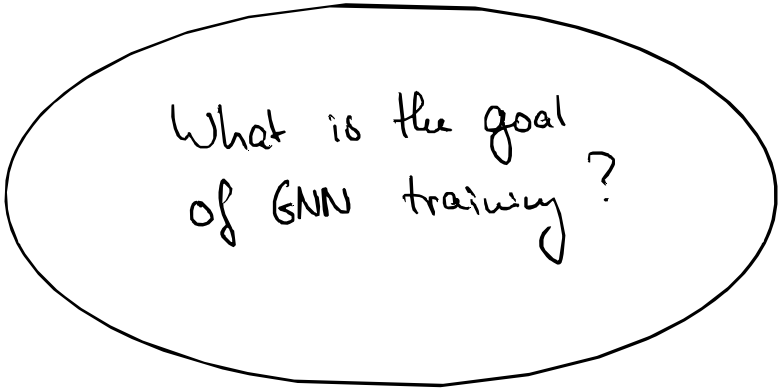
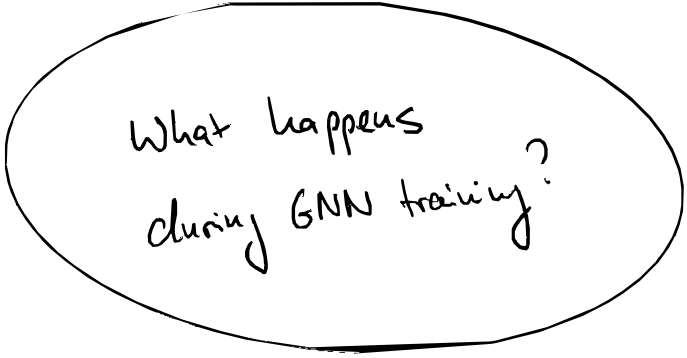
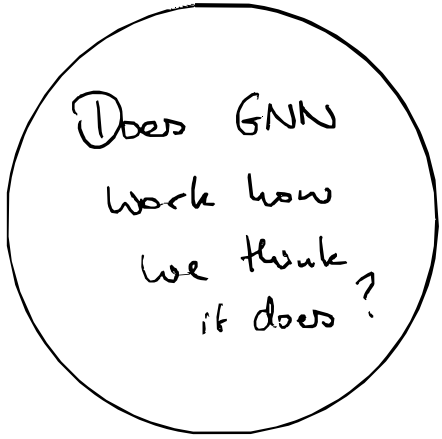
GNNs don't need Backprop

Benoit Gaujal, Fabian Jogl, Pascal Welke

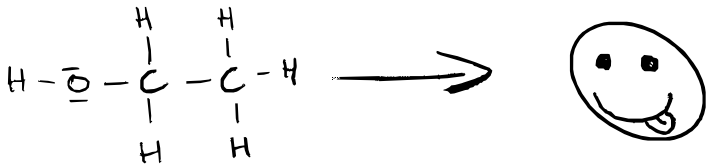
Ecole Polytechnique TU Wien Lancaster University Leipzig


What is the goal of GNN training?

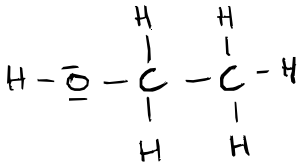

What happens during GNN training?


Does GNN work how we think it does?

Supervised Learning on Graphs



Neural Networks on Graphs



Message Passing Graph Neural Network Layer 1

...

Message Passing Graph Neural Network Layer k

Graph level pooling

Feed forward neural network

To train, we usually use

Stochastic Gradient Descent

To train, we usually use

Stochastic Gradient Descent

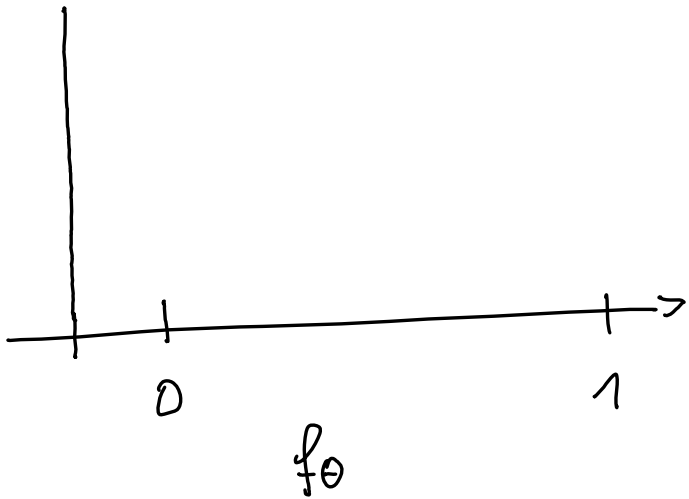
But do we need to?

