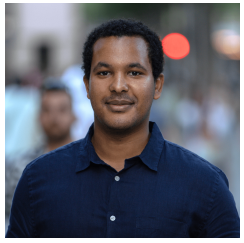
Graph Learning for and with Graph Neural Networks

Johannes Lutzeyer

Data Science and Mining Team, Laboratoire d'Informatique (LIX),
École Polytechnique, Institut Polytechnique de Paris

June 06, 2024

Today I present work that was done in collaboration with



Yassine Abbahaddou
PhD Student LIX

Sofiane Ennadir
PhD Student KTH

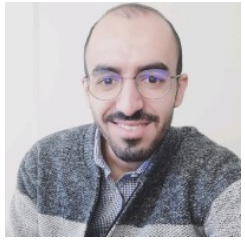
Gaspard Michel
PhD Student Deezer

Ariel Ramos Vela
Data Scientist Amadeus

Michalis Chatzianastasis
PhD Student LIX

Dr. Changmin Wu
Researcher Huawei

Dr. George Dasoulas
Postdoc Harvard



Dr. Mohamed Seddik
Senior Researcher TII

Dr. Guillaume Salha Galvan
Research Coordinator Deezer

Dr. Giannis Nikolentzos
Assistant Professor
University of Peloponnese

Prof. Henrik Boström
Professor KTH

Dr. Anastasios Giovanidis
Data Science Researcher
Ericsson

Dr. Romain Hennequin
Lead Research Scientist
Deezer

Prof. Michalis Vazirgiannis
Distinguished Professor LIX

Overview of Today's Talk

1) Brief Introduction to Graph Neural Networks (GNNs);

2) GNNs where graph structure is altered/learned **previous** to training;

3) GNNs where graph structure is altered/learned **during** training;

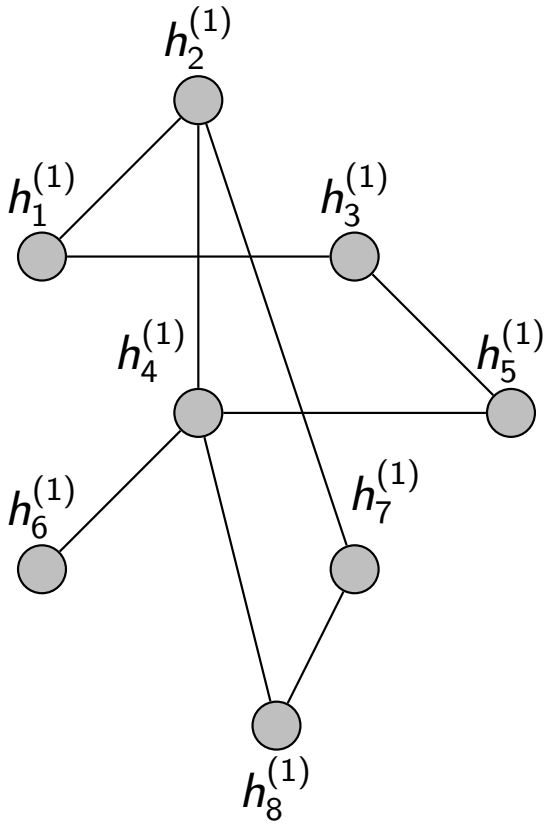
4) Robustness of GNNs.

Graph Neural Networks

Graph Neural Networks (GNNs) are neural networks that take graph-structured data as input.

In this talk we will only see a specific type of GNN, the Message Passing Neural Networks.

$$m_v^{(k)} = M^{(k)}\left(\left\{h_w^{(k-1)} : w \in \mathcal{N}(v)\right\}\right),$$
$$h_v^{(k)} = U^{(k)}\left(h_v^{(k-1)}, m_v^{(k)}\right).$$



E.g., the Graph Convolutional Network (GCN, Kipf and Welling, 2017)

$$H^{(1)} = \mathrm{ReLU}\left(\tilde{A}XW^{(1)}\right).$$

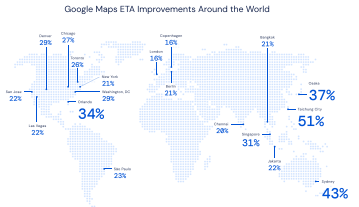
Iteratively performing the message-passing and update computations allows us to build 'deep' learning models, e.g., a 3-layer GCN

