Application of Machine Learning Techniques for the retrieval of Cloud Properties for the Copernicus Satellites Sentinel-4 (S4) and TROPOMI / Sentinel-5 Precursor (S5P)

Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

Fabian Romahn[1], Diego G. Loyola[1], Victor Molina Garcia[1], Ronny Lutz[1]
(1) German Aerospace Center (DLR)

1. Why Machine Learning for processing data of Copernicus Satellite Sensors?

- The amount of data from remote sensing satellites that has to be processed, dramatically increased in the recent years, especially with the Copernicus program
- The processing is even further challenging since there are near real time (NRT) requirements for many products
- Therefore, the retrieval algorithms not only have to be accurate, but also very fast aswell
- In recent years, the application of machine learning techniques, especially neural networks, has become increasingly popular in order to improve the performance of classical retrieval algorithms
- A successful example is the use of neural networks for the retrieval of the operational CLOUD product of the Sentinel-5 Precursor satellite (S5P)

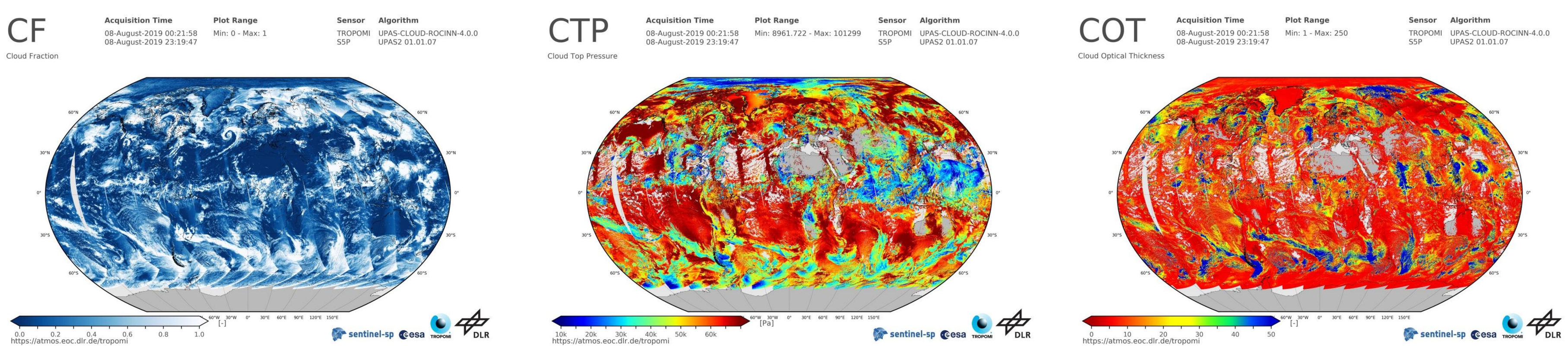

Figure 1.1: cloud fraction (CF), cloud top pressure (CTP) and the cloud optical thickness (COT) from the operational S5P CLOUD product from 08-08-2019

2. Inversion with a radiative transfer model vs inversion with a neural network

- Atmospheric retrieval can often be formulated in terms of mathematical inversion problems
- There, the goal is to find a set of parameters x that minimize the residual $\|F(x) - y\|_2$ between a known vector y and the mapping of the parameters $F(x)$ - where F is a predefined function
- In the context of atmospheric retrieval algorithms x then represents the state of the atmosphere, y a measured spectrum and F a radiative transfer model (RTM) that predicts the spectrum $F(x)$
- For the inversion algorithm the specific implementation of F is not relevant – it can be a complex RTM or a fast neural network (NN)
- **GODFIT** (**GO**ME **D**irect **FIT**ting) is an example for an inversion algorithm with a RTM as forward model
  - It produces the S5P ozone total column product, is computationally very expensive but has no NRT requirements
- **ROCINN** (**R**etrieval **O**f **C**loud **I**nformation using **N**eural **N**etworks) is an example for an inversion algorithm with NNs as forward models
  - It is part of the S5P CLOUD product and has strict NRT requirements

