Efficient and Simplify Hypergraph Evolution Measurement Using Entropy

Yu Liu*, Qi Luo†, Mengbai Xiao*, Yanwei Zheng*, Xiuzhen Cheng*, Dongxiao Yu*
*Shandong University, Qingdao, China
†University of New South Wales, Sydney, Australia
yuliu@mail.sdu.edu.cn, qi.luo1@unsw.edu.au, {xiaomb, zhengyw, xzcheng, dxyu}@sdu.edu.cn

***Abstract*—Hypergraphs are powerful tools for modeling complex multidimensional relationships, which are widely applied in social networks, link prediction, and bioinformatics. *Hypergraph evolution* is defined by the dynamic changes in its vertices and hyperedges, which is essential for understanding its structural dynamics. However, existing methods of measuring hypergraph evolution struggle to balance exactness and computational efficiency, limiting their scalability, and these methods fail to provide a concise and interpretable representation, making it challenging to quantify the extent of structural changes effectively. To address these, we propose a novel entropy-driven framework for hypergraph evolution analysis that precisely quantifies structural changes while ensuring computational effectuation. At the key of our approach is the hypergraph information entropy metric, which captures the hypergraph evolution by analyzing fluctuations in vertex and hyperedge probability distributions. This metric provides a concise scalar representation, enabling efficient and interpretable analysis. Building on this foundation, we introduce two applications: hypergraph merging and anomalous vertex/hyperedge detection. Our method leverages vertex and hyperedge similarity for hypergraph merging, grounded in the relationship between similarity and entropy changes. This reduces redundancy while preserving essential structures. For anomaly detection, we propose a theory to dynamically evaluate the impact of different types of hyperedges/vertices on the hypergraph entropy, solving identification of structural abnormalities. Experiments on eight datasets validate the accuracy and scalability of the proposed entropy-based metric. Our method maintains a KL divergence of no more than 0.4 between the original and merged hypergraphs in fundamental and application-specific properties in hypergraph merging. The proposed similarity metric outperforms Jaccard by 20%, with ablation studies showing 10%-15% performance gains from each optimization. In anomaly detection, our method consistently identifies inserted anomalous hyperedges with high accuracy, outperforming existing metrics. Moreover, it achieves over 60% higher efficiency compared to traditional approaches, demonstrating its robustness and computational advantage.**

***Index Terms*—Hypergraph evolution, information entropy, hypergraph merge, anomalous vertex/hyperedge detection**

I. Introduction

The hypergraph [1], [2] offers powerful modeling capabilities for complex structures, finding widespread applications in social networks [3], [4], link prediction [5], [6], and bioinformatics [7]. Unlike traditional pairwise graphs that only capture binary relationships, hypergraphs provide a more expressive framework where each hyperedge can connect multiple vertices simultaneously [8], [9]. An inherent characteristic of hypergraphs is structural evolution [10], [11], especially in large-scale datasets and dynamic systems. Studying this evolutionary behavior is crucial for analyzing and understanding the structural dynamics of hypergraphs. For instance, in anomaly detection [12], recognizing irregular group formation patterns, such as sudden shifts in consumer co-purchasing behaviors, can help identify potential fraudulent activities [13]. In social network analysis [14], monitoring the evolution of community structures can reveal emerging trends or the decline of interest groups. In this paper, we attempt to capture and quantify such dynamic changes in hypergraphs, which is refereed as *Hypergraph Evolution*.

Existing approaches to quantify *Hypergraph Evolution* often utilize general metrics such as Hamming distance [15] and hypergraph edit distance [16]. While these metrics can capture certain aspects of hypergraph properties, they are not specifically designed to measure the hypergraph evolution. Hamming distance provides computational efficiency but oversimplifies hypergraphs, failing to capture intricate changes. In contrast, hypergraph edit distance provides more accurate comparisons but introduces significant computational overhead due to its reliance on the exhaustive hypergraph traversal. Additionally, graph similarity computation [17], [18], designed for static snapshot comparisons, cannot capture continuous dynamic changes or structural evolution within a single hypergraph, such as vertex and hyperedge modifications. Therefore, current approaches struggle to address the unique demands of hypergraph evolution analysis, particularly in capturing dynamic changes with both exact and efficiency. Building on these, we summarize these key challenges as follows:

- Existing methods struggle to balance exact and efficiency. Fast measurement, hamming distance, approaches are often overly simplified, capturing only shallow changes in hypergraphs and failing to reflect the multidimensional complexity of dynamic hypergraphs. On the other hand, accurate methods like hypergraph edit distance can capture intricate dynamic evolutions but are computationally expensive, as they require traversal of hypergraph vertices and hyperedges to compute the minimal edit operations, particularly in large-scale or real-time scenarios. This trade-off between fast but imprecise and accurate but slow is a core bottleneck in

The corresponding author is Qi Luo.

hypergraph evolution analysis.

- Hypergraph edit distance, as a representative precise measurement method, relies on elaborate computational results to represent differences between hypergraphs. It often uses a collection of minimal edit operations as metrics, leading to overly complex and less intuitive representations, especially in large-scale hypergraphs. This complexity makes it difficult to directly determine the extent of hypergraph evolution, hindering its straightforward application in various hypergraph processing tasks.

To overcome these challenges, we propose a novel information entropy-driven approach to measure hypergraph evolution, which leverages the probability distributions of vertices and hyperedges to precisely quantify their evolving relationships.

Exact and Efficient Measurement: To address the challenges of balancing exact and efficiency, our method integrates information entropy as the core metric, leveraging vertex and hyperedge probability distributions to precisely quantify hypergraph dynamics. By capturing structural changes such as scale variations, neighborhood adjustments, and hyperedge size shifts, our approach ensures exceptional accuracy. Simultaneously, the reliance on probability-driven entropy calculations eliminates the need for traversal-based operations, enabling independent and parallel processing of vertex and hyperedge entropy. This design significantly reduces computational costs, ensuring scalability and efficiency for large-scale datasets and real-time hypergraph systems. By combining precision with computational efficiency, our method effectively resolves the trade-off between fast but imprecise and accurate but slow. **Simplicity of Results:** In contrast to traditional methods that rely on complex representations such as hypergraph edit distance, our approach computes hypergraph evolution into a single scalar value. This concise and interpretable representation simplifies the analysis of structural changes, facilitates rapid and intuitive comparisons between hypergraphs, and enables seamless integration into various algorithms and applications.

Our approach integrates precise measurement with efficient computation, achieving an optimal balance between accuracy and speed. By quantifying hypergraph dynamics as a single scalar value, it simplifies the analysis of complex structures while ensuring computational scalability. This versatility makes the method suitable for diverse applications, effectively capturing intricate hypergraph evolution and detecting subtle structural changes.

Applications. To demonstrate the applicability of our approach, we present two representative applications. The first is hypergraph merging, which aggregates similar vertices and hyperedges while retaining key structural features. The second is anomalous vertex/hyperedge detection, aimed at identifying significant structural changes in evolving hypergraphs. These application cases highlight the adaptability, precision, and computational efficiency of our entropy-driven approach.

Hypergraph Merging: Existing hypergraph merging methods often rely on exploring the topological substructures of hypergraphs, such as subgraph pattern matching [19], [20]. This process requires complex traversal and computation, which becomes computationally expensive, especially in large-scale hypergraphs. To address these challenges, we propose a similarity metric based on entropy changes, replacing traditional topological structure analysis with similarity computations between vertices and hyperedges, thereby significantly reducing computational complexity.

However, implementing this method involves several theoretical and technical challenges. The first challenge is to ensure that the similarity metric accurately reflects the semantic relationships between vertices and hyperedges while remaining consistent with changes in information entropy. Another critical issue is balancing entropy loss during the merging process to preserve core information and maintain structural features. Furthermore, the design of efficient algorithms plays a key role in processing large-scale hypergraphs, as it directly impacts the feasibility and scalability of the approach. To address these issues, we define an entropy-driven similarity metric with theoretical guarantees. This metric quantifies the similarity of vertices and hyperedges based on differences in their probability distributions, directly linking similarity to changes in information entropy. Additionally, a tolerance threshold for entropy loss is introduced during merging, ensuring the retention of high-entropy vertices and hyperedges to maintain the integrity of the core information. Finally,we introduce a sampling-based similarity computation method that focuses on neighborhood regions to enhance efficiency.

Anomalous Vertex/Hyperedge Detection: A key theoretical challenge in anomalous vertex/hyperedge detection lies in accurately determining how the insertion of different types of hyperedges impacts the information entropy of the hypergraph. Hyperedges of varying sizes and compositions contribute differently to the overall entropy, depending on their connectivity patterns, the probability distributions of their associated vertices. This variability makes it difficult to establish a uniform criterion for assessing the impact of new hyperedges.

To address this, we provide a theoretical proof that establishes the rationality of dynamically evaluating the entropy contribution of each hyperedge type by analyzing the shifts in probability distributions caused by its insertion. By monitoring fluctuations in information entropy based on this proof, we can capture the nuanced effects of different hyperedge insertions on the hypergraph's structure. This allows us to rigorously determine whether a new hyperedge aligns with the existing structure and preserves the overall integrity of the hypergraph. Additionally, our method leverages these entropy-driven insights to enhance the efficiency of dynamic change detection. Rather than relying on computationally expensive traversal methods, we focus on localized probability changes and similarity metrics, enabling scalable and accurate detection of structural variations in large-scale and evolving hypergraphs.

Our contributions are as follows:

1) **Hypergraph Information Entropy:** We propose a novel hypergraph information entropy metric, which quantifies hypergraph evolution by analyzing entropy fluctuations

derived from vertex and hyperedge probability distributions. By leveraging the inherent properties of entropy, this approach effectively captures dynamic structural changes while ensuring computational efficiency.

