7 arcsec). PA is then converted from its [0,360] degree range to two independent parameters in the [-1,1] range through sin and cos functions, which works better than a direct prediction of the angle due to angular symmetries and periodicity. i is converted from its [0,90] range to the [0,1] range through the cos function, which also flattens the otherwise inhomogeneous distribution of i. In summary, with six extra parameters, all targets and predictions are expressed in the form of a 14-element vector $\langle x, y, z, w, h, d, P, O, f, S, w_{20}, \sin(PA), \cos(PA), \cos(i)\rangle$. The individual γ^{p} scaling values are given in the top row of Table 2.

The remaining method hyperparameters are listed in Tables 2 and 3. All fIoU thresholds are given for a DIoU-based association function. We note a few differences compared to the SDC1 setup, like the absence of association refinement parameters, which are not required with a single detection unit per grid element. Difficulty-flagging parameters are specified here, but discussed in the next section.

3.3.1. Training example generation and augmentation

Training examples are dynamically generated with augmentation from the normalized LDEV cube (Sect. 2.4) and the selected LDEV truth source catalog (Sect. 2.3). As discussed in Sect. 2.4, we make the approximation that the frequency axis is an invariant spatial coordinate, meaning we train our 3D detector to be insensitive to the input subvolume position on the frequency axis. We define two ways of generating training subcubes. The first one, in 70% of the cases, extracts a cutout at a random position in the LDEV cube regardless of its content. Sources with their central coordinate inside the cutout are added to the target list. This mostly returns empty cubes, but it exposes the network to the full diversity of background signals and artifacts. The second one, in 30% of the cases, draws a random source from the selected truth sample and extracts a random cutout within the subregion of the LDEV cube where the selected source is guaranteed to be included. If other sources have their center in the same region, they are also added to the target list. With this approach, we ensure that a minimum proportion of examples contains at least one detectable source.

Extracted subcubes are augmented to increase the diversity of examples. We add masks on the edges of the subcubes to mimic the effect of added padding when applying the method to the edge of a large cube. Masks are added independently to each axis at a 20% rate, always on a single side picked at random, and masking up to half the subcube in the given axis. We also apply flips at a 50% rate for each axis independently. Whereas this is a standard augmentation for sky axes, it is generally not used in the frequency dimension. It requires assuming that the source morphology is symmetrical over the frequency axis, which is a stronger approximation than treating the source frequency position as invariant. Still, enabling this augmentation results in an improvement in the achievable score of a few percent, indicating that the increased example diversity is more beneficial than the potential drawbacks of this approximation. Target PA values are updated to account for the flips when necessary.

The input resolution of the example subcubes is set to $64 \times 64 \times 256$. As we discussed in Sect. 3.8 of Paper I, when splitting a finite-sized volume, the choice of chunk size directly impacts the accessible diversity inside each chunk. The smaller the input volume, the higher the diversity, and the lower the chances of overtraining. A small context window also prevents the detector from inferring the position of the input within the large cube, which would cause overtraining. On the other hand, large input sizes are numerically efficient and reduce the proportion of the full volume affected by edge effects and truncated content when making predictions (see Sect. 3.5). We observed that adding augmentations improves the results, indicating that the intrinsic diversity of the LDEV cube is a limiting factor at the selected input size.

Our dynamic augmentation scheme does not fit the classical epoch definition. Instead, we group the generated examples into iterations over 1600 example subcubes. Due to the limited number of labeled sources in the LDEV catalog, splitting them into training and validation sets is impractical. Constituting a large enough validation set so it is statistically relevant would remove too many sources from the training sample. We therefore chose to use all labeled sources from the LDEV catalog for training and relied on occasional scoring over the MAIN SDC2 cube to verify the absence of overtraining, acting as a reliable validation and calibration dataset. During the challenge, the number of scoring was limited to prevent indirect optimization against the hidden MAIN catalog. Yet, it was possible to score a few save states from each trained model to test for blatant overfitting over the LDEV cube. In this paper, we use the scorer more extensively to provide a detailed analysis of our results. Still, it remains prohibitively costly to score more often than every few hundred iterations. Monitoring the training loss directly is not very useful, as most input examples are devoid of visible sources. We instead define a subset composed of cutouts centered on 800 random sources from the training catalog and use it as a proxy validation dataset to monitor the fine-grained loss evolution. Validation loss values are, therefore, only comparable within a given training.

3.4. Training setup and bootstrapping

Training is performed with the `CIANNA` framework, which implements our YOLO-CIANNA method, using an RTX 6000 Ada with (91 FP16 TFLOPS). We use FP16C_FP32A mixed-precision training, meaning that computations are done at a 16-bit floating point precision with 32-bit floating point accumulators (Micikevicius et al. 2017). All data flowing through the network uses the reduced 16-bit precision. A master copy of the weight is maintained at 32-bit precision and is used to accumulate the weight updates. Training at this precision yields similar scores to those obtained from full 32-bit training. We set the batch size to 16, which offers a good balance between training efficiency and memory footprint. The learning rate starts at 1.2×10^{-5} and increases linearly over the first 32 000 examples to 6×10^{-4}. It then decays exponentially as a function of the number of training iterations at a rate of 5×10^{-4} toward a minimum of 1.2×10^{-5}. The optimizer is a simple mini-batch stochastic gradient descent with a momentum set at 0.7 and a weight decay of 5×10^{-4}. A typical training over 6000 iterations takes around 36 hours on an RTX 6000 Ada, with an average processing speed of ~100 input cubes per second. The best score is generally achieved around iteration 4000.

We observed that small changes in the selection function described in Sect. 2.3 can significantly impact the final result, potentially even preventing training altogether. This is due to an oversimplified selection criterion that does not reflect the real capability of our detector. During training, sources that can be detected but are not included in the target list will be considered false detections, forcing the model to lower the detection probability of all sources that present similar characteristics. Conversely, sources that are undetectable but still included in the target list will force the model to increase the detection probability of background noise. These two types of mislabelling affect the fitting of the objectness curve, resulting in suboptimal detection performances. One possible solution is to rely on the detectors' self-assessed detection capability to refine the selection function. This is achieved by first training a BASE model from scratch with random weight initialization (using Glorot normal, Glorot & Bengio 2010) based on our handcrafted selection function. The best iteration is identified using the SDC2 scorer over the MAIN cube following the prediction pipeline. This model is then applied to the LDEV cube that was used for its training to produce a list of detections with a predicted score confidence. We extract the 1600 detections with the highest objectness score. From this, we define a new selection function with lower line-flux and volume brightness thresholds as

$$(f \geq 100) \quad \text{or} \quad (f > 14 \text{ and } V_b > 0.011), \tag{16}$$

but with the additional constraint that the selected sources must match one of the detections produced by the previous model based on a DIoU threshold of 0.1. By doing so, we include fainter true sources considered detectable by the first trained model. However, most added sources are difficult to detect and would likely complicate the training startup of a new model. To compensate, we use the difficult flagging introduced in Appendix A.6.4 of Paper I, and flag all the new sources that do not fulfill the original selection function. This allows the detector to learn from the obvious sources first and progressively refine its solution to include fainter ones. Sources fulfilling the original selection function but not detected by the first model could be removed, but we chose to also flag them as difficult instead. These missed sources might become detectable thanks to the added diversity provided by the new targets. We then train a new bootstrap model based on the refined training source catalog.

This bootstrap process can be repeated until the training sample converges. For the SDC2, we train a BASE model and two successive bootstrap models, BT1 and BT2. Each model is used to refine the following selection function, resulting in training samples containing 1383 (BASE), 1573 (BT1), and 1565 (BT2) target sources. This indicates a form of convergence after a single bootstrap step as the number of selected sources stabilizes, which is confirmed by the best score achieved by each model in Sect. 4.1. Instead of training every new model from scratch, we reuse part of the previously trained model as a starting point. We remove the last three layers from the loaded model and replace them with identically shaped layers with randomly initialized weights. Although this does not improve the final score, it allows faster convergence of bootstrap models.