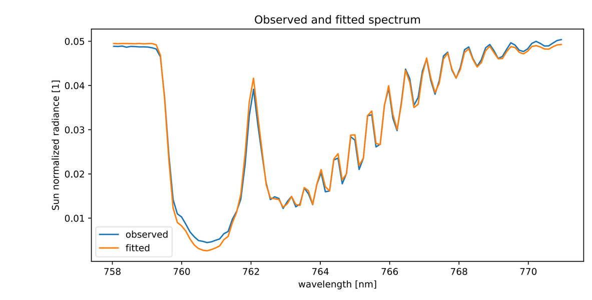

Figure 2.1: Example of an observed and fitted spectrum in the O_2 A-band – the fitted spectrum is a linear combination of a clear sky- and fully cloudy spectrum weighted by the cloud fraction

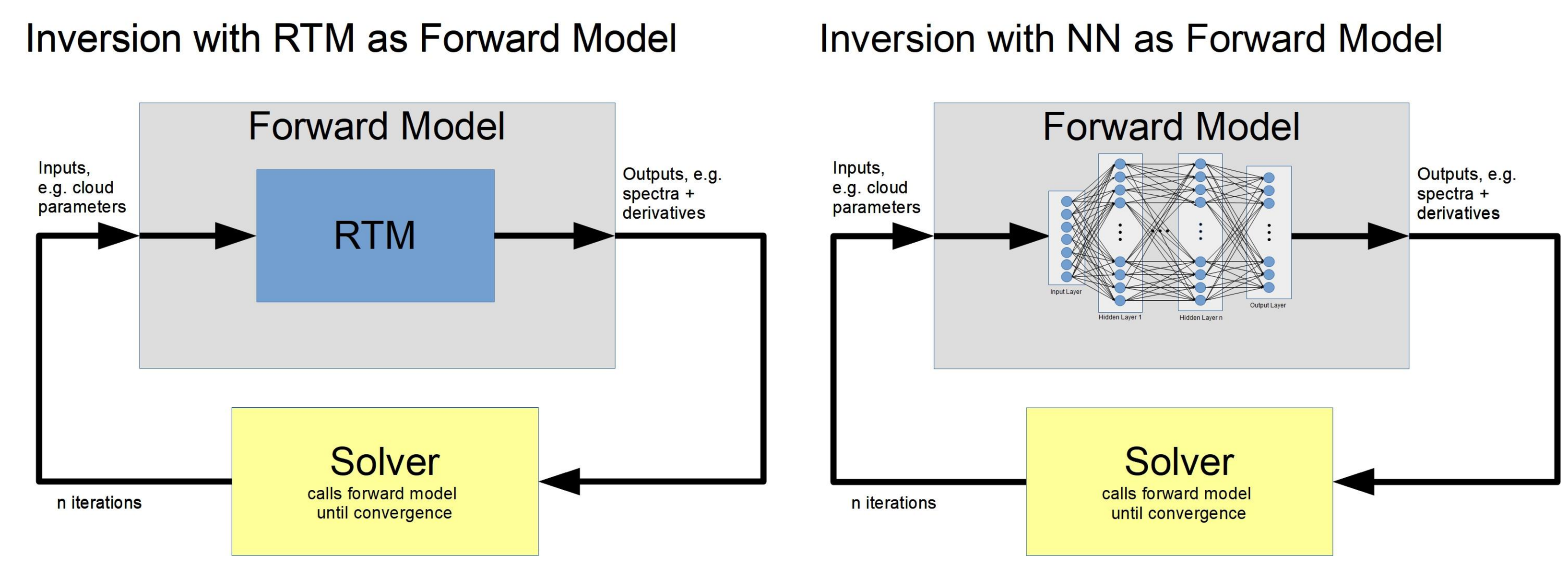

Figure 2.2: The solver can either use a RTM or a NN as forward model

3. How to get from a radiative transfer model to a neural network?

- In order to replace the RTM of an inversion algorithm by a NN a general method was developed which is applicable to arbitrary RTMs and thus can be used for many retrieval algorithms

