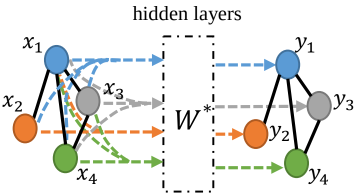
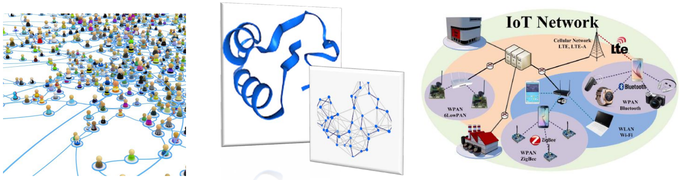
Motivation: Graph Structured Data

Graph neural networks $\Longrightarrow$ Graph structured data.

The output of each node depends on the input of the node and its neighbor nodes;





(a) Social Networks (b) Protein-Protein Interaction (PPI) Networks (c) Internet of Thing (IoT) Networks

Figure 1: Sampling applications in (a) social networks, (2) PPI Networks, and (3) IoT networks.

Motivation: Data & Computation Inefficiency of GNNs

- Sample complexity highly depends on the degree of the nodes/graph.
  – Sample complexity is proved to be a *quadratic* function of the degree of the graph.
- “Neighborhood explosion” during aggregation stages + High DNN computation.

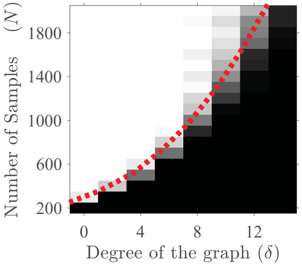


Figure 2: Phrase transition of number of samples against the degree of graph [Zhang et al. ICML’20]

Motivation: Data & Computation Inefficiency of GNNs

- Sample complexity highly depends on the degree of the nodes/graph.
  – Sample complexity is proved to be a *quadratic* function of the degree of the graph.
- "Neighborhood explosion" during aggregation stages + High DNN computation.
- The computational cost of a 2-layer GNN with $\sim$ 230 thousand nodes can be 2X as a 50-layer CNN with $\sim$ 14 million images.

Node: *232,965*
Average degree: *492*

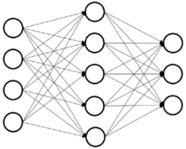
+

Parameter
size: 18K

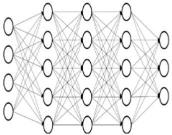
Imagenet:
14,197,122

+

Parameter
size: 25.6M

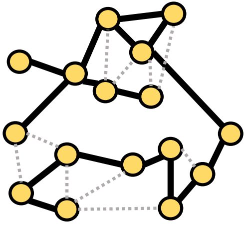


$>$ **2x**

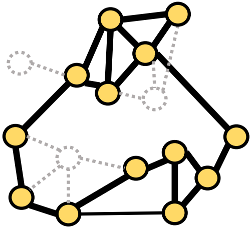


Background: Graph Topology Sampling

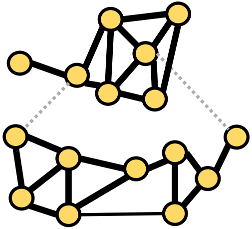
- Graph topology sampling: edge sampling, node sampling, sub-graph clustering.
- Why sampling? To reduce sample complexity & memory costs.



GraphSage [Hamilton et al.17]



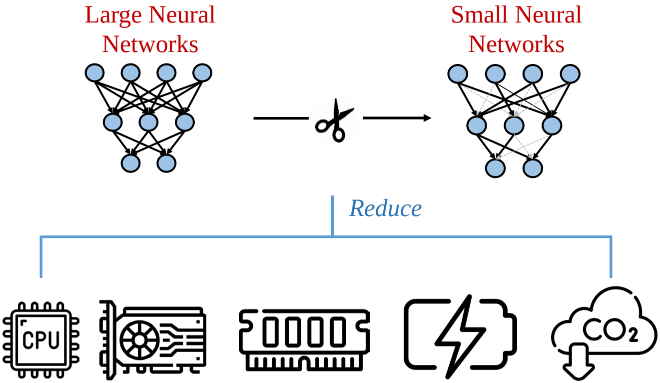
FastGCN [Chen et al.18]



Cluster-GCN [Chiang et al.19]

Background: Neural Network Pruning

- Remove (unnecessary) parameters of the neural networks.
- Reduce compute cost, memory cost, energy consumption, and carbon footprint.



