CONTRACTIVE SYSTEMS IMPROVE GRAPH NEURAL NETWORKS AGAINST ADVERSARIAL ATTACKS

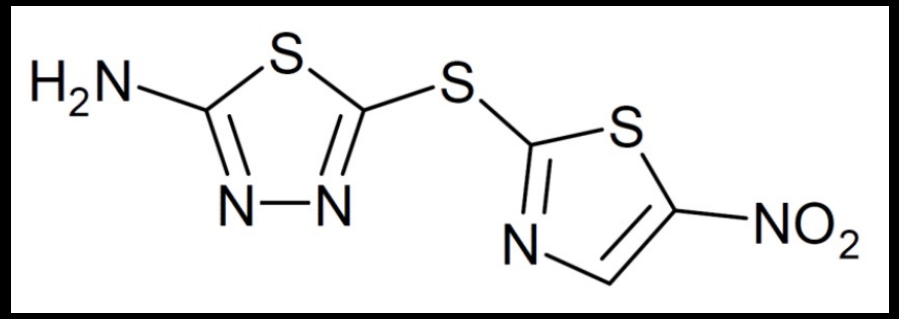
Anonymous authors
Paper under double-blind review

ABSTRACT

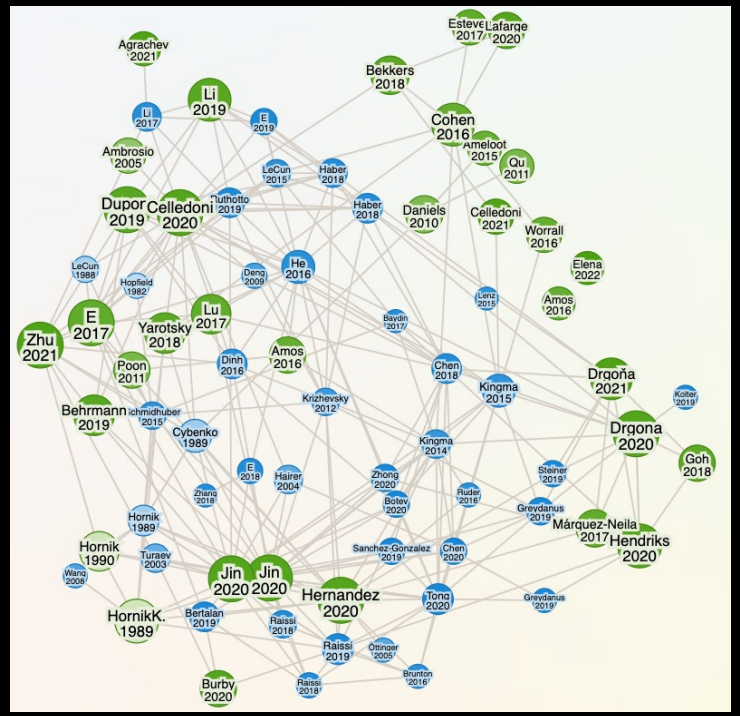
Graph Neural Networks (GNNs) have established themselves as a key component in addressing diverse graph-based tasks. Despite their notable successes, GNNs remain susceptible to input perturbations in the form of adversarial attacks. This paper introduces an innovative approach to fortify GNNs against adversarial perturbations through the lens of contractive dynamical systems. Our method introduces graph neural layers based on differential equations with contractive properties, which, as we show, improve the robustness of GNNs. A distinctive feature of the proposed approach is the simultaneous learned evolution of both the node features and the adjacency matrix, yielding an intrinsic enhancement of model robustness to perturbations in the input features and the connectivity of the graph. We mathematically derive the underpinnings of our novel architecture and provide theoretical insights to reason about its expected behavior. We demonstrate the efficacy of our method through numerous real-world benchmarks, reading on par or improved performance compared to existing methods.

