Nonparametric Teaching for Graph Property Learners

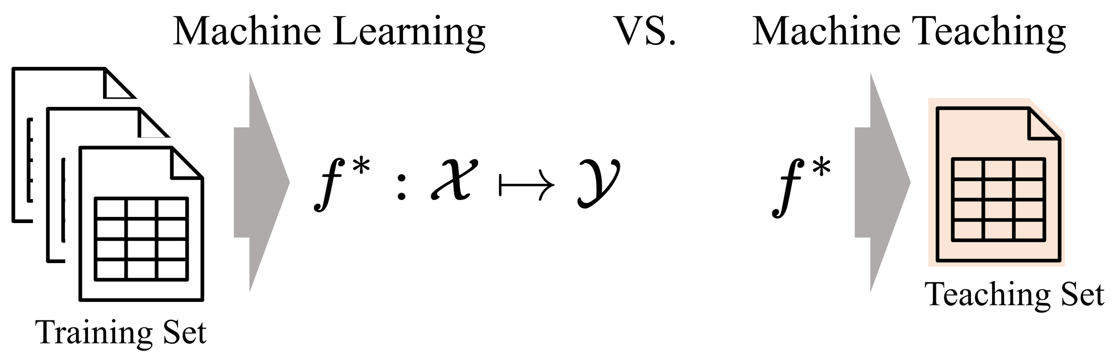
Chen Zhang[1]*, Weixin Bu[2]*, Zeyi Ren[1], Zhengwu Liu[1], Yik-Chung Wu[1], Ngai Wong[1]

[1]The University of Hong Kong [2]Reversible Inc

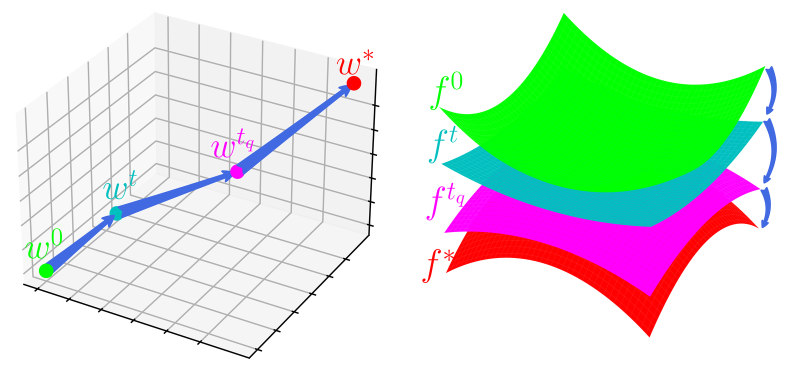
Nonparametric Teaching

Nonparametric teaching (NT) (Zhang et al., 2023b;a; 2024a) presents a theoretical framework to facilitate efficient example selection when the target function is nonparametric, i.e., implicitly defined.

It builds on the idea of *machine teaching* (Zhu, 2015; Zhu et al., 2018), which involves designing a training set (dubbed the teaching set) to help the learner rapidly converge to the target functions.



NT (Zhang et al., 2023b;a; 2024a) relaxes the assumption of target functions† f being parametric (Liu et al., 2017; 2018), which is f can be represented by a set of parameters $\boldsymbol{w}$, *e.g.*, $f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle$ with input $\boldsymbol{x}$, to encompass the teaching of nonparametric target functions.



(a) Parametric IMT (b) Nonparametric IMT

†The loss $\mathcal{L}$ can be general for different tasks, *e.g.*, square loss for regression and hinge loss for classification.

Graph Property Learning

Graph-structured data, commonly referred to as graphs, are typically represented by vertices and edges (Hamilton et al., 2017; Chami et al., 2022). The vertices, or nodes, contain individual features, while the edges link these nodes and capture the structural information, collectively forming a complete graph.