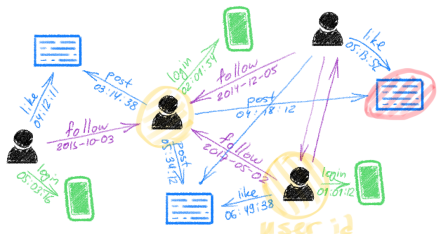
$$\hat{y} = \sigma\left(\tilde{A}\,\mathrm{ReLU}\left(\tilde{A}\,\mathrm{ReLU}\left(\tilde{A}XW^{(1)}\right)W^{(2)}\right)W^{(3)}\right).$$

Academic and Industrial Success of GNNs

Empirical and Theoretical **Research**:

- expressivity analysis of GNNs (Xu et al., 2019; Geerts and Reutter, 2022);
- bottlenecks, e.g., oversmoothing and oversquashing (Alon and Yahav, 2020; Deac et al., 2022)
- robustness to adversarial attacks and noise (Günnemann, 2022; Zhou et al., 2020).





Successful **Applications** of GNNs:

- Google Maps (Lange and Perez, 2020);
- Twitter (Bronstein, 2020);
- Amazon, Alibaba, Pinterest & Uber Eats (Virinchi et al., 2022; Wang et al., 2018; Ying et al., 2018; Jain et al., 2019);
- Discovery of two *new antibiotics* (Stokes et al., 2020; Liu et al., 2023);
- LinkedIn (Borisyuk et al., 2024).

Overview

We shall now categorise Graph Neural Networks (GNNs) into GNNs that

1) alter or learn the graph structure **previous** to training;
2) alter or learn the graph structure **during** training.

GNNs that do not modify the graph structure at all, such as the GCN (Kipf and Welling, 2017) and GIN (Xu et al., 2019), are omitted in this talk.

Graph is Altered/Learned Before Training

Reweight existing edges:

- GCN-k (Seddik et al., 2022, AISTATS)
  Reweight edges based on node feature kernels

Add/Delete individual edges:

- SDRF (Topping, Di Giovanni et al., 2022)
  Rewiring according to curvature metrics on graphs
- Modularity-Aware (V)GAE (Salha-Galvan et al., 2022, Neural Networks)
  Add edges based on Louvain Clustering

Completely redraw the graph:

- PPRGo (Bojchevski et al., 2020)
  Rewiring according to thresholded Personalised PageRank Scores
- CorePPR (Ramos Vela et al., 2022, NeurIPS Workshop)
  Rewiring according to thresholded Personalised PageRank and CoreRank Scores

Represent the graph via substructures:

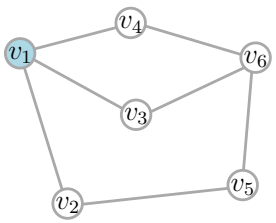
- PathNNs (Michel et al., 2023, ICML)
  We represent graphs as collections of paths

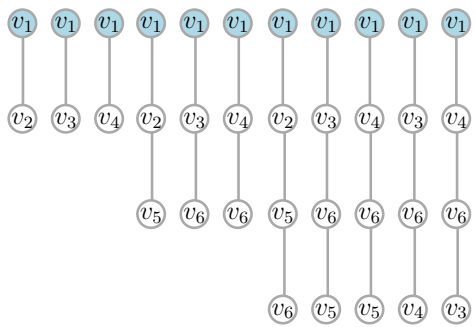
Paths Collections

Idea

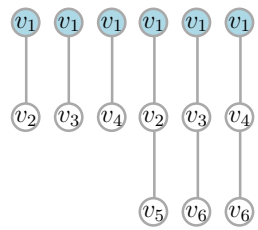
Explicitly representing paths (and nodes) instead of only nodes in GNNs should lead to more accurate and expressive models.

We make use of three different collections of paths:

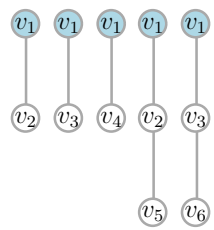




all paths ($\mathcal{AP}$)



all shortest paths ($\mathcal{SP}^+$)