With Moshe Eliasof, Ferdia Sherry and Carola Schönlieb

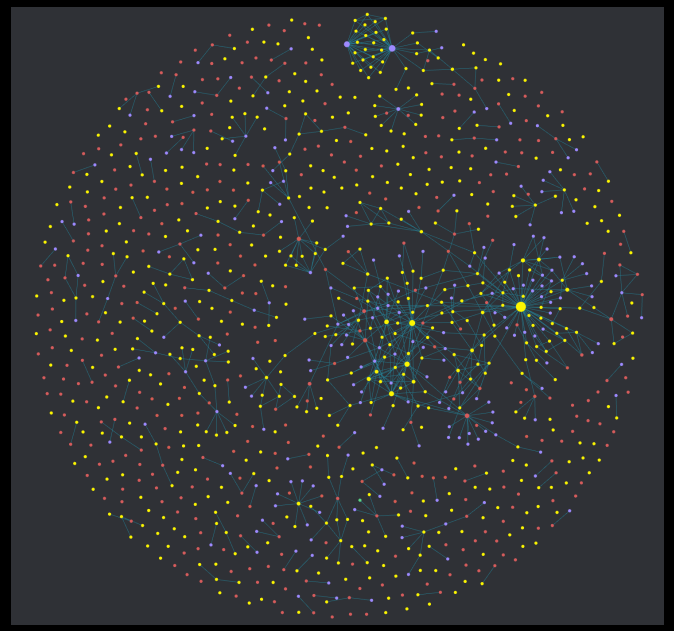
Graphs are everywhere



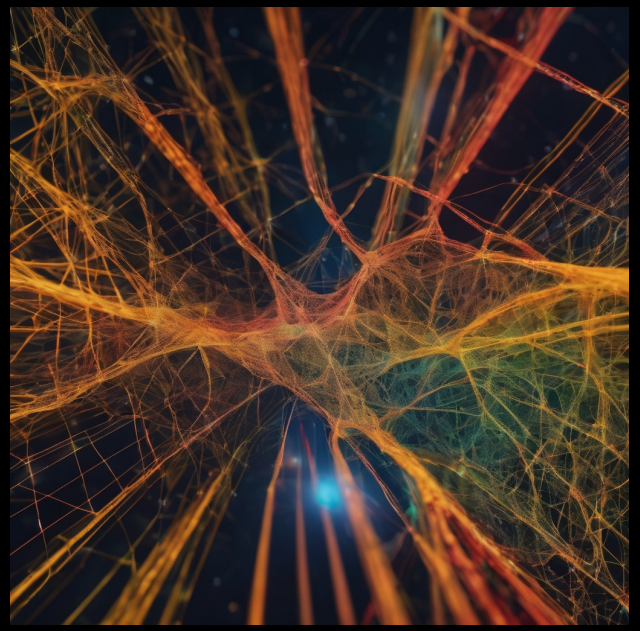
Molecule structure



