Neural Networks Applied to Spectral Modeling of AGN Jet Emission

A. Tzavellas, NKUA

Collaborators: G. Vasilopoulos, M. Petropoulou, A. Mastichiadis, S.I. Stathopoulos

Workshop on Numerical Multi-messenger Modeling
APC Laboratory, Paris

Introduction

What is a Blazar?

A supermassive black hole at the center of a galaxy, with powerful jets of material and one of those jets happens to be pointed right at us :o

Why is Blazar study important?

- Study accretion onto supermassive black holes across redshifts
- Offers insight to physics in extreme conditions

How are Blazars studied?

- Combine spectra from different instruments and use Spectral Energy Distributions
- SEDs reveal radiative processes involved
- Requirement for fast numerical models

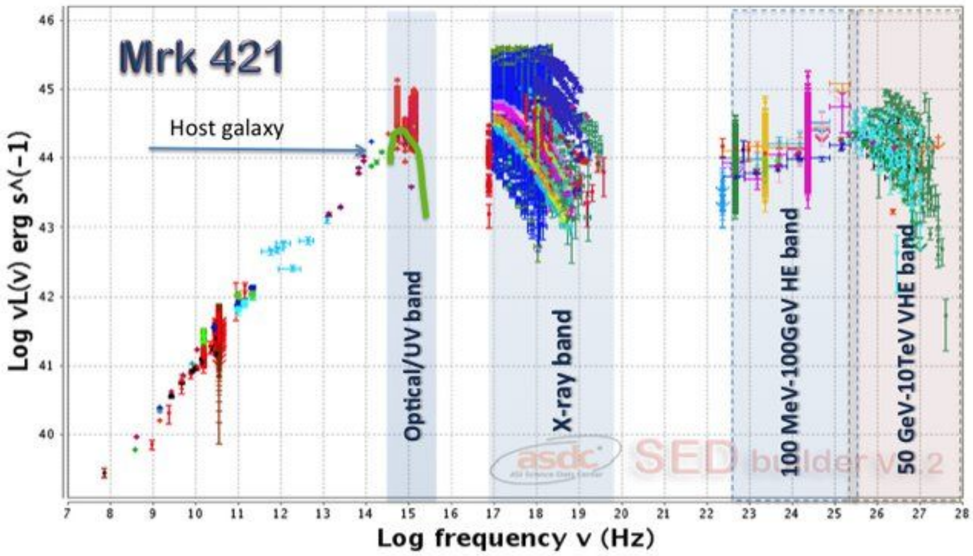


Image Credit: Padovani et al. (2017)

Existing Numerical Models

- What are they? e.g. solvers of stiff coupled PDEs
- Computational complexity increases as more radiative processes are introduced
- Leptonic model computation time: a few seconds to a couple of minutes
- Leptohadronic model computation time: ATHEνA[1](~30min) LeHaMoC[2](~10min)

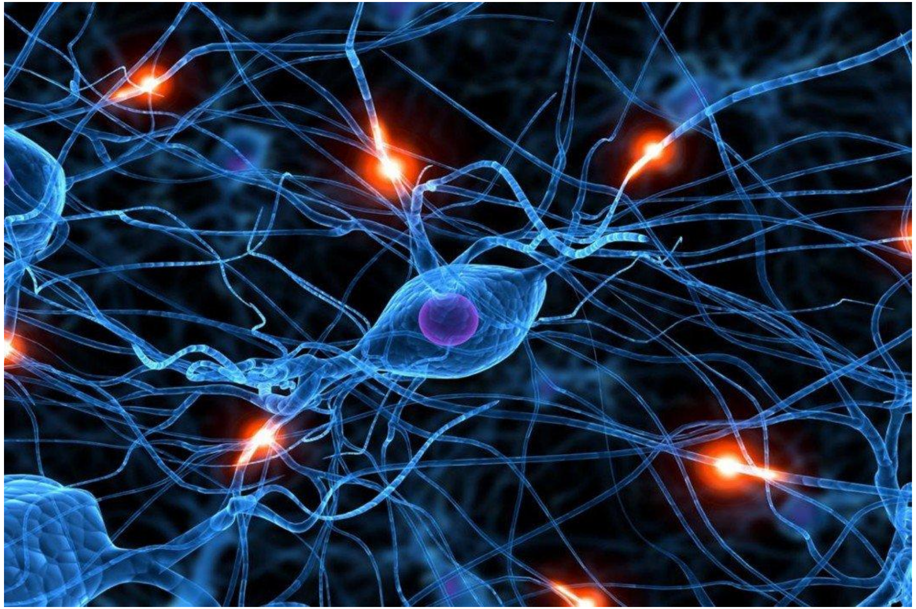
Shortcoming: Cannot use with Markov Chain Monte Carlo

