Graph Data Mining and Learning: Modern applications, challenges, and paradigms

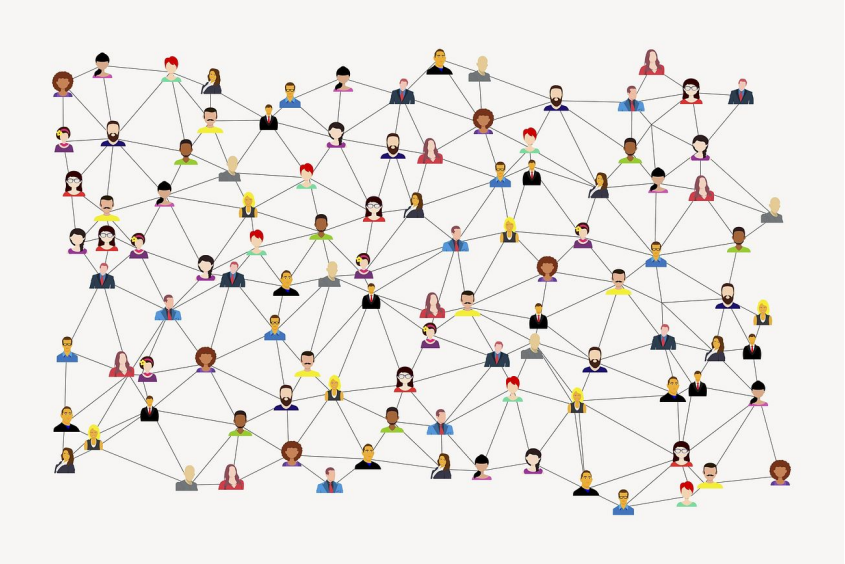
Naheed Anjum Arafat

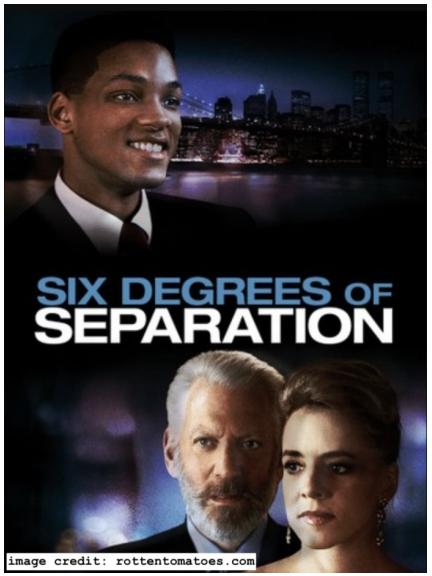
Post-doc
Nanyang Technological University, Singapore (2021-2024)
PhD
National University of Singapore

January 22, 2025

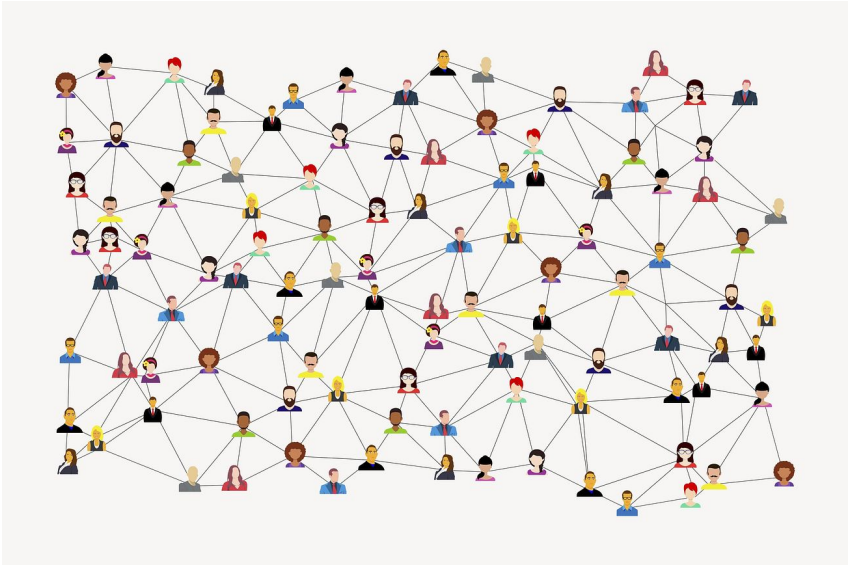
Outline

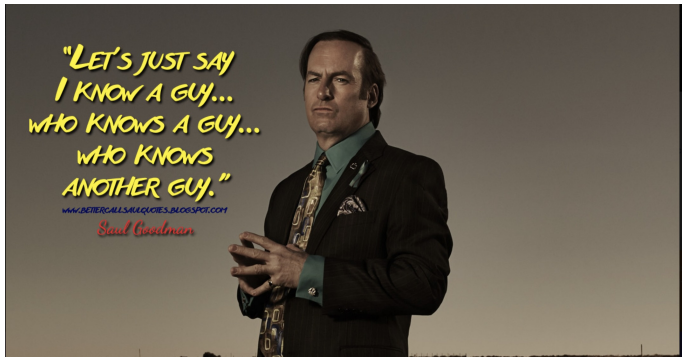
Graphs before..





Graphs before..

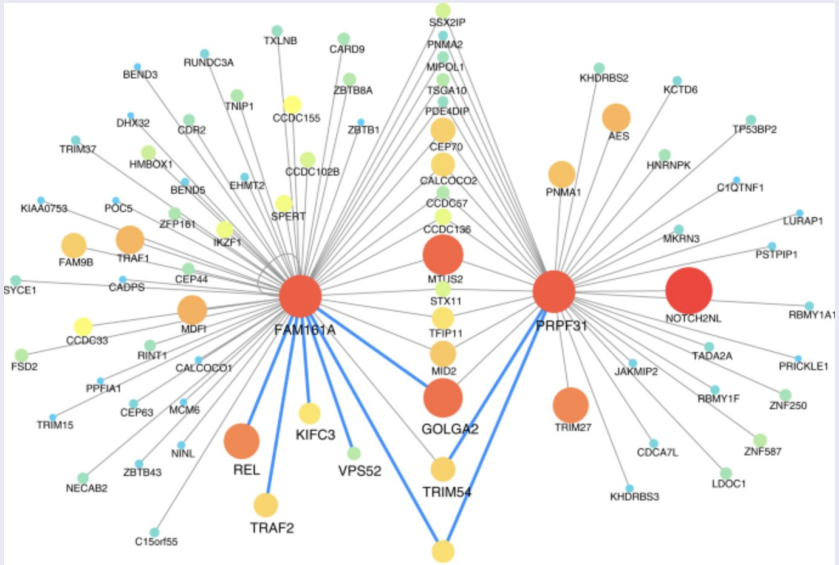




Graphs now..

Well, they are everywhere

Biology



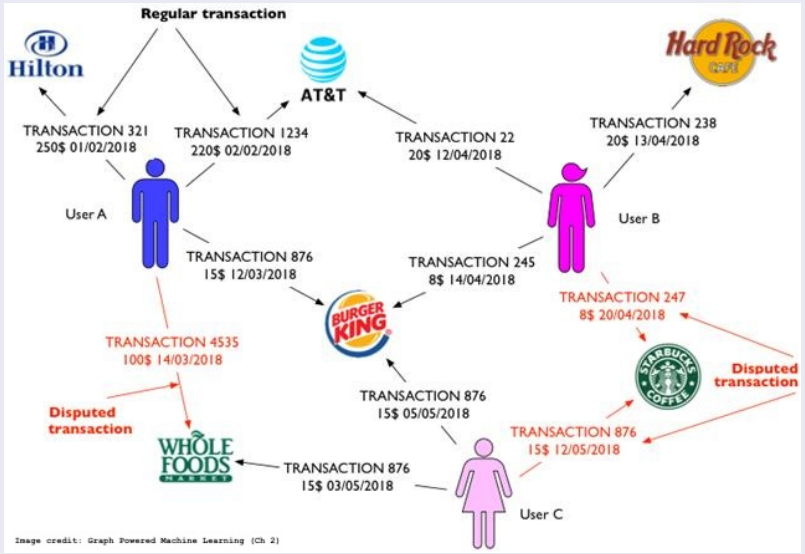
Protein-protein interactions

Image credit: Kovács, István A., et al. "Network-based prediction of protein interactions." Nature comm., 2019

Graphs now..

Well, they are everywhere

Finance

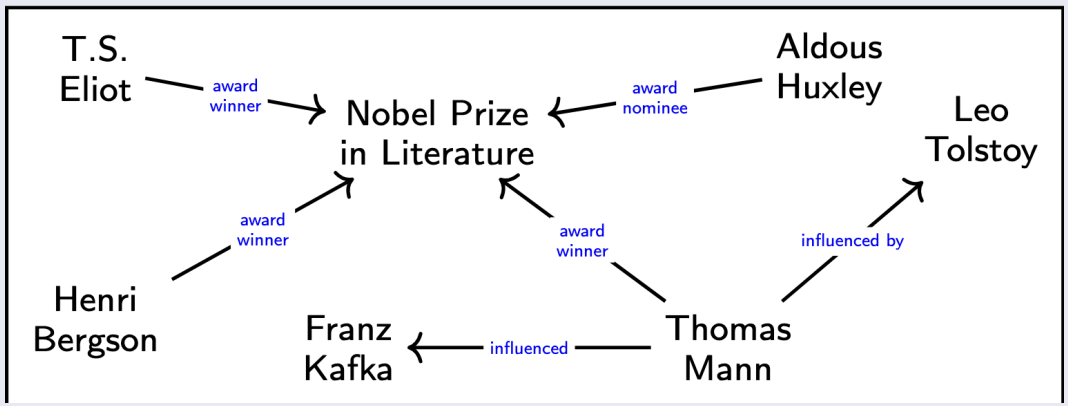


Financial transactions

Graphs now..

Well, they are everywhere

Human centered computing



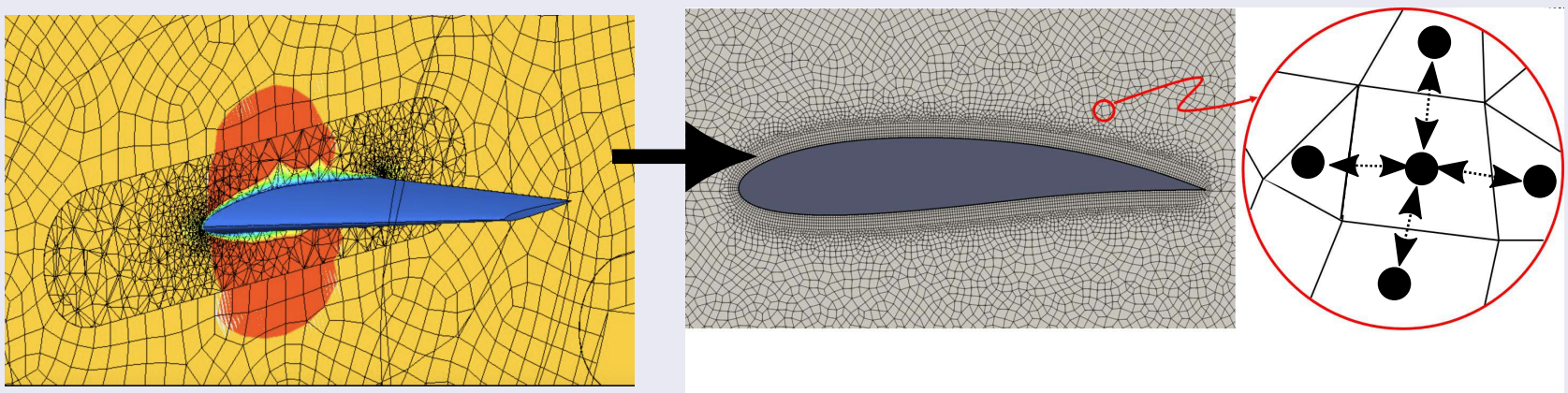
Knowledge graphs for explainable AI

Modern paradigm: Human-understandable explanation to black-box AI models. For instance, Recommending *War and Piece* because you have read *The Magic Mountain* by Thomas Mann who was influenced by works of Tolstoy.

Graphs now..

Well, they are everywhere

Physics



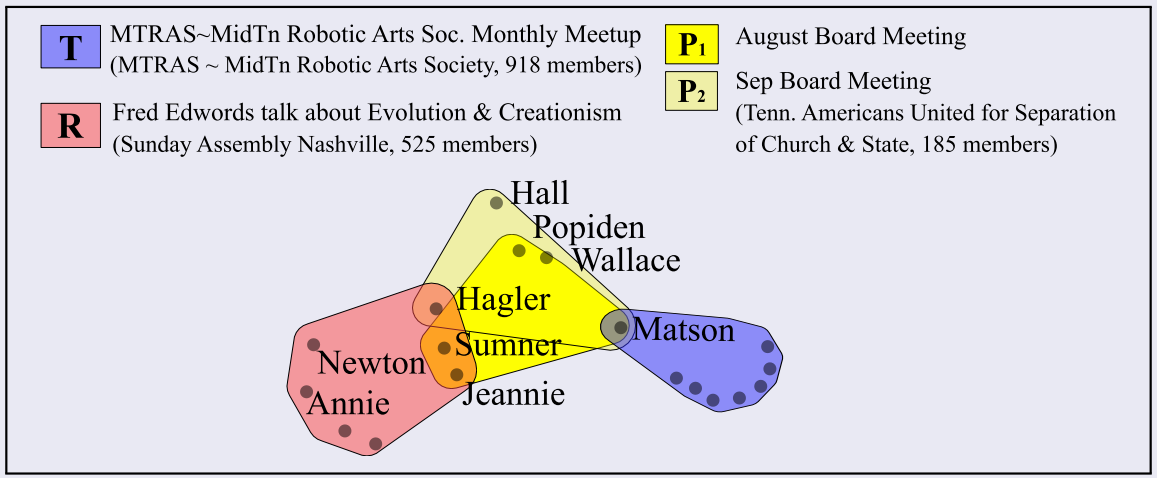
Fluid simulation at cross-section of an airplane wing (aka airfoil) and a close-up view

Modern paradigm: Modeling fluid flow as message passing on graphs.

Graphs now..

