Journal of Data Mining and Knowledge Engineering
Volume 2, Issue 1, January-June 2017

Hypergraph and Higher-Order Data Mining: Concepts, Models and Applications

Awdhesh Pandey[1], Gaurav Triwedi[2], Buwaneshwer Patil[3], Rahul Sakya[4]
Associate Professor, Assistant Professor
Department of Information Systems
Gyan Jyoti College, Uttar Pradesh, India
Email:Awdheshpandey85@gmail.com[1], gauravtriwedi779@yahoo.com[2],
patil_1bhuwaneshwer@rediffmail.com[3]

Abstract

Traditional data mining techniques mostly rely on pairwise relationships represented by graphs or relational tables. However, many real-world systems naturally contain interactions among more than two entities simultaneously, such as co-authorship networks, biological pathways, and group communications. These complex multi-entity relations can be better modeled using hypergraphs and higher-order structures. Hypergraph and higher-order data mining has emerged as an important research direction to capture group-level dependencies, collective patterns, and multi-way associations that cannot be discovered using conventional graph-based methods.

This paper presents a comprehensive review of hypergraph and higher-order data mining. We first introduce the theoretical foundations of hypergraphs and their properties. Then we discuss hypergraph representation learning, clustering, classification, and community detection methods. We also examine higher-order network mining including simplicial complexes and tensor-based approaches. Applications in social networks, recommender systems, bioinformatics, and knowledge graphs are analyzed. A comparative analysis of algorithms and challenges is also presented. Finally, open research directions such as scalability, interpretability, and dynamic hypergraphs are discussed. The study shows that hypergraph-based mining provides richer structural *understanding compared to traditional graph mining and is essential for modeling modern complex data systems.*

Keywords: *Hypergraph mining, higher-order networks, multi-relational data, hypergraph clustering, tensor mining, group interactions*

INTRODUCTION

Data mining and knowledge discovery traditionally assume pairwise relations among entities. Graphs and relational databases represent such binary relationships effectively. However, many datasets contain interactions involving multiple entities at same time. For example, a research paper has multiple authors, a biochemical reaction involves several molecules, and a group chat includes many participants simultaneously. Modeling such relationships using simple edges results in loss of structural information.

Hypergraphs provide a natural extension of graphs where a hyperedge can connect any number of vertices. This allows representing group-level relations directly instead of decomposing them into pairwise links. Recently, higher-order data mining techniques have been developed to analyze these complex relations. They include hypergraph clustering, higher-order centrality, tensor decomposition, and simplicial network analysis.

The motivation for hypergraph mining arises from limitations of traditional graph mining:

- Pairwise reduction loses group semantics
- Clique expansion causes information distortion
- Higher-order dependencies remain hidden
- Complex relational datasets cannot be modeled properly

Therefore, hypergraph and higher-order data mining is becoming essential in modern analytics domains such as social computing, systems biology, and recommendation systems.

This paper provides a structured survey of methods and applications in hypergraph and higher-order data mining.

FUNDAMENTALS OF HYPERGRAPHS

Hypergraphs generalize classical graphs by allowing relationships among more than two entities. In many real datasets, interactions are not simply pairwise but involve groups. Hypergraph theory provides mathematical tools to represent and analyze such multi-way relations. Understanding the basic structure and types of hypergraphs is essential before applying higher-order data mining algorithms.

1. Definition of Hypergraph

A hypergraph is formally defined as an ordered pair:

H=(V,E)H = (V, E)H=(V,E)

Where:

- **V** is a finite set of vertices (or nodes)
- **E** is a set of hyperedges
- Each hyperedge e∈Ee \in Ee∈E is a non-empty subset of V

Thus, a hyperedge can connect any number of vertices, unlike a standard graph edge that connects exactly two vertices. If a hyperedge contains *k* vertices, it is called a *k-ary relation* or *k-uniform relation* in relational modeling.

In simple terms, a hypergraph represents relationships among groups rather than pairs.

Difference from Graph

In a traditional graph:

G=(V,E),E⊆V×VG = (V, E), \quad E \subseteq V \times VG=(V,E),E⊆V×V

Each edge is an ordered or unordered pair of vertices.

But in hypergraph:

E⊆2VE \subseteq 2^VE⊆2V

That means each hyperedge is a subset of vertices, so it can include 2, 3, or many vertices.

Example: Co-authorship Network

Consider a research paper written by three authors:

e={Author1,Author2,Author3}e = \{Author1, Author2,
Author3\}e={Author1,Author2,Author3}

This single hyperedge represents a joint collaboration among all three authors simultaneously.

