Convolutional neural networks as an alternative to Bayesian retrievals for interpreting exoplanet transmission spectra

F. Ardévol Martínez1,2,3,4, M. Min2, I. Kamp1, and P. I. Palmer3,4

1 Kapteyn Astronomical Institute, University of Groningen, Groningen, The Netherlands
e-mail: ardevol@astro.rug.nl
2 Netherlands Space Research Institute (SRON), Leiden, The Netherlands
3 Centre for Exoplanet Science, University of Edinburgh, Edinburgh, UK
4 School of GeoSciences, University of Edinburgh, Edinburgh, UK

Received XXX; accepted YYY

ABSTRACT

Context. Exoplanet observations are currently analysed with Bayesian retrieval techniques to constrain physical and chemical properties of their atmospheres. Due to the computational load of the models used to analyse said observations, a compromise is usually needed between model complexity and computing time. Analyses of observational data from future facilities, such as the James Webb Space Telescope (JWST), will require more complex models, and this will increase the computational load of retrievals, prompting the search for a faster approach for interpreting exoplanet observations.

Aims. Our goal is to compare machine learning retrievals of exoplanet transmission spectra with nested sampling (Bayesian retrieval) and to understand if machine learning can be as reliable as a Bayesian retrieval for a statistically significant sample of spectra while being orders of magnitude faster.

Methods. We generated grids of synthetic transmission spectra and their corresponding planetary and atmospheric parameters, with one using free chemistry models and the other using equilibrium chemistry models. Each grid was subsequently rebinned to simulate both HST (Hubble Space Telescope), WFC3 (Wide Field Camera 3), and JWST NIRSpec (Near-InfraRed Spectrograph) observations, yielding four datasets in total. Convolutional neural networks (CNNs) were trained with each of the datasets. We performed retrievals for a set of 1,000 simulated observations for each combination of model type and instrument with nested sampling and machine learning. We also used both methods to perform retrievals for real WFC3 transmission spectra of 48 exoplanets. Additionally, we carried out experiments to test how robust machine learning and nested sampling are against incorrect assumptions in our models.

Results. Convolutional neural networks reached a lower coefficient of determination between predicted and true values of the parameters. Neither CNNs nor nested sampling systematically reached a lower bias for all parameters. Nested sampling underestimated the uncertainty in ~8% of retrievals, whereas CNNs correctly estimated the uncertainties. When performing retrievals for real WFC3 observations, nested sampling and machine learning agreed within 2σ for ~86% of spectra. When doing retrievals with incorrect assumptions, nested sampling underestimated the uncertainty in ~12% to ~41% of cases, whereas for the CNNs this fraction always remained below ~10%.

Key words. planets and satellites: atmospheres – planets and satellites: gaseous planets – planets and satellites: composition

1. Introduction

The number of confirmed exoplanets has grown to over 4,900 in recent years1. Transiting exoplanets provide a golden opportunity for characterisation. The wavelength-dependent transit depth contains information about the chemical and physical structure of the exoplanet’s atmosphere. Yet retrieving this information is not an easy task. Traditional Bayesian retrievals involve computing tens to hundreds of thousands of forward models and comparing them to observations to derive probability distributions for different physical and chemical parameters. Nested sampling (Skilling 2006), and in particular the MultiNest (MN) implementation (Feroz et al. 2009), is currently the preferred Bayesian sampling technique. The high computational load of Bayesian retrievals often makes it necessary to simplify the forward models by parameterising different aspects of the atmosphere, such as the temperature structure, the chemistry, and/or the clouds. More complex models that can compute these self-consistently paired with data with higher S/N and larger wavelength coverage, such as what will be provided by future facilities like the JWST (James Webb Space Telescope, Gardner et al. 2006) or ARIEL (Atmospheric Remote-sensing Exoplanet Large-survey, Tinetti et al. 2018), will only exacerbate this problem.

Machine learning approaches have recently started being considered as means to reduce the computational load of atmospheric retrievals (e.g. Waldmann 2016; Zingales & Waldmann 2018; Márquez-Neila et al. 2018; Cobb et al. 2019; Nixon & Madhusudhan 2020). In principle, once a supervised learning algorithm has been trained with pairs of forward models and their corresponding parameters, it should be possible to perform retrievals in seconds. The feasibility of this approach was first demonstrated by Waldmann (2016), who used a deep belief network (DBN) to recognise molecular signatures in exoplanetary emission spectra. DBNs are generative models, which means that they learn to replicate the inputs (in this case the emission spectra) in an unsupervised manner. Subsequently, they can be trained with supervision to perform classification. Indeed,
this first method did not do a full retrieval and could only predict whether different molecules were present in the spectrum. Another generative model was used by Zingales & Waldmann (2018), who trained a deep convolutional generative adversarial network (DCGAN). In their case, the DCGAN was trained to generate a 2D array containing both the spectra and the parameters. This is accomplished by pitting two convolutional neural networks against each other, one trying to generate fake inputs and the other trying to discriminate the real inputs from the fake ones. Training concludes when the discriminator can no longer tell which is which. The DCGAN can then be used to reconstruct incomplete inputs, in this case a 2D array missing the parameters. Having to train two neural networks puts this method at a disadvantage, as more fine-tuning is necessary to optimise the architectures of both networks. Lastly, Zingales & Waldmann (2018) find that a very large training set of $\sim 5 \cdot 10^5$ forward models is necessary for the DCGAN to learn, while the methods presented below can be trained with $\sim 5 \cdot 10^3$ forward models, a factor of ten fewer.

Aside from generative models, random forests and neural networks have also been used to retrieve atmospheric parameters from transmission spectra of exoplanets. Márquez-Neila et al. (2018) trained a random forest to do atmospheric retrievals of transmission spectra, obtaining a posterior distribution for WASP-12b consistent with a Bayesian retrieval. Random forests are ensemble predictors composed of multiple regression trees, each trying to find which features and thresholds best split the data into categories which best match the labels. They are simple, easy to interpret, and fast to train (although they can require large amounts of RAM). However, due to their simplicity, they are outperformed by neural networks in complex tasks, such as the ones we will be using in this work (see Section 6). Cobb et al. (2019) trained a dense neural network able to predict both the means and covariance matrix of the parameters, showing again good agreement with nested sampling for WASP-12b. Additionally, machine learning has also been used to interpret albedo data (Johnsen et al. 2020), as well as planetary interiors (Baumeister et al. 2020) and high-resolution ground-based transmission spectra (Fisher et al. 2020). More recently, Yip et al. (2021) investigated the interpretability (that is, understanding what these algorithms are ‘looking at’ when making their predictions) of these deep learning approaches, showing that the decision process of the neural networks is largely aligned with our intuitions.

