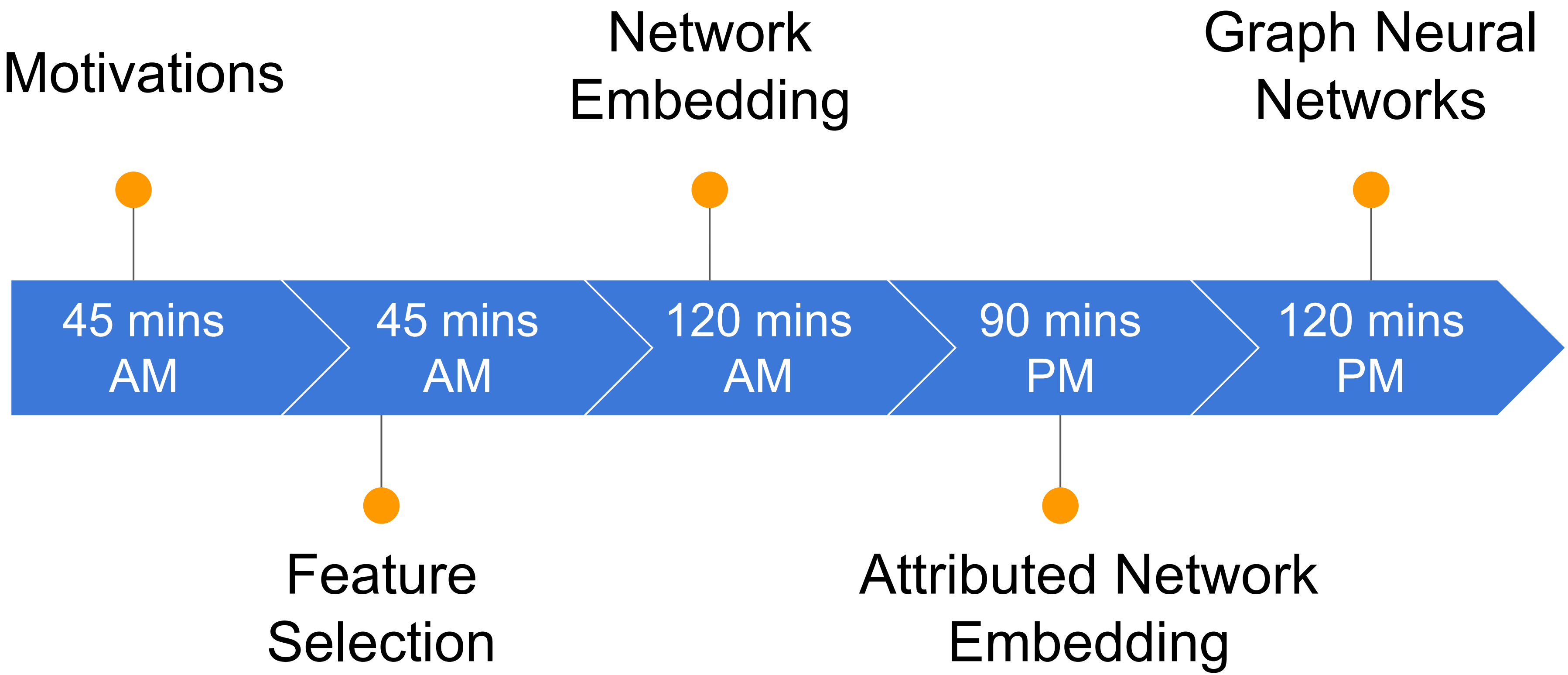
KDD 2019

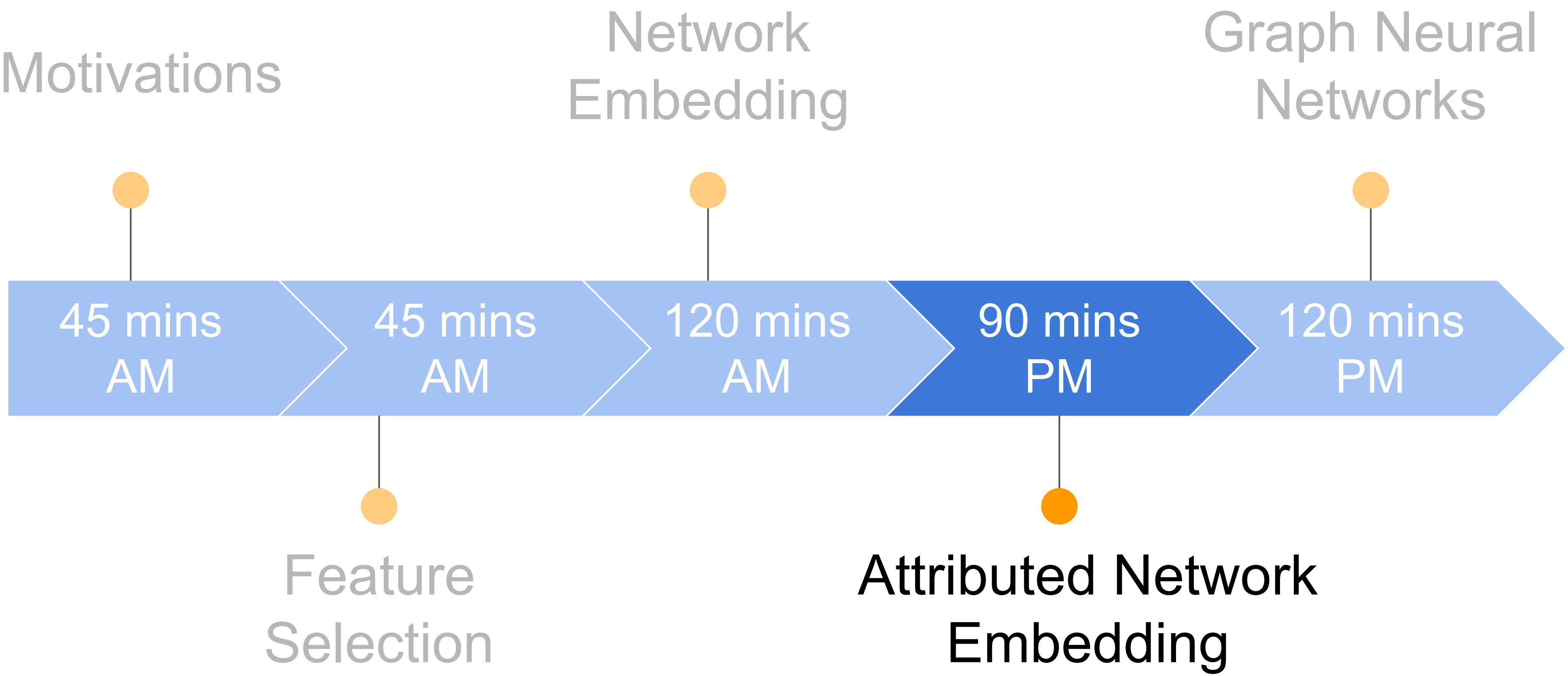
Learning From Networks

——*Algorithms, Theory, & Applications*

Xiao Huang, Peng Cui, Yuxiao Dong, Jundong Li, Huan Liu, Jian Pei, Le Song, Jie Tang, Fei Wang, Hongxia Yang, Wenwu Zhu

xhuang@tamu.edu; cuip@tsinghua.edu.cn; yuxdong@microsoft.com; jundongl@asu.edu; huan.liu@asu.edu; jpei@cs.sfu.ca; le.song@antfin.com; jietang@tsinghua.edu.cn; few2001@med.cornell.edu; yang.yhx@alibaba-inc.com; wwzhu@tsinghua.edu.cn;

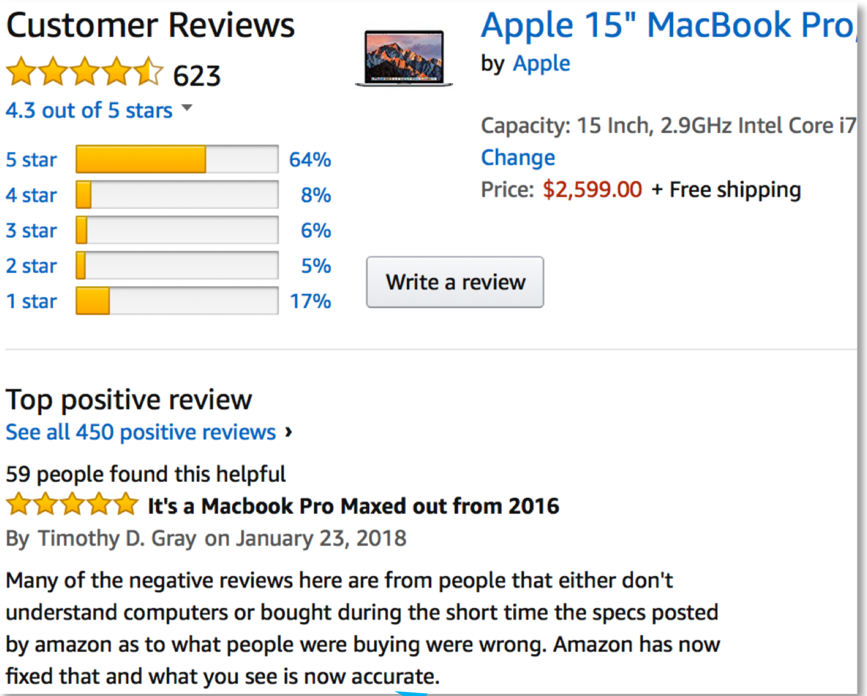

Motivations
45 mins AM
Feature Selection
45 mins AM
Network Embedding
120 mins AM
Attributed Network Embedding
90 mins PM
Graph Neural Networks
120 mins PM


Motivations
45 mins AM
Feature Selection
45 mins AM
Network Embedding
120 mins AM
Attributed Network Embedding
90 mins PM
Graph Neural Networks
120 mins PM

Attributed network embedding

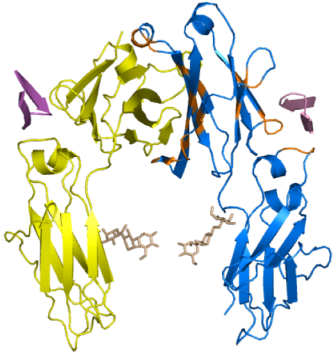
- **Motivations & challenges**
  - What are attributed networks and why embedding
  - Formal definitions and challenges
- **Mining attributed networks with shallow embedding**
- **Mining attributed networks with deep embedding**
- **Human-centric network analysis**

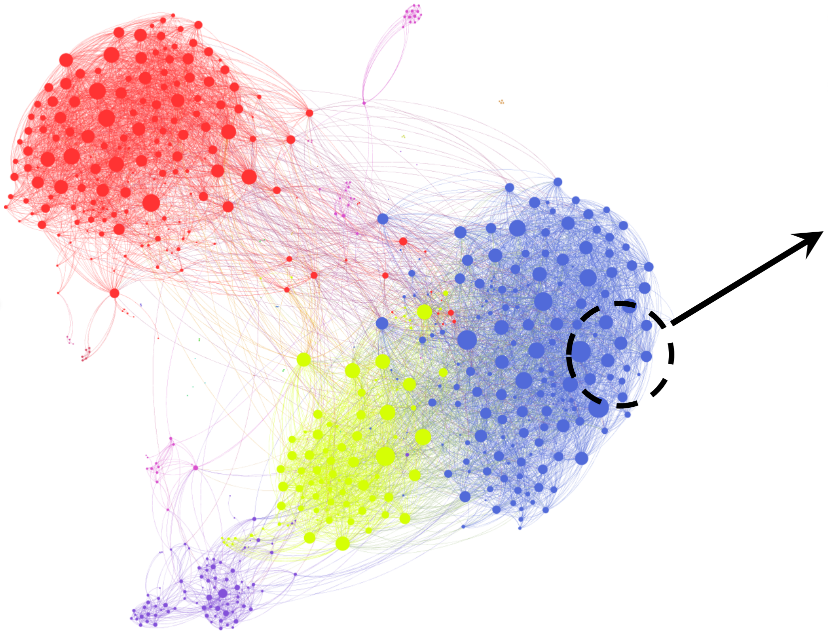
Example of node attributes



- Examples: user content in social media, reviews in co-purchasing networks, & paper abstracts in citation networks

Attributed networks are prevalent in practice





Nodes Have
Different Attributes

- Node attributes: a rich set of data describing the unique characteristics of each node

Node attributes & network are correlated

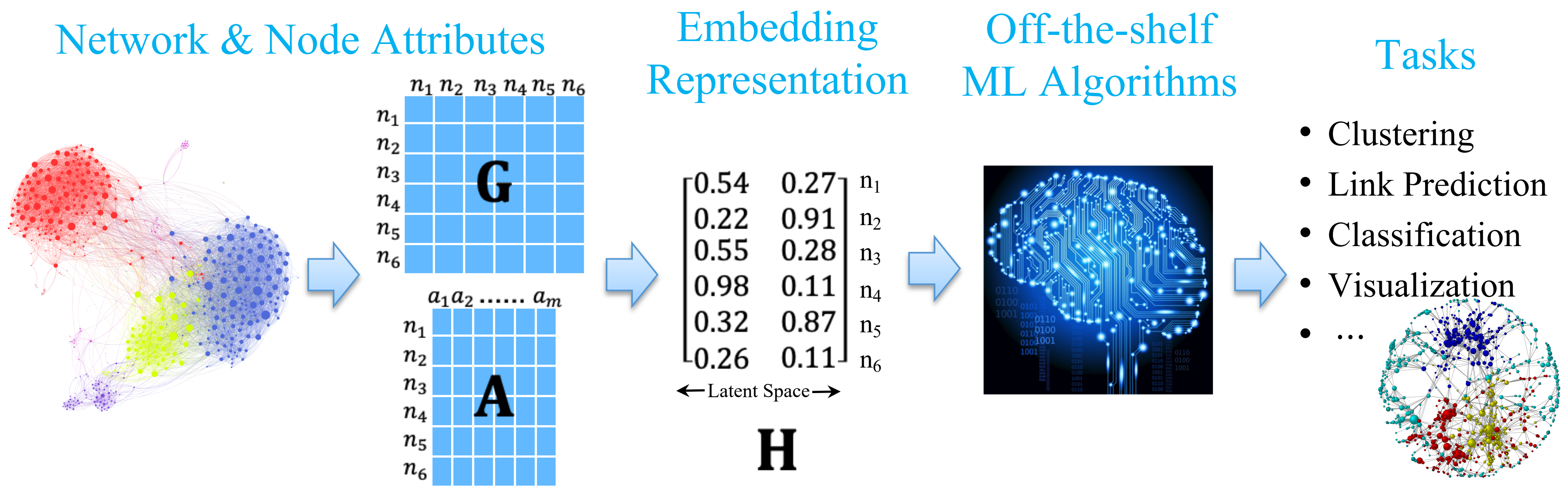
- Node attributes and network influence each other and are inherently correlated
  - Explained by Homophily & social influence
  - High correlation of user posts & following relationships
  - Strong association between customer reviews & co-purchasing networks

Hypothesis testing on correlation

Dataset	Scenarios	CorrCoef	p-value
BlogCatalog	Real-world	**3.69e-002**	**0.00e-016**
	RandomMean	3.14e-005	0.18
	RandomMax	1.40e-003	4.42e-016
Flickr	Real-world	**1.85e-002**	**0.00e-016**
	RandomMean	2.15e-005	0.49
	RandomMax	5.48e-004	3.37e-003

- Hypothesis: there is no correlation between network affinities and node attribute affinities (a significance level of 0.05)
- CorrCoef: Pearson correlation coefficient of two types of affinities
- Real-world network vs randomly-generated networks
    - Mean and max results of 100 synthetic networks