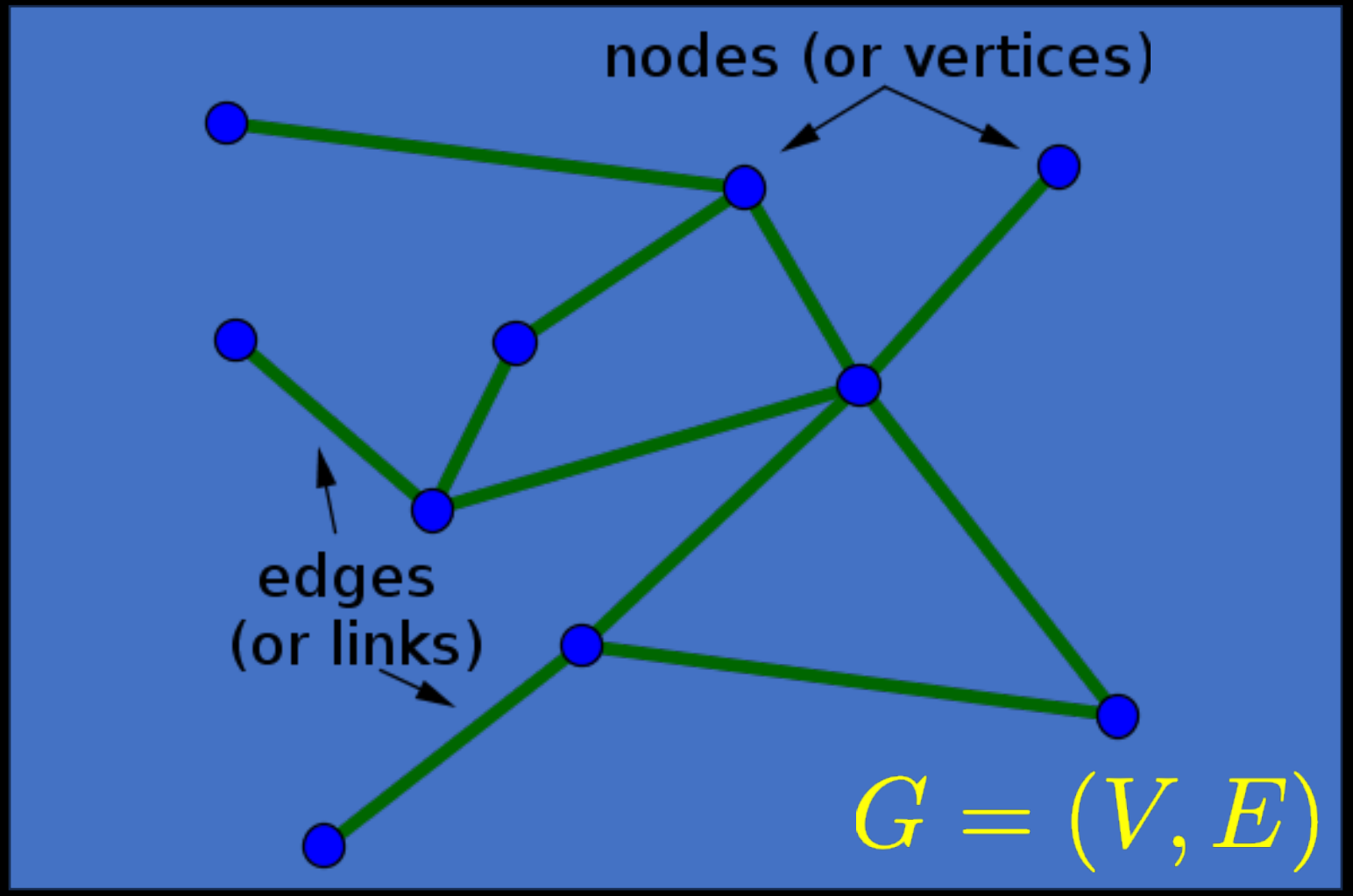
Citation graph of papers I saved



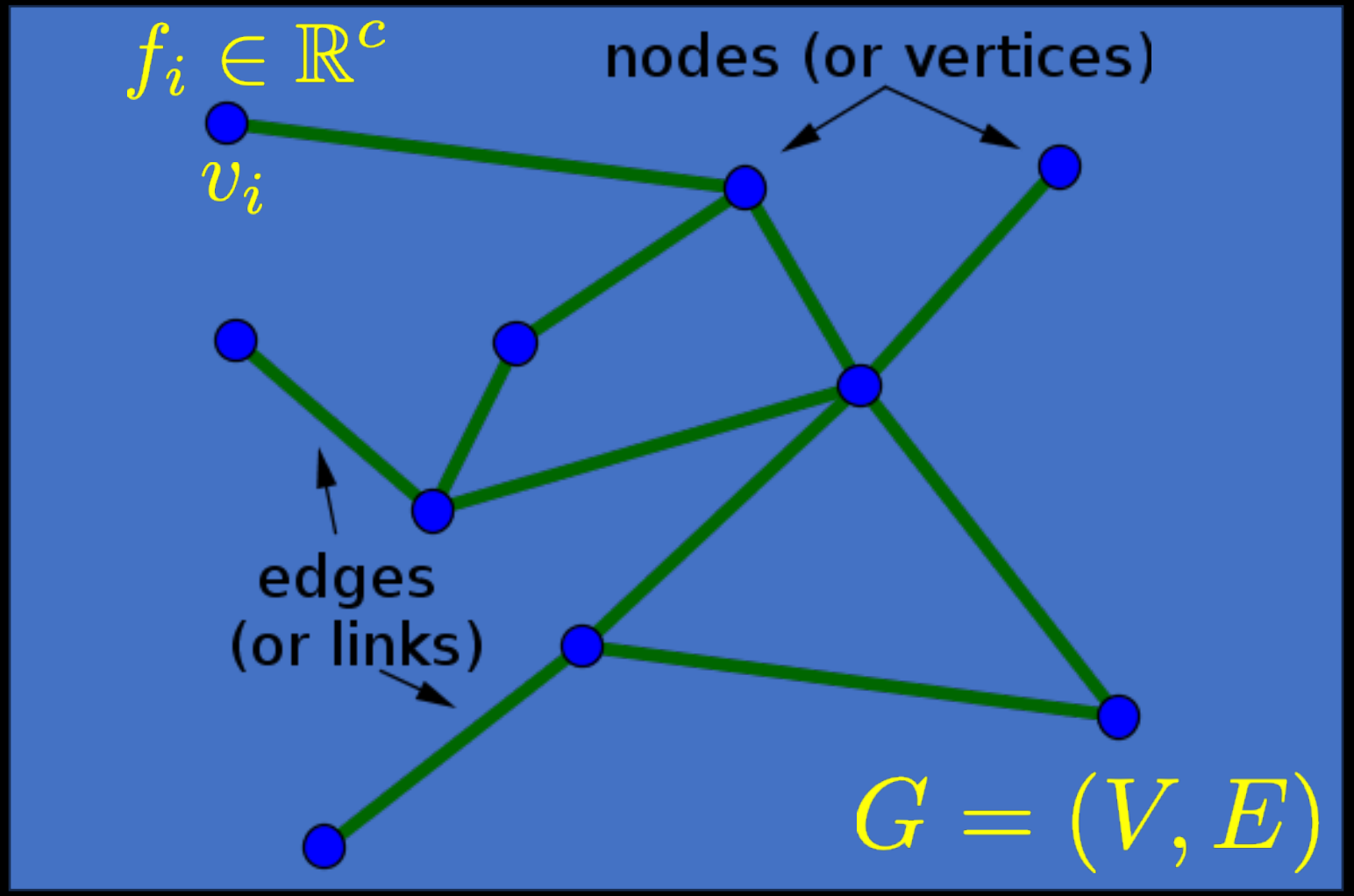
Reference graph some notes
I wrote in Obsidian



