INTERPRETABLE AI IN PHYSICS

Savannah Thais

BIRS Interpretability in AI Workshop

05/05/2022

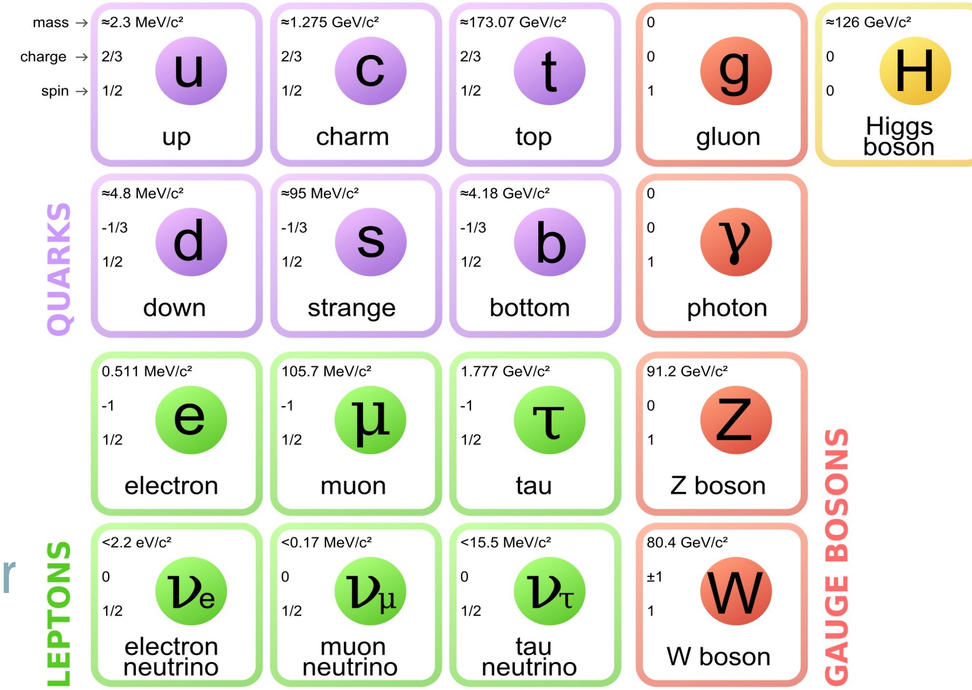
Physics and ML are concerned with characterizing the true probability distributions of nature, how do we understand which model is most accurate and predictive?

Particle Physics Data

Particle Physics Data

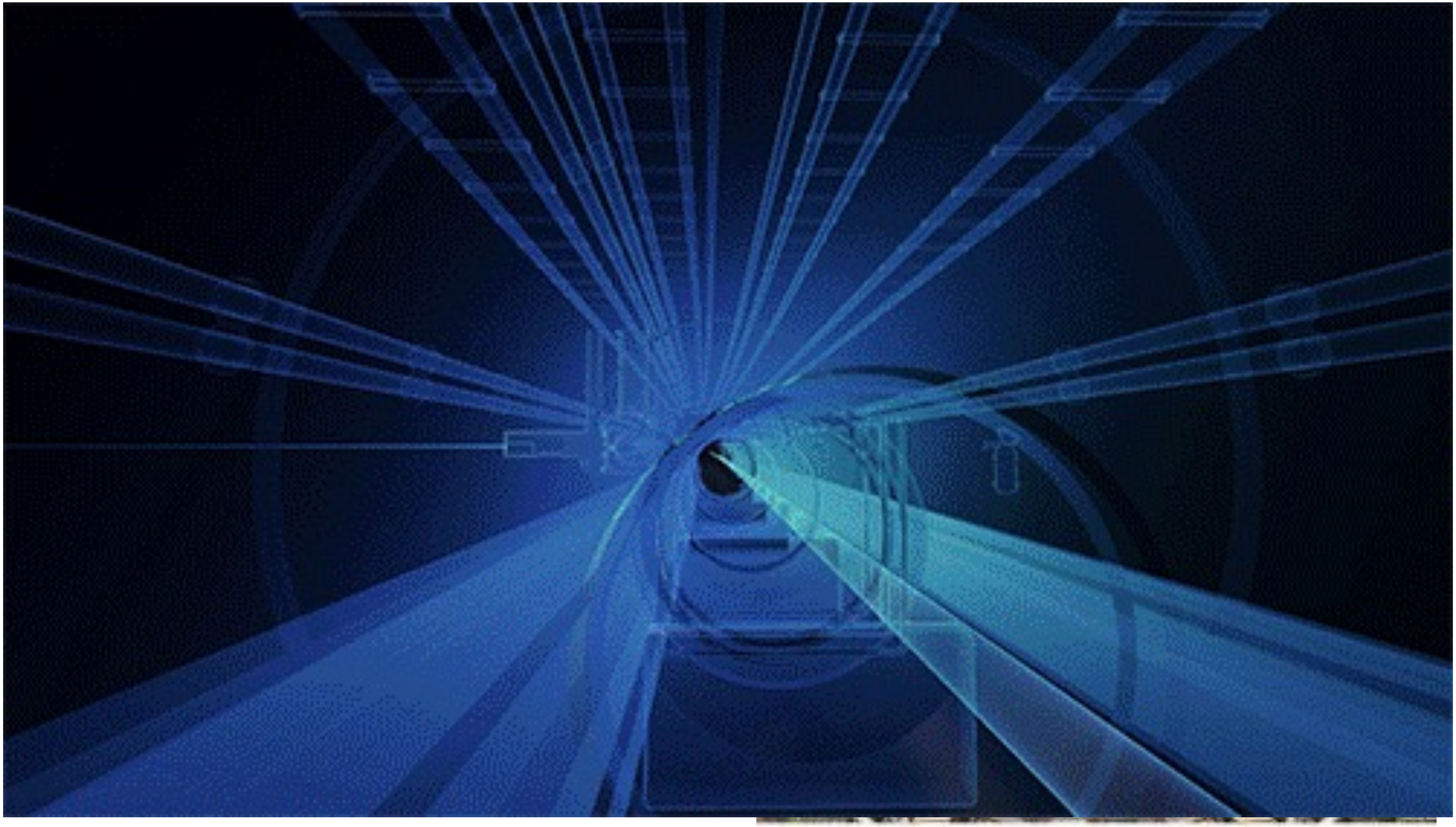
- The Standard Model mathematically (probabilistically) describes the fundamental constituents of all visible matter and most forces in the universe
  - But there are still many open questions and tensions between predictions and nature
- We use the Large Hadron Collider to collect massive datasets to study these tensions

1. Accelerate protons to .99x the speed of light
2. Collide the accelerated particles
3. E=mc^2, so the high energy collisions create rare, exciting particles
4. Measure the decay products with specialized detectors.



$$
\begin{aligned}
L = & -\tfrac{1}{4} W_{\mu\nu} W^{\mu\nu} - \tfrac{1}{4} B_{\mu\nu} B^{\mu\nu} - \tfrac{1}{4} G_{\mu\nu} G^{\mu\nu} \\
& + \psi_j \gamma^\mu \left(i\, \delta_\mu - g^\tau W_\mu - g^i\, Y_i\, B_\mu - g_j\, T_j\, G_\mu \right) \psi_j \\
& + |D_\mu\, \phi\, |^2 + \mu^2\, |\, \phi\, |^2 - \lambda |\, \phi\, |^4 \\
& - \left(y_i\, \psi_{iL}\, \phi\, \psi_{jR} + y_i\, \psi_{jL}\, \phi_c\, \phi_{jR} + \text{conjugate} \right)
\end{aligned}
$$

