Bayesian Inference on Stellar Spectra with Flow-based Model

Bhavya Gupta, Chang Liu, Brigette Vazquez-Segovia, Lingfeng Wei, Jiajian Zhu

June 15, 2024

Github Link: https://github.com/wei-lingfeng/Group7-Project/

1 Introduction

In astrophysics, modeling observational data is very important to comprehend the real physicalities of stars. The usual methods typically use complicated likelihood functions for inferring model parameters. However, with the progress made in generative neural networks lately, a Bayesian method has become possible that gives a more adaptable and effective alternative. Normalizing flow is one that has been proposed for Bayesian inference. It can change a basic distribution, such as a Gaussian, into one more complicated by learning a series of invertible and differentiable transformations. This method helps us figure out the probability distribution for best-fit parameters in stellar spectra — where uncertainties are important astrophysical measurements. Once trained, the model spares the computational resources required by standard sampling and can infer the posteriors rapidly. Also, the model allows generating new samples using the flow transformation on the initial distribution (Papamakarios et al., 2019).

In the project, we intend to model labeled simulated APOGEE stellar spectra utilizing a Residual Neural Network (ResNet), the Gate Recurrent Unit (GRU), followed by a normalizing flow model. A similar work is done in Zhang et al. (2023). Flow-based generative inference models have already been implemented in the literature, for instance, `LAMPE`(Rozet et al., 2021), `sbi`(Tejero-Cantero et al., 2020), and `nbi`(Zhang et al., 2023)

2 Methodology

2.1 Data

Parameter	**Notation**	**Prior**[1]	**Unit**
Effective Temperature	Teff	$U\,(2300, 7000)$	K
Radial Velocity	RV	$U\,(-200, 200)$	km/s
Rotational Velocity	Vsini	$U\,(0, 100)$	km/s
Surface Gravity	logg	$U\,(2.5, 6)$	–
Metallicity	metal	$U\,(-2.12, 0.5)$	–

Table 1: Stellar Parameter Prior Distributions

We initially proposed to use the survey spectral data from The Apache Point Observatory Galactic Evolution Experiment (APOGEE; Majewski et al. 2017) conducted by the 2.5 meter Sloan Digital Sky Survey (SDSS) and model the spectra with the PHOENIX-ACES-AGSS-COND-2011 stellar atmospheric model (Husser et al., 2013) using the Spectral Modeling Analysis and RV Tool (SMART Hsu et al., 2021a,b). Each spectrum is a 1-dimensional array of spectral flux as a function of wavelength from 1.514 to 1.696 microns. An example of the data can be found here. However, we encountered problems with removing the continuum in the observed spectra. Additionally, there are many other instrumental dimensions unrelated to the physics property of the star itself. With limited time and computational resources, we therefore generated simulated spectra with noise using the stellar atmospheric model for this work. We eventually generated $50,000$ simulated spectra varying 5 stellar parameters: effective temperature (teff), radial velocity (rv), rotational velocity (vsini), and metallicity (metal). The ranges of the parameters are listed in Table 1.

[1]The U stands for uniform distribution.

2.2 Model

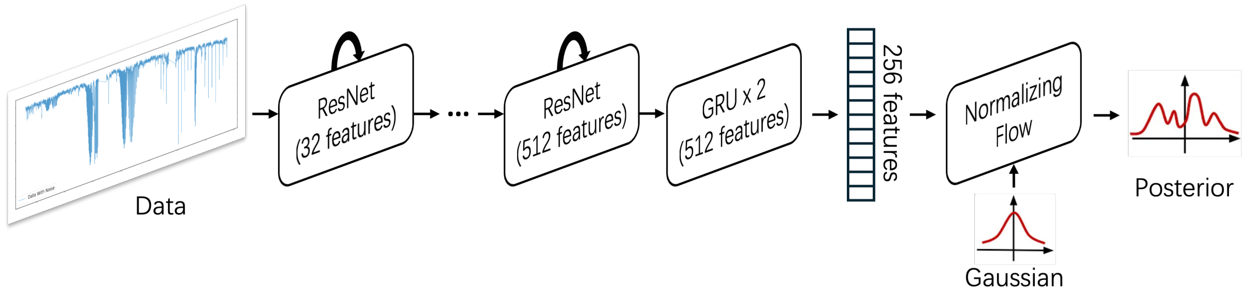

Figure 1: Model architecture. The model consists of a feature extractor (featurizer) with ResNet and GRU, and a normalizing flow that samples the posterior distribution.

The model we used consists of two main parts: a feature extractor, or a featurizer, and a normalizing flow. The featurizer learns to extract the features from the original spectra and feeds them into the normalizing flow. The flow model then performs transformations of variables on base Gaussian distributions to sample the posterior distributions of the stellar parameters according to these features (Papamakarios et al., 2017; Zhang et al., 2023). An illustration of the model is shown in Figure 1.