Connectivity of neurons
in the brain



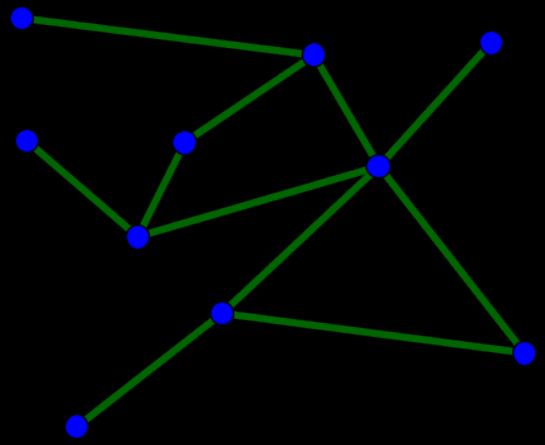
$$A \in \{0,1\}^{n \times n},\ A^T = A$$



$$F \in \mathbb{R}^{n \times c}, e_i^T F = f_i$$

Classical tasks solved with GNNs

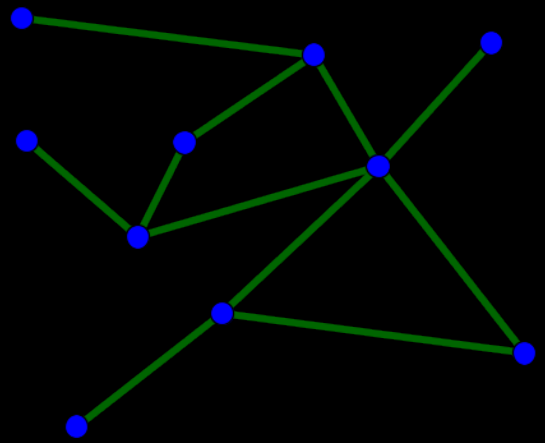
Graph classification



GNN

It is a protein

Node classification



GNN

Red
Blue
Blue
…
Green

Usual structure of GNNs

$$F^{(0)} = F$$

$$F^{(l+1)} = T_l\left(F^{(l)}, A\right), l = 0, \ldots L-1$$

$$F^{(O)} = \mathrm{MLP}\left(\mathrm{F}^{(\mathrm{L})}\right) =: \mathbf{GNN}(\mathbf{F}, \mathbf{A})$$