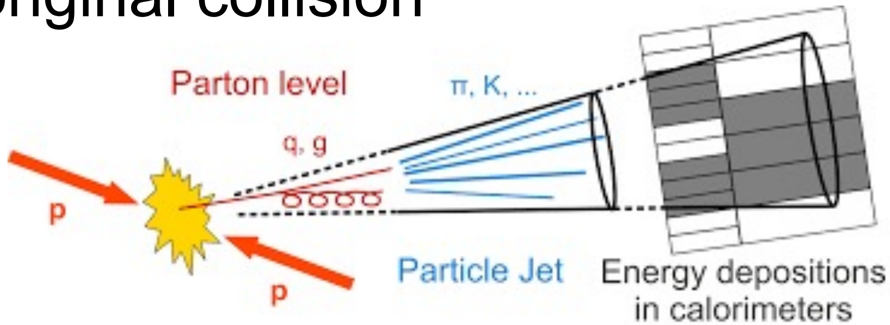
Particle Collisions at the LHC

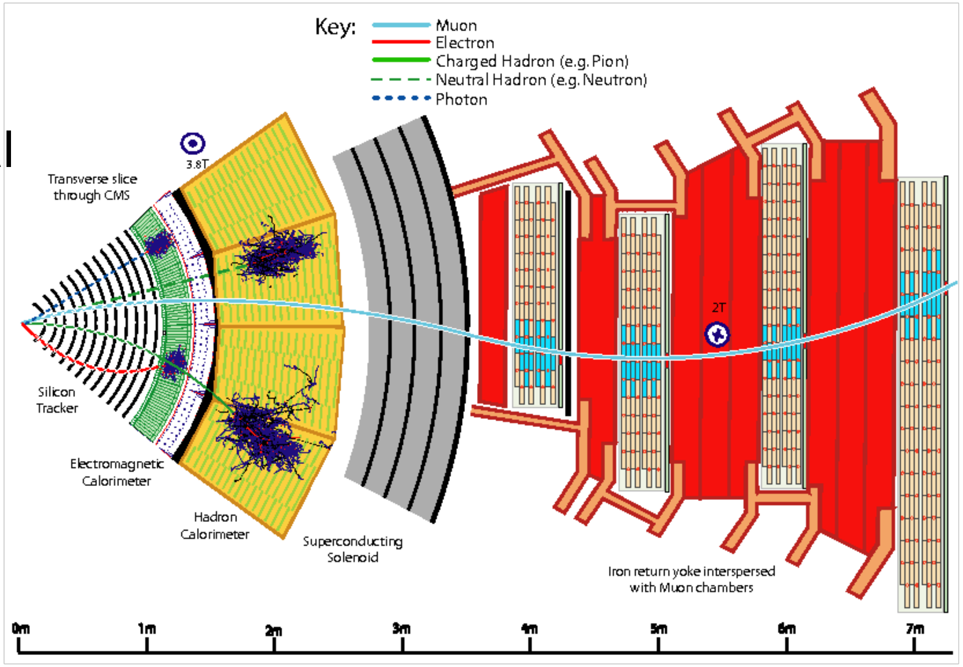


Particle Physics and ML

Raw data consists of energy deposits in different types of detectors, specialized software must then reconstruct what happened in the original collision

1. Object construction: identify detector data belonging to individual particle and its decays
2. Object identification/tagging: identify what type of particle created the reconstructed data
3. Event reconstruction: using physics knowledge, extrapolate to what likely happened at the original collision



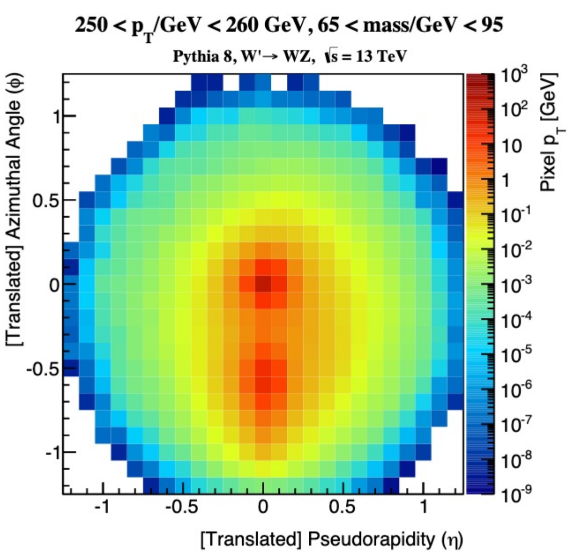


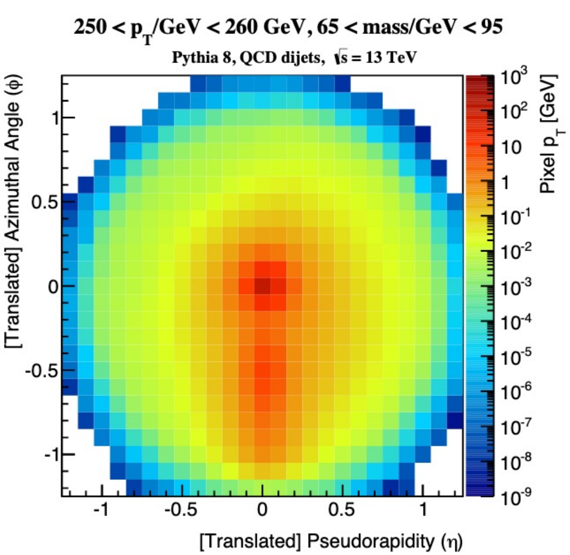
- All of these steps are inherently probabilistic
- We ultimately want to know if the model is discovering something new about the universe
  - And if it's respecting what we already know

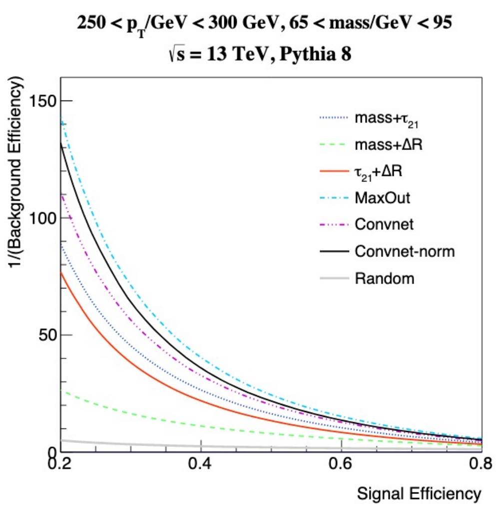
Feature Importance and Relevance Propagation

Particles as Images

- Heavy particles hadronize into collimated sprays (jets) and are absorbed in the granular calorimeter
  - Want to distinguish different types of jets based on their energy patterns
- Can achieve higher classification accuracy using CV
  - Standard approach uses cuts on physics-inspired features
- 'Unroll' the detector and map each cell to an image pixel
  - Apply preprocessing (normalization, rotation, translation) to standardize
  - Train CNN to classify jets (simplify to binary classification)



