Pacific Northwest
NATIONAL LABORATORY

Hypergraph Edit Distance: Theory, Algorithms, and Applications

Emilie Purvine
Chief Data Scientist
Team Lead of Complex Data Models

Joint with:
Cliff Joslyn, Audun Myers

U.S. DEPARTMENT of ENERGY BATTELLE

PNNL is operated by Battelle for the U.S. Department of Energy

PNNL-SA-224559

Pacific Northwest
NATIONAL LABORATORY

Plan of talk

- Motivation
- Graph and hypergraph edit distance
  - Definitions
  - Equivalences
  - Special cases
- Applications
  - Entity resolution in bibliographic database
  - Cyber host anomaly detection

Relational or "Tabular" Data is Common

Raw data

- **Boolean Attributes:** Entities (rows) are indicated as having specific attributes, properties, or behaviors (columns)
- **Joint relationships:** Entities jointly participate in some relationship or activity
- **Numeric data:** consider thresholding the data

	Supervisor?	Brown Eyes?	Female?	Surgeon?
John	X	X		
Sally	X	X	X	X
Spoe				X
Gertie		X	X	X

	camcountry.com	crowmedicine.com	sonymusicnashville.com	elvisthemusic.com
104.16.236.100	X	X	X	
104.16.237.100	X	X		X
104.16.238.100	X	X	X	X
104.16.235.100		X	X	X

Data from https://activednsproject.org/

Protein	MERS_WT_0HRS	MERS_WT_12HRS	MERS_WT_24HRS	MERS_WT_36HRS
AAAS	0.089651	-0.081078	-0.136514	-0.268651
AACS	-0.039801	0.344122	-0.970295	0.106712
AADAC	0.017425	-0.018319	0.050566	-0.159475
AAK1	-0.112657	0.235510	0.480794	-0.852893
AAMP	0.021689	0.131453	-0.574124	0.011710

Relational or "Tabular" Data is Common

Binary data

- **Boolean Attributes:** Entities (rows) are indicated as having specific attributes, properties, or behaviors (columns)
- **Joint relationships:** Entities jointly participate in some relationship or activity
- **Numeric data:** consider thresholding the data
  e.g., cell value > 0

	Supervisor?	Brown Eyes?	Female?	Surgeon?
John	1	1	0	0
Sally	1	1	1	1
Spoe	0	0	0	1
Gertie	0	1	1	1

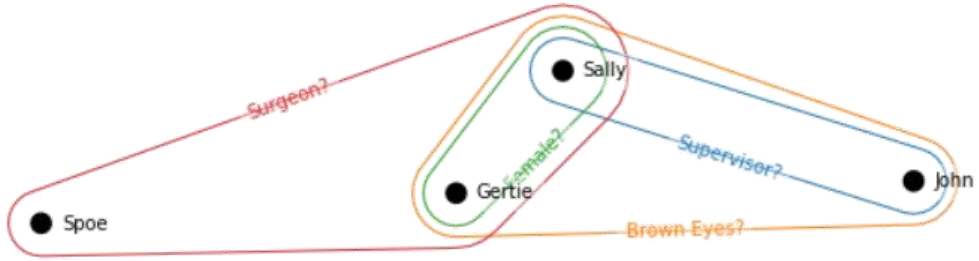
	camcountry.com	crowmedicine.com	sonymusicnashville.com	elvisthemusic.com
104.16.236.100	1	1	1	0
104.16.237.100	1	1	0	1
104.16.238.100	1	1	1	1
104.16.235.100	0	1	1	1

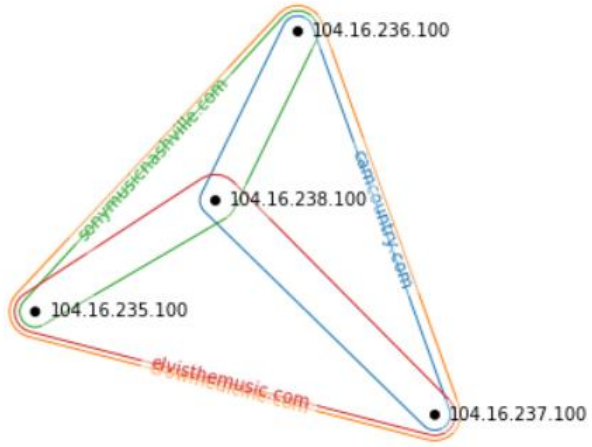
Data from https://activednsproject.org/

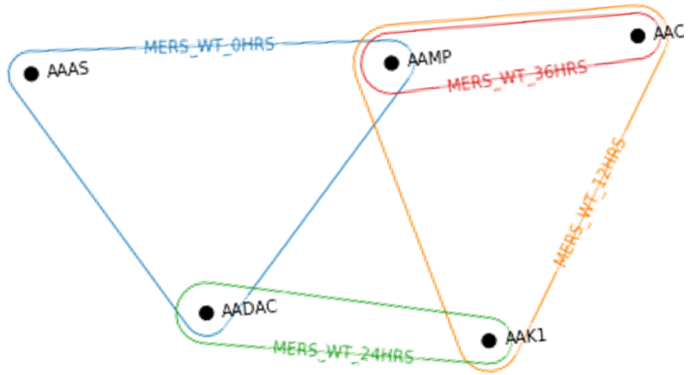
Protein	MERS_WT_0HRS	MERS_WT_12HRS	MERS_WT_24HRS	MERS_WT_36HRS
AAAS	1	0	0	0
AACS	0	1	0	1
AADAC	1	0	1	0
AAK1	0	1	1	0
AAMP	1	1	0	1

Relational or “Tabular” Data is Common

Inducing groups







	Supervisor?	Brown Eyes?	Female?	Surgeon?
John	1	1	0	0
Sally	1	1	1	1
Spoe	0	0	0	1
Gertie	0	1	1	1

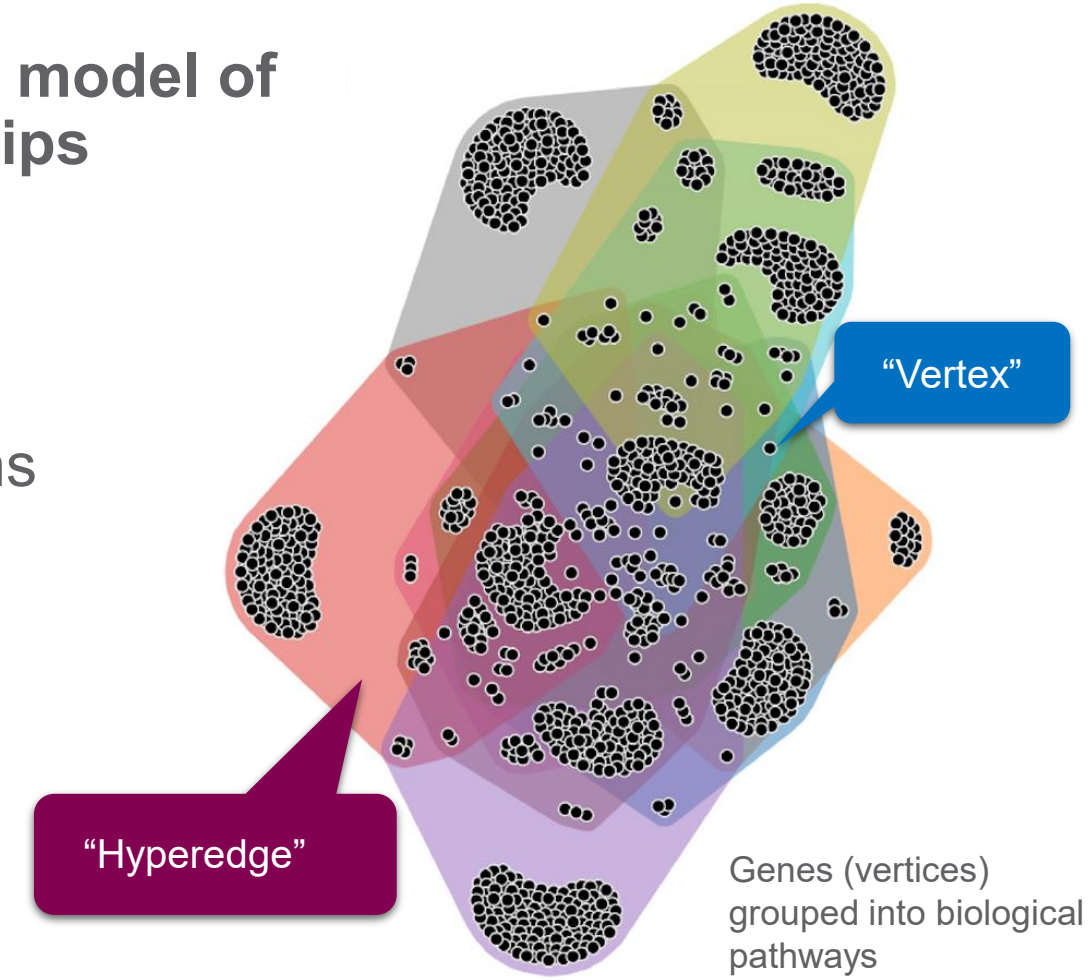
	camcountry.com	crowmedicine.com	sonymusicnashville.com	elvisthemusic.com
104.16.236.100	1	1	1	0
104.16.237.100	1	1	0	1
104.16.238.100	1	1	1	1
104.16.235.100	0	1	1	1

Data from https://activednsproject.org/

Protein	MERS_WT_0HRS	MERS_WT_12HRS	MERS_WT_24HRS	MERS_WT_36HRS
AAAS	1	0	0	0
AACS	0	1	0	1
AADAC	1	0	1	0
AAK1	0	1	1	0
AAMP	1	1	0	1

Hypergraphs can provide accurate models of multi-way interactions in a way that graphs can't

- **Hypergraphs provide a mathematical model of data focused on multi-way relationships**
  - To ***ask*** certain kinds of questions
    - ✓Connectivity of entities
    - ✓Clustering structure
  - To ***model*** certain kinds of interactions
    - ✓Multi-way relationships
- **Examples of systems hypergraphs can model:**
  - Social networks
  - Biological systems
  - Shared behaviors among entities
  - Documents and keywords or topics



Genes (vertices) grouped into biological pathways (hyperedges)