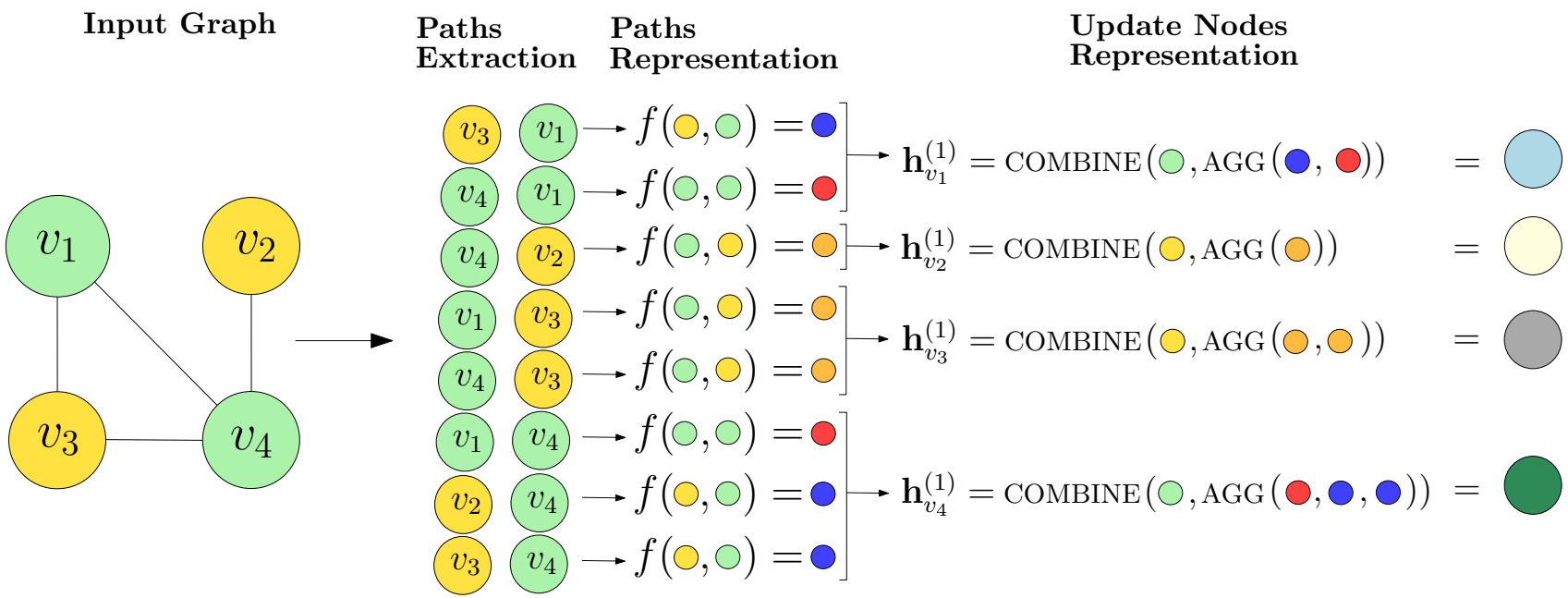


single shortest paths ($\mathcal{SP}$)

Path Neural Networks

1) Given a collection of paths in a graph, we apply an LSTM to learn path representations.
2) We then aggregate the representations of all paths emanating from a node to form updated node representations.

At layer ℓ of the model we process paths of length ℓ, e.g., $\ell = 2$.



Results on Real-World Datasets

- PathNNs demonstrate convincing performance on **Synthetic Datasets** designed to benchmark the expressivity of GNNs.
- **6 Real-World Graph Classification Datasets**

	DD	PROTEINS	NCI1	ENZYMES	IMDB-B	IMDB-M
GIN	75.3 ± 2.9	73.3 ± 4.0	80.0 ± 1.4	59.6 ± 4.5	71.2 ± 3.9	48.5 ± 3.3
GraphSAGE	72.9 ± 2.0	73.0 ± 4.5	76.0 ± 1.8	58.2 ± 6.0	68.8 ± 4.5	47.6 ± 3.5
GAT	73.9 ± 3.4	70.9 ± 2.7	77.3 ± 2.5	49.5 ± 8.9	69.2 ± 4.8	48.2 ± 4.9
SPN ($K = 1$)	72.7 ± 2.6	71.0 ± 3.7	80.0 ± 1.5	67.5 ± 5.5	NA	NA
SPN ($K = 5$)	77.4 ± 3.8	74.2 ± 2.7	78.6 ± 3.8	69.4 ± 6.2	NA	NA
PathNet ($N = 10, K = 2$)	OOM	70.5 ± 3.9	64.1 ± 2.3	69.3 ± 5.4	70.4 ± 3.8	49.1 ± 3.6
Nested GNN	**77.8** ± 3.9	74.2 ± 3.7	NA	31.2 ± 6.7	NA	NA
PathNN-$\mathcal{P}$ ($K = 1$)	76.9 ± 3.7	**75.2** ± 3.9	77.5 ± 1.6	**73.0** ± 5.2	**72.6** ± 3.3	**50.8** ± 4.5
PathNN-$\mathcal{SP}$ ($K = 2$)	75.3 ± 2.7	73.1 ± 3.1	82.0 ± 1.6	71.6 ± 6.4	70.8 ± 3.5	50.0 ± 4.1
PathNN-$\mathcal{SP}$ ($K = 3$)	77.0 ± 3.1	72.2 ± 2.7	82.2 ± 1.7	69.2 ± 4.7	-	-
PathNN-$\mathcal{SP}^+$ ($K = 2$)	74.7 ± 3.0	73.1 ± 3.7	81.0 ± 1.4	72.5 ± 5.3	70.5 ± 3.4	50.7 ± 4.5
PathNN-$\mathcal{SP}^+$ ($K = 3$)	76.5 ± 4.6	73.2 ± 3.3	**82.3** ± 1.9	70.4 ± 3.1	-	-
PathNN-$\mathcal{AP}$ ($K = 2$)	75.0 ± 4.4	73.1 ± 4.9	81.3 ± 1.8	71.8 ± 4.8	71.7 ± 3.6	49.8 ± 4.2
PathNN-$\mathcal{AP}$ ($K = 3$)	OOM	73.1 ± 4.0	82.3 ± 1.7	69.0 ± 5.3	OOM	OOM