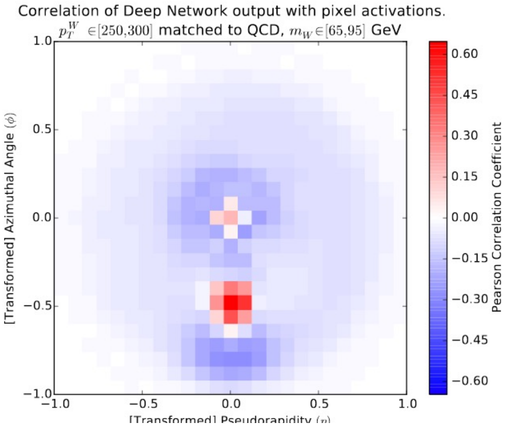


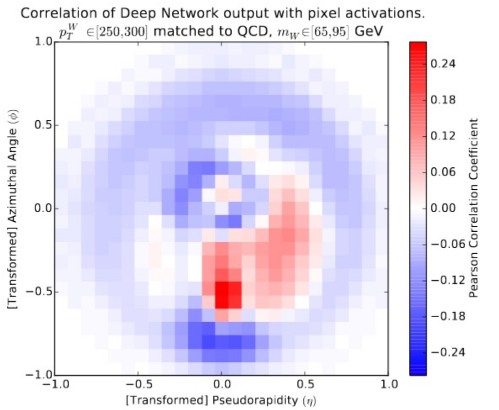


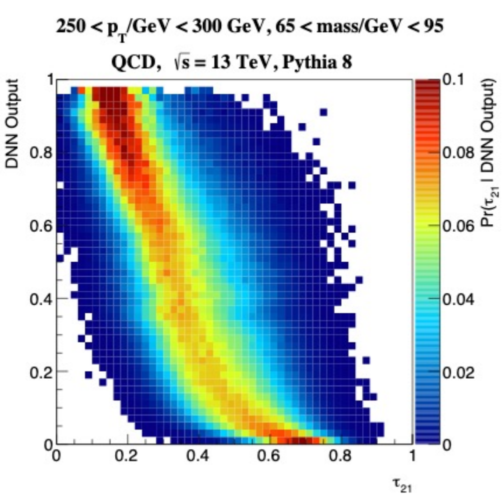
paper

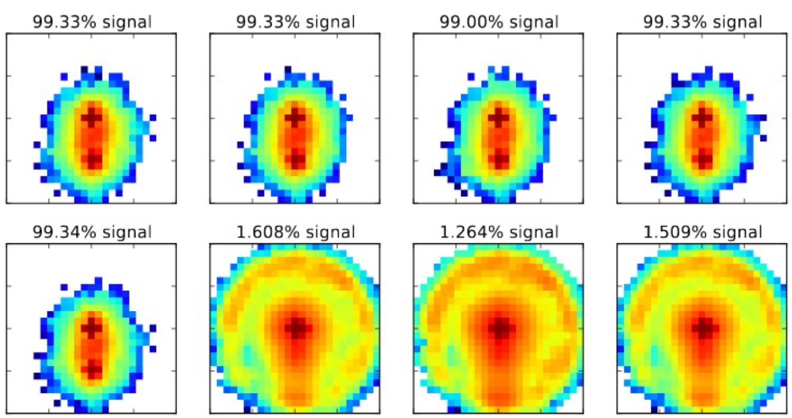
CNN Interpretation

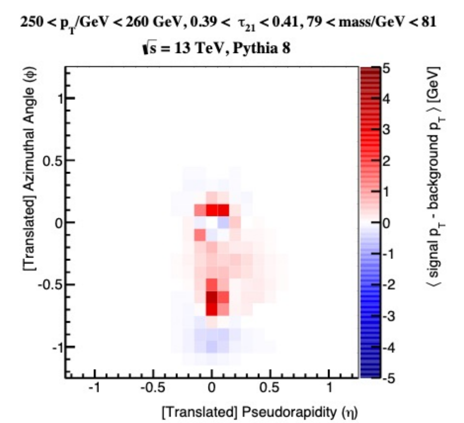
- Look at correlation of CNN output with standard physics features → it's learning thing we expect to be important
- Look at average of images with highest activations for last hidden layer → presence of secondary core is informative
- Look at per pixel correlation with CNN output (doesn't map to a known physical function)
    - Reweight samples to remove known physical variables → the radiation around the second core seems to matter
    - Look at only jets with W-like mass → radiation between cores seems to matter → learning about color flow?





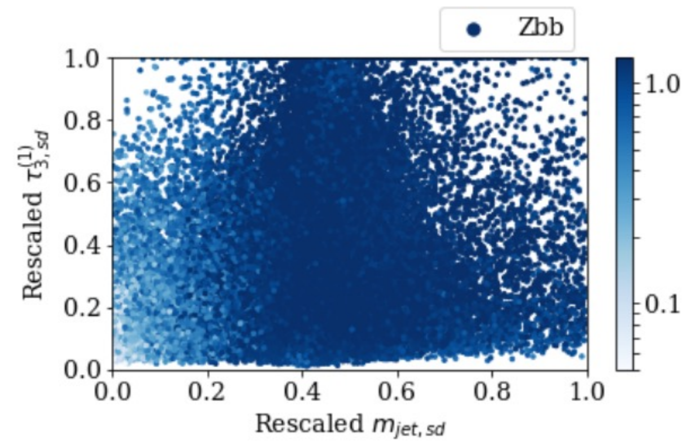


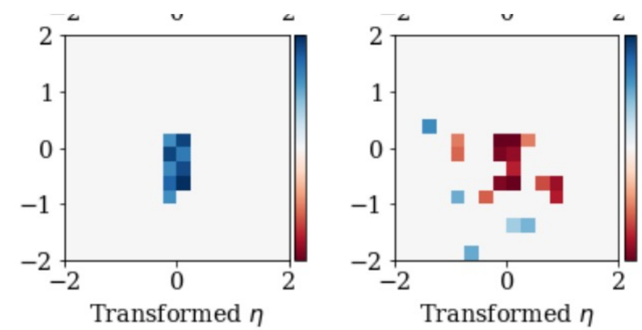


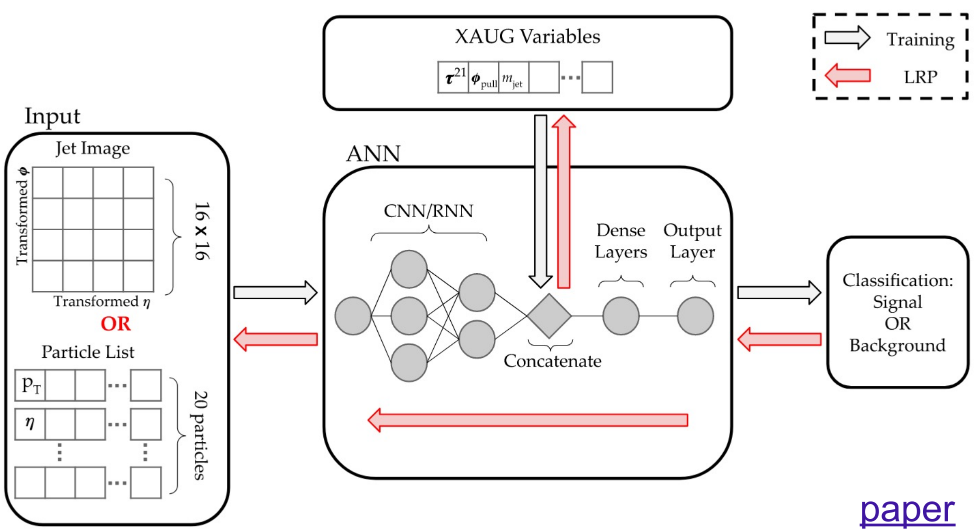


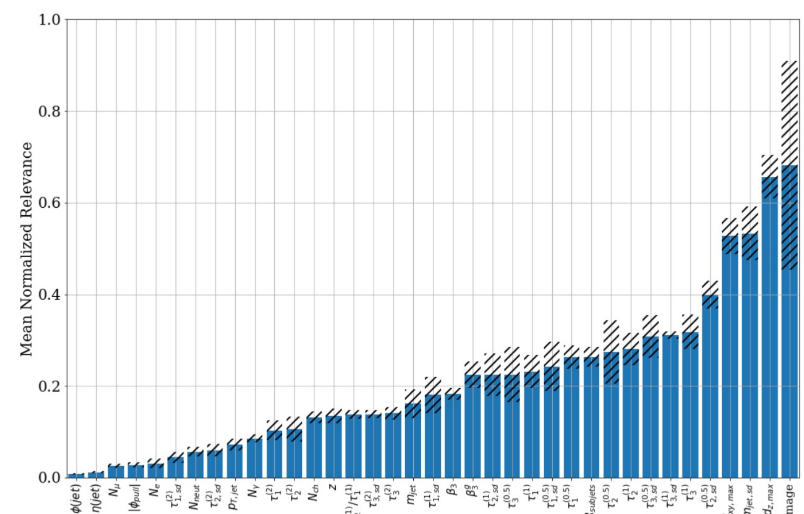
Adding In Expertise

- Augment the CNN with physics-motivated features after initial prediction
- Use LRP to understand what information the network is using
  - Can you replace the learned representation with engineered features
- Demonstrates the network learns expected physical relationships
  - But image representation is most important feature → some new information









[paper]

Implications and Limitations

- We can (sort of) check if a model is learning about physics features we know
- But how do we interpret what else it is learning
  - No clear way to map image relevances to mathematical information
- No way to identify if relevances are due to true generalizable physics or statistical artifacts
- These methods don’t characterize model performance on edge cases or difficult samples

