GNN INTERPRETABILITY IN HEP

Savannah Thais

Mini-workshop on GNNs for Tracking

06/03/2022

iris hep

Princeton Research Computing

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

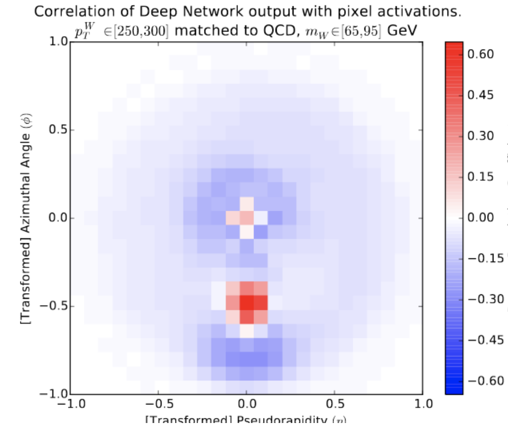
Outline

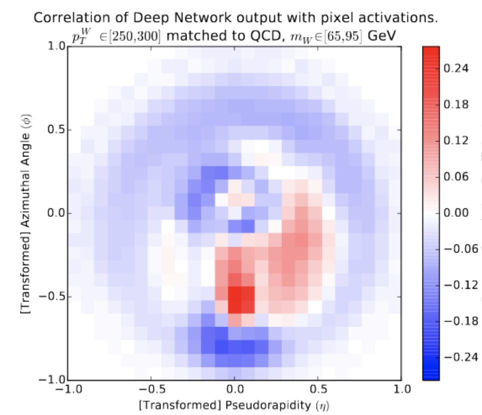
- Three examples of interpretability techniques applied to GNNs (or similar models) in HEP
    1. Feature Importance and relevance propagation
    2. Decision approximation
    3. Symbolic regression
- GNN-focused interpretability techniques from industry
    1. Perturbation-based explainability
    2. Graph filters and kernels
    3. Disentangled representation learning
- Some prompts for discussion

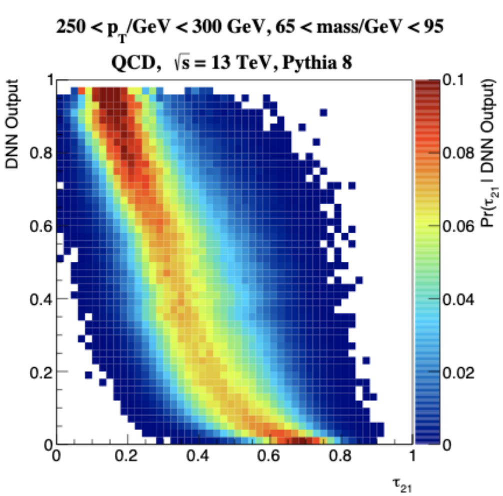
Feature Importance and Relevance Propagation

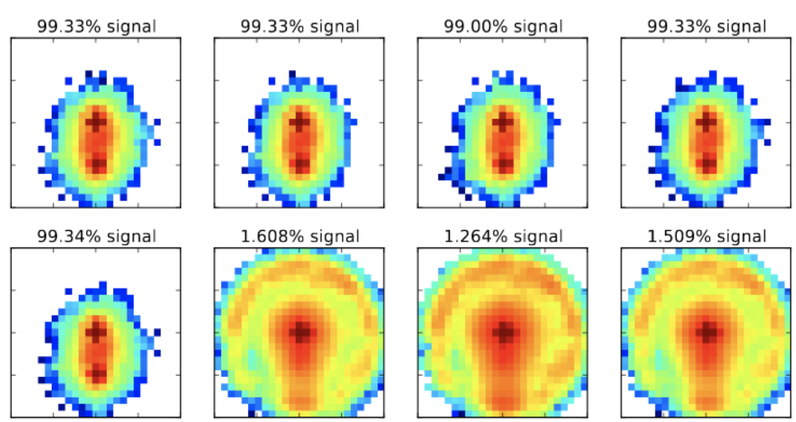
CNN Interpretation

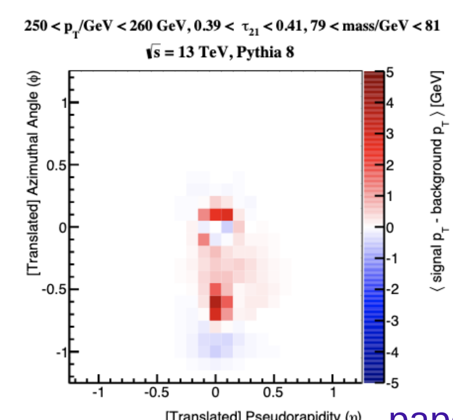
- Look at correlation of CNN output with standard physics features → it's learning thing we expect to be important
- Look at average of images with highest activations for last hidden layer → presence of secondary core is informative
- Look at per pixel correlation with CNN output (doesn't map to a known 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?