1 ATHEνA

2 LeHaMoC

Neural Networks and Applications

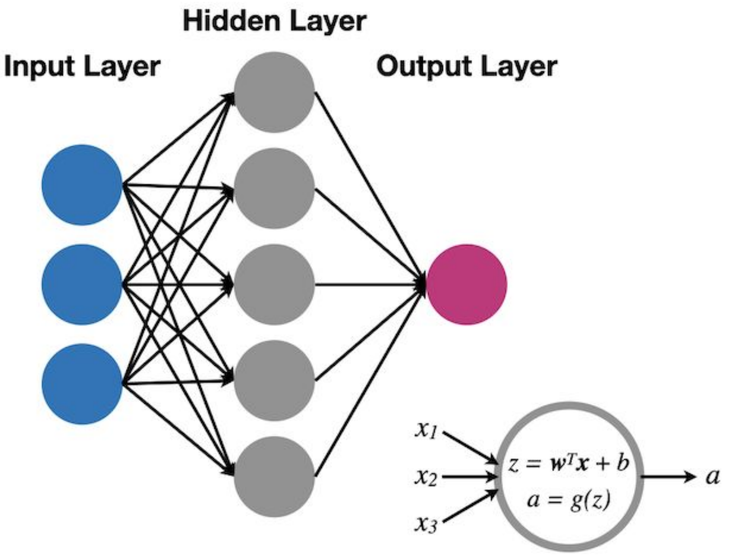
1. Facial recognition
2. Speech recognition
3. Recommendation engines (Netflix)
4. Healthcare - Medical image analysis
5. Biology - Emulate mechanism-based biological models
6. Finance - Market prediction
7. Self-driving cars
8. Image and music generation
9. Large Language Models (Chat GPT)



Research Aim

Replace a numerical model with a Neural Network

1. Proof-of-concept: replace an average complexity model with a trained neural network
2. Evaluate trained model



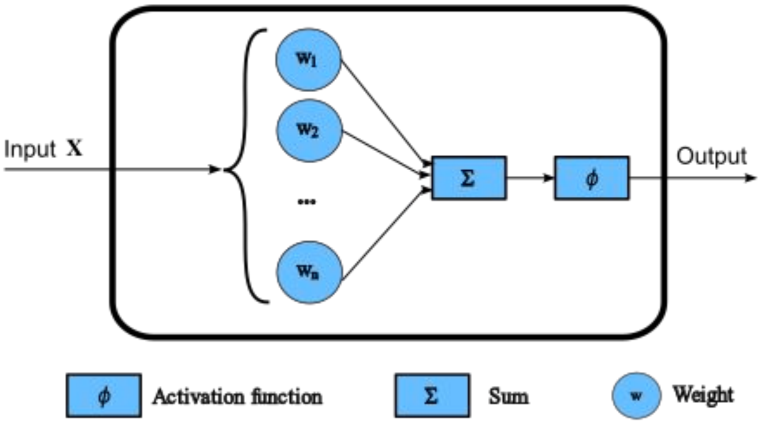
What is a Neural Network?

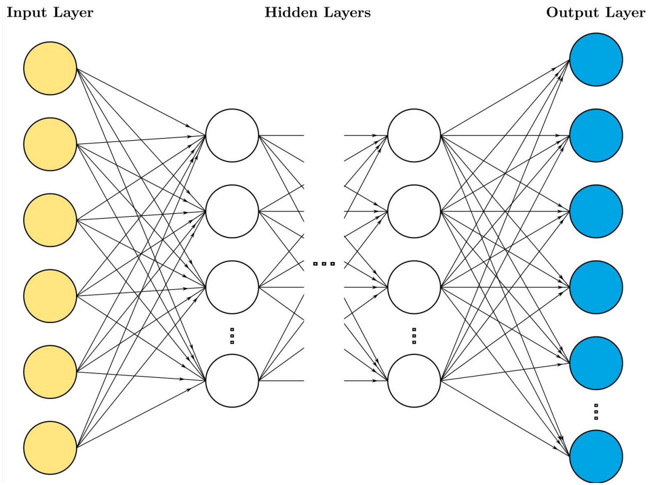
Numerical constructs whose "name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another"[1]

Stack of Neurons: Hidden Layer

Stack of Hidden Layers: Neural Network

Artificial Neuron (ANN cell): For input $X = (x_1, \ldots, x_n)$, apply dot product with $W = (w_1, \ldots, w_n)$ and pass result to activation function ϕ. The output is fed to the next layer.