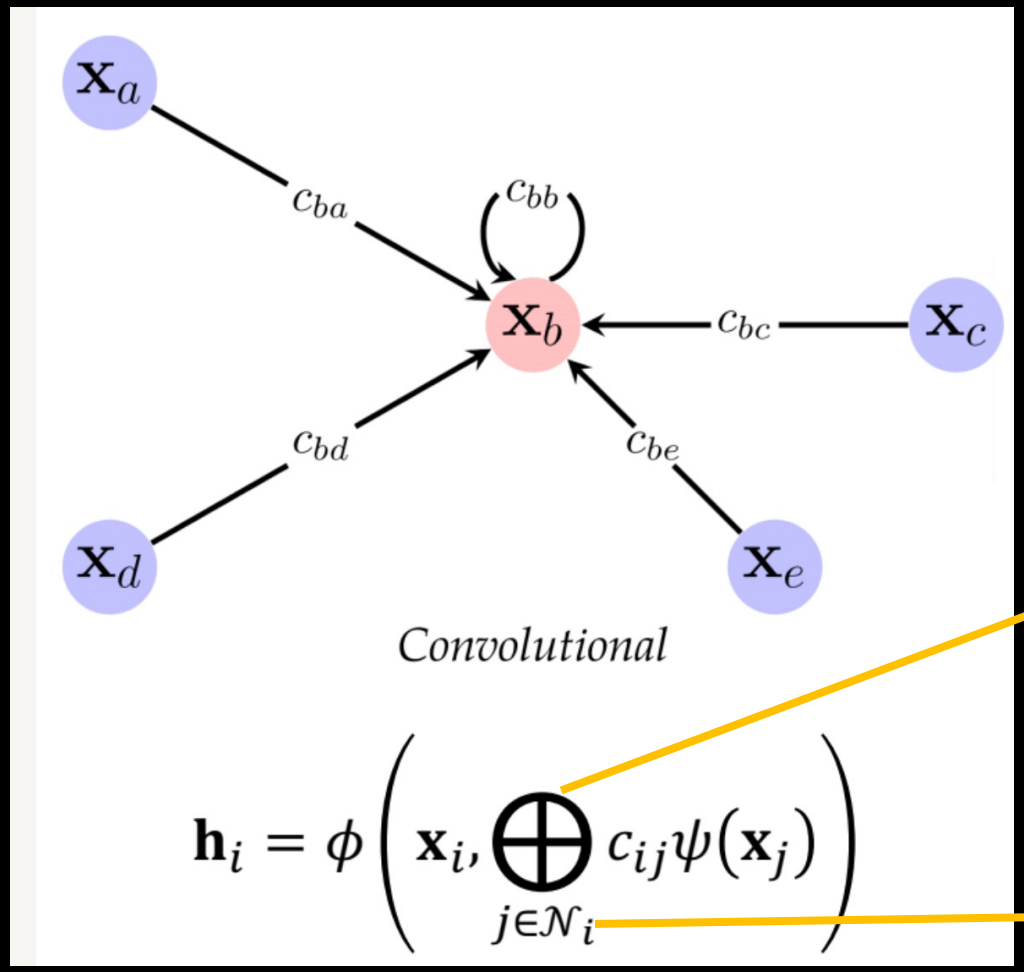
Invariant

$$\mathrm{GNN}(\mathrm{F}, \mathrm{A}) = \mathrm{GNN}(\mathrm{PF}, \mathrm{PAP}^{\mathrm{T}})$$

$$P\mathrm{GNN}(\mathrm{F}, \mathrm{A}) = \mathrm{GNN}(\mathrm{PF}, \mathrm{PAP}^{\mathrm{T}})$$

Equivariant

Usual structure of GNNs

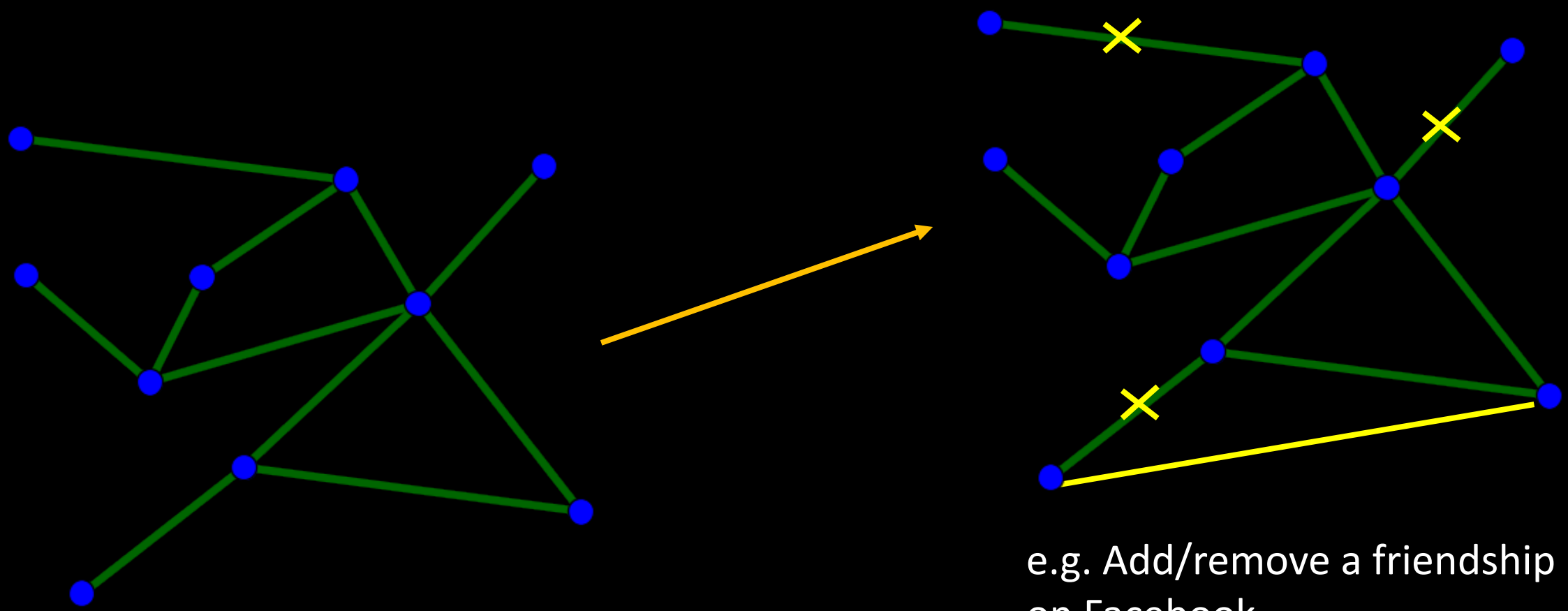


Permutation invariant aggregator, like a sum

Neighbourhood of the i-th node

Source : https://geometricdeeplearning.com/lectures/

Adversarial attacks



e.g. Add/remove a friendship on Facebook

Adversarial attacks

$$A_* = A + \delta A, \quad \|\delta A\|_0 \leq \varepsilon_1$$

$$F_* = F + \delta F, \quad \|\delta F\|_F \leq \varepsilon_2$$

Attacks do not break the properties of symmetry generally

Goal:

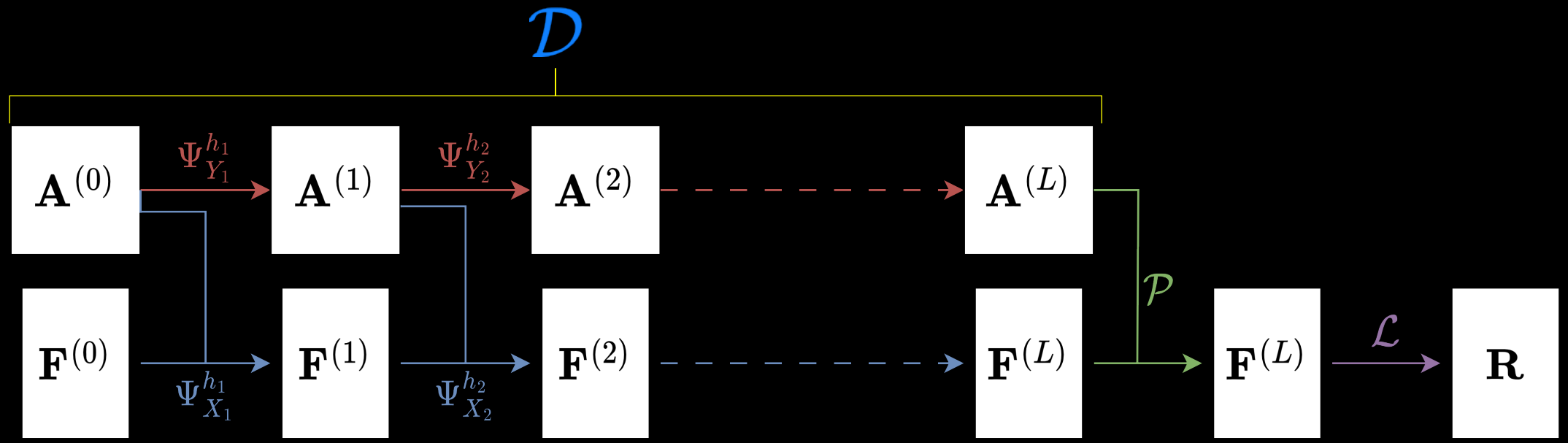
$$\mathrm{GNN}(\mathrm{F}, \mathrm{A}) \approx \mathrm{GNN}(\mathrm{F}_*, \mathrm{A}_*)$$

Remark on Nuclear Norm

$$A \in \{0,1\}^{n\times n} \implies \|A\|_0 = \#\{i,j \in \{1,\dots,n\} : A_{ij} \neq 0\} = \|\mathrm{vec(A)}\|_{\ell^1}$$

The 1-norm of the vectorisation is better suited for what we do, and we will use such norm instead of the nuclear norm.

Our proposed architecture: CSGNN



$$(F^{(0)}, A^{(0)}) := (\mathcal{K}(F_*), A_*)$$

$$\Psi_{X_i}^{h_i}(F, A) = F - h_i G(A)^T \sigma(G(A) F W_i) W_i^T \left(\frac{K_i + K_i^T}{2} \right)$$

$$\Psi_{Y_i}^{h_i}(A) = A + h_i \sigma(M_i(A))$$

Our proposed architecture: CSGNN

$$M(A) = k_1 A + k_2 \mathrm{diag}(\mathrm{diag}(A)) + \frac{k_3}{2n}(A\mathbf{1}_n\mathbf{1}_n^\top + \mathbf{1}_n\mathbf{1}_n^\top A) + k_4 \mathrm{diag}(A\mathbf{1}_n)$$
$$+ \frac{k_5}{n^2}(\mathbf{1}_n^\top A\mathbf{1}_n)\mathbf{1}_n\mathbf{1}_n^\top + \frac{k_6}{n}(\mathbf{1}_n^\top A\mathbf{1}_n)I_n + \frac{k_7}{n^2}(\mathbf{1}_n^\top \mathrm{diag}(A))\mathbf{1}_n\mathbf{1}_n^\top$$
$$+ \frac{k_8}{n}(\mathbf{1}_n^\top \mathrm{diag}(A))I_n + \frac{k_9}{2n}(\mathrm{diag}(A)\mathbf{1}_n^\top + \mathbf{1}_n(\mathrm{diag}(A))^\top)$$

$$M(PAP^T) = PM(A)P^T, \quad (M(A))^T = M(A)$$

Robustness of the network

Theorem 2 (Equation (8) can induce contractive node dynamics)**.** *There are choices of* $(\mathbf{W}_l, \mathbf{K}_l) \in \mathbb{R}^{n\times n} \times \mathbb{R}^{c\times c}$*, for which the explicit Euler step in Equation* (8) *is contractive for a small enough* $h_l > 0$*, i.e.*

$$\|\Psi^{h_l}_{X_l}(\mathbf{F}+\delta\mathbf{F},\mathbf{A}) - \Psi^{h_l}_{X_l}(\mathbf{F},\mathbf{A})\|_F \leq \|\delta\mathbf{F}\|_F, \quad \delta\mathbf{F} \in \mathbb{R}^{n\times c}. \tag{10}$$