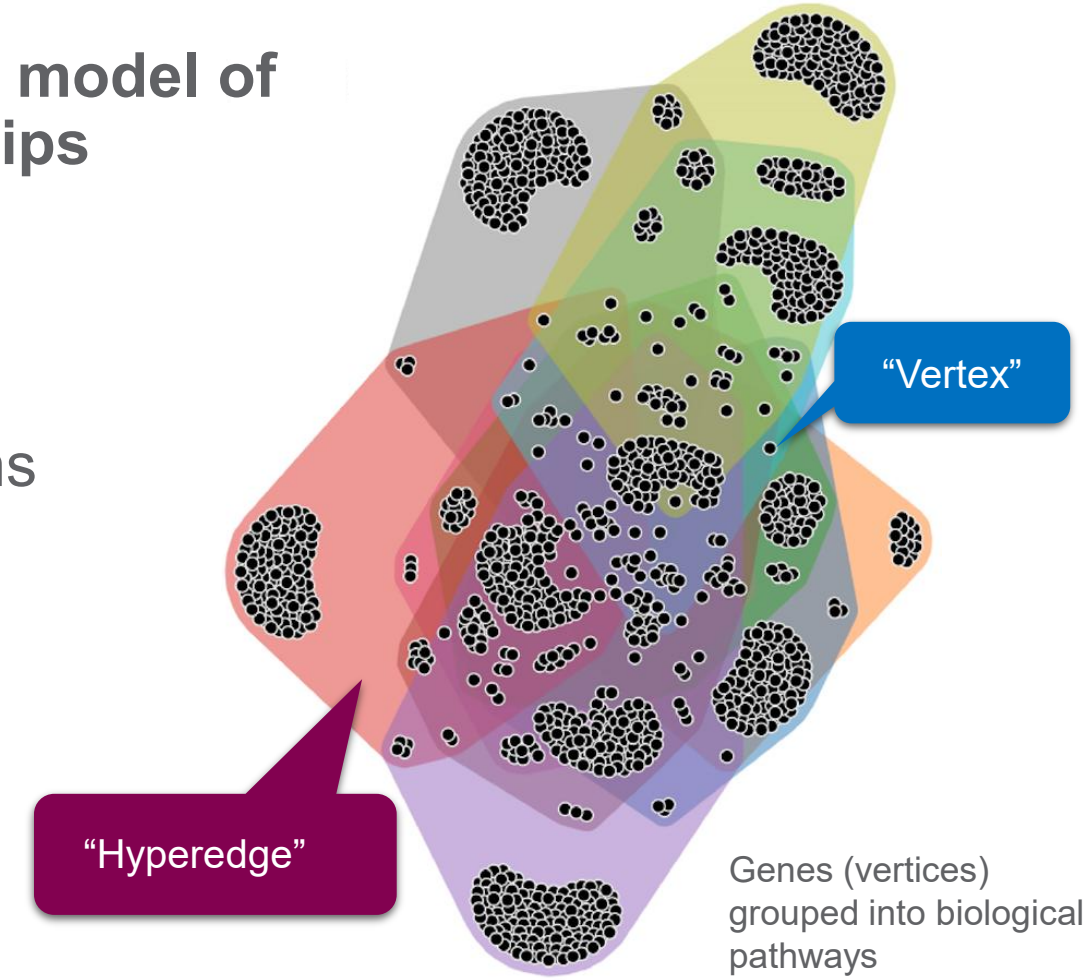
Hypergraphs can provide accurate models of multi-way interactions in a way that graphs can't

- **Hypergraphs provide a mathematical model of data focused on multi-way relationships**
  - To ***ask*** certain kinds of questions
    - ✓Connectivity of entities
    - ✓Clustering structure
  - To ***model*** certain kinds of interactions
    - ✓Multi-way relationships

Formally, $H = (V, E, \epsilon)$ where

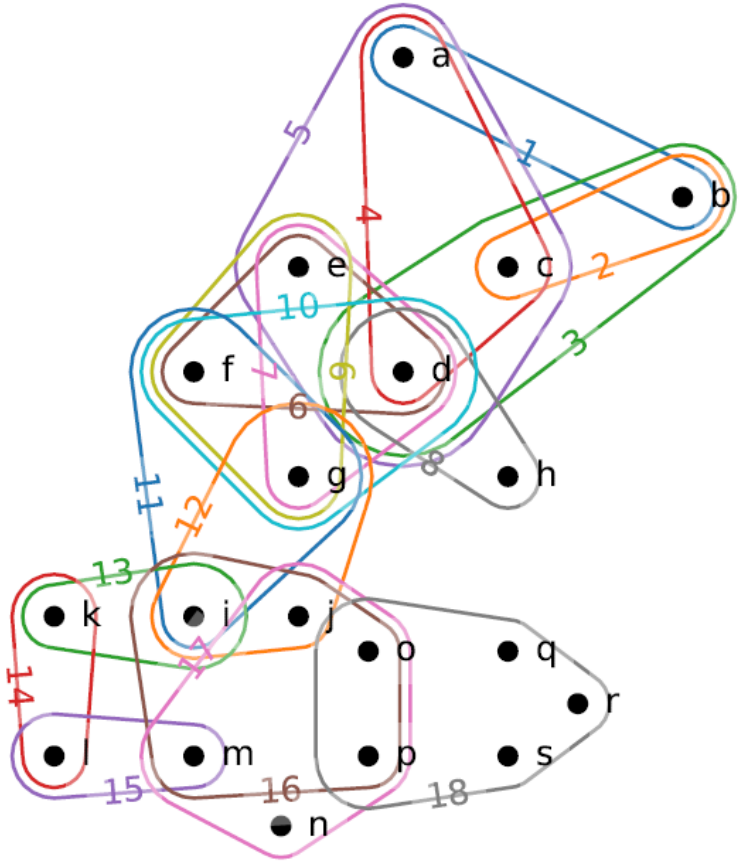
- V is a set of vertices,
- E is a set of edge ids,
- $\epsilon : E \to 2^V$ maps edge ids to sets of vertices defining the edges as sets

This defines a **multi-hypergraph**



Genes (vertices) grouped into biological pathways (hyperedges)

Traditional analysis of individual hypergraphs: analogs of network science



Hypergraph properties

- Degree (distribution)
- Edge size (distribution)
- s-Walk, s-Path, s-Diameter
- s-Connected components
- s-Centrality
- Clustering coefficient?
- Triangle counting?

More complex, not in this talk (but interesting)

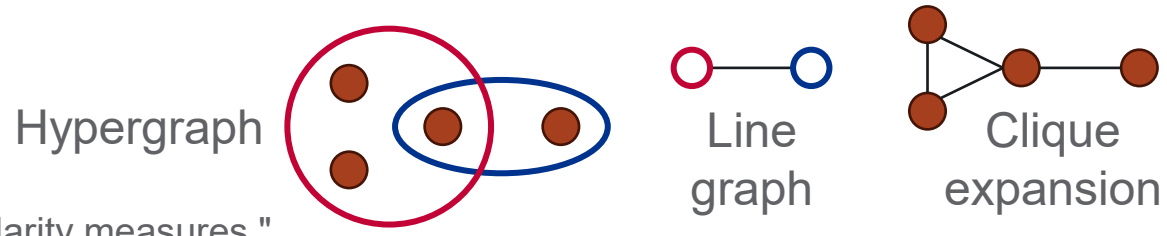
- …

https://github.com/pnnl/HyperNetX

S.G. Aksoy, C.A. Joslyn, C.M. Ortiz Marrero, B.L. Praggastis, and E. Purvine. “Hypernetwork Science via High-Order Hypergraph Walks.” EPJ Data Science, accepted June 2020.

What if we have a collection of (hyper)graphs and want to compare them?

- Global similarity measures: based on properties agnostic of edge node and edge labels
  - Compare spectra of Laplacian matrices
  - Global feature vectors, e.g., degree / edge size distributions, triangle or motif counts, diameter
  - Embedding-based metrics via graph neural networks (GNNs)
- Label-dependent similarity measures: Assuming approx. same label set
  - Hamming or Jaccard distance of adjacency or incidence matrices
  - Compute vertex feature vectors, e.g., centrality measures, local motif counts
- Hypergraph comparison could be based on graph projections
  - Problems arise when multiple hypergraphs have same line graph and/or clique expansion



Surana, Amit, Can Chen, and Indika Rajapakse. "Hypergraph similarity measures." *IEEE Transactions on Network Science and Engineering* 10, no. 2 (2022): 658-674.

Plan of talk

- Motivation
- Graph and hypergraph edit distance
  - Definitions
  - Equivalences
  - Special cases
- Applications
  - Entity resolution in bibliographic database
  - Cyber host anomaly detection

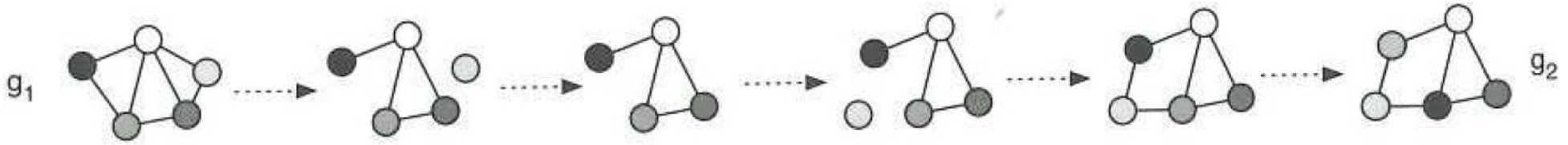
Example from Riesen, Kaspar, and Horst Bunke. *Graph classification and clustering based on vector space embedding*. Vol. 77. World Scientific, 2010. [Chapter 3]

Graph edit distance as inspiration

- **Definition:** Let G_1, G_2 be two (finite, simple) graphs. Their graph edit distance (GED) is given by

$$GED(G_1, G_2) = \min_{(\epsilon_1,\ldots,\epsilon_k)\in E(G_1,G_2)} \sum_{i=1}^{k} c(\epsilon_i)$$

  where $E(G_1, G_2)$ is set of all **edit paths**, $(\epsilon_1, \ldots, \epsilon_k)$, to transform G_1 into G_2.
  - ϵ_i are atomic edit operations: vertex/edge addition, deletion, or modification
  - $c(\epsilon_i)$ is the cost of the atomic edit



- **Atomic edit notation:** $v \to u, \varepsilon \to u, v \to \varepsilon$ for vertex edits, analogs for edges
  - Edge edits come "for free" from vertex edits. If $v \to u, w \to x$ then we know what must happen with edges $(v, w) \in E_1$ and $(u, x) \in E_2$ if either or both of them exist.
  - Vertex (and edge) merging / splitting not allowed. Implies partial injection from V_1 to V_2

Pacific Northwest NATIONAL LABORATORY

A word on cost functions

- Application dependent, provides flexibility but also additional burden
- A few assumptions to make edit paths finite
  - $c(\epsilon) \geq 0$ for all edit operations
  - $c(\epsilon) > 0$ for all removals and additions (prevents add then remove same vertex/edge)
  - Triangle inequality

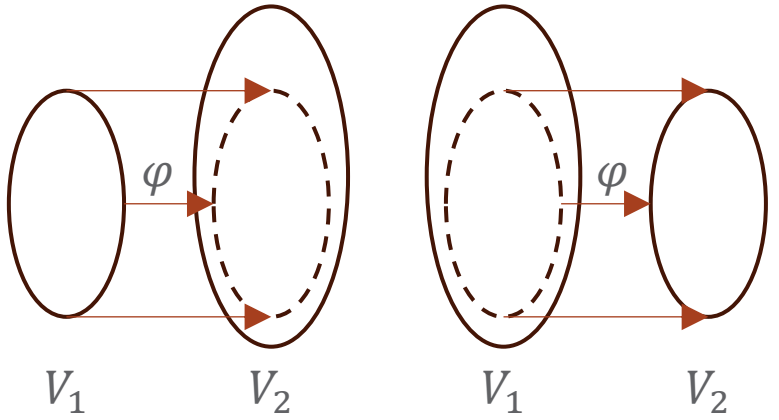
$$c(v \to u) \leq c(v \to w) + c(w \to u)$$
$$c(v \to \varepsilon) \leq c(v \to w) + c(w \to \varepsilon)$$
$$c(\varepsilon \to u) \leq c(\varepsilon \to w) + c(w \to u)$$

    (limits to only one operation per vertex). Same for edges.
- Another triangle-type inequality ($c(v \to u) \leq c(v \to \varepsilon) + c(\varepsilon \to u)$, and for edges) guarantees min. edit path will have removals or additions, not both.
- To guarantee GED is a distance:
  - $c(v \to u) = 0$ if v and u are identical (same for edges) – isomorphic graphs are dist. 0
  - $c(\epsilon) = c(\epsilon^{-1})$ – needed for $GED(G_1, G_2) = GED(G_2, G_1)$

Alternate formulation for GED using partial injection

- Given $\varphi: \hat{V}_1 \hookrightarrow V_2$ partial injection with $\hat{V}_1 \subseteq V_1$ we can define a corresponding sequence of atomic edits on vertices:
  - For every $v \in \hat{V}_1,\ v \to \varphi(v)$ – vertex substitution
  - For every $v \in V_1 \setminus \hat{V}_1,\ v \to \varepsilon$ – vertex deletion
  - For every $u \in V_2 \setminus \varphi(\hat{V}_1), \varepsilon \to u$ – vertex addition

