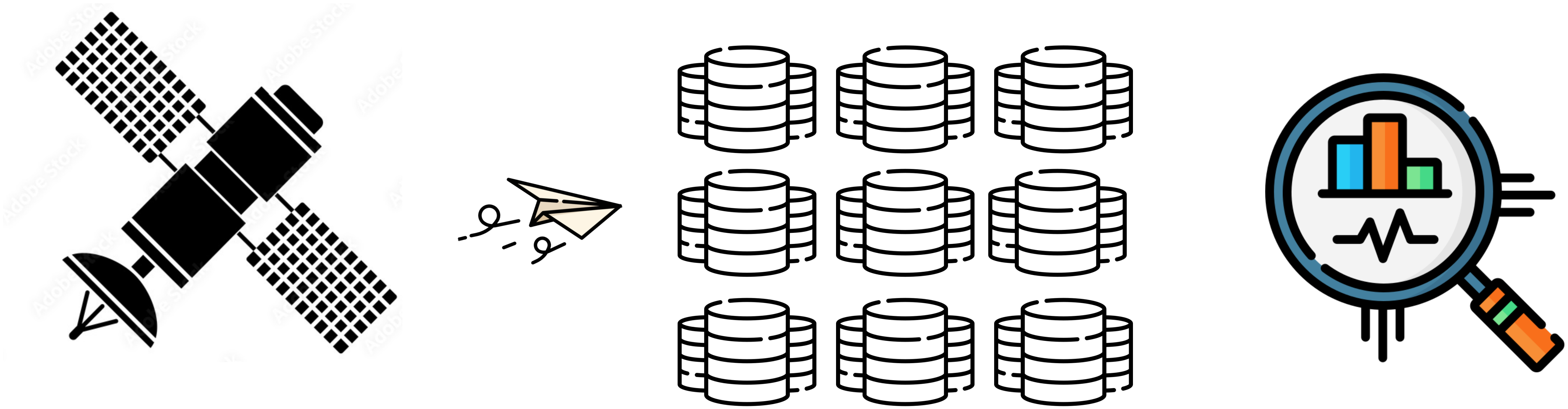
Can Artificial Neural Networks help to understand X-ray spectra?

Laura Manduchi, ETH Zürich

ESA/ESO SCIOPS 2022

Collaborators: Guillermo Ayllón, Norbert Schartel, Richard Saxton, Maria Santos-Lleo, Felix Fuerst

X-ray Spectra

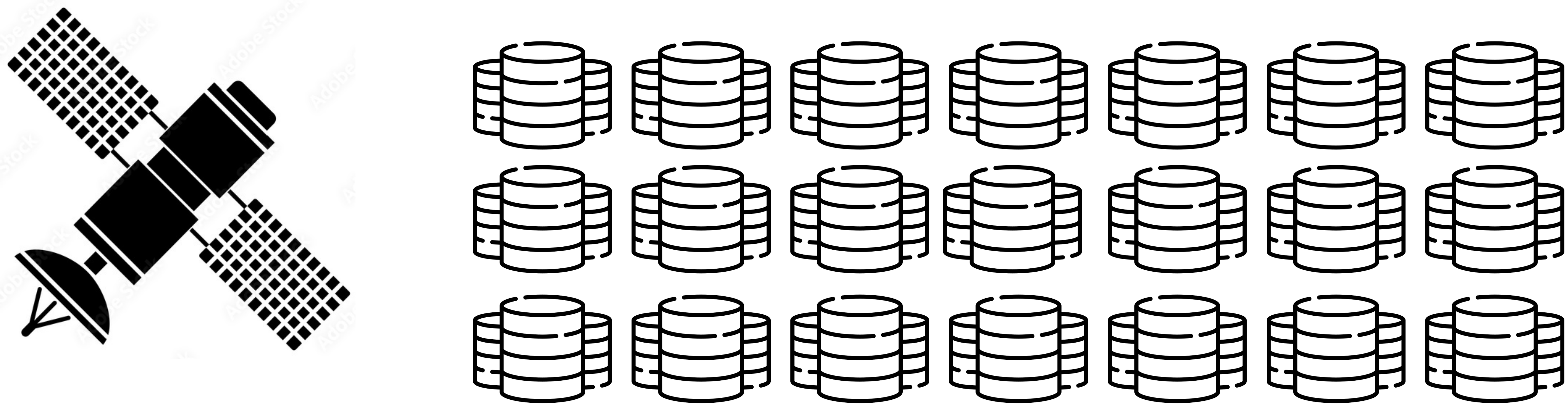


- X-ray emissions from astronomical objects describe the radiation under different scenarios.

- **Model-fitting** is used to extract the underlying physical parameters of X-ray spectra.

- Automatic ways for few standard models using grid-search, e.g. XSPEC.

X-ray Spectra



Limitations:

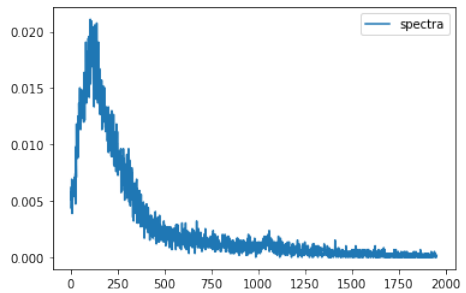
- **Local minimum** could be found for complex spectra features.
- **Computational time** scales exponentially with the number of parameters and the number of model components.

➡ **Its usage is limited for a growing number of available spectra !**

Could we use neural networks to infer physical parameters from X-ray spectra?

Spectra Fitting vs Neural Networks

- **Spectra fitting :**
  - Simplify the problem using physical model that we can **understand.**
  - Fit the data to find the parameters that minimise an objective function.
- **Neural Networks:**
  - Define a very **complex non linear relationship** between input and outputs.
  - **Training:** Fit the data to find the parameters that minimise an objective function.
  - **Testing:** Predict the output of new input data almost **instantaneously**.



→ **Neural Network** → **Predicted Spectra Parameters** ↔ **True Spectra Parameters**

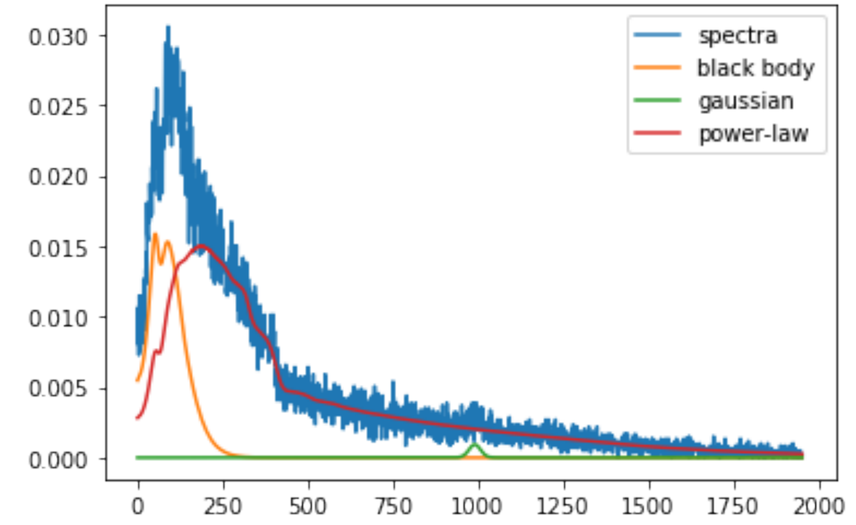
Data Generation

Richard Saxton

- Simulated spectra from the **Active Galactive Nuclei** using **XSPEC**.
- We include 30 different **backgrounds** taken from XMM-Newton observations.
- The spectral model is defined as:

$$M = wabs * (pow + gaus + bbody)$$

- We randomise the parameters over the ranges

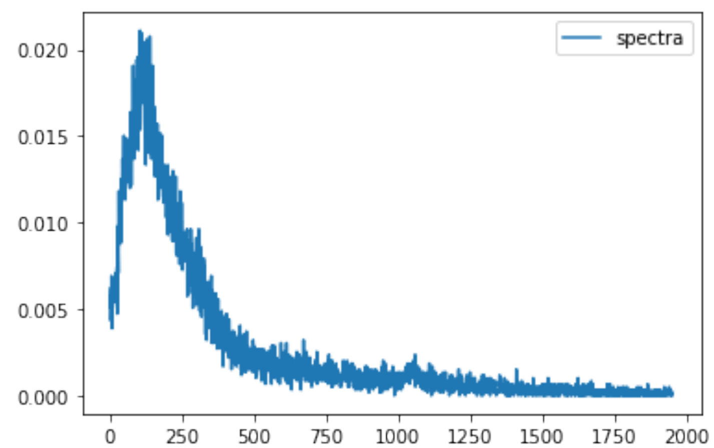


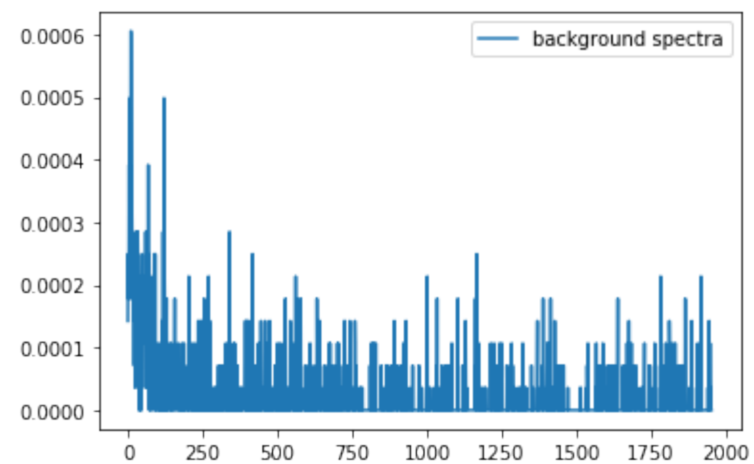
Absorption	Power-law	Gaussian	Black-body
wabs=0.01-1.0	pow_slope=1.0-3.0 pow_norm=4.0E-5 - 4.0E-3	gaus_energy=5.0-7.0 gaus_norm=1.0E-6 - 5E-5	bbody_kT=120-200 eV bbody_norm=1.0E-6 - 5E-5

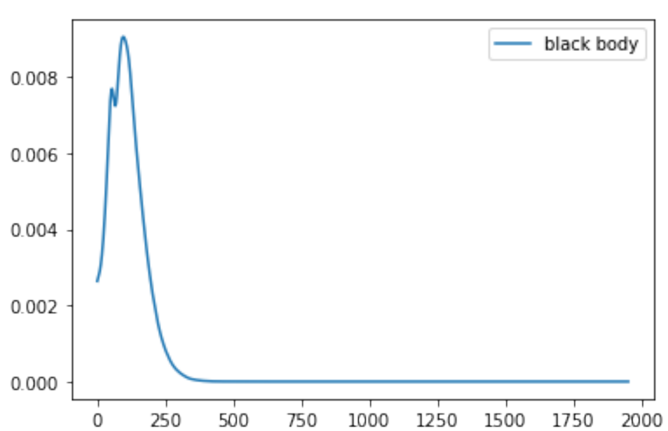
Data Generation

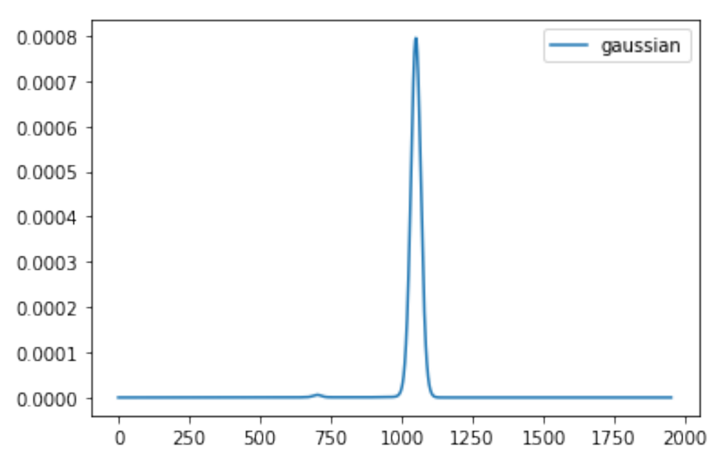
The "fakeit" command produces:

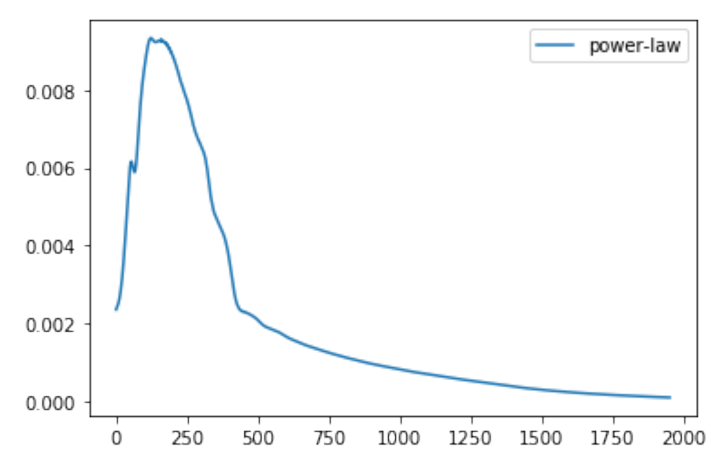
- Source+background spectrum with poissonian statistics.
- The corresponding background spectrum with poiossonian statistics.
- Pow*wabs, bbod*wabs, gauss files without a background and without statistics.