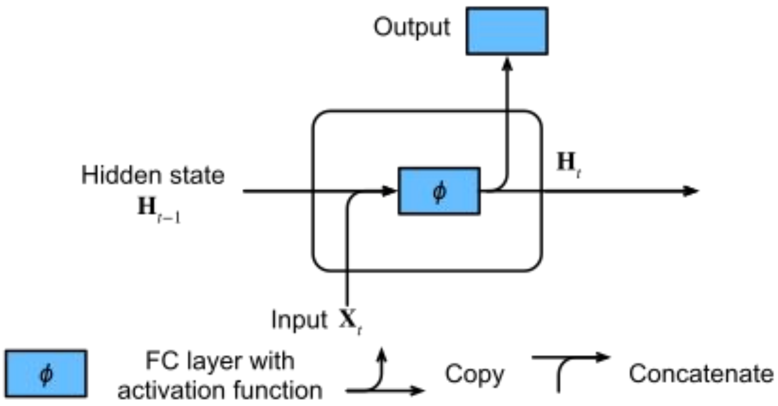


1 IBM

2 Image Credit Top picture: Dive into Deep

Recurrent Neural Networks

- Possess loops that allow information persistence (“memory” of past inputs)
- Suited for tasks like time-series prediction
- 3 types of RNN cells: simple Recurrent Neuron **(Vanilla)**, Gated Recurrent Unit **(GRU)**, Long-Short Term Memory **(LSTM)**
- Vanilla: Useful for understanding the concept, impractical due to unstable gradient during training

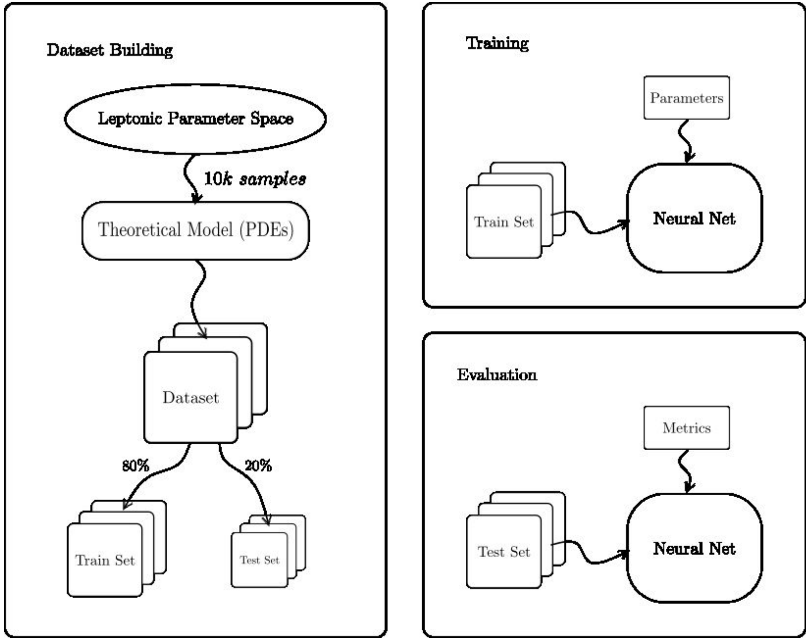


Neural Network Concepts

- **Activation Function ϕ:** A mathematical function applied to the output of a neuron in a neural network. e.g. ReLU(x) = max(0, x).
- **Loss Function**: Measures the difference between the predicted output of a neural network and the actual desired output. e.g. MSE, MAE, RMSLE
- **Training:** Forward-pass, Loss Calculation, Back-propagation, Rinse and Repeat!
- **Optimization Algorithm (GD)**: calculates gradients numerically, possible numerical instabilities (vanishing/exploding gradients)

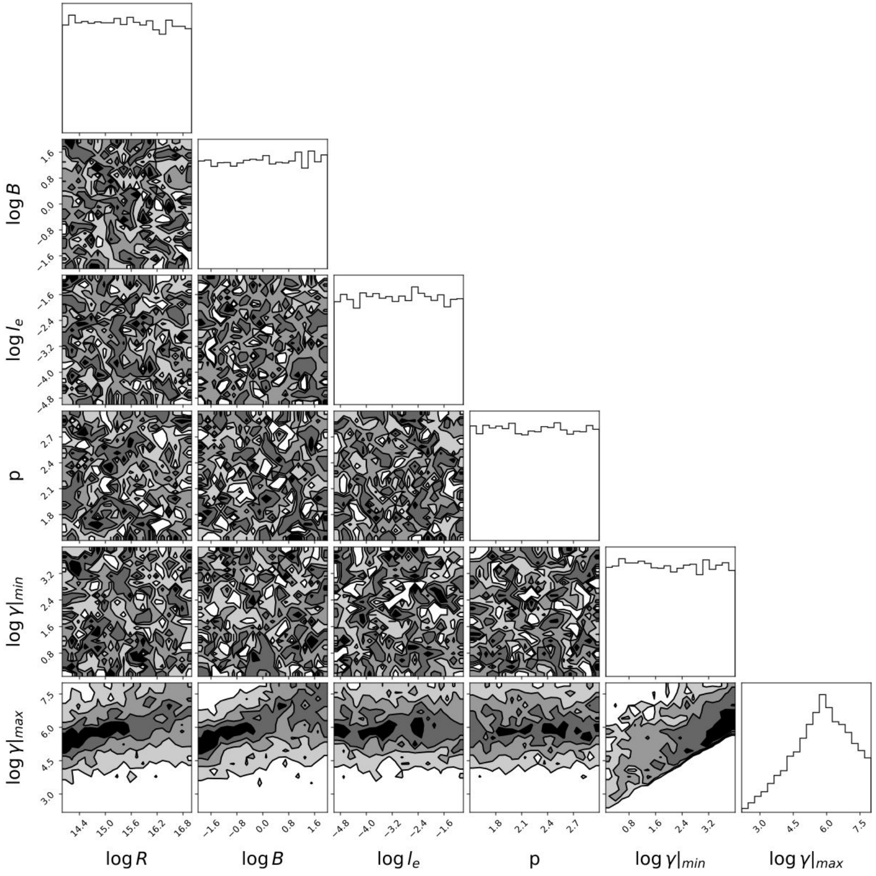
What did we do?