Attributed network embedding



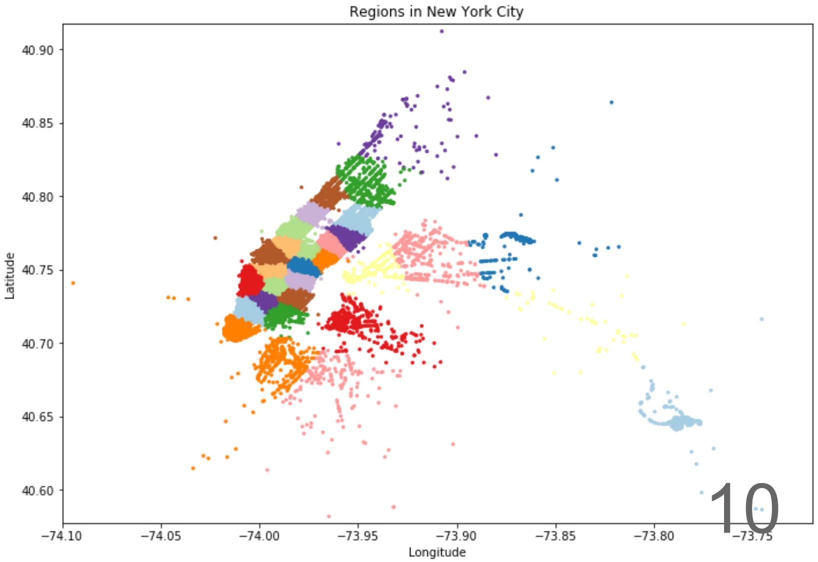
- Given $\mathbf{G}$ and $\mathbf{A}$, we aim to represent each node as a *d*-dimensional vector $\mathbf{h}_i$, such that $\mathbf{H}$ can preserve node proximity both in network and node attributes

Why attributed network embedding

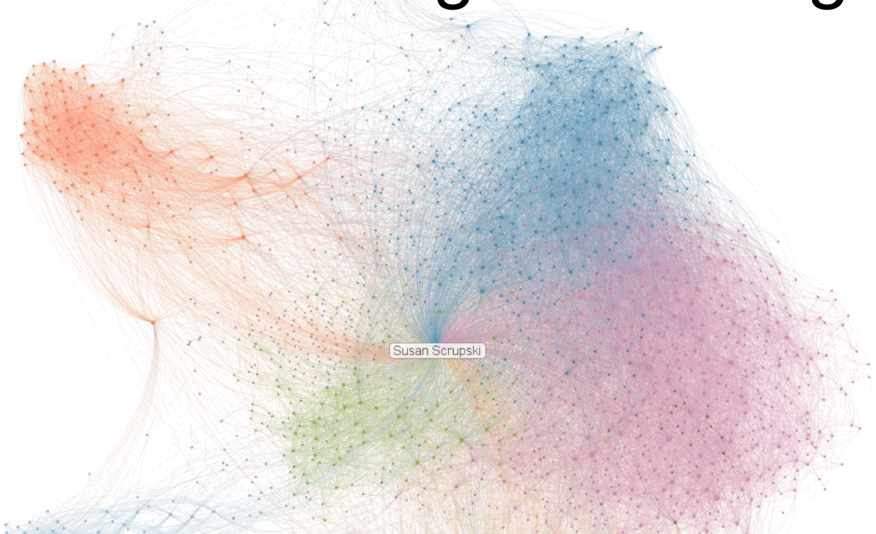
- Traditional graph theory based analysis achieves suboptimal in large-scale networks with complex tasks
  - Shortest path, maximum flow, centrality

- Aim to take advantage of off-the-shelf machine learning algorithms

- Provide general ways to handle the heterogeneous info in networked systems
  - Friend recommendation: social links, textual posts, categorical attributes, images.
  - Taxi demand forecast: region networks, demographic and meteorological data.

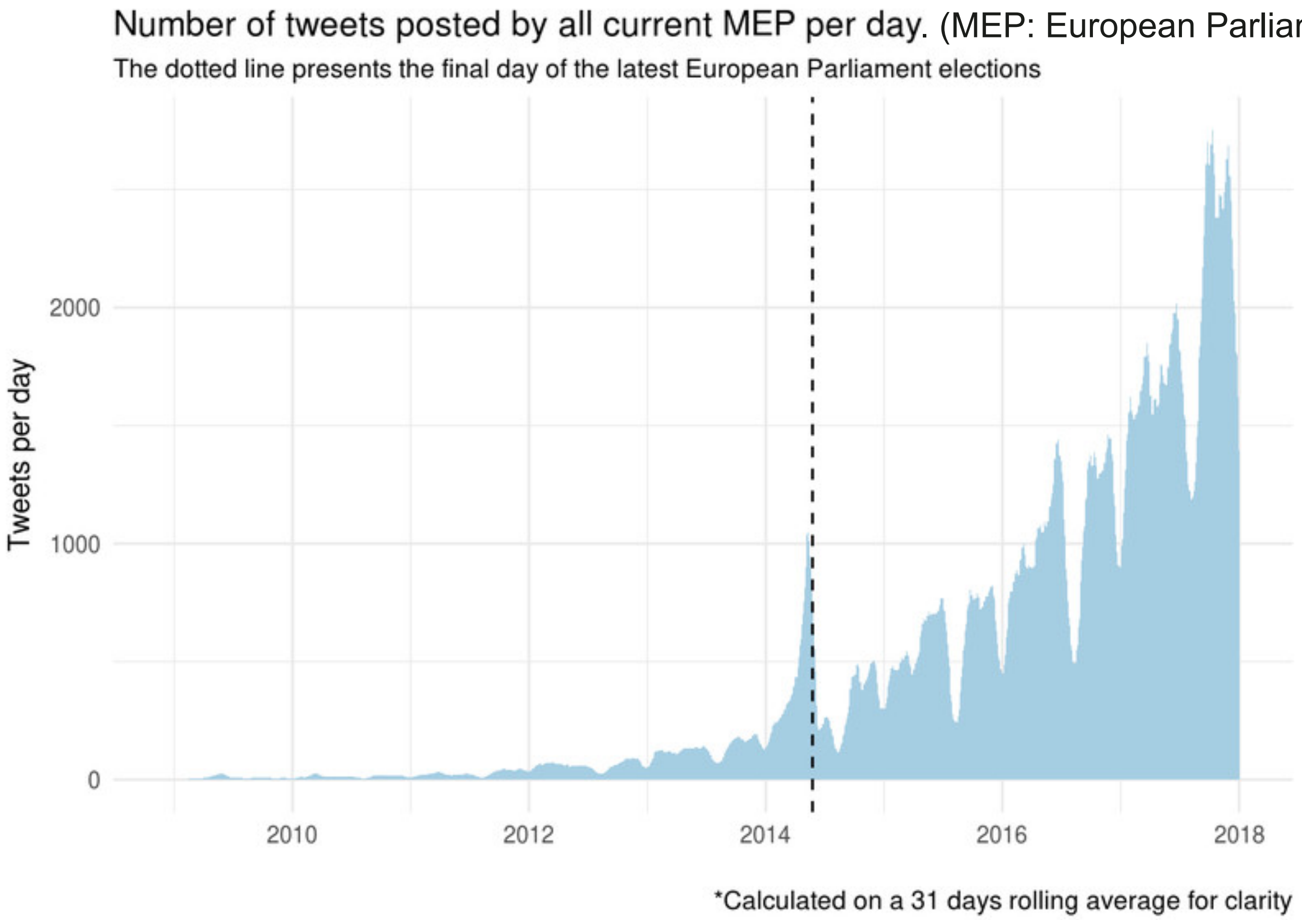


Challenges: heterogeneity & large scale



- Difficult to jointly assess node proximity from the heterogeneous information
  - Distinct modalities: topological structures & node attributes
- Number of nodes and dimension of attributes could be large
  - It could be expensive to store or manipulate the high-dimensional matrices such as node attribute similarity

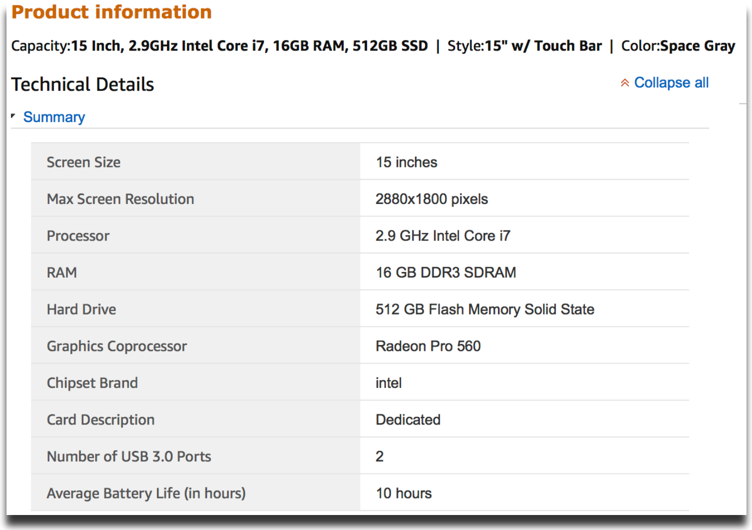
Real-world attributes are high-dimensional



(MEP: European Parliament)

https://euobserver.c

Data characteristics vary significantly



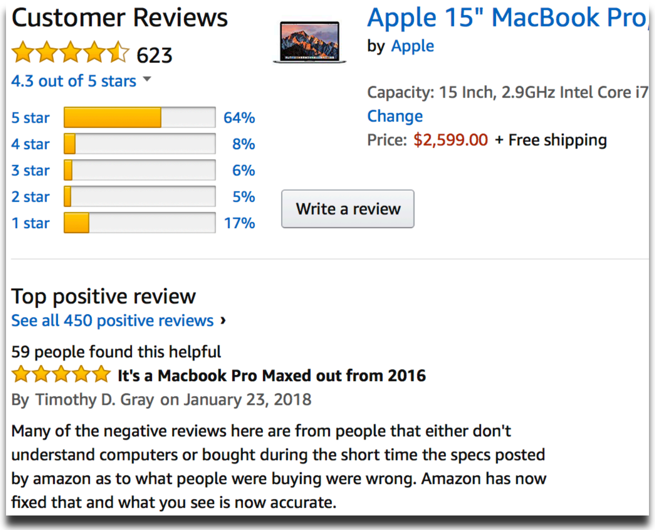
Product information

Capacity:**15 Inch, 2.9GHz Intel Core i7, 16GB RAM, 512GB SSD** | Style:**15" w/ Touch Bar** | Color:**Space Gray**

Technical Details — Collapse all

Summary

Screen Size	15 inches
Max Screen Resolution	2880x1800 pixels
Processor	2.9 GHz Intel Core i7
RAM	16 GB DDR3 SDRAM
Hard Drive	512 GB Flash Memory Solid State
Graphics Coprocessor	Radeon Pro 560
Chipset Brand	intel
Card Description	Dedicated
Number of USB 3.0 Ports	2
Average Battery Life (in hours)	10 hours



- Different types of useful heterogeneous info, such as multiple networks, multiple types of node attributes, & labels
  - Facebook: attributes in introduction, words in posts, content in photos, predefined groups etc.
  - Amazon: product info, customer reviews, customer purchasing records, customer viewing history, etc.

Attributed network embedding

- Motivations & challenges

- **Mining attributed networks with shallow embedding**
  - Coupled spectral embedding
  - Coupled matrix & tri-factorization
  - Random walk based embedding

- Mining attributed networks with deep embedding

- Human-centric network analysis

Coupled spectral embedding

- Spectral embedding on plain networks:

$$\underset{\mathbf{U}}{\text{minimize}} \qquad \frac{1}{2}\sum_{i,j=1}^{n} g_{ij}\|\frac{\mathbf{u}_i}{\sqrt{d_i}} - \frac{\mathbf{u}_j}{\sqrt{d_j}}\|_2^2 = \text{Trace}[\mathbf{U}^\top(\boxed{\mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{G}\mathbf{D}^{-\frac{1}{2}}})\mathbf{U}]$$

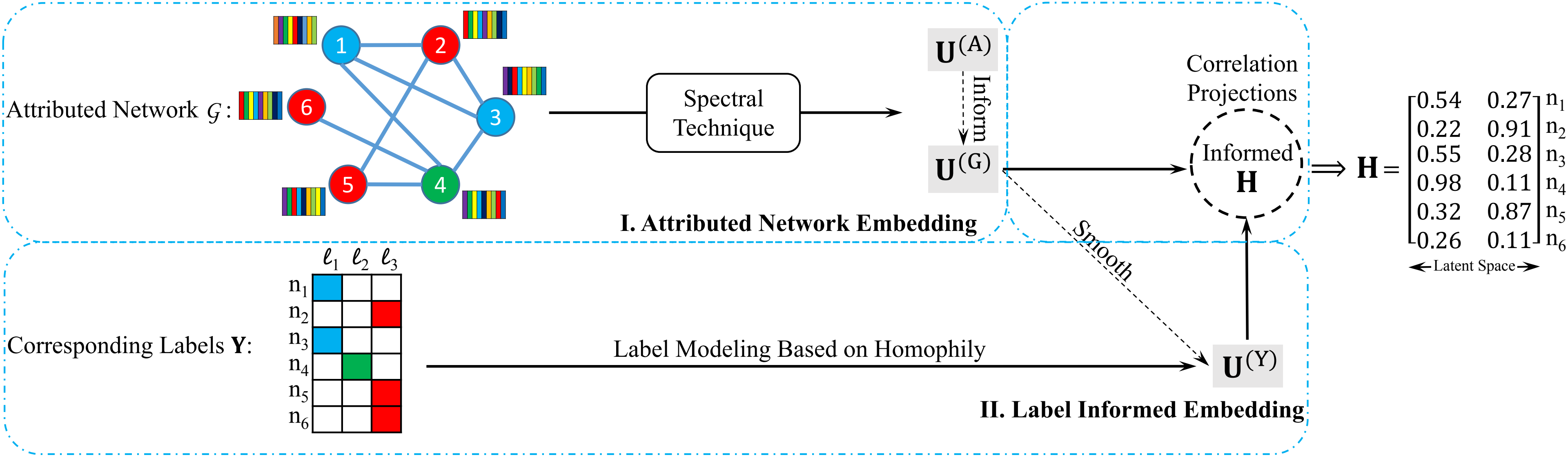
Normalized Graph Laplacian

- For each pair of nodes i and j, larger $\boldsymbol{g}_{ij}$ tends to make their vector representations more similar

- Spectral Graph Theory: Eigenvalues are strongly connected to almost all key invariants of a graph

- How to extend spectral embedding to attributed networks?
    - Challenges: Heterogeneity & Large Scale

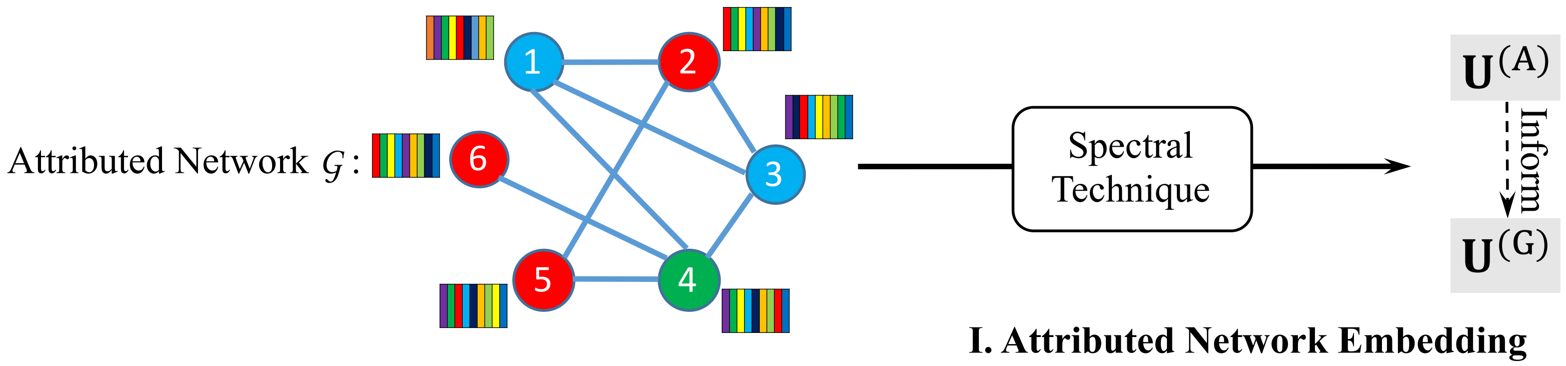
Label informed attributed network embedding



