Structural Deep Embedding for Hyper-Networks

Ke Tu
Tsinghua U

Peng Cui
Tsinghua U

Xiao Wang
Tsinghua U

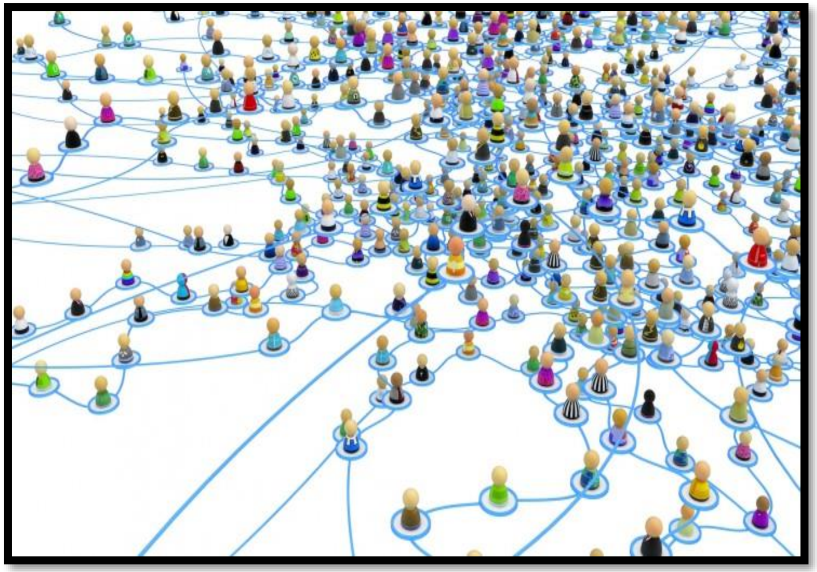
Fei Wang
Cornell U

Wenwu Zhu
Tsinghua U

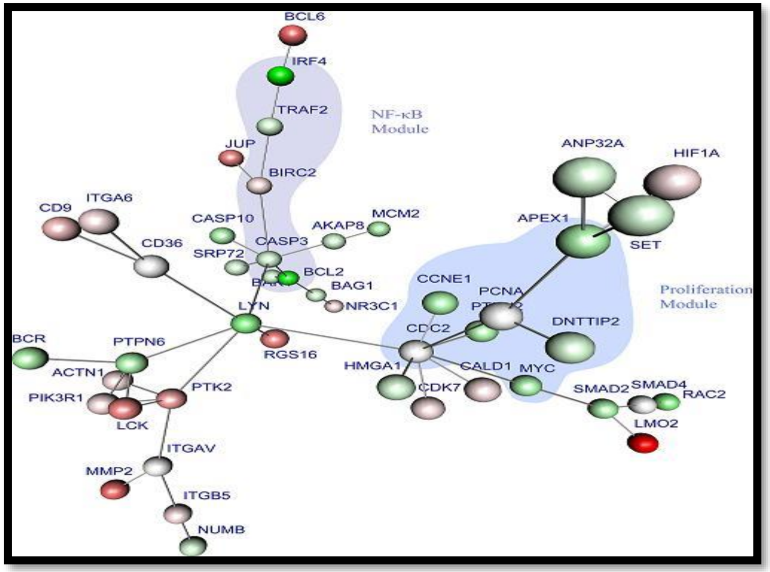




Network Analytics



Social Networks

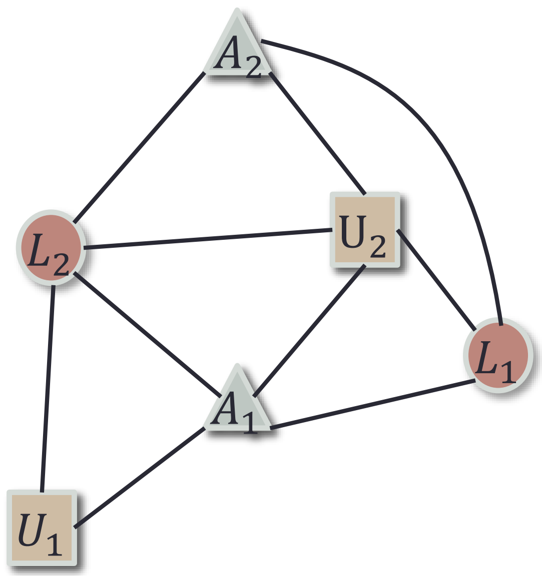


biology Networks

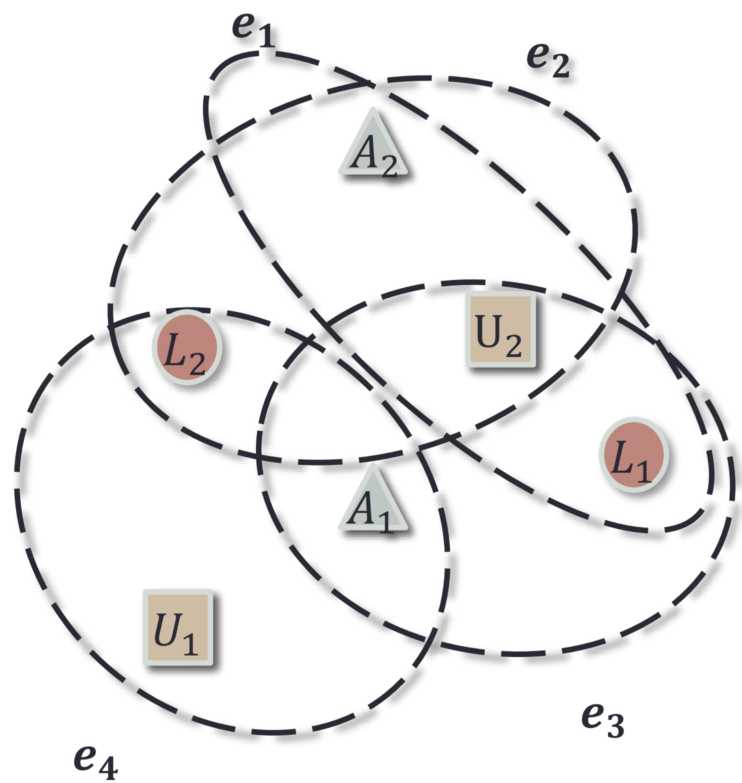
Networks are widely used to represent the rich pairwise relationships of data objects

However, in real world applications, the relationships among data points could go beyond pairwise