- Our code is available: https://github.com/gasmichel/PathNNs_expressive
- Graph Learning on the basis of substructures is expensive but accurate.

Graph is Altered/Learned During Training

Reweight existing edges:

- GAT (Veličković et al., 2018)
  Message-Passing Operation: $A_{att}X$,
- GATv2 (Brody et al., 2022)
  Message-Passing Operation: $A_{attv2}X$,
- PGSO-GNN (Dasoulas et al., 2021, ICLR)

  Learn a graph-level graph representation via a particular parameterisation of the message passing operator

Completely redraw the graph:

- Graph Transformers (Dwivedi and Bresson, 2021)
  Freely learn the message passing operator using the Transformer attention mechanism

Represent neighbourhoods as sequences:

- GOAT (Chatzianastasis et al., 2023, AAAI)
  1) A self-attention mechanism is used to obtain a ranking of nodes in neighbourhoods.
  2) An LSTM processed the ordered neighbourhoods to produce updated node representation.

Learning Parametrised Graph Shift Operators (2021, ICLR)

Recall, that MPNNs iterate between message passing and update steps.

Observation

In MPNNs, the choice of message passing function corresponds to a choice of matrix used to represent the graph, e.g., $D_1^{-1/2} A_1 D_1^{-1/2}$ in the GCN

$$H^{(l+1)} = \sigma\left(D_1^{-1/2} A_1 D_1^{-1/2} H^{(l)} W^{(l)}\right).$$

Definition

We define the parametrised graph shift operator (PGSO), denoted by $\gamma(A, \mathcal{S})$, as

$$\gamma(A, \mathcal{S}) = m_1 D_a^{e_1} + m_2 D_a^{e_2} A_a D_a^{e_3} + m_3 I_n, \tag{1}$$

where $A_a = A + aI_n$, $D_a = \mathrm{Diag}(A_a \mathbf{1}_n)$ and $\mathcal{S} = (m_1, m_2, m_3, e_1, e_2, e_3, a)$.

The GCN-PGSO is defined as $H^{(l+1)} = \sigma\left(\gamma(A, \mathcal{S}) H^{(l)} W^{(l)}\right)$, where the seven scalablar parameters in $\mathcal{S}$ are trainable.

Hence, we learn different edge weights as functions of node degrees.

Bounding the Expected Robustness of Graph Neural Networks Subject to Node Feature Attacks

Abbahaddou*, Ennadir*, Lutzeyer, Vazirgiannis & Boström (2024, ICLR)

(Graph) Adversarial Attacks

Goal: Adversarial attacks apply a *small* change to the input to achieve a *large* change in the output of our model.