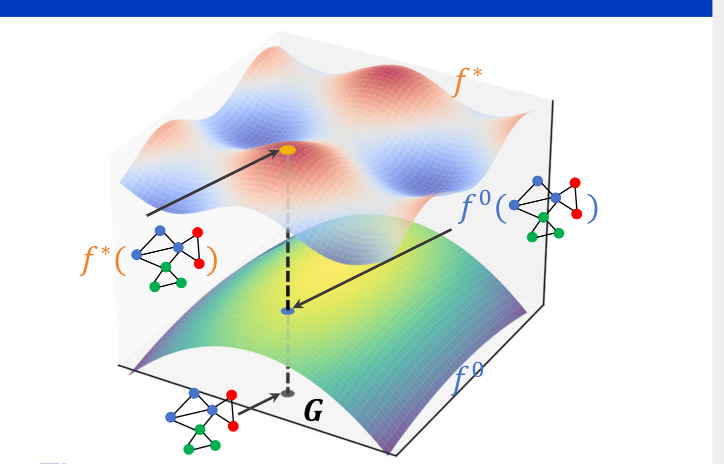


Figure: The implicit mapping f^* between a graph G and its property $f^*(G)$.

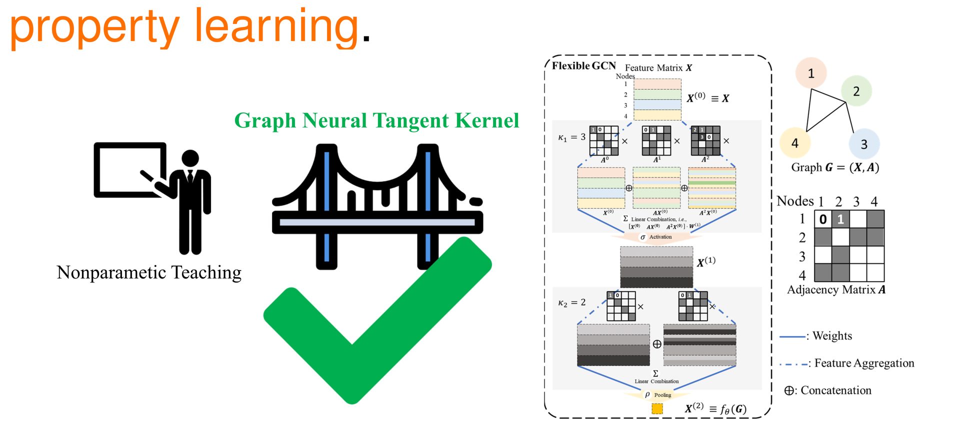
Graph properties can be categorized as either node-level or graph-level. For example, the node category is a node-level property in social network graphs (Fan et al., 2019), while the solubility of molecules is a graph-level property in molecular graphs (Ramakrishnan et al., 2014). Inferring these graph properties essentially involves learning the implicit mapping from graphs to these properties (Hamilton et al., 2017).

Content by: Chen Zhang.

The Bridge Between NT and Graph Property Learning: Graph Neural Tangent Kernel

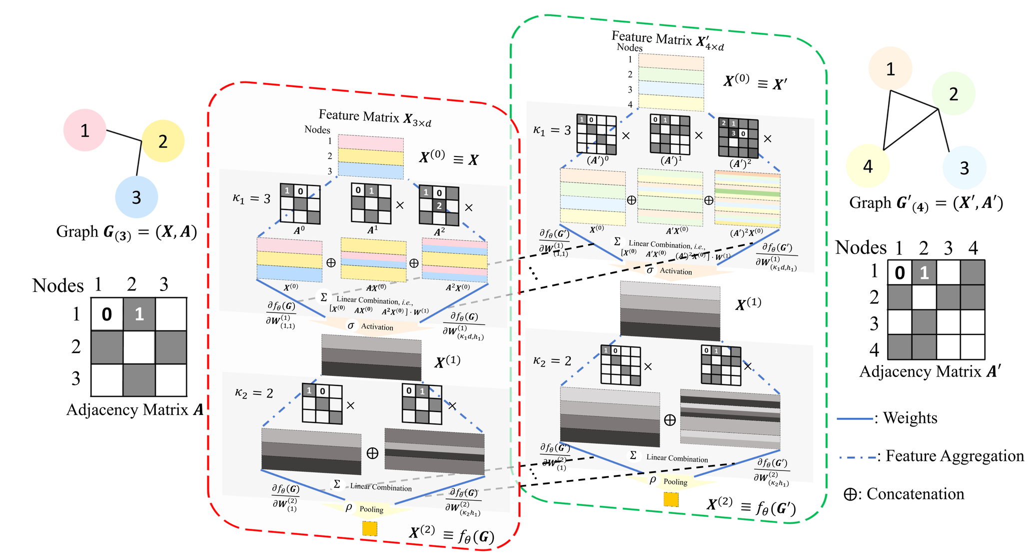
The evolution of a Graph Convolutional Network (GCN) is typically achieved by gradient descent on its parameters, whereas nonparametric teaching involves functional gradient descent as the means of function evolution.