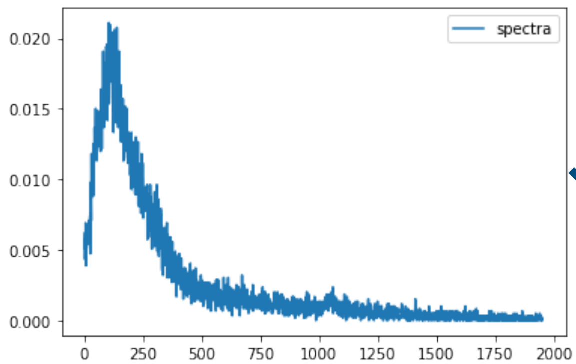


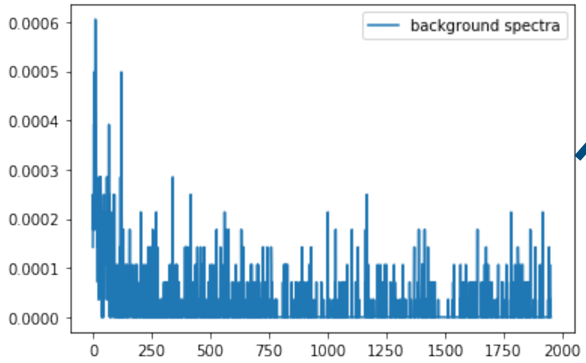






Objective





Neural Network

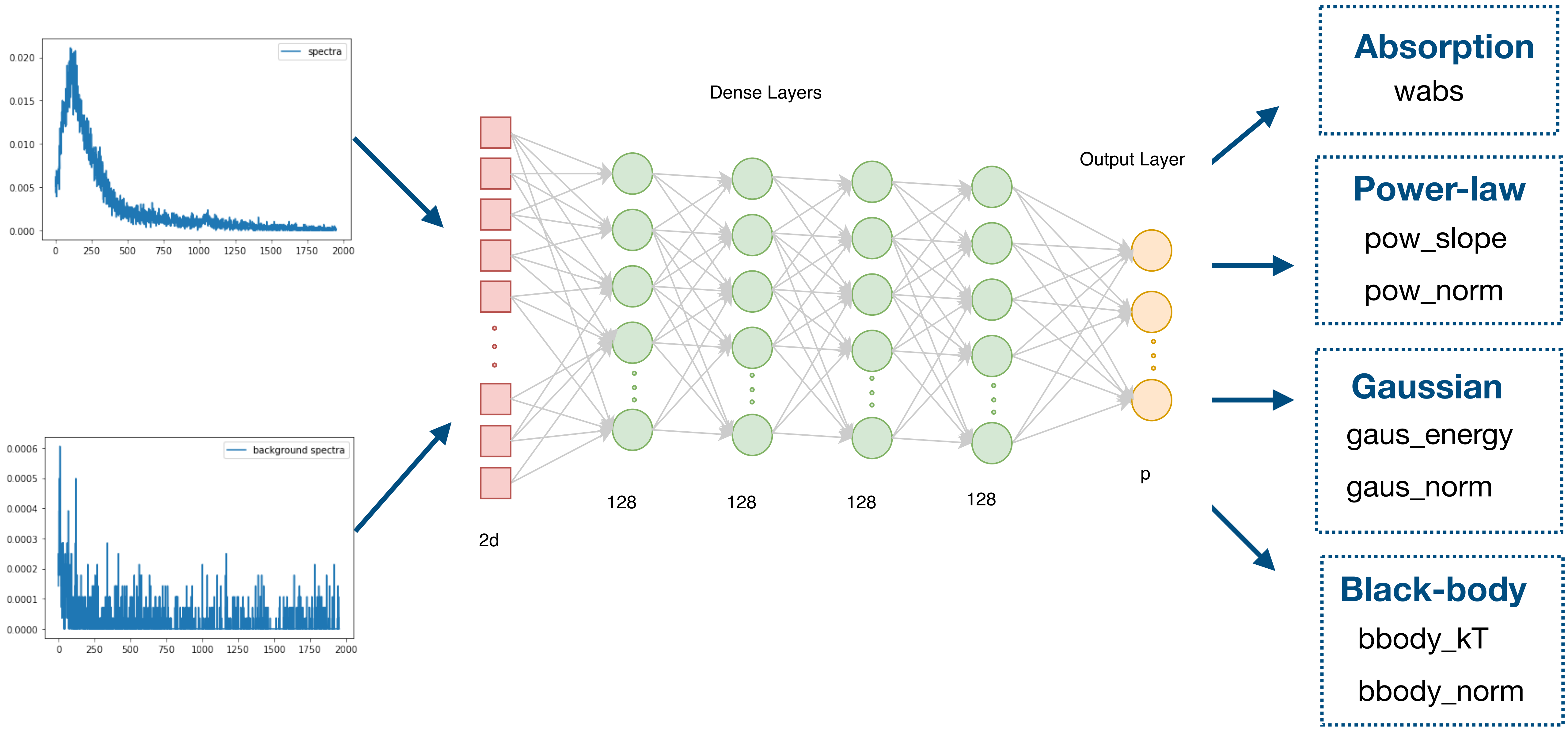
Absorption
wabs

Power-law
pow_slope
pow_norm

Gaussian
gaus_energy
gaus_norm

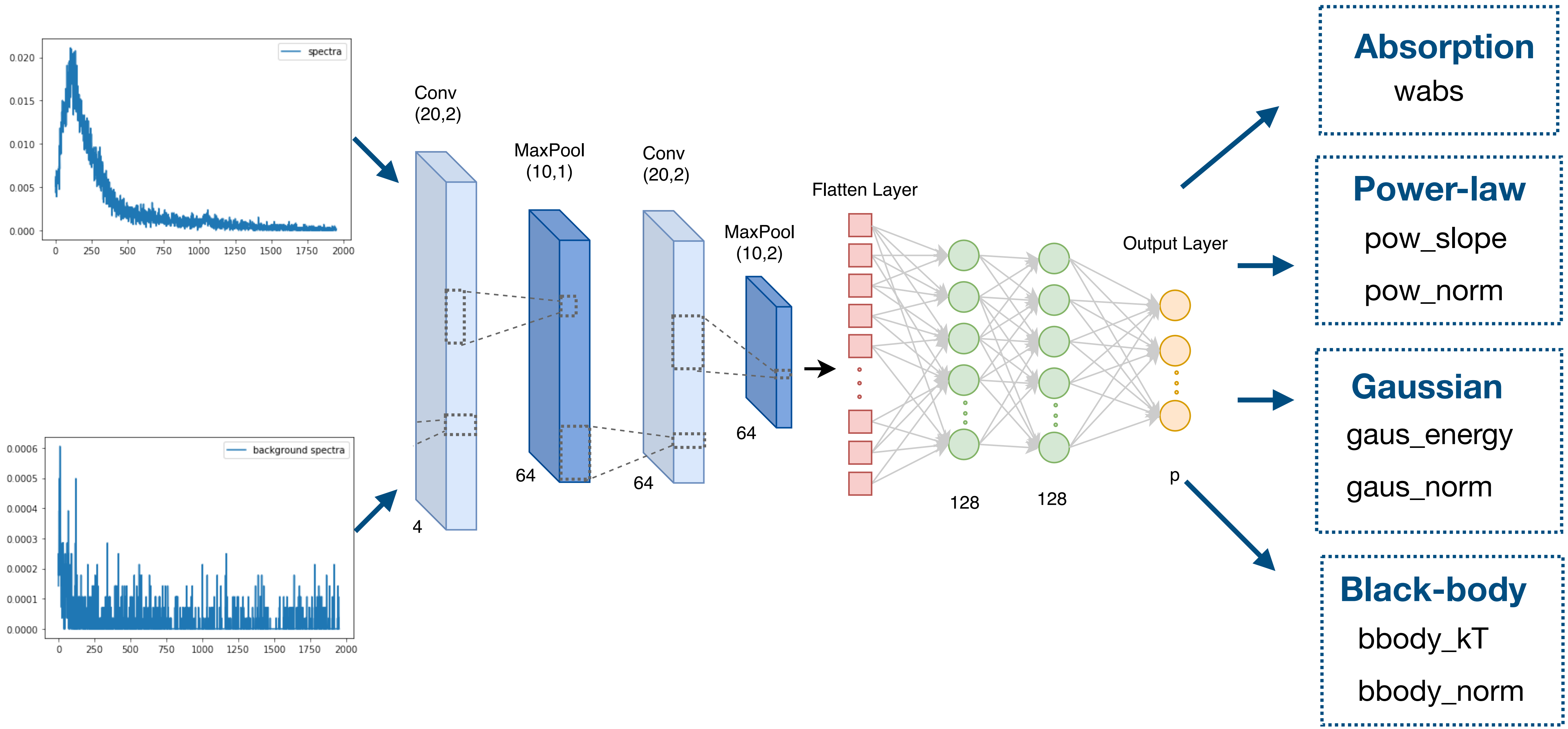
Black-body
bbody_kT
bbody_norm

MLP



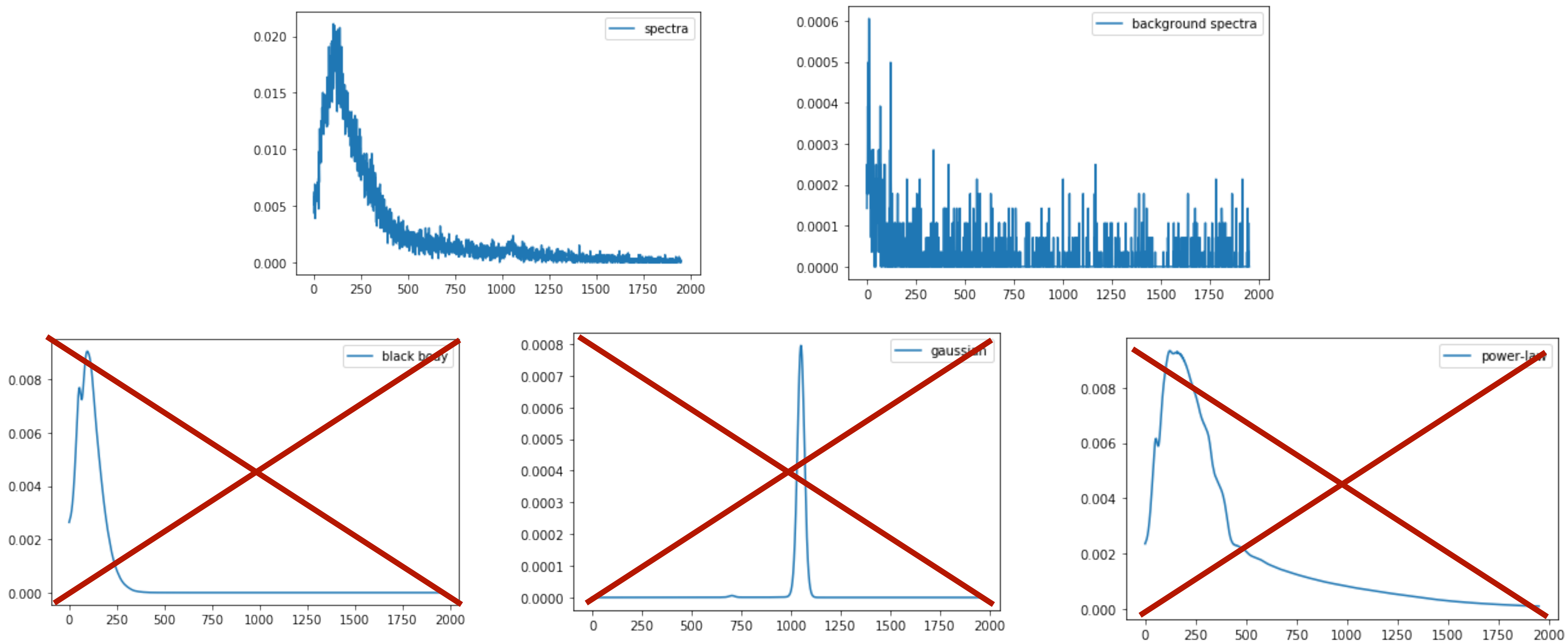
spectra
background spectra
Dense Layers
Output Layer
2d
128
128
128
128
p
Absorption
wabs