The featurizer takes in 1-dimensional spectra and uses a combination of the residual network (ResNet) and gated recurrent unit (GRU). Starting at a dimension of 32, each layer has twice as many dimensions as the previous one until it reaches an upper limit followed by 2 layers of GRUs, and finally collapses to a 1D feature array. We implemented 8 layers of ResNet (`depth`), and set the maximum hidden dimension of the convolution neural network in the ResNet to be 512 (`dim_conv_max`), and the output feature dimension to be 256 (`dim_out`). For the normalizing flow model, we set the number of Masked Autoregressive Flow (MAF) to 15 (`num_blocks`), and set the hidden dimension for each MAF block to be 256 (`flow_hidden`). The number of the mixture of Gaussians in base density distribution is set to 4 (`n_mog`). The model is then trained on the data with 100 epochs and a batch size of 256. The learning rate is set to 10^{-4}, and the early stop patience is set to 20 epochs. The model uses the Adam optimizer for the training.

3 Results

This section presents the outcomes of modeling stellar spectra. To understand the flow-based Bayesian inference model, we first started with a PyTorch implementation of GRU + Flow model to infer the simulated sine curve with varying frequency and phase, as in the notebook `GRU_nflows_sine_sample.ipynb`. However, the same model with an increased number of hidden dimensions did not work for the stellar spectra. We think the model is not complicated enough for high-dimensional data such as the stellar spectra. Therefore, we turned to use the `nbi` package.

The notebook `model_spectra_3_params.ipynb` shows the results for modeling the first 3 stellar spectra, and the 5 parameter modeling results are presented in the notebook `model_spectra_5_params.ipynb`.

Figure 2 shows an example of the modeled posterior distributions of the 5 stellar parameters. The posteriors before and after the importance sampling are shown on the left and right respectively. Importance sampling is the process of assigning a weight to each of the samples in the parameter space. The weight is proportional to the inverse of χ^2, the square of the difference between the data and the model spectra computed at the given sample. Therefore, this step requires a model function. As can be seen, the importance sampling shrinks down the contours close to the truth values, and results in sharper posterior distributions, making the uncertainties even smaller. The uncertainty of around 70 K in effective temperature is decent and not unreasonably small. The uncertainty of about 0.1 km/s in radial velocity is ideally expected from the high-resolution nature of the spectra. Figure 3 shows the reconstructed spectra with the best-fit stellar parameters. It can clearly be seen that the model captures the features such as the absorption lines in the spectra. From all the above perspectives, the model is very successful in modeling the simulated spectra.

The training and validation loss curves are shown in Figure 4, both of which show a clear and steady decreasing trend that confirms the strength of our model. The training and validation loss decreases hand in hand and show no sign of overfitting.

Note that we did not have time to build the model completely by ourselves, but we looked into the code and made

