The Astronomical Journal, 162:195 (29pp), 2021 November https://doi.org/10.3847/1538-3881/ac1744

Peeking inside the Black Box: Interpreting Deep-learning Models for Exoplanet Atmospheric Retrievals

Kai Hou Yip, Quentin Changeat, Nikolaos Nikolaou, Mario Morvan, Billy Edwards, Ingo P. Waldmann, and Giovanna Tinetti
Department of Physics and Astronomy, University College London, Gower Street, WC1E 6BT London, UK; kai.yip.13@ucl.ac.uk
Received 2020 November 22; revised 2021 July 20; accepted 2021 July 20; published 2021 October 13

Abstract

Deep-learning algorithms are growing in popularity in the field of exoplanetary science due to their ability to model highly nonlinear relations and solve interesting problems in a data-driven manner. Several works have attempted to perform fast retrievals of atmospheric parameters with the use of machine-learning algorithms like deep neural networks (DNNs). Yet, despite their high predictive power, DNNs are also infamous for being "black boxes." It is their apparent lack of explainability that makes the astrophysics community reluctant to adopt them. What are their predictions based on? How confident should we be in them? When are they wrong, and how wrong can they be? In this work, we present a number of general evaluation methodologies that can be applied to any trained model and answer questions like these. In particular, we train three different popular DNN architectures to retrieve atmospheric parameters from exoplanet spectra and show that all three achieve good predictive performance. We then present an extensive analysis of the predictions of DNNs, which can inform us–among other things–of the credibility limits for atmospheric parameters for a given instrument and model. Finally, we perform a perturbation-based sensitivity analysis to identify to which features of the spectrum the outcome of the retrieval is most sensitive. We conclude that, for different molecules, the wavelength ranges to which the DNNs predictions are most sensitive do indeed coincide with their characteristic absorption regions. The methodologies presented in this work help to improve the evaluation of DNNs and to grant interpretability to their predictions.

Unified Astronomy Thesaurus concepts: Convolutional neural networks (1938); Neural networks (1933); Exoplanet atmospheric composition (2021); Transit instruments (1708); Astronomical instrumentation (799)

1. Introduction

Exoplanetary science is one of the fastest-expanding fields in astronomy. The increasing number of discovered exoplanets has provided necessary motivation for subsidiary disciplines to grow.

Exoplanet atmosphere characterization, in particular, is one of the frontiers of the field. Transit spectroscopy, which consists of observing transits at different wavelengths, has allowed astronomers to robustly detect chemical species such as water vapor, carbon-bearing molecules, oxides, and alkali species in the atmosphere (Fossati et al. 2010; Linsky et al. 2010; Berta et al. 2012; de Kok et al. 2013; Mandell et al. 2013; Ehrenreich et al. 2014; Barman et al. 2015; Macintosh et al. 2015; MacDonald & Madhusudhan 2017; Arcangeli et al. 2018; Edwards et al. 2020). These successes are built upon the foundation of generations of ground-based and space-based instruments, such as the Very Large Telescope, the Spitzer Space Telescope, and the Hubble Space Telescope. The accumulation of such observations over the years has enabled large-scale statistical studies of subpopulations of exoplanets, e.g., on a wide range of Hot Jupiter atmospheres (Iyer et al. 2016; Sing et al. 2016; Fisher & Heng 2018; Tsiaras et al. 2018).

Looking forward, the next generation of space missions, dedicated to exoplanet characterization, such as Ariel (Tinetti et al. 2018), Twinkle (Edwards et al. 2019b), and JWST (Greene et al. 2016), will be launching within the next decade, delivering spectra with broader wavelength coverage and higher spectral resolution. The prospect of better data quality has encouraged further development in forward modeling of exoplanet spectra and atmospheric retrieval techniques (e.g., Irwin et al. 2008; Madhusudhan & Seager 2009; Line et al. 2013; Al-Refaie et al. 2021; Ormel & Min 2019; Zhang et al. 2019).

Artificial intelligence, and in particular deep learning (DL), has risen in popularity in recent years. Deep-learning algorithms have proven successful in efficiently deriving useful models from large amounts of high-dimensional data. Such models, which are capable of capturing highly nonlinear relationships, have been used to solve hard problems in a wide range of application domains, such as image classification (Al-Saffar et al. 2017), natural language processing (Young et al. 2017), and time series analysis (Ismail Fawaz et al. 2019). Typically, training such complex models without overfitting requires a large amount of data.

The use of DL and general machine-learning (ML) algorithms has become widespread in the field of exoplanet research. McCauliff et al. (2015) applied Random Forests (RFs) to identify candidate transit signals from Kepler light curves. Subsequently, Shallue & Vanderburg (2018) and Pearson et al. (2018) demonstrated the potential of DL in transit candidate vetting and inspired follow-up applications on other instruments (e.g., Chaushev et al. 2019; Dattilo et al. 2019; Yu et al. 2019; Osborn et al. 2020). Schanche et al. (2019) developed a combination of shallow ML and DL models to improve vetting accuracy on WASP data. Additionally, Long Short-Term Memory Network (Morvan et al. 2020) and RF (Krick et al. 2020) were implemented to model and correct the systematics of Spitzer IRAC exoplanet transit modeling and detection.

On the planetary characterization front, despite the fact that retrieval has always been the standard, universal approach when it comes to inferring atmospheric properties, retrieval frameworks are not without weaknesses. It is essentially a fitting

algorithm that attempts to estimate a best-fit solution given a forward model and a list of parameters with their bounds. The limitation imposed by the observational data means that multiple solutions, regardless of their feasibility, could exist, and it is left for the user to judge the feasibility of the outcome. A neural network, on the other hand, is able to learn the intricate, nonlinear relationships between parameters and the observational data. The development of a neural network driven retrieval is still at its early stages, but there have been attempts to infer atmospheric properties from a network. Waldmann (2016) pioneered the application of DL models to identify the existence of molecular species in a transmission spectrum. Márquez-Neila et al. (2018) used RFs to infer atmospheric properties, such as temperature and water abundance, from exoplanet spectra. The success of RFs inspired further applications. Fisher et al. (2020) applied the same algorithm on high-resolution ground-based observations. Nixon & Madhusudhan (2020) built upon the work of Márquez-Neila et al. (2018) and produced an RF-generated posterior distribution with excellent agreement to one from a fully Bayesian retrieval. On the neural network front, Zingales & Waldmann (2018) utilized a Generative Adversarial Network (GAN), a DL network architecture that can generate the closest synthetic spectrum and its associated atmospheric properties for a given observed spectrum. Cobb et al. (2019) developed a Bayesian Neural Network to model the posterior distribution between atmospheric parameters. To speed up the computationally expensive radiative transfer simulation process, Himes et al. (2020) trained an ML surrogate forward model and demonstrated its potential to significantly reduce retrieval time.

However, despite their predictive power, models generated using some of the most powerful ML algorithms –DL being the most prominent example–are often regarded as "black boxes." Deep neural networks trained on large, high-dimensional data sets are models that typically contain thousands or even millions of parameters learnt from data. This is what allows them to model complex nonlinearities within the data and ultimately leads to their accurate predictions. But the same complexity also makes it difficult to understand what factors contribute the most to DNN predictions. Developing methodologies to make such models more interpretable is a growing area in the field of ML (Molnar 2019). Better interpretability and a more robust understanding of the DNN's uncertainties may lead to a broader adoption of these methods in the physical sciences. It allows us to understand if our models make correct predictions for the right reasons, why our models are wrong—when they are—and how to correct for their biases. Beyond this, interpretability methods allow for identifying the ever-present biases in the data set—especially relevant in the case of astrophysics where a lot of effort is dedicated to analyzing simulated data in preparation for deploying a new instrument. Finally, understanding ML models in terms we can relate to the underlying theory of the application domain (e.g., astrophysics) can provide us with new theoretical insights.

In this paper, we will investigate the use of several DNN architectures (MLPs,[1] CNNs, and LSTMs) in the problem of exoplanet atmospheric retrievals. We emphasize that our goal is not to train a neural network to perform retrieval—the two methodologies might have similar outcomes but they are different in their approach. Our goal is to investigate how to probe into the inner workings of DNN models trained to perform this prediction task. We will demonstrate how to analyze the performance of a trained model and answer questions like: "What is the true abundance range if the model predicts an abundance of H_2O of 10^{-5}?" We will use this analysis to explore the performance of an instrument/observational strategy (in our case, the Deep survey by the ESA Ariel space telescope; Tinetti et al. 2018). Moreover, we will present a general methodology that can be applied to any trained neural network (or rather, any statistical predictive model) to understand how its input affects its predictions. The proposed method is a quantification of the sensitivity of the trained model to the various features of the input. In the context of atmospheric retrievals, we will visualize how different features of the spectrum affect the quality of the retrieval. In other words, we ask: "Where does a neural network look in the spectrum to determine the value of each of the retrieved parameters?" As we will see, the answer mostly agrees with our physical intuition, yet it occasionally brings to light interesting new insights about the model, the data, or the underlying physics. Our implementation is available on Github[2] and Zenodo (Yip 2021).

2. Problem Statement, Data, and Models

2.1. Objectives

There are three main objectives in this investigation:

1. To train DNNs to infer different atmospheric parameters from a transmission spectrum. We demonstrate that several DNN architectures (MLPs, CNNs, and LSTMs) are capable of producing models that achieve good predictive performance in this task. We use the best model obtained at this stage as an example model for the next two stages (which, we should note, are not tied to any specific model or learning algorithm).
2. To present a detailed evaluation methodology to investigate the quality of the predictions of any given trained model. We move beyond the naive regression visualizations and demonstrate how to decompose the error of the model into its bias and variance components, how to check for interactions among variables, and how to assess the credibility of its predictions. In doing so, we also infer the limits of credibility on each target on the given data set under our model.
3. To introduce a perturbation-based sensitivity analysis approach for visualizing regions of the input that are most relevant for the predictions of any given trained model. Doing so allows us to understand whether the regions of the input to which the model is most sensitive align with our physical intuition.

2.2. Data Generation

For the purposes of this study, we generated synthetic[3] planetary atmospheres from planets contained in the Ariel Target list (Edwards et al. 2019a). A total of 11,940 transmission spectra were produced. A transmission spectrum records the λ dependency change in transit depth (Δt_λ). This large-scale spectrum generation is made possible through the function *Alfnoor-forward* in Alfnoor (Changeat et al. 2020a), a

[1] Multilayer Perceptrons (MLPs) are fully (i.e., "densely") connected feed-forward neural networks. They are the oldest type of DNN developed and the most commonly used in structured data.

[2] https://github.com/ucl-exoplanets/Spectra_Sensitivity_analysis

[3] Although the predictive performance and sensitivity analysis results of the specific models trained on this data set are problem-specific, it is important to clarify that all methodologies presented in this work are applicable to any ML model trained on a given data set.

pipeline consisting of TauREx3 and ArielRad, the Ariel Radiometric Model (Mugnai et al. 2020). Each generated spectrum is binned to Ariel Tier-2 resolution with error bars calculated based on Deep survey requirements and realistic estimates of the instrument, observations taken, and planet observed. The Ariel Tier-2 resolution is kept the same throughout the investigation; any binning process is not performed in wavelength space. This setup was used by Changeat et al. (2020a) in their investigation, which provided a benchmark for us to compare in Section 3.4. Here, we denote the binned spectrum as the mean, ground-truth spectrum $\bar{\boldsymbol{X}} = [\bar{x}_1, \bar{x}_2, ..., \bar{x}_{52}]$, where x_i represents the transit depth at the ith wavelength bin in ascending order. The associated uncertainty for each x_i will be denoted as σ_i.

All generated spectra are subject to the same assumptions: the atmosphere for each spectrum is assumed to have a constant He/H_2 ratio of 0.17, a hypothesis corresponding to a primary atmosphere with solar composition. Rayleigh scattering and Collision Induced Absorption for H_2-H_2 and H_2-He are included. The T-P profile is assumed to be isothermal, and trace gases are introduced to the atmosphere with isoabundance profiles (profiles constant with altitude). Other planetary parameters necessary to producing a transmission spectrum and estimating the observational uncertainties (spectrum and error bars), such as stellar radius (R_s), planet radius (R_p), planet mass (M_p), planet temperature (T_p), and other orbital parameters (semimajor axis, distance to the star, eccentricity) are taken from the predictions in Edwards et al. (2019a).

To generate an unbiased sample of spectra, we added a number of trace gases. For each of the constituent trace gases (H_2O, CH_4, CO, CO_2, and NH_3), we uniformly sampled their log abundance from -9 to -3[4]. The line lists of different molecules are taken from ExoMol (Tennyson et al. 2016), HITRAN (Gordon et al. 2016), and HITEMP (Rothman & Gordon 2014). Additionally, we have also added gray clouds with a cloud deck pressure $\log(P_{\rm cloud})$ uniformly sampled from 2.7 to 6. Table 1 summarizes the sampling range, sampling method, and their respective scales. For a detailed discussion of the data-generation process, we refer the interested readers to Section 2.2 of Changeat et al. (2020a).

2.3. Data Preprocessing

The term "parameter" is defined differently under the context of ML and exoplanet atmospheric retrievals. To explicitly distinguish the different contexts, we will refer to Atmospheric Model Parameters as "AMPs" and Deep Neural Network parameters (synaptic weights) as "weights" hereafter.

The synthetic spectra and their corresponding AMPs are standardized (normalized so that each feature, i.e., wavelength bin and each AMP has zero mean and unit variance) to facilitate the training of the DNN models (see Figure 1 for empirical comparison of sample spectra before and after standardization). The standardized data set is then split uniformly at random into three subsets: the original training set (70%), the validation set (10%), and the test set (20%).

The original training set is not directly used in training. Rather, it is used to generate an augmented training set. For each data point (spectrum) $\bar{\boldsymbol{X}}$ in the original training set, we generate

[4] The range was chosen to explore Ariel Deep survey's ability at capturing low molecular abundances. Log abundance values > -3 are omitted as they can easily be detected via current retrieval methods.

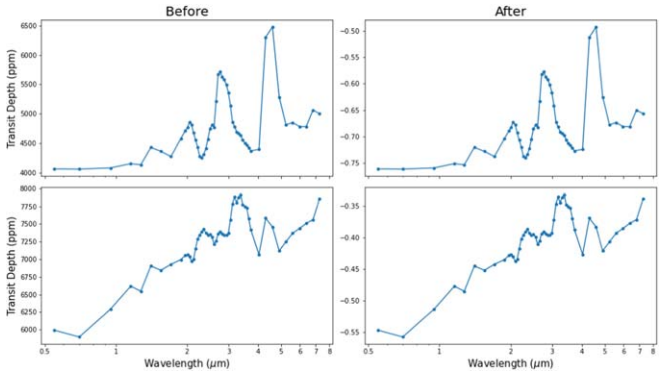

Figure 1. Empirical comparison of a sample of synthetic spectra before and after standardization. The comparison shows that our transformation only scales the spectral features, without distorting their relative shape.

Table 1
Sampling Range, Scale and Sampling Method Used for Different AMPs in the Synthetic Data Set

AMPs	Range	Scale	Sampling
H_2O	-9 to -3	log	Uniform
CH_4	-9 to -3	log	Uniform
CO	-9 to -3	log	Uniform
CO_2	-9 to -3	log	Uniform
NH_3	-9 to -3	log	Uniform
M_p ($\log(M_J)$)	-3.00 to 1.43	log	Edwards et al. (2019a)
R_p (R_J)	0.07 to 2.39	linear	Edwards et al. (2019a)
T_p (K)	1393 to 3999	linear	Edwards et al. (2019a)
Cloud ($\log$(Pa))	2.7 to 6	log	Uniform

50 data points $\tilde{\mathcal{X}}$ for the augmented training set. Each $\tilde{\mathcal{X}}$ is produced by sampling a new $\tilde{x}_i$ from a Gaussian distribution centered at $\bar{x}_i$ and having a standard deviation defined by the corresponding σ_i. The original ground-truth (i.e., noise-free) spectra are thus discarded and the models are trained only on these noisy, more realistic samples. The same applies for the validation set, whereas the test set is kept noise-free.

2.4. Model Training

We trained a DNN to perform a multi-output regression task. The task is to predict nine targets, $\log(X_{H_2O})$, $\log(X_{CH_4})$, $\log(X_{CO})$, $\log(X_{CO_2})$, $\log(X_{NH_3})$, R_p, $\log(M_p)$, T_p, and $\log(P_{\rm cloud})$, from a given spectrum. For ease of referencing, we denote the AMPs as $\boldsymbol{y} = [y_1, y_2, ...y_9]$, where y_j represents the jth AMP (as ordered above). The model is trained in a supervised manner by minimizing the Mean Squared Error (MSE) between the predicted values $\hat{\boldsymbol{y}}$ and the ground-truth $\boldsymbol{y}$, averaged across all targets. Details of how the neural networks were trained can be found in Appendix A. The results and figures shown in this paper are selected from our best-performing model, a one-dimensional Convolutional Neural Network (1D-CNN).

3. Evaluation of Predictive Models

3.1. Prediction versus Truth Plot

In Figure 2, we compare for each of the individual AMPs to be retrieved, the value predicted by the model (y-axis) against the true value (x-axis). These predictions were generated for noise-free spectra from the test set. Each blue point in every subplot represents a prediction from a single (test) spectrum.