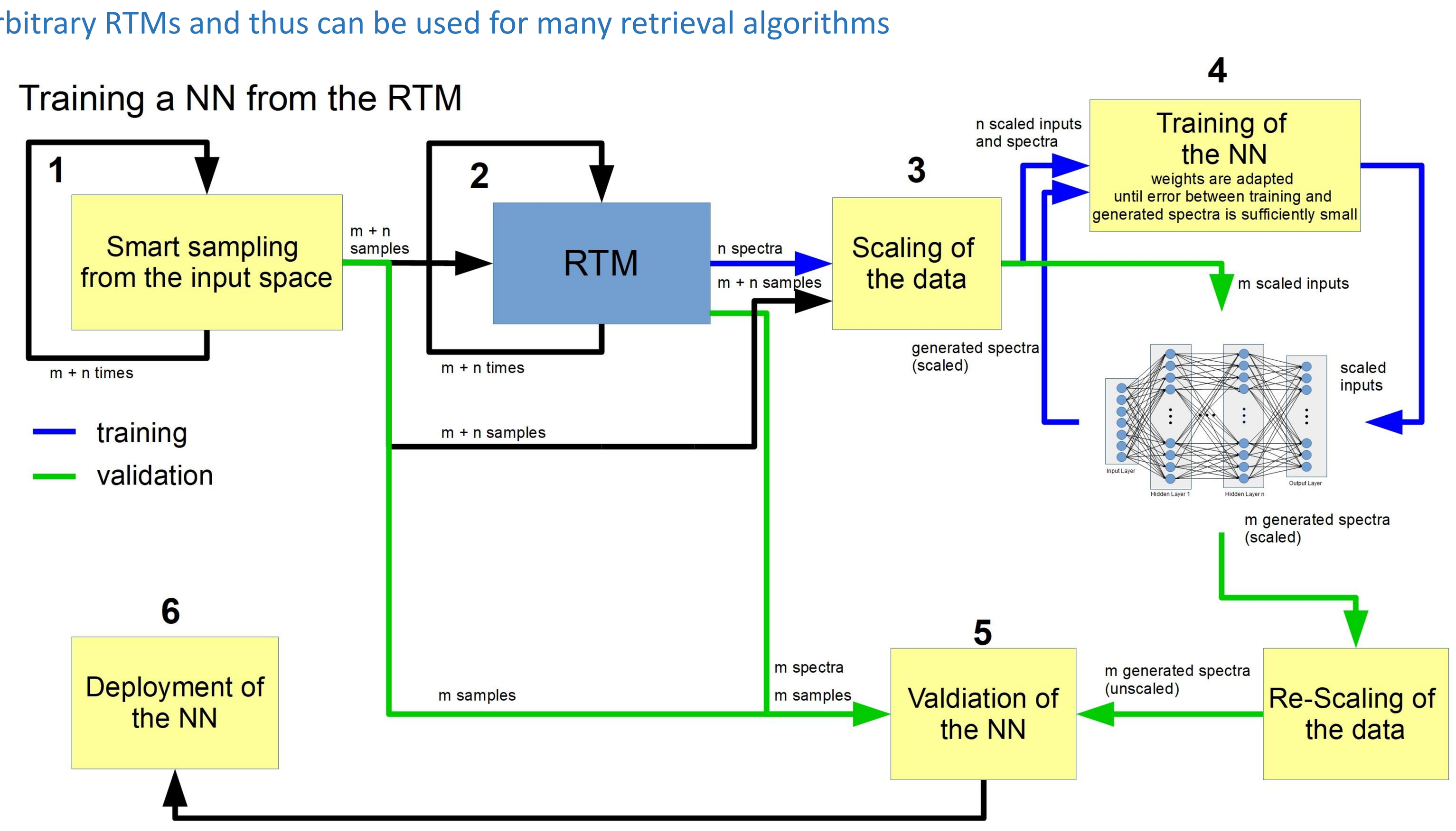

Figure 3.1: Illustration of the complete NN lifecycle – from data sampling to deployment

- It consists of the following steps:
  1. **Smart sampling:**
     The training data needed for the NN consists of input / output pairs. In case of the ROCINN algorithm the input consists of up to seven parameters:
     - Surface parameters (suface height, surface albedo)
     - Geometry (solar zenith angle, viewing zenith angle, relative azimuth angle)
     - Cloud properties (cloud height, cloud optical thickness) – in case of cloudy scenes

     Samples of this input space are chosen with the Halton sequence. Additionally, Importance Sampling can be used to account for the distribution of the different parameters.
  2. **Generation** of the training data:
     - The corresponding outputs are generated using the RTM
     - For ROCINN, these are spectra in the O_2 A-band, calculated by the RTM VLIDORT
     - The final results are then saved (together with the inputs) in a netCDF-4 file
  3. **Scaling** of the data:
     Inputs and outputs of the training data are scaled to the interval [0,1] to improve the stability of the weights during the training process
  4. **Training** of the NN:
     Tools based on keras were implemented which allow:
     - easy definition of the network topology, activation functions and training parameters
     - saving of the network as well as metadata in an hdf5 file
     - iterative training by loading of pre-trained networks
  5. **Validation:**
     After the training, the NN is validated with an indenpendent data set
  6. **Deployment** of the NN:
     A neural network module was developed which implements:
     - Reading of NNs defined in hdf5 files at runtime
     - Transparent scaling of inputs and outputs
     - Computation of the derivatives
     - Support of arbitrary network topologies and different activation functions