Theorem 3 (Equation (14) defines contractive adjacency dynamics)**.** *Let* $\alpha \leq 0$*,* $\sigma : \mathbb{R} \to \mathbb{R}$ *be a Lipschitz continuous function, with* $\sigma'(s) \in [0,1]$ *almost everywhere. If* $0 \leq h_l \leq \hat{h}_l^{\text{adj}} := \frac{2}{(2\sum_{i=2}^{9}|k_i|)-\alpha}$*, then the explicit Euler step*

$$\mathbf{A}^{(l)} = \Psi^{h_l}_{Y_l}(\mathbf{A}^{(l-1)}) := \mathbf{A}^{(l-1)} + h_l\sigma\left(M(\mathbf{A}^{(l-1)})\right), \tag{15}$$

where $k_1 = \left(\alpha - \sum_{i=2}^{9}|k_i|\right)$*, is contractive in the vectorized* ℓ^1 *norm.*

$$\begin{aligned} d(\mathcal{D}(\mathbf{F}^{(0)},\mathbf{A}^{(0)}),\mathcal{D}(\mathbf{F}^{(0)}_*,\mathbf{A}^{(0)}_*)) &:= \|\operatorname{vec}(\mathbf{A}^{(L)}) - \operatorname{vec}(\mathbf{A}^{(L)}_*)\|_1 + \|\mathbf{F}^{(L)} - \mathbf{F}^{(L)}_*\|_F \\ &\leq \varepsilon_1 + \varepsilon_2\left(1 + \sum_{i=1}^{L}\operatorname{Lip}(X_{i,\mathbf{F}^{(i-1)}})h_i\right) \\ &=: \varepsilon_1 + c(h_1,\ldots,h_L)\varepsilon_2. \end{aligned}$$