Well, they are everywhere

Public Health and Epidemiology

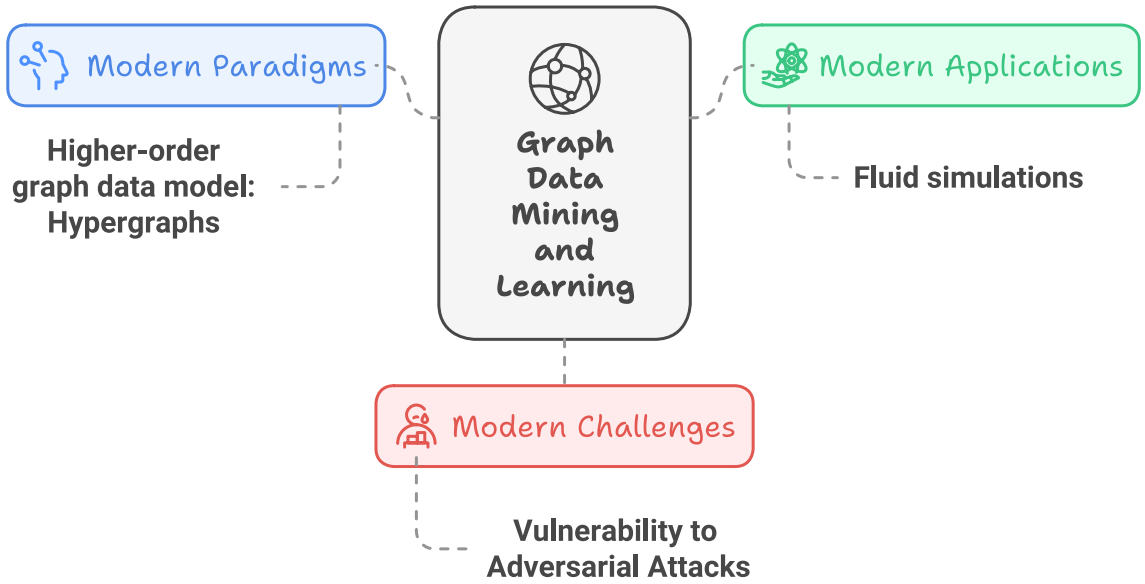


Social meet-ups (each meet-up event is a hyperedge)

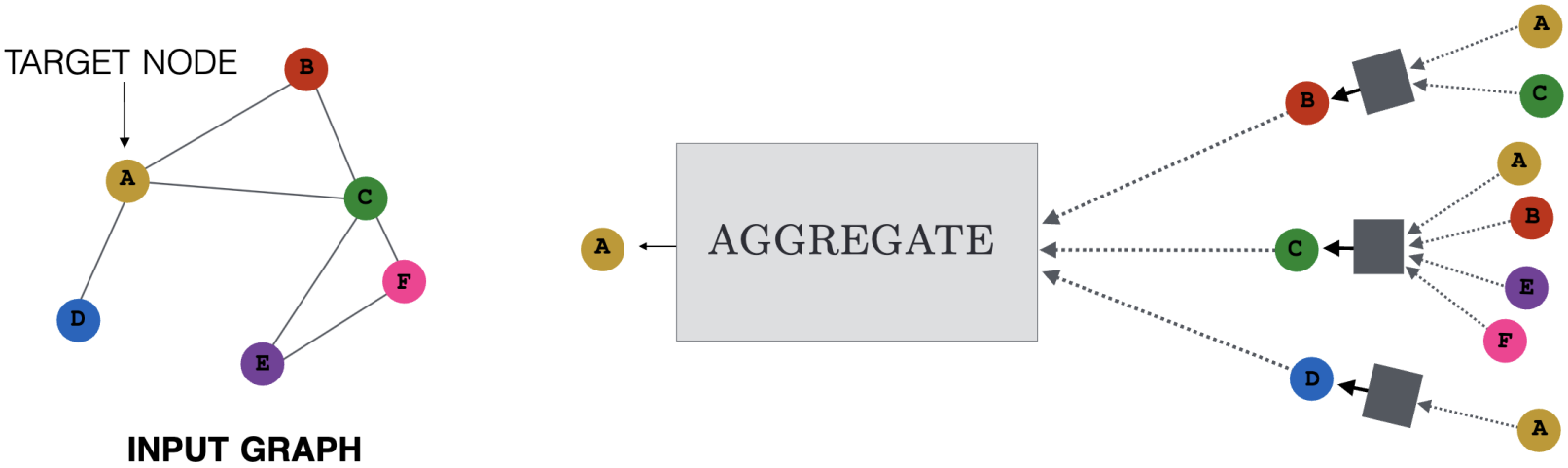
Modern paradigm: Modeling higher-order interactions as Hypergraphs.

My Research: Overview

My research aims to advance our capabilities in extracting insights from graphs and learning on graphs.



My Research: Learning on Graph



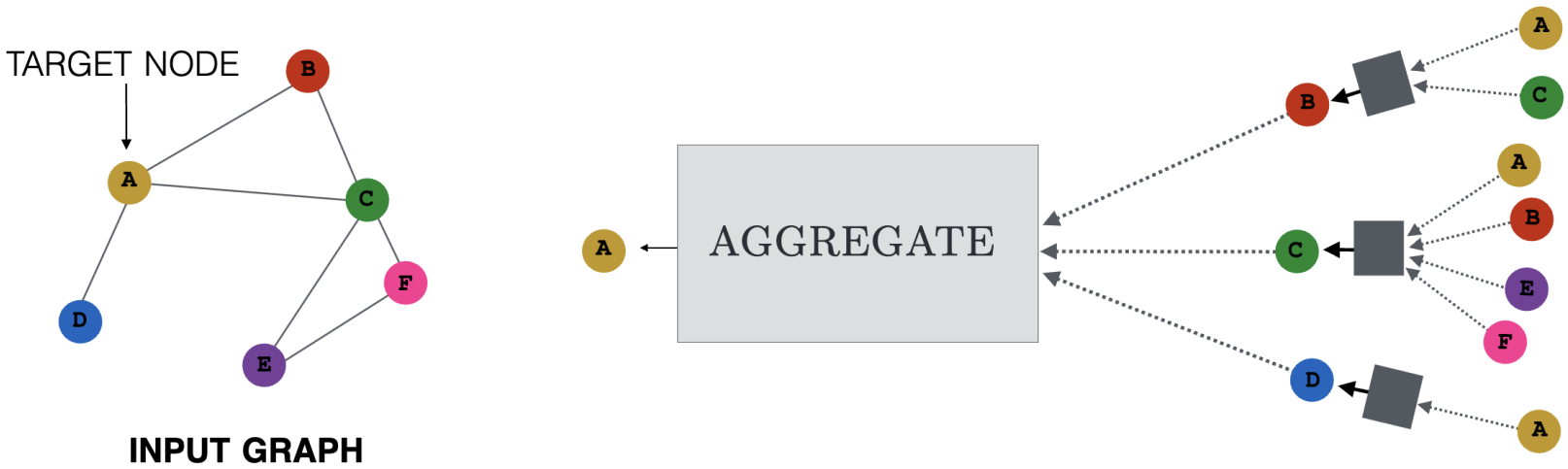
Graph Neural Network

When GNNs (applications):

- Node classification(identifying fraudulant transactions)
- Link prediction (recommending movies, videos, posts, products)
- Graph classification (identifying toxic chemicals)

Basically, any learning task involving entities connected by relations.

My Research: Learning on Graph



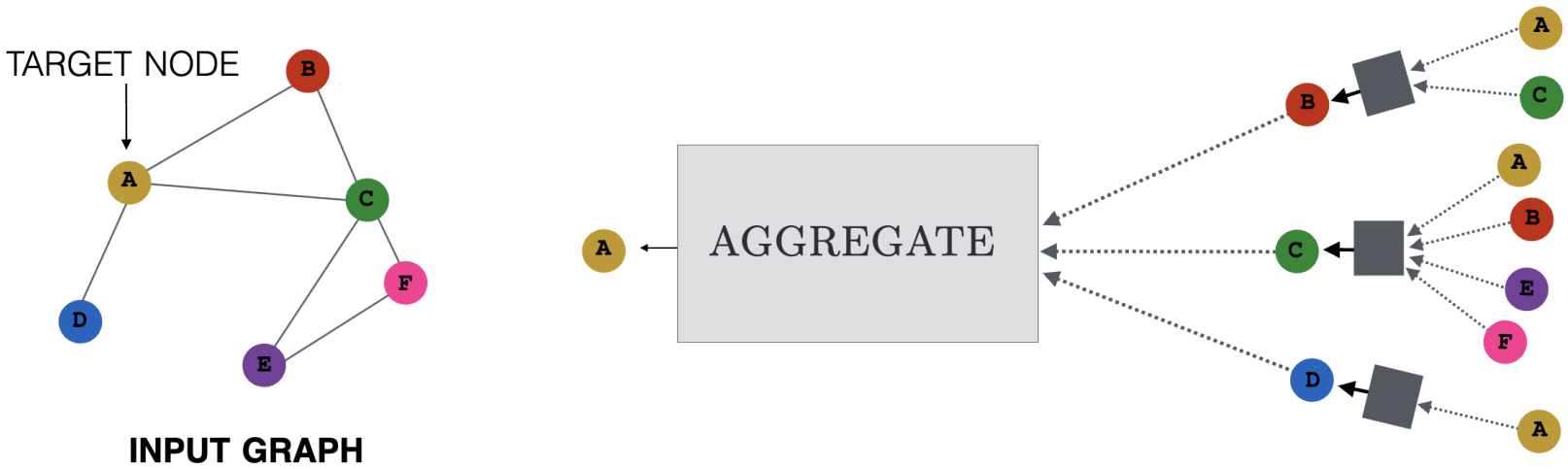
Graph Neural Network

When GNNs (applications):

- Node classification(identifying fraudulant transactions)
- Link prediction (recommending movies, videos, posts, products)
- Graph classification (identifying toxic chemicals)

Basically, any learning task involving entities connected by relations.

My Research: Learning on Graph



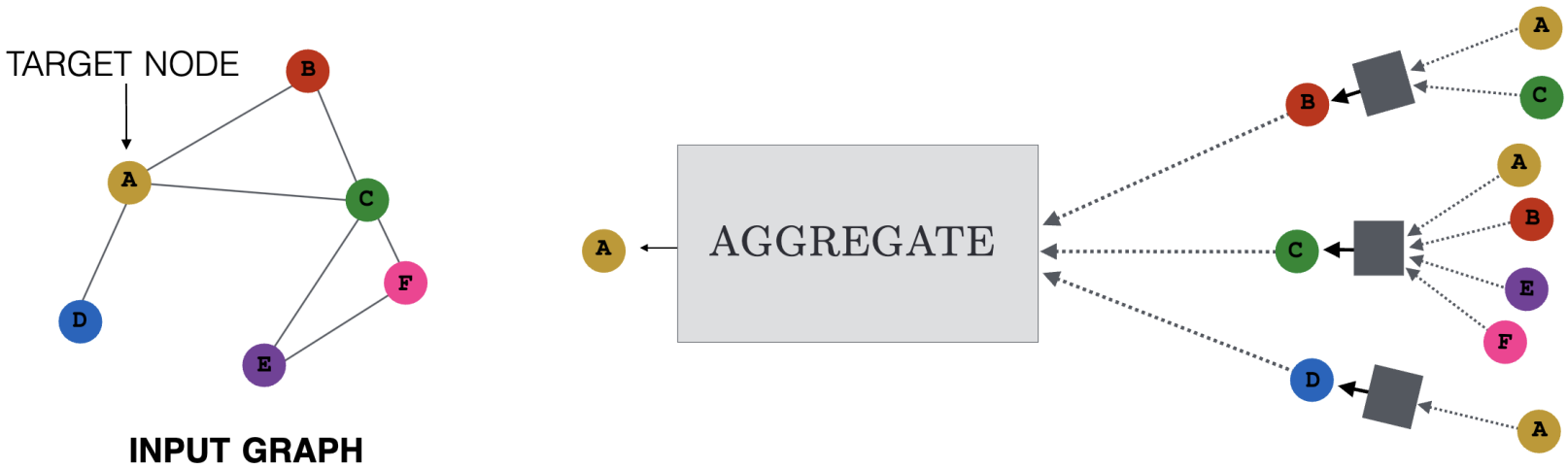
Graph Neural Network

When GNNs (applications):

- Node classification(identifying fraudulant transactions)
- Link prediction (recommending movies, videos, posts, products)
- Graph classification (identifying toxic chemicals)

Basically, any learning task involving entities connected by relations.

My Research: Learning on Graph



Graph Neural Network

When GNNs (applications):

- Node classification(identifying fraudulant transactions)
- Link prediction (recommending movies, videos, posts, products)
- Graph classification (identifying toxic chemicals)

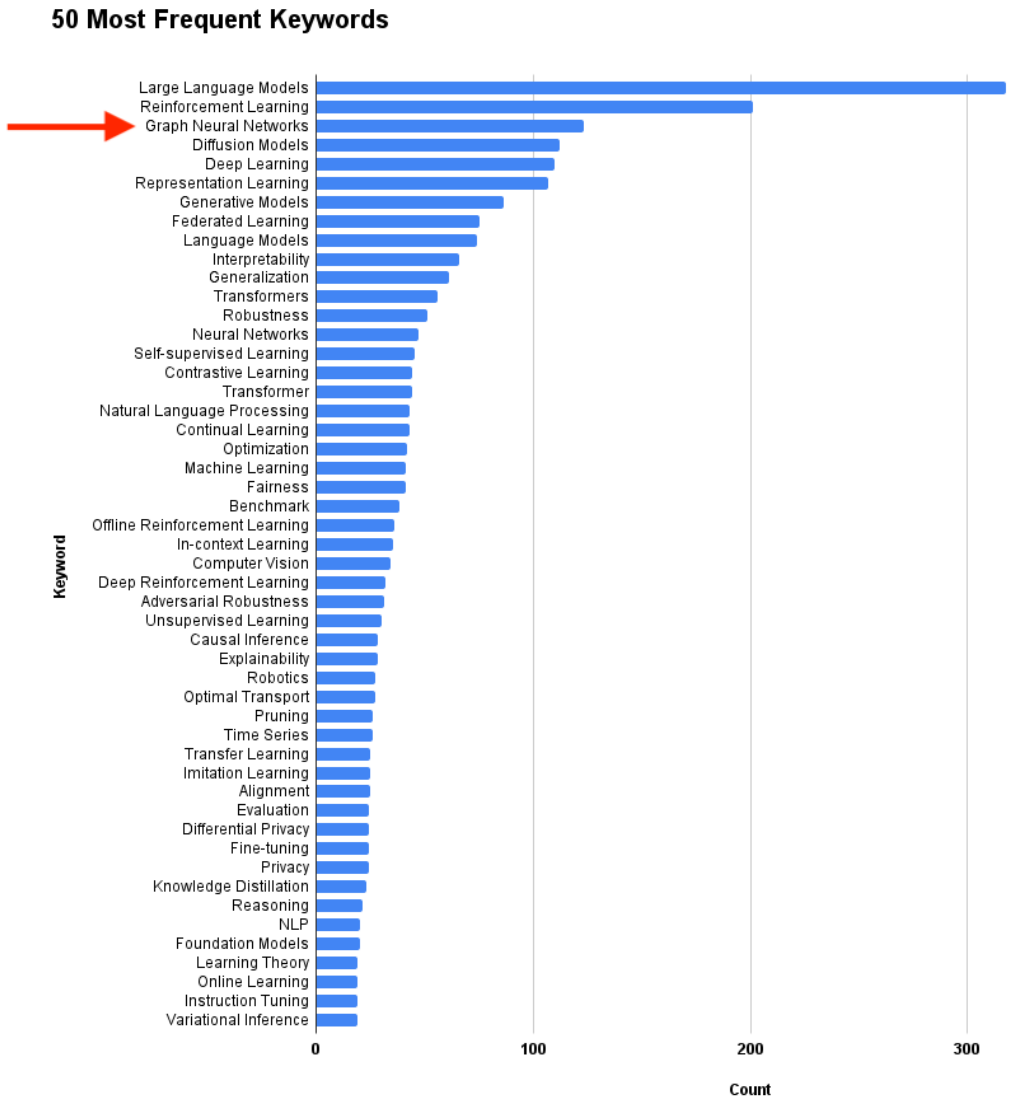
Basically, any learning task involving entities connected by relations.