Power-law
pow_slope
pow_norm
Gaussian
gaus_energy
gaus_norm
Black-body
bbody_kT
bbody_norm

CNN


spectra
background spectra
Conv
(20,2)
4
MaxPool
(10,1)
64
Conv
(20,2)
64
MaxPool
(10,2)
64
Flatten Layer
128
128
Output Layer
p
Absorption
wabs
Power-law
pow_slope
pow_norm
Gaussian
gaus_energy
gaus_norm
Black-body
bbody_kT
bbody_norm

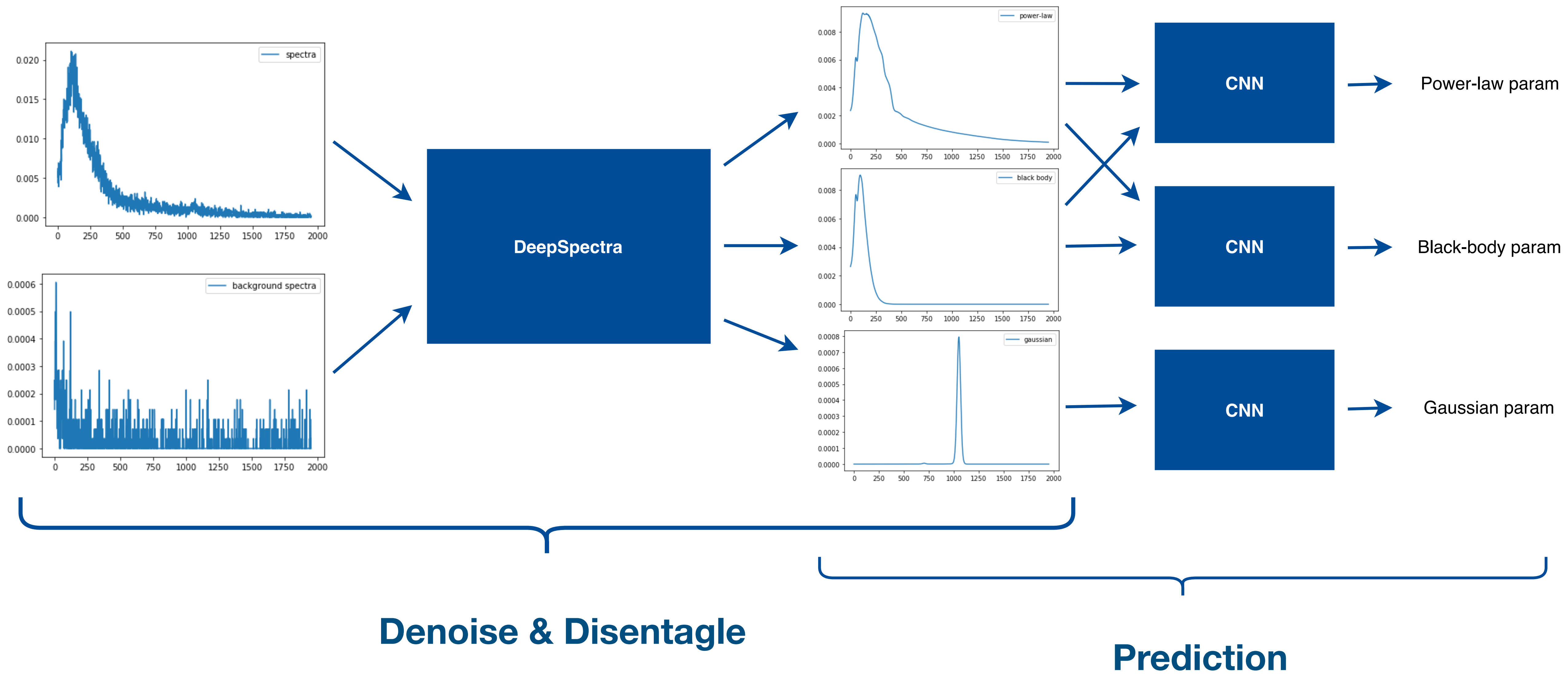
Architecture: Limitations

- We are not using all the information that we have.