Usually, we use a loss function that is some sort of <u>distance</u> (well, more or less) to a target value.

$$\text{Loss}(G) = \left| f_\theta(G) - y(G) \right|_2^2$$

$y(\text{  }) = 1$

$y(\text{  }) = 0$

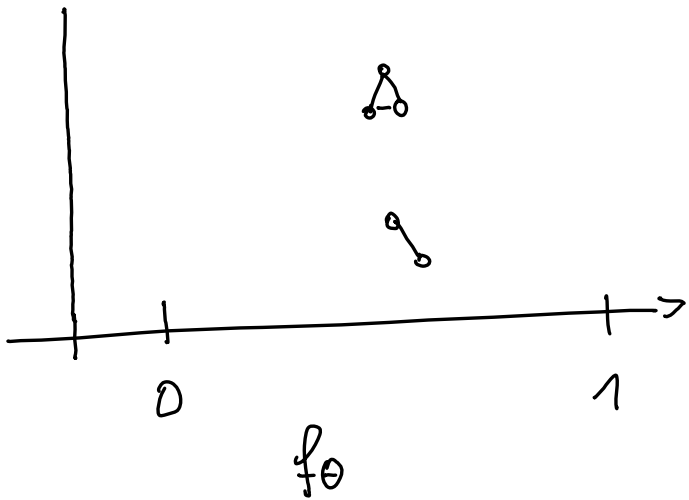


Usually, we use a loss function that is some sort of distance (well, more or less) to a target value.