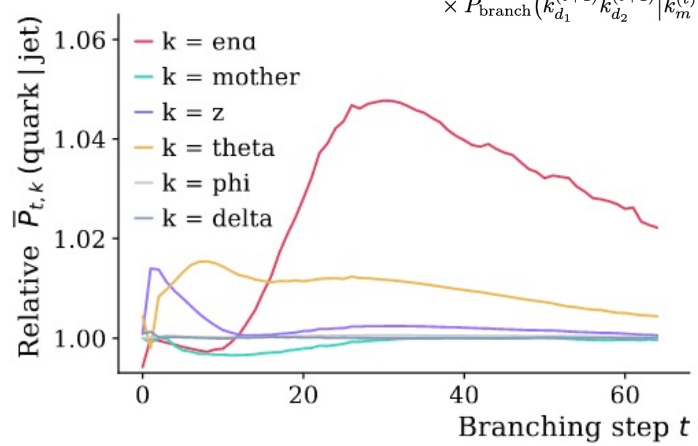
Using Physics Knowledge as a Basis

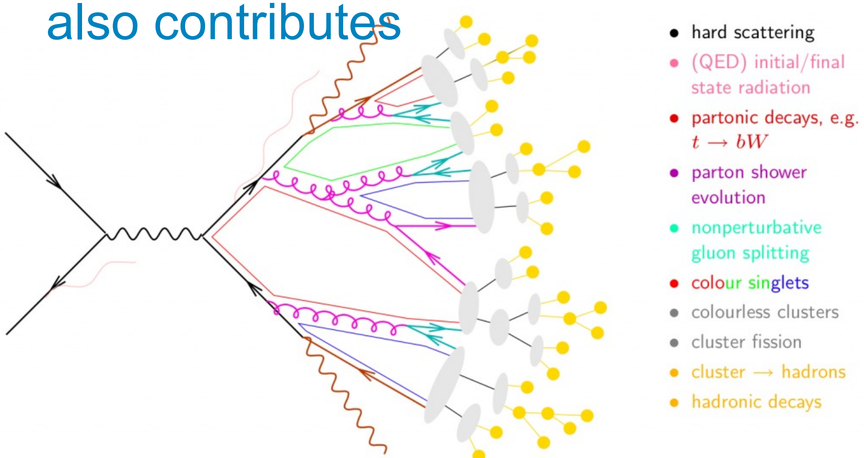
Building a Physical Model

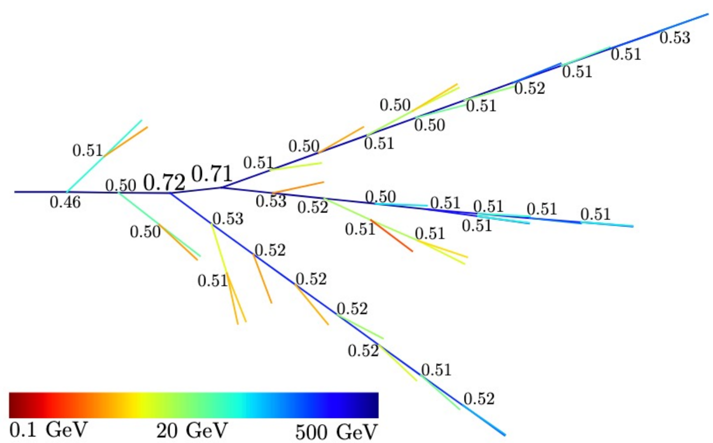
- JUNIPR builds jets by clustering components into a binary tree
  - Then learns the probability of that clustering being found in the sample
  - Maximize the log likelihood over the training data (can be used for discrimination with likelihood ratio thresholding)

$$P^{(t)}\left(k_1^{(t+1)}\dots\middle|k_1^{(t)}\dots\right) = P_{\text{end}}\left(\text{false}\middle|h^{(t)}\right) \times P_{\text{mother}}\left(m^{(t)}\middle|h^{(t)}\right) \times P_{\text{branch}}\left(k_{d_1}^{(t+1)} k_{d_2}^{(t+1)}\middle|k_m^{(t)}\, h^{(t)}\right) \quad (3)$$

- Can look at the classification probability at different branchings to understand what information is relevant to the decision
  - Expected three prong structure is most important
- Can look at what information is used at each branching
  - Multiplicity matters most, but angle (width) also contributes







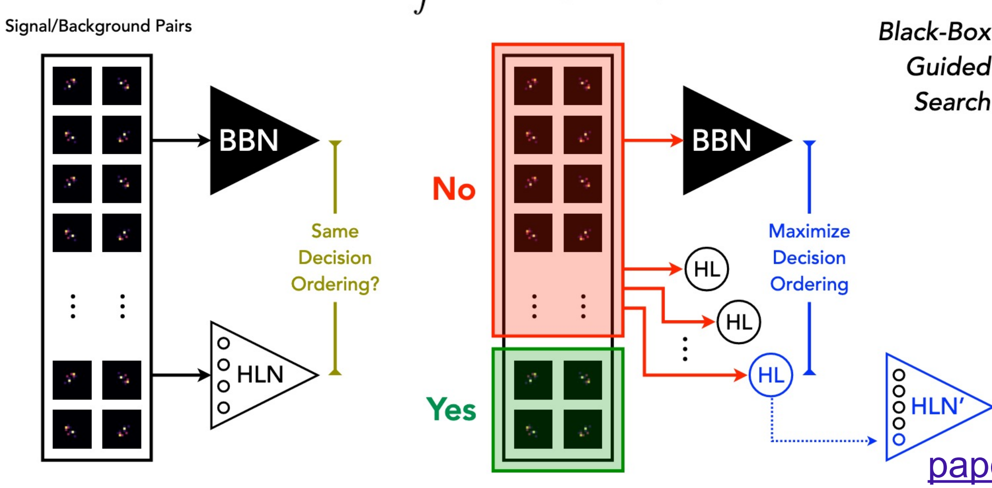
paper

Constructing Learned Information

- Use a CNN trained on low level information (jet images) to guide the construction of a simplified classifier based on high level interpretable features
  - Use average decision ordering to maximize the similarity between the decision boundaries of the two models
  - Use a black box guided search: iteratively selecting HL features that maximize ADO with the LL classifier
  - At each search step separate samples where HL and LL classifiers disagree
- The bulk of the CNN's power can be captured by 6 known jet features

$$\mathrm{DO}[f,g](x,x') = \Theta\Big(\big(f(x)-f(x')\big)\big(g(x)-g(x')\big)\Big)$$

$$\mathrm{ADO}[f,g] = \int dx\, dx'\, p_{\mathrm{sig}}(x)\, p_{\mathrm{bkg}}(x')\, \mathrm{DO}[f,g](x,x').$$

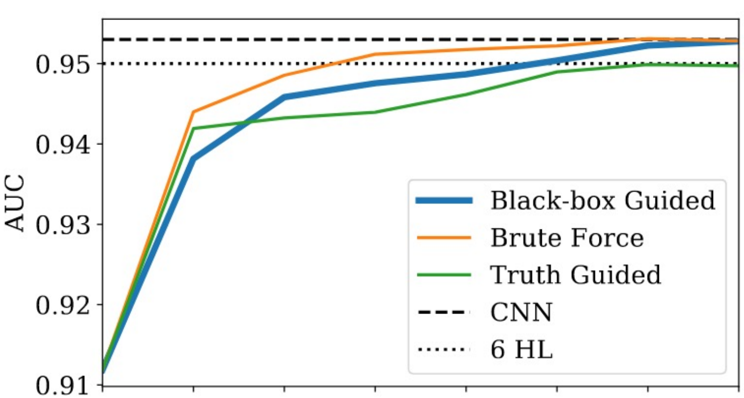


paper

Observable	AUC	ADO[CNN, Obs.]
M_{jet}	0.898 ± 0.004	0.807
$C_2^{\beta=1}$	0.660 ± 0.006	0.584
$C_2^{\beta=2}$	0.604 ± 0.007	0.548
$D_2^{\beta=1}$	0.790 ± 0.005	0.743
$D_2^{\beta=2}$	0.807 ± 0.005	0.762
$\tau_2^{\beta=1}$	0.662 ± 0.006	0.600
6HL	0.9504 ± 0.0002	0.971
CNN	0.9531 ± 0.0002	1.000
488HL	0.9535 ± 0.0002	0.978
$7\mathrm{HL}_{\text{black-box}}$	0.9528 ± 0.0003	0.971

Constructing Learned Information

- Define a basis space that captures a broad spectrum of physically interpretable information
  - Energy Flow Polynomials (EFPs): functions of momentum fraction of calorimeter cell and pairwise angular distance between cells
- Define a subspace of samples where 6-feature NN did not match CNN performance and search for EFP with max ADO
  - Identifies a new EFP that seems to help on edge cases
- Can use black box guided search directly on space of EFPs
  - Some EFPs identified are substantially different than traditional jet features



Iteration (n)	EFP	κ	β	Chrom #	$\mathrm{ADO}[\mathrm{EFP},\mathrm{CNN}]_{X_{n-1}}$	AUC[EFP]	$\mathrm{ADO}[\mathrm{HLN}_n,\mathrm{CNN}]_{X_{\mathrm{all}}}$	$\mathrm{AUC}[\mathrm{HLN}_n]$
0	$M_{\mathrm{jet}} + p_{\mathrm{T}}$	–	–	–	–	–	0.9259	0.9119
1		2	$\frac{1}{2}$	2	0.8144	0.8190	0.9570	0.9382
2		0	2	2	0.6377	0.8106	0.9673	0.9458
3		0	–	1	0.5460	0.6737	0.9692	0.9476
4		1	$\frac{1}{2}$	2	0.5274	0.8464	0.9712	0.9487
5		−1	–	1	0.5450	0.5882	0.9714	0.9504
6		1	$\frac{1}{2}$	4	0.5382	0.7678	0.9734	0.9523
7		−1	$\frac{1}{2}$	2	0.5561	0.5957	0.9741	0.9528