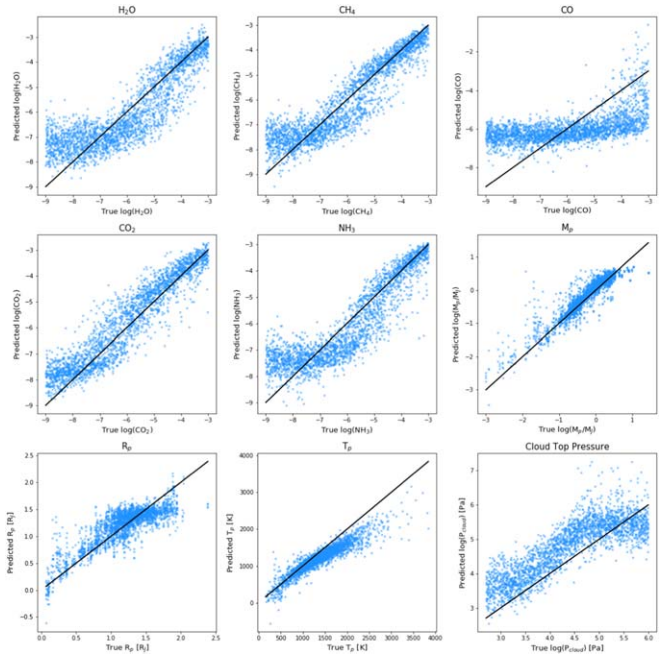

Figure 2. Prediction versus Truth Plot for each AMP in their respective units. Each blue point represents a single spectrum from the test set. Prediction from the model (*y*-axis) is plotted against the corresponding ground truth (*x*-axis). The black diagonal line represents the distribution of perfect predictions.

The diagonal line represents the predictions of a perfect model (one that always predicts the ground truth). This is a classic visualization of regression results. It is useful for obtaining an overall sense of the model's performance: ideally "points should not deviate much from the diagonal." But significant information is obscured by the fact that (i) the density of points is not uniform and that (ii) "deviation" can assume different mathematical meanings (e.g., "average deviation" or "standard deviation around the mean").

3.2. Bias and Variance Visualization

To get a deeper understanding of the model's performance, we need to go beyond Figure 2. The error (in our case, the MSE) of a regression model can be decomposed into three terms: bias, variance, and irreducible noise (Murphy 2012). Here, we state explicitly our definition of bias and variance, to avoid any confusion with the terminologies. Bias refers to the mean absolute deviation of the model's predictions from the true value. Variance refers to the spread/width of the model's prediction. Irreducible Noise (third term) refers to variance inherent to the data.[5] We note here that the variance we computed for each wavelength bin in the following figures will inherently contain variance from both the model's prediction and the irreducible noise.

An approximate visualization of the bias and variance components of the error for each AMP as shown in Figure 3 [6] can be more illuminating. Each subplot is generated by equal frequency binning of the true values (each bin contains 100 data points). For each bin, we calculate the mean absolute difference between pairs of predicted and true values $|\hat{y}_j - y_j|$ (a proxy of the model's bias) and the standard deviation of these differences (a proxy of its variance and variance from irreducible noise). Ideally, one would like to keep both components of the error low, i.e., to consistently predict values close to the truth. Using Figure 3, we can inspect regions where the model's predictions suffer from high bias (deviation), high variance (spread), or both.

In our application on the simulated Ariel-like data set, the model's predictions on the gases exhibit a similar trend: the prediction starts off with small bias and variance at high abundances, and both the bias and the variance gradually become higher as the abundance drops, reaching a peak at certain abundance. At lower abundances, below the credibility limit (see Section 3.4) of the corresponding gas, the network resorts to–on average–outputting an "average low" value. This results in a characteristic trough (its minimum indicating the "average low" value for each gas). This behavior is expected. A molecule's absorption feature is most prominent at high abundances, and this helps to tightly constrain the model's predictions. However, as gas abundance decreases, so does the magnitude of the corresponding feature, making it easier for other absorption features to partially, or–in some cases–even completely mask it. The task therefore becomes progressively harder, which contributes to a higher variance and a significantly biased mean deviation. If the level of abundance becomes low enough, the model can no longer constrain the prediction, due to presence of features from other molecules. At this point, the best strategy for a loss-minimizing model is to restrict its output and output a limited range of values centered at an average value in the low-abundance region (see discussion in Section 3.4).

The above trend is generally followed by most gases except CO, which has most of its predictions clustered around $\log(X) = -6$ exhibiting large bias. The poor quality of the predictions for CO is expected, given the spectral coverage of the instrument and lack of broadband features from the molecule itself. The lack of information on CO causes the model to minimize the loss by always predicting a restricted range of values with an average volume mixing ratio of $\log(X) = -6$ (see Figure 6).

On the other hand, for planetary parameters such as M_p, R_p, and T_p, the performance of the model varies. The model's M_p predictions are generally characterized by low bias. The ability of the model to accurately predict M_p suggests that the Ariel Tier-2 spectra alone contain sufficient information to constrain M_p, confirming the findings in Changeat et al. (2020b). The model's performance on R_p and T_p, however, is not as satisfactory. While the model is able to accurately predict smaller planetary radii, its predictions on the very largest radii in the sample are characterized by high bias. In particular, the radius is consistently underestimated. For T_p, the model becomes progressively more biased in hotter temperatures starting from $T_p = 1500$ K, again consistently underestimating them (see Figure 2).

Regarding the model's prediction of $\log(P_{\rm cloud})$, it appears that most of the predictions have been stratified into two levels, a sign that the model is only able tell qualitatively whether the atmosphere is cloudy or not.

The poor performance in R_p, T_p, and $\log(P_{\rm cloud})$ can be explained by several factors. Most notably:

[5] The noise component is due to the inherent uncertainty in predicting the targets (AMPs) from the data (spectra), even with a "perfect" model. As such, it is irreducible.

[6] The interested reader can find the same type of plots generated by the other DNN models examined in Figures 10 and 11.

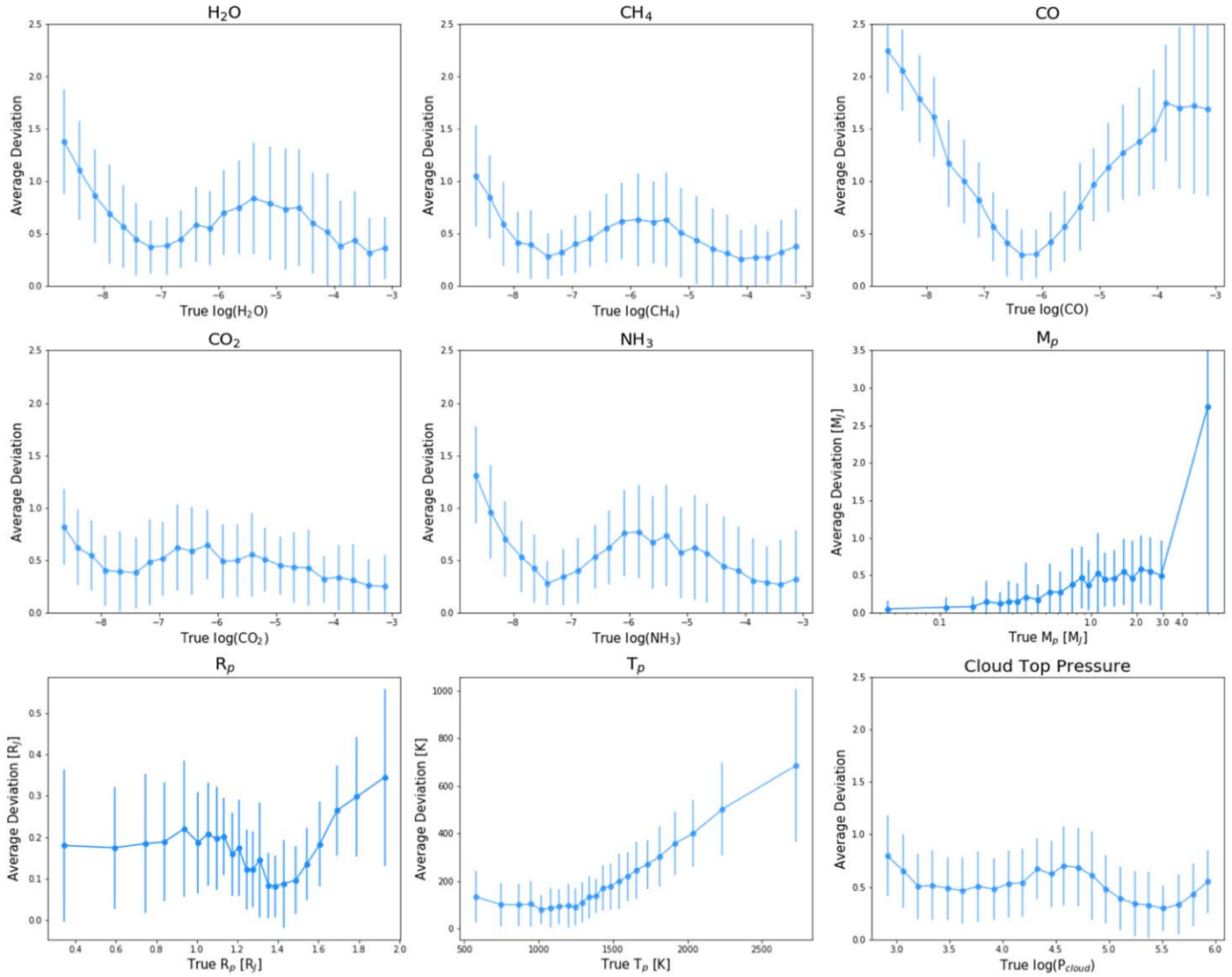

Figure 3. Visualization of bias and variance for different AMPs. Each point represents the average absolute deviation of the model's prediction from the ground truth (a measure of the model's bias), and the associated error bar represents the 1σ spread of the predictions (a measure of the model's variance). Note that the model's variance also includes contributions from the irreducible noise.

1. The degeneracy involving these quantities. It is well-known that the interaction between these quantities could produce very similar spectral features (e.g., Brown 2001; Fortney 2005; Lecavelier des Etangs et al. 2008; de Wit & Seager 2013; Griffith 2014; Rocchetto et al. 2016; Fisher & Heng 2018; Tinetti et al. 2018; Changeat et al. 2020b). For example, the model tends to underestimate T_p and R_p and overestimate $\log(P_{\rm cloud})$ (i.e., predict a less cloudy atmosphere). These AMPs are degenerated, i.e., multiple combinations of AMPs exist for (almost) the same spectra. As there is more than one possible solution, our model (being a deterministic function outputting a single prediction for each AMP) fails to always identify the "'true" (i.e., in the data generation sense) solution given the limited information from the data. Ideally, in an atmospheric retrieval setting, rather than predicting the most probable values of the AMPs, we would rather predict their posterior distribution or at least capture their covariance. In Section 3.3, we perform an initial investigation of interactions among AMPs.
2. The standardizing step in Section 2.3 was performed by extracting the overall mean and standard deviation of the training set; the order-of-magnitude differences between spectra may have significantly reduced the dynamic range within a spectrum, dwarfing any molecular signatures.
3. The nonuniform distribution of R_p, T_p, and M_p in the generated data. The nonuniformity means that the model will focus on more accurately predicting values in the densely populated areas of the target space to the potential detriment of the quality of its predictions elsewhere, if that means achieving a lower MSE.[7]

Our example above has investigated the average deviation and spread of the prediction at each prediction level. This quantification and visualization of bias and variance can be further utilized to help us determine the optimal model

[7] The nonuniform distribution could be alleviated with better knowledge on permitted combinations of R_p, T_p, and M_p, an alternative way is to adjust the weight of each sample based on its rarity (heavier loss on uncommon examples).

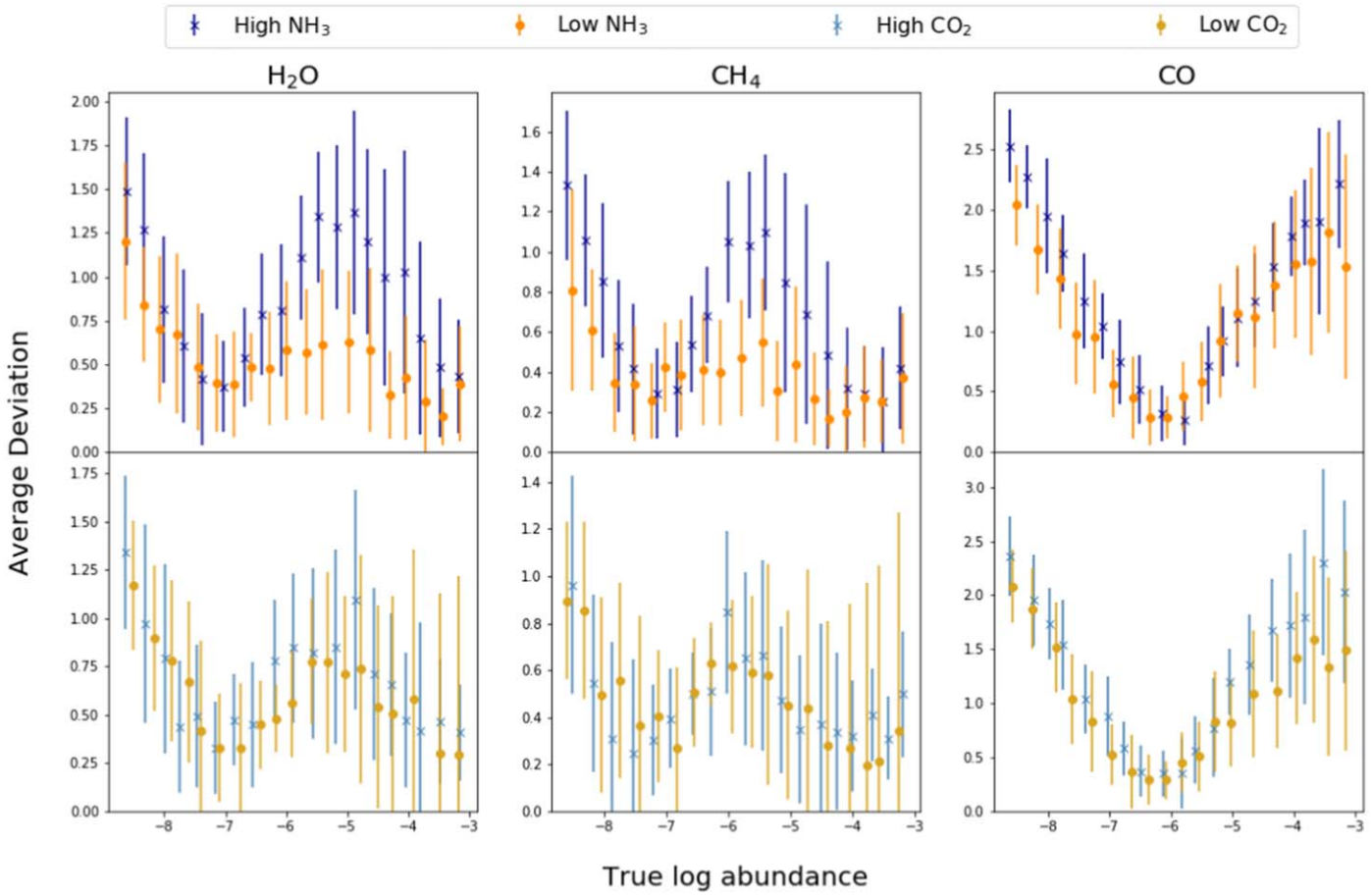

Figure 4. Visualization of bias and variance for H_2O, CH_4, and CO at high and low NH_3 (top) and CO2 (bottom) (log-) abundance. H_2O and CH_4 prediction improved at low NH_3 abundance level. Please note that the model's variance also includes contributions from the irreducible noise.

complexity for a given predictive task (see Appendix B for a detailed discussion).

3.3. Interactions between AMPs

Next, we inspect whether the model aligns with our physical intuition on the problem. More specifically, we ask "Does it perform worse when we expect it to?" One way to investigate this is to measure how its predictions on the jth AMP $\hat{y}_j$ vary conditioned on the true value of another AMP y_k, $k \neq j$ having a "low" or a "high" value. For the purposes of this visualization, we focused on gases and defined any log abundance in the lowest quartile (i.e., the lowest 25% of the values) of the population as "low" and any log abundance in the highest quartile (i.e., the highest 25% of the values) as "high".

Figure 4 shows how the (binned) predictions of H_2O, CH_4, and CO change under high (> -4) and low (< -7) abundances of NH_3 and CO_2. The binning procedure is similar to the one described Section 3.2, but the bin size is reduced to 30 samples.

We can observe that a high or low abundance of NH_3 gives rise to a distinctive contrast in the quality of predictions for most molecules. In particular, the quality of the predictions of CH_4 is highly affected by the abundance level of NH_3. This observation aligns with our expectations. As ammonia's absorption feature spans from 2–4 μm, it can partially or fully cover any other absorption features within that range at high abundances, reducing the model's ability to accurately predict the abundance of molecules such as CH_4, and vice versa. The same issue, however, should not arise for H_2O, as the molecule possesses several broadband features outside 2–4 μm, i.e., a well-trained network should be able to rely on information available outside this range to predict water abundance, which means there should not be a dramatic improvement when NH_3 is low. This somewhat unexpected improvement in performance hints at the mechanism behind the model's prediction; this mechanism is further discussed in Section 4.2.

On the other hand, the model's performance on CH_4 does not change as much under different levels of CO_2. This is also an expected outcome, as CO_2's absorption feature lies in 5–6 μm, thus distinct features in CH_4 are less likely to be masked by changes in CO_2's features (Sharp & Burrows 2007).

It is possible to construct similar plots for other AMPs beyond NH_3 and CO_2. The purpose of this work is to demonstrate general evaluation tools that elucidate the inner workings of ML models. To avoid overly emphasizing our analysis on our particular (data set, model) combination, we shall forgo an exhaustive discussion on all combinations of AMPs interactions.[8]

Another well-known way to visualize the covariance between different AMPs is to visualize the learned posterior distribution. We sample the parameter space of the model by varying the same input spectrum according to its uncertainty.

[8] Interested readers can inspect the full results in Figures 12–20; the results shown here are produced using our 1D-CNN model.