$$\text{Loss}(\theta) = \left| f_\theta(\theta) - y(\theta) \right|_2^2$$

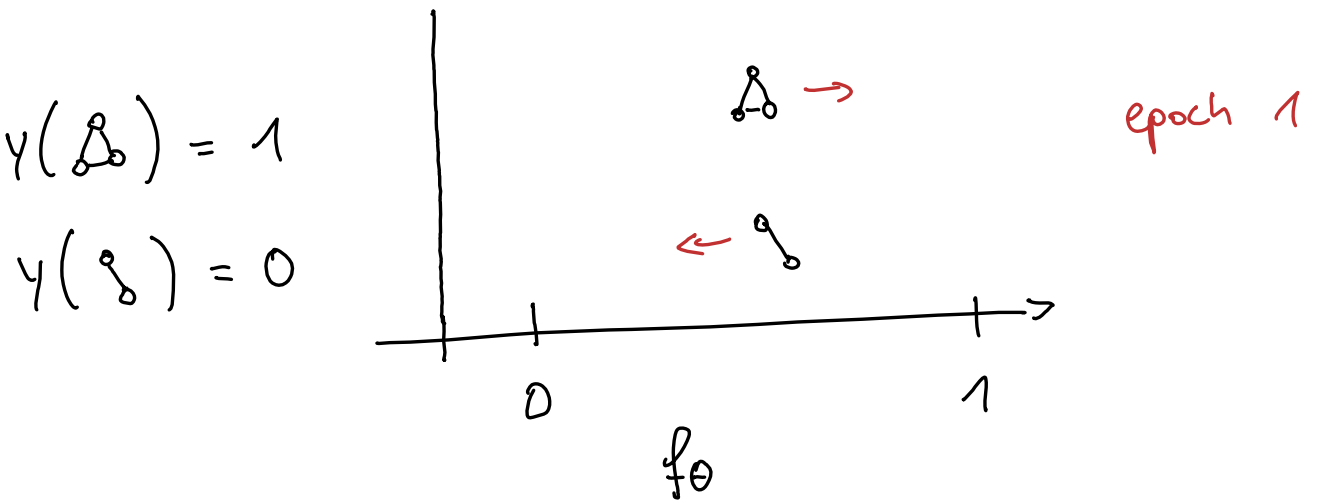
$y(\triangle) = 1$

$y(\text{molecule}) = 0$



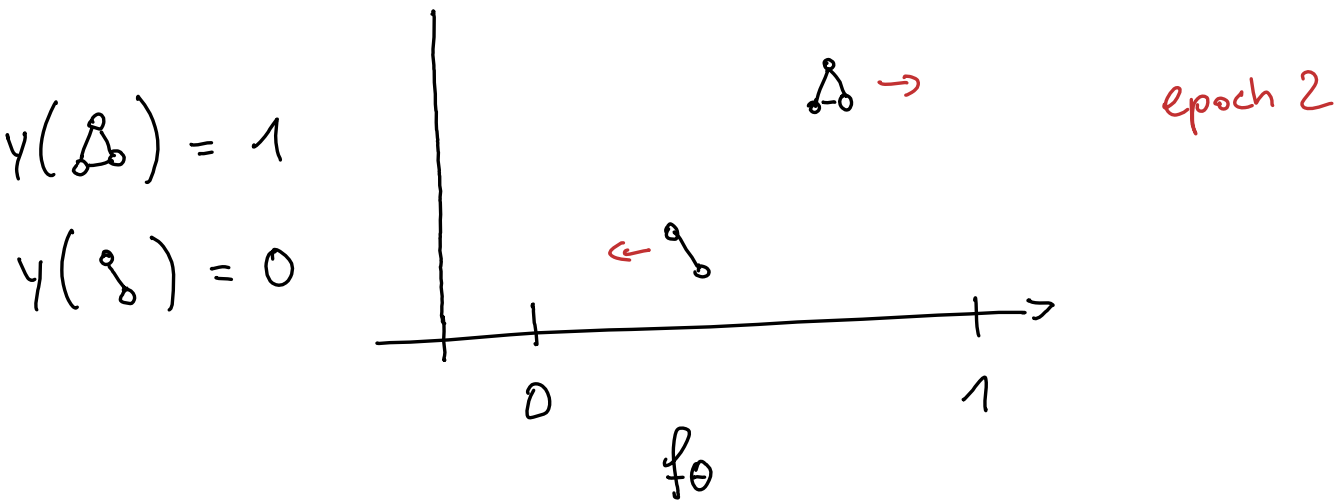
Usually, we use a loss function that is some sort of distance (well, more or less) to a target value.

$$\text{Loss}(\theta) = \left| f_\theta(\theta) - y(\theta) \right|_2^2$$



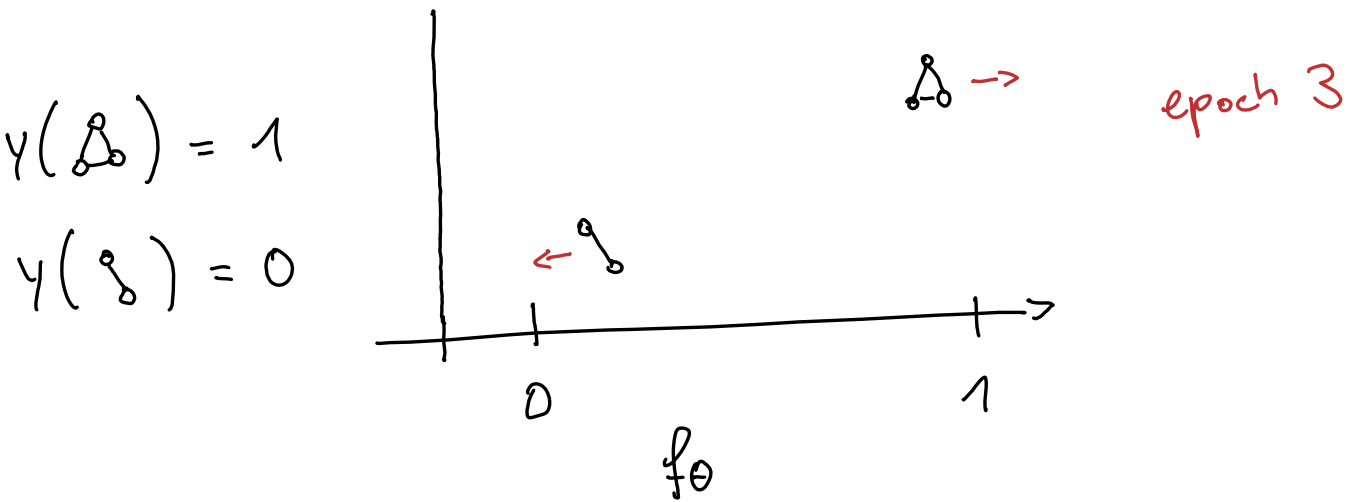
Usually, we use a loss function that is some sort of distance (well, more or less) to a target value.