$|V_1| \leq |V_2|$ or $|V_1| \geq |V_2|$



- Vertex edits imply edge edits:
  - For every $e = (v, w) \in E_1$
    - ✓ If $v, w \in \hat{V}_1$ and $(\varphi(v), \varphi(w)) = f \in E_2$ then $e \to f$, otherwise $e \to \varepsilon$
  - For any $f \in E_2$ that isn't mapped to, $\varepsilon \to f$
- Given an edit path we can define the vertex map directly by just looking at the $v \to u$ edits. This forms $\hat{V}_1$ (those v that are mapped).
- Then $GED(G_1, G_2) = \min_{\hat{V}_1 \subseteq V_1,\ \varphi: \hat{V}_1 \hookrightarrow V_2} GEC(G_1, G_2, \varphi)$, GEC = graph edit cost

Generalize GED to hypergraphs

- **Definition:** Let $\mathcal{H}_1, \mathcal{H}_2$ be two (finite) hypergraphs. Their hypergraph edit distance (HGED) is given by

$$HGED(\mathcal{H}_1, \mathcal{H}_2) = \min_{(\epsilon_1,\dots,\epsilon_k)\in E(\mathcal{H}_1,\mathcal{H}_2)} \sum_{i=1}^{k} c(\epsilon_i)$$

  where $E(\mathcal{H}_1, \mathcal{H}_2)$ is set of all **edit paths**, $(\epsilon_1, \dots, \epsilon_k)$, to transform $\mathcal{H}_1$ into $\mathcal{H}_2$.
  - ϵ_i are atomic edit operations: *not so straightforward*
  - $c(\epsilon_i)$ is the cost of the atomic edit

- Two paradigms for atomic edits:
  - "Immutable edges" – edges must map exactly where their vertices map
  - "Individual edits" – can remove vertices from edges and add vertices to edges

"Immutable edges" – direct analog of GED

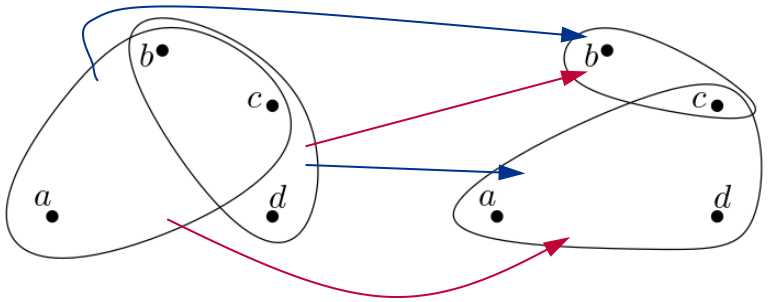
- Atomic edits same type as GED: vertex / edge substitution, addition, deletion
- Edit path again bijective with partial injection on vertices, slight mod. for edges
  - For every $e = (v_1, \dots, v_t) \in E_1$
    - ✓ If $v_1, \dots, v_t \in \hat{V}_1$ and $(\varphi(v_1), \dots, \varphi(v_t)) = f \in E_2$ then $e \to f$, otherwise $e \to \varepsilon$
  - For any $f \in E_2$ that isn't mapped to, $\varepsilon \to f$

- Then define $HGED_{imm}(\mathcal{H}_1, \mathcal{H}_2) = \min_{\hat{V}_1 \subseteq V_1,\, \varphi:\hat{V}_1 \hookrightarrow V_2} HGEC_{imm}(\mathcal{H}_1, \mathcal{H}_2, \varphi)$
  - Number of possible maps $\sum_{k=1}^{\min(|V_1|,|V_2|)} \binom{|V_1|}{k}\binom{|V_2|}{k} k!$
  - In practice the optimal edit path has only additions or deletions of vertices, but HGEC is defined for any partial injection even with $|\hat{V}_1| < \min(|V_1|, |V_2|)$

Horst Bunke, Peter Dickinson, Miro Kraetzl, Michel Neuhaus, and Marc Stettler. Matching of Hypergraphs — Algorithms, Applications, and Experiments, pages 131–154. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

“Individual edits” – more types of atomic edits

- Same minimum over edit paths, but allow adding or removing *incidences*
  - Atomic edits as before plus $(v, e) \to \varepsilon$ and $\varepsilon \to (u, f)$
  - *To remove an edge (resp. vertex) must first remove all its incidences, then remove empty edge (resp. isolated vertex). Analogously (in reverse) for adding edges (resp. vertices).*
- No longer uniquely defined by vertex map, need edge map too
- Vertex and edge maps imply incidence edits



In hypergraph, vertex map consistent with multiple edge maps

$$HGED_{ind}(\mathcal{H}_1, \mathcal{H}_2) = \min_{\substack{f_V: \hat{V}_1 \hookrightarrow V_2 \\ f_E: \hat{E}_1 \hookrightarrow E_2}} HGEC_{ind}(\mathcal{H}_1, \mathcal{H}_2, f_V, f_E)$$

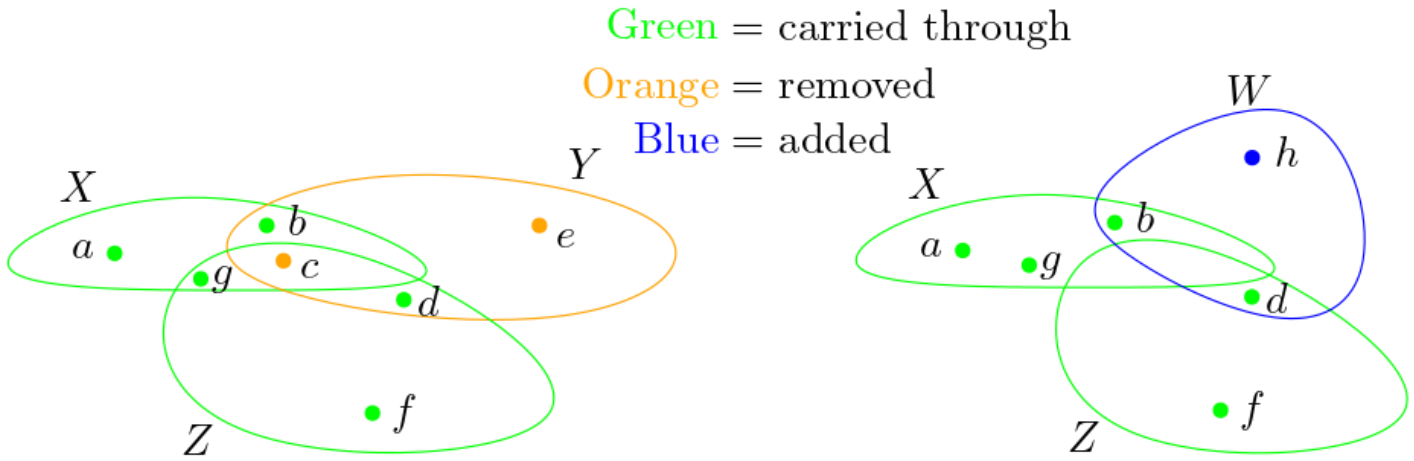
$(a, 1) \to \varepsilon$ $\quad$ $(b, 1) \to \varepsilon$

$(b, 2) \to \varepsilon$ $\quad$ $\varepsilon \to (d, f_E(1))$

$\varepsilon \to (a, f_E(2))$ $\quad$ $(d, 2) \to \varepsilon$

Hongchao Qin, Rong-Hua Li, Ye Yuan, Guoren Wang, and Yongheng Dai. Explainable hyperlink prediction: A hypergraph edit distance-based approach. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 245–257. IEEE, 2023.

Example to compare individual vs. immutable



- **Individual edits:**
  - Remove c: $(c, X), (c, Y), (c, Z) \rightarrow \varepsilon, c \rightarrow \varepsilon$
  - Remove e: $(e, Y) \rightarrow \varepsilon, e \rightarrow \varepsilon$
  - Remove Y: $(b, Y), (d, Y) \rightarrow \varepsilon, Y \rightarrow \varepsilon$
  - Add h: $\varepsilon \rightarrow h$
  - Add W: $\varepsilon \rightarrow W, \varepsilon \rightarrow (b, W), (d, W), (h, W)$
  - Remove g from Z: $(g, Z) \rightarrow \varepsilon$
- **Immutable edges:** All edges change memberships so must remove all edges and add back in

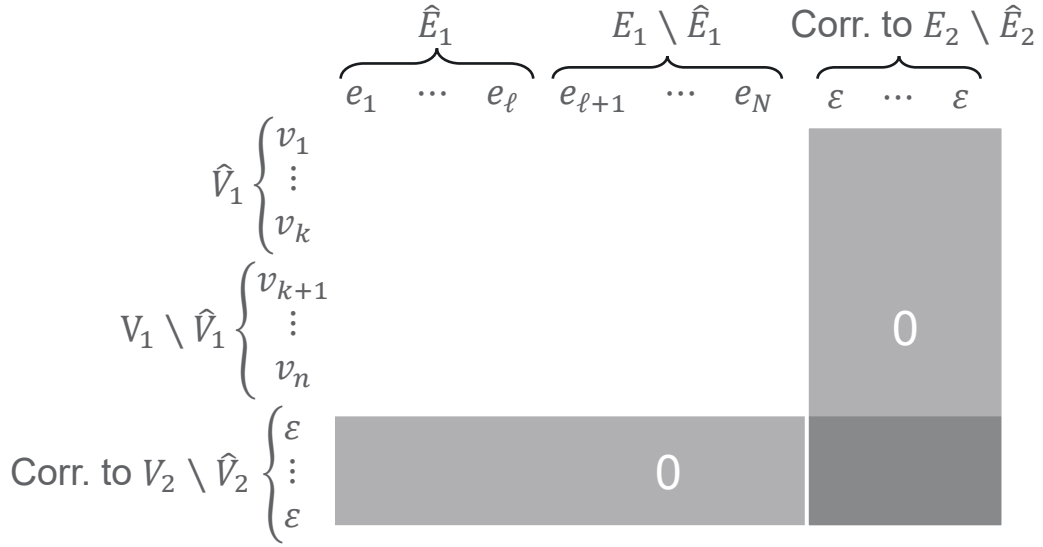
Computational algorithms for (H)GED

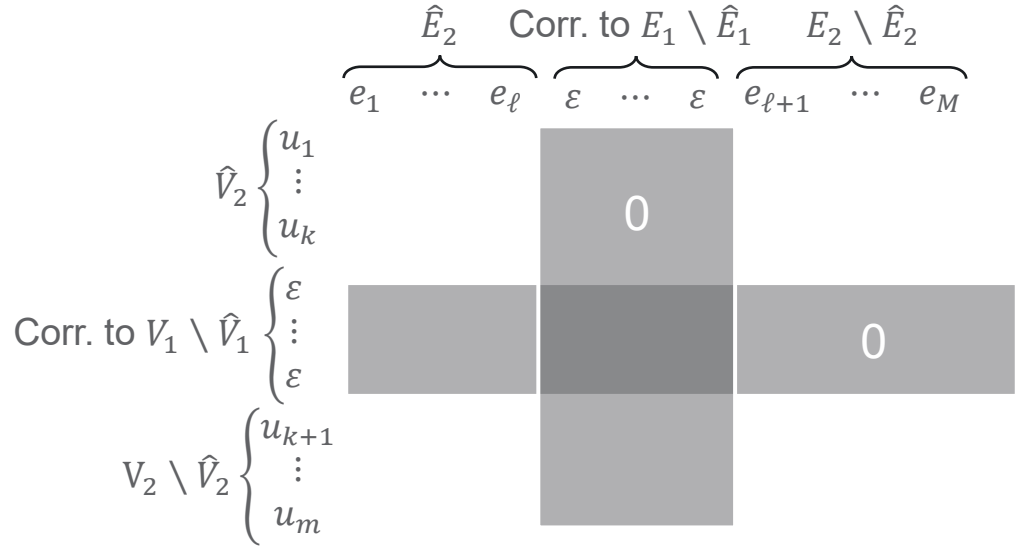
- Exact computation is NP-hard [1] – equivalent to sub-(hyper)graph isomorphism.
  - Exact algorithms based on pruning combinatorial search space [2]
  - Approximation or bounding algorithms explored using local search, linear programming, linear sum assignment [3]
- Bunke, et al. defined $HGED_{imm}$ for hypergraph matching to find maximum common sub-hypergraph
  - They discuss algorithms for computation
  - Experiments on real and synthetic hypergraph data
- Qin, et al. defined $HGED_{ind}$ for hyperedge prediction and node similarity
  - Proposed BFS- and DFS-based algorithms for computation

1. Zhiping Zeng, Anthony KH Tung, Jianyong Wang, Jianhua Feng, and Lizhu Zhou. Comparing stars: On approximating graph edit distance. *Proceedings of the VLDB Endowment*, 2(1):25–36, 2009.
2. David B Blumenthal and Johann Gamper. On the exact computation of the graph edit distance. *Pattern Recognition Letters*, 134:46–57, 2020.
3. David B Blumenthal, Nicolas Boria, Johann Gamper, Sebastien Bougleux, and Luc Brun. Comparing heuristics for graph edit distance computation. *The VLDB journal*, 29(1):419–458, 2020.