To quantify the robustness of a function processing graph structured data, i.e., $f : (\mathcal{G}, \mathcal{X}) \to \mathcal{Y}$ we need:

- a distance on the input space
  $d_2^{\alpha,\beta}([G, X], [\tilde{G}, \tilde{X}]) = \min_{P \in \Pi} \left(\alpha \|A - P\tilde{A}P^T\|_2 + \beta \|X - P\tilde{X}\|_2 \right),$
- and a distance on the output space
  $d_1(f(\tilde{G}, \tilde{X}), f(G, X)) = \|f(\tilde{G}, \tilde{X}) - f(G, X)\|_1.$

Expected Adversarial Robustness

Let the *expected vulnerability* of a graph function f be defined as
$Adv_\epsilon^{\alpha,\beta}[f] = \mathbb{P}_{(G,X)\sim\mathcal{D}_{\mathcal{G},\mathcal{X}}}[(\tilde{G}, \tilde{X}) \in B^{\alpha,\beta}(G, X, \epsilon) : d_{\mathcal{Y}}(f(\tilde{G}, \tilde{X}), f(G, X)) > \sigma]$,
with $B^{\alpha,\beta}(G, X, \epsilon) = \{(\tilde{G}, \tilde{X}) : d^{\alpha,\beta}([G, X], [\tilde{G}, \tilde{X}]) < \epsilon\}$ for any budget $\epsilon \geq 0$.

Then, a graph function $f : (\mathcal{G}, \mathcal{X}) \to \mathcal{Y}$ is $((d^{\alpha,\beta}, \epsilon), (d_{\mathcal{Y}}, \gamma))$–*robust* if its vulnerability $Adv_\epsilon^{\alpha,\beta}[f]$ can be upper-bounded by γ, i.e., $Adv_\epsilon^{\alpha,\beta}[f] \leq \gamma$.

Problem Set-Up & Theoretical Results

Recall, Graph Neural Networks (GNNs) take both a graph A and node features X as input.

Problem: Most defense approaches for GNNs defend structural attacks altering A. There exists very little work on how to defend against attacks on the node features X.

Upper Bound on GCN Vulnerability

We consider node-feature attacks on the input graph (A, X), with a budget ϵ and L-layer GCNs with weight matrices $W^{(i)}$ for $i \in \{1, \ldots, L\}$.
Then, the vulnerability of GCNs is upper bounded by

$$\gamma = \prod_{i=1}^{L} \|W^{(i)}\|_1 \prod_{i=1}^{L} \|W^{(i)}\|_1 \frac{\epsilon \sum_{u \in \mathcal{V}} \hat{w}_u}{\sigma},$$

with $\hat{w}_u$ denoting the sum of normalized walks of length $(L-1)$ starting from node u.

Insight: Our upper bound on the vulnerability of a GCN is **smaller for small** $\prod_{i=1}^{L} \|W^{(i)}\|_1$ yielding a **more robust GCN.**

Methodology

Fact: Orthonormal matrices have norm 1.

$\Rightarrow$ According to our bound a GNN with orthonormal weight matrices should be more robust.

Björk Orthonormalisation Algorithm

Given a weight matrix W we iteratively alter it to approximate the closest orthonormal matrix $\hat{W}$. When $\hat{W}_0 = W$, we recursively compute

$$\hat{W}_{k+1} = \hat{W}_k \left(I + \tfrac{1}{2} \left(I - \hat{W}_k^T \hat{W}_k \right) + \ldots + (-1)^p \binom{-1/2}{p} \left(I - \hat{W}_k^T \hat{W}_k \right)^p \right).$$

Proposed Solution: In our *GCORN* model we propose the inclusion of several Björk Orthonormalisation iterations in each forward pass during the training of a GCN, **yielding weight matrices that approach orthonormality and thereby a more robust GNN**.

Results

Table: Node classification accuracy ($\pm$ standard deviation) for feature-based attacks.