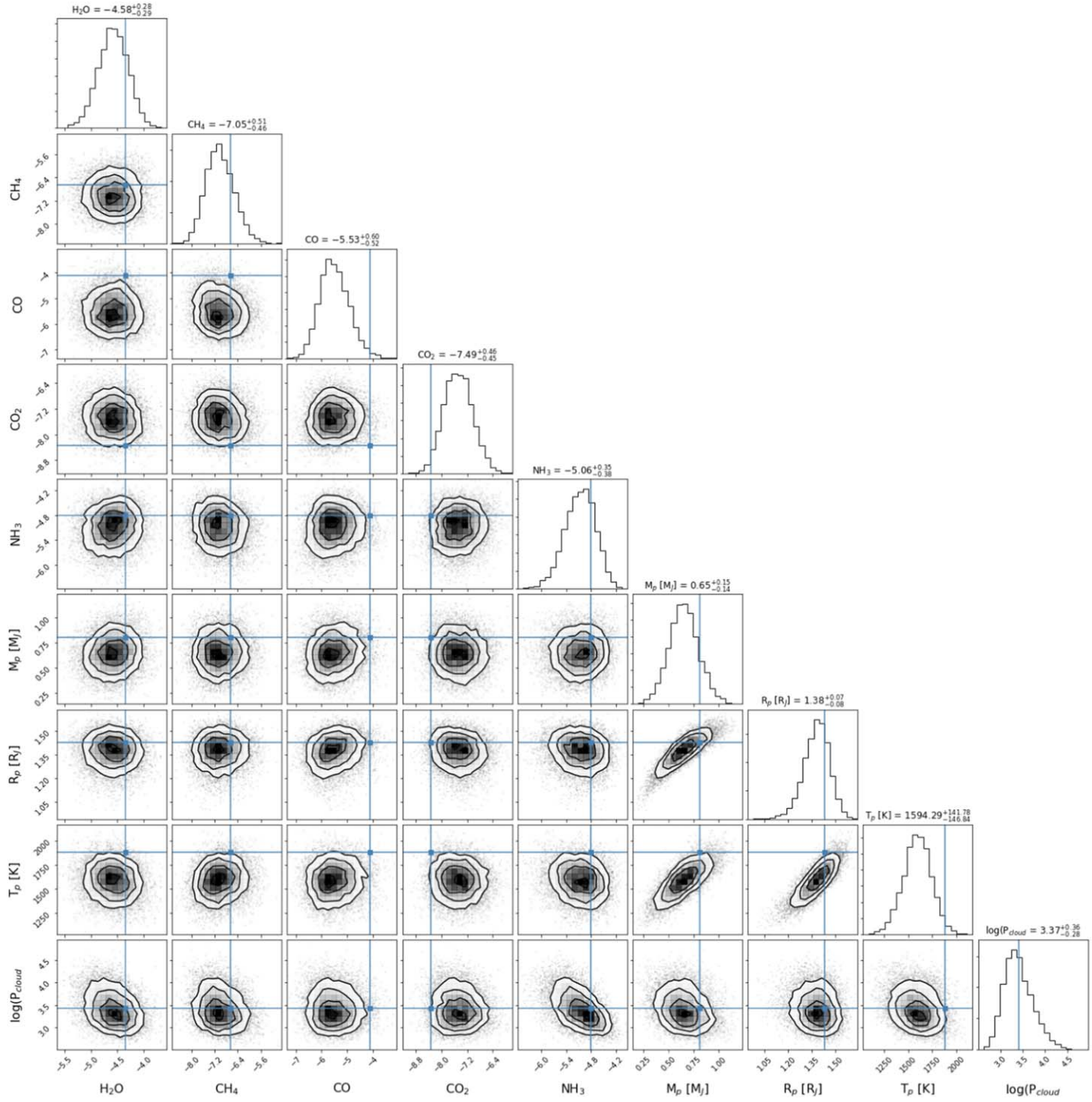

Figure 5. Posterior distribution from a test set spectrum. The ground truth is indicated by the blue line. The model tends to perform well on abundant trace gases with the exception of CO. It is also able to capture some correlation between AMPs.

Figure 5 shows an example posterior distribution produced using a spectrum from the test set. The model managed to predict within 2σ of the ground truth values (indicated by the blue line) when the log abundances of the gases are higher than -6, which is expected from our analysis in previous sections. We can see that the model is able to capture some of the correlations such as M_p versus R_p, as well as R_p versus T_p, which are among some of the worst-performing AMPs. On the other hand, the model failed to capture other well-known correlations such as H_2O versus R_p and H_2O versus clouds. This analysis is thus highlighting a shortcoming of this model.

Upon discovering ways in which a model's behavior is poor (either in terms of predictive performance, or in terms of capturing aspects of the underlying physics) we can take further measures to improve it. In our analysis above, we noticed that certain known correlations among the targets (AMPs) were not captured by the model. There are ways to explicitly introduce domain knowledge like this into the architecture of the neural network, e.g., by sharing parameters across targets as discussed in Reyes & Ventura (2019). As this work is focused on analyzing models and diagnosing problems, applying such methods to this setting is reserved for future work.

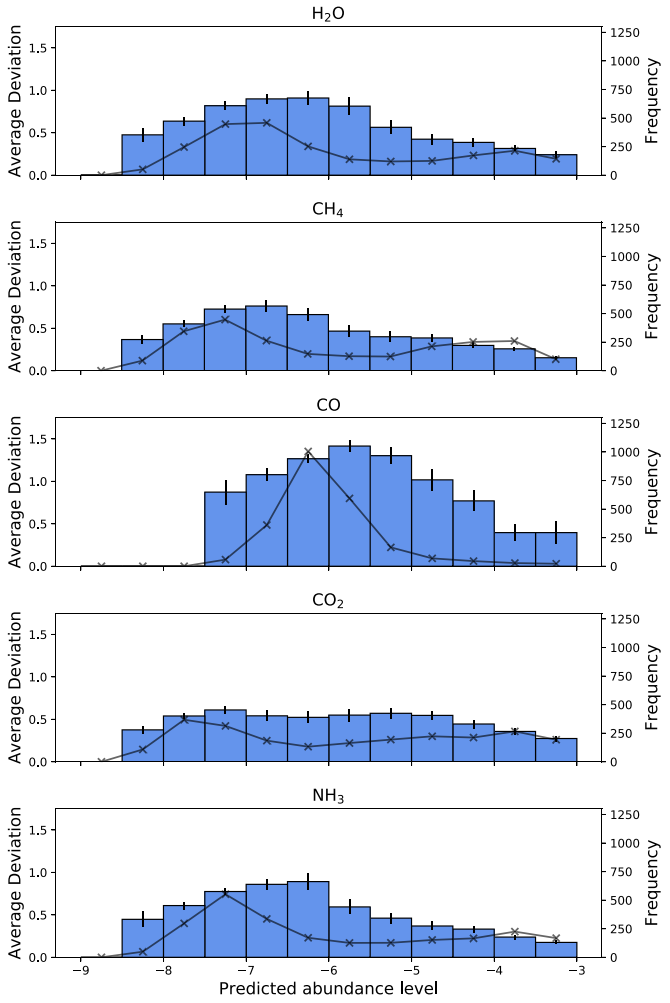

Figure 6. Average deviation for different molecules at different predicted abundance. The error bars on each bar represent 95% confidence intervals around the mean. The black curve represents the total number of predictions made by the model (frequency) within each bin.

3.4. Credibility of Predictions

So far, we have looked at how the quality of a model's predictions varies across different ground truth values. During the training phase, this is useful for identifying problems with our models: targets whose predictions are problematic, areas of particularly high bias or variance, and interactions among AMPs suggestive of degeneracies, among others.

However, in real life, we rarely have access to the ground truth, thus a more practical question would be: "What is the expected deviation of a given prediction from the ground truth?" In other words, we would like to know how credible a given prediction by our model is.

Our analysis in Sections 3.2 and 3.3 has provided us with some qualitative intuition regarding the credibility of the model's predictions. For example, the model's prediction is more reliable at higher molecular abundances and when there is less interference from other molecules. To obtain a more quantitative measure, we follow an approach similar to that in Section 3.2 and compute the average deviation at different prediction levels for each AMP (Figure 6). The error bars on each bin represent 95% confidence intervals. Instead of binning with equal frequency like Section 3.2, we performed equal-width binning, and thus each bin will have a different number of data points. The black line shows the number of data points per bin. Bins with fewer than 20 data points are omitted.

The distribution of average deviations aligns with our discussion in Section 3.2. For high predicted abundances, the model starts off with low average deviation, and as the predicted abundance level goes down, the model struggles to predict well and begins to have higher average deviation. However, counter to our intuition, the average deviations do not increase monotonically; instead, they begin to decrease after a certain abundance level. This peak corresponds to the trough we saw in the figures of Section 3.2. This provides us with clues regarding the model's loss-minimization strategy. The model is restricting its output to a limited range of values at low abundance levels, centered around some average value. This can be evidenced by the distribution of counts (black line) being centered at some value in the low-abundance region and few or zero counts at the lowest abundance level ($\log(X_{\rm gas}) = -9$).

Another important insight that can be drawn from Figure 6 is that the trustworthiness of the prediction varies across abundance levels. We propose a method to qualitatively assess the credibility limit of each gas—the limit at which predictions remain meaningful to the model's user. First, we compute the probability P that the model's prediction ($\hat{y}_j$) does not deviate more than a positive real value ϵ from the ground truth y_j. We then require that P be at least $1-\delta$ to consider the prediction credible. A detailed discussion on the method is included in Appendix C. In Figure 7, we demonstrate an example where we have chosen $\epsilon = 0.5$, and defined a credibility threshold $\delta = 0.3$, so that any prediction level with probability $P \geqslant 1-\delta$ is credible. We can then define the lowest predicted abundance level that satisfies this as the credibility limit of that gas.

This limit is specific to the chosen δ and ϵ, as well as the trained model. Table 2 summarizes the estimated limit for each molecule. Although this approach is useful, we should still be mindful of its limitations. Note that, for several gases, the probability that the prediction error will not deviate from the acceptable region seems to increase at the lower end of the log abundance. However, this increase does not necessarily imply a higher predictive power at low abundances. In the aforementioned cases, this apparent increase in predictive power can be most likely attributed to the small number of instances that fall within these bins, as evidenced by Figure 6. It is thus a small-sample effect.

Similar work by Changeat et al. (2020a) determined the detection limit from a retrieval perspective. In their work, they have generated 164 planets using the same setup as the one presented in Section 2.2 and determined the lowest abundance at which they can constrain the molecular abundance within 1 order of magnitude of error. Our limit here addresses the trustworthiness of the neural network, which cannot be directly compared with results from retrievals. Despite the differences, both studies suggest that Tier-2 Ariel spectra are capable of allowing for the consistent detection of some molecules in abundances as low as $\log(X) = -5.8$. While it is possible that, given a different architecture, the credibility limit could be improved, we would like to reiterate that our goal is not to compete against retrieval frameworks, and thus we will leave this for future work.

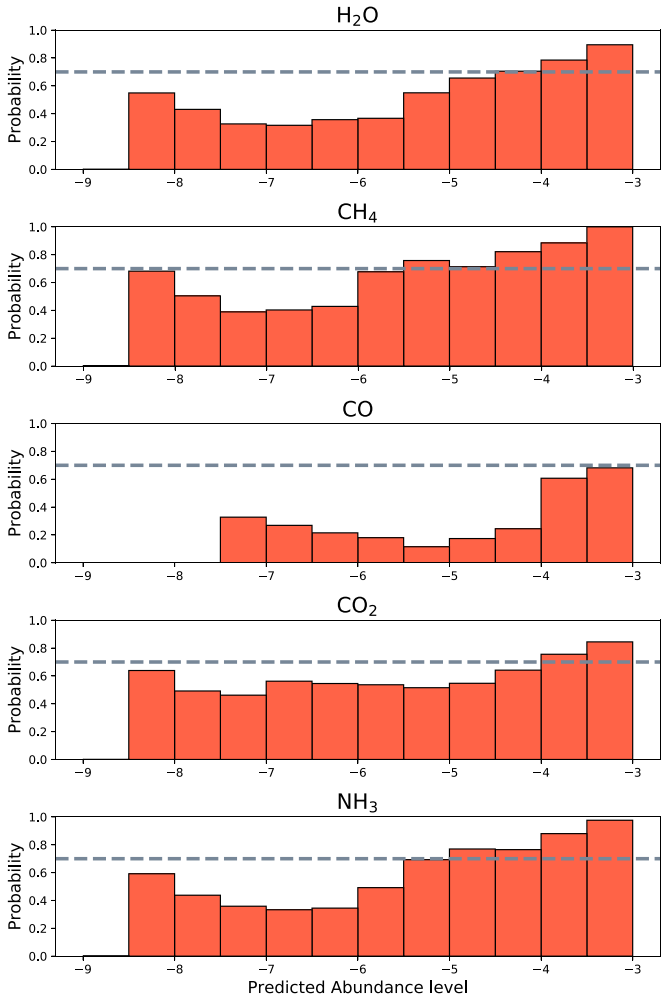

Figure 7. Probability of finding a prediction within $\epsilon = 0.5$ to the ground truth for each molecule. The gray dotted line represents the probability threshold at 0.7 $(1 - \delta)$. Any bin with $P > 0.7$ is considered credible in this case.

Table 2
Credibility Limit for Each Molecule at $\delta = 0.3$, $\epsilon = 0.5$

Molecules	Credibility Limit ($\log_{10}$)	Detection Limit
H_2O	−4.3	−6.5
CH_4	−5.8	−7
CO	N/A	−5.5
CO_2	−3.8	−7
NH_3	−5.3	−6.5

Note. This limit is derived using the lowest credible (log-) abundance level, following the credibility definition in Appendix C. The detection limit is reproduced from Changeat et al. (2020a) as a comparison to retrieval methods.

Interested readers could refer to Figure 21 for the distribution of average deviation of all AMPs. Credibility limits for nongaseous AMPs are less straightforward, and are more influenced by the training set. As the training set is not uniformly distributed w.r.t. these AMPs, trained models will have a tendency to focus on the regions containing a higher density of examples, biasing the predictions. Thus, the derived limit first and foremost would depend on biases of the training set, and for this reason we chose to omit it here to avoid overinterpretation.

4. Sensitivity Analysis for Model Interpretation

4.1. Method

Given any trained predictive model $\mathcal{M}$ (e.g., a neural network) that takes an input $\boldsymbol{x}$ and outputs the corresponding prediction $\hat{\boldsymbol{y}}$, a perturbation-based sensitivity test can be performed to assess the change in the prediction $\hat{y}$ when a set of features (transit depth, in our case) x_i (consecutive or not) is perturbed.

This approach assesses quantitatively how $\hat{\boldsymbol{y}}$ varies as a set of x_i (transit depths) vary. The intuition is that perturbations in the regions of the input containing more information about the target will yield larger deviations in the output of the model.

Below, we outline a general procedure for such a sensitivity analysis:

1. Produce a reference prediction $\hat{\boldsymbol{y}}_r$ on an unperturbed input $\boldsymbol{x}_r$.
2. Perturb the input $\boldsymbol{x}_p$.
3. Predict $\hat{\boldsymbol{y}}_p$ on the perturbed input $\boldsymbol{x}_p$.
4. Compare $\hat{\boldsymbol{y}}_p$ and $\hat{\boldsymbol{y}}_r$.
5. Repeat step 2–4 for different sets of features.

The form of perturbation depends on the context of the problem. Zeiler & Fergus (2013) demonstrated the idea on models performing image classification. They perturbed the input image by setting a region to zero pixel value, and produced a heat map of sensitivity by covering each region systematically. In this investigation, we adapted this procedure to our multi-target regression problem. Instead of setting x_i to zero like Zeiler & Fergus (2013), we applied the perturbation by sampling each wavelength bin x_i according to its respective error bars (i.e., from a Gaussian centered at its unperturbed value and with standard deviation σ_i). There are three main reasons for this choice: 1. *Physical plausibility.* Setting a window of the spectrum to zero would render it nonphysical, as a transit depth of 0 would mean $R_p = 0.2$. *Statistical plausibility.* Neural networks excel at interpolation but not extrapolation. Perturbing the input spectrum within its error bars would still result in valid input (i.e., a sample from the actual data distribution) for the Neural Network. 3. *Instrument plausibility.* The result of the test under these conditions provides realistic measurement of the relative sensitivity of each wavelength bin for the purposes of determining each of the parameters to be retrieved, in the context of Ariel Deep survey specifications (Tier-2 spectra). This also means any derived result will be specific to the instrument and observing strategy.

At each iteration, we select a random number of x_i and apply the perturbation by scattering these points according to $\mathcal{N}(x_i, \sigma_i^2)$. For computational efficiency, at each iteration the number of x_i is chosen from 27 (half of the total number of wavelength bins) down to 2 (parts of a feature). The intention is to account for the influence from both broad and narrow features, as well as the interdependencies between different wavelength bins. We repeat the above procedure 1000 times with 300 spectra randomly chosen from the test set and calculate the average mean squared difference per AMP (i.e., parameter to be retrieved) between $\hat{\boldsymbol{y}}_p$ and $\hat{\boldsymbol{y}}_r$ for each wavelength bin. The result is a sensitivity map of the model's output for each AMP, w.r.t. each feature. A detailed implementation of the test is discussed in Appendix D.