LANE [Huang et al. WSDM, 2017]

- Goal: embed nodes with similar network structure, attribute proximity, or same label into similar vector representations

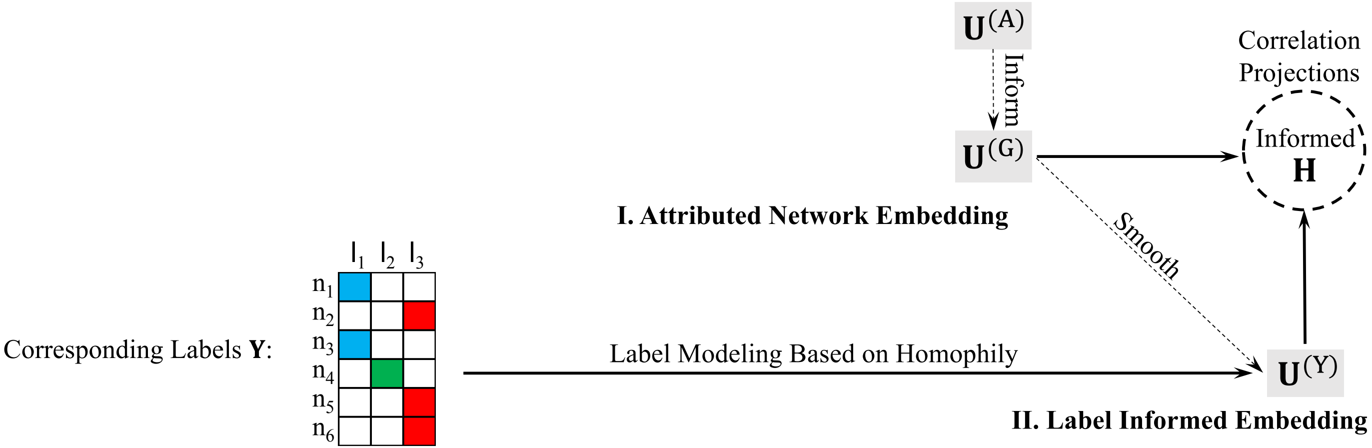
Couple embedding via correlation projection



- Though network $\mathbf{G}$, node attributes $\mathbf{A}$, labels $\mathbf{Y}$ are heterogeneous, node proximities defined by $\mathbf{G}$, $\mathbf{A}$, $\mathbf{Y}$ are homogeneous
- We map the node proximities in network and node attributes into two latent representations $\mathbf{U}^{(G)}$ and $\mathbf{U}^{(A)}$ via spectral embedding and fuse them by extracting their correlations

$$\underset{\mathbf{U}^{(G)},\mathbf{U}^{(A)}}{\text{maximize}} \ \mathrm{Tr}(\mathbf{U}^{(G)\top}\mathcal{L}^{(G)}\mathbf{U}^{(G)} + \alpha\mathbf{U}^{(A)\top}\mathcal{L}^{(A)}\mathbf{U}^{(A)} + \alpha\mathbf{U}^{(A)\top}\mathbf{U}^{(G)}\mathbf{U}^{(G)\top}\mathbf{U}^{(A)})$$

Uniform projections



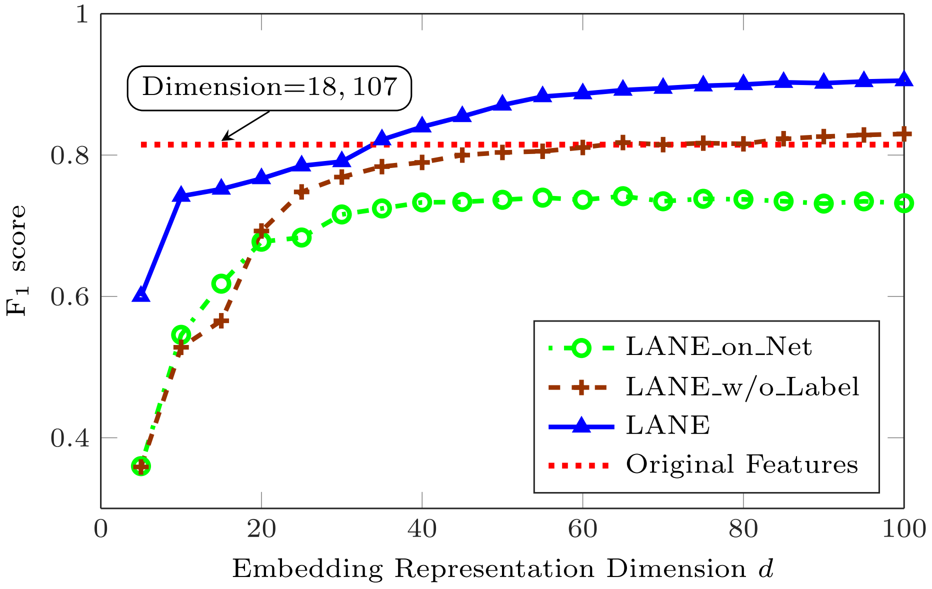
- Consider nodes with the same label as a clique, and employ the learned network proximity to smooth the label information

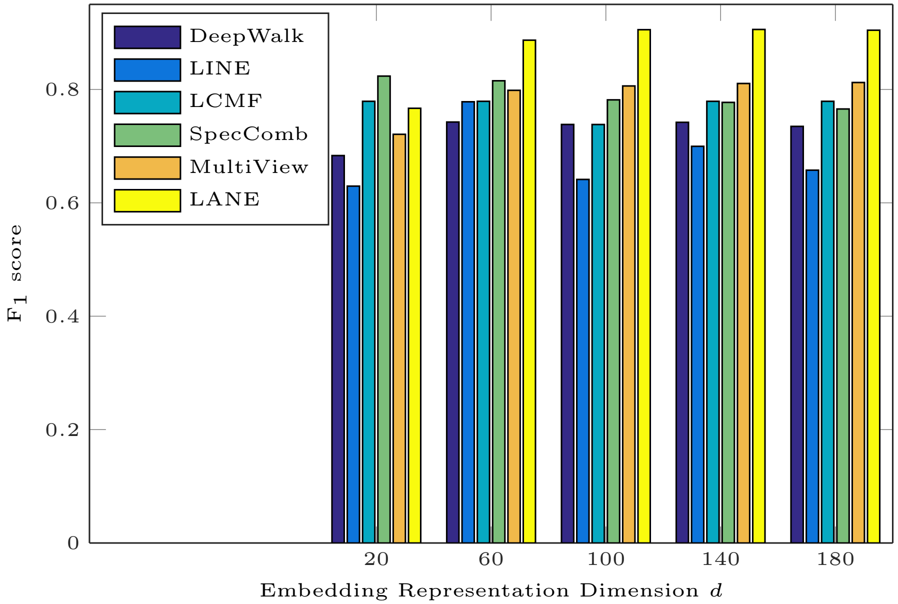
$$\underset{\mathbf{U}^{(G)},\mathbf{U}^{(Y)}}{\text{maximize}}\ \mathrm{Tr}\left(\mathbf{U}^{(Y)\top}(\mathcal{L}^{(YY)}+\mathbf{U}^{(G)}\mathbf{U}^{(G)\top})\mathbf{U}^{(Y)}\right)$$

- Uniformly project all of the learned latent representations into $\mathbf{H}$

$$\underset{\mathbf{U}^{(G)},\mathbf{U}^{(A)},\mathbf{U}^{(Y)},\mathbf{H}}{\text{maximize}}\ \mathrm{Tr}\left(\mathbf{H}^\top(\mathbf{U}^{(G)}\mathbf{U}^{(G)\top}+\mathbf{U}^{(A)}\mathbf{U}^{(A)\top}+\mathbf{U}^{(Y)}\mathbf{U}^{(Y)\top})\mathbf{H}\right)$$

Experimental results





- LANE and its variation outperform Original Features
- LANE achieves significantly better performance than the state-of-the-art embedding algorithms

Summary of coupled spectral embedding

I. Convert node attributes into a network by computing the affinity matrix and couple multiple spectral embedding
   - Label informed attributed network embedding, WSDM 2017
   - Co-regularized multi-view spectral clustering, NIPS 2011

$$\underset{\mathbf{U}^{(G)},\mathbf{U}^{(A)}}{\text{maximize}} \ \mathrm{Tr}({\mathbf{U}^{(G)}}^\top \mathcal{L}^{(G)}\mathbf{U}^{(G)} + \alpha {\mathbf{U}^{(A)}}^\top \mathcal{L}^{(A)}\mathbf{U}^{(A)} + \alpha {\mathbf{U}^{(A)}}^\top \mathbf{U}^{(G)}{\mathbf{U}^{(G)}}^\top \mathbf{U}^{(A)})$$

   - ANE for learning in a dynamic environment, CIKM 2017
     - Initialization:

$$\underset{\mathbf{p},\mathbf{q}}{\text{maximize}} \ \mathbf{p}^\top {\mathbf{U}^{(G)}}^\top \mathbf{U}^{(G)}\mathbf{p} + \mathbf{p}^\top {\mathbf{U}^{(G)}}^\top \mathbf{U}^{(A)}\mathbf{q} + \mathbf{q}^\top {\mathbf{U}^{(A)}}^\top \mathbf{U}^{(G)}\mathbf{p} + \mathbf{q}^\top {\mathbf{U}^{(A)}}^\top \mathbf{U}^{(A)}\mathbf{q}$$

     - Joint representations:

$$\mathbf{H} = [\mathbf{U}^{(G)}, \mathbf{U}^{(A)}] \times [\mathbf{P}, \mathbf{Q}]$$

Summary of coupled spectral embedding

II. Leverage spectral embedding to handle networks and couple with other low-rank approximations, including matrix factorization

- Exploring context and content links in social media, TPAMI 2012

$$\underset{\mathbf{H}}{\text{minimize}} \quad \|\mathbf{A} - \mathbf{H}\|_{\mathrm{F}}^2 + \lambda \mathrm{Trace}[\mathbf{H}^\top (\mathbf{D} - \mathbf{G})\mathbf{H}] + \gamma \|\mathbf{H}\|_*$$

- Attributed signed network embedding, CIKM 2017
  - Use spectral embedding to encode node attribute affinity matrix

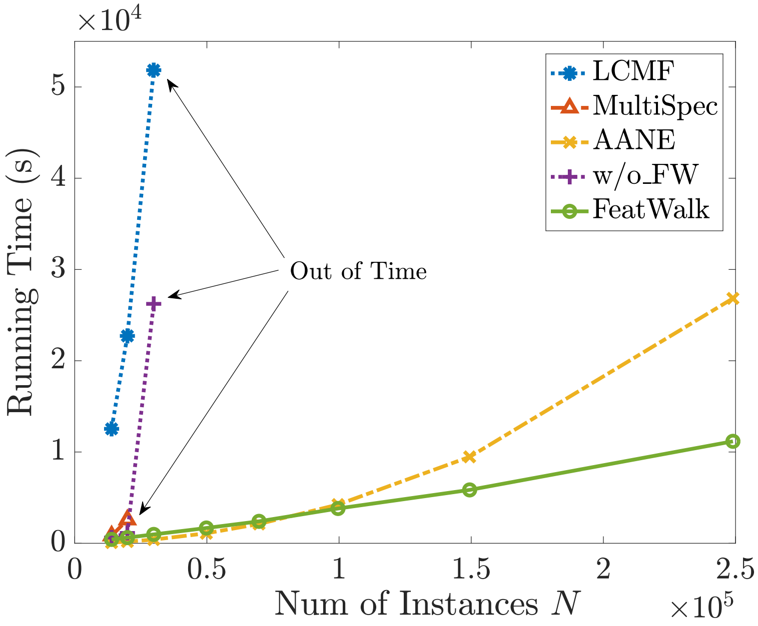
III. Spectral filters in graph neural networks

- Eigenvalues & Eigenvectors are identified as the frequencies of graph & graph Fourier modes
- CNN on graphs with fast localized spectral filtering, NIPS 2016
- Semi-supervised classification with graph convolutional networks, 2016
- GCN networks with complex rational spectral filters, 2019

Coupled matrix & tri- factorization

- Learning a unified representation from two matrices is trivial

$$\min_{\mathbf{H},\mathbf{U},\mathbf{V}} \quad \|\mathbf{G} - \mathbf{H}\mathbf{U}\|_{\mathrm{F}}^2 + \alpha\|\mathbf{A} - \mathbf{H}\mathbf{V}\|_{\mathrm{F}}^2$$

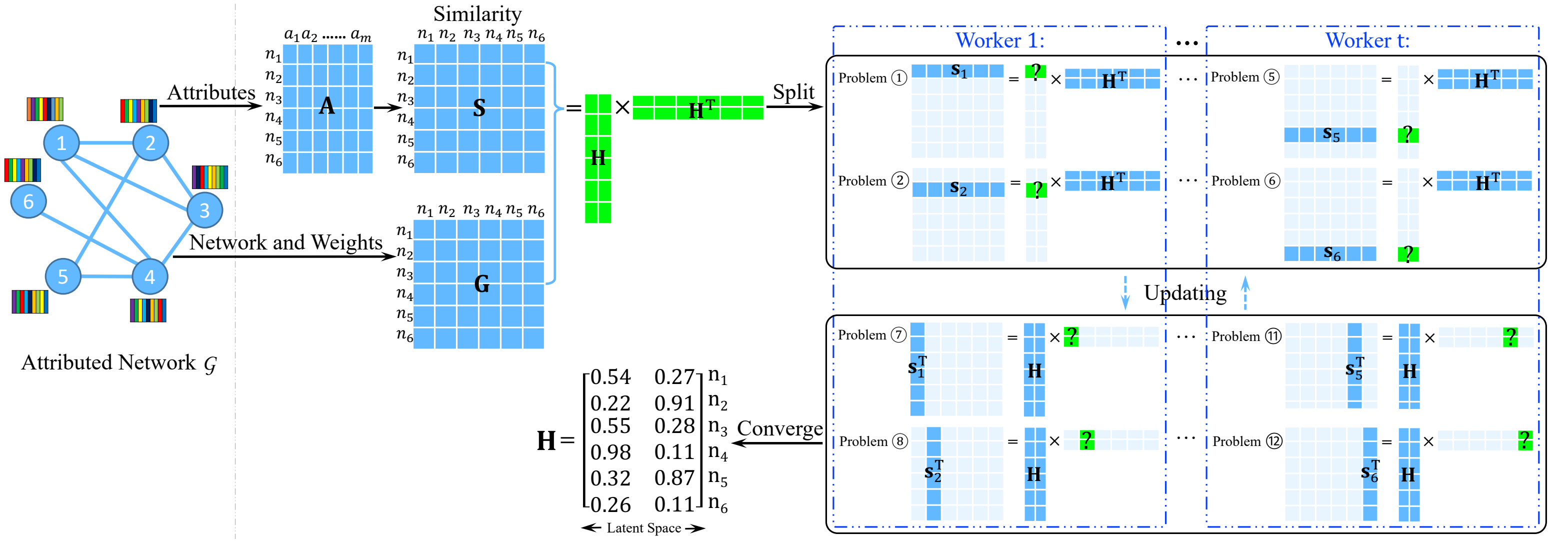


- Intuitive solutions:
  - Combining Content and Link for Classification using Matrix Factorization, 2007 (LCMF)

$$\min_{\mathbf{H},\mathbf{U},\mathbf{V}} \|\mathbf{G} - \mathbf{H}\mathbf{U}\mathbf{H}^\top\|_{\mathrm{F}}^2 + \alpha\|\mathbf{A} - \mathbf{H}\mathbf{V}\|_{\mathrm{F}}^2 + \gamma\|\mathbf{U}\|_{\mathrm{F}}^2 + \beta\|\mathbf{V}\|_{\mathrm{F}}^2$$