Hyper-network embedding



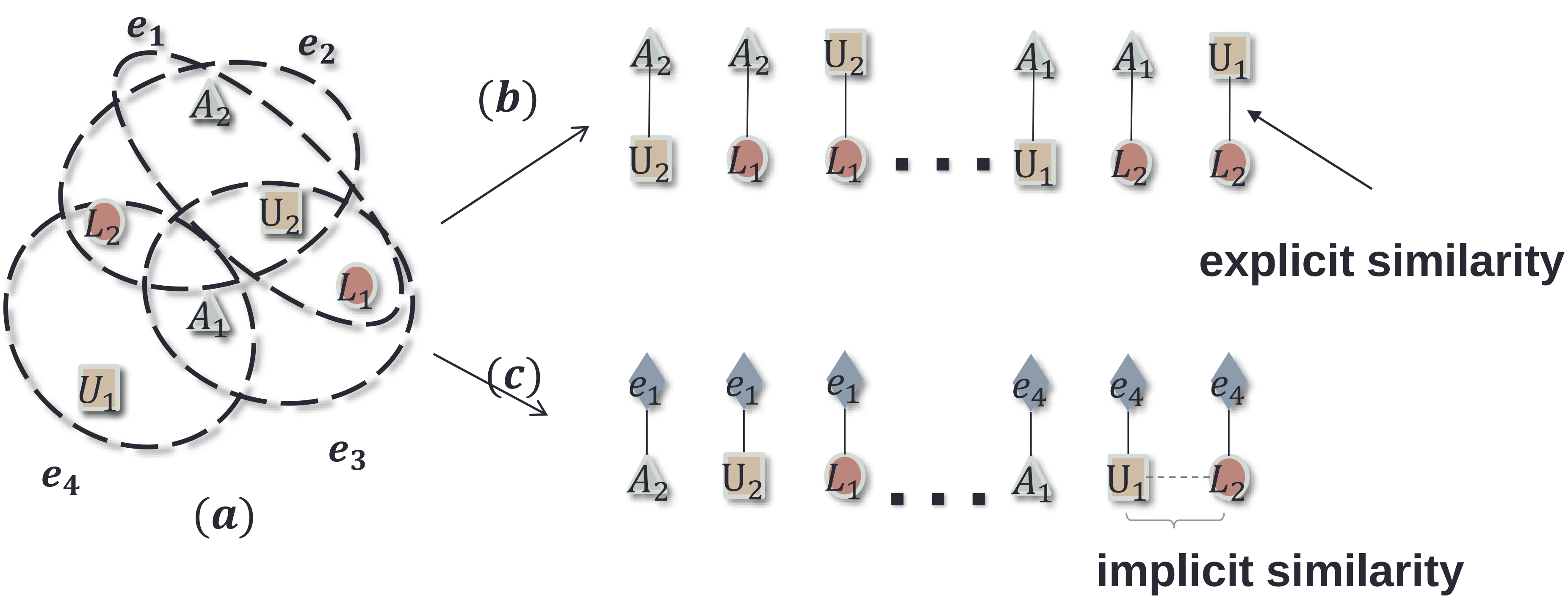
Networks



Hyper-Networks

- A hyper-network is a network in which an edge can include any number of nodes

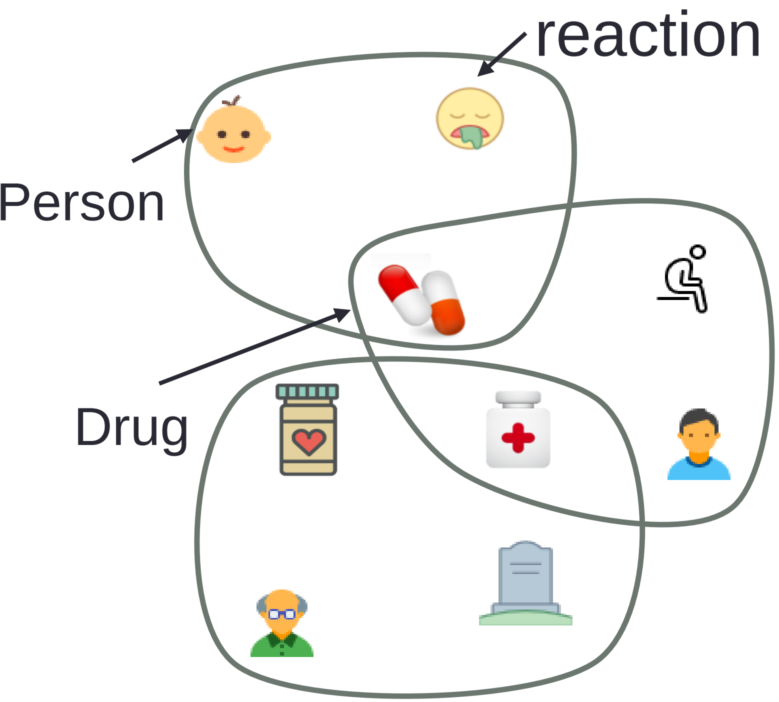