1. Choose Machine-Learning Framework: Tensorflow
2. Create a sufficiently large dataset
3. Train several NNs of different architectures (number of hidden layers, number of neurons per layer, neuron types, etc.)
4. Compare the best candidates and reveal the optimal
5. Evaluate the optimal model



Dataset Creation

- Use a theoretical model to generate AGN spectra: ATHEvA
- For a leptonic model, six input parameters are required
- Sample the six input parameter space
- Decide on the number of samples (~10k)
- For each sample, use theoretical model to generate its spectrum
- Interpolate the spectrum: 500 points
- Store sample & spectrum as a dataset entry



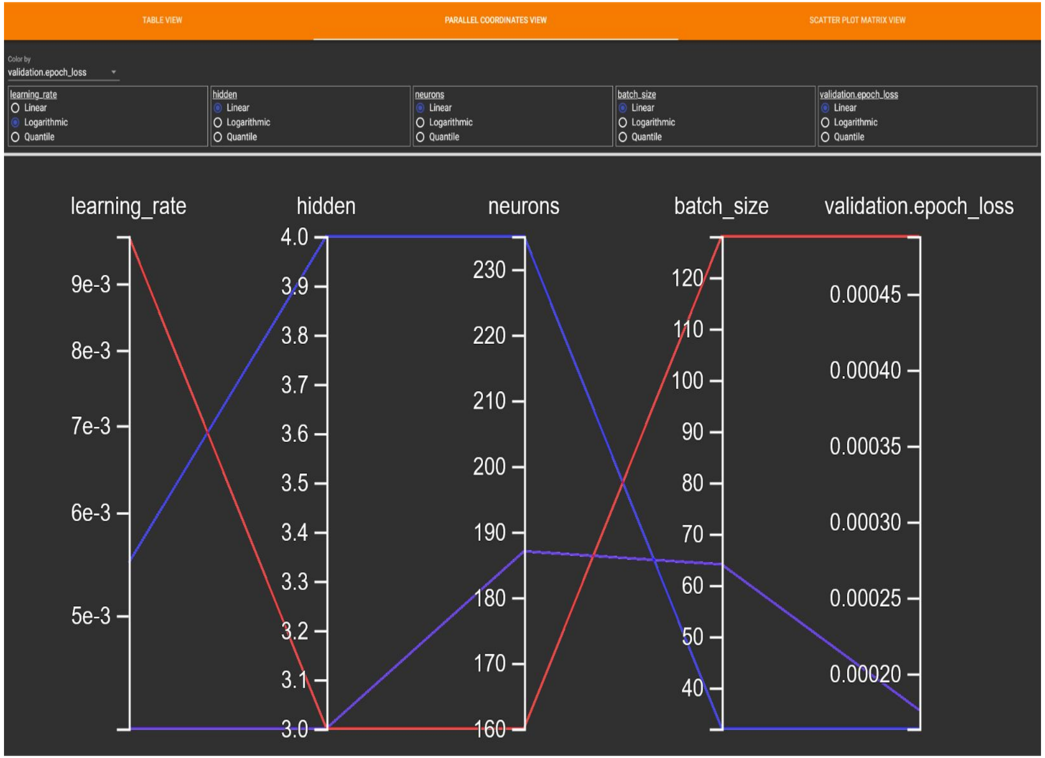
Neural Network Architecture

- Explore 3 types of neurons: ANN, GRU, LSTM
- Define Hyper-parameters: neurons per layer, number of layers, learning rate, batch size
- For each neuron type, tune Hyper-parameters to find a sub-optimal network of that type.

