# 网络科学第三讲

Network Embedding

2019版

主讲教师：罗铁坚
电子邮箱：tjluo@ucas.ac.cn

中国科学院大学

# 提纲

- 三篇文章介绍
- Structure Preserving Network（2007）
- A Survey on Network Embedding(2017)
- A Tutorial on Network Embeddings(2018)

# 为什么要embedding

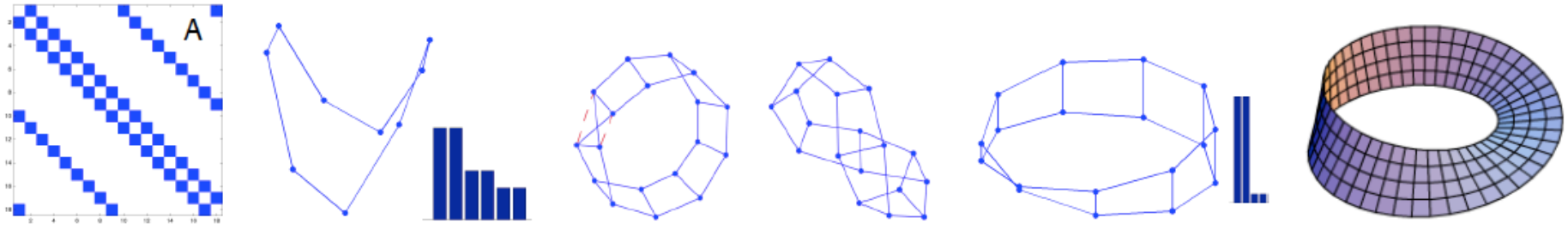



*Figure 1.* Embedding the classical Mobius Ladder Graph. Given the adjacency matrix (left), the visualizations produced by spectral embedding and spring embedding (middle) do not accurately capture the graph topology. The SPE embedding is compact and topologically correct.

---

## Structure Preserving Embedding

---

**Blake Shaw** BLAKE@CS.COLUMBIA.EDU
**Tony Jebara** JEBARA@CS.COLUMBIA.EDU
Department of Computer Science, Columbia University, 1214 Amsterdam Ave, New York, NY 10027

# 两种Embedding

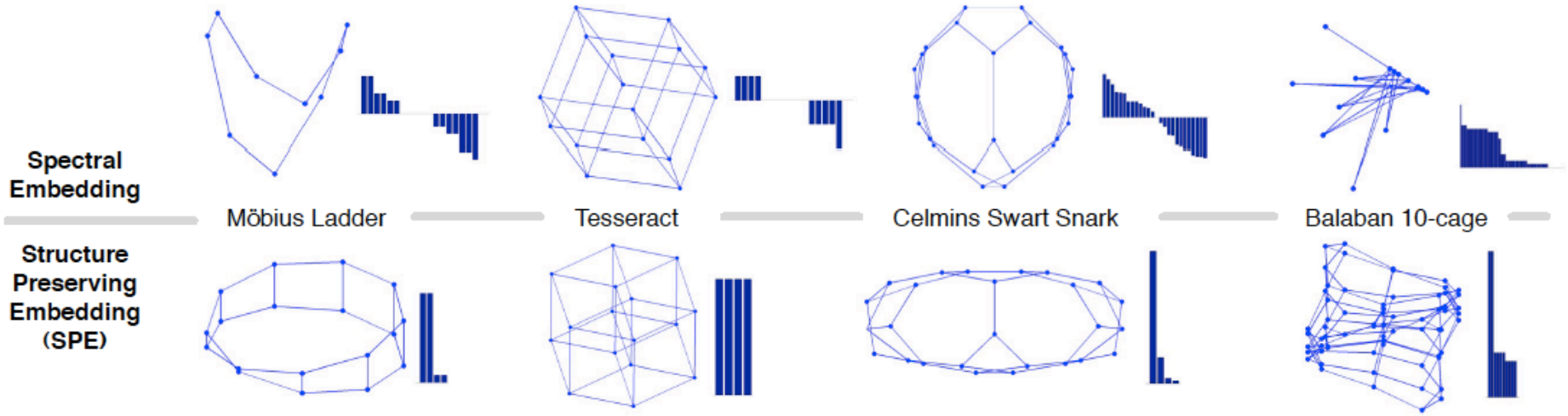



*Figure 2.* Classical graphs embedded with spectral embedding (above), and SPE w/ kNN (below). Eigenspectra are shown to the right. SPE finds a small number of dimensions that highlight many of the symmetries of these graphs.

# SPE 应用

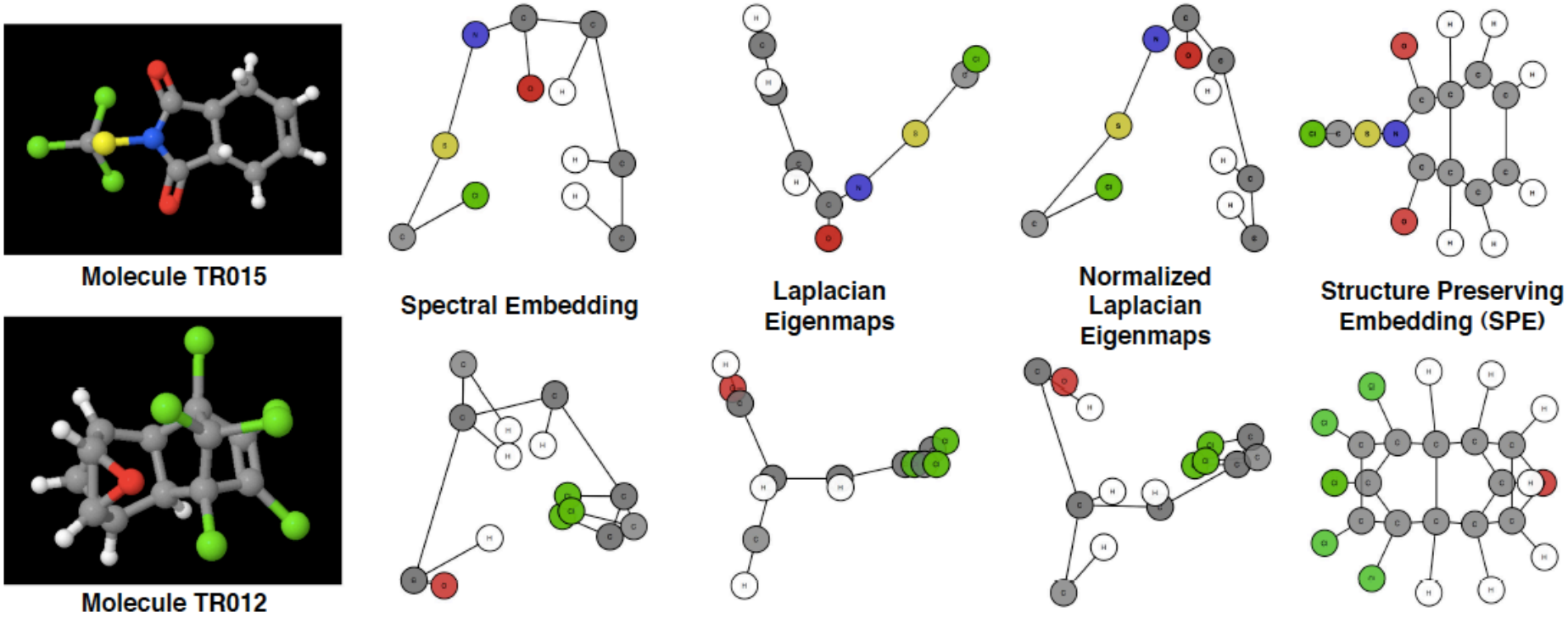


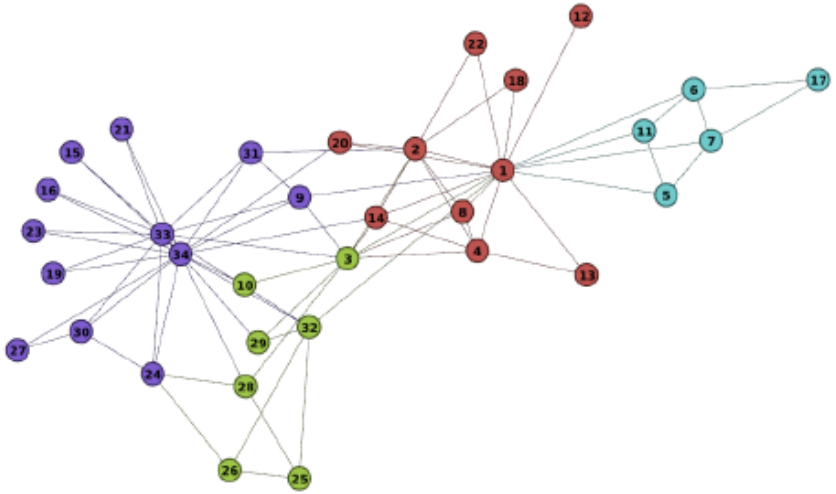

*Figure 4.* Two comparisons of molecule embeddings (top row and bottom). The SPE w/kNN embedding (right) more closely resembles the true physical embedding of the molecule (left), despite being given only connectivity information.

# A Tutorial on Network Embeddings

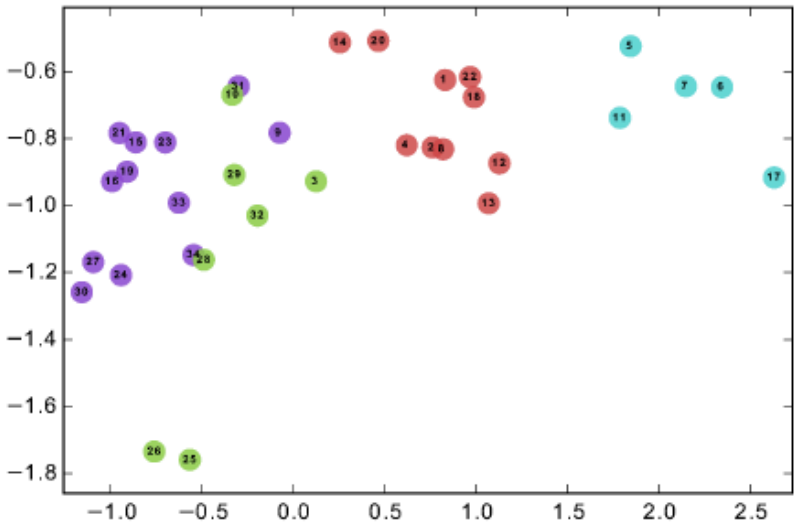
Haochen Chen[1], Bryan Perozzi[2], Rami Al-Rfou[2], and Steven Skiena[1]

[1]Stony Brook University
[2]Google Research
{haocchen, skiena}@cs.stonybrook.edu, bperozzi@acm.org, rmyeid@google.com

August 9, 2018

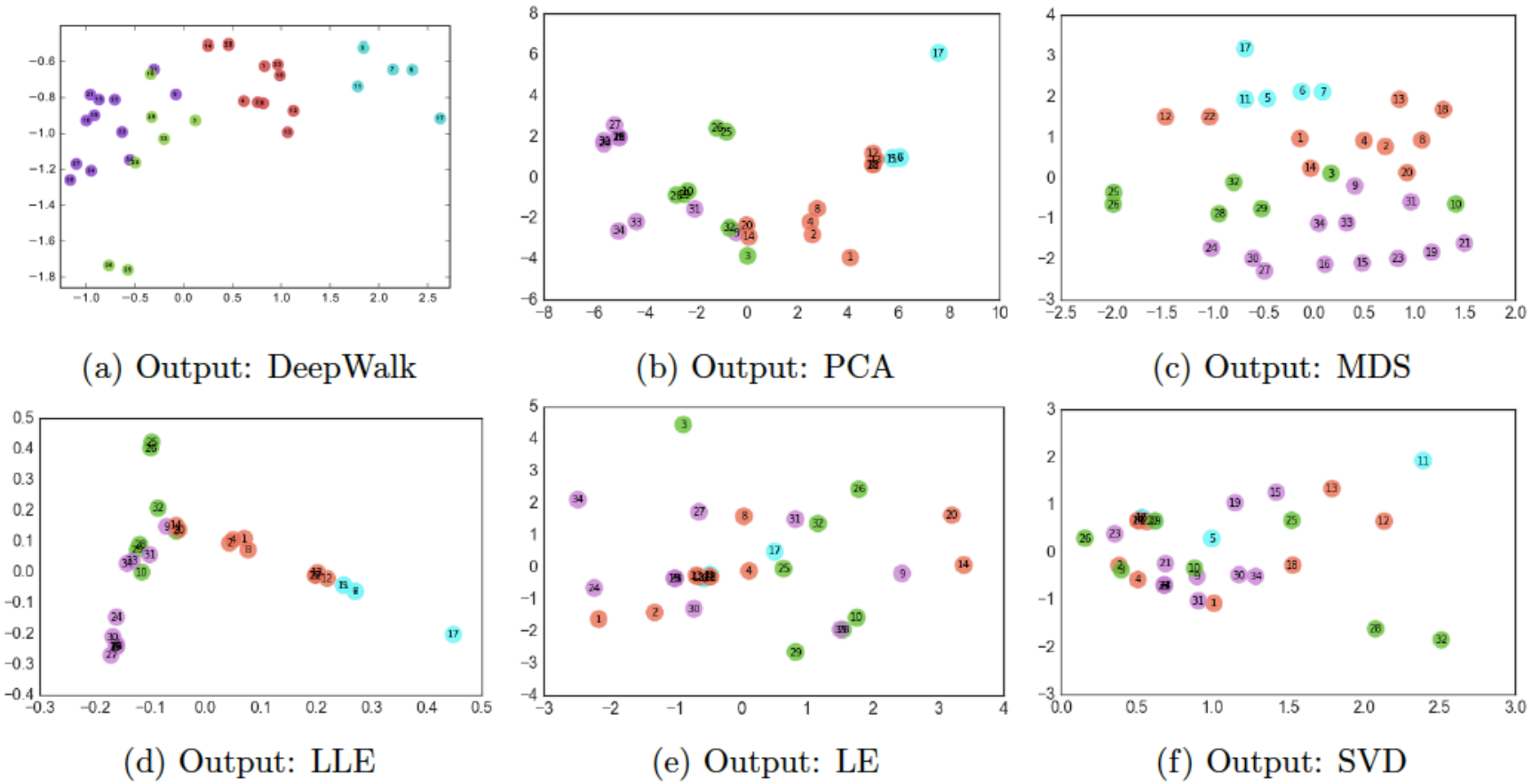


(a) Input: Karate Graph




(b) Output: Network Embedding

# 6种Network Embedding 对比



(a) Output: DeepWalk (b) Output: PCA (c) Output: MDS

(d) Output: LLE (e) Output: LE (f) Output: SVD

Figure 2: Two-dimensional embeddings of the Karate graph using DeepWalk and several early dimension reduction techniques. The input is the adjacency matrix for DeepWalk and SVD, and the geodesic matrix for the other four methods.

# Deep Learning On Graph 的设计选择

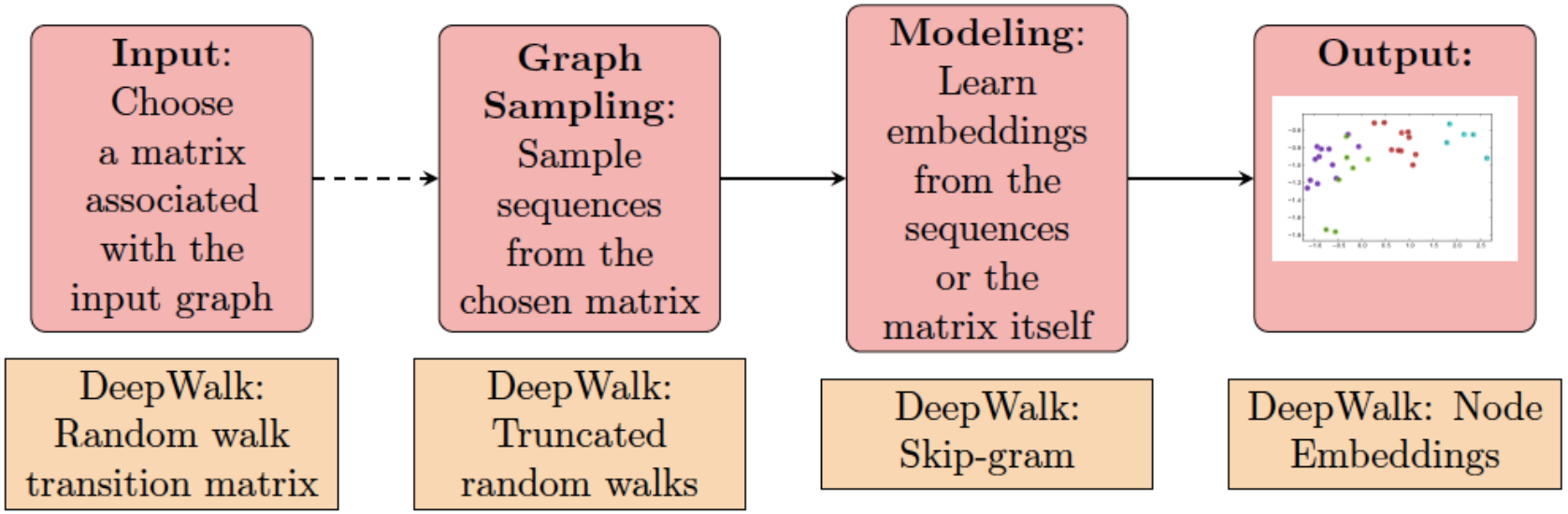



Figure 3: A paradigm for *Deep Learning On Graphs* with DeepWalk's design choice for each building block.

- **Definition 1** *(graph)* A simple undirected **graph** $G = (V, E)$ is a collection $V$ of $n$ vertices $v_1, v_2, \cdots, v_n$ together with a set $E$ of edges, which are *unordered* pairs of the vertices. In other words, the edges in an undirected graph have no orientation.

  The adjacency matrix $A$ of $G$ is an $n \times n$ matrix where $A_{ij} = 1$ if there is an edge between $v_i$ and $v_j$, and $A_{ij} = 0$ otherwise. Unless otherwise stated, we use both graph and network to refer to a simple undirected graph.

- **Definition 2** *(network embedding)* For a given a network $G$, a **network embedding** is a mapping function $\Phi : V \mapsto \mathbb{R}^{|V| \times d}$, where $d \ll |V|$. This mapping $\Phi$ defines the latent representation (or *embedding*) of each node $v \in V$. Also, we use $\Phi(v)$ to denote the embedding vector for node $v$.

- **Definition 3** *(directed graph)* A **directed graph** $G = (V, E)$ is a collection $V$ of $n$ vertices $v_1, v_2, \cdots, v_n$ together with a set $E$ of edges, which are *ordered* pairs of the vertices. The only difference between a directed graph and an undirected graph is that the edges in a directed graph have orientation.

- **Definition 4** *(heterogeneous network)* A **heterogeneous network** is a network $G = (V, E)$ with multiple types of nodes or multiple types of edges. Formally, $G$ is associated with a node type mapping $f_v : v \to O, \forall v \in V$ and an edge type mapping $f_e : e \to Q, \forall e \in E$, where $O$ is the set of all node types and $Q$ is the set of all edge types.

- **Definition 5** *(signed graph)* A **signed graph** is a graph where each edge $e \in E$ is associated with a weight $w(e) \in \{-1, 1\}$. An edge with weight of 1 denotes a positive link between nodes, whereas an edge with weight of -1 denotes a negative link. Signed graphs can be used to reflect agreement or trust.

# DeepWalk 算法

**Algorithm 1** DeepWalk($G$, $w$, $d$, $\gamma$, $t$)

**Input:** network $G(V, E)$
window size $w$
embedding size $d$
walks per vertex $\gamma$
walk length $t$

**Output:** matrix of vertex representations $\Phi \in \mathbb{R}^{|V| \times d}$

Initialization: Sample $\Phi$ from $\mathcal{U}^{|V| \times d}$
Build a binary Tree $T$ from $V$
**for** $i = 0$ to $\gamma$ **do**
  $\mathcal{O} = \text{Shuffle}(V)$
  **for each** $v_i \in \mathcal{O}$ **do**
    $\mathcal{W}_{v_i} = RandomWalk(G, v_i, \text{t})$
    SkipGram($\Phi$, $\mathcal{W}_{v_i}$, $w$)
  **end for**
**end for**

# 八种方法的特点

| Method | Source of Context Nodes | Embedding Learning Method |
|---|---|---|
| DeepWalk [33] | Truncated Random Walks | Skip-gram with Hierarchical Softmax |
| LINE [41] | 1-hop and 2-hop Neighbors | Skip-gram with Negative Sampling |
| Node2vec [21] | Biased Truncated Random Walks | Skip-gram with Negative Sampling |
| Walklets [34] | $A^i$ where $i = 1, 2, \cdots, k$ | Skip-gram with Hierarchical Softmax |
| GraRep [5] | $A^i$ where $i = 1, 2, \cdots, k$ | Matrix Factorization |
| GraphAttention [2] | $A^i$ where $i = 1, 2, \cdots, k$ | Graph Likelihood |
| SDNE [48] | 1-hop and 2-hop Neighbors | Deep Autoencoder |
| DNGR [6] | Random surfing | Stacked Denoising Autoencoder |

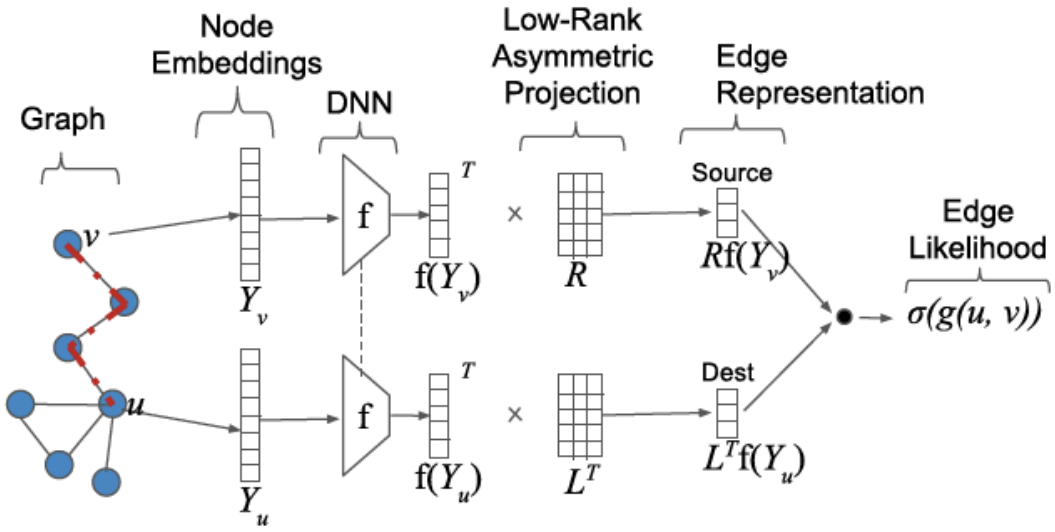
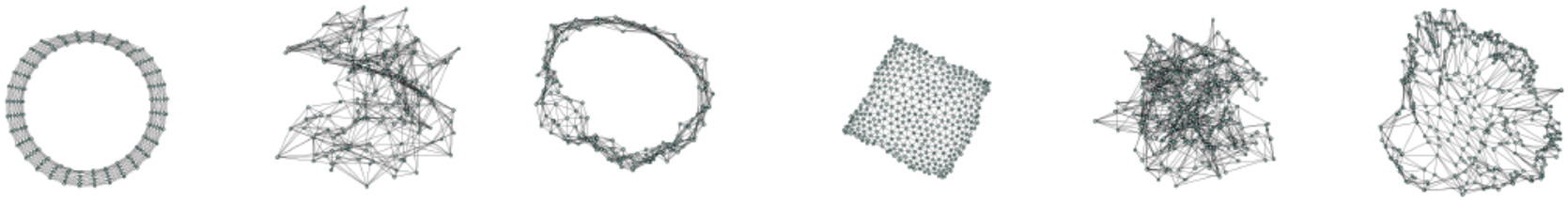



Figure 5: Depiction of the edge representation method in [1]. On the left: a graph, showing a random walk in dotted-red, where nodes $u, v$ are close in the walk (i.e. within a configurable context window parameter). Their method accesses the trainable embeddings $Y_u$ and $Y_v$ for the nodes and feed them as input to Deep Neural Network (DNN) $f$. The DNN outputs manifold coordinates $f(Y_u)$ and $f(Y_v)$ for nodes $u$ and $v$, respectively. A low-rank asymmetric projection transforms $f(Y_u)$ and $f(Y_v)$ to their source and destination representations, which are used by $g$ to represent an edge.

[1] Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. Learning edge representations via low-rank asymmetric projections. CIKM '17, 2017.

# 比较几种方面的效果



(a) Can_187 (b) LINE (c) HARP (d) Poisson 2D (e) LINE (f) HARP

Figure 6: Comparison of two-dimensional embeddings from LINE and HARP, for two distinct graphs. Observe how HARP's embedding better preserves the higher order structure of a ring and a plane.

# Network Embedding 的应用

知识表示

### 5.1 Knowledge Representation

The problem of knowledge representation is concerned with encoding facts about the world using short sentences (or tuples) composed of subjects, predicates, and objects. While it can be viewed as strictly as a heterogeneous network, it is an important enough application area to mention here in its own right:

- GenVector [58] studies the problem of learning social knowledge graphs, where the goal is to connect online social networks to knowledge bases. Their multi-modal Bayesian embedding model utilizes DeepWalk for generating user representations in social networks.
- RDF2Vec [38] is an approach for learning latent entity representations in Resource Description Framework (RDF) graphs. RDF2Vec first converts RDF graphs into sequences of graph random walks and Weisfeiler-Lehman graph kernels, and then adopt CBOW and Skip-gram models on the sequences to build entity representations.

# 推荐系统

## 5.2 Recommender Systems

Another branch of work attempts to incorporate network embeddings into recommender systems. Naturally, the interactions between users, users' queries and items altogether form a heterogeneous network which encodes the latent preferences of users over items. Network embedding on such interaction graphs could serve as an enhancement to recommender systems.

- Chen et al. [8] exploit the usage of social listening graph to enhance music recommendation models. They utilize DeepWalk to learn latent node representations in the social listening graph, and incorporate these latent representations into factorization machines.
- Chen et al. [9] propose Heterogeneous Preference Embedding to embed user preference and query intention into low-dimensional vector space. With both user preference embedding and query embedding available, recommendations can be made based on the similarity between items and queries.

# 自然语言处理

## 5.3 Natural Language Processing

State-of-the-art network embedding methods are mostly inspired by advances in the field of natural language processing, especially neural language models. At the same time, network embedding methods also lead to better modeling of human language.

- PLE [37] studies the problem of label noise reduction in entity typing. Their model jointly learns the representations of entity mentions, text features and entity types in the same feature space. These representations are further used to estimate the type-path for each training example.
- CANE [45] is a context-aware network embedding framework. They argue that one node may exhibit different properties when interacting with different neighbors, thus its embedding with respect to these neighbors should be different. CANE achieves this goal by employing mutual attention mechanism.
- Fang et al. [18] propose a community-based question answering (cQA) framework which leverages the social interactions in the community for better question-answering matching. Their framework treats users, questions and answers and the interactions between them as a heterogeneous network and trains a deep neural network on random walks in the network.
- Zhao et al. [65] study the problem of expert finding in community-based question answering (cQA) site. Their method adopts the random-walk method in DeepWalk for embedding social relations between users and RNNs for modeling users' relative quality rank to questions.

## 5.4 Social Network Analysis

Social networks are prevailing in the real world, and it is not suprising that network embedding methods have become popular in social network analysis. Network embeddings on social network have prove to be powerful features for a wide spectrum of applications, leading to improved performance on a lot of downstream tasks.

- Perozzi et al. [35] study the problem of predicting the exact age of users in social networks. They learn the user representations in social networks with DeepWalk, and adopts linear regression on these user representations for age prediction.

- Yang et al. [56] propose a neural network model for modeling social networks and mobile trajectories simultaneously. They adopt DeepWalk to generate node embeddings in social networks and the RNN and GRU models for generating mobile trajectories.

- Dallmann et al. [16] show that by learning Wikipedia page representations from both the Wikipedia link network and Wikipedia click stream network with DeepWalk, they can obtain concept embeddings of higher quality compared to counting-based methods on the Wikipedia networks.

- Liu et al. [28] propose Input-output Network Embedding (IONE), which use network embeddings to align users across different social networks. IONE achieves this by preserving the proximity of users with similar followers and followees in a common embedding space.

- Chen and Skiena [14] demonstrate the efficacy of network embedding methods in measuring similarity between historical figures. They construct a network between historical figures from the interlinks between their Wikipedia pages, and use DeepWalk to obtain vector representations of historical figures. It is shown that the similarity between the DeepWalk representations of historical figures can be used as an effctive decent similarity measurement.

# 其他应用

## 5.5 Other Applications

- Geng et al. [19] and Zhang et al. [62] develop deep neural network models which learns distributed representations of both users and images from an user-image co-occurrence network. The representation learning process in the network is analogous to that of DeepWalk [33], except that they also incorporate image features extracted with a DCNN into the optimization process.
- Wu et al. [52] treat the click data collected from users' searching behavior in image search engines as a heterogeneous graph. The nodes in the click graph are text queries and images returned as search results, while the edges indicates the click count of an image given a search query. By proposing a neural network model based on truncated random walks, their method learns multimodal representations of text and images, which are shown to boost cross-modal retrieval performance on unseen queries or images.
- Zhang et al. [63] apply DeepWalk to large-scale social image-tag collections to learn both image features and word features in a unified embedding space.

# 结论和未来方向

## 6 Conclusions and Future Directions

Network embedding is an exciting and rapidly growing research area which attracts researchers from various communities, especially data mining, machine learning and natural language processing. While most work concerned about general methods for network embedding, we argue that the applications of network embedding is even more underresearched. We anticipate a large body of work on additional applications of network embeddings, such as improving the performance of natural language processing and information retrieval models, mining biology network and social networks, to name a few.

Also, much work has been done for graphs which possess different properties and from different domains. In terms of graph properties, various methods are proposed for directed graphs, signed graphs, heterogeneous graphs and attributed graphs. In terms of application domains, network embedding methods are applied to a wide spectrum of graphs including knowledge graphs, biology graphs and social networks. However, doubtlessly much more work can be done on this front by exploiting the unique characteristics of these graphs.

## 6.1 The search for the right context

Inspired by the two-phase network embedding learning framework presented in DeepWalk, various strategies have been proposed for searching for the right context, as discussed in Table 1. However, most of these strategies relies on a rigid definition of context nodes identical for all networks, which is not desirable.

Under this background, there is much effort recently on unifying different network embedding under a general framework [13, 36]. GEM-D [13] decomposes graph embedding algorithms into three building blocks: node proximity function, warping function and loss function. They show that algorithms such as Laplacian Eigenvectors, DeepWalk, LINE, and node2vec can all be unified under this framework. By testing different design choices for each building block on real-world graphs, they pick the triple which works the best empirically: the combination of the finite-step transition matrix, exponential warping function and warped Frobenius norm loss. However, such design decisions are purely made based on models' empirical performance on a limited number of networks, which may not work well for all networks.

A promising approach is the attention model recently proposed in GraphAttention[2]. By parameterizing the attention over the power series of the transition matrix, GraphAttention automatically learns different attention parameters for different networks.

## 6.2 Improved Losses / Optimization Models

Another issue with the neural embedding methods is their dependence upon general loss functions and optimization models, such as Skip-gram. These optimization goals and models are not tuned for any particular task. As a result, though the learned network embeddings have been proven to achieve competitive performance on a variety of tasks such as node classification and link prediction, they are suboptimal when compared with end-to-end embeddings methods designed specifically for a task.

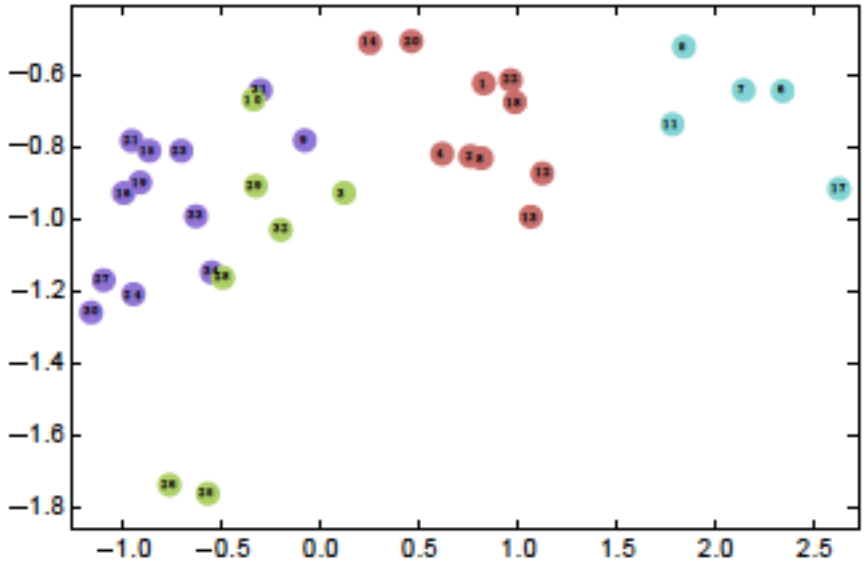
Thus, another future direction for network embedding algorithms is to design loss functions

# A Survey on Network Embedding

**Peng Cui[1], Xiao Wang[1], Jian Pei[2], Wenwu Zhu[1]**
[1]Department of Computer Science and Technology, Tsinghua University, China
[2]School of Computing Science, Simon Fraser University, Canada
cuip@tsinghua.edu.cn, wangxiao007@mail.tsinghua.edu.cn,
jpei@cs.sfu.ca, wwzhu@tsinghua.edu.cn

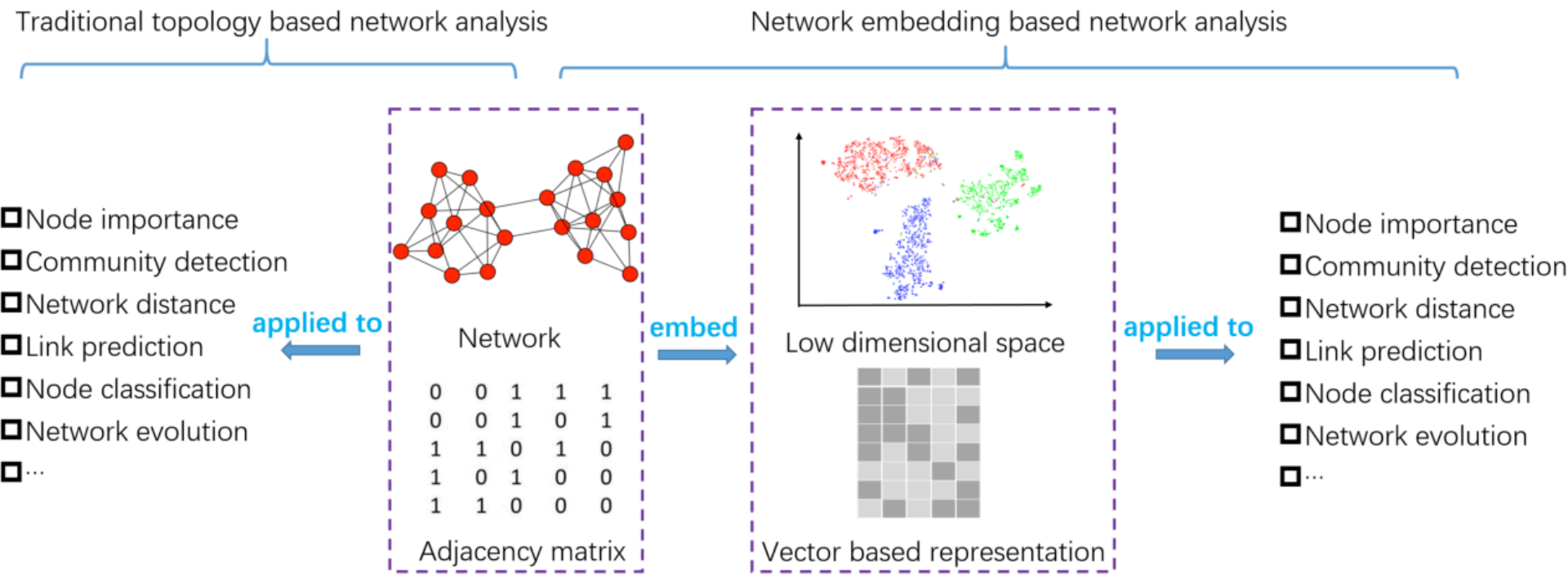
# 跆拳道俱乐部成员网络分析

(a) Input: karate network




(b) Output: representations

# 2种不同网络建模和分析方法的对比



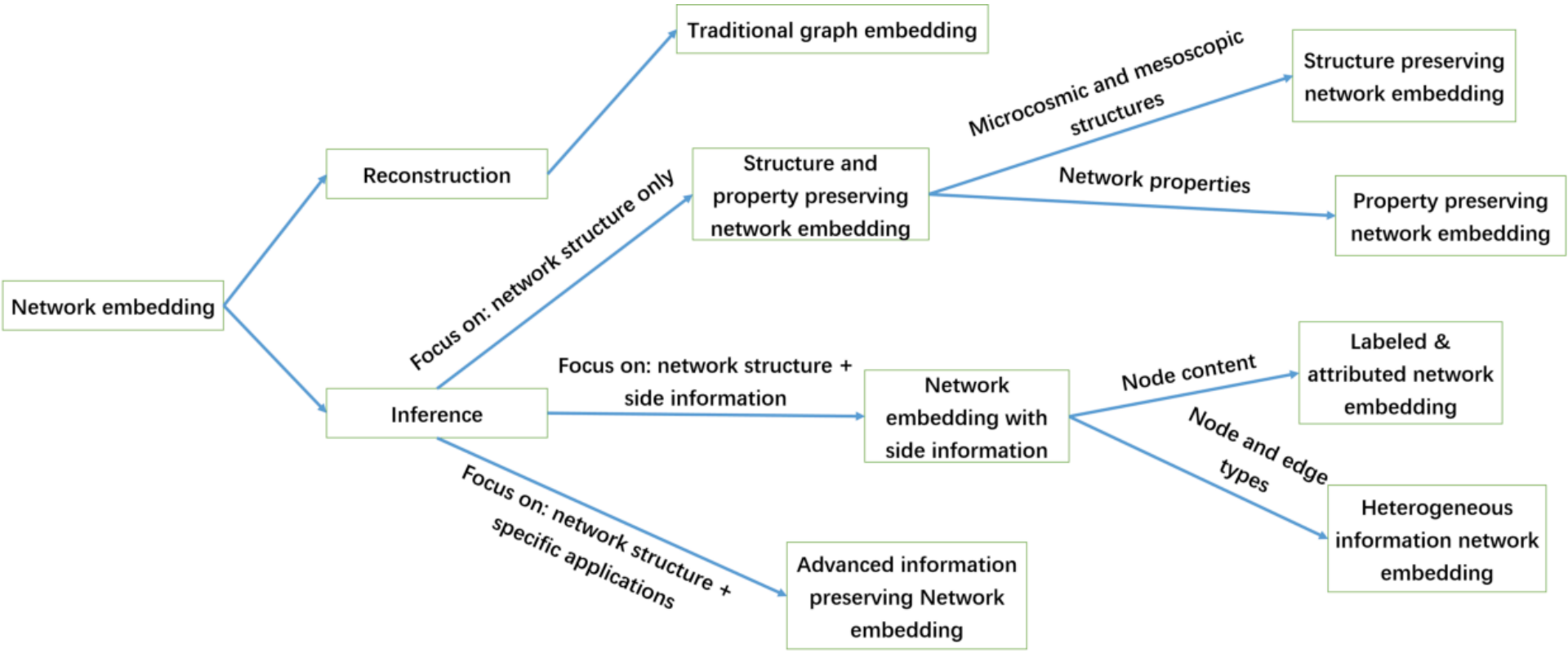
# 现实世界网络特点

| networks | structure and property preserving network embedding | network embedding with side information | classification | link prediction | clustering | visualization |
|---|---|---|---|---|---|---|
| BLOGCATALOG | √ | | √ | √ | √ | √ |
| FLICKR | √ | | √ | √ | √ | √ |
| YOUTUBE | √ | | √ | √ | √ | √ |
| Twitter | √ | | | √ | | |
| DBLP | √ | √ | √ | √ | √ | √ |
| Cora | √ | √ | √ | √ | √ | √ |
| Citeseer | √ | √ | √ | √ | √ | √ |
| ArXiv | √ | | | √ | | |
| Wikipedia | √ | | √ | √ | √ | √ |
| PPI | √ | | | √ | | |

# Network Embedding 方法的代码和实例

| Structure and property preserving network embedding | |
|---|---|
| Methods | Source code |
| DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) | https://github.com/phanein/deepwalk |
| LINE (Tang et al. 2015) | https://github.com/tangjianpku/LINE |
| GraRep (Cao, Lu, and Xu 2015) | https://github.com/ShelsonCao/GraRep |
| SDNE (Wang, Cui, and Zhu 2016) | http://nrl.thumedia.org/structural-deep-network-embedding |
| Node2vec (Grover and Leskovec 2016) | https://github.com/aditya-grover/node2vec |
| DNGR (Cao, Lu, and Xu 2016) | https://github.com/ShelsonCao/DNGR |
| M-NMF (Wang et al. 2017b) | http://nrl.thumedia.org/community-preserving-network-embedding |
| GED (Chen et al. 2017) | https://users.ece.cmu.edu/~sihengc/publications.html |
| Ou *et al.* (Ou et al. 2015) | http://nrl.thumedia.org/non-transitive-hashing-with-latent-similarity-components |
| HOPE (Ou et al. 2016) | http://nrl.thumedia.org/asymmetric-transitivity-preserving-graph-embedding |
| **Network embedding with side information** | |
| Methods | Source code |
| MMDW (Tu et al. 2016) | https://github.com/thunlp/mmdw |
| TADW (Yang et al. 2015) | https://github.com/thunlp/tadw |
| TriDNR (Pan et al. 2016) | https://github.com/shiruipan/TriDNR |
| **Advanced information preserving network embedding** | |
| Methods | Source code |
| Information diffusion (Bourigault et al. 2014) | https://github.com/ludc/social_network_diffusion_embeddings |
| Cascade prediction (Li et al. 2017) | https://github.com/chengli-um/DeepCas |
| Anomaly detection (Hu et al. 2016) | https://github.com/hurenjun/EmbeddingAnomalyDetection |
| Collaboration prediction (Chen and Sun 2017) | https://github.com/chentingpc/GuidedHeteEmbedding |

# Network embedding 分析类型



# 不同方法的可视化效果对比

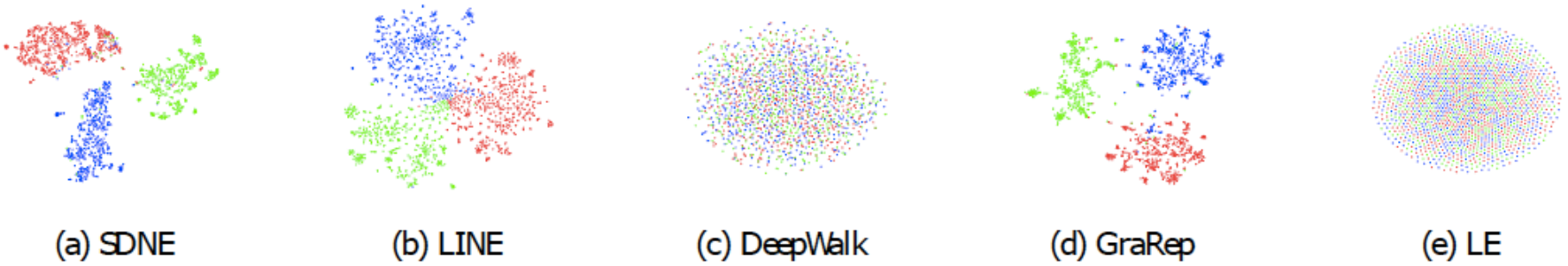
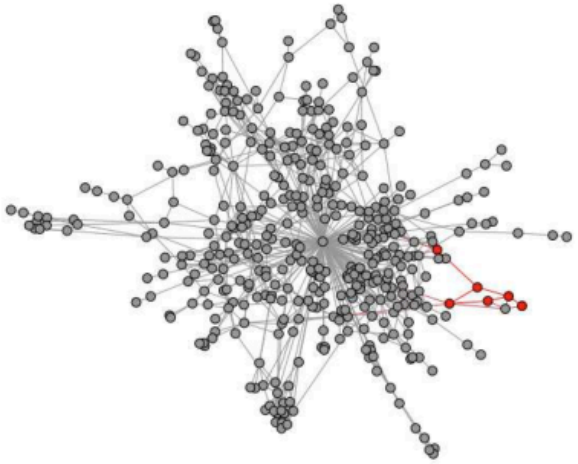
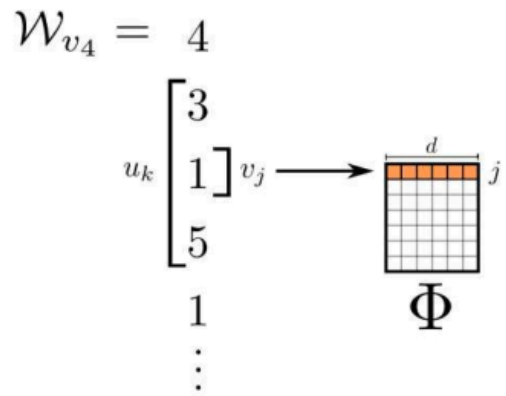



Figure 15: Network visualization of 20-NewsGroup by different network embedding algorithms, i.e., SDNE (Wang, Cui, and Zhu 2016), LINE (Tang et al. 2015), DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), GraRep (Cao, Lu, and Xu 2015), LE (Belkin and Niyogi 2003). Image extracted from SDNE (Wang, Cui, and Zhu 2016).

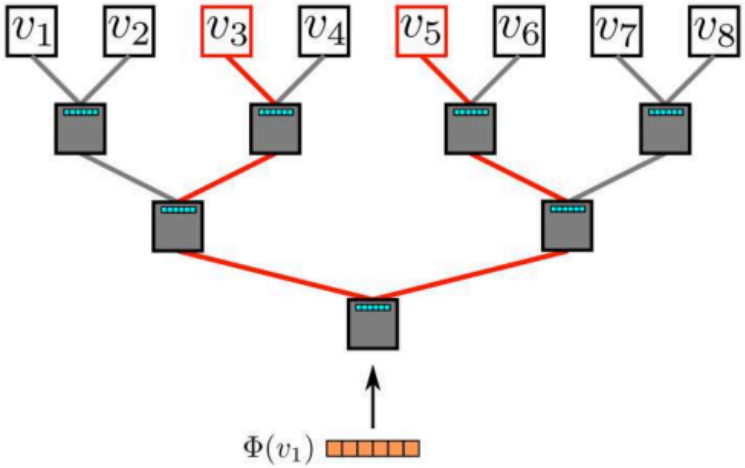
# Deep Walk



(a) Random walk generation.




(b) Representation mapping.




(c) Hierarchical Softmax.

# NE基础知识

- Softmax functions
- Distributional semantics
- Word2vec
  - CBOW
  - Skip-gram

# NE基础知识

- Representation learning
  - Using machine learning techniques to derive data representation
- Distributed representation
  - Different from one-hot representation, it uses dense vectors to represent data points
- Embedding
  - Mapping information entities into a low-dimensional space

# Softmax 函数

Input:[1, 2, 3, 4, 1, 2, 3],
Output:[0.024, 0.064, 0.175, 0.475, 0.024, 0.064, 0.175]

[0.1, 0.2, 0.3, 0.4, 0.1, 0.2, 0.3] (which sums to 1.6) the softmax would be [0.125, 0.138, 0.153, 0.169, 0.125, 0.138, 0.153].
in fact, de-emphasizes the maximum value (note that 0.169 is not only less than 0.475, it is also less than the initial proportion of 0.4/1.6=0.25).

```
>>> import numpy as np
>>> z = [1.0, 2.0, 3.0, 4.0, 1.0, 2.0, 3.0]
>>> softmax = lambda z:np.exp(z)/np.sum(np.exp(z))
>>> softmax(z)
array([0.02364054, 0.06426166, 0.1746813 , 0.474833  , 0.02364054,
       0.06426166, 0.1746813 ])
```

# 结束语

- Network Embedding 是一种表示，目的是便于观察到大规模和复杂网络中的模式、不变量等性质。
- 三篇的切入点不同，贡献不一样，你能给出一个评价吗？
- 如果你要讲这个主题，你将如何讲解？