A&A, 689, A142 (2024)
https://doi.org/10.1051/0004-6361/202449150
© The Authors 2024

Astronomy & Astrophysics

Machine learning for exoplanet detection in high-contrast spectroscopy

Combining cross-correlation maps and deep learning on medium-resolution integral-field spectra

R. Nath-Ranga[1], O. Absil[1], V. Christiaens[1,2], and E. O. Garvin[3]

[1] STAR Institute, University of Liège, 19C Allée du Six Août, 4000 Liège, Belgium
e-mail: rakesh.nath@uliege.be
[2] Institute of Astronomy, KU Leuven, Celestijnenlaan 200D, Leuven, Belgium
[3] Institute for Particle physics and Astrophysics, ETH Zürich, Wolfgang Pauli Strasse 27, 8093 Zürich, Switzerland

Received 2 January 2024 / Accepted 23 April 2024

ABSTRACT

Context. The advent of high-contrast imaging instruments combined with medium-resolution spectrographs allows spectral and temporal dimensions to be combined with spatial dimensions to detect and potentially characterize exoplanets with higher sensitivity.

Aims. We developed a new method to effectively leverage the spectral and spatial dimensions in integral-field spectroscopy (IFS) datasets using a supervised deep-learning algorithm to improve the detection sensitivity to high-contrast exoplanets.

Methods. We began by applying a data transform whereby the four-dimensional (two spatial dimensions, one spectral dimension, and one temporal dimension) IFS datasets are replaced by four-dimensional cross-correlation coefficient tensors obtained by cross-correlating our data with young gas giant spectral template spectra. Thus, the spectral dimension is replaced by a radial velocity dimension and the rest of the dimensions are retained 'as is'. This transformed data is then used to train machine learning (ML) algorithms. We trained a 2D convolutional neural network with temporally averaged spectral cubes as input, and a convolutional long short-term memory memory network that uses the temporal data as well. We compared these two models with a purely statistical (non-ML) exoplanet detection algorithm, which we developed specifically for four-dimensional datasets, based on the concept of the standardized trajectory intensity mean (STIM) map. We tested our algorithms on simulated young gas giants inserted into a SINFONI dataset that contains no known exoplanet, and explored the sensitivity of algorithms to detect these exoplanets at contrasts ranging from 10^{-3} to 10^{-4} for different radial separations.

Results. We quantify the relative sensitivity of the algorithms by using modified receiver operating characteristic curves (mROCs). We discovered that the ML algorithms produce fewer false positives and have a higher true positive rate than the STIM-based algorithm. We also show that the true positive rate of ML algorithms is less impacted by changing radial separation than the STIM-based algorithm. Finally, we show that preserving the velocity dimension of the cross-correlation coefficients in the training and inference plays an important role in ML algorithms being more sensitive to the simulated young gas giants.

Conclusions. In this paper we demonstrate that ML techniques have the potential to improve the detection limits and reduce false positives for directly imaged planets in IFS datasets, after transforming the spectral dimension into a radial velocity dimension through a cross-correlation operation and that the presence of the temporal dimension does not lead to increased sensitivity.

Key words. methods: data analysis – techniques: image processing – techniques: imaging spectroscopy – techniques: radial velocities – techniques: spectroscopic – planets and satellites: detection

1. Introduction

Direct imaging combined with low-resolution integral field spectrographs (IFS), such as in VLT/SPHERE (Beuzit et al. 2019), Gemini/GPI (Macintosh et al. 2014), and Subaru/CHARIS (Groff et al. 2015), or with medium-resolution IFS, such as in VLT/SINFONI (Bonnet et al. 2004), VLT/ERIS (Davies et al. 2023), and Keck/OSIRIS (Cepa et al. 2000), allows exoplanet hunters to access both images and infrared spectra of substellar companions around nearby stars. Since the advent of such IFS instruments, their combination with high-contrast imaging techniques has strongly contributed to the direct detection of young gas giants such as PDS 70b (Keppler et al. 2018), or 51 Eridani b (Macintosh et al. 2015). In addition, the spectral dimension provided by the IFS has allowed us to characterize directly imaged exoplanets such as HR 8799bcde (e.g. Konopacky et al. 2013; Bonnefoy et al. 2016), β Pic b (e.g. Bonnefoy et al. 2014; Chilcote et al. 2017), HD 206893b (e.g. Delorme et al. 2017), PDS 70b (e.g. Müller et al. 2018; Christiaens et al. 2019), or HIP 99770b (Currie et al. 2023), and has provided clues on the nature of various low-mass substellar companions such as HD142527B (e.g. Christiaens et al. 2018) or the as-yet-unidentified type of companion CrA 9b (Christiaens et al. 2021). The early science operations of JWST/NIRSpec (Böker et al. 2022), including specific high-contrast imaging applications (Ruffio et al. 2023), further demonstrates the detection ability of such instruments.

However, these discoveries have been few and far in between because of the specific challenges posed by direct imaging data. One of the more difficult challenges is to be able to separate the exoplanetary signal from the ever-present stellar glare. The flux ratio between the exoplanet and its host star, termed contrast, can be as high as 10^{-10} for a Jupiter-like planet in a Solar System analogue in visible light (e.g. Galicher & Mazoyer 2023). Therefore, we work in the regime of high-contrast imaging (HCI), where the planetary flux-to-stellar flux ratio defines the sensitivity of our detections. In order to work in the high-contrast regime, a number of observing strategies have been employed, the most common for exoplanet imaging being angular differential imaging (ADI; Liu 2004; Marois et al. 2006) and spectral differential imaging (SDI; Sparks & Ford 2002). These two techniques can even be combined in a hybrid mode referred to as spectral angular differential imaging (SADI; e.g. Christiaens et al. 2019). While these strategies have their own unique features, in general, they produce a model for the stellar halo that can be subtracted from the data, leaving the residual data largely free of stellar contamination. However, unsubtracted speckle signals in the data can still be interpreted as exoplanet companions. Therefore, astronomers are limited to contrasts where it is possible to confidently differentiate between the exoplanet companions and speckles. These achievable sensitivity limits are generally much shallower than what would theoretically be allowed by the fundamental photon noise limit.

A possible way out of this confusion is to combine high-contrast imaging data with high-resolution spectra to improve the detectability of exoplanets, as suggested by Snellen et al. (2015), who propose to rely on the exoplanetary molecular absorption lines present in the spectra. The molecular absorption lines produced in the atmosphere of a Jupiter analogue have long been used to characterize the atmospheres of spatially unresolved exoplanets using cross-correlations with template spectra (e.g. Snellen et al. 2010; Birkby et al. 2013) that contain molecular absorption features (e.g. Baraffe et al. 2003). The additional presence of spatial information in spectroscopic data allows astronomers to converge on the region in the spatial dimension where the molecular cross-correlation is the highest. Such maps are generally referred to as molecular maps, as introduced by Hoeijmakers et al. (2018), who produced detection maps for H_2O and CO in the SINFONI datasets of β Pic. In order to produce such maps, known molecular templates are cross-correlated with the data. This cross-correlation produces coefficients that can be placed in the same pixel positions in lieu of the original photon values to produce molecular cross-correlation coefficient maps. These molecular maps can now provide spatial and spectral characterization of the exoplanet. However, deriving quantitative constraints on molecular abundances in an exoplanet atmosphere from cross-correlation measurements requires using a non-trivial atmospheric retrieval framework (Brogi & Line 2019). Instead, exoplanet characterization based on medium-resolution IFS spectra has therefore generally relied on the extraction of the (possibly continuum-subtracted) exoplanet spectrum, followed by atmospheric retrieval, as performed for instance on HR 8799e (Mollière et al. 2020), HR 7672AB (Wang et al. 2022), YSES 1b (Zhang et al. 2022), or HR 4747B (Xuan et al. 2022).

In addition to the challenge of retrieving abundances from cross-correlation measurements, another crucial question is how can astronomers claim a detection based on the value of the cross-correlation coefficients alone? A common method of claiming a detection is to estimate whether the cross-correlation coefficient with the companion spectrum is 5σ higher than the cross-correlation noise level. In spatial photon-based maps, such detections are usually subjected to measures such as spatial S/N (Mawet et al. 2014a). An equivalent of such an S/N computed from the cross-correlation coefficient vector itself has been attempted by several authors (e.g. Hoeijmakers et al. 2018), but this still suffers from an auto-correlation bias in the cross-correlation values and from the non-Gaussianity of the noise. Some remedies for the auto-correlation bias have been suggested by Ruffio et al. (2019), who measure the noise separately and then divide the auto-correlation signal out. However, these techniques also continue to rely on the assumption that noise is Gaussian in distribution. There is also the intrinsic difficulty in measuring the S/N within a correlation pattern, as described in Mâlin et al. (2023) and the solutions therein. There is thus a lack of consensus in defining the 5σ limit of detection when using cross-correlation maps. In addition, the S/N computation method used by some authors (e.g. in Cugno et al. 2021) requires computing large radial velocity cross-correlations, which are computationally more intensive and need to be computed one pixel at a time, which also makes them unsuited for large datasets.

In order to tackle some of these issues, we propose the use of supervised machine learning (ML) techniques to exploit the multi-dimensional character of IFS datasets. In the literature, ML techniques have already been proposed to process the spectral and spatial dimensions separately for direct exoplanet characterization tasks. In particular, Fisher et al. (2020) demonstrated that using high-resolution spectra in an ML framework is not efficient, and proposed to feed the ML models with cross-correlation coefficients instead. We therefore only consider cross-correlation maps, which preserve the multi-dimensionality of the datasets, where the wavelength dimension is replaced by the radial velocity of the cross-correlation and the photon values by the cross-correlation coefficients. The presence of these multiple dimensions allows us to explore the potential of using ML algorithms to claim detections as opposed to using metrics such as S/N. Using spatial data with ML algorithms has had particular success in the direct imaging community, to detect point-like sources (Gomez Gonzalez et al. 2018; Cantero et al. 2023) or to build the reference PSF to be subtracted in ADI images (Gebhard et al. 2022; Flasseur et al. 2023). However, research in the field of exoplanet detection using spectral and direct imaging data with ML algorithms still has plenty of scope to develop. This is a gap that this paper, and a companion paper (Garvin et al. 2024), will be addressing. While in this paper we focus on how spatial, spectral (i.e., replaced by the cross-correlation vector), and temporal dimensions can be most effectively leveraged together in a single ML framework. The companion paper focuses only on the cross-correlated spectral dimension. This companion paper provides a more instrument-agnostic approach in as much as the spatial dimension is treated separately, which allows us to use this approach on long-slit spectroscopy datasets as well, for instance. Both of these studies aim to illustrate the as-yet-untapped potential of ML techniques to tackle spectrally resolved exoplanet datasets for exoplanet detection. Having two parallel and complementary studies allows us to explore the potential of ML techniques in a more comprehensive manner without having to force either study to be a one-size-fits-all solution.

The main research question of the present paper is whether ML can improve upon the detection capability of non-ML algorithms, taking advantage of all the available dimensions in high-contrast IFS datasets. This involves addressing some intermediate questions, such as the definition of an appropriate non-ML algorithm to compare with, and the definition of performance metrics. To this end, this paper is organized as follows. Sect. 2 describes the data, its spatial and spectral pre-processing, the generation of cross-correlation cubes, and the insertion of simulated young gas giants. Section 3 describes cross-correlation map-based detection algorithms, including one non-ML algorithm and two ML algorithms that we developed for this project. Section 4 describes the important process of preparing our data to train the ML algorithms. Section 5 describes how results from the ML and non-ML algorithms can be evaluated with the same baseline comparison methods. Finally, in Sect. 6 we discuss the reasons why using data in specific ways allows us to achieve higher detection sensitivity, and conclude with the limitations of our study[1].