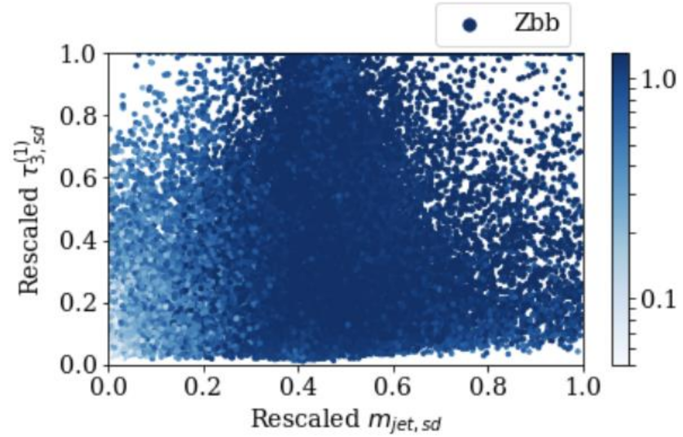
paper

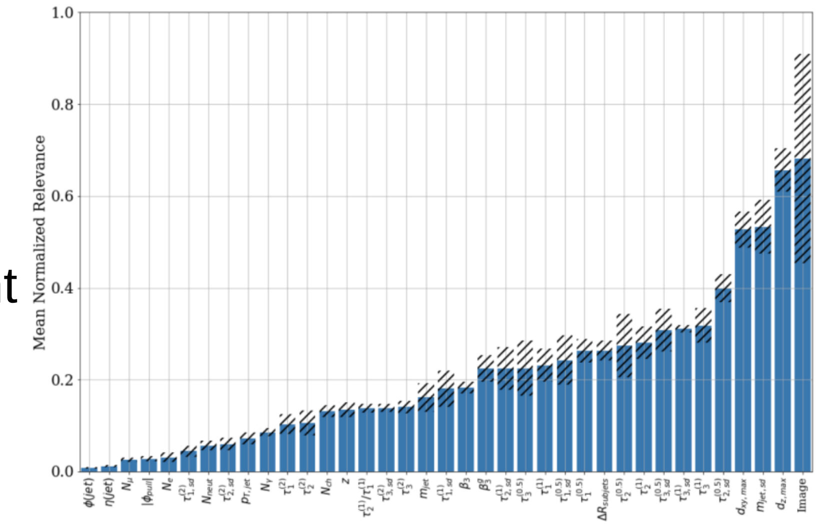
Adding In Expertise

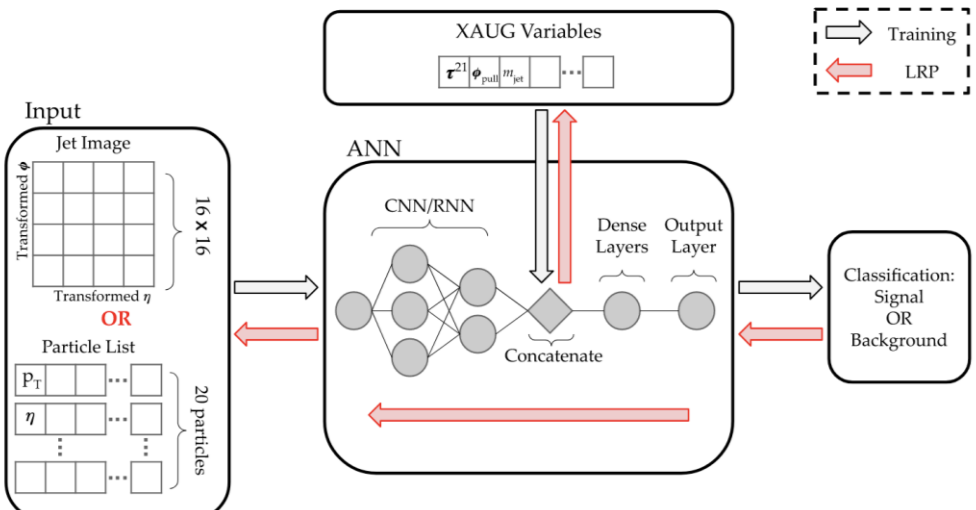
- Augment the CNN with physics-motivated features after initial prediction
- Use LRP to understand what information the network is using

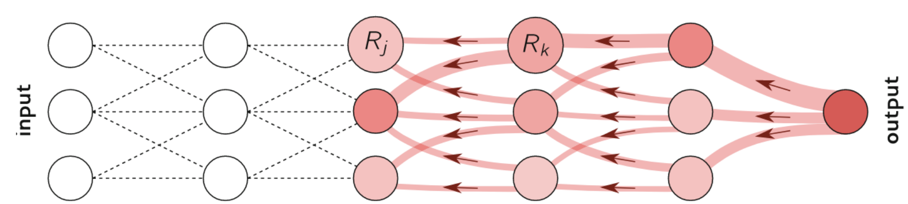
$$R_j = \sum_k \frac{z_{jk}}{\sum_j z_{jk}} R_k.$$

  - 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



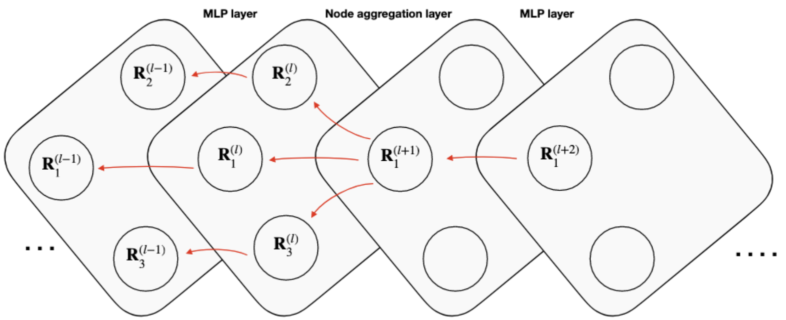






paper

Extending to GNNs



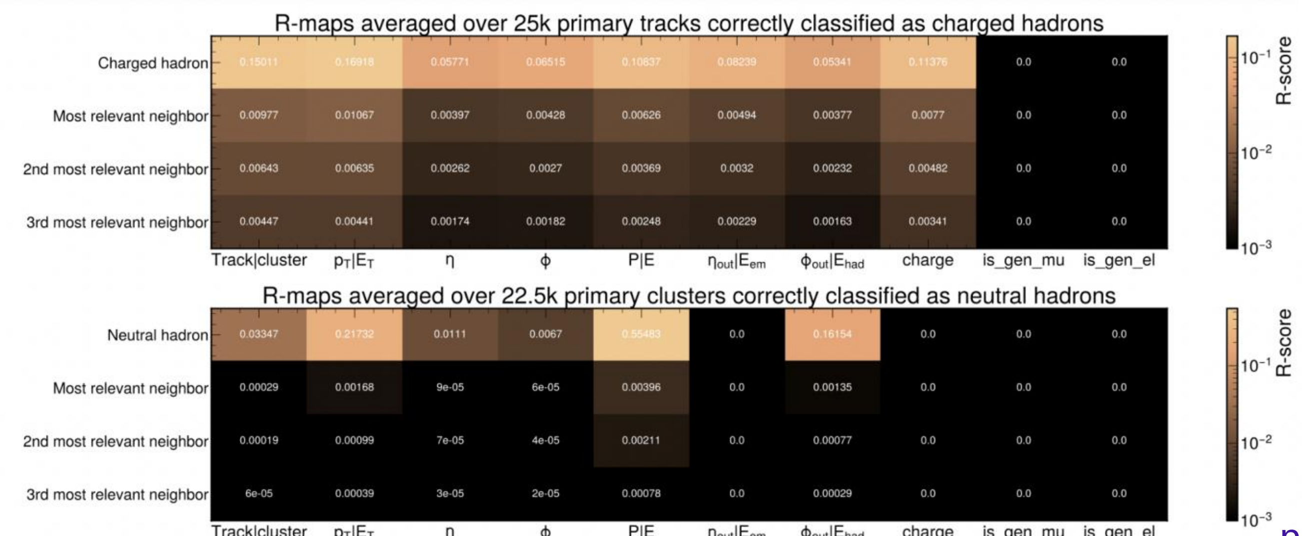
- MLPF uses graphs of reconstructed detector elements for node classification