- Focuses:
  - Factorizing networks
  - Improving efficiency

Accelerated attributed network embedding



$$\mathbf{H} = \begin{bmatrix} 0.54 & 0.27 \\ 0.22 & 0.91 \\ 0.55 & 0.28 \\ 0.98 & 0.11 \\ 0.32 & 0.87 \\ 0.26 & 0.11 \end{bmatrix} \begin{array}{l} n_1 \\ n_2 \\ n_3 \\ n_4 \\ n_5 \\ n_6 \end{array}$$

AANE [Huang et al. SDM, 2017]

- Goal: Preserve the network & node attributes into a unified latent representation, in an efficient way

- AANE accelerates the optimization by decomposing it into low complexity sub-problems

Network structure modeling

- Objective function: $\min_{\mathbf{H}} \quad \mathcal{J} = \|\mathbf{S} - \mathbf{H}\mathbf{H}^\top\|_{\mathrm{F}}^2 + \boxed{\lambda \sum_{(i,j)\in\mathcal{E}} g_{ij}\|\mathbf{h}_i - \mathbf{h}_j\|_2},$

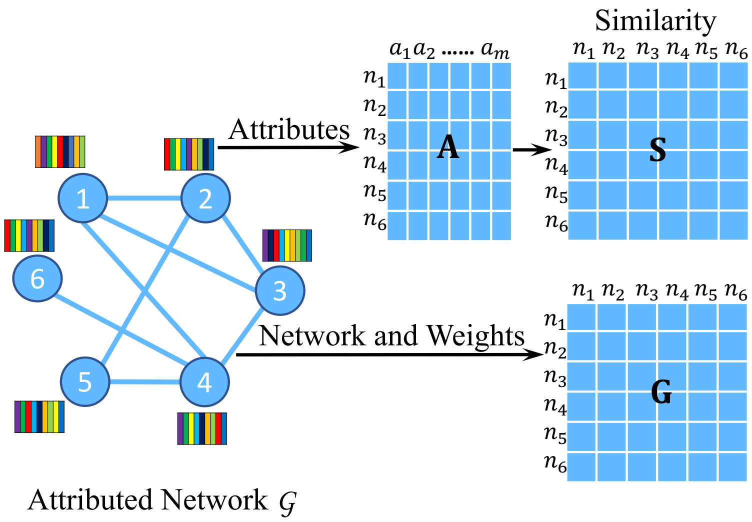
  Network Lasso

- Network lasso [Hallac et al. KDD, 2015]:
  - If we use squared norms, it would reduce to Laplacian regularization
  - A generalization of group lasso, encouraging $h_i = h_j$ across the edge
  - For each edge i to j, set $\{(h_{i1} - h_{j1}),\ (h_{i2} - h_{j2}), \dots\}$ as a group
  - Group lasso: $\min_{\boldsymbol{\beta}} \quad \|\mathbf{y} - \mathbf{X}\beta\|_2^2 + \lambda \sum_{\mathcal{I}=1,\dots,I} \|\boldsymbol{\beta}_{\mathcal{I}}\|_2$

- λ adjusts the size of clustering groups

- ℓ_2-norm alleviates the impacts from outliers and missing data

Incorporating node attribute affinities



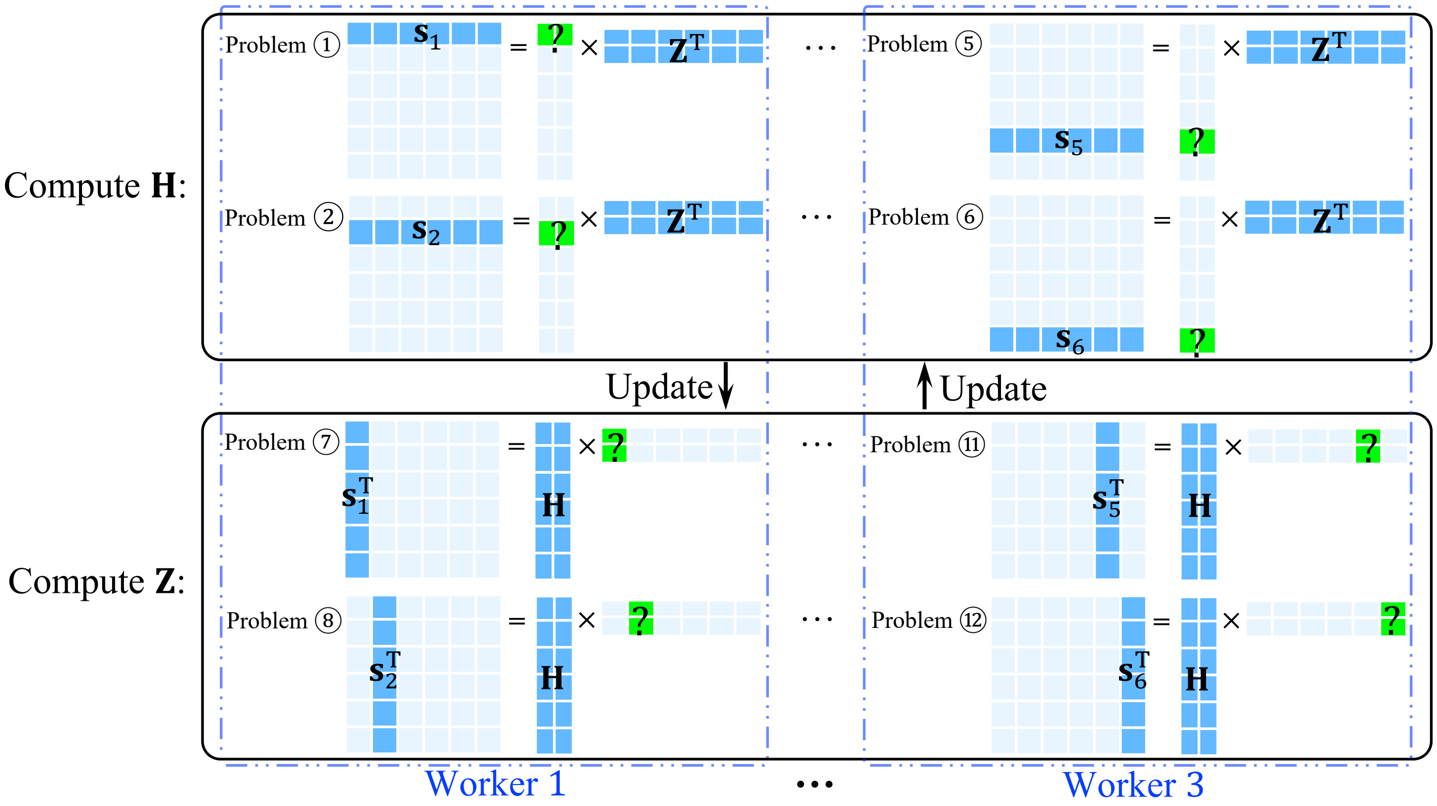
Objective functions:

$$\min_{\mathbf{H}} \quad \mathcal{J} = \|\mathbf{S} - \mathbf{H}\mathbf{H}^\top\|_F^2 + \boxed{\lambda \sum_{(i,j)\in\mathcal{E}} g_{ij}\|\mathbf{h}_i - \mathbf{h}_j\|_2}$$

Network Lasso

- Though network & node attributes are heterogeneous info, node proximity defined by attributes is homogenous with network
- Based on the decomposition of similarities defined by attributes and penalty of embedding difference between connected nodes

Acceleration via distributed optimization



- Make sub-problems independent to each other to allow parallel computation

Low-complexity independent sub-problems

- Make a copy of $\mathbf{H}$, named $\mathbf{Z}$
- Reformulate objective function into a linearly constrained problem

$$\begin{aligned} \min_{\mathbf{H}} \quad & \sum_{i=1}^{n} \|\mathbf{s}_i - \mathbf{h}_i \mathbf{Z}^\top\|_2^2 + \lambda \sum_{(i,j)\in\mathcal{E}} g_{ij} \|\mathbf{h}_i - \mathbf{z}_j\|_2, \\ \text{subject to} \quad & \mathbf{h}_i = \mathbf{z}_i, \ i = 1, \ldots, n. \end{aligned}$$

- Given fixed $\mathbf{H}$, all the row $\mathbf{z}_i$ could be calculated independently
- Each sub-problem only needs row $\mathbf{s}_i$, not the entire $\boldsymbol{S}$
- Time complexity of updating $\mathbf{h}_i$ is $\mathcal{O}(d^3 + dn + d|N(i)|)$, with space complexity $\mathcal{O}(n)$

Summary of coupled matrix & tri- factorization

I. Accelerate coupled matrix factorization via distributed optimizations
   - Accelerated attributed network embedding, SDM 2017
   - Accelerated local anomaly detection via resolving AN, IJCAI 2017
     - $\min_{\mathbf{H},\mathbf{V}} \quad \|\mathbf{G}-\mathbf{H}\mathbf{H}^\top\|_\mathrm{F}^2+\alpha\|\mathbf{A}-\mathbf{H}\mathbf{V}\|_\mathrm{F}^2+\gamma(\|\mathbf{H}\|_\mathrm{F}^2+\|\mathbf{V}\|_\mathrm{F}^2)$
     - A parallel mini-batch SGD to accelerate the optimization

$$\|\mathbf{G}-\mathbf{H}\mathbf{H}^\top\|_\mathrm{F}^2$$

$$\begin{array}{cc} & \begin{bmatrix} \boldsymbol{h}_1^\top & \boldsymbol{h}_2^\top & \boldsymbol{h}_3^\top & \boldsymbol{h}_4^\top & \boldsymbol{h}_5^\top & \cdots & \boldsymbol{h}_n^\top \end{bmatrix} \\ \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \boldsymbol{h}_3 \\ \boldsymbol{h}_4 \\ \boldsymbol{h}_5 \\ \vdots \\ \boldsymbol{h}_\mathrm{n} \end{bmatrix} & \begin{bmatrix} g_{1,1} & g_{1,2} & g_{1,3} & g_{1,4} & g_{1,5} & \cdots & g_{1,n} \\ g_{2,1} & g_{2,2} & g_{2,3} & g_{2,4} & g_{2,5} & \cdots & g_{2,n} \\ g_{3,1} & g_{3,2} & g_{3,3} & g_{3,4} & g_{3,5} & \cdots & g_{3,n} \\ g_{4,1} & g_{4,2} & g_{4,3} & g_{4,4} & g_{4,5} & \cdots & g_{4,n} \\ g_{5,1} & g_{5,2} & g_{5,3} & g_{5,4} & g_{5,5} & \cdots & g_{5,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\ g_{\mathrm{n},1} & g_{\mathrm{n},2} & g_{\mathrm{n},3} & g_{\mathrm{n},4} & g_{\mathrm{n},5} & \cdots & g_{\mathrm{n},n} \end{bmatrix} \end{array}$$

$$\|\mathbf{A}-\mathbf{H}\mathbf{V}\|_\mathrm{F}^2$$

$$\begin{array}{cc} & \begin{bmatrix} \boldsymbol{v}_1 & \boldsymbol{v}_2 & \boldsymbol{v}_3 & \boldsymbol{v}_4 & \boldsymbol{v}_5 & \cdots & \boldsymbol{v}_m \end{bmatrix} \\ \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \boldsymbol{h}_3 \\ \boldsymbol{h}_4 \\ \boldsymbol{h}_5 \\ \vdots \\ \boldsymbol{h}_\mathrm{n} \end{bmatrix} & \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & a_{1,5} & \cdots & a_{1,m} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} & \cdots & a_{2,m} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & \cdots & a_{3,m} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} & \cdots & a_{4,m} \\ a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} & a_{5,5} & \cdots & a_{5,m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\ a_{\mathrm{n},1} & a_{\mathrm{n},2} & a_{\mathrm{n},3} & a_{\mathrm{n},4} & a_{\mathrm{n},5} & \cdots & a_{\mathrm{n},m} \end{bmatrix} \end{array}$$

Summary of coupled matrix & tri- factorization

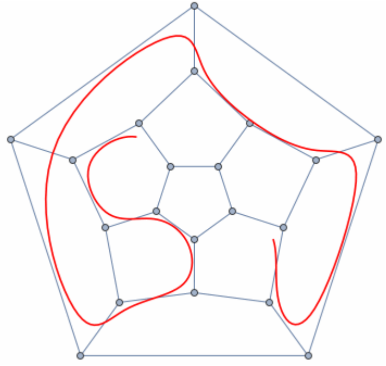
II. Modeling networks via matrix tri-factorization

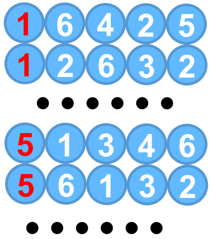
- Network Representation Learning with Rich Text Information, IJCAI 2015
  - Let $\mathbf{T}$ be the transition matrix of the PageRank on $\mathbf{G}$, and $\mathbf{M} = (\mathbf{T} + \mathbf{T}^2)/2$
  - $\min_{\mathbf{H},\mathbf{V}} \quad \|\mathbf{M} - \mathbf{H}\mathbf{V}\mathbf{A}^\top\|_{\mathrm{F}}^2 + \frac{\lambda}{2}(\|\mathbf{H}\|_{\mathrm{F}}^2 + \|\mathbf{V}\|_{\mathrm{F}}^2)$
- Preserving Proximity and Global Ranking for Network Embedding, 2017
  - Lemma: Matrix tri-factorization $\mathbf{H}^\top\mathbf{V}\mathbf{H} \approx \mathbf{M}^{\mathrm{PMI}}$ preserves the second-order proximity, where (shifted) pointwise mutual information is defined as follows

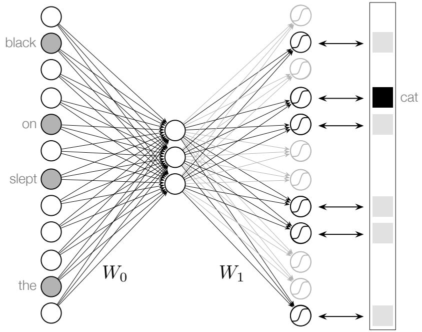
$$\mathbf{M}^{\mathrm{PMI}} = \begin{cases} \max\{0, \log \frac{p_{s,t}(i,j)}{p_s(i)p_t(j)} - \log\alpha\}, & \text{if } (i,j) \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases}$$

  - $p_{s,t}(i,j) = \frac{1}{|\mathcal{E}|}$, $p_s(i) = \frac{\mathrm{degree}_{\mathrm{out}}^i}{|\mathcal{E}|}$, $p_t(j) = \frac{\mathrm{degree}_{\mathrm{in}}^j}{|\mathcal{E}|}$
  - Negative values are filtered since less informative [Levy and Goldberg, 2014]

Random walk based embedding





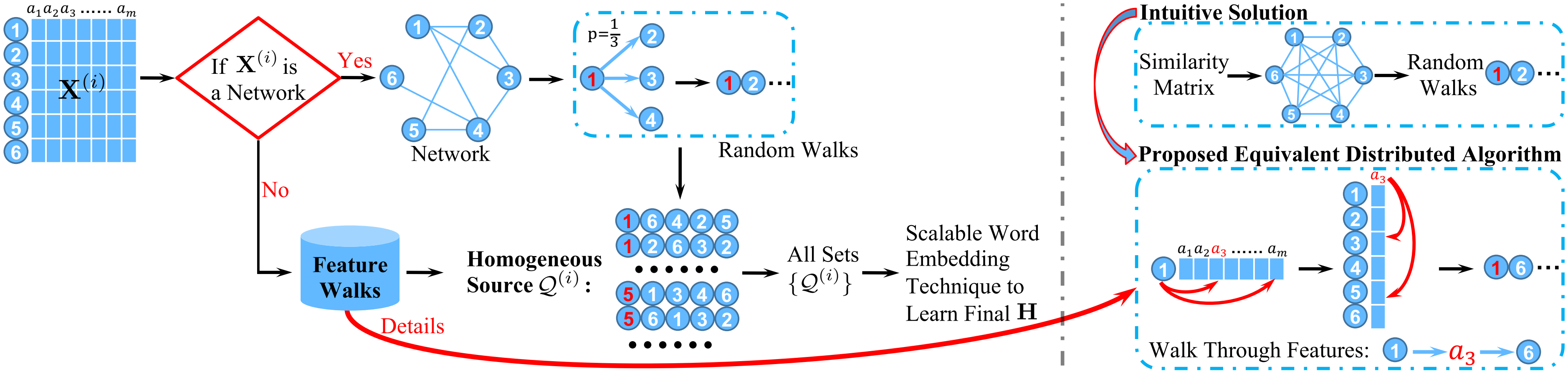


$$\mathbf{H} = \begin{bmatrix} 0.54 & 0.27 \\ 0.22 & 0.91 \\ 0.55 & 0.28 \\ 0.98 & 0.11 \\ 0.32 & 0.87 \\ 0.26 & 0.11 \end{bmatrix}$$

- Random walks on plain networks:
  - Conduct random walks on a network and record the walking trajectories
  - Treat nodes as words and sequences as sentences to learn embedding
- Nodes' co-occurrence probabilities ≈ linking probabilities
- It converts geometric structures into structured sequences while alleviating the issues of sparsity and curse of dimensionality
- Random walks on attributed networks? (Heterogeneity)

Word2vec Figure from Franziska Horn, "Context Encoders as A Simple But Powerful Extension of Word2vec", 2017.