2. From 4D IFS tensors to cross-correlation tensors

This section describes the step-wise transformation of the data from 4D IFS (i.e. 2D spatial, 1D spectral and 1D temporal) tensors to cross-correlation tensors that can then be used to train ML algorithms. We start with describing these 4D tensors in Sect. 2.1. We follow by describing the injection of simulated young gas giants in this data in Sect. 2.2. Finally, we describe the conversion of these 4D IFS tensors into 4D cross-correlation tensors in Sect. 2.3.

2.1. Observations and data calibration

The ESO pipeline calibration includes flat-fielding, wavelength calibration, detector linearity correction and extraction of spectral cubes from each raw detector frame. For this study, we used a dataset obtained on HD 179218 with the SINFONI instrument on the VLT (Bonnet et al. 2004; Eisenhauer et al. 2003) as part of ESO program 093.C-0526. We chose this dataset because of the absence of any known companion. The data was calibrated as in Christiaens et al. (2018), using the SINFONI pipeline implemented in the ESO common pipeline library (EsoRex version 3.10.2; Abuter et al. 2006), apart from the sky subtraction, which was performed manually before running the ESO pipeline. The ESO pipeline calibration includes flat-fielding, wavelength calibration, detector linearity correction and extraction of spectral cubes from each raw detector frame. Each spectral cube is composed of ~2000 channels spanning from 1.45 to 2.45 µm (i.e. H and K bands). This is the spectral dimension. Each of the calibrated spectral cubes are then used as input for further preprocessing. Each spectral frame is made of 64 × 64 pixels with a plate scale of 12.5 mas pixel^{-1}. However, due to a twice coarser vertical sampling, each pair of consecutive rows have the same values. This forms the spatial dimension. We have a sequence of 83 such spectral cubes captured in pupil-tracking mode, with the instrument rotator turned off. This constitutes the temporal dimension. Each of the calibrated spectral cubes are then used as input for further preprocessing. The wavelength vector is recalculated by cross-correlating the sky signal present in spectral channels at the edges of the H and K band channels with the expected ESOCalc transmission profile following Hoeijmakers et al. (2018). All of the pre-processing is carried out in accordance with the recipe set out in the SINFONI data reduction pipeline (Abuter et al. 2006). We performed basic pre-processing of the data in order to:

- correct bad pixels in all images,
- determine the centroid location of the star and
- produce the final wavelength solution of the cube.

We identify bad pixels with an iterative sigma-clipping algorithm, and correct them with a 2D Gaussian kernel, using routines of the Vortex Image Processing (VIP) package (Gomez Gonzalez et al. 2017; Christiaens et al. 2023). We resample each image vertically by a factor two, to avoid the redundant rows. We find the stellar centroid by fitting a 2D Gaussian model to each spectral channel. These fits also enable us to estimate the FWHM of the PSF in each channel. In this paper we explore the detection of a population of planets which are relatively young (typically a few tens of Myr old) where they still retain a large fraction of their formation entropy, with a $T_{\rm eff} \approx 1000{-}1500$ K. These exoplanets are called young gas giants because of their age and gas giant composition. In order to simulate the presence of a young gas giant in the data we inject a simulated young gas giant at a specific radial distance and position angle (PA) from the frame centre. We use the substellar templates characterized by their effective temperature ($T_{\rm eff}$) and surface gravity ($\log(g)$), and contain detailed spectra which model the atmosphere for brown dwarfs, late M-type stars, and young gas-giant planets or young gas giants. Young gas giants are particularly interesting as they seem to be present in the 5−20 au distance range Bryan et al. (2016). For this paper we use the models from BT-SETTL as it allows us to have a large enough wavelength range in the infrared. With the BT-SETTL models we have access to wavelengths between 1 and 30 µm. We choose a young gas giant template with a $T_{\rm eff}$ = 1300 K and a $\log(g)$ = 3.0. The wavelength vector is recalculated by cross-correlating the sky signal present in spectral channels at the edges of the H and K band channels with the expected ESOCalc transmission profile following Hoeijmakers et al. (2018). The spectral frames are then cropped around the identified stellar centroid positions,which falls typically at ≈2−3 pixels from the centre of the frames during the observation. This step both recentres the images on the star and removes artefacts that affect spaxels at the edge of the frames. This in turn reduces the field of view of the frames to 61 × 61 pixels. Finally, we collate all re-centred cubes into a single master 4D tensor. The tensor now has 83 temporal cubes with 2000 wavelength bins with 61 rows and 61 columns of pixels.

2.2. Injection of simulated young gas giants in the data

In order to simulate the presence of a young gas giant in the data, we inject a simulated planet using a template spectrum at a specific radial distance and position angle (PA) from the frame centre. In this paper we use the template set provided by the BT-SETTL models (Allard et al. 1997, 2011; Baraffe et al. 2003).We use substellar templates characterized by their effective temperature ($T_{\rm eff}$) and surface gravity ($\log(g)$), containing detailed model spectra for the atmosphere of brown dwarfs, late M-type stars, and young gas-giant planets. For this paper we use the BT-SETTL models they allows us to have a large enough wavelength range in the infrared, ranging between 1µm and 30 µm. For most of our test, we choose a young gas giant template with a $T_{\rm eff}$ = 1300 K and a $\log(g)$ = 3.0. When injecting this template, we follow these steps:

1. the template is re-sampled to the same spectral resolution as SINFONI and the wavelength range is limited to that of SINFONI (i.e. between 1.4 and 2.4 µm);
2. we broaden each of the bins in the template spectrum to match the spectral PSF width of the SINFONI instrument (derived from the user manual);
3. we remove the spectral bins between 1.8 and 1.91 µm corresponding to the telluric absorption range, and then

[1] The code used in this paper is available at `https://github.com/digirak/NathRanga2024.git`

re-normalize the spectrum so that the total number of photons in each spectrum is 1;

4. we then multiply the total photon count in the spectrum by a factor of the total photon count in the spectrum extracted at the located of the star in our IFS dataset (this factor is the contrast);
5. finally, we chose a target pixel and add a PSF-scaled equivalent of the young gas giant spectrum in a FWHM-sized aperture around that pixel.

We use routines from the VIP package to perform the spatial injection at a specified radial separation and position angle, and at a given contrast with respect to the star. Every insertion in each temporal cube is done taking into account the parallactic rotation of the frame (or spectral cube in this case). In order to rule out the possibility of a real young gas giant biasing our tests, the simulated planets are inserted using the opposite parallactic angles to the ones associated with the data, and these are later used for final derotation of the frames.

2.3. Producing cross-correlation cubes

After injecting fake planets into our 4D IFS data cubes, we now proceed with the computation of the cross-correlation, which consists of two steps: (i) reducing the amount of stellar feature contamination in the data, and (ii) cross-correlating each spaxel with a similarly pre-processed version of a BT-SETTL planetary template with similar T_{eff} and $\log(g)$ to the one used for the injection. The idea of using a similar but not exact same template is to take into account the possible mismatch between BT-SETTL models and real planets, and avoid a too optimistic set of cross-correlation tensors. The chosen templates are typically picked two grid points away from the injected template. This is expected to preserve a relatively high cross-correlation with the injected template, as cross-correlations are not very sensitive to T_{eff} and $\log(g)$. Picking templates further away from the injected template within the BT-SETTL grid may result in unrealistically low cross-correlation values. In practice, when performing planet detection in a yet-undetected planetary system, a large number of templates will generally be used, which is expected to result in relatively high cross-correlation in at least a few cross-correlation tensors. This is the situation that we try to reproduce here.

2.3.1. Stellar spectral feature subtraction

To remove any lower order residual contamination, we compute the spectrum filtered using a first order Savitzky-Golay filter (Savitzky & Golay 1964) with a window size of 101 and subtract it from the extracted spectrum. The original spectrum is then replaced by this residual spectrum at the same pixel. Before computing the cross-correlation on each spaxel of our IFS cubes, it is important to first reduce the amount of stellar light in the data. While classical HCI post-processing techniques such as ADI or SDI rely on the spatial structure and behaviour of stellar speckles to remove them from the data, here we rely on the spectral behaviour following the first step of the High Resolution Spectral Differential Imaging (HRSDI) introduced by Haffert et al. (2019). This step consists of constructing a reference spectrum that contains the stellar spectral features and then dividing (or subtracting, in the case of Haffert et al. 2019) this reference spectrum from spectra extracted from every pixel using aperture photometry. To produce a reference spectrum we have the following sequential steps:

1. we first extract the stellar spectrum using aperture photometry at each frame centre with an aperture radius of 1 FWHM for each wavelength for every temporal cube;
2. we then normalize this extracted spectrum by the total photon counts for every wavelength in every cube;
3. and finally we remove the telluric bins (1.81–1.93 μm) by masking the values in these bins.

This spectrum is the reference spectrum, which we compute for every temporal cube so that we end up with as many reference spectra as temporal cubes (in our case 83 spectra). We then divide the rest of the spectra extracted from each of the spaxels in each temporal cube by its own reference spectrum.

2.3.2. Cross-correlation of the template with target spectrum

Cross-correlation is a method used to compare two different spectra. In this paper, such a comparison is made between an extracted spectrum and the BT-SETTL template spectrum, which is downsampled and broadened to match the SINFONI spectra (same as when it was injected) and then mean-subtracted as described above. Cross-correlation is the result of introducing a small velocity shift between the model and the extracted spectrum and computing the product of the two. This product is the cross-correlation coefficient and it is computed for every velocity shift. The cross-correlation C between the model template M_λ and the extracted spectrum (F_λ) at different velocities v is thus given by

$$C(v) = \sum_{\lambda} F_\lambda \times (M_{v,\lambda} - M_{\mathrm{SG,v},\lambda}), \qquad (1)$$

where v is the velocity shift, $M_{v,\lambda}$ the velocity-shifted version of M_λ, and $M_{\mathrm{SG},\lambda}$ the model template after applying the Savitzky-Golay filter. In this paper, we sample the velocities at every 10 km s^{-1} between −100 and 100 km s^{-1}, resulting in a total of 20 velocity bins. Thus, for each pixel, F_λ is now replaced by $C(v)$. This lowers the total size of the data although the number of data dimensions remain the same, and we have a 4D tensor once again with the dimensions $(83 \times 20 \times 61 \times 61)$ where the 20 corresponds to v replacing the 2000 wavelength bins.

3. Detection algorithms

Cross-correlations as defined in Sect. 2.3.2 are typically used to compare spectra, by comparing the value of the cross-correlation coefficient computed at $v = 0$ (i.e. $C(0)$) to the standard deviation of the cross-correlation signal $C(v)$ at higher velocity shifts. This ratio is then used to define the signal-to-noise ratio (S/N) for the cross-correlation. This, however, biases the noise estimation for a cross-correlation because of the presence of auto-correlation. Thus, the noise is typically overestimated. However, if the standard deviations are computed at sufficiently high velocity shifts from the central peak (e.g. for $|v| \geq 250$ km s^{-1} in Hoeijmakers et al. 2018) then this bias is somewhat mitigated. Even then, Ruffio et al. (2019) show that this problem is not completely mitigated.