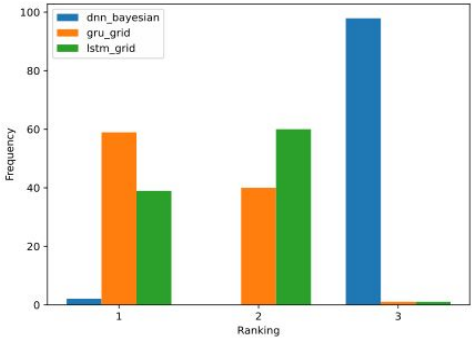
Parameter	**Range**
Hidden Layers	$[4, 7]$
Neurons per Layer	$[64, 256]$
Learning Rate	$(10^{-5}, 10^{-2})$
Batch Size	$\{32, 64, 128, 256\}$

Tuning of Hyper-Parameters

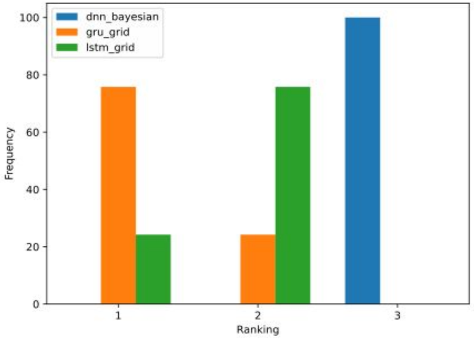
- Tuning involves running a search algorithm.
- Possible algorithms: Grid Search, Random Search, Bayesian Optimization
- All exist in Tensorflow