Attack	Dataset	GCN	GCN-k	AirGNN	RGCN	ParsevalR	GCORN
Random ($\psi = 0.5$)	Cora	68.4 $\pm$ 1.9	69.2 $\pm$ 2.6	73.5 $\pm$ 1.9	71.6 $\pm$ 0.3	72.9 $\pm$ 0.9	**77.1 $\pm$ 1.8**
	CiteSeer	57.8 $\pm$ 1.5	62.3 $\pm$ 1.2	64.6 $\pm$ 1.6	63.7 $\pm$ 0.6	65.1 $\pm$ 0.8	**67.8 $\pm$ 1.4**
	PubMed	68.3 $\pm$ 1.2	71.2 $\pm$ 1.1	70.9 $\pm$ 1.3	71.4 $\pm$ 0.5	71.8 $\pm$ 0.8	**73.1 $\pm$ 1.1**
	CS	85.3 $\pm$ 1.1	86.7 $\pm$ 1.1	87.5 $\pm$ 1.6	88.2 $\pm$ 0.9	87.6 $\pm$ 0.6	**89.8 $\pm$ 1.2**
	OGBN-Arxiv	68.2 $\pm$ 1.5	52.8 $\pm$ 0.5	66.5 $\pm$ 1.3	63.8 $\pm$ 1.9	68.3 $\pm$ 1.9	**69.1 $\pm$ 1.8**
Random ($\psi = 1.0$)	Cora	41.7 $\pm$ 2.1	46.3 $\pm$ 2.8	53.7 $\pm$ 2.2	52.8 $\pm$ 1.6	55.3 $\pm$ 1.2	**57.6 $\pm$ 1.9**
	CiteSeer	38.2 $\pm$ 1.3	45.3 $\pm$ 1.4	49.8 $\pm$ 2.1	43.7 $\pm$ 2.2	51.2 $\pm$ 1.2	**57.3 $\pm$ 1.7**
	PubMed	60.1 $\pm$ 1.7	62.3 $\pm$ 1.3	62.4 $\pm$ 1.2	61.9 $\pm$ 1.2	61.3 $\pm$ 1.7	**65.8 $\pm$ 1.4**
	CS	69.9 $\pm$ 1.3	73.2 $\pm$ 0.9	76.7 $\pm$ 2.8	76.2 $\pm$ 1.4	78.7 $\pm$ 1.2	**81.3 $\pm$ 1.6**
	OGBN-Arxiv	66.4 $\pm$ 1.9	46.6 $\pm$ 0.6	62.7 $\pm$ 1.6	63.0 $\pm$ 2.4	66.1 $\pm$ 0.7	**67.3 $\pm$ 2.1**
PGD	Cora	54.1 $\pm$ 2.4	58.3 $\pm$ 1.6	68.2 $\pm$ 1.8	62.5 $\pm$ 1.2	68.6 $\pm$ 1.7	**71.1 $\pm$ 1.4**
	CiteSeer	52.3 $\pm$ 1.1	59.6 $\pm$ 1.6	59.3 $\pm$ 2.1	61.9 $\pm$ 1.1	62.1 $\pm$ 1.5	**65.6 $\pm$ 1.4**
	PubMed	66.1 $\pm$ 2.1	67.3 $\pm$ 1.3	70.8 $\pm$ 1.7	69.5 $\pm$ 0.9	68.9 $\pm$ 2.1	**72.3 $\pm$ 1.3**
	CS	71.3 $\pm$ 1.1	74.1 $\pm$ 0.8	76.3 $\pm$ 2.1	76.6 $\pm$ 1.2	77.3 $\pm$ 0.6	**79.6 $\pm$ 1.2**
	OGBN-Arxiv	67.5 $\pm$ 0.9	49.9 $\pm$ 0.7	55.7 $\pm$ 0.9	63.6 $\pm$ 0.7	67.6 $\pm$ 1.2	**68.1 $\pm$ 1.1**
Nettack	Cora	60.9 $\pm$ 2.5	64.2 $\pm$ 5.2	66.7 $\pm$ 3.8	63.4 $\pm$ 3.8	67.5 $\pm$ 2.5	**68.3 $\pm$ 1.4**
	CiteSeer	55.8 $\pm$ 1.4	71.7 $\pm$ 1.4	67.5 $\pm$ 2.5	70.8 $\pm$ 3.8	69.2 $\pm$ 3.8	**77.5 $\pm$ 2.5**
	PubMed	60.0 $\pm$ 2.5	65.8 $\pm$ 2.9	69.2 $\pm$ 1.4	**71.7 $\pm$ 3.8**	68.3 $\pm$ 1.4	70.8 $\pm$ 1.4
	CS	55.8 $\pm$ 1.4	71.6 $\pm$ 1.4	76.7 $\pm$ 1.4	71.7 $\pm$ 2.9	75.8 $\pm$ 2.8	**78.3 $\pm$ 1.4**
	OGBN-Arxiv	49.2 $\pm$ 2.9	53.3 $\pm$ 1.4	**56.7 $\pm$ 1.4**	52.6 $\pm$ 2.5	55.8 $\pm$ 1.4	55.8 $\pm$ 1.4

- Our **GCORN model often outperforms** existing defense approaches when subject to feature based attacks.
- GCORN is also effective against **structure-based, as well as combined structure and feature attacks**.

A Simple and Yet Fairly Effective Defense for Graph Neural Networks

Ennadir, Abbahaddou, Lutzeyer, Vazirgiannis & Boström (2024, AAAI)

Problem Set-Up

Problem: Available defense methods often have high computational complexity and training time (often increasing with increasing graph size).