2) **Entropy-Driven Hypergraph merging:** We present an innovative hypergraph merging approach that integrates entropy changes with a similarity metric. By calculating similarities between vertices and hyperedges, whose probabilities contribute to entropy computation, the method identifies highly similar pairs for merging. It significantly eliminates redundancy and retains the core structure and essential information. Furthermore, we provide rigorous theoretical proof demonstrating that entropy changes effectively capture structural redundancies, ensuring that the merging process preserves the hypergraph's core characteristics and critical information.
3) **Efficient Anomalous Vertex/Hyperedge Detection:** We introduce an innovative detection method specifically designed for dynamic hypergraphs. By leveraging information entropy changes derived from vertex and hyperedge probabilities, this method enables precise and efficient evaluation of structural changes. Combining high accuracy with computational efficiency, it serves as a robust solution for detecting dynamic variations in hypergraphs. Additionally, we establish a theoretical foundation to validate that entropy changes accurately reflect the impact of newly added or removed hyperedges, ensuring structural consistency and reliability in anomalous vertex/hyperedge detection tasks.
4) **Extensive Experiments:** Our experiments on eight datasets demonstrate the effectiveness of the proposed entropy-driven method in two applications. For hypergraph merging, the method preserves key properties while limiting KL divergence to 0.4 and improves the similarity metric by 20% over Jaccard. Ablation studies reveal 10%-15% performance gains from each optimization. The method also scales well for large datasets. For anomalous vertex/hyperedge detection, it achieves superior accuracy in identifying abnormal hyperedges, surpassing existing metrics, and boosts efficiency by over 60%.

II. Related Work

Here we review related work from three aspects: measuring the evolution in hypergraphs, reducing the scale of hypergraphs, and anomalous vertex/hyperedge detection.

Hypergraph Evolution Measurement: Traditional metrics and entropy-driven methods provide complementary insights. Pinheiro et al. introduced Hamming Distance [15], a straightforward measure that captures element-level differences but lacks the ability to reflect global structural changes in hypergraphs. LeFevre et al.'s Reconstruction Error (RE) [21] assesses approximation quality, though it does not fully capture the intricate structure of hypergraphs. Qin et al. proposed Hypergraph Edit Distance [16], a more comprehensive metric for structural comparison; however, its computational intensity limits its scalability to large hypergraphs. Entropy-based methods offer a deeper understanding of hypergraph structural changes. Bloch et al. developed an entropy measure based on local structures within hypergraphs [22], producing an entropy vector that quantifies the complexity of relationships within various regions, thus suitable for fine-grained structural analysis. Hu et al. introduced von Neumann entropy [23] to identify critical nodes, accounting for higher-order interactions in complex networks, which enhances the identification of key structural features. Chen et al. proposed an entropy measure using Higher-Order Singular Value Decomposition of Laplacian tensors [24], offering a comprehensive approach to understanding structural complexity in uniform hypergraphs.

Reducing Hypergraph Scale: Sampling and compressing methods are commonly employed. Sampling techniques focus on selecting representative subsets to retain primary characteristics. For instance, Chen et al.'s graph sparsification [25] reduces edge counts to lower computational complexity, and Gou et al.'s Graph Stream Sketch [26] compresses vertices through hash mapping to generate subgraphs. Although these methods effectively improve computational efficiency, they often provide only local approximations and may fail to preserve the global structure essential in capturing high-order interactions within complex networks. Conversely, merging/compressing methods emphasize retaining global structural integrity. Borici et al. proposed a graph compression technique based on semantic features [27], which partitions and maps the graph into a hypergraph framework, enhancing information representation completeness. Riondato et al. applied k-means clustering to merge vertices into super-nodes and super-edges [28], minimizing reconstruction error to suit applications requiring global structural retention.

Anomalous Vertex/Hyperedge Detection: Anomalous vertex/hyperedge detection, crucial for understanding complex systems, focuses on efficiently identifying structural changes over time in large-scale hypergraphs. Traditional methods like full traversal or repetitive sampling are often computationally prohibitive. Recent advancements address these challenges: hypergraph change point detection leverages the symmetrized combinatorial Laplacian for efficient structural analysis [29]; Yan et al. [30] introduced hypergraph contrastive learning to detect pair-wise and multivariate interaction anomalies; Surana et al. [31] proposed transforming hypergraphs into graphs for standard dissimilarity measures or using tensor methods for multi-way relations. These approaches underscore tailored techniques that exploit hypergraph structures.

III. Preliminaries

We consider a weighted and undirected hypergraph $G = (V, E, W)$. V denotes the set of vertices, and E denotes the set of hyperedges, where each hyperedge is composed of multiple vertices. W is a weight function that assigns weights to vertices and hyperedges, denoted as $W_G(v)$ for each vertex v and $W_G(e)$ for each hyperedge e. The weights reflect the merging history of the hyperedges and vertices. Initially, each vertex and hyperedge is assigned a weight of 1, representing the base state where no merging operations have been performed. The incident hyperedge set $N_G(v)$ for a vertex v includes all hyperedges that contain v. The degree of a vertex, $d_G(v)$, is the sum of the weights of its incident hyperedges, while the cardinality of a hyperedge, $c_G(e)$, is the sum of the weights of its vertices. The symbol of the hypergraph is omitted in contexts where it is clear.

IV. Hypergraph Information Entropy

In this section, we explain our approach from three key aspects: the definition of information entropy, problem definition, and base Observation.

A. Hypergraph Information Entropy Definition

Entropy [32]–[34] is a fundamental tool for quantifying uncertainty and information content in a system. Viewing a hypergraph as a complex information set, we extend the concept of entropy to provide a novel framework for quantifying its structural changes. Hypergraph information entropy enables precise and efficient evaluation of the impacts of adding or removing vertices and hyperedges, offering a powerful lens to analyze hypergraph evolution. Recognizing the dual-layered nature of hypergraphs, where vertices represent entities and hyperedges define high-order relationships among them, we design the metric from two complementary perspectives: vertex information entropy and hyperedge information entropy. This dual-perspective design allows for a nuanced and detailed analysis, effectively reflecting both local and global structural characteristics of hypergraphs. Specifically, we consider the following perspectives:

- **Vertex-Centric Perspective:** Vertices are the building blocks of a hypergraph, and their connections to hyperedges define the local interaction patterns. By measuring the degree distribution of vertices, we quantify how evenly or unevenly vertices are connected. A more uneven distribution indicates higher complexity, making vertex information entropy a natural choice for assessing local structural uncertainty.
- **Hyperedge-Centric Perspective:** Hyperedges encapsulate the high-order interactions among vertices, defining the global relationship patterns of the hypergraph. By quantifying the diversity in hyperedge sizes and compositions, hyperedge information entropy captures the richness of these interactions, reflecting the hypergraph's global structural complexity.
- **Dynamic Adaptability:** Hypergraphs are dynamic systems, where vertices and hyperedges are frequently added or removed. The decomposition into vertex and hyperedge components allows us to separately and precisely capture the impacts of these changes, ensuring that our metric remains sensitive to both local and global dynamics while maintaining computational efficiency.

In the following, we define several key concepts for quantifying information entropy in hypergraphs, focusing on both vertex and hyperedge.

Definition 1: (**Vertex Probability**) Given a hypergraph $G = (V, E, W)$ and a vertex $u \in V$, the probability of u is defined as:

$$p_G(v) = \frac{d_G(v)}{\sum_{e \in E} |e|}, \quad v \in V \tag{1}$$

where $|E|$ is the total number of hyperedges in G.

Definition 2: (**Vertex Information Entropy**) Given a hypergraph $G = (V, E, W)$, the vertex information entropy of G is defined as:

$$H_V(G) = -\sum_{v \in V} W_G(v) \cdot p_G(v) \cdot \ln p_G(v) \tag{2}$$

Definition 3: (**Hyperedge Probability**) Given a hypergraph $G = (V, E, W)$ and a hyperedge $e \in E$, the probability of e is defined as:

$$p_G(e) = \frac{c_G(e)}{\sum_{v \in V} |N(v)|}, \quad e \in E \tag{3}$$

where $|V|$ is the total number of vertices in G.

Definition 4: (**Hyperedge Information Entropy**) Given a hypergraph $G = (V, E, W)$, the hyperedge information entropy of G is defined as:

$$H_E(G) = -\sum_{e \in E} W_G(e) \cdot p(e) \cdot \ln p(e) \tag{4}$$

Building on vertex information entropy and hyperedge information entropy, we introduce the concept of hypergraph information entropy.

Definition 5: (**Hypergraph Information Entropy**) Given a hypergraph $G = (V, E, W)$ and a parameter $\alpha \in [0, 1]$, which controls the ratio between vertex information entropy and hyperedge information entropy, the hypergraph information entropy is defined as:

$$H(G) = \alpha \cdot H_V(G) + (1 - \alpha) \cdot H_E(G) \tag{5}$$

Equation 5 elegantly balances the contributions of vertex and hyperedge entropy, providing a comprehensive measure of the overall complexity of a hypergraph.

B. Problem Definition

As hypergraphs evolve with dynamic changes in vertices and hyperedges, measuring structural changes becomes essential. Traditional graph metrics are inadequate for capturing the complexity of hypergraph relationships, necessitating specialized metrics to quantify these structural changes. To address this, we propose a hypergraph entropy-driven metric to measure hypergraph evolution, and we apply it to two key areas: hypergraph merging, where identifying core structures reduces storage needs, and anomalous vertex/hyperedge detection, enabling real-time monitoring of structural changes. We provide specific solutions for each application to achieve efficient and accurate hypergraph change detection.

C. Base Observation

This section presents key observations on the probabilities of vertices and hyperedges in a large-scale real-world hypergraph.

