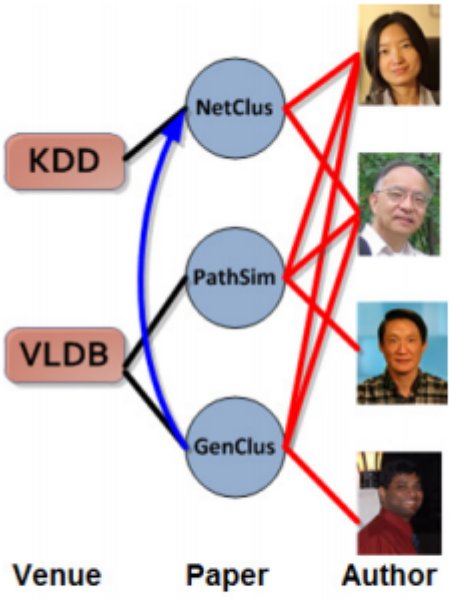
DATA MINING

/CISC 873 - Steven Ding
/Graph Representation

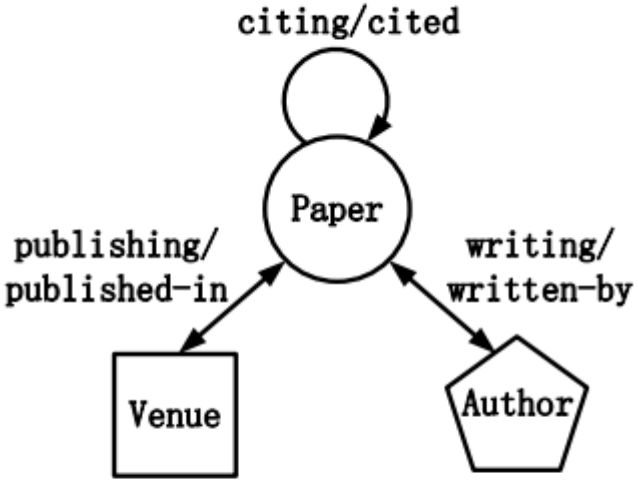
- Part III: Network
  - ***Supervised/Self-supervised learning on information network***

Graph Types

- Homogeneous Network vs Heterogenous Information Network

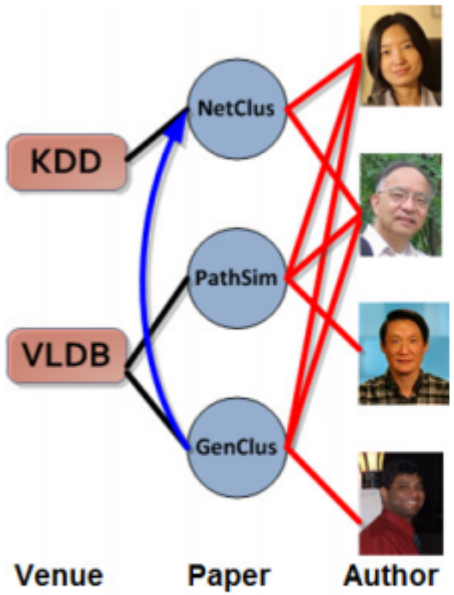


(a) Network instance

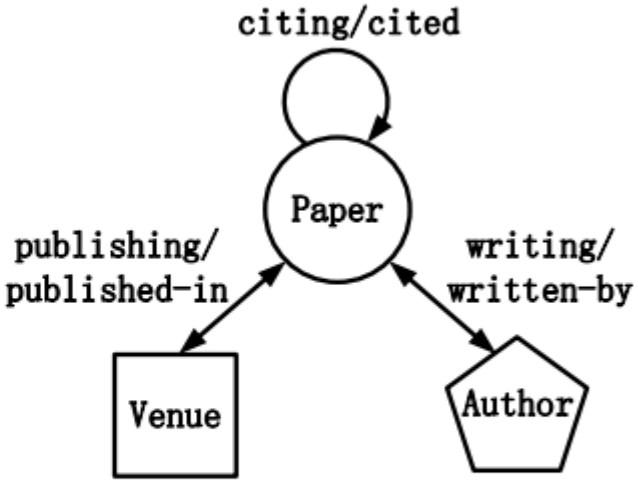


(b) Network schema

Graph Types



(a) Network instance

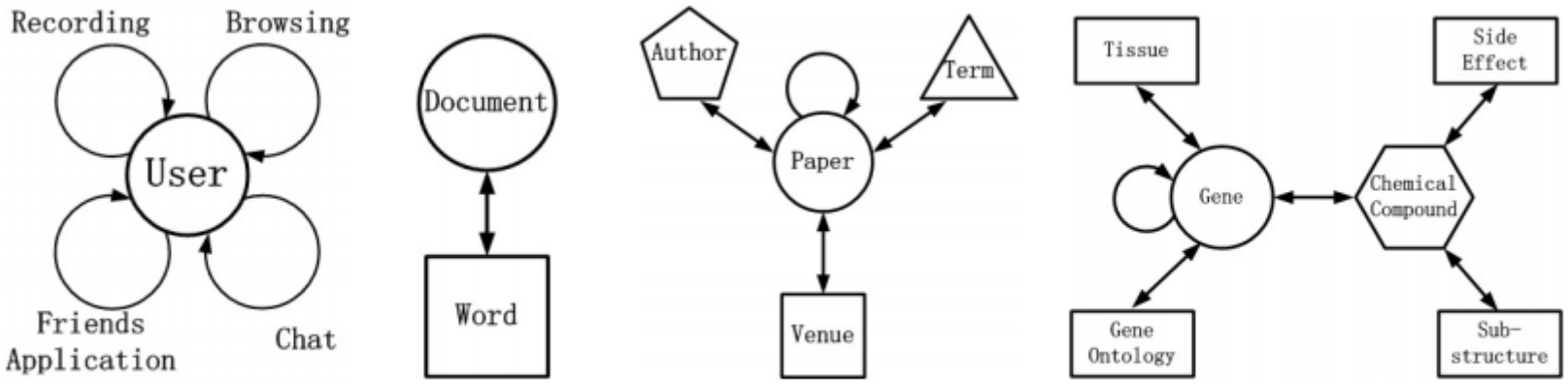


(b) Network schema

DEFINITION *2:* **Heterogeneous/homogeneous information Network**. The information network is called **heterogeneous information network** if the types of objects $|\mathcal{A}| > 1$ or the types of relations $|\mathcal{R}| > 1$; otherwise, it is a **homogeneous information network**.

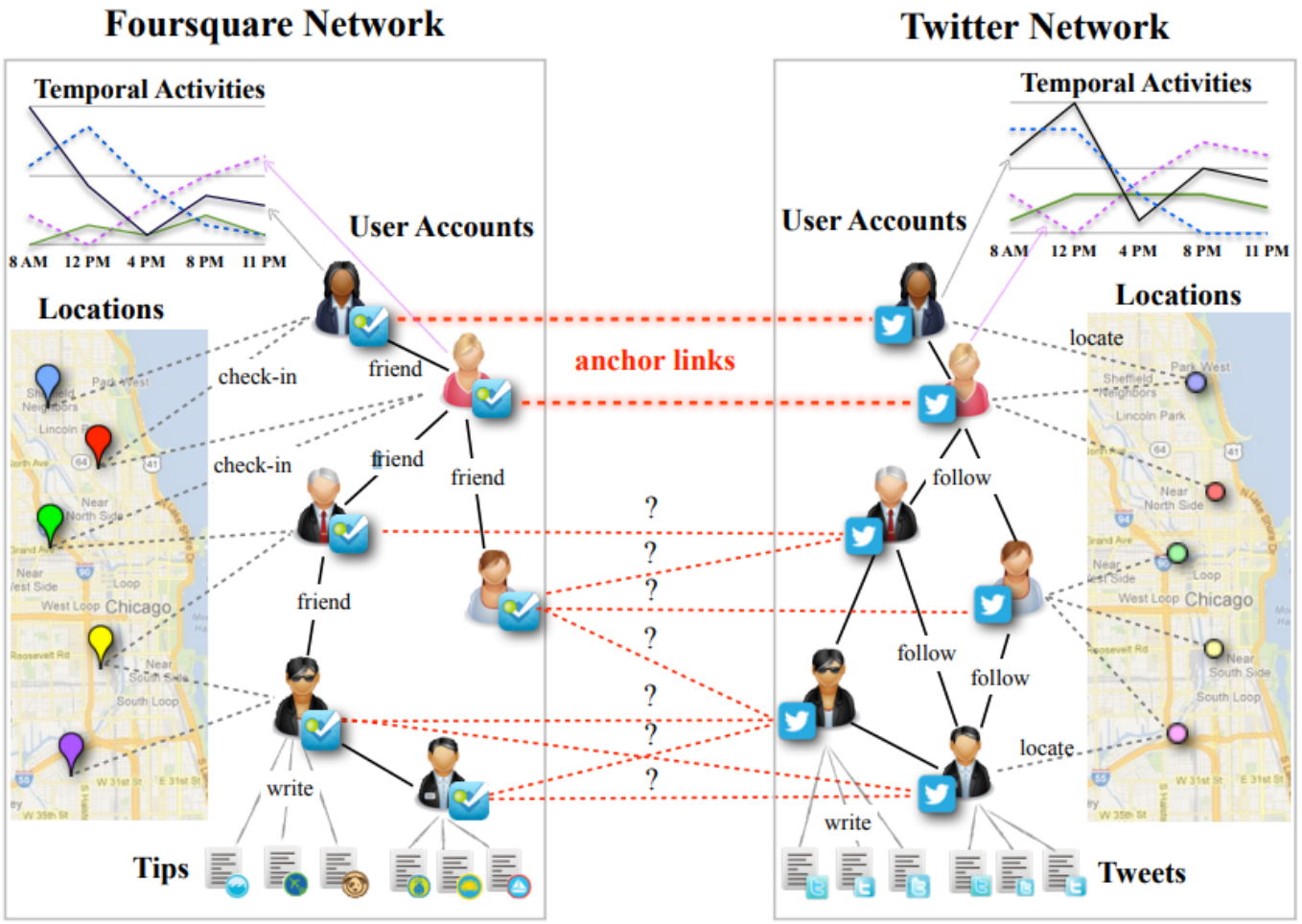
Graph Types

- Heterogenous Information Network
  - Example graph?
  - Social Media Platform
    - Users, Tags, Posts
  - Health Care System:
    - Doctors, Patients, Diseases, Devices



(a) Multi-relation (b) Bipartite (c) Star-schema (d) Multiple-hub

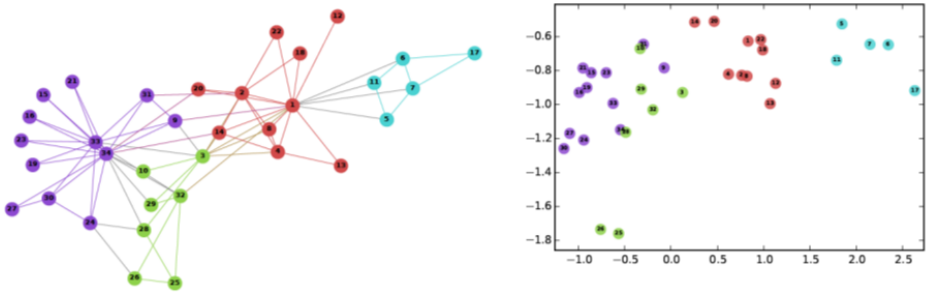
Reality is much more complicated



Assumption

- Graph Data
  - One-hot encoding of ***first-order neighbor***
    - [1, 0 , 0, 0, 1, 0, 0 , 0, 1, 1, 0, 0, 0]
  - One-hot encoding of ***second-order neighbor***
    - [0, 0 , 1, 0, 0, 0, 0 , 0, 0, 0, 0, 1, 1]