Solution Approach: We propose a GNN, called the *NoisyGNN*, in which **hidden states are perturbed** by random noise following a normal distribution $N \sim \mathcal{N}(0, \beta I)$, i.e., our GNNs are of the form

$$\hat{y} = \sigma\left(\tilde{A}\,\mathrm{ReLU}\left(\tilde{A}XW^{(1)} + N\right)W^{(2)}\right).$$

Theoretical Results

Upper Bounds on GNN Vulnerability

We consider structural perturbations of the input graph (A, X), with a budget ϵ and 2-layer GNNs with 1-Lipschitz continuous activation functions and weight matrices $W^{(1)}, W^{(2)}$.

- Then, the vulnerability of GCNs is upper bounded by

$$\frac{2(\|W^{(2)}\|\|W^{(1)}\|\|X\|\epsilon)^2}{\beta};$$

- Then, the vulnerability of GINs is upper bounded by

$$\frac{(\|W^{(2)}\|\|W^{(1)}\|\|X\|\epsilon(2\|A\|+\epsilon))^2}{2\beta}.$$

Insight: Our upper bound on the vulnerability of a GNN is **smaller for large** β yielding a **more robust GNN.**

Experimental Results

Table: Classification accuracy (± standard deviation) of combining defense methods with the proposed noise injection on different benchmark datasets.

Method	Cora	CiteSeer	PolBlogs
GINGuard	61.8±0.5	55.6±1.8	82.7±0.6
+ Noisy	**66.2±1.3**	**58.3±1.9**	**83.6±0.8**
GIN-Jaccard	70.4±1.1	61.2±2.3	-
+ Noisy	**72.9±0.8**	**64.9±1.8**	-
GCNGuard	69.5±0.7	66.2±0.6	64.7±0.8
+ Noisy	**72.4±1.2**	**68.9±0.9**	**65.8±1.3**
GCN-Jaccard	66.7±0.5	61.2±1.1	-
+ Noisy	**69.6±0.9**	**63.1±0.6**	-

- Our NoisyGCNs **sometimes outperform** other defense methods.
- NoisyGNNs are **faster to train** than most other defense methods.
- When **combined with other defense methods**, best performance is achieved.

Conclusions

- Graph Neural Networks (GNNs) are a versatile and powerful tool, that you may want to consider using.
- The categorisation of GNNs into *when and how* they alter the graph structure is insightful and could give rise to new models.
- The robustness of GNNs is one of many interesting problem domains that have opened up in graph learning.

I may be looking to hire Postdocs & PhD students soon!

Please email me if you are interested.

Thank you for your attention!

@JLutzeyer

References

Y. Abbahaddou, S. Ennadir, J. F. Lutzeyer, M. Vazirgiannis & H. Boström, "Bounding the Expected Robustness of Graph Neural Networks Subject to Node Feature Attacks," *International Conference on Learning Representations(ICLR),* 2024.

U. Alon & E. Yahav, "On the Bottleneck of Graph Neural Networks and its Practical Implications," In: *International Conference on Learning Representations (ICLR)*, 2020.

A. Bojchevski, J. Gasteiger, B. Perozzi, A. Kapoor, M. Blais, B. Rózemberczki, M. Lukasik & S. Günnemann, "Scaling graph neural networks with approximate pagerank," *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 2464-2473, 2020.

F. Borisyuk, S. He, Y. Ouyang, M. Ramezani, P. Du, X. Hou, C. Jiang, N. Pasumarthy, P. Bannur, B. Tiwana, P. Liu, "LiGNN: Graph Neural Networks at LinkedIn," *arXiv:2402.11139*, 2024.

S. Brody, U. Alon & E. Yahav, "How Attentive Are Graph Attention Networks?," In: *International Conference on Learning Representations (ICLR)*, 2022.

M. Bronstein, "Graph ML at Twitter," *Twitter Engineering Blog Post*, `https://blog.twitter.com/engineering/en_us/topics/insights/2020/graph-ml-at-twitter`, 2020.

M. Chatzianastasis, J. F. Lutzeyer, G. Dasoulas & M. Vazirgiannis, "Graph Ordering Attention Networks," *Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI),* 2023.

G. Dasoulas, J. F. Lutzeyer & M. Vazirgiannis, "Learning Parametrised Graph Shift Operators," In: *International Conference on Learning Representations (ICLR)*, 2021.

A. Deac, M. Lackenby & P. Veličković, "Expander Graph Propagation," *arXiv:2210.02997*, 2022.

V.P. Dwivedi & X. Bresson, "A Generalization of Transformer Networks to Graphs," *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, 2021.