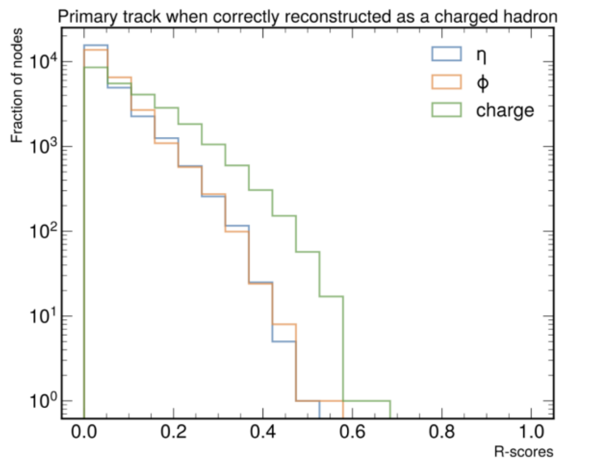
$$x_i = [\text{type}, p_{\mathrm{T}}, \eta, \phi, \eta_{\text{out}}, \phi_{\text{out}}, E_{\text{ECAL}}, E_{\text{HCAL}}, \text{charge}, \text{is\_gen\_el}, \text{is\_gen\_mu}]$$
$$y_i = [\text{PID}, p_{\mathrm{T}}, E, \eta, \phi, \text{charge}]$$

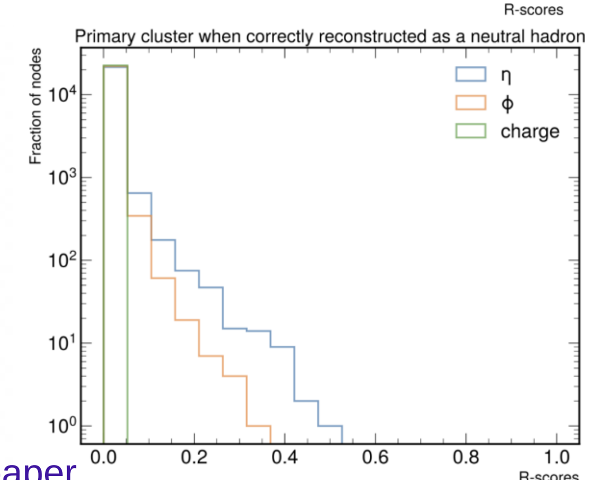
- GNNs are NNs with an aggregation step
  - Modify LRP formula to distribute aggregated neighborhood information to un-aggregated nodes in previous layer

$$\mathbf{R}_j^{(l)} = \sum_k \frac{x_j A_{jk}}{\sum_m x_m A_{mk}} \mathbf{R}_k^{(l+1)}$$

  - Construct R-score tensor for output neurons of each node
- Create R-maps: sample tensors for output classes, sort nodes by relevance, average over events







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

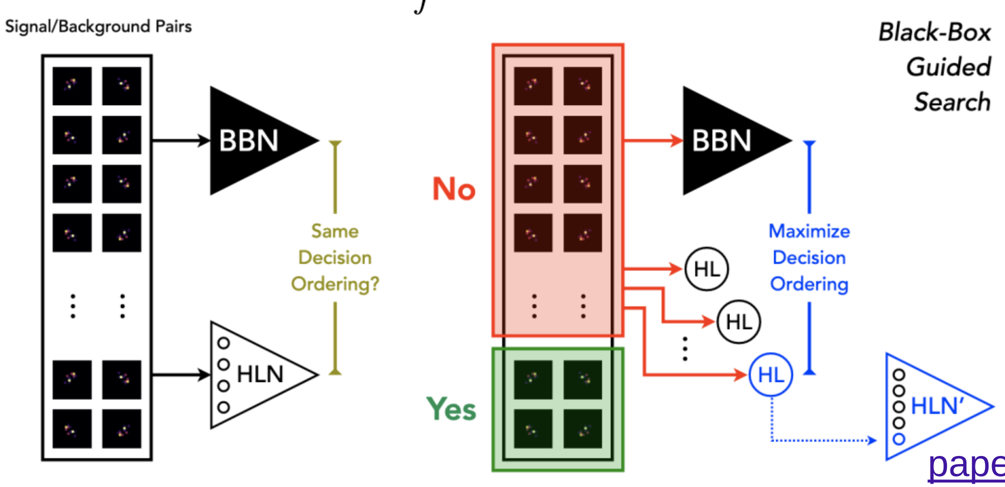
Using Physics Knowledge as a Basis

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

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

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

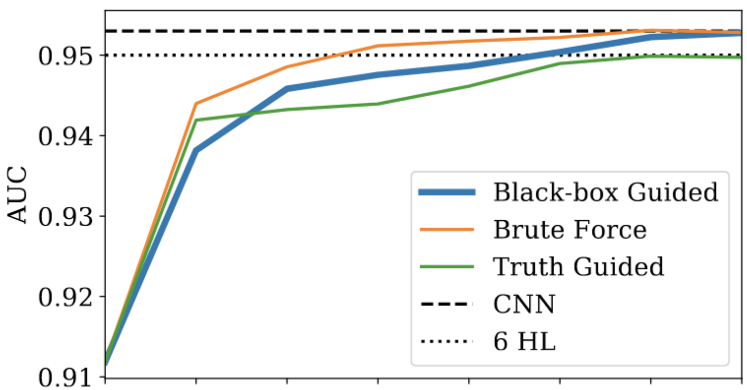


paper

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 #	$\mathrm{ADO[EFP, CNN]}_{X_{n-1}}$	AUC[EFP]	$\mathrm{ADO[HLN}_n\mathrm{, CNN]}_{X_{\mathrm{all}}}$	$\mathrm{AUC[HLN}_n]$
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