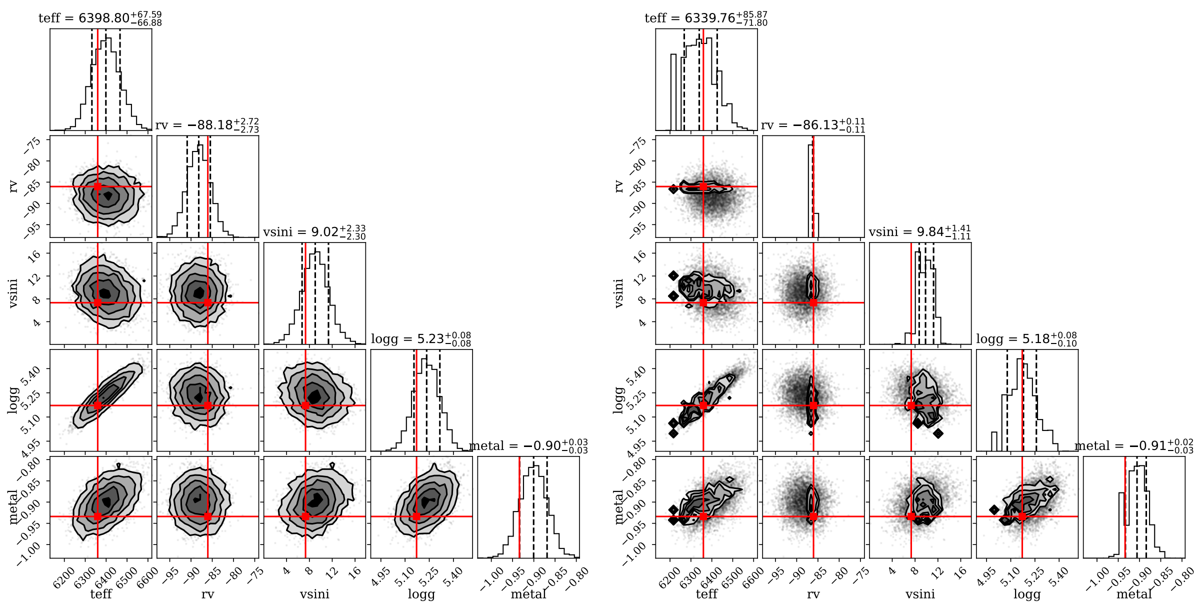

Figure 2: Modeled posterior distributions of stellar parameters before (*left*) and after (*right*) importance sampling. Importance sampling assigns a weight to each of the samples proportional to the inverse of χ^2, the square of the difference between the data and the model at the given sample.

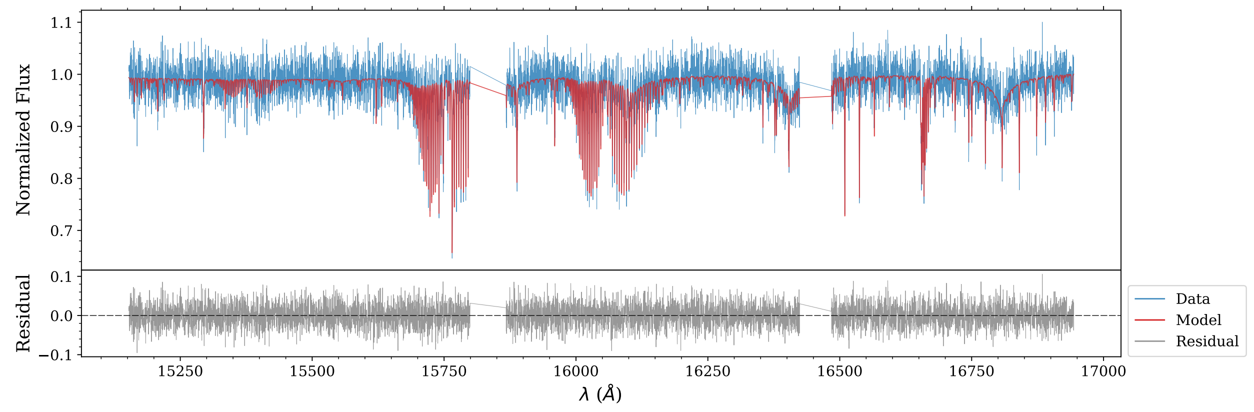

Figure 3: Reconstructed spectra with the modeled stellar parameters shown in red and the original data shown in blue. The bottom pannel shows the residual.

sure that we understand the structure of the model which is illustrated in Section 2.2. An attempt was made to replace the featurizer with a subclass of PyTorch's `nn.Modules` but ended with failure. Another attempt to model our downloaded real data also resulted in a failure as the flow model tries to sample outside the prior ranges for unknown reasons (See the output in `model_real_spectra.ipynb`)

4 Comparison with MCMC

As a comparison with the standard Monte Carlo Markov Chain (MCMC) methods, we performed an MCMC Bayesian inference on the same simulated data. Figure 5 shows the MCMC result of the posterior distributions. The first feature that caught one's eye is that the MCMC and flow-based model both capture the correlation between effective temperature and rotational velocity in the parameter space. The results are also similar. This supports the claim in Zhang et al. (2023) that the normalizing flow would converge to the same results as the MCMC. However, there are times when we observe degeneracies in the radial velocity which are unique in the MCMC results. This could potentially be attributed to the high level of noise we added, as in real data the scale of noise is much smaller, and such degeneracies in RV are rarely seen in MCMC results on real data.