3.5. Prediction pipeline

To apply a trained detector to the whole SDC2 MAIN cube, we decompose it into input regions along all three axes. To ensure that a source is fully contained in at least one input region, we need a minimum overlap of at least half the maximum source size in each dimension. In addition, the overlap must be a multiple of the backbone reduction factor to align subregions mapped by detection units from adjacent inputs, which is necessary for multiple detection filtering. We settled on an overlap of 8 pixels in the sky axes and of 32 channels in the frequency axis. This imposes constraints on the detector input size, which must be large enough to minimize the proportion of overlapping volume, as it directly impacts the inference pipeline efficiency. The required wide frequency overlap was one of our motivations in having 256 frequency channels per input (Sects. 3.2 and 3.3.1).

Storing the MAIN cube in system memory to decompose it dynamically is technically challenging. Thus, we rely on the cube slicing described in Sect. 2.4. The detector is applied to all input regions of all subcubes, preserving only detections with an objectness score above 0.1. Multiple detections within each input are filtered through a first non-maximum suppression (NMS) using a 3D DIoU threshold of 0.1. Multiple detections from overlapping inputs are also removed using a secondary NMS with a threshold of -0.3. Detections are then transformed back to the SDC2 catalog format by converting 3D positions to RA, Dec, and central frequency, and by inverting the normalizations for the predicted source parameters.

Classical computer vision tasks are often evaluated in a way that accounts for the detector's confidence score. In contrast, the SDC2 scorer expects a list of considered real detections. In practice, we can filter our prediction catalog using an objectness threshold deducted from the objectness histogram or by a rough estimate of the expected number of sources. The final threshold can be fine-tuned with a few calls to the scorer to optimize the result, which was our approach during the original challenge. However, to comprehensively analyze our results, we need to perform numerous and comparable scoring over the MAIN cube, which requires an automated threshold search. This is achieved by scoring an unfiltered catalog, from which we extract the list of matches from the scorer products, including the individual score of each source. By binning our catalog by objectness and computing the average score in each bin, we can identify the threshold below which the added sources contribute negatively to the score. Since the detection difficulty likely increases with the frequency, we split the detected sources into 20 frequency bins and optimize independent thresholds for each of them. We acknowledge that this post-process threshold optimization technically breaks the challenge conditions, but it only increases the achievable score by about 1% compared to a manual threshold.

On an RTX 6000 Ada GPU, with inputs of $64 \times 64 \times 256$, and using the FP16_FP32A compute resolution, we reach a processing speed of 300 inputs per second, or 315 million voxels per second. The GPU memory footprint in inference is around 450 MB per input in the batch, allowing the model to be used for prediction on lighter hardware. The total processing time for the 450 GB datacube is approximately 30 minutes, evenly split between GPU processing and data loading from a high-speed NVMe SSD. This excludes the preprocessing time of the raw MAIN cube, which depends strongly on the available system memory. Post-processing time to filter predictions is negligible.

4. Results

4.1. MAIN cube scoring

We present results from three successive bootstrap trainings, BASE, BT1, and BT2. We show the evolution of all natural loss subparts on our proxy validation dataset as a function of the iteration for these three trainings in Fig. 4. The score on the MAIN cube is computed periodically to confirm the absence of overtraining. We observe a loss plateau at the beginning of the training. This is well visible in the BASE model loss, but absent from BT1 and BT2 losses due to the use of pre-trained layers. This plateau corresponds to a regime in which the network is yet incapable of identifying the relevant signal for the task. Its length depends strongly on the method setup and backbone, with the worst-case scenario being an infinite plateau. The subspace of method parameters and architecture for which the network overcomes this state is small and difficult to find, highlighting the difficulty of the task. Selecting only the few hundred brightest sources as targets reduces the plateau's length, but this comes at the cost of fainter sources never being detected, resulting in a lower score. This is one of the motivations for using a bootstrap training approach, which allows the selection function to be refined over successive trainings. For a given selection function, there is no clear correlation between the plateau's length and the maximum achievable score when varying other parameters. Here, we optimized our method to reach the best score regardless of the plateau's length. The short-term instability of both the loss and the score is caused by the low average source density, resulting in substantial variations in the typical source content between two iterations. For each training, we select the iteration that results in the highest score on the MAIN cube as our final model state. In principle, the best iteration should be determined based on an independent validation dataset distinct from both the training and final test datasets. However, as discussed in Sect. 3.3.1, splitting the small LDEV catalog would cause stronger issues, leading us to consider the MAIN catalog as our validation dataset to identify the best iteration.

Table 4 presents a detailed score result comparison of the catalogs produced by our three models and of teams that scored above 10 000 points in the original challenge. This table highlights the number of candidate detections, the matches based on the scorer criteria, the false positives, the purity, and the average source score. A small description of each team's method and a detailed comparison of the produced catalogs are provided in the SDC2 summary paper (Hartley et al. 2023). Here, we summarize a few additional observations regarding the original challenge scores. ML methods were used for all or part of the pipeline of several teams, notably by the two top-scoring teams and three out of four teams with the lowest submitted score. This illustrates both the potential and the intrinsic difficulty of implementing such approaches for analyzing astronomical data. Also, most teams that applied denoising as a preprocessing failed to achieve a high score. This step likely removes signal from fainter sources, which are only detectable through their complex 3D signal distribution in a high noise environment. Although the first few layers of our backbone can act as a denoiser, the network learn to identify which signal should be compressed or preserved regarding the specific SDC2 task. Most teams using classical approaches relied on the same software, `SOFIA` (Serra et al. 2015; Westmeier et al. 2021). Team SOFIA notably achieved third place in the challenge. The second best-scoring team, FORSKA-Sweden, used a 3D U-net for source segmentation, but characterized them with `SOFIA` as well. Still, the remaining score difference between the classical `SOFIA` method and the two top-scoring approaches based on 3D CNNs highlights a specific strength of this type of model in extracting complex 3D patterns in high noise contexts (Appendix B). Interestingly, this contrasts with the results of Barkai et al. (2023), which performed a comparative study on a few HI detection methods and observed that `SOFIA` and the V-net 3D-CNN-based method performed similarly. This highlights again the importance of the architecture, method setup, and quality of the training data. Although some teams published a detailed version of their pipeline after the challenge, such as FORSKA-Sweden (Håkansson et al. 2023), there is currently no published post-challenge score comparable to ours. To facilitate future comparisons, all models and catalogs presented in the paper are made publicly available (Sect. 6).

Our new BASE model improves our top challenge score by 6.1%, and the BT1 model pushes this improvement to 9.5%, placing it 13.2% higher than the second top score. The number of matched sources also increases by 10.6% between our BT1 model and our top challenge score. We note that the typical variation in best score over multiple training of the same setup is typically around ±0.5%. Additional bootstrap trainings after the first one produce no significant improvement, except for faster convergence, as illustrated by our BT2 model. This indicates that our base selection function is already a good starting point. In Appendix D, we show that the bootstrap approach can help reach similar scores when starting from a degraded selection function. We highlight that our MINERVA v0.1 score was achieved through a combination of two methods, a prototype version of YOLO-CIANNA, and a hybrid classical and ML method called CHADHOC (Sect. 4.1.7 from Hartley et al. 2023). At the time, the YOLO-CIANNA prototype was limited to ~21 000 points, meaning that we improved our method-specific score by around 20%. The distribution of predicted sources is mostly homogeneous over all three dimensions of the MAIN cube. We represent cutouts centered on a few matched sources in Appendix B.