Naïve solution: Expand into Conventional Networks



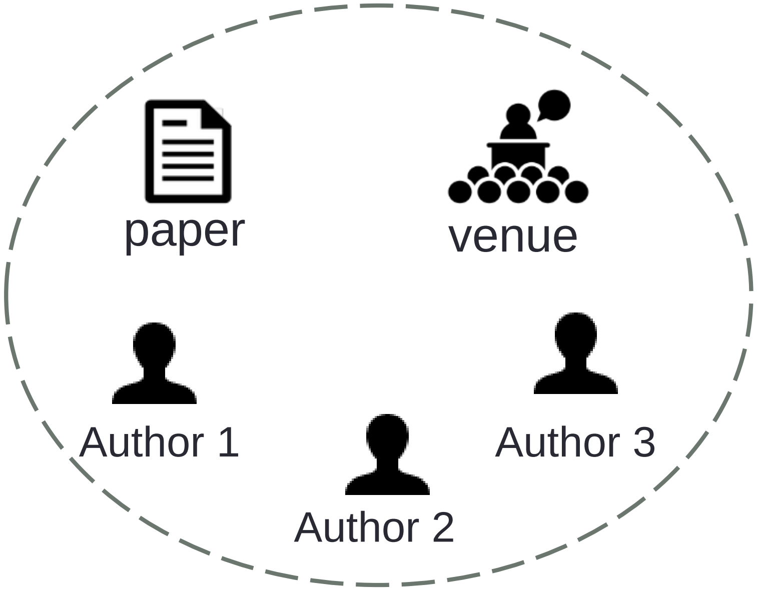
(a) A hyper-network (b) The clique expansion. (c) The star expansion

We usually assume that the hyper-edges are decomposable.

