NeurIPS 2022 Ariel Data Challenge Solution Description

Stefan Stefanov

The solution method is to model each target variable distribution(planet_temp, log_H2O, etc.) as an independent mixture model of a normal and a uniform distribution. The mixture model parameters are fit to the trace data using Expectation-Maximization (EM). The mixture model parameters are predicted by a neural network. Multi-task learning is applied for neural network training - in addition to mixture model parameters the neural network is predicting FM parameters, quartiles for the light track and performs spectrum reconstruction. Semi-supervised learning is used where the network is also trained to reconstruct the spectrum on the test data. The predictions for the regular track are sampled from the mixture model using the neural network predicted parameters. The light track quartiles are directly predicted by the neural network. The final submission is an ensemble of 10 such models. The regular track submission is constructed from one tenth of each model samples. The light track submission is using the mean of quartiles predictions of the individual models.

1. How did you use the training data?

Tracedata is used to fit mixture model parameters using Expectation-Maximization implemented by external library `pomegranate`(https://github.com/jmschrei/pomegranate).
QuartilesTable is used for training the neural network to predict light track quartiles.
FM_Parameter_Table is used for training the neural network to predict FM parameters.
AuxillaryTable features are used as additional input to the model
Both train and test SpectralData is used as input and target to train the model to reconstruct the spectrum.

2. Did you perform any data preprocessing step?

The spectrum values are clipped to max 0.5 as there are extreme values for some planets.
The spectrum mean is subtracted from the spectrum and the result is used as an additional model input.
Deep Adaptive Input Normalization (DAIN) layer(https://github.com/passalis/dain) is used as part of the neural network for trainable normalization of the spectrum input.
Standard scaling is used for the auxiliary features.
Targets are normalized using min-max scaling according to default prior bounds.

3. What kind of model did you go for?

The model is an autoencoder neural network consisting of encoder and decoder components. The encoder component is using a convolutional block based on ResNet architecture adapted for 1D to encode the spectrum data. The output of the convolutional block is combined with the auxiliary features and fed into a dense block. There are multiple heads on top of the dense block for predicting multiple outputs: mixture model parameters, FM parameters and quartiles. These outputs of the encoder are fed into the decoder using 1D transposed convolution layers to reconstruct the spectrum.

4. What is the input/output of the model?

The inputs to the model are:

- spectrum data: spectrum, spectrum noise and spectrum with subtracted mean
- all auxiliary features

The outputs of the model are:

- Mixture model parameters: mu and sigma of the normal distribution, uniform_a and uniform_b - upper and lower bound of the uniform distribution, alpha - the mixing coefficient between them
- FM parameters predictions
- Quartiles predictions
- Reconstructed spectrum data: spectrum, spectrum noise and spectrum with subtracted mean

5. Did you do any post-processing to the output?

Inverse scaling is applied to predictions as train targets were normalized with min-max scaling.

6. Did you perform any sampling step? If so please describe the sampler.

The predictions for the regular track are sampled from the mixture model with parameters predicted by the neural network. Initially 6000 samples are generated and the ones that are outside of the default prior bounds are rejected. If the remaining samples are above 5000 then 5000 are randomly drawn for submission. In rare cases when the remaining samples are below 5000, 5000 are sampled with replacement so that all planets predictions have equal size and regular track format function can be used.

7. Did you use any external library and/or forward model?

No forward model is used.

`pomegranate` external library is used for fitting mixture models.

`dain` external DAIN_Layer is used for spectrum normalization.

References

1. Schreiber, Jacob, pomegranate: Fast and Flexible Probabilistic Modeling in Python, https://jmlr.org/papers/volume18/17-636/17-636.pdf, https://github.com/jmschrei/pomegranate
2. N. Passalis, A. Tefas, J. Kanniainen, M. Gabbouj and A. Iosifidis, Deep adaptive input normalization for time series forecasting, IEEE Transactions on Neural Networks and Learning Systems, 2019 https://github.com/passalis/dain