My Research: Why GNNs?

GNNs in Industry:

- Uber (food/restaurant recommendations)
- Alibaba (product recommendation)
- Snapchat (content recommendations)
- Pinterest (Pins recommendation)
- Google (Google Maps and Deepmind)
- Kumo.ai (fraud detection)
- Ant financial Services (fraud detection)
- Meta (preventing spread of misinformation, fake account detection)

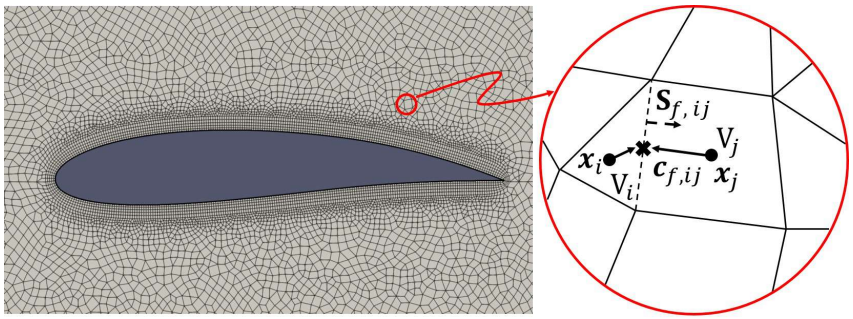
GNNs in Academia: Top 50 keywords in submitted research papers at ICLR 2024.



Fluid Simulation: a Modern Application of GNNs

Finite Volume Features, Global Geometry Representations, and Residual Training for Deep Learning-based CFD Simulation, ICML 2024 (spotlight)

- **Problem**: Predict velocity, pressure at every point in the domain using AI.
- **Motivation:**
  - Numerical simulations of fluids are computationally expensive often requiring days.
  - Recently, data-driven GNNs have been deployed as an efficient alternative. (e.g. MeshGraphNet from Deepmind)
  - Can we improve the fidelity of data-driven GNNs?



(a) CFD mesh with an airfoil body surrounded by different sizes of cells. (b) Illustration of cell characteristics, namely cell centroids, x_i and x_j, face centroid, $c_{f,ij}$, face area normal vector, $S_{f,ij}$, and cell volumes, V_i and V_j.

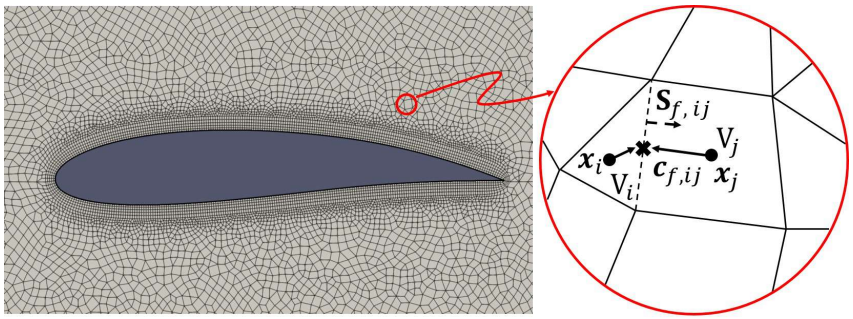
Contributions

1. Geometric features to inform GNN nodes of long-range interactions.
2. Finite-volume features in the graph convolution layer.
3. *Super-resolution:* exploited residual training w/ low-res simulation data to ease the learning.

Fluid Simulation: a Modern Application of GNNs

Finite Volume Features, Global Geometry Representations, and Residual Training for Deep Learning-based CFD Simulation, ICML 2024 (spotlight)

- **Problem**: Predict velocity, pressure at every point in the domain using AI.
- **Motivation:**
  - Numerical simulations of fluids are computationally expensive often requiring days.
  - Recently, data-driven GNNs have been deployed as an efficient alternative. (e.g. MeshGraphNet from Deepmind)
  - Can we improve the fidelity of data-driven GNNs?

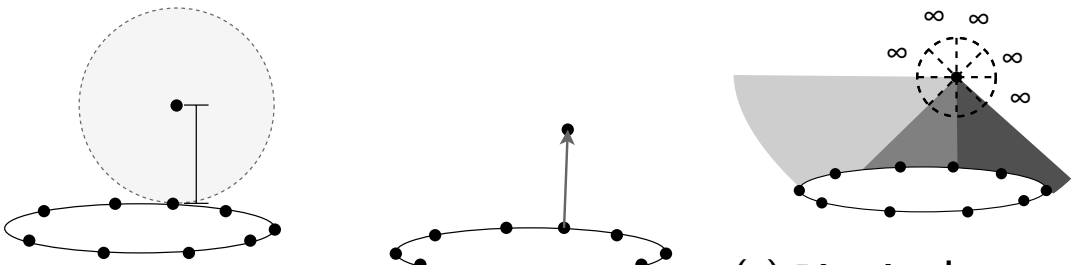


(a) CFD mesh with an airfoil body surrounded by different sizes of cells. (b) Illustration of cell characteristics, namely cell centroids, x_i and x_j, face centroid, $c_{f,ij}$, face area normal vector, $S_{f,ij}$, and cell volumes, V_i and V_j.

Contributions

1. Geometric features to inform GNN nodes of long-range interactions.
2. Finite-volume features in the graph convolution layer.
3. *Super-resolution:* exploited residual training w/ low-res simulation data to ease the learning.

Impact of this work



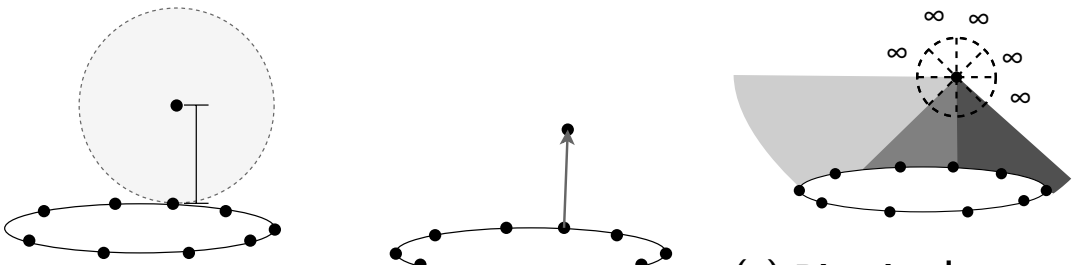
(a) **Signed Distance Function (SDF)**

(b) **Shortest Vector (SV)**

(c) **Directional Integrated Distance (DID)**

1. Existing geometric features (SDF) only indicate the presence of the closest boundary point somewhere along the circle's circumference. SV provides both distance and direction from the nearest boundary point. DID gives the average distance of all boundary points within a given angle range.
2. We showed that using cell characteristics, such as cell volume as node features, while face surface area, and face centroid as edge features improves the fidelity.
3. We were able to improve the fidelity of MeshGraphNet by 41% (and others).
4. Patent at UK IP office.

Impact of this work



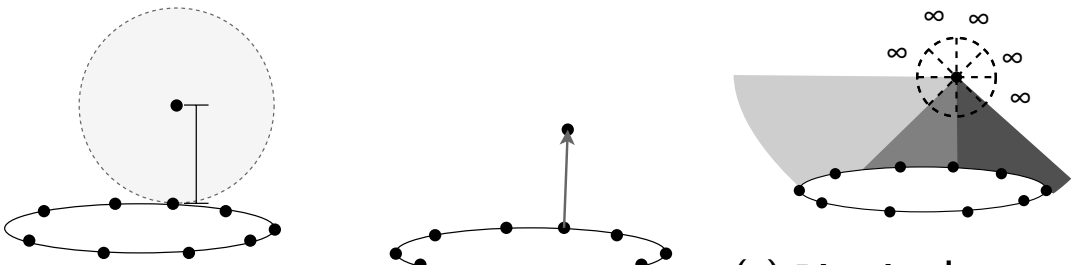
(a) **Signed Distance Function (SDF)**

(b) **Shortest Vector (SV)**

(c) **Directional Integrated Distance (DID)**

1. Existing geometric features (SDF) only indicate the presence of the closest boundary point somewhere along the circle's circumference. SV provides both distance and direction from the nearest boundary point. DID gives the average distance of all boundary points within a given angle range.
2. We showed that using cell characteristics, such as cell volume as node features, while face surface area, and face centroid as edge features improves the fidelity.
3. We were able to improve the fidelity of MeshGraphNet by 41% (and others).
4. Patent at UK IP office.

Impact of this work



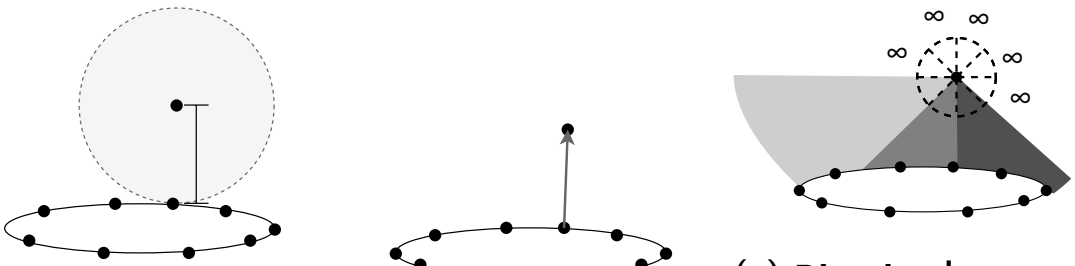
(a) **Signed Distance Function (SDF)**

(b) **Shortest Vector (SV)**

(c) **Directional Integrated Distance (DID)**

1. Existing geometric features (SDF) only indicate the presence of the closest boundary point somewhere along the circle's circumference. SV provides both distance and direction from the nearest boundary point. DID gives the average distance of all boundary points within a given angle range.
2. We showed that using cell characteristics, such as cell volume as node features, while face surface area, and face centroid as edge features improves the fidelity.
3. We were able to improve the fidelity of MeshGraphNet by 41% (and others).
4. Patent at UK IP office.

Impact of this work



(a) **Signed Distance Function (SDF)**

(b) **Shortest Vector (SV)**

(c) **Directional Integrated Distance (DID)**

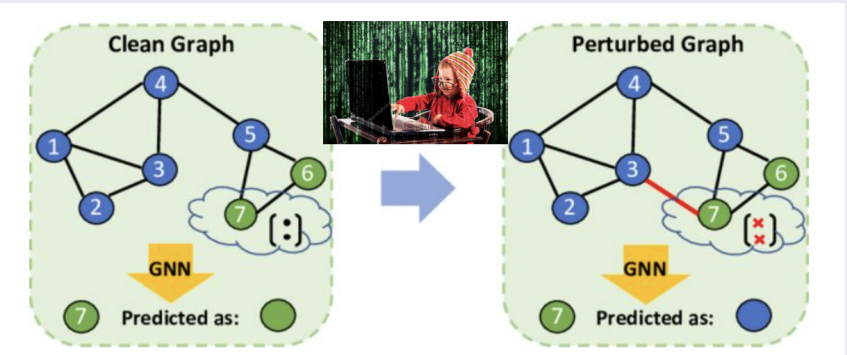
1. Existing geometric features (SDF) only indicate the presence of the closest boundary point somewhere along the circle's circumference. SV provides both distance and direction from the nearest boundary point. DID gives the average distance of all boundary points within a given angle range.
2. We showed that using cell characteristics, such as cell volume as node features, while face surface area, and face centroid as edge features improves the fidelity.
3. We were able to improve the fidelity of MeshGraphNet by 41% (and others).
4. Patent at UK IP office.

Adversarial attack: a Modern Challenge to GNNs

When Witnesses Defend: A Witness Graph Topological Layer for Adversarial Graph Learning. (AAAI'25)

Adversarial attack on Graph learning algorithms.

Attacker misleads a learning algorithm (e.g. GNN) into making incorrect predictions or classifications by deliberately perturbing a small number of edges (e.g. remove/add edges) or node features.



Adversarial perturbation (around target node 7) causes misclassification.

Contributions

1. We introduced a novel topological adversarial defense, namely, the *Witness Graph Topological Layer (WGTL)*.
2. WGTL integrates not only local but also global higher-order graph characteristics and controls their potential defense role via a topological regularizer.