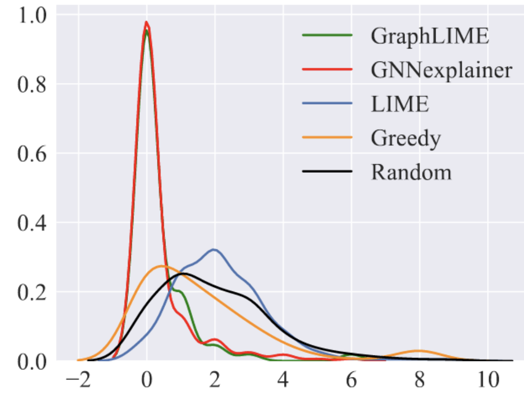
Extending to GNNs

- Local black box approximator learns an interpretable non-linear model locally in the subgraph of an individual node
  - E.g. linear regression, decision trees, etc
- Use HSIC Lasso for feature selection to approximate decision in an n-hop node-neighborhood
  - Minimize Lasso loss across pre-determined set of features

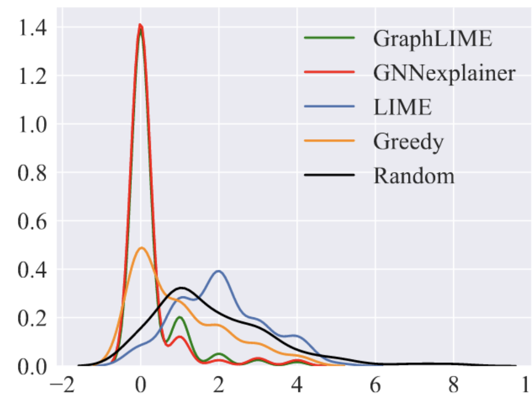
$$\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \frac{1}{2}\|\bar{\boldsymbol{L}} - \sum_{k=1}^{d} \beta_k \bar{\boldsymbol{K}}^{(k)}\|_F^2 + \rho\|\boldsymbol{\beta}\|_1$$

$$\text{s.t.} \quad \beta_1, \ldots, \beta_d \geq 0,$$

- Demonstrated robustness to noisy and correlated features
- Still a bit of a gap on how to interpret the locality of a decision



(a)



Algorithm 1 Locally nonlinear Explanation: GraphLIME

Input: GNN classifier f, Number of explanation features K
Input: the graph $\mathcal{G}$, the node v being explained
Output: K explanation features

$\boldsymbol{X}_n = N\_hop\_neighbor\_sample(v)$
$\boldsymbol{Z} \leftarrow \{\}$
for all $x_i \in \boldsymbol{X}_n$ **do**
    $y_i = f(\boldsymbol{x}_i)$
    $\boldsymbol{Z} \leftarrow \boldsymbol{Z} \cup (\boldsymbol{x}_i, y_i)$
end for
$\boldsymbol{\beta} \leftarrow$ HSIC Lasso$(\boldsymbol{Z})$ $\triangleright$ with x_i as features, y_i as label
$\zeta(v) \leftarrow$ Top-K features as explanations based on $\boldsymbol{\beta}$