These previous studies were limited to retrieving isothermal atmospheres, free chemistry (meaning the abundances of chemical species are free parameters) with only a few molecules, and grey clouds. Retrieving the chemistry self consistently is necessary to constrain macroscopic parameters such as the C/O or the metallicity, which are crucial to understand the formation history of the planet (e.g., Madhusudhan et al. 2016; Cridland, Alex J. et al. 2020). Additionally, the previous literature only contains comparisons between machine learning and Bayesian retrievals for a handful of spectra. More extensive testing is needed to establish machine learning retrieval frameworks as valid alternatives to nested sampling.

We therefore trained convolutional neural networks (CNN) to do free and equilibrium chemistry retrievals of NIRSpec and WFC3 transmission spectra of exoplanets. We also compared CNN and Multinest retrievals for large samples of spectra.

In this paper, we first introduce the generation of the data used to train the CNNs and its pre-processing in Section 2. In Section 3, a brief explanation of how CNNs work is presented, followed by a description of our implementation and the evaluation techniques employed. Our results are presented in Section 4. In Section 5, we test the robustness of both retrieval methods against incorrect assumptions in our forward models. In Section 6, the results presented previously are discussed. Finally, in Section 7, we present our conclusions.

2. Generation of the training data

Training a supervised learning algorithm requires training examples containing features (inputs) and labels (outputs). In the case of inferring physical and chemical parameters from transmission spectra of exoplanets, the features are the transit depths at each wavelength and the labels are the planetary and atmospheric parameters. We generated two grids of transmission spectra using two different types of models with ARtful modelling of Cloudy exoplanet atmosphereS (ARCIS, Min et al. 2020), each containing 100,000 spectrum-parameters pairs.

ARtful modelling of Cloudy exoplanet atmosphereS is an atmospheric modelling code attempting to strike a balance between fully self-consistent and parametric codes. This balance allows it to predict observations from physical parameters while limiting the computational load, making it suitable for nested sampling retrievals. It is a flexible framework that allows us to compute models of different complexities. In this work, we used two different types of models, which we hereinafter refer to as ‘type 1’ and ‘type 2’. The difference between the two lies in the chemistry. Type 1 models were computed with free chemistry, meaning the abundances of the chosen molecules (in our case, H$_2$O, CO, CO$_2$, CH$_4$, and NH$_3$) were free parameters. In the type 2 models, the thermochemical equilibrium abundances were computed using GGchem (Woitke et al. 2018) from the carbon to oxygen ratio and the metallicity of the atmosphere. The following species were considered in the chemistry as well as in the opacity sources: H$_2$O, CO, CO$_2$, CH$_4$, NH$_3$, HCN, TiO, VO, AlO, FeH, OH, C$_2$H$_2$, CrH, H$_2$S, MgO, H$^+$, Na, K.

In both cases, the atmospheres were characterised by an isothermal temperature structure. We also included grey clouds parameterised by the cloud top pressure $P_{\text{cloud}}$; in this parameterisation, the atmosphere is transparent for $P < P_{\text{cloud}}$ and opaque otherwise. Finally, hazes were also included, which were modelled by adding a grey opacity $k_{\text{haze}}$ to the atmosphere.

We generated two grids of transmission spectra, one for each type of model. For more details regarding the forward models, the ARGIS input files used to generate them are provided in the replication package.

Tables 1 and 2 detail the parameters that were allowed to vary for both grids, along with the ranges from which they were randomly sampled.

All of the forward models were generated for NIRSpec’s wavelength range and spectral resolution obtained from a Pan-Dexo (Batalha et al. 2017) simulation, and then rebinned to WFC3’s spectral range and resolution. The exact wavelength bins were extracted from real WFC3 observations reduced by the Exoplanets-A project (Pye et al. 2020).

2 For supersolar C/O ratios, ARCIS adjusts the oxygen abundance while keeping the carbon at solar abundance, and, for subsolar C/O ratios, it adjusts the carbon abundance while oxygen remains at a solar abundance. All other elements are then scaled so that the solar Si/O is matched. The metallicity is then adjusted by scaling the H and He abundances.

3 All data and codes necessary to reproduce the results of this work can be found here: https://gitlab.astro.rug.nl/ardevol/exocnn.
To make the retrievals as realistic as possible, and to facilitate comparison to Bayesian retrievals, we trained the algorithms with noisy spectra. To this end, we generated a number of noisy copies of each model, drawing from normal distributions with standard deviations described below. This process is called ‘data augmentation’, and it allows to make the training sets larger without computing more forward models, which would be significantly more time-consuming. Section 3.3 contains a discussion on the number of models and noisy copies necessary to train the machine learning algorithms.

For the WFC3 forward models, we introduced constant noise throughout the wavelength range with \( \sigma = 50 \) ppm for consistency with the previous literature (Márquez-Neila et al. 2018). For NIRSpec forward models we calculated the noise with PandExo, setting the noise floor to 10 ppm. We used the stellar parameters of HD209458, which is an optimistic case due to its brightness (\( m_J = 6.591 \)), but realistic nevertheless. HD209458 is a solar-type host to HD209458b and is one of the best studied hot Jupiters and target of several accepted JWST observing programs. In addition, we limited the noise calculation to one transit. In any case, scaling the noise for a different number of transits or a star of a different magnitude is straightforward, and a new simulation would only be required for a star of a different spectral type.