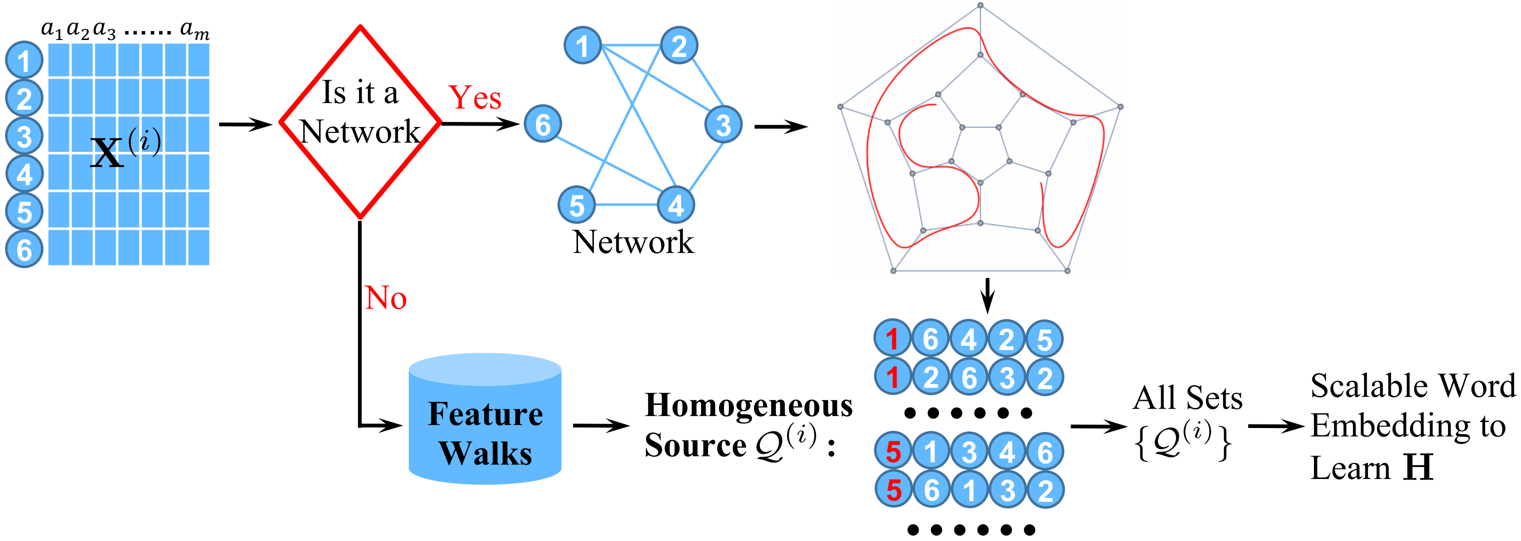
Large-scale heterogeneous feature embedding



FeatWalk [Huang et al. AAAI, 2019]

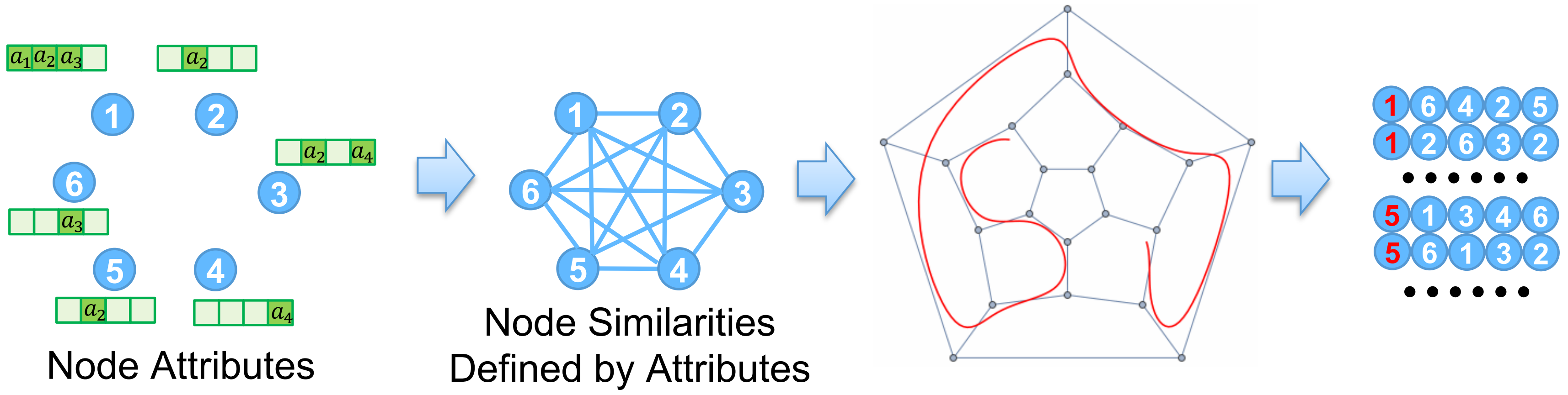
- Goal: Incorporate multiple networks & multiple types of high-dimensional node attributes into a unified latent representation
- E.g., amazon products have product info, customer reviews, etc. Networks: customer purchase record, & customer viewing history

Learn node proximities to handle heterogeneity



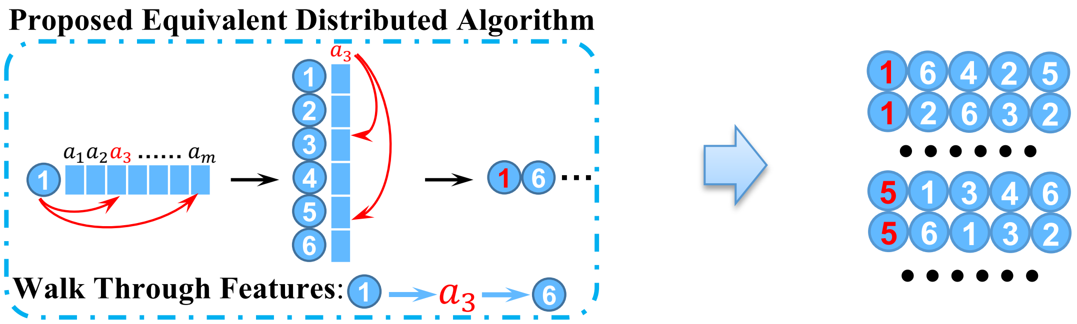
- Node proximity: Similarities between nodes defined by links or attributes of nodes, i.e., rows of each $\mathbf{X}^{(i)}$
- Node proximities learned from different $\{\mathbf{X}^{(i)}\}$ are homogeneous
- FeatWalk projects each node proximity into a set of node sequences $\mathcal{Q}^{(i)}$, and learns $\mathbf{H}$ from all $\{\mathcal{Q}^{(i)}\}$

The intuitive solution



- To learn $\mathcal{Q}^{(i)}$, intuitive solution is to compute node similarity matrix $\boldsymbol{S}$ based on $\mathbf{A}^{(i)}$, and perform random walks on $\boldsymbol{S}$
- Random Walks: In $\mathcal{Q}^{(i)}$, a sequence of node indices, probability of i follows j approaches their similarity in $\boldsymbol{S}$
- Expensive: $\boldsymbol{S}$ is dense with $\mathrm{n} \times n$ dimensions

Equivalent similarity-based random walks



- Theorem 1. Probability of walking from i to j via FeatWalk is equal to the one via random walks on $\boldsymbol{S}$, where

$$\mathbf{S} = \mathbf{Y}\mathbf{D}\mathbf{Y}^\top$$

- $\mathbf{Y}$ is the node attribute matrix after special normalizations
- FeatWalk learns the same sequences as the intuitive solution, while avoiding the computation of node similarities $\boldsymbol{S}$

FeatWalk walks via features

- Given the initial i, we walk to the m^{th} attribute category with probability

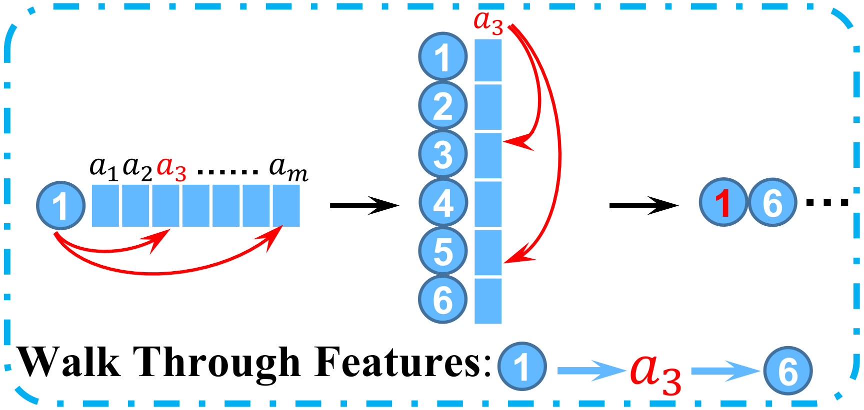
$$P(i \to a_m) = \frac{\hat{x}_{im}}{\sum_{p=1}^{M} \hat{x}_{ip}}$$

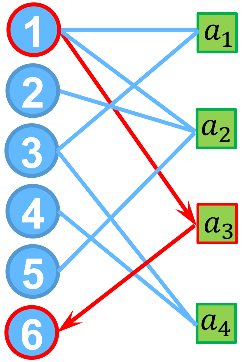
- We focus on the m^{th} attribute category and walk from a_m to j with probability

$$P(a_m \to j) = \frac{y_{jm}}{\sum_{n=1}^{N} y_{nm}}$$

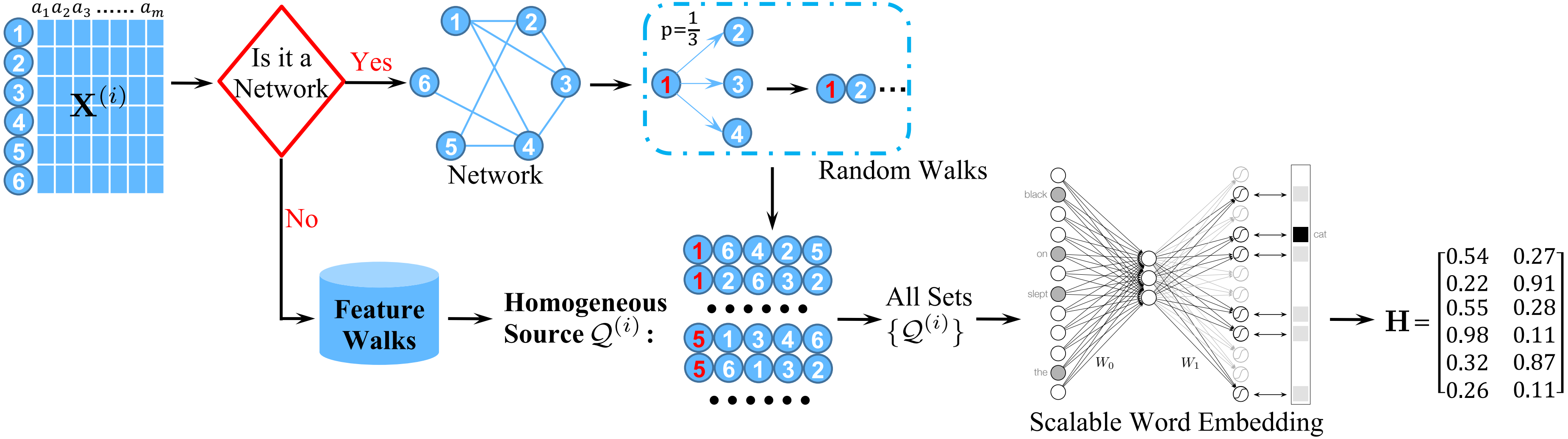
- $\hat{x}_{im}$ and y_{jm} are normalized node attributes

Proposed Equivalent Distributed Algorithm



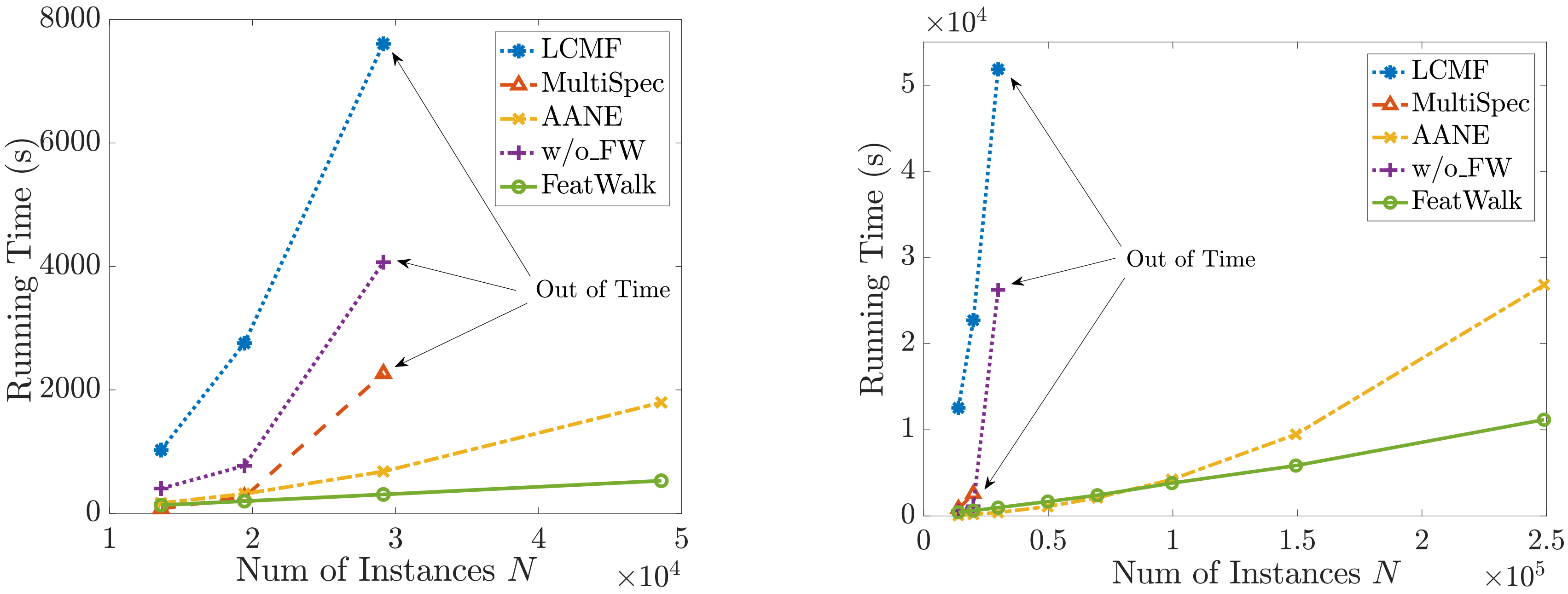


Summary of FeatWalk



- Project each node proximity into a set of node sequence $\mathcal{Q}^{(i)}$
- Consider nodes as words and truncated sequences as sentences
- Apply a scalable word embedding technique to all $\{\mathcal{Q}^{(i)}\}$ to learn a joint embedding representation $\mathbf{H}$