Hyper-edges are often indecomposable

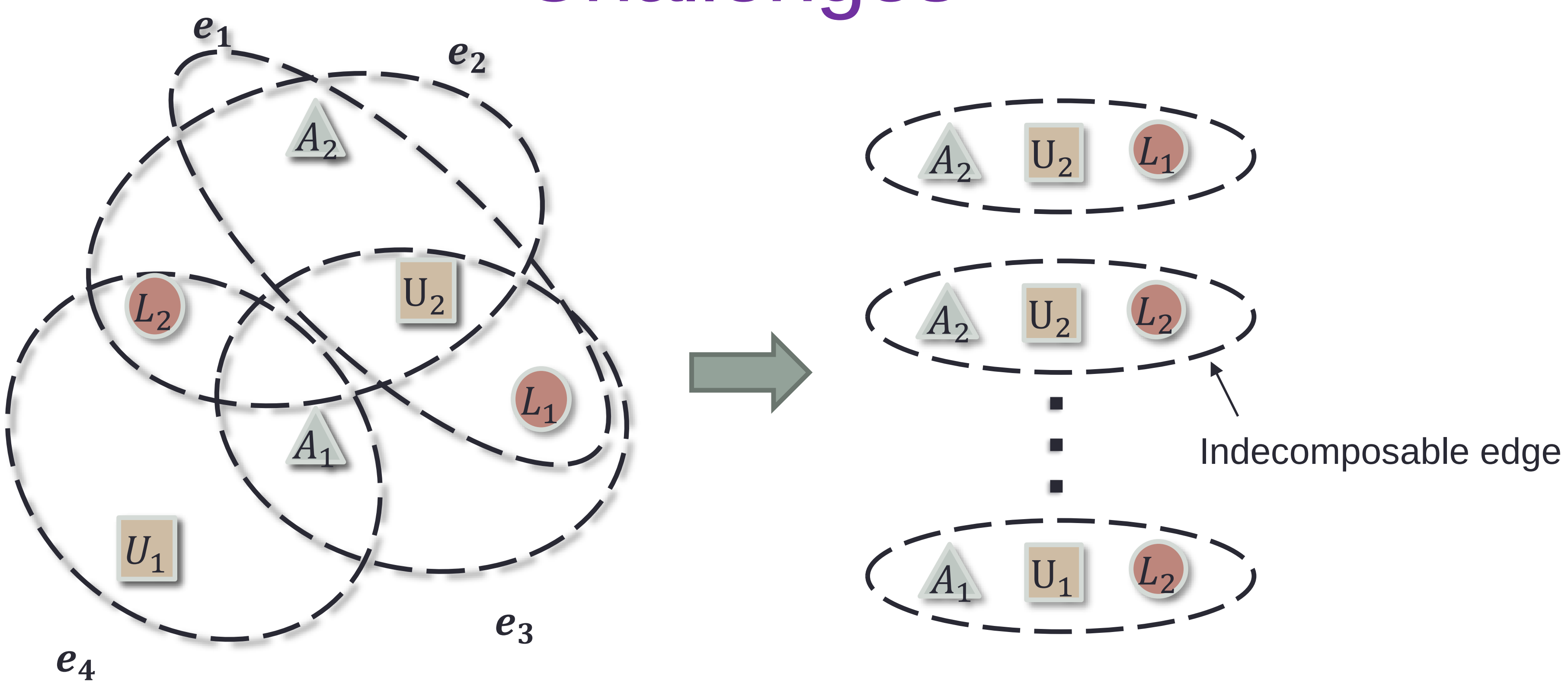


Adverse Drug Network



Bibliographic Network

Challenges



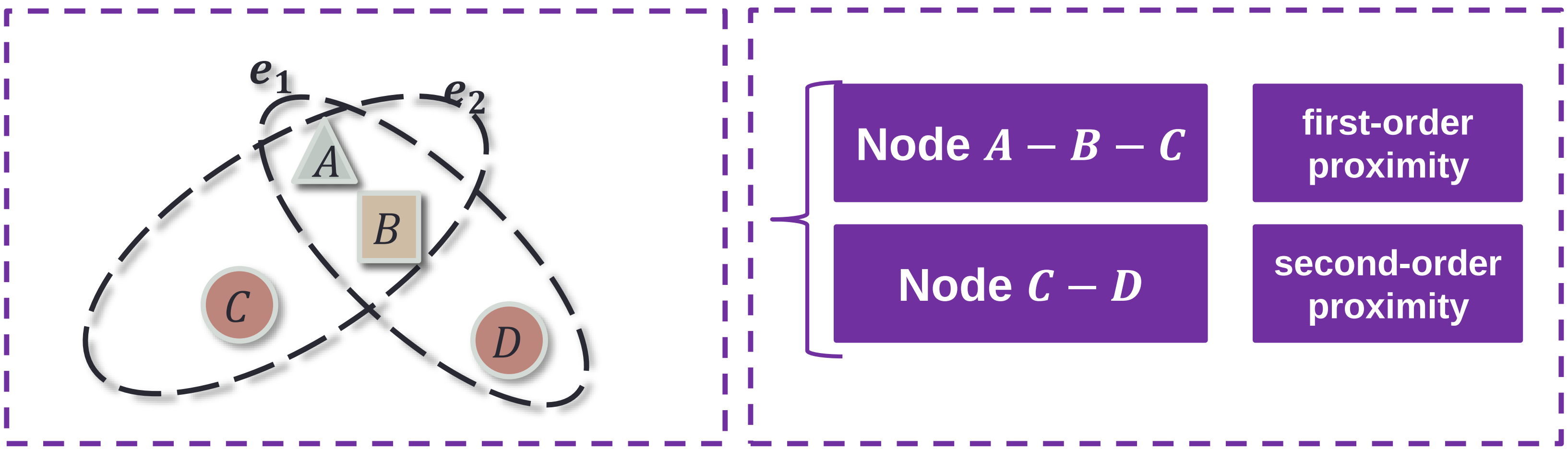
- How to preserve the indecomposable relationships while learning representations?
- How to preserve structures for sparse hyper-networks?