Some theoretical observations

- GED, $HGED_{imm}$, and $HGED_{ind}$ are true distances if assumptions on cost function defined earlier hold
  - Consider graph of all finite (hyper)graphs where edges connect two (hyper)graphs if there is an atomic edit between them, weight of edge is cost of edit
  - GED, $HGED_{imm}$, $HGED_{ind}$ are then realized by shortest weighted path distance here
- Relation to Hamming distance: Given $\mathcal{H}_1, \mathcal{H}_2, f_V, f_E$ create incidence matrices respecting the mapping functions. Assume costs are 0 or 1.





Pacific Northwest NATIONAL LABORATORY

More theoretical observations

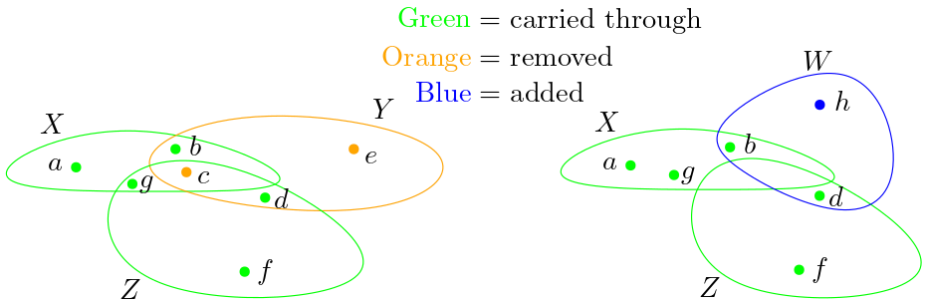
- $GED(G_1, G_2) = HGED_{imm}(G_1, G_2)$, i.e., if the hypergraphs are graphs the immutable edits paradigm agrees with graph edit distance
  - Follows directly from definitions

- Let $\mathcal{B}(\mathcal{H})$ be the bipartite graph associated with hypergraph $\mathcal{H}$, then
$$HGEC_{ind}(\mathcal{H}_1, \mathcal{H}_2, f_V, f_E) = GEC(\mathcal{B}(\mathcal{H}_1), \mathcal{B}(\mathcal{H}_2), f_V \sqcup f_E).$$
Whether $HGED_{ind} = GED$ depends on structure. We can't force vertices to map to vertices and edges to map to edges in the optimal GED vertex map.

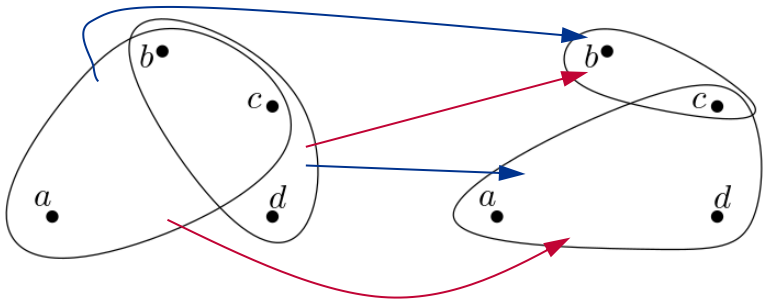
- Duality: $HGED_{ind}(\mathcal{H}_1, \mathcal{H}_2) = HGED_{ind}(\mathcal{H}_1^*, \mathcal{H}_2^*)$.
  - If one were smaller than the other then the mappings that realize it could be turned into mappings for the dual, contradicting that one was smaller.

When vertex and/or edge labels matter

- If both vertex and edge IDs provide mappings this is very computationally tractable via Hamming distance or set operations
  - $HGED_{ind}$ with only add/remove operations, no substitutions. Identity maps on vertices / edges.
  - Could relax this to create SME-informed mappings that include substitutions but still rely on labels for both



- If only vertex labels provide a mapping must optimize over edge mappings. We use linear sum assignment (minimum cost matching), cost of matching two edges is $-\left|e_i \cap e_j\right|$.



Plan of talk

- Motivation
- Graph and hypergraph edit distance
  - Definitions
  - Equivalences
  - Special cases
- Applications
  - Entity resolution in bibliographic database
  - Cyber host anomaly detection

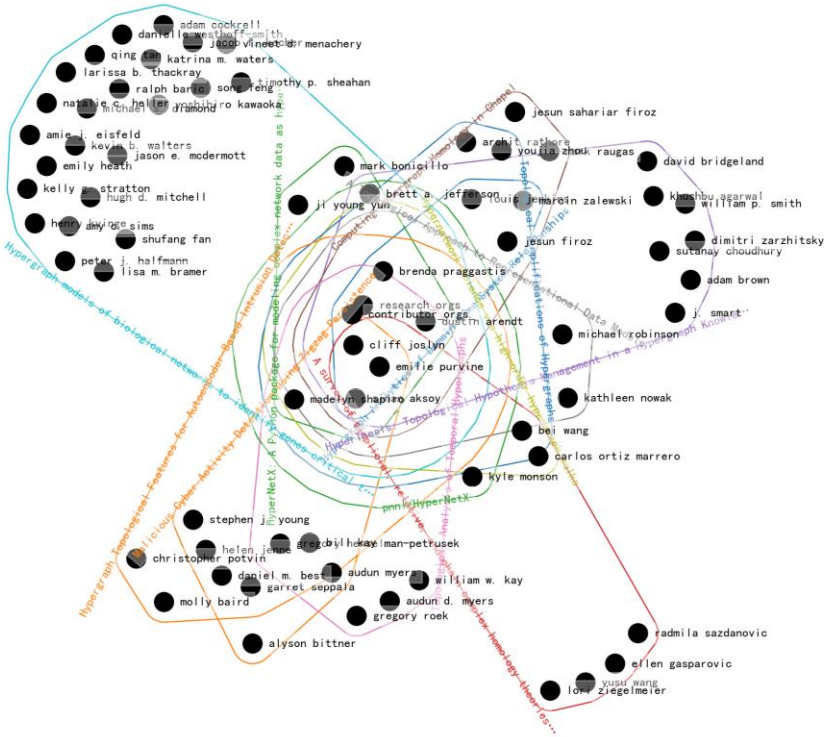
Pacific Northwest
NATIONAL LABORATORY

Entity resolution for bibliographic data

Entity resolution problem and approach

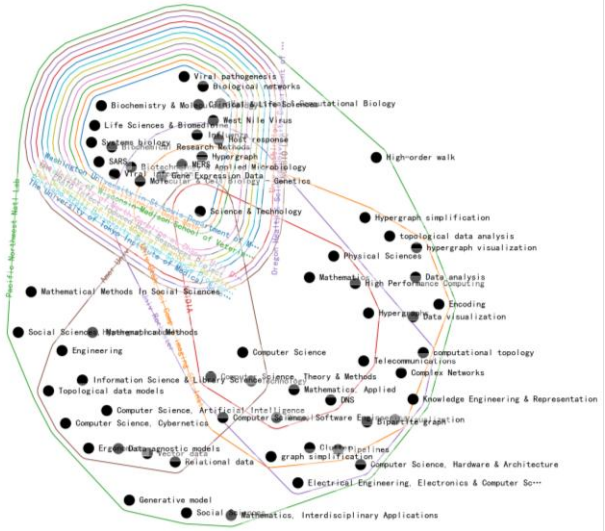
- **Problem:** publication databases include multiple versions of the same author despite all best efforts to track which are the same person (e.g., ORCiD). *How do we disambiguate "author records" to group those that represent the same person?*
- There are many approaches, problem is not solved. Methods include:
  - Pairwise scoring (often ML-based) followed by clustering
  - Build heterogeneous graph of full corpus, learn node embeddings using GNN, use link prediction or clustering in high dimensional space
  - LLMs to compare author records / contexts to get a match score. Due to high cost this is often done after candidates generated by other methods.
- Our approach:
  - Candidate clusters based on text embedding and low threshold ← (won't discuss here)
  - Within each cluster, build hypergraphs per author record, compare using edit cost
  - Logistic regression based on all evidence to get probability of record match

Construction of hypergraphs for each entity in a candidate cluster – restrict to papers by entity



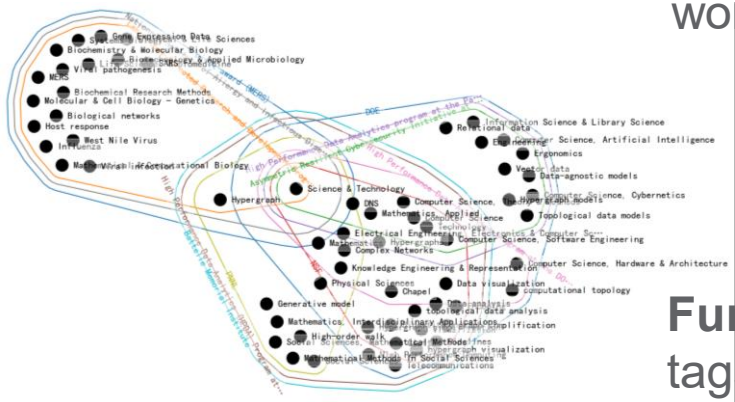
Coauthors: V = authors, E = papers

Examples here shown for author entity “Emilie Purvine”