Experimental setup

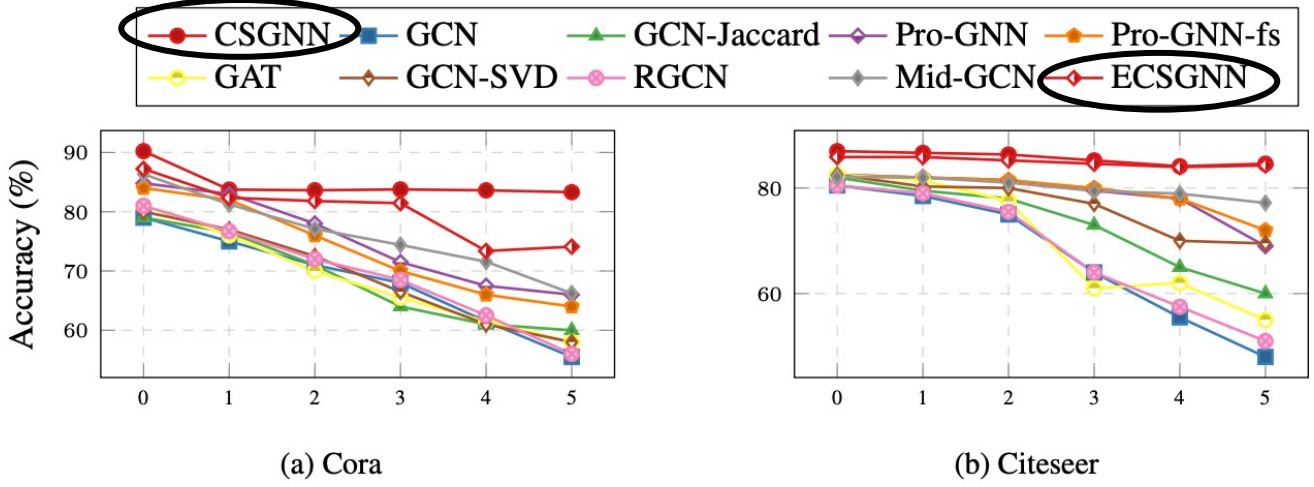
Table 3: Hyperparameter ranges

Hyperparameter	Range	Distribution
input/output embedding learning rate	$[10^{-5}, 10^{-2}]$	uniform
node dynamics learning rate	$[10^{-5}, 10^{-2}]$	uniform
adjacency dynamics learning rate	$[10^{-5}, 10^{-2}]$	uniform
input/output embedding weight decay	$[5 \cdot 10^{-8}, 5 \cdot 10^{-2}]$	log uniform
node dynamics weight decay	$[5 \cdot 10^{-8}, 5 \cdot 10^{-2}]$	log uniform
adjacency dynamics weight decay	$[5 \cdot 10^{-8}, 5 \cdot 10^{-2}]$	log uniform
input/output embedding dropout	$[0, 0.6]$	uniform
node dynamics dropout	$[0, 0.6]$	uniform
share weights between time steps	$\{\text{yes}, \text{no}\}$	discrete uniform
step size h	$[10^{-2}, 1]$	log uniform
adjacency contractivity parameter α	$[-2, 0]$	uniform
#layers L	$\{2, 3, 4, 5\}$	discrete uniform
#channels c	$\{8, 16, 32, 64, 128\}$	discrete uniform

Some experimental results

Method	Cora			Citeseer		
	nettack	metattack	random	nettack	metattack	random
$\text{CSGNN}_{\text{noAdj}}$	81.90	70.25	77.19	82.20	70.17	71.28
CSGNN	83.29	74.46	78.38	84.60	72.94	72.70

Table 6: The influence of learning the adjacency dynamical system $\Psi_{Y_l}^{h_l}$. The results show the node classification accuracy (%) with and without learning the adjacency dynamical system.



(a) Cora (b) Citeseer

Figure 8: Node classification accuracy (%) of ECSGNN and other baselines, under a targeted attack generated by nettack. The horizontal axis describes the number of perturbations per node.

We target the nodes with degree at least 10 and flip few of their incident edges

Some experimental results

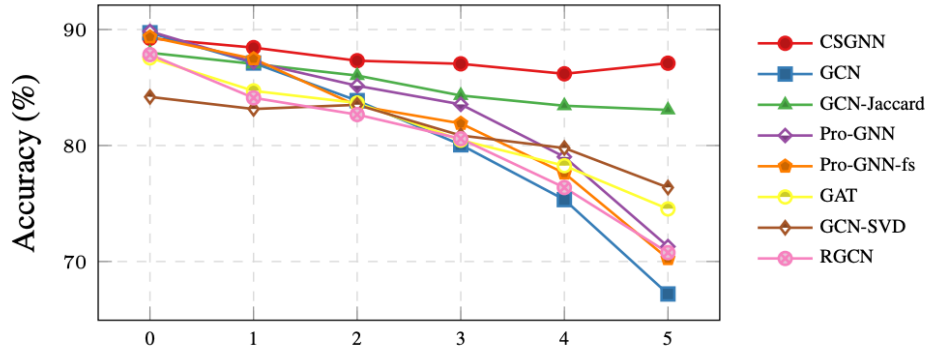


Figure 5: Node classification accuracy (%) on the Pubmed dataset using nettack as attack method. The horizontal axis describes the number of perturbations per node.

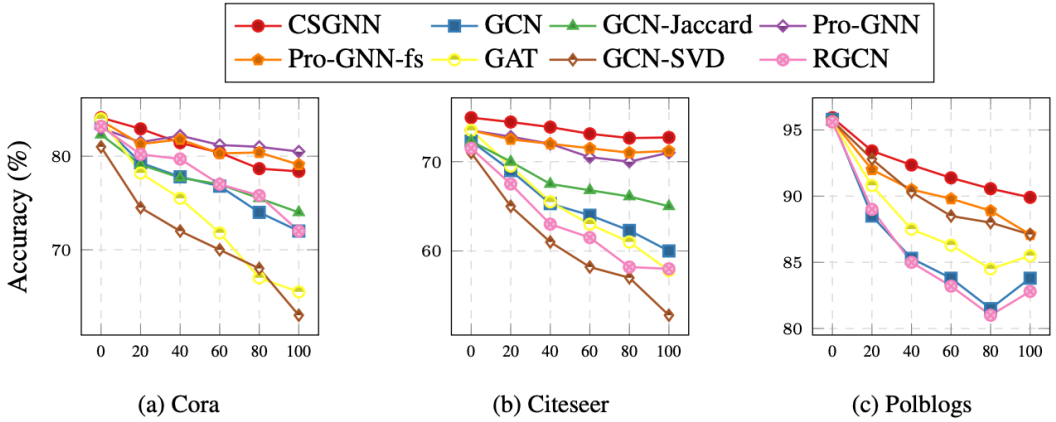


Figure 3: Node classification accuracy (%) under a random adjacency matrix attack. The horizontal axis describes the attack percentage.

The adjacency matrix is attacked by adding random fake edges, from 0% to 100 %of the number of edges in the true adjacency matrix