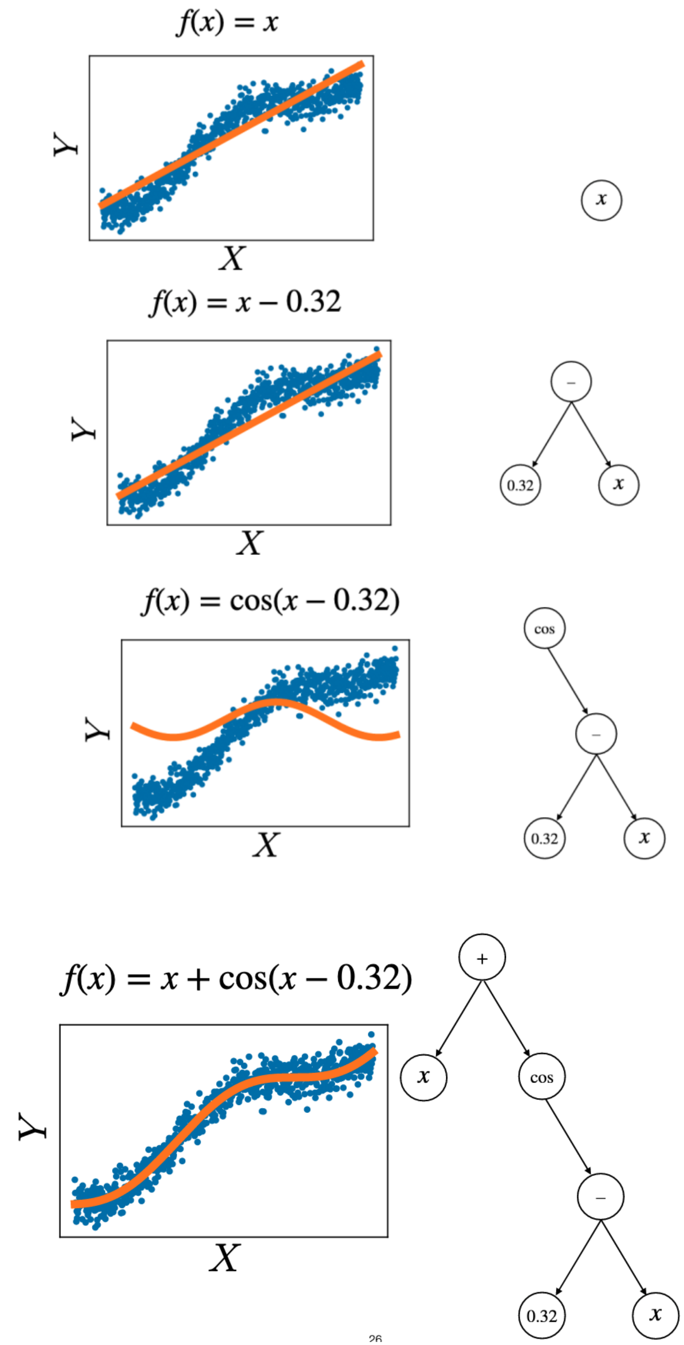
Implications and Limitations

- 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
- Saliency maps have known issues for discrete/sparse input structures like adjacency matrices
- 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

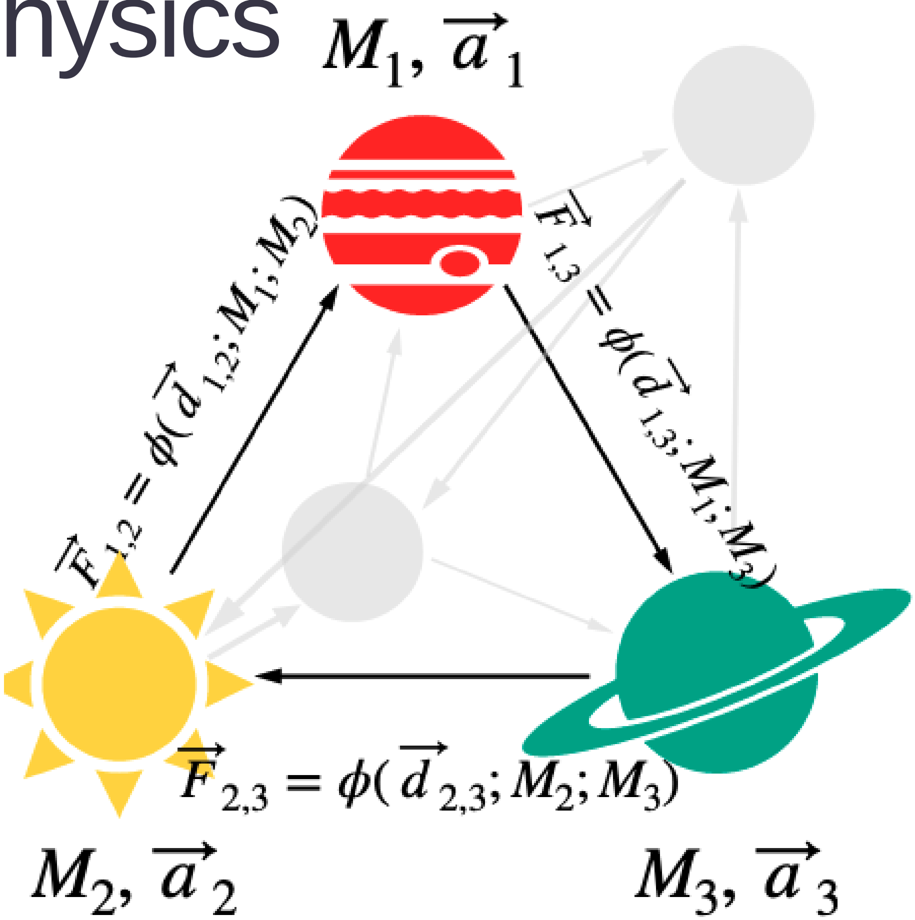
Mapping Back to Math

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



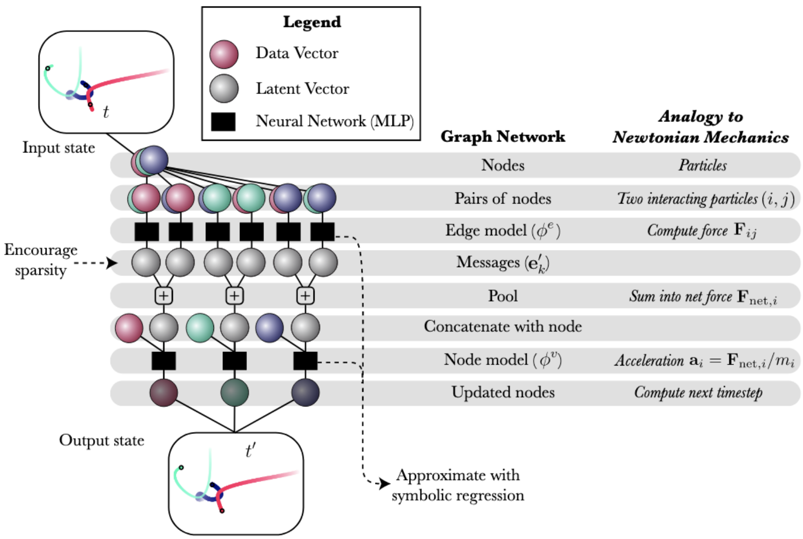
1. Our inputs are the positions of the bodies
2. They are converted into pairwise distances
3. **Our model tries to guess a mass for each body**
4. **It then also guesses a force, that is a function of distance and masses**
5. **Using Newton's laws of motion ($\sum \vec{F} = M\vec{a}$) it converts the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