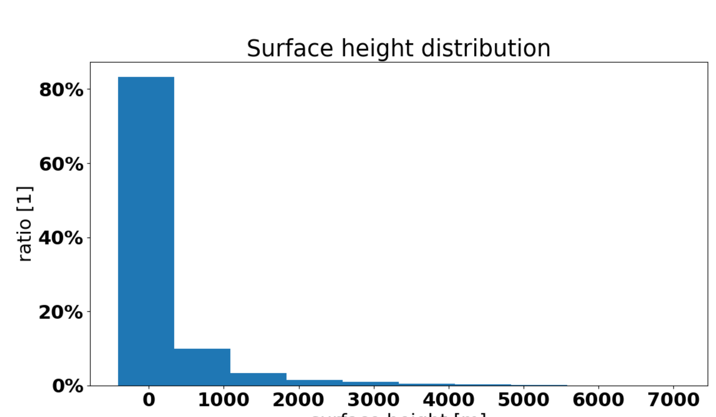

Figure 3.2: Histogram of the surface height from data of the whole earth

4. Evaluation

- Finding a neural network with the optimal performance is very challenging (many aspects impact the results)
- Following key aspects were investigated:
  1. **Scaling** of the data
  2. **Topology** of the network
  3. **Activation Functions** of the neurons
  4. **Sampling** of the parameters for the training set
- Other aspects not (yet) investigated: Optimizer + learning-rate, number of samples, batch size
- For the evaluation Sentinel-4 Clear-sky spectra (5 input, 345 output parameters) were used
- Training settings: 2000 Epochs, batch size: 200, Optimizer: Adam with learning rate 1e-5

1. Scaling

- Scaling the inputs and/or outputs is very important as it influences the stability of the weights during training

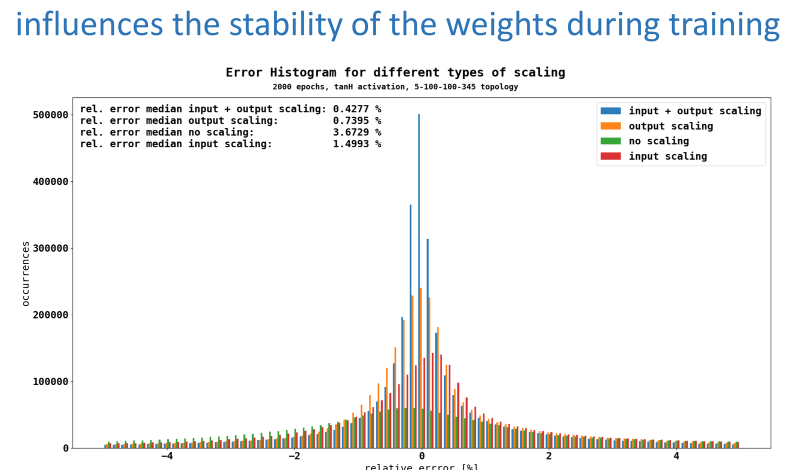

Figure 4.1: Relative errors of NNs trained with input and/or output scaling ar no scaling at all

→ Scaling both, inputs and outputs, is necessary

2. Topology

- Selecting the network topology is hard as it is very arbitrary (key factors: complexity, number hidden layers)

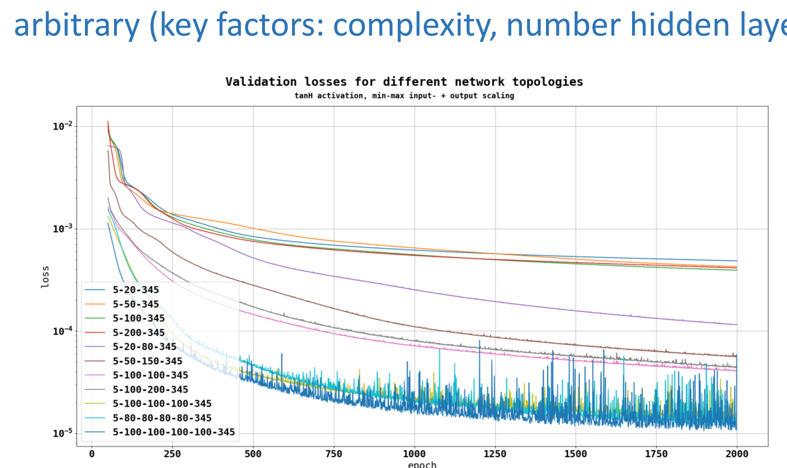

Figure 4.2: Validation losses during training for different NN topologies

→ 1 hidden layer is insufficient, 3 – 4 hidden layers with ~40000 parameters (5-80-80-80-80-345) is a good choice