We represent histograms of the integrated line flux for both the matched and false detections in comparison to the complete MAIN truth catalog for BT1 in Fig. 5. The first one is based on the target values and used to compute the completeness as a function of the real line flux, whereas the second one is based on the predictions and used to calculate the purity as a function of the predicted line flux. We also represent a histogram of the matched prediction smoothed S/N, as defined in Sect. 6.2 of Hartley et al. (2023). This S/N estimate was added to the MAIN truth catalog after the challenge and is therefore only used for analyzing the results. Finally, this figure shows a 2D histogram of the matched sources volume against the target line flux. These plots show that our method can generalize outside our selection function, which is reinforced by the bootstrap steps, but is also the case

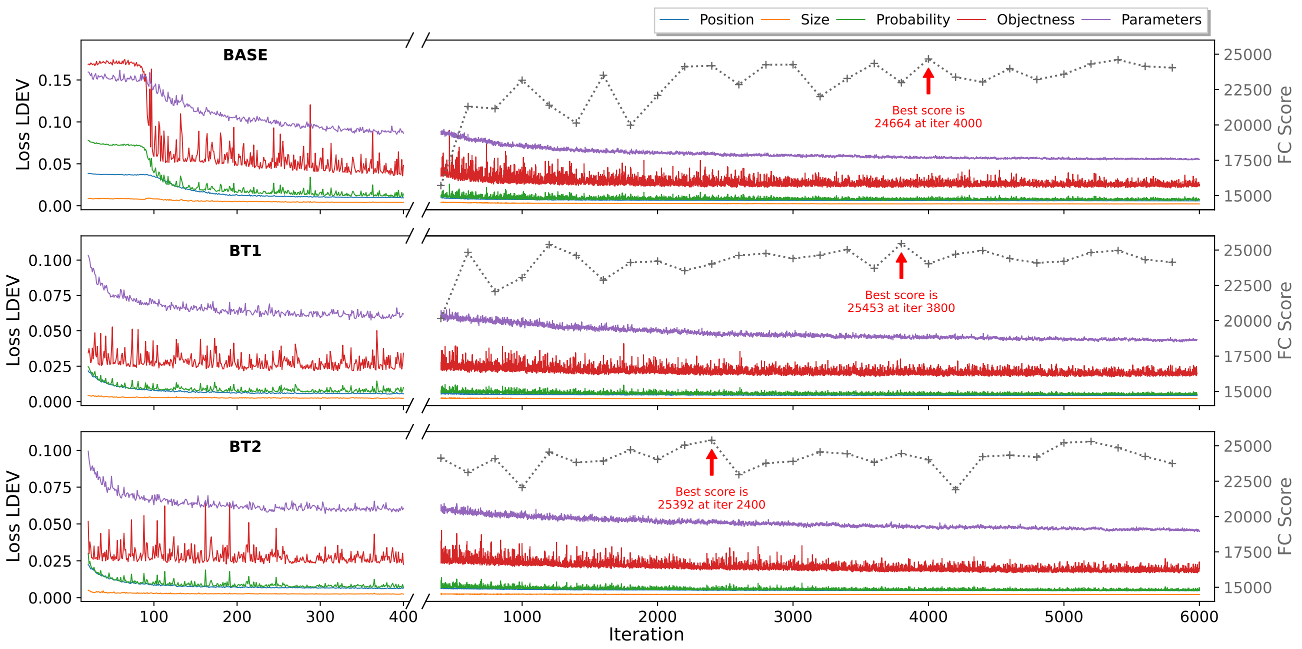

Fig. 4: Evolution of the validation loss subparts (natural, Sect. A.7 in Paper I) during training for our three successive models. The first 20 iterations are omitted. Scores computed over the MAIN cube are indicated every 200 iterations after 400.

Table 4: . SDC2 scores for source catalogs from different teams and methods.

Team / method / model	M_s (Score)	N_{det}	N_{match}	N_{false}	Purity	$\bar{s}$
Post-challenge results						
YOLO-CIANNA v1.0						
- BASE	**24 664**	36 360	33 404	2962	91.97%	0.8269
↪ *purity threshold*	18 459	22 126	21 927	199	**99.10%**	0.8509
- BT1	**25 453**	36 971	34 115	2862	92.28%	0.8298
↪ *purity threshold*	19 631	23 505	23 294	212	**99.10%**	0.8518
- BT2	**25 392**	36 438	33 785	2658	92.72%	0.8301
↪ *purity threshold*	19 484	23 354	23 135	220	**99.06%**	0.8517
Challenge results (M_s > 10 000)						
MINERVA v0.1*	**23 254**	32 652	30 841	1811	94.5%	0.81
FORSKA-Sweden	**22 489**	33 294	31 507	1787	94.6%	0.77
Team SOFIA	**16 822**	24 923	23 486	1437	94.2%	0.78
NAOC-Tianlai	**14 416**	29 151	26 020	3131	89.3%	0.67
HI-FRIENDS	**13 903**	21 903	20 828	1075	95.1%	0.72
...	...	...	...	...	...	...

Note. The bold elements highlight the optimized metric for each result.
*Combination of the catalogs obtained with a prototype version of YOLO-CIANNA 3D and CHADHOC (Hartley et al. 2023).

for our BASE model. False detections arise mainly in the low flux regimes where completeness and purity drop smoothly.

Our method produces a list of detections ordered by objectness confidence. Instead of selecting the threshold that maximises the SDC2 score, we can search for a threshold that enforces a purity of at least 99%. We report the scores and properties of these purity-optimized catalogs for our three models in Table 4. Interestingly, the resulting catalogs still score 10 to 17% higher than team SOFIA, the best non-ML approach. On the contrary, lowering the threshold results in more detection candidates. When applied to real observational data, these low-confidence detections could serve as follow-up targets in an observing program that aims to confirm some of them. The real strength of predicting a self-assessed confidence score is that a trained model samples the full detectability range. Therefore, as we did in Sect. 5.1.4 of Paper I, we compute an alternative metric similar to an average precision (AP). For this, we lower our initial objectness prediction clip to 0.05, order the prediction by objectness, and pass it to the SDC2 scorer to obtain the list of matches. We then compute the running precision (purity)