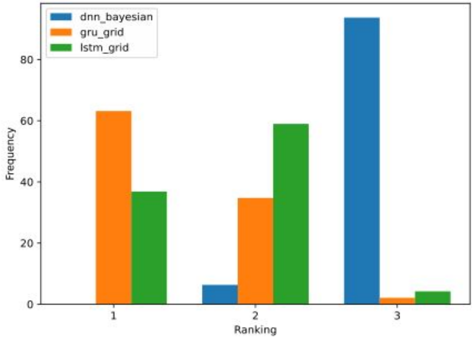
Comparison of the Best Candidates



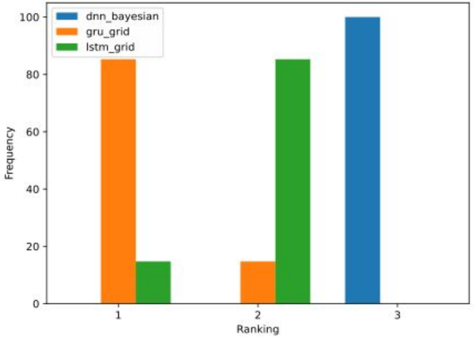
(a) DTW metric



(b) MSE metric



(c) KST metric

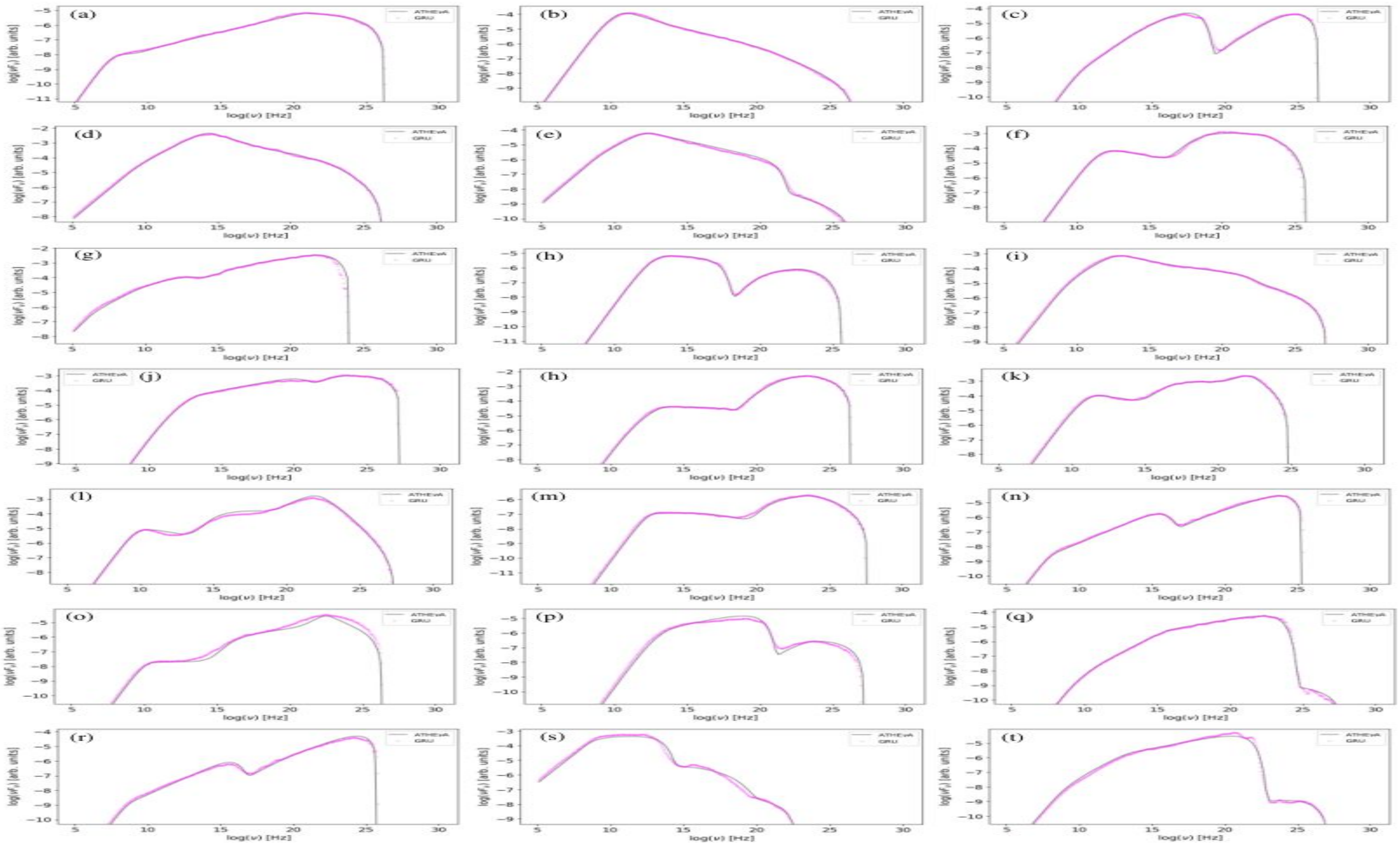


(d) EMD metric

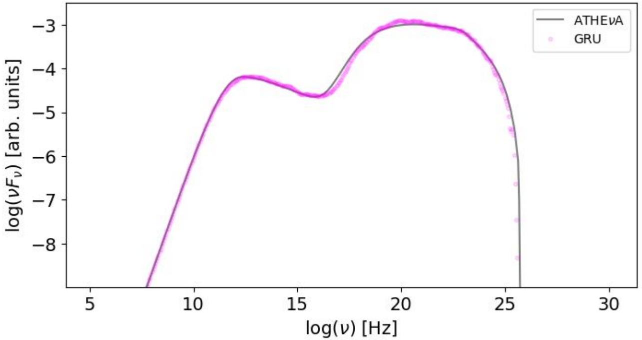
For a test sample of 100 cases

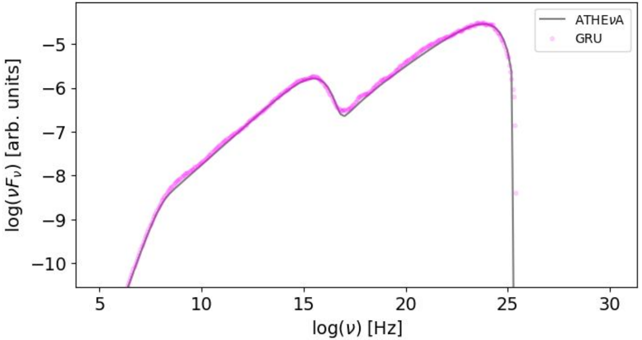
- Apply a measure of dissimilarity between the predicted SED and the theoretical. The smaller the metric the better the match.
- Rank the candidates based on the metric.
- Metrics used: DTW, MSE, KST, EMD

