Symbolic Distillation of Graph Neural Networks

Executive Summary

Liz Tan — Data Intensive Science MPhil — University of Cambridge

MOTIVATION

Understanding the underlying physical laws of a system from data is a central goal in the physical sciences. Historically, empirical physical laws were found by eye. However, with the rise of high-dimensional data and complex systems, this manual approach becomes infeasible. Deep learning architectures are efficient in analysing high-dimensional data, but these models are often black boxes offering limited interpretability of the mechanisms they capture. In contrast, symbolic regression provides interpretable equations but the algorithm struggles to scale with high-dimensional inputs.

This project reproduces a framework introduced by Cranmer (2020) [1] that combines Graph Neural Networks (GNNs) with symbolic regression to reconstruct known force laws from high-dimensional particle simulation data.

METHODS

Training

We trained GNNs to predict particle accelerations from simulated 2-D n-body datasets, each governed by a known physical force law. The force laws used were the charge force, a force that scales as $\propto r^{-1}$, a force that scales as $\propto r^{-2}$ and the spring force. The input data contained particle features including the positions (x, y), discretised velocities $(\dot{x}, \dot{y})$, mass (m) and charge (q). The target data were the instantaneous accelerations.

GNNs are well-suited to modelling n-body systems due to their inductive biases. They naturally represent particles as nodes and interactions as edges, making them permutation-invariant to particle ordering. Their architecture reflects physical principles: the edge model computes messages between pairs of particles based on their features — analogous to pairwise forces — using the features of the two connected nodes as input. These messages are summed for each receiving node and passed, along with that node's own features, to the node model, which computes an updated state, which in our case are the accelerations [2]. Both the edge model and the node model are Multilayer Perceptrons (MLPs). This structure reflects Newtonian mechanics: interactions are computed pairwise and then summed to update dynamics.

Each GNN was trained for 1.1 million optimisation steps using the Adam optimiser [3] and a Cosine Annealing learning rate schedule.

Model Variations

If the system is described by a n-dimensional force law, then in a trained GNN, the edge messages (outputs of the edge model) should be linear combinations of the true force vectors as long as the message dimensionality matches that of the system. Hence, we want to encourage sparse representations in the edge model. We train different variations of a GNN:

- *Standard*: no regularisation. The dimensionality of the edge messages, $\mathcal{L}^e$, are not constrained ($\mathcal{L}^e = 100$).
- *Bottleneck*: The dimensionality of the edge messages are constrained to dimensionality of the system ($\mathcal{L}^e = 2$).
- *L1*: Added L1 regularisation of the edge messages to the loss function.

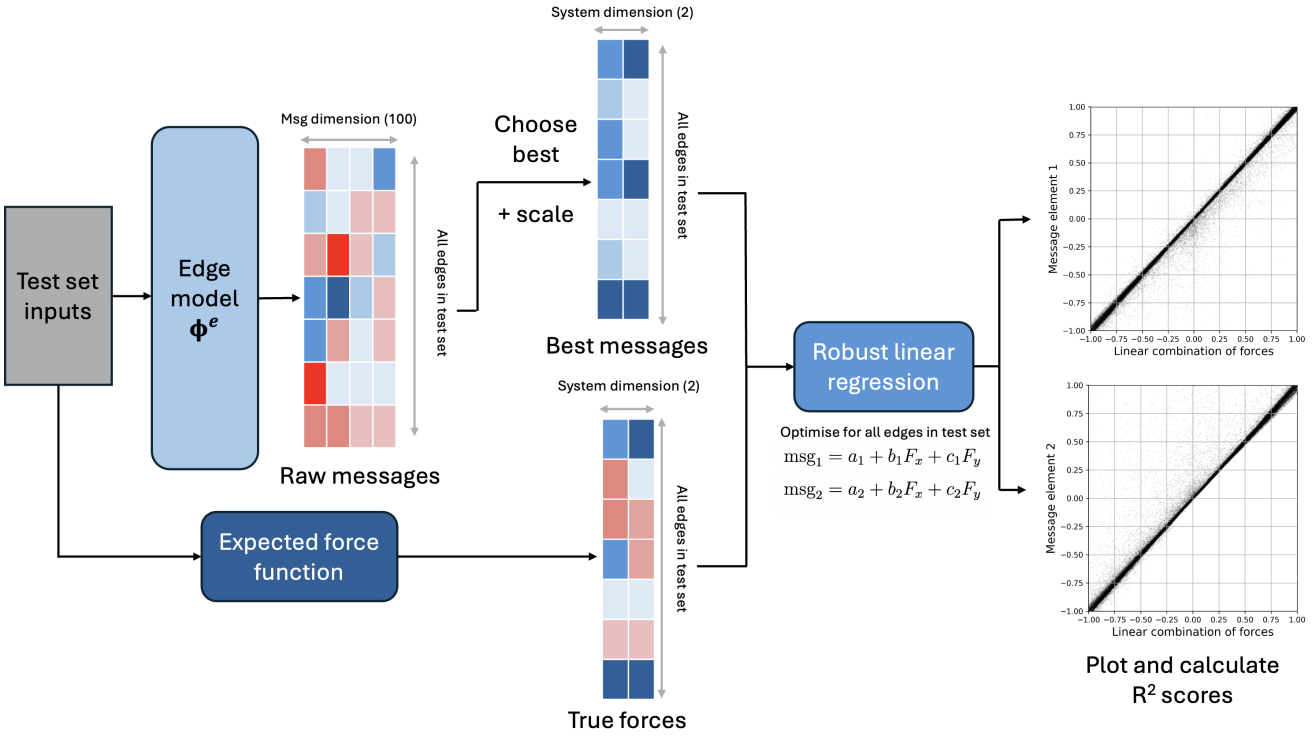

Figure 1: We evaluate the model on the test set and extract the edge messages. We then choose the most two important messages as the ones with the highest standard deviation (for standard and L1 models) or highest KL-divergence (KL model). The pruning and bottleneck models are already constrained to the dimensionality of the system. We perform a robust linear regression, where we ignore the 10% outliers, to fit the true force vectors to the message elements. A R^2 score that is near 1 indicates that the model has successfully learnt the true forces.