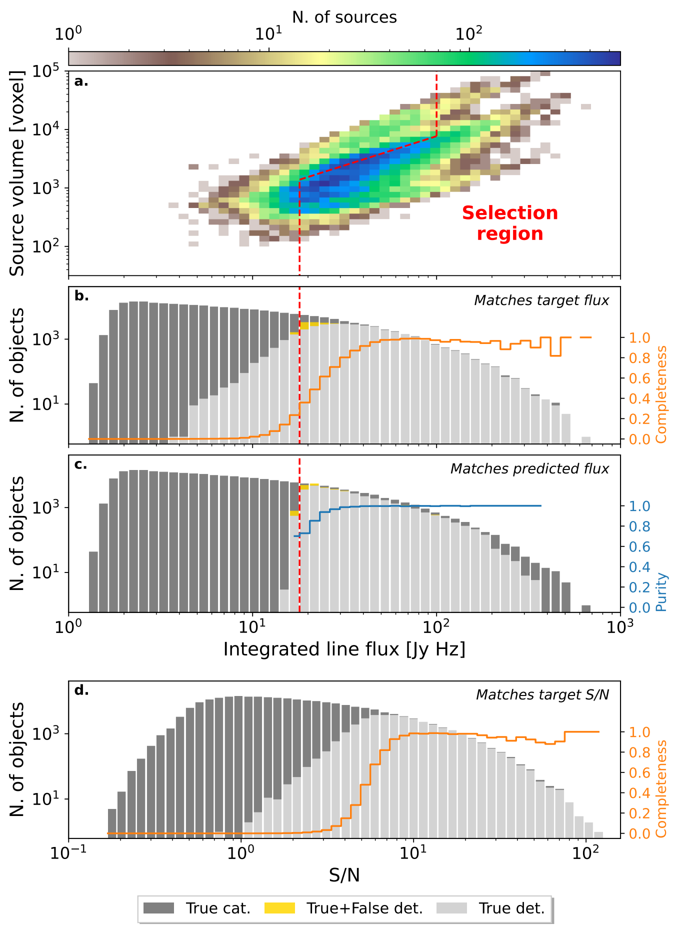

Fig. 5: (a) Two-dimensional histogram of the MAIN cube detection matches target volume as a function of their target integrated line flux. The red dashed line represents the selection function as in Fig. 2. (b) Histogram of the target line flux for the MAIN truth catalog and the matched sources. False positives are added based on their predicted flux. Completeness is overplotted for relevant bins. (c) Histogram of the predicted line flux for the matched predicted sources and false positives. The MAIN truth catalog is plotted in the background based on the target line flux. Purity is overplotted for relevant bins. (d) Histogram of the S/N for the MAIN truth catalog and the matched sources. Completeness is overplotted for relevant bins.

and recall (completeness) over the catalog and convert them into a smoothed precision-recall curve, whose integral defines the SDC2-based AP. We report AP values of 18.35, 18.32, and 18.34 for our BASE, BT1, and BT2 models, respectively. We also find the number of matches found regardless of the objectness value to be around 50 000 for all three models. To estimate the chances of random matches with a background source, we compute the AP of the BASE catalog with the 3D positions randomized, which produces only 361 false matches. This indicates that detections in the low objectness regime contain a fair amount of real detections, but at a purity lower than the SDC2 scorer requirement. We also compute a purely geometrical DIoU-based AP score for all models, which is presented in Appendix C.

4.2. Source characterization quality

Our BT1 model achieves an average characterization score of $\bar{s} = 0.8298$, improving our original challenge submission by 0.02, which was already the highest characterization score among all teams. Multiplied by the approximate 34 000 matches, this results in a direct increase of about 680 points. However, an average characterization score is not suitable for comparing methods with varying detection performances. This penalizes strong detectors that successfully detect many faint sources, which are likely also difficult to characterize. This is illustrated by our alternative purity-optimized catalogs that all achieve $\bar{s} > 0.85$ by selecting only the most obvious detections. A better approach would be to evaluate the average score on the catalog intersections for each pair of methods to be compared, so it is computed on the same list of sources. However, to date, the catalogs submitted by teams that participated in the original challenge have not been publicly distributed. This is one of the motivations for distributing all our SDC2 catalogs along with the paper, which should enable future comparison (Sect. 5).

We also highlight that the scorer uses relative errors for which the positive and negative sides are asymmetric, and for which the positive side is unbound and the negative side is limited to minus one. Combined with symmetric score response functions, it implies that catalogs that systematically underestimate the parameters achieve higher scores than those with a zero-centered error distribution. This limit is inherited from the SDC1 scorer as discussed in Sect. 5.1.3 of Paper I.

To thoroughly evaluate the characterization accuracy of our method, we rely on the detailed scoring information exposed by the scorer code. For each match, it provides the predicted and target source properties, the matching distance, and all error subpart values E as presented in Table 1, as well as the final source score recombined by Eq. 7. From this, we can draw individual error subparts histograms and compare the resulting error distribution to the associated score response function, which are represented in grey in Fig. 6. For each error subpart, we indicate the region of the distribution where the score is saturated using red dashed lines. The more the error distribution is contained in this area, the higher the average subpart score. To push the analysis further, for each error subpart, we identify an appropriate source target property against which it would be interesting to represent the error distribution. For example, we represent the flux error as a function of the target's integrated line flux, which is natural considering that E^{flux} is simply a relative flux error. The sky position, the HI size, and PA are represented against the target beam-convolved HI size. The frequency position, the line width, and the inclination are represented against the target line width. All these distributions are represented in the form of 2D histograms, presented alongside their 1D counterparts in Fig. 6.

From this figure, we observe that faint sources exhibit a strong positive error. This results from multiple effects. Firstly, the cube noise imposes a hard limit on the minimum predictable flux value. Secondly, the random fluctuation of the noise can either boost or reduce the apparent flux. Therefore, sources with a per-voxel flux close to the noise level will only be detected when the noise contribution is positive, acting as a selection effect of sources with overestimated flux. This effect is reinforced by the detector being unlikely to predict a flux lower than the minimum value it has seen during training. This is illustrated by the dashed black line over the 2D E^{flux} error distribution representing the error that would arise by systematically predicting the minimum training flux. Considering that the scorer includes the flux error in its matching criteria (Eq. 1), properly detected faint sources might be rejected based on high flux error, even though we reached the best achievable flux prediction for the noise level. Although the scorer does not expose the D_{tot} value for rejected sources, we observe that the faintest matches already have values

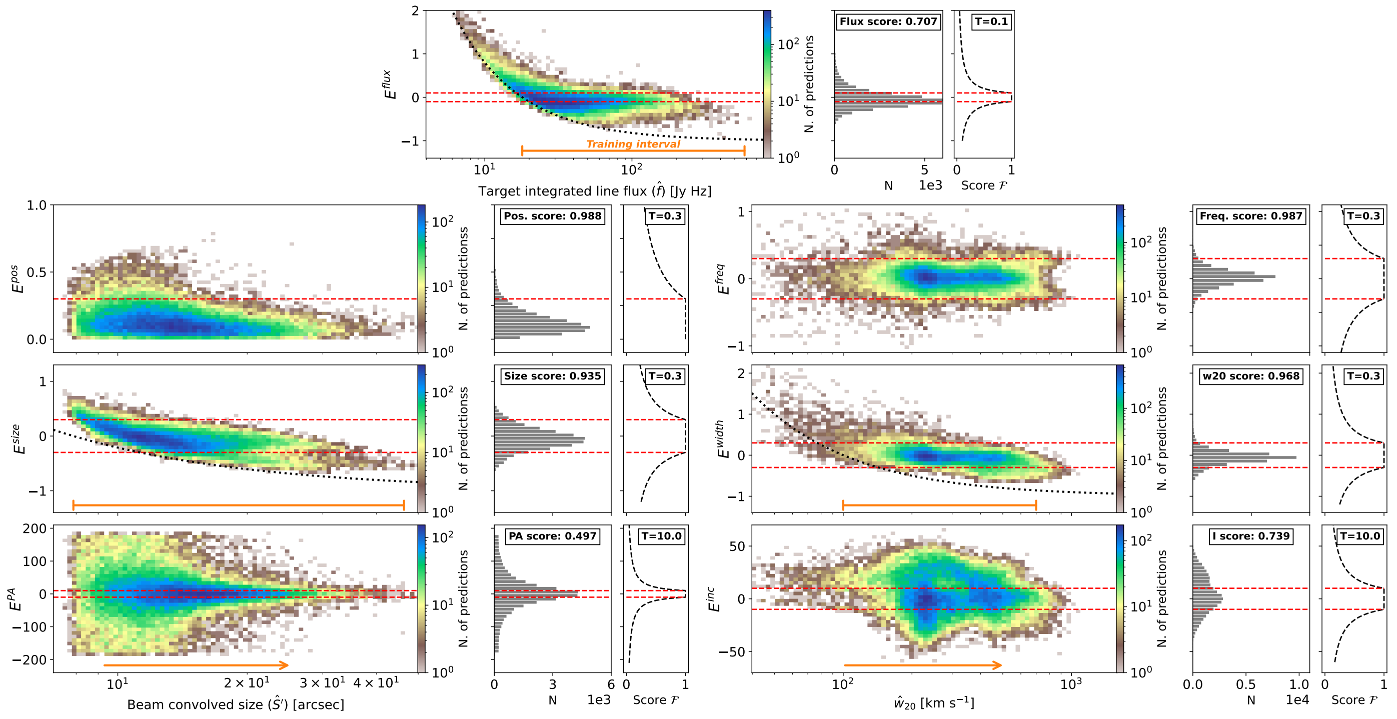

Fig. 6: Subpart error distributions, presented as 2D histograms against relevant target quantities or scales, along with the corresponding 1D error histogram and the associated score response functions. The red dashed lines indicate the error region for which the score is saturated to one. Orange lines and arrows show the range of values from the training sample when relevant. Dotted black lines represent the error obtained when predicting the minimum value from the training sample across the full comparison scale.