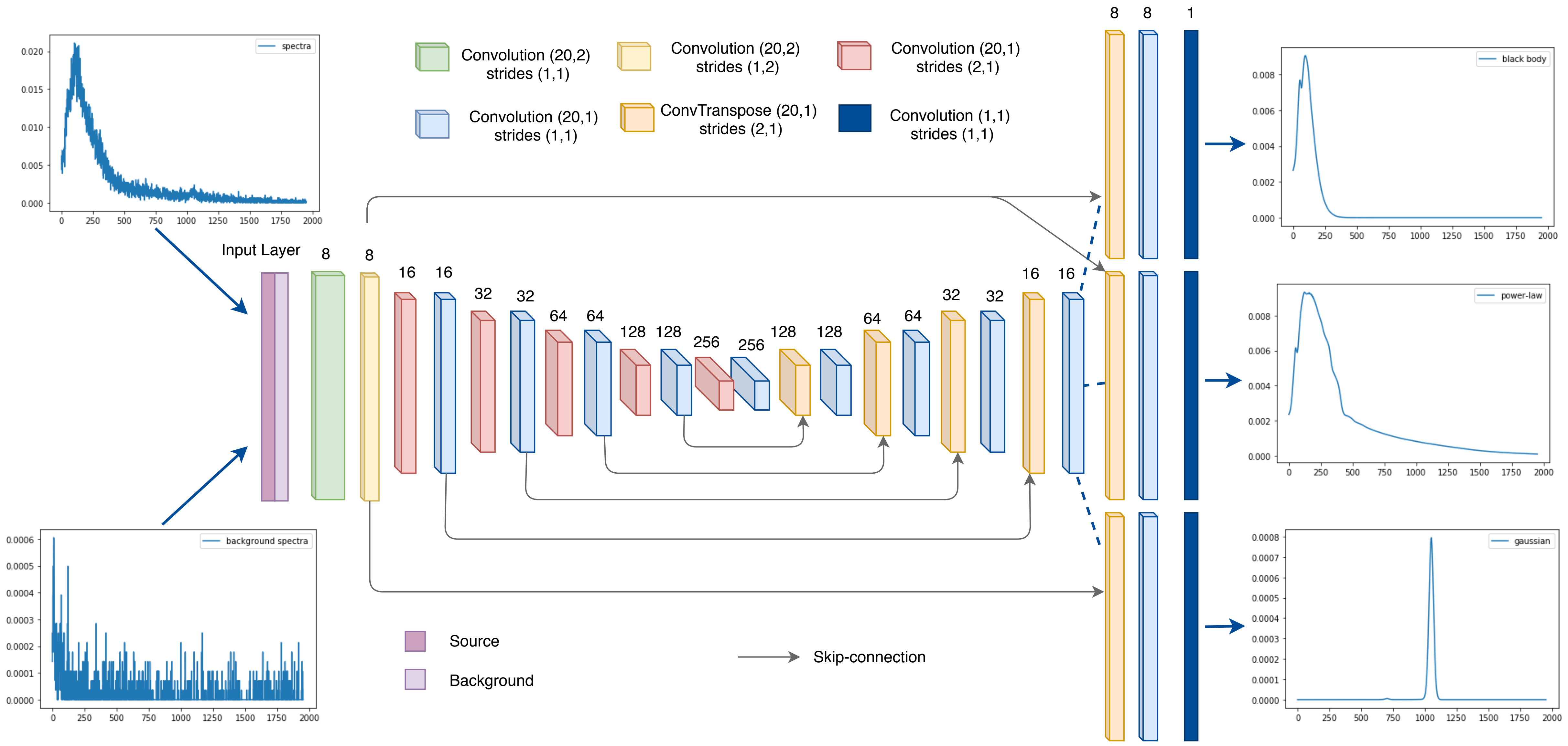
- **Idea:** enforce the algorithm to decompose the spectra first and then infer the parameters.

DeepSpectra



Note: the true spectra components are used as labels to **train** the network and they are **not** available for **testing**.

DeepSpectra



Skip-connection: Combine fine layers and coarse layers to make local predictions that **respect global structure.**

Results: accuracy

MFE	Power-law			Gaussian		Black-body	
	wabs	slope	norm	energy	norm	kT	norm
SpectraFitting	5.95%	2.33%	5.33%	2.65%	33.46%	9.28%	77.83%
MLP	10.5%	2.60%	8.41%	1.29%	18.28%	6.1%	46.8%
CNN	8.78%	2.00%	7.28%	0.89%	12.47%	5.61%	40.6%
DeepSpectra	**4.85%**	**1.47%**	**4.65%**	**0.56%**	**10.4%**	**5.58%**	**40.3%**

Table 2: Prediction performance using Mean Fractional Error.

$$\mathrm{MFE} = \frac{1}{n}\sum_{j=1}^{n}\left|\frac{y_j - \hat{y}_j}{y_j}\right|$$

$y_j = true \;\; spectra \;\; parameter$

$\hat{y}_j = predicted \;\; spectra \;\; parameter$

Results: accuracy

RMSE	Power-law			Gaussian		Black-body	
	wabs (10^{-2})	slope (10^{-2})	norm (10^{-5})	energy (10^{-1})	norm (10^{-6})	kT (10^{-2})	norm (10^{-6})
SpectraFitting	4.53	15.0	9.82	4.70	11.0	2.42	15.5
MLP	3.62	7.62	9.95	1.60	2.8	1.31	7.49
CNN	3.00	5.67	8.03	1.27	1.99	**1.21**	**6.89**
DeepSpectra	**2.72**	**4.96**	**6.25**	**1.23**	**1.81**	**1.20**	**6.83**

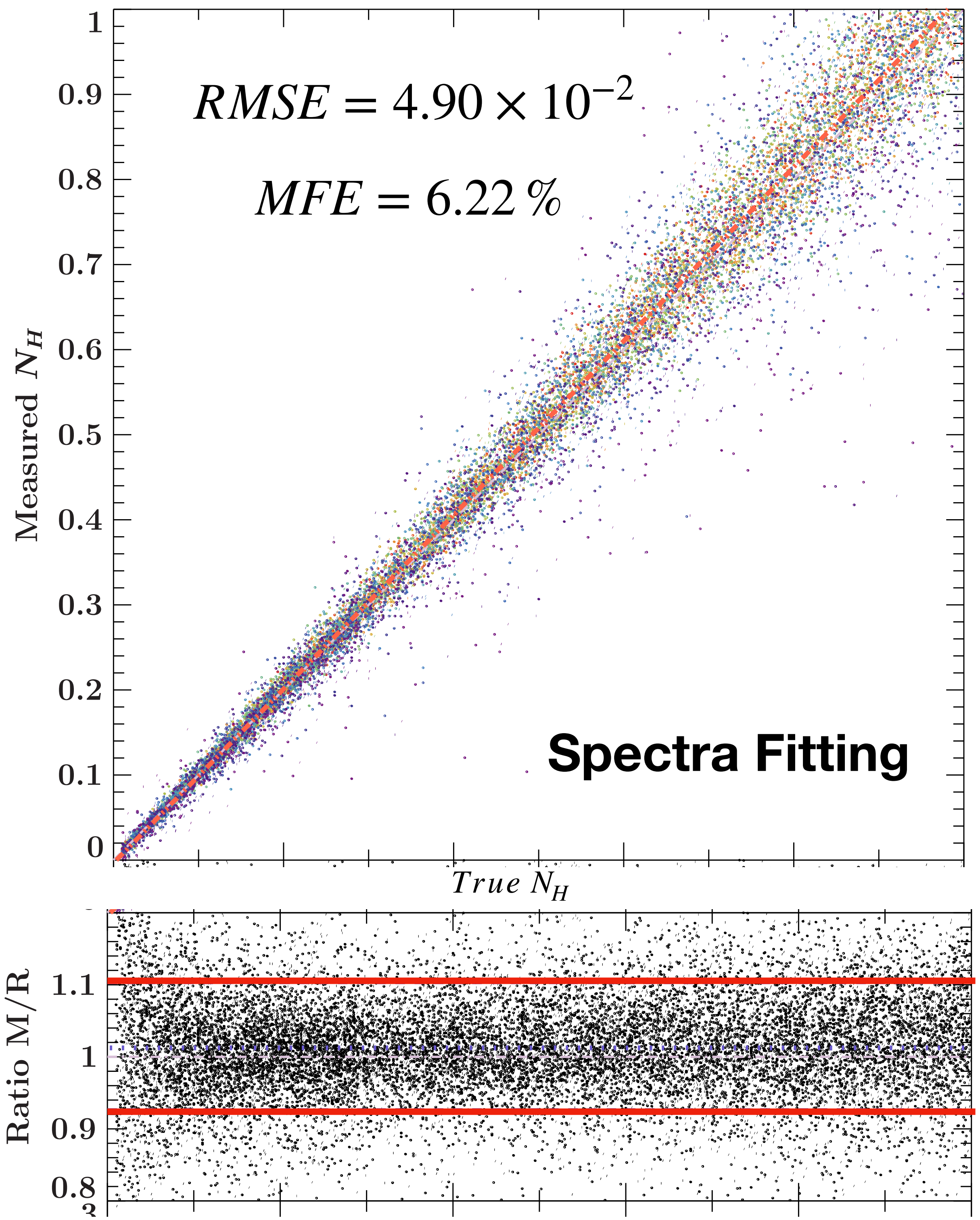
Table 1: Prediction performance using Root Mean Squared Error.