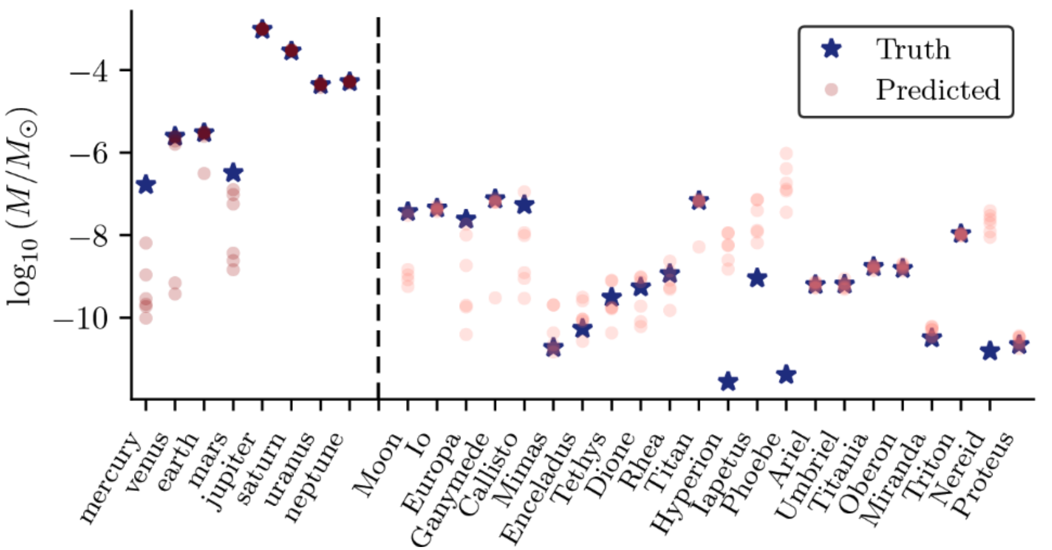
Minimize

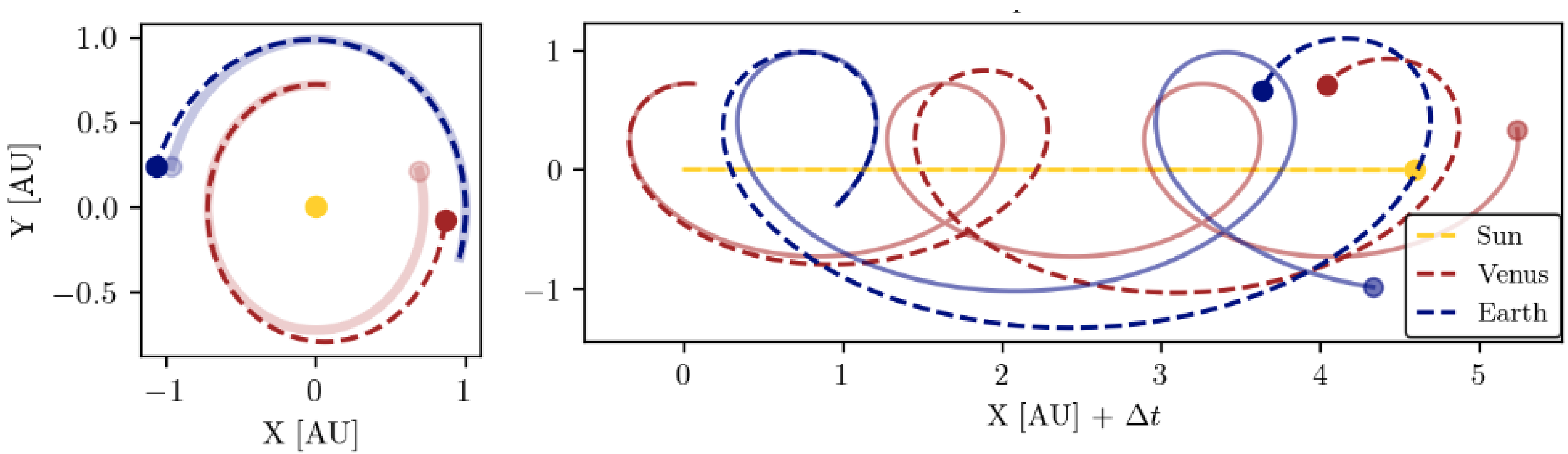
$$\left| \vec{a}(\text{pred}) - \vec{a}(\text{true}) \right|^2$$