Background: Pruning in Neural Networks

Sparse neural networks:

- 90% of the parameters can be pruned.
- Reduce computational cost by $5\times$.

Table 1: Network pruning makes neural networks sparse. Source from Han et al.15

Neural Network	# Parameters			MACs
	Before Pruning	After Pruning	Reduction	Reduction
Alexnet	61M	6.7M	**9** X	**3** X
VGG-16	138M	10.3M	**12** X	**5** X
GoogleNet	7M	2.0M	**3.5** X	**5** X
ResNet50	26M	7.47M	**3.4** X	**6.3** X
SqueezeNet	1M	0.38M	**3.2** X	**3.5** X

In addition, a good pruned neural network:

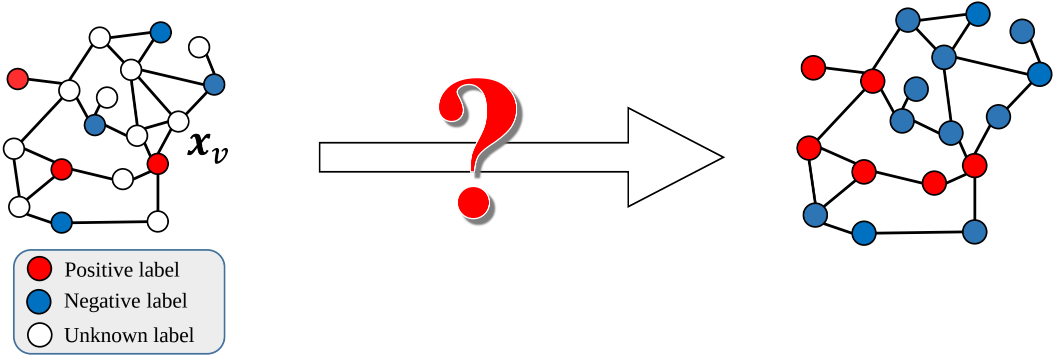
- Improved test accuracy
- Faster convergence rate

Table 2: Improved test accuracy of training pruned network. Source: Adapted from , [Chen et al.20], [Chen et al.22].

Neural Network	Dataset	Accuracy (%)	
		Before Pruning	After Pruning
LetNet-5	MINST	98.05	**98.41**
Conv-6	Cifar-10	77.52	**80.02**
ResNet-50	Cifar-10	94.31	**94.82**
ResGCN-28	Cora	80.02	**81.88**
BERT	MNLI	82.39	**83.08**

Problem Formulation: Node Classification

- Node feature $\boldsymbol{x}_v \in \mathbb{R}^d$ & Node label $y_v \in \{+1, -1\}$.
- Given partial labels of $\{y_v\}_{v \in \mathcal{D}}$ and all input feature $\{\boldsymbol{x}_v\}_{v \in \mathcal{V}}$, the goal is to predict the labels for all nodes $v \in \mathcal{V}/\mathcal{D}$.
  – $\mathcal{D}$ is a subset of nodes set $\mathcal{V}$.



Algorithm: Joint Topology-Model Sparsification

1. **(Initialization.)** Initialize $\boldsymbol{w}_k$ as random Gaussian and b_k uniformly from $\{+1, -1\}$.

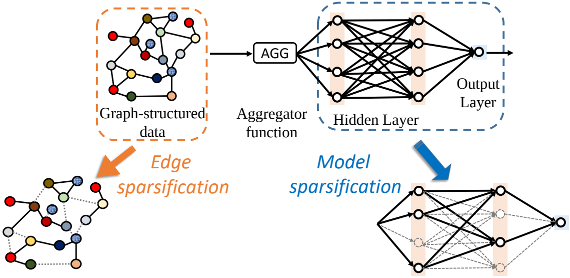


Figure 2: An illustration of the joint topology-model sparsification.