  - lie in a low dimensional manifold

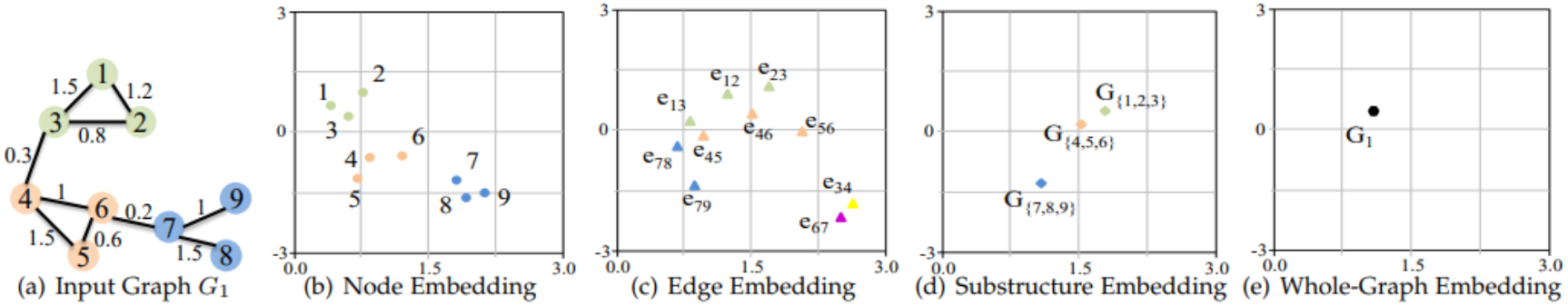


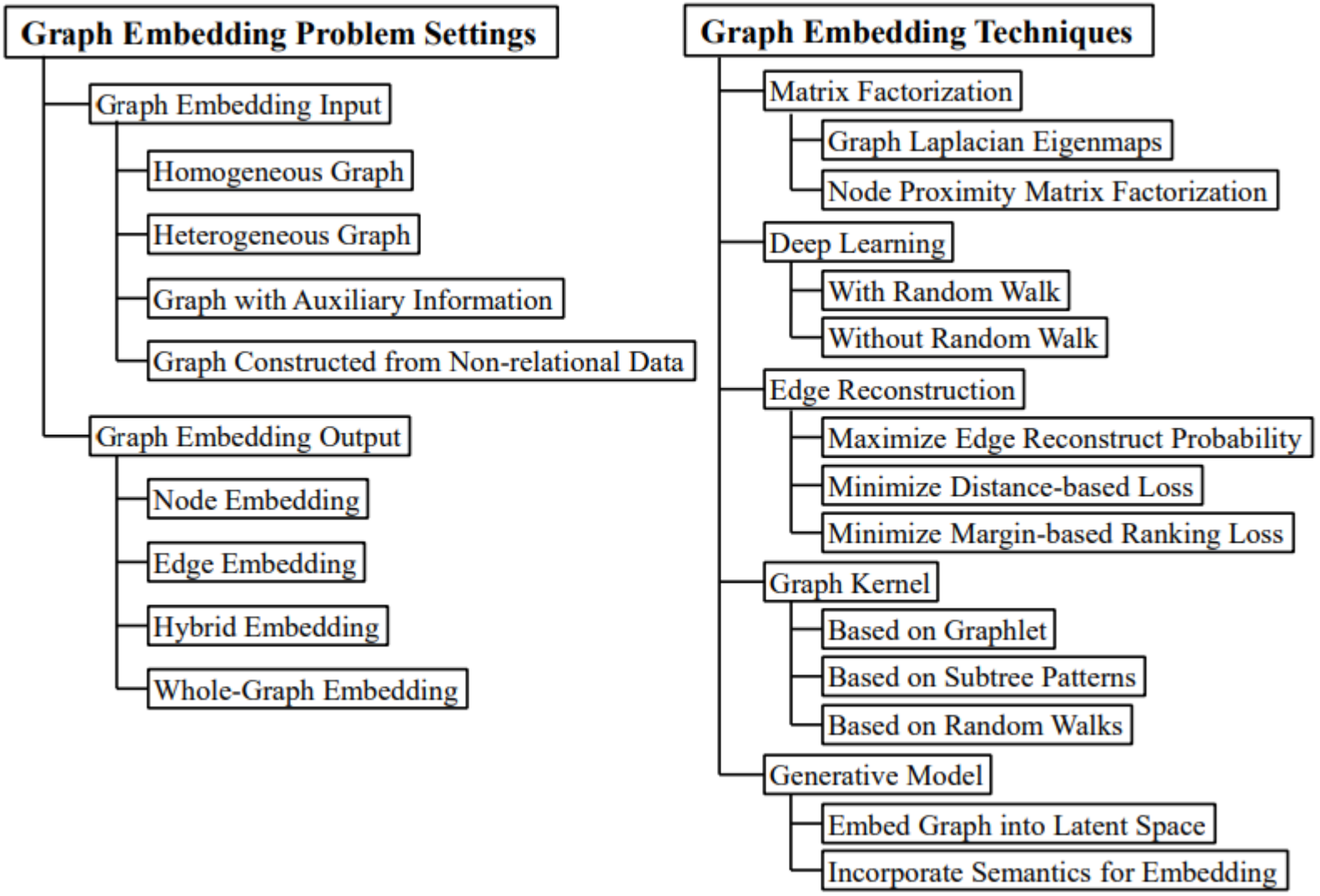
Input Output

- Complex non-relational graph structure -> low dimensional embedding space

Why Graph Embedding

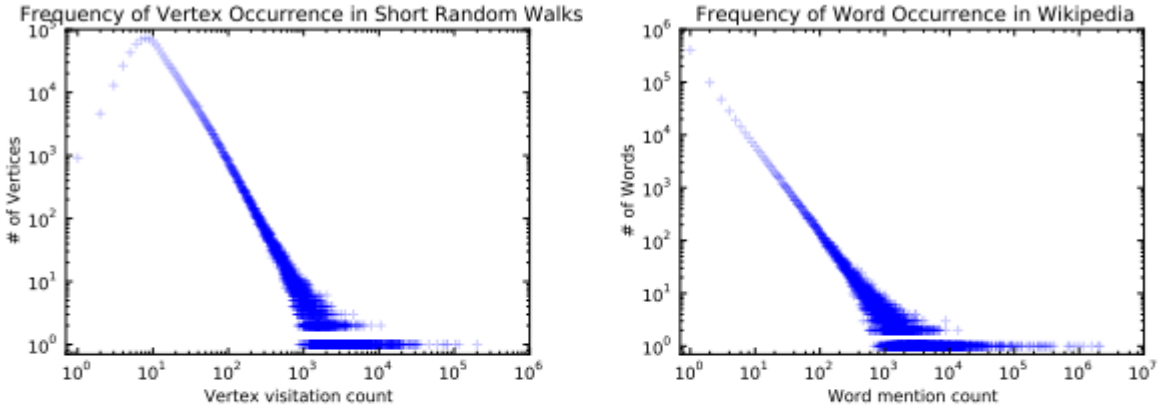
- Latent Embedding -> Hidden Regularities of the Data
- Similar to Language Model, useful for a lot of downstream task
  - Link prediction
  - Node prediction
  - Recommendation
  - Tag Prediction




Graph Embedding Problem Settings
Graph Embedding Input
Homogeneous Graph
Heterogeneous Graph
Graph with Auxiliary Information
Graph Constructed from Non-relational Data
Graph Embedding Output
Node Embedding
Edge Embedding
Hybrid Embedding
Whole-Graph Embedding
Graph Embedding Techniques
Matrix Factorization
Graph Laplacian Eigenmaps
Node Proximity Matrix Factorization
Deep Learning
With Random Walk
Without Random Walk
Edge Reconstruction
Maximize Edge Reconstruct Probability
Minimize Distance-based Loss
Minimize Margin-based Ranking Loss
Graph Kernel
Based on Graphlet
Based on Subtree Patterns
Based on Random Walks
Generative Model
Embed Graph into Latent Space
Incorporate Semantics for Embedding

DeepWalk

- Basic Idea -> Graph is similar to Text!
  - (if we randomly travel to through the graph)
  - Connection: Power laws



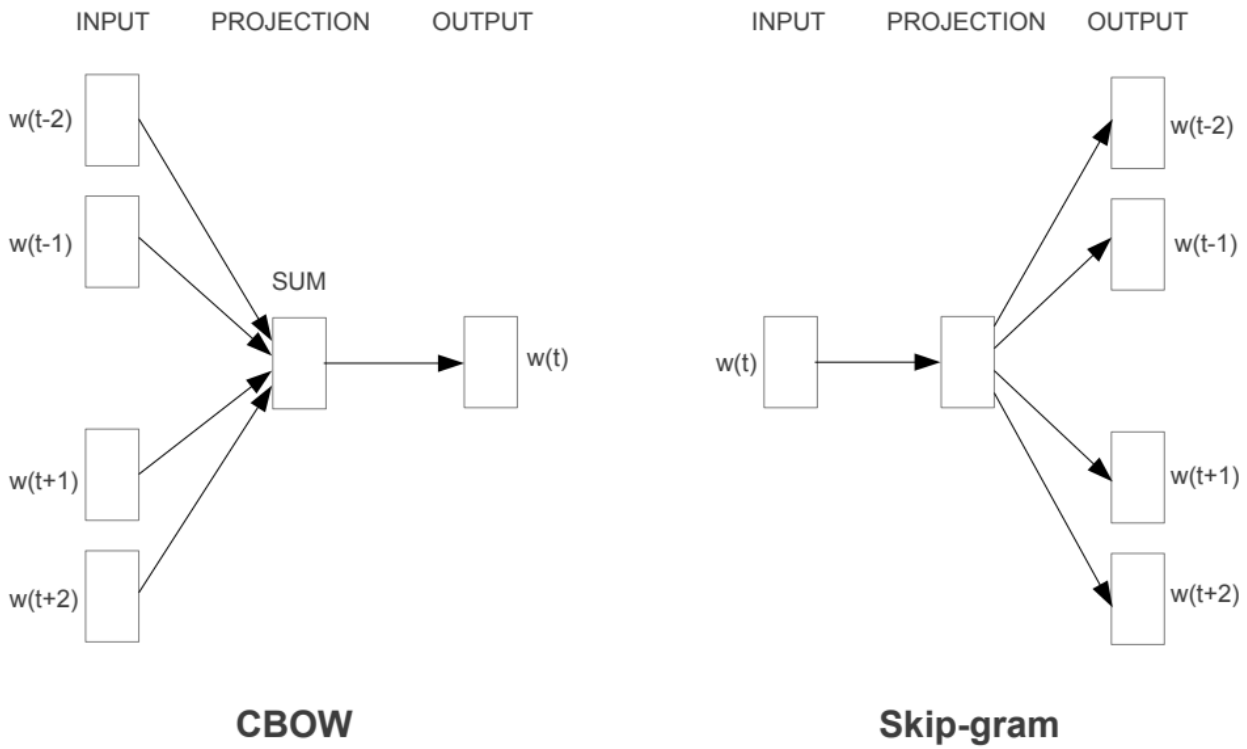
(a) YouTube Social Graph (b) Wikipedia Article Text

Figure 2: The power-law distribution of vertices appearing in short random walks (2a) follows a power-law, much like the distribution of words in natural language (2b).

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

DeepWalk

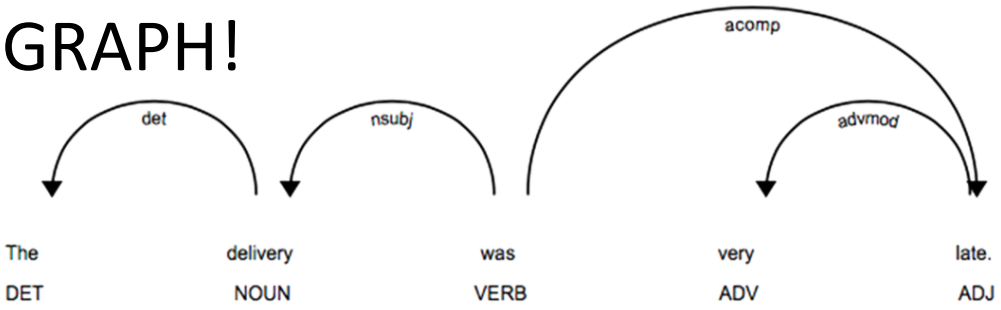
- Word2Vec
  - Recall:



Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

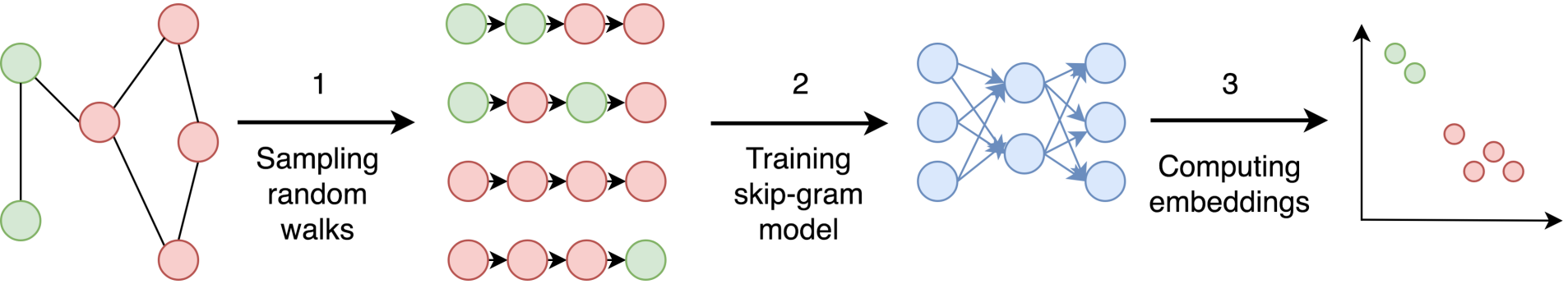
DeepWalk

- Word2Vec
  - Create questions (and true answers) from the dataset to train the model.
  - The, baby, is, **smiling**, at, her, mom
    - Question: The, baby, is, ___, at, her, mom
    - Answer: smiling
  - The, baby, is, **flying**, at, her, mom
    - Question: The, baby, is, ___, at, her, mom
    - Answer: flying
- In fact, Sentence is GRAPH!



Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

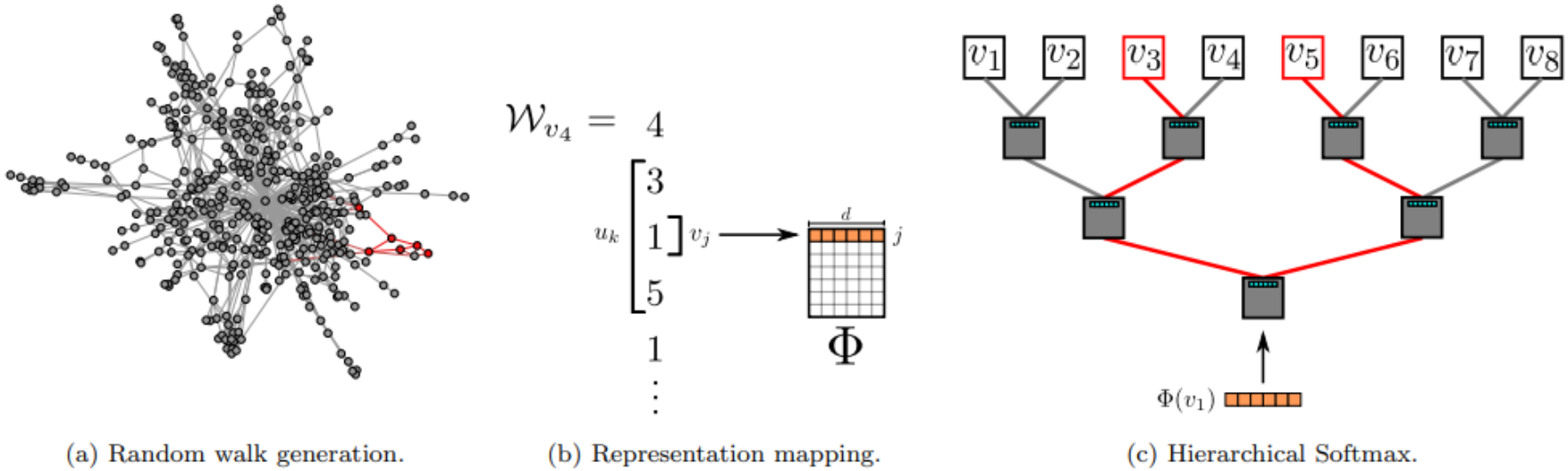
DeepWalk



Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

DeepWalk

- Three steps:
  - Sequence Generation (random walk)
  - Representation mapping (embedding layer)
  - Prediction (through word2vec)



(a) Random walk generation. (b) Representation mapping. (c) Hierarchical Softmax.

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

DeepWalk

- Performance?
  - BlogCataLog classification
  - Flicker label prediction
  - Youtube Label Prediction

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

DeepWalk

	% Labeled Nodes	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1(%)	DeepWalk	**36.00**	**38.20**	**39.60**	**40.30**	**41.00**	**41.30**	41.50	41.50	42.00
	SpectralClustering	31.06	34.95	37.27	38.93	39.97	40.99	**41.66**	**42.42**	**42.62**
	EdgeCluster	27.94	30.76	31.85	32.99	34.12	35.00	34.63	35.99	36.29
	Modularity	27.35	30.74	31.77	32.97	34.09	36.13	36.08	37.23	38.18
	wvRN	19.51	24.34	25.62	28.82	30.37	31.81	32.19	33.33	34.28
	Majority	16.51	16.66	16.61	16.70	16.91	16.99	16.92	16.49	17.26
Macro-F1(%)	DeepWalk	**21.30**	**23.80**	25.30	26.30	27.30	27.60	27.90	28.20	28.90
	SpectralClustering	19.14	23.57	**25.97**	**27.46**	**28.31**	**29.46**	**30.13**	**31.38**	**31.78**
	EdgeCluster	16.16	19.16	20.48	22.00	23.00	23.64	23.82	24.61	24.92
	Modularity	17.36	20.00	20.80	21.85	22.65	23.41	23.89	24.20	24.97
	wvRN	6.25	10.13	11.64	14.24	15.86	17.18	17.98	18.86	19.57
	Majority	2.52	2.55	2.52	2.58	2.58	2.63	2.61	2.48	2.62

Table 2: Multi-label classification results in BlogCatalog

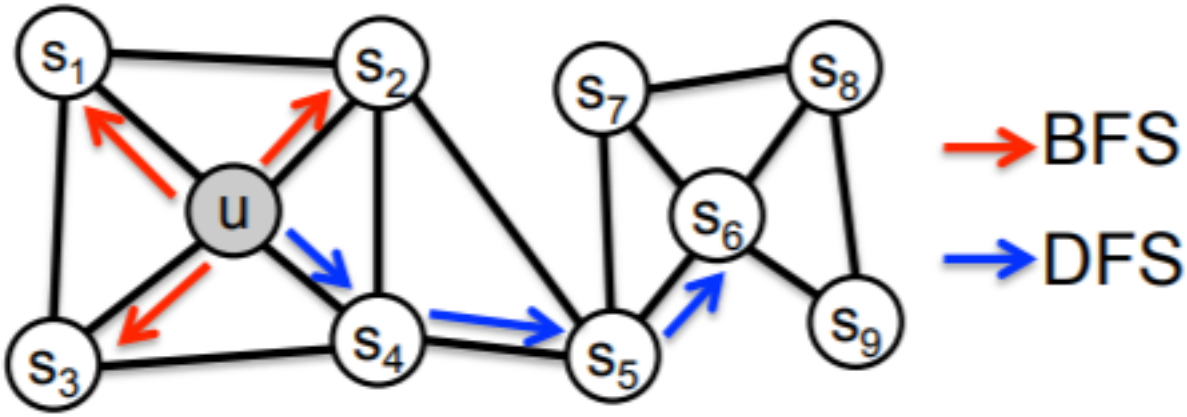
	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1(%)	DeepWalk	**32.4**	**34.6**	**35.9**	**36.7**	**37.2**	**37.7**	**38.1**	**38.3**	**38.5**	**38.7**
	SpectralClustering	27.43	30.11	31.63	32.69	33.31	33.95	34.46	34.81	35.14	35.41
	EdgeCluster	25.75	28.53	29.14	30.31	30.85	31.53	31.75	31.76	32.19	32.84
	Modularity	22.75	25.29	27.3	27.6	28.05	29.33	29.43	28.89	29.17	29.2
	wvRN	17.7	14.43	15.72	20.97	19.83	19.42	19.22	21.25	22.51	22.73
	Majority	16.34	16.31	16.34	16.46	16.65	16.44	16.38	16.62	16.67	16.71
Macro-F1(%)	DeepWalk	**14.0**	17.3	**19.6**	**21.1**	**22.1**	**22.9**	**23.6**	**24.1**	**24.6**	**25.0**
	SpectralClustering	13.84	**17.49**	19.44	20.75	21.60	22.36	23.01	23.36	23.82	24.05
	EdgeCluster	10.52	14.10	15.91	16.72	18.01	18.54	19.54	20.18	20.78	20.85
	Modularity	10.21	13.37	15.24	15.11	16.14	16.64	17.02	17.1	17.14	17.12
	wvRN	1.53	2.46	2.91	3.47	4.95	5.56	5.82	6.59	8.00	7.26
	Majority	0.45	0.44	0.45	0.46	0.47	0.44	0.45	0.47	0.47	0.47

Table 3: Multi-label classification results in Flickr

Perozzi, B., Al-Rfou, R., & Skiena, S. (2014, August). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 701-710).

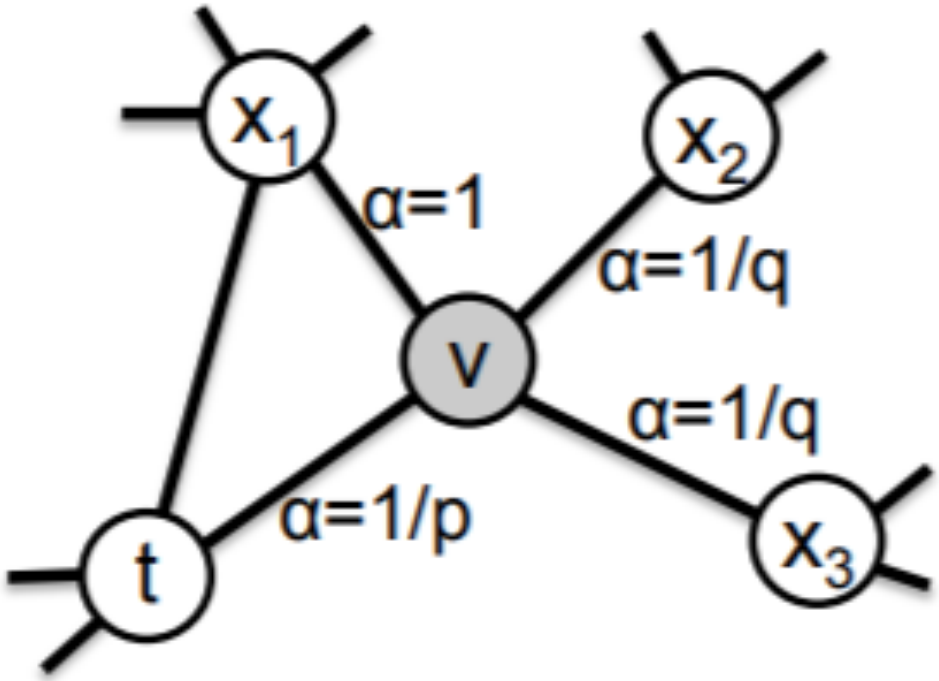
Node2Vec

- Second order traversal
  - Breadth-first Sampling (BFS)
  - Depth-first Sampling (DFS)
  - BFS, DFS traversal yields very different sequence
  - BFS -> emphasize 1st order neighbor
  - DFS -> emphasize >2nd order neighbor



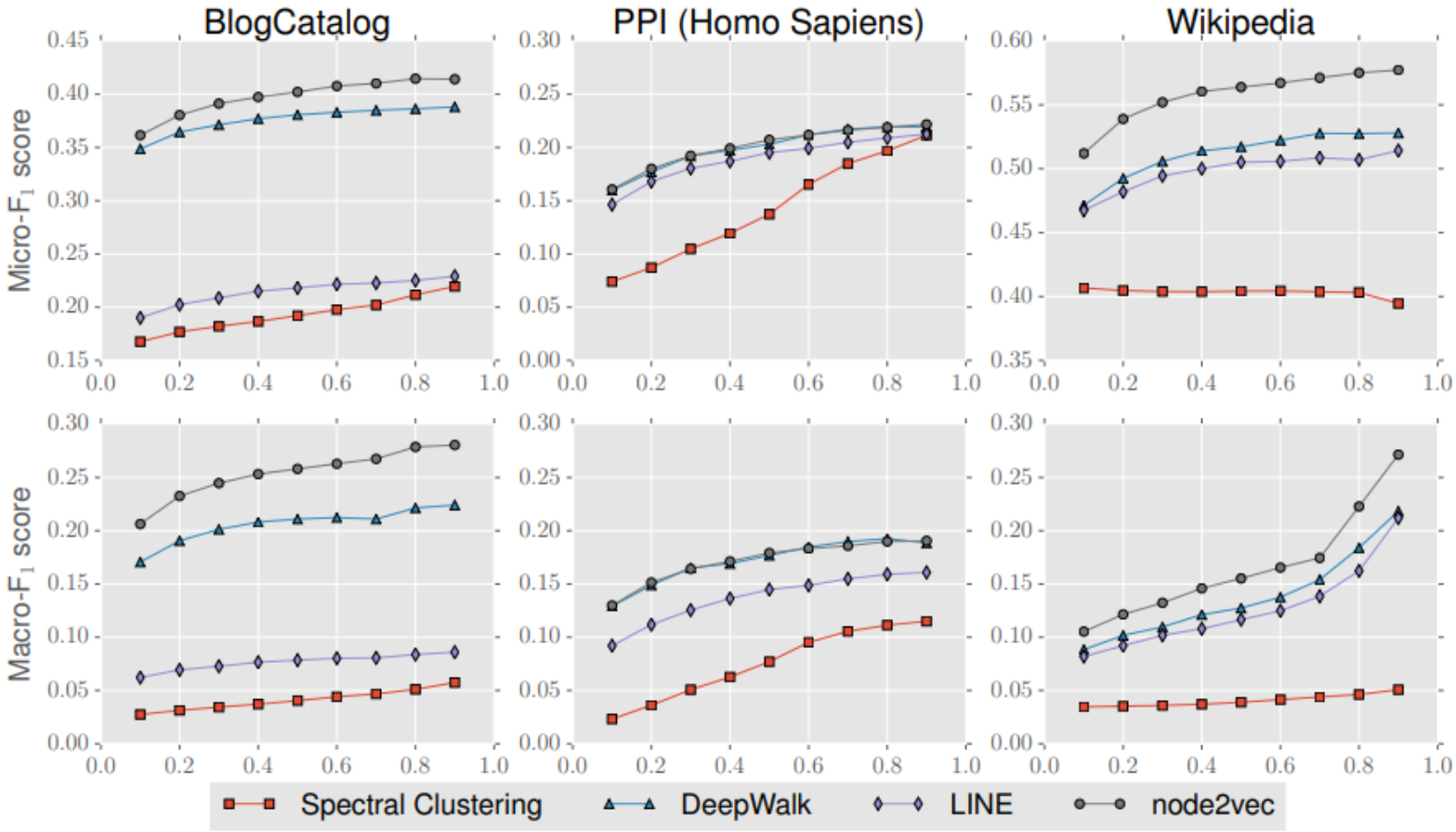
Node2Vec

- Controlled Random Walk
  - p — return rate
    q — exploration rate



Node2Vec

- Learning? Same as DeepWalk and Word2Vec

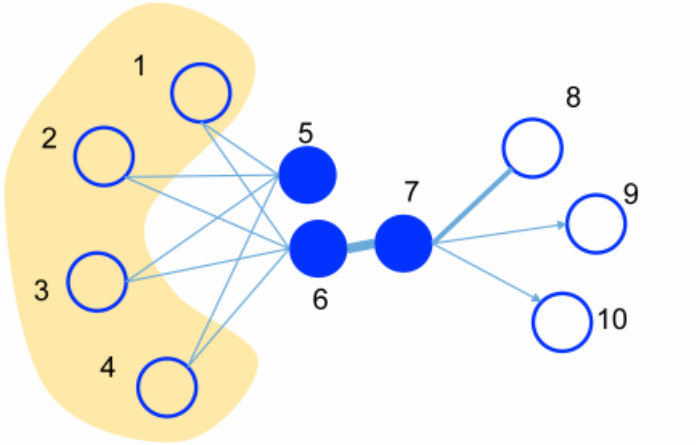


LINE

- Edge sampling instead of random walk
- Separate embedding space into
  - 1st order proximity
  - 2nd order proximity
- Avoid sequence generation
- More efficient and scalable
- the consideration of the second order proximity effectively complements the sparsity of the first-order proximity and better preserves the global structure of the network.

LINE

- Edge sampling instead of random walk
- Separate embedding space into
  - 1st order proximity
  - 2nd order proximity



- Avoid sequence generation
- More efficient and scalable
- the consideration of the second order proximity effectively complements the sparsity of the first-order proximity and better preserves the global structure of the network.

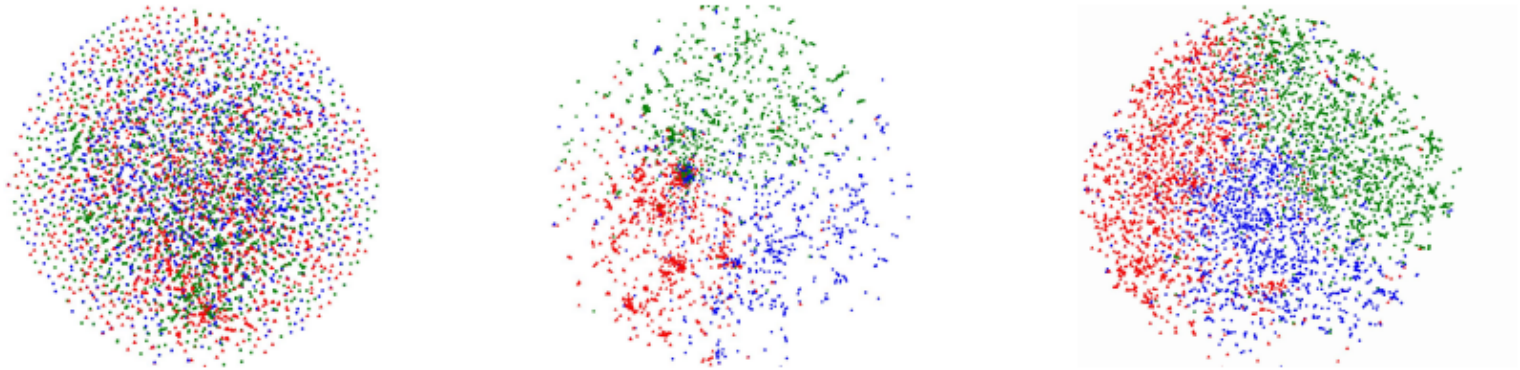
LINE

Table 3: Results of Wikipedia page classification on WIKIPEDIA data set.

Metric	Algorithm	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1	GF	79.63	80.51	80.94	81.18	81.38	81.54	81.63	81.71	81.78
	DeepWalk	78.89	79.92	80.41	80.69	80.92	81.08	81.21	81.35	81.42
	SkipGram	79.84	80.82	81.28	81.57	81.71	81.87	81.98	82.05	82.09
	LINE-SGD(1st)	76.03	77.05	77.57	77.85	78.08	78.25	78.39	78.44	78.49
	LINE-SGD(2nd)	74.68	76.53	77.54	78.18	78.63	78.96	79.19	79.40	79.57
	LINE(1st)	79.67	80.55	80.94	81.24	81.40	81.52	81.61	81.69	81.67
	LINE(2nd)	79.93	80.90	81.31	81.63	81.80	81.91	82.00	82.11	82.17
	LINE(1st+2nd)	**81.04****	**82.08****	**82.58****	**82.93****	**83.16****	**83.37****	**83.52****	**83.63****	**83.74****
Macro-F1	GF	79.49	80.39	80.82	81.08	81.26	81.40	81.52	81.61	81.68
	DeepWalk	78.78	79.78	80.30	80.56	80.82	80.97	81.11	81.24	81.32
	SkipGram	79.74	80.71	81.15	81.46	81.63	81.78	81.88	81.98	82.01
	LINE-SGD(1st)	75.85	76.90	77.40	77.71	77.94	78.12	78.24	78.29	78.36
	LINE-SGD(2nd)	74.70	76.45	77.43	78.09	78.53	78.83	79.08	79.29	79.46
	LINE(1st)	79.54	80.44	80.82	81.13	81.29	81.43	81.51	81.60	81.59
	LINE(2nd)	79.82	80.81	81.22	81.52	81.71	81.82	81.92	82.00	82.07
	LINE(1st+2nd)	**80.94****	**81.99****	**82.49****	**82.83****	**83.07****	**83.29****	**83.42****	**83.55****	**83.66****

Significantly outperforms GF at the: ** 0.01 and * 0.05 level, paired t-test.

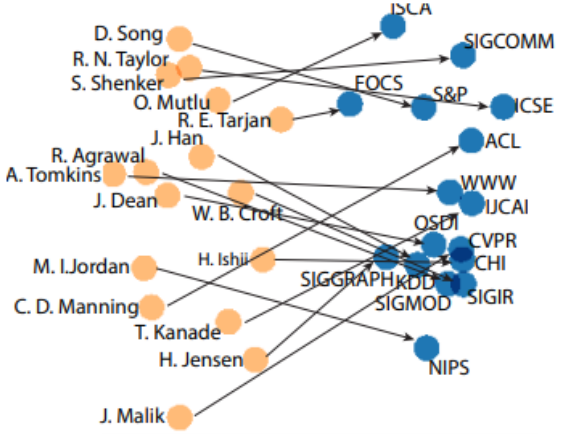
LINE



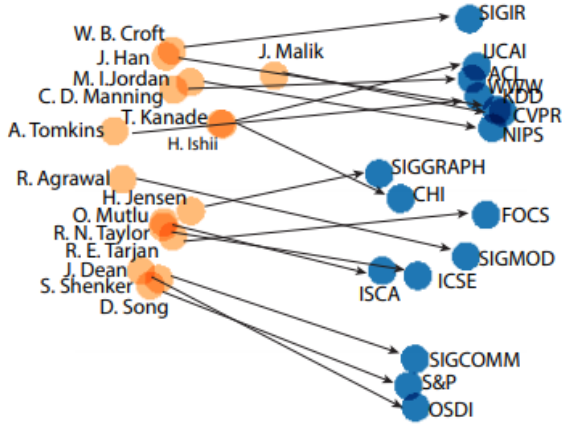
(a) GF (b) DeepWalk (c) LINE(2nd)

Figure 2: Visualization of the co-author network. The authors are mapped to the 2-D space using the t-SNE package with learned embeddings as input. Color of a node indicates the community of the author. Red: "data Mining," blue: "machine learning," green: "computer vision."

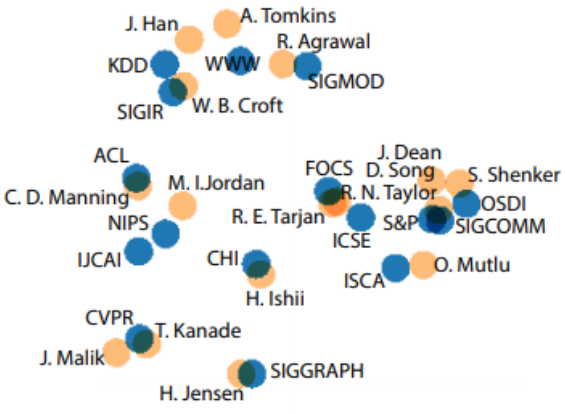
metapath2vec



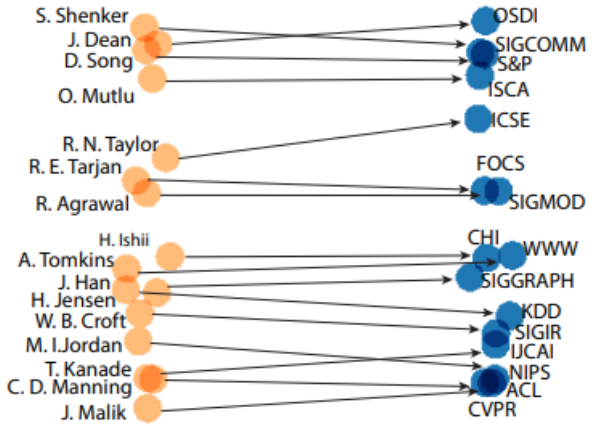
(a) DeepWalk / node2vec



(b) PTE



(c) *metapath2vec*



(d) *metapath2vec++*

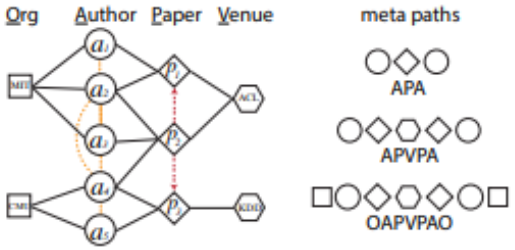
metapath2vec

- consider the nature of multi-typed node & entities

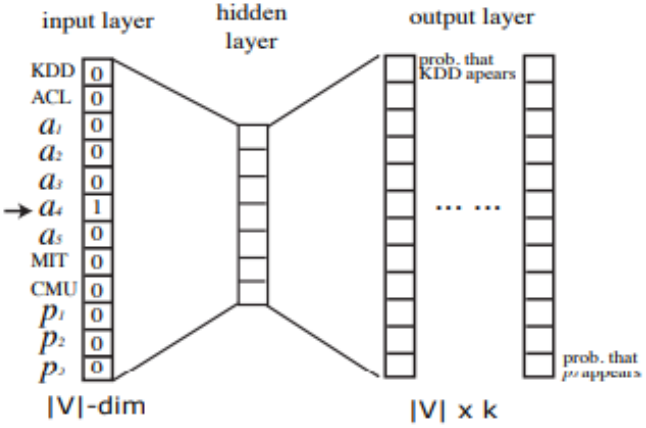
META PATH EXAMPLES AND THEIR PHYSICAL MEANINGS ON BIBLIOGRAPHIC DATA.

Path instance	Meta path	Physical meaning
Sun-NetClus-Han Sun-PathSim-Yu	Author-Paper-Author (*APA*)	Authors collaborate on the same paper
Sun-PathSim-VLDB-PathSim-Han Sun-PathSim-VLDB-GenClus-Aggarwal	Author-Paper-Venue-Paper-Author (*APVPA*)	Authors publish papers on the same venue
Sun-NetClus-KDD Sun-PathSim-VLDB	Author-Paper-Venue (*APV*)	Authors publish papers at a venue

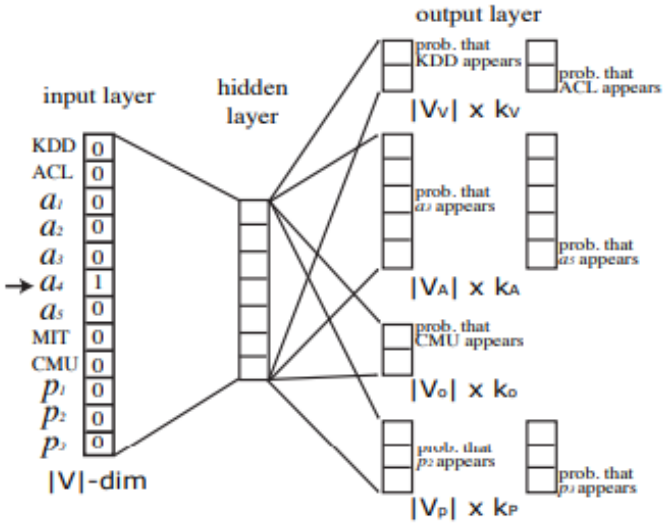
Metapath2vec



(a) An academic network



(b) Skip-gram in *metapath2vec*, node2vec, & DeepWalk



(c) Skip-gram in *metapath2vec++*

Metapath2vec

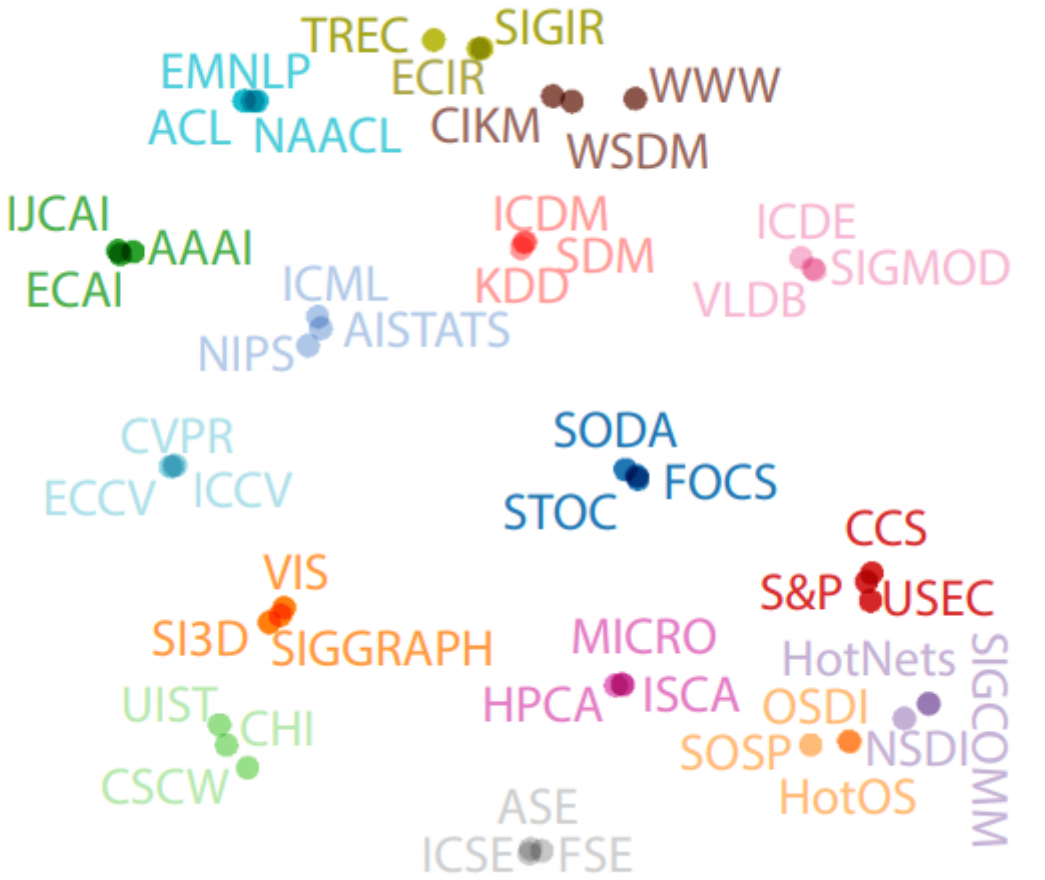


Table 2: Multi-class venue node classification results in AMiner data.

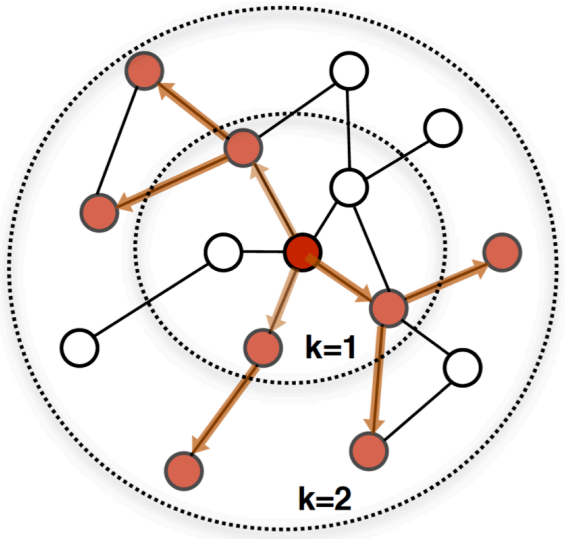
Metric	Method	5%	10%	20%	30%	40%	50%	60%	70%	80%	90%
Macro-F1	DeepWalk/node2vec	0.0723	0.1396	0.1905	0.2795	0.3427	0.3911	0.4424	0.4774	0.4955	0.4457
	LINE (1st+2nd)	0.2245	0.4629	0.7011	0.8473	0.8953	0.9203	0.9308	0.9466	0.9410	0.9466
	PTE	0.1702	0.3388	0.6535	0.8304	0.8936	0.9210	0.9352	0.9505	0.9525	0.9489
	metapath2vec	0.3033	0.5247	0.8033	0.8971	0.9406	0.9532	0.9529	0.9701	0.9683	0.9670
	metapath2vec++	0.3090	0.5444	0.8049	0.8995	0.9468	0.9580	0.9561	0.9675	0.9533	0.9503
Micro-F1	DeepWalk/node2vec	0.1701	0.2142	0.2486	0.3266	0.3788	0.4090	0.4630	0.4975	0.5259	0.5286
	LINE (1st+2nd)	0.3000	0.5167	0.7159	0.8457	0.8950	0.9209	0.9333	0.9500	0.9556	0.9571
	PTE	0.2512	0.4267	0.6879	0.8372	0.8950	0.9239	0.9352	0.9550	0.9667	0.9571
	metapath2vec	0.4173	0.5975	0.8327	0.9011	0.9400	0.9522	0.9537	0.9725	0.9815	0.9857
	metapath2vec++	0.4331	0.6192	0.8336	0.9032	0.9463	0.9582	0.9574	0.9700	0.9741	0.9786

Table 3: Multi-class author node classification results in AMiner data.

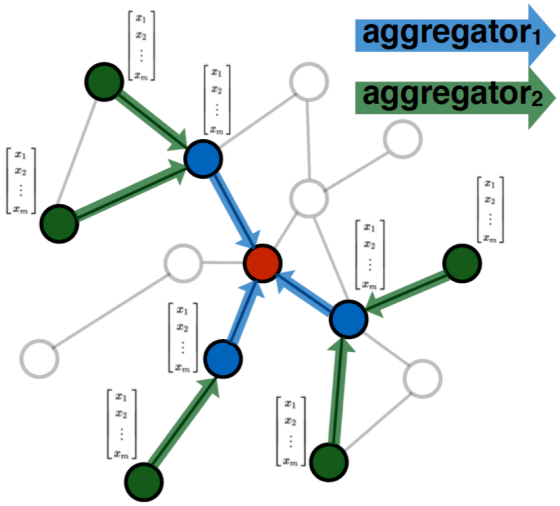
Metric	Method	5%	10%	20%	30%	40%	50%	60%	70%	80%	90%
Macro-F1	DeepWalk/node2vec	0.7153	0.7222	0.7256	0.7270	0.7273	0.7274	0.7273	0.7271	0.7275	0.7275
	LINE (1st+2nd)	0.8849	0.8886	0.8911	0.8921	0.8926	0.8929	0.8934	0.8936	0.8938	0.8934
	PTE	0.8898	0.8940	0.897	0.8982	0.8987	0.8990	0.8997	0.8999	0.9002	0.9005
	metapath2vec	0.9216	0.9262	0.9292	0.9303	0.9309	0.9314	0.9315	0.9316	0.9319	0.9320
	metapath2vec++	0.9107	0.9156	0.9186	0.9199	0.9204	0.9207	0.9207	0.9208	0.9211	0.9212
Micro-F1	DeepWalk/node2vec	0.7312	0.7372	0.7402	0.7414	0.7418	0.7420	0.7419	0.7420	0.7425	0.7425
	LINE (1st+2nd)	0.8936	0.8969	0.8993	0.9002	0.9007	0.9010	0.9015	0.9016	0.9018	0.9017
	PTE	0.8986	0.9023	0.9051	0.9061	0.9066	0.9068	0.9075	0.9077	0.9079	0.9082
	metapath2vec	0.9279	0.9319	0.9346	0.9356	0.9361	0.9365	0.9365	0.9365	0.9367	0.9369
	metapath2vec++	0.9173	0.9217	0.9243	0.9254	0.9259	0.9261	0.9261	0.9262	0.9264	0.9266

GraphSAGE

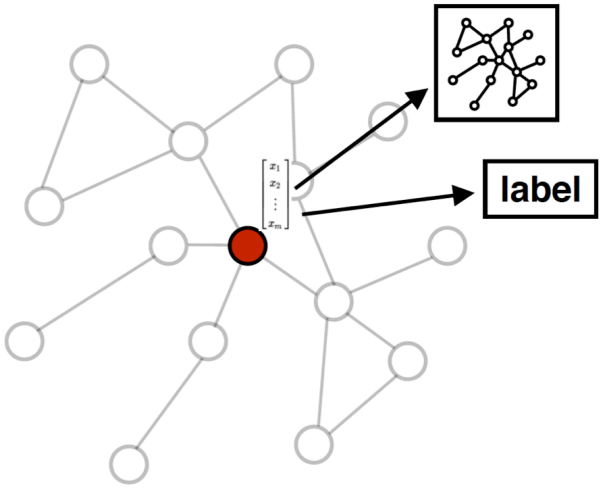
- Semi-supervised learning



1. Sample neighborhood



2. Aggregate feature information from neighbors



3. Predict graph context and label using aggregated information

GraphSAGE

Table 1: Prediction results for the three datasets (micro-averaged F1 scores). Results for unsupervised and fully supervised GraphSAGE are shown. Analogous trends hold for macro-averaged scores.

	Citation		Reddit		PPI	
Name	Unsup. F1	Sup. F1	Unsup. F1	Sup. F1	Unsup. F1	Sup. F1
Random	0.206	0.206	0.043	0.042	0.396	0.396
Raw features	0.575	0.575	0.585	0.585	0.422	0.422
DeepWalk	0.565	0.565	0.324	0.324	—	—
DeepWalk + features	0.701	0.701	0.691	0.691	—	—
GraphSAGE-GCN	0.742	0.772	**0.908**	0.930	0.465	0.500
GraphSAGE-mean	0.778	0.820	0.897	0.950	0.486	0.598
GraphSAGE-LSTM	0.788	0.832	**0.907**	**0.954**	0.482	**0.612**
GraphSAGE-pool	**0.798**	**0.839**	0.892	0.948	**0.502**	0.600
% gain over feat.	39%	46%	55%	63%	19%	45%

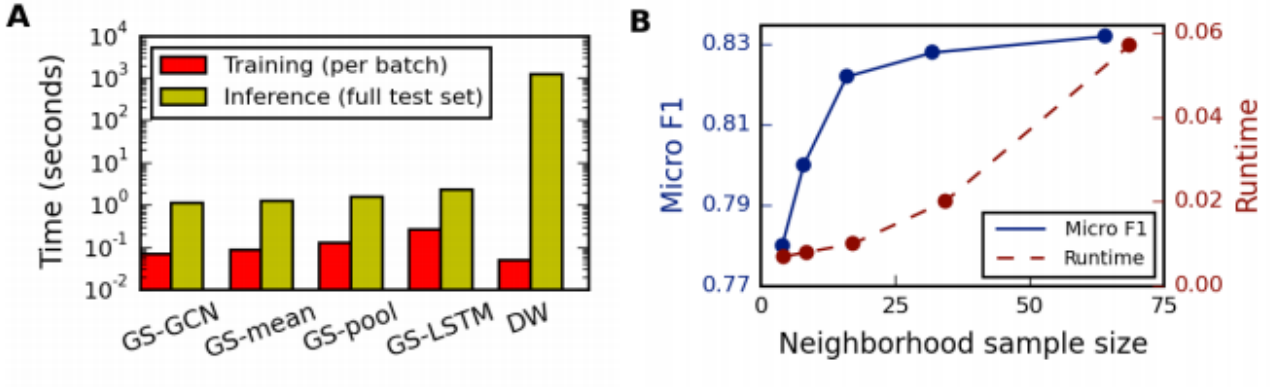


Figure 2: **A**: Timing experiments on Reddit data, with training batches of size 512 and inference on the full test set (79,534 nodes). **B**: Model performance with respect to the size of the sampled neighborhood, where the "neighborhood sample size" refers to the number of neighbors sampled at each depth for $K = 2$ with $S_1 = S_2$ (on the citation data using GraphSAGE-mean).

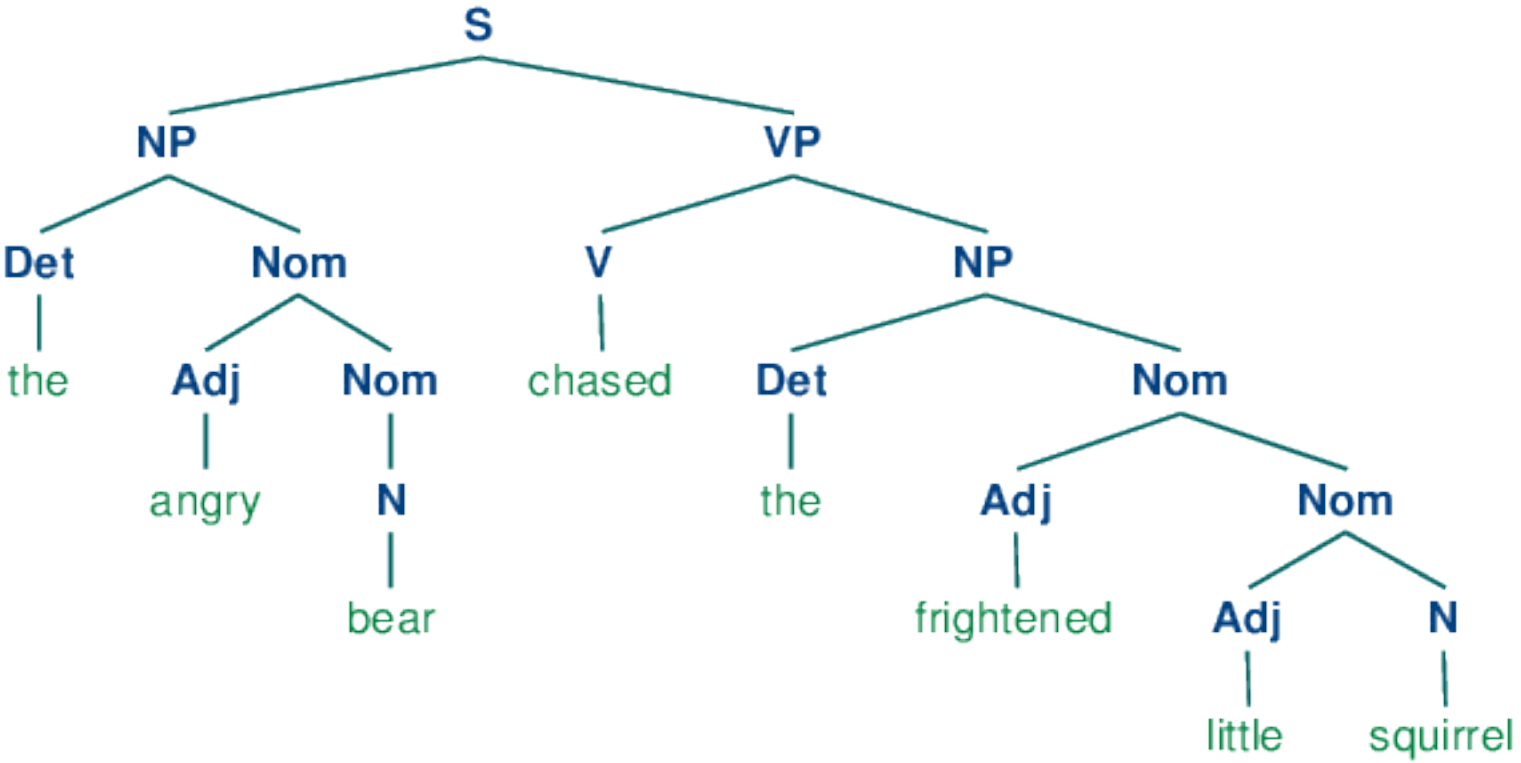
DATA MINING

/CISC 873 - Steven Ding
/GCN

- Part III: Network
  - ***Classification/Prediction Task over graph***
    - ***Each same is a graph!***

Example

- Natural Language

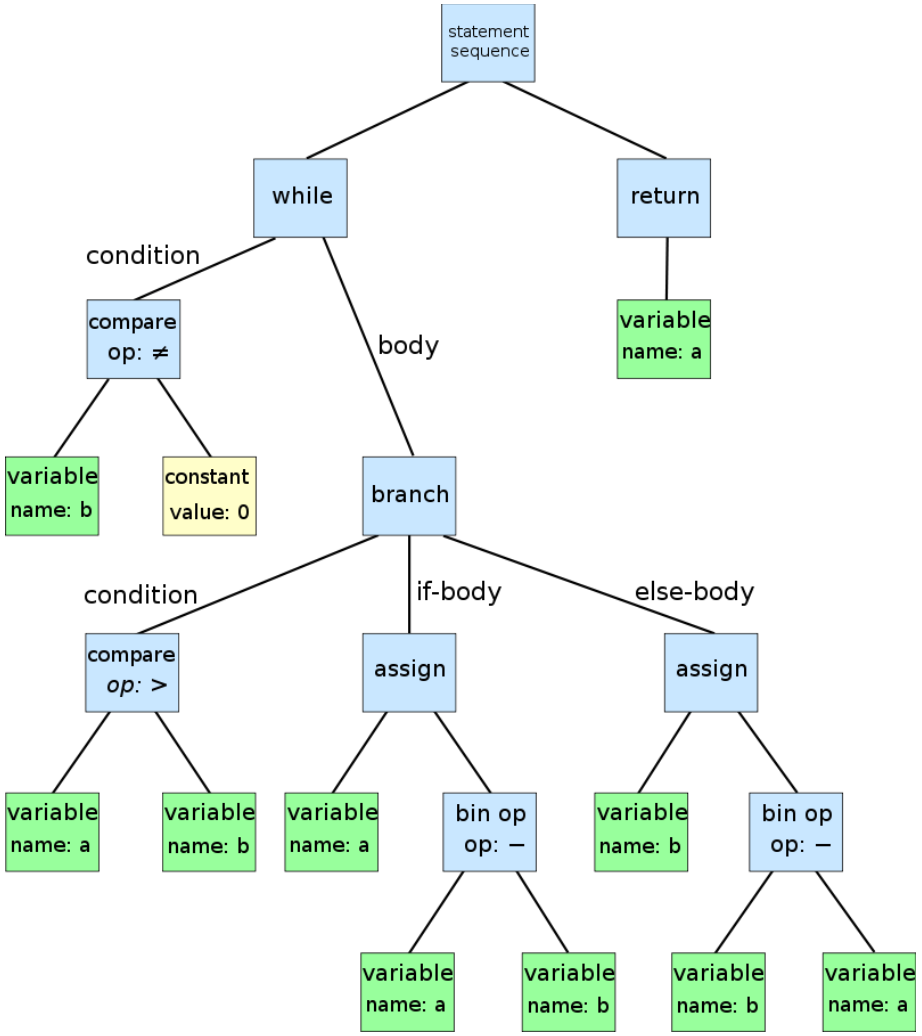


https://www.nltk.org/book/ch08.html

Example

- Source Code

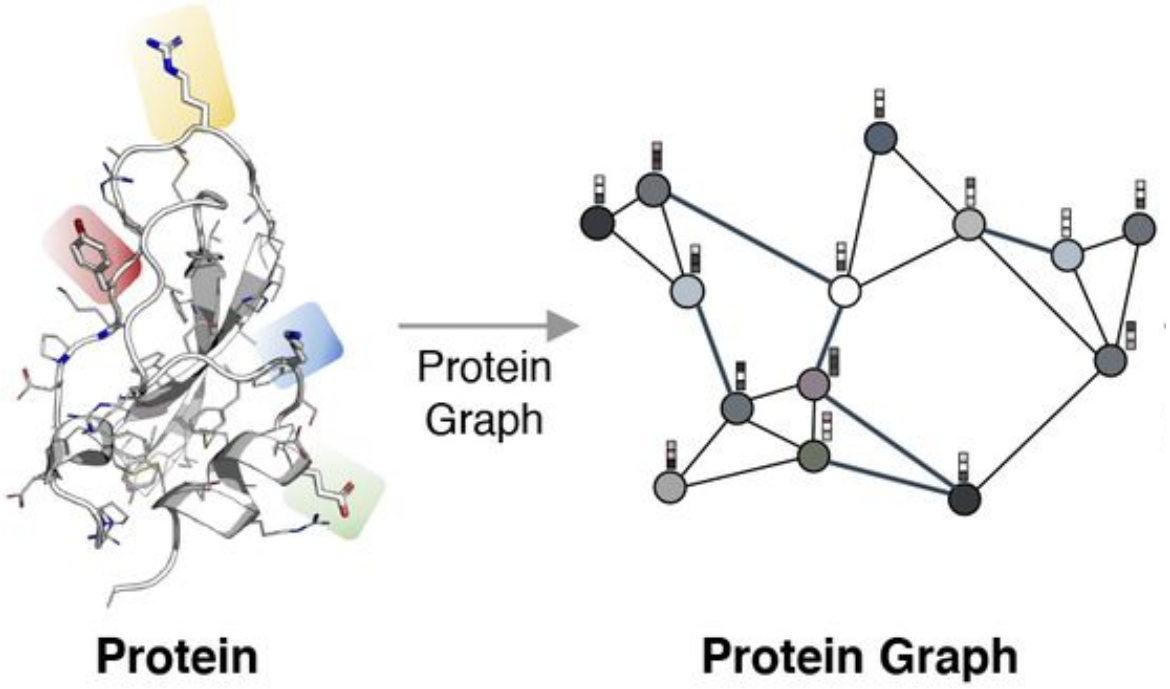
```
while b ≠ 0
    if a > b
       a := a − b
    else
        b := b − a
    return a
```



https://medium.com/@dinis.cruz/ast-abstract-syntax-tree-538aa146c53b

Example

- Protein



https://www.biorxiv.org/content/10.1101/2020.04.06.028266v1.full

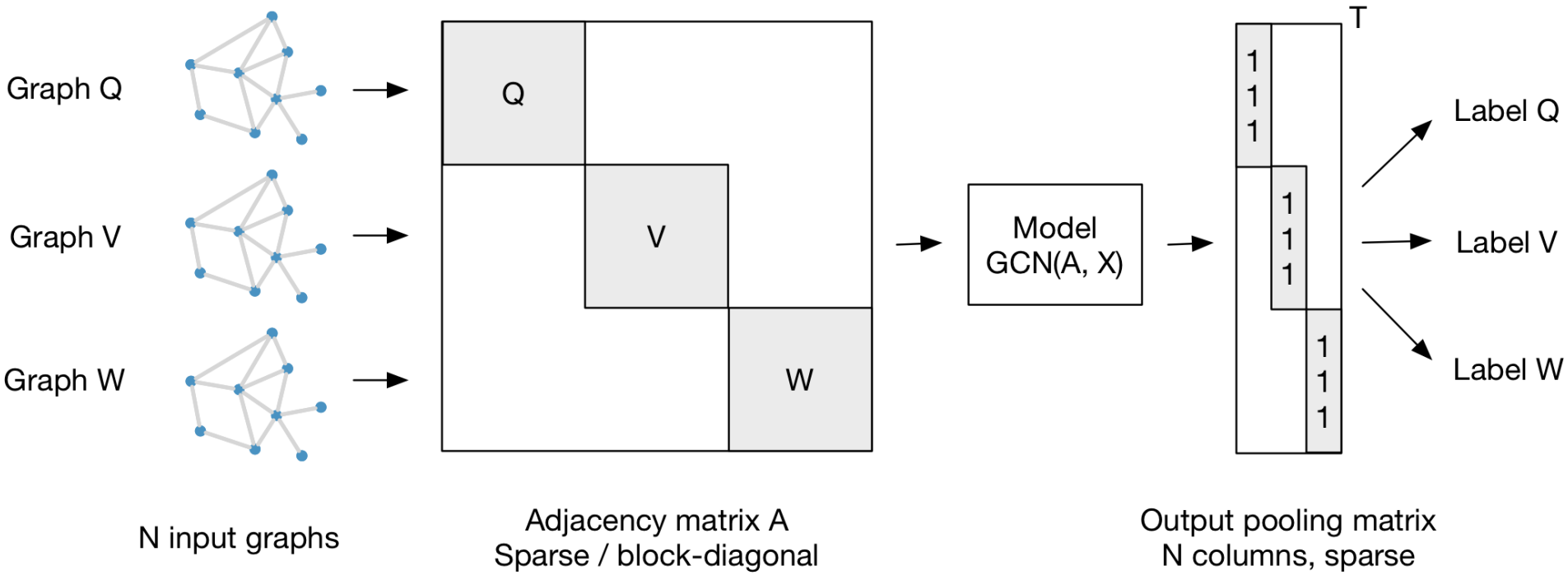
Problem

- Graph-level sample
  - Node originally encoded with some features
    - Manually engineer
    - Learned/Transformed features e.g.
      - Token -> embedding
      - Text -> RNN-ed / Attention-ed
      - Image -> Convoluted
  - BUT node has neighbors
  - GCN -> Better node representation by considering the neighbors

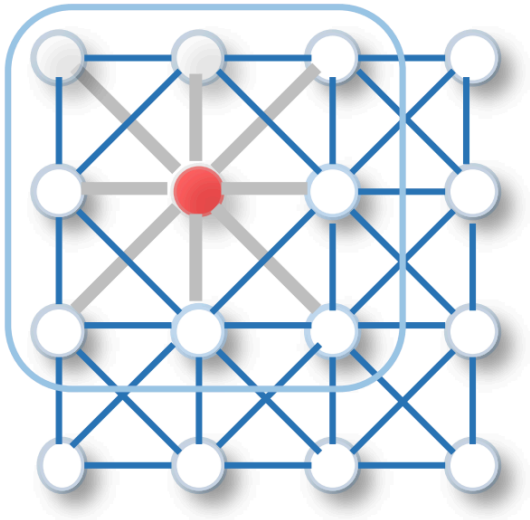
Batch Rep

Notation: $\mathcal{G} = (\mathbf{A}, \mathbf{X})$

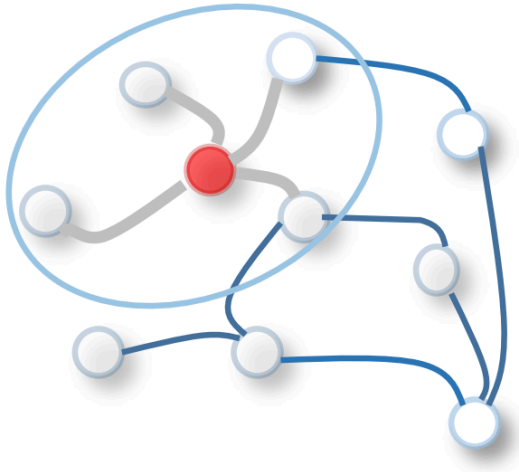
- Adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$
- Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F}$



https://github.com/tkipf/gcn



(a) 2D Convolution. Analogous to a graph, each pixel in an image is taken as a node where neighbors are determined by the filter size. The 2D convolution takes the weighted average of pixel values of the red node along with its neighbors. The neighbors of a node are ordered and have a fixed size.

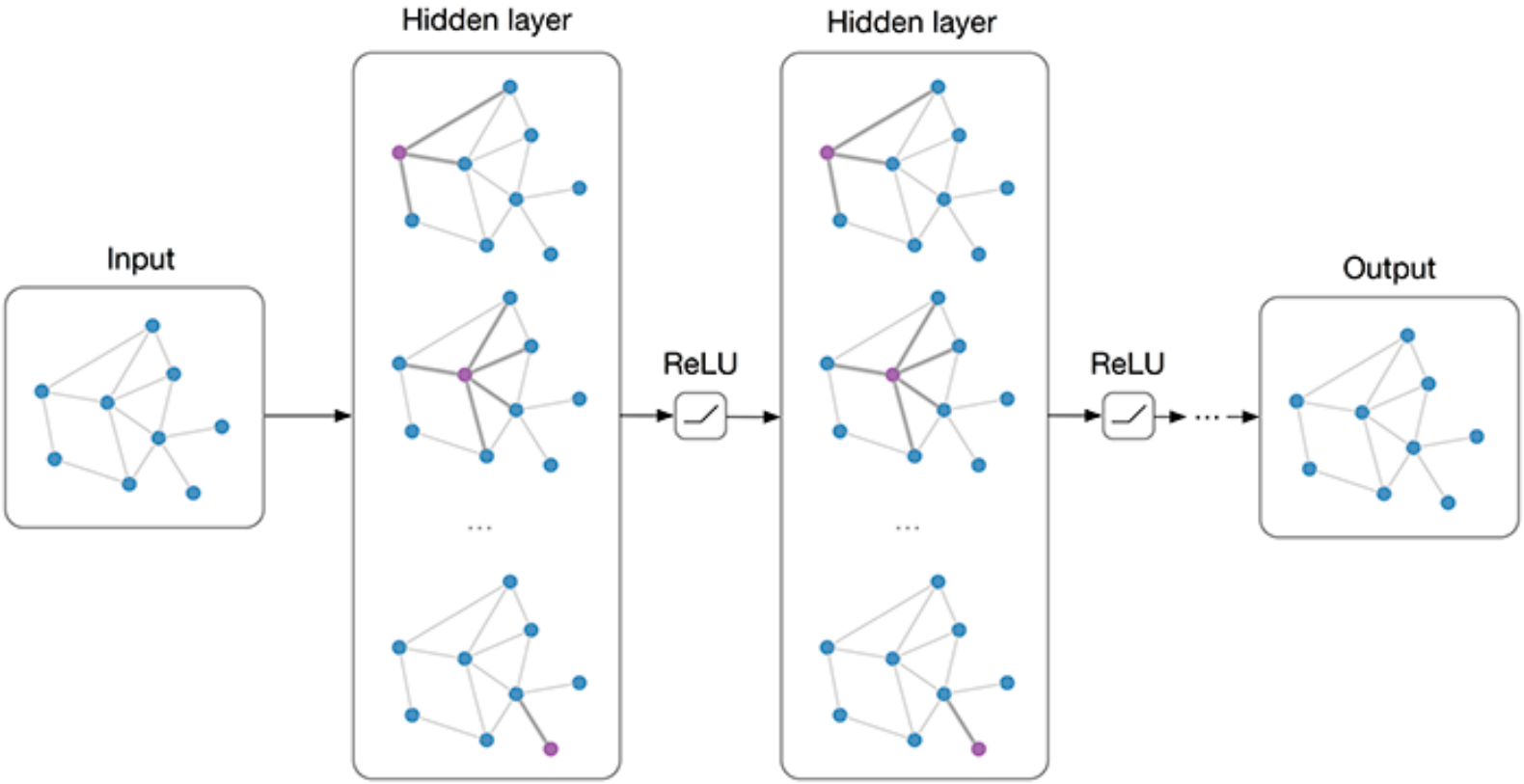


(b) Graph Convolution. To get a hidden representation of the red node, one simple solution of the graph convolutional operation is to take the average value of the node features of the red node along with its neighbors. Different from image data, the neighbors of a node are unordered and variable in size.

Fig. 1: 2D Convolution vs. Graph Convolution.

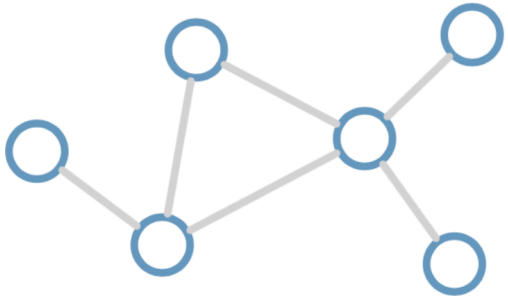
https://arxiv.org/pdf/1901.00596.pdf

GCN – `Messages` Passing

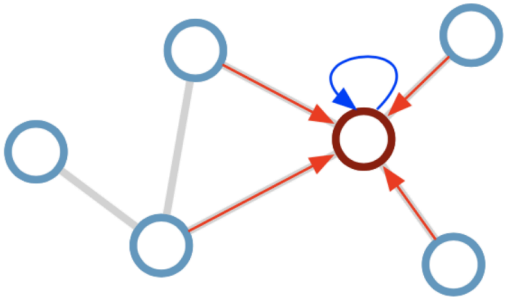


Single Layer – reduce_mean/sum/..

Consider this undirected graph:



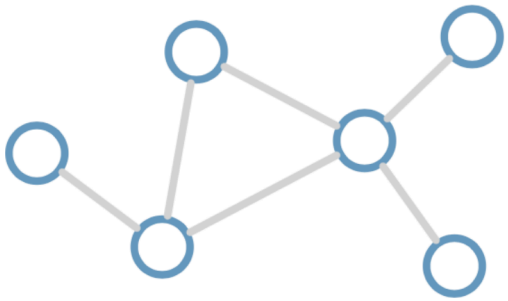
Calculate update for node in red:



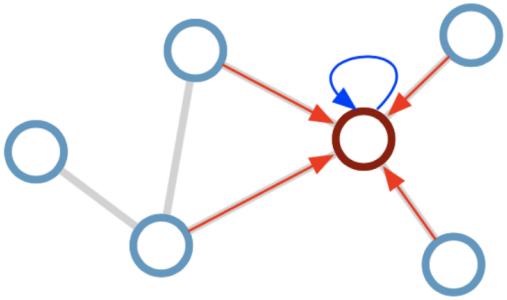
http://deeploria.gforge.inria.fr/thomasTalk.pdf

Single Layer – Dense Aggregation

Consider this undirected graph:



Calculate update for node in red:

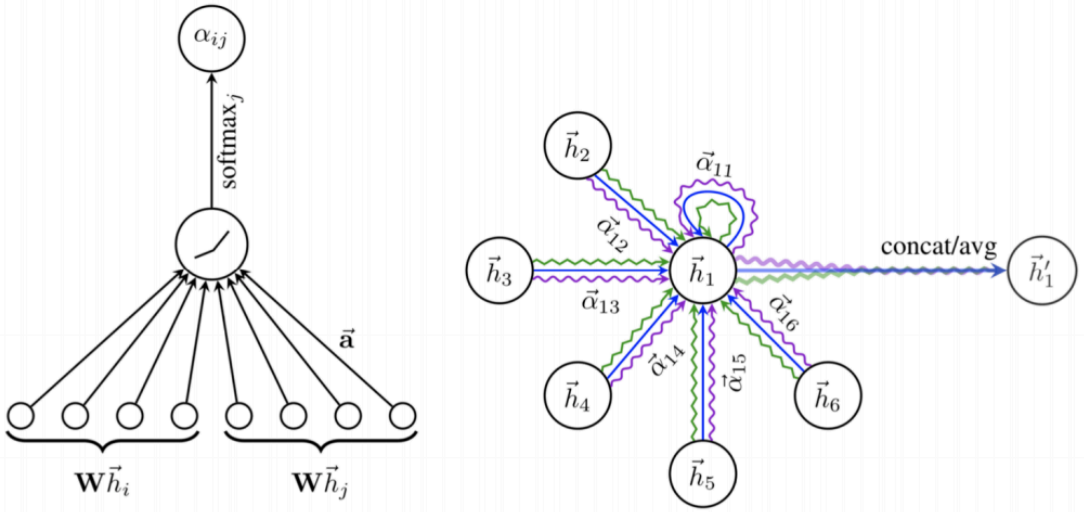


Update rule:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

http://deeploria.gforge.inria.fr/thomasTalk.pdf

Single Layer - Attention



[Figure from Veličković et al. (ICLR 2018)]

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right) \qquad \alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i\|\mathbf{W}\vec{h}_k]\right)\right)}$$

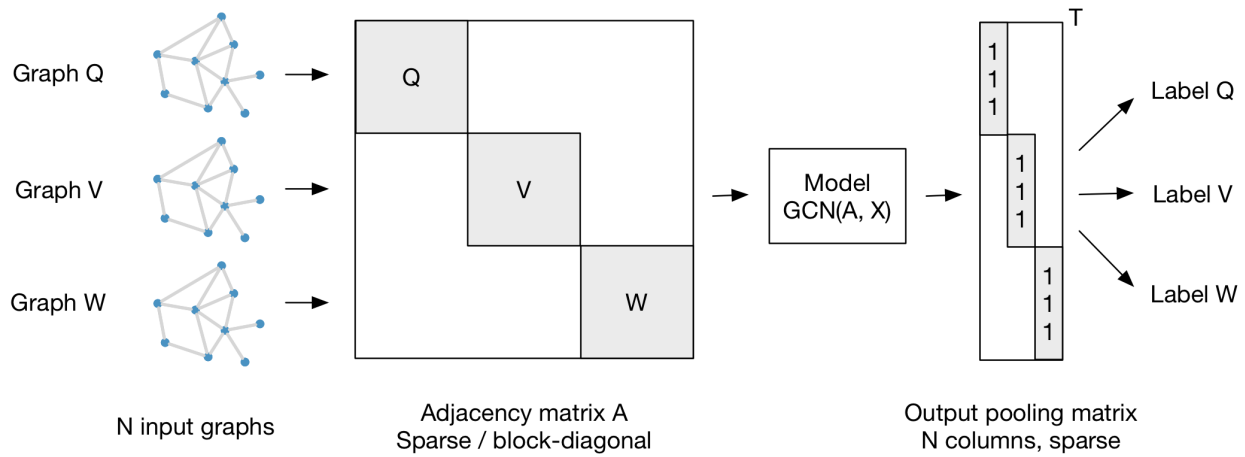
http://deeploria.gforge.inria.fr/thomasTalk.pdf

Example - Keras

```
import keras
from keras_gcn import GraphConv


DATA_DIM = 3

data_layer = keras.layers.Input(shape=(None, DATA_DIM))
edge_layer = keras.layers.Input(shape=(None, None))
conv_layer = GraphConv(
    units=32,
    step_num=1,
)([data_layer, edge_layer])
```



https://github.com/CyberZHG/keras-gcn

DATA MINING

/CISC 873 - History & Appreciation

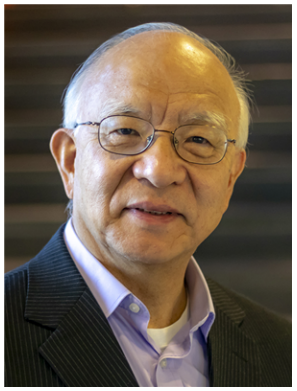
CISC 873 – Data Mining

Originally created by **Prof. David Skillicorn** in 2000

CISC 873 – Data Mining

- Materials

Dr. Jiawei Han
Univ. of Illinois at Urbana-Champaign



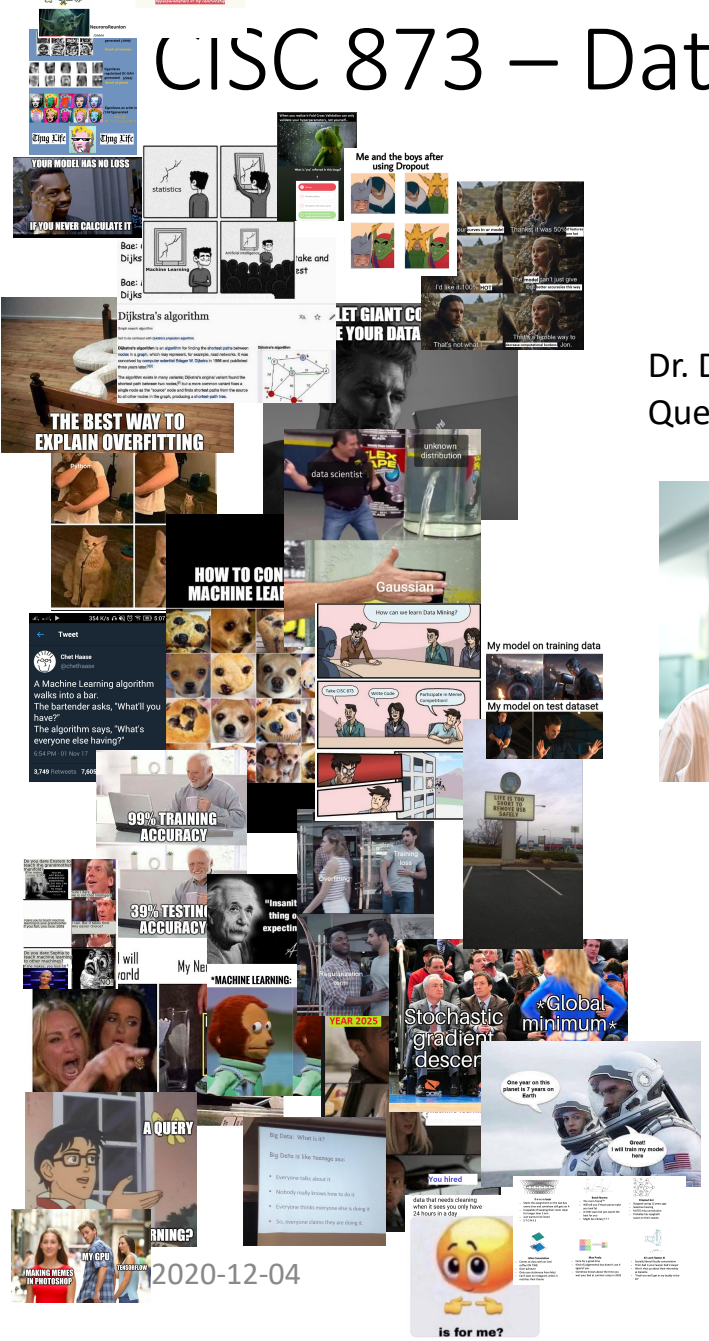
Dr. Pei Jian
Simon Fraser University

Dr. David Skillicorn
Queen's Computing

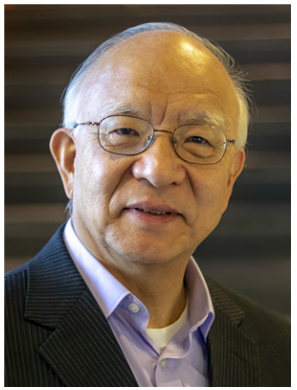
Dr. Benjamin C. M. Fung
McGill University

GLIS 873 2020

CISC 873 – Data Mining



Dr. Jiawei Han
Univ. of Illinois at Urbana-Champaign



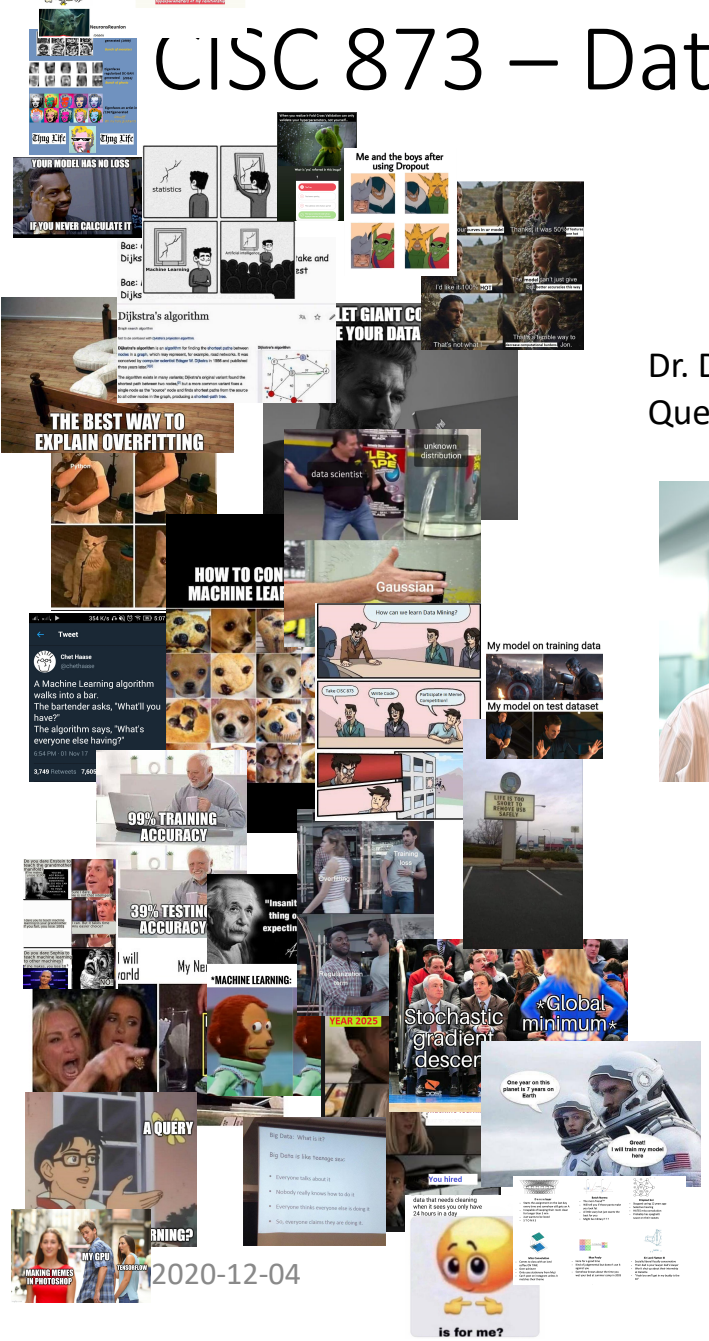
Dr. Pei Jian
Simon Fraser University

Dr. David Skillicorn
Queen's Computing

Dr. Benjamin C. M. Fung
McGill University

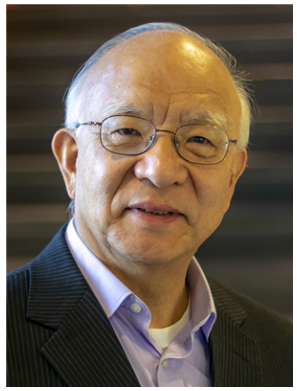
GLIS 873 2020

CISC 873 – Data Mining



Dr. Pei Jian
Simon Fraser University

Dr. Jiawei Han
Univ. of Illinois at Urbana-Champaign



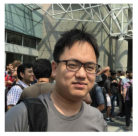
Dr. David Skillicorn
Queen's Computing

Dr. Benjamin C. M. Fung
McGill University

GLIS 873 2020


WeihanO


Li Tao Li


Gitsamshi


Sal Choueib


strixy16