paper

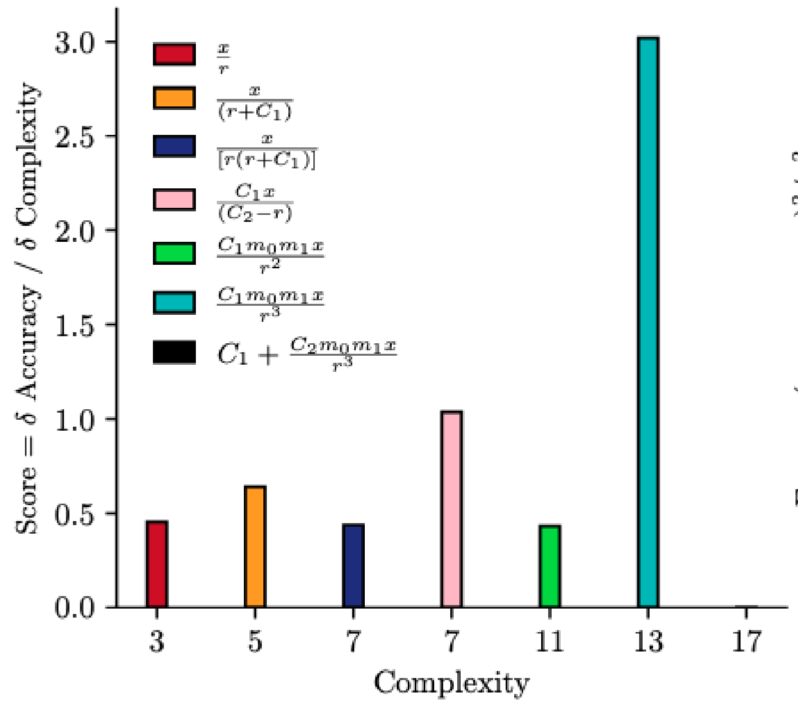
Learning Astrophysics

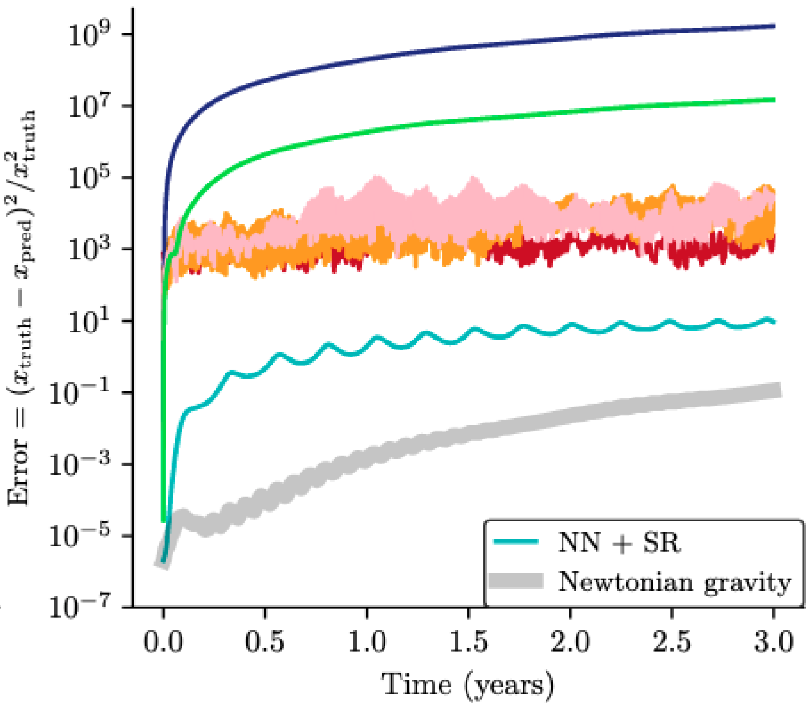






Extracting the Physics





- 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

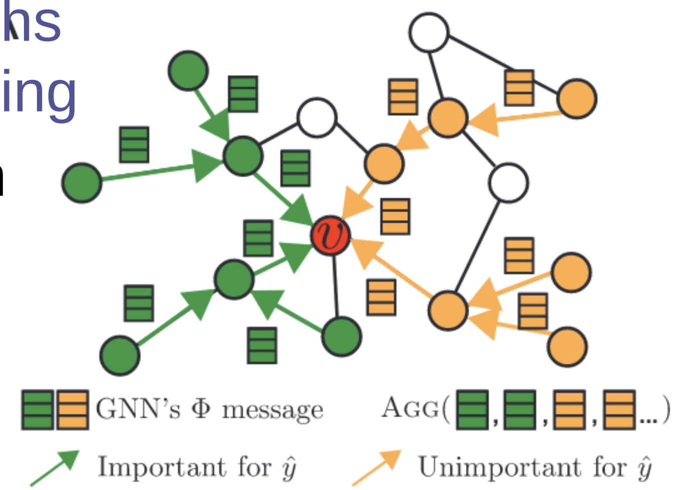
- How do we decide what constraints and priors to incorporate into the model
  - Doesn’t allow for the possibility that any of these constraints are wrong

- How do you account for uncertainties/mismodelings in the synthetic data or reconstruction software
  - Is the ML model decision actually describing nature

Other GNN Interpretability Methods

Perturbation-based Explainability

- **Idea**: identify maximally relevant subgraphs that contribute to mutual information sharing
- GNNExplainer learns a real-valued graph mask with mean field variational approximation
  - Maximize change in prediction probability by reducing computation to subgraph
  - Qualitatively allows edge/node based counterfactuals
  - Can also learn a feature mask

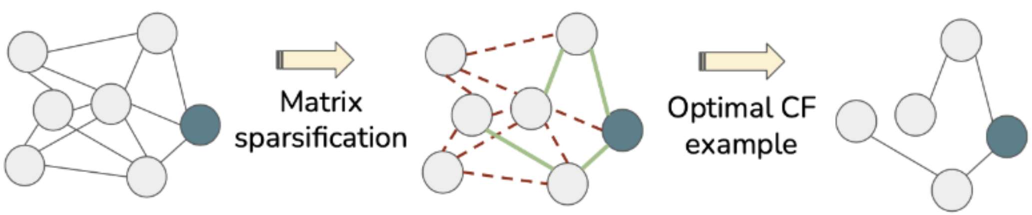


$$\max_{G_S} MI\left(Y, (G_S, X_S)\right) = H(Y) - H(Y|G = G_S, X = X_S).$$

$$H(Y|G=G_S, X=X_S) = -\mathbb{E}_{Y|G_S,X_S}\left[\log P_\Phi(Y|G=G_S, X=X_S)\right]$$

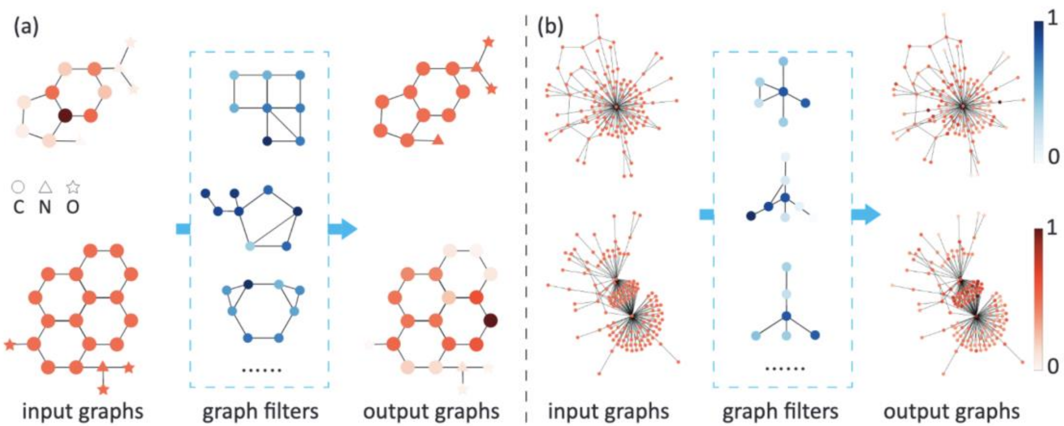
- CF-GNNExplainer uses dynamic edge-deletion to identify the minimum prediction-altering subgraph
  - Use adjacency matrix sparsification to minimize difference between performance on original and perturbed graph

$$\mathcal{L} = \mathcal{L}_{pred}(v, \bar{v} \mid f, g) + \beta \mathcal{L}_{dist}(v, \bar{v} \mid d),$$