If converted into a normal graph, it would produce three pairwise edges:

- (Author1, Author2)
- (Author1, Author3)

- (Author2, Author3)

This conversion loses the fact that collaboration occurred as one group. Hypergraph keeps the original group semantics intact.

Example: Transaction Data

In market basket analysis:

Transaction T1 = {Milk, Bread, Eggs}

This can be modeled as a hyperedge connecting all three items. Hypergraph mining can discover frequent itemsets directly without pairwise reduction.

Degree Concepts in Hypergraph

Two degree notions exist:

Vertex degree

Number of hyperedges containing a vertex

deg(v)=|{e∈E:v∈e}|deg(v) = |\{e \in E : v \in e\}|deg(v)=|{e∈E:v∈e}|

Hyperedge size (cardinality)

Number of vertices in hyperedge

|e||e||e|

These metrics help analyze node participation in group interactions.

Uniform Hypergraph

If all hyperedges contain same number of vertices *k*, the hypergraph is called **k-uniform**.

Example:

Every hyperedge has exactly 3 nodes → 3-uniform hypergraph

Uniform hypergraphs are mathematically convenient and often used in theoretical studies.

Incidence Structure

Hypergraphs can be described by an incidence relation:

I⊆V×EI \subseteq V \times EI⊆V×E

If (v,e) ∈ I, vertex v belongs to hyperedge e.

This relation forms basis for incidence matrix representation used in algorithms.

TYPES OF HYPERGRAPHS

Hypergraphs appear in different forms depending on whether hyperedges have direction, weight, size constraints, or attributes. These variations allow modeling diverse real-world systems.

1. Uniform Hypergraph

A hypergraph is called **k-uniform** if every hyperedge connects exactly k vertices.

∀e∈E, ∣e∣=k\forall e \in E, \ |e| = k∀e∈E, ∣e∣=k

Example:

3-uniform hypergraph representing triangular interactions in social groups.

Applications:

- Simplicial complexes
- Motif analysis
- Higher-order network theory

Uniform hypergraphs simplify algorithm design because hyperedge size is fixed.

2. Non-Uniform Hypergraph

If hyperedges have varying sizes, the hypergraph is non-uniform.

Example:

Research collaborations:

- e1 = {A,B}
- e2 = {A,B,C}
- e3 = {A,B,C,D}

Most real datasets produce non-uniform hypergraphs because group sizes vary.

3. Weighted Hypergraph

In weighted hypergraph, each hyperedge has an associated weight:

w:E→R+w : E \rightarrow \mathbb{R}^+w:E→R+

Weight may represent:

- Frequency of interaction
- Strength of relation
- Probability
- Cost

Example:

Shopping basket hyperedge weight = number of transactions containing that itemset.

Weighted hypergraphs are important in recommender systems and network analysis.

4. Directed Hypergraph

In directed hypergraph, hyperedges have input and output vertex sets:

e=(Te,He)e = (T_e, H_e)e=(Te,He)

Where:

- TeT_eTe = tail (source vertices)
- HeH_eHe = head (target vertices)

This models many-to-many directional relations.

Example:

Chemical reaction:

Reactants → Products

$e = (\{H_2, O_2\}, \{H_2O\})$

Directed hypergraphs are widely used in:

- Metabolic networks
- Data flow systems
- Knowledge representation

Table 1.

Type	Description	Example
Uniform Hypergraph	All hyperedges have same size	3-uniform collaboration
Weighted Hypergraph	Hyperedges have weights	Co-purchase frequency
Directed Hypergraph	Ordered relation among nodes	Chemical reactions
Attributed Hypergraph	Nodes/edges have features	Social groups

HYPERGRAPH REPRESENTATIONS

Hypergraphs can be represented in several mathematical and computational forms depending on the analysis task and algorithmic requirements. Unlike simple graphs, hypergraphs do not have a single universal adjacency structure because hyperedges may connect more than two vertices. Therefore, different representations are used to convert hypergraph relations into matrix, graph, or tensor forms that can be processed by data mining algorithms.

The most commonly used hypergraph representations include incidence matrices, clique expansion graphs, star expansion graphs, and tensor representations. Each representation preserves different structural properties and introduces its own computational trade-offs.

1. Incidence Matrix Representation

The most fundamental representation of a hypergraph is the **incidence matrix**. For a hypergraph H=(V,E)H = (V,E)H=(V,E) with |V|=n|V| = n|V|=n vertices and |E|=m|E| = m|E|=m hyperedges, the incidence matrix is defined as:

H∈Rn×m\mathbf{H} \in \mathbb{R}^{n \times m}H∈Rn×m

Where:

H(v,e)={1if v∈e0otherwiseH(v,e) = \begin{cases} 1 & \text{if } v \in e \\ 0 & \text{otherwise} \end{cases}H(v,e)={10if v∈eotherwise

Thus, rows correspond to vertices and columns correspond to hyperedges.