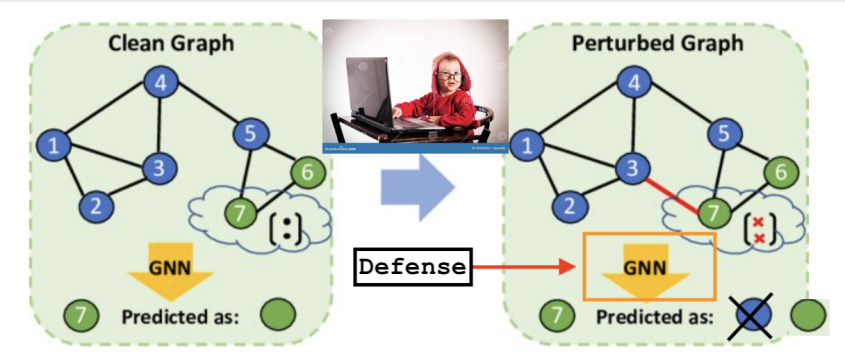
Adversarial attack: a Modern Challenge to GNNs

When Witnesses Defend: A Witness Graph Topological Layer for Adversarial Graph Learning. (AAAI'25)

Adversarial attack on Graph learning algorithms.

Attacker misleads a learning algorithm (e.g. GNN) into making incorrect predictions or classifications by deliberately perturbing a small number of edges (e.g. remove/add edges) or node features.

Problem



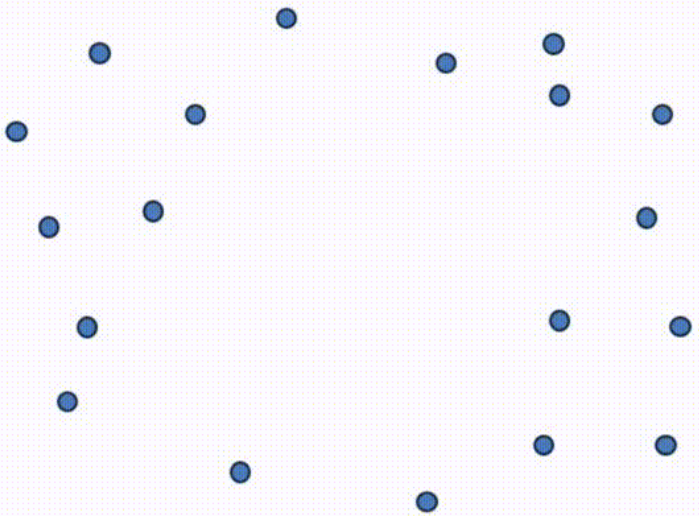
Design a defense algorithm that mitigates the effect of adversarial attack

Contributions

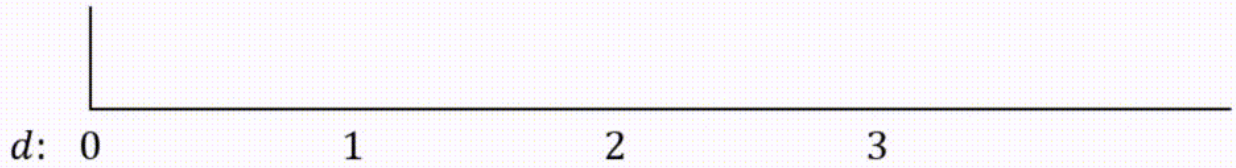
1. We introduced a novel topological adversarial defense, namely, the *Witness Graph Topological Layer (WGTL)*.
2. WGTL integrates not only local but also global higher-order graph characteristics and controls their potential defense role via a topological regularizer.

Topological Features

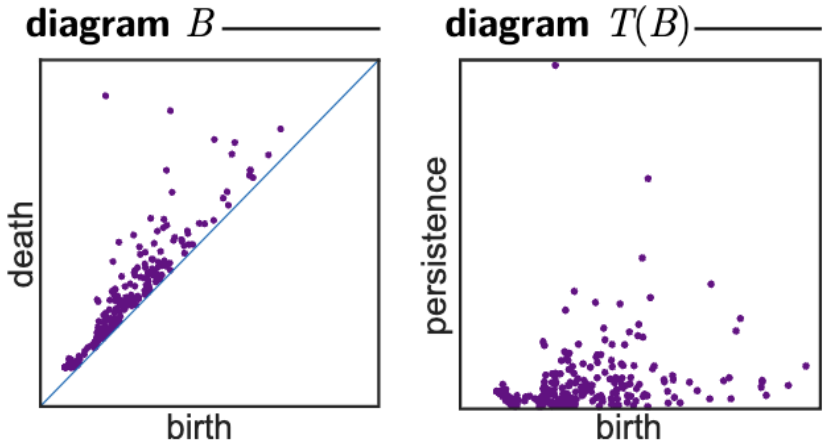
Example:



Record the barcode:



Topological Features

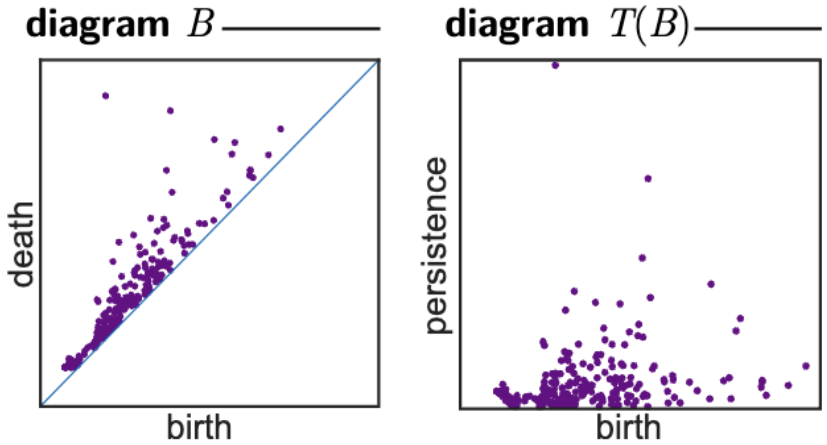


A persistence diagram is transformed using function $T : (x, y) \to (0, y - x)$.

Why Topological features?

Stability theorem: Small ($<= \epsilon$) change in the data (graph/points) only result in small ($<= \epsilon$) changes in the persistence diagram.

Topological Features



A persistence diagram is transformed using function $T : (x, y) \to (0, y - x)$.

Why Topological features?

Stability theorem: Small ($<= \epsilon$) change in the data (graph/points) only result in small ($<= \epsilon$) changes in the persistence diagram.

Impact of this work

1. This is the first work that shows that topological features can make GNNs robust against adversarial perturbations.
2. Effective against several types of attacks, for instance,
   - Targetted poisoning attack (Graybox, modify the neighbors of a target node and their features)
   - Global poisoning attack (Graybox, instead of targetting specific neighborhood modify whichever edges maximizes attackers objective)
   - Adaptive attacks (White-box, assumes that the model architecture, parameters and defense mechanisms are known to the attacker)
   - Node feature attack
3. WGTL improves existing defenses such as Pro-GNN, GNNGuard, and SimP-GCN respectively by 5%, 15%, and 5%.

Impact of this work

1. This is the first work that shows that topological features can make GNNs robust against adversarial perturbations.
2. Effective against several types of attacks, for instance,
   - Targetted poisoning attack (Graybox, modify the neighbors of a target node and their features)
   - Global poisoning attack (Graybox, instead of targetting specific neighborhood modify whichever edges maximizes attackers objective)
   - Adaptive attacks (White-box, assumes that the model architecture, parameters and defense mechanisms are known to the attacker)
   - Node feature attack
3. WGTL improves existing defenses such as Pro-GNN, GNNGuard, and SimP-GCN respectively by 5%, 15%, and 5%.

Impact of this work

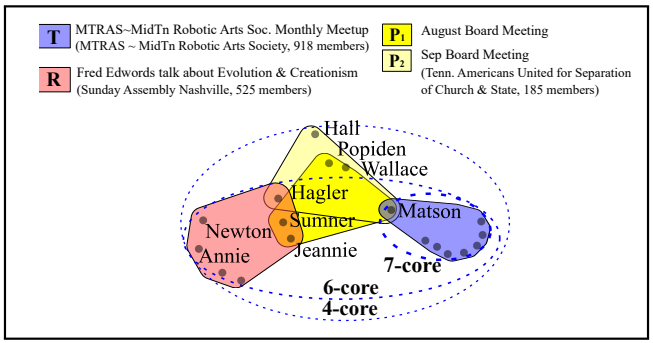
1. This is the first work that shows that topological features can make GNNs robust against adversarial perturbations.
2. Effective against several types of attacks, for instance,
   - Targetted poisoning attack (Graybox, modify the neighbors of a target node and their features)
   - Global poisoning attack (Graybox, instead of targetting specific neighborhood modify whichever edges maximizes attackers objective)
   - Adaptive attacks (White-box, assumes that the model architecture, parameters and defense mechanisms are known to the attacker)
   - Node feature attack
3. WGTL improves existing defenses such as Pro-GNN, GNNGuard, and SimP-GCN respectively by 5%, 15%, and 5%.

Hypergraphs: a modern paradigm in higher-order graph data mining

Neighborhood based hypergraph core decomposition (pVLDB 2023).

Neighborhood-based core decomposition

Decomposition of a hypergraph into nested, maximal subhypergraphs/cores such that all nodes in the k-core have at least k neighbors in that subhypergraph.



The set of events $H = \{T, R, P_1, P_2\}$ forms a hypergraph. 6-core => $\{T, R\}$, 7-core => $\{T\}$

Contributions

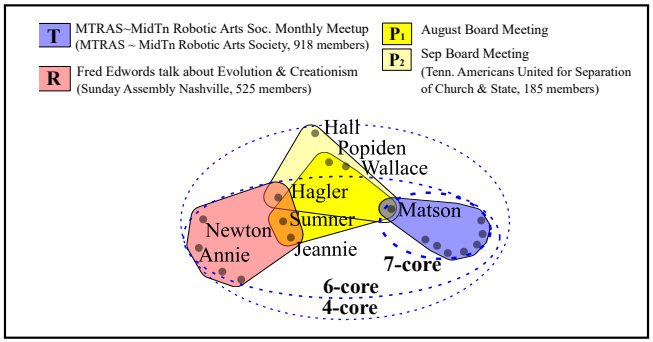
- We introduced this novel core decomposition.
- We proposed an efficient local algorithm that scales to million-node hypergraph.
- **Applications**:
  - **Densest subhypergraph extraction.** Our novel volume-densest subhypergraphs capture important meetup events.
  - **Diffusion intervention.** Our decomposition is more effective than other graph-based decompositions in intervening diffusion e.g. epidemic spread.

Hypergraphs: a modern paradigm in higher-order graph data mining

Neighborhood based hypergraph core decomposition (pVLDB 2023).

Neighborhood-based core decomposition

Decomposition of a hypergraph into nested, maximal subhypergraphs/cores such that all nodes in the k-core have at least k neighbors in that subhypergraph.



The set of events $H = \{T, R, P_1, P_2\}$ forms a hypergraph. 6-core => $\{T, R\}$, 7-core => $\{T\}$

Contributions

- We introduced this novel core decomposition.
- We proposed an efficient local algorithm that scales to million-node hypergraph.
- **Applications**:
  - **Densest subhypergraph extraction.** Our novel volume-densest subhypergraphs capture important meetup events.
  - **Diffusion intervention.** Our decomposition is more effective than other graph-based decompositions in intervening diffusion e.g. epidemic spread.

Impact of this work

1. Fastest hypergraph core-decomposition algorithm to date: Can decompose ArnetMiner hypergraph with 27M nodes and 17M hyperedges in-memory within 91 seconds.
2. Laid the foundation for more recent core decomposition methods such as (k,g)-core [CIKM'23], (k,t)-hypercore [ECML-PKDD'23], and Dual-Layer Hierarchy [CIKM'24].

(Brief) Future Plans

Problems I want to study and relevant impact areas:

1. Cyber-security.
   - **Privacy-preserving graph neural networks** that are robust against adversaries.
   - **Adversarial robustness of Higher-order graph (hypergraph) learning.**
   - Efficient **uncertainty estimation of GNNs.** We want increased trustworthiness of AI systems by providing transparency about model confidence. In cybersecurity, uncertainty estimation can help prioritize high-risk alerts for further investigation.
2. Human-centered AI.
   - **Inconsistencies and bias in reasoning ability of LLMs.** Can we develop benchmark datasets to assess the deductive reasoning ability of LLMs?
   - How to model rich datasets e.g. those arising from **multiple modalities and multiple sources** using a **Knowledge Hypergraph**. How to handle complex queries on such knowledge hypergraph?
3. Physical science.
   - **Explainability and reliability** of ML models (particularly GNNs) in the **physics domain** e.g. fluid dynamics, additive manufacturing etc.