Graph Filters and Kernels

- **Idea**: integrate graph kernels (as filters) into the GNN message passing step and use CNN filter interpretation methods
- Substitute graph kernels for neighborhood feature aggregation
  - Take kernel function to compare trainable hidden subgraphs with local node neighborhood and use kernel to update the node
  - The ‘filter’ is used to learn the hidden subgraph
  - Kernels can be calculated in different ways
- Can directly visualize the graph filters and application to input graph
  - Interpretability here is not precise, but can provide some intuition
  - Gain a sense of structure shapes that are important across dataset



Algorithm 1: Forward pass in l-th KerGNN layer

Input: Graph $G = (\mathcal{V}, \mathcal{E})$; Input node feature maps $\{\phi_{l-1}(v) : v \in \mathcal{V}\}$; Graph filters $\{H_i^{(l)} : i = 1, ..., d_l\}$; Graph kernel function K

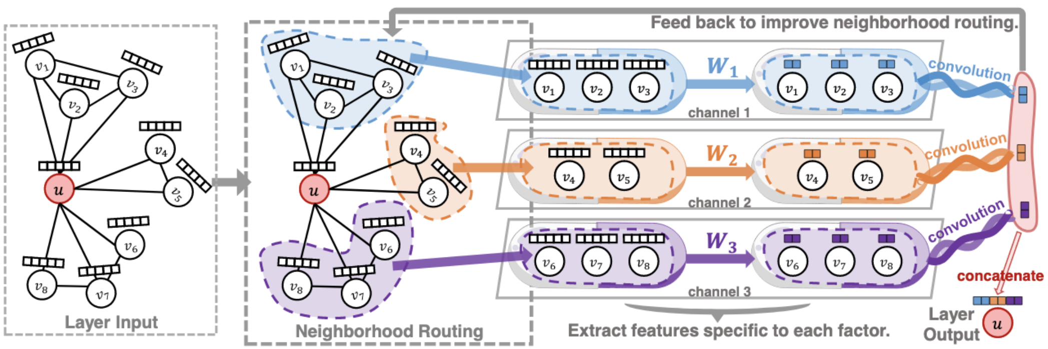
Output: Graph $G = (\mathcal{V}, \mathcal{E})$; Output node feature maps $\{\phi_l(v) : v \in \mathcal{V}\}$

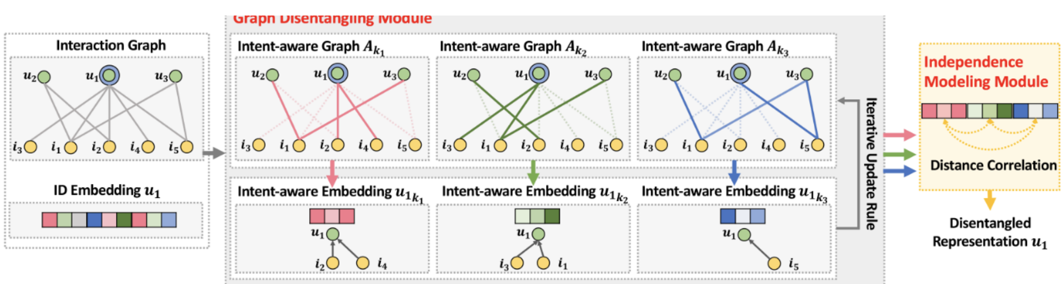
for $v \in \mathcal{V}$ **do**
  $G_v = \text{subgraph}(\{v\} \cup \mathcal{N}(v))$;
  for $i = 1$ **to** d_l **do**
    $\phi_{l,i}(v) = K(G_v, H_i^{(l)})$;
  end for
end for

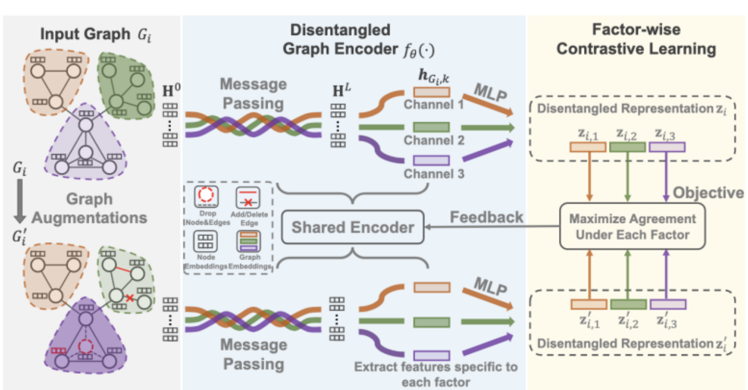
paper

Disentangled Representation Learning

- **Idea**: separate input information into features in learned latent space such that they can be mapped to interpretable values
- Different methods are proposed to learn these representations
  - Disentangled graph convolutional networks: use neighborhood routing to identify the factor that contributes to an edge relationship by assigning neighbors to learned channels
  - Disentangled graph collaborative filtering: learn a distribution over defined features for each edge relation
  - Disentangled contrastive learning: use a factor-wise discrimination objective in training to force embedding dimensions to describe different information







Discussion Prompts

- Is interpretability necessary? If we can demonstrate robustness and accuracy do we need to understand the model?
- How precise does interpretability need to be?
  - Do we need to map to fundamental physical values?
  - Are things like counterfactual explanations valuable in a physics context?
- How do we encourage the adoption of interpretable and explainable AI techniques from broader ML field?
- For GNNs specifically, can we consider methods like symmetry enforcement or attention mechanisms as interpretability?
- Is interpretability fundamentally in opposition to robustness?
  - Recent paper "Attribution based explanations that provide recourse cannot be robust"

Thank you!

Looking forward to an interesting discussion!

sthais@princeton.edu @basicsciencesav