$$\text{Loss}(\theta) = \left| f_\theta(\theta) - y(\theta) \right|_2^2$$



Usually, we use a loss function that is some sort of distance (well, more or less) to a target value.

$$\text{Loss}(\theta) = |f_\theta(\theta) - y(\theta)|_2^2$$



– Often repeated assumption in learning –

„ Similar graphs behave similarly "

– Often repeated assumption in learning –

"Similar graphs behave similarly"

What about:

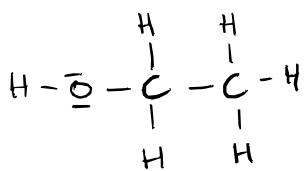
"Two graphs are similar
if they behave similarly
under γ"

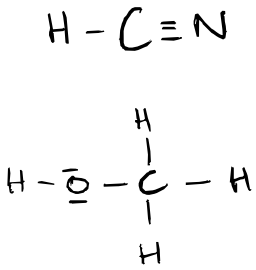
"Two graphs are similar if they behave similarly under y"

The best distance function we can define for a given learning problem is

$$d_y(G,H) = d'(y(G), y(H))$$

where d' is some sensible distance function on the target labels



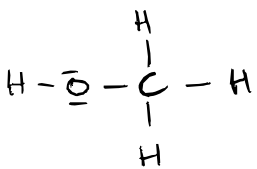


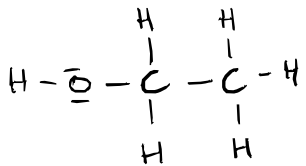
$$d'\left(\text{H-C}\equiv\text{N},\ \text{H-O-CH}_2\text{-CH}_3\right) = 0$$

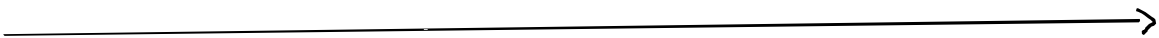
$$d'\left(\text{H-C}\equiv\text{N},\ \text{H-O-CH}_2\text{-CH}_2\text{-H}\right) = 1$$

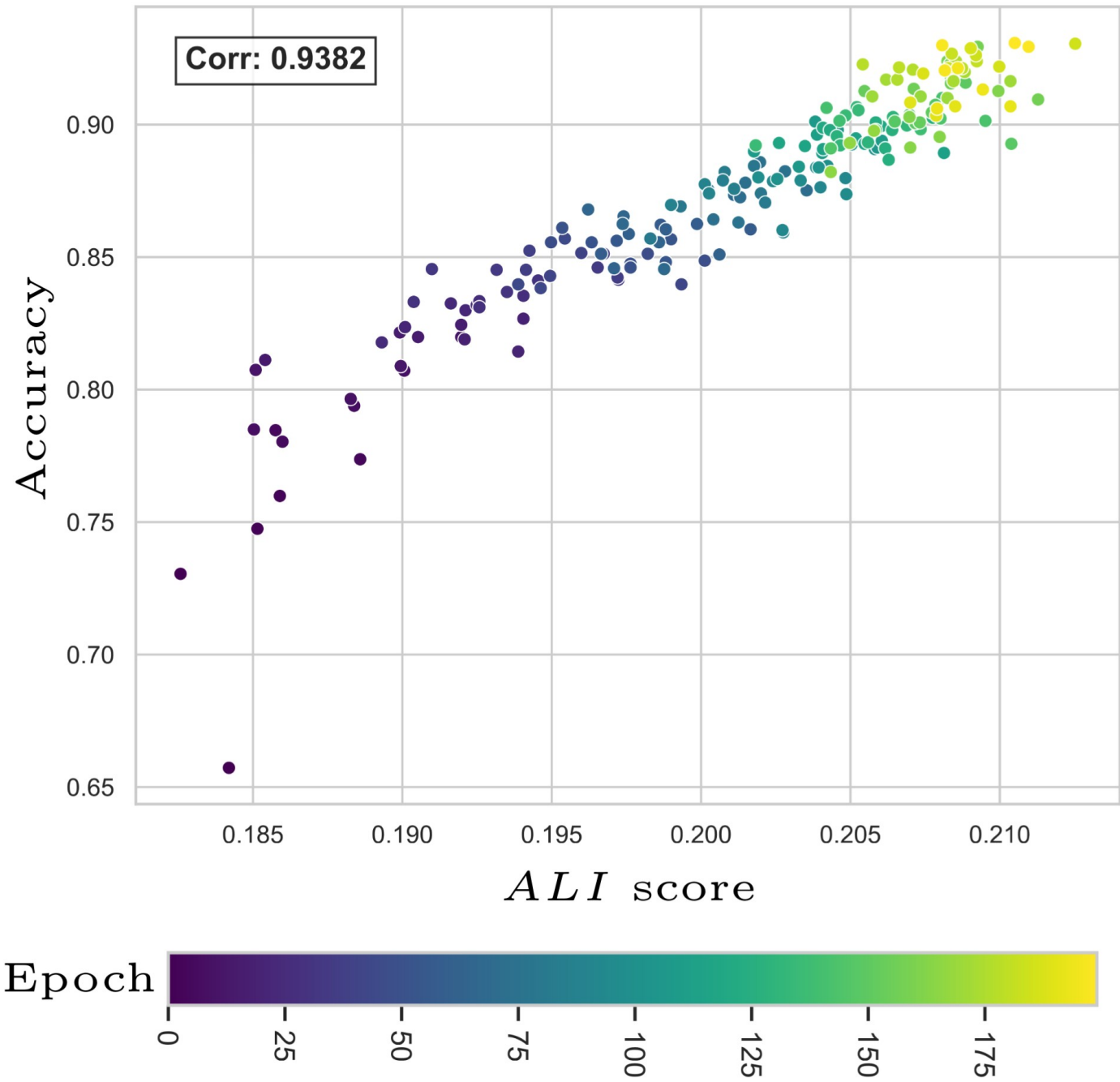
$$d'\left(\text{H-O-CH}_2\text{-H},\ \text{H-O-CH}_2\text{-CH}_2\text{-H}\right) = 1$$