Observation 1: In large-scale hypergraphs $G = (V, E, W)$, typically found in real-world datasets, the probabilities associated with both vertices and hyperedges tend to be significantly lower than $\exp(-1)$. Formally:

$$\begin{cases} p(v) < \exp(-1), & \forall v \in V \\ p(e) < \exp(-1), & \forall e \in E \end{cases} \tag{6}$$

Proof: Observation 1 applies predominantly to large-scale hypergraphs in practical applications. Although this observation may not hold for small hypergraphs or those with few hyperedges, empirical analyses of diverse real-world hypergraphs show that, when the hypergraph contains thousands of vertices and hyperedges, the probabilities $p(v)$ for vertices and $p(e)$ for hyperedges are consistently below $\exp(-1)$. This pattern, observed across numerous large-scale hypergraphs, substantiates the validity of this observation in practical contexts. ■

V. Entropy-Driven Hypergraph Merging

In this section, we propose an entropy-driven framework for hypergraph merging, leveraging information entropy as a guiding principle to ensure minimal structural and informational loss. We begin by defining similarity measures for vertices and hyperedges, which are fundamental to optimizing the merging process. Next, we provide theoretical analyses to demonstrate how merging impacts hypergraph entropy, establishing conditions for minimizing entropy changes. Finally, we introduce efficient algorithms, including a sampling-based approach and an adaptive merging strategy, to enhance computational efficiency and scalability while preserving hypergraph integrity.

A. Vertex/Hyperedge Similarity

In the process of hypergraph merging, accurately measuring the similarity between vertex pair and hyperedge pair is fundamental. An effective similarity calculation metric is crucial for minimizing information loss. As outlined in Theorem 3 and Theorem 4, the larger the intersection of the incident hyperedge sets of vertex pairs or hyperedge pairs, the smaller the entropy loss during the merging process. Building on this insight, we have developed specialized metrics that precisely calculate the similarity between vertices and hyperedges. These metrics not only enable accurate similarity assessments but also significantly reduce potential information entropy loss during merging, ensuring that the merged hypergraph retains as much structural and informational integrity as possible.

1) ***Vertex Similarity****:* In hypergraph, vertex similarity is primarily determined by their shared hyperedges. Traditionally, Jaccard similarity [35] quantifies vertex similarity as the ratio of shared to total hyperedges, as follows:

$$S(v_0, v_1) = \frac{|N(v_0) \cap N(v_1)|}{|N(v_0) \cup N(v_1)|} \tag{7}$$

Although intuitive, Equation 7 neglects the internal structure and varying sizes of hyperedges, potentially skewing results. Given the variability in hyperedge sizes, this oversight can lead to skewed similarity assessments. To address this limitation, we refine the Jaccard approach by incorporating the cardinality of the hyperedges, resulting in an enhanced metric that accounts for both the presence and significance of shared hyperedges. The revised equation incorporates both the overlap of hyperedge sets and their cardinalities, yielding a weighted similarity measure:

$$S(v_0, v_1) = \frac{|N(v_0) \cap N(v_1)|}{|N(v_0) \cup N(v_1)|} \cdot \frac{\sum_{e \in N(v_0) \cap N(v_1)} c(e)}{\sum_{e \in N(v_0) \cup N(v_1)} c(e)} \tag{8}$$

Equation 8 enhances traditional similarity measures by considering both the sets of hyperedges incident to two vertices and the sum of the cardinalities of these hyperedges. This refined measure captures both shared connections and their cardinalities, providing a more accurate vertex similarity assessment.

2) ***Hyperedge Similarity****:* In hypergraphs, hyperedge similarity is primarily determined by the vertices they share. Traditionally, Jaccard similarity is defined as the ratio of shared to distinct vertices between hyperedges:

$$S(e_0, e_1) = \frac{|\{v \mid v \in e_0 \cap e_1\}|}{|\{v' \mid v' \in e_0 \cup e_1\}|} \tag{9}$$

Equation 9 has limitations, particularly for high-degree vertices, as it ignores their broader network context. To address this limitation, we propose a refined method that incorporates vertex degrees, emphasizing that hyperedges sharing vertices with higher degrees are structurally more similar. Equation 10 captures this refinement by considering both the number of shared vertices and the sum of their degrees:

$$S(e_0, e_1) = \frac{|\{v \mid v \in e_0 \cap e_1\}|}{|\{v \mid v \in e_0 \cup e_1\}|} \cdot \frac{\sum_{v \in e_0 \cap e_1} d(v)}{\sum_{v \in e_0 \cup e_1} d(v)} \tag{10}$$

Equation 10 refines hyperedge similarity by considering both the number of shared vertices and their degrees, capturing their connectivity within the broader network. This approach accounts for direct connections and their broader impact, providing a more comprehensive similarity measure.

B. Theoretical Analysis

Building on Observation 1, we analyze how hyperedge information entropy evolves during the merging process, providing insights into the structural and informational transformations of the hypergraph.

Theorem 1: Given a hypergraph $G = (V, E, W)$ and two hyperedges e_0 and e_1, let the resulting hypergraph after merging e_0 and e_1 be denoted by $G' = (V', E', W')$. Then, the information entropy satisfies:

$$H(G') \geq H(G) \tag{11}$$

Proof: Let the new hyperedge in G' formed by merging e_0 and e_1 be denoted as e, with weight $W(e) = W(e_0) + W(e_1)$.

We examine the change in hyperedge information entropy as follows:

$$\begin{aligned}\Delta H &= H(G') - H(G)\\ &= -(W(e_0) + W(e_1)) \cdot p(e) \cdot \ln p(e)\\ &\quad + W(e_0) \cdot p(e_0) \cdot \ln p(e_0) + W(e_1) \cdot p(e_1) \cdot \ln p(e_1)\\ &= W(e_0) \cdot (p(e_0) \cdot \ln p(e_0) - p(e) \cdot \ln p(e))\\ &\quad + W(e_1) \cdot (p(e_1) \cdot \ln p(e_1) - p(e) \cdot \ln p(e)).\end{aligned} \tag{12}$$

Since the hyperedges e_0 and e_1 are merged into a new hyperedge e, the cardinality of e satisfies $c(e) \geq c(e_0)$ and $c(e) \geq c(e_1)$. According to Equation 3, this implies $p(e_0) \leq p(e)$ and $p(e_1) \leq p(e)$. The function $f(x) = x \ln x$ is strictly increasing on the interval $(0, \exp(-1)]$, and since $0 < p(e_0), p(e_1), p(e) \leq \exp(-1)$, it follows that:

$$\begin{cases} p(e_0) \leq p(e) \implies p(e_0) \ln p(e_0) \leq p(e) \ln p(e)\\ p(e_1) \leq p(e) \implies p(e_1) \ln p(e_1) \leq p(e) \ln p(e)\end{cases} \tag{13}$$

Therefore, $\Delta H \geq 0$ indicates that the information entropy of the hypergraph increases after the merge. ■

The results of Theorem 1 confirm that merging hyperedges increases the information entropy of the hypergraph. By creating hyperedges that encompass more vertices, the structural complexity grows, directly contributing to higher entropy. Notably, this process does not reduce the total number of hyperedges.

Theorem 2: Given a hypergraph $G = (V, E, W)$ and two vertices $v_0, v_1 \in V$, let the resulting hypergraph after merging v_0 and v_1 be denoted by $G' = (V', E', W')$. Then, it holds that:

$$H(G') \geq H(G) \tag{14}$$

To analyze the increase in entropy more effectively, we propose a new theorem that identifies the conditions for minimizing changes in hyperedge information entropy, thereby facilitating more efficient hypergraph merging strategies.

Theorem 3: Let $G = (V, E, W)$ be a hypergraph, and $e_0, e_1 \in E$ be two distinct hyperedges. Define the merged hyperedge $e = e_0 \cup e_1$. The change in hyperedge information entropy ΔH resulting from merging e_0 and e_1 is minimized when the cardinality of their intersection, $|e_0 \cap e_1|$, is maximized. Therefore, we have:

$$|e_0 \cap e_1| \text{ maximized} \Rightarrow \Delta H \text{ minimized} \tag{15}$$

Proof: The change in hyperedge information entropy after merging e_0 and e_1 can be expressed as:

$$\begin{aligned}\Delta H = &- W(e) \cdot p(e) \cdot \ln p(e)\\ &+ W(e_0) \cdot p(e_0) \cdot \ln p(e_0) + W(e_1) \cdot p(e_1) \cdot \ln p(e_1),\end{aligned} \tag{16}$$

where $p(e)$ denotes the probability of the new hyperedge e formed by merging e_0 and e_1.

Substituting $p(e)$ from Equation 3, the expanded form of ΔH is

$$\begin{aligned}\Delta H = &- W(e) \cdot \frac{c(e)}{\sum_{e \in E} |e|} \cdot \ln \frac{c(e)}{\sum_{e \in E} |e|}\\ &+ W(e_0) \cdot \frac{c(e_0)}{\sum_{e \in E} |e|} \cdot \ln \frac{c(e_0)}{\sum_{e \in E} |e|}\\ &+ W(e_1) \cdot \frac{c(e_1)}{\sum_{e \in E} |e|} \cdot \ln \frac{c(e_1)}{\sum_{e \in E} |e|}\end{aligned} \tag{17}$$

Since e_0 and e_1 are fixed, Equation 16 indicates that ΔH decreases as $c(e)$, the size of new hyperedge, is minimized. Using Inclusion-Exclusion principle [36], $c(e)$ is expressed as:

$$|c(e)| = |c(e_0 \cup e_1)| = |c(e_0)| + |c(e_1)| - |c(e_0 \cap e_1)|. \tag{18}$$

Thus, minimizing $c(e)$ requires maximizing the intersection term $c(e_0 \cap e_1)$. This result implies that hyperedges with larger intersections lead to smaller changes in hyperedge information entropy after merging, making such pairs preferable for minimizing entropy increases. ■

Following a similar approach to Theorem 3, we can derive the conditions for minimizing the change in vertex information entropy during the merging of vertices.

Theorem 4: Let $G = (V, E, W)$ be a hypergraph, and $u, v \in V$ be two distinct vertices with adjacent hyperedge sets $N(u)$ and $N(v)$. Define the merged vertex $w = u \cup v$ with an adjacent hyperedge set $N(w) = N(u) \cup N(v)$. The change in vertex information entropy ΔH resulting from merging u and v is minimized when the cardinality of the intersection of their adjacent hyperedge sets, $|N(u) \cap N(v)|$, is maximized. Therefore, we have:

$$\max_{u,v \in V} |N(u) \cap N(v)| \Rightarrow \min_{u,v \in V} \Delta H \tag{19}$$

Next, we introduce a theorem that serves as the foundation for subsequent optimizations, enabling the analysis of changes in hypergraph entropy during the merging of multiple vertices or hyperedges.