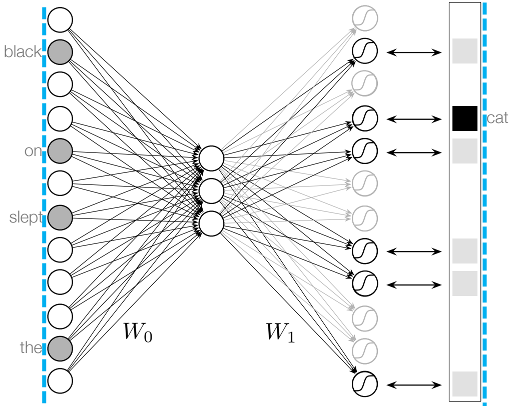
Efficiency evaluation



- Running time of FeatWalk is almost linear to N
- FeatWalk achieves a significant acceleration compared to the intuitive solution w/o_FW

Summary of random walk based embedding

Skip-Gram Model & Negative Sampling



Word2vec: Distributed Representations of Words and Phrases and their Compositionality
DeepWalk: Online Learning of Social Representations
FeatWalk: Large-Scale Heterogeneous Feature Embedding
TriDNR: Tri-Party Deep Network Representation
Gat2vec: Representation Learning for Attributed Graphs

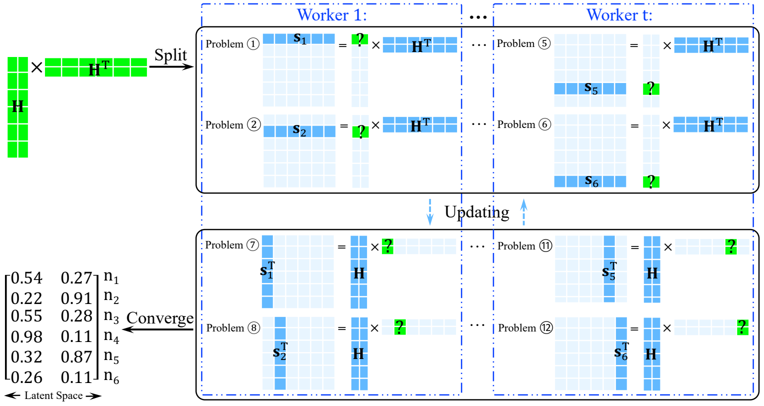
Word2vec:	words	surrounding words	[2013]
DeepWalk:	nodes	neighbors	[2014]
FeatWalk:	nodes	neighbors defined by edges nodes with same attributes	[2019]
TriDNR Gat2vec:	nodes attributes labels	neighbors defined by edges nodes with same attributes nodes with same labels	[2016] [2019]

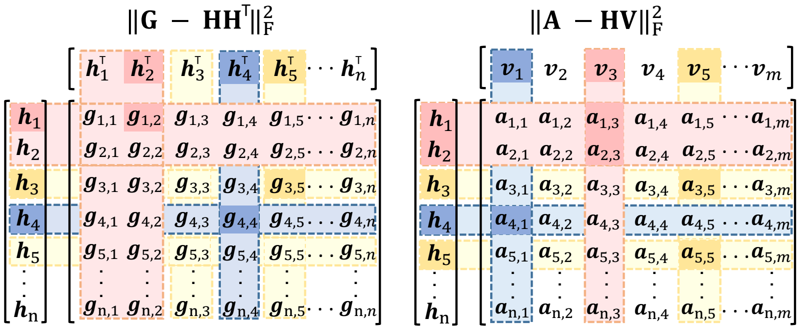
Mining attributed networks with shallow embedding

- Focuses:
  Joint learning, embedding networks, & accelerating optimization

- Methods:
  Coupled spectral embedding
  Coupled matrix & tri-factorization
  Random walk based embedding

- Techniques:
  Spectral graph theory, Coupling, distributed optimization, joint random walks, etc.





Attributed network embedding

- Motivations & challenges
- Mining attributed networks with shallow embedding
- **Mining attributed networks with deep embedding**
  - Objective function based deep embedding
  - Graph neural networks
- Human-centric network analysis

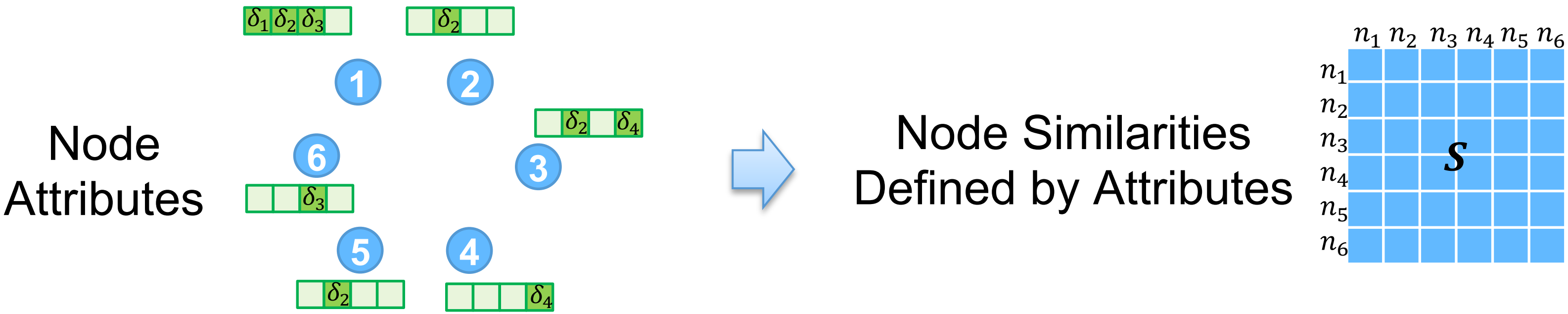
Objective function based deep embedding

- Objective function of DeepWalk:

$$\mathcal{J}_{\text{DeepWalk}} = -\log(\sigma(\mathbf{h}_u^\top \mathbf{h}_v)) - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log(\sigma(-\mathbf{h}_u^\top \mathbf{h}_{v_n}))$$

- v is a node that co-occurs near u on fixed-length random walks
- σ is the sigmoid function. Q is the number of negative samples
- $P_n(v)$ is a negative sampling distribution, based on the node frequencies in the entire node sequences
- It trains a unique embedding representation for each node via a representation look-up table
- How to incorporate node attributes in deep architectures?

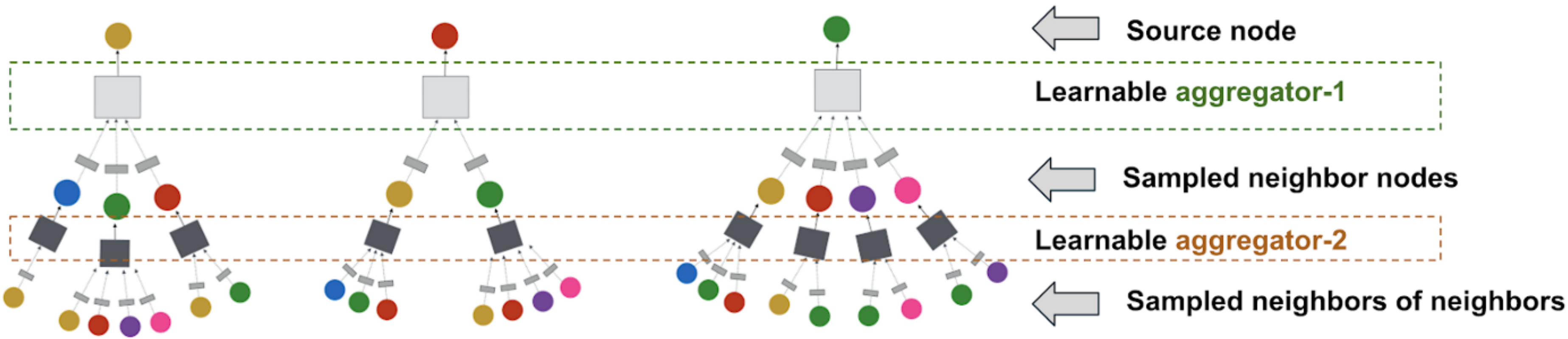
Property preserving network embedding



- Compute the node similarity matrix $\boldsymbol{S}$ defined by node attributes
- Objective function: $\mathcal{J} = \mathcal{J}_{\text{DeepWalk}} + \sum\limits_{i \in \text{pos}(v) \cup \text{neg}(v)} \mathbf{s}_{vi} d(v, i)$
- $\mathbf{s}_{vi}$ is the attribute similarity between u and i
- $d(v,i) = \sqrt{(\mathbf{h}_v - \mathbf{h}_i)^\top (\mathbf{h}_v - \mathbf{h}_i)}$ measures distance in embedding space
- $\text{pos}(v)$ and $\text{neg}(v)$ are sets of top-k similar and dissimilar nodes according to $\boldsymbol{S}$

Li et al., "PPNE: Property Preserving Network Embedding", DASFAA, 2019.

Graph neural networks

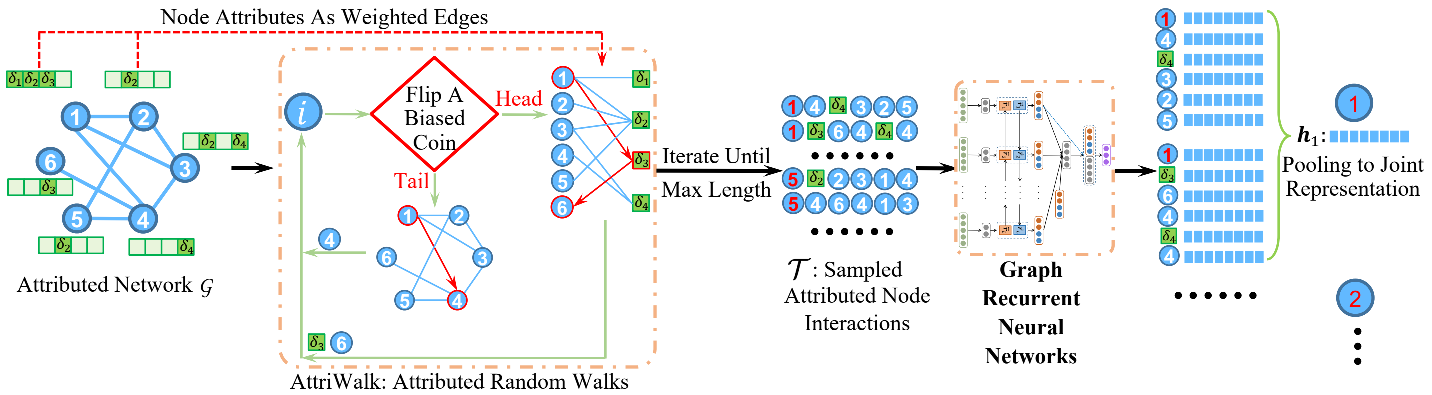


- Key ideas of graph convolutional networks and GraphSage:
  - Use node attributes or random vectors as initial latent representations
  - Each node's representation is learned via averaging its neighbors' representations in previous layer

- It could be considered as a first-order approximation of spectral graph convolutions

Hamilton et al., "Inductive Representation Learning on Large Graphs", NIPS, 2017.

Graph recurrent networks with attributed walks

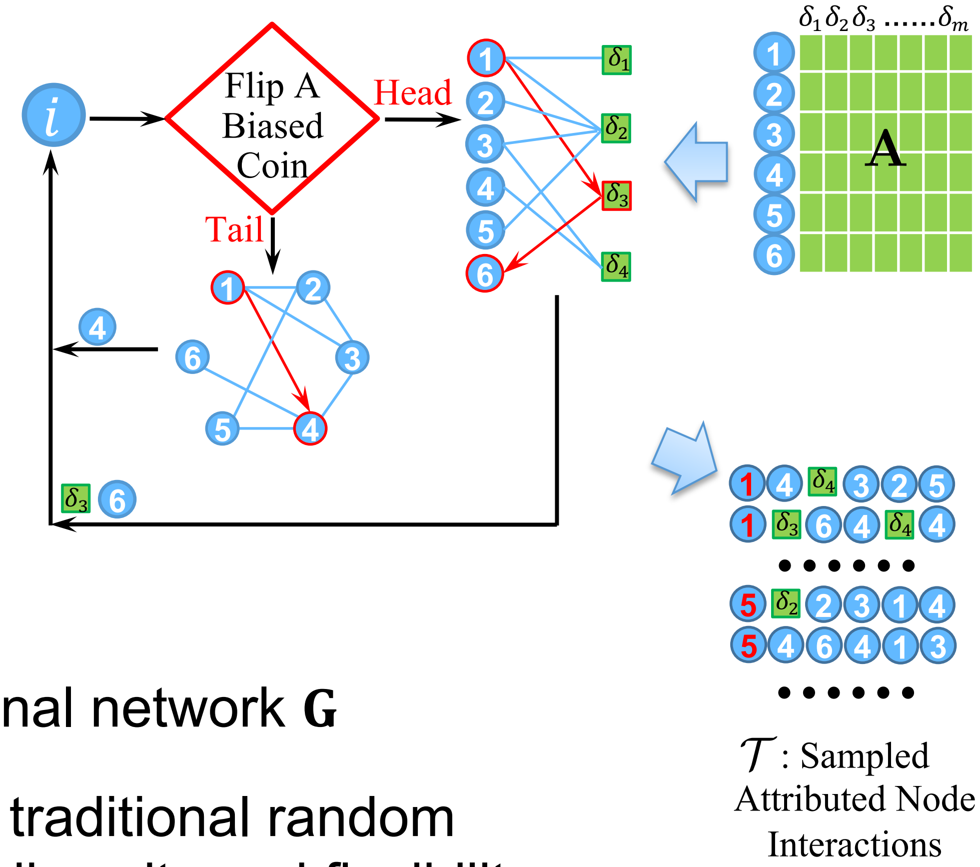


GraphRNA [Huang et al. KDD, 2019]

- A unified walking mechanism is proposed to jointly sample networks and node attributes
- Graph recurrent network (GRN) could preserve node order information
- Nodes are allowed to interact in GRN via the same way as they interact in the original attributed network