3. Activation Functions

- Activation functions are responsible for nonlinearity of the NN
- The following activation functions were evaluated: relu, sigmoid, tanh

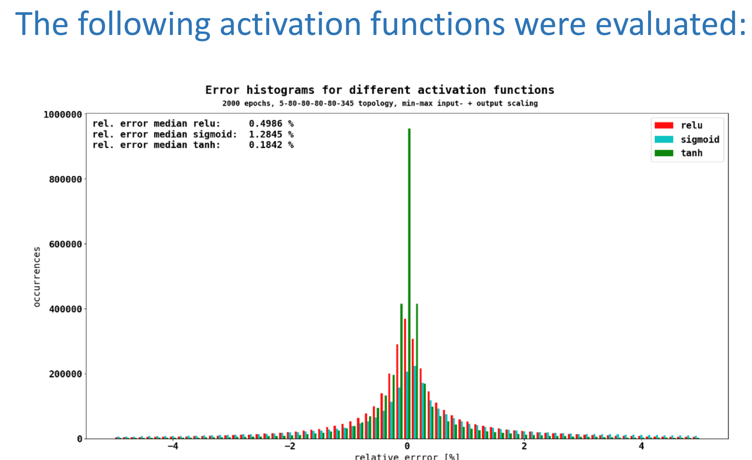

Figure 4.3: Relative errors of NNs with different activation functions

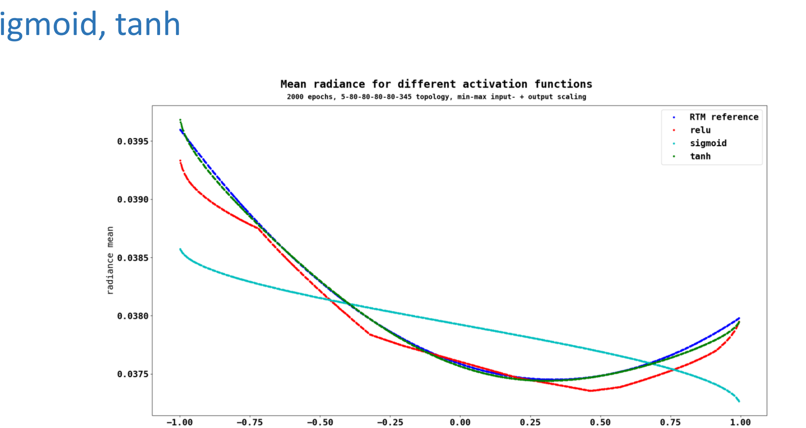

Figure 4.4: Mean spectra-radiance depending on the cosine of the relative azimuth angle for NNs with different activation functions

→ tanH is the preferred activation function: best relative error (Figure 4.3), smoothest, no discontinuities (Figure 4.4)

4. Sampling

- Parameter distribution in training set is significant → Importance Sampling
- Comparison for surface height with actual vs. Uniform distribution