Theorem 5: Let $G = (V, E, W)$ be a hypergraph and let $S_E \subseteq E$ be a set of hyperedges. Define $H(e_{all})$ as the information entropy obtained from the simultaneous merging of all hyperedges in S_E and $H(e_{pairwise})$ as the information entropy obtained from the pairwise merging of the hyperedges in S_E. Then, it holds that:

$$H(e_{all}) = H(e_{pairwise}) \tag{20}$$

Proof: The calculation of hyperedge information entropy, as defined in Equation 4, depends solely on the final structure of the hypergraph and is unaffected by the sequence of merging operations. Therefore, whether the hyperedges in S_E are merged simultaneously or through a series of pairwise mergers, the resulting hypergraph structure remains the same. Consequently, the hyperedge information entropy obtained from these two merging methods is identical, leading to the conclusion:

$$H(e_{all}) = H(e_{pairwise}) \tag{21}$$

■

Similarly, a corresponding theorem applies to vertex information entropy during the merging of multiple vertices.

Theorem 6: Let $G = (V, E, W)$ be a hypergraph and let $S_V \subseteq V$ be a set of vertices. Define $H(v_{all})$ as the information entropy obtained from the simultaneous merging of all vertices in S_V and $H(v_{pairwise})$ as the information entropy obtained from the pairwise merging of the vertices in S_V. Then, it holds that:

$$H(v_{all}) = H(v_{pairwise}) \tag{22}$$

Finally, we present the relationship between the similarity threshold and hypergraph information entropy, supported by experimental demonstrations conducted in the laboratory.

Theorem 7: Let $G = (V, E, W)$ be a hypergraph, and let $v_0, v_1 \in V$ be two distinct vertices. The similarity $S(v_0, v_1)$, defined in Equation 8, affects the change in vertex information entropy ΔH resulting from merging v_0 and v_1. Specifically, as $S(v_0, v_1)$ increases, the change in information entropy ΔH decreases. Formally, we have:

$$S(v_0, v_1) \uparrow \Rightarrow \Delta H \downarrow \tag{23}$$

Proof: 1) **Increase in Common Adjacent Hyperedges:** An increase in $S(v_0, v_1)$ implies a larger ratio of the intersection $|N(v_0) \cap N(v_1)|$ to the union $|N(v_0) \cup N(v_1)|$. This indicates that a greater proportion of hyperedges are shared between v_0 and v_1, which results in a more efficient merging process and thus reduces the change in information entropy, as many connections are preserved. 2) **Increase in Shared Hyperedge Weights:** Additionally, as $S(v_0, v_1)$ increases, the ratio of the sum of weights $\sum_{e \in N(v_0) \cap N(v_1)} c(e)$ in the intersection to the sum of weights $\sum_{e \in N(v_0) \cup N(v_1)} c(e)$ in the union also increases. This suggests that a larger portion of the total hyperedge weight is concentrated in the shared connections, maximizing the informational content in the merged vertex and further minimizing the change in entropy.

In summary, as the similarity $S(v_0, v_1)$ increases, both the efficiency in maintaining common hyperedges and the concentration of hyperedge weights lead to a decrease in the change of vertex information entropy ΔH. Thus, we conclude that:

$$S(v_0, v_1) \uparrow \Rightarrow \Delta H \downarrow \tag{24}$$

■

Similarly, we can give the relationship between the hyperedge similarity threshold and the hypergraph entropy:

Theorem 8: Let $G = (V, E, W)$ be a hypergraph, and $e_0, e_1 \in E$ be two distinct hyperedges. As the similarity $S(e_0, e_1)$ increases, the change in hyperedge information entropy ΔH resulting from merging e_0 and e_1 decreases. Formally, we have:

$$S(e_0, e_1) \uparrow \Rightarrow \Delta H \downarrow \tag{25}$$

Algorithm 1 EDHM-Sample

```
Input: Hypergraph $G = (V, E, W)$;
       Hypergraph information entropy change threshold $\psi$;
       Vertex/Hyperedge similarity threshold $\theta$;
       The count of merging operations $C$;
Output: mergeded hypergraph $G^{'} = (V^{'}, E^{'}, W^{'})$;
 1: $G^{'} \leftarrow G$;
 2: $c \leftarrow 0$;
 3: while $\frac{H(G^{'})-H(G)}{H(G)} \leq \psi$ or $c \leq C$ do
 4:     repeat
 5:         Randomly sample the vertex pair $(u, v)$ from $G^{'}$;
 6:     until $S(u, v) > \theta$
 7:     Merge the vertex pair $(u, v)$ in $G^{'}$;
 8:     repeat
 9:         Randomly sample the hyperedge pair $(e_0, e_1)$ from
            $G^{'}$;
10:     until $S(e_0, e_1) > \theta$
11:     Merge the vertex pair $(e_0, e_1)$ in $G^{'}$;
12:     $c \leftarrow c + 1$;
    return $G'$;
```

C. Entropy-driven Hypergraph Merging Using Sampling

To mitigate the computational inefficiency during merging, we propose a sampling-based approach to accelerate the process. This method utilizes random sampling of vertex and hyperedge pairs, assessing their similarity against a predefined threshold that serves as the merging criterion. By focusing on sampled pairs, the approach significantly reduces computational overhead, improving both efficiency and practicality. In each iteration, the algorithm randomly selects vertex or hyperedge pairs and evaluates their similarity to ensure that merging results in minimal changes to hypergraph information entropy. This process is repeated until no pairs satisfy the similarity threshold, signaling the completion of the merging operation. The detailed implementation is outlined in Algorithm 1.

Algorithm 1 accelerates hypergraph merging by using a sampling-based approach, reducing the hypergraph's size while preserving its informational structure. The inputs include the hypergraph $G = (V, E, W)$, an entropy change threshold ψ, a similarity threshold θ, and a merge operation limit C. The output is a smaller, merged hypergraph $G' = (V', E', W')$.

The algorithm initializes G' as G and a counter c to zero (Lines 1-2). It then iteratively performs random sampling of vertex and hyperedge pairs, merging those with similarity exceeding θ, until the entropy change threshold ψ or the operation limit C is reached (Lines 3-12). Each successful merge increments the counter c.

The key advantage of this method is its computational efficiency, achieved by selectively merging high-similarity pairs without recalculating all pairwise similarities. This approach ensures the essential structure and information of the hypergraph are preserved while significantly reducing computational complexity.

Time Complexity Analysis of Algorithm 1. Assuming the computation of hypergraph information entropy requires $O(h)$, merging operations take $O(m)$, and finding a vertex/hyperedge pair meeting the similarity threshold incurs $O(n)$, the worst-case time complexity of Algorithm 1 is $O(C(h+n+m))$.

The algorithm's complexity is dominated by the iterative process (Lines 3-12), which runs for at most $O(C)$ iterations, where C is the maximum iteration count or the stopping condition based on the entropy difference threshold ψ. Within each iteration, key operations include:

- Random sampling of vertex/hyperedge pairs ($O(n)$)
- Merging operations ($O(m)$)
- Computing hypergraph information entropy ($O(h)$)

Thus, each iteration has a time complexity of $O(h+n+m)$, and the overall complexity is $O(C(h+n+m))$.

Space Complexity Analysis of Algorithm 1 The space complexity is $O(|V|+|E|)$.

Initializing the hypergraph $G^{'}$ requires storage for $|V|$ vertices and $|E|$ hyperedges. During execution, all operations, including merging, are performed in place, requiring no additional storage. Therefore, the space complexity remains $O(|V|+|E|)$.

D. Optimal Strategies

Despite the notable improvements in efficiency and quality achieved by Algorithm 1, further optimizations are possible. We have made major advancements in three areas: algorithmic efficiency, merge result quality, and adaptability.

To improve operational efficiency, we introduce a new strategy for simultaneously merging multiple vertices or hyperedges, reducing runtime without compromising merge quality. Additionally, a more efficient sampling algorithm has been developed that leverages hypergraph distribution characteristics to prioritize high-similarity pairs, thus improving precision. For usability, we have simplified parameter settings, allowing users to input desired information entropy error and expected merge counts. This automated configuration process enhances accessibility, making the algorithm easier to use for users of all expertise levels.

1) Multi Vertices/Hyperedges Merging: Our method introduces an efficient approach for merging multiple vertices or hyperedges simultaneously, enhancing efficiency beyond traditional pairwise merging. Starting with vertex pairs that meet a similarity threshold, additional vertices or hyperedges are iteratively added until the group reaches a sampling threshold, after which they are merged in a single operation. This reduces iterations and improves runtime while preserving merge quality. Theorems 5 and 6 confirm that this approach maintains the hypergraph's entropy properties and structural integrity, making it particularly effective for large-scale hypergraphs.

2) Efficient Sampling Method: Sampling vertex and hyperedge pairs that meet a specified similarity threshold is challenging due to the complexity of hypergraph structures, often requiring multiple rounds with basic random sampling (Algorithm 1). Leveraging the power-law [37] distribution of hypergraphs, we propose a priority-based sampling algorithm that focuses on high-degree vertices and large hyperedges, significantly improving sampling efficiency. Specifically, the sampling probabilities for vertices and hyperedges are designed to prioritize those with higher degrees or larger cardinality:

$$
\begin{aligned}
p_{\text{sample}}(u) &= \frac{d(u)}{\sum_{v \in V} d(v)}, \quad u \in V \\
p_{\text{sample}}(e) &= \frac{c(e)}{\sum_{e' \in E} c(e')}, \quad e \in E
\end{aligned}
\tag{26}
$$

Experimental results demonstrate superior speed and quality compared to uniform random sampling.

3) Adaptive Algorithm: We propose an adaptive algorithm to dynamically adjust the similarity parameter θ, optimizing performance across diverse hypergraph structures without manual tuning. Inspired by adaptive learning rate techniques, the algorithm updates θ during execution using a weighted balance of historical and new similarity data. This eliminates the need for fixed parameter settings, enhancing flexibility and accessibility for users of all expertise levels.