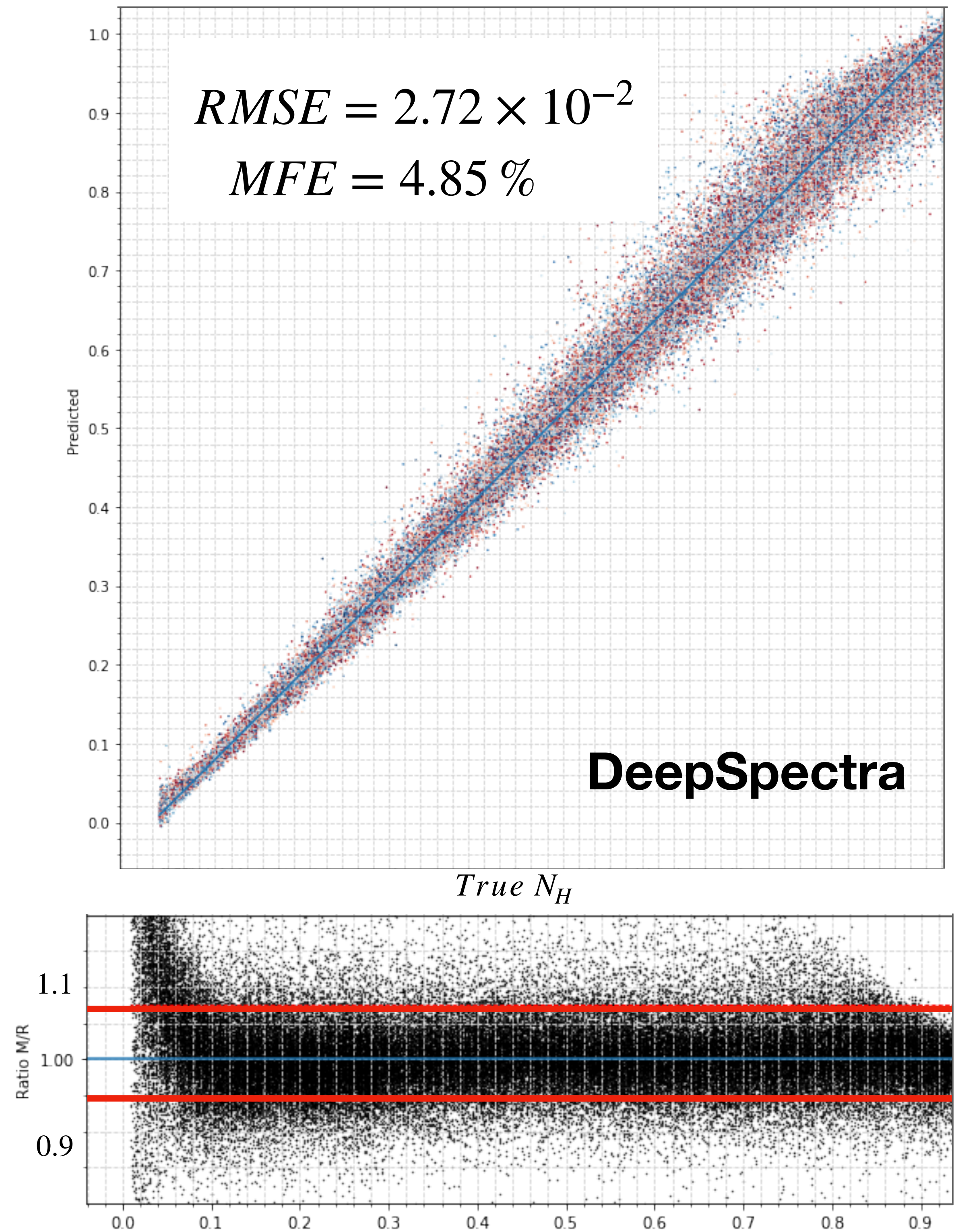
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{y}_j\right)^2}$$

$$y_j = true \quad spectra \quad parameter$$

$$\hat{y}_j = predicted \quad spectra \quad parameter$$

Results: SpectraFitting vs DeepSpectra



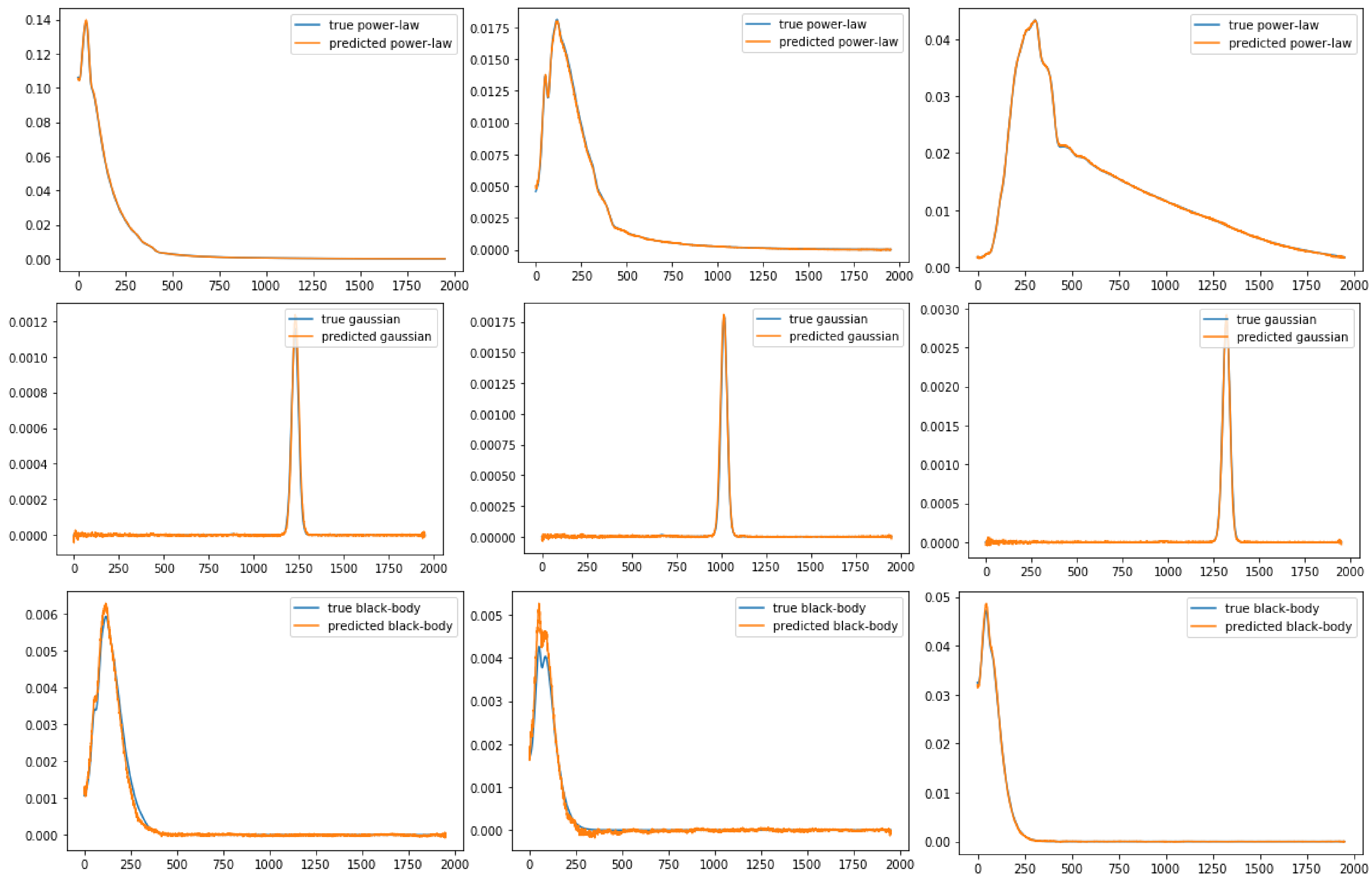


Results: Generated Components

Power-law

Gaussian

Black-body



Results: Computational Time

NEURAL NETWORKS:

Training:

- **MLP** ~ 3 hours
- **CNN** ~ 4 hours
- **DeepSpectra:** ~ 10 hours

Testing

MLP / CNN / DeepSpectra:

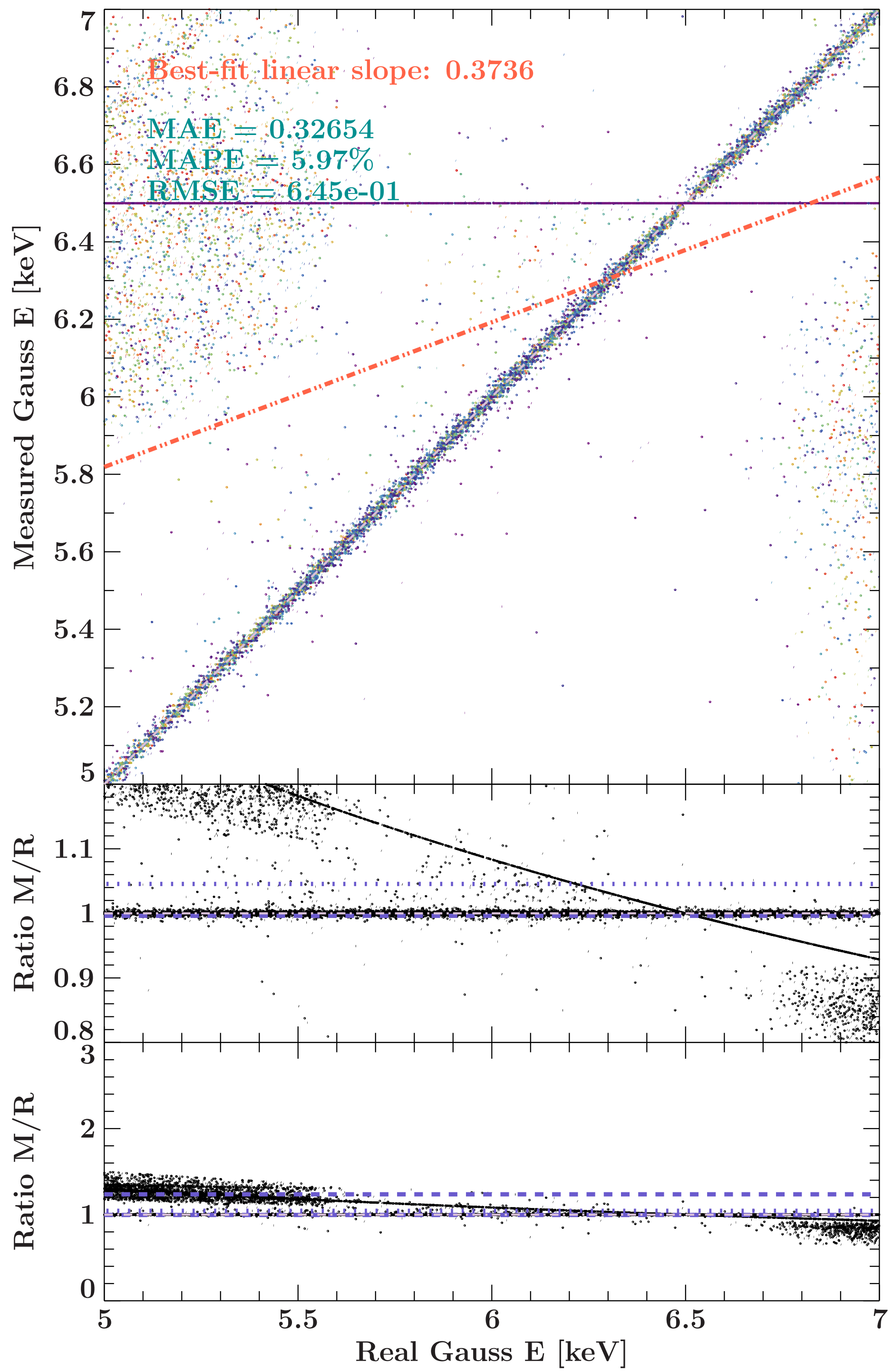
~ few seconds for 20000 samples

SPECTRA FITTING: ~ 27 hours for 20000 samples

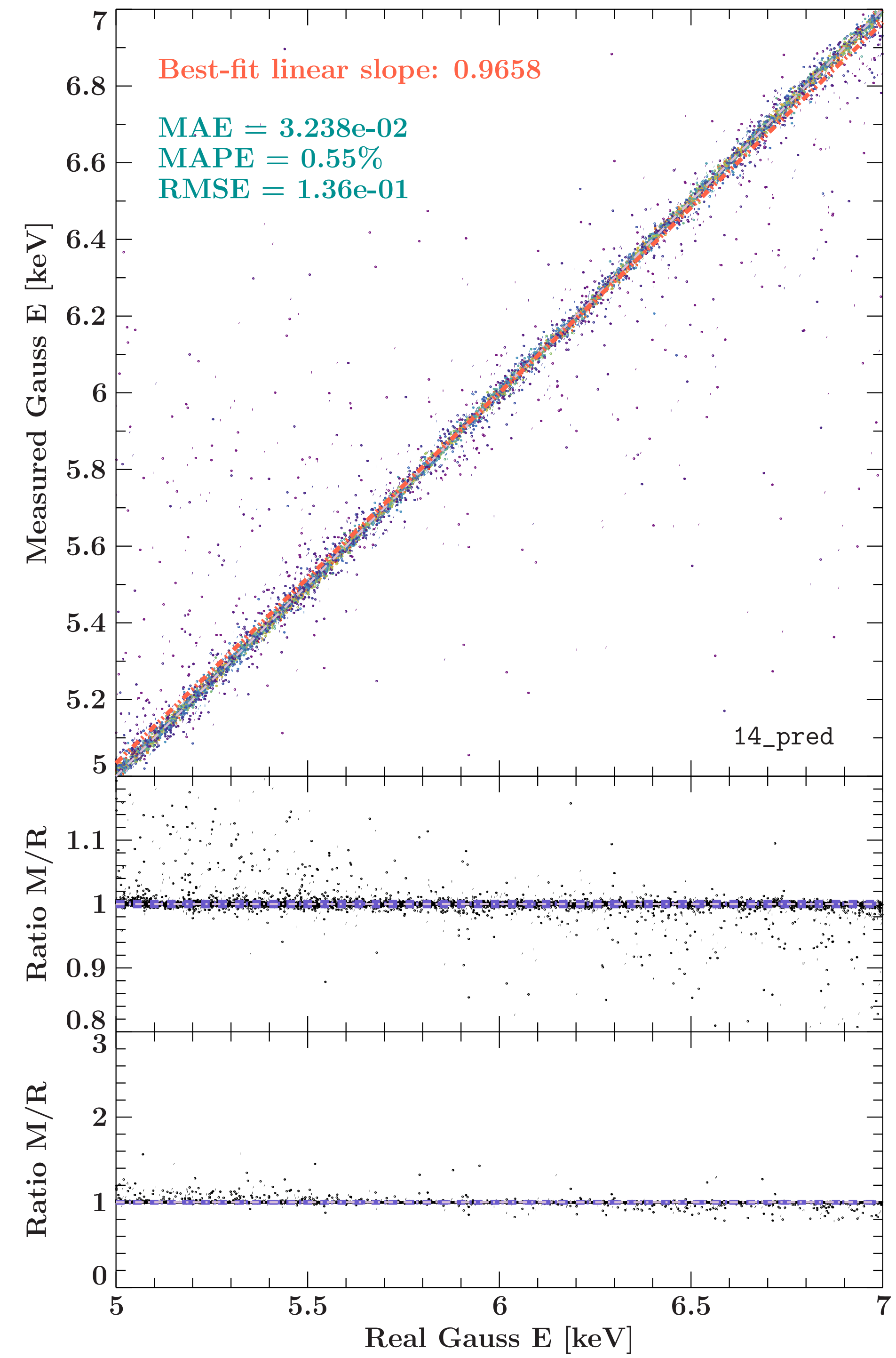
Results: Spectra Fitting

Initial parameters value:

Random



DeepSpectra predicted parameters



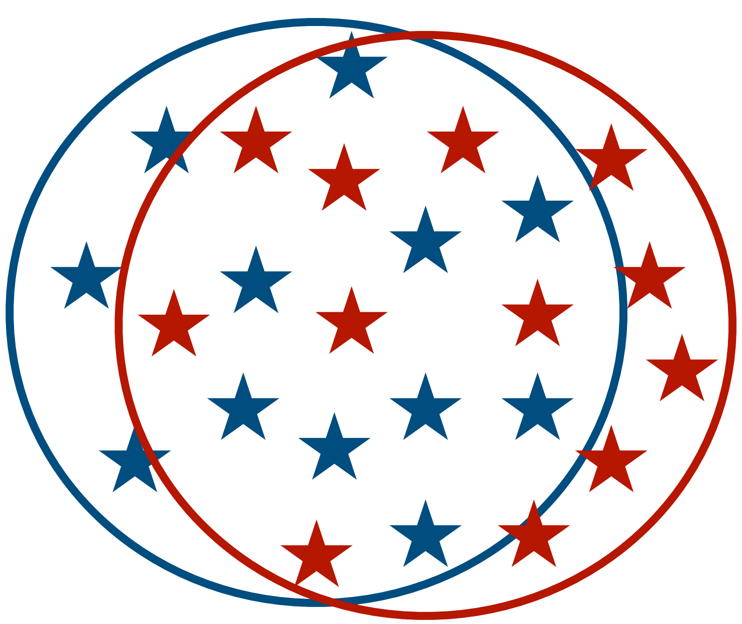
Felix Fuerst

Conclusion

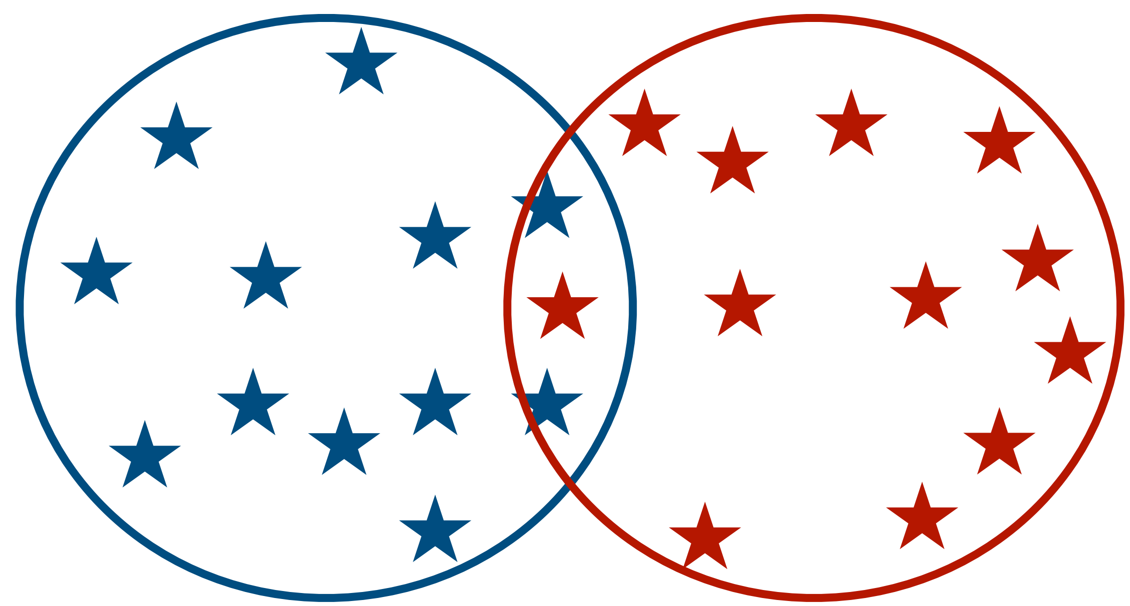
- Neural networks could be used in spectra analysis to speed up computational time and to increase the accuracy of the results.
- **Basic architectures** such as MLPs represent an improvement over the standard spectra fitting routines for certain parameters.
- **CNN** shows improvements over the MLP baseline.
- **DeepSpectra** outperforms Spectra Fitting, the baseline and the CNN and it successfully denoises and disentagles different spectra components.

Final Goal: Real Observations

- **Application on real observations from XMM-Newton Catalogue.**
- Results on real data are still not satisfactory:
  - Real-world observations are noisy.
  - **Distributional shift**: train and test dataset have different distributions.



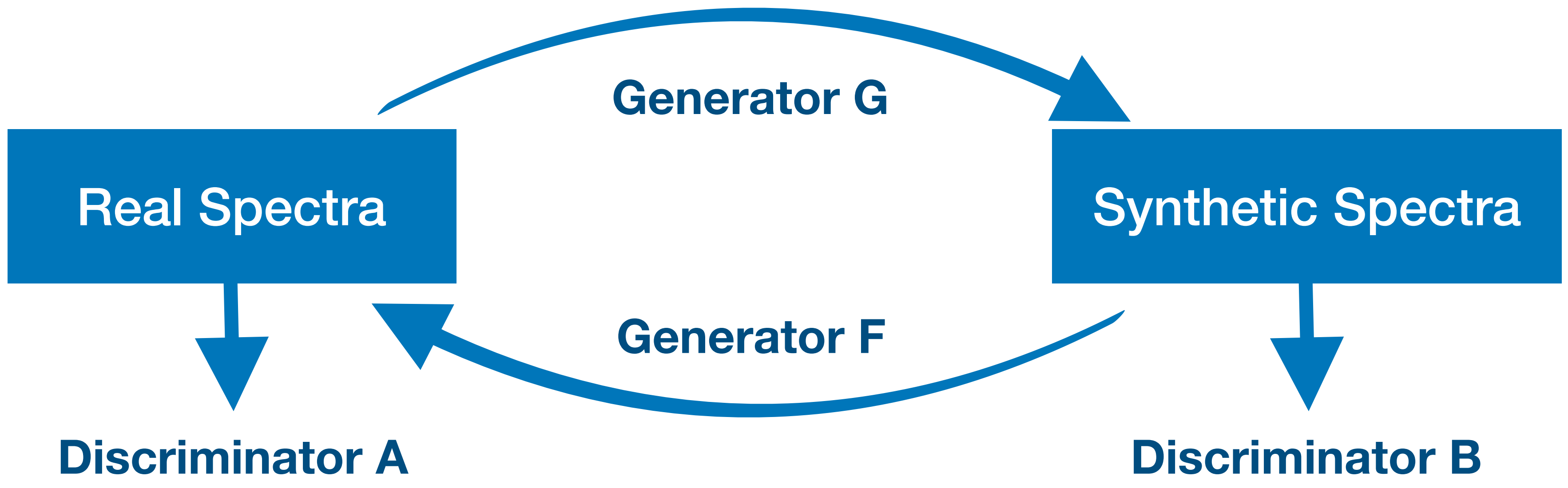
Training data Test data



Training data Test data

Future Work

- **Data:**
    - Use approximate labels to train on both real and synthetic data.
    - Pre-processing of the real observations.
    - Real-to-synthetic translation model (e.g., Cycle GAN).
- **Model:**
    - Increase robustness through augmentations.
    - Use a semi-supervised model.



Questions?