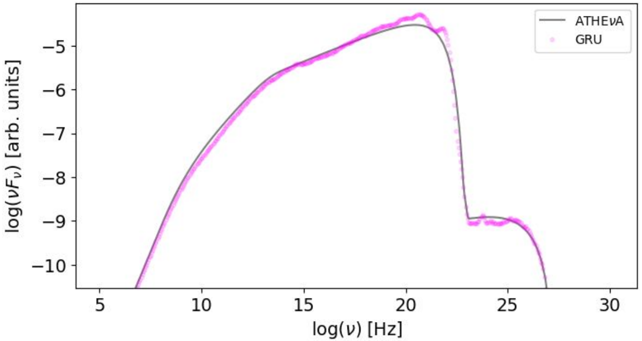
ATHEvA model vs NN - Visual Comparison



ATHEvA model vs NN - Visual Comparison 2



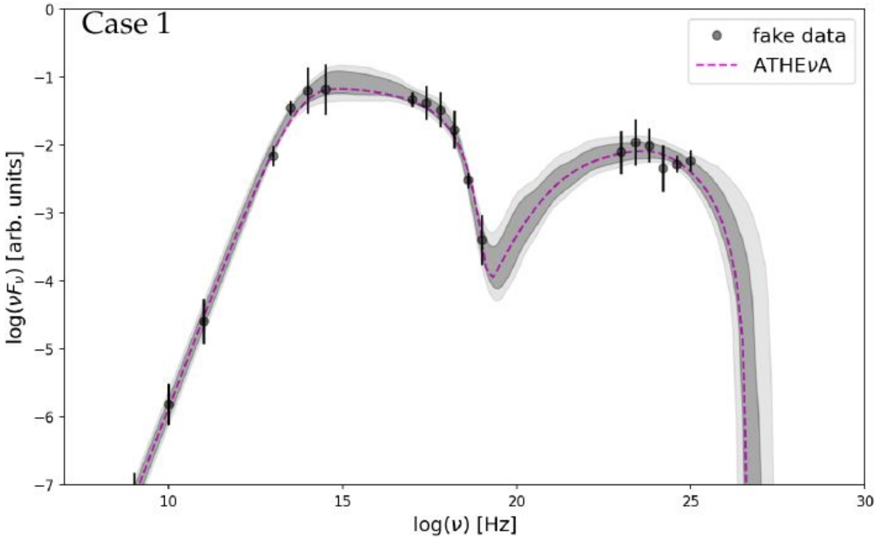


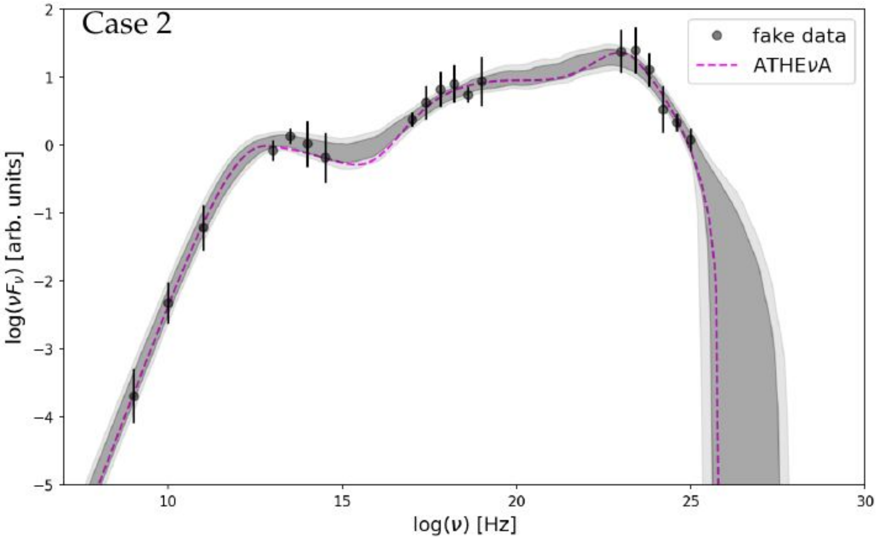


Test: Recovering parameters from simulated spectra

- Generate a simulated SED (25 points) using ATHEvA.
- Add Gaussian noise of std 0.1 to the logarithmic flux values (variability)
- Choose points to represent Radio, UV, X-ray and γ-ray bands.
- Use the NN model to recover the parameters of the simulated SED

Recovered spectra - Plots





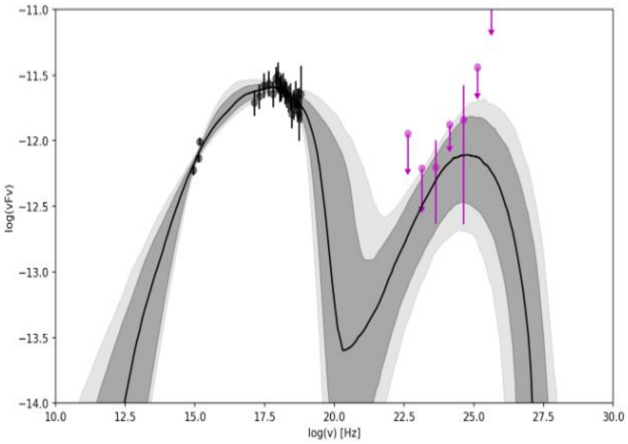
Recovered spectra - Values

	Case 1		Case 2	
Parameter [unit]	True	Fit	True	Fit
$\log R$ [cm]	16.60	$16.04^{+0.61}_{-0.97}$	15.42	$14.78^{+0.66}_{-0.53}$
$\log B$ [G]	-0.34	$-0.49^{+0.36}_{-0.37}$	-0.12	$0.24^{+0.38}_{-0.38}$
$\log \gamma_{min}$	3.37	$3.13^{+0.35}_{-0.24}$	2.31	$2.20^{+0.31}_{-0.26}$
$\log \gamma_{max}$	5.55	$5.51^{+0.25}_{-0.19}$	4.76	$5.57^{+1.07}_{-0.98}$
$\log \ell_e$	-4.57	$-5.08^{+0.40}_{-0.49}$	-1.82	$-1.83^{+0.46}_{-0.29}$
p	2.20	$2.76^{+0.17}_{-0.34}$	2.62	$2.64^{+0.20}_{-0.22}$
δ	1	$1.3^{+0.2}_{-0.2}$	1	$1.00^{+0.16}_{-0.14}$