Algorithm 2 EDHM-Adaptive

```
Input: Hypergraph $G = (V, E, W)$;
       Hypergraph information entropy change threshold $\psi$;
       The count of merging operations $C$;
Output: merged hypergraph $G^{'} = (V^{'}, E^{'}, W^{'})$;
 1: $G^{'} \leftarrow G$;
 2: $c \leftarrow 0$;
 3: $\theta \leftarrow 1$;
 4: while $\frac{H(G^{'})-H(G)}{H(G)} \leq \psi$ or $c \leq C$ do
 5:     $simS \leftarrow \emptyset$;
 6:     repeat
 7:         Randomly sample the vertex pair $(u, v)$ from $G^{'}$;
 8:         $simS \leftarrow simS \cup S(u, v)$;
 9:     until $S(u, v) > \theta$ or $|simS| > 100$
10:     if $|simS| < 100$ then
11:         Merge the vertex pair $(u, v)$ in $G^{'}$;
12:     else
13:         Select the top 10% of values in $simS$ and compute
            their average, denoted as $\theta'$.
14:         $\theta \leftarrow 0.5\theta + 0.5\theta'$;
15:     Merge hyperedge pair by a similar approach in Line
        5-13;
16:     $c \leftarrow c + 1$;
    return $G^{'}$;
```

Algorithm 2 initializes the similarity threshold θ to 1 and iteratively samples vertex and hyperedge pairs. Pairs exceeding θ are merged, while others are stored for future reference. If no pairs surpass θ after 100 iterations, sampling halts. The threshold θ is dynamically updated using a weighted formula to adapt to evolving data. Each successful merge increments the counter c, ensuring efficient and adaptive handling of vertex and hyperedge pairs throughout the process.

Time Complexity Analysis of Algorithm 2. Assuming the computation of hypergraph information entropy takes $O(h)$, merging operations require $O(m)$, and the average size of the set of similarity scores is denoted as $\overline{simS}$, the overall time complexity of Algorithm 2 is $O(C \cdot (h + m + \overline{simS}))$, where C is the upper limit on the number of iterations.

The time complexity is primarily driven by the number of iterations and the operations within each iteration. The algorithm continues until the rate of information gain falls below a threshold ψ, or until a maximum of C iterations is reached. In each iteration:

- Random sampling of vertex pairs and similarity computation is an $O(1)$ operation.
- The inner loop runs until either condition is met or a maximum of 100 attempts are reached, leading to an average complexity of $O(\overline{simS})$ for handling the similarity scores.
- Calculating the mean of the top 10% similarity scores also contributes $O(\overline{simS})$.

Thus, the overall time complexity is $O(C \cdot (h + m + \overline{simS}))$.

Space Complexity Analysis of Algorithm 2. The space complexity of Algorithm 2 is $O(|V| + |E|)$, which is the same as that of Algorithm 1.

VI. Anomalous Vertex and Hyperedge Detection

In dynamic hypergraphs, where vertices and hyperedges are constantly added or removed, tracking structural changes is essential for understanding their evolution. Traditional change detection methods, such as full traversal or repetitive sampling, are often computationally expensive and impractical for large-scale hypergraphs. To address these challenges, we propose an efficient approach based on global information entropy to detect significant structural changes.

We first establish the theoretical foundation of our method, proving its effectiveness and robustness in capturing structural changes. Subsequently, we present a detailed algorithmic design that implements this entropy-driven approach, ensuring both computational efficiency and scalability for real-world dynamic hypergraph scenarios.

A. Theoretical Analysis

We analyze the relationship between changes in hypergraph information entropy and the insertion of hyperedges in dynamic hypergraphs.

Theorem 9: When a new hyperedge is added, the change in hypergraph entropy will be smaller if it interacts more with the existing vertices of the hypergraph.

Proof: Consider adding a new hyperedge e to the hypergraph. Let this hyperedge consist of m vertices from the existing vertex set V, denoted as V_m, and n newly introduced vertices, denoted as V_n. After adding this hyperedge, the updated vertex set is V', and the updated edge set is E'.

1) Information Entropy of the Hyperedge

For the new hyperedge e with a weight $c'_G(e)$, we define its probability in the hypergraph as:

$$p'_G(e) = \frac{c'_G(e)}{\sum_{v \in V'} |N(v)|} \tag{27}$$

Here, $\sum_{v \in V'} |N(v)|$ represents the total number of hyperedges adjacent to all vertices in the hypergraph. Since the new hyperedge e contains m existing vertices and n new vertices, the total number of adjacent hyperedges for the updated vertex set V' becomes:

$$\sum_{v \in V'} |N(v)| = \sum_{v \in V} |N(v)| + m + n \tag{28}$$

Thus, the probability of the newly added hyperedge becomes:

$$p'_G(e) = \frac{c'_G(e)}{\sum_{v \in V} |N(v)| + m + n} \tag{29}$$

From this, it is evident that the probability of the new hyperedge depends on the total number $m + n$, rather than the specific ratio of m to n. The change in the hyperedge probability is thus not directly dependent on the balance between m and n.

2) Information Entropy of the Vertex

Next, we consider the impact of the new hyperedge on the degrees and probabilities of the vertices. The probability of a vertex v in the hypergraph is defined as:

$$p_G(v) = \frac{d'_G(v)}{\sum_{e \in E'} |e|} \tag{30}$$

Here, $d'_G(v)$ is the degree of vertex v, and $\sum_{e \in E'} |e|$ is the total number of vertices participating in all hyperedges in the hypergraph. When a new hyperedge is inserted, if a vertex is not part of the newly added hyperedge, its degree remains unchanged, meaning its probability also remains unchanged. Therefore, we only need to consider the probability changes for the vertices that are part of the new hyperedge. We need to evaluate how the insertion of the new hyperedge affects the entropy of the hypergraph. The entropy of the vertex set, denoted as $H_V(G)$, is given by:

$$H_V(G) = -\sum_{v \in V} p_G(v) \cdot \ln p_G(v) \tag{31}$$

After adding the new hyperedge, the degree of each vertex $v \in V_m$ in the hyperedge increases by 1, i.e., $d_G(v) \to d_G(v) + 1$. Hence, the probability of these vertices changes, and the entropy change can be represented as:

$$\begin{aligned} \Delta H_V(G) = &-\sum_{v \in V_m} w(v) \cdot \frac{d_G(v)+1}{\sum_{e \in E'} |e|} \cdot \ln \frac{d_G(v)+1}{\sum_{e \in E'} |e|} \\ &+ \sum_{v \in V_m} w(v) \cdot \frac{d_G(v)}{\sum_{e \in E'} |e|} \cdot \ln \frac{d_G(v)}{\sum_{e \in E'} |e|} \\ &- \sum_{v \in V_n} w(v) \cdot \frac{1}{\sum_{e \in E'} |e|} \cdot \ln \frac{1}{\sum_{e \in E'} |e|} \end{aligned} \tag{32}$$

With $m + n$ as a constant, as n increases, the increment of $\Delta H_V(G)$ also grows.

Based on the above analysis, we observe that as the new hyperedge includes more non-existing vertices (i.e., as n increases and m decreases), the change in entropy is primarily influenced by the increase in the degrees of existing vertices.

Algorithm 3 Anomalous Vertex/Hyperedge Detection

```
Input: Hypergraph $G = (V, E)$, threshold $\Delta H_{th}$ for significant entropy change
       New vertex/hyperedge added or removed
Output: Abnormal change indicators
Initialize $H_{entropy} \leftarrow$ CalculateEntropy$(G)$
while hypergraph $H$ is updated do
    Input: New vertex/hyperedge added or removed
    Recalculate entropy $H_{new} \leftarrow$ CalculateEntropy$(H)$
    Compute entropy change $\Delta H = |H_{new} - H_{entropy}|$
    if $\Delta H > \Delta H_{th}$ then
        Mark as an abnormal change
        Update global entropy $H_{entropy} \leftarrow H_{new}$
    else
        Update global entropy $H_{entropy} \leftarrow H_{new}$
return Abnormal change
```

Specifically, when we add an existing vertex to a hyperedge, the degree increment of that vertex causes a smaller change in the overall uncertainty (entropy) compared to introducing a completely new vertex, which adds significant uncertainty. Therefore, we conclude that the more a new hyperedge interacts with the existing vertices, the smaller the entropy change. Conversely, the more non-existing vertices a new hyperedge includes, the greater the increase in entropy. ■

Similarly, we present the Theorem when a hyperedge is deleted.

Theorem 10: When a new hyperedge is added, the change in entropy will be smaller if it interacts more with the existing vertices of the hypergraph.

B. Anomalous Detection Algorithm

Our approach recalculates the global information entropy of the hypergraph whenever updates occur, capturing structural fluctuations caused by additions or deletions of vertices and hyperedges. By tracking the overall entropy change, we can monitor shifts in the hypergraph's complexity and connectivity, identifying significant structural changes over time.

The following algorithm describes our approach for detecting changes in a dynamic hypergraph by recalculating the global entropy and assessing structural shifts based on a predefined threshold.

The algorithm begins by calculating the initial global entropy of the hypergraph G, which serves as a baseline for detecting future changes. Whenever the hypergraph is updated with a new vertex or hyperedge addition or removal, the global entropy is recalculated to reflect any overall structural change, capturing shifts in complexity and connectivity. The entropy change ΔH is then computed as the absolute difference between the new entropy H_{new} and the previous entropy $H_{entropy}$. If ΔH exceeds a predefined threshold ΔH_{th}, the change is marked as abnormal, indicating a significant structural shift. Regardless of whether the change is abnormal, the baseline global entropy is updated to H_{new} to ensure that subsequent changes are assessed relative to the current structural state. Finally, the algorithm returns indicators of any abnormal changes, providing a continuous measure of significant shifts in the hypergraph's evolution.