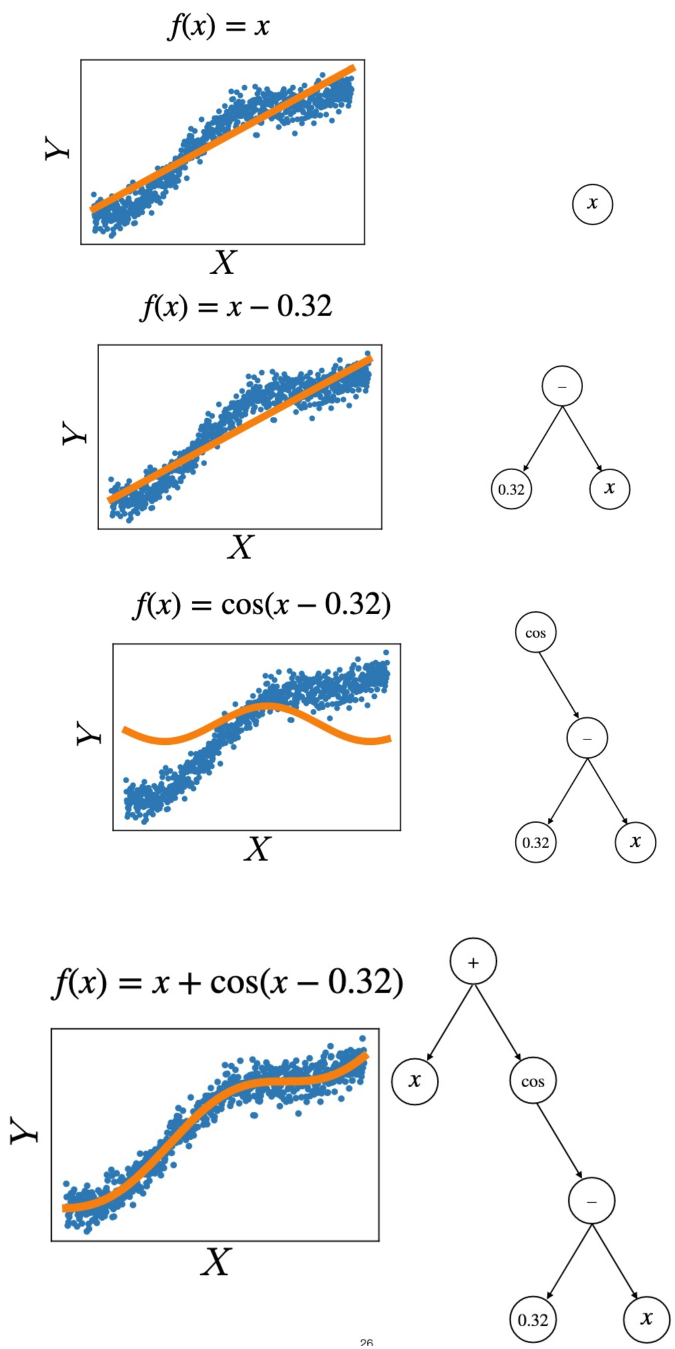
Implications and Limitations

- These methods give us a specific quantification on what the network is learning in terms of what we already know
- By directly parametrizing the information in terms of known features we ensure learned information is not a statistical artifact
- Building a robust classifier with a reduced feature set enables better uncertainty quantification
- For some problems we don't have a nice basis space of features to search over
  - These bases don't provide full coverage, unable to characterize other learned information

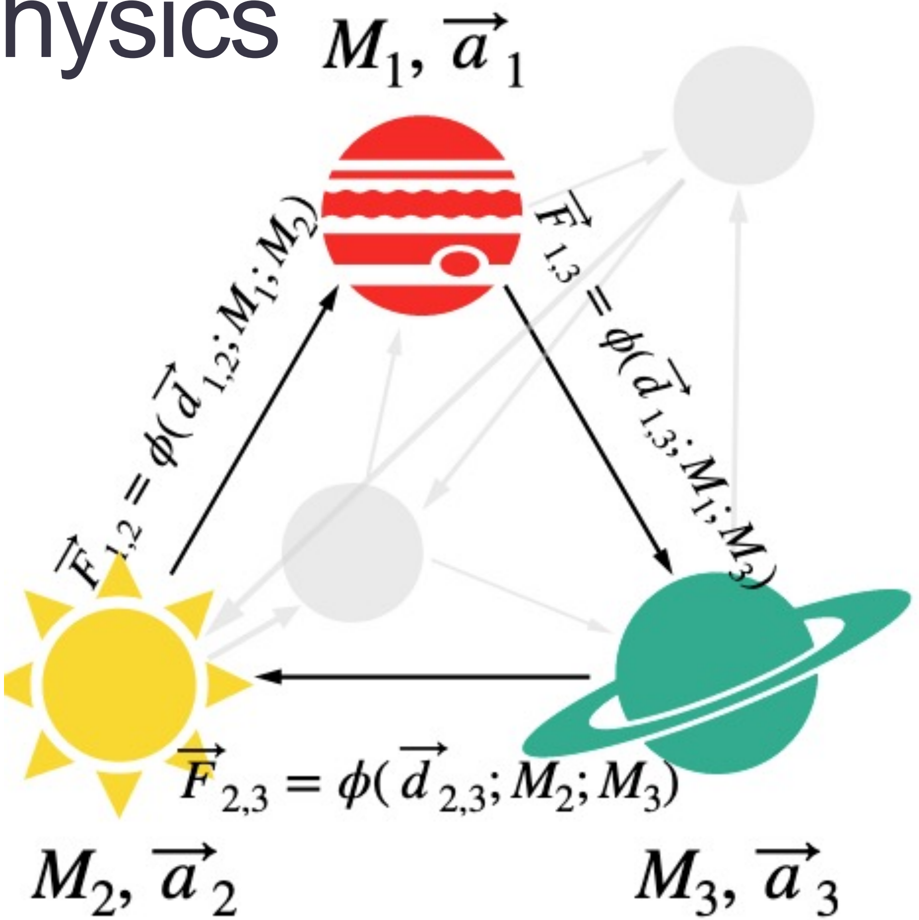
Mapping Back to Math

Symbolic Regression

- Finds an analytic equation that mimics the predictions of a trained ML model
  - Find the analytic function that maps your inputs to the outputs of your model
  - By cleverly setting up your ML model you reduce the space of functions to search over
- Typically done with a genetic algorithm
  - Recursively build a function using basis space of input variables, operators, and constants (through crossover and mutation)
  - Minimize error between function and ML prediction
  - Result is a set of possible equations
  - Can enforce constraints like penalizing complexity