(Brief) Future Plans

Problems I want to study and relevant impact areas:

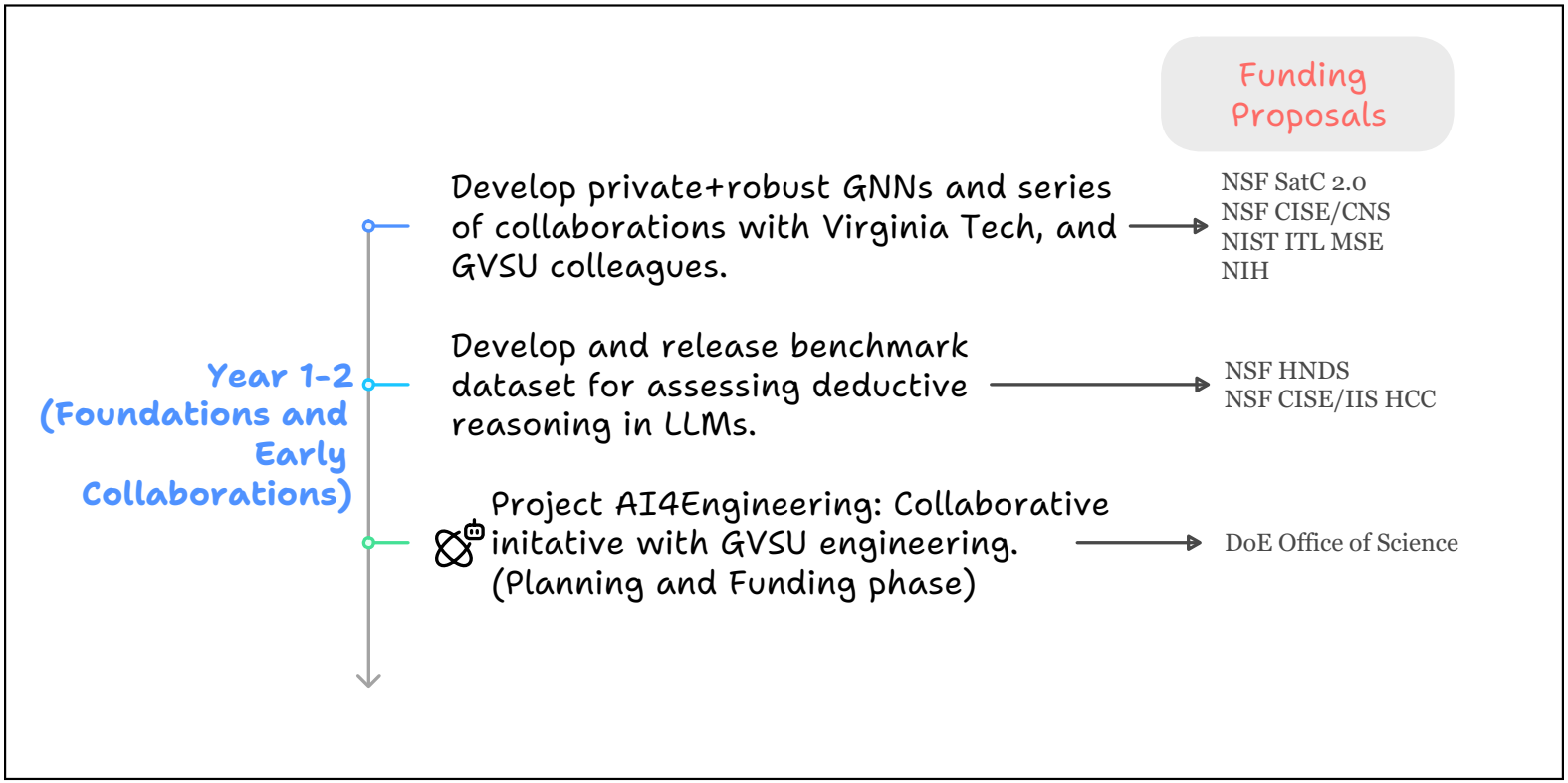
1. Cyber-security.
   - **Privacy-preserving graph neural networks** that are robust against adversaries.
   - **Adversarial robustness of Higher-order graph (hypergraph) learning.**
   - Efficient **uncertainty estimation of GNNs.** We want increased trustworthiness of AI systems by providing transparency about model confidence. In cybersecurity, uncertainty estimation can help prioritize high-risk alerts for further investigation.
2. Human-centered AI.
   - **Inconsistencies and bias in reasoning ability of LLMs.** Can we develop benchmark datasets to assess the deductive reasoning ability of LLMs?
   - How to model rich datasets e.g. those arising from **multiple modalities and multiple sources** using a **Knowledge Hypergraph**. How to handle complex queries on such knowledge hypergraph?
3. Physical science.
   - **Explainability and reliability** of ML models (particularly GNNs) in the **physics domain** e.g. fluid dynamics, additive manufacturing etc.

(Brief) Future Plans

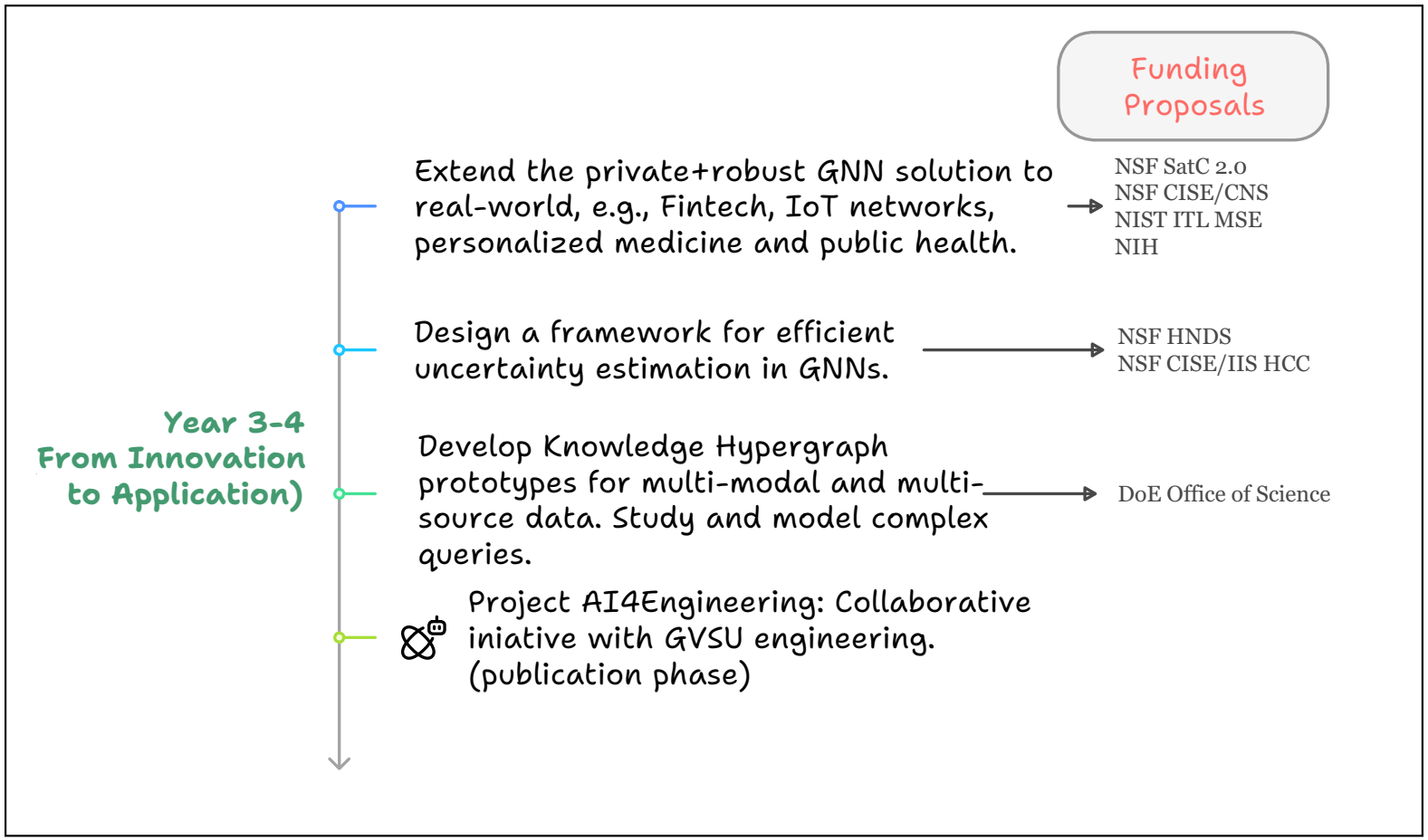
Problems I want to study and relevant impact areas:

1. Cyber-security.
   - **Privacy-preserving graph neural networks** that are robust against adversaries.
   - **Adversarial robustness of Higher-order graph (hypergraph) learning.**
   - Efficient **uncertainty estimation of GNNs.** We want increased trustworthiness of AI systems by providing transparency about model confidence. In cybersecurity, uncertainty estimation can help prioritize high-risk alerts for further investigation.
2. Human-centered AI.
   - **Inconsistencies and bias in reasoning ability of LLMs.** Can we develop benchmark datasets to assess the deductive reasoning ability of LLMs?
   - How to model rich datasets e.g. those arising from **multiple modalities and multiple sources** using a **Knowledge Hypergraph**. How to handle complex queries on such knowledge hypergraph?
3. Physical science.
   - **Explainability and reliability** of ML models (particularly GNNs) in the **physics domain** e.g. fluid dynamics, additive manufacturing etc.

Short term (years 1-2)



Mid term (years 3-4)



Long term (year 5)

Funding Proposals

Year 5 (Expansion and Long-term impact)

- Implement Uncertainty Estimation methods in collaboration with cybersecurity industry partners.
- Secure a multi-institutional grant for Graph Learning for cyber-security and Uncertainty estimation in GNNs. Organize and host a workshop. → To be decided
- Deploy Knowledge Hypergraph models in interdisciplinary applications (e.g., healthcare).
- Project AI4Engineering: Collaborative initiative with GVSU engineering. (Extend the scope to other disciplines) → To be decided

Interdisciplinary Works and Collaborations

Interdisciplinary collaboration

- **Rolls-Royce plc.**, Dept. of Mechanical & Aerospace Engineering (NTU)
  - 1 Patent @UK IP office, 1 ICML paper.
  - Rolls-Royce has plans to internally adopt our solution.

Collaborations across the world

- *North-America*:
  - **Virginia Tech**
  - **Purdue University**
  - **University of California Riverside**
  - **Pacific Northwest National Lab (PNNL)**
- *Europe*:
  - Aalborg University, **Denmark**
  - Inria @ Univ. of. Lile, **France**
  - University of Vienna, **Austria**
  - CENTAI, **Italy**
  - Max Planck Institute, **Germany**
- *Asia*: NUS, NTU (Singapore), BUET (Bangladesh).

Interdisciplinary Works and Collaborations

Interdisciplinary collaboration

- **Rolls-Royce plc.**, Dept. of Mechanical & Aerospace Engineering (NTU)
  - 1 Patent @UK IP office, 1 ICML paper.
  - Rolls-Royce has plans to internally adopt our solution.

Collaborations across the world

- *North-America*:
  - **Virginia Tech**
  - **Purdue University**
  - **University of California Riverside**
  - **Pacific Northwest National Lab (PNNL)**
- *Europe*:
  - Aalborg University, **Denmark**
  - Inria @ Univ. of. Lile, **France**
  - University of Vienna, **Austria**
  - CENTAI, **Italy**
  - Max Planck Institute, **Germany**
- *Asia*: NUS, NTU (Singapore), BUET (Bangladesh).

Mentoring

I have had the pleasure to mentor and collaborate with the following young researchers:

- **PhD students:**
  - Siddhartha Shankar Das, PhD student@Purdue University (2024-) (To join PNNL as post-doc next summer)
  - Bishwamittra Ghosh, PhD student@NUS (2021-2023)(Now post-doc at Max Planck Institute, Germany)
  - Ehsan B. Mobaraki, PhD student@AAU (2023-2024)
  - Sarah Hasan, PhD student@AAU (2023-2024)
  - Loh Sher En Jessica, PhD student@NTU (2021-2024)
  - Debabrata Mahapatra, PhD student@NUS (2019)
- **Undergraduate student:**
  - Arpit Kumar Rai, IIT Kanpur, Intern @NTU (2021-2022) (Now Software Engineer @ Glean, Palo Alto, California)

Thank You

Q&A

Supplementary Slides

CFD+AI

Step 1. Modeling the laws of fluid flow (Physics) with Equations.

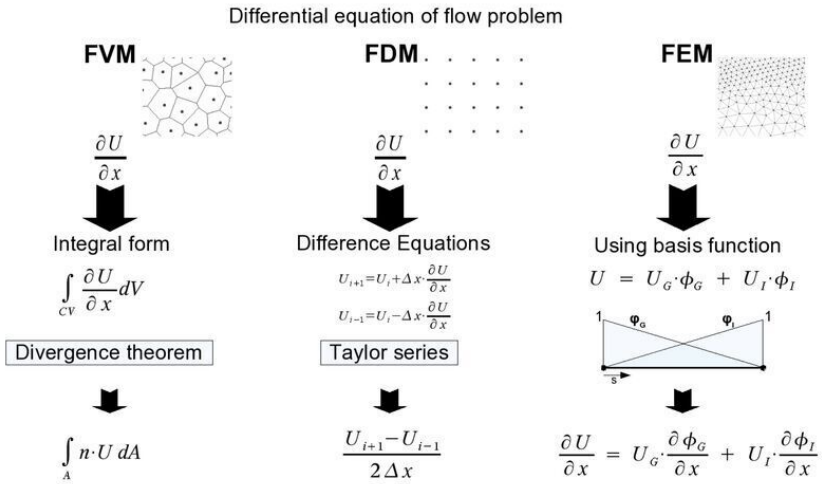
$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\varrho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$
$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\varrho}\frac{\partial p}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right)$$
$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = -\varrho\left(\frac{\partial u}{\partial x}\frac{\partial u}{\partial x} + 2\frac{\partial u}{\partial y}\frac{\partial v}{\partial x} + \frac{\partial v}{\partial y}\frac{\partial v}{\partial y}\right)$$

2D Navier-Stokes Equation

Step 2. Conditions: Prescribe the fluid properties (e.g. density, viscosity), boundary conditions of the domain, and initial conditions for the flow.

Traditional (Numerical) Fluid Simulation (Contd.)

Step 3. Discretization: To solve these equations, the fluid domain is divided into a large number of small regions (grids or control volumes), and the equations are solved numerically for each region.



Various methods of approximating the terms in governing PDEs.

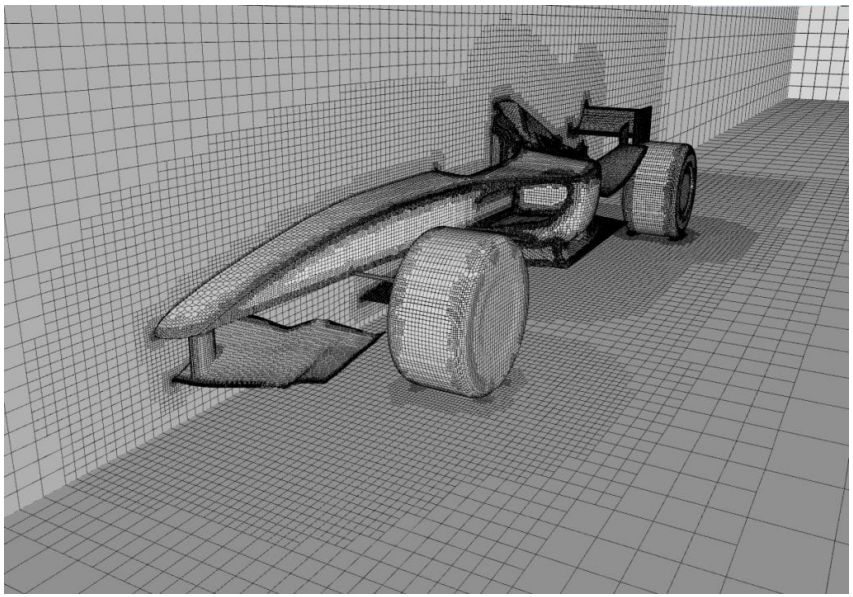
For instance, FDM linearizes the first equation at step 1

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} + u_{i,j}^{n} \frac{u_{i,j}^{n} - u_{i-1,j}^{n}}{\Delta x} + v_{i,j}^{n} \frac{u_{i,j}^{n} - u_{i,j-1}^{n}}{\Delta y} =$$
$$- \frac{1}{\varrho} \frac{p_{i+1,j}^{n} - p_{i-1,j}^{n}}{2\Delta x} + \nu \left(\frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}}{\Delta x^2} + \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{\Delta y^2} \right)$$

Step 4. Solve the system of equations using numerical solvers (e.g. Gauss-Siedel).