TABLE I
REAL-WORLD HYPERGRAPH DATASETS

Dataset	$\|V\|$	$\|E\|$	d_{max}	d_{avg}	c_{max}	c_{avg}
NDCS	3767	29810	5901	38	187	4
MaAn	73851	5446	173	1	1784	24
WaTr	88860	69906	5733	5	25	6
ThAU	90054	117764	2247	3	14	2
ThMS	153806	563710	12403	9	21	2
CoMH	503868	308934	1077	1	925	2
CoGe	1091979	1045462	1125	3	284	3
CoDB	1836596	2955129	1399	5	280	3

Theorems 9 and 10 demonstrate that the degree of change in global information entropy varies with the type of hyperedge inserted or deleted. Hyperedges with distinct connectivity patterns and influence contribute differently to the entropy, allowing more precise detection and quantification of structural changes by analyzing their specific impacts.

VII. EXPERIMENT

In this section, we conduct a detailed experimental analysis of eight real-world hypergraphs to evaluate the performance and effectiveness of our proposed hypergraph merging algorithm. First, we provide an overview of the datasets, experimental setup, algorithms used, and evaluation metrics. Then, we perform extensive experiments from two perspectives: hypergraph merging and anomalous vertex/hyperedge detection, demonstrating the advantages of our approach.

A. Datasets and Settings

Datasets. Our algorithms are assessed across 8 unique datasets obtained from ARB [1].

The NDCS hypergraphs [38] from the pharmaceutical domain represent substances as vertices and drug compositions as hyperedges. The MaAn dataset [39] models Math Overflow tags as vertices and groups of answered questions as hyperedges. WaTr [40] represents Walmart products as vertices and co-purchased items as hyperedges. ThAU and ThMS [38] are temporal networks with vertices as users on askubuntu.com and math.stackexchange.com, and hyperedges as discussions within 24-hour periods. CoMH and CoGe [38], [41] are temporal academic graphs with vertices as authors and hyperedges as contributions in ”History” and ”Geology.” The CoDB dataset [38] features vertices as authors and hyperedges as their DBLP publications.

In the preprocessing phase, isolated vertices that do not affect hypergraph connectivity are removed to improve data coherence for analysis. Table I summarizes the datasets' key statistics, including d_{max} and d_{avg} (maximum and average vertex degrees) and c_{max} and c_{avg} (maximum and average hyperedge cardinalities).

[1] https://www.cs.cornell.edu/~arb/data/

TABLE II
METRICS USED IN THE EXPERIMENTAL ANALYSIS

Metric	Description
Degree Distribution [25]	The distribution of the number of hyperedges connected to each vertex in the hypergraph.
Cardinality Distribution [25]	The distribution of the number of vertices within each hyperedge in the hypergraph.
Degree Pair Distribution [25]	The distribution of the number of hyperedges shared between pairs of vertices in the hypergraph.
Intersection Distribution [25]	The distribution of the number of vertices shared between pairs of hyperedges in the hypergraph.
KCore Distribution [25]	The distribution of the number of vertices in various k-cores of the hypergraph.
PageRank Distribution [25]	The distribution of PageRank values across the vertices in the hypergraph.
Closeness [25]	The distribution of closeness centrality among the vertices in the hypergraph.
LCC Distribution [25]	The distribution of Local Clustering Coefficient (LCC) values across all vertices in the hypergraph, where $LCC(v) = \frac{\lvert e_{jk}: j,k \in N_v, e_{jk} \in E \rvert}{k_v(k_v-1)}$, N_v is the neighbor set of vertex v, and k_v is its size.
KL [42]	The Kullback-Leibler divergence between the attribute distributions of the merged and original hypergraphs.
Ratio	The information entropy of the merged hypergraph G' relative to the original hypergraph G, i.e., $ratio = \frac{H(G')}{H(G)}$.
Compression Ratio	The ratio of the size of the merged hypergraph to the original hypergraph, i.e., $CR = \frac{\lvert E(G') \rvert + \lvert V(G') \rvert}{\lvert E(G) \rvert + \lvert V(G) \rvert}$.
Overlapping	The average number of hyperedges per vertex in the sub-hypergraph, defined as $Overlapping(H) = \frac{\sum_{e \in E_H} \lvert e \rvert}{\lvert V_H \rvert}$.

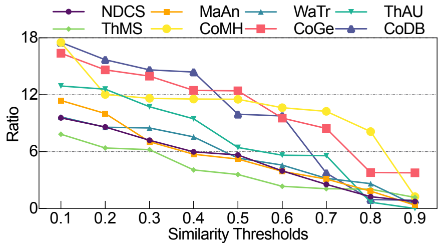


Fig. 1. Exp-1: Impact of similarity threshold on hypergraph information entropy.

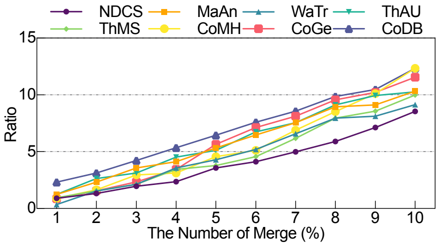


Fig. 2. Exp-2: Impact of merge number on hypergraph information entropy.

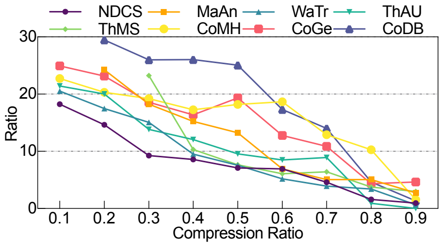


Fig. 3. Exp-3: Impact of compression ratio on hypergraph information entropy.

Settings. All algorithms are implemented in C++ and compiled with GNU GCC version 11.4.0. Experiments are conducted on a server with an Intel Xeon Platinum 8358P CPU (2.60 GHz), 512 GB memory, and Ubuntu 22.04.5 LTS.

Algorithms. We list all the methods used in the experiment along with their detailed configurations as follows:

- *Hamming Distance [15]:* Measures the number of differing elements.
- *Hypergraph Edit Distance [16]:* Represents the minimum number of edits needed to transform one hypergraph into another.
- *EDHM-Sample:* Entropy-Driven hypergraph merging algorithm by sampling in Section V-C.
- *EDHM-Multi:* Entropy-Driven hypergraph merging algorithm by sampling and multi vertices/hyperedges merge in Section V-D1.
- *EDHM-Efficient Sample:* Entropy-Driven hypergraph merging algorithm by efficient sampling in Section V-D2.
- *EDHM-Adaptive:* Entropy-Driven hypergraph merging algorithm by adaptive approach in Section V-D3.

Evaluation Metrics and Concepts. We outline all the metrics and concepts utilized for evaluation in our experiments in Table II.

B. Hypergraph Merging

The experiments begin by demonstrating the relationships between the proposed hypergraph entropy, similarity threshold, number of merged, and compression ratio. Following this, we examine the superiority of our proposed similarity measurement metrics to the traditional Jaccard similarity. We then analyze the consistency between the merged hypergraph and the original hypergraph across various fundamental attributes, such as degree distribution, cardinality distribution, degree pair distribution, and intersection distribution, as well as application-related attributes like k-core, PageRank, LCC, and closeness [25]. Finally, we compare the performance and scalability of our algorithm against different optimization techniques.

Exp-1: Impact of Similarity Threshold on Hypergraph Information Entropy. We evaluated EDHM-Sample across different datasets under varying similarity thresholds, comparing the ratio of the merged hypergraph's information entropy to the original after the same merging iterations. As shown in Fig. 1, higher thresholds generally increase the ratio, merging more similar hyperedges while preserving hypergraph performance. Lower thresholds may overmerge, affecting results. Notably, a critical balance between merging efficiency and information retention occurs around the 0.5 threshold, marked by significant changes in the ratio for several datasets.

Exp-2: Impact of Merge Number on Hypergraph Information Entropy. In this experiment, we evaluated the performance of different datasets under varying merge counts using the EDHM-Sample algorithm with a similarity threshold

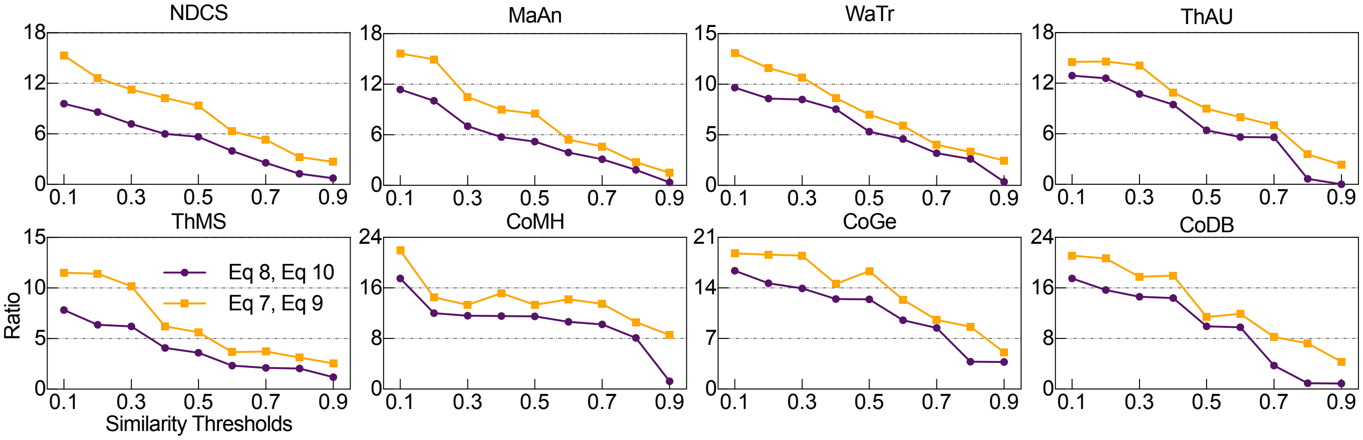
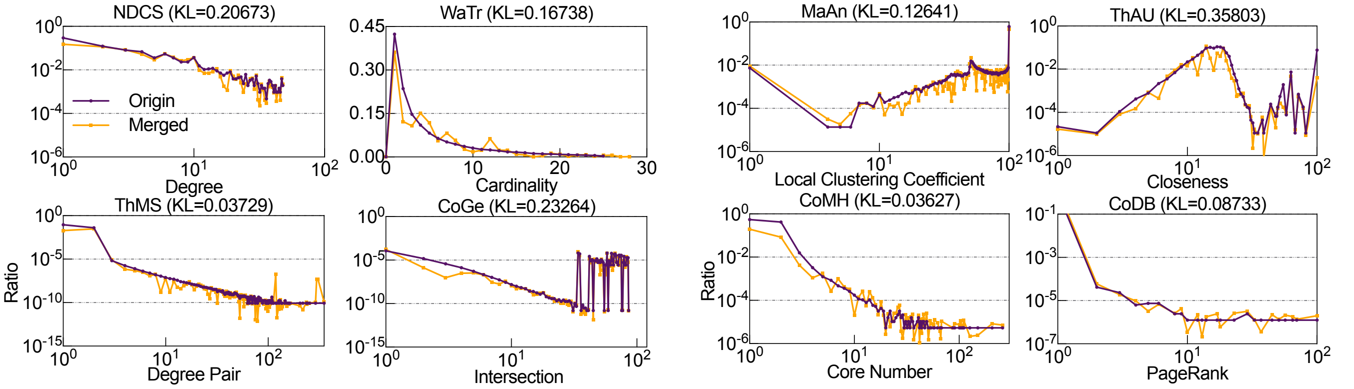


Fig. 4. Exp-4: Comparison of Proposed Similarity Measure and Traditional Jaccard Similarity.