Learning Astrophysics

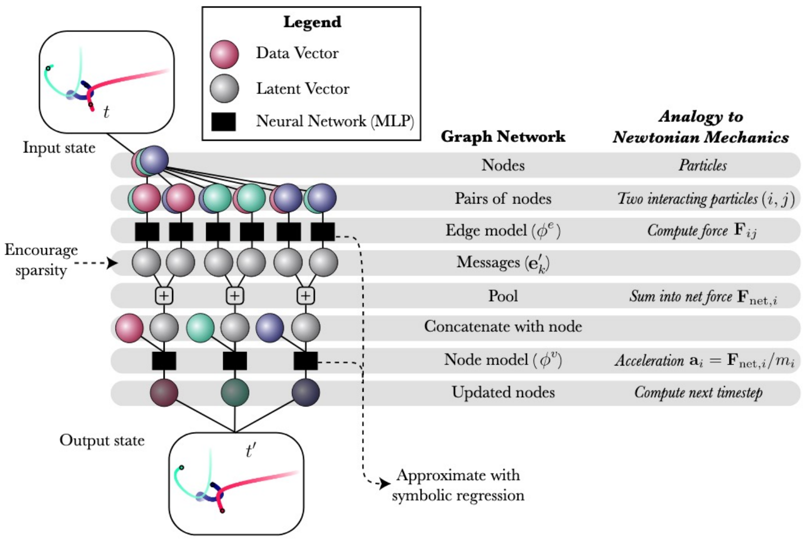


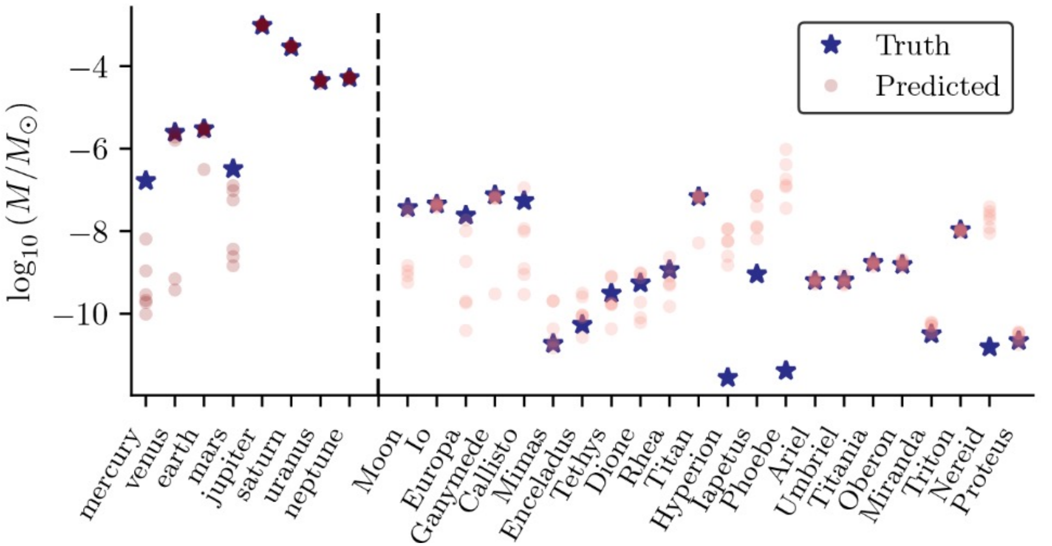
1. Our inputs are the positions of the bodies
2. They are converted into pairwise distances
3. **Our model tries to guess a mass for each body**
4. **It then also guesses a force, that is a function of distance and masses**
5. **Using Newton's laws of motion ($\sum \vec{F} = M\vec{a}$) it converts the forces into accelerations**
6. **Finally, it compares this predicted acceleration, with the true acceleration from the data**

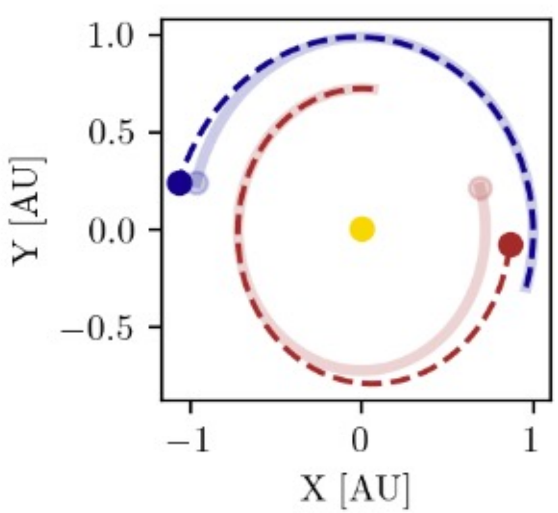
Minimize
$$\left| \vec{a}(\text{pred}) - \vec{a}(\text{true}) \right|^2$$

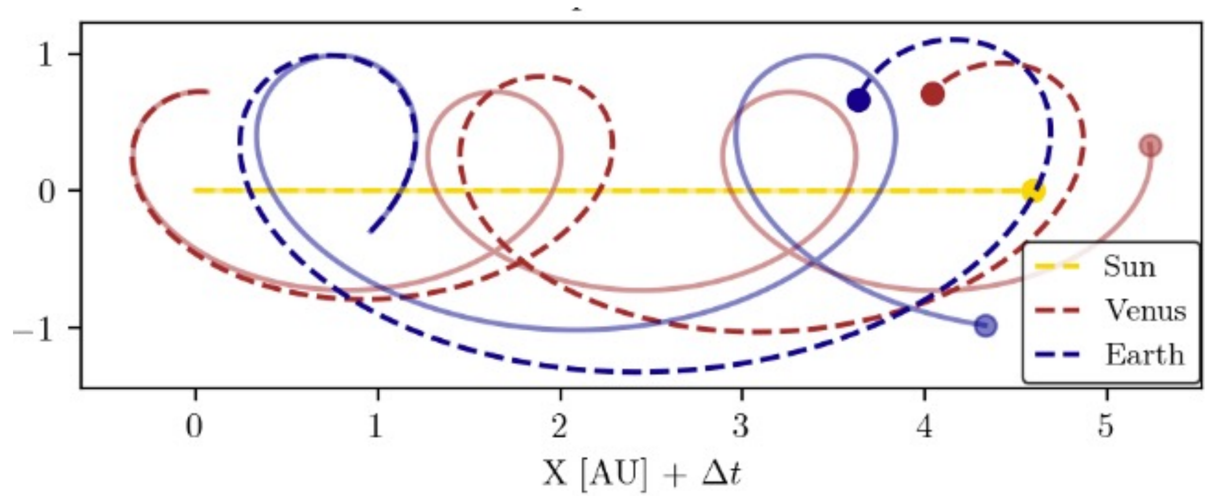
paper

Learning Astrophysics

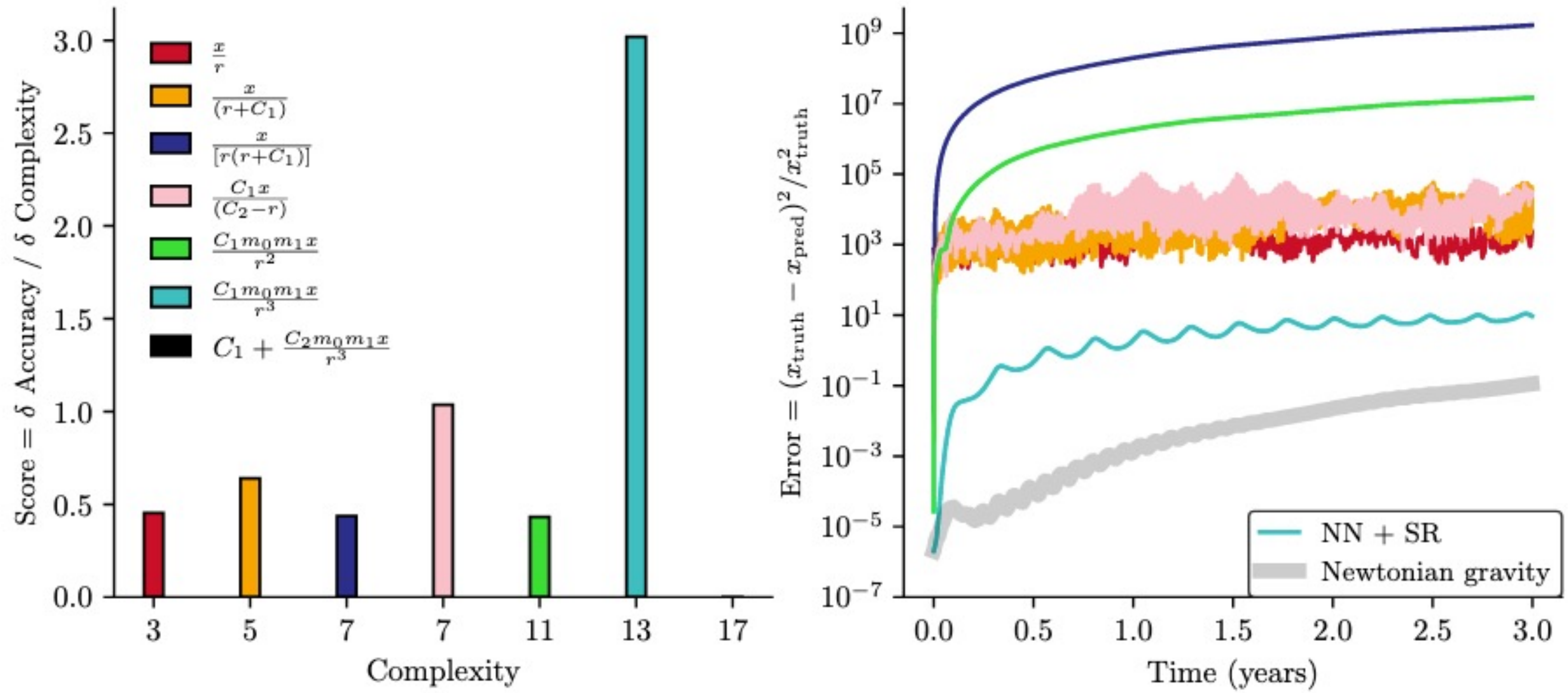








Extracting the Physics



- Apply symbolic regression to the GNN messages (forces) with a constraint to balance accuracy and equation complexity
- Can substitute learned equation for the force guess to improve the simulator and the mass predictions (node predictions)