Why Numerical Solvers are expensive?

1. Ensuring **numerical stability and accuracy** often requires fine grids and small time steps, which leads to solving millions or even billions of coupled equations iteratively.
2. Real-world objects are complex with many **complicated parts design**.



Given node feature $h_i \in \mathcal{R}^{d_{in}}$ at the i^{th} node and its spatial location $\boldsymbol{x}_i \in \mathbb{R}^t$, a convolution using FVF is defined as

$$\bar{h}_i(U, b, N_i) = \sum_{j \in N_i} ReLU(U^T(q_{ij}) + b) \odot (h_j \oplus p_j) \,, \tag{1}$$

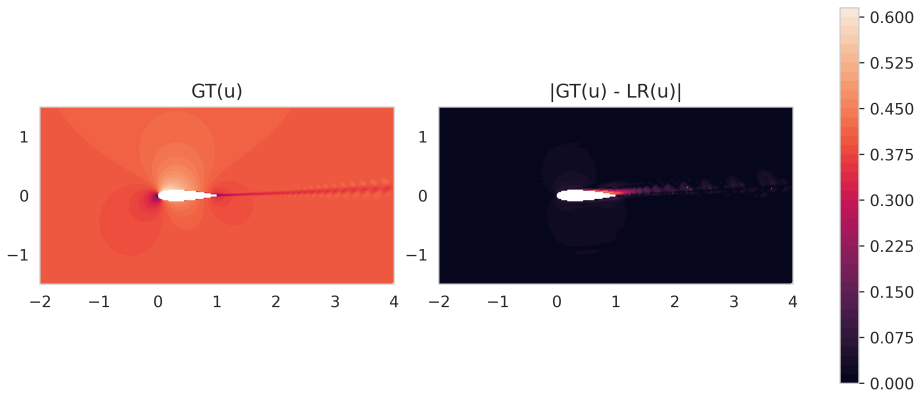
where $U \in \mathbb{R}^{3t \times (d_{in}+1)}$ and $b \in \mathbb{R}^{d_{in}+1}$ are trainable parameters, t is the spatial dimension of the CFD simulation, and N_i is an index set indicating the neighbourhood of the node i and $\odot$ is the element-wise multiplication.

In our model, the node attributes are its associated cell volume, denoted as $p_j = V_j \in \mathbb{R}^1$, and the edge attributes are its associated face area normal vector and the relative spatial location of its face centroid to the nodes, denoted as

$$q_{ij} = \mathbf{S}_{f,ij} \oplus (\boldsymbol{c}_{f,ij} - \boldsymbol{x}_i) \oplus (\boldsymbol{c}_{f,ij} - \boldsymbol{x}_j) \in \mathbb{R}^{3t}$$

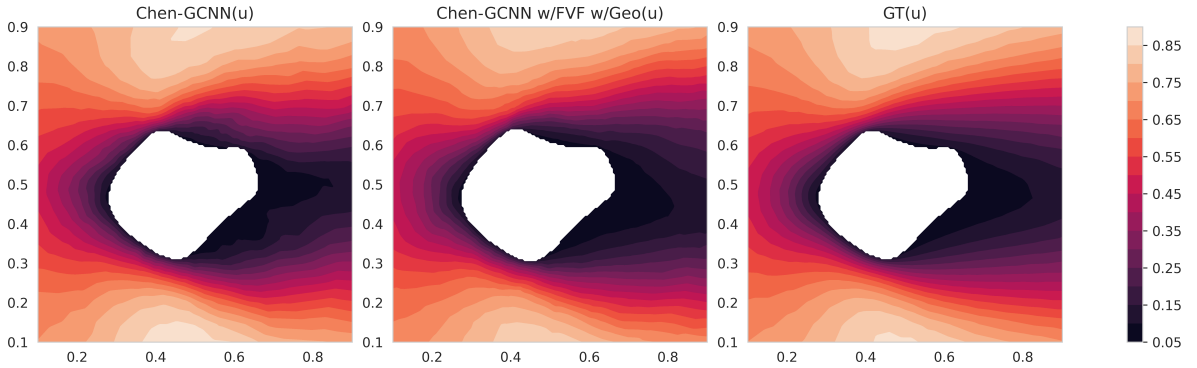
Residual training

- Train the network to predict the residual field (GT - LR), where LR is the upsampled low-resolution field, instead of the original field GT itself.
- Much of the residual field will be close to zero, which
  - eases the learning,
  - helps the model to focus on areas where the LR fields tend to be inaccurate



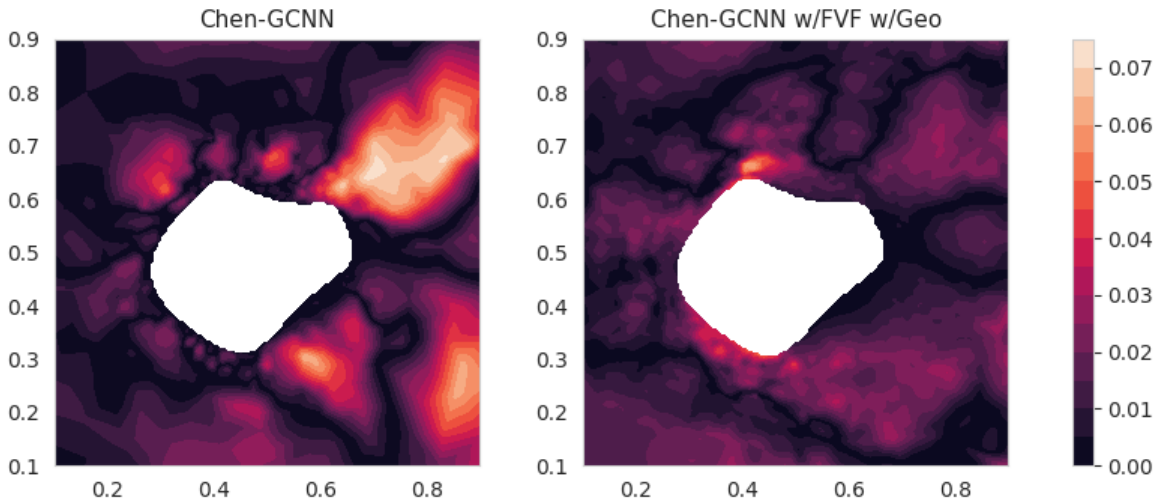
Prediction on an airfoil shows that the velocity x-component, u remains invariant in majority regions (left), hence the absolute value of residual ($|GT(u) - \text{Upsample}(LR(u))|$) is close to 0 in those regions (right).

Some visualisations



(a) **Velocity x-component (u) visualisation.**

$|\hat{U} - U|$



(b) **Absolute error of predictions.**

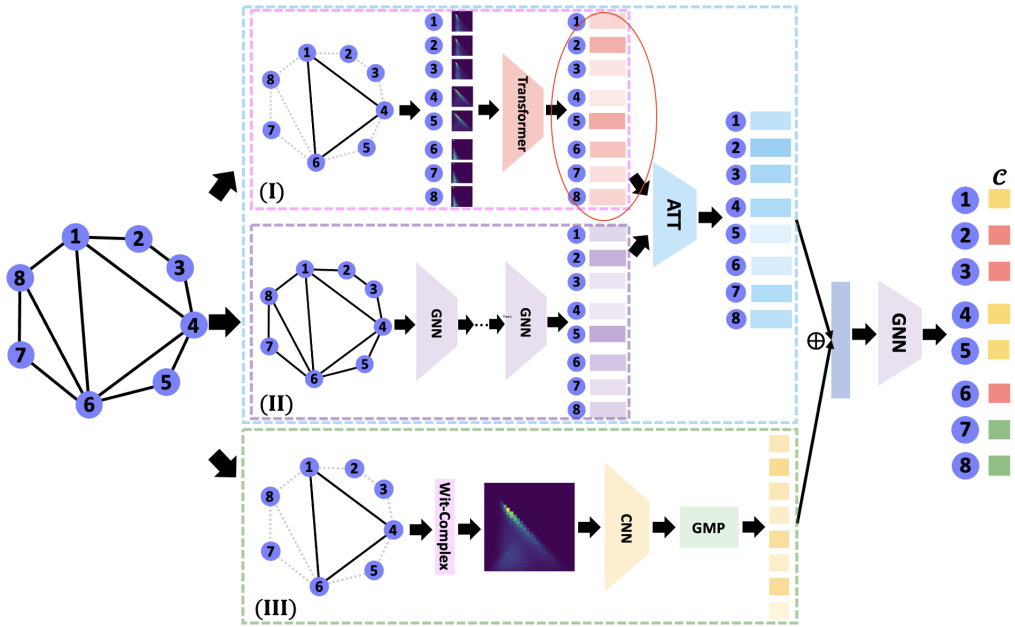
Chen-GCNN w/ FVF w/ Geo has a comparatively smaller high-error ($[0.050, 0.075]$) region than the baseline. Darkest shade = low-error region.

Efficiency of Geometric feature computations.

	Average Computation Time (s)	
	Coarse-AirfRANS	AirfRANS
SV	0.001	0.018
DID	0.031	4.276
FVF	0.001	0.032

Computational time of SV, DID and FVF at two mesh resolutions. Airfoils in Coarse-AirfRANS has $\sim$20 times fewer nodes than AirfRANS.

Adversarial robustness



Architecture of Witness Graph Topological Layer.

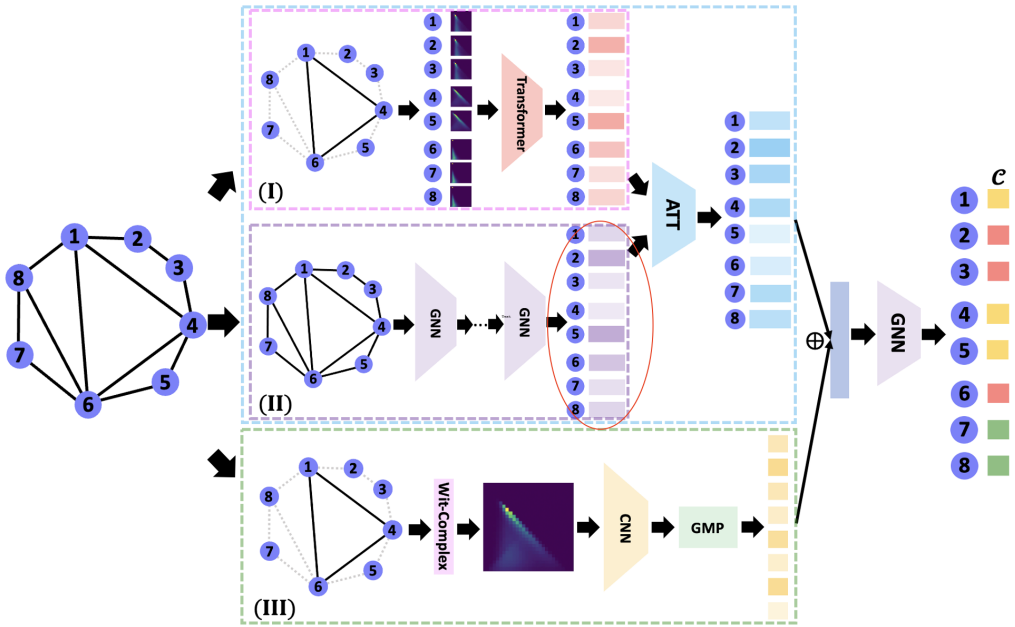
1. **Local Topology Encoding:** Encodes local topological features of every node. (Z_{T_L})
2. Node Representation Learning. Learns node representations using any backbone GNN. (Z_G)
3. Global Topology Encoding. Encodes topological feature of the entire graph. (Z_{T_G})
4. Aggregated Topological Encoding. Encodes local and global topological priors. (Z_{WGTL})

$$z = [Z_{T_L}, Z_G]$$

Attention coefficients, $\alpha_i = \mathbf{Softmax}(W_2 \cdot \tanh(W_1 z_i + b_1))$

Additive attention, $Z_{\mathrm{AGG}} = \alpha_1 \times Z_{T_L} + \alpha_2 \times Z_G$

$$Z_{\mathrm{WGTL}} = Z_{\mathrm{AGG}} Z_{T_G}$$



Architecture of Witness Graph Topological Layer.