Institutions: V = topic tags, E = institutions

Fields of study: V = content words from abstract, E = papers

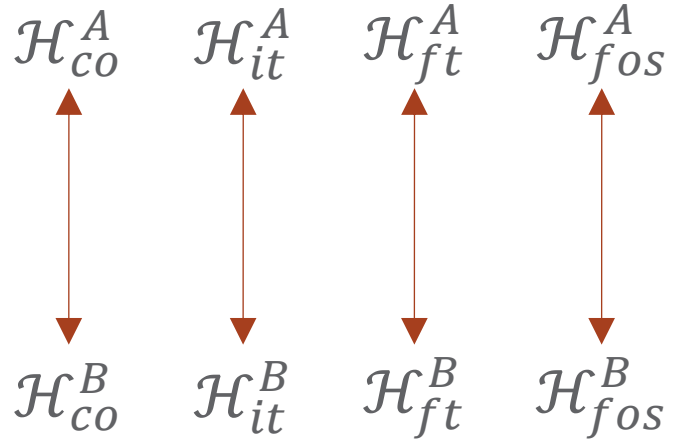


Funding sources: V = topic tags, E = funding agencies

Data from arXiv, bioRxiv, Web of Science, DBLP, OSTI

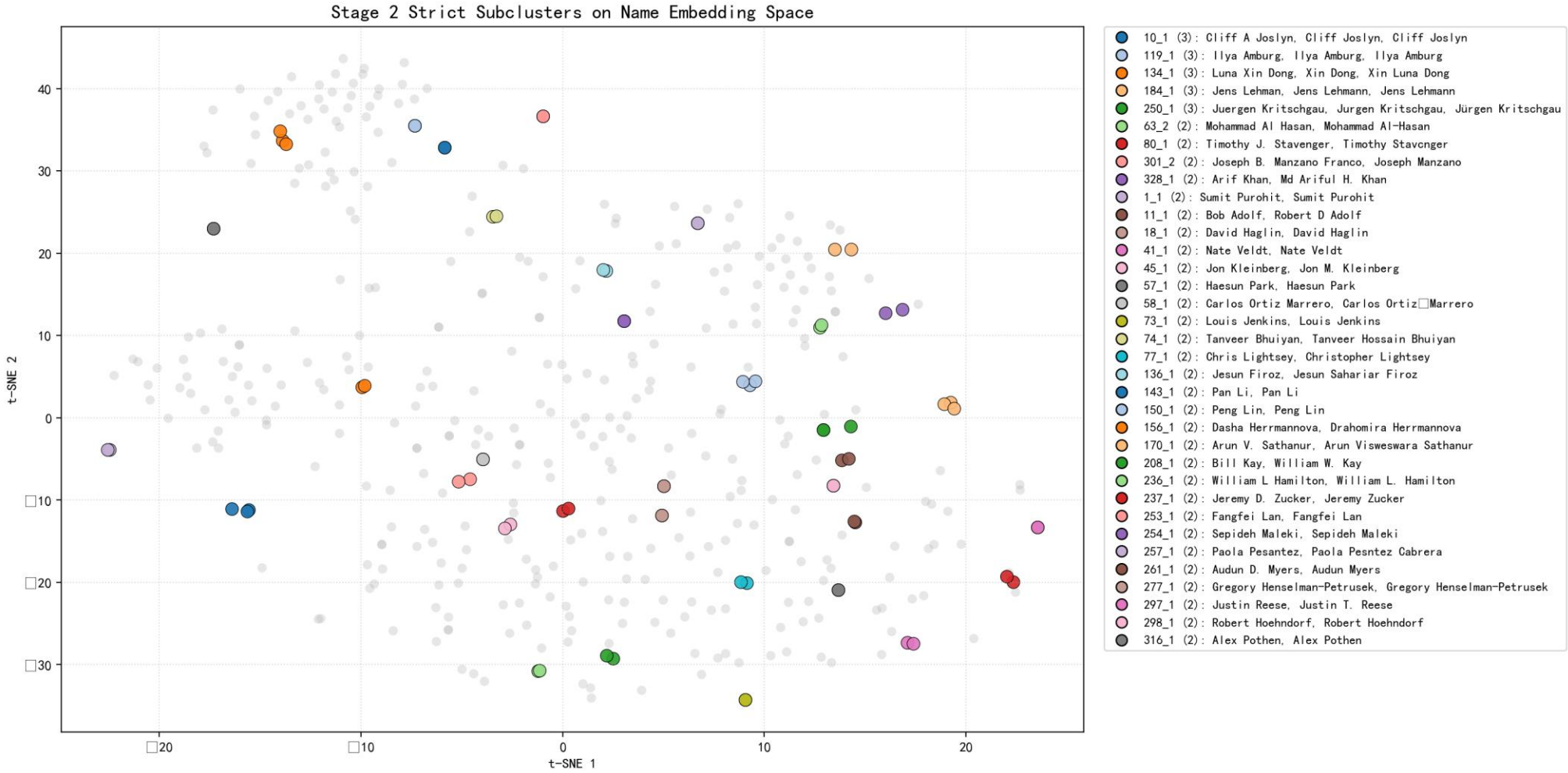
Comparison of hypergraphs for each pair of entities and each "channel"

- Compare corresponding hypergraphs using edit cost
  - f_V by label, f_E optimized: coauthor, field of study
  - f_V and f_E by label: institutions, funding sources
- Record various measures of similarity including:
  - Jaccard on nodes and edges
  - Normalized HGEC and breakdown into node, edge, and incidence scores
  - Directional growth in $[-1, 1]$ – measuring if there are more adds (positive) or more deletes (negative)
- If all four channel hypergraphs are close, mark as high-confidence matched pair.
  - Will use these in training logistic regression
  - Could also use negative pairs



HGEC = Node add
+ Node remove
+ Node substitute
+ Edge add
+ Edge remove
+ Edge substitute
+ Incidence add
+ Incidence remove

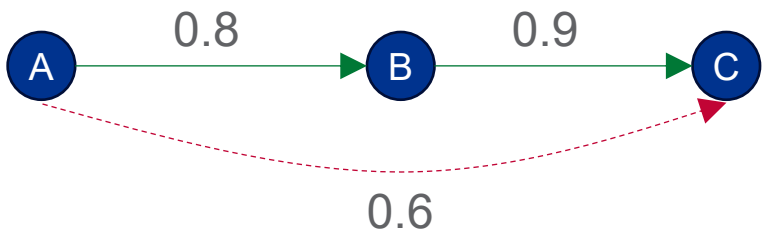
High confidence pairs displayed on text embedding of author name t-SNE projection



Example on corpus of several pub databases (arXiv, bioRxiv, Web of Science, DBLP, OSTI) restricted to a set of 1,579 paper-author rows from a subset of PNNL authors

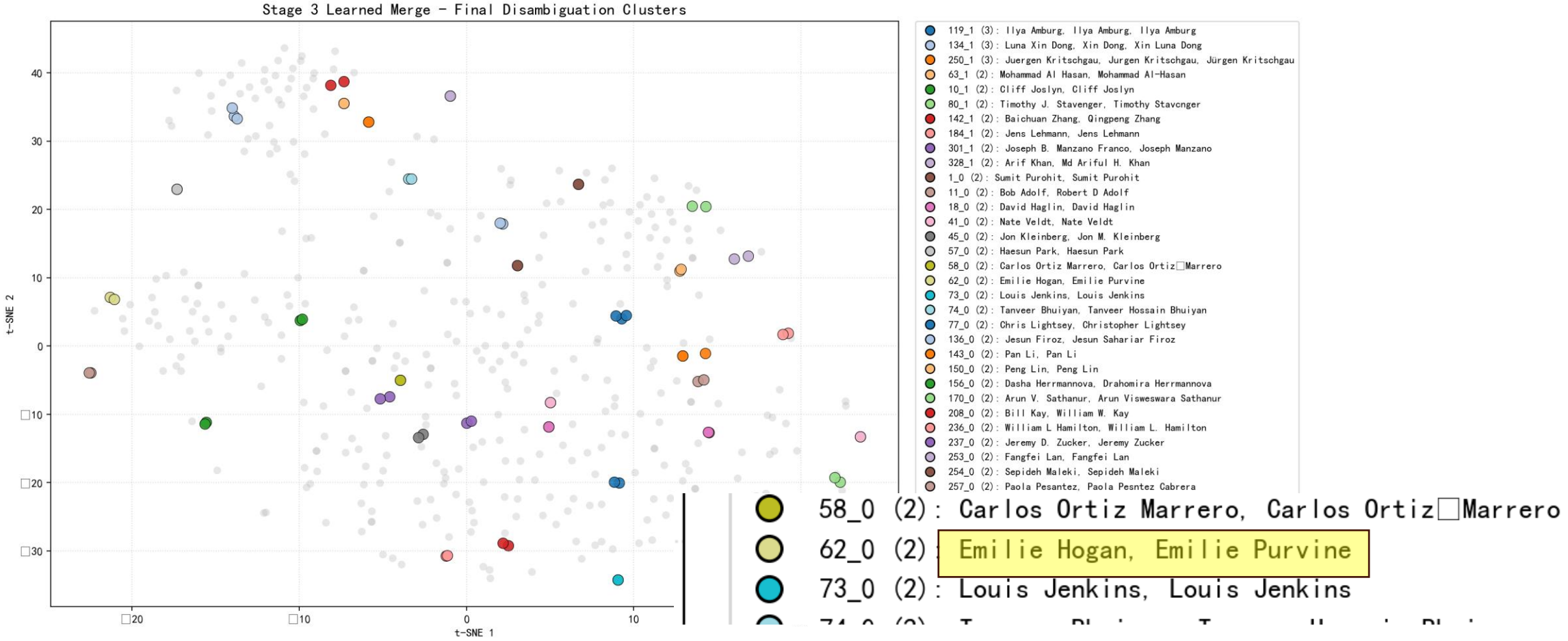
Ensemble of logistic regression models for final disambiguation

- For each channel train a logistic regression model:
  - Input = statistics vector for each pair for that channel
  - Output = probability that pair is the same person
  - Training set: “high confidence pairs” with output value 1.0. Future experiment will also use low confidence pairs with output value 0.0
- Meta-classifier then fuses the four per-channel probabilities to give each author pair a probability of being the same person.
  - A $P(same) \geq 0.75$ cutoff is applied
  - Resulting connected components are noted as individuals



- This pattern puts {A, B, C} all as same person even though (A, C) is not high enough probability
- This is necessary. Consider temporal effects like name changes or changing research interests over time.

Final clusters for small example (we have scaled up to larger paper sets with similar results)



We compared our results to an LLM-only disambiguation and saw very low adjusted Rand index (0.00008) and normalized mutual information (0.00165)) between the two clusterings.

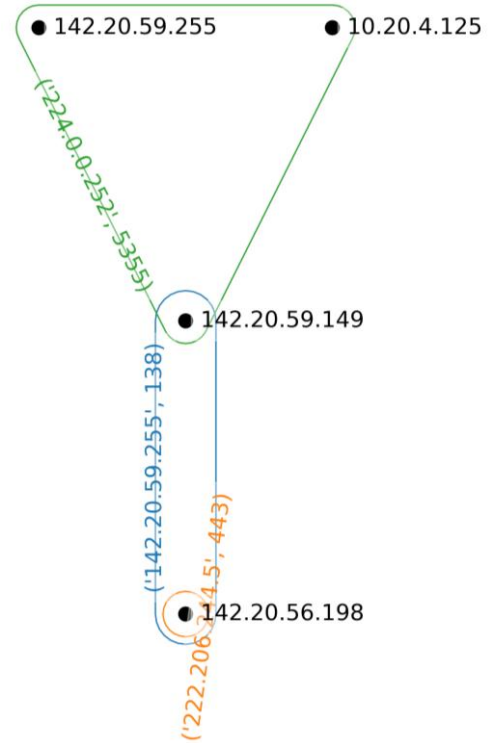
Pacific Northwest
NATIONAL LABORATORY

Anomalous host identification in cyber network

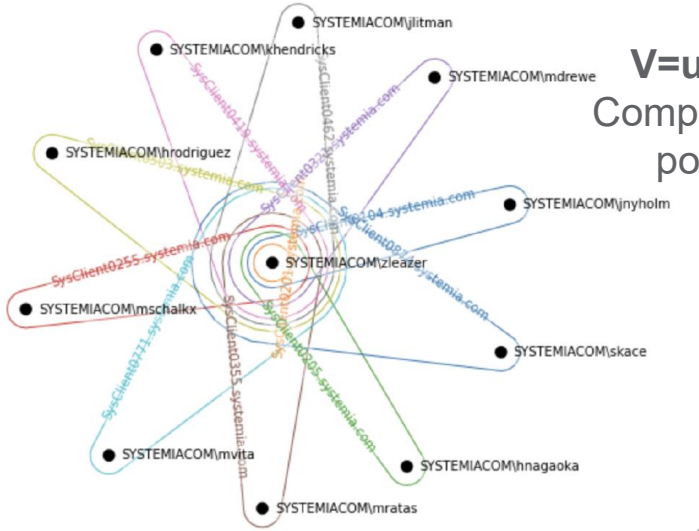
A Hypergraph Construction from Cyber Data

hostname	principal	pid	src_ip	dest_ip	dest_port	l4protocol	image_path
SysClient0201.systemia.com	NT AUTHORITY\SYSTEM	4	142.20.56.198	142.20.59.255	138	UDP	System
SysClient0201.systemia.com	NT AUTHORITY\NETWORK SERVICE	864	10.20.4.125	224.0.0.252	5355	UDP	svchost.exe
SysClient0201.systemia.com	NT AUTHORITY\NETWORK SERVICE	864	142.20.59.255	224.0.0.252	5355	UDP	svchost.exe
SysClient0201.systemia.com	SYSTEMIACOM\zleazer	636	142.20.56.198	222.206.244.5	443	TCP	firefox.exe
SysClient0201.systemia.com	NT AUTHORITY\SYSTEM	4	142.20.59.149	142.20.59.255	138	UDP	System
SysClient0201.systemia.com	NT AUTHORITY\NETWORK SERVICE	864	142.20.59.149	224.0.0.252	5355	UDP	svchost.exe