(a) Fundamental Properties

(b) Application Properties

Fig. 5. Exp-5: Comparison of fundamental and application properties before and after hypergraph merging.

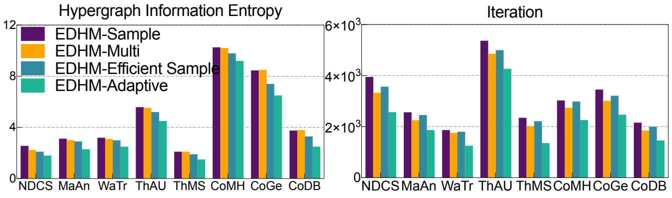


Fig. 6. Exp-6: Ablation Study on EDHM Algorithms.

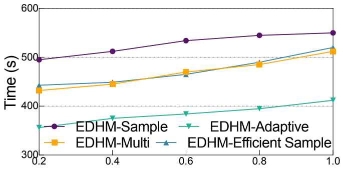


Fig. 7. Exp-7: Scalability Analysis of EDHM Algorithms.

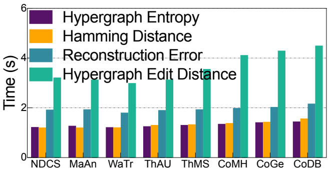


Fig. 8. Exp-8: Comparison Performance of different metrics on anomalous vertex/hyperedge detection.

of 0.7. Hyperedges were merged in increments of 1% to 10% of the total number of hyperedges, and the ratio of the merged hypergraph's information entropy to the original was tracked after each step. As shown in Fig. 2, the ratio generally increases across all datasets as the number of merges grows. Initially, up to 5-6% merges, the increase is slight, indicating effective redundancy reduction with minimal information loss. Beyond this point, some datasets exhibit a sharper rise in the ratio, signaling excessive information loss and reduced merging benefits. This pattern implies that there is an optimal range of merging operations that maximizes merging efficiency while maintaining a balance with information retention. Beyond this optimal point, additional merges may start to degrade the quality of the merged hypergraph.

Exp-3: Impact of Compression Ratio on Hypergraph Information Entropy. This experiment examined the relationship between compression ratio and information entropy using the EDHM-Sample algorithm with a similarity threshold of 0.7. Hyperedges were progressively merged, and the information entropy ratio was recorded at various compression ratios. As shown in Fig. 3, certain datasets, such as MaAn, ThMS, and CoDB, could not achieve lower compression ratios, e.g., 0.1 or 0.2, under the given threshold. The results reveal that at higher compression ratios, the information entropy ratio remains low, indicating minimal information loss due

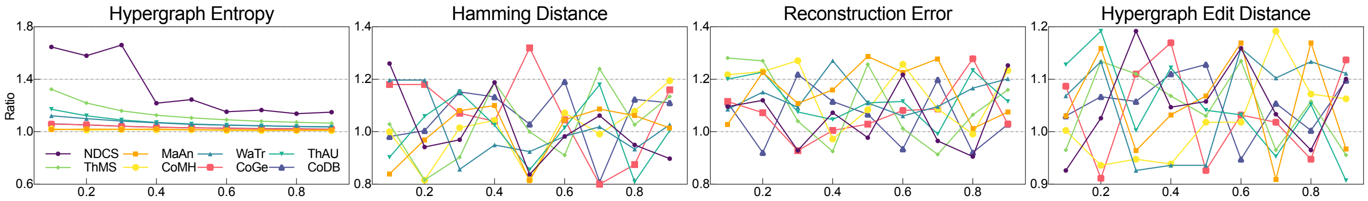


Fig. 9. Exp-9: Comparison Effectuation of different metrics on anomalous vertex/hyperedge detection.

to the removal of redundant edges and nodes during initial merging stages. However, as the compression ratio decreases, the entropy ratio increases significantly, suggesting that further merging introduces excessive information loss. This highlights the trade-off between compression and information retention.

Exp-4: Comparison of Proposed Similarity Measure and Traditional Jaccard Similarity. In this experiment, we compared the traditional Jaccard similarity with our proposed similarity measure. At the same compression ratio and after the same number of merging operations, we calculated the degree of entropy change. The results, as depicted in Fig. 4, illustrate the differences between the two similarity measures. For each dataset, we examined how the entropy ratio changes as a function of the similarity threshold. The graphs show that the proposed similarity measure (represented by Eq. 8 and Eq. 10) generally results in a lower entropy ratio compared to the traditional Jaccard similarity (represented by Eq. 7 and Eq. 9) at similar thresholds. This suggests that our similarity measure is more effective in preserving the original information of the hypergraph during merging. In summary, the experiment demonstrates that, under the same conditions, our proposed similarity measure results in less information loss, making it a more advantageous choice for hypergraph merging tasks.

Exp-5: Comparison of Fundamental and Application Properties Before and After Hypergraph Merging. In this experiment, we compared the distributions of fundamental properties and application-related properties metrics of hypergraphs before and after merging across the same datasets.

- *Fundamental Properties:* We analyzed degree distribution, cardinality distribution, degree pair distribution, and intersection distribution (Fig. 5(a)). The results show that these distributions remain highly consistent before and after merging, with low KL divergence values across datasets, indicating that the merging process preserves the essential structural characteristics of the hypergraph.
- *Application Properties:* We evaluated application-related metrics, including k-core, PageRank, Local Clustering Coefficient (LCC), and closeness centrality (Fig. 5(b)). These metrics also exhibit strong similarity between the original and merged hypergraphs. The low KL divergence values confirm that the merging process has minimal impact on functionality, ensuring that the merged hypergraph retains high usability.

Overall, the experiments demonstrate that the hypergraphs maintain their fundamental and application-related properties after merging, with only minor variations. This indicates that the merging technique effectively reduces the size of the hypergraph without significantly altering its structural or functional characteristics.

Exp-6: Ablation Study on EDHM Algorithms. This experiment conducts an ablation study to analyze the contributions of different components in the EDHM algorithms, including EDMH-Sample, EDMH-Multi, EDMH-Efficient Sample, and EDMH-Adaptive, across multiple datasets. As shown in Fig. 6, EDMH-Efficient Sample and EDMH-Adaptive achieve lower information entropy compared to EDMH-Sample and EDMH-Multi, highlighting their superior ability to preserve information during hypergraph merging. Furthermore, the fewer iterations required by EDMH-Efficient Sample and EDMH-Adaptive indicate their enhanced computational efficiency. By contrast, EDMH-Sample and EDMH-Multi exhibit higher entropy and longer iteration times, revealing increased information loss and reduced efficiency. These results demonstrate that each optimization in EDMH-Efficient Sample and EDMH-Adaptive contributes significantly to improving both effectiveness and efficiency, making them the most robust configurations for hypergraph merging.

Exp-7: Scalability Analysis of EDHM Algorithms. This experiment evaluated the scalability of EDHM algorithms using the CoMH dataset, scaled down to 20%, 40%, 60%, and 80% of its hyperedges. Processing times for each algorithm were compared across these dataset sizes. As shown in Fig. 7, processing times increase with dataset size, as expected. EDMH-Efficient Sample and EDMH-Adaptive consistently exhibit better scalability, with lower and more gradual increases in processing time compared to EDMH-Sample and EDMH-Multi. In contrast, EDMH-Multi shows the steepest increase in processing time, indicating poor scalability. Overall, EDMH-Adaptive and EDMH-Efficient Sample are more efficient and suitable for large-scale hypergraph merging.

C. Anomalous Vertex/Hhyperedge Detection

In this experiment, we analyze changes in four metrics—hypergraph entropy, Hamming distance, reconstruction error, and hypergraph edit distance—as hyperedges are incrementally added. Starting with the base hypergraph structure, we gradually increase the number of hyperedges from 10% to 90%, applying two insertion schemes at each step. Scheme 1 inserts 10% unrelated hyperedges, artificially constructed without association to the original structure. Scheme 2 inserts 10% related hyperedges, connected to the existing structure and following its distribution pattern. For each insertion, we calculate the total time and compare the four metrics between the two schemes to evaluate their impact on the hypergraph structure.

Exp-8: Dynamic Hyperedge Insertion Performance Analysis. The results in Fig. 8 reveal that hypergraph entropy consistently demonstrates the fastest computation time across all datasets, making it highly efficient and well-suited for real-time dynamic hypergraph analysis. Hamming distance shows moderate computation times, performing slightly slower than entropy but still manageable for most datasets, though its scalability becomes a concern for very large hypergraphs. Reconstruction error requires significantly more time, particularly for larger datasets, due to its reliance on computationally expensive iterative calculations. Hypergraph edit distance, however, is consistently the slowest metric, with substantial computational overhead arising from exhaustive traversal and edit operations. These findings underscore hypergraph entropy as the efficient and practical metric for analyzing dynamic hypergraphs, particularly in time-sensitive applications.