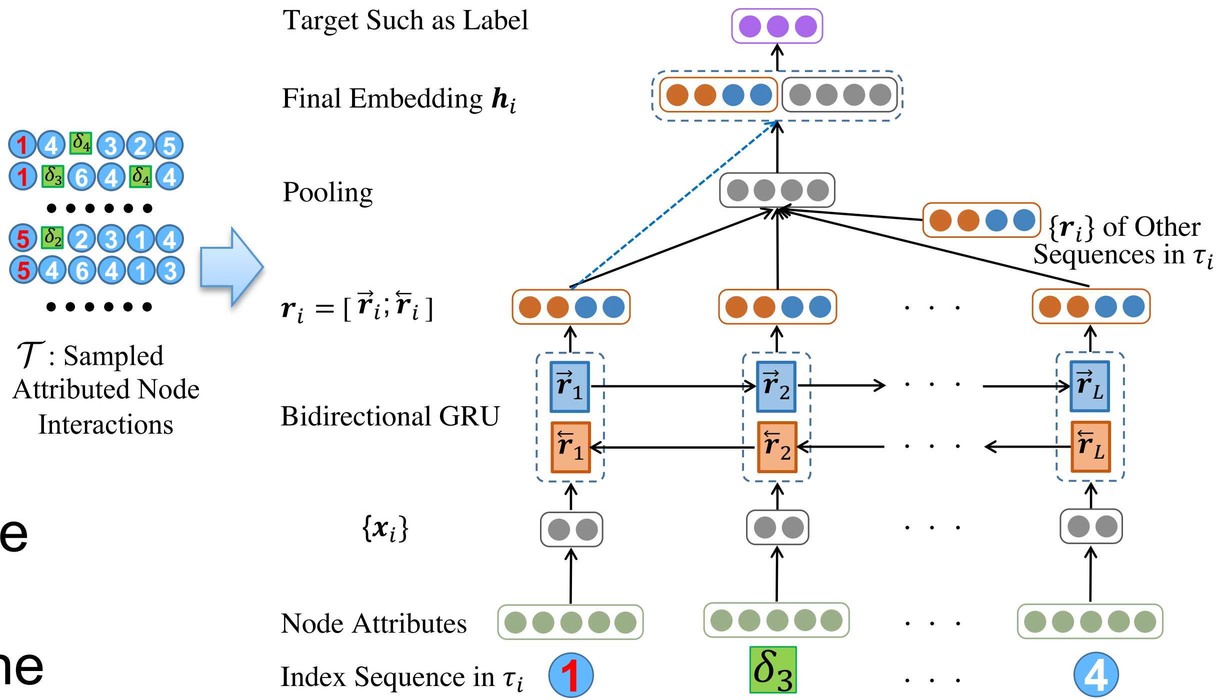
A joint walking mechanism - AttriWalk

- Construct a bipartite network based on $\mathbf{A}$
- Flip a biased coin in each step
- If head, walk two steps on the bipartite network
  - Jump to an attribute category δ_k
  - From δ_k, jump to a node j
- If tail, walk one step on the original network $\mathbf{G}$
- Walks on $\mathbf{G}$ inherit properties of traditional random walks; walks on $\mathbf{A}$ increase the diversity and flexibility



Graph recurrent neural networks - GRN

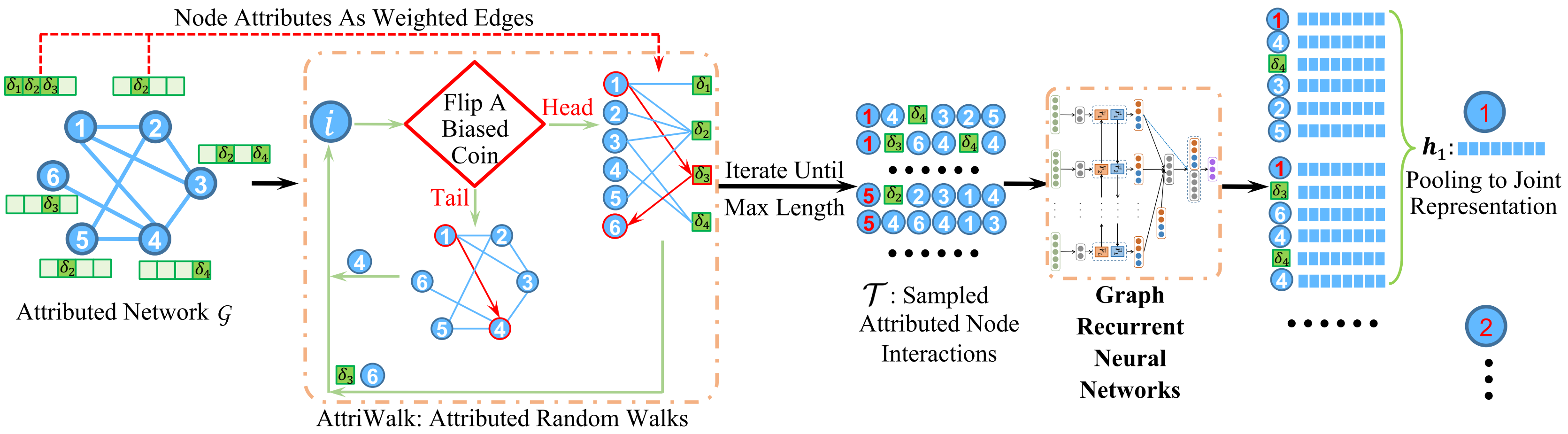
- Hidden state sequences in RNN naturally accord with sampled node interactions



- Pooling layers combine indices within each sequence, and combine all sequences of each node

- It concatenates the first embedding representation for self loop

Task-specific objective function & multiple sources



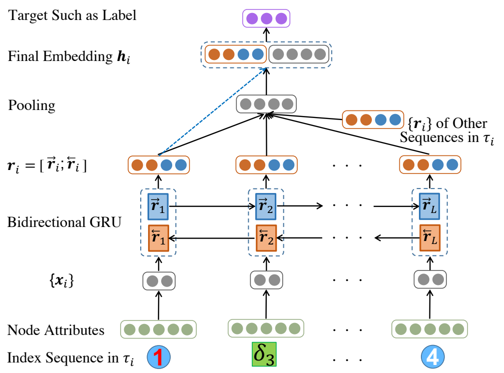
- GraphRNA could be trained with an unsupervised, supervised, or task-specific objective functions, e.g.,

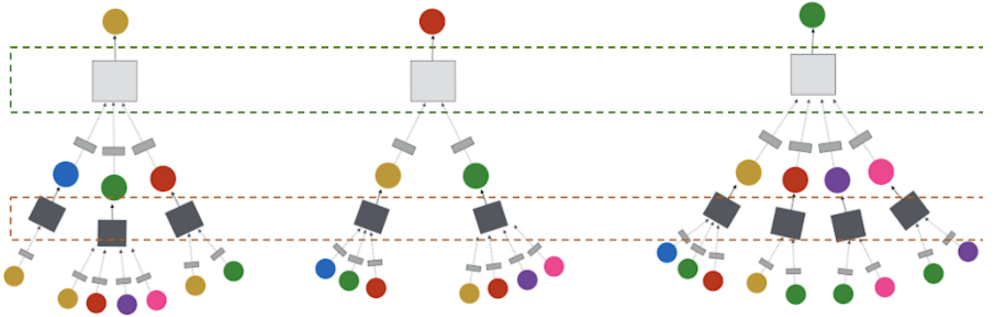
$$\mathcal{L} = -\sum_{i \in \mathcal{V}} \mathbf{y}_i^\top \log(\mathrm{softmax}(\sigma(\mathbf{h}_i \mathbf{W}_h + \mathbf{b}_h)))$$

- Graph neural networks could be an embedding model or an end-to-end model for different tasks

Mining attributed networks with deep embedding

- Focuses:
  Deep architectures for networks & joint learning
- Methods:
  Objective function based deep embedding
  Graph neural networks
- Architectures:
  Graph convolutional networks
  Graph recurrent networks

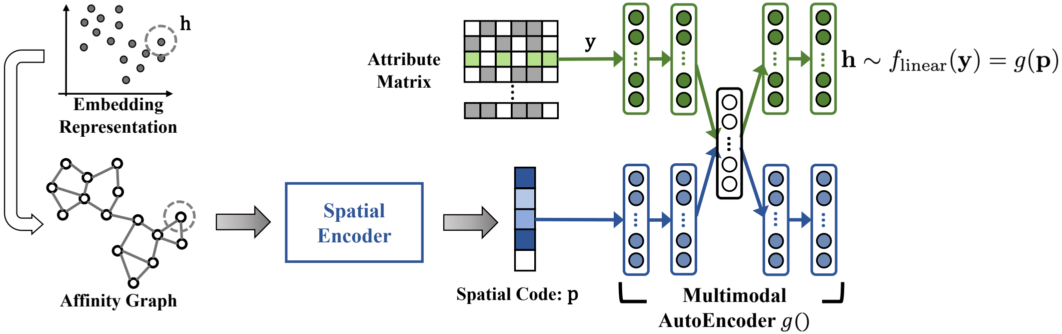




Attributed Network Embedding

- **Motivations & challenges**
- **Mining attributed networks with shallow embedding**
  - Coupled spectral embedding
  - Coupled matrix & tri-factorization
  - Random walk based embedding
- **Mining attributed networks with deep embedding**
  - Objective function based deep embedding
  - Graph neural networks
- **Human-centric network analysis**
  - Interpretable node representation learning
  - Attributed network analysis with humans in the loop

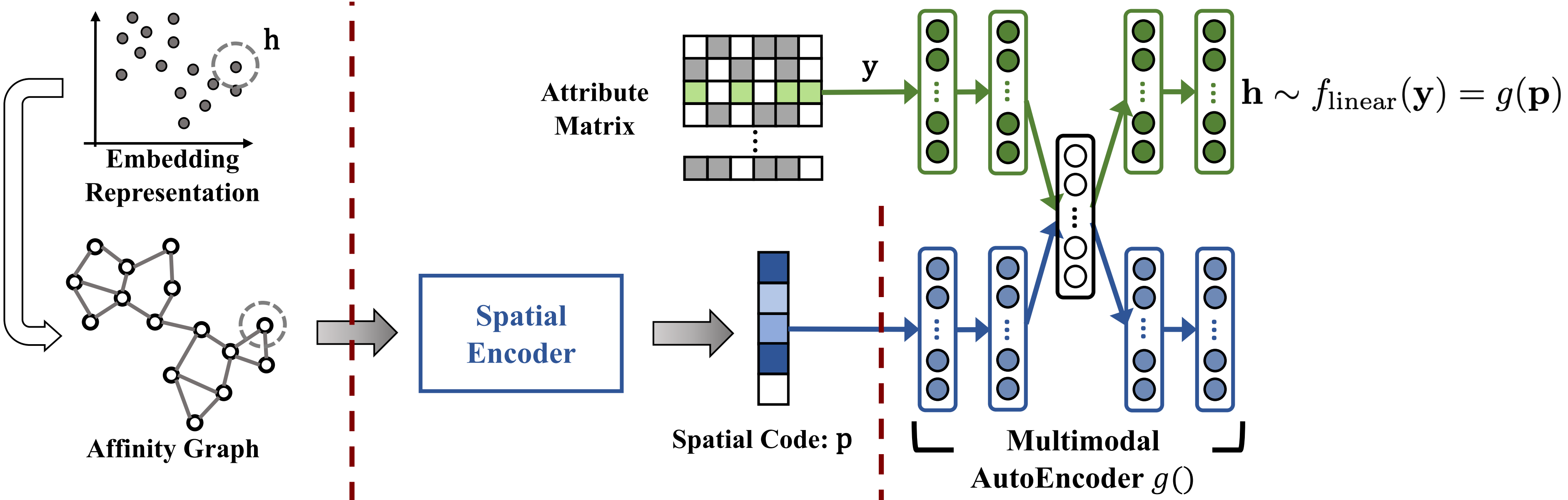
Interpretable node representation learning



[Liu et al. WSDM, 2019]

- Opacity of embedding space
  - How representation vectors distribute in the embedding space?
  - What information is encoded in different embedding space regions?
  - Existing methods for explaining classifiers are not directly applicable
- Comprehensible node attributes are available
- Goal: Mining explainable structures and identifying characteristic factors from the mass of representation vectors

Spatial encoding and multimodal analytics

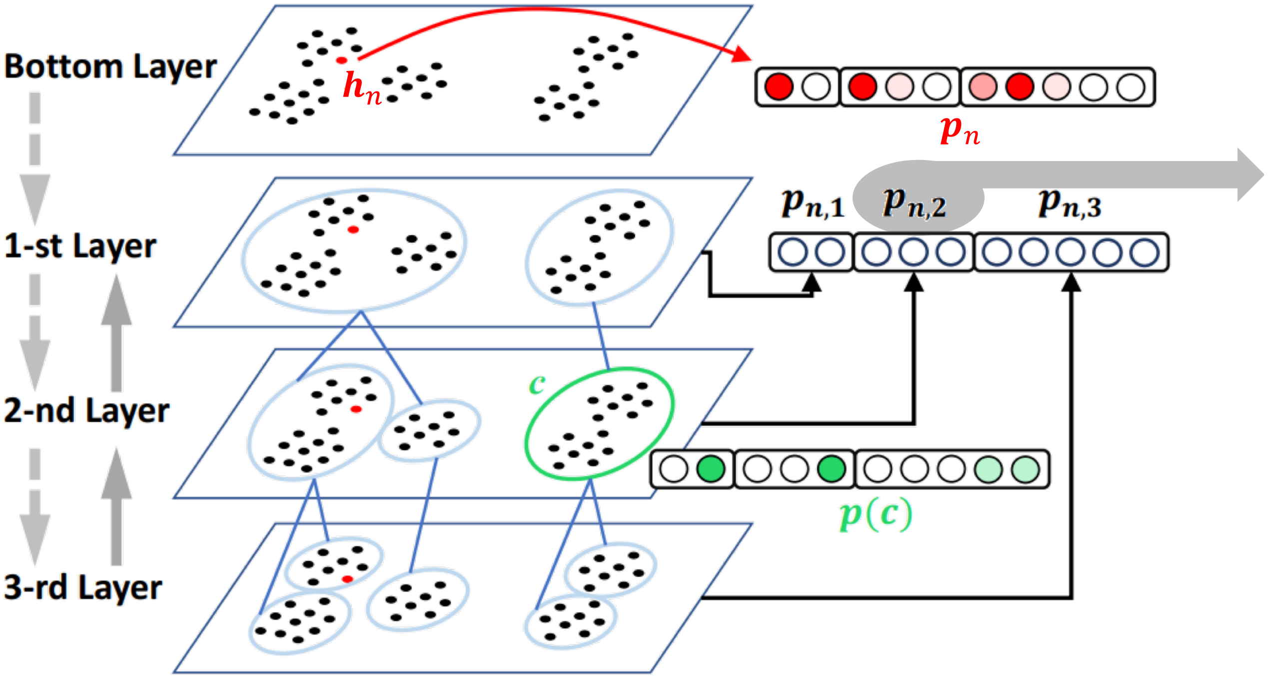


Affinity Graph Construction

Encoding Spatial Position

Interpretation Generation

Spatial encoding



$$\mathbf{p}_{n,l} = \mathbf{e}_n \mathbf{W}_L \mathbf{W}_{L-1} \dots \mathbf{W}_l$$

$\mathbf{e}_n \in \mathbb{R}^N$ is a one-hot vector, where $\mathbf{e}_n(i) = 1$ for $i = n$

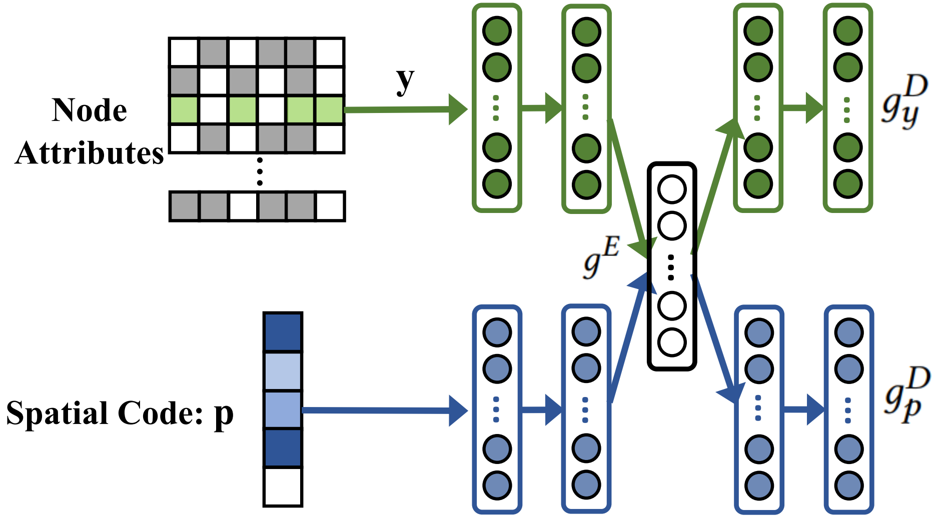
The spatial code for node n is $\mathbf{p}_n = [\hat{\mathbf{p}}_{n,1}, \hat{\mathbf{p}}_{n,2}, \dots, \hat{\mathbf{p}}_{n,L-1}, \hat{\mathbf{p}}_{n,L}]$