Existing Methods

- Spectral hypergraph representation learning (Liu et al. 2013) and HyperEdge Based Embedding (HEBE) (Gui et al. 2016)
  - cannot preserve the structure of indecomposable hyperedges
- Tensor decomposition (Kolda and Bader 2009)
  - The time cost is very expensive so it cannot scale efficiently to large network

In summary, none of existing methods solve the hyper-network embedding problem well.

First- and second-order proximities



- The first-order proximity of hyper-network measures the N-tuplewise similarity between nodes
- The second-order proximity of hyper-network measures the proximity of two nodes with respect to their neighborhood structures.

N-tuplewise similarity requires non-linear model

- Preserve N-tuplewise similarity in embedding space

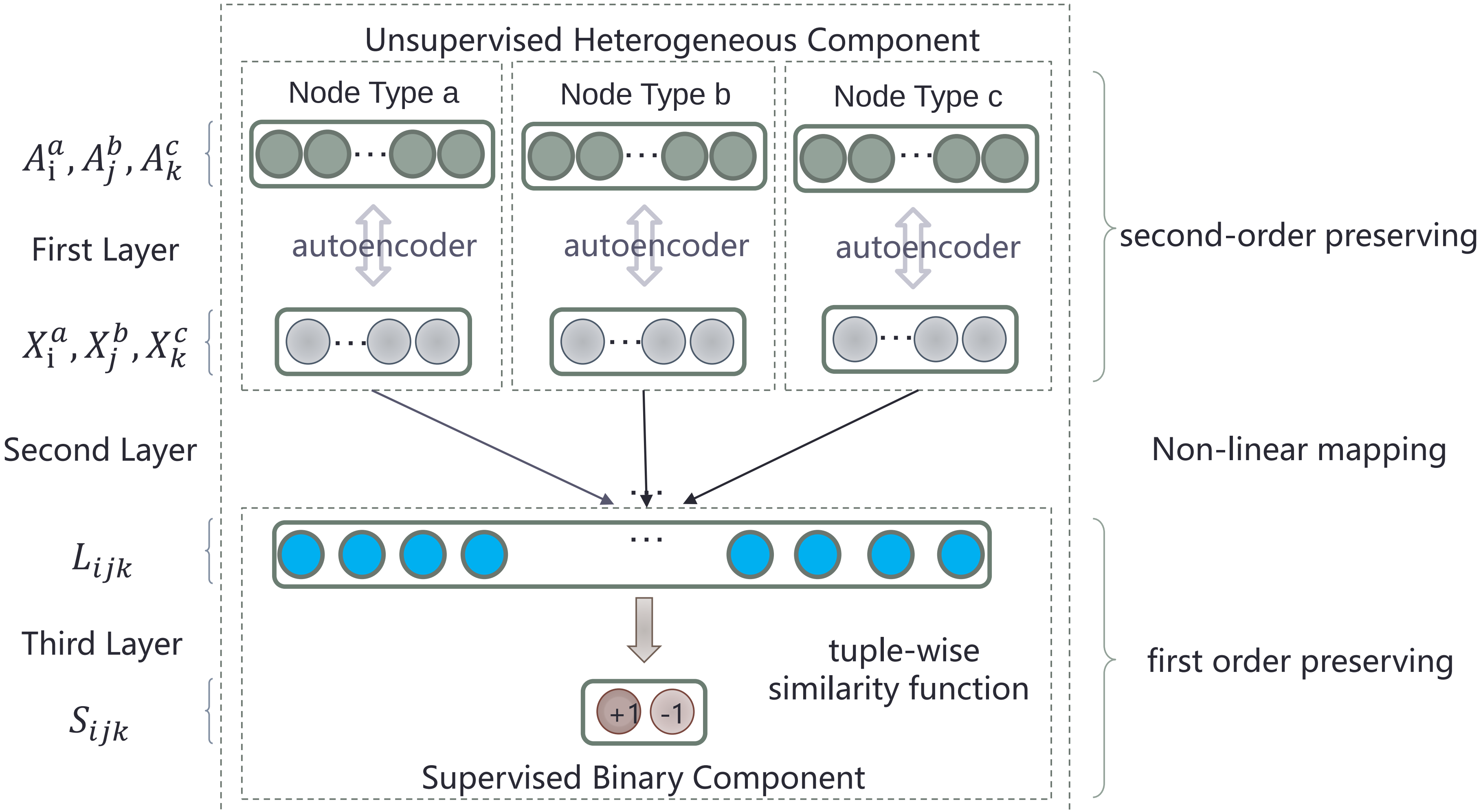
Property 1. *We mark* $\mathbf{X}_i$ *as the embedding of node* v_i *and* $\mathcal{S}$ *as* N*-tuplewise similarity function.*