$\text{H-C}\equiv\text{N}$








Corr: 0.9382
Accuracy
0.90
0.85
0.80
0.75
0.70
0.65
0.185
0.190
0.195
0.200
0.205
0.210
ALI score
Epoch
0
25
50
75
100
125
150
175

Now to the technical part

If we want a distance function

$$d(G,H) = d'(\gamma(G), \gamma(H))$$

Now to the technical part

If we want a distance function

$$d(G,H) \approx d'(y(G), y(H))$$

$$\Downarrow$$

$$|f_\theta(G) - f_\theta(H)|_2$$

GNN

Now to the technical part

If we want a distance function

$$d(G,H) \approx d'(y(G), y(H))$$

ss

$$|f_\theta(G) - f_\theta(H)|_2$$

↑ GNN →

Do we really need to use SGD?

Navarin et al. An Untrained Graph Neural Network for fast and accurate Graph Classification ICANN '23

"Take a randomly initialized GNN and just train a linear model on top. Done."

Böker et al. Fine-grained Expressivity of GNNs. Neurips '23

"Untrained GNNs behave as well as trained ones"

Folklore:

Randomly initialized GNNs are basically Weisfeiler Leman

(and that works well for learning)

Bui et al. Random Propagations in GNNs. Unireps '24

Huang et al:

You can have better GNNs by not training at all LoG '22

Zaman et al.

Graph Random Neural Features for Distance Preserving Graph Representations

ICML '20

Outside of GNNs: (a few examples)

Rahimi & Recht:

Weighted Sums of Random Kitchen Sinks

NeurIPS '08

Jaeger & Haas:

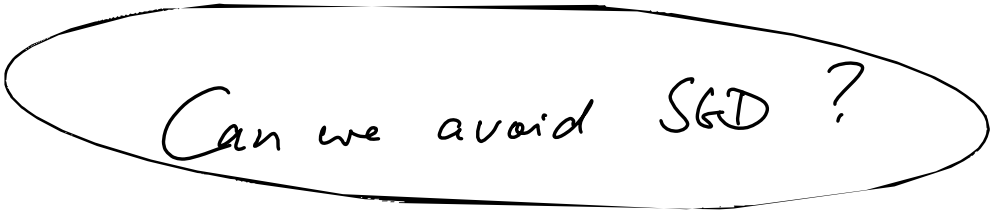
Harnessing Nonlinearity: Predicting Chaotic Systems and saving Energy in Wireless Communication