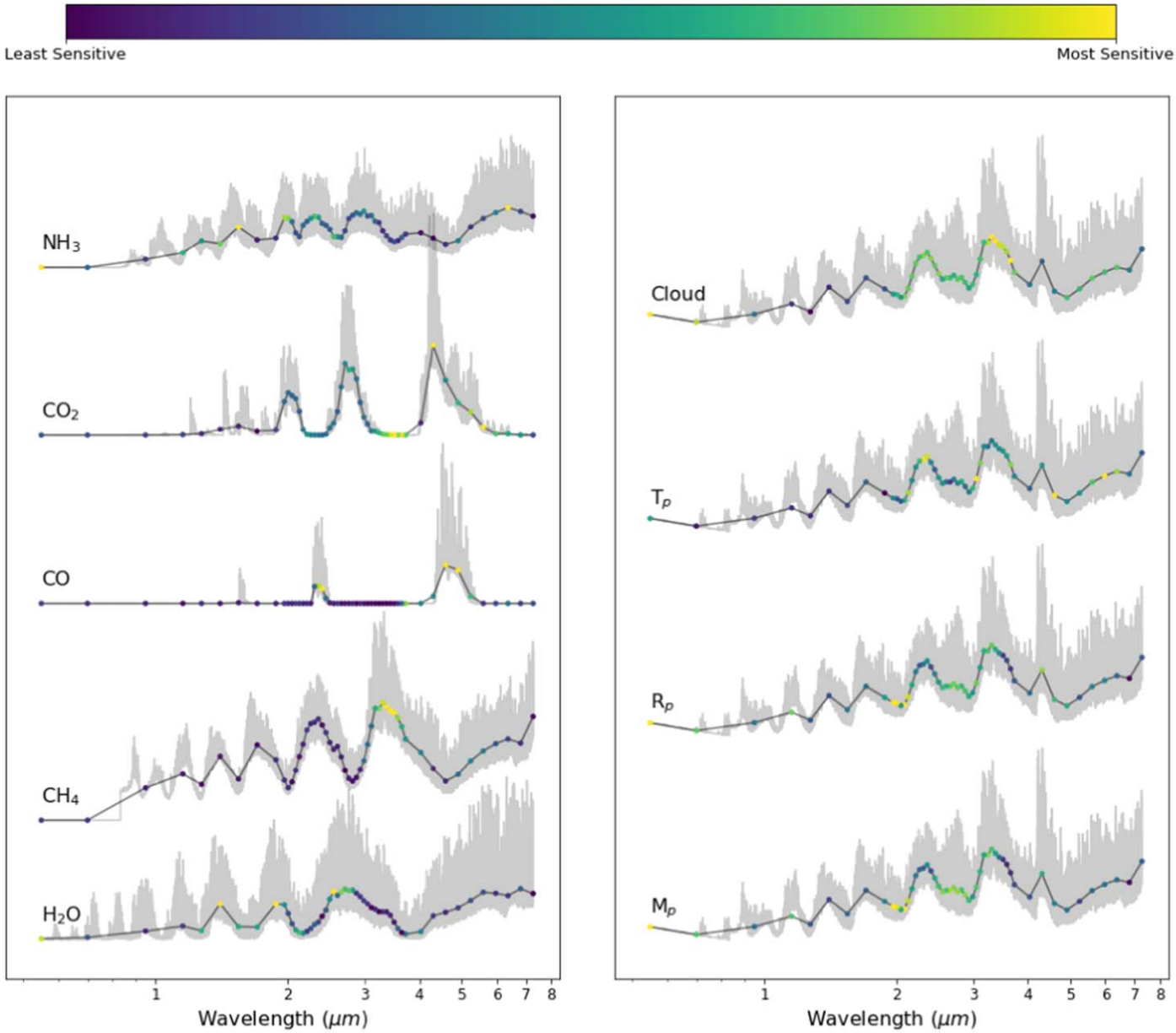

Figure 8. Sensitivity map for each AMP. Each wavelength bin is color-coded to reflect the relative sensitivity level among bins, with yellow being the most sensitive and black being the least. Each spectrum shows the corresponding molecule's characteristic absorption features in Ariel Tier-2 resolution.

The sensitivity map is a tool for us to visualize what factors drive the model's predictions. As such, it allows us to investigate whether the model aligns with our physical intuition. This can also shed light to potential biases of the model or the training data. Finally, it can even aid us in identifying potentially undiscovered relationships among features.

4.2. Sensitivity Map

We applied the general procedure outlined in Section 4.1 and produced sensitivity maps for the different atmospheric parameters of interest, using as the model $\mathcal{M}$ the 1D-CNN we trained in Section 2.4. We shall explicitly ignore regions where the model's prediction is uncertain, and restrict the sensitivity analysis to cases with $\log(X_{\rm gas}) > -5$.[9] We will first investigate whether the model's predictions align with our physical intuitions regarding inferring AMPs from spectra. Figure 8 summarizes the results of the sensitivity analysis for all the AMPs. The left subfigure summarizes the molecular species, and the right subfigure summarizes the planetary parameters and clouds.

4.2.1. Sensitivity Map for Active Molecules

Each spectrum on the left subfigure displays the corresponding molecule's characteristic absorption features in Ariel Tier-2 spectra, and each bin in the spectrum is color-coded to reflect the relative sensitivity of the model's prediction of the corresponding molecular abundance due to changes in the value of said bin. We normalized each spectrum according to its respective minimum and maximum.

We can see that many of the highlighted regions correspond to the major absorption features of the molecules. This alignment is evidence that the neural network is recognizing

[9] There are different limits for some gases, but for simplicity we chose a conservative value.

individual molecular features and basing its predictions of the corresponding molecular abundance on the peaks and troughs of these absorbing regions. Even in the case of CO, whose abundance the model generally fails to accurately predict, it nonetheless manages to highlight the absorption bands of the molecule as the most important region for predicting it.

So far, we see that the predictions of the neural network are based on factors that agree with our physical intuition. However, we can also see that, for some molecules, not all peaks are highlighted by the model, e.g., for H_2O only the peaks at 2–3 μm are highlighted. The model is tasked to jointly predict all quantities of interest. As a result, it is compromising performance across individual molecular species to identify the optimal features to predict them jointly.

Sensitivity maps like these are useful for improving the transparency—and thus, our confidence in the predictions of the model. On the other hand, they also give us an indication of where most of the information is coming from for the model in question. Here, we only present maps for $\log(X_{\rm gas}) > -5$. It is possible that, in the face of different combinations of abundances, the sensitivity map will change accordingly. As the purpose of this study is to explain the methodology, the discussion of sensitivity maps at different abundances will be left for future work.

4.2.2. *Sensitivity Map for* M_p, R_p, T_p, *and Clouds*

For AMPs other than gaseous species, their corresponding sensitivity maps are summarized on the right side of Figure 8. We used the same randomly sampled spectrum to investigate their sensitivity to each wavelength bin. Below, we provide our observations and offer an interpretation of these maps.

M_p, R_p, T_p, and clouds are interconnected via the computation of scale height, $H_{\rm sc} = \frac{k_b T R_p^2}{\mu G M_p}$. The photometric points at the blue end of the spectrum are "calibration" points, as they are the lowest points of the spectrum. These points help to provide an estimate for R_p, but are often masked in the presence of clouds. In the absence of M_p from external sources, the model examined here attempts to derive it from the spectrum, which again is correlated with R_p, and clouds, as described in Changeat et al. (2019), and can be visualized via the similarities between their respective sensitivity maps.

Temperature, on the other hand, is highlighted in three distinct regions: the photometric wave band, and the 3 μm and 5 μm regions. The model appears to be relying on the most probable highest features in the spectrum, combined with the photometric points, to derive the scale height via the features' size,[10] and subsequently the temperature as well. However, the aforementioned degeneracy between M_p, R_p, and clouds means that temperature is not accurately determined, as can be seen from Figure 3.

4.3. Choice of Network

We should keep in mind that sensitivity maps like the one shown in Figure 8 are model-specific, i.e., different models can have different sensitivity maps. The sensitivity analysis described here does not directly measure the information in the features that is relevant for predicting the AMPs, but rather, it captures the degree to which a given model uses said information to predict the AMPs.

[10] The size of the features is determined by a quasi-linear function of H_{sc}.

Figures 22 and 23 show the sensitivity maps for the networks of the other two DNN architectures we trained as outlined in Section 2.4. Interestingly, we find that all three models were able to highlight most of the peaks and troughs of the molecules' characteristic features. This is evidence[11] that these regions of the spectra are indeed important features for determining their corresponding molecular abundances.

There are also notable differences in the sensitivity maps of each model, in particular the maps obtained for nonmolecular AMPs. For example, the MLP tends to focus on 4–5 μm to derive quantities such as M_p, R_p, T_p, and cloud top pressure, while the LSTM and the 1D-CNN tend to also focus on 2–3 μm features. Despite any differences in sensitivity across different models, non-trace-gas AMPs exhibit high sensitivity to the same regions for a given model, highlighting the degeneracy between these AMPs.

5. Conclusion

In the context of exoplanet atmospheric retrievals using simulated data from Ariel, we investigated the use of three different types of DNN architectures (MLP, CNN, and LSTM) for inferring atmospheric model parameters from exoplanet spectra. We presented a suite of methodologies for analyzing the performance of any regression model, identifying its main source of error by leveraging the concepts of bias and variance, and quantifying the credibility of its predictions. Applying these evaluation methodologies to the three DNN models we trained, we found that they all behaved similarly for this data set, and that they are capable of reliably determining molecular abundances down to as low as $10^{-5.8}$.

We also introduced a perturbation-based sensitivity analysis that allows us to assess the relative importance of each feature (wavelength bin) in predicting each target (atmospheric parameter), for a given trained predictive model. Our analysis confirmed that the predictions of the DNN models we constructed largely align with our physical intuition with respect to each atmospheric parameter's spectral signature, our understanding of Ariel's instrument specification, and Ariel Deep survey observational strategy.

The evaluation and interpretability methods presented in this paper are applicable to any predictive model learned from data, and only require access to the model's predictions and the training data. These tools allow us to analyze the predictions of a model, identify potential biases in the model itself or the data, understand the factors driving the model's predictions, and investigate whether these agree with our current knowledge of the underlying physics, whether the model's predictions are "right for the wrong reasons," or whether it can provide us with new theoretical insights. Ultimately, they can make predictive models more transparent and thus easier to adopt by domain experts.

We appreciate suggestions from the anonymous reviewer, which have improved the quality of the manuscript. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 758892, ExoAI) and the European Union's Horizon 2020 COMPET

[11] Naturally, the class of models explored here is constrained to DNNs, so we cannot make any strong model-independent claims. Yet in this case, all three models agree with one another and with physical intuition.

program (grant agreement No. 776403, ExoplANETS A). Furthermore, we acknowledge funding by the Science and Technology Funding Council (STFC) grants: ST/K502406/1, ST/P000282/1, ST/P002153/1, and ST/S002634/1. We are grateful for the support of the NVIDIA Corporation through the NVIDIA GPU Grant program.

Software: ArielRad: (Mugnai et al. 2020), TauREx3 (Al-Refaie et al. 2021), h5py (Collette 2013), Matplotlib (Hunter 2007), Pandas (McKinney 2011), Numpy (Oliphant 2006), Keras (Chollet et al. 2015), Tensorflow (Abadi et al. 2015).

Appendix A
Implementation details

Table 3 summarizes the architecture details of the three types of neural networks we explored. The hyperparameters were selected after performing a grid search on the number hidden units per layer, the number of layers, and the number of filters. In all cases, the models were trained for 100 epochs with an initial learning rate of 0.01 and a learning rate decay of 10^{-4} using the Adam optimizer. Any unspecified hyperparameters were set to default Keras/Tensorflow values. All the networks were developed using the open source Keras (Version 2.3.1) Python module (Chollet et al. 2015), with Tensorflow (Version 2.4.1) as backend (Abadi et al. 2015).

Table 4 shows the average performance of each architecture across five runs with identical hyperparameter setup but different weight initialization, and under different training/validation splits. We also compare their complexity as measured by the number of weights to be learned. All three architectures yielded models with comparable predictive performances. However, we chose to present our main results using the CNN due to its lower complexity and subsequently faster training and inference time.

Table 3
The Three Different Neural Network Architectures Examined in This Study

Neural Network Architecture								
MLP			CNN			LSTM		
Layer Type	Config.	Output	Layer Type	Config.	Output	Layer Type	Config.	Output
Input		(m,52)	Input		(m,52,1)	Input		(m,52,1)
FC-RELU	$h = 320$	(m,320)	Conv-BN-RELU	$f = 32, 3 \times 3, s = 1$	(m,52,32)	LSTM	$h = 200$	(m,52,200)
FC-RELU	$h = 240$	(m,240)	Maxpooling	2×2	(m,26,32)	LSTM	$h = 200$	(m,200)
FC-RELU	$h = 160$	(m,160)	Conv-BN-RELU	$f = 64, 3 \times 3, s = 1$	(m,26,64)	FC-LeakyRELU	$h = 16$	(m,16)
FC-RELU	$h = 80$	(m,80)	Maxpooling	2×2	(m,13,64)	Dropout	$p = 0.3$	(m,16)
Dropout	$p = 0.3$	(m,80)	Conv-BN-RELU	$f = 96, 3 \times 3, s = 1$	(m,13,96)	FC-Linear	$h = 9$	(m,9)
FC-Linear	$h = 9$	(m,9)	Flatten		(m,1248)			
			FC-leakyRELU	$h = 128$	(m,128)			
			Dropout	$p = 0.3$	(m,128)			
			FC-Linear	$h = 9$	(m,9)			

Note. "BN," "FC," and "Conv" denote Batch Normalization, Fully Connected, and 1D Convolutional layer, respectively. With "*h*," "*f*," "*s*," and "*p*," we denote the hidden layer size, the filter size, the stride, and the dropout probability, respectively.

Table 4
Average Performance of the Three Different Architectures across Five Runs versus Model Complexity (as Measured by Number of Parameters)

Best Performance versus Model Complexity		
Architecture	Performance	# of weights
MLP	0.28 ± 0.01	248,889
1D-CNN	0.28 ± 0.01	186,665
LSTM	0.29 ± 0.01	325,769

Appendix B
Model Complexity

Quantifying the bias and variance of a model is also useful in determining whether the model is underfitting or overfitting the data. An underfitting model lacks the complexity (capacity) to learn the underlying pattern in the training set. This results in high error in the training set. It also results in poor performance on he test set. In this scenario, the prediction error is dominated by high bias. On the other hand, an overfitting model is "excessively complex" for the task at hand. This complexity can result in fitting not only the underlying pattern of interest but also the noise in the training data, leading to a good fit on the training set but poor generalization in the test set. In this scenario, the prediction error is dominated by high variance. The ideal model for the task is the one with just enough complexity to neither underfit nor overfit.

If we identify that the model is underfitting on a given task, then we should increase its complexity. In the case of DNNs, this can be achieved by, e.g., increasing the number of layers or the number of hidden units per layer. If we detect that a model is overfitting, one solution is to decrease the complexity of the model. We can either draw models from a richer model family (e.g., in the case of DNNs, choose an architecture with more hidden layers and/or hidden units per layer), or we can introduce some form of regularization (e.g., L_1-regularization, L_2-regularization, batch normalization, or dropout). Alternatively, we can use an ensemble of several predictors (e.g., combine the predictions of multiple DNNs). Finally, if such a thing is possible, we can increase the amount of training data.

The example below demonstrates the effect of increasing the amount of training data available to a model that underfits the data (insufficient complexity, high bias) and a model that overfits the data (excessive complexity, high variance). We first train two models: a "simple" and a "complex" one. The "simple" model consists of two CNN layers with eight filters each, and the "complex" model is composed of three CNN layers with 128 filters each. Both are trained using 1000 training data points, repeating the data generation and training 20 times. The top part of Figure 9 shows the average deviation of the predictions of the two models for the different AMPs (a

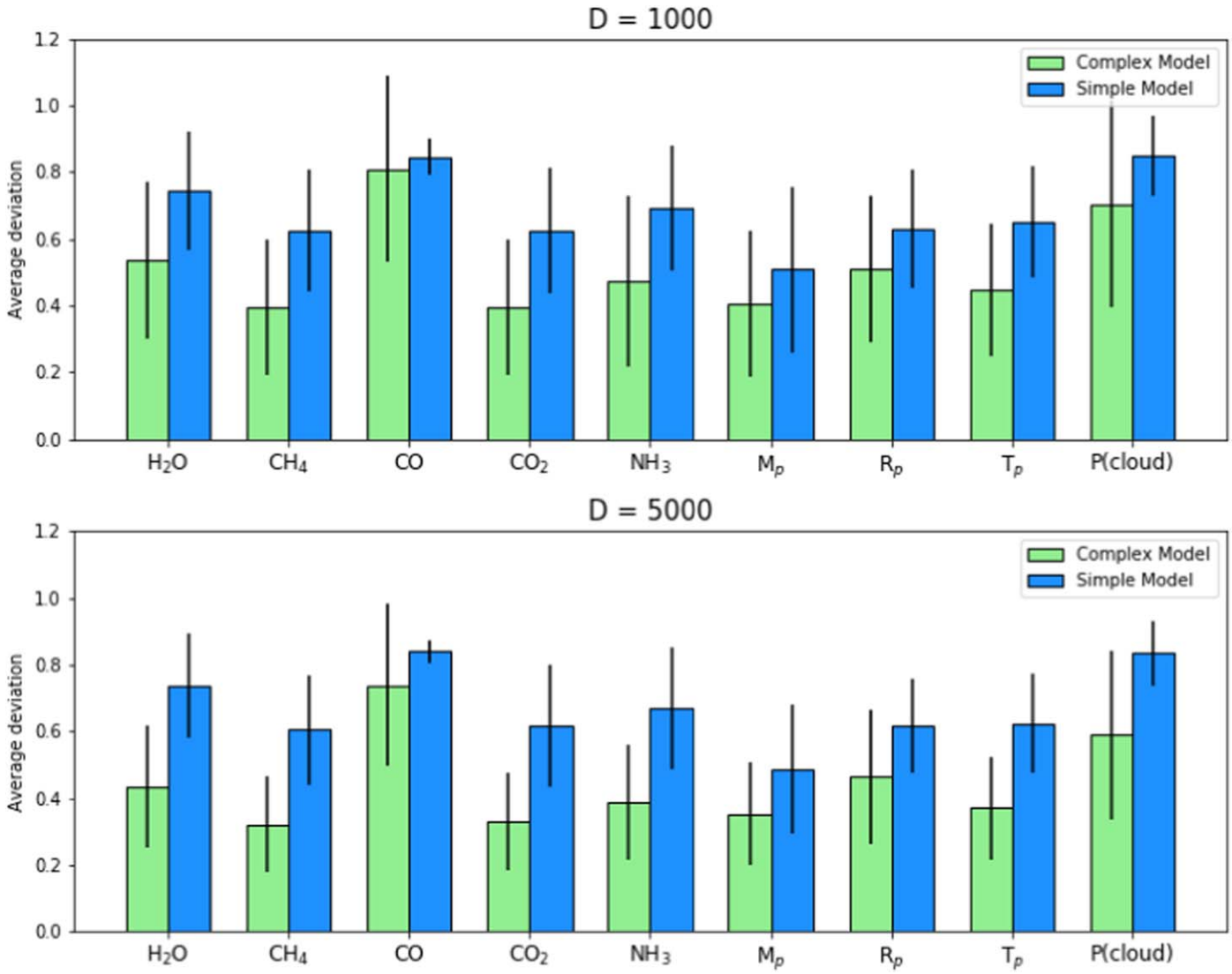

Figure 9. The performance of a "simple" and a "complex" model, on each of the AMPs, under different amounts of data points. The predictions of the "simple" model exhibit a higher average deviation from the true value (higher bias), while those of the "complex" model exhibit a higher variance. Training the two models on a larger training set allows the "complex" model to reduce its variance, but only marginally improves the predictions of the "simple model."