Direct imaging data as used in this paper allows us to produce cross-correlation maps for a simulated young gas giant as described in Sect. 2.3. The image dimension gives us access to neighbouring pixels where the noise of the cross-correlation can be measured as suggested for example by Patapis et al. (2022). All of these methods rely on computing a large set of cross-correlations for every spaxel making the problem computationally intensive. An alternative way is to use the spatial S/N

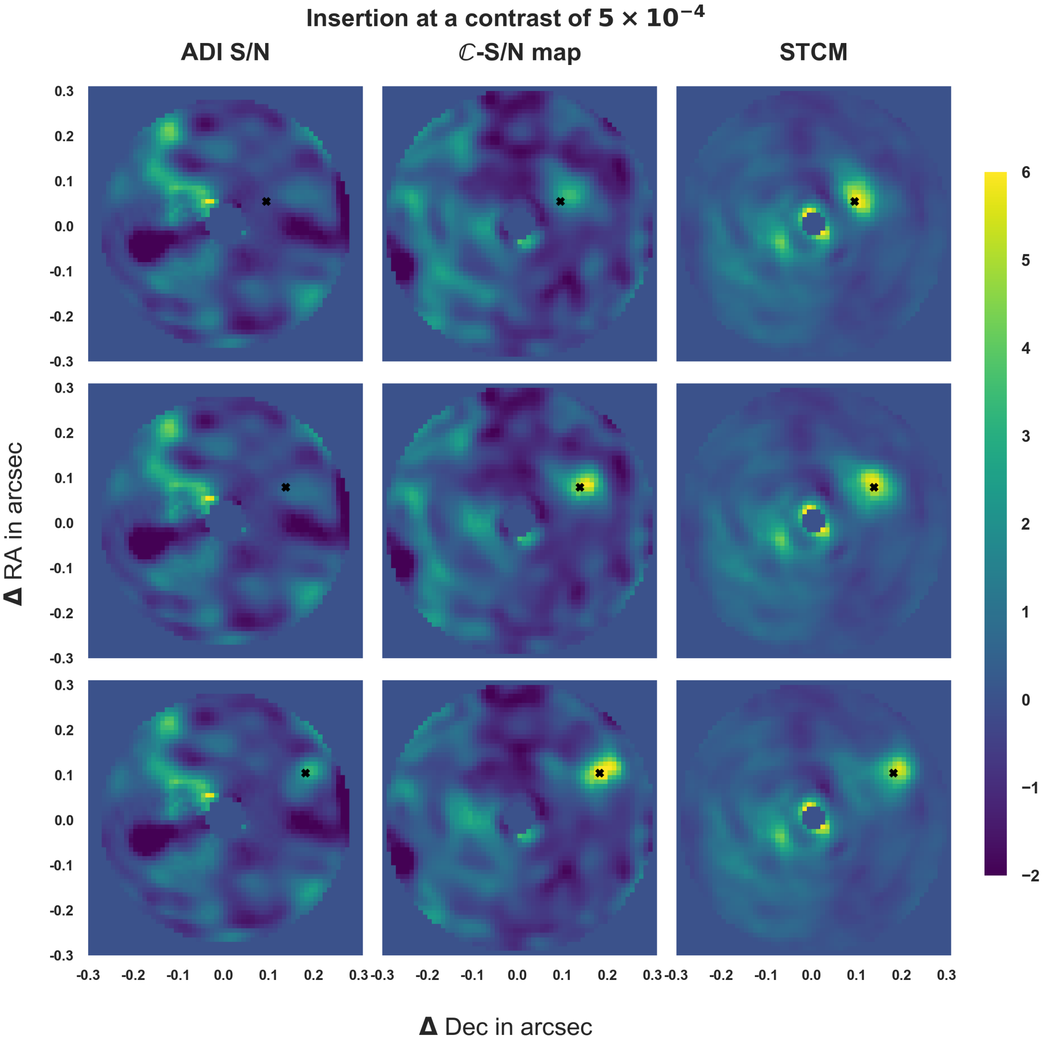

Fig. 1. Detection maps obtained with different algorithms, where a young gas giant was inserted at a specific contrast at different separations from the frame centre. Column 1 corresponds to a classical ADI algorithm, where an intensity map is computed for each wavelength, and the S/N map is derived from the median of all the ADI-wavelength maps. Column 2 is obtained by computing an S/N map on the cross-correlation map obtained with $v = 0$ km s^{-1}. Column 3 represents the STCM map, produced as described in the text. The rows represent maps produced when the same companion is inserted at different radial separations from the frame centre, respectively at 2.3, 3.3, and 4.3 FWHM. The simulated young gas giant is inserted at the same contrast of 5×10^{-4} in all maps.

definition, that does not rely on the cross-correlation noise at all, and is commonly used in ADI algorithms as defined in Mawet et al. (2014a). Instead of using this S/N definition, which once again relies on some definition of spatial noise distribution,and thus implicitly making assumptions about noise distributions, we choose to adapt the standardized trajectory intensity mean (STIM, Pairet et al. 2019) map to our cross-correlation cubes. A major advantage of using STIM is that it makes use of the temporal axis in the residual cubes, which has shown to reduce false positives in ADI cubes.

3.1. Standardized trajectory correlation mean (STCM) maps

Detecting an exoplanet in an imaging dataset requires the estimation of the signal versus the noise distribution of the dataset, and tools such as S/N maps provide a means of doing this as demonstrated by Mawet et al. (2014a). Here, we introduce the standardized trajectory correlation mean (STCM) map as an analogous way to estimate the relative strength of the signal. To produce an STCM map we perform the following steps. We first compute the cross-correlation tensor as described in Sect. 2.3.2 and derotate the cubes so that the pixels at which the simulated young gas giant was inserted now align in all the cubes. Next we choose the velocity at which $C(v)$ is the highest for this pixel[2] and recompose the tensor into a cube consisting of the spatial and the temporal dimensions only. Finally the STIM algorithm is applied to this cube along the temporal axis, resulting in an STCM map.

In Fig. 1 we demonstrate the application of three different ways of estimating the relative signal strength of the simulated young gas giant that we insert in the data as described in Sect. 2.2. Each column in Fig. 1 is the detection map for an injection at a fixed contrast of 5×10^{-4} produced using different algorithms, with each row corresponding to a different radial separation for the injection. The first column is produced when we compute an S/N map (as defined in Mawet et al. 2014a) after

[2] Since we do not apply an explicit frame shift and both the template and the spectrum are at the same rest frame, the highest $C(v)$ should be at $v = 0$ in principle.

performing classical ADI on each 4D tensor. In order to do this we compute the ADI for a series of data cubes at every wavelength. Then, we compute the mean ADI cube from this result. The second column is produced by calculating the S/N map (as defined in Mawet et al. 2014a) on the cross-correlation cube in two steps: first we compute the median along the temporal dimension (after applying appropriate de-rotation), and then we compute the cross-correlation map and apply the standard spatial S/N to it. The third column is produced by the STCM algorithm. The range of values on each of these maps is normalized to be between −2 and 6 so that we get a fair visual estimation of the noise values produced by each of these algorithms, based on the global minima and maxima of these nine maps. From the figures it appears that the STCM detection map recovers the exoplanet more consistently across different separations. We interpret this as being caused by the original STIM algorithm design, which aims to alleviate small sample statistics to perform better than standard S/N maps at small separations.

3.2. Map-based detection through deep-learning algorithms

Application of artificial neural networks and particularly deep neural networks to exoplanet detection is now an established method in astrophysics (Fluke & Jacobs 2020). The exoplanet community has benefited from the use of deep neural networks for Kepler light curves (Pearson et al. 2018), for direct imaging detection (Gomez Gonzalez et al. 2018), and to accurately produce a simulated model of the PSF of an instrument using half-sibling regression (Gebhard et al. 2022). We seek to use deep-learning algorithms to produce maps akin to the STCM maps described in the previous section.

The building block of a neural network is a neuron. A neuron consists of an input that is operated on by an activation function, resulting in an output. The activation function is usually a non-linear mathematical operation (for instance a hyperbolic tangent) that is performed on the input. A neural network usually contains more than one neuron, and a set of such independent neurons each operating on a part of the input is called a layer. If the output of this layer is used as input to another layer, this layer is called a hidden layer. When one or more hidden layers are present, such a neural network is called a deep neural network. A layer of neurons can have multidimensional input: typically a deep neural network is used with one dimensional data and a two dimensional deep convolutional neural network is used with image data. Typically, memory-based networks such as a long short-term memory and transformers can be used with data that have some temporal coherence. Deep neural networks have the versatility to also use images through 2D convolutional neural networks (CNN, Shi et al. 2015) and to combine images with temporal data in convolutional long short term memory (convLSTM, Hochreiter & Schmidhuber 1997; Liao et al. 2022). Deep neural networks have been applied to astronomical spectra (e.g. Leung & Bovy 2019; Tao et al. 2020) to classify them and specifically in direct imaging. Fisher et al. (2020) characterized exoplanets using cross-correlations of the spectra from the HARPS-N spectrograph. A neuron, despite its input dimensionality, has a set of weights and biases as parameters. These parameters form the model of the neural network and are produced when we train the algorithm on data. Training is the term used to make the neurons identify intrinsic relationships between different data dimensions and the necessary output. In this case we train the neural networks in this paper to predict the presence of an exoplanet in the input cross-correlation tensor. Training typically consists of providing the neural networks with large number of labelled examples of the classes one wishes to predict. The neural network in turn predicts the probability of each class given the input data. Training neural networks is very important to achieve the desired result; we explain this in detail in Sect. 4.

In this section we describe the two different deep neural network architectures that are used in this paper. Architectures are typically defined by the number of neurons, number of layers, and the kind of neurons used. Thus, while describing the deep-learning algorithms we have two parts, the first being its architecture, which is necessary to explain the neuron type, its activation function, and its output, and the second being its training methodology. We explore the detectability of young gas giants with different dimensionalities of the data with both a CNN-based deep neural network and a convLSTM-based network. When using the CNN-based network, we compute a temporal mean reducing the dimensionalities to x, y, and v. For the spatial-velocity dimension, we use a CNN-based deep-learning framework called Cross-Correlation and Convolutional neural network-based exoPlanet detectOr (C3PO). In order to include the temporal dimension, we also introduce a 2D convolutional long short-term memory (LSTM)-based algorithm called Cross-correlation and convolutional LSTM based exoplANet DetectOr (C-LANDO). We have designed these algorithms so that they are able to produce probability maps akin to the STCM maps. The input to these deep-learning algorithms will be the cross-correlation tensors provided pixel by pixel and the output will be the probability of that pixel containing an exoplanet.

3.2.1. C3PO

Our architecture consists of a set of convolutional layers and set of dense layers finally terminating in a single neuron, which provides a probabilistic output. The x and y dimensions of the convolutional layer is called kernel size. Several such kernels are stacked in a 3D block and the depth is the number of such kernels that are stacked. The input image is passed through each of these layers in a block for each feedforward pass. In our case, an input consists of the cross-correlation coefficients in the x, y, and v dimensions. The pooling layer downsamples the kernels of the convolutional layer based on a summary statistic of the output of the convolutional layer (e.g. maximum value). The output of this layer is then flattened to remove 3D elements and then connected to a multi-layer perceptron, with a sigmoid activation as final layer. The latter will act as the classifier output, which is the probability of a pixel to contain a simulated young gas giant. The detailed architecture of C3PO is as follows:

1. a convolutional layer with kernel size of $(3, 3)$ with 50 such kernels with a hyperbolic tangent activation function;
2. a pooling layer that uses the maximum value statistic with a pool size $(2, 2)$ such that at this point the data dimensionality follows the dimensions set by C3PO, and we now have a new dimensionality to the output based on the kernel size, number of velocity bins, and number of such kernels;
3. a convolutional layer with kernel size $(2, 2)$ with 25 such kernels with the same activation function;
4. another pooling layer of size $(1, 1)$;
5. a final convolutional layer of kernel size $(1, 1)$ with 25 such kernels;
6. a fully connected layer with 128 neurons with a rectifying linear activation function, followed by a dropout and final neuron with a sigmoid activation.