Algorithm: Joint Topology-Model Sparsification

1. (**Initialization.**) Initialize $\boldsymbol{w}_k$ as random Gaussian and b_k uniformly from $\{+1, -1\}$.
2. (**Edge sampling.**) For each node, aggregate a subset of neighbor nodes via (randomly) edge sampling.

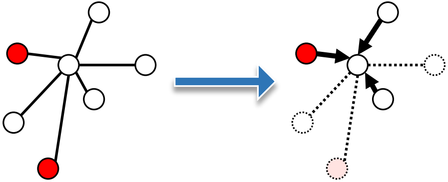


Figure 2: The illustration of edge sampling

Algorithm: Joint Topology-Model Sparsification

1. (**Initialization.**) Initialize $\boldsymbol{w}_k$ as random Gaussian and b_k uniformly from $\{+1, -1\}$.
2. (**Edge sampling.**) For each node, aggregate a subset of neighbor nodes via (randomly) edge sampling.
3. (**Pre-training.**) Update $\boldsymbol{w}_k$ through gradient descent algorithm based on the sub-graph.
4. (**Pruning.**) Pruning β fraction of neurons with the smallest magnitude.
5. (**Re-training.**) Update $\boldsymbol{w}_k$ through gradient descent algorithm.

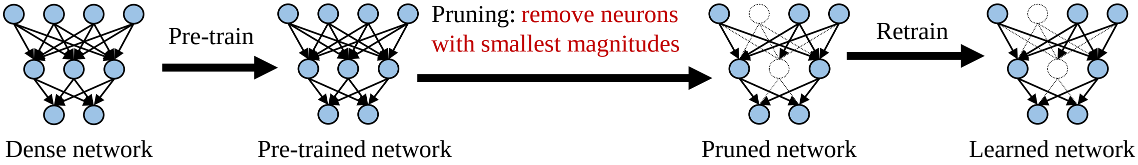


Figure 2: The illustration of magnitude-based neuron pruning

Related Works: graph sampling & pruned network

Only separate theoretical explanations for *either* graph sampling *or* network pruning.

- Pruned neural networks are *slightly worse* than the original dense network in terms of the expressive power and training accuracy [Arora et al.18, Baykal et al.18, Ben et al.20, Malach et al.20].
- [Zhang et.al NeurIPS'21] characterizes the benefits of training "winning tickets" but in *feedforward neural networks* and *cannot explain how to find* "winning ticket".
- Focus on the expressive power of sampled graphs [Hamilton et al.17; Cong et al.21; Chen et al.18; Zou et al.19].
- [Li et al.22] shows improved generalization using graph sampling, but assuming *the adjacency matrices of the sampled and original graph are similar*.

Related Works: graph sampling & pruned network

Only separate theoretical explanations for *either* graph sampling *or* network pruning.

- Pruned neural networks are *slightly worse* than the original dense network in terms of the expressive power and training accuracy [Arora et al.18, Baykal et al.18, Ben et al.20, Malach et al.20].
- [Zhang et.al NeurIPS'21] characterizes the benefits of training "winning tickets" but in *feedforward neural networks* and *cannot explain how to find* "winning ticket".
- Focus on the expressive power of sampled graphs [Hamilton et al.17; Cong et al.21; Chen et al.18; Zou et al.19].
- [Li et al.22] shows improved generalization using graph sampling, but assuming *the adjacency matrices of the sampled and original graph are similar*.

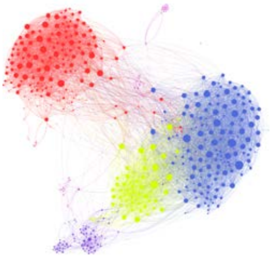
No theoretical guarantees for the joint model-topology sparsification.

Takeaways of Theoretical Findings

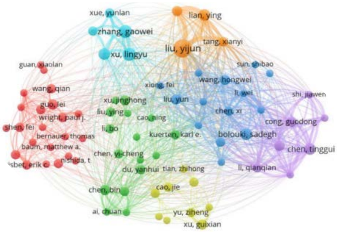
1. Edge sampling reduces the sample complexity.
2. Magnitude-based neuron pruning reduces the sample complexity and accelerates the convergence rate.
3. Edge and model sparsification is a *win-win* strategy.