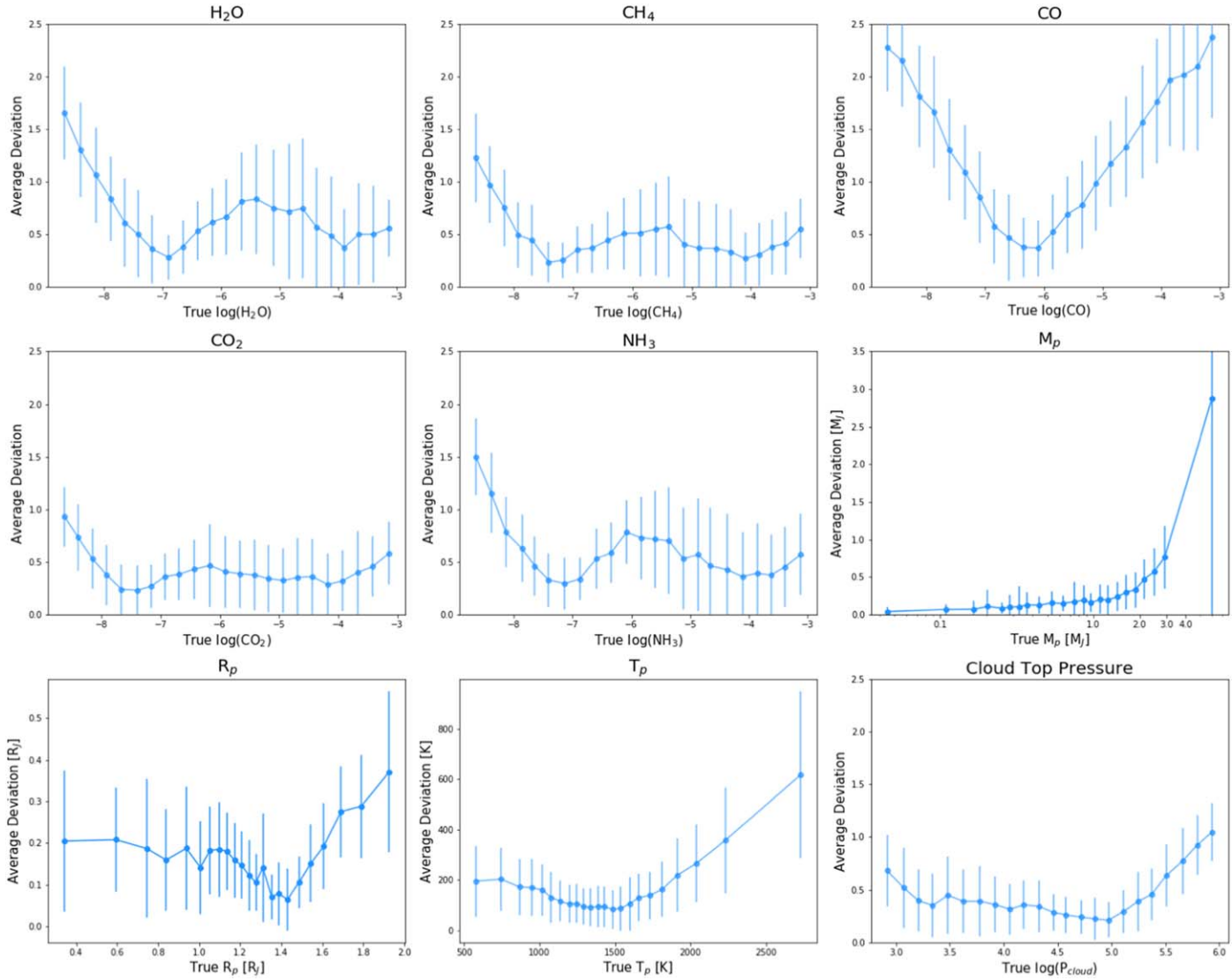

Figure 10. Visualization of bias and variance for different AMPs produced by the MLP model in their actual units. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. The performance is mostly similar to the one derived from the CNN model.

measure of the bias component of the prediction error), along with their standard deviation across the 20 runs (a proxy for the variance of the error). We then repeat the same experiment, but this time we provide 5000 training data points to the two models. The bottom part of Figure 9 corresponds to the results when the two models are trained with an increased training sample size.

We can see that the "simple" model has a lower predictive performance than the "complex" one. Moreover, its performance only slightly improves when provided with more data. The results suggest that the error of the "simple" model is mainly due to bias and it cannot be decreased when trained on more data. The model has a limited capacity that is smaller than the one required to model the data set in this situation, and thus it underfits. On the other hand, the "complex" model exhibits a lower bias (as measured by the average deviation) than the "simple" one. However, its error is characterized by a notably larger variance (as approximated by the standard deviation) compared to the "simple" model. Finally, when provided with a larger training sample, the variance of its predictions decreases.

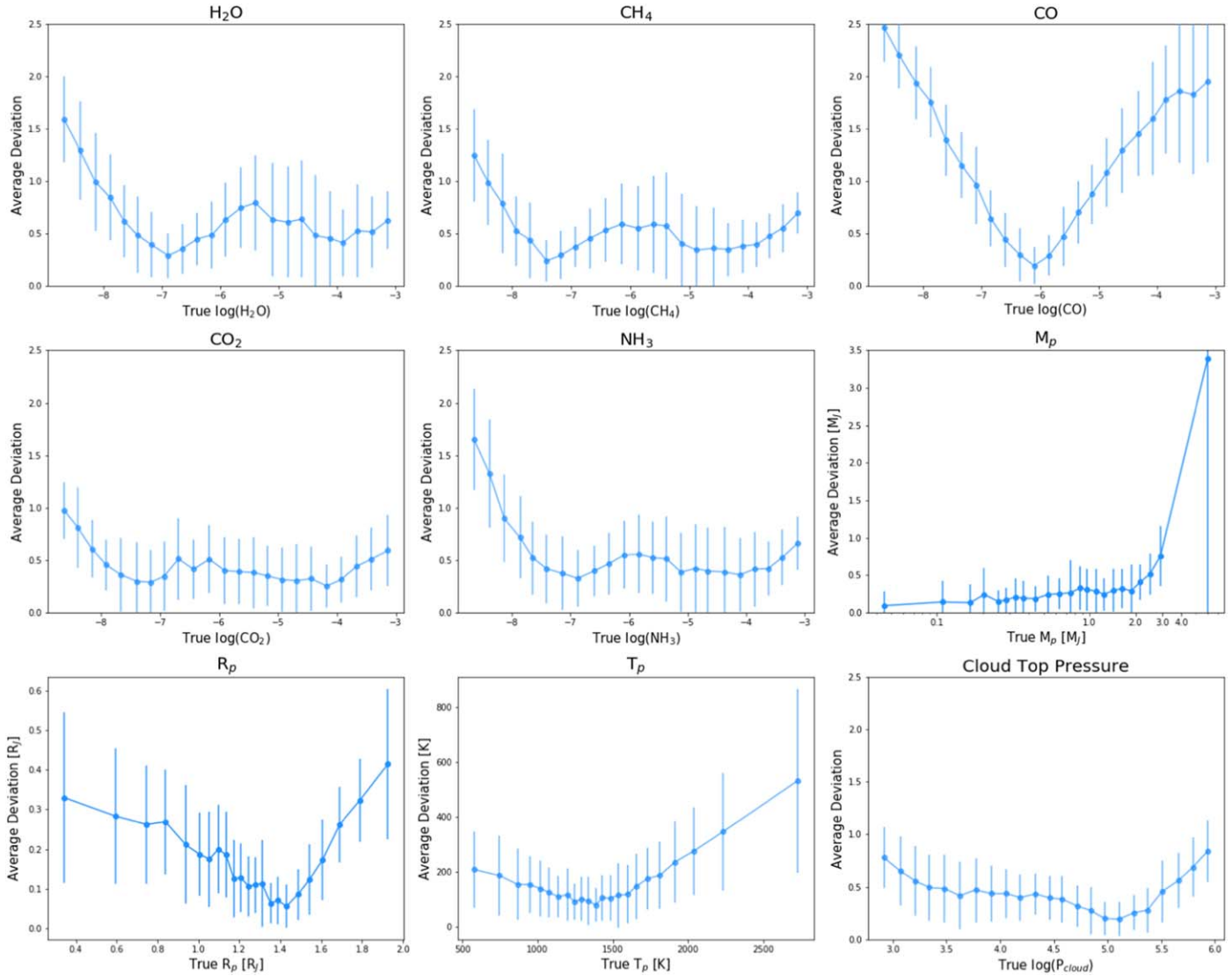

Figure 11. Visualization of bias and variance for different AMPs produced from the LSTM model. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. The performance is mostly similar to the one derived from the CNN model.

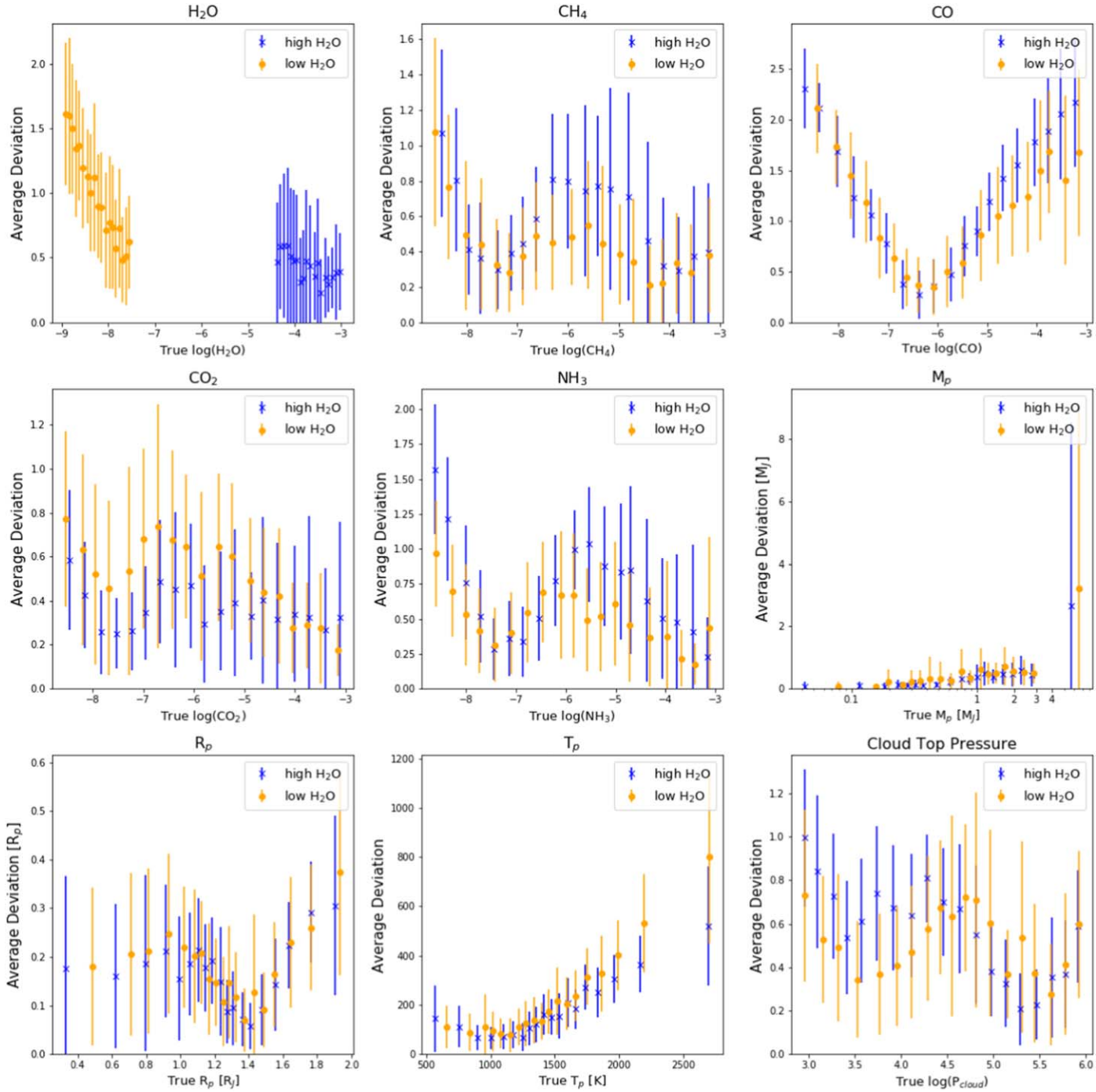

Figure 12. Visualization of bias and variance for different AMPs at high and low H_2O (log-)abundance. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

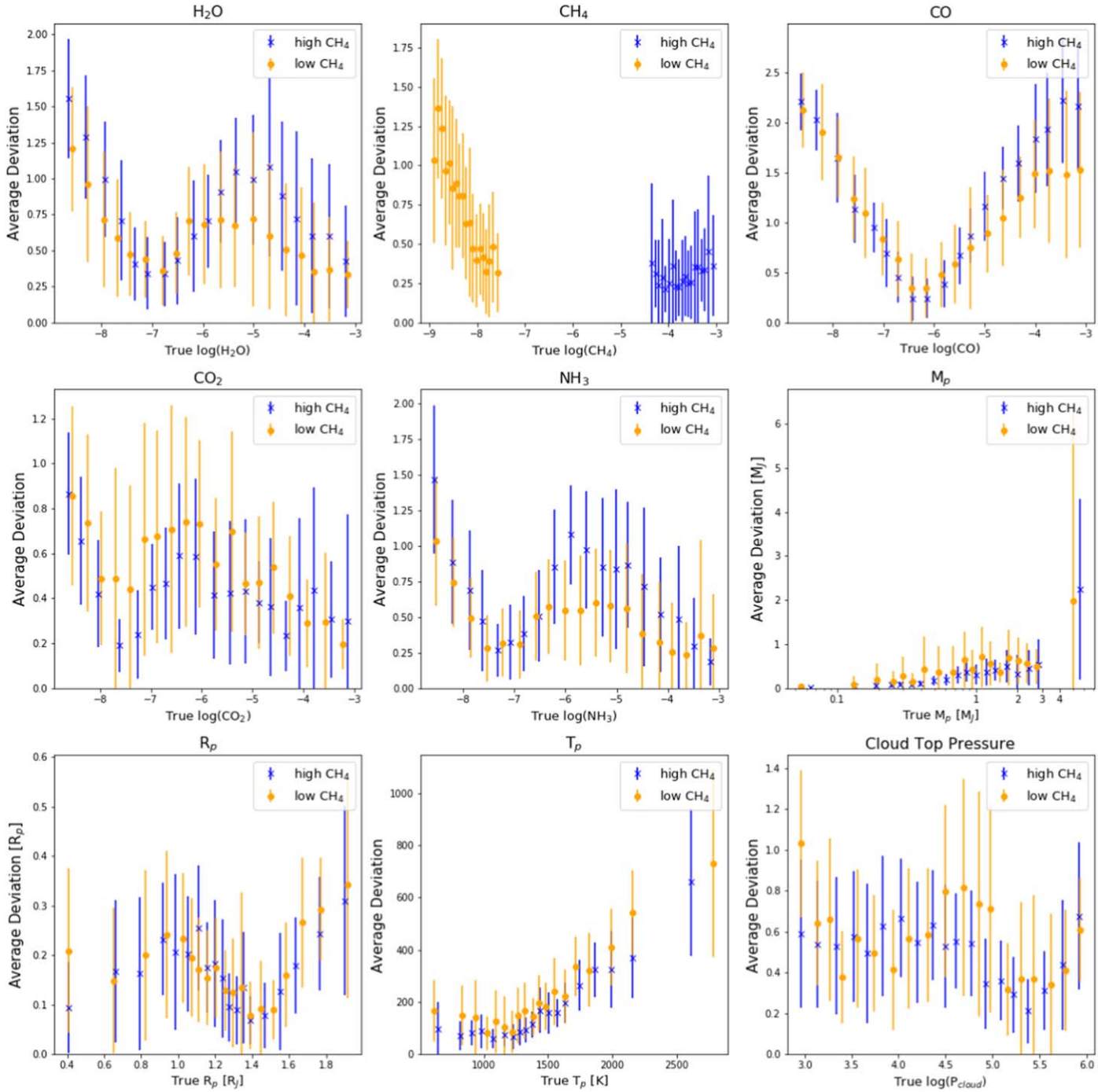

Figure 13. Visualization of bias and variance for different AMPs at high and low CH_4 (log-)abundance. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