- Multi-dimensional data set: *n*D-array, *n*-column data frame
- Specify column set for hyperedges (blue)
  - **Unique combinations:**
    (142.20.59.255, 138), (224.0.0.252, 5355), (222.206.244.5, 443)
- Specify disjoint column set for vertices (red)
  - **Unique vertices:**
    142.20.59.149, 10.20.4.125, 142.20.59.255, 142.20.56.198
- A vertex is contained in a hyperedge if there is a record with that combination in the data.
  - Think: “hyperedges = common behaviors”

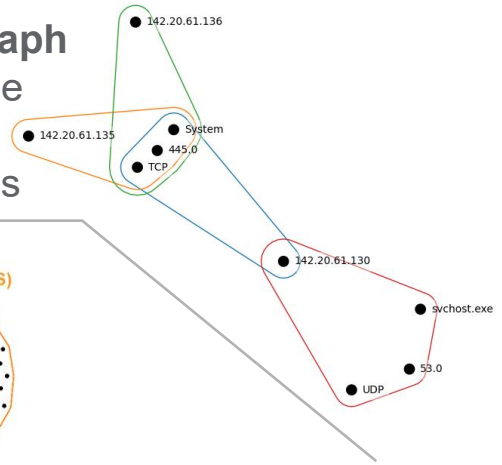


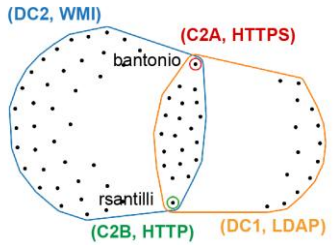
Other Hypergraph Constructions

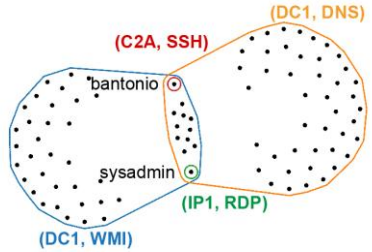


V=user; E=host
Components show possible lateral movement

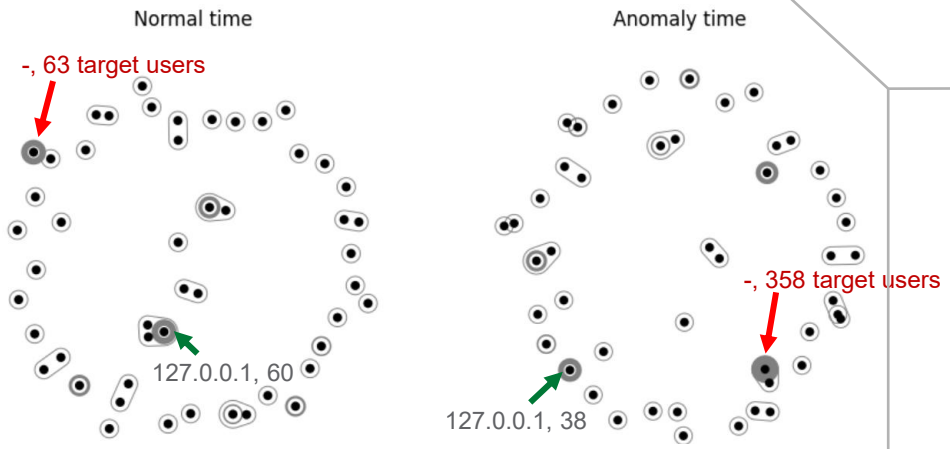
"k-column" hypergraph
Each edge is a log line with vertices as the values for the columns



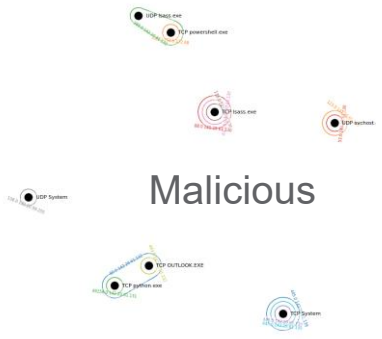




V=(src ip, host); E=(dst ip, dst port)
This pattern is rare and often associated with red team activity.



V=src ip; E=TargetUser
Looking for known password spray



V=(dst ip, dst port); E=(protocol, image_path)
Hypergraphs from single host's data

Workflow from All Data to Identify Outliers

When is an entity in my network
(e.g., user, host, process, …)
"behaving" out of the ordinary?

All log data, all time

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
9/23/23 13:14	START-FLOW	864	142.20.57.199	5355	svchost.exe
9/23/23 13:14	LOAD-MODULE	5628	NA	NA	ping.exe
9/23/23 13:14	MODIFY-FILE	4	NA	NA	system
⋮					
9/23/23 23:59	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:59	LOAD-MODULE	3464	NA	NA	conhost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.1.234	5355	svchost.exe

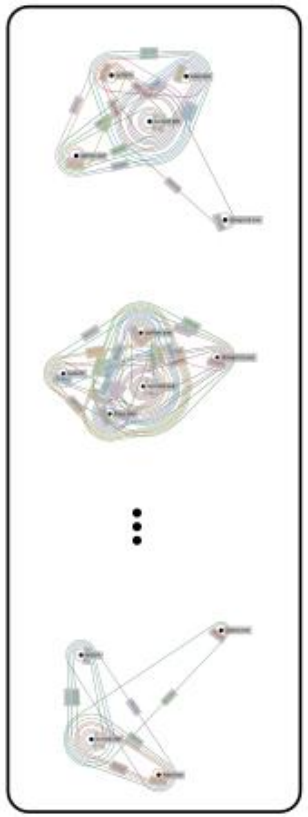
One window of log data

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
9/23/23 13:14	START-FLOW	864	142.20.57.199	5355	svchost.exe
9/23/23 13:14	LOAD-MODULE	5628	NA	NA	ping.exe
9/23/23 13:14	MODIFY-FILE	4	NA	NA	system
⋮					
9/23/23 23:59	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:59	LOAD-MODULE	3464	NA	NA	conhost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.1.234	5355	svchost.exe

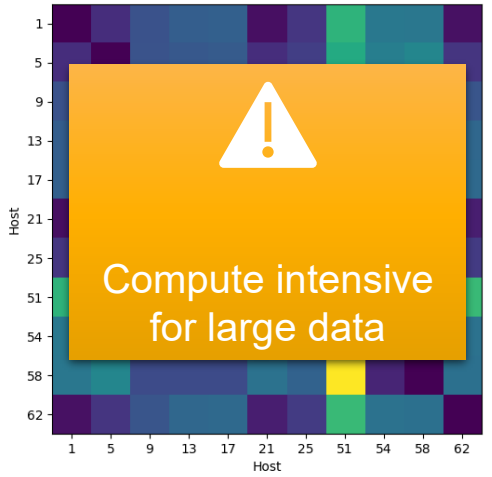
Entity log data
(one table for each entity)

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
⋮					
9/23/23 13:24	START-FLOW	864	10.20.1.202	5355	svchost.exe
9/23/23 13:24	START-FLOW	864	10.20.1.167	5355	svchost.exe
9/23/23 13:24	OPEN-PROCESS	2984	NA	NA	cmd.exe
⋮					
9/23/23 13:19	MESSAGE-FLOW	864	142.20.61.11	60999	svchost.exe
9/23/23 13:19	CREATE-FILE	1200	NA	NA	lwabeat.exe
9/23/23 13:19	MESSAGE-FLOW	4	10.50.1.6	138	system
⋮					
9/23/23 13:29	START-FLOW	4	142.20.57.8	138	system
9/23/23 13:29	START-FLOW	864	10.20.2.104	5355	svchost.exe
9/23/23 13:29	START-FLOW	4	142.20.57.2	138	system
⋮					
9/23/23 23:49	START-FLOW	864	10.50.0.116	5355	svchost.exe
9/23/23 23:49	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:49	START-FLOW	864	10.20.3.171	5355	svchost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.5.25	5355	svchost.exe
9/23/23 23:59	START-FLOW	864	10.20.2.47	5355	svchost.exe
9/23/23 23:59	START-FLOW	4	10.50.1.46	137	system

Hypergraphs
(one for each entity)



Edit Cost Matrix



Outlier score for each entity

Workflow from All Data to Identify Outliers…

And more!

When is an entity in my network
(e.g., user, host, process, …)
"behaving" out of the ordinary?

All log data, all time

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
9/23/23 13:14	START-FLOW	864	142.20.57.199	5355	svchost.exe
9/23/23 13:14	LOAD-MODULE	5628	NA	NA	ping.exe
9/23/23 13:14	MODIFY-FILE	4	NA	NA	system
⋮					
9/23/23 23:59	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:59	LOAD-MODULE	3464	NA	NA	conhost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.1.234	5355	svchost.exe

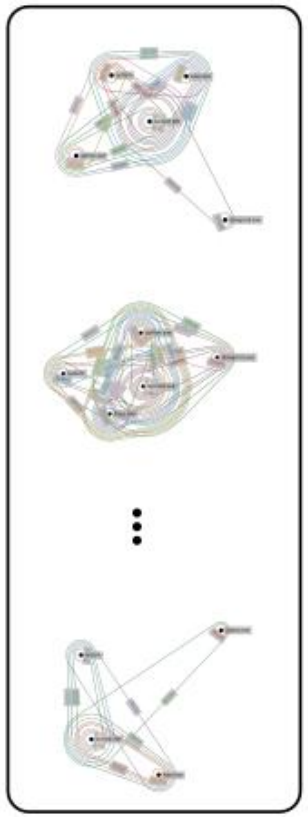
One window of log data

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
9/23/23 13:14	START-FLOW	864	142.20.57.199	5355	svchost.exe
9/23/23 13:14	LOAD-MODULE	5628	NA	NA	ping.exe
9/23/23 13:14	MODIFY-FILE	4	NA	NA	system
⋮					
9/23/23 23:59	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:59	LOAD-MODULE	3464	NA	NA	conhost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.1.234	5355	svchost.exe

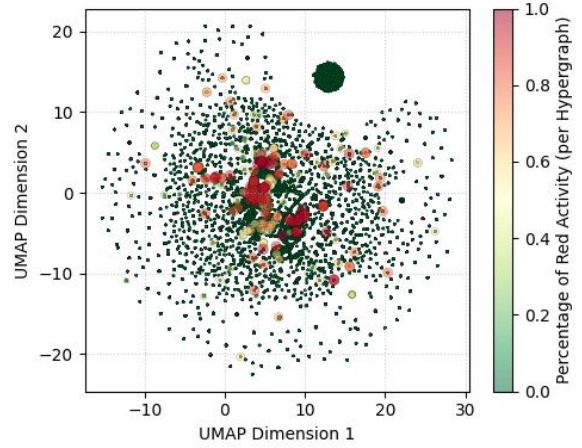
Entity log data
(one table for each entity)