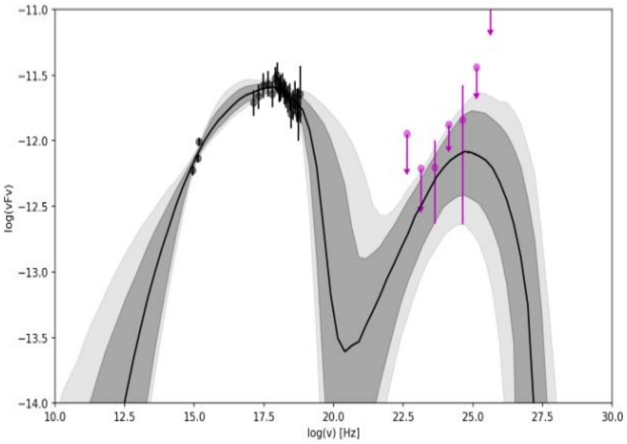
Test: Recovering parameters from observed spectra

Use blazar HSP J095507.9+355101 as test case. Why?

- Combines measurements with small errors at lower energies with several upper limits in the γ-rays
- Same source modeled with LeHaMoC



emcee

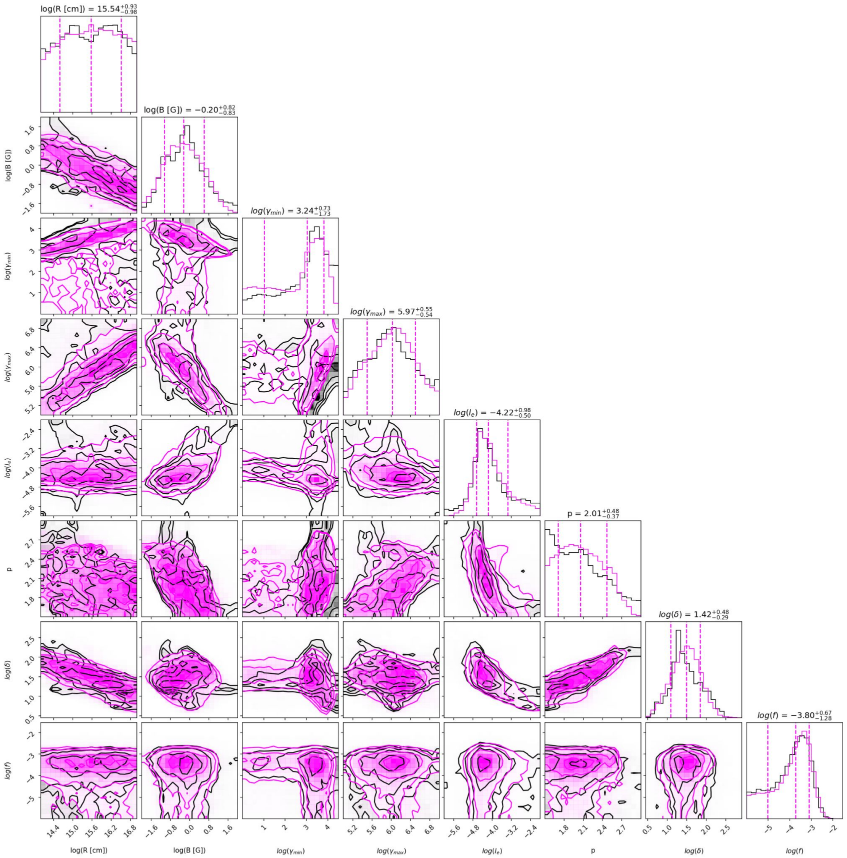


Ultranest

	ATHEνA	LeHaMoC	GRU NN
Single SSC model	1-4 min	2-3 s	3 ms
emcee*	–	20 d	20 min
UltraNest	–	–	21 hr

* For 48 walkers and 10,000 steps for each chain.

CompariSon with LehaMoC+emcee



Conclusions - Future Work

- GRU NN performed better than ANN and LSTM.
- Leptonic (SSC) NN achieves ~ms computational times!
- Train a leptonic NN including more physical processes:
  - external Compton scattering,
  - SSA
  - $\gamma\gamma$ pair production
- Train a lepto-hadronic NN with LeHaMoC or ATHEvA

Links:Github, arXiv:2311.06181, accepted in A&A

Questions?


NOT SURE IF THEY'RE CLAPPING FOR MY PRESENTATION
OR BECAUSE ITS FINISHED

Differences with Bégué et all (2023)

- Use Convolutional Network neurons
- Different mix in the output (not just SED values, but also finite differences)

DTW