The architecture of C3PO is illustrated in Fig. 2. We provide more details on the input dimensionalities in Sect. 4.

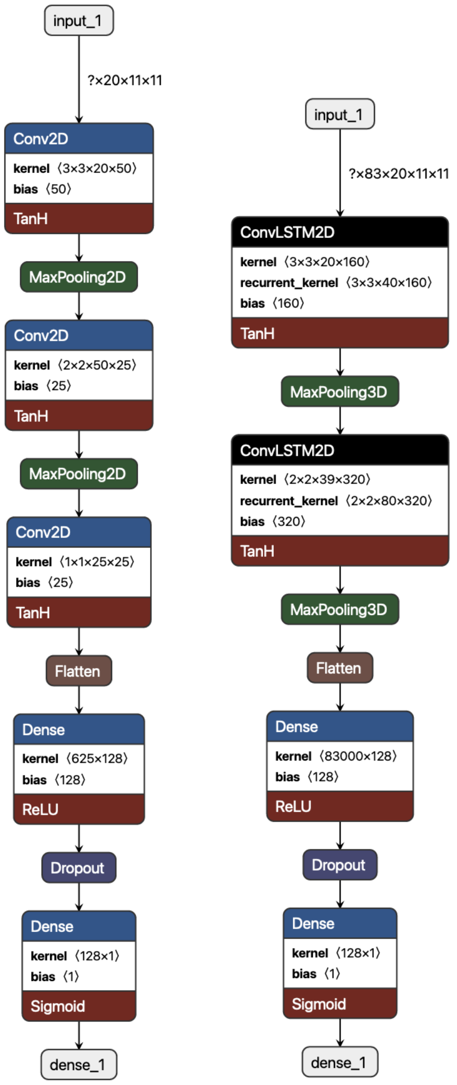

Fig. 2. Schematic of the C3PO (left) and C-LANDO (right) architectures, showing the different layers and sizes of the input, the dilation of this input at different layers, and finally the output format. The top grey block represents the input, with dimensions printed next to the arrow (e.g., for C3PO: 20 velocity bins, 11 × 11 pixel image size as explained in Sect. 4, and batch size given as input labelled with a question mark). Each block represents a layer and has three parts: the blue or black part represents the type of neuron, the white part is the number of neurons represented by a convolutional kernel size and number of bias units, and the red part represents the non-linearity (hyperbolic tangent). The kernel has four dimensions: the first three represent the input shape and the last represents the depth of the kernel. The kernel shape also represents the output dimensions of a layer and the input to its following layer. Between each layer we have pooling layers marked in green, a flatten layer in brown, and a dropout layer in the end. The output is just a single neuron with a sigmoid activation (dense_1).

3.2.2. C-LANDO

The basic unit of C-LANDO is a convolutional LSTM, in the same way as the basic unit of C3PO is a CNN. A convolutional LSTM is a combination of the standard LSTM (Hochreiter & Schmidhuber 1997) with a convolutional neural network (Zhang et al. 1990), which allows this kind of network to combine C3PO features of velocity and spatial PSF structure with the temporal structure embedded in the 83 temporal cubes that are present. Our convolutional LSTM architecture intends to exploit the recurrence of spatial and velocity signatures in each temporal cubes, in the presence of a planet. The architecture of C-LANDO, illustrated in Fig. 2, consists of the following layers:

1. a convolutional LSTM layer consisting of a 3×3 kernel with 40 filters;
2. this is followed by a 3D max pooling layer, which pools the convolutional kernels of the LSTM two pixels at a time;
3. this is followed by another convolutional LSTM layer consisting of a 2×2 kernel with 80 filters once again followed by a max pooling layer as the previous layer, which once again pools the convolutional kernels two pixels at a time;
4. this is followed by a flatten layer which converts the 3D shape into 1D that can then be fed to a feedforward neural network layer with 128 neurons with a Rectifying Linear unit (ReLu) activation;
5. the output of the feedforward neural network is then fed into a dropout layer that randomly drops 25% of the neurons;
6. finally, the output is a single sigmoid neuron that gives the probability of our output class.

4. Training the deep-learning algorithms

At the beginning of training we initialize the parameters of the neural networks such as weights and biases to random values, which will then be tuned through the training process. An important concept that will be used repeatedly is the loss function (hereafter, referred to as “loss”), which depends on the output type, and quantifies the difference between the ground truth and prediction of a network. For example, in our case where the output is a binary classifier, the categorical cross-entropy loss is used. This loss is back-propagated through the network each time the data is processed (passed) through the network (Rojas 1996). At this time the network parameters such as the weights and the biases are modified slightly and the loss is once again computed for the next data point. In practice, a series of labelled examples (known as data samples) are input and passed through each layer of the neural network and the error between the output of our architecture and the label is computed for every example. Typically, this process is repeated once through the whole dataset and such a training step is called an epoch. In order to complete the training, several such training iterations or epochs have to be completed. Each time the error is back-propagated through the network, the weights and biases are adjusted so that the error is minimized.

After calculating this loss over multiple epochs, the network parameters undergo significant adjustments to ensure that the subsequent processing by the network does not result in further loss reduction. At this point, the network is referred to as a trained model. Training the model necessitates the generation of diverse dataset, a crucial step to prevent the network from memorizing recurring noise features and to ensure the dataset encompasses a sufficient array of data features for the network to learn from. The consideration of these challenges is pivotal in the training process, as the potential problems in the data, such as overfitting (Dietterich 1995) and gradient stall (Patel 2017), could arise if not effectively addressed. Hence, in the following section, we outline the process of generating training data and provide a detailed description of the training process for both C3PO and C-LANDO.

4.1. Generating the training dataset

To train the C3PO and C-LANDO networks presented in Sect. 3.2, we insert simulated young gas giants at different

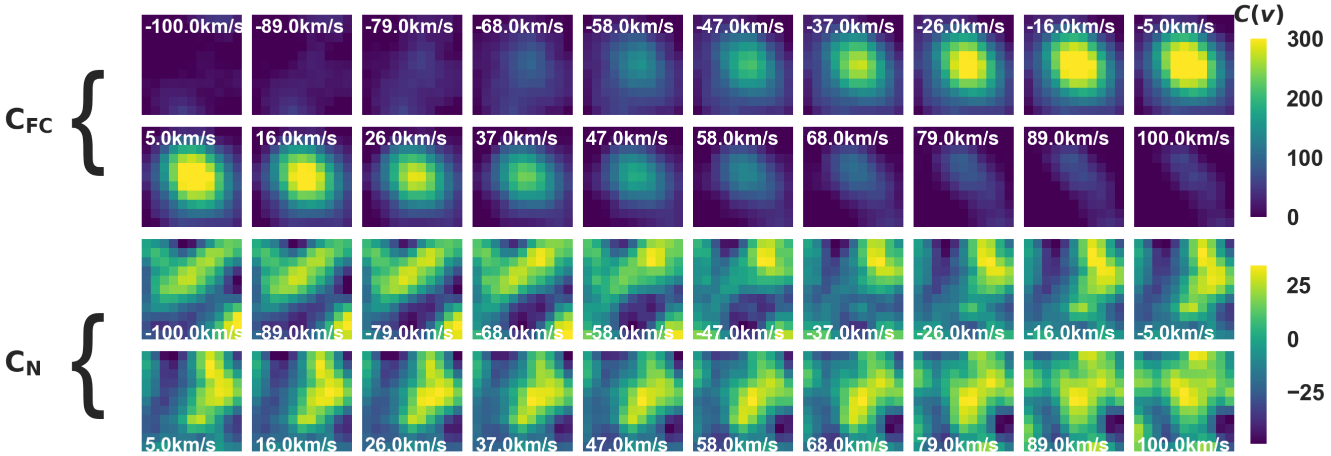

Fig. 3. Evolution of spatial crops with respect to radial velocity. The rows indicate the different velocities and each square shows the spatial noise and signal diversity of the patch. This is just a single spatio-temporal sample (i.e. one pixel from a single temporal frame). The colour-coded pixel value corresponds to the cross-correlation signal from Eq. (1).

selected pixels with a range of contrasts and at different PAs from the frame centre, as described in Sect. 2.2. In practice these contrasts range from 10^{-2} to 10^{-4} and we insert these at each pixel with a PA sampling rate of one insertion for every 30° of PA, to make sure that at the closest radial separations we still have a spaced out PA separation between insertions. From previous studies using deep learning to detect exoplanets in images (e.g. Gomez Gonzalez et al. 2018; Cantero et al. 2023), we learned that using full-frame images to train deep neural networks on spatial features is not optimal, as the features corresponding to an exoplanet are confined to a few percent of the total pixels that we train on. The effect of such imbalance can result in suboptimal training of ML algorithms. Therefore, we crop the field of view that is used as an input to the neural network to a region of two FWHM around the inserted fake companions (or around another position to learn the noise structure of non-detections). This results in data with the following dimensions: a temporal dimension of 83 temporal cubes, a velocity dimension composed of 20 velocity shifts starting from −100 and extending up to 100 km s^{-1} and two spatial dimensions of two FWHM size each (11 × 11 pixels). Thus, each pixel now corresponds to a tensor of these dimensions as an input to C3PO and C-LANDO. Both C3PO and C-LANDO produce as output a probability based on their input and that output is interpreted as the probability of a simulated young gas giant being present at central pixel of the input tensor. We use VIP to spatially crop each full frame into non-overlapping smaller 11 × 11 pixel sized patches.

Patches are produced for a pixel where the simulated young gas giant is inserted, but also for those pixels where nothing is present. The patches that contain the simulated young gas giant are part of the C_{FC} class (where FC stands for fake companion) and the rest, which are just noise samples, become part of the C_{N} class (where N stands for noise). Figure 3 shows a sample for both $\mathrm{C_{FC}}$ and $\mathrm{C_N}$ side by side. Spatially the $\mathrm{C_{FC}}$ has an intensity distribution very similar to the PSF shape originally measured in the data whereas the intensity distribution in $\mathrm{C_N}$ class appears to be random. The range of cross-correlation values in the $\mathrm{C_{FC}}$ and the $\mathrm{C_N}$ also offer a point of difference. The first two lines are those representing $\mathrm{C_{FC}}$ and the last two rows are produced in the same manner with $\mathrm{C_N}$. In any data cube if an injection is made at a specific PA and radial separation and then the cube is cropped down into 2 FWHM sized sub-cubes, we will have a total of 35 non-identical $\mathrm{C_N}$ and 1 $\mathrm{C_{FC}}$ patch tensor for every cross-correlation tensor (produced from the insertion at the same PA and radial separation) that have the spatial, velocity and temporal features. Diversity in the $\mathrm{C_{FC}}$ class is achieved by inserting the simulated young gas giant at different separations from the frame centre and at different contrasts to the stellar flux and then producing cross-correlation tensors. The temporal dimension (not shown here) is such that each pixel in the $\mathrm{C_{FC}}$ class that contains the exoplanet shows consistently the same pixel value and a set of pixels tend to maintain a PSF-like shape for a fixed velocity. The features that we intend C3PO to learn are this spatial variation and the velocity variation which will be learnt as a ‘colour’ by the CNN. C-LANDO will have the same features in addition to the repeating format temporally of $\mathrm{C_{FC}}$ and more random variation temporally.