Multimodal autoencoder

- $\boldsymbol{y}$ are comprehensible node attributes

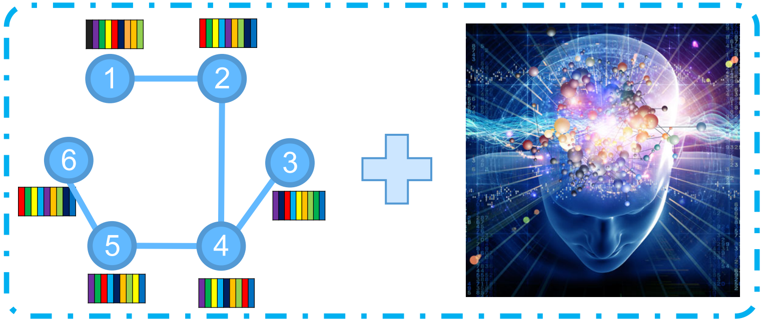
- Variational autoencoder is used to reconstruct $\boldsymbol{y}$ and $\boldsymbol{p}$

- After training the autoencoder, the interpretation for embedding representation $\boldsymbol{h}$ is,
  - $\mathbf{h} \sim f_{\text{linear}}(\mathbf{y}) = g(\mathbf{p}) = g_y^D \circ g^E(\mathbf{p}, \mathbf{0})$
  - The input to the node attribute side is set to be absent
  - The output from node attribute decoder is used as the interpretation



Attributed network analysis with humans in the loop

Initial Attributed Network



Embedding Representation

$$\mathbf{H} = \begin{bmatrix} 0.54 & 0.27 \\ 0.22 & 0.91 \\ 0.55 & 0.28 \\ 0.98 & 0.11 \\ 0.32 & 0.87 \\ 0.26 & 0.11 \end{bmatrix}$$

Tasks

- Classification
- Clustering
- Link Prediction
- Visualization
- …

[Huang et al. WSDM, 2018]

- Attributed network embedding (ANE) serves as infrastructures of various real-world applications
- We aim to learn cognition from experts and incorporate it into ANE to advance downstream analysis algorithms

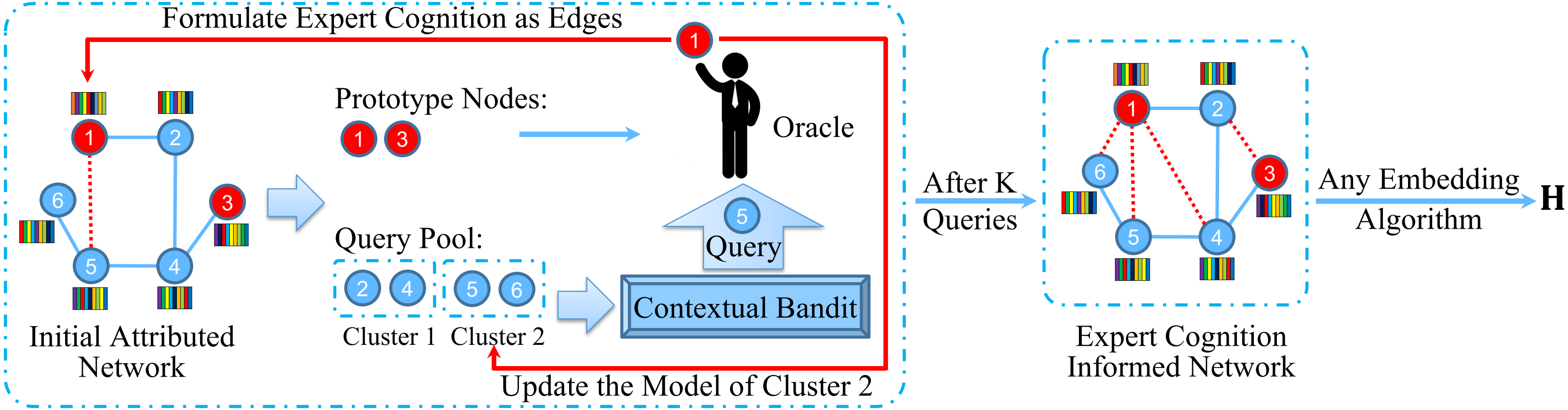
Expert cognition benefits data analysis

- Definition: Meaningful and Intelligence-related info that experts know beyond the data

- Understanding of domain knowledge
- Awareness of conventions
- Perception of latent relations

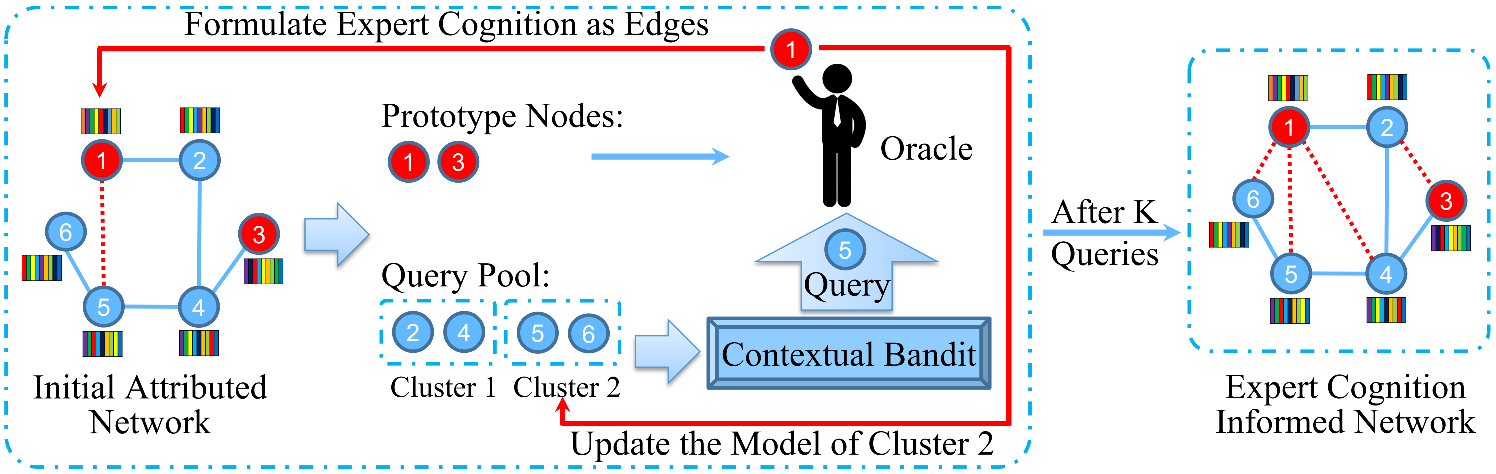
- Example: Human understand the sentiment in product reviews. This cognition could be applied to enhance the recommendations

Network embedding with expert cognition - NEEC



- Convert the abstract and meaningful cognition of domain experts into concrete answers
- Incorporate answers into ANE towards a more informative **H**
- Employ a general and concise form of queries to learn expert cognition from the oracle while greatly saving his/her effort

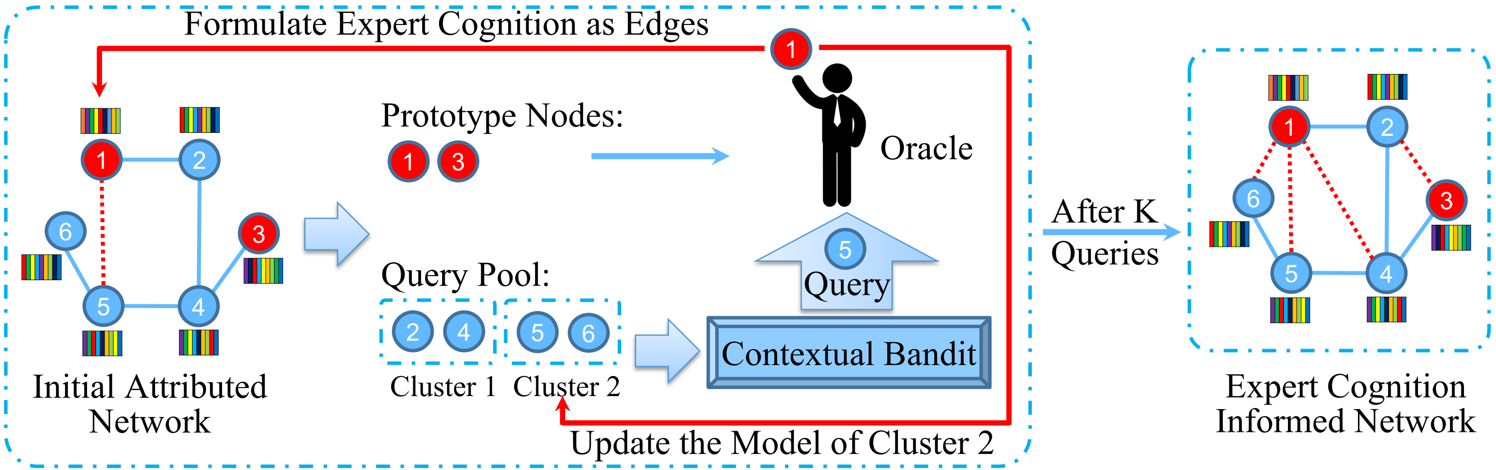
Strategies of framework NEEC



- Two steps to find the top K meaningful queries
  - Find few representative and distinct nodes (in red) as prototypes
  - Iteratively select K nodes from the remaining nodes (in blue) with the largest amount of expected learned expert cognition

- Oracle needs to indicate a node from the prototypes (e.g., $j = 1$) that is the most similar to the queried node $i = 5$

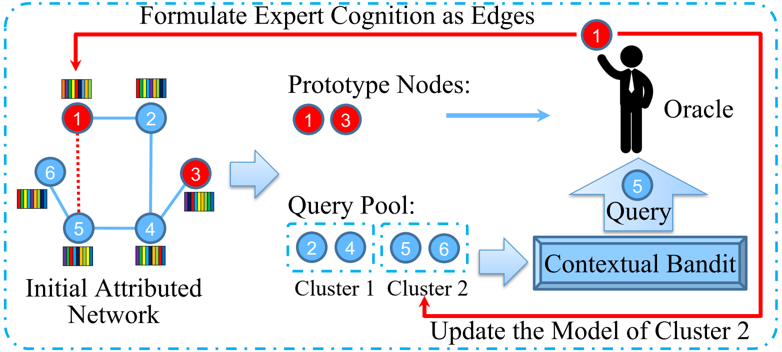
Strategies of framework NEEC



- Answers will be added into the network structure in the form of weighted edges, named as cognition edges (red dotted lines)
- With these cognition edges, different ANE methods can be directly applied to the expert cognition informed network to learn $\mathbf{H}$

Human-centric network analysis

- Focuses:
  Interpretable embedding, & utilizing network embedding to incorporate human knowledge
- Methods:
  Interpretable node representation learning
  Attributed network analysis with humans in the loop
- Techniques:
  Linking embedding with interpretable node attributes, converting knowledge into links, etc.



Summary of attributed network embedding

- ANE learns low-dimensional vectors to represent all nodes, bridging the gap between real-world systems & ML algorithms
- Challenges: Heterogeneity, large-scale, & Data Characteristics Vary Significantly
- Compare with other research topics
  - Multiview learning: Learn a unified representation of instances from multiple feature matrices observed from different aspects
  - Multimodal learning: Embed multiple sources with distinct modalities such as networks, images, and audio
  - Attributed network embedding: Preserve proximity information in networks and (one or multiple types of) node attributes

Summary of Attributed Network Embedding

- Shallow attributed network embedding:
    - Coupled spectral embedding
    - Coupled matrix & tri-factorization
    - Random walk based embedding
- Deep attributed network embedding:
    - Objective function based deep embedding
    - Graph neural networks
- Comprehensible node attributes help humans interact with systems.
    - Interpretable node representation learning
    - Attributed network analysis with humans in the loop

Acknowledgments

- DATA Lab and collaborators

COMPUTER SCIENCE & ENGINEERING TEXAS A&M UNIVERSITY

Arizona State University

DATA LAB

- Funding agencies
  - National Science Foundation
  - Defense Advanced Research Projects Agency

- Everyone attending the talk

References

- Hongchang Gao and Heng Huang. 2018. Deep Attributed Network Embedding. In IJCAI. 3364–3370.
- Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In NIPS. 1024–1034.
- Xiao Huang, Jundong Li, and Xia Hu. 2017. Accelerated Attributed Network Embedding. In SDM. 633–641.
- Xiao Huang, Jundong Li, and Xia Hu. 2017. Label Informed Attributed Network Embedding. In WSDM. 731–739.
- Xiao Huang, Jundong Li, Na Zou, and Xia Hu. 2018. A General Embedding Framework for Heterogeneous Information Learning in Large-Scale Networks. TKDD 12 (2018).
- Xiao Huang, Qingquan Song, Jundong Li, and Xia Hu. 2018. Exploring Expert Cognition for Attributed Network Embedding. In WSDM. 270–278.
- Xiao Huang, Qingquan Song, Fan Yang, and Xia Hu. 2019. Large-Scale Heterogeneous Feature Embedding. In AAAI.
- Xiao Huang, Qingquan Song, Yuening Li, and Xia Hu. 2019. Graph Recurrent Networks with Attributed Random Walks. In KDD.
- Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In ICLR.
- David Hallac, Jure Leskovec, and Stephen Boyd. 2015. Network Lasso: Clustering and Optimization in Large Graphs. In KDD. 387–396.
- Abhishek Kumar, Piyush Rai, and Hal Daume. 2011. Co-regularized Multi-view Spectral Clustering. In NIPS. 1413–1421.
- Jundong Li, Harsh Dani, Xia Hu, Jiliang Tang, Yi Chang, and Huan Liu. 2017. Attributed Network Embedding for Learning in a Dynamic Environment. In CIKM. 387–396.
- Jiongqian Liang, Peter Jacobs, Jiankai Sun, and Srinivasan Parthasarathy. 2018. Semi-Supervised Embedding In Attributed Networks With Outliers. In SDM. 153–161.
- Ninghao Liu, Xiao Huang, and Xia Hu. 2017. Accelerated Local Anomaly Detection via Resolving Attributed Networks. In IJCAI. 2337–2343.
- Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. 2016. Tri- Party Deep Network Representation. In IJCAI. 1895–1901.
- Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In KDD. 701–710.
- Guo-Jun Qi, Charu Aggarwal, Qi Tian, Heng Ji, and Thomas S. Huang. 2012. Exploring Context and Content Links in Social Media: A Latent Space Method. TPAMI 34, 5 (2012), 850–862.
- Ulrike von Luxburg. 2007. A Tutorial on Spectral Clustering. Statistics and Computing 17, 4 (2007), 395–416.

- Hongchang Gao and Heng Huang. 2018. Deep Attributed Network Embedding. In IJCAI. 3364–3370.

- Lizi Liao, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. 2018. Attributed Social Network Embedding. TKDE 30, 12 (2018), 2257–2270.
- Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. 2015. Network Representation Learning with Rich Text Information. In IJCAI. 2111– 2117.
- Shenghuo Zhu, Kai Yu, Yun Chi, and Yihong Gong. 2007. Combining Content and Link for Classification Using Matrix Factorization. In SIGIR. 487–494.
- Zhen Zhang, Hongxia Yang, Jiajun Bu, Sheng Zhou, Pinggang Yu, Jianwei Zhang, Martin Ester, and Can Wang. 2018. ANRL: Attributed Network Representation Learning via Deep Neural Networks. In IJCAI. 3155–3161.