Exp-9: Dynamic Hyperedge Insertion Effectuation Analysis. Fig. 9 shows the ratio change for hypergraph entropy. It can be observed that inserting related hyperedges causes minimal entropy change, whereas inserting unrelated hyperedges significantly increases entropy, resulting in a ratio consistently greater than 1. This suggests that entropy can reliably detect the insertion of unrelated hyperedges and effectively distinguish between the two types of hyperedges. However, Fig. 9 display the ratio changes for Hamming distance and reconstruction error, with both ratios fluctuating around 1. This indicates that these two metrics are unable to consistently distinguish between related and unrelated hyperedges, making them less effective for detection. Overall, entropy demonstrates high stability and discriminative power for detecting unrelated hyperedges, whereas Hamming distance, reconstruction error, and edit distance are less effective in this regard.

VIII. CONCLUSION

This paper introduces a novel information entropy-based metric for measuring *Hypergraph Evolution*, offering an efficient and precise solution for analyzing dynamic structural changes in hypergraphs. By leveraging probability distributions of vertices and hyperedges, the proposed method effectively captures their significance and connectivity, enabling accurate and scalable hypergraph analysis. The method is applied to two tasks: hypergraph merging and anomalous vertex/hyperedge detection. For hypergraph merging, we present an entropy-driven approach that aggregates similar vertices and hyperedges, significantly reducing storage requirements while preserving core structural and functional properties. For anomalous vertex/hyperedge detection, we establish a framework to identify structural changes by analyzing entropy variations during hyperedge insertions, enabling efficient detection of anomalies and dynamic variations. Extensive experimental results validate the efficiency and precision of the hypergraph information entropy metric, demonstrating its ability to effectively preserve the original properties of hypergraphs while achieving superior scalability, making it suitable for processing and analyzing large-scale hypergraphs.

References

[1] A. Bretto, "Hypergraph theory: An introduction," *An introduction. Mathematical Engineering. Cham: Springer*, 2013.

[2] E. Estrada and J. A. Rodriguez-Velazquez, "Complex networks as hypergraphs," *arXiv preprint physics/0505137*, 2005.

[3] D. Yang, B. Qu, J. Yang, and P. Cudré-Mauroux, "Revisiting user mobility and social relationships in lbsns: A hypergraph embedding approach," *The World Wide Web Conference*, 2019.

[4] W. Yang, G. Wang, M. Z. A. Bhuiyan, and K.-K. R. Choo, "Hypergraph partitioning for social networks based on information entropy modularity," *Journal of Network and Computer Applications*, vol. 86, pp. 59–71, 2017.

[5] D. Li, Z. Xu, S. Li, and X. Sun, "Link prediction in social networks based on hypergraph," in *Proceedings of the 22nd international conference on world wide web*, 2013, pp. 41–42.

[6] H. Fan, F. Zhang, Y. Wei, Z. Li, C. Zou, Y. Gao, and Q. Dai, "Heterogeneous hypergraph variational autoencoder for link prediction," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 8, pp. 4125–4138, 2021.

[7] W. Li, B. Xiang, F. Yang, Y. Rong, Y. Yin, J. Yao, and H. Zhang, "scmhnn: a novel hypergraph neural network for integrative analysis of single-cell epigenomic, transcriptomic and proteomic data." *Briefings in bioinformatics*, vol. 24 6, 2023.

[8] Q. Luo, D. Yu, Z. Cai, Y. Zheng, X. Cheng, and X. Lin, "Core maintenance for hypergraph streams," *World Wide Web*, vol. 26, no. 5, pp. 3709–3733, 2023.

[9] Q. Luo, D. Yu, Y. Liu, Y. Zheng, X. Cheng, and X. Lin, "Finer-grained engagement in hypergraphs," in *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 2023, pp. 423–435.

[10] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *ACM transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, pp. 2–es, 2007.

[11] W. Aiello, F. Chung, and L. Lu, "Random evolution in massive graphs," *Handbook of massive data sets*, pp. 97–122, 2002.

[12] S. S. Srinivas, R. K. Sarkar, and V. Runkana, "Hypergraph learning based recommender system for anomaly detection, control and optimization," in *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 2022, pp. 1922–1929.

[13] A. Huang, Z. Fang, Z. Wu, Y. Tan, P. Han, S. Wang, and L. Zhang, "Multi-view heterogeneous graph learning with compressed hypergraph neural networks," *Neural Networks*, vol. 179, p. 106562, 2024.

[14] Z. Zhang, H. Lin, Y. Gao, and K. BNRist, "Dynamic hypergraph structure learning." in *IJCAI*, 2018, pp. 3162–3169.

[15] A. Pinheiro, H. P. Pinheiro, and P. K. Sen, "The use of hamming distance in bioinformatics," in *Handbook of Statistics*. Elsevier, 2012, vol. 28, pp. 129–162.

[16] H. Qin, R. Li, Y. Yuan, G. Wang, and Y. Dai, "Explainable hyperlink prediction: A hypergraph edit distance-based approach," *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pp. 245–257, 2023.

[17] Y. Bai, H. Ding, S. Bian, T. Chen, Y. Sun, and W. Wang, "Simgnn: A neural network approach to fast graph similarity computation," in *Proceedings of the twelfth ACM international conference on web search and data mining*, 2019, pp. 384–392.

[18] D. Koutra, J. T. Vogelstein, and C. Faloutsos, "Deltacon: A principled massive-graph similarity function," in *Proceedings of the 2013 SIAM international conference on data mining*. SIAM, 2013, pp. 162–170.

[19] M. Qiao, H. Zhang, and H. Cheng, "Subgraph matching: on compression and computation," *Proceedings of the VLDB Endowment*, vol. 11, no. 2, pp. 176–188, 2017.

[20] L. Zhang, Z. Zhang, G. Wang, Y. Yuan, S. Zhao, and J. Xu, "Hyperiso: Efficiently searching subgraph containment in hypergraphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 8, pp. 8112–8125, 2022.

[21] K. LeFevre and E. Terzi, "Grass: Graph structure summarization," in *SDM*, 2010.

[22] I. Bloch and A. Bretto, "A new entropy for hypergraphs," in *Discrete Geometry for Computer Imagery: 21st IAPR International Conference, DGCI 2019, Marne-la-Vallée, France, March 26–28, 2019, Proceedings 21*. Springer, 2019, pp. 143–154.

[23] F. Hu, K. Tian, and Z.-K. Zhang, "Identifying vital nodes in hypergraphs based on von neumann entropy," *Entropy*, vol. 25, no. 9, p. 1263, 2023.

[24] C. Chen and I. Rajapakse, "Tensor entropy for uniform hypergraphs," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2889–2900, 2020.

[25] Y. Chen, H. Ye, S. Vedula, A. Bronstein, R. Dreslinski, T. Mudge, and N. Talati, "Demystifying graph sparsification algorithms in graph properties preservation," *Proceedings of the VLDB Endowment*, vol. 17, no. 3, pp. 427–440, 2023.

[26] X. Gou, L. Zou, C. Zhao, and T. Yang, "Graph stream sketch: Summarizing graph streams with high speed and accuracy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 6, pp. 5901–5914, 2023.

[27] A. Borici and A. Thomo, "Semantic graph compression with hypergraphs," in *2014 IEEE 28th International Conference on Advanced Information Networking and Applications*. IEEE, 2014, pp. 1097–1104.

[28] M. Riondato, D. García-Soriano, and F. Bonchi, "Graph summarization with quality guarantees," *Data mining and knowledge discovery*, vol. 31, pp. 314–349, 2017.

[29] H. Matsumoto, T. Yoshida, R. Kondo, and R. Hisano, "Hypergraph change point detection using adapted cardinality-based gadgets: Applications in dynamic legal structures," *arXiv preprint arXiv:2409.08106*, 2024.

[30] B. Yan, C. Yang, C. Shi, J. Liu, and X. Wang, "Abnormal event detection via hypergraph contrastive learning," in *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*. SIAM, 2023, pp. 712–720.

[31] A. Surana, C. Chen, and I. Rajapakse, "Hypergraph dissimilarity measures," *arXiv preprint arXiv:2106.08206*, 2021.

[32] R. M. Gray, *Entropy and information theory*. Springer Science & Business Media, 2011.

[33] C. Tsallis, "Entropy," *Thermodynamic Weirdness*, 2022.

[34] S. M. Pincus, "Approximate entropy as a measure of system complexity." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 88, pp. 2297 – 2301, 1991.

[35] A. H. Murphy, "The finley affair: A signal event in the history of forecast verification," *Weather and Forecasting*, vol. 11, pp. 3–20, 1996.

[36] A. Björklund and T. Husfeldt, "Inclusion–exclusion algorithms for counting set partitions," *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pp. 575–582, 2006.

[37] M. Faloutsos, P. Faloutsos, and C. Faloutsos, "On power-law relationships of the internet topology," *ACM SIGCOMM computer communication review*, vol. 29, no. 4, pp. 251–262, 1999.

[38] A. R. Benson, R. Abebe, M. T. Schaub, A. Jadbabaie, and J. Kleinberg, "Simplicial closure and higher-order link prediction," *Proceedings of the National Academy of Sciences*, 2018.

[39] N. Veldt, A. R. Benson, and J. Kleinberg, "Minimizing localized ratio cut objectives in hypergraphs," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, 2020.

[40] I. Amburg, N. Veldt, and A. R. Benson, "Clustering in graphs and hypergraphs with categorical edge labels," in *Proceedings of the Web Conference*, 2020.

[41] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-J. P. Hsu, and K. Wang, "An overview of microsoft academic service (MAS) and applications," in *Proceedings of the 24th International Conference on World Wide Web*. ACM Press, 2015.

[42] T. Van Erven and P. Harremos, "Rényi divergence and kullback-leibler divergence," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.