In order to produce diverse noise samples, we resort to data augmentation similar to what was done by Gomez Gonzalez et al. (2018). Random patch sequences are subjected to one of the following augmentation techniques during insertion and before producing the C_N patch tensors themselves: random rotations between 0° and 360°, random subpixel shifts of up to 0.5 pixels in either x or y direction and production of additional samples by adding two non-sequential and randomly chosen samples. Spatially, a C_{FC} sample has an intensity distribution very similar to the PSF shape originally measured in the data whereas the intensity distribution in C_N class appears to be random. The range of cross-correlation values in the C_{FC} and the C_N also offer a point of difference. This is the type of differentiating feature that allows a deep-learning algorithm to make fine distinctions. Such a differentiating feature was observed in the PCA dimension by Gomez Gonzalez et al. (2018). The temporal dimension (not shown here) is such that each pixel in the C_{FC} class that contains the exoplanet shows consistently the same pixel value and a set of pixels tend to maintain a PSF-like shape for a fixed velocity.

4.2. Training the deep-learning algorithms

As stated earlier, ML algorithms always run the risk of memorizing the data, but there are several ways to overcome this. One of the several strategies is to divide the data into three independent datasets called training, validation, and testing. Overfitting is characterized by exceptionally good performance in the training dataset but conversely produces poor performance in the validation and test dataset. The training dataset is used to primarily train the algorithm to make predictions, and the validation is used to verify the training accuracy and make small modifications to hyperparameters such as learning rate, number of neurons, and activation functions. The test data is reserved to test and benchmark the algorithm and is typically used when the training and validation accuracies are very close to each other. The three datasets are typically separated before beginning training so that the algorithm never sees the other two parts when training. The testing samples are first removed from data by choosing 25% of the PA values and removing all C_{FC} and C_{N} samples corresponding to those. For the purposes of this paper, this corresponds to four PA values. Then we split the remaining data into 90% training and 10% validation. A key metric we use in the validation data is the number of false positives the algorithm generates. The validation comprises of patch tensors corresponding to just one PA value with all of the different radial separations so that we probe the different noise levels. As with the test, the C_{N} and C_{FC} are removed before training. We describe the computation of false positives in Sect. 5. We use the number of false positives in the validation data to tune the mix of contrasts and the parameters of the ML algorithms, such as the number of layers, neurons.

We started with a small network and grew it layer by layer as the validation data produced more true positives (i.e. started detecting the simulated young gas giant over multiple separations and for higher contrasts) and fewer false positives. It became quickly clear that hyperparameters of the network (such as the number of neurons, number of layers, and activation functions) themselves have little impact on the number of false positives produced in the validation data. Initially, we started with the full range of contrasts starting from 5×10^{-2} to 10^{-4} with no subpixel shifts. We noticed that, as we increase the number of high-contrast examples in the data, we encounter higher false positives generated. Therefore, we limit the contrast to $<10^{-3}$ (i.e. to simultated new gas giants that are brighter than 10^{-3}) with the full range of separations starting from ~1 FWHM up to ~5 FWHM. We use the `binary_cross_entropy` loss function (Li & Lee 1993) to compare the prediction of the deep-learning algorithms with the truth value. We use the Adam optimizer (Kingma & Ba 2015) to minimize the loss and overcome local minimas.

5. Comparing the performance of the algorithms

We benchmark all our algorithms based on their ability to accurately detect simulated young gas giants at contrasts dimmer than 10^{-3} with the fewest number of mistaken detections. At this point, it is important to make the distinction between the contrasts supplied when training the ML algorithms, which is limited to simulated young gas giants brighter than 10^{-3} to have less overfitting, while during testing we used the full range of contrasts (i.e. 10^{-2}–10^{-4}). In this section, we describe the production of detection map and binary maps with our test data described in the previous section. The binary maps are then used to compute true and false positives, and we finally describe the metric used for the comparison of our algorithms, based on modified receiver operating characteristic (mROC) curves.

5.1. Producing detection and binary maps on test data

To produce a test dataset, we choose four PAs corresponding to 60°, 120°, 180°, and 360°. The choice of these angles is somewhat arbitrary, but it is also done so that all of the tests can be performed with the same angles and there are lesser systematic effects when testing the three algorithms together. We then go through the procedure of inserting simulated young gas giants as described in Sect. 2.2, where we insert young gas giants at different radial separations over a range of contrasts between 10^{-2} and 10^{-4}. We then produce cross-correlation cubes as described in Sect. 2.3.2, and save these cross-correlation tensors without any further processing. As explained in the previous section, once the algorithms have trained, we pass the test dataset through the neural networks to produce detection maps. A detection map is a map of the relative weighting of pixels, where noisy pixels carry lower weight and those pixels with an exoplanet carry higher weight. These maps can be composed using S/Ns, STCMs, or probabilities, and in each case we define thresholds to define which weights constitute a detection (for example $S/N \geq 5$). An example of such detection maps for non-ML algorithms is shown in Fig. 1. Detection maps intrinsically hold the biases of the algorithm that produces them. One of the ways of quantifying these biases is the use of binary maps. Binary maps are produced when thresholds are applied to a detection map and each pixel above the threshold is assigned a binary value of 1, while the rest of the pixels are set to 0.

We produce detection maps from each of the algorithms as shown in the first column of Fig. 4. Each row in Fig. 4 starts with the detection map followed by a binary map produced with the threshold indicated in the title of each of the plots in columns two through four. These detection and binary maps are produced on the test data. The STCM algorithm (first row in Fig. 4) produces a detection map as its natural output. In order to compare the STCM and ML algorithms, we present the ML algorithms with overlapping patches where the central pixel of each patch is the pixel of interest, whose probability of containing the simulated young gas giant will be predicted by the ML algorithm. This method is thus also adapted to searching for exoplanets in the dataset as we consider each pixel as the centre of an overlapping patch. These probabilities are used to build a detection map. This constitutes row two and three of the first column of Fig. 4. Columns two to four in Fig. 4 consist of binary maps produced by thresholding the detection maps. These binary maps are then used to compute the total number of false positives and true positives for each insertion or each frame (since we have exactly one simulated young gas giant inserted in each frame).

5.2. Computing true and false positives

In principle, the threshold is the cutoff that allows us to identify the pixels where the exoplanet is present. Utilizing thresholding on a detection map facilitates the exploration of the tradeoff between false positives and the detection of faint companions. Lower thresholds, for example, increase the likelihood of identifying faint simulated young gas giants but concurrently result in a higher incidence of false positives. Counting the number of false positives and true positives in every binary map and comparing the two values as a function of the threshold allows us to produce classical receiver operating characteristic (ROC) curves. ROC curves are particularly useful when posing a problem as a two-class classification problem to define the ability of a classifier to make the trade-off between predicting a class and admitting false positives. They allow to benchmark both ML and non-ML pipelines and make a relative comparison.

In this paper, we define all the connected pixels in a binary map within ~1 FWHM of the original insertion of the simulated young gas giant as a true positive (TP) and all other non-zero pixels as a false positive (FP). We perform this analysis over multiple insertions and thereby derive the number of TPs per

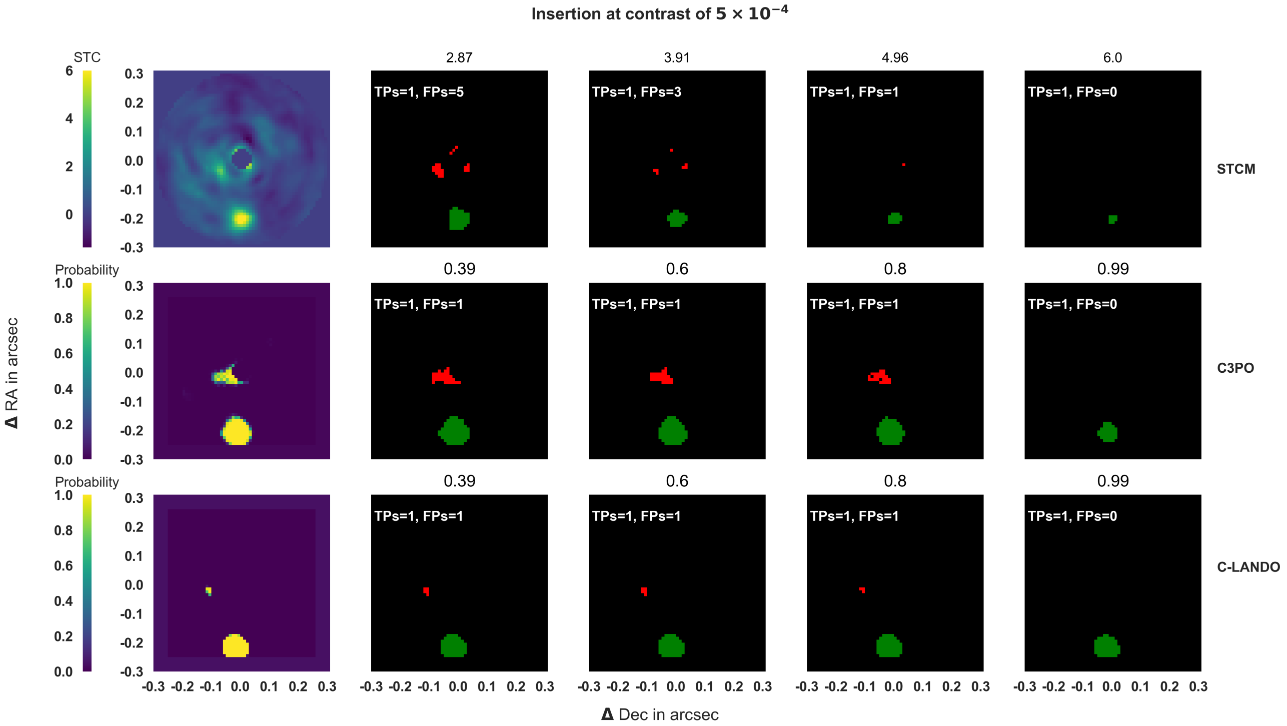

Fig. 4. Illustration of the TP and FP counting process. Column 1 shows the detection map used to produce four different binary maps, thresholded at different intensity levels (Cols. 2–5), for the three algorithms (respectively STCM, C3PO, and C-LANDO from top to bottom). A fake companion was inserted at position ΔDec = 0.0 and ΔRA = −0.2. TPs and FPs are respectively shown in green and in red for the thresholds indicated in the titles of each image. Each red blob or point is counted as a single FP or TP, with blobs representing many connected pixels at the same binary intensity level.

insertion, also known as a true positive rate (TPR), and the mean number of FPs per detection map, or a mean full frame FP as described in Gomez Gonzalez et al. (2018). Because we consider the whole field of view to count a mean number of FPs, instead of testing each single resolution element at a time to derive a false positive rate (FPR), we use adapted ROC curves referred to as mROC curves. In comparison to ROC curves, these mROC curves are arguably more relevant to the problem at hand, where the goal is to work at very low FPR (e.g. 3×10^{-7} for an equivalent Gaussian 5σ detection). As we produce these mROC curves curves for each detection map, we mask a circular region of 3.5 pixels from each frame centre.