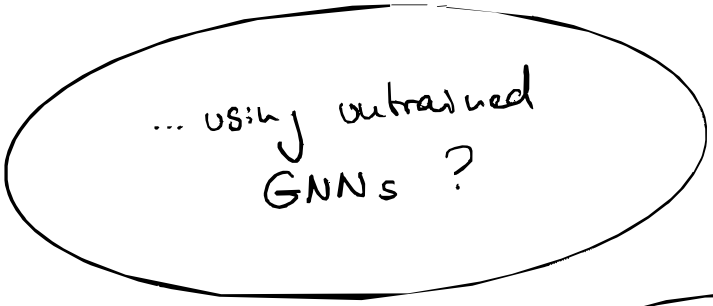
Science '04

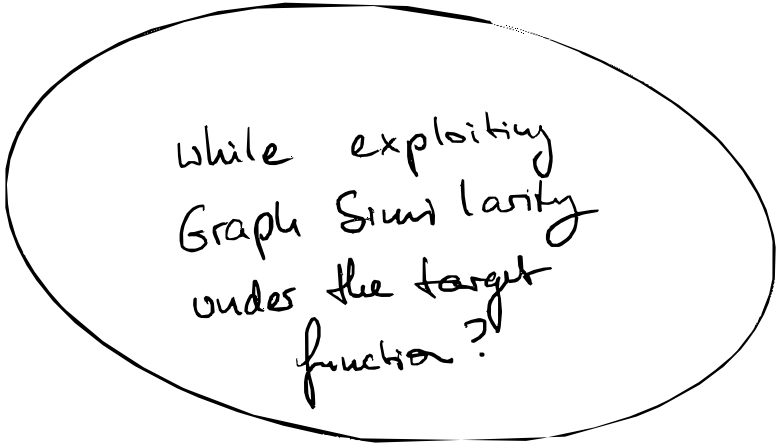
My Data Mining work:

Sampling a few graphs (trees, low treewidth graphs, ...) and using these as features in learning works well

Wale et al '14 - '23


Can we avoid SGD ?


... using untrained
GNNs ?


while exploiting
Graph Similarity
under the target
function ?

Can alignment between

the distance of representations

(1) obtained from randomly sampled GNNs

Can alignment between

the distance of representations

(1) obtained from randomly sampled GNNs

and

distance between target values y

(2) be used for learning?

The Algorithm (TM)

1. Sample a gazillion randomly initialized GNNs

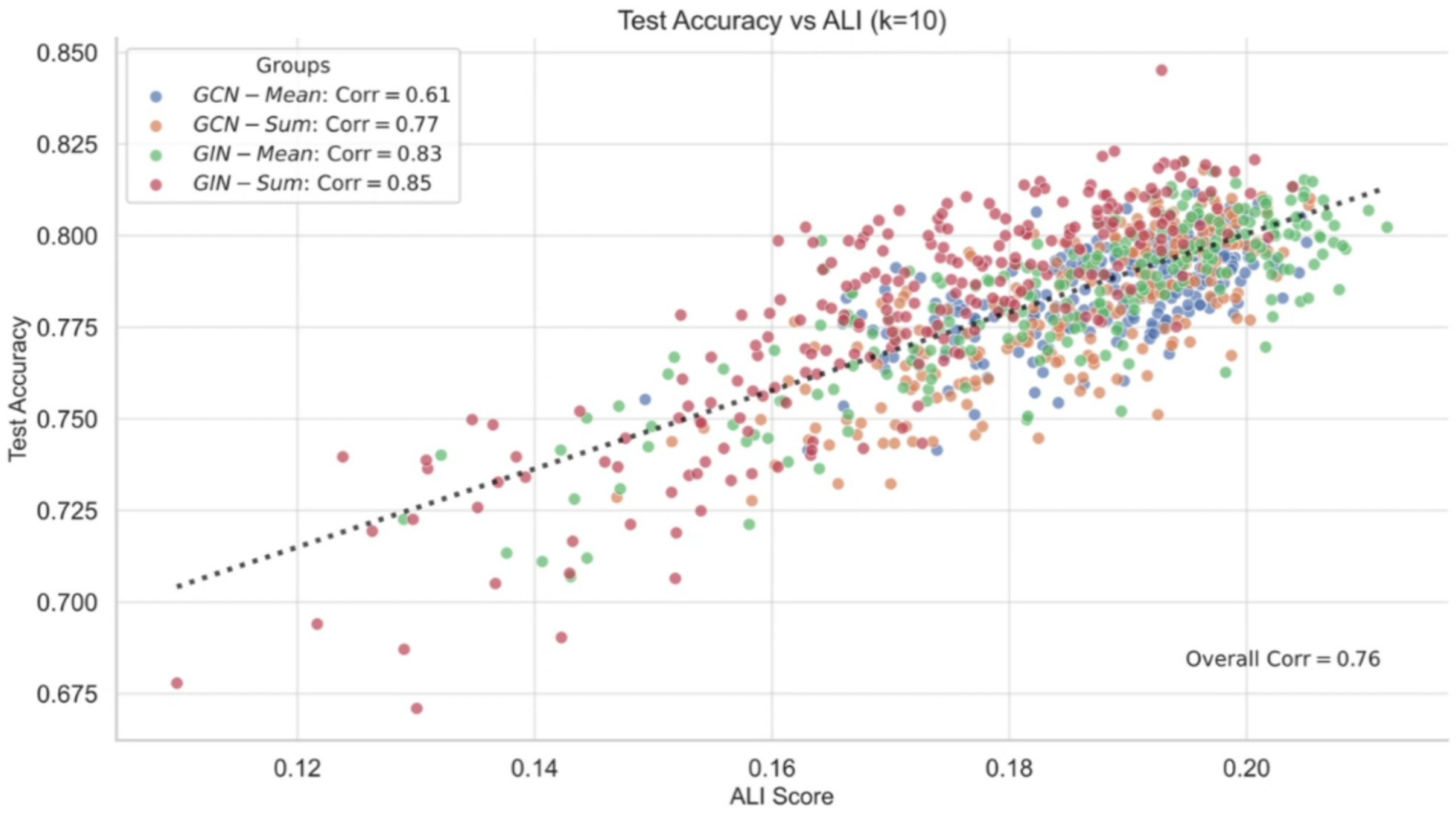
The Algorithm (TM)