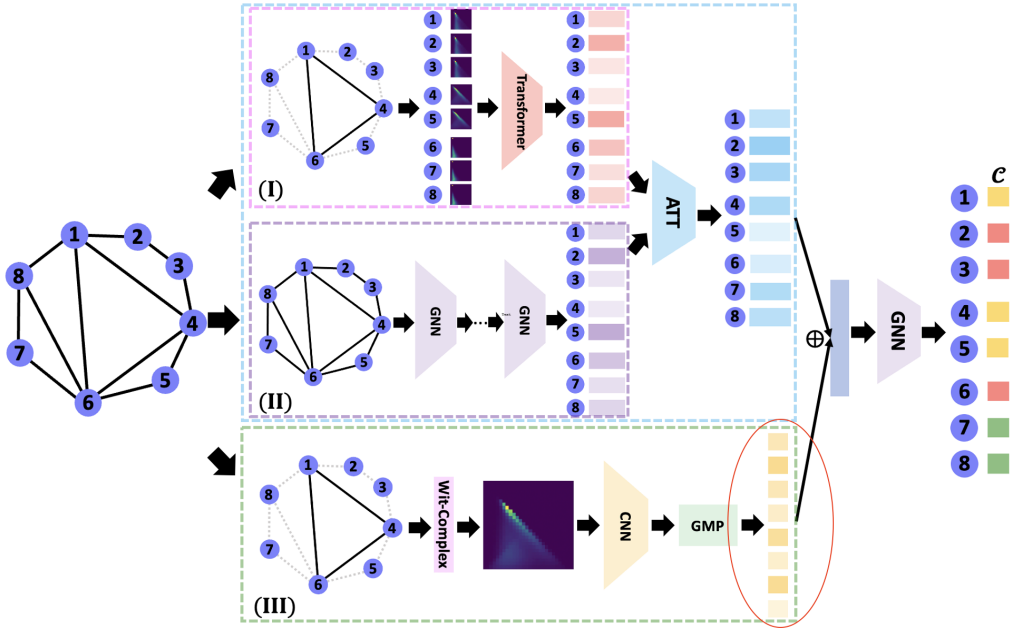
1. **Local Topology Encoding:** Encodes local topological features of every node. (Z_{T_L})
2. **Node Representation Learning.** Learns node representations using any backbone GNN. (Z_G)
3. Global Topology Encoding. Encodes topological feature of the entire graph. (Z_{T_G})
4. Aggregated Topological Encoding. Encodes local and global topological priors. (Z_{WGTL})

$$z = [Z_{T_L}, Z_G]$$

Attention coefficients, $\alpha_i = \mathbf{Softmax}(W_2 \cdot \tanh(W_1 z_i + b_1))$

Additive attention, $Z_{\mathrm{AGG}} = \alpha_1 \times Z_{T_L} + \alpha_2 \times Z_G$

$$Z_{\mathrm{WGTL}} = Z_{\mathrm{AGG}} Z_{T_G}$$



Architecture of Witness Graph Topological Layer.

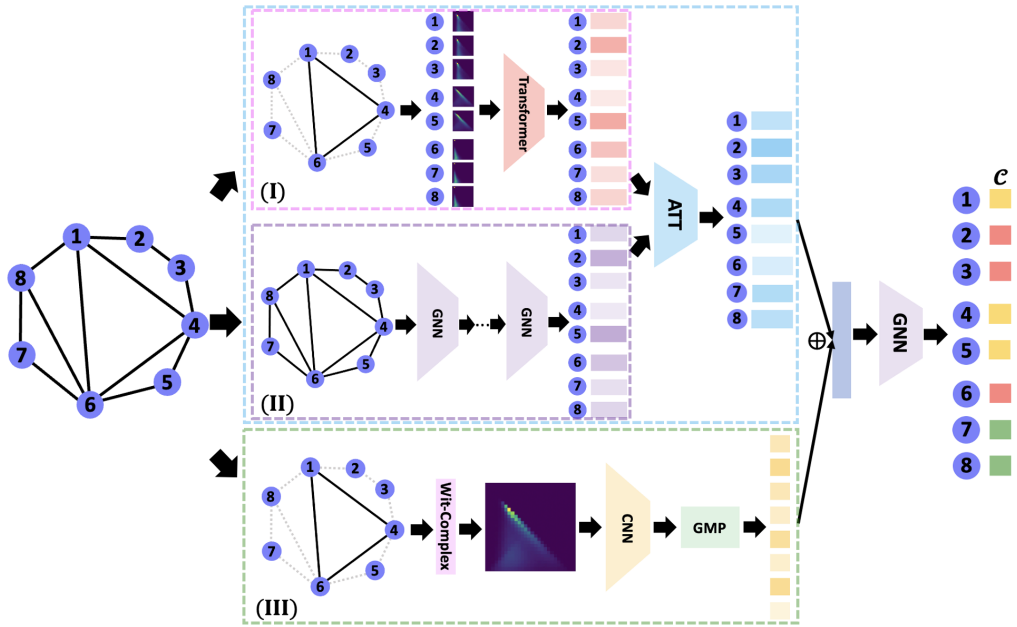
1. **Local Topology Encoding:** Encodes local topological features of every node. (Z_{T_L})
2. **Node Representation Learning.** Learns node representations using any backbone GNN. (Z_G)
3. **Global Topology Encoding.** Encodes topological feature of the entire graph. (Z_{T_G})
4. **Aggregated Topological Encoding.** Encodes local and global topological priors. (Z_{WGTL})

$$z = [Z_{T_L}, Z_G]$$

$$\text{Attention coefficients, } \alpha_i = \mathbf{Softmax}(W_2 \cdot \tanh(W_1 z_i + b_1))$$

$$\text{Additive attention, } Z_{\mathrm{AGG}} = \alpha_1 \times Z_{T_L} + \alpha_2 \times Z_G$$

$$Z_{\mathrm{WGTL}} = Z_{\mathrm{AGG}} Z_{T_G}$$



Architecture of Witness Graph Topological Layer.

1. **Local Topology Encoding:** Encodes local topological features of every node. (Z_{T_L})
2. **Node Representation Learning.** Learns node representations using any backbone GNN. (Z_G)
3. **Global Topology Encoding.** Encodes topological feature of the entire graph. (Z_{T_G})
4. **Aggregated Topological Encoding.** Encodes local and global topological priors. (Z_{WGTL})

$$z = [Z_{T_L}, Z_G]$$

Attention coefficients, $\alpha_i = \mathbf{Softmax}(\boldsymbol{W_2} \cdot \tanh(\boldsymbol{W_1} z_i + \boldsymbol{b_1}))$

Additive attention, $Z_{\mathrm{AGG}} = \alpha_1 \times Z_{T_L} + \alpha_2 \times Z_G$

$$Z_{\mathrm{WGTL}} = Z_{\mathrm{AGG}} Z_{T_G}$$

WGTL: Topological Regularizer

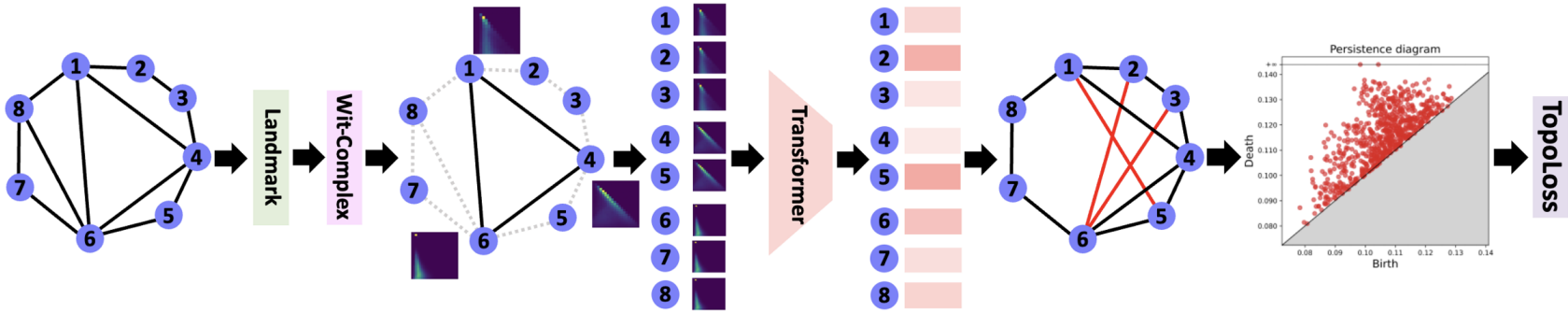


Illustration of Witness Complex-based topological regularizer L_{Topo}.

$$L_{topo}(\mathrm{T}(\mathcal{G})) \triangleq \sum_{i=1}^{m}(d_i - b_i)^2 \left(\frac{d_i + b_i}{2}\right)^2, \tag{2}$$

- A localized attack (perturbing certain nodes or edges) appears as topological noise in the final persistent diagram, and exhibit lower persistence.
- And minimising L_{topo} forces the Transformer to learn local topology encodings (Z_{T_L}) which produces PD with small persistence, i.e., $(d_i - b_i)$.

WGTL: Topological Regularizer

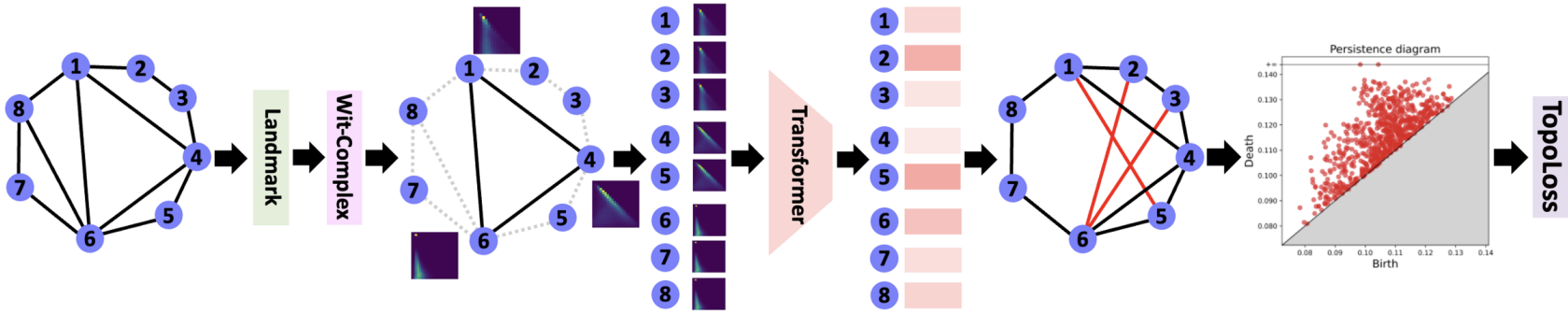


Illustration of Witness Complex-based topological regularizer L_{Topo}.

$$L_{topo}(\mathrm{T}(\mathcal{G})) \triangleq \sum_{i=1}^{m}(d_i - b_i)^2 \left(\frac{d_i + b_i}{2}\right)^2, \tag{2}$$

- A localized attack (perturbing certain nodes or edges) appears as topological noise in the final persistent diagram, and exhibit lower persistence.
- And minimising L_{topo} forces the Transformer to learn local topology encodings (Z_{T_L}) which produces PD with small persistence, i.e., $(d_i - b_i)$.

Table 1: Comparison of performances (avg. accuracy±std.) with existing defenses under mettack.

Dataset	Models	Perturbation Rate		
		0%	5%	10%
Cora-ML	Pro-GNN	82.98±0.23	80.14±1.34	71.59±1.33
	Pro-GNN+WGTL	**83.85±0.38**	**81.90±0.73**	**72.51±0.76**
	GCN+GNNGuard	83.21±0.34	76.57±0.50	69.13±0.77
	GCN+GNNGuard+WGTL	*__84.78±0.43__	*__83.23±0.82__	*__79.96±0.49__
	SimP-GCN	79.52±1.81	74.75±1.40	70.87±1.70
	SimP-GCN+WGTL	**81.49±0.52**	**76.65±0.65**	**72.88±0.83**
Citeseer	ProGNN	72.34±0.99	68.96±0.67	67.36±1.12
	ProGNN+WGTL	**72.83±0.94**	**71.85±0.74**	**70.70±0.57**
	GCN+GNNGuard	71.82±0.43	70.79±0.22	66.86±0.54
	GCN+GNNGuard+WGTL	**73.37±0.63**	**72.57±0.17**	**66.93±0.21**
	SimP-GCN	73.73±1.54	73.06±2.09	72.51±1.25
	SimP-GCN+WGTL	*__74.32±0.19__	*__74.05±0.71__	*__73.09±0.50__
Pubmed	Pro-GNN	87.33±0.18	87.25±0.09	87.20±0.12
	Pro-GNN + WGTL (ours)	**87.90±0.30**	*__87.77±0.08__	*__87.67±0.22__
	GCN+GNNGuard	83.63±0.08	79.02±0.14	76.58±0.16
	GCN+GNNGuard+WGTL	OOM	OOM	OOM
	SimP-GCN	*88.11±0.10	86.98±0.19	86.30±0.28
	SimP-GCN+WGTL	OOM	OOM	OOM
Polblogs	GCN+GNNGuard	95.03±0.25	73.25 ±0.16	72.76±0.75
	GCN+GNNGuard+WGTL	*__96.22±0.25__	*__73.62±0.22__	*__73.72±1.00__
	SimP-GCN	89.78±6.47	65.75±5.03	61.53±6.41
	SimP-GCN+WGTL	**94.56±0.24**	**69.78±4.10**	**69.55±4.42**

Table 2: Efficiency of WGTL. All the times are in seconds.

Datasets/ (# Landmarks)	Landmark selection time	Local feat. comput. time	Global feat. comput. time
Cora-ML/124	0.01±0.01	0.12±0.03	5.11±0.13
Citeseer/105	0.01±0.01	0.16±0.02	5.23±1.22
Polblogs/61	0.01±0.00	0.07±0.01	4.64±0.2
Snap-patents/91	0.03±0.02	0.64±0.00	7.54±1.15
Pubmed/394	0.07±0.01	0.51±0.03	27.83±0.47
OGBN-arXiv/84	1.02 ±0.00	12.79±0.31	83.04±2.19

Hypergraphs

Hypergraph

A **hypergraph** (V, E) consists of a set of nodes V and a collection of subsets of nodes E called *hyperedges*. Unlike edges in a graph, hyperedge may contain more than 2 nodes.
Examples: co-authorship in papers, event-participant relations in meet-ups, etc.

Neighbors

Pair of nodes that co-occur in a hyperedge are neighbours.

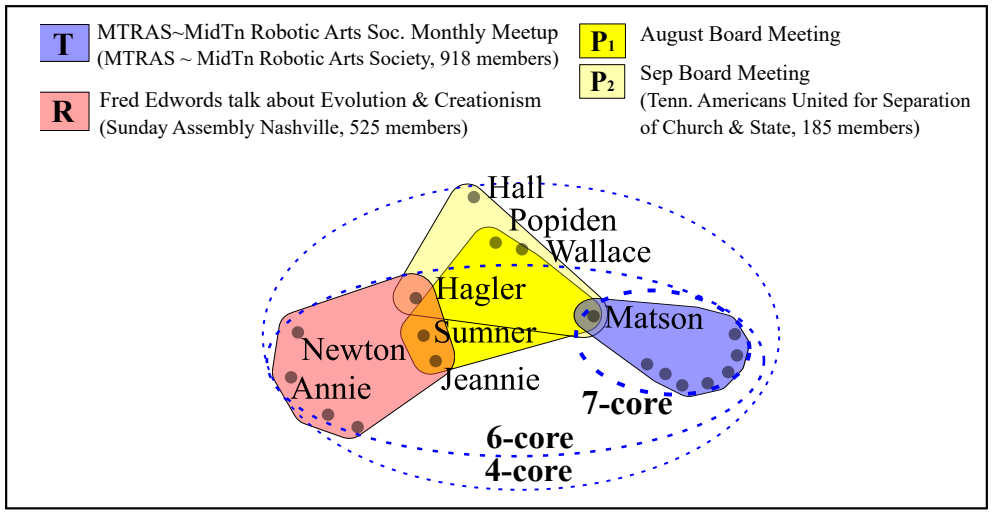


Figure 3: The set of events $H = \{T, R, P_1, P_2\}$ forms a hypergraph. *Annie* and *Newton* are neighbors. *Newton* has 6 neighbours.