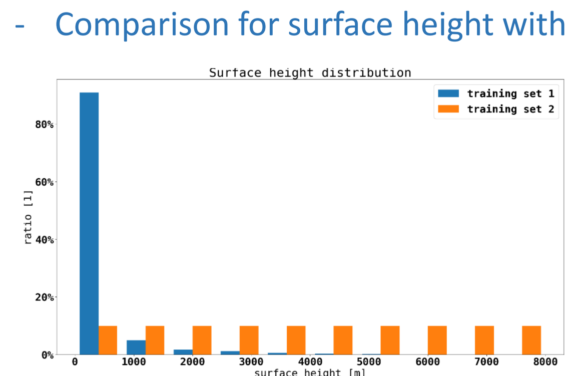

Figure 4.5: Surface distribution for the two training sets

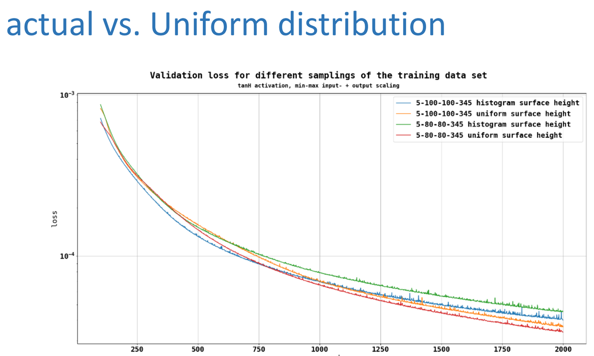

Figure 4.6: Validation loss during training for NNs with different training sets

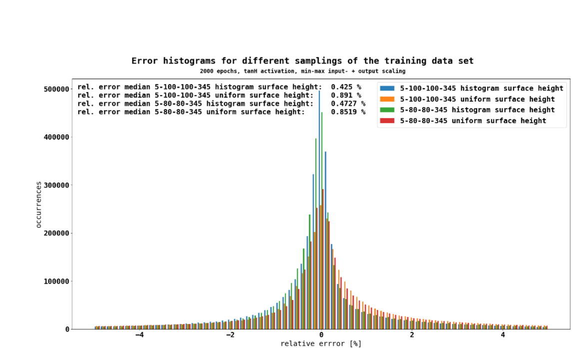

Figure 4.7: Relative errors of NNs with different training sets

→ Desipite worse validation loss during training, NNs trained with actual parameter distribution lead to better results

5. Application

- Neural networks for S5P already used for operational CLOUD product
- Neural networks for S4 implemented in first version of CLOUD processor
- Current cleark-sky neural network for S4 has slightly better performance than clear-sky network for S5P, cloudy network for S4 needs further improvement
- Different NNs for different scene types and cloud models are in use (clear-sky, cloudy CRB, cloudy CAL), development for other cloud types (ice-clouds) is ongoing

Model	# parameters	exec time	exec. time with gradient
RTM Vlidort, 32 threads	-	17h, 9min, 0.625s	-
S5P Clear Sky (5-100-100-107)	21507	0.62s	3.48s
S4 Clear Sky (5-80-80-80-80-345)	47865	3.28s	14.67s

Table 5.1: Computational performance for generating 250000 spectra

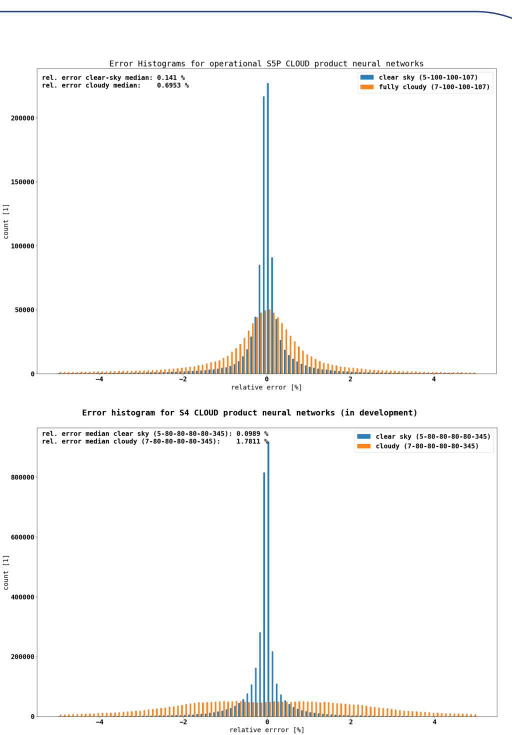

Figure 5.1: Relative errors for S5P (top) and S4 (bottom) NNs (clear-sky + cloudy)

6. Conclusion

1. Neural networks offer a way to **drastically increase the performance** of classical retrieval algorithms
   - Many orders of magnitudes faster than RTMs, successful use in CLOUD product for S5P (operational) and S4
2. Neural network lifecycle chain offers **general procedure for replacing RTM by NN**
   - Allows use of specific NNs for specific problems (e.g. for different types of clouds)
3. Neural networks can provide **sufficient accuracy** to replace RTM
   - finding best structure configuration is challenging, approaches to evaluate and determine well suited structures have been presented, more investigations are ongoing
4. Neural netowrks allow for **new possibilities for inversion algorithms**
   - Computational performance allows many foward model calls, gradients available
     → possibility for global optimization techniques