1. Sample a gazillion randomly initialized GNNs
2. Return the GNN f_θ that maximizes the alignment $\mathrm{ALI}_K(f_\theta, d')$ on the training dataset $\mathcal{G}$

The Algorithm (Th)

1. Sample a gazillion randomly initialized GNNs
2. Return the GNN f_θ that maximizes the alignment $ALI_K(f_\theta, d')$ on the training dataset $\mathcal{G}$
3. Train a classifier on top of the representations given by f_θ

ALI-Accuracy Correlation



distance between targets

$$A(G, f_\theta, y) = \frac{1}{k} \sum_{H \in S_k(f_\theta, G)} d'(y(G), y(H))$$

$S_k(f_\theta, G)$: k Nearest Neighbors under f_θ

distance between targets

$$A(G, f_\theta, \mathcal{G}) = \frac{1}{k} \sum_{H \in S_k(f_\theta, G)} d'(y(G), y(H))$$

$S_k(f_\theta, G)$: k Nearest Neighbors under f_θ

$$B(G, f_\theta, \mathcal{G}) = \frac{1}{|\mathcal{G}| - k - 1} \sum_{H \notin S_k(f_\theta, G)} d'(y(G), y(H))$$

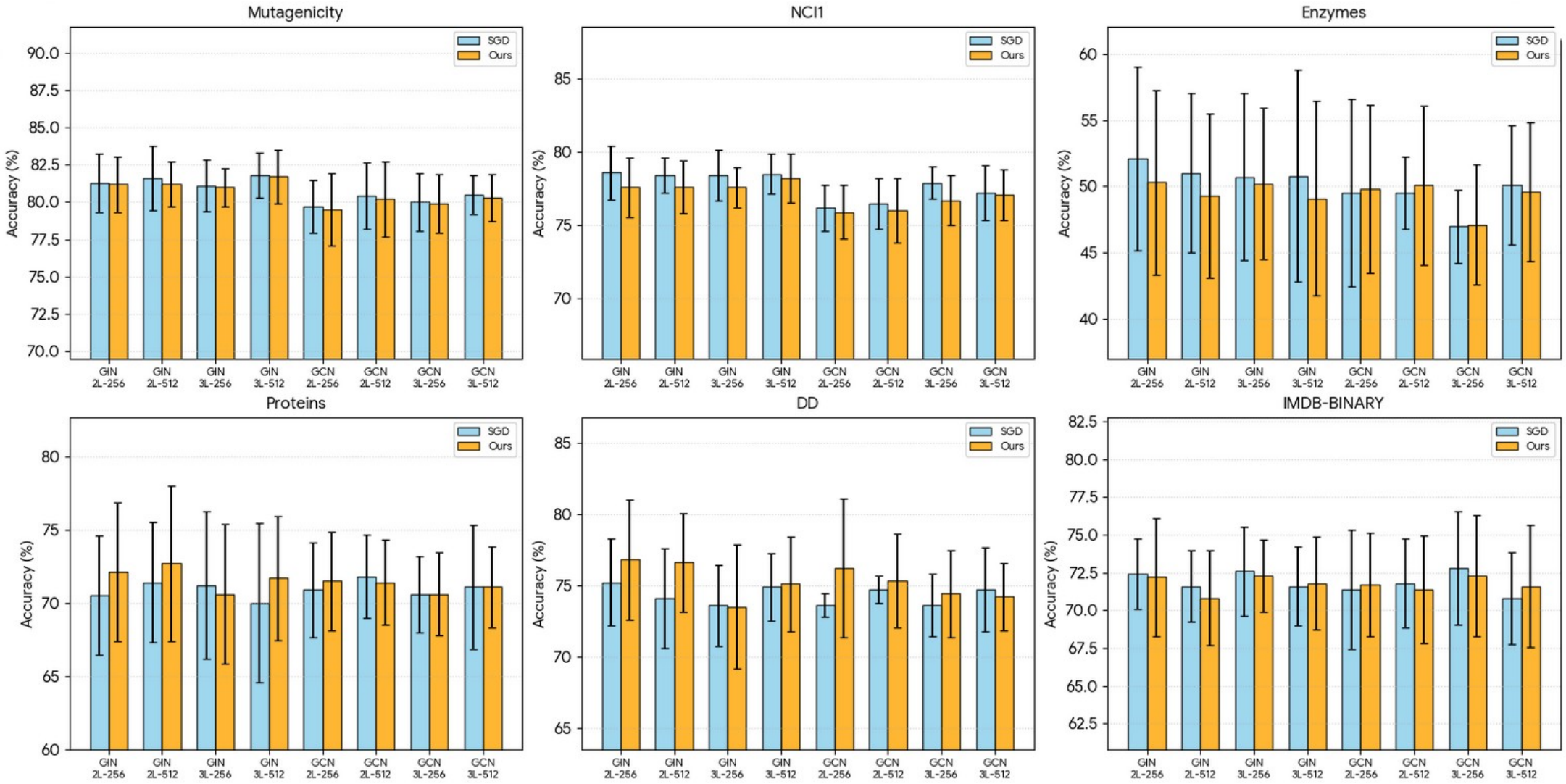
distance between targets

$$A(G, f_\theta, \mathcal{G}) = \frac{1}{k} \sum_{H \in S_k(f_\theta, G)} d'(y(G), y(H))$$

$S_k(f_\theta, G)$: k Nearest Neighbors under f_θ

$$B(G, f_\theta, \mathcal{G}) = \frac{1}{|\mathcal{G}| - k - 1} \sum_{H \notin S_k(f_\theta, G)} d'(y(G), y(H))$$

$$ALI_k(f_\theta, d') = \sum_{G \in \mathcal{G}} -A(G, f_\theta, \mathcal{G}) + B(G, f_\theta, \mathcal{G})$$


Mutagenicity
NCI1
Enzymes
Proteins
DD
IMDB-BINARY
SGD
Ours
Accuracy (%)
GIN 2L-256
GIN 2L-512
GIN 3L-256
GIN 3L-512
GCN 2L-256
GCN 2L-512
GCN 3L-256
GCN 3L-512

Conclusion

- Alignment is a good predictor of GNN performance
- Our sampling approach can achieve similar accuracy to SGD

Conclusion

- Alignment → a good predictor of GNN performance
- Our sampling approach can achieve similar accuracy to SGD

Limitations

- Struggles with deeper GNNs
- Runtime of alignment computation → quadratic in training set size

Conclusion

- Alignment → a good predictor of GNN performance
- Our sampling approach can achieve similar accuracy to SGD

Limitations

- Struggles with deeper GNNs
- Runtime of alignment computation → quadratic in training set size

Future Work

- Ensembles of sampled GNNs
- Evaluate on regression tasks + node level tasks
- Evaluate on larger datasets
- Robustness to noisy data?

Last page as a bonus

in case

it vanishes