### 2.2. Pre-conditioning of the training data

The bulk of the transit depth is due to the planetary radii, with small variations due to the atmospheric features. As a result, spectra of planets with different radii can look essentially flat when compared to each other. This is illustrated in Fig. 2 (Top). If no pre-processing is applied, the CNNs will need to learn how the different features look at every radius and will learn to predict the radius virtually perfectly, but their performance for the rest of the parameters will lag behind. After testing different approaches, we found that simply subtracting the mean from each individual spectrum was the best way to increase the performance of the CNNs. As can be seen in Fig. 2 (Bottom), by removing the contribution from the planet itself, the atmospheric features become easily comparable between different spectra. This way, the CNNs can learn how spectral features look for all radii simultaneously. In order not to discard information, both the original and the normalised spectra were given as inputs as a two-column array. Finally, the spectra were converted from \( R_p \) to \( R_p/R_J \) units. In this way, the trained algorithms did not depend on the host-star properties.

As well as the spectra (features), the parameters (labels) also needed to be normalised. Due to the parameters having different ranges, the CNNs can reduce the loss by focusing on more accurately predicting the parameters with greater absolute values, such as the temperature \( T \) in our case. Because this parame-
ter can be between 500 and 5,000 K, improving the temperature predictions will have a higher impact on minimising the loss than improving the C/O, which only varies between 0.1 and 2. In order to avoid this, we used scikit-learn’s MinMaxScaler to linearly transform all parameters to be between 0 and 1.

Some combinations of parameters (namely high $T$ and low log $g$) caused unphysically large transit depths. To avoid this, we imposed the condition $R_T^2(\lambda) < 1.5R_{P,0}^2$, where $R_T^2(\lambda)/R_{P,0}^2$ is the average transit depth across the whole wavelength range and $R_{P,0}$ is the planetary radius at a pressure $P = 10$ bar. This left us with 68,000 and 64,000 training examples for the type 1 and type 2 retrievals, respectively. In both cases, 8,000 spectra were used for validation and 1,000 were reserved for testing.

3. Training and evaluation of the networks

We used the datasets described in the previous section to train CNNs. The choice of using convolutional layers for our neural networks was informed by previous results in the literature, which found that CNNs were the best performers amongst different machine learning algorithms (e.g. Soboczenski et al. 2018; Yip et al. 2021), while at the same time having a lower number of trainable weights than a fully connected neural network. A brief description of what CNNs are, along with our implementation and how they were evaluated, is presented below.

3.1. Convolutional neural networks

A dense neural network is a succession of linear transformations followed by non-linear activations. Given an input $x$ with $N$ features, the activations $a_i$ of a dense layer with $M$ neurons can be written as follows:

$$a_i = g\left(\sum_{j=0}^{N} w_{ij} x_j\right) \quad (0 \leq i \leq M),$$

where $w_{ij}$ are the network weights and $g$ is a non-linear function referred to as an activation function. These activations are subsequently used as the inputs for the next layer.

A 1D convolutional neural network (CNN) substitutes some of the dense layers with 1D convolutional layers. A 1D convolutional layer is defined by a number of filters of a certain size. Given an input $x$ of size $N \times L$, the activations of a 1D convolutional layer with $K$ filters $f_k$ of size $M \times L (M < N)$ can be written as follows:

$$a_k = g\left(\sum_{j=0}^{L} \sum_{i=0}^{M} f_{ji} x_{i+j}\right) \quad (0 \leq i \leq N - M),$$

In our case, $N$ is the number of wavelength bins, and since we feed both the original and pre-processed spectra as a two-column array, the input size is $N \times 2$. After applying $K$ convolutional filters, the output size is $N \times K$. A graphical representation of the inner workings of a CNN can be seen in Fig. 3. The filter ‘slides’ along the input, using Eq. 2 to obtain the output. Because the same filter is applied throughout the input, the number of trainable weights is reduced considerably when compared to fully connected layers. In addition, because the filter is applied to a group of inputs, CNNs are particularly suited for feature detection and are commonly used for tasks involving images. Training consists of finding the $w_{ij}$ or $f_{ji}$ that minimise a loss function (for a more detailed explanation, see Goodfellow et al. (2016)).

We used the Python package Tensorflow (Abadi et al. 2015) to implement our CNN. The network architecture is shown in Table 3. To select the architecture, a grid search in hyper-parameter space was conducted, choosing the smallest network possible without sacrificing performance. The total number of trainable weights for each combination of instrument and model type is shown in Table 4.

Max-pooling was used after each convolutional layer. Max-pooling reduces the size of an input by taking the maximum value over a group of inputs. We chose a pooling size of 2, downsizing an $N \times L$ input to a size of $(N/2 + 1) \times L$ or $[(N + 1)/2] \times L$ depending on whether $N$ is odd or even. The activation function was Rectified Linear Unit (ReLU) for every layer except for the last one, for which we used a sigmoid activation function. The ReLU activation function is defined as follows:

$$f(x) = \max(0, x),$$

The sigmoid activation function returns a value between 0 and 1 and is defined as

$$f(x) = \frac{1}{1 + e^{-x}}.$$  

Although a linear activation function in the output layer was found to reach a virtually identical performance, ultimately the sigmoid function was chosen to ensure that the predictions are within the selected parameter ranges. The optimisation algorithm used to update the network’s weights during training was the Adam optimizer (Kingma & Ba 2014).

![Diagram of 1D convolutional layer with a single filter. Adapted from Goodfellow et al. (2016).]

### Table 3. Architecture of our CNN. We denote the number of filters and the filter size in convolutional layers by ‘f’, and the number of neurons in the dense layer by ‘h’.

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Type</th>
<th>Trainable weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>WFC3</td>
<td>1</td>
<td>52,689</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>48,819</td>
</tr>
<tr>
<td>NIRSpec</td>
<td>1</td>
<td>437,713</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>433,843</td>
</tr>
</tbody>
</table>

### Table 4. Number of trainable weights of the four CNNs.
Loss
20
19
18
L

Cobb et al. (2019):ATIVE log-likelihood of the multivariate Gaussian, as presented in
Σ =

The latter is predicted via its Cholesky decomposition
the means and the covariance matrix of a multivariate Gaussian.

predict a single value for the parameters given a spectrum, but
depends on the input size, and it is therefore different for WFC3
and NIRSpec spectra.

to obtain a probability distribution, our CNN does not only
predict a single value for the parameters given a spectrum, but
the means and the covariance matrix of a multivariate Gaussian.

