Lecture 12 EN.553.744 Prof. Luana Ruiz

Today: Stability of GNNs to graph perturbations

→ Sampling-based GNN training

- Step 1: Divide V in random batches $V_{B_1}, V_{B_2}, \dots$
  (number of batches will depend on batch size — hyperparameter you set)

- Step 2: (batches) for $i=1$ to number of batches, do:
  (elements in batch) for j in B_i, do:
  (layers) for $\ell=1$ to L, do:

$$[x_\ell]_j = \sigma\Big(\sum_{k=0}^{K-1} \sum_{p\in\mathcal{N}(j)} [S]_{jp}[x_{\ell-1}]_p H_{k\ell}\Big)$$

only the embeddings of $j \in V_{B_i}$ are updated here

$$H_{k\ell}^{i} = H_{k\ell}^{i-1} - \eta \sum_{\substack{j\in V_{B_i} \\ \& \\ j\in\mathcal{T}}} \nabla \ell([Y]_j, [X_L]_j)$$

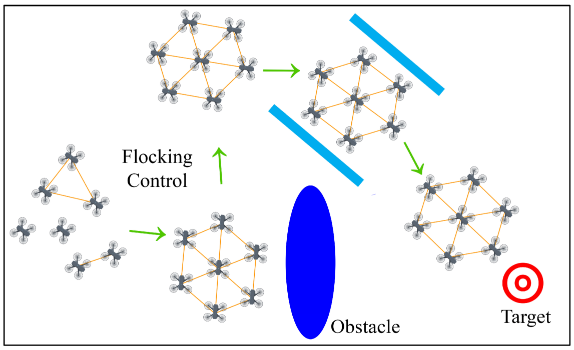
→ loss & weight updates only computed for nodes in batch

if semi-supervised, only look at training set $\mathcal{T}$ ←

Back to stability:

→ Why stability?

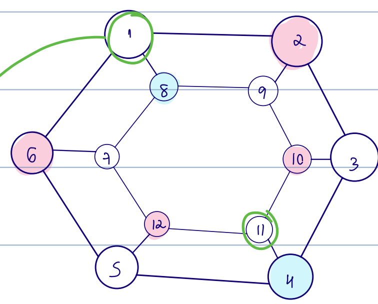
1) Graphs are susceptible to small perturbations, but GNN outputs should vary as little as possible;
e.g., GNN controller trained on offline trajectories has to perform well in online trajectories



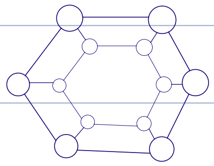
2) Recall that permutation equivariance functions as implicit data augmentation:

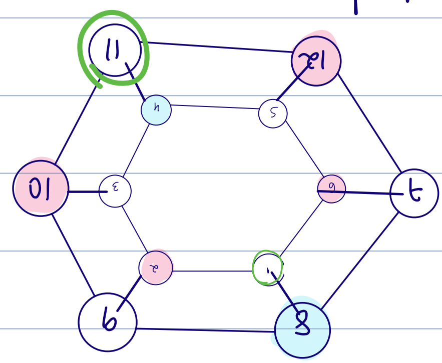
suppose we observe 1's label y_1 at training time, but not 11's

GNN learns to output $\tilde{y}_{11}$



What label will the GNN output for node 11?





automorphism P:
homomorphism
from the graph
onto itself:

$S = PSP^T \qquad x = Px$

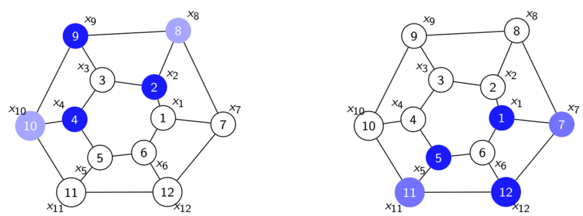
perm. equiv.