- *if* $(\mathrm{v}_1, \mathrm{v}_2, ..., \mathrm{v}_N) \in \mathbf{E}$, $\mathcal{S}(\mathbf{X}_1, \mathbf{X}_2, .., \mathbf{X}_N)$ *should be large (without loss of generality, large than a threshold* l*).*
- *if* $(\mathrm{v}_1, \mathrm{v}_2, ..., \mathrm{v}_N) \notin \mathbf{E}$, $\mathcal{S}(\mathbf{X}_1, \mathbf{X}_2, .., \mathbf{X}_N)$ *should be small (without loss of generality, smaller than a threshold* s*).*

Theorem 1. *Linear function* $\mathcal{S}(\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_N) = \sum_i \mathbf{W}_i \mathbf{X}_i$ *cannot satisfy Property 1.*

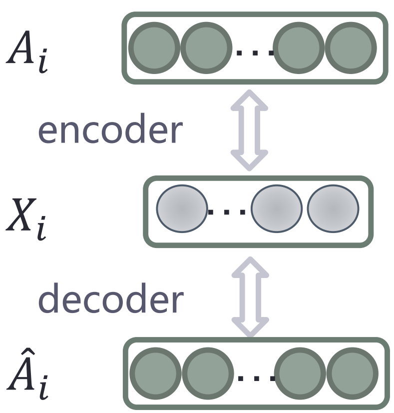
- We use deep neural network to solve this non-linear problem.

Structural Deep Network for Hyper-network



Preserve second-order proximity

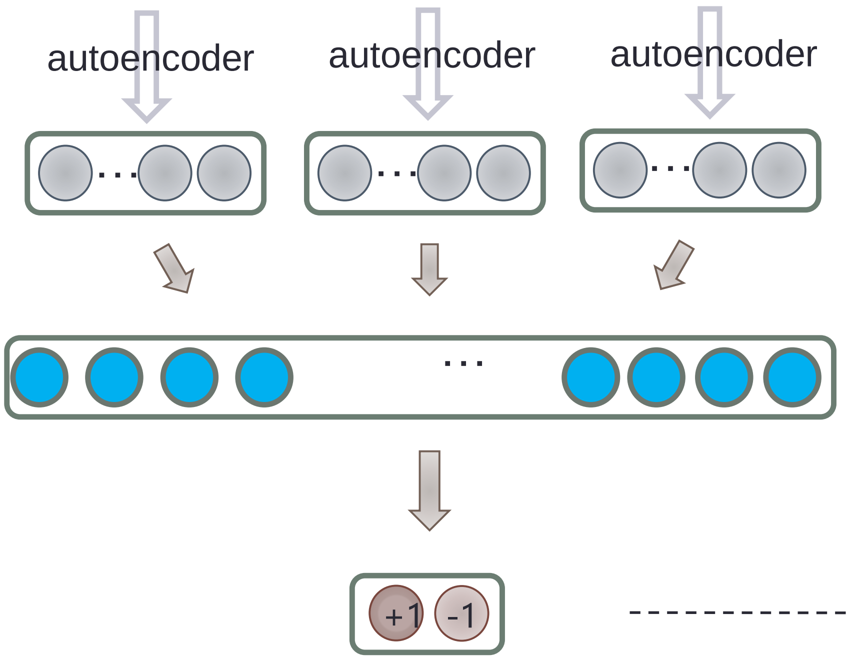
- For a hypergraph G = (V;E),
  - Incidence matrix H with entries h(v; e) = 1 if v ∈ e and 0 otherwise
  - D_v denotes the diagonal matrix containing the vertex degree
  - The adjacency matrix A = $HH^T - D_v$
    - the i-th row of adjacency matrix A shows the neighborhood structure of vertex v_i



- encoder: $X_i = \sigma\left(W^{(1)} * X_i + b^{(1)}\right)$
- decoder: $\hat{A}_i = \sigma\left(\widehat{W}^{(1)} * X_i + \hat{b}^{(1)}\right)$
- Loss = $\|\, sign(A_i) \odot (A_i - \hat{A}_i) \|_F^2$