We use procedures in VIP (Gomez Gonzalez et al. 2017; Christiaens et al. 2023) to compute TPs and FPs from a detection map. First, each detection map is subjected to the map-based ROC analysis tool available in VIP. In this tool we set the values of the number of thresholds as 10, varying from ≈1 to ≈6 for the STCM maps and from 0.1 to 0.9999 for the outputs of C3PO and C-LANDO. The FWHM is set to 4.8 pixels, which is the mean FWHM of the SINFONI instruments in all wavelength bands. In order to count the number of TPs and FPs, we use the function `compute_binary_map` described in the VIP documentation[3], with the following parameters: the `overlap_threshold` variable is set to 0.7, the `max_blob_fact` variable is set to 3 and `npix` variable set to 2 because of computational errors resulting in slightly different values for higher thresholds even though it is part of the same blob. We produce such maps for many separations and a range of contrasts. For every contrast we insert exactly one simulated young gas giant at a specific radial separation and position angle. Therefore, every detection map will contain at most one TP, and potentially many FPs for the lowest threshold. As we progressively increase the threshold the FPs will reduce until at the maximum threshold we have no remaining FP. True positives in Fig. 4 are shown in green and false positives in red. In order to produce a smooth mROC curve, we repeat this experiment T_i number of times, where T_i is the product of the total number of radial separations (in this case we pick five separations corresponding to ≈1 FWHM each), the total number of PA that are part of the test set (four angles), and the number of contrast levels at which the injection is performed. For each insertion, we choose between two contrast values respectively just above and just below the chosen contrast value (e.g. 5×10^{-4} and 3×10^{-4} for a chosen contrast of 4×10^{-4}), and inject one companion of each contrast at each position, leading to a total of 40 insertions to produce one mROC curve. This results in a total of 40 detection maps like those in Fig. 4, which are then turned into binary maps with various thresholds. Each point in Fig. 5 corresponds to 40 such binary maps for each algorithm, whose TPR and mean full frame FPs are computed as follows:

$$\mathrm{TPR} = \frac{\sum_{T_\mathrm{i}} \mathrm{TPs}}{T_\mathrm{i}}, \quad \text{mean FP} = \frac{\sum_{T_\mathrm{i}} \mathrm{FPs}}{T_\mathrm{i}}. \tag{2}$$

It should be noted that the mean number of FPs is always an integer number as the number of FPs remains the same for binary maps of a fixed threshold. In the unlikely event that a fake companion was inserted at the exact location where a FP would appear for the empty dataset, we still account for it as we compute the mean FP. The mean number of FPs is only

[3] https://vip.readthedocs.io/en/latest/_modules/vip_hci/metrics/roc.html?highlight=compute_binary_map

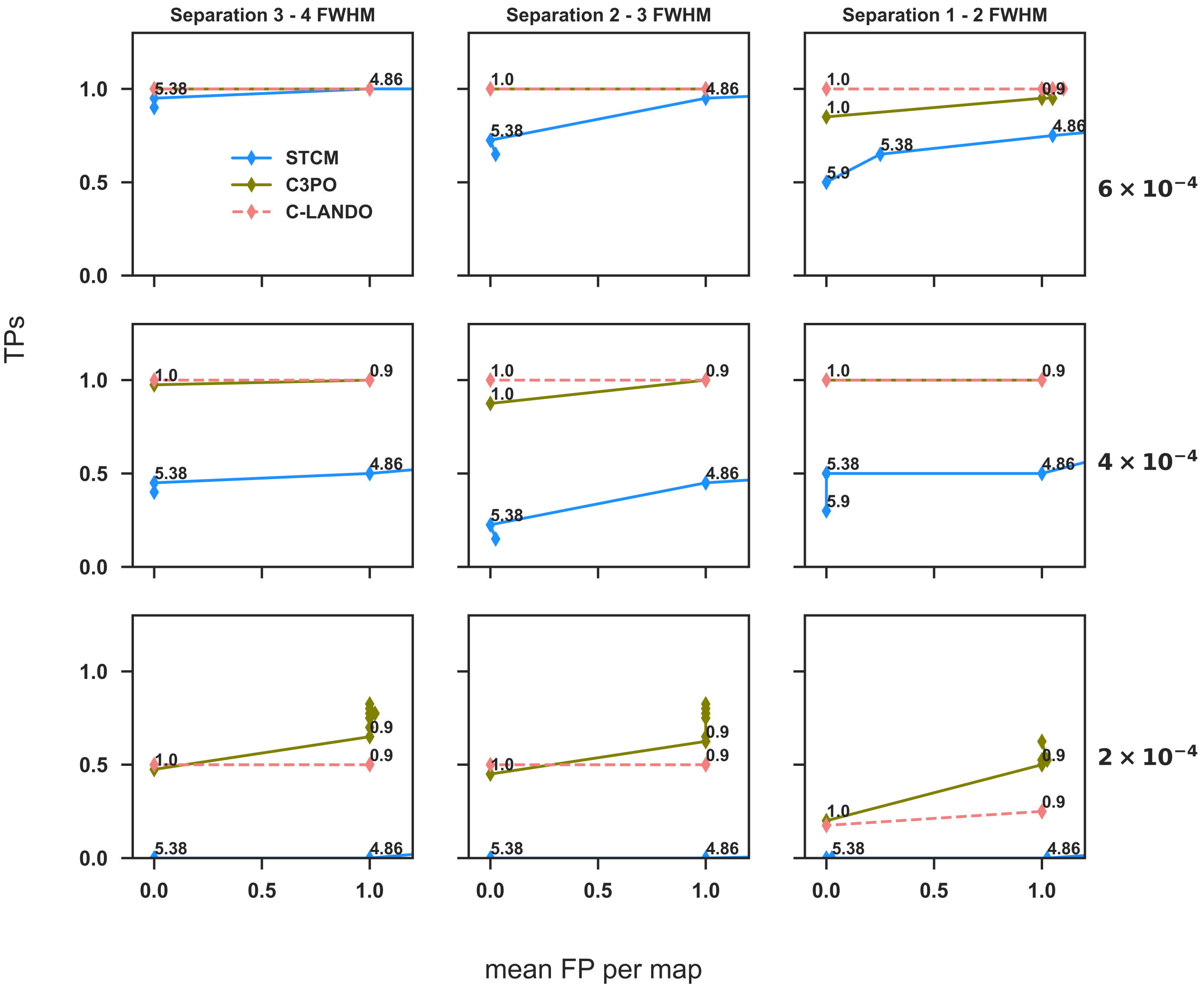

Fig. 5. mROC curves produced by inserting a total of 40 exoplanets for a set of contrasts (6×10^{-4}, 4×10^{-4} and 2×10^{-4} from top to bottom) at annuli of different separations in the ranges 3−4, 2−3, and 1−2 FWHM (left, middle, and right columns, respectively). The *x*-axis is the mean FP per map and the *y*-axis corresponds to the TPR computed using Eq. (2). The numbers next to each data point correspond to the threshold that was applied to compute the number of TPs and FPs for each binary map (as described in Fig. 4).

bounded by the number of pixels in our field-of-view, but we limit the range of the *x*-axis in our mROC curves to 1, as this is the largest number of FPs produced by the ML algorithm for the considered thresholds, and because this is the regime astronomers are usually interested in. The numbers around each point in Fig. 5 represent the threshold for each binary map produced by each algorithm. We consider the highest threshold as that where FP = 0. For the ML algorithms this is satisfied at the highest threshold value of 0.999, and for the STCM algorithm at a STCM ∼ 6.0.

5.3. Comparing the mROC curves

To compare the different ML and non-ML algorithms, we computed their mROC curves for different separations, as each separation corresponds to a different noise regime. This corresponds to the columns in Fig. 5, with the rows corresponding to mean contrasts starting from 6×10^{-4} down to 2×10^{-4}. We have chosen this range of contrasts specifically as it allows us to observe the change in detection sensitivity of the algorithms. The panels start with a similar sensitivity for all algorithms (all the algorithms detect the simulated young gas giant in every insertion with a TPR ≈ 1.0 for the different values of thresholds). As we step through different noise levels (left to right) and brightness levels (top to bottom), the STCM algorithm evolves towards a complete insensitivity. The shape of an mROC curve defines the response of the algorithm to increasing the thresholds and thereby its ability to continue to have TPs while the FPs are removed. An ideal detection algorithm is where the TPR = 1 even when the threshold changes (i.e. the mROC curve is flat, and lies at the very top of the plot). This case is illustrated in Fig. 5 (cf. first row first column) for all three algorithms and remains the case for the ML algorithms in most of the first two rows, while STCM begins to show a drastic drop in TPR as the simulated young gas giant gets fainter in the second row. Conversely, when we look at the bottom right set of plots, which correspond to the higher contrast noisy regime, a decrease in threshold will produce more FPs but will also produce higher TPR.

The performance of the STCM is the baseline that establishes whether a contrast is detectable by non-ML methods. This corresponds to contrasts in the top left of the plot, particularly the top left extreme where all three algorithms perform similarly in terms of the TPs i.e., TPR = 1 for FP = 0. Thus, for this paper,

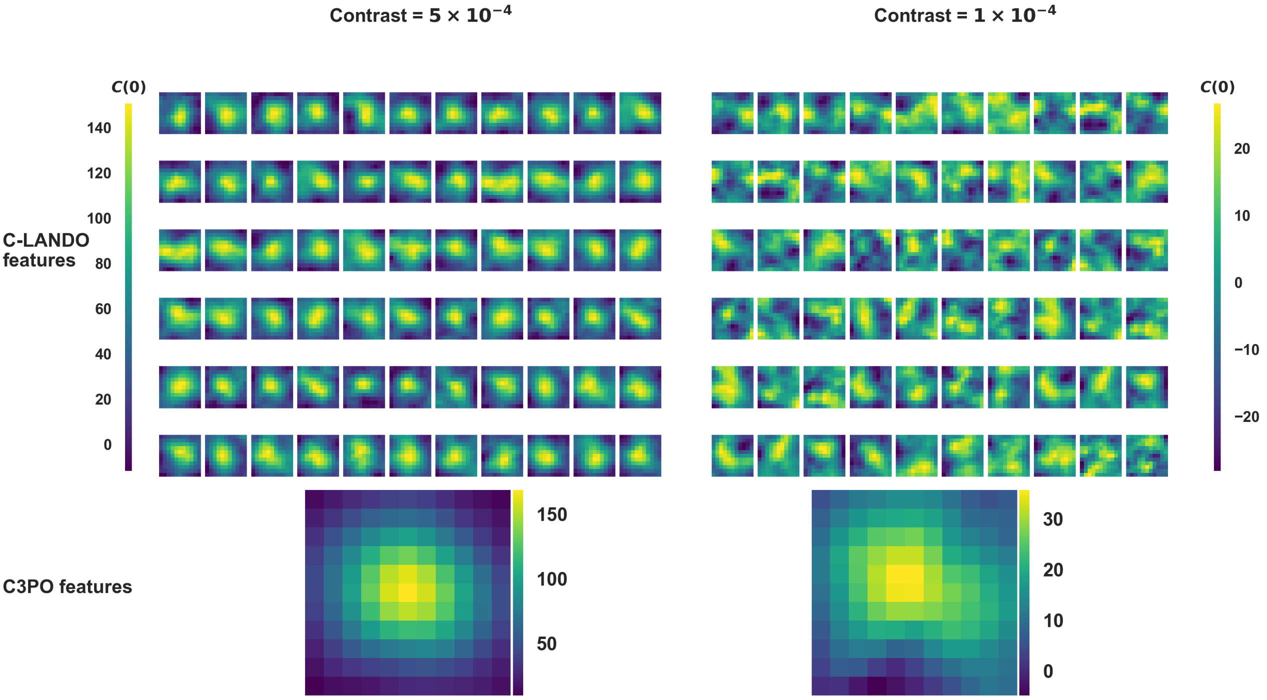

Fig. 6. Data presented to the ML algorithms for training, for two different contrast levels. The upper rows of miniatures depict the spatial distribution of the cross-correlation signature along the temporal dimension, used for C-LANDO training, while the lower images depict the temporal mean of the same cube, used for C3PO training.

we study those contrasts at which STCM produces TPR < 1 and compare how ML algorithms perform for these contrasts and separations. As we go from the top left towards the bottom right we see TPRs going down across all algorithms, until we reach a TPR = 0 at the bottom row for STCM. In general, ML algorithms continue to detect the majority of simulated young gas giants (TPR ≥ 0.5) other than for the highest contrast companion closest to the frame centre. For the lowest contrast, we see that STCM produces a TPR = 0 for almost all separations and values of the thresholds. Conversely, the ML algorithms continue to be able to produce a TPR > 0 for all the considered contrasts. The additional advantage offered by ML algorithms is that the FPs do not seem to increase indefinitely, whereas with STCM it is obvious that the FPs will continue to increase at lower thresholds. An interesting observation is that the brightness of the young gas giant seems to be the single most important astronomical aspect which impacts detectability with ML algorithms. We see that the TPR does not drop when reducing radial separation (i.e. across columns in Fig. 5), but rather drops with increasing contrast. Conversely, the brightness of the companion and the background stellar contamination seems to both be impacting factors for the STCM, as we observe a drop in TPR both with increasing contrast and reducing radial separations (i.e. both rows and columns in Fig. 5).