Bridging this (**theoretical** + **practical**) gap is of great value and calls for more examination prior to the application of nonparametric teaching algorithms in the context of graph property learning.



Graph Neural Tangent Kernel (Jacot et al., 2018; Du et al., 2019; Krishnagopal & Ruiz, 2023) is a symmetric and positive definite kernel function, which is derived from the analysis of the evolution of a GCN.

$$K_{\theta^t}(\boldsymbol{G}_i, \cdot) := \left\langle \frac{\partial f_{\theta^t}(\boldsymbol{G}_i)}{\partial \theta^t}, \frac{\partial f_{\theta^t}(\cdot)}{\partial \theta^t} \right\rangle$$



Main Contribution

Our key contributions are:

- We propose **Gra**ph **N**eural **T**eaching (GraNT) that interprets graph property learning within the theoretical context of nonparametric teaching. This enables the use of greedy algorithms from the latter to effectively enhance the learning efficiency of the graph property learner, GCN.
- We unveil a strong link between the evolution of a GCN using gradient descent on its parameters and that of a function using functional gradient descent in nonparametric teaching. These connect nonparametric teaching theory to graph property learning, thus expanding the applicability of nonparametric teaching in the context of graph property learning.
- We demonstrate the effectiveness of GraNT through extensive experiments in graph property learning. Specifically, GraNT saves training time for graph-level regression (-36.62%), graph-level classification (-38.19%), node-level regression (-30.97%) and node-level classification (-47.30%), while upkeeping its generalization performance.

GraNT Algorithm

Algorithm 1 GraNT Algorithm

Input: Target mapping f^* realized by a dense set of graph-property pairs, initial GCN f_{θ^0}, the size of selected training set $m \leq N$, small constant $\epsilon > 0$ and maximal iteration number T.

Set $f_{\theta^t} \leftarrow f_{\theta^0}, t = 0$.

while $t \leq T$ and $\|[f_{\theta^t}(\boldsymbol{G}_i) - f^*(\boldsymbol{G}_i)]_N\|_2 \geq \epsilon$ **do**

  The teacher selects m teaching graphs:

  `/* (Graph-level) Graphs corresponding to the m largest` $|f_{\theta^t}(\boldsymbol{G}_i) - f^*(\boldsymbol{G}_i)|$. `*/`

  $$\{\boldsymbol{G}_i\}_m{}^* = \underset{\{\boldsymbol{G}_i\}_m \subseteq \{\boldsymbol{G}_i\}_N}{\arg\max} \|[f_{\theta^t}(\boldsymbol{G}_i) - f^*(\boldsymbol{G}_i)]_m\|_2.$$

  `/* (Node-level) Graphs associated with the m largest` $\frac{\|f_{\theta^t}(\boldsymbol{G}_i) - f^*(\boldsymbol{G}_i)\|_2}{n_i}$. `*/`

  $$\{\boldsymbol{G}_i\}_m{}^* = \underset{\{\boldsymbol{G}_i\}_m \subseteq \{\boldsymbol{G}_i\}_N}{\arg\max} \left\| \left[\frac{f_{\theta^t}(\boldsymbol{G}_i) - f^*(\boldsymbol{G}_i)}{n_i} \right]_m \right\|_{\mathcal{F}},$$

  with Frobenius norm $\|\cdot\|_{\mathcal{F}}$.

  Provide $\{\boldsymbol{G}_i\}_m{}^*$ to the GCN learner.

  The learner updates f_{θ^t} based on received $\{\boldsymbol{G}_i\}_m{}^*$:

  `// Parameter-based gradient descent.`

  $$\theta^t \leftarrow \theta^t - \frac{\eta}{m} \sum_{\boldsymbol{G}_i \in \{\boldsymbol{G}_i\}_m{}^*} \nabla_\theta \mathcal{L}(f_{\theta^t}(\boldsymbol{G}_i), f^*(\boldsymbol{G}_i)).$$

  Set $t \leftarrow t + 1$.