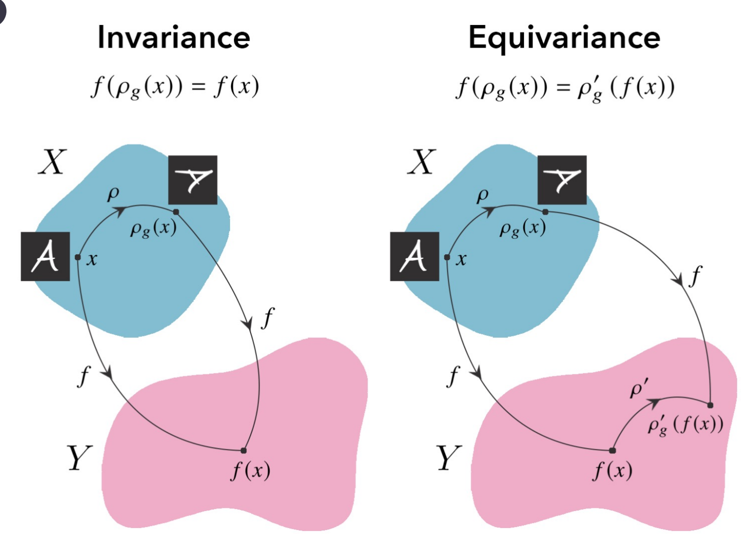
Implications and Limitations

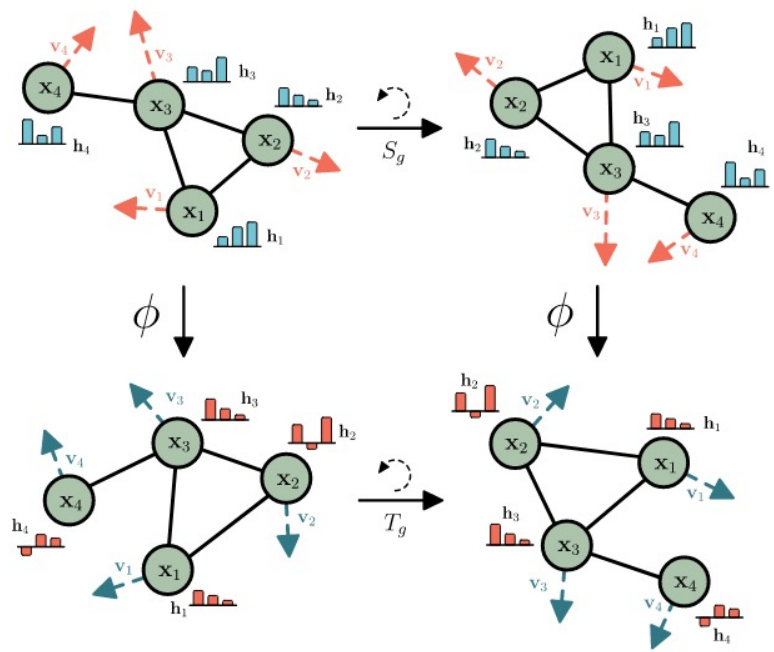
- This process had been successfully applied to more complex systems (estimating galaxies dark matter halo)
- 'New' equations could be used to guide future experiments
  - Can we validate an equation's predictions are accurate, does it describe a new particle or force with additional implications?
- How do we know which equation to pick (smallest error might not always be the correct equation)
  - Simplicity of an equation as a decision factor is a big assumption
- How do we decide what constraints and priors to incorporate into the model
  - Doesn't allow for the possibility that any of these constraints are wrong
- How do you account for uncertainties/mismodelings in the synthetic data or reconstruction software
  - Is the ML model decision actually describing nature

My Current Problem

Equivariance in GNNs

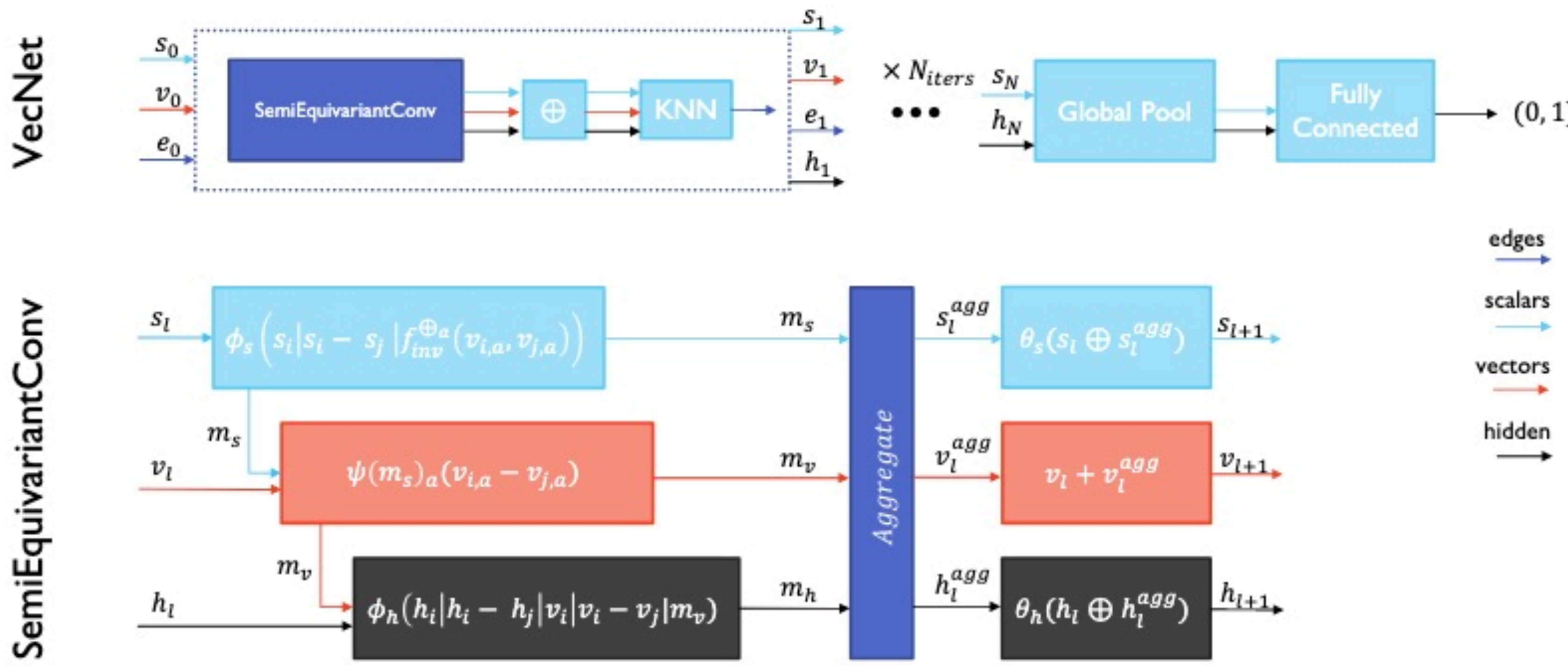
- Many physics datasets are governed by natural symmetries
  - Invariance: output of a model doesn't change when inputs transformed under symmetry
  - Equivariance: output of model changes in a specific way when inputs transformed under symmetry
- Constraining the functions learned by a network could help reduce model size and training resources





VecNet

Built an architecture that abstracts out different GNN design choices (including equivariance) as hyperparameters



Paper

Building the Most Efficient Model

- Looked at top jet tagging
  - Each jet candidate has up to 200 constituents
- Enforced Lorentz equivariance
  - Described by $O^+(1,3)$ group
- Conduct hyperparameter sweep and importance analysis on accuracy and ant factor

The effect of a Lorentz boost in the x-direction,

$$\begin{pmatrix} t' \\ x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} \gamma & -\gamma v & 0 & 0 \\ -\gamma v & \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} t \\ x \\ y \\ z \end{pmatrix}, \qquad \gamma = \frac{1}{\sqrt{1-v^2}},$$

Spacetime interval:

$$|v_0^\mu - v_1^\mu|^2 = (v_0^0 - v_1^0)^2 - \sum_{i=1}^3 (v_0^i - v_1^i)^2$$

$$ant = \frac{\text{accuracy}}{\text{model size}} = \frac{\epsilon_B^{-1}}{N_{params}}$$