S. Ennadir, Y. Abbahaddou, J. F. Lutzeyer, M. Vazirgiannis & H. Boström, "A Simple and Yet Fairly Effective Defense for Graph Neural Networks," *Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI),* 2024.

F. Geerts & J. L. Reutter, "Expressiveness and Approximation Properties of Graph Neural Networks," *International Conference on Learning Representations (ICLR)*, 2022.

I. J. Goodfellow, J. Shlens, & C. Szegedy, "Explaining and harnessing adversarial examples," International Conference of Learning Representations (ICLR), 2015.

S. Günnemann, "Graph Neural Networks: Adversarial Robustness," *Graph Neural Networks: Foundations, Frontiers, and Applications*, pp. 149–176, 2022.

A. Jain, I. Liu, A. Sarda & P. Molino, "Food Discovery with Uber Eats: Using Graph Learning to Power Recommendations," *Uber Engineering Blog Post*, `https://eng.uber.com/uber-eats-graph-learning/`, 2019.

Thomas N. Kipf & M. Welling, "Semi-supervised classification with graph convolutional networks," *International Conference on Learning Representations (ICLR)*, 2017.

O. Lange & L. Perez, "Traffic prediction with advanced Graph Neural Networks," *DeepMind Research Blog Post*, `https://deepmind.com/blog/article/traffic-prediction-with-advanced-graph-neural-networks`, 2020.

G. Liu, D. B. Catacutan, K. Rathod, K. Swanson, W. Jin, J. C. Mohammed, A. Chiappino-Pepe, S. A. Syed, M. Fragis, K. Rachwalski, J. Magolan, M. G. Surette, B. K. Coombes, T. Jaakkola, R. Barzilay, J. J. Collins, J. M. Stokes, "Deep learning-guided discovery of an antibiotic targeting Acinetobacter baumannii," *Nature Chemical Biology*, pp. 1–9, 2023.

G. Michel, G. Nikolentzos, J. Lutzeyer & M. Vazirgiannis, "Path Neural Networks: Expressive and Accurate Graph Neural Networks," *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

A. R. Ramos Vela, J. F. Lutzeyer, A. Giovanidis & M. Vazirgiannis, "Improving Graph Neural Networks at Scale: Combining Approximate PageRank and CoreRank," *NeurIPS New Frontiers in Graph Learning Workshop*, 2022.

G. Salha-Galvan, J. F. Lutzeyer, G. Dasoulas, R. Hennequin & M. Vazirgiannis, "Modularity-Aware Graph Autoencoders for Joint Community Detection and Link Prediction," *arxiv:2202.00961*, 2022.

M. E. A. Seddik, C. Wu, J. F. Lutzeyer & M. Vazirgiannis, "Node Feature Kernels Increase Graph Convolutional Network Robustness," *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

J. M. Stokes, K. Yang, K. Swanson, W. Jin, A. Cubillos-Ruiz, N. M. Donghia, C. R. MacNair, S. French, L. A. Carfrae, Z. Bloom-Ackermann, V. M. Tran, A. Chiappino-Pepe, A. H. Badran, I. W. Andrews, E. J. Chory, G. M. Church, E. D. Brown, T. S. Jaakkola, R. Barzilay & J. J. Collins, "A Deep Learning Approach to Antibiotic Discovery," *Cell*, pp. 688–702, 2020.

J. Topping, F. Di Giovanni, B. P. Chamberlain, X. Dong & M. M. Bronstein, "Understanding Over-Squashing and Bottlenecks on Graphs via Curvature," *International Conference on Learning Representations (ICLR)*, 2022.

P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò & Y. Bengio, "Graph Attention Networks," *6th International Conference on Learning Representations (ICLR),* 2018.

S. Virinchi, A. Saladi & A. Mondal, "Recommending Related Products Using Graph Neural Networks in Directed Graphs," In: *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2022.

J. Wang, P. Huang, H. Zhao, Z. Zhang, B. Zhao & Dik Lun Lee, "Billion-scale Commodity Embedding for E-Commerce Recommendation in Alibaba," In Proceedings of the *24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 839–848, 2018.

K. Xu, W. Hu, J. Leskovec & S. Jegelka."How powerful are graph neural networks?", *International Conference on Learning Representations (ICLR)*, 2019.

R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton & J. Leskovec, "Graph Convolutional Neural Networks for Web-Scale Recommender Systems," In Proceedings of the *24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 974–983, 2018.

Y. Zhou, H. Zheng & X. Huang, "Graph Neural Networks: Taxonomy, Advances and Trends," *arXiv:2012.08752*, 2020.