Example

Vertices: V = {A, B, C, D}

Hyperedges:

- e_1 = {A, B, C}
- e_2 = {B, D}

Incidence matrix:

	e_1	**e_2**
A	1	0
B	1	1
C	1	0
D	0	1

Properties

- Captures exact vertex–hyperedge membership
- Supports algebraic operations

- Enables Laplacian construction
- Preserves higher-order structure

Hypergraph Laplacian often defined as:

L=Dv−HWDe−1HTL = D_v - H W D_e^{-1} H^TL=Dv−HWDe−1HT

Where:

DvD_vDv = vertex degree matrix

DeD_eDe = hyperedge degree matrix

WWW = hyperedge weight matrix

Advantages

- Exact and lossless representation
- Suitable for spectral clustering
- Supports hypergraph neural networks
- Mathematically tractable

Limitations

- Matrix becomes very large for big hypergraphs
- Sparse storage required
- Not directly compatible with standard graph algorithms

2. Clique Expansion Graph

Clique expansion converts each hyperedge into a fully connected subgraph (clique) among its vertices.

If hyperedge e={v1,v2,…,vk}e = \{v_1, v_2, …, v_k\}e={v1,v2,…,vk}, then edges are added between all vertex pairs:

(vi,vj),∀i≠j(v_i, v_j), \quad \forall i \neq j(vi,vj),∀i□=j

Thus, a hypergraph is reduced to a simple graph.

Example

Hyperedge: {A, B, C}

Clique expansion produces edges:

- (A,B)
- (A,C)
- (B,C)

Weighted Clique Expansion

Often edges weighted to preserve hyperedge size:

wij=∑e∋i,jwe|e|−1w_{ij} = \sum_{e \ni i,j} \frac{w_e}{|e|-1}wij=e∋i,j∑|e|−1we

This avoids over-counting large hyperedges.

Advantages

- Compatible with standard graph algorithms
- Easy to implement
- Enables use of graph neural networks
- Efficient for small hyperedges

Limitations

- Loses group identity
- Introduces artificial pairwise relations
- Distorts higher-order semantics
- Large hyperedges create dense cliques

Clique expansion is widely used but considered an approximation of hypergraph structure.

3. Star Expansion (Bipartite Representation)

Star expansion represents a hypergraph as a bipartite graph between vertices and hyperedges. A new node is introduced for each hyperedge, and edges connect hyperedge nodes to member vertices.

So hypergraph H=(V,E)H=(V,E)H=(V,E) becomes bipartite graph:

G′=(V∪E,F)G' = (V \cup E, F)G′=(V∪E,F)

Where:

(v,e)∈Fif v∈e(v,e) \in F \quad \text{if } v \in e(v,e)∈Fif v∈e

Example

Hyperedges:

- e_1 = {A,B,C}
- e_2 = {B,D}

Star expansion graph:

- e_1 connected to A,B,C

- e_2 connected to B,D

Properties

- Equivalent to incidence matrix
- Preserves hyperedge identity
- Converts hypergraph into graph form
- Bipartite structure

Advantages

- No loss of higher-order information
- Works with graph algorithms
- Supports message passing
- Used in hypergraph neural networks

Limitations

- Doubles number of nodes
- Path lengths change
- Requires bipartite algorithms

Star expansion is often preferred for machine learning because it preserves hyperedges explicitly.

4. Tensor Representation

Hypergraphs can also be represented as higher-order adjacency tensors, especially when hyperedges have fixed size (uniform hypergraphs).

For a k-uniform hypergraph, adjacency tensor:

A∈Rn×n×⋯×n(k dimensions)\mathcal{A} \in \mathbb{R}^{n \times n \times \dots \times n} \quad (k\text{ dimensions})A∈Rn×n×⋯×n(k dimensions)

Ai1i2…ik={1if {vi1,…,vik}∈E0otherwise\mathcal{A}_{i_1 i_2 … i_k} = \begin{cases} 1 & \text{if } \{v_{i_1},…,v_{i_k}\} \in E \\ 0 & \text{otherwise} \end{cases}Ai1i2…ik={10 if {vi1,…,vik}∈Eotherwise

Example (3-uniform hypergraph)

Hyperedge: {A,B,C}

Tensor entries:

AABC=AACB=ABAC=1A_{ABC} = A_{ACB} = A_{BAC} = 1AABC=AACB=ABAC=1

(all permutations)

Applications

- Tensor decomposition
- Higher-order clustering
- Motif detection
- Multi-way relations

Tensor representation directly models multi-dimensional interactions such as:

User × Item × Time

Author × Paper × Venue

Advantages

- Preserves full higher-order structure
- Suitable for multilinear algebra
- Captures multi-mode relations
- Enables latent factor mining

Limitations

- Extremely high dimensional
- Memory intensive
- Computationally expensive
- Difficult visualization

HIGHER-ORDER NETWORK MODELS

Higher-order data mining extends beyond hypergraphs to more general structures capturing multi-way relations.

1. Simplicial Complexes

A simplicial complex is a collection of simplices representing higher-order interactions:

- 0-simplex: node
- 1-simplex: edge
- 2-simplex: triangle
- k-simplex: group of k+1 nodes

These structures preserve hierarchy of interactions.

Example: If {A,B,C} is a 2-simplex, pairwise edges also exist.

2. Tensor-Based Models

Many higher-order datasets naturally form tensors:

Examples:

- User × Item × Time
- Author × Paper × Venue
- Gene × Condition × Experiment

Tensor decomposition methods such as CP and Tucker extract latent factors across multiple dimensions.

3. Higher-Order Random Walks

Instead of node-to-node transitions, transitions occur across hyperedges or simplices. This models group-level diffusion and influence propagation.

HYPERGRAPH DATA MINING TASKS

1. Hypergraph Clustering

Clustering groups vertices based on shared hyperedges rather than pairwise similarity.

Approaches:

- Spectral hypergraph clustering
- Partitioning via Laplacian
- Tensor clustering

Hypergraph Laplacian generalizes graph Laplacian and preserves group structure.

Advantages:

- Captures multi-entity similarity
- Avoids clique distortion
- Better community detection

2. Hypergraph Classification

Nodes or hyperedges can be classified using higher-order features.

Methods:

- Hypergraph neural networks
- Hyperedge feature aggregation
- Semi-supervised learning

Applications:

- Document categorization
- Disease classification
- User interest prediction

3. Hypergraph Community Detection

Communities in hypergraphs represent groups of entities frequently co-occurring.

Techniques:

- Hypergraph modularity optimization
- Probabilistic generative models
- Non-negative tensor factorization

This is more accurate than graph community detection for group data.

4. Frequent Hyperedge Pattern Mining

Extends frequent itemset mining to hypergraphs.

Goal: discover recurring groups of nodes.

Example:

Frequent author teams

Frequent gene interaction sets

Methods:

- Apriori-like hyperedge mining
- Hyperclique patterns
- Tensor pattern mining

HYPERGRAPH REPRESENTATION LEARNING

Representation learning aims to embed nodes or hyperedges into vector space while preserving higher-order structure.

1. Hypergraph Embedding Methods

Method	Idea	Strength
Spectral Embedding	Laplacian eigenvectors	Theoretical basis
Random Walk	Higher-order walks	Scalable

Method	Idea	Strength
Neural Hypergraph	Message passing	Nonlinear patterns
Tensor Factorization	Multi-mode factors	Rich relations

2. Hypergraph Neural Networks (HGNN)

HGNN extends graph neural networks to hypergraphs.

Process:

- Aggregate features from nodes in hyperedge
- Update hyperedge representation
- Propagate back to nodes

This two-stage aggregation captures group context.

3. Higher-Order Graph Neural Networks

Higher-order GNNs operate on simplices instead of edges.

Capabilities:

- Capture triadic closure
- Model group dynamics
- Learn motif-level features

These models improve performance in complex relational tasks.

APPLICATIONS OF HYPERGRAPH AND HIGHER-ORDER MINING

1. Social Network Analysis

Many social interactions occur in groups rather than pairs:

Examples:

- Group chats
- Events participation
- Collaboration teams

Hypergraph mining helps:

- Detect communities
- Identify influencers in groups
- Model group diffusion

2. Recommender Systems

User preferences often involve sets of items.

Examples:

- Shopping baskets
- Playlist creation
- Travel packages

Hypergraph models capture co-consumption patterns better than pairwise similarity.

3. Bioinformatics and Systems Biology

Biological processes involve multi-molecule interactions.

Examples:

- Protein complexes
- Metabolic pathways
- Gene regulatory modules

Hypergraphs represent biochemical reactions naturally.