Sparse constraint

Preserve first-order proximity



Input embeddings of 3 nodes

$$X_i^a, X_j^b, X_k^c$$

Latent space

$$\mathbf{L}_{ijk} = \sigma(\mathbf{W}_A^{(2)} * \mathbf{X}_i^A + \mathbf{W}_B^{(2)} * \mathbf{X}_j^B + \mathbf{W}_C^{(2)} * \mathbf{X}_k^C + \mathbf{b}^{(2)})$$

N-tuplewise similarity

$$\mathbf{S}_{ijk} \equiv \mathcal{S}(\mathbf{X}_i^A, \mathbf{X}_j^B, \mathbf{X}_k^C) = \sigma(\mathbf{W}^{(3)} * \mathbf{L}_{ijk} + \mathbf{b}^{(3)}).$$

Loss

$$\mathcal{L}_1 = -(\mathbf{R}_{ijk} log \mathbf{S}_{ijk} + (1 - \mathbf{R}_{ijk}) log(1 - \mathbf{S}_{ijk})).$$

$R_{ijk} = 1$ if (v_i, v_j, v_k) is a hyperedge and 0 otherwise

Time Complexity Analysis

- The time complexity of training process is $O((nd + dl + l)bI)$
  - n is the number of nodes
  - d is the dimension of embedding vectors, set as 16, 32, 64, 128
  - l is the size of latent layer, set as $3d$
  - b is the batch size
  - I is the number of iterations(epochs), set as 10

- Conclusion: the complexity of training process is linear to the number of nodes n.

Experiment

- four different types of hyper-networks
  - GPS network: GPS
  - social network: MovieLens
  - medical network: drug
  - semantic network: wordnet
- three applications
  - reconstruction, link prediction, node classification

Table 2: Statistics of the datasets

datasets	node type			#(V)			#(E)
GPS	user	location	activity	146	70	5	1436
MovieLens	user	movie	tag	2113	5908	9079	47957
drug	user	drug	reaction	12	1076	6398	171756
wordnet	head	relation	tail	40504	18	40551	145966

Experiment: reconstruction

Table 3: AUC value for network reconstruction

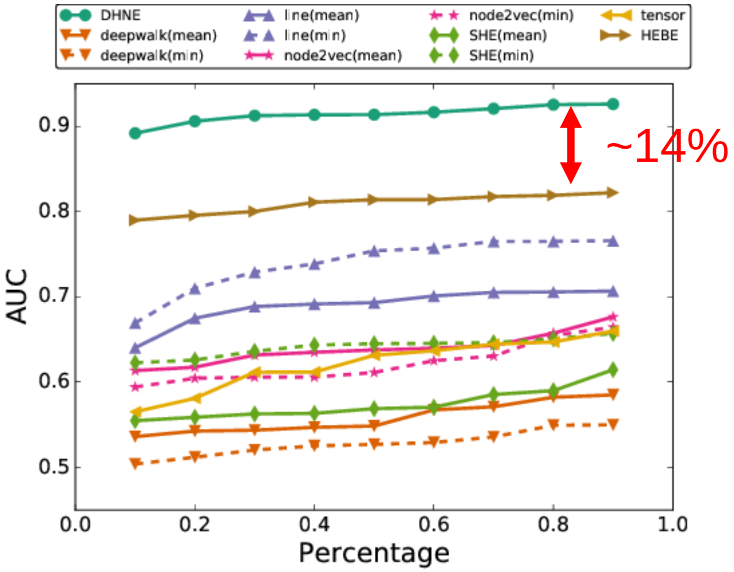
methods		GPS	MovieLens	drug	wordnet
DHNE		**0.9598**	**0.9344**	**0.9356**	**0.9073**
mean	deepwalk	0.6714	0.8233	0.5750	0.8176
	line	0.8058	0.8431	0.6908	0.8365
	node2vec	0.6715	0.9142	0.6694	0.8609
	SHE	0.8596	0.7530	0.5486	0.5618
min	deepwalk	0.6034	0.7117	0.5321	0.7423
	line	0.7369	0.7910	0.7625	0.7751
	node2vec	0.6578	0.9100	0.6557	0.8387
	SHE	0.7981	0.7972	0.6236	0.5918
tensor		0.9229	0.8640	0.7025	0.7771
HEBE		0.9337	0.8772	0.8236	0.7391