close to the $D_{\rm tot} \geq 5$ rejection limit, suggesting that some of our false positives might fall in this category. On the contrary, the flux of the brightest sources is underestimated, which is mainly due to the scarcity of bright examples in the training sample.

We observe similar behavior for both the source size and line width errors, also resulting from observational limits, specifically the beam size and channel resolution. These subparts have high average scores due to less strict error thresholds. Both the sky position and the central frequency are well characterized, which arises from the fact that poorly positioned detections have already been rejected by the matching criteria that are more strict than the associated subscores. We note that despite their resemblance, $E^{\rm pos}$ and $D_{\rm pos}$ are two different quantities, as they rely on different reference sizes ($S' \neq S''$), which explains why the position error distribution does not expand up to the matching rejection limit of one. Finally, the position angle is well characterized for sources with a sufficient HI size but predicted at random for small sources, which is also the result of the observational limit imposed by the beam size. The inclination angle achieves a better score. However, the substructures visible in the distribution as a function of the line width suggest that this angle might be inferred from correlated parameters rather than directly measured from the cube. We confirm that the network learns parameter correlations to some degree in Appendix A by representing the source distribution for a selected set of parameter couples.

5. Discussion

This second edition of the SKAO SDCs improves the challenge format in several aspects compared to the first one. The simulated cube includes more physical and instrumental effects, improving its realism, which is also a strength of the subsequent SDC editions (see the SDC3a, Bonaldi et al. 2025). The LDEV cube is a proper representation of the MAIN cube, allowing for training supervised ML methods without requiring representativity corrections like we implemented in Paper I. Still, the use of a similar scoring method to the SDC1 results in the same bias and limits, such as strong coupling between detection and characterization performance evaluation, or the absence of a confidence score weighting like in most computer-vision challenges. We also acknowledge that using simulated data provides an unrealistic advantage for training supervised ML methods compared to real observations. All sources are known and perfectly characterized, allowing for the definition of unambiguous targets, which would not be the case with observational labeled catalogs.

The results obtained with our YOLO-CIANNA 3D method, combined with the fact that the two top-scoring teams from the SDC2 utilized 3D CNNs, demonstrate that this type of structure offers significant advantages for processing noisy hyperspectral data. However, as discussed in Paper I, this requires building dedicated approaches that account for the specificities of astronomical data deeply in their design, which was facilitated by the low-level access granted by our `CIANNA` framework. Nevertheless, several aspects of our method could be improved. In terms of backbone architecture, `CIANNA` V-1.0 does not implement residual and skip connection layers (He et al. 2016), which limit the maximum network depth. Such layers have been added to the experimental build of `CIANNA`, which is currently only provided upon specific request but is expected to be publicly released on a short timescale. Unlike some versions of YOLO (Redmon & Farhadi 2018), we did not split predictions across scales, which is also related to current architectural limitations. In addition, having a fixed number of detection units on a finite-sized grid constrains both the positioning resolution and the density of detectable sources. This specific limit could likely be overcome with the addition of attention layers (Vaswani et al.

2017; Carion et al. 2020; Zhang et al. 2021) and some changes in the output layer definition. We also consider the possibility of pre-training our backbone using the Denoising Diffusion Probabilistic Model (DDPM, Ho et al. 2020) formalism. For example, we could have pre-trained our model over the MAIN cube without requiring any source targets. This would likely shorten the initial loss plateau, resulting in more stable training and possibly better results. Finally, we could inject context information, such as the instrumental setup, at multiple stages of the backbone to guide the source extraction (Lu et al. 2019). This would enable us to train generic detection models compatible with multiple setups or instruments.

The natural next step after achieving state-of-the-art results on simulated data is to generalize our method application to observational data from SKA precursors such as the LADUMA (Looking at the Distant Universe with the MeerKAT Array, Blyth et al. 2016) and WALLABY (Widefield ASKAP L-band Legacy All-sky Blind surveY, Koribalski et al. 2020) surveys. Our method could also be generalized to detect line emission in hyperspectral surveys covering a different frequency domain, such as MUSE. However, it requires a large set of labeled examples to build the training and validation sets. Although simulations can be used to generate large example datasets for pre-training highly parametric statistical models, they often lose accuracy when generalized to real observations. A complementary training phase on real labeled data is almost always necessary, and the calibration of the inference pipeline must necessarily be done on a labeled validation dataset. For this, we can combine labels from classical detection methods, crowdsourced science, or existing catalogs from other instruments. The supervised nature of our method implies that the trained detector will likely inherit uncertainties and biases from the methods used to define the targets. Combining independent labeling methods might help in this regard. In all cases, the bootstrap training approach can be employed to identify candidate detection that can then be refined using different labeling approaches or new observational programs, iteratively improving detection sensitivity. Preparatory works indicate that direct porting of our SDC2-trained models generalizes well over preliminary LADUMA data after a simple rescaling, producing candidate detection that is compatible with catalogs obtained with `SOFIA`. Specific retraining for this survey is currently being explored.

We conclude with an update of the numerical environmental footprint estimated in Paper I, which already accounts for most of the impact related to the method development and testing for 2D and 3D datasets. For this SDC2 study, we estimate the additional compute time at 4000 GPU hours, which includes the 1000 hours granted to MINERVA on Jean-Zay during the original challenge. Using the same estimates of 0.5 kW of average power consumption when training with a single GPU, and France's carbon intensity estimated by the ADEME for 2022 at 52 CO_2-e/kWh, the added impact is about 104 kg of CO_2-e.

6. Conclusion

In this paper, we generalized our YOLO-CIANNA deep learning and characterization method to 3D hyperspectral data from the SKAO SDC2. We detail the required modifications to the box definition, the position and size regression loss subpart, and the match association metric. We present a backbone optimized for HI line emission detection and describe the construction of our training sample, along with our training and inference pipelines.

Our results show that 3D CNN structures efficiently extract complex 3D patterns specific to HI sources and are capable of best-in-class source characterization. However, we highlight that implementing such a method on complex astronomical data is difficult and that only a narrow region of the parameter and architecture space allows successful training. We also introduce a bootstrap training strategy that relies on the self-assessed detection capability of a first poorly constrained model to refine the training selection function iteratively.

These state-of-the-art results are encouraging regarding the generalization of the method to observed surveys from SKA precursor instruments, with preparatory work currently underway on the LADUMA and WALLABY surveys. Although 3D CNN approaches should not be considered drop-in replacements for classical analysis tools, which often offer greater interpretability and consistency, our results demonstrate that deep learning detectors can efficiently produce complete and pure first-guest catalogs from large hyperspectral datasets.

