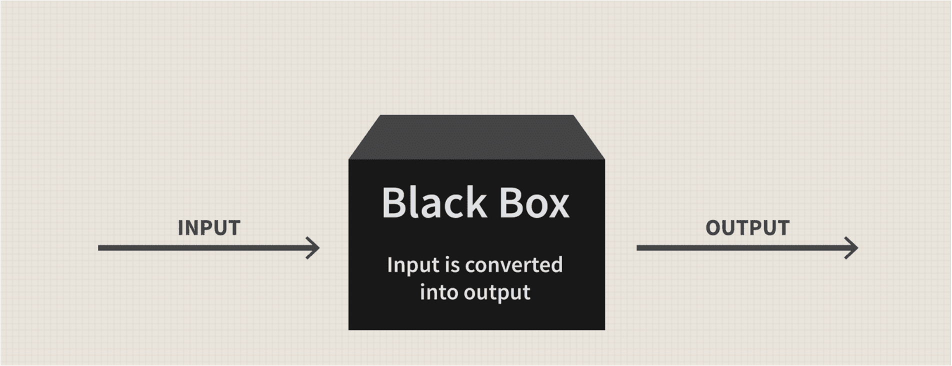
Scalable GNN Explanation with Shapley Values

Ariful Azad
Texas A&M University
ariful@tamu.edu

Applied and Computational Discrete Algorithms in AI and Data Science

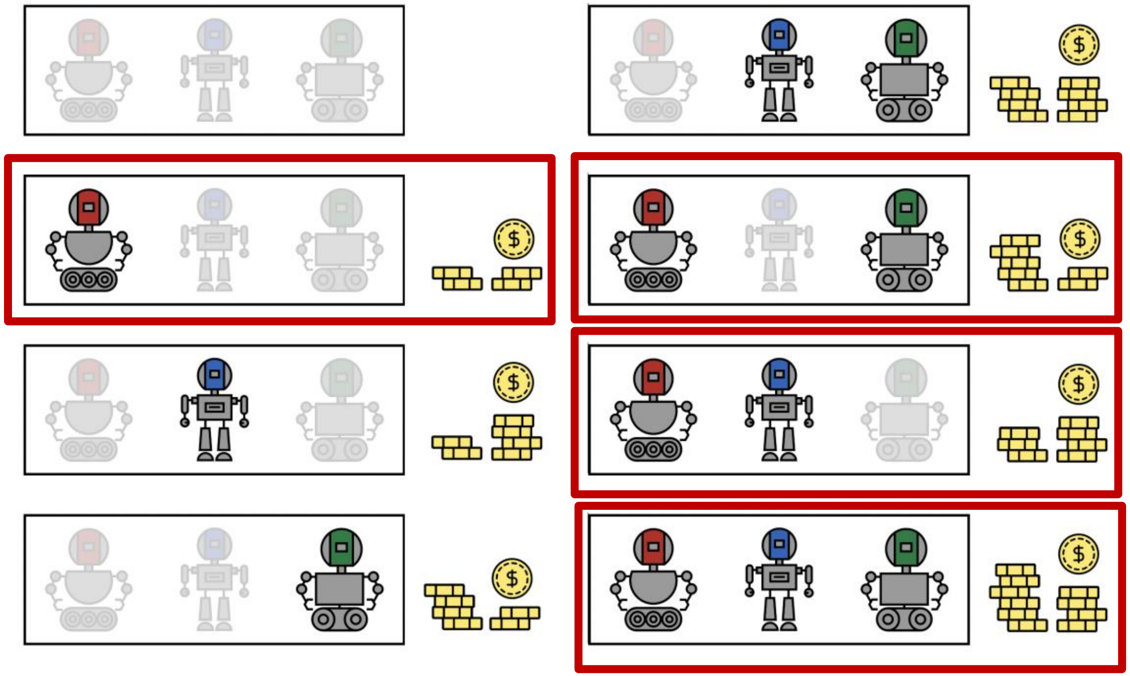
AI Explainability

- Modern ML models are accurate but opaque
- **Explainability**: Generally, post-hoc property
  - Explainability refers to methods that provide **explanations of a model's predictions**, often **after the model is trained**.
  - Which inputs mattered most for this prediction
- **Interpretability**: Generally, model-intrinsic property
  - A model is *interpretable* if we can directly understand **how it works** and **how inputs map to outputs (e.g., Random Forest)**



Shapley Value

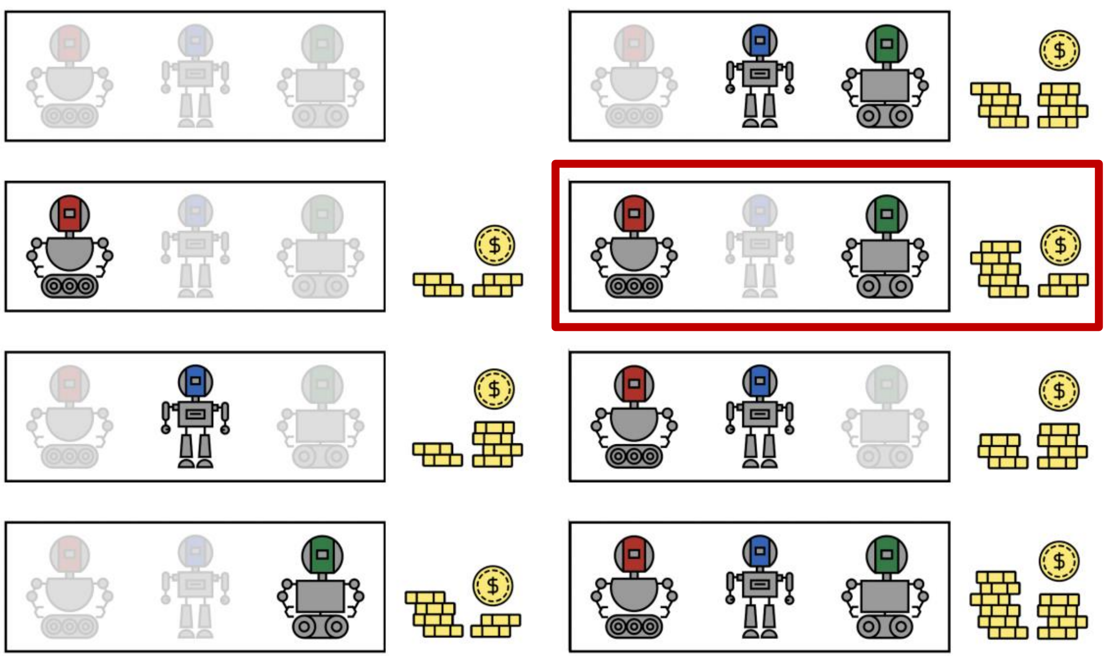
- Determines the contribution of each player in **collaborative** work
- What is the contribution of the red player?
  - **Consider 2^{N-1} coalitions excluding the red player (N: number of players)**
  - The contribution of a player is the sum of its **additional contributions** to all other teams



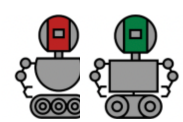
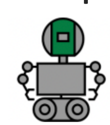
3 player:
2^3 = 8 coalitions

Shapley Value

- What is the contribution of the red player?
  - The sum of its **additional contributions** to all other teams
- Let F(S): Total contribution of S, a coalition/team of a set of players



Contribution of the red player

F() – F ()

Shapley Value

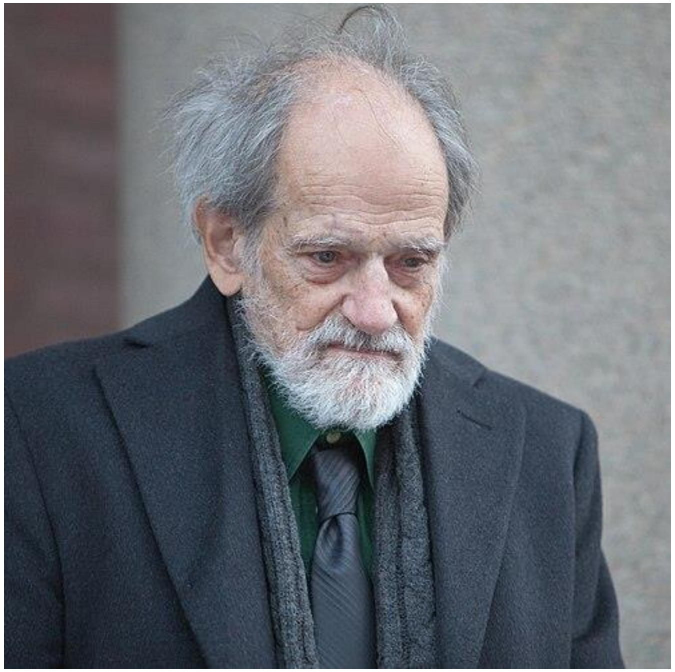
N: number of players
S: a coalition
F(S): Total contribution of a coalition
ϕ_{ni}: The contribution of the ith player to all coalition

$$\phi_{n_i} = \sum_{S \subseteq \{1,2,..,N\} \setminus \{i\}} \underbrace{\frac{|S|!(N-|S|-1)!}{N!}}_{\text{Weights}} \underbrace{(F(S \cup \{n_i\}) - F(S))}_{\text{Marginal contribution}}$$

For example, in a team project where each member contributed differently,

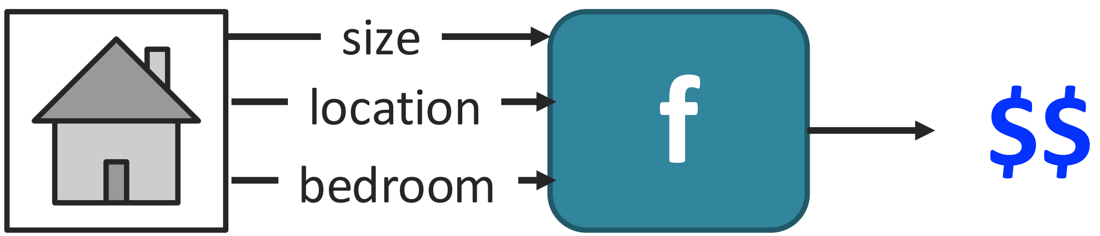
- The Shapley value provides a way to determine **how much credit or blame** each member deserves.

❑ Nobel Prize in Economics, 2012



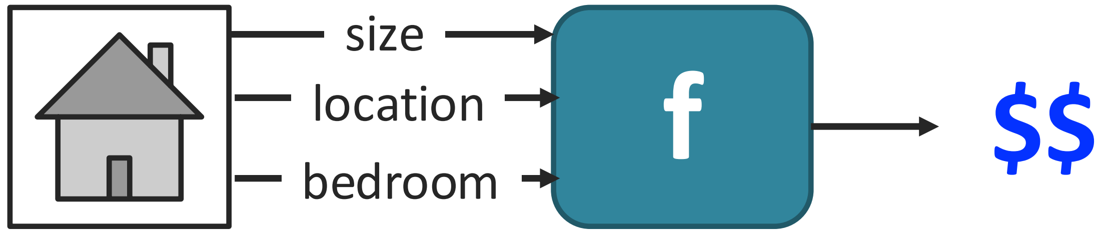
Shapley Value in Machine Learning

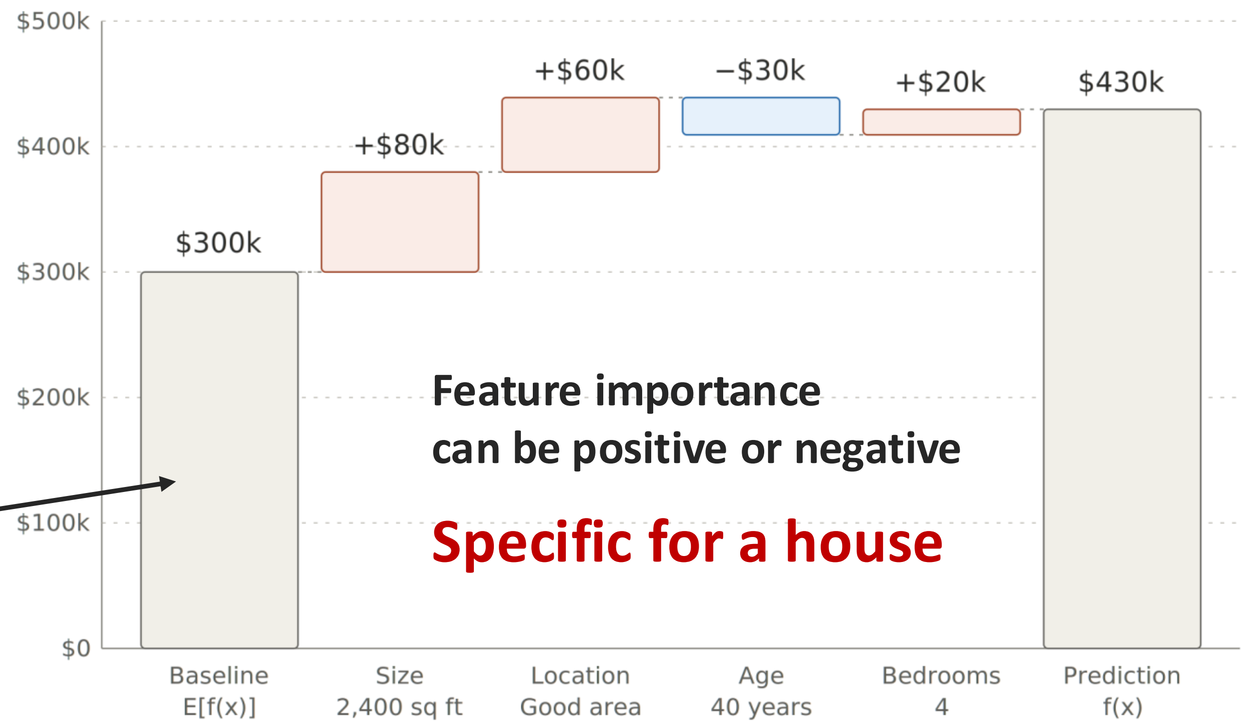
- Consider a model f designed to predict the price of a house
- **f(size, location, age, bedroom….) = price**
- The model is trained.



- Selling agent: What should be the price of a house that they are putting to the market? **The model can give a price.**
- The buyer: What aspects of the house contribute to the price? **Here comes Shapley values.**

Shapley Value in Machine Learning





Feature importance can be positive or negative

Specific for a house

Baseline: The model's average prediction across the training data

Shapley Value in Machine Learning

N: number of players
S: a coalition
F(S): Total contribution of a team
ϕ_i: The contribution of the ith player to all coalitions

$$\phi_{n_i} = \sum_{S \subseteq \{1,2,..,N\} \setminus \{i\}} \underbrace{\frac{|S|!(N - |S| - 1)!}{N!}}_{\text{Weights}} \underbrace{(F(S \cup \{n_i\}) - F(S))}_{\text{Marginal contribution}}$$

N: model parameters/data features influencing the model prediction
S: a coalition of a subset players
F: An ML model (e.g., CNN, GNN, regression, etc.)
F(S): Prediction of the model with S
ϕ_i: The contribution of the ith parameter or feature to model prediction

- The value is between: [-1,1]. Measures how influential a parameter or feature is to the prediction of the model.
- Treat the model as a black box. It does not care how accurate the model is.

Shapley Value in Machine Learning

N: number of players
S: a coalition
F(S): Total contribution of a team
ϕ_i: The contribution of the ith player to all coalitions

$$\phi_{n_i} = \sum_{S \subseteq \{1,2,..,N\} \setminus \{i\}} \underbrace{\frac{|S|!(N-|S|-1)!}{N!}}_{\text{Weights}} \underbrace{(F(S \cup \{n_i\}) - F(S))}_{\text{Marginal contribution}}$$

Computation Complexity: **2^N inference with S parameters**

Example: 20 parameters => $N=2^{20}$: 1,048,576 coalitions

Impractical for most ML models.

Solution: Sampling or surrogate model to approximate

SHAP (Lundberg et al. 2017): 50K citations

- Sample coalitions: e.g., 2000 samples << 2^{20}
- Linear surrogate model

SHAP: Approximate Shapley Values

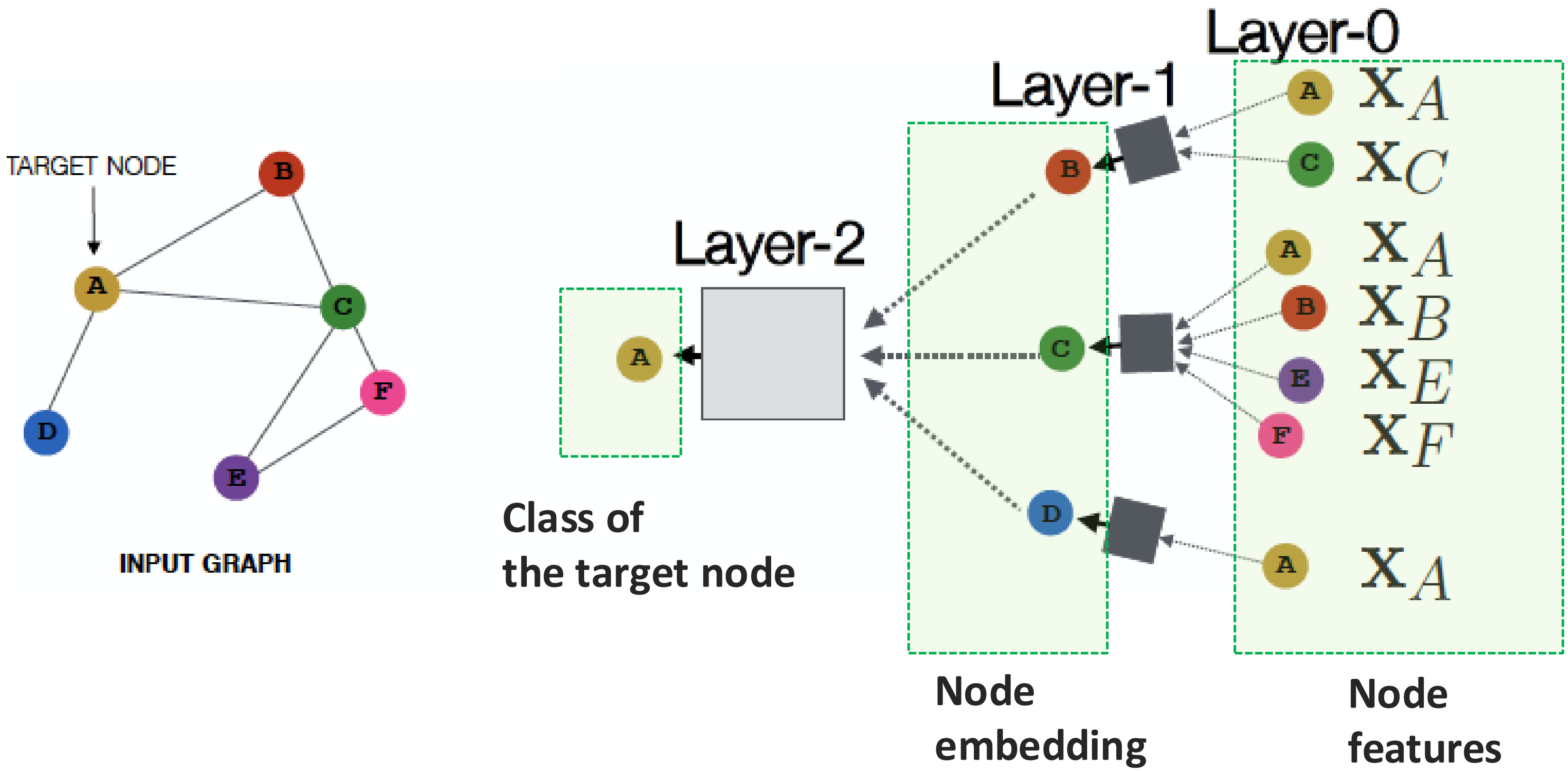
- **Key idea:** approximate Shapley values via **weighted linear regression** on sampled coalitions of features.
- Treats the model as a *black box*.
  - Randomly sample subsets $S \subseteq \{1,2,..N\}$.
  - Compute model outputs f(S) with masked/absent features replaced by background values.
  - Fit a **locally weighted linear model**: $g(S) = \phi_0 + \sum_{i=1}^{N} \phi_i m_i$

    where ($m_i = 1$) if feature i in S , else 0.
  - Solve for ϕ_i via weighted least squares.

  Solution:

$$\phi = (M^T W M)^{-1} M^T W \hat{y}$$

Graph Neural Networks



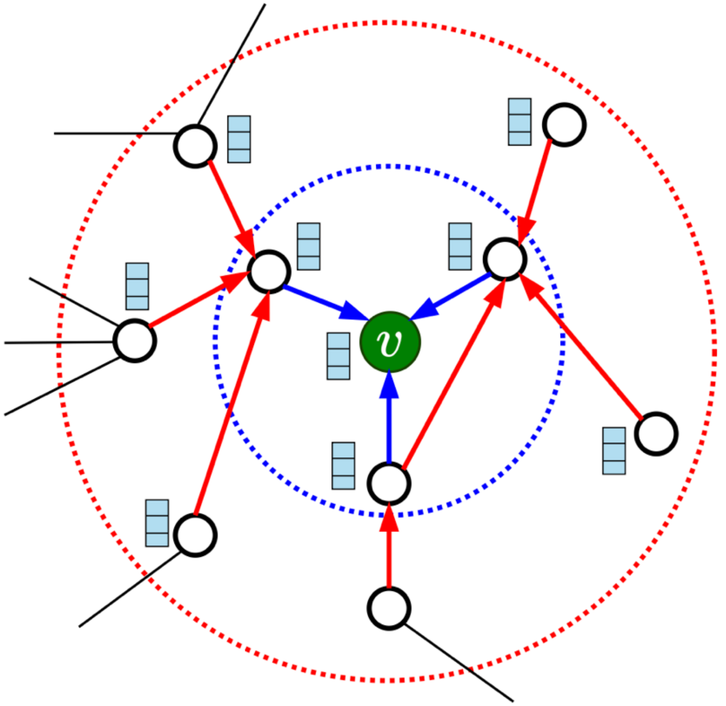
Inference time: GNN(v) = class of v

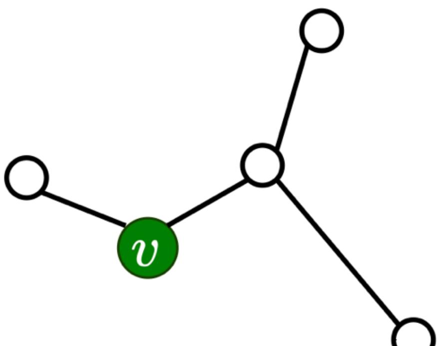
Explanations: features and edges important for this prediction

Photo credit: Jure Leskovec's lecture

GNN Explanation Goals

- Consider a **pretrained** 2-layer GNN (GCN, GAT, etc) for the node-classification task.
- v's prediction is influenced by 2-hop nodes and edges. Called **the computational graph**.
- Simplest goal: which edges influence v's prediction the most.

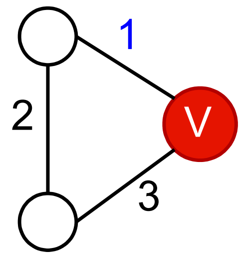


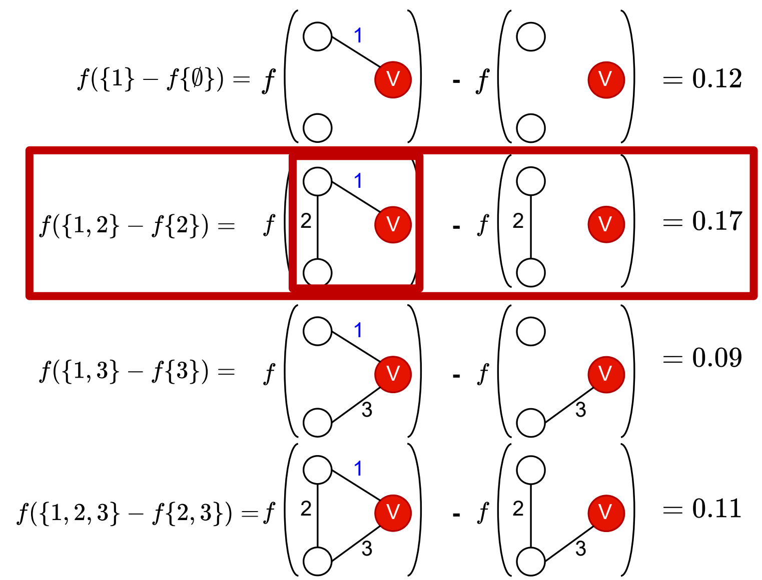


An explanation

GNN Explanation With Shapley

- **Players: Edges** {1, 2, 3}
  - explain v, find importance of edge 1
- **Coalition:** {1, 2} (A subgraph)
- **Marginal contribution** of 1: f({1, 2}) – f({2}) = 0.17
- **2^{N-1} marginal contributions (2^2=4 in the example)**
- **Weighted average**:
  - More weights: small and large coalitions
  - $\phi_1 = \frac{1}{3} * 0.12 + \frac{1}{6} * 0.17 + \frac{1}{6} * 0.09 + \frac{1}{3} * 0.11 = 0.12$



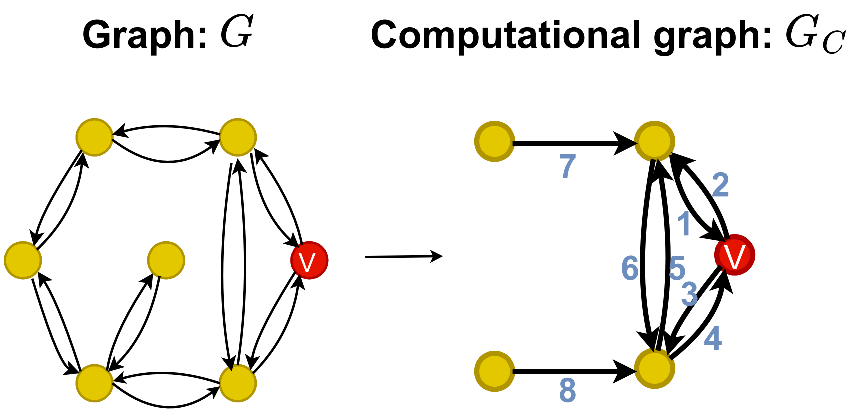


$$\phi_{n_i} = \sum_{S \subseteq \{1,2,..,N\} \setminus \{i\}} \underbrace{\frac{|S|!(N-|S|-1)!}{N!}}_{\text{Weights}} \underbrace{(F(S \cup \{n_i\}) - F(S))}_{\text{Marginal contribution}}$$

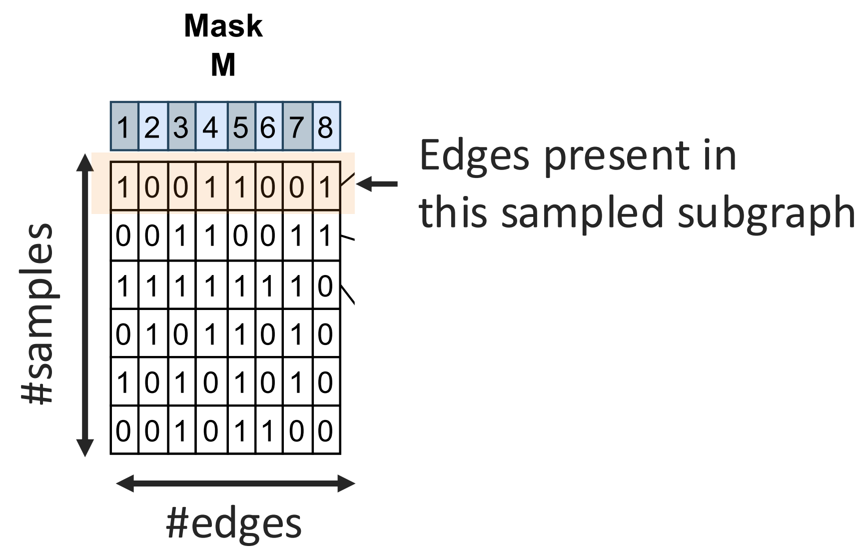
Similarly for other edges!

GNNShap: Shapley Based GNN Explanation
(Akkas and Azad, the Web Conference, 2024)

Step 1: Find players: 2-hop computational graph



Step 2: Sampling via mask matrix



- ✓ Other edges are not used in inference

So, their importance is 0 and removed from consideration

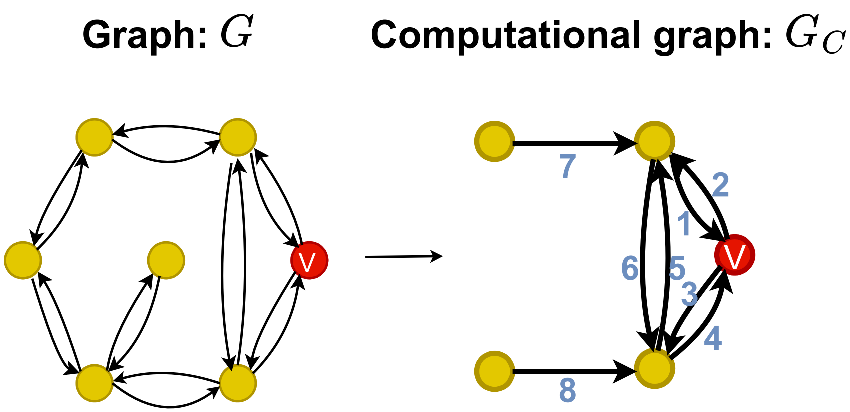
- ✓ **Exact calculation require GNN inference with 2^7 subgraphs**

Need to sample from the subgraph space

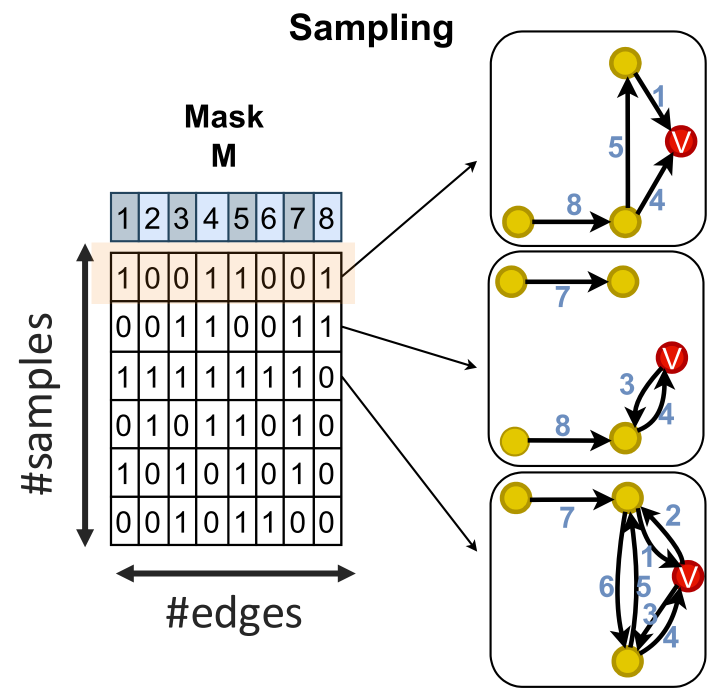
GNNShap: Shapley Based GNN Explanation

(Akkas and Azad, the Web Conference, 2024)

Step 1: Find players: 2-hop computational graph



Step 2: Sampling via mask matrix

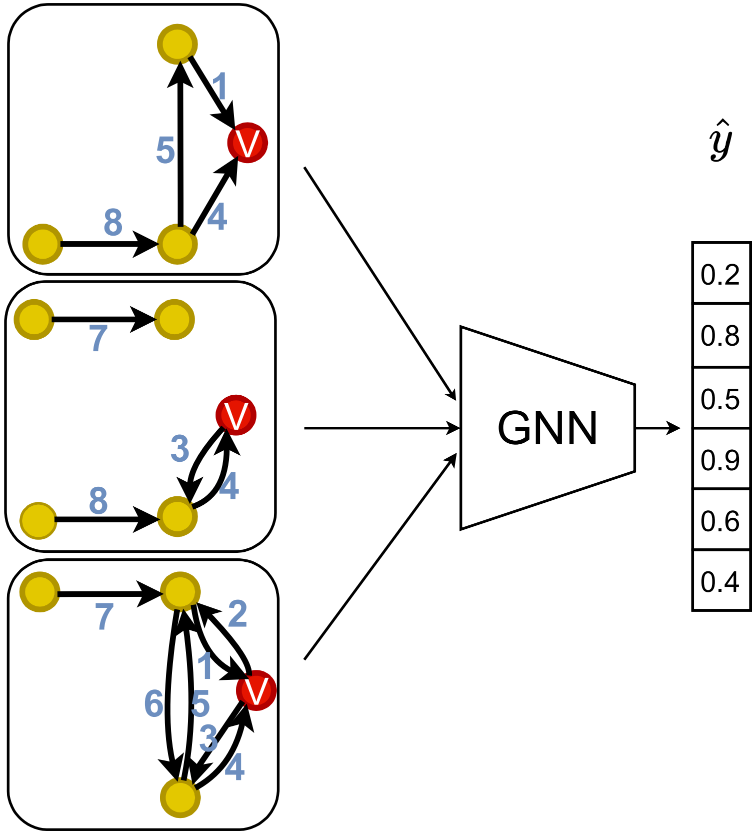


GNNShap: Shapley Based GNN Explanation

(Akkas and Azad, the Web Conference, 2024)

Step 3: Model predictions

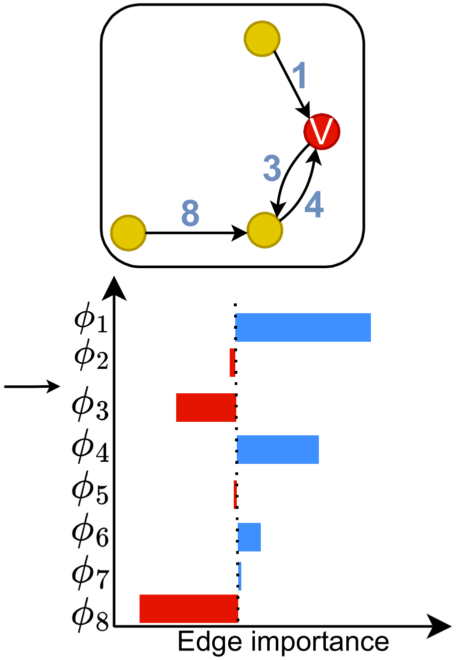
Model predictions



Step 4: Shapley computation

Shapley Computation

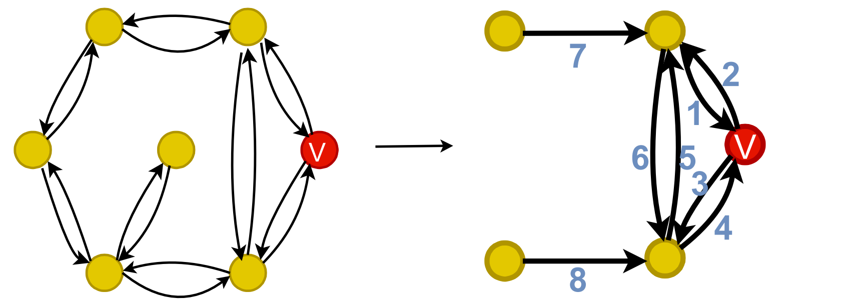
$$\phi = (M^T W M)^{-1} M^T W \hat{y}$$



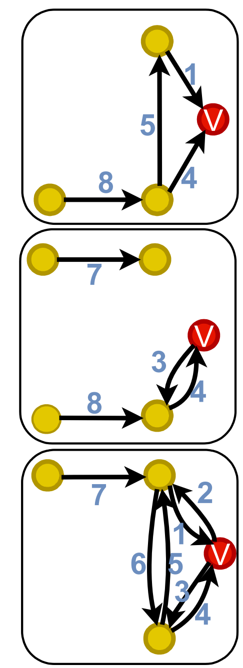
How to sample from the large space?

- Example: 8 edges to consider.
- Exact calculation require GNN inference with 2^7 subgraphs
- Need to sample from the subgraph space
- How to sample from the subgraph space?

Graph: G **Computational graph:** G_C

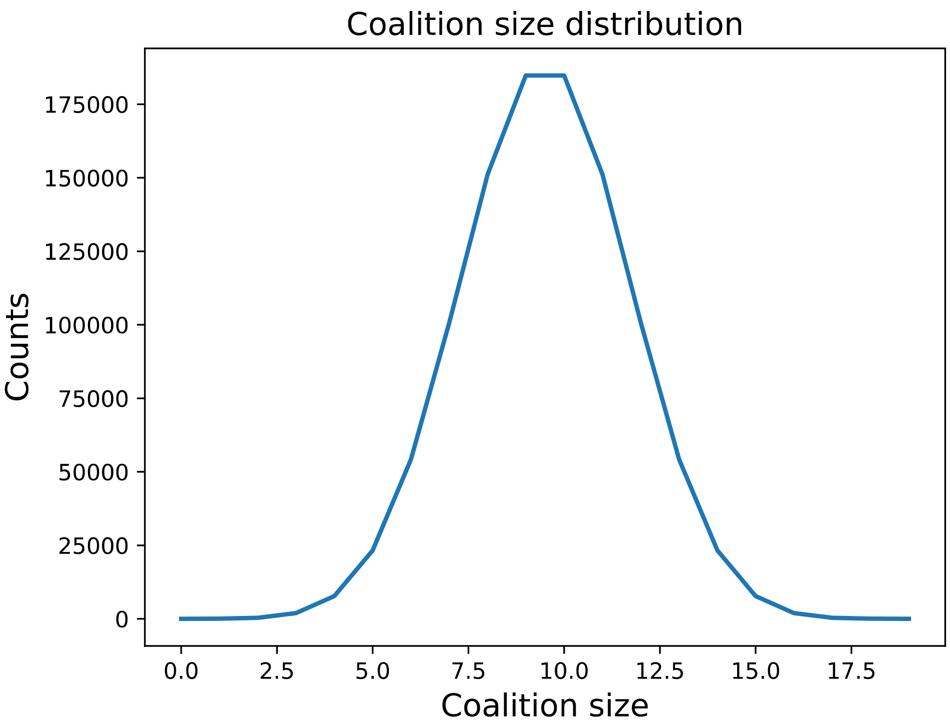


Samples (subgraphs)



How to sample from the large space?

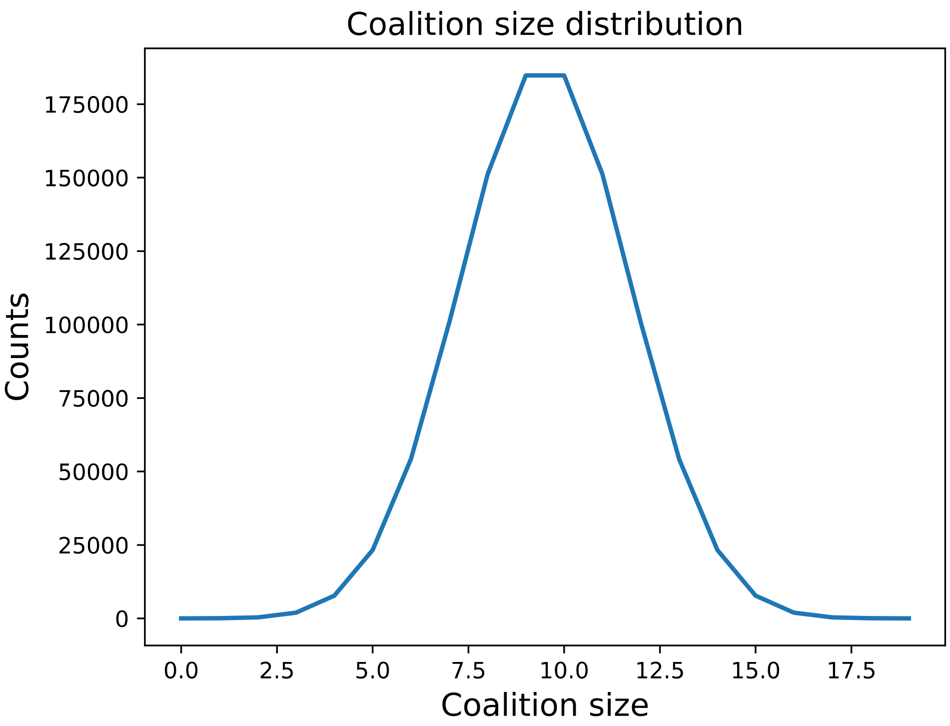
- Sampling space
  - A few small and large subgraphs
  - A lot of mid-size subgraphs
- How should we pick subgraphs from this space?



Number of coalitions for 20 players (subgraphs of given sizes)

How to sample from the large space?

- ❑ Uniform sampling?
  - – Sample more from mid-sized coalitions
- ❑ However, amall and large samples (subgraphs) are more informative

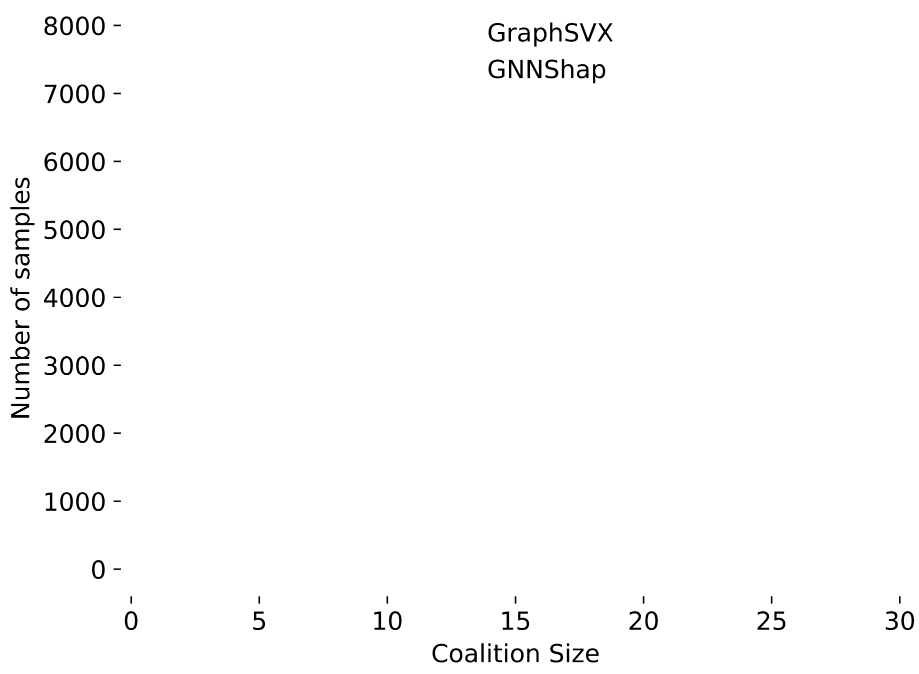


Number of coalitions for 20 players (subgraphs of given sizes)

How to Samples from the large space?

- Small and large samples (subgraphs) are more informative
- **GraphSVX** (Duval et al. 2021): only sample small and large subgraphs GraphSVX
- **GNNShap**: Sample from everywhere, but more from small and large subgraphs

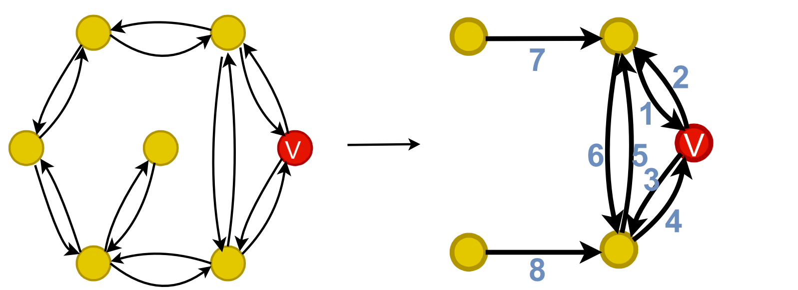
Sample distribution for 30 players



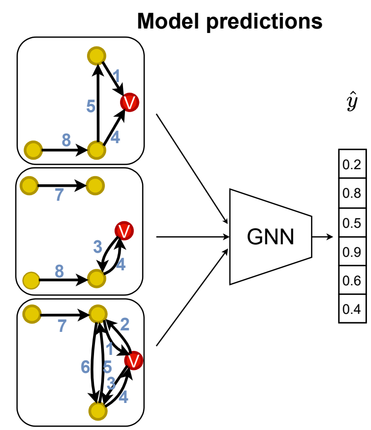
GNNShap: The overall algorithm
(Akkas and Azad, WWW 2024)

Step 1: Find players: l-hop subgraph graph

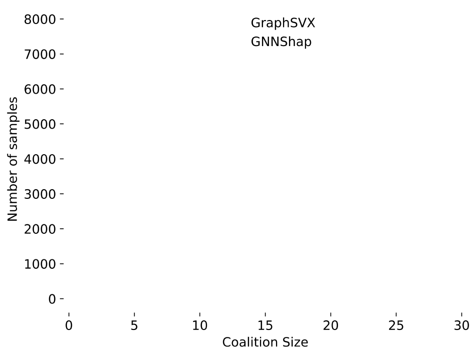
Graph: G **Computational graph:** G_C

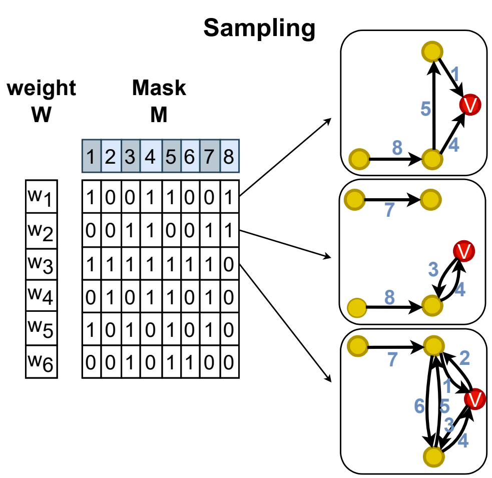


Step 3: Model predictions

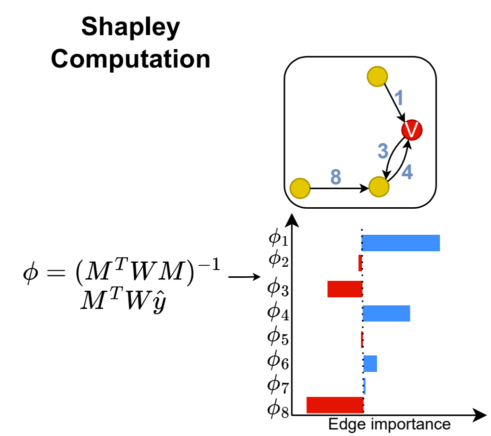


Step 2: Sampling (mask matrix)



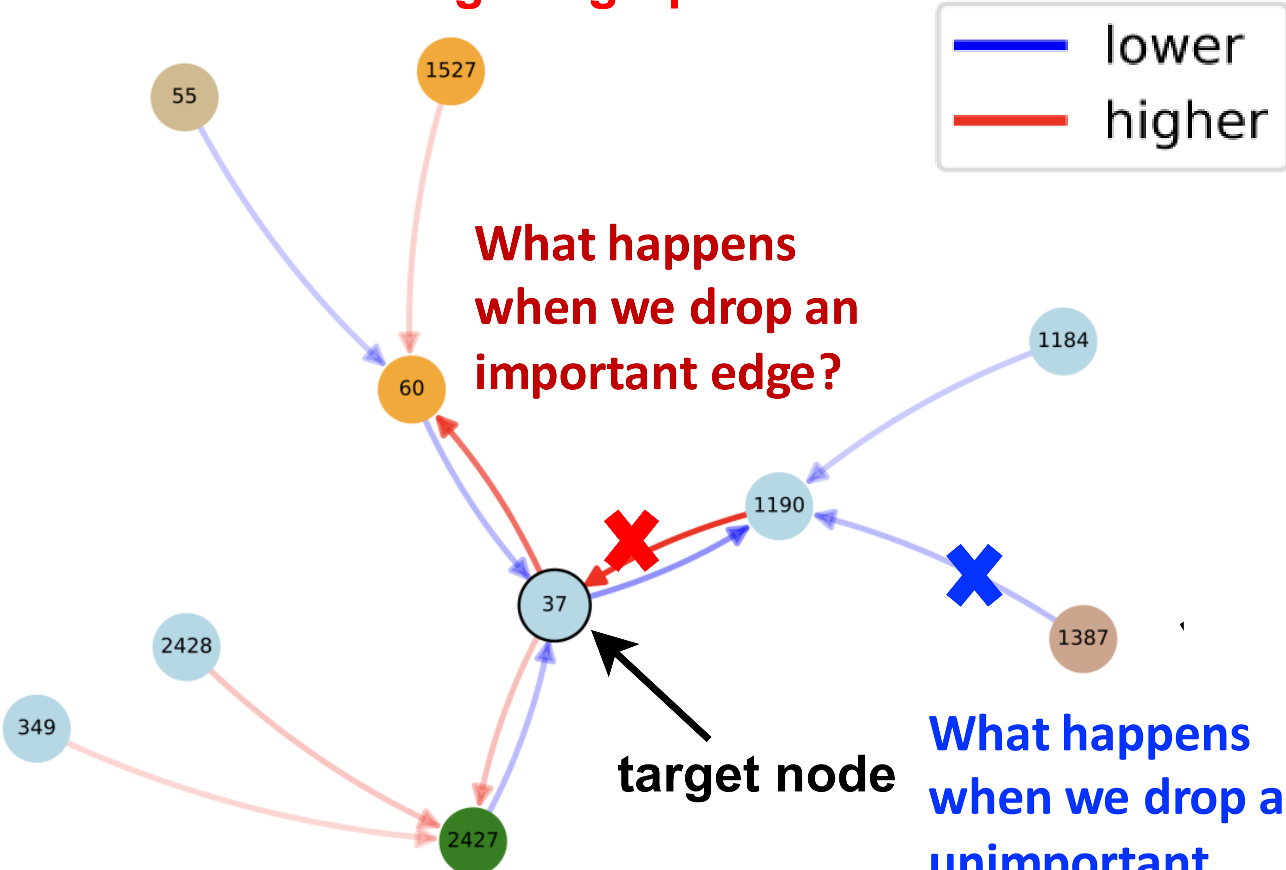


Step 4: Shapley computation



Output of GNNShap

original graph



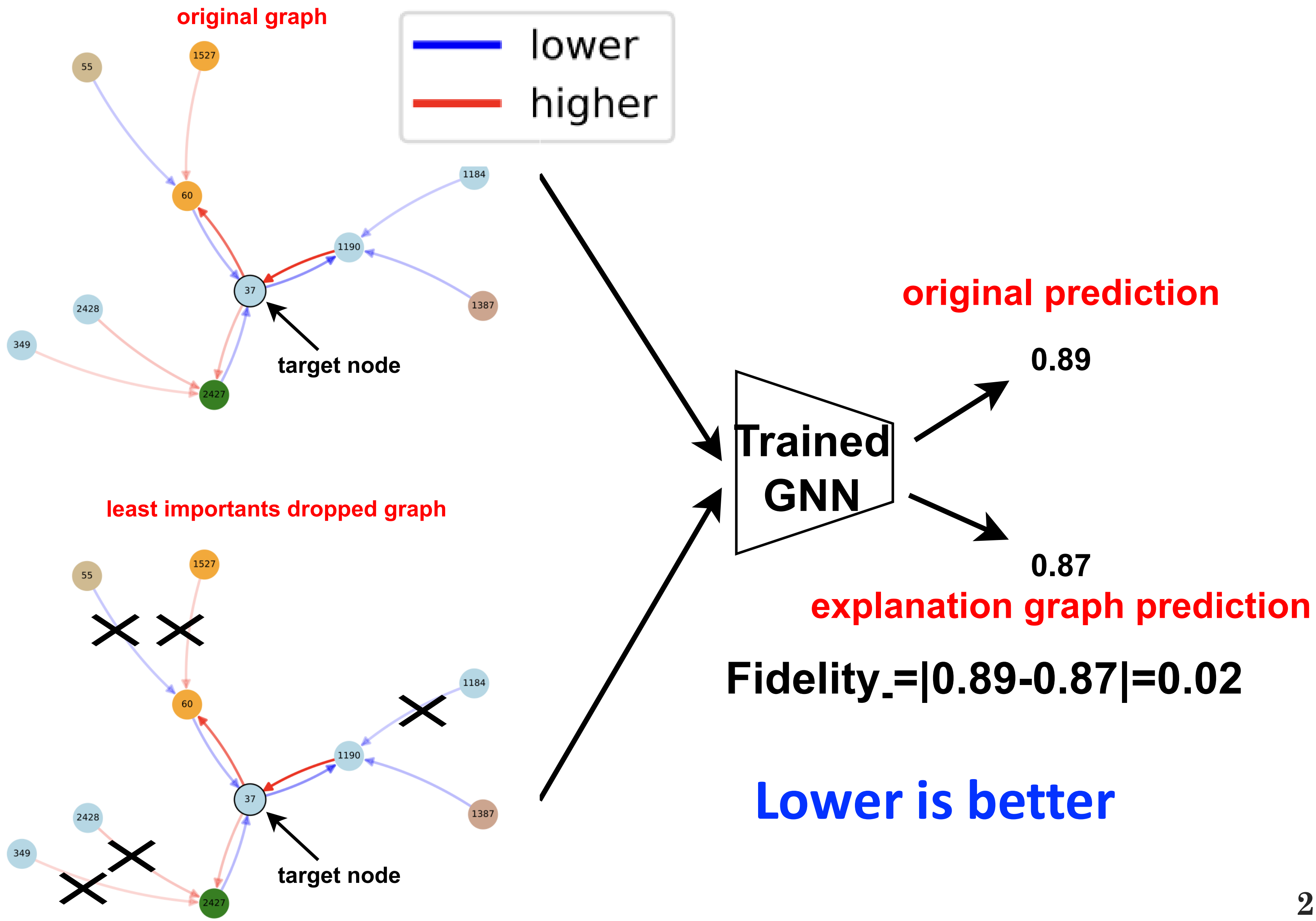
What happens when we drop an important edge?

Model prediction changes significantly

What happens when we drop an unimportant edge?

Model prediction does not change

Explanation Fidelity (Fidelity-)

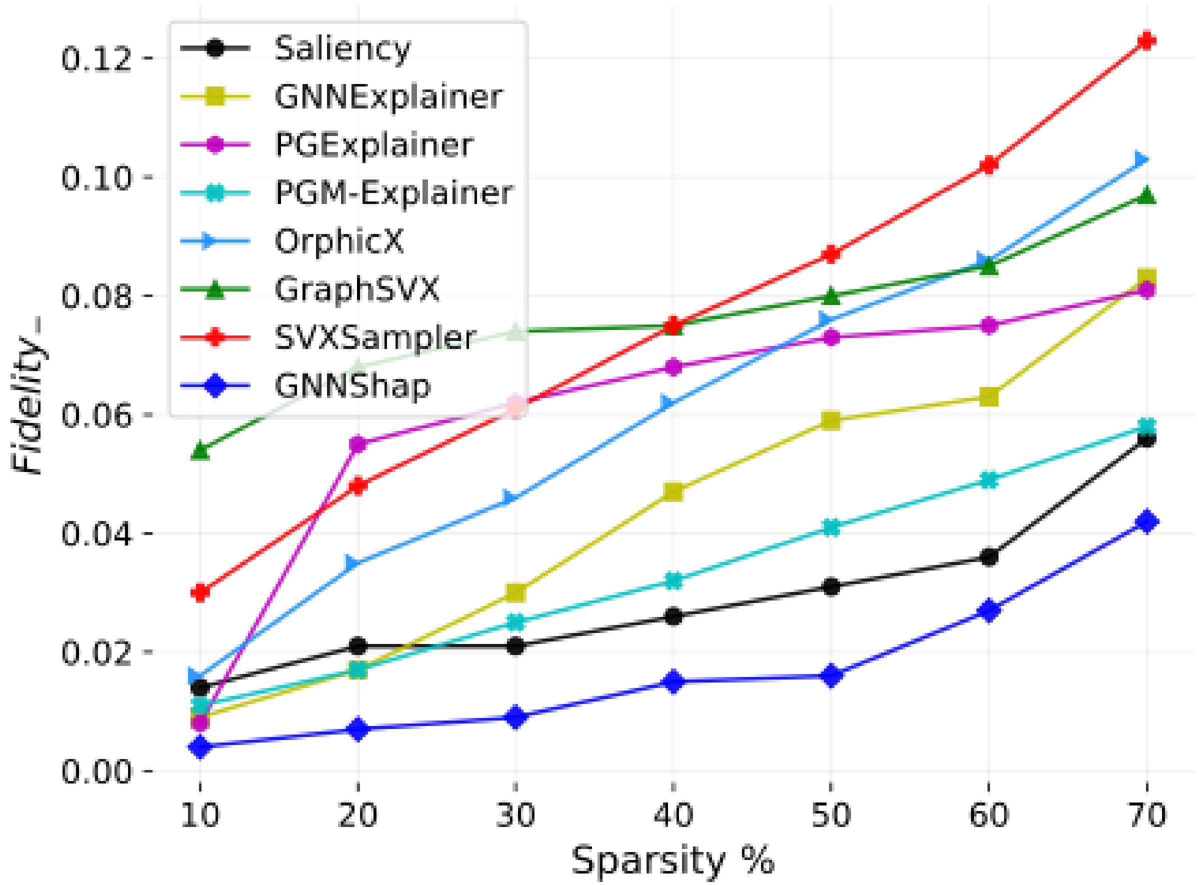


$\text{Fidelity}_{-}=|0.89-0.87|=0.02$

Lower is better

Explanation Quality (**Fidelity-**)

Fidelity-: *Drop least important edges identified by a method. Then measure the difference in prediction score.* ***GNN(E) – GNN(E\e)***. Lower is better.

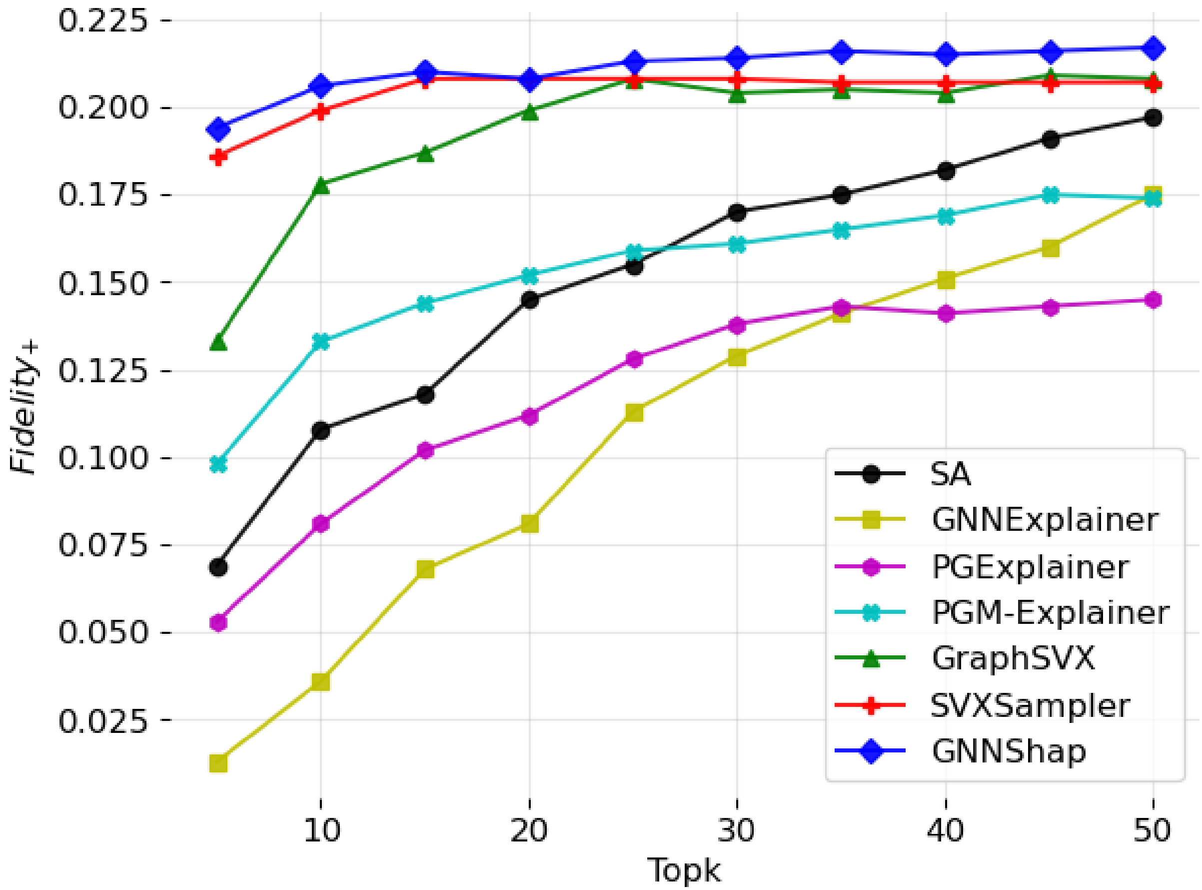


Model: 2-layer GCN
Data: Cora
Task: node classification for 50 test nodes

Removing least important edges by different methods.

Explanation Quality (**Fidelity+**)

Fidelity+: *Drop most important edges identified by a method. Then measure the difference in prediction score.* ***GNN(E) – GNN(E\e)***. Higher is better.



Model: 2-layer GCN
Data: Cora
Task: node classification for 50 test nodes

Removing top-k most important edges by different methods.

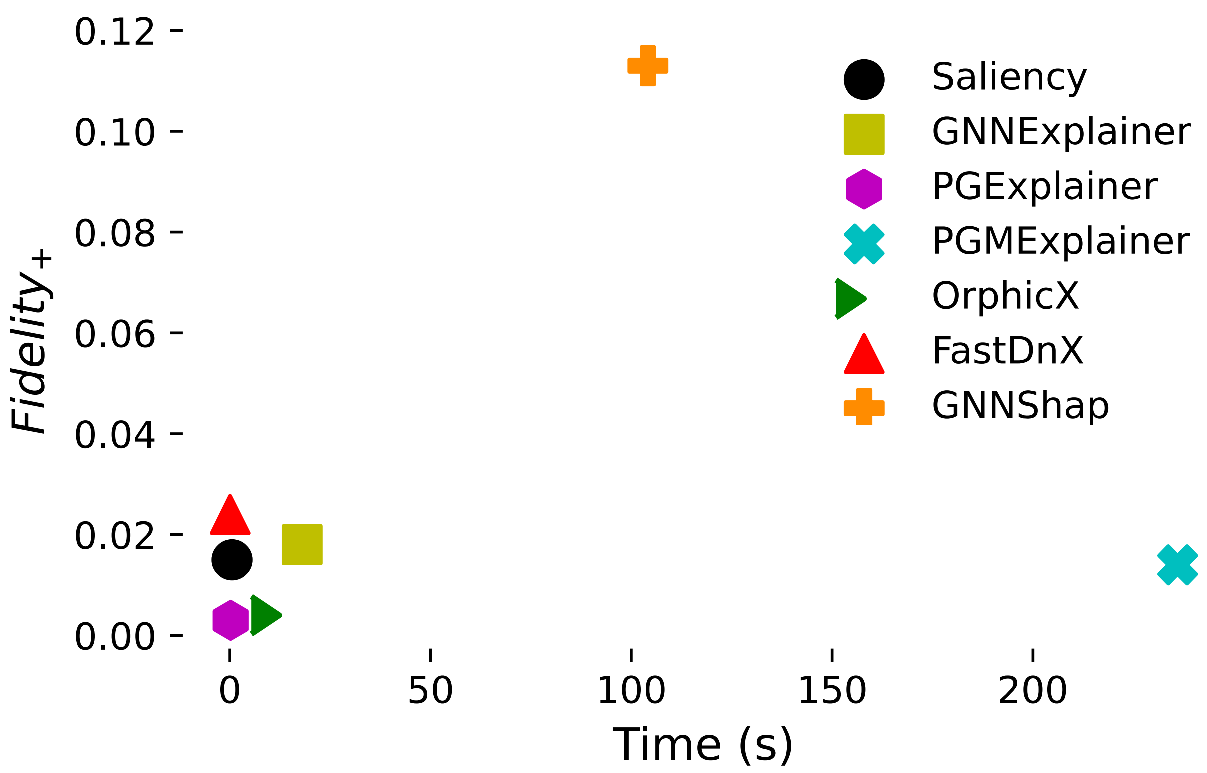
Computational Complexity

GNNShap Computational Complexity

- Consider explaining prediction of a node with 32 edges in its two-hop subgraph
  - **Exact Shapley computations:** evaluates 2^{32} =4 billion subgraphs
  - **GNNSHap**: ~ 20,000 subgraphs using sampling
  - Still, we need to run 20,000 GNN inferences on different subgraphs, which is embarrassingly parallel but expensive
  - This computation is just to explain the prediction of a node

GNNShap Runtime Performance

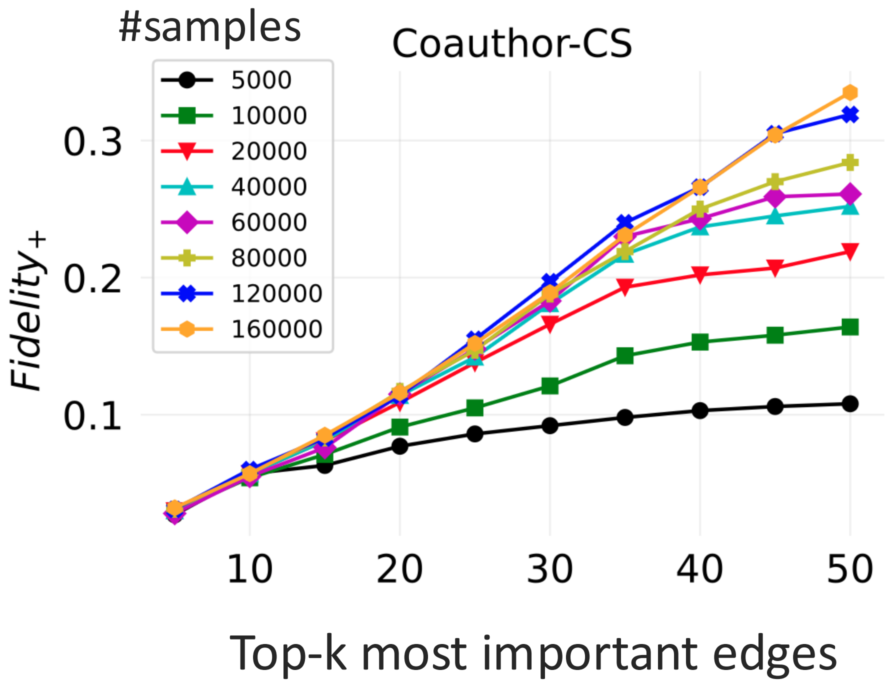
- GNNShap high quality, but still computationally expensive



Scalability Challenge

- Shapley-based methods are more accurate, but computationally expensive
  - **More samples => better fidelity**
  - **More sample => expensive**
- **Need a distributed algorithm that can use multiple GPUs/CPUs**

GNNShap Fidelity+ (higher is better)



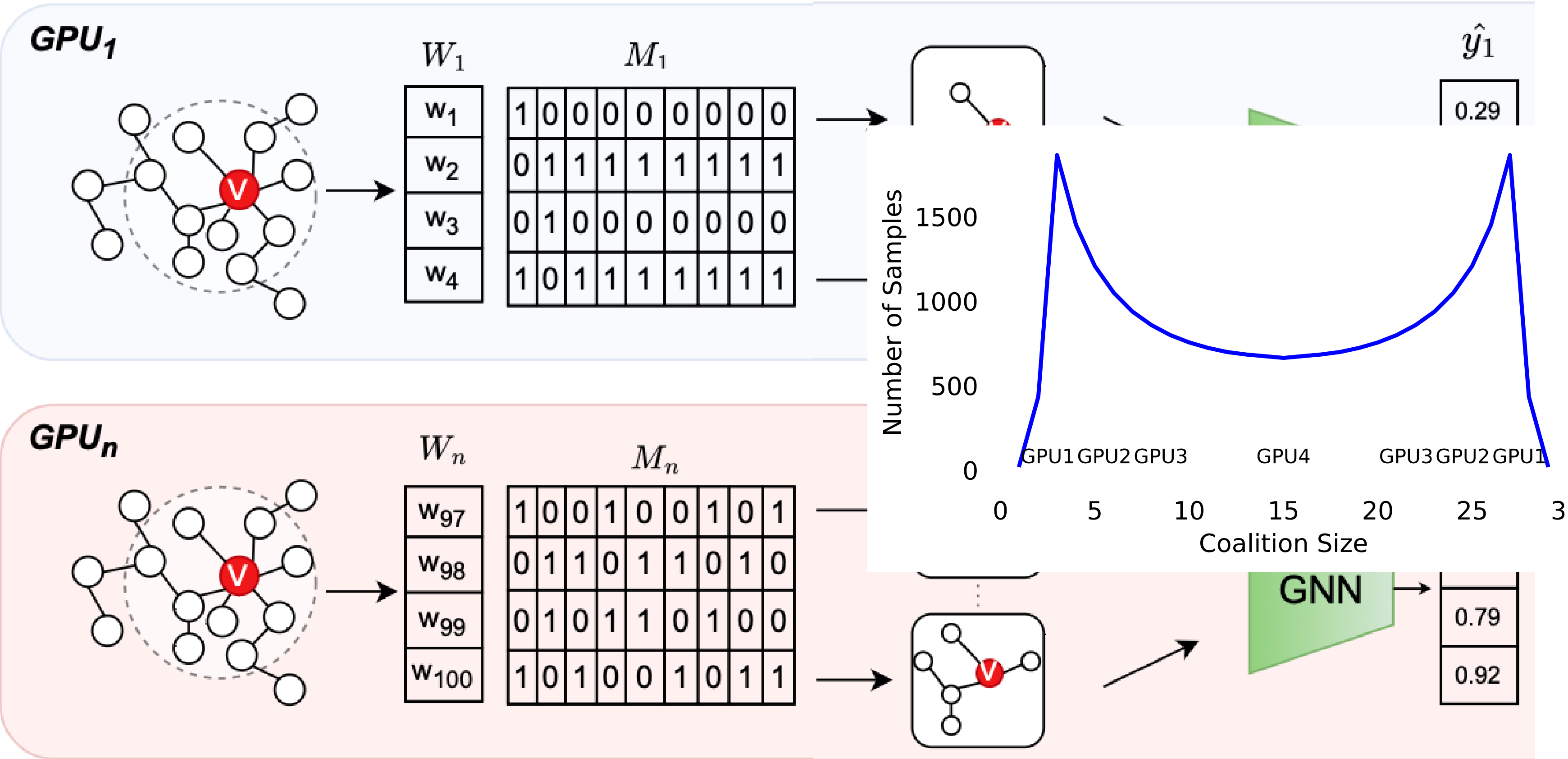
DistShap Overview

(Akkas, Devarakonda, and Azad, VLBD, 2026)

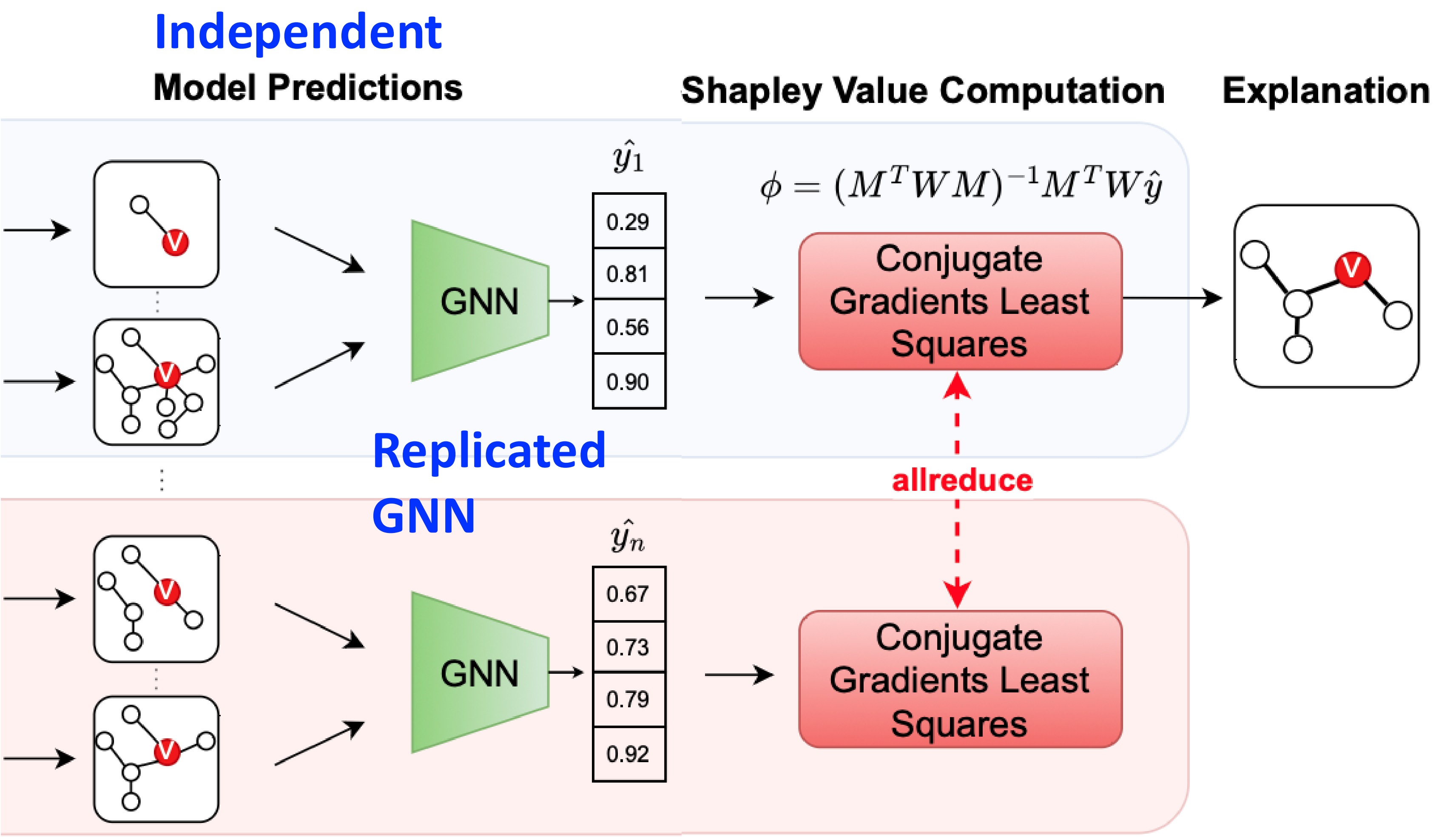
Replicated Computational Graph

Independent Local Sampling

Independent Model Predictions

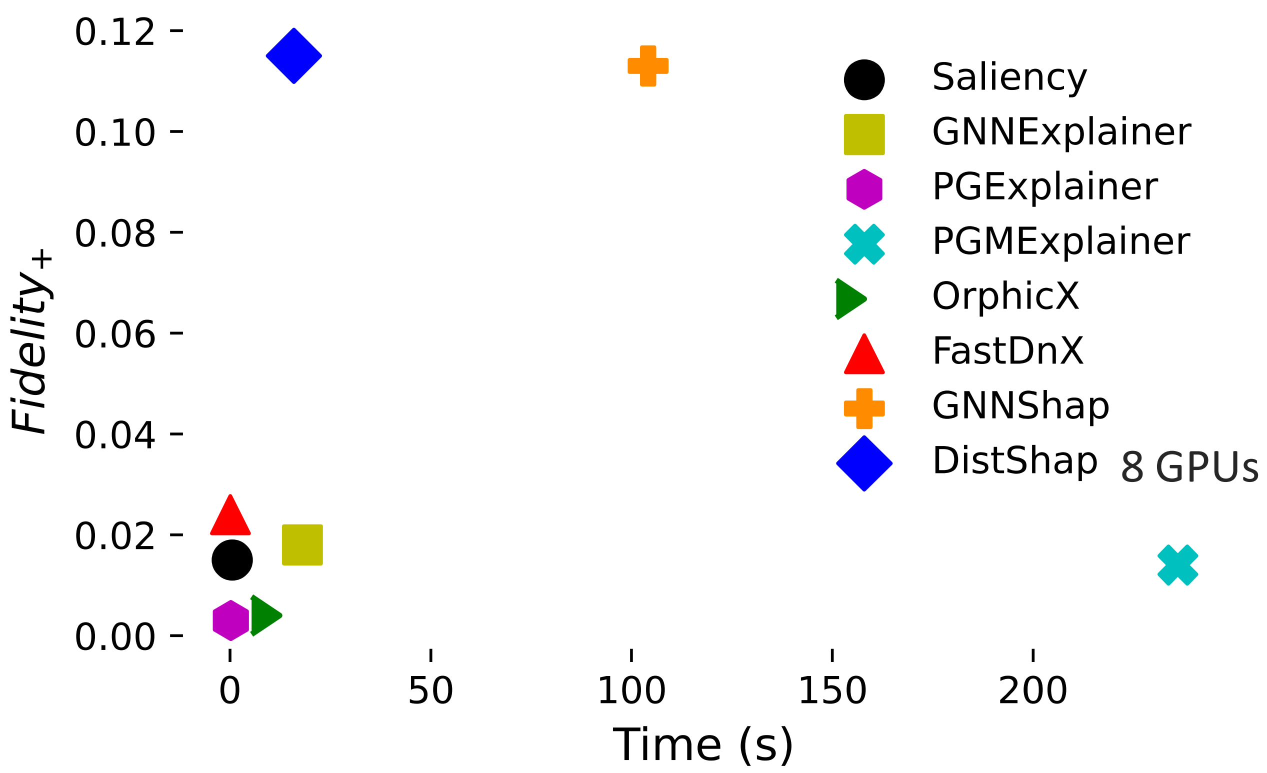


DistShap Overview



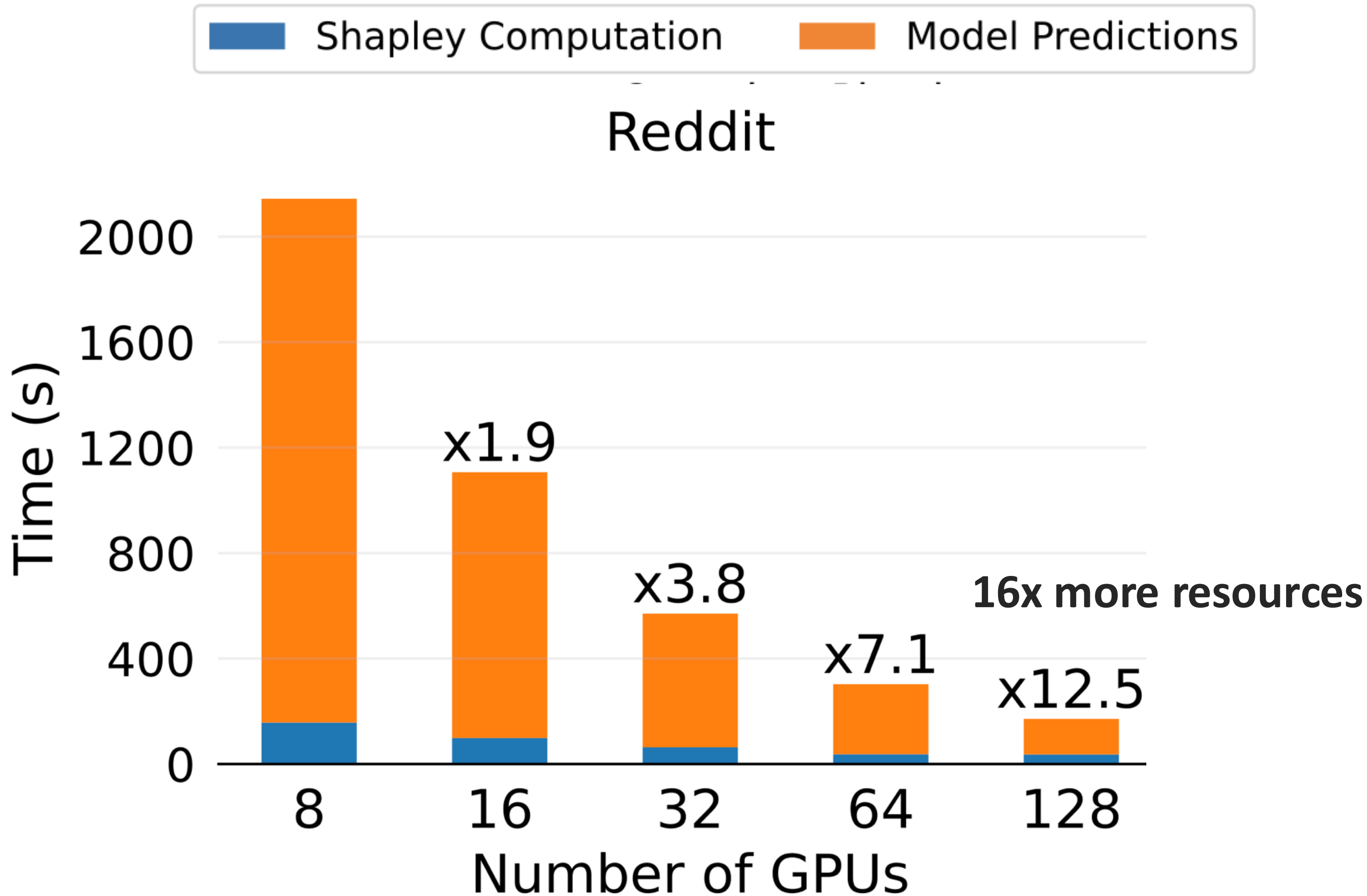
DistShap Performance

- ❑ DistShap: Quality same as GNNShap
- ❑ Faster with more GPUs (skipping the method)

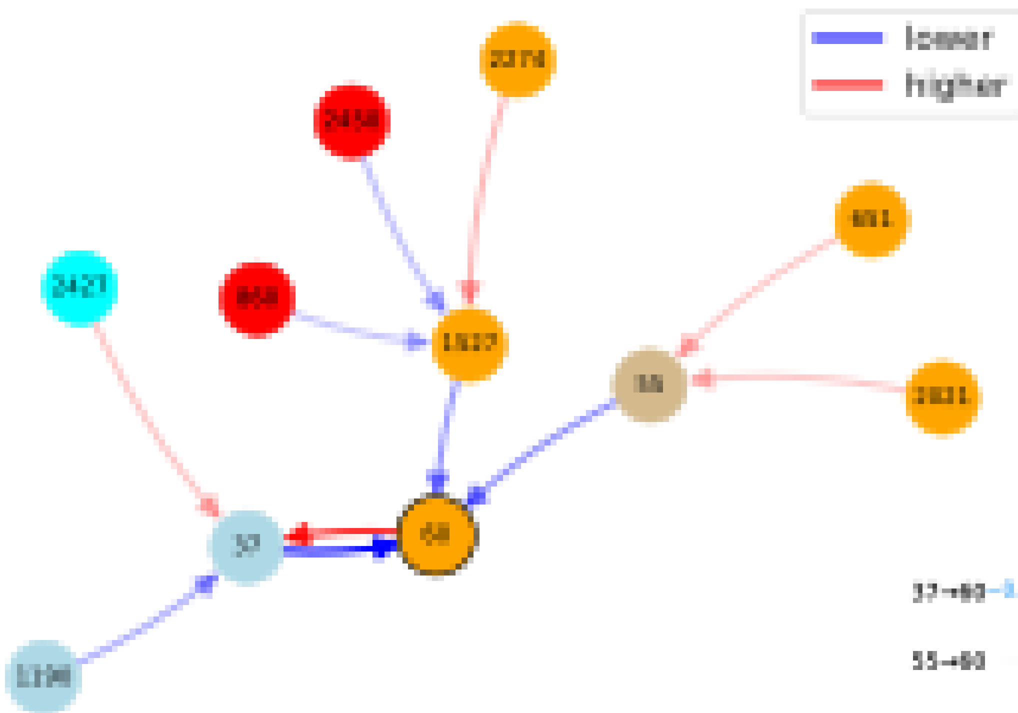


Scalability of DistShap

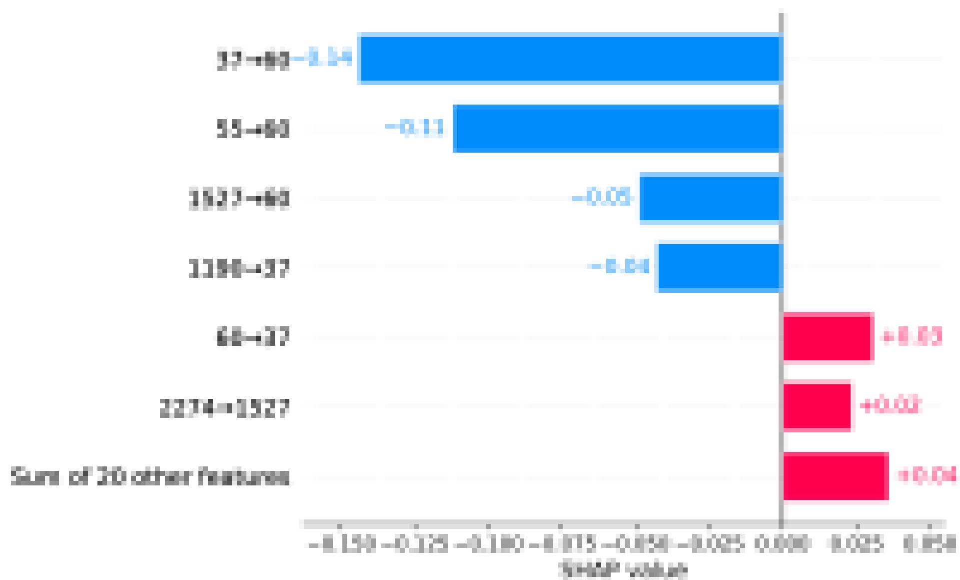
Large dataset: 50-100K players, 600K samples
8 to 128 NVIDIA A100 GPUs



Application in Graph Sparsification

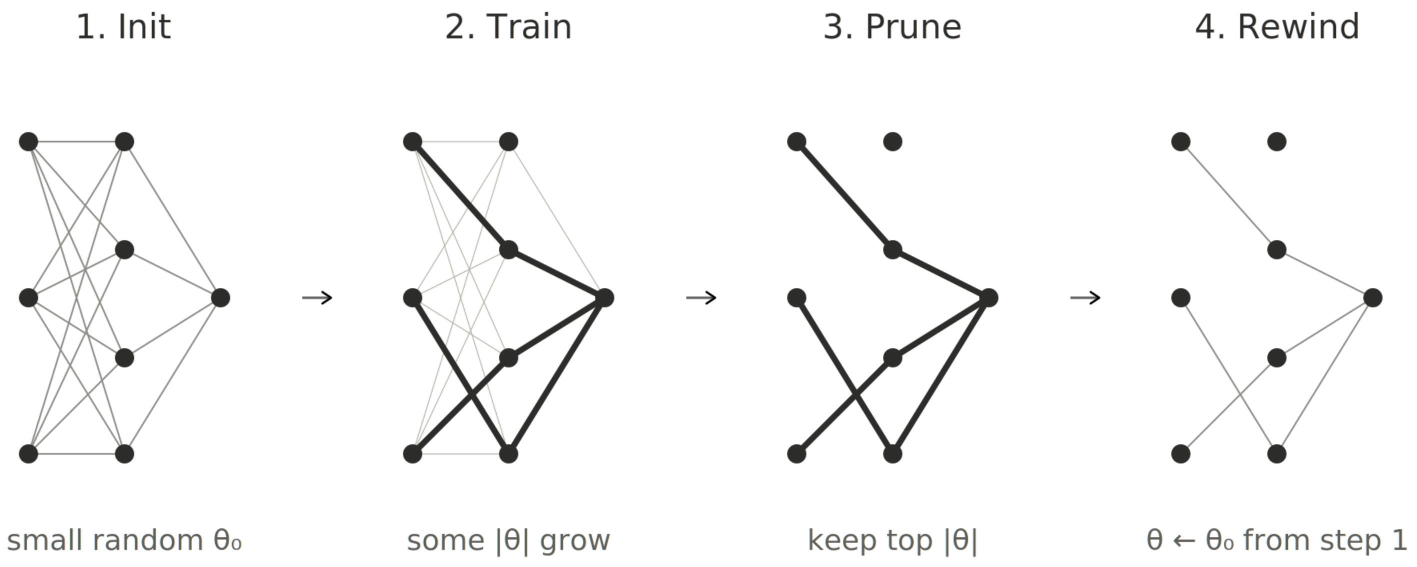


Can we remove less important edges from the graph entirely?



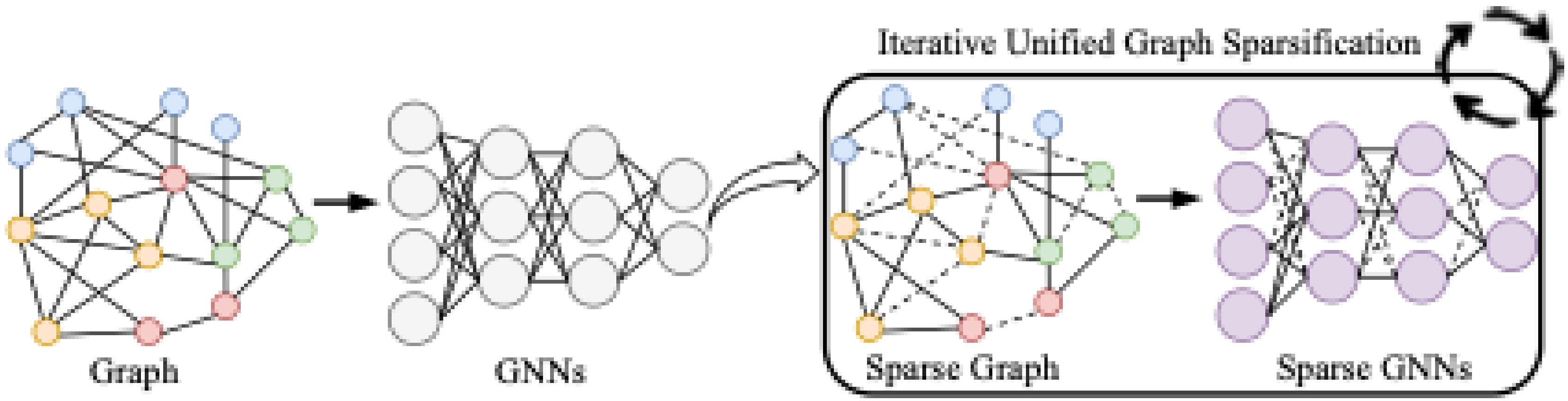
Lottery Ticket Hypothesis

- **Frankle & Carbin, ICLR 2019**: A dense, randomly initialized network contains a sparse subnetwork that, when trained in isolation from the *same* initialization, can match the dense network's test accuracy.



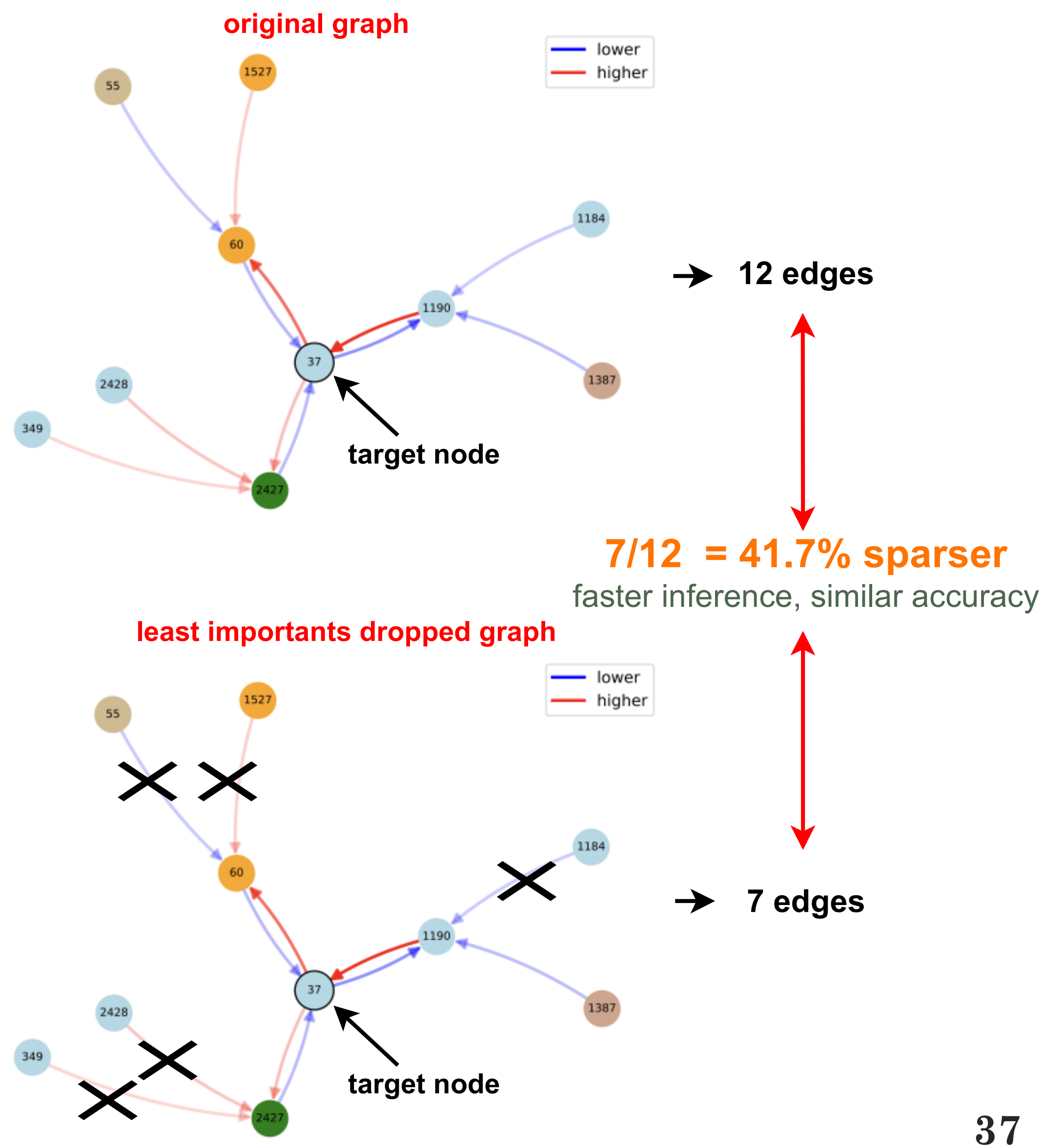
Lottery Ticket Hypothesis For GNN

- There is no practical benefit to sparsifying the model, as GNN models are already small.
- Lottery Ticket Hypothesis For GNN: How to sparsify the input graph!
  - **Graph Lottery Ticket (GLT).** A sparsified subgraph matches a GNN on the full graph. Chen et al., ICML 2021
  - Need a lot of training

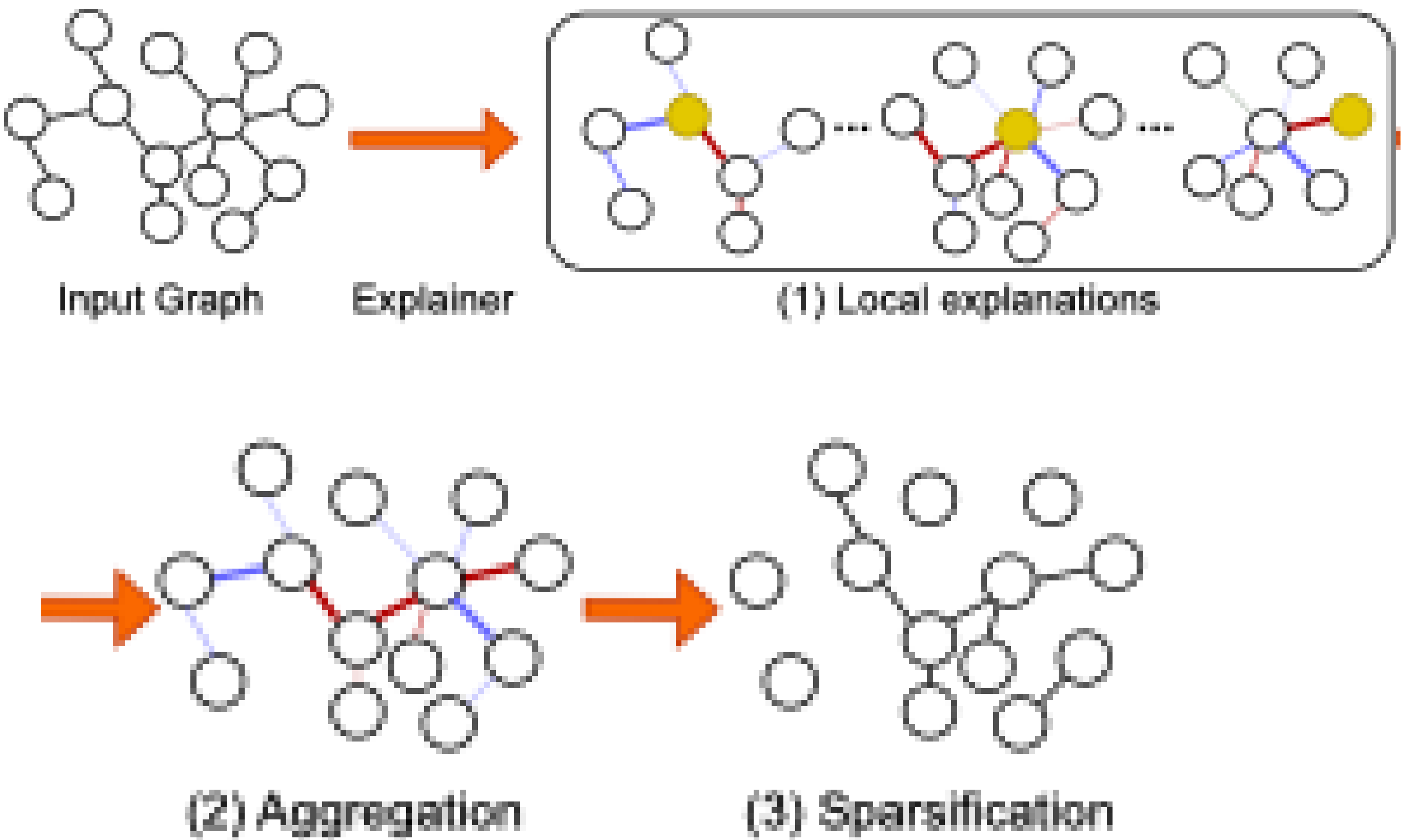


Use of Shapley in Graph Sparsification

- Remove unimportant edges for fast inference
- But this is local for the target node!
- What if these edges are important for other nodes?

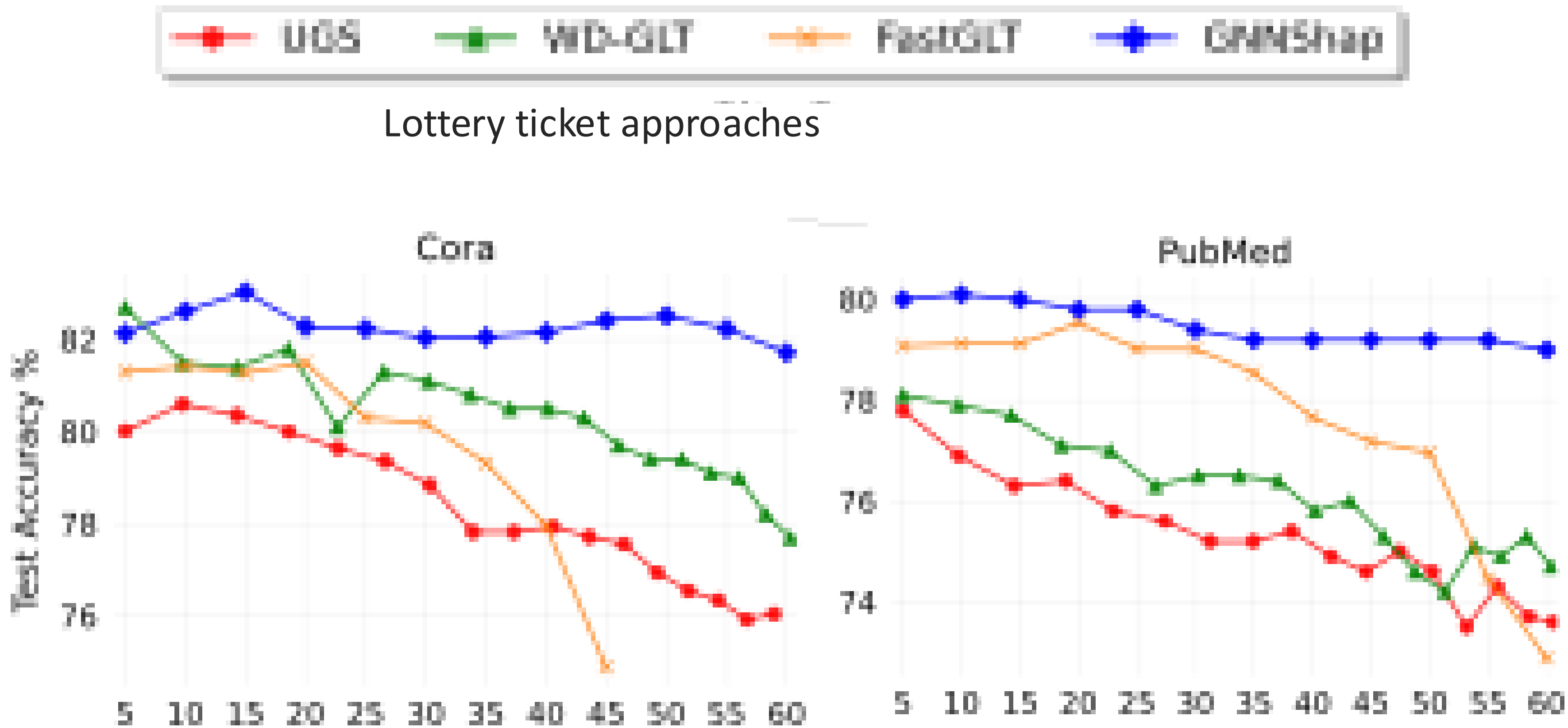


Use of Shapley in Graph Sparsification



Akkas and Azad, WWW 2026

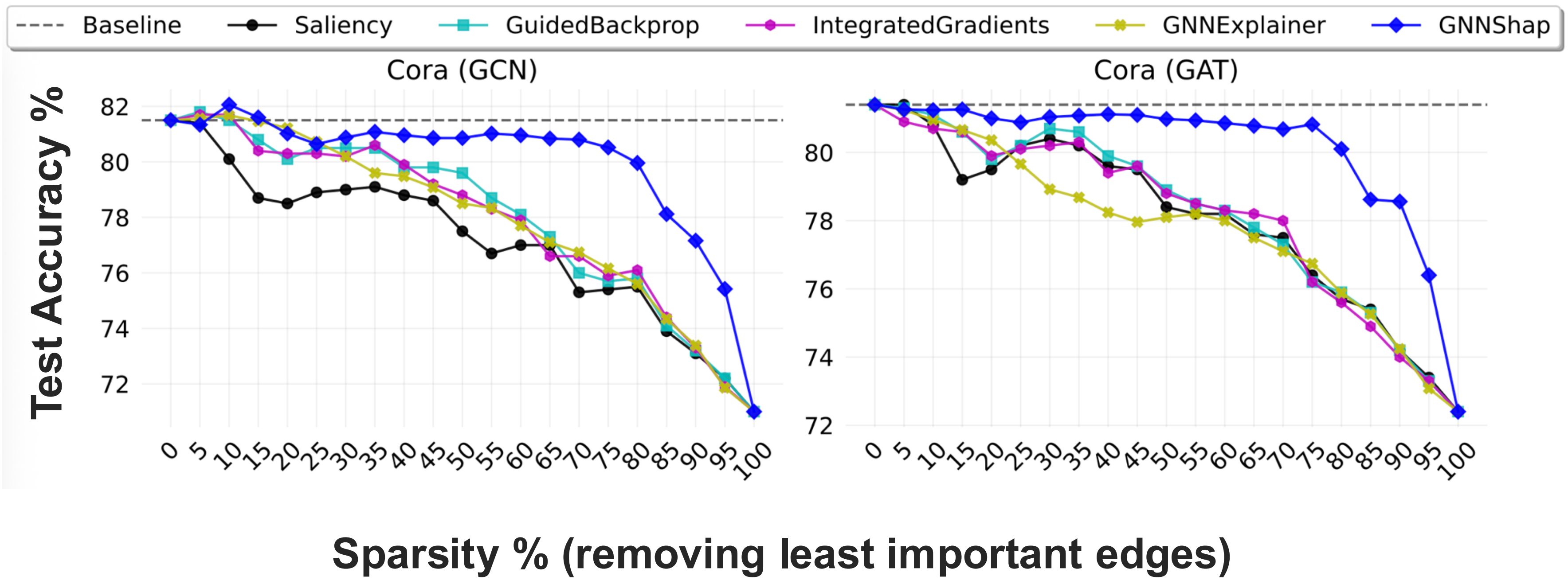
Comparison with Lottery Ticket Approaches



Sparsity % (removing least important edges)

Comparison with Other Explanation Methods

- ❑ **GNNShap offers better sparsity compared to other explanation approaches**



Summary

- Shapley values are arguably the most used method in explainable AI
- GNNShap showed that it is better than other popular GNN explanation methods such as GNNExplainer
- Shapley values can be used to sparsify the graph for faster inference

Possible Extensions

- Shapley-based methods for **relational GNNs** and for **dynamic graphs**
- Better **sampling** algorithms
- **Faster Shapley computation for other models**. SHAP is notoriously slow.
- Shapley-style methods for **spectral GNN**

References:

1. Akkas and Azad, GNNShap: Scalable and Accurate GNN Explanation using Shapley Values, WWW 2024
2. Akkas, Devarakonda, and Azad, DistShap: Scalable GNN Explanations with Distributed Shapley Values, VLDB 2026 (to appear)
3. Akkas and Azad, Explainable Graph Sparsification with Shapley Values, WWW 2026