Experiment: link prediction

Table 4: AUC value for link prediction

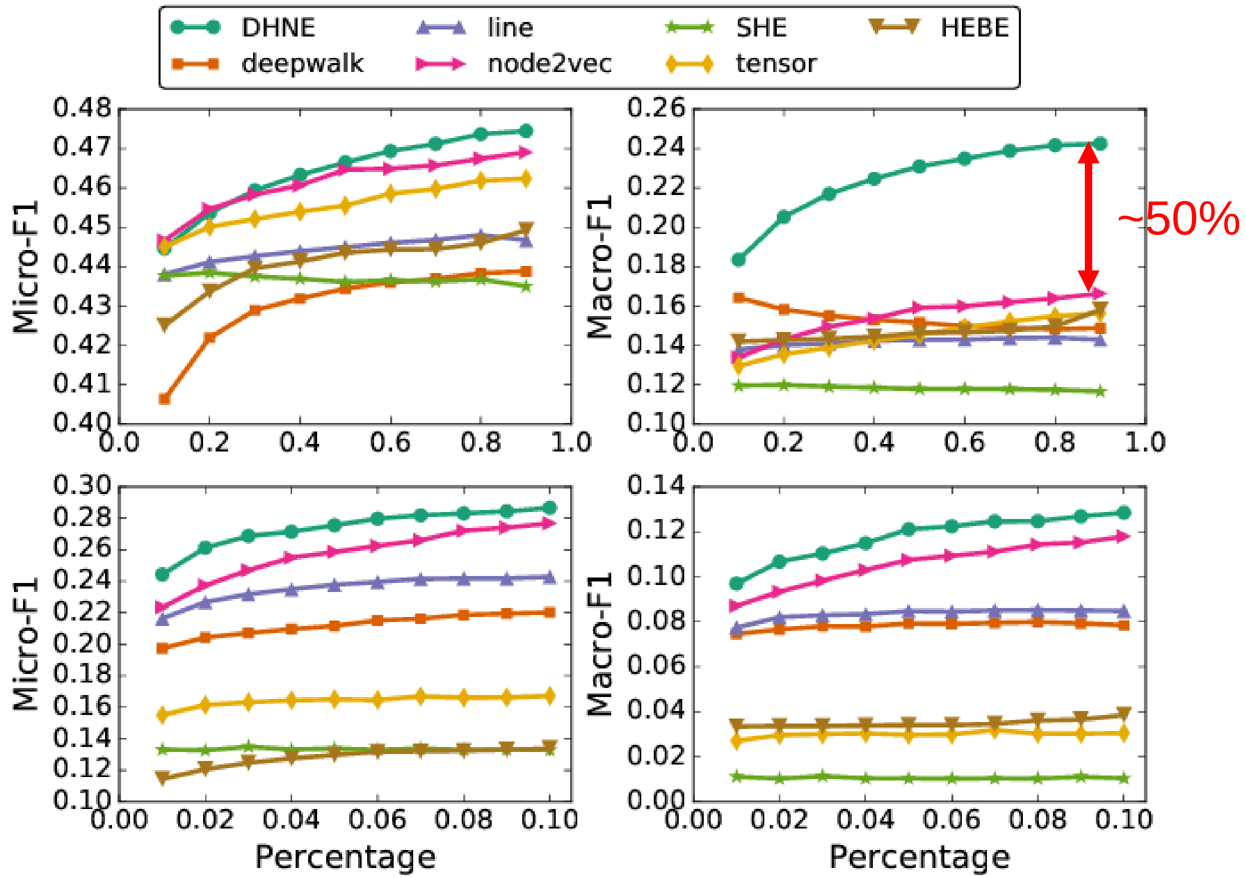
methods		GPS	MovieLens	drug	wordnet
DHNE		**0.9166**	**0.8676**	**0.9254**	**0.8268**
mean	deepwalk	0.6593	0.7151	0.5822	0.5952
	line	0.7795	0.7170	0.7057	0.6819
	node2vec	0.5835	0.8211	0.6573	0.8003
	SHE	0.8687	0.7459	0.5899	0.5426
min	deepwalk	0.5715	0.6307	0.5493	0.5542
	line	0.7219	0.6265	0.7651	0.6225
	node2vec	0.5869	0.7675	0.6546	0.7985
	SHE	0.8078	0.8012	0.6508	0.5507
tensor		0.8646	0.7201	0.6470	0.6516
HEBE		0.8355	0.7740	0.8191	0.6364

the overall performance



Performance on networks of different sparsity

Experiment: classification



top: multi-label classification on MovieLens dataset;
bottom: multi-class classification on wordnet dataset

Conclusion

- Firstly investigate the problem of indecomposable hyper-network embedding.
  - theoretically prove that any linear similarity metric cannot maintain the indecomposibility property
- Propose a novel deep model.
  - maintain the indecomposibility as well as the sparsity issue.
  - linear time complexity to the number of node
- Extensive experiments on three applications
  - four different types of hyper-networks

Thanks!

Ke Tu, Tsinghua University

tuke1993@163.com

Reference:

[1] Tu, K; Cui, P; Wang, X; Wang, F; Zhu, W. Structural Deep Embedding for Indecomposable Hyper-Networks. AAAI, 2018.