end

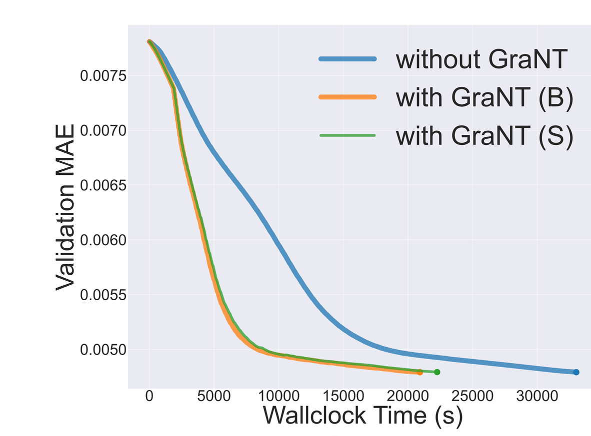
By comparing the disparity between the property true values and the GCN outputs, the nonparametric teacher selectively chooses examples (graphs) of the greatest disparity, instead of using all, to feed to the GCN learner who undergoes learning (*i.e.*, training).

Experiments and Results

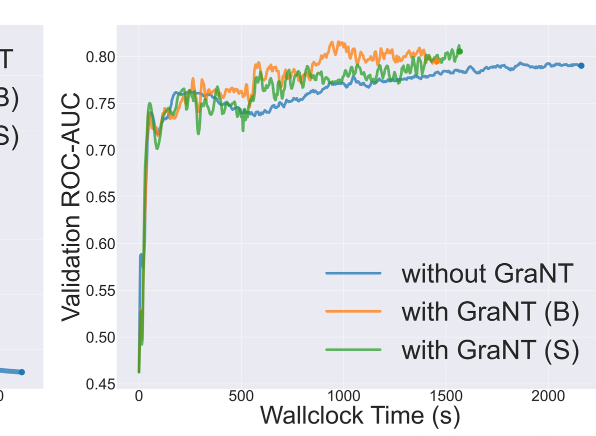
We conduct extensive experiments to validate the effectiveness of GraNT.

GraNT		Dataset	Time (s)	Loss ↓	MAE ↓	ROC-AUC ↑	AP ↑
✗		QM9	9654.81	2.0444	0.0051±0.0009	-	-
		ZINC	33033.82	3.1160	0.0048±0.0004	-	-
		ogbg-molhiv	2163.50	0.1266	-	0.7572±0.0005	-
		ogbg-molpcba	130191.26	0.0577	-	-	0.3270±0.0000
		gen-reg	3344.78	0.0086	0.0007±0.0001	-	-
		gen-cls	11662.25	0.1314	-	0.9150±0.0024	-
✓	B	QM9	6392.26 (-33.79%)	2.0436	0.0051±0.0009	-	-
		ZINC	20935.24 (-36.62%)	3.1165	0.0048±0.0004	-	-
		ogbg-molhiv	1457.39 (-32.64%)	0.1238	-	0.7676±0.0036	-
		ogbg-molpcba	80465.06 (-38.19%)	0.0577	-	-	**0.3358±0.0001**
		gen-reg	2308.97 (-30.97%)	0.0086	0.0007±0.0001	-	-
		gen-cls	6145.72 (-47.30%)	0.1314	-	**0.9157±0.0013**	-
	S	QM9	7076.37 (-26.71%)	2.0443	0.0051±0.0009	-	-
		ZINC	22265.83 (-32.60%)	3.1170	0.0048±0.0004	-	-
		ogbg-molhiv	1597.69 (-26.15%)	0.1421	-	**0.7705±0.0027**	-
		ogbg-molpcba	89858.65 (-30.98%)	0.0575	-	-	0.3351±0.0025
		gen-reg	2337.46 (-30.12%)	0.0086	0.0007±0.0001	-	-
		gen-cls	8171.21 (-29.93%)	0.1313	-	**0.9157±0.0014**	-

Table 1: Training time and testing results across different benchmarks. GraNT (B) and GraNT (S) demonstrate similar testing performance while significantly reducing training time compared to the "without GraNT", across graph-level (QM9, ZINC, ogbg-molhiv, ogbg-molpcba) and node-level (gen-reg, gen-cls) datasets, for both regression and classification tasks.



(a) ZINC MAE.



(b) ogbg-molhiv ROC-AUC.

Figure: Validation set performance for graph-level tasks: ZINC (regression) and ogbg-molhiv (classification).

Project Page