Example scripts and notebooks are provided in the `CIANNA` git repository to help reproduce our results. We also publish all detection catalogs and trained models presented in this paper at xx.xxxx/zenodo.xxxxxxxx[4], along with ancillary models trained over the whole MAIN cube to serve as pre-training states for transfer learning toward other data or surveys (Appendix E).

Acknowledgements. We acknowledge the support granted to the MINERVA project by the Paris Observatory, including material and financial support. We acknowledge the Paris Observatory DIO shared computing resources and their handling of the MINERVA dedicated compute resources. We acknowledge the computing resources from the GENCI (Jean-Zay GPU partition) provided to the MINERVA team in the context of the SDC2. DC acknowledges the MINERVA project for funding a two-year post-doctoral contract. DC acknowledges its PSL fellowship as part of the MCA-IA program and in collaboration with PR[AI]RIE-PSAI. We highlight the great work done by SKAO in organizing all the SDCs and thank them for allowing post-challenge access to the datasets.

References

Barkai, J. A., Verheijen, M. A. W., Talavera, E., & Wilkinson, M. H. F. 2023, A&A, 670, A55
Blyth, S., Baker, A. J., Holwerda, B., et al. 2016, in MeerKAT Science: On the Pathway to the SKA, 4
Bonaldi, A., An, T., Brüggen, M., et al. 2021, MNRAS, 500, 3821
Bonaldi, A., Bonato, M., Galluzzi, V., et al. 2019, MNRAS, 482, 2
Bonaldi, A., Hartley, P., Braun, R., et al. 2025, arXiv:2503.11740
Braun, R., Bourke, T., Green, J., Keane, E., & Wagg, J. 2015, in Conférence: Advancing Astrophysics with the Square Kilometre Array, 174
Carion, N., Massa, F., Synnaeve, G., et al. 2020, arXiv:2005.12872
Cornu, D. 2024, Deyht/CIANNA: CIANNA V-1.0
Cornu, D., Salomé, P., Semelin, B., et al. 2024, A&A, 690, A211
Glorot, X. & Bengio, Y. 2010, in Proc. Mach. Learn. Res., Vol. 9, , 249–256
Hartley, P., Bonaldi, A., Braun, R., et al. 2023, MNRAS, 523, 1967
He, K., Zhang, X., Ren, S., & Sun, J. 2016, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
Ho, J., Jain, A., & Abbeel, P. 2020, arXiv:2006.11239
Håkansson, H., Sjöberg, A., Toribio, M. C., et al. 2023, A&A, 671, A39
Koribalski, B. S., Staveley-Smith, L., Westmeier, T., et al. 2020, Ap&SS, 365, 118
Lu, J., Batra, D., Parikh, D., & Lee, S. 2019, arXiv:1908.02265
Micikevicius, P., Narang, S., Alben, J., et al. 2017, arXiv:1710.03740
Redmon, J. & Farhadi, A. 2017, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
Redmon, J. & Farhadi, A. 2018, arXiv:1804.02767
Scaife, A. M. M. 2020, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 378, 20190060
Serra, P., Westmeier, T., Giese, N., et al. 2015, MNRAS, 448, 1922
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. 2014, Journal of Machine Learning Research, 15, 1929
Vaswani, A., Shazeer, N., Parmar, N., et al. 2017, arXiv:1706.03762
Westmeier, T., Kitaeff, S., Pallot, D., et al. 2021, MNRAS, 506, 3962
Wu, Y. & He, K. 2018, in Proc. of the ECCV
Zhang, Z., Lu, X., Cao, G., et al. 2021, in 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2799–2808
Zheng, Z., Wang, P., Liu, W., et al. 2020, Proceedings of the AAAI Conference on Artificial Intelligence, 34, 12993

[4] Pre-review archive at https://share.obspm.fr/s/swyCT7BgEGjtZK3

Appendix A: Source parameter space correlations

Both the MAIN and LDEV truth catalogs exhibit correlations between source parameters. As we observed in Sect. 4.2, the inclination angle error distribution exhibits substructures, indicating that the network likely infers this parameter from other source properties. We identified parameter pairs showing specific correlations and represent them in the form of 2D histograms for all matched sources from our BT1 catalog in Fig A.1. To determine if our detector reconstructs these correlations, we show the distribution for both the predicted and associated target values. We also display histograms for candidate detections rejected by the scorer's matching criteria and for the training sample to determine if the parameter space coverage allows for the identification of these correlations.

We observe a linear correlation between the flux and the HI source size. We also notice substructured distributions in both histograms that involve the inclination angle. Our detector fails to capture most fine-grained substructures, reducing them to mostly linear relations, which is likely due to the sparse sampling of the parameter spaces in the training sample. Still, the observed relations confirm that the model learns to correlate source properties, allowing it to infer some parameters from others that are easier to evaluate. Finally, we observe that most false detections are faint sources with a small HI size and a line width that falls in the lower half of the distribution. Although it is not illustrated here, we note that false detections are uniformly distributed in sky position and central frequency.