Assumptions: Data Model

1. Nodes connected to each other tend to have the same label.
2. Some nodes have a stronger influence than the other nodes.
   – *Important nodes* v.s. *Unimportant nodes*



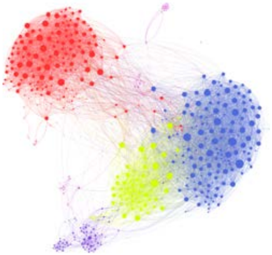
Social Network



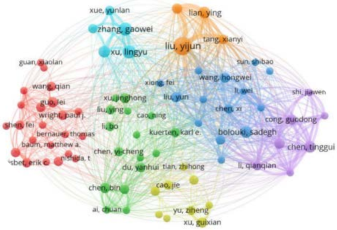
Citation Network

Assumptions: Data Model

1. Nodes connected to each other tend to have the same label.
2. Some nodes have a stronger influence than the other nodes.
   – *Important nodes* v.s. *Unimportant nodes*



Social Network



Citation Network

Influence of Edge Sampling

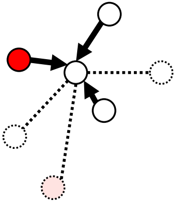
- (Pro.) Sample complexity is a quadratic function of the node degree, indicating that edge sampling reduces the sample complexity.

Influence of Edge Sampling

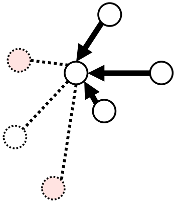
- (Pro.) Sample complexity is a quadratic function of the node degree, indicating that edge sampling reduces the sample complexity.
  – Sample complexity $N = \Theta(r^2)$. (r: number of sampled neighbors for each node)

Influence of Edge Sampling

- (Pro.) Sample complexity is a quadratic function of the node degree, indicating that edge sampling reduces the sample complexity.
  – Sample complexity $N = \Theta(r^2)$. (r: number of sampled neighbors for each node)
- (Con.) Edge sampling leads to possible labeling information lost, indicating an increased sample complexity and iteration number for convergence.



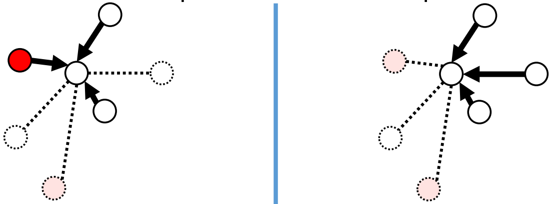
Important node (red node) is sampled



Important node is NOT sampled

Influence of Edge Sampling

- (Pro.) Sample complexity is a quadratic function of the node degree, indicating that edge sampling reduces the sample complexity.
  – Sample complexity $N = \Theta(r^2)$. (r: number of sampled neighbors for each node)
- (Con.) Edge sampling leads to possible labeling information lost, indicating an increased sample complexity and iteration number for convergence.
  – Sample complexity $N = \Theta(\alpha^{-2})$, Number of iterations $T = \Theta(\alpha^{-1})$.
  – α: average rate of at least one important node is sampled.



Important node (red node) is sampled 

Important node is NOT sampled 

Influence of Edge Sampling

Uniform sampling can save sample complexity by a faction of $1/c$.

- c is the average number of important nodes (red nodes in Figure 3) in the neighbor.

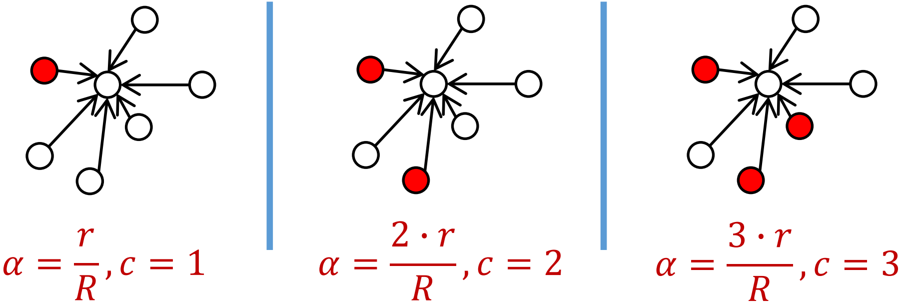


Figure 3: Illustration of different α and c in different graphs

Benefits from Magnitude-based Model Pruning

Two types of neuron weights:

- "Good" neuron:
  – small angle $\longrightarrow$ learns features of important nodes (class-relevant features).
  – have large magnitudes.
- "Bad" neuron:
  – large angle $\longrightarrow$ learns features of unimportant nodes (class-irrelevant features).
  – have small magnitudes.

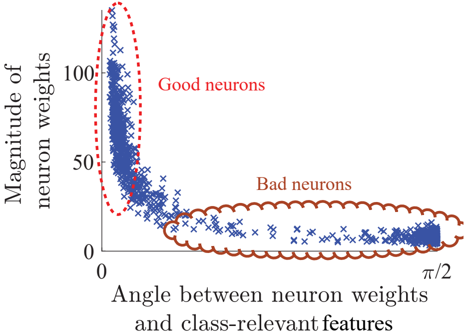


Figure 4: Distribution of the neuron weights.

Benefits from Magnitude-based Model Pruning

Two types of neuron weights:

- "Good" neuron:
  – small angle $\longrightarrow$ learns features of important nodes (class-relevant features).
  – have large magnitudes.
- "Bad" neuron:
  – large angle $\longrightarrow$ learns features of unimportant nodes (class-irrelevant features).
  – have small magnitudes.

Proposition 1

For a "good" neuron with weights $\boldsymbol{W}_1$ and "bad" neuron with weights $\boldsymbol{W}_2$, we have

$$\|\boldsymbol{W}_1\| - \|\boldsymbol{W}_2\| \geq 1 - \Theta(1/\sqrt{N}).$$

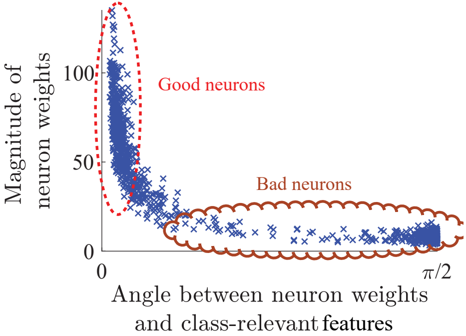


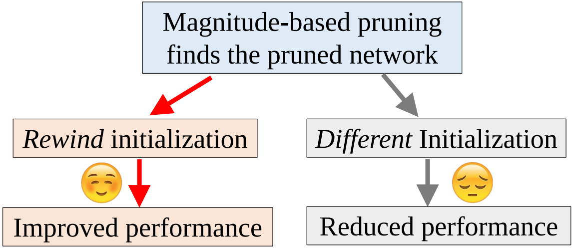
Figure 4: Distribution of the neuron weights.

Benefits from Model Pruning

The initial weights determine whether a neuron is “good” or ”bad”.

Proposition 2

A “good” neuron weight at initialization is still “good” at next iterations.

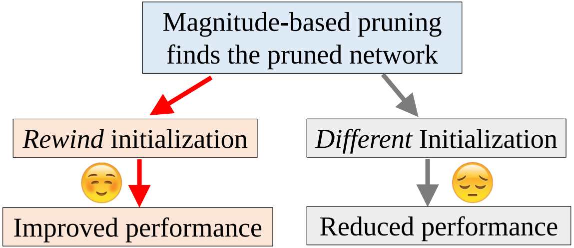


Benefits from Model Pruning

The initial weights determine whether a neuron is “good” or ”bad”.

Proposition 2

A “good” neuron weight at initialization is still “good” at next iterations.



A numerical justification on a shallow neural network.

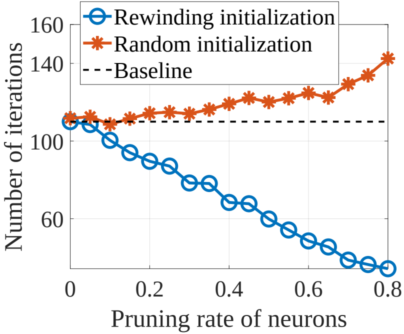


Figure 5: Number of iterations against the pruning rate.

Numerical Justification

- Our joint topology-model sparsification significantly improves test accuracy over random pruning.
- Save the computational cost by up to 18×.

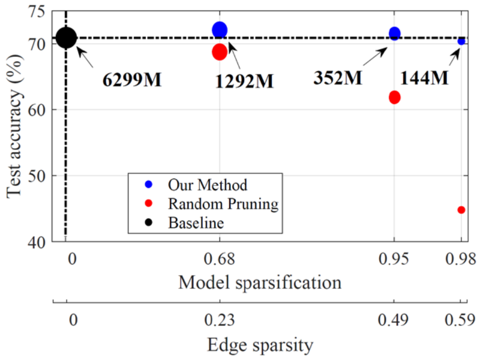


Figure 6: Performance at different model and edge sparsity on Citeseer node classification.

Proof Sketchy

1. A sufficient large fraction of neurons is “good neuron”.

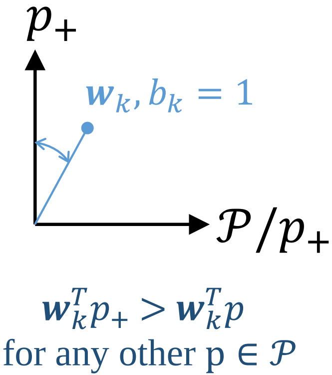


Figure 7: Example of a “good” neuron

Proof Sketchy

1. A sufficient large fraction of neurons is "good neuron".
2. "Good neurons": increase along the direction of class-relevant features; zigzag in other directions.

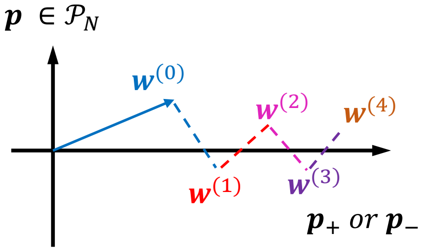


Figure 7: Illustration of iterations of "good" neurons

Proof Sketchy

1. A sufficient large fraction of neurons is "good neuron".
2. "Good neurons": increase along the direction of class-relevant features; zigzag in other directions.
3. "Bad neuron": increase slowly along any direction with a sufficiently large number of samples.

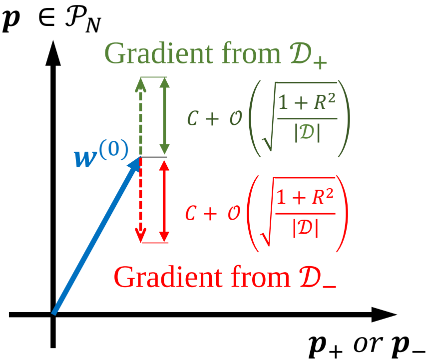


Figure 7: Illustration of iterations of "bad" neurons

Proof Sketchy

1. A sufficient large fraction of neurons is "good neuron".
2. "Good neurons": increase along the direction of class-relevant features; zigzag in other directions.
3. "Bad neuron": increase slowly along any direction with a sufficiently large number of samples.
4. With a sufficiently large number of iterations, the output of the graph neural network is determined by the "good neurons" (and the class-relevant features).

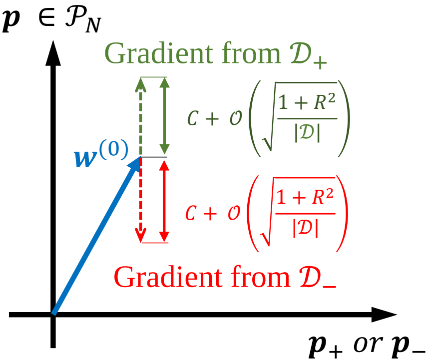


Figure 7: Illustration of iterations of "bad" neurons