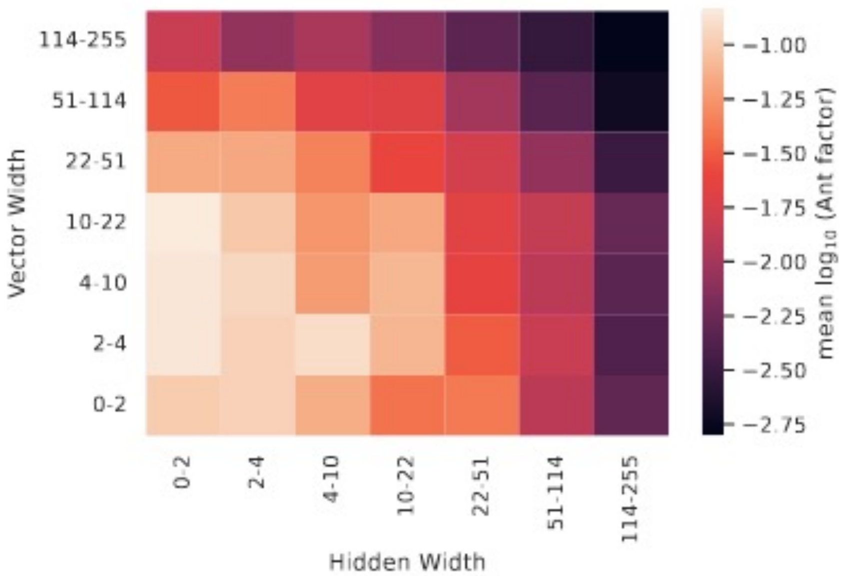
Hyperparameter	AUC		Ant Factor	
	Importance	Coefficient	Importance	Coefficient
Hidden width	0.344	0.044	0.443	-0.092
Vector width	0.193	-0.055	0.232	-0.049
Scalar width	0.194	-0.005	0.226	-0.056
Batch norm	0.017	-0.006	0.016	-0.01
Layer norm	0.029	0.006	0.034	-0.001
Num. node layers	0.157	0.126	0.01	0.014
Num. edge layers	0.056	0.036	0.023	0.01
LR decay factor	0.012	-0.057	0.016	0.086

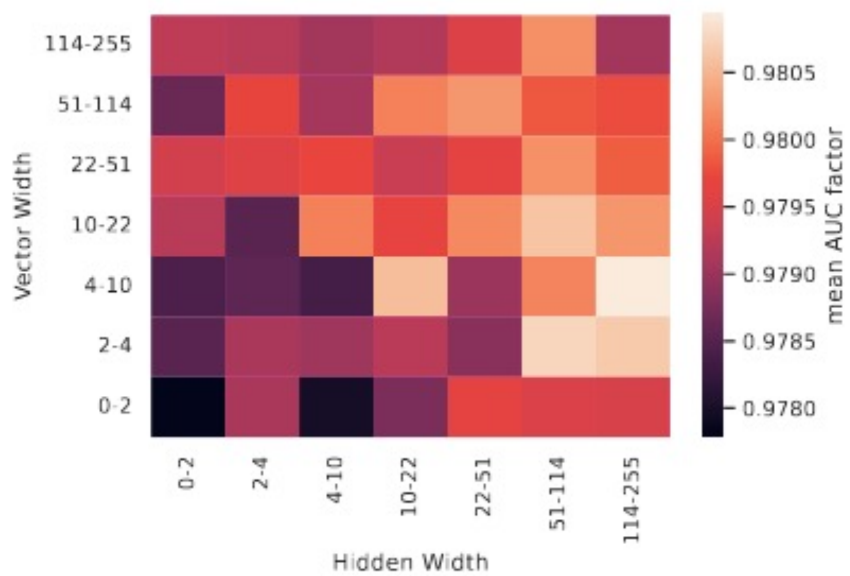
Model	Accuracy	AUC	ϵ_B^{-1}	N_{params}	Ant
ResNeXt	0.936	0.984	1122 ± 47	1.46M	0.0007
ParticleNet	0.938	0.985	1298 ± 46	498k	0.0026
EFP	0.932	0.980	384	1k	0.384
LGN	0.929	0.964	435 ± 95	4.5k	0.097
VecNet (Ours)	0.935	0.984	4046	633k	0.006
	0.931	0.981	3482	15k	0.229

Semi-Equivariance

We find there is a non-trivial optimum combining equivariant and non-equivariant information

- Most resource efficient configuration combined 4 hidden channels, 2 vector channels, and 8 scalar channels





What to do?

- Want to characterize what the non-equivariant model channels are learning
- Thought: is this just a feature of constrained optimization problems being harder to solve?
- Ideas:
    - Local white-box approximators (but EFPs and other jet features should be equivariant)
    - Graph rewiring (does changing the allowed information flow affect the decision)
    - Relevance propagation (but how to interpret)
    - Suggestions?!

Thank you!

Happy to answer any questions!

sthais@princeton.edu @basicsciencesav

E(N) Approach to Equivariance

- Develops an architecture that is translation, rotation, and reflection equivariant
  - Without requiring parameterizing the basis space of the model transformations using spherical harmonics/group representations
- Provides relative square distance of node coordinates (equivariant quantity) as input to the edge convolution
  - Node positions are updated as weighted sum of relative distance
  - Maintains equivariance without limiting the space of convolutions (better expressivity)
  - Can extend to vector quantities on nodes

$$\mathbf{m}_{ij} = \phi_e\left(\mathbf{h}_i^l, \mathbf{h}_j^l, \left\|\mathbf{x}_i^l - \mathbf{x}_j^l\right\|^2, a_{ij}\right)$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + C\sum_{j\neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x\left(\mathbf{m}_{ij}\right)$$

$$\mathbf{m}_i = \sum_{j\neq i}\mathbf{m}_{ij}$$

$$\mathbf{h}_i^{l+1} = \phi_h\left(\mathbf{h}_i^l, \mathbf{m}_i\right)$$

$$\mathbf{v}_i^{l+1} = \phi_v\left(\mathbf{h}_i^l\right)\mathbf{v}_i^{\text{init}} + C\sum_{j\neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x\left(\mathbf{m}_{ij}\right)$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + \mathbf{v}_i^{l+1}$$

Original Paper

	GNN	Radial Field	TFN	Schnet	EGNN
Edge	$\mathbf{m}_{ij} = \phi_e(\mathbf{h}_i^l, \mathbf{h}_j^l, a_{ij})$	$\mathbf{m}_{ij} = \phi_{\text{rf}}(\|\mathbf{r}_{ij}^l\|)\mathbf{r}_{ij}^l$	$\mathbf{m}_{ij} = \sum_k \mathbf{W}^{lk}\mathbf{r}_{ji}^l\mathbf{h}_i^{lk}$	$\mathbf{m}_{ij} = \phi_{\text{cf}}(\|\mathbf{r}_{ij}^l\|)\phi_s(\mathbf{h}_j^l)$	$\mathbf{m}_{ij} = \phi_e(\mathbf{h}_i^l, \mathbf{h}_j^l, \|\mathbf{r}_{ij}^l\|^2, a_{ij})$ $\hat{\mathbf{m}}_{ij} = \mathbf{r}_{ij}^l\phi_x(\mathbf{m}_{ij})$
Agg	$\mathbf{m}_i = \sum_{j\in\mathcal{N}(i)}\mathbf{m}_{ij}$	$\mathbf{m}_i = \sum_{j\neq i}\mathbf{m}_{ij}$	$\mathbf{m}_i = \sum_{j\neq i}\mathbf{m}_{ij}$	$\mathbf{m}_i = \sum_{j\neq i}\mathbf{m}_{ij}$	$\mathbf{m}_i = \sum_{j\neq i}\mathbf{m}_{ij}$ $\hat{\mathbf{m}}_i = C\sum_{j\neq i}\hat{\mathbf{m}}_{ij}$
Node	$\mathbf{h}_i^{l+1} = \phi_h(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + \mathbf{m}_i$	$\mathbf{h}_i^{l+1} = w^{ll}\mathbf{h}_i^l + \mathbf{m}_i$	$\mathbf{h}_i^{l+1} = \phi_h(\mathbf{h}_i^l, \mathbf{m}_i)$	$\mathbf{h}_i^{l+1} = \phi_h\left(\mathbf{h}_i^l, \mathbf{m}_i\right)$ $\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + \hat{\mathbf{m}}_i$
	Non-equivariant	E(n)-Equivariant	SE(3)-Equivariant	E(n)-Invariant	E(n)-Equivariant