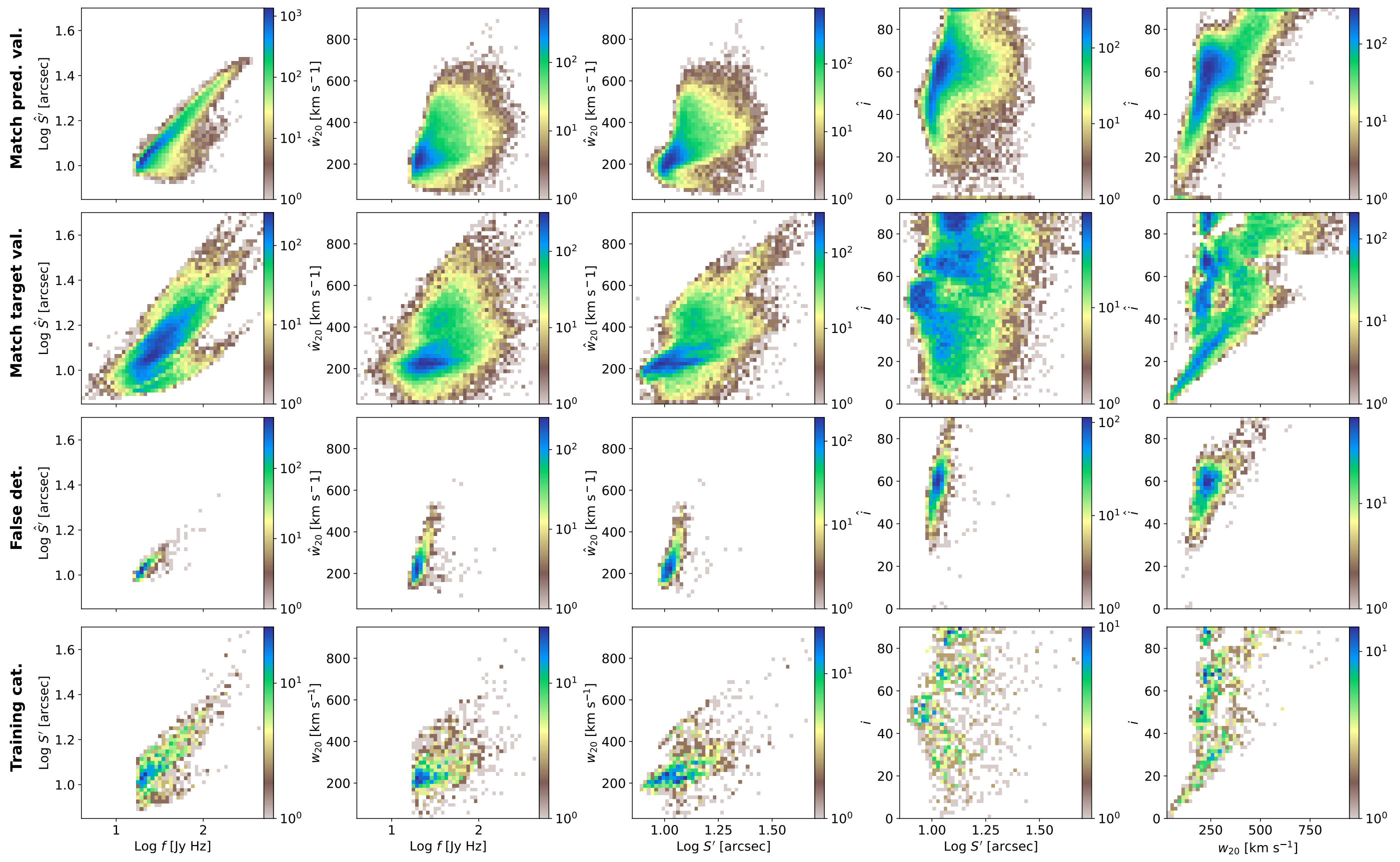

Fig. A.1: Two-dimensional histograms of selected source parameter pairs. Rows correspond to different source catalogs: *first* and *second* rows represent matched sources from the MAIN from BT1 using the predicted and target source values, respectively; *third* row represents false detections; *fourth* row represents the selected training catalog from the LDEV cube.

Appendix B: Local reprojection effect on detectability

Most participating teams started by visually exploring the SDC2 cubes, identifying only a handful of very bright sources. Smoothing the LDEV in the frequency axis can increase the number of visually identified sources up to a hundred ($\sim$ 1% of the LDEV catalog). Visualizing the source signal in 3D, like we did with Fig. 1, is only possible for the brightest sources. Fainter sources can be visualized by taking a cutout centered on the source coordinate and projecting their signal along one of the cube axes based on the source extension, which boosts the S/N. This per-source approach obviously outperforms global cube smoothing in enhancing the individual source signal. However, it requires knowing the position and extension of each source in advance, defeating the very purpose of a detection and characterization task. Still, this test confirms that around 10% of sources technically contribute enough signal to be detectable, which is still lower than our detector capabilities.

Here, we aim to identify potential explanations for the robust detection capability of our method on this task. We hypothesize that 3D CNNs can learn to recombine the cube information in a way that allows them to explore multiple 3D reprojections and

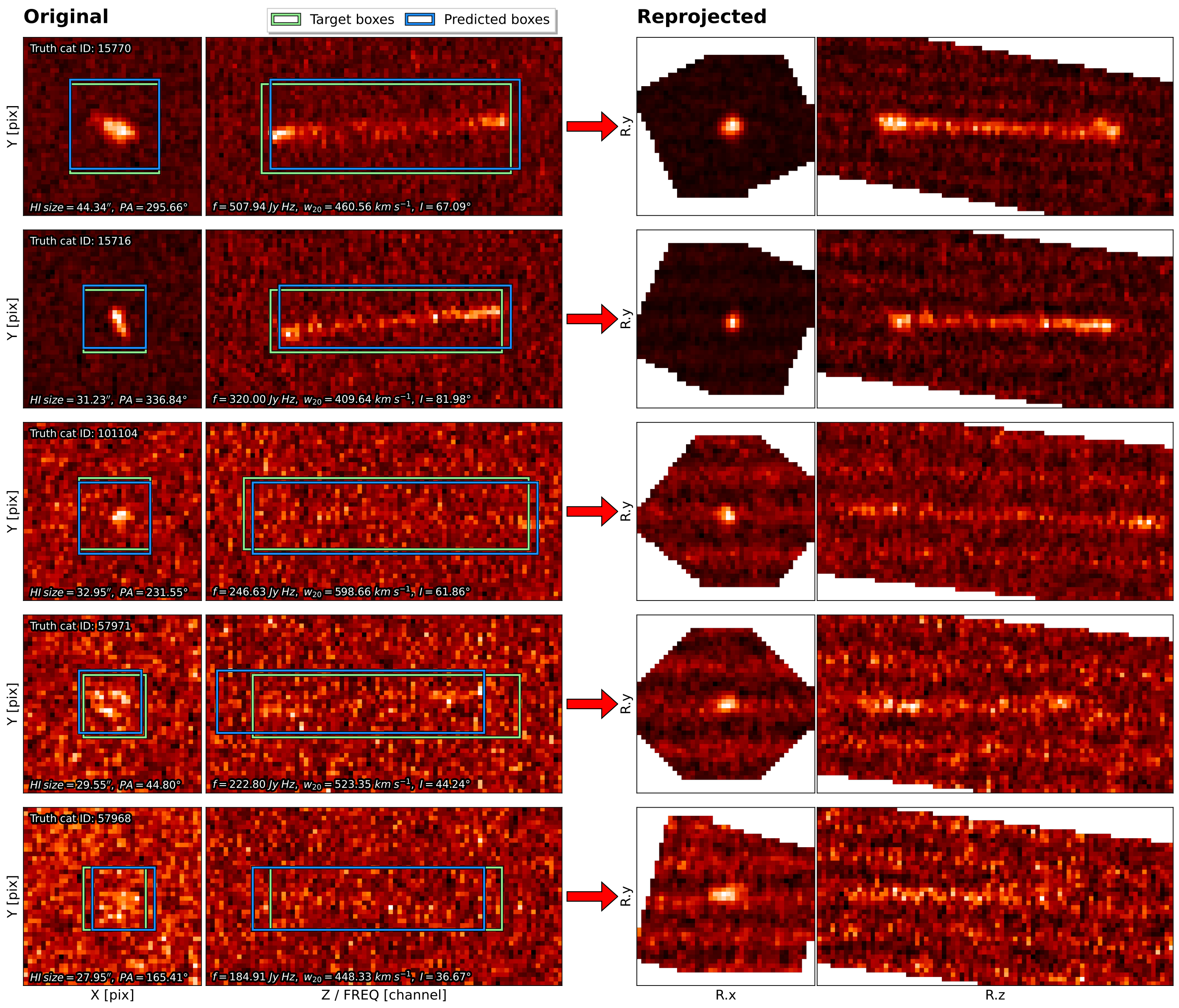

Fig. B.1: Signal projection for a subset of matched MAIN-cube detections. Each row corresponds to a source. The *left* column is obtained from cutouts centered on the source in the original MAIN cube, whereas the *right* column is obtained after reprojecting the cutout based on each source's *PA* and *i*. For each cutout, we present two 2D projections, one in the plane of the sky by averaging over the frequency axis, and the second in a sky-frequency plane by averaging over one sky axis. Images are in arbitrary units inversely proportional to the normalized voxel intensity. The colormap is adjusted to maximise the per-image contrast. The truth catalog `id` and the target source properties are indicated for reference. Boxes are overplotted on the *left* column for each source and represent the target box in green and the corresponding detector's prediction in blue.

various source extensions dynamically based on the input context. To test this, we can try to visualize reprojected sources into their preferred plane based on the truth angles. We do this for a few successfully detected sources from the MAIN cube and present the results in Fig. B.1. To better illustrate the effect of the reprojection, we select sources covering an extensive range of S/N and compare their 2D projections in the original cube and in the projected cutout. The first source is the same as the one represented in Fig. 1. Horizontal periodic lines visible in the reprojected space are artifacts resulting from the reprojection method we used. Boxes corresponding to our target definition and the associated network prediction are overplotted on the projections from the original cube.