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
⋮					
9/23/23 13:24	START-FLOW	864	10.20.1.202	5355	svchost.exe
9/23/23 13:24	START-FLOW	864	10.20.1.167	5355	svchost.exe
9/23/23 13:24	OPEN-PROCESS	2984	NA	NA	cmd.exe
⋮					
9/23/23 13:19	MESSAGE-FLOW	864	142.20.61.11	60999	svchost.exe
9/23/23 13:19	CREATE-FILE	1200	NA	NA	lwabeat.exe
9/23/23 13:19	MESSAGE-FLOW	4	10.50.1.6	138	system
⋮					
9/23/23 13:29	START-FLOW	4	142.20.57.8	138	system
9/23/23 13:29	START-FLOW	864	10.20.2.104	5355	svchost.exe
9/23/23 13:29	START-FLOW	4	142.20.57.2	138	system
⋮					
9/23/23 23:49	START-FLOW	864	10.50.0.116	5355	svchost.exe
9/23/23 23:49	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:49	START-FLOW	864	10.20.3.171	5355	svchost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.5.25	5355	svchost.exe
9/23/23 23:59	START-FLOW	864	10.20.2.47	5355	svchost.exe
9/23/23 23:59	START-FLOW	4	10.50.1.46	137	system

Hypergraphs
(one for each entity)



High dimensional embedding



Outlier score for each entity

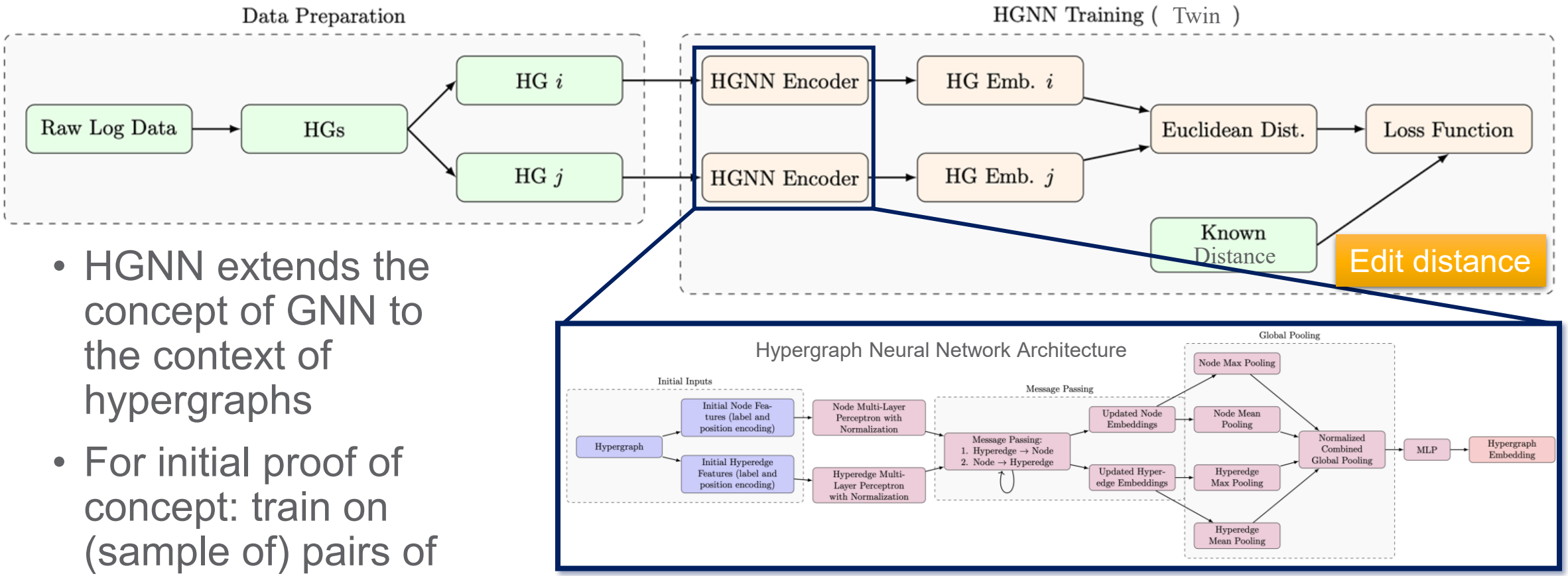
Classification

Clustering

Prediction

Detour: From Distance to Embedding

- Twin Neural Networks are efficient at learning distance measures



- HGNN extends the concept of GNN to the context of hypergraphs
- For initial proof of concept: train on (sample of) pairs of all hypergraphs

Example Walkthrough on OpTC FLOW

Setup

OpTC Day 2 –
24 hours of log data

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
9/23/23 13:14	START-FLOW	864	142.20.57.199	5355	svchost.exe
9/23/23 13:14	LOAD-MODULE	5628	NA	NA	ping.exe
9/23/23 13:14	MODIFY-FILE	4	NA	NA	system
⋮					
9/23/23 23:59	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:59	LOAD-MODULE	3464	NA	NA	conhost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.1.234	5355	svchost.exe

One 20-min window
of log data

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
9/23/23 13:14	START-FLOW	864	142.20.57.199	5355	svchost.exe
9/23/23 13:14	LOAD-MODULE	5628	NA	NA	ping.exe
9/23/23 13:14	MODIFY-FILE	4	NA	NA	system
⋮					
9/23/23 23:59	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:59	LOAD-MODULE	3464	NA	NA	conhost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.1.234	5355	svchost.exe

Entity log data
(one table for each host,
restricted to dominant source IP)

Time	Action-Object	PID	Source IP	Destination Port	Executable
9/23/23 13:14	WRITE-FILE	4	NA	NA	system
9/23/23 13:14	READ-FILE	5076	NA	NA	ping.exe
9/23/23 13:14	COMMAND-SHELL	2952	NA	NA	powershell.exe
⋮					
9/23/23 13:24	START-FLOW	864	10.20.1.202	5355	svchost.exe
9/23/23 13:24	START-FLOW	864	10.20.1.167	5355	svchost.exe
9/23/23 13:24	OPEN-PROCESS	2984	NA	NA	cmd.exe
⋮					
9/23/23 13:19	MESSAGE-FLOW	864	142.20.61.13	60999	svchost.exe
9/23/23 13:19	CREATE-FILE	1200	NA	NA	lwabeat.exe
9/23/23 13:19	MESSAGE-FLOW	4	10.50.1.6	138	system
⋮					
9/23/23 13:29	START-FLOW	4	142.20.57.8	138	system
9/23/23 13:29	START-FLOW	864	10.20.2.104	5355	svchost.exe
9/23/23 13:29	START-FLOW	4	142.20.57.2	138	system
⋮					
9/23/23 23:49	START-FLOW	864	10.50.0.116	5355	svchost.exe
9/23/23 23:49	START-FLOW	2236	10.20.2.66	5000	python.exe
9/23/23 23:49	START-FLOW	864	10.20.3.171	5355	svchost.exe
⋮					
9/23/23 23:59	START-FLOW	864	10.20.5.25	5355	svchost.exe
9/23/23 23:59	START-FLOW	864	10.20.2.47	5355	svchost.exe
9/23/23 23:59	START-FLOW	4	10.50.1.46	137	system

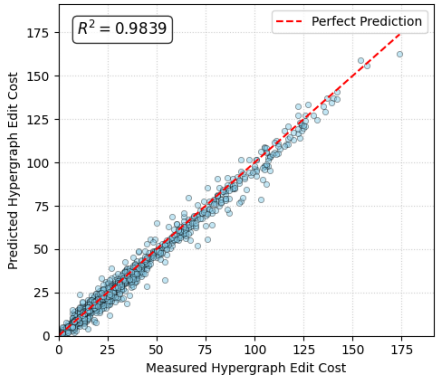
V=(dest_port, dest_IP)
E=(protocol, image_path
(one for each host)

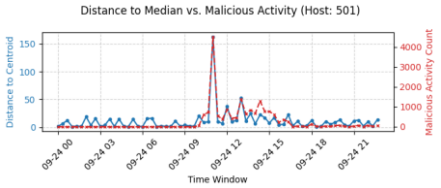
Example walkthrough on OpTC FLOW

Fit and results

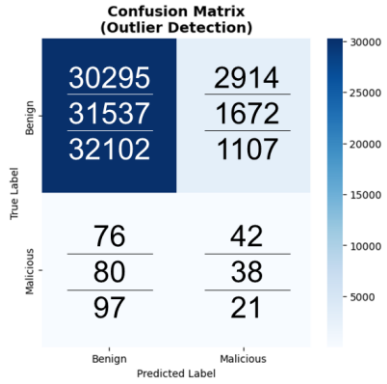
Initial Workflow:

- Trained on:
  - 10% sample of pairs of HGs from day 2





- For each host, for each time window:
  - Compute distance to centroid of all other hosts in window
  - Flag as anomalous if its **modified z-score** distance **> 3, 5, or 7**
  - Ground truth anomaly if any malicious activity

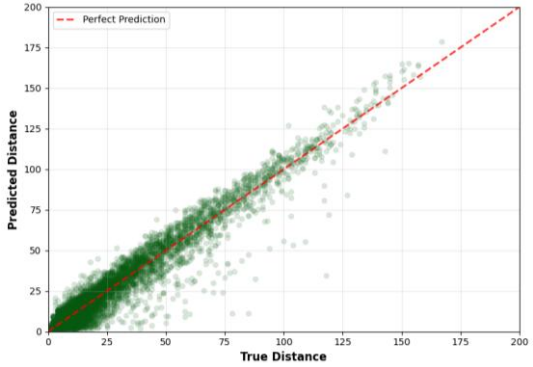


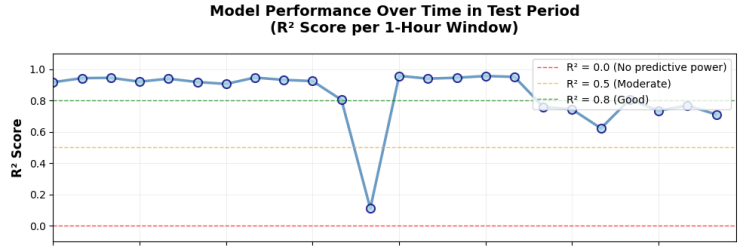
Limitation: trained on sample of **all** data

- Can we train on some initial data and then run in inference mode?
- (How often) is retraining necessary? How to trigger?