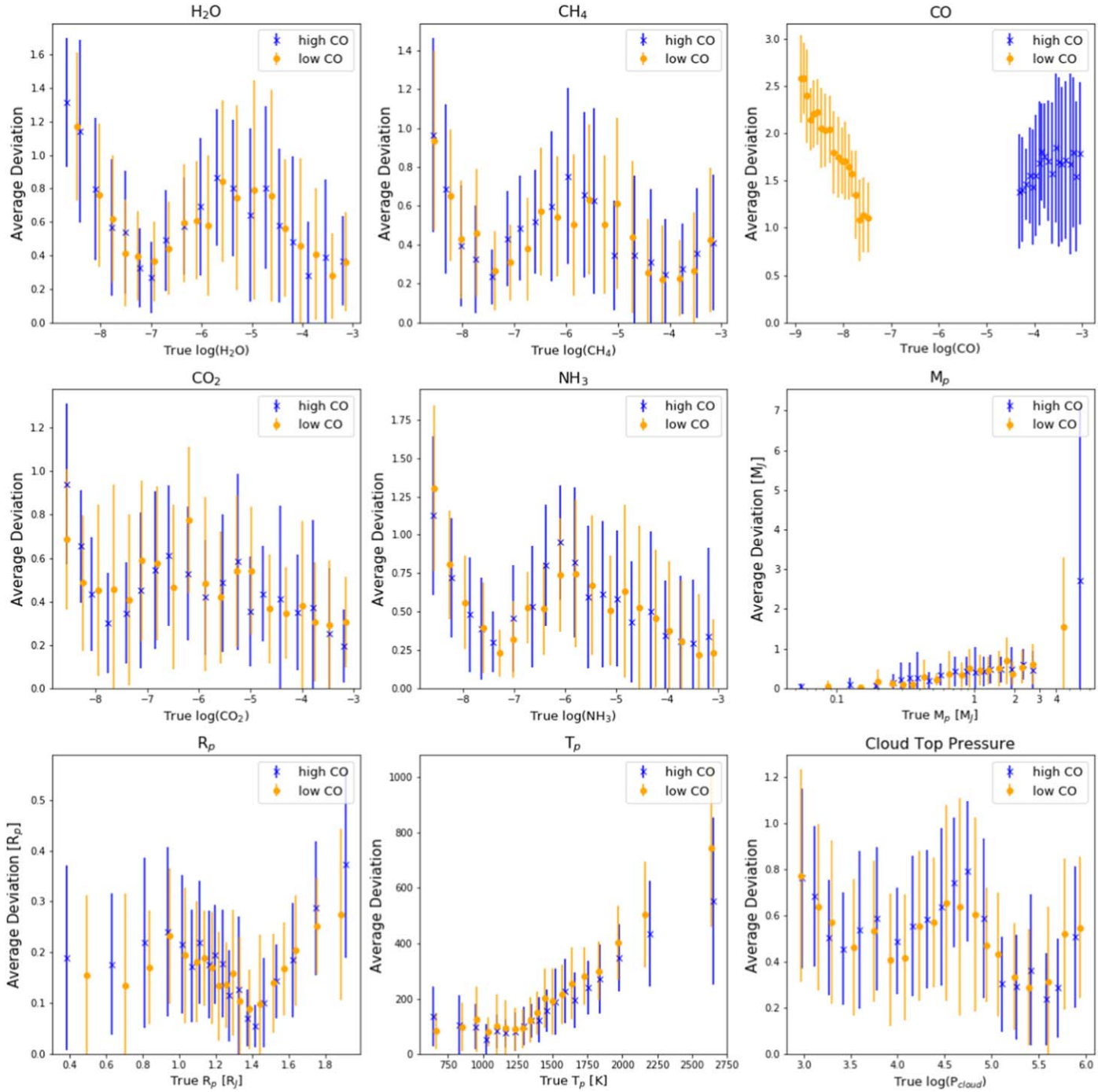

Figure 14. Visualization of bias and variance for different AMPs at high and low CO abundance. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

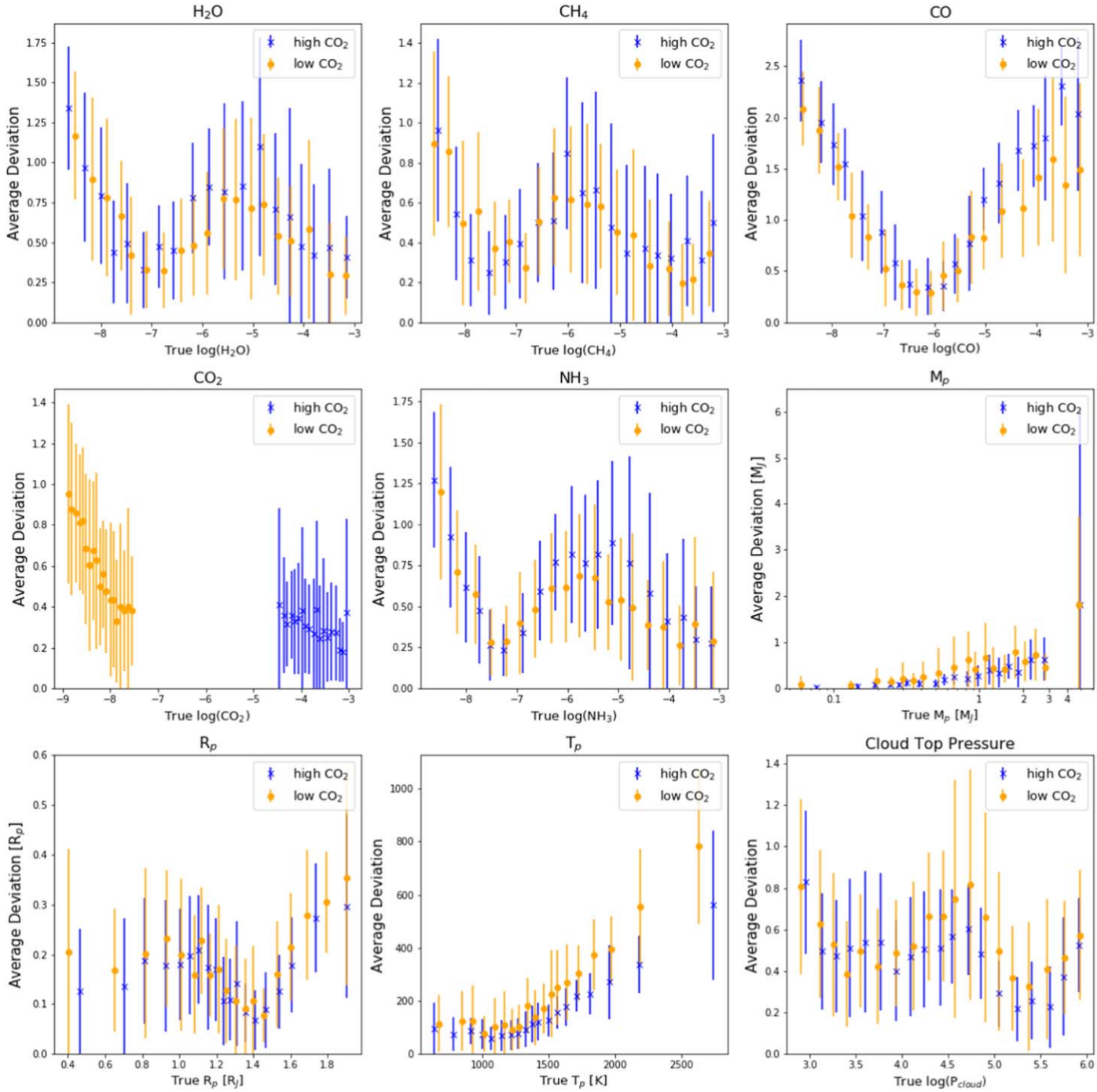

Figure 15. Visualization of bias and variance for different AMPs at high and low CO_2 (log-)abundance. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

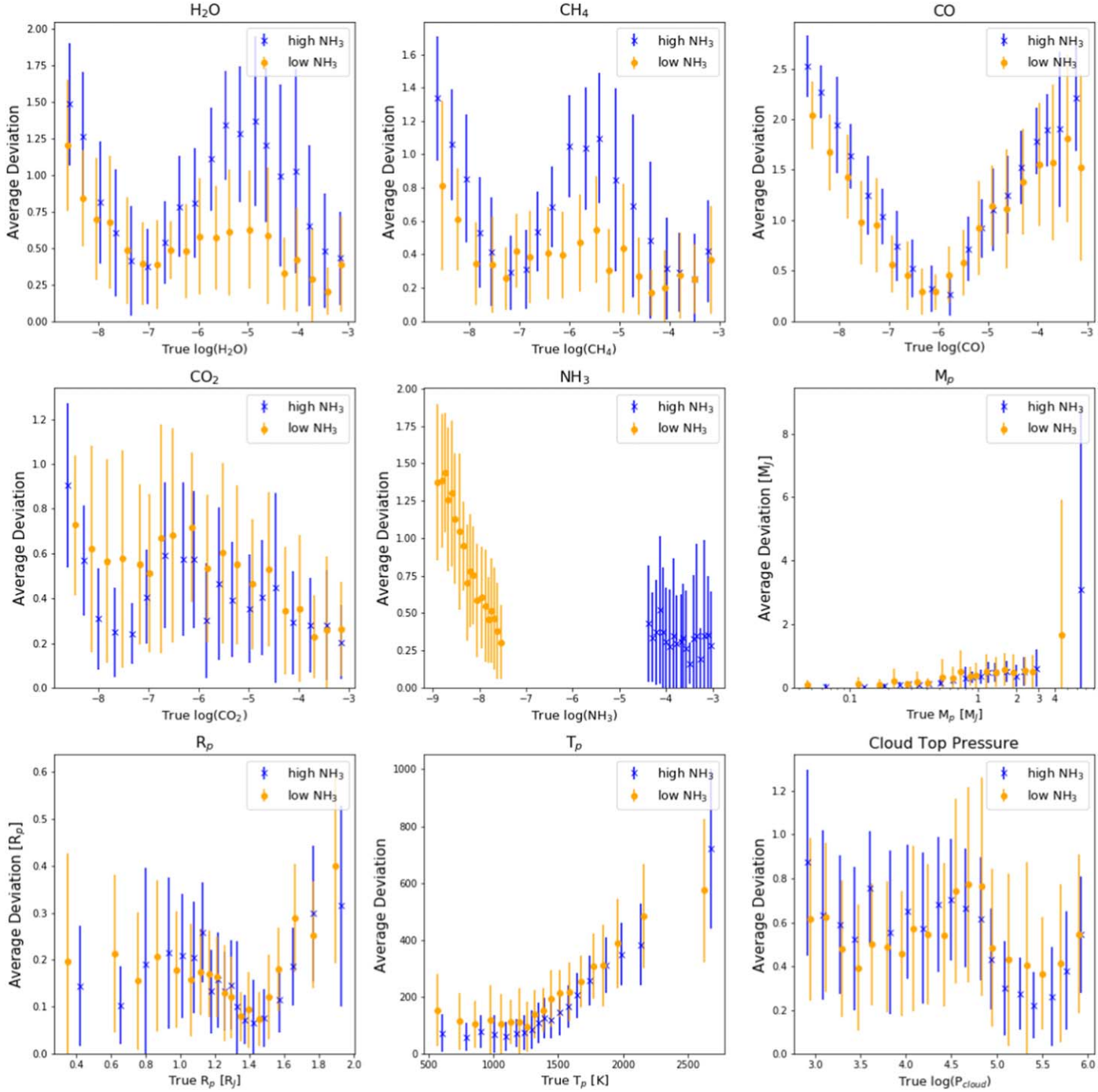

Figure 16. Visualization of bias and variance for different AMPs at high and low NH_3 abundance. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

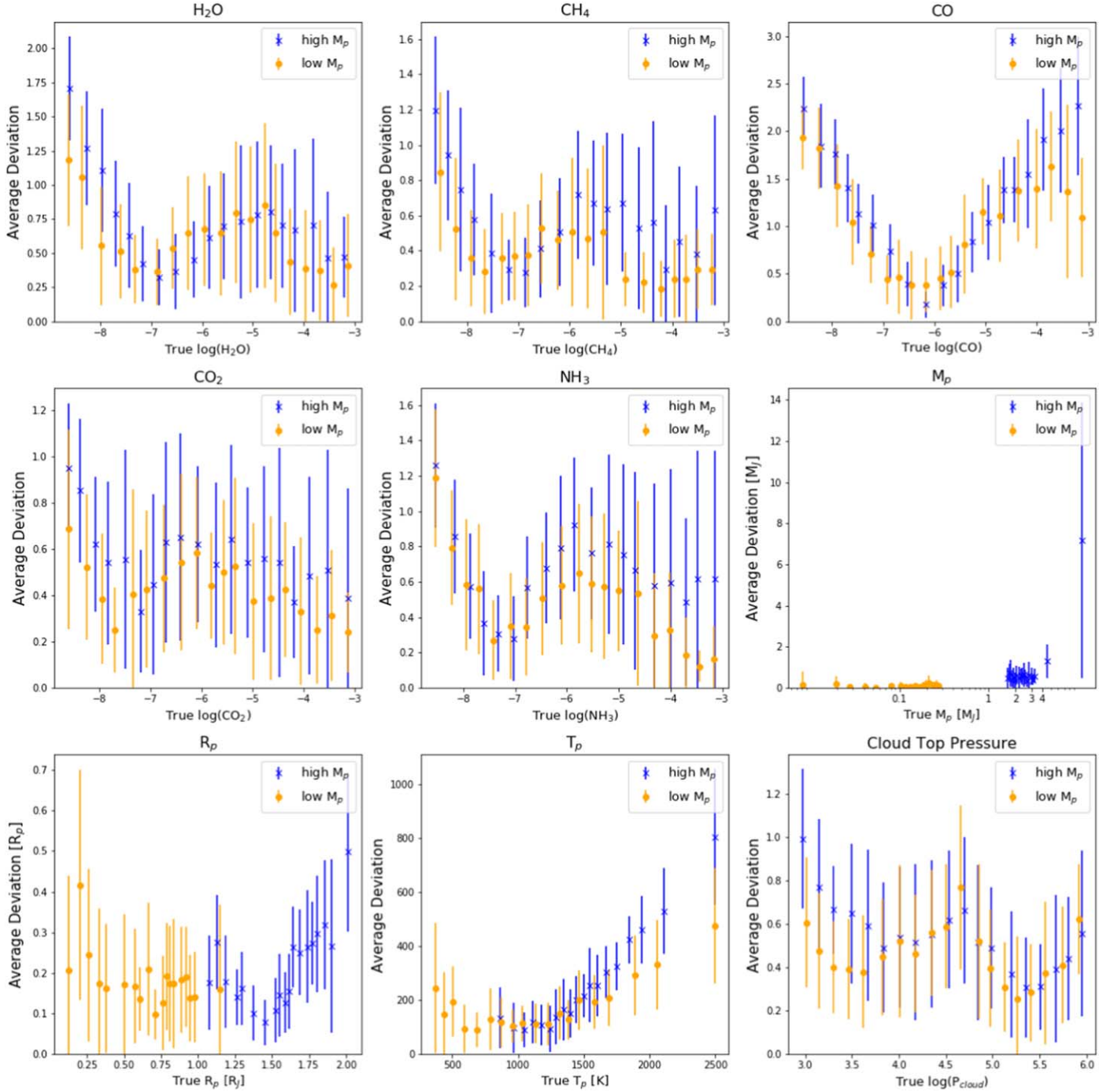

Figure 17. Visualization of bias and variance for different AMPs at high and low M_p. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

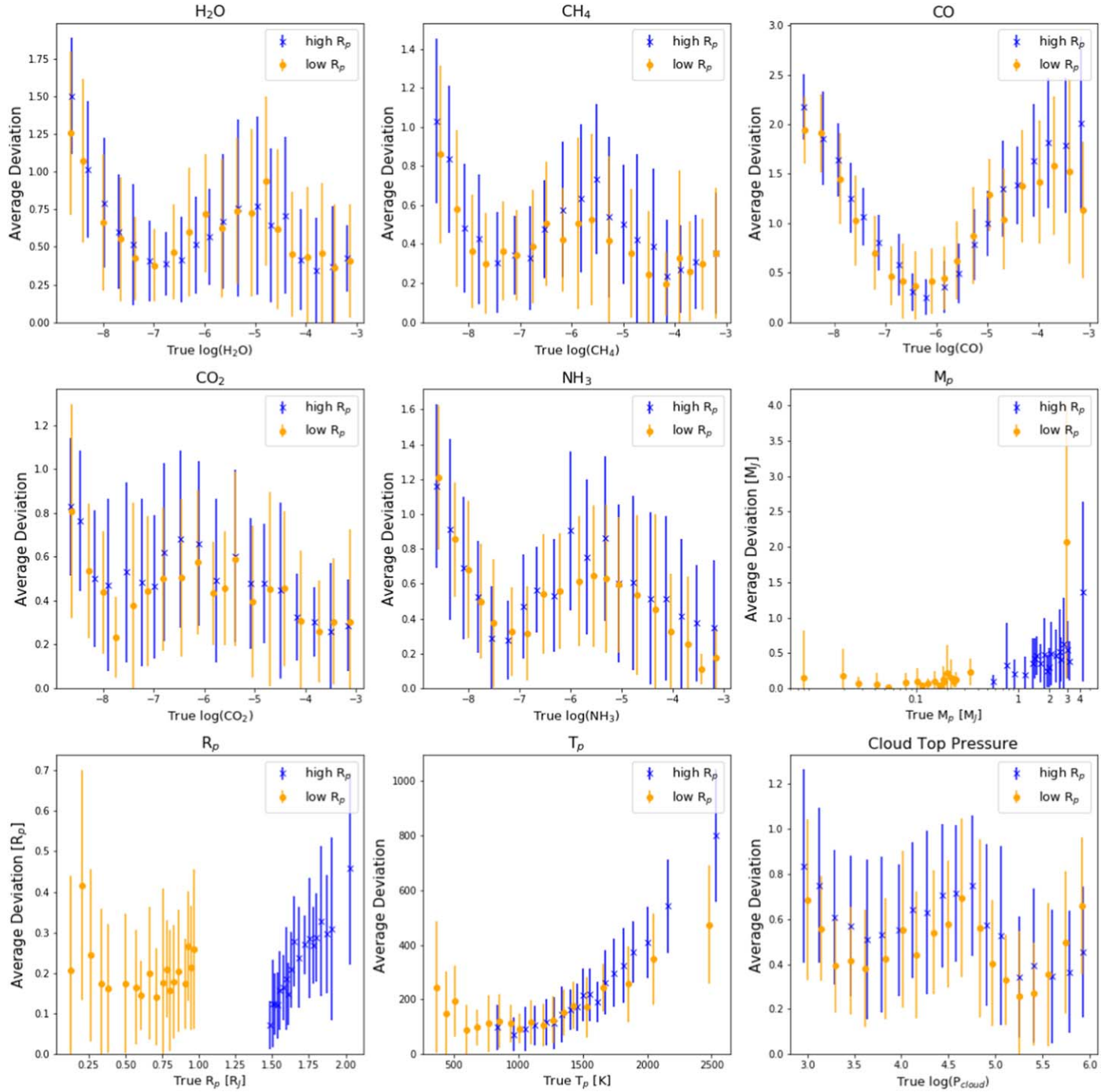

Figure 18. Visualization of bias and variance for different AMPs at high and low R_p. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

