Network Embedding

壽險程設科 崔嘉祐

Outline

- Introduction
- Network Embedding
- SVD
- DeepWalk

Introduction

- To process network data effectively, the first critical challenge is to find effective network data representation.
- Traditionally, we usually represent a network as a graph $G = \langle V, E \rangle$, where V is a vertex set representing the nodes in a network, and E is an edge set representing the relationships among the nodes.

Introduction(cont.)

- High computational complexity
- Low parallelizability
- Inapplicability of machine learning methods

Introduction(cont.)

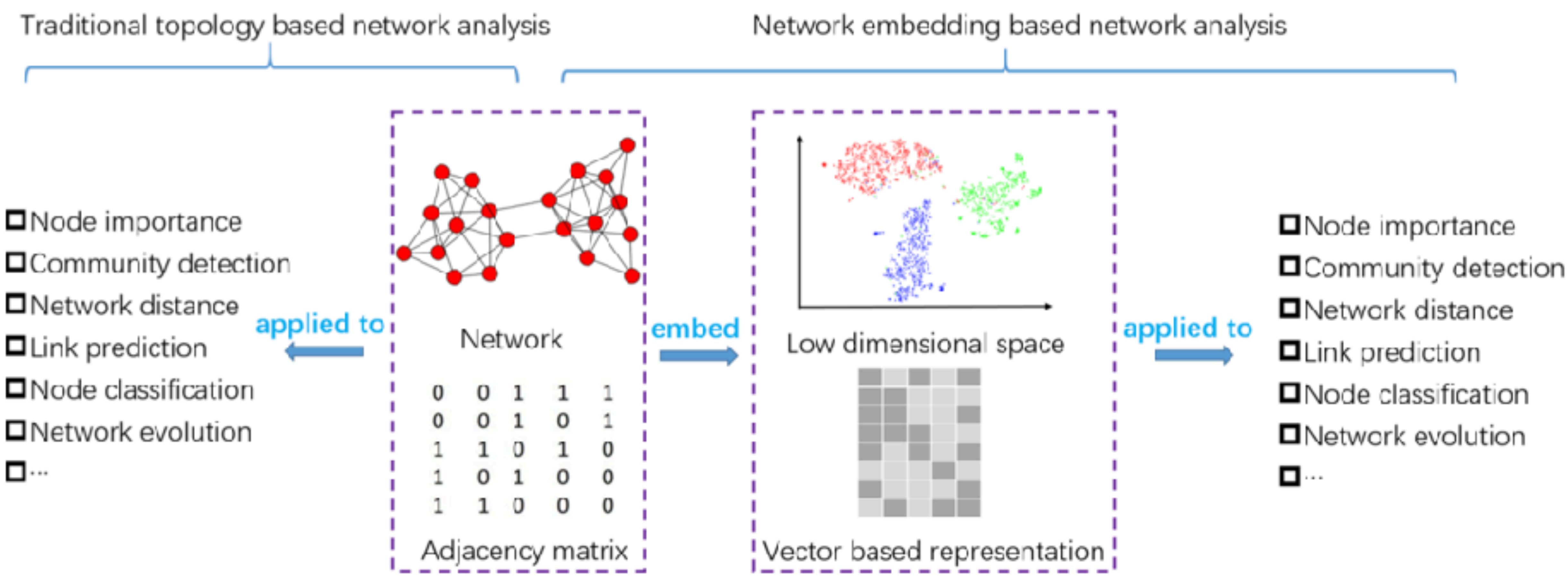


Figure 2: A comparison between network topology based network analysis and network embedding based network analysis

Introduction(cont.)