As expected, we see that reprojecting based on known source characteristics boosts the signal. Any classical detection methods applied directly to this reprojected space would likely achieve similar performance to our detector. The strength of our method lies in its ability to identify such transformations automatically on the fly for each source. The fully convolutional 3D translational-equivariant structure of our backbone implies that our detector behaves similarly to simultaneously applying a given detector centered on every subregion mapped by our network, as defined by the output grid. Considering that this detector is built upon stacks of 3D convolutions, it can simultaneously scan for a vast diversity of 3D patterns that could typically emulate the expected projections. Achieving the same result with a classical method would require following a sliding window approach, testing all possible positions in the 3D cubes with multiple detector configurations to account for the possible reprojections, which would be technically prohibitive in terms of compute time.

Appendix C: Detection-only task performances

In Sect. 4.2, we discussed how the SDC2 scorer correlates the detection and source characterization and how it affects the evaluation of the detector's capability. Here, we test whether merging detection and characterization into a single model compromises pure detection performance. This question arises because the YOLO-CIANNA loss combines multiple objectives balanced by scaling factors (Eq. 14 in Paper I), which may compete against each other. This question is addressed for the specific case of the SDC1 in Appendix D of Paper I, where a DIoU-based AP score is used to evaluate the purely geometrical detection capabilities of our method across different training setups. It concludes that our combined detection and characterization model performs systematically better than equivalent models for which the parameter prediction subloss is deactivated. Although a naïve combined loss would lead to slightly lower detection performances, the cascading loss design in YOLO-CIANNA leverages the additional information, improving both detection and characterization results. These observations align with our general hypothesis that simultaneously predicting correlated variables often yields better results than making independent predictions.

In this section, we do the same verification for the SDC2 by computing purely geometrical 3D DIoU-based AP scores at different thresholds for our three reference models from Sect. 4.1, to which we add a model trained with parameter prediction turned off (No-param). Our bootstrap training approach is based on matches provided by the scorer, which requires source parameter predictions. Consequently, this new model is trained using the BASE model configuration and with our default selection function (Sect. 2.3). We compute the AP scores for DIoU thresholds of -0.3, 0.1, and 0.5, which correspond to the DIoU limits used in the second and first NMS of the prediction pipeline, and to a strict high-quality match criteria, respectively. The absence of source characteristic predictions prevents us from using the scorer to identify the best iteration. We therefore rely directly on the AP scores for the No-param model. Since we use the previously identified best iteration for all other models, this might grant a slight unfair advantage to the No-param scores. We present the AP scores in Table C.1, along with the scorer-based AP for our three reference models. The model without source characterization matches our BASE model AP scores for the first two thresholds but falls behind for the stricter one. It appears that the added parameter prediction helps improve source positioning and sizing, but has no substantial impact on source detectability. Still, we can confirm that simultaneous detection and characterization have no adverse effect on pure detection accuracy using our YOLO-CIANNA method on the SDC2.

Table C.1: Detection-only score metrics with and without the extra-parameter prediction enabled during training.

Model	$\mathrm{AP}_{-0.3}$	$\mathrm{AP}_{0.1}$	$\mathrm{AP}_{0.5}$	$N^m_{-0.3}$	$N^m_{0.1}$	$N^m_{0.5}$	AP_{SDC2}	N^m_{SDC2}	$\bar{s}$
With param. BASE	19.20	16.72	9.01	54022	46040	27476	18.35	51520	0.7816
With param. BT1	19.34	16.99	9.70	52524	45323	28567	18.32	49306	0.7908
With param. BT2	19.33	16.87	9.29	52768	45302	27806	18.34	49579	0.7903
No-param.	19.24	16.69	8.18	55996	46997	26807	N/A	N/A	N/A

Appendix D: Bootstrap from a degraded selection function

We show in Sect. 4.1 that our bootstrap training approach introduced in Sect. 3.4 allows us to improve the score from our BASE model significantly. However, the improvement may appear modest with respect to the increased training time and pipeline complexity. We hypothesize that our initial selection function is already near optimal, and that the bootstrap method would allow us to converge toward a similar final score if we started from a poorer selection. To test this, we define a degraded selection function as

$$(f \geq 100) \quad \text{or} \quad (f > 20 \text{ and } V_b > 0.016) \tag{D.1}$$

selecting 984 sources as detectable, which represents approximately 70% of the sources used for our original BASE training. Selecting only brighter sources should accelerate model training convergence, but is likely to result in a model unable to detect fainter sources. As we did for our reference training pipeline, we apply two successive bootstrap training steps, resulting in a total of three trained models, B-BASE, B-BT1, and B-BT2. We summarize the results in Table D.1.

The B-BASE model catalog scores 6% below our BASE model with a 4% drop in the number of matches and a lower average source score, which is consistent with a reduced training parameter-space coverage. After a single bootstrap step, the B-BT1 model already achieves a similar score to our BASE model. The second bootstrap further improves the score by placing B-BT2 in between our BASE and BT1 models, confirming our hypothesis. Achieving higher scores with additional bootstrap steps would likely require some adjustments in our bootstrap strategy. As detailed in Sect. 3.4, all new sources are flagged as difficult instead of confident targets, which gives more weight to sources from the original selection function compared to the new ones, which is an issue if the starting selection function is too conservative. When generalizing this approach to observational data, we would recommend refining the process to remove the difficult flag of some targets based on their predicted objectness score.

Table D.1: . SDC2 scores for models trained from a degraded selection function.

Model	M_s (Score)	N_{det}	N_{match}	N_{false}	Purity	$\bar{s}$
B-BASE	23 238	34 688	32 031	2664	92.34%	0.8084
B-BT1	24 657	36 548	33 612	2941	91.97%	0.8209
B-BT2	24 978	36 938	33 816	3130	91.55%	0.8310

Appendix E: Ancillary models for transfer learning

In this paper, we followed the original challenge conditions as closely as possible to produce results that can be compared with other teams. Here, we aim to take a step forward in generalizing our method to observed hyperspectral datasets by distributing models pre-trained on the whole MAIN cube. For this purpose, we modify our training pipeline to generate examples from the MAIN cube at each iteration. This results in a much larger training sample with many detectable sources to train from than what we had with the LDEV cube. This would technically allow us to constrain a more complex and wider backbone and to relax some hyperparameters. However, it would require a new tedious exploration of the associated parameter space. In addition, using the small LDEV catalog as a validation dataset makes it difficult to compare the performances of different setups accurately. Here, we instead use the exact setup from our LDEV cube training, for which we have confirmed the high level of detection performance. We use the same starting selection function as our BASE training, and perform two bootstrap steps, resulting in three MAIN cube models: MC-BASE, MC-BT1, and MC-BT2.

The inference pipeline works the same way as before, but is applied to the smaller LDEV cube instead to monitor for blatant overtraining. The objectness selection threshold is optimized against the LDEV score. We summarize LDEV scorer-based results in Table E.1. Due to the small number of LDEV sources, the scores we obtain are difficult to compare and likely sensitive to retraining variability. Still, we observe a clear 5% score improvement between MC-BASE and MC-BT1, while the additional score increase provided by MC-BT2 is negligible. We distribute these three models publicly, along with all other models presented in the paper, at xx.xxxx/zenodo.xxxxxxxx[5]. These MAIN-cube models should be used preferentially when transferring SDC2 pre-trained models to new datasets, as they have been exposed to greater example diversity. They are also the models used in the interactive SDC2 inference notebook provided in the CIANNA git repository, as it is practically easier to do the inference over the small LDEV cube.

Table E.1: . LDEV SDC2 scores for models trained on the MAIN cube.

Model	M_s (Score)	N_{det}	N_{match}	N_{false}	Purity	$\bar{s}$
MC-BASE	1179	1770	1614	156	91.2%	0.827
MC-BT1	1246	1761	1650	111	93.7%	0.823
MC-BT2	1262	1761	1652	109	93.8%	0.830

[5] archive at https://share.obspm.fr/s/swyCT7BgEGjtZK3