6. Discussion

The results in the previous section indicate that ML algorithms provide an advantage in noisier regimes, closer to the frame centre and particularly with higher contrast exoplanets. But the reality is that the algorithm uses data features (temporal, spatial and velocity) diversity naturally present in the data efficiently and consistently in the test data. One of the questions that is raised is which features or dimensions of the data are producing this high detection sensitivity and robustness that make them more reliable? We also have several dimensions such as the spatial, the cross-correlation velocity dimension, and the temporal dimension. In this section, we analyse this question in the light of our data and results. We start by discussing the reasons for the relatively higher TPR produced by C3PO when compared to C-LANDO. This will serve to illuminate as well, why the temporal dimension does not lend much more detection sensitivity than just using the image dimension alone. We then make a test of the same data both with and without the velocity to provide evidence of the importance of the velocity dimension to our results.

6.1. Impact of the temporal dimension

The previous section showed us that C3PO is more sensitive both at higher contrast and noisier regimes than C-LANDO. We posit that one of the reasons for this higher detection sensitivity is the consistency in the presence of signal in the input cubes to C3PO. While both C3PO and C-LANDO have been trained with the same data and are considered equally well trained, the difference in performance is a measure of the features necessary to make a prediction being present in the data. Therefore, it appears that while for lower contrasts (i.e., brighter exoplanets) C-LANDO sees the signature of the exoplanet being consistently present, in the temporal dimension this behaviour changes with higher contrast insertions. We use Fig. 6 to explain this behaviour. This figure shows the input to both algorithms, composed of cross-correlation tensors that are cropped to the 2-FWHM patch tensors. In the first six rows of the figure, we depict patches of some of the 83 temporal cubes (i.e. the temporal features that are used to train C-LANDO), while the final row shows the mean of these temporal cubes (i.e. the features used to train C3PO). The two large columns represent two different contrasts at which the simulated young gas giant is inserted. On the left is a contrast corresponding to the first column second row of Fig. 5, where both algorithms perform very similarly. Both in amplitude and spatial features the first large column shows quite similar features, where C-LANDO just seems to have an additional 83 copies of the features that C3PO receives. The data features the algorithm looks for is the spatial distribution of the cross-correlation coefficients, as much as signal consistency in the temporal dimension. The second large column corresponds to a young gas giant inserted for the same separation at a higher contrast. Thus, the only difference between the columns is the brightness of the simulated young gas giant. In this column the first difference that we notice is the difference in amplitude between the first six rows and the last row. The first six rows have an amplitude range closer to the class C_{N} in Fig. 3, whereas the final row seems to have linearly scaled amplitude to its lower contrast (brighter) counterpart (first column).

The spatial signature of the cross-correlation plays another, in our view, crucial role in the higher TPR produced by C3PO for higher contrasts. When we compare the inputs to both the algorithms between the different contrasts, the spatial distribution of the cross-correlation coefficients is better conserved for the input to C3PO than for C-LANDO, although the mean cross-correlation strength is reduced for a higher contrast. This seems to be another key differentiating factor between the two algorithms where it appears that computing the mean to produce the last row focuses the features better to enable an ML algorithm to be more sensitive to a high-contrast exoplanet.

6.2. Importance of the velocity dimension

The importance of how spatial features are driving the higher performance of C3PO was illustrated in the previous section. The STCM algorithm sees the same features as C3PO, but with $v = 0$, and thus only sees the $C(0)$ coefficient. Since velocity seems to be the key feature difference between C3PO and STCM, it is relevant to discuss the importance of the velocity feature in the data. In order to appreciate the importance of this dimension, we train the ML algorithms both with the full range of the velocity shift (i.e. between -100 and $100\ \mathrm{km\,s^{-1}}$) to produce ROC curves in Fig. 5, but also train them by choosing only the velocity where the cross-correlation value is the maximum (i.e. $v = 5\ \mathrm{km\,s^{-1}}$ in Fig. 3) We note that because of the medium resolution of SINFONI, there is no significant difference between the spatial distribution and mean value of cross-correlation between $v = -5$, 0, and 5 $\mathrm{km\,s^{-1}}$. Then, we re-create C3PO and C-LANDO such that the value for the number channels is 1 instead of 20 as in Fig. 2. All other dilations/changes are appropriately altered automatically as a consequence of this but the hyperparameters (i.e. number of neurons, layers, the mix of contrasts used for training, optimization algorithm, and error function) remain the same. We perform training exactly in the manner described in Sect. 3.2, compute the TPs and FPs in frame exactly as in Sect. 5, and then produce sample ROC curves in Fig. 7, where the two plots are obtained for insertions of simulated young gas giants at different contrasts but at a fixed separation from the frame centre. The dashed lines in Fig. 7 are those produced with no explicit velocity evolution in the data. The solid lines are the same as those produced in Fig. 5.

The first noteworthy point is that we have extended the FP axis up to 3 FPs, because both ML algorithms without access to the velocity dimension produce more FPs. The second point is that the evolution (i.e. shape of the mROC curves) of the TPRs produced by the ML algorithms remains similar both with and without the velocity dimension between two different contrasts (i.e. the TPR falls by a fraction of ~50% between contrasts). However, within the same contrast regime (in this case each sub-plot), the TPR produced by C-LANDO with velocity is ~20% higher than without the velocity evolution (for a fixed FP), whereas the evolution of the TPR produced by C3PO is less dramatic. The velocity dimension produces a distinct systematic variation of spatial features (i.e. both cross-correlation values and the spatial distribution of the cross-correlation). This is becomes evident in Fig. 3, when comparing the evolution of the spatial features with velocity between C_{FC} and C_{N}. This produces an additional dimension that can be relied on to make this distinction between different pixels, making the detection less confused ergo more sensitive. This should particularly be useful when one of the dimensions used are not able to make these distinctions for a specific insertion. Therefore, while introducing the velocity dimension, C-LANDO produces an improvement in both TPR and FPs, the improvement in C3PO is mostly confined to having fewer FPs.

A final noteworthy point is that given the data dimensionality used by both the ML algorithms and STCM, it is truly fair to compare the solid blue line in Fig. 7 and the dashed lines produced by the ML algorithms, and we notice that the TPRs produced by the ML algorithms are higher, however the FPs produced by C-LANDO are similar, when reducing the threshold, to those produced by STCM (FP = 3). When we consider the performance of the ML and non-ML algorithms, it is clear that even with the same data dimensions C3PO outperforms STCM in detection sensitivity.

7. Conclusion

In this paper we explored the detectability of young gas giants in high-contrast imaging data, while having access to medium-resolution spectra for each pixel of the image, with several such cubes produced temporally. The fundamental question we addressed is whether the use of such data with ML algorithms produces any improvement in detection sensitivity to young gas giants. In order to efficiently use the spectral dimension, we used cross-correlations to produce cross-correlation tensors, which were then subjected to further statistical analysis to determine if they contain an injected exoplanet. We used ML and non-ML algorithms that are capable of detecting the cross-correlation signature of an exoplanet. In order to answer this question, we used ROC curves as a method to identify the trade-off between incorrectly labelling speckles as exoplanets in test data and detecting high-contrast exoplanets. To build ROC curves, we inserted simulated young gas giants in a dataset devoid of any known companions, at different radial separations from the host star to simulate different noise regimes and at different contrast to simulate exoplanets of different brightness. To detect these simulated young gas giants we devised a non-ML algorithm (STCM), a CNN-based algorithm (C3PO), and a CNN-LSTM-based algorithm (C-LANDO). We trained the ML algorithms on part of the data and reserved the rest to test and benchmark our algorithms. We trained our algorithms so that the validation set had the fewest FPs, and tuned our algorithms accordingly.

The most notable results of our study are the following:

- The ability to detect TPs reduces for the STCM algorithm for companions closer to the frame centre, while it remains relatively unchanged for ML algorithms.
- The ability of ML algorithms to harness additional dimensionalities such as velocity seems to be at the root of their increased sensitivity.