- **Experiment:** train model on sample of Day 1, see how it performs on Day 2

Test on Day 2

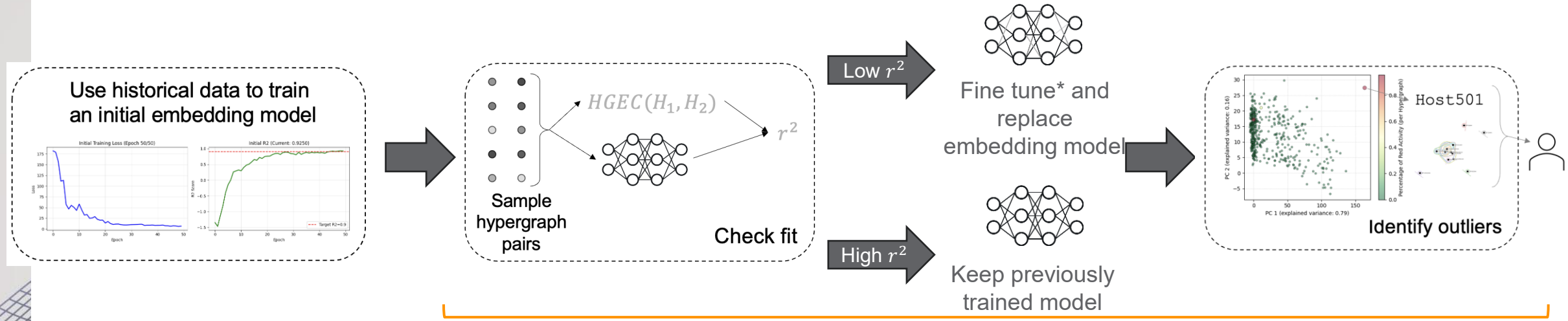




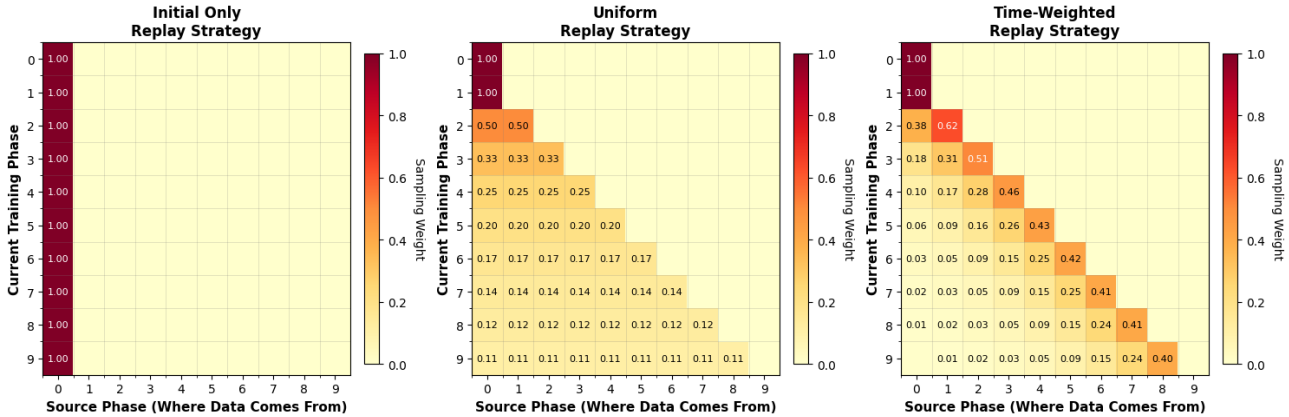
Not pictured: a single outlier with true distance ~4500 and predicted distance ~200

Updated Analytic Pipeline

Dynamic Re-Training

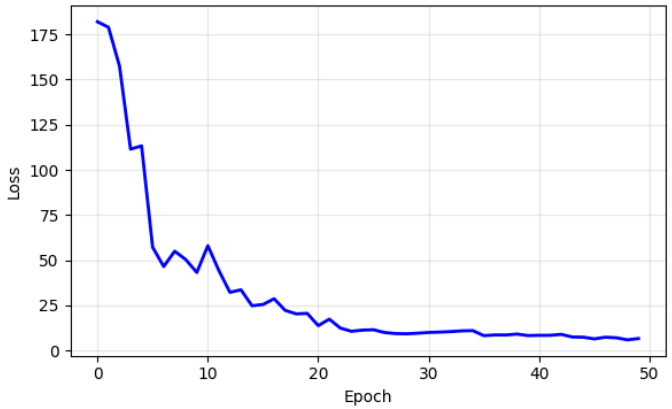


To fine tune use different strategies based on how much want to “remember”

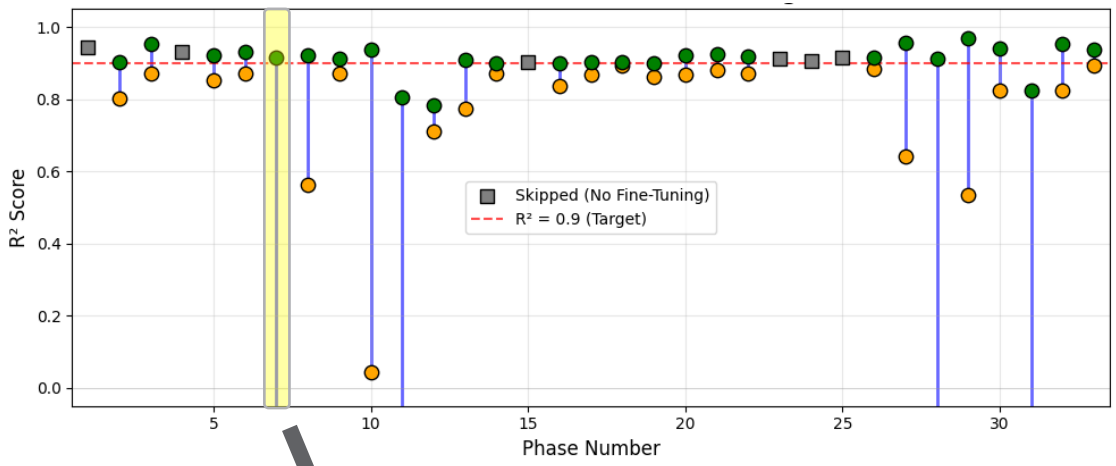


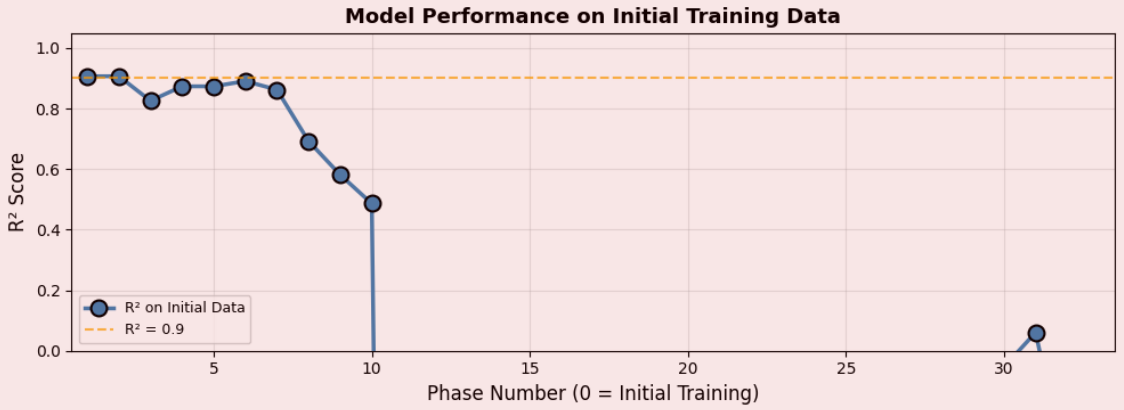
In the experiment in next slides we use Time-Weighted replay with a decay rate of 0.6.

Initial Train (2 hours of data)



Continuous Learning Phases

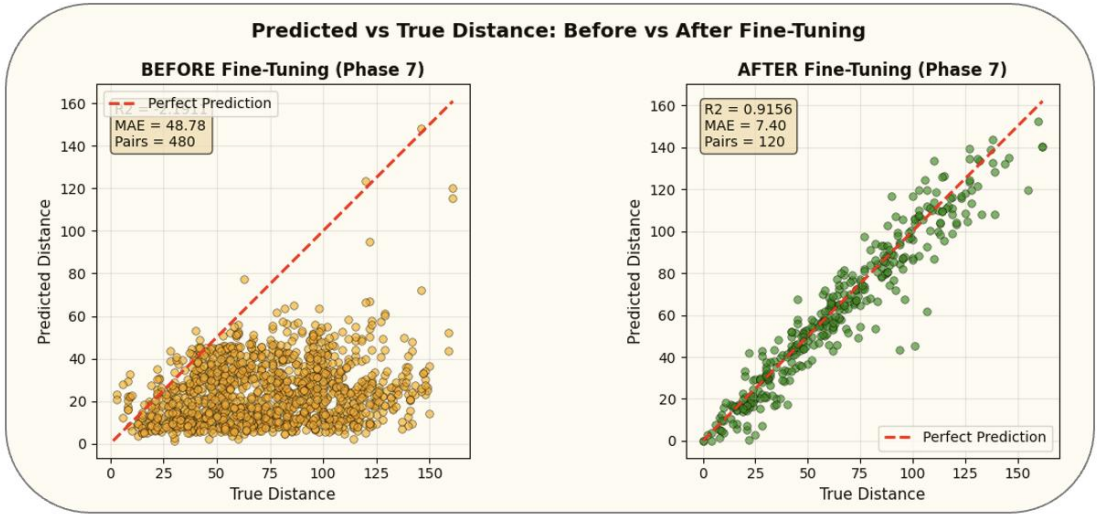




Temporally weighted random sampling of previous hypergraphs causes degradation of performance of initial data over time

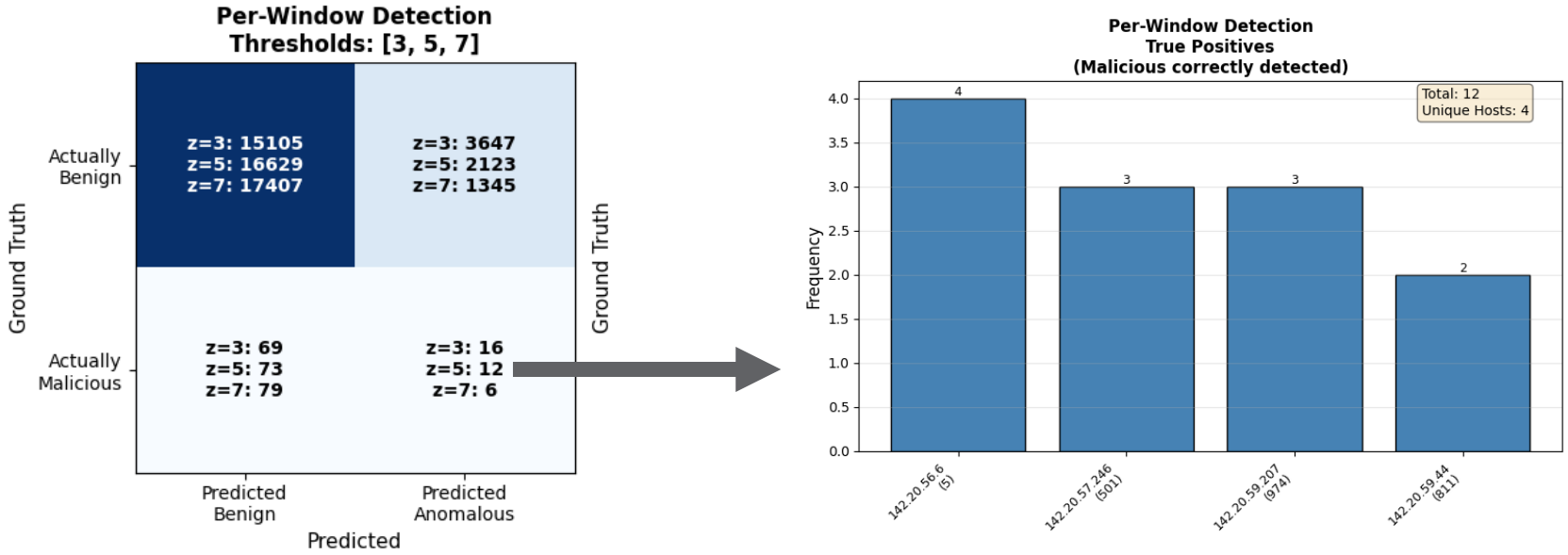
If this is undesirable, a different regularization strategy should be used

Significant improvement in embedding after fine tuning if there is change in data.



Summary of Results for Z=5

- Total malicious hosts in dataset: 5
- Percent of malicious hosts detected: 80% (4/5)
- Percent of all true malicious activity (host, time window pairs) detected (recall): 14.1% (12/85)
- Looking into what the FPs and FNs are and how to decrease both and increase recall



Final thoughts

- Hypergraph edit distance:
  - Measures transformations between hypergraphs
  - Flexible – definition and cost of atomic edits
  - Based on graph edit distance
  - NP-hard to compute exactly but there are approximations
- Hypergraph edit cost:
  - Appropriate when labels matter
  - Computationally tractable
- Examples show utility for entity disambiguation, anomaly detection

Pacific Northwest
NATIONAL LABORATORY

Thank you

Emilie Purvine
emilie.purvine@pnnl.gov