So: $Py = H(\underbrace{PSP^T}_{S})\underbrace{Px}_{x} = H(S).x = y \implies \tilde{y}_{11} = \tilde{y}_1$!

automorphism

=> We do not need to know y_{11} at training time — only y_1!

In practice, most graphs don't have such automorphisms (symmetries), but quasi-symmetries:



↳ Stability to perturbations ensures "data augmentation" under "quasi-symmetries"

To be more formal, first define:

(DEF) Operator distance modulo permutations:

$$\|\Psi - \Psi'\|_{\mathcal{P}} = \min_{P \in \mathcal{P}} \max_{x : \|x\| = 1} \|P^T \Psi(x) - \Psi'(P^T x)\|_2$$

Equivariance to permutations of graph filters

means: $\|S - S'\|_{\mathcal{P}} = 0 \Rightarrow \|H(S) - H(S')\|_{\mathcal{P}} = 0$

(for GNNs, $\|S - S'\|_{\mathcal{P}} = 0 \Rightarrow \|\Phi(S) - \Phi(S')\|_{\mathcal{P}} = 0$)

when $\|S - S'\|_{\mathcal{P}} \leq \varepsilon \Rightarrow$ quasi-symmetry

↳ we want $\|\Phi(S) - \Phi(S')\|_{\mathcal{P}} \leq C.\,\varepsilon$

Why stability only to graph perturbations?

$$y = \sum_{k=0}^{K-1} h_k S^k x$$

↳ linear in h_k & x

- if $x' = x + \varepsilon$, $\|y - y'\| \leq (K.\max_k h_k \lambda_1^{k-1})\,\varepsilon$
- if $h_k' = h_k + \varepsilon$, $\|y - y'\| \leq (\lambda_1^k \|x\|)\,\varepsilon$

$\Rightarrow$ graph convolutions are naturally Lipschitz stable to perturbations in x & h_*

(DEF) Lipschitz stability / continuity

A function $f: x \mapsto y$ is C-Lipschitz stable:
if $\| f(x) - f(x') \| \leq C . \| x - x' \| \quad \forall x, x'$
or, $\| f'(x) \| \leq C \quad \forall x$

↳ but graph convolution is nonlinear in S $\Rightarrow$ more challenging stability

We'll study stability to 3 perturbation types.

(First for graph convolutions, then for GNNs)

1) Dilations:

$S' = (1+\varepsilon) S \quad \Rightarrow \quad \| S' - S \| = \varepsilon \| S \|$

↳ edges scaled by $(1+\varepsilon)$; reasonable as edges change proportional to their values

↳ unrealistic as proportion is the same for all edges;

no additions & deletions possible

(Thm) Let $S' = (1+\varepsilon)S$ and consider graph convolution $H(S)$. If $H(S)$ is an integral Lipschitz filter with constant C, then

$$\| H(S) - H(S') \| \leq C \cdot \varepsilon + \mathcal{O}(\varepsilon^2)$$

Quick detour: Lipschitz and integral Lipschitz filters

Recall that, in the spectral domain,

$y = H(S)x = \sum_{k=0}^{K-1} h_k S^k x$ has frequency response:

$$\hat{y} = \sum_{k=0}^{K-1} h_k \Lambda^k \hat{x}$$

$$\Rightarrow$$

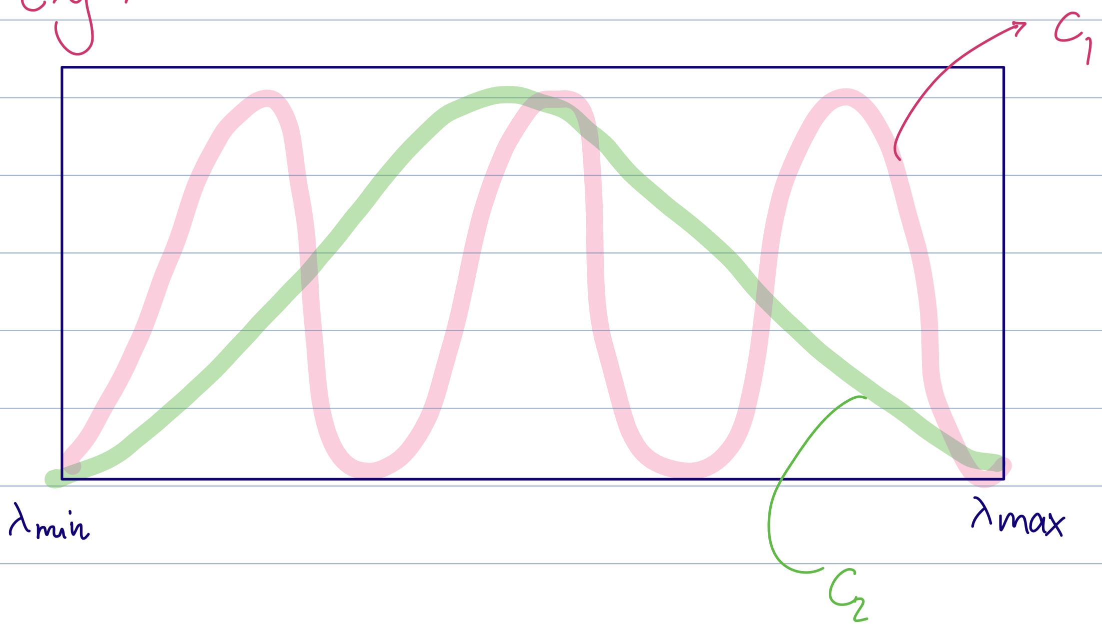
$$\hat{y} = h(\Lambda)\hat{x}$$

where $h(\lambda) = \sum_{k=0}^{K-1} h_k \lambda^k$

- **Lipschitz filters:** $h(\lambda)$ is Lipschitz, i.e., $\exists\, c$ s.t.

$$|h(\lambda) - h(\lambda')| \leq c\,|\lambda - \lambda'| \quad \underbrace{\forall\, \lambda, \lambda'}_{\text{within an interval}}$$

E.g.:

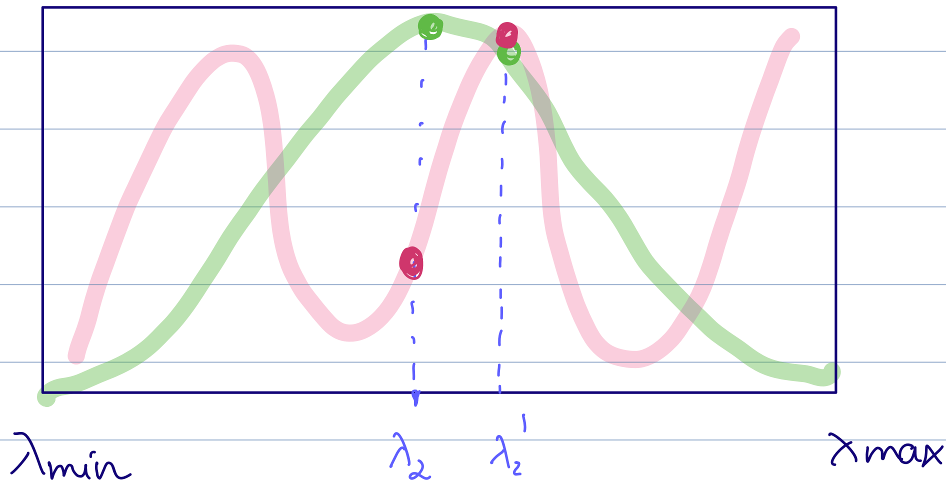


$$c_1 > c_2$$

C can be as large (or as small) as you want.

Larger c means the filter is more discriminative.

E.g.:



↳ same discriminability at all frequencies