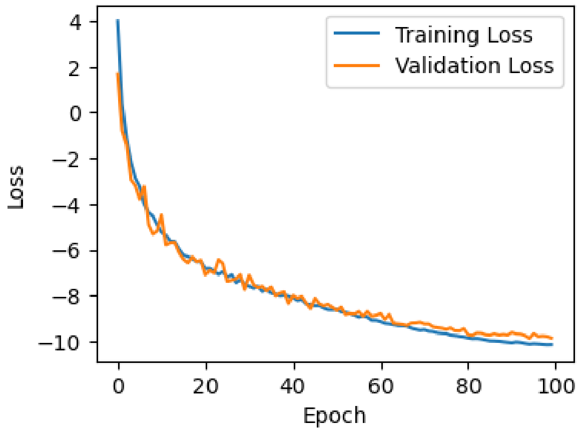

Figure 4: Training and Validation Loss Curves

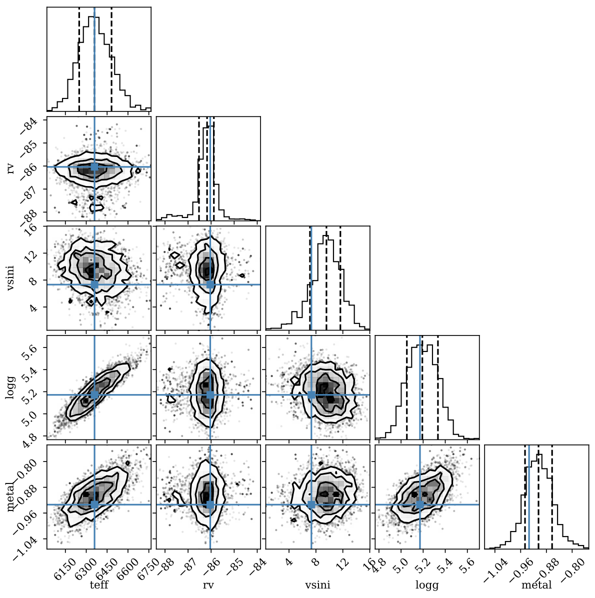

Figure 5: MCMC results of stellar parameter posteriors

5 Conclusion

We showed how Bayesian inference with the flow-based model can be used to model simulated stellar spectra using ResNet, GRU, and normalizing flow. By applying these machine learning methods, we could model the posterior distributions of stellar parameters such as effective temperature, radial velocity, rotational velocity, surface gravity, and metallicity with great precision and reliability. The training and validation loss curves showed a consistent decrease, which confirmed that the model was learning well without overfitting. Additionally, the reconstructed spectrum matches the observed data. We also sharpened the posterior distribution by performing importance sampling. Finally, the comparison between the traditional MCMC methods and the machine learning methods shows consistency and both capture the correlation in the parameter space. In summary, the flow-based model shows a strong capability in Bayesian inference. It may greatly expedite the modeling of astronomical data and save enormous computational resources that are required by traditional MCMC methods without sacrificing the accuracy of the inference.

6 Group Contribution

Bhavya Gupta: Setup `nbi` and constructed the GRU + Flow model using PyTorch modeling sine curve; Attempted transformer as featurizer; *Model* section in presentation; *Methodology* section in the report.

Chang Liu: Trained the sine wave sample and got good results; Modeled 3 stellar parameters; Helped the presentation slides. Improved Introduction section, data section, wrote results section, and conclusion section for the final report.

Brigette Vazquez-Segovia: Get the sine curve code running on DataHub and worked on changing parameters to decrease training loss; *Motivation* section in presentation; *Introduction* and *Methodology* in final report.

Lingfeng Wei: Started and maintained Github repo; Downloaded and generated simulated spectra; Modeled 3 and 5 stellar parameters; Compared with MCMC results; *Results* section in presentation; *Results* and *Comparison with MCMC* sections in the report;

Jiajian Zhu: Trained the sine curve model; Started overleaf and presentation documents; *Introduction* section in presentation; Helped improving the final report.

References

Hsu, C.-C., Theissen, Christopher A.and Burgasser, A. J., & Birky, J. L. 2021a, SMART: The Spectral Modeling Analysis and RV Tool, v1.0.0, Zenodo, doi: 10.5281/zenodo.4765258

Hsu, C.-C., Burgasser, A. J., Theissen, C. A., et al. 2021b, ApJS, 257, 45, doi: 10.3847/1538-4365/ac1c7d

Husser, T. O., Wende-von Berg, S., Dreizler, S., et al. 2013, A&A, 553, A6, doi: 10.1051/0004-6361/201219058

Majewski, S. R., Schiavon, R. P., Frinchaboy, P. M., et al. 2017, AJ, 154, 94, doi: 10.3847/1538-3881/aa784d

Papamakarios, G., Nalisnick, E., Jimenez Rezende, D., Mohamed, S., & Lakshminarayanan, B. 2019, arXiv e-prints, arXiv:1912.02762, doi: 10.48550/arXiv.1912.02762

Papamakarios, G., Pavlakou, T., & Murray, I. 2017, arXiv e-prints, arXiv:1705.07057, doi: 10.48550/arXiv.1705.07057

Rozet, F., Delaunoy, A., Miller, B., et al. 2021, LAMPE: Likelihood-free Amortized Posterior Estimation, doi: 10.5281/zenodo.8405782

Tejero-Cantero, A., Boelts, J., Deistler, M., et al. 2020, The Journal of Open Source Software, 5, 2505, doi: 10.21105/joss.02505

Zhang, K., Bloom, J. S., van der Walt, S., & Hernitschek, N. 2023, nbi: the Astronomer's Package for Neural Posterior Estimation. https://arxiv.org/abs/2312.03824

Zhang, K., Jayasinghe, T., & Bloom, J. 2023, in Machine Learning for Astrophysics, 39, doi: 10.48550/arXiv.2312.05687