4. Knowledge Graphs and Semantic Mining

Knowledge graphs contain multi-entity relations such as:

(Author, Paper, Venue)

(Drug, Target, Disease)

Higher-order mining discovers relational patterns and ontology structures.

5. Computer Vision and Multimedia

Images contain groups of objects and regions.

Hypergraph models:

- Image segmentation
- Object co-occurrence
- Video event detection

They capture spatial and semantic grouping.

COMPARATIVE ANALYSIS: GRAPH VS HYPERGRAPH MINING

Aspect	Graph Mining	Hypergraph Mining
Relation Type	Pairwise	Multi-entity
Information Loss	High	Low
Community Quality	Moderate	High
Model Complexity	Lower	Higher
Representation	Matrix	Tensor/Incidence
Applications	Networks	Group systems

Hypergraph mining preserves original interaction semantics better.

CHALLENGES IN HYPERGRAPH AND HIGHER-ORDER MINING

Despite advantages, several challenges exist.

1. Scalability

Hypergraphs often large and dense.

Complexity grows exponentially with hyperedge size.

Issues:

- Large incidence matrices
- Tensor explosion
- Memory constraints

2. Data Sparsity

Higher-order relations sparse in many datasets.

Example:

Few large groups exist compared to pairs.

This affects learning stability.

3. Interpretability

Hypergraph embeddings and tensor factors difficult to interpret.

Understanding group semantics remains challenge.

4. Dynamic Hypergraphs

Real-world interactions evolve over time:

Examples:

- Changing collaboration teams
- Temporal group chats
- Biological evolution

Temporal hypergraph mining still developing.

5. Evaluation Metrics

Standard graph metrics not suitable.

Need new measures:

- Hyperedge modularity
- Higher-order centrality
- Group cohesion

FUTURE RESEARCH DIRECTIONS

Hypergraph mining is rapidly growing field. Several promising directions exist.

1. Scalable Hypergraph Learning

Development of distributed and approximate algorithms for large hypergraphs.

2. Explainable Higher-Order Models

Interpretable embeddings and group explanations needed.

3. Dynamic and Streaming Hypergraphs

Continuous mining of evolving group relations.

4. Integration with Deep Learning

Combining hypergraph structure with deep architectures.

5. Cross-Domain Higher-Order Mining

Joint analysis of multiple hypergraphs such as social + biological networks.

CONCLUSION

Hypergraph and higher-order data mining provides a powerful framework for analyzing complex relational data involving multi-entity interactions. Traditional graph-based approaches often fail to capture group semantics and higher-order dependencies, leading to information loss and inaccurate patterns. Hypergraphs, simplicial complexes, and tensor models overcome these limitations by directly representing multi-way relations.

This paper reviewed the foundations, mining tasks, representation learning methods, and applications of hypergraph and higher-order data mining. We discussed clustering, classification, community detection, and embedding techniques along with practical applications in social networks, recommender systems, bioinformatics, and knowledge graphs. Comparative analysis shows that hypergraph mining yields more meaningful structural insights than pairwise graph mining.

However, challenges such as scalability, sparsity, interpretability, and dynamic modeling remain open research problems. Future work should focus on scalable algorithms, explainable higher-order learning, and temporal hypergraph analysis. As real-world datasets increasingly involve group interactions, hypergraph and higher-order data mining will become essential for next-generation knowledge discovery and intelligent systems.

REFERENCES

1. Berge, C. *Hypergraphs: Combinatorics of Finite Sets*. North-Holland, 1989.
2. Zhou, D., Huang, J., & Schölkopf, B. "Learning with Hypergraphs: Clustering and Classification." *NIPS*, 2006.
3. Agarwal, S., Branson, K., & Belongie, S. "Higher Order Learning with Graphs." *ICML*, 2006.
4. Benson, A., Gleich, D., & Leskovec, J. "Higher-Order Organization of Complex Networks." *Science*, 2016.
5. Battiston, F., et al. "Networks Beyond Pairwise Interactions." *Nature Physics*, 2020.
6. Yadati, N., et al. "Hypergraph Neural Networks." *NeurIPS*, 2019.
7. Aksoy, S., et al. "Hypernetwork Science via High-Order Hypergraph Walks." *EPJ Data Science*, 2020.

8. Kolda, T., & Bader, B. “Tensor Decompositions and Applications.” *SIAM Review*, 2009.
9. Chitra, U., & Raphael, B. “Random Walks on Hypergraphs.” *MLG Workshop*, 2019.
10. Feng, Y., You, H., Zhang, Z., Ji, R., & GAO, Y. “Hypergraph Neural Networks.” *AAAI*, 2019.