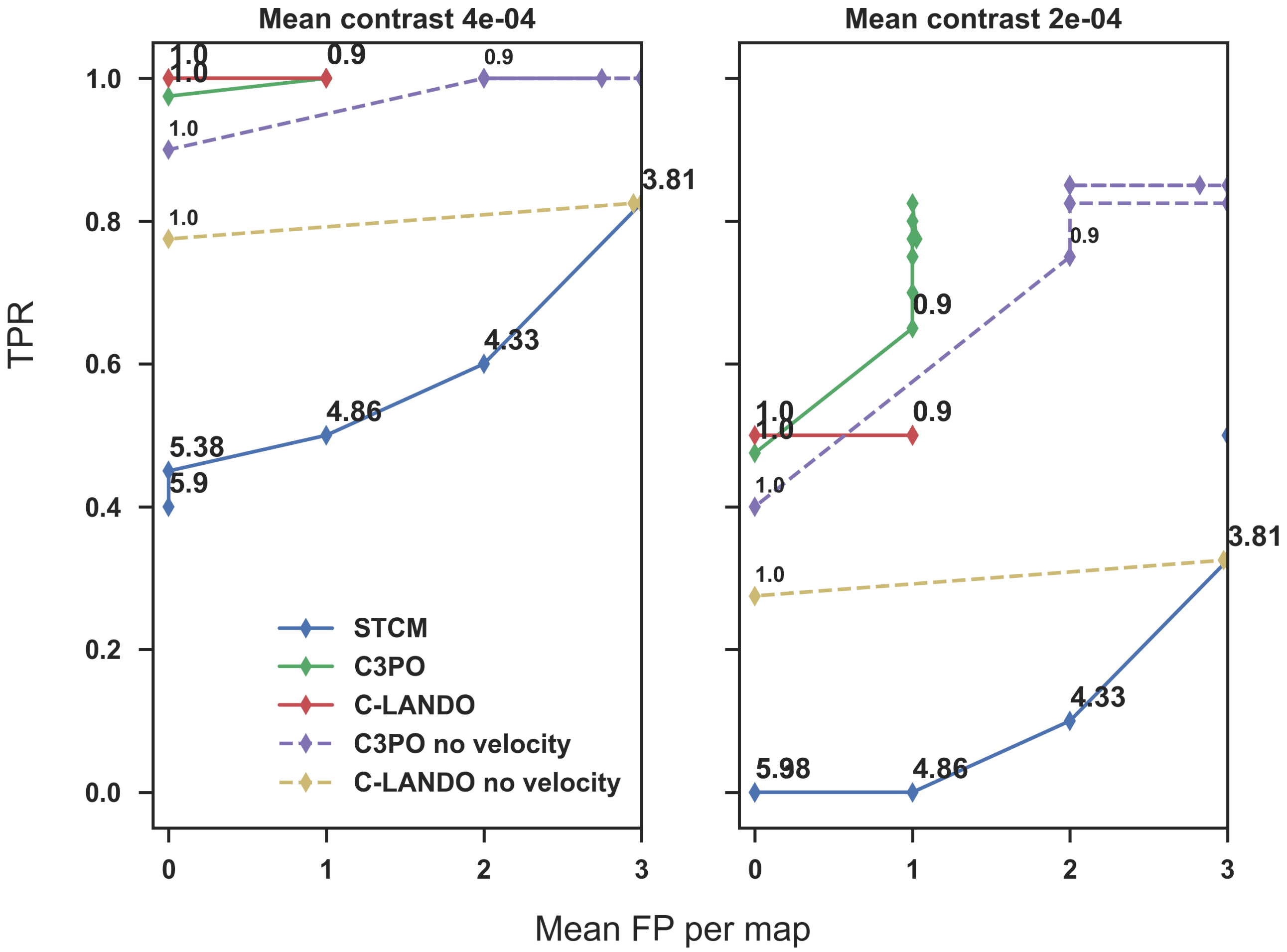

Fig. 7. Comparison of mROC curves produced with and without using the velocity dimension, using otherwise the same dataset, at two contrast levels (4×10^{-4} and 2×10^{-4}) and for a separation of 3–4 FWHM. In the cases where the velocity dimension is not used, illustrated with the purple and yellow dashed lines for C3PO and C-LANDO, respectively, the velocity dimension is replaced by the central velocity bin. This makes the comparison to STCM a little more balanced as STCM does not have the capability to use the velocity dimension. A final noteworthy point is that given the data dimensionality used by both the ML algorithms and STCM, it is truly fair to compare the solid blue line in Fig. 7 and the dashed lines produced by the ML algorithms, and we note the TPRs produced by the ML algorithms are higher; however, the FPs produced by C-LANDO are similar, when reducing the threshold, to those produced by STCM (FP = 3). We note that the solid curves remain the same as in Fig. 5.

– The use of the temporal dimension does not seem to provide an added value in terms of detectability.

There are a few limitations to this work that merit a mention. The first is related to the use of a unique type of exoplanet, while the use of cross-correlation also allows the use of multiple templates for multiple types of exoplanets. A companion paper (Garvin et al. 2024) addresses this topic, where they show that it is possible to account for template variability by incorporating several spectral templates as convolution filter depth for CNNs. This accounts for uncertainties in the planet's atmospheric characteristics when performing detection in spectral data. To increase this flexibility, they also use individual molecular line lists instead of full atmospheric templates. Another limitation of this work is the limited data that was used for this study, which means that the selection of test data reduced the amount of data available to train the algorithms. Despite this significant limitation, the ML algorithms still continued to be more sensitive to high-contrast exoplanets, even at smaller radial separations without any evidence of overfitting. If our algorithms overfit the training data, we typically expect to see a large number of FPs in the test data, but very few in the training data. The fact that this has performed well on this dataset, which is very noisy and lacks a coronagraph to enable high-contrast imaging, gives us confidence in a future iteration to try this algorithm on a less noisy dataset. Finally, ML algorithms have evolved rapidly in the last few years, with the advent of Vision Transformers (Dosovitskiy et al. 2021) and recipes to train them on smaller datasets (e.g. Gani et al. 2022). Those algorithms have recently been shown to be very effective on the same data dimensions, and would be a direction to explore for future research. Although we used relatively standard network architectures, this paper sets the methodology to leverage a computer vision algorithm effectively with spectral and temporal dimensions.

Acknowledgements. R.N.R., O.A., and V.C. are funded by the Fund for Scientific Research (F.R.S.-FNRS) of Belgium. R.N.R was funded for this project through the FRIA grant N° 40004011. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 819155). R.N.R also wishes to thank the members of PSILab headed by O.A and Dr Sascha Quanz for the kind inputs and discussions. E.O.G gratefully acknowledges the financial support from the Swiss National Science Foundation (SNSF) under project grant number 200020_200399.

References

Abuter, R., Schreiber, J., Eisenhauer, F., et al. 2006, New A Rev., 50, 398
Allard, F., Hauschildt, P. H., Alexander, D. R., & Starrfield, S. 1997, ARA&A, 35, 137
Allard, F., Homeier, D., & Freytag, B. 2011, ASP Conf. Ser., 448, 91
Baraffe, I., Chabrier, G., Barman, T. S., Allard, F., & Hauschildt, P. H. 2003, A&A, 402, 701
Beuzit, J. L., Vigan, A., Mouillet, D., et al. 2019, A&A, 631, A155
Birkby, J., De Kok, R., Brogi, M., et al. 2013, MNRAS, 436, L35
Böker, T., Arribas, S., Lützgendorf, N., et al. 2022, A&A, 661, A82
Bonnefoy, M., Marleau, G. D., Galicher, R., et al. 2014, A&A, 567, L9
Bonnefoy, M., Zurlo, A., Baudino, J. L., et al. 2016, A&A, 587, A58
Bonnet, H., Conzelmann, R., Delabre, B., et al. 2004, SPIE Conf. Ser., 5490, 130
Brogi, M., & Line, M. R. 2019, AJ, 157, 114
Bryan, M. L., Knutson, H. A., Howard, A. W., et al. 2016, ApJ, 821, 89
Cantero, C., Absil, O., Dahlqvist, C. H., & Van Droogenbroeck, M. 2023, A&A, 680, A86
Cepa, J., Aguiar, M., Escalera, V. G., et al. 2000, SPIE Conf. Ser., 4008, 623
Chilcote, J., Pueyo, L., De Rosa, R. J., et al. 2017, AJ, 153, 182
Christiaens, V., Casassus, S., Absil, O., et al. 2018, A&A, 617, A37
Christiaens, V., Cantalloube, F., Casassus, S., et al. 2019, ApJ, 877, L33
Christiaens, V., Ubeira-Gabellini, M. G., Canovas, H., et al. 2021, VizieR Online Data Catalog: J/MNRAS/502/6117
Christiaens, V., Gonzalez, C., Farkas, R., et al. 2023, J. Open Source Softw., 8, 4774
Cugno, G., Patapis, P., Stolker, T., et al. 2021, A&A, 653, A12
Currie, T., Brandt, G. M., Brandt, T. D., et al. 2023, Science, 380, 198
Davies, R., Absil, O., Agapito, G., et al. 2023, A&A, 674, A207
Delorme, P., Schmidt, T., Bonnefoy, M., et al. 2017, A&A, 608, A79
Dietterich, T. 1995, ACM Comput. Surveys, 27, 326
Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. 2021, in 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, OpenReview.net
Eisenhauer, F., Abuter, R., Bickert, K., et al. 2003, SPIE Conf. Ser., 4841, 1548
Fisher, C., Hoeijmakers, H. J., Kitzmann, D., et al. 2020, AJ, 159, 192
Flasseur, O., Bodrito, T., Mairal, J., et al. 2023, MNRAS, 527, 1534
Fluke, C. J., & Jacobs, C. 2020, WIREs Data Mining and Knowledge Discovery, 10, e1349
Galicher, R., & Mazoyer, J. 2023, Comptes Rendus. Phys., 24, 69
Gani, H., Naseer, M., & Yaqub, M. 2022, in 33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022 (BMVA Press)
Garvin, E. O., Bonse, M. J., Hayoz, J., et al. 2024, A&A, 689, A143
Gebhard, T. D., Bonse, M. J., Quanz, S. P., & Schölkopf, B. 2022, A&A, 666, A9
Gomez Gonzalez, C. A., Wertz, O., Absil, O., et al. 2017, AJ, 154, 7
Gomez Gonzalez, C. A., Absil, O., & Van Droogenbroeck, M. 2018, A&A, 613, A71
Groff, T. D., Kasdin, N. J., Limbach, M. A., et al. 2015, SPIE Conf. Ser., 9605, 96051C
Haffert, S. Y., Bohn, A. J., de Boer, J., et al. 2019, Nat. Astron., 3, 749
Hochreiter, S., & Schmidhuber, J. 1997, Neural Comput., 9, 1735
Hoeijmakers, H. J., Schwarz, H., Snellen, I. A. G., et al. 2018, A&A, 617, A144
Keppler, M., Benisty, M., Müller, A., et al. 2018, A&A, 617, A44
Kingma, D. P., & Ba, J. 2015, in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, eds. Y. Bengio, & Y. LeCun
Konopacky, Q. M., Barman, T. S., Macintosh, B. A., & Marois, C. 2013, Science, 339, 1398
Leung, H. W., & Bovy, J. 2019, MNRAS, 483, 3255
Li, C. H., & Lee, C. K. 1993, Pattern Recog., 26, 617
Liao, J., Liu, L., Duan, H., et al. 2022, JMIR Med Inform, 10, e28880
Liu, M. C. 2004, Science, 305, 1442
Macintosh, B., Graham, J. R., Ingraham, P., et al. 2014, Proc. Natl. Acad. Sci., 111, 12661
Macintosh, B., Graham, J. R., Barman, T., et al. 2015, Science, 350, 64
Mâlin, M., Boccaletti, A., Charnay, B., Kiefer, F., & Bézard, B. 2023, A&A, 671, A109
Marois, C., Lafrenière, D., Doyon, R., Macintosh, B., & Nadeau, D. 2006, ApJ, 641, 556
Mawet, D., Milli, J., Wahhaj, Z., et al. 2014a, ApJ, 792, 97
Mollière, P., Stolker, T., Lacour, S., et al. 2020, A&A, 640, A131
Müller, A., Keppler, M., Henning, T., et al. 2018, A&A, 617, L2
Pairet, B., Cantalloube, F., Gomez Gonzalez, C. A., Absil, O., & Jacques, L. 2019, MNRAS, 487, 2262
Patapis, P., Nasedkin, E., Cugno, G., et al. 2022, A&A, 658, A72
Patel, V. 2017, arXiv e-prints [arXiv:1702.00317]
Pearson, K. A., Palafox, L., & Griffith, C. A. 2018, MNRAS, 474, 478
Rojas, R. 1996, The Backpropagation Algorithm (Berlin, Heidelberg: Springer Berlin Heidelberg), 149
Ruffio, J.-B., Macintosh, B., Konopacky, Q. M., et al. 2019, AJ, 158, 200
Ruffio, J.-B., Perrin, M. D., Hoch, K. K. W., et al. 2023, arXiv e-prints [arXiv:2310.09902]
Savitzky, A., & Golay, M. J. E. 1964, Analy. Chem., 36, 1627
Shi, X., Chen, Z., Wang, H., et al. 2015, Advances in neural information processing systems (Cambridge, Massachusetts: The MIT Press), 28
Snellen, I. A., De Kok, R. J., De Mooij, E. J., & Albrecht, S. 2010, Nature, 465, 1049
Snellen, I., de Kok, R., Birkby, J. L., et al. 2015, A&A, 576, A59
Sparks, W. B., & Ford, H. C. 2002, ApJ, 578, 543
Tao, Y., Zhang, Y., Cui, C., & Zhang, G. 2020, ASP Conf. Ser., 522, 421
Wang, J., Kolecki, J. R., Ruffio, J.-B., et al. 2022, AJ, 163, 189
Xuan, J. W., Wang, J., Ruffio, J.-B., et al. 2022, ApJ, 937, 54
Zhang, W., Itoh, K., Tanida, J., & Ichioka, Y. 1990, Appl. Opt., 29, 4790
Zhang, Y., Snellen, I. A. G., Brogi, M., & Birkby, J. L. 2022, Res. Notes Am. Astron. Soc., 6, 194