- *Kullbeck-Leibler*: Variational model. Edge model outputs both the mean (μ_e) and log-variance ($\log(\sigma^2)$) of the messages. We add regularisation term to the loss equivalent to Kullbeck-Leibler divergence between $\mathcal{N}(\mu_e, \sigma^2)$ and a standard normal.

- *Pruning*: The dimensionality of the edge messages is gradually reduced during training until it matches that of the system, with unimportant elements zero-masked. Through hyperparameter tuning, we found that using a Cosine Annealing pruning schedule, with pruning completed at 65% of training, yielded the best performance. This model variation is an extension to the original paper.

ANALYSIS & KEY FINDINGS

Validating the Edge Model Learns True Forces

Figure 1 shows how we validated whether the GNN has broadly learnt the true force. Table 1 show the R^2 values of the linear fits of the true forces to the most important message elements. The L1, bottleneck and pruning models performed the best for all datasets demonstrating the greatest success in capturing the underlying interaction forces.

Reconstructing Force Equations

We used the *PySR* package [4] to perform symbolic regression on the latent representations of the trained GNNs, aiming to reconstruct the true force laws learned by

Simulation	Standard	Bottleneck	L1	KL	Pruning
Charge	0.466	0.995	0.692	0.029	0.998
r^{-1}	0.414	0.839	0.844	0.594	0.838
r^{-2}	0.833	0.967	0.830	0.866	0.973
Spring	0.920	1.000	0.998	0.801	0.952

Table 1: Averaged R^2 values of the linear fits of the forces to the most important messages for the different simulations and model types. The 10% outliers were not included in the linear fit or R^2 calculation.

the model. *PySR* uses a genetic algorithm to search for the best equations - balancing accuracy and complexity - to describe a dataset (target variable) given input variables and operators [5]. The target variables of the symbolic regression were the two most important edge messages, as extracted previously (see Figure 1). The input variables were the masses and charges of the two interacting particles, as well as their relative positions: $\Delta x = x_1 - x_2$, $\Delta y = y_1 - y_2$, and $r = \sqrt{\Delta x^2 + \Delta y^2 + \epsilon}$, where ϵ is a small constant added for numerical stability. *PySR* produces a Pareto front of candidate equations across varying complexity levels, from which we identify successful reconstructions of the true force laws.

The operators that were allowed in the symbolic regression were $+, -, \times, \texttt{inv}(\cdot)$, `exp`, `log` and `^`. We chose a random set of 5,000 examples from the test set for the symbolic regression. For all of the tests, we ran the symbolic regression for enough iterations for the Pareto front to remain stable (6000 iterations for all datasets except charge which we ran for 7000 iterations).

Table 2 summarises the success of symbolic regression in recovering the underlying force laws across various systems and model variations. The bottleneck and pruning models performed the best, suggesting that directly constraining the dimensionality of the edge messages — rather than sparsity-inducing regularisation — is the most effective strategy for learning meaningful representations.

Simulation	Message	Standard	Bottleneck	L1	KL	Pruning
Charge	1	✗	✓	✓	✗	✓
	2	✗	✓	✗	✗	✓
r^{-1}	1	✓	✓	✓	✗	✓
	2	✓	✓	✓	✗	✓
r^{-2}	1	✗	✓	✗	✓	✓
	2	✗	✓	✗	✗	✓
Spring	1	✓	✓	✓	✗	✓
	2	✓	✓	✓	✗	✓

Table 2: Symbolic regression results for each message component. ✓= correct form of force law recovered; ✗= failure.

DISCUSSION & NEXT STEPS

A limitation of this framework is that the edge model is not guaranteed to learn the true force equations directly. Since the GNN is trained to predict accelerations, the edge model may learn the forces scaled by an arbitrary function of the receiving node's features, provided the node model then divides by the same function.

The pruning model variation is an extension to the original work. This approach performed comparably to the bottleneck model — as expected, given that both impose constraints on message dimensionality — but does not require the target dimensionality to be specified beforehand.

We have shown that this framework is successful in reconstructing known force laws from simulated datasets. In the original paper, the authors applied this approach to cosmological data and found a new analytical equation to describe dark matter concentrations. The next steps would be to apply this framework to more complex, real-world physical systems enabling the discovery of novel empirical laws.

REFERENCES

[1] Miles D. Cranmer, Alvaro Sanchez-Gonzalez, Peter W. Battaglia, Rui Xu, Kyle Cranmer, David N. Spergel, and Shirley Ho. "Discovering Symbolic Models from Deep Learning with Inductive Biases". In: *CoRR* abs/2006.11287 (2020). arXiv: 2006.11287. URL: https://arxiv.org/abs/2006.11287.

[2] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. *Relational inductive biases, deep learning, and graph networks*. 2018. arXiv: 1806.01261 [cs.LG]. URL: https://arxiv.org/abs/1806.01261.

[3] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980.

[4] *PySR Code Repository*. URL: https://github.com/MilesCranmer/PySR/blob/master/examples/pysr_demo.ipynb.

[5] Miles Cranmer. *Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl*. arXiv:2305.01582 [astro-ph, physics:physics]. May 2023. DOI: 10.48550/arXiv.2305.01582. URL: http://arxiv.org/abs/2305.01582 (visited on 07/17/2023).