To obtain a probability distribution, our CNN does not only
predict a single value for the parameters given a spectrum, but
the means and the covariance matrix of a multivariate Gaussian.

where

y

µ

is the number of dimensions,

D

l_{dd}

the diagonal elements

L

y

the true values,

µ

the predicted values,

N

the number of spectra.

The coefficient of determination is measuring the normalised sum
of squared differences between predicted and true values, therefore
making it a joint proxy for the bias and the variance of the model.

In order to disentangle both, we also calculated the mean bias:

MB = \frac{1}{N} \sum_{i=1}^{N} (\mu_i - y_i).

Both a high \( R^2 \) and low mean bias are desired. Computing both
metrics is useful to determine whether a lower \( R^2 \) is due to a
higher bias or a lower variance, and vice versa.

Additionally, we also compared the predicted values with the
true values of the parameters. This is helpful to identify trends
and compare the performance of the different retrieval methods
in different regions of parameter space.

Retrieving atmospheric parameters from transmission spectra
is a degenerate problem (e.g. Brown 2001; Fortney 2005;
Griffith 2014; Fisher & Heng 2018). As such, the retrieved values
are often incorrect. When evaluating the performance of a
retrieval method it is crucial that we also consider the uncertainties
of the retrieved values. To do so for our large samples of
simulated observations, we computed the difference between
the predicted and true values in units of standard deviation. Because
the posterior probability distributions are typically not Gaussian,
a true standard deviation cannot be calculated. Instead, we de-
netermined the percentile of the posterior distribution that the true
value of the parameter fell in, or in other words, the fraction of
points \( p_{\%} \) in the posterior distribution that were smaller than the
true value. This was then converted into an equivalent standard
deviation:

\sigma_{eq} = \sqrt{2} \text{erf}^{-1}(1 - 2p_{\%}),

where \text{erf} is the error function and is defined as

\text{erf} z = \frac{2}{\sqrt{\pi}} \int_{0}^{z} e^{-t^2} dt.

We then evaluated the histogram of these differences in \( \sigma_{eq} \) to
see whether the retrieval method was overconfident, underconfi-
dent, or predicting accurate uncertainties.

3.2. Evaluating the performance of the CNNs
To evaluate the machine learning methods, as well as to facilitate
the comparison with nested sampling, we generated sets of 1,000
simulated observations for each combination of type of model
(1 and 2) and instrument (WFC3 and NIRSpec), yielding four
sets in total. We then performed retrievals for these simulated
observations with our CNNs and Multinest. All the Multinest
retrievals had flat priors for all of the parameters.

The first metric we took into account was the coefficient of
determination \( R^2 \) between the predicted and true values of
the parameters. We considered the predicted value to be the
median of the posterior distribution. In particular, we used the
scikit-learn (Pedregosa et al. 2011) implementation of the
\( R^2 \), defined as follows:

\begin{equation}
R^2_{y,\mu} = 1 - \frac{\sum_{i=1}^{N}(y_i - \mu_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2},
\end{equation}

where \( y_i \) are the true values of a given parameter, \( \bar{y} \) the average of
the \( y_i \), \( \mu_i \) the predicted value, and \( N \) the number of spectra.

Figure 4 shows a schematic representation of the architecture.
The blue, green, and red boxes represent the convolutional
filters. Although in the diagram they are only drawn in the first
column, the filters are 2D and are applied to all columns. The
column height decreases after each max-pooling operation and
depends on the input size, and it is therefore different for WFC3
and NIRSpec spectra.

To obtain a probability distribution, our CNN does not only
predict a single value for the parameters given a spectrum, but
the means and the covariance matrix of a multivariate Gaussian.

where

\Sigma = (LL^T)^{-1}.

The loss function used to achieve this was the negative
log-likelihood of the multivariate Gaussian, as presented in
Cobb et al. (2019):

\begin{equation}
\mathcal{L} = -2 \sum_{d=1}^{D} \log(l_{dd}) + (y - \mu)^T LL^T (y - \mu),
\end{equation}

where

\Sigma

the covariance matrix

D

the number of dimensions

l_{dd}

diagonal elements

L

y

the true values

\mu

the predictions.

Figure 5 shows

how the training and validation losses decreased as the training

progressed. We stopped the training if the validation loss did not

improve for ten epochs.

Once trained, to account for the observational error during
a retrieval, we generated 1,000 noise realisations and did 1,000
forward passes of the CNN, one for each noise realisation. Fi-

nally, we randomly picked one point from each of the 1,000 nor-
mal distributions returned, which collectively formed the proba-

bility distribution of the parameters.

Figure 4 shows a schematic representation of the architecture.
The blue, green, and red boxes represent the convolutional
filters. Although in the diagram they are only drawn in the first
column, the filters are 2D and are applied to all columns. The
column height decreases after each max-pooling operation and
depends on the input size, and it is therefore different for WFC3
and NIRSpec spectra.

To obtain a probability distribution, our CNN does not only
predict a single value for the parameters given a spectrum, but
the means and the covariance matrix of a multivariate Gaussian.

The latter is predicted via its Cholesky decomposition
\Sigma = (LL^T)^{-1}.

The loss function used to achieve this was the negative
log-likelihood of the multivariate Gaussian, as presented in
Cobb et al. (2019):

\begin{equation}
\mathcal{L} = -2 \sum_{d=1}^{D} \log(l_{dd}) + (y - \mu)^T LL^T (y - \mu),
\end{equation}

where

D

the number of dimensions

l_{dd}

diagonal elements

L

y

the true values

\mu

the predictions.

Figure 5 shows

how the training and validation losses decreased as the training

progressed. We stopped the training if the validation loss did not

improve for ten epochs.

Once trained, to account for the observational error during
a retrieval, we generated 1,000 noise realisations and did 1,000
forward passes of the CNN, one for each noise realisation. Fi-

nally, we randomly picked one point from each of the 1,000 nor-
This metric is arguably more important than the $R^2$ or the $MB$, at least for our goal of characterising exoplanetary atmospheres. Predicted values that are further from the ‘truth’ but have uncertainties that account for this distance will be preferred over predictions which are closer to the ‘truth’, but due to too small uncertainties, are statistically incompatible with the true value.

### 3.3. On the number of training examples

We investigated the number of training examples that were needed to train our CNNs. This is an important factor to take into account when considering using machine learning, as its use may not be advantageous enough if too many forward model computations are needed to train it. We trained our CNNs using training sets of varying sizes and measured how the loss decreased with increasing training set size.

First of all, we modified the number of noisy copies made of every forward model computed with ARCiS. Fig. 6 (Top) shows how the loss decreases with increasing number of noisy copies, although there is little improvement beyond 20 noisy copies, if any at all. To facilitate comparison, all the curves were shifted so they reached a loss of zero.

When we instead modified the number of forward models used to train the CNNs while keeping the number of noisy copies constant (20), we saw a much more drastic decrease in performance with a low number of forward models (see Fig. 6 (Bottom)). This is an intuitive result as the CNN saw fewer combinations of parameters during training; in the previous case, it still saw many parameter combinations, but it did not see as many noise realisations. When increasing the number of training examples, the loss improved much more dramatically for NIRSpec than for WFC3, with the latter showing very little improvement beyond 30,000 examples. Since the NIRSpec spectra contain more information, it was to be expected that more examples would be needed to train the CNNs.

### 3.4. Limitations of the current approach

The machine learning retrieval framework presented in this paper has some limitations, some related to our current implementation, and others inherent to supervised machine learning approaches. In our current implementation, the noise level is not an input but instead all training examples are sampled from the same noise distribution. This results in sub-optimal results when performing retrievals for observations with different noise levels. It is possible to retrain the CNNs with the observational noise, which we do in Section 4.3, but it is far from ideal. Additionally, we trained the CNNs to predict the means and covariance matrix of a multivariate Gaussian, constraining the shape of the posterior distributions. This makes it impossible to predict multimodal or uniform distributions. The latter in particular are very common as often only an upper or lower bound for a parameter can be retrieved.

The limitations related to the supervised learning approach itself are the fixed wavelength grid and the fixed model type. The training examples contain only the transit depth, no wavelength axis is provided. Therefore to perform a retrieval for a given observation, its wavelength axis must match the one used during training exactly. This is a lesser inconvenience, as observations from the same instruments will share the same wavelength axis and so one could simply train networks for specific instruments. Lastly, this approach lacks flexibility regarding the kinds of models used in the retrieval. Here, we trained CNNs for two different kinds of models, and if we wanted to analyse an observation with a different model (say, for example, with a different list of chemical species), it would be necessary to generate a completely new training set and train a CNN. Of course, since eventually the different kinds of models would likely be used to analyse more than one observation, using machine learning would probably still be computationally advantageous in the long run.

### 4. Results

In order to test the performance of the CNNs and have a benchmark to compare it against, we generated four sets of 1,000 simulated observations (one for each combination of instrument and model type), on which retrievals were performed with the CNNs and Multinest. Below, we present the results for the different evaluation metrics discussed in Section 3. A summary of these results can be found in Table 5.
Table 5. Summary statistics of CNN and Multinest bulk retrievals.

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Type</th>
<th>CNN</th>
<th>MN</th>
<th>CNN</th>
<th>MN</th>
</tr>
</thead>
<tbody>
<tr>
<td>WFC3</td>
<td>1</td>
<td>0.30</td>
<td>0.30</td>
<td>0.0</td>
<td>8.5</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.36</td>
<td>0.29</td>
<td>0.1</td>
<td>6.2</td>
</tr>
<tr>
<td>NIRSpec</td>
<td>1</td>
<td>0.67</td>
<td>0.76</td>
<td>0.1</td>
<td>10.6</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.71</td>
<td>0.76</td>
<td>0.2</td>
<td>8.2</td>
</tr>
</tbody>
</table>

Fig. 7. Predicted vs. true values of temperature and log g for type 1 retrievals performed for simulated WFC3 transmission spectra. (Left) Multinest. (Right) CNN.

4.1. Predicted versus true values

The predicted versus true values plots for all parameters, instrument, and model types can be seen in Appendix A. For WFC3 type 1 retrievals, the CNN and Multinest reached the same $R^2$, whereas for WFC3 type 2, the CNN actually outperformed Multinest in this metric. The CNNs also achieved a lower MB (except for the log $g_{\text{hand}}$ in the type 2 retrievals), although neither method showed large biases.

For NIRSpec retrievals, Multinest reached a higher average $R^2$ in both cases. Yet, if we focus on the type 2 retrievals, the $R^2$ of all the parameters was actually very close between both methods, with the CNN only doing significantly worse for the C/O and the $P_{\text{cloud}}$. Regarding the bias, there was no clear winner, and no method reached systematically lower biases for all parameters. However, as for the WFC3 spectra, the bias was low for both methods.

More importantly, when comparing the predicted and true values, we observed similar trends for both methods, which is an indication that the lack of predictive power in some regions of parameter space is mostly due to the lack of information in the data itself. There were, however, a few differences worth mentioning. Firstly, for WFC3 data (for both type 1 and 2 retrievals), the predicted versus true values plots for the temperature $T$ and the log $g$ are markedly different, as can be seen in Fig. 7 (only illustrated for type 1 retrievals; type 2 retrievals can be found in Appendix A). In both cases, the CNN achieved a higher $R^2$ and lower MB.

Fig. 8. Predicted vs. true values of NH$_3$ abundance for type 1 retrievals performed for simulated NIRSpec transmission spectra. (Left) Multinest. (Right) CNN.

The other main differences between both methods can be seen for parameters where the predicted versus true values clump along two ‘tails’. This can be interpreted as a degeneracy in the data for which Multinest predicts one of two possible values. In these cases, we observed that the CNN predictions were distributed between both ‘tails’ without clumping. This is very evident in the predictions of the molecular abundances in type 1 retrievals. Fig. 8 illustrates this for the retrieved values of the NH$_3$ abundance from NIRSpec spectra.

4.2. Uncertainty estimates

The coefficients of determination between predicted and true values discussed above are, of course, only half of the picture. As we have seen, there are many regions in parameter space for which the correct values of the parameters were not retrieved. Whether the retrieved parameters agree with the ‘ground truth’ then depends on the entire posterior distribution. Following the method detailed in Section 3, we quantified which method got closer to the true values of the parameters.

For type 2 retrievals of NIRSpec spectra, the histogram of the distances (in units of standard deviation) between predictions and true values for both methods can be seen in Fig. 9. The ‘ideal case’ represents our statistical expectation that ∼68% of predictions should be within 1σ of the ground truth, ∼27% within 2σ, and so on. For all other model-type and instrument combinations, the figures can be found in Appendix B. Here, our CNNs followed the expected values closely, doing better than Multinest. For the CNNs, the fraction of predictions lying more than 3σ away from the true value was always below 0.3%. While this remained true for the retrievals of synthetic WFC3 spectra, in this case we found an excess of predictions between 1σ and 2σ from the true value and a lack of predictions within 1σ and between 2σ and 3σ of the ground truth. Multinest, on the other hand, can be off by more than 3σ in between ∼2% and ∼20% of retrievals, depending on the parameter, the model type, and the instrument. This means that there was a non-negligible fraction of spectra for which Multinest underestimated the uncertainties of the parameters it retrieved. This goes against our expectations as it is typically assumed that Multinest finds the ‘true’ posterior probability distribution. Although finding the reasons behind this would be extremely valuable to the community, our goal in this paper is simply to compare the results obtained with Multinest and our CNNs.
A& téléphone: manuscript no. output

4.3. Real HST observations

Until now, the CNNs have only been tested with simulated observations. The Exoplanets-A database of consistently reduced WFC3 transmission spectra of 48 exoplanets (Pye et al. 2020) provided us with a unique opportunity to test our CNNs on a relatively large sample of real data. With this dataset, we can study the real world performance of our CNNs and compare it with Multinest. Retrievals of these spectra were run with both our CNNs and Multinest, and the results were compared. In order to apply our machine learning technique to the real dataset, for each of the planets the CNNs were retrained with the observational noise from the real data applied to the training sets (see Section 2). This is not ideal, and work is ongoing to avoid it in the future; however, for now it allows us to test our framework with real observations. Examples of the corner plots for retrievals for WASP-12b can be seen in Appendix C.

The differences (measured in $\sigma$) between the parameters retrieved by the two methods can be seen in Fig. 10. These were calculated by assuming, for the method that predicted a lower value, a normal distribution with mean and standard deviation the median and the 84th minus the 50th percentile of the posterior respectively; and similarly for the other method but with the 50th minus 16th percentile as the standard deviation. For most planets, both methods agreed within 1$\sigma$, with very few cases in which they disagreed by more than 2$\sigma$. These differences are discussed in Sect. 6.

5. Uncomfortable retrievals

The forward models we use to analyse exoplanet observations will virtually always be an incomplete description of the real atmosphere. On top of this, the assumptions we make in the models can be wrong, causing our models to never be a one-to-one match of the actual observation. In machine learning jargon, these are generally referred to as ‘adversarial examples’. It is therefore important to know beforehand what to expect in these cases. Will our retrieval framework still provide good results? Will it mislead us into an incorrect characterisation of the exoplanet? Or, will it break, letting us know that this is not some-
thing it was prepared for? To figure out how our machine learning retrieval framework responds in these scenarios, and to compare it with nested sampling, we modified our synthetic observations in different ways while keeping the retrieval frameworks identical. In particular, we ran three experiments, namely adding an extra chemical species to our type 1 models, removing species from our type 2 models, and simulating the effect of stellar spots.

5.1. Type 1 retrievals of synthetic observations with AlO

Firstly, we added AlO to our type 1 models with an abundance log AlO = −5.25. This choice was influenced by Chubb et al. (2020) reporting AlO on WASP-43b with this abundance. We generated 735 simulated NIRSpec observations (the original set was 1,000, but we filtered unphysical atmospheres as discussed in section 2), sampled randomly from the parameter space. Retrievals on these synthetic spectra were then performed using the same retrieval frameworks as in section 4, which assume that no AlO is present in the atmosphere. An example of how the addition of AlO at an abundance of log AlO = −5.25 modifies the transmission spectrum can be seen in Fig. 11.

We analysed the retrievals as detailed in Section 3, just as we did in Section 4. As expected, both Multinest and the CNN achieved a lower $R^2$, with Multinest’s $R^2$ still being better than that of the CNN ($R^2 = 0.39$ vs. $R^2 = 0.26$). Multinest also achieved a lower MB for most parameters. However, when we look at the whole picture and take the uncertainties into consideration (Fig. 12), the CNN would still be the preferred method, with only ~9% of predictions further than 3σ from the truth compared to ~41% for Multinest.

5.2. Type 2 retrievals of synthetic observations without TiO or VO

To test the effects of an incorrect list of chemical species in type 2 retrievals, we removed TiO and VO and observed the effect it had on the retrieved parameters. TiO and VO were expected to be major opacity sources in the atmospheres of hot Jupiters (e.g. Hubeny et al. 2003; Fortney et al. 2008); however, the lack of detection in multiple planets (e.g. Helling, Ch. et al. 2019; Merritt et al. 2020; Hoeijmakers et al. 2020) suggests that perhaps these species are condensing out of the gas phase at the day-night limb. We therefore investigated how different assumptions regarding the presence of TiO and VO in the gas phase affected the results of the retrievals, stressing again that our goal here is simply to compare how Multinest and our CNN behave in this scenario. To compare the behaviour of the CNN and Multinest under this scenario, we generated a set of 349 simulated observations (filtered down from 500 as discussed in Section 2 to get rid of extreme atmospheres) and performed retrievals on them with our CNN and Multinest, in both cases assuming TiO and VO were part of the gas phase chemistry of the atmosphere. Removing these species changes only the optical part of the spectrum (see Fig. 13), and thus HST is unable to observe the differences. We therefore limited this experiment to simulated NIRSpec observations.

The CNN reached a coefficient of determination between predictions and truths of $R^2 = 0.67$, falling from the regular value of $R^2 = 0.71$, while for Multinest it falls from $R^2 = 0.76$ to $R^2 = 0.63$. In particular, the C/O predictions made by our CNN in this scenario are significantly better than those of Multinest, as can be seen in Fig. 14. We found the bias to be small with both methods, with no method doing systematically better for all parameters. Most importantly, however, we can see in Fig. 15 that Multinest was much more overconfident, with ~26% of predictions more than 3σ away from the true value compared to only ~3% for the CNN.

5.3. Simulating unocculted star spots in the synthetic observations

Finally, we also simulated the effect of unocculted star spots in the transmission spectra following the method described by Zellem et al. (2017). If the spot coverage of the host star $f$ (defined as the fraction of the surface covered by spots) is heterogeneous and the planet transits in front of an area with no spots (or fewer than in the unocculted region), the planet will be blocking a hotter, bluer region than the overall stellar disc, and thus the transit depth will be larger in shorter wavelengths. The transit depth modulated by the unocculted stellar activity can be calculated using

$$\delta_{\text{active,transit}} = \left( \frac{R_p}{R_\star} \right)^2 \left[ 1 + f \left( \frac{B_{\text{spots}}}{B_{\text{stars}}} - 1 \right) \right]^{-1},$$

where $\delta_{\text{active,transit}}$ is the transit depth and $B_{\text{spots}}$ and $B_{\text{stars}}$ are the black-body spectra of the spots and the star, respectively.

To test how this changes the retrieved parameters, we assumed a star with $T_{\text{star}} = 5500K$, $T_{\text{spot}} = 4000K$ and a spot coverage of $f = 0$ in the transiting region and $f = 0.01$ in the non-transiting region. These values are consistent with each other and were taken from Rackham et al. (2019). Fig. 16 shows how these spots modify the observed transmission spectrum.

Again, we generated a set of 500 simulated NIRSpec observations of exoplanets orbiting heterogeneously spotted host stars as described above (with type 2 model complexity) and performed retrievals on them with our CNN and Multinest. Similarly to previous experiments, we observed a reduction in $R^2$, although Multinest still reached a higher coefficient of determination ($R^2 = 0.70$) than our CNN ($R^2 = 0.64$). Despite this, the CNN retrievals would be preferred since Multinest was again quite overconfident, as is made clear by Fig. 17. As for the previous experiments, the bias was low for all parameters with no method performing systematically better than the other for all parameters. Averaging over all parameters, ~12% of Multinest predictions were more than 3σ away from the ground truth, whereas this fraction remained at the expected value of ~0.3% for our CNN.

6. Discussion

As presented in Section 4, our CNNs outperformed Multinest when doing retrievals on WFC3 simulated observations, achiev-
that both methods recovered similar information from the data, and both struggled in the same regions of parameter space. Interestingly, when both methods differed significantly (which was only for some parameters in retrievals of WFC3 simulated data), the CNN achieved a higher $R^2$ and lower $MB$.

Aside from the CNN retrievals and their comparison to Multinest, to the best of our knowledge the exercise of retrieving a large sample of simulated data with Multinest had not been done previously. This was useful in and of itself, as we now have statistical information on how commonly Multinest returns too narrow posterior distributions, as well as on how well different parameters can be retrieved in different regions of parameter space. The results of these bulk retrievals are worthy of a more in-depth analysis, which falls outside the scope of this paper.

One of the advantages of our CNNs is the ability to still provide a plausible answer even when the models with which they were trained did not match the observation to be retrieved, as shown in Section 5. Multinest did poorly in these experiments. Yip et al. (2021) showed that CNNs focus solely on specific features to predict the parameters. On the other hand, Multinest tries to find the best fit to the whole spectrum, and it is therefore more easily thrown off by modifications to the spectrum, even when the features caused by the parameter we want to predict remain unchanged. This ability is crucial in the real world, as the models used will always be simplifications of the real atmospheres, and/or will be based on a set of limiting assumptions.

On the other hand, Multinest has the advantage that it calculates the Bayesian evidence, making it possible to compare models with different assumptions and select the best one. This does not negate the advantage that our CNNs had in these experiments, as running a model selection requires multiple retrievals, incurring in a higher computational cost.

The main limitation of our machine learning framework is that it can only predict Gaussian posteriors. This is a poor approximation of multi-modal posteriors or uniform posteriors with lower and upper bounds. The latter is often the case for the molecular abundances in type 1 retrievals, for which an upper bound might be the only information that can be inferred from the spectrum; for the C/O, where it might only be possible to infer whether the atmosphere is carbon or oxygen rich; or for $k_{haze}$ and $P_{cloud}$, for which sometimes only an upper or lower bound, respectively, might be inferred. However, when we

---

**Fig. 12.** Distance in $\sigma$ between predictions and ground truths for level 1 retrievals of simulated NIRSpec observations of exoplanet atmospheres with $\log AlO = -5.25$. (Left) Multinest. (Right) CNN.

**Fig. 13.** Comparison between transmission spectra of two identical atmospheres, except for the removal of TiO and VO in one of them. The red dashed lines indicate WFC3’s wavelength coverage, clearly showing its inability to detect this difference.

**Fig. 14.** Predicted vs. true values of the C/O of simulated spectra without TiO and VO. (Left) Multinest. (Right) CNN.

---

It is also important to highlight the similar trends in the predicted values when compared with the ground truths for the CNNs and Multinest, particularly for NIRSpec data. This tells us
tested likelihood-free machine learning methods such as a random forest or a CNN with Monte Carlo dropout (Gal & Ghahramani 2016) trained with the MSE loss (mean squared error), they performed worse than the CNNs trained with the negative log likelihood loss. A random forest reached a lower $R^2$ and was slightly worse at estimating the uncertainties, while a likelihood-free CNN reached a similar (even slightly higher) $R^2$ but was very overconfident and yielded too narrow posterior distributions.

The other limitations of our CNNs are the fixed wavelength grid and noise, which really hinder its real world usability. For the purpose of retrieving real WFC3 transmission spectra, we retrained the CNNs with the observational noise of each observation, which is impractical. Future work will focus on adding flexibility in this regard, as well as making it possible to retrieve non-Gaussian posterior distributions.

Regarding the retrievals of the Exoplanets-A transmission spectra of 48 exoplanets, we want to highlight that although we benchmark our CNNs against Multinest, when both methods disagree, our previous analyses point in the direction of the CNN’s predictions being more trustworthy, even if less constraining. Because no ‘ground truth’ exists for these spectra, it is difficult to say which is actually correct.

Not surprisingly, where the CNNs excel is in the time they take to do a retrieval. On a regular laptop (Intel i7-8550U with 16GB of RAM) it takes ~ 1 s with 1000 points in the posterior (meaning 1000 forward passes of the network, each with a different noise realisation of the spectrum). With Multinest it would take from ~ 10 min to multiple hours, depending on whether it is a WFC3 or NIRSpec observation and the model complexity.

The CNNs were trained on a cluster with 80 Intel Xeon Gold 6148 CPUs and 753 GB of RAM. Table 6 summarises the training times of the four CNNs.

This difference will only become larger when using higher model complexity or data from upcoming facilities with lower noise, higher spectral resolution, and larger wavelength coverage. As an example, 3D nested sampling retrievals with self-consistent clouds and chemistry might not be feasible or practical with current techniques, and methods such as machine learning might be the solution.

Because of their speed, machine learning retrievals are also specially suited for any application in which multiple retrievals need to be done. An example of this might be retrievability studies, in which many spectra are retrieved across parameter space to see what information can be extracted and how different parameters affect the retrievability of other parameters. This information is of great relevance to inform the design of future instrumentation or observing programmes, and the increased speed of machine learning retrievals provides a huge advantage.

### 7. Conclusions

In this paper, we present extensive comparisons between machine learning and nested sampling retrievals of transmission spectra of exoplanetary atmospheres. This represents a step forward from the previous literature where comparisons had been limited to a few test cases.

Machine learning methods can be a powerful alternative to Bayesian sampling techniques for performing retrievals on trans-
mission spectra of exoplanetary atmospheres. Our work shows that CNNs can achieve a similar performance and even behave more desirably than Multinest in certain situations. In particular, when taking into account the retrieved values and their uncertainties, the answers our CNN provided were virtually always correct, contrary to those of Multinest. For the latter, despite being the current standard for retrieving exoplanetary transmission spectra, we found that for a significant fraction of spectra (~ 8%), its answer was off by more than 3σ.

When comparing both methods with real HST transmission spectra, we found them to agree within 1σ of each other for most cases. However, in the cases where they disagreed it was difficult to reach a verdict on which of the two was correct.

Finally, when retrieving spectra with different characteristics than those used in our retrieval frameworks (such as having different chemistry or being contaminated by stellar activity), the CNNs performed significantly better, with Multinest underestimating the uncertainty in ~ 12% to ~ 41% of cases.

Acknowledgements. We wish to thank Daniela Huppenkothen for her fruitful discussions and her valuable comments on the manuscript. We would also like to thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Peregrine high performance computing cluster. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 860470.

References


Fig. 17. Distance in σ between predictions and ground truths for type 2 retrievals of simulated NIRSpec observations of exoplanets orbiting around heterogeneously spotted host stars. (Left) Multinest. (Right) CNN.

Márquez-Neila, P., Fisher, C., Szuntatt, R., & Heng, K. 2018, Natural Astronomy, 2, 719
Skilling, J. 2006, Bayesian Analysis, 1, 833
Appendix A: Prediction versus truth plots

Here, we present the full comparisons between predicted and true values of the parameters for all the different retrievals we ran in this work. In these figures, each blue dot corresponds to an individual spectrum and the predicted value of the parameter is the median of the retrieved posterior. The red diagonal line represents perfect predictions.

![Prediction versus truth plots](image)

**Fig. A.1.** Predicted vs. true values for type 1 retrievals performed for simulated WFC3 transmission spectra. (Left) Multinest. (Right) CNN.
Fig. A.2. Predicted vs. true values for type 2 retrievals performed for simulated WFC3 transmission spectra. (Left) Multinest. (Right) CNN.
Fig. A.3. Predicted vs. true values for type 1 retrievals performed for simulated NIRSpec transmission spectra. (Left) Multinest. (Right) CNN.
Fig. A.4. Predicted vs. true values for type 2 retrievals performed for simulated NIRSpec transmission spectra. (Left) Multinest. (Right) CNN.
Fig. A.5. Predicted vs. true values for type 1 retrievals performed for simulated NIRSpec transmission spectra with log AlO = −5.25. (Left) Multinest. (Right) CNN.
Fig. A.6. Predicted vs. true values for type 2 retrievals performed for simulated NIRSpec transmission spectra of atmospheres without TiO or VO. (Left) Multinest. (Right) CNN.
Fig. A.7. Predicted vs. true values for type 2 retrievals performed for simulated NIRSpec transmission spectra of exoplanets orbiting heterogeneously spotted host stars. (Left) Multinest. (Right) CNN.
Appendix B: Difference between predicted and true values of the parameters

Here, we present the distributions of $\sigma_{e,q}$ between the predicted and true values of the parameters that were not shown in the main text. The ‘ideal case’ represents the fraction of predictions we would expect to find in each bin if both methods were correctly estimating the uncertainty.

Fig. B.1. Distance in $\sigma$ between predictions and ground truths for type 1 retrievals performed for simulated WFC3 observations. (Left) Multinest. (Right) CNN.

Fig. B.2. Distance in $\sigma$ between predictions and ground truths for type 2 retrievals performed for simulated WFC3 observations. (Left) Multinest. (Right) CNN.
Fig. B.3. Distance in $\sigma$ between predictions and ground truths for type 1 retrievals performed for simulated NIRSpec observations. (Left) Multinest. (Right) CNN.
Appendix C: Corner plots for WASP-12b

Here, we present the corner plots of the posterior distributions retrieved for WASP-12b. We note that both our CNN and Multi-nest show good agreement for the type 2 retrievals, but type 1 retrievals disagree for $T$ and $\log g$.

![Corner plots for WASP-12b](image)

Fig. C.1. Corner plots for type 1 retrievals for WASP-12b.
Fig. C.2. Corner plots for type 2 retrievals for WASP-12b.