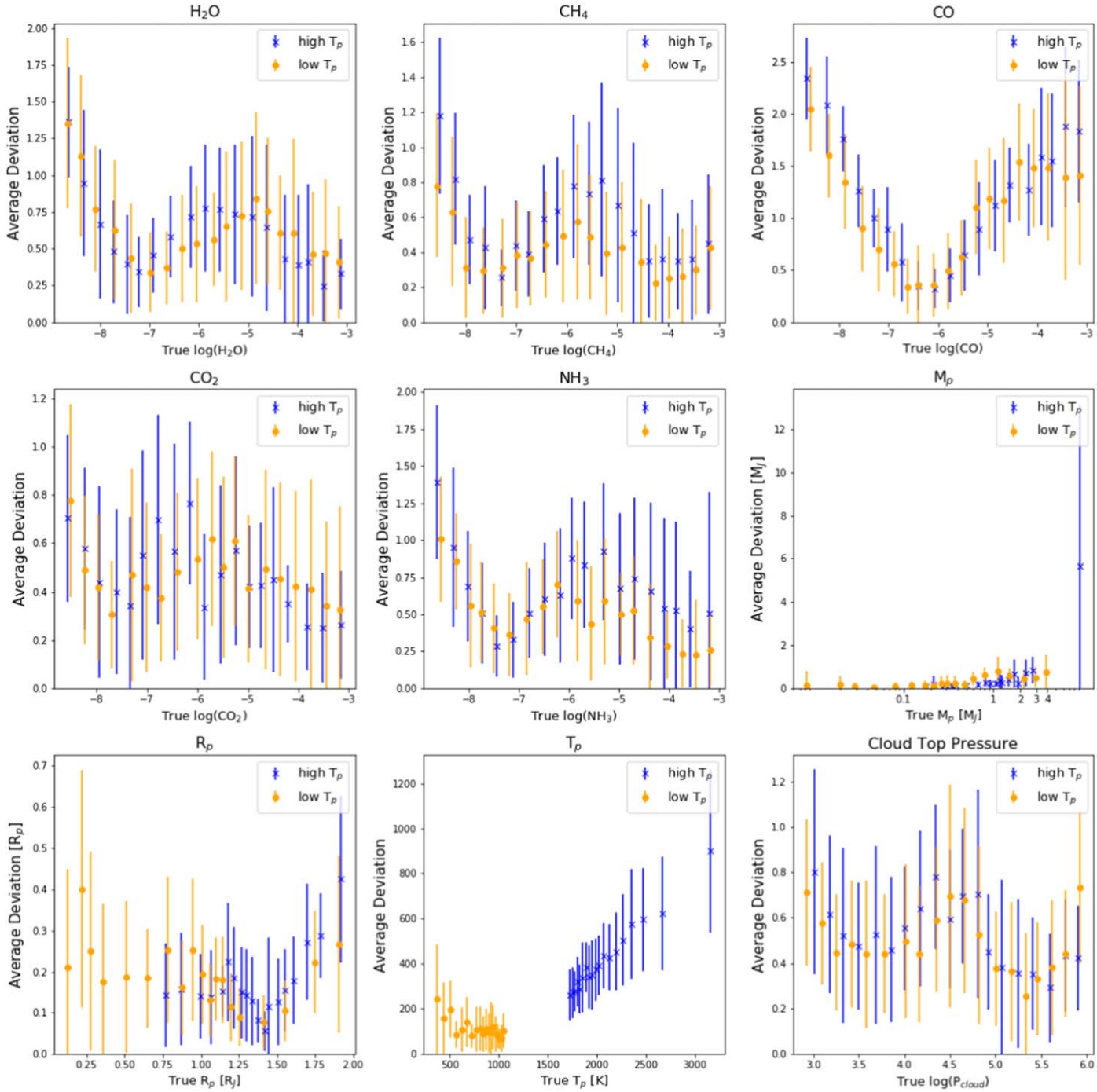

Figure 19. Visualization of bias and variance for different AMPs at high and low T_p. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction. This plot is generated using the CNN model.

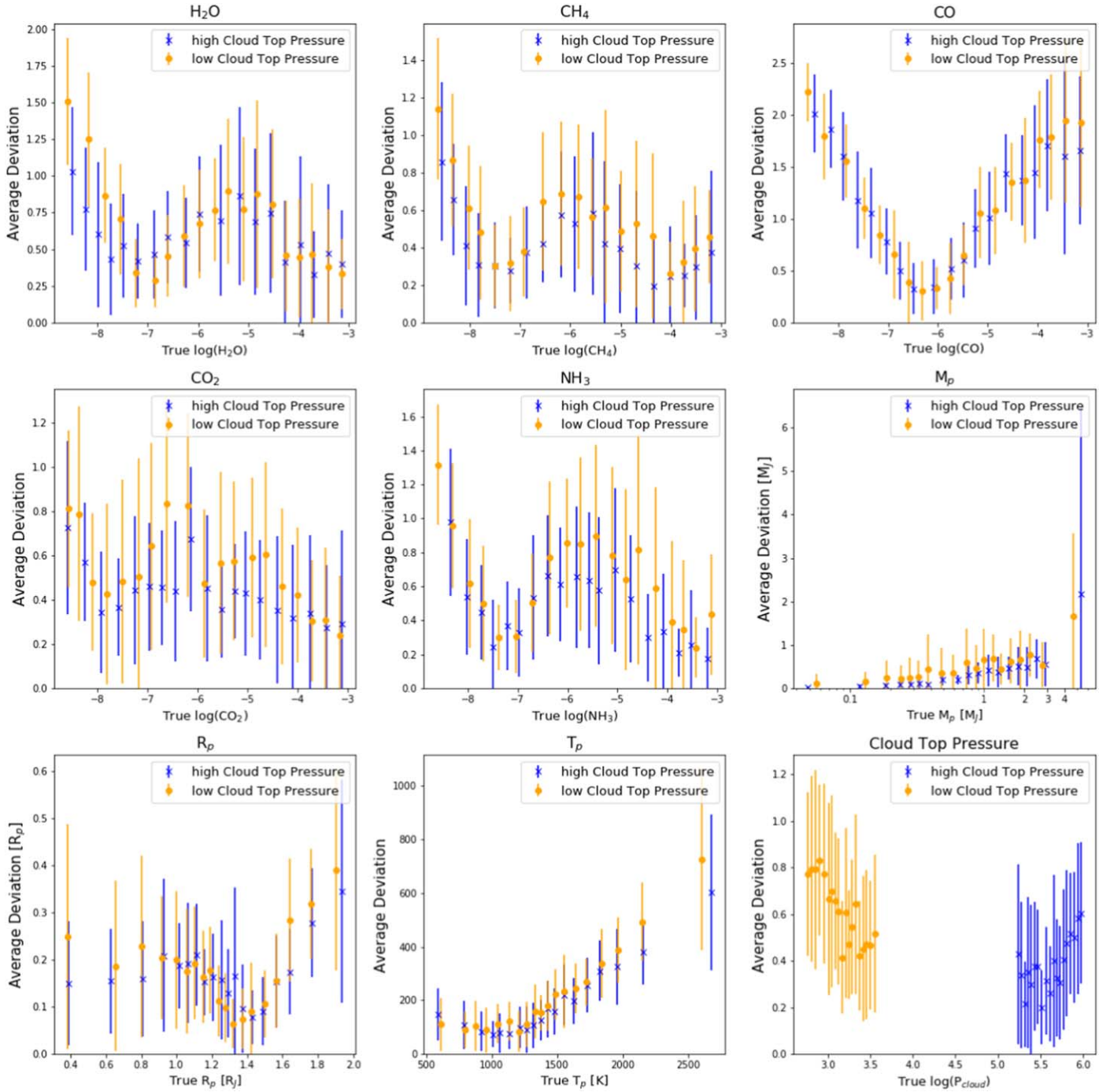

Figure 20. Visualization of bias and variance for different AMPs at high and low Cloud Top Pressure. Each point represents the average deviation at that level, and its error bar represents the 1σ spread of the prediction

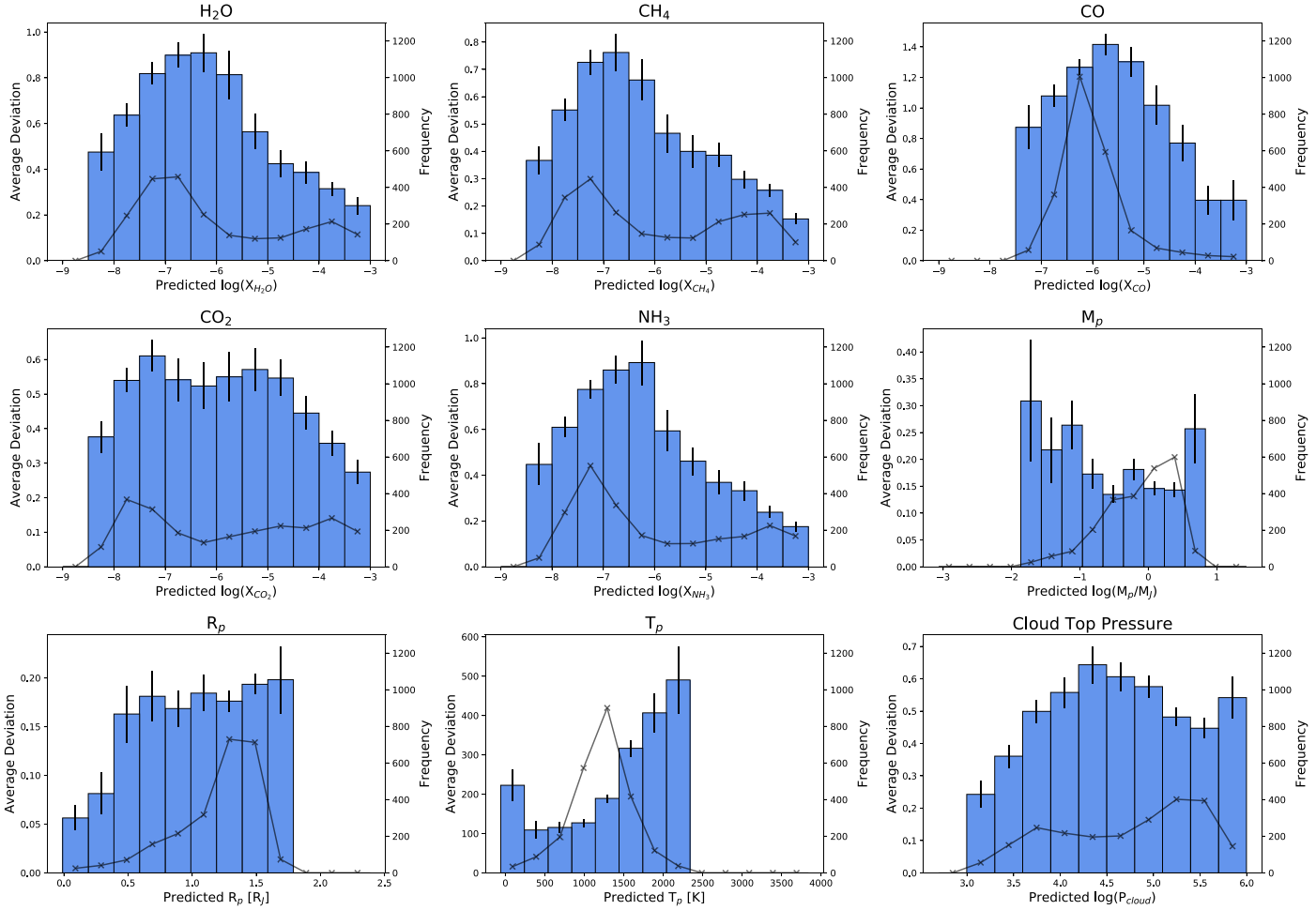

Figure 21. Distribution of average deviation from the ground truth at each prediction level for different AMPs. The deviation for each AMP is presented in actual unit. The black curve represents the frequency of model predictions for that bin. Bins are sampled at equal width with sample size less than 20 ignored. Note that M_p is defined as $\log(M_p/M_J)$ for better visualization. The model generally performs better when R_p and T_p is small. While for M_p, the model struggles with extreme cases, i.e., very light or heavy planets, this could be due to lack of examples in those regions.

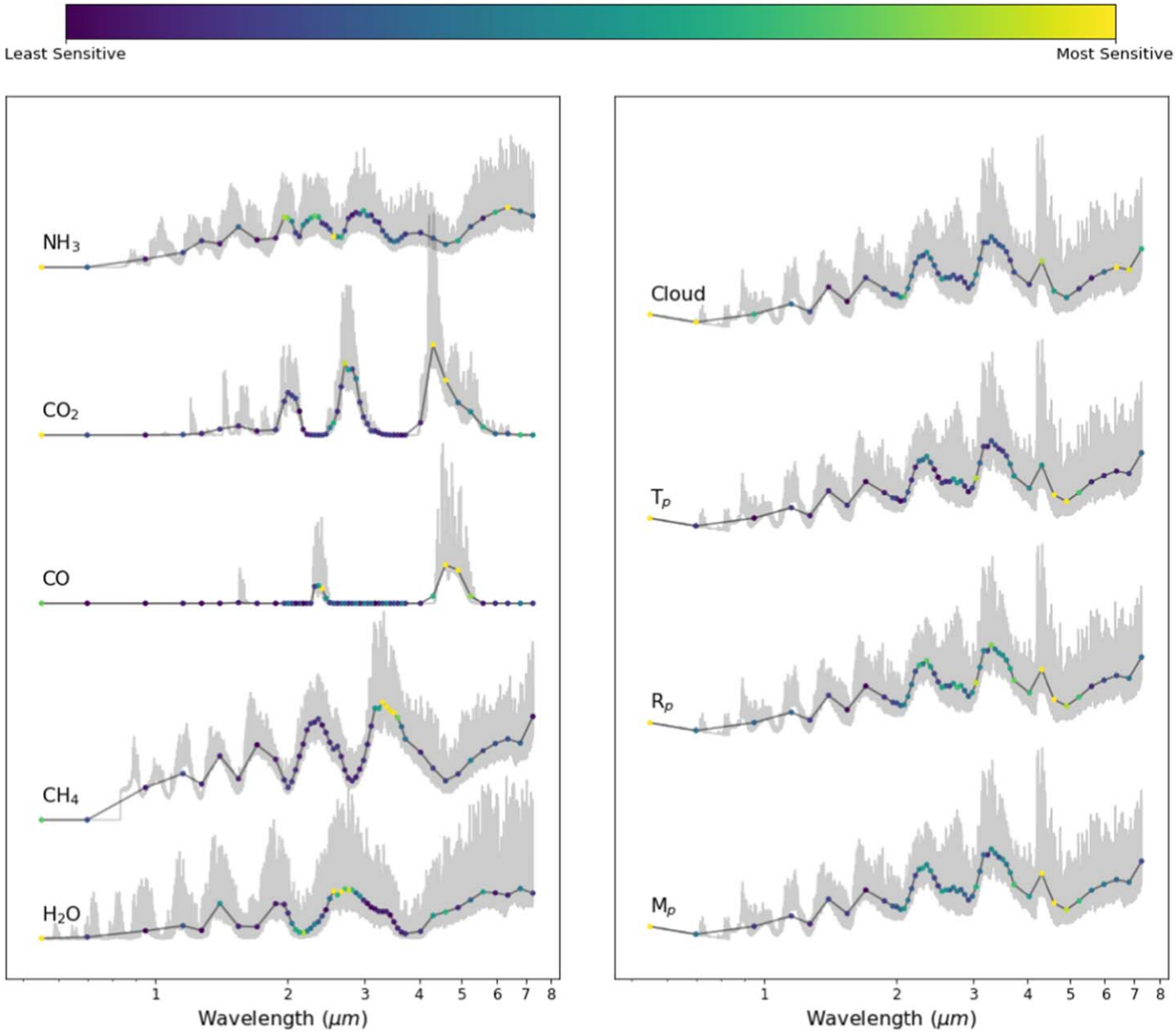

Figure 22. Sensitivity map for the MLP model. For each AMP, we visualize the sensitivity of the model to each wavelength w.r.t. predicting it. Yellow denotes the wavelength to which the model is most sensitive, and black the one to which it is the least. The MLP model is more sensitive to the photometric points in most cases, which could be linked to obtaining a baseline for the model. For nongaseous AMPs, in contrast to the CNN and LSTM (see below) models, the MLP focuses more on other parts of the spectrum than the region 2–3 μm. This hints that the MLP captures different aspects of the underlying physics than the other two models. This is not surprising, as both CNN and LSTM architectures encode more local structure constraints than MLPs.

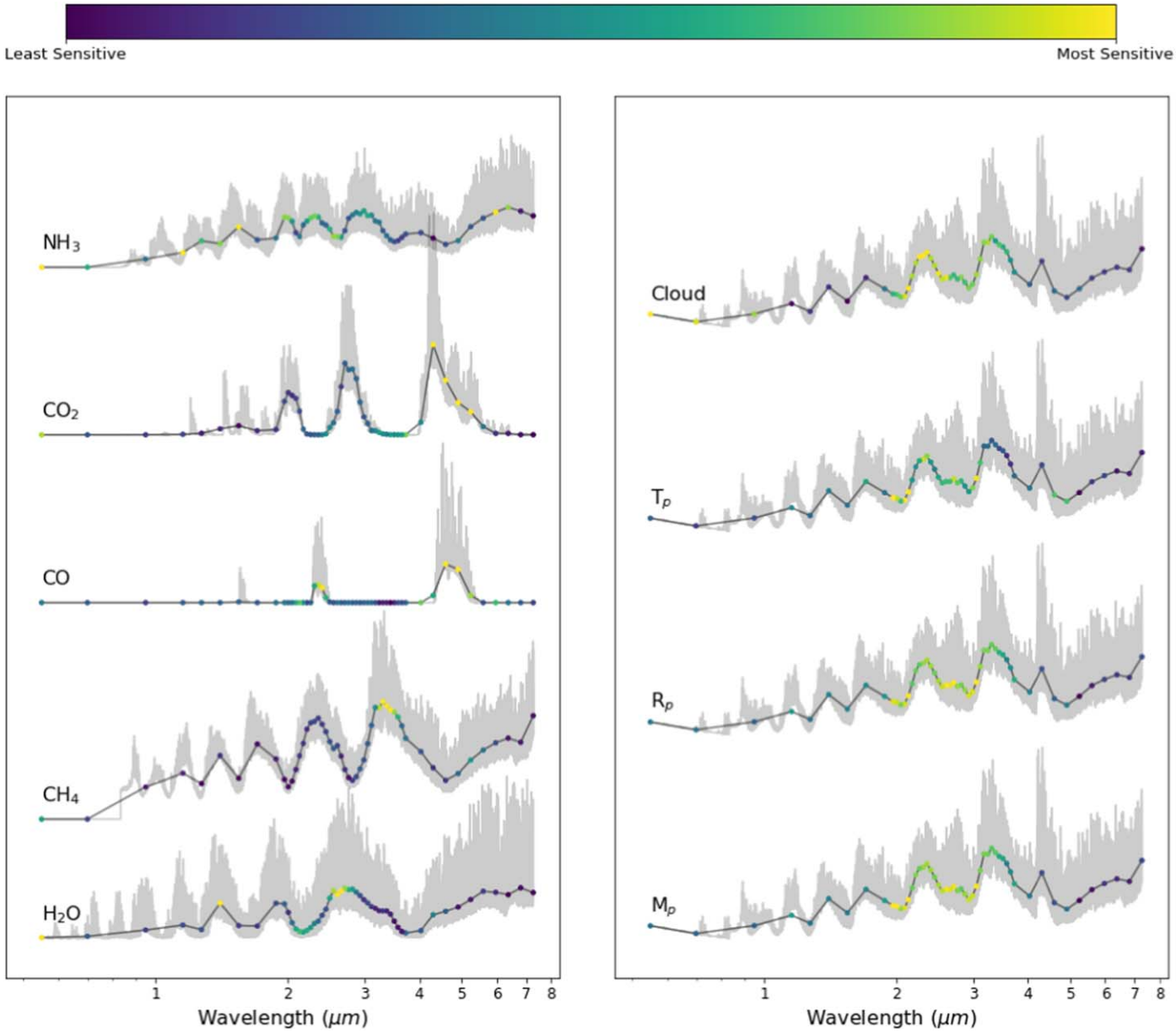

Figure 23. Sensitivity map for the LSTM model. For each AMP, we visualize the sensitivity of the model to each wavelength w.r.t. predicting it. Yellow denotes the wavelength to which the model is most sensitive, and black the one to which it is the least. Similar to the CNN model, the LSTM model focuses in the region 2–3 μm for nongaseous AMPs.

Appendix C
Credibility Limit Calculations

Consider random variables Y and $\hat{Y}$ that represent the ground truth value of an AMP for a particular instance and a model's prediction on an AMP (i.e., $\log X_{H_2O}$ abundance) respectively, we can define the credibility limit $L(\delta, \epsilon)$ for this AMP as the lowest possible predicted value $\hat{y}$ for which:

$$P(Y = \hat{y} \pm \epsilon | \hat{Y} = \hat{y}) > 1 - \delta, \tag{C1}$$

where $P(Y = \hat{y} \pm \epsilon | \hat{Y} = \hat{y})$ is the probability of finding a model's prediction $\hat{y}$ within ϵ of the ground truth y. It can be computed for the entire set or per each abundance bin as:

$$P = \frac{\#(Y \in |\hat{y} - \epsilon, \hat{y} + \epsilon| \cap \hat{Y} = \hat{y})}{\#(\hat{Y} = \hat{y})}, \tag{C2}$$

where $\#(\cdot)$ denotes the number of data points that satisfy the conditions enclosed in the parentheses.

Appendix D
Sensitivity Map Calculation: Implementation

This section provides a detailed description on how to compute the sensitivity map for a particular target. The notation follows that of the general procedure outlined in Section 4.1.

Given any trained predictive model $\mathcal{M}$ (e.g., a neural network) that takes an input spectrum $\boldsymbol{x}$ and outputs the corresponding prediction $\hat{\boldsymbol{y}}$, a perturbation-based sensitivity test can be performed to assess the change in the prediction $\hat{\boldsymbol{y}}$ when a set of features (transit depth, in our case) x_i (consecutive or not) is perturbed. Algorithm 1 and 2 demonstrates our procedure to compute the sensitivity map for a single spectrum on a single prediction.

Our implementation assumes no prior knowledge on the size and distribution of spectral features. Perturbation is applied to a randomly selected bin or group of bins at each iteration. Parameter k controls the number of bins selected each time. Here, we iterate through different values of k to capture spectral features at different scales. To quantify the deviation of the perturbed prediction from the unperturbed, we chose the Mean Squared Difference (MSE), due to its sensitivity to large differences, which helps to highlight sensitive regions.

The general procedure for generating sensitivity maps described in Section 4.1 is not tied to any specific distance measure, way of selecting the wavelength bins to be perturbed, or perturbation method. The only requirement is that perturbed wavelength bin(s) be attributed a score based on the magnitude of the distance of the two predictions.

Algorithm 1. Sensitivity map calculation

Data: unperturbed spectrum $\boldsymbol{x_r}$, associated uncertainty vector $\boldsymbol{\sigma}$ and number of repetitions N
Result: sensitivity score, score, of size $(|\boldsymbol{x_r}| \times |\hat{\boldsymbol{y}}_r|)$ for each input feature (wavelength bin) x_i and output variable AMP.
begin
  // get prediction from an unperturbed spectrum;
  $\hat{\boldsymbol{y}}_r \leftarrow$ `ModelPrediction` $(\boldsymbol{x_r})$;
  // empty array to store results;
  delta $\leftarrow$ `Zeros` $(|\boldsymbol{x_r}|, |\hat{\boldsymbol{y}}_r|)$;
  $l \leftarrow |\boldsymbol{x_r}|$;
  for $n \leftarrow 0$ **To** $N - 1$ do
    $\boldsymbol{L} \leftarrow$ `Zeros` $()|\boldsymbol{x_r}|, |\hat{\boldsymbol{y}}_r|$;
    // Extract position index from an array;
    index $\leftarrow$ `GetIndex` $(\boldsymbol{x_r})$;
    // shuffle index at random;
    shuffled_index $\leftarrow$ `Shuffle(index)`;
    $k \leftarrow l$;
    while $k \geqslant 2$ **do**;
      $k \leftarrow$ `Ceil` $(k/2)$;
      // randomly draw k positions without repetition;
      $\boldsymbol{pos} \leftarrow$ `DrawWithoutRepetition(shuffled_index, k)`;
      $\boldsymbol{x_p} \leftarrow$ `Perturb` $(\boldsymbol{x_r}, \boldsymbol{\sigma}, \boldsymbol{pos})$;
      $\hat{\boldsymbol{x}}_p \leftarrow$ `ModelPrediction` $(\boldsymbol{x_p})$;
      // assign $\hat{\boldsymbol{x}}_p$ to $\boldsymbol{L}$ at position $\boldsymbol{pos}$;
      $\boldsymbol{L}[\boldsymbol{pos}, :] \leftarrow \hat{\boldsymbol{x}}_p$;
    end;
    // user-defined distance metric;
    diff $\leftarrow$ `SquaredDifference` $(\hat{\boldsymbol{y}}_r, \boldsymbol{L})$;
    delta $\leftarrow$ delta + diff;
    $n \leftarrow n + 1$;
  end;
  score $\leftarrow$ delta/N;
end

Algorithm 2. Perturbation method used in this investigation

Function Perturb $(\boldsymbol{x_r}, \boldsymbol{\sigma}, \boldsymbol{pos})$;
Data: unperturbed spectrum $\boldsymbol{x_r}$, associated uncertainty vector $\boldsymbol{\sigma}$ and set of positions in which to apply perturbation $\boldsymbol{pos}$
Result: perturbed spectrum $\boldsymbol{x_p}$
begin
  $\boldsymbol{x_p} \leftarrow$ `Copy` $(\boldsymbol{x_r})$;
  sample $\leftarrow$ `DrawFromGaussian` $(\boldsymbol{x_p}[\boldsymbol{pos}], \boldsymbol{\sigma}[\boldsymbol{pos}])$;
  // replace values at position $\boldsymbol{pos}$ with perturbed values;
  $\boldsymbol{x_p}[\boldsymbol{pos}] \leftarrow$ sample;
end

ORCID iDs

Kai Hou Yip https://orcid.org/0000-0002-9616-1524
Quentin Changeat https://orcid.org/0000-0001-6516-4493
Nikolaos Nikolaou https://orcid.org/0000-0001-8453-7574
Mario Morvan https://orcid.org/0000-0001-8587-2112
Billy Edwards https://orcid.org/0000-0002-5494-3237
Ingo P. Waldmann https://orcid.org/0000-0002-4205-5267
Giovanna Tinetti https://orcid.org/0000-0001-6058-6654

References

Abadi, M., Agarwal, A., Barham, P., et al. 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, http://tensorflow.org/
Al-Refaie, A. F., Changeat, Q., Waldmann, I. P., & Tinetti, G. 2021, ApJ, 917, 37
Al-Saffar, A. A. M., Tao, H., & Talab, M. A. 2017, in Int. Conf. on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET) (Piscataway, NJ: IEEE), 26
Arcangeli, J., Désert, J.-M., Line, M. R., et al. 2018, ApJL, 855, L30
Barman, T. S., Konopacky, Q. M., Macintosh, B., & Marois, C. 2015, ApJ, 804, 61
Berta, Z. K., Charbonneau, D., Désert, J.-M., et al. 2012, ApJ, 747, 35
Brown, T. M. 2001, ApJ, 553, 1006
Changeat, Q., Al-Refaie, A., Mugnai, L. V., et al. 2020a, AJ, 160, 80
Changeat, Q., Edwards, B., Waldmann, I. P., & Tinetti, G. 2019, ApJ, 886, 39
Changeat, Q., Keyte, L., Waldmann, I. P., & Tinetti, G. 2020b, ApJ, 896, 107
Chaushev, A., Raynard, L., Goad, M. R., et al. 2019, MNRAS, 488, 5232
Chollet, F., et al. 2015, Keras, https://keras.io
Cobb, A. D., Himes, M. D., Soboczenski, F., et al. 2019, AJ, 158, 33
Collette, A. 2013, Python and HDF5 (Sebastopol, CA: O'Reilly Media, Inc.)

Dattilo, A., Vanderburg, A., Shallue, C. J., et al. 2019, AJ, 157, 169
de Kok, R. J., Brogi, M., Snellen, I. A. G., et al. 2013, A&A, 554, A82
de Wit, J., & Seager, S. 2013, Sci, 342, 1473
Edwards, B., Changeat, Q., Baeyens, R., et al. 2020, AJ, 160, 8
Edwards, B., Mugnai, L., Tinetti, G., Pascale, E., & Sarkar, S. 2019a, AJ, 157, 242
Edwards, B., Rice, M., Zingales, T., et al. 2019b, ExA, 47, 29
Ehrenreich, D., Bonfils, X., Lovis, C., et al. 2014, A&A, 570, A89
Fisher, C., & Heng, K. 2018, MNRAS, 481, 4698
Fisher, C., Hoeijmakers, H. J., Kitzmann, D., et al. 2020, AJ, 159, 192
Fortney, J. J. 2005, MNRAS, 364, 649
Fossati, L., Haswell, C. A., Froning, C. S., et al. 2010, ApJL, 714, L222
Gordon, I., Rothman, L. S., Wilzewski, J. S., et al. 2016, AAS Meeting, 48, 421.13
Greene, T. P., Line, M. R., Montero, C., et al. 2016, ApJ, 817, 17
Griffith, C. A. 2014, Philos. Trans. Royal Soc. A, 372, 20130086
Himes, M. D., Harrington, J., Cobb, A. D., et al. 2020, arXiv:2003.02430
Hunter, J. D. 2007, CSE, 9, 90
Irwin, P. G. J., Teanby, N. A., de Kok, R., et al. 2008, J. Quant. Spec. Radiat. Transf., 109, 1136
Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. 2019, Data Min. Knowl. Disc., 33, 917
Iyer, A. R., Swain, M. R., Zellem, R. T., et al. 2016, ApJ, 823, 109
Krick, J., Fraine, J., Ingalls, J., & Deger, S. 2020, ApJ, 160, 99
Lecavelier des Etangs, A., Pont, F., Vidal-Madjar, A., & Sing, D. 2008, A&A, 481, L83
Line, M. R., Wolf, A. S., Zhang, X., et al. 2013, ApJ, 775, 137
Linsky, J. L., Yang, H., France, K., et al. 2010, ApJ, 717, 1291
MacDonald, R. J., & Madhusudhan, N. 2017, ApJL, 850, L15
Macintosh, B., Graham, J. R., Barman, T., et al. 2015, Sci, 350, 64
Madhusudhan, N., & Seager, S. 2009, ApJ, 707, 24
Mandell, A. M., Haynes, K., Sinukoff, E., et al. 2013, ApJ, 779, 128
Márquez-Neila, P., Fisher, C., Sznitman, R., & Heng, K. 2018, NatAs, 2, 719
McCauliff, S. D., Jenkins, J. M., Catanzarite, J., et al. 2015, ApJ, 806, 6
McKinney, W. 2011, in Python for High Performance and Scientific Computing, Vol. 14
Molnar, C. 2019, Interpretable Machine Learning, https://christophm.github.io/interpretable-ml-book/
Morvan, M., Nikolaou, N., Tsiaras, A., & Waldmann, I. P. 2020, AJ, 159, 109
Mugnai, L. V., Pascale, E., Edwards, B., Papageorgiou, A., & Sarkar, S. 2020, arXiv:2009.07824
Murphy, K. P. 2012, Machine Learning: A Probabilistic Perspective (Cambridge, MA: MIT Press)
Nixon, M. C., & Madhusudhan, N. 2020, MNRAS, 496, 269
Oliphant, T. E. 2006, A guide to NumPy, Vol. 1 (USA: Trelgol Publishing)
Ormel, C. W., & Min, M. 2019, A&A, 622, A121
Osborn, H. P., Ansdell, M., Ioannou, Y., et al. 2020, A&A, 633, A53
Pearson, K. A., Palafox, L., & Griffith, C. A. 2018, MNRAS, 474, 478
Reyes, O., & Ventura, S. 2019, IJNS, 29, 1950014
Rocchetto, M., Waldmann, I. P., Venot, O., Lagage, P. O., & Tinetti, G. 2016, ApJ, 833, 120
Rothman, L. S., & Gordon, I. E. 2014, in 13th International HITRAN Conf., June 2014 (Cambridge, MA: DPS)
Schanche, N., Collier Cameron, A., Hébrard, G., et al. 2019, MNRAS, 483, 5534
Shallue, C. J., & Vanderburg, A. 2018, AJ, 155, 94
Sharp, C. M., & Burrows, A. 2007, ApJS, 168, 140
Sing, D. K., Fortney, J. J., Nikolov, N., et al. 2016, Natur, 529, 59
Tennyson, J., Yurchenko, S. N., Al-Refaie, A. F., et al. 2016, JMoSp, 327, 73, new Visions of Spectroscopic Databases, Volume II
Tinetti, G., Drossart, P., Eccleston, P., et al. 2018, ExA, 46, 135
Tsiaras, A., Waldmann, I. P., Zingales, T., et al. 2018, AJ, 155, 156
Waldmann, I. P. 2016, ApJ, 820, 107
Yip, K. H. 2021, sensitivity analysis code, v1.0, Zenodo doi:10.5281/zenodo.4587343
Young, T., Hazarika, D., Poria, S., & Cambria, E. 2017, arXiv:1708.02709
Yu, L., Vanderburg, A., Huang, C., et al. 2019, AJ, 158, 25
Zeiler, M. D., & Fergus, R. 2013, arXiv:1311.2901
Zhang, M., Chachan, Y., Kempton, E. M. R., & Knutson, H. A. 2019, PASP, 131, 034501
Zingales, T., & Waldmann, I. P. 2018, AJ, 156, 268