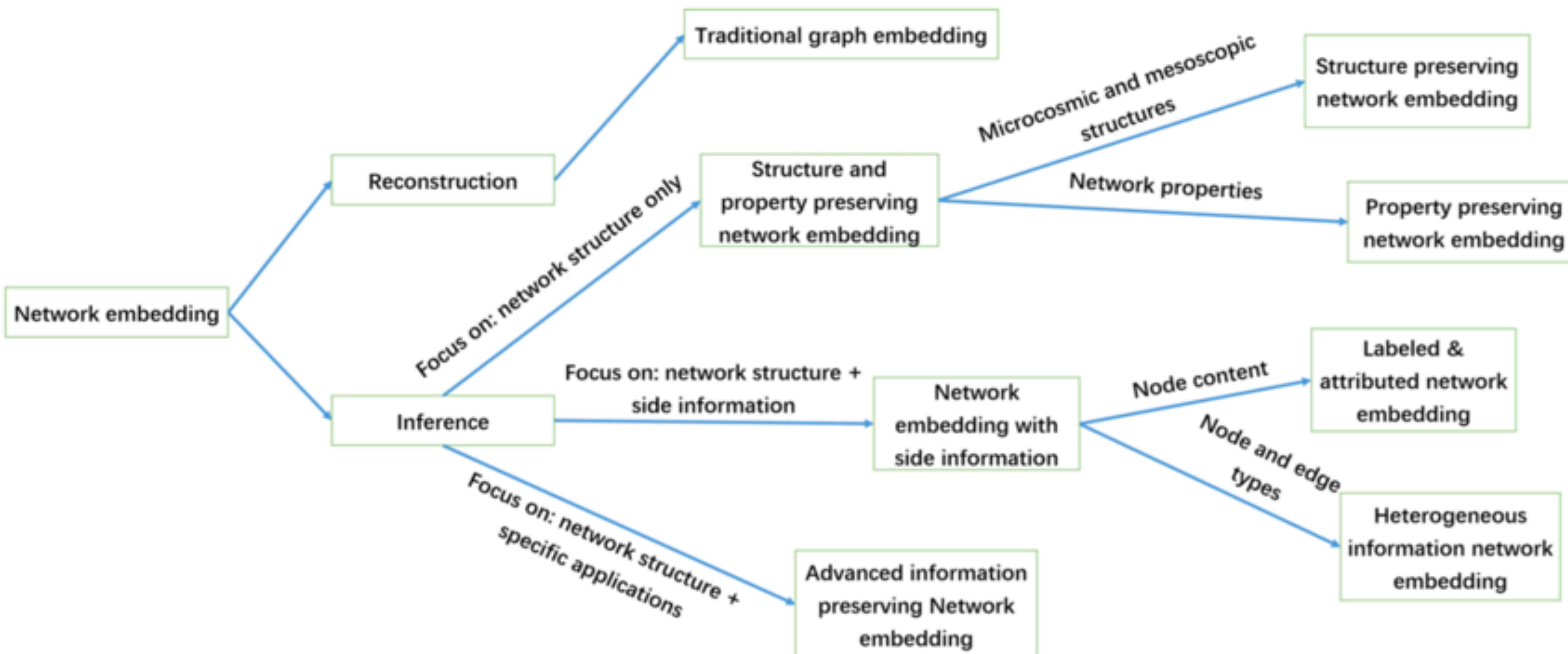


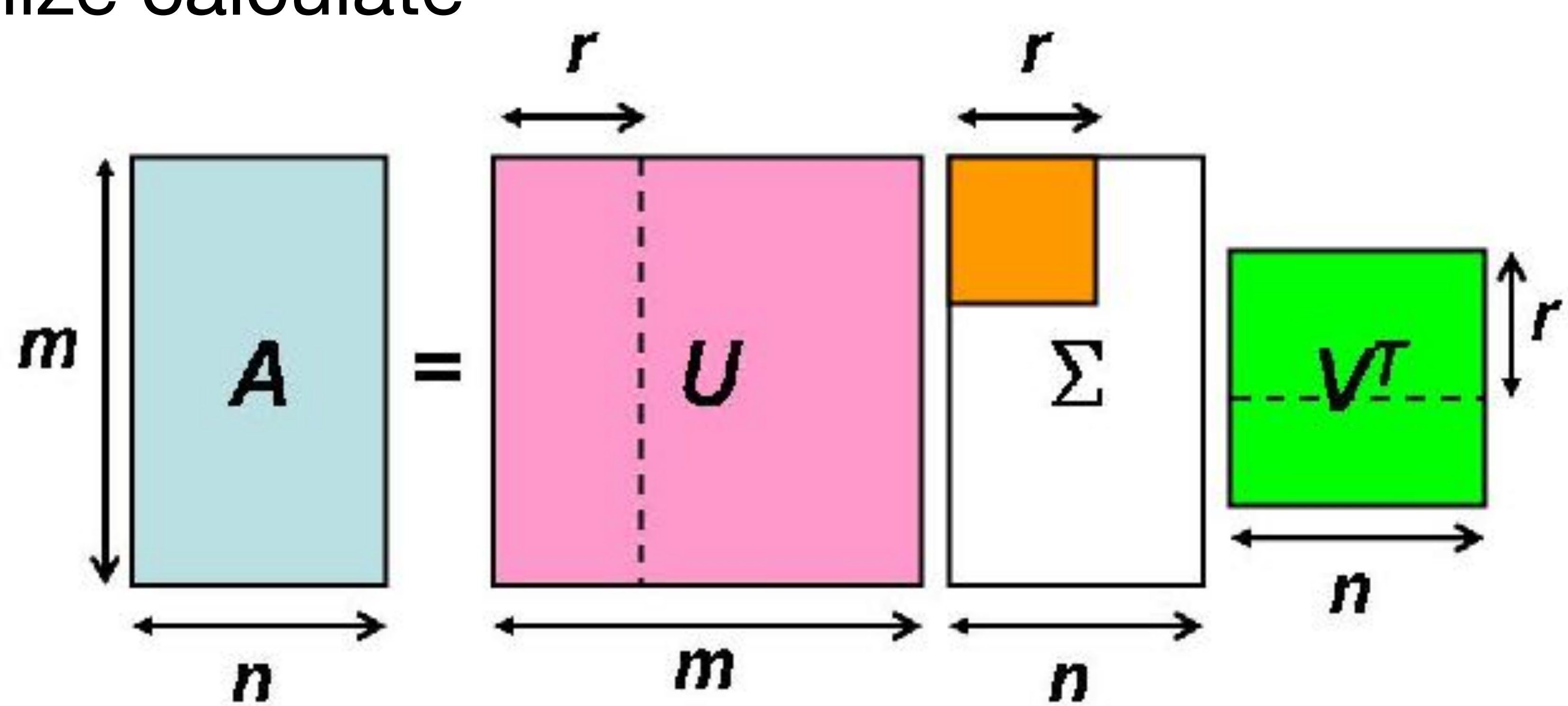
Figure 3: An overview of different settings of network embedding.

Network Embedding

- Network structure & properties preserving network embedding
- Network embedding with side information
- Advanced information preserving network embedding
  - Matrix Factorization (SVD)
  - Random Walk (DeepWalk)

SVD