Limitations of existing methods.

Hypergraph Degree-based decomposition may not be informative

Reduced Hypergraph Reducing to Clique graph and bipartite graph and then applying graph core-decompositions produces non-intuitive results.

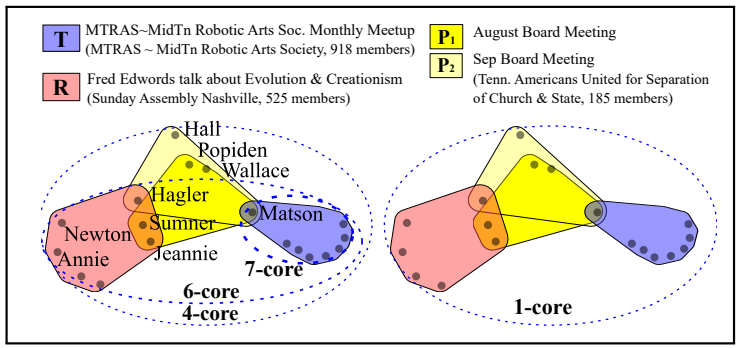


Figure 4: (left) Neighborhood-based and (right) degree-based core decomposition of a hypergraph H

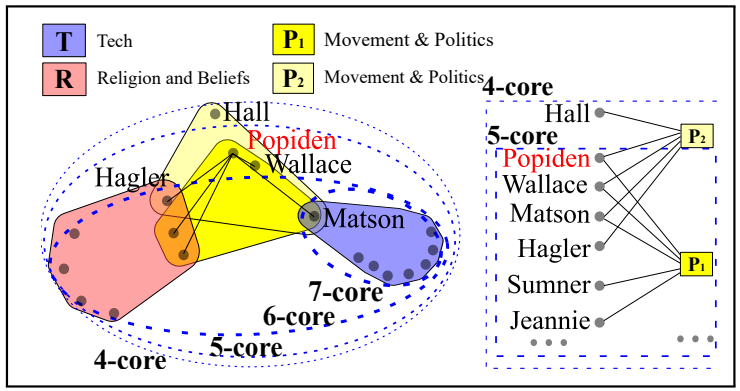


Figure 5: Alternative decompositions (a) Core decomposition of clique graph of H and (b) Dist-2 core decomposition of the bipartite graph of H. **Non-intuitiveness:** Similar events (P_1 and P_2) in different cores.

Challenges

Peeling paradigm In the classic peeling algorithm for graph, a node removal reduces its neighbors' degree by 1 (Linear time algorithm). However, in a neighborhood-based hypergraph core decomposition, its neighboring nodes # neighbors may reduce by more than 1 (Polynomial time).

Local algorithm paradigm Graph h-index reports incorrect neighborhood-based core.

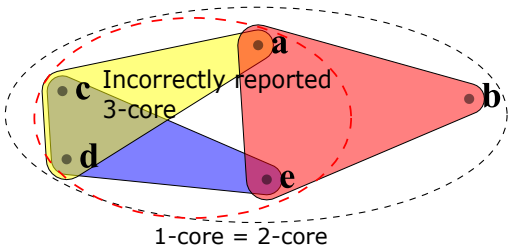


Figure 6: For any $n > 1$, the h-index of node a never reduces from $h_a^{(1)} = \mathcal{H}(2, 3, 3, 4) = 3$ to its correct core-number 2. Because a will always have at least 3 neighbors (c, d, and e) whose h-indices are at least 3. An incorrect 3-core reported.

Problem Statement

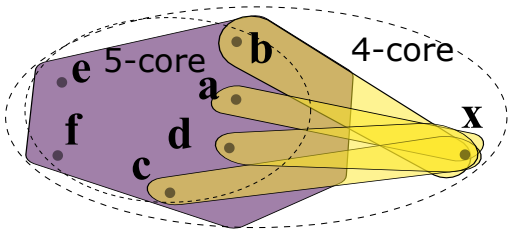
How to correctly and efficiently compute neighborhood-based hypergraph cores.

Naive Peeling algorithm: **Peel**

1. At each iteration $k \in \{1, 2, \cdots, |V|\}$,
   1. Remove the node with # neighbors $\leq k$.
   2. Report k as the core-number of the removed node.
   3. Recompute the #neighbors of neighboring nodes.
2. Complexity: $\mathcal{O}(|V| . d_{nbr} . (d_{nbr} + d_{hpe})$, here d_{nbr} (d_{hpe}) is the #neighbor (degree) of the node with largest #neighbors (degree).

Can we do better?.

Delay # neighbors recomputation of nodes with core-number $> k$ based on lower-bound.



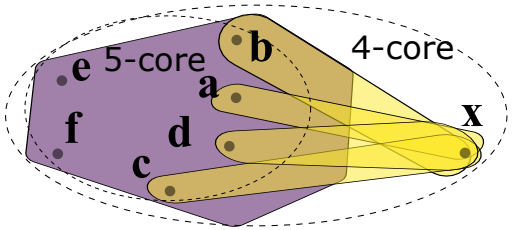
Node b's #neighbors is recomputed twice: (1) when x is peeled and later (2) when e is peeled. Can we delay the recomputation until e is peeled?

Better Peeling algorithm: **E-Peel**

1. Compute the core-number lower bound for all nodes.
2. At each iteration $k \in \{1, 2, \cdots, |V|\}$,
   1. Remove the node with # neighbors $\leq k$.
   2. Report k as the core-number of the removed node.
   3. Recompute the #neighbors of a neighboring node v only if $k >= LB(v)$.

$$LB(v) = \max\left(|e_m(v)| - 1, \min_{u \in V} |N(u)|\right)$$

Here $e_m(v)$ is the maximal cardinality hyperedge containing v



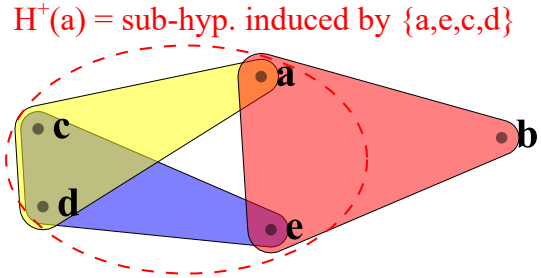
Node b's #neighbors computation is delayed until e is peeled.

Best algorithm: Local core and optimisations

Input: Hypergraph $H = (V, E)$
Output: Core-number $c(v)$ for each node $v \in V$

for all $v \in V$ **do**
  $\hat{h}_v^{(\mathbf{0})} = h_v^{(\mathbf{0})} \leftarrow |N(v)|$.
for all $n = 1, 2, \ldots, \infty$ **do**
  for all $v \in V$ **do**
    $h_v^{(n)} \leftarrow \min\left(\mathcal{H}(\{\hat{h}_u^{(n-\mathbf{1})} : u \in N(v)\}), \hat{h}_v^{(n-\mathbf{1})}\right)$
  for all $v \in V$ **do**
    $c(v) \leftarrow \hat{h}_v^{(n)} \leftarrow$ **Core-correction** $(v, h_v^{(n)}, H)$
  if $\forall v, \hat{h}_v^{(n)} == h_v^{(n)}$ **then**
    Terminate Loop
Return c

Core correction:



Reduce h-index h_a by 1 until the #neighbors of a in $H^+(a) \geq h_a$: Node a's corrected h-index = 2.

Hypergraph h-index of order n

The Hypergraph h-index of order n for node v, denoted as $\hat{h}_v^{(n)}$, is defined for any natural number $n \in \mathbb{N}$ by the following recurrence relation:

$$\hat{h}_v^{(n)} = \begin{cases} |N(v)| & n = 0 \\ h_v^{(n)} & n > 0 \wedge LCCSAT(h_v^{(n)}) \\ \max\{k \mid k < h_v^{(n)} \wedge LCCSAT(k)\} & n > 0 \wedge \neg LCCSAT(h_v^{(n)}) \textbf{(Core-correction)} \end{cases} \quad (3)$$

Hypergraph h-index has a limit

For any node $v \in V$ of a hypergraph $H = (V, E)$, the two sequences $(h_v^{(n)})$ and $(\hat{h}_v^{(n)})$ have the same limit: $\lim_{n\to\infty} h_v^{(n)} = \lim_{n\to\infty} \hat{h}_v^{(n)}$.

The limiting value is the core-number

If the local coreness-constraint is satisfied for all nodes $v \in V$ at the terminal iteration, the corrected h-index at the terminal iteration $\hat{h}_v^{(\infty)}$ satisfies: $\hat{h}_v^{(\infty)} = c(v)$.

Convergence time guarantee

Given a node $v \in N_i$ in a hypergraph H, it holds that $\forall n \geq i,\ \hat{h}_v^{(n)} = c(v)$. Here N_i is the i-th *neighborhood hierarchy*, which contains the set of nodes that have the minimum number of neighbors in $H[V']$, where $V' = V \setminus \cup_{0 \leq j < i} N_j$

Optimisations and parallelisation of Local core

- **Optimisations:** We have proposed 4 optimisations to make **Local-core** more efficient.
- **Parallisation:** We have proposed **Local-core(p)**, a shared-memory, data parallel programming adaptation of Local-core.
- **Generalised core model**: We have proposed a generalised hypergraph core model (*neighborhood, degree)-core* that simultaneously considers degree constraint and neighborhood constraint.

Efficiency evaluation

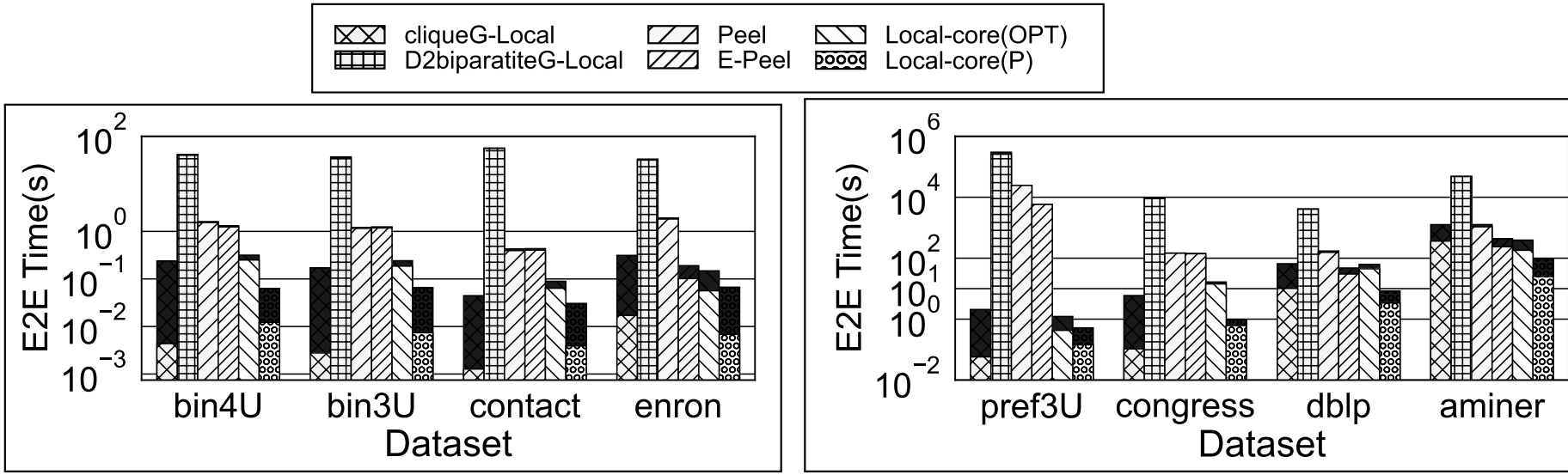


Figure 7: (a)-(b) End-to-end (E2E) running time of our algorithms: **Peel**, **E-Peel**, **Local-core(OPT)**, **Local-core(P)** with 64 Threads vs. those of baselines: **Clique-Graph-Local** and **Distance-2 Bipartite-Graph-Local**. End-to-end (E2E) running time = data structure initialization time (shaded with dark-black on top of each bar) + algorithm's execution time.

Our OpenMP parallel implementation **Local-core(P)** decomposes *aminer* hypergraph with 27M nodes, 17M hyperedges in 91 seconds.

Application 1: Densest subgraph discovery

- A new notion of densenset sub-hypergraph.

Volume-densest subhypergraph

The **volume-densest subhypergraph** is a subhypergraph which has the largest volume-density among all subhypergraphs. The **volume-density** $\rho^N[S]$ of a subset $S \subseteq V$ of nodes in a hypergraph $\rho^N[S] = \frac{\sum_{u \in S} |N_S(u)|}{|S|}$.

- **Greedy approximation algorithm** for volume-densest subhyp. recovery is $(d_{pair}(d_{card} - 2) + 2)$-approximate, where hyperedge-cardinality and node-pair co-occurrence (# hyperedges containing that pair) are at most d_{card} and d_{pair}, resp.

Case study: Nashville Meetup Dataset

- The degree-densest subhyp. contains casual, frequent gatherings from only one socializing group. (**Not informative**)
- The degree-densest subgraph of the clique graph captures technical events arranged by diverse, yet niche activity groups (e.g. 5 participants on avg.) (**Informative**)
- The volume-densest subhyp. captures technical events arranged by diverse and vibrant activity groups (78 participants on avg.) (**More informative**)

Application 2: Influence spreading and intervention

Initially, all nodes except one called a *seed*- are at the *susceptible* state. The seed node is initially at the *infectious* state. At each time step, each infected node infects its susceptible neighbors with probability β and then becomes *immunized*. Once a node is immunized, it is never re-infected.

1. Inner-cores produced by our decomposition contain influential spreaders.
2. Our decomposition produces the best order of important nodes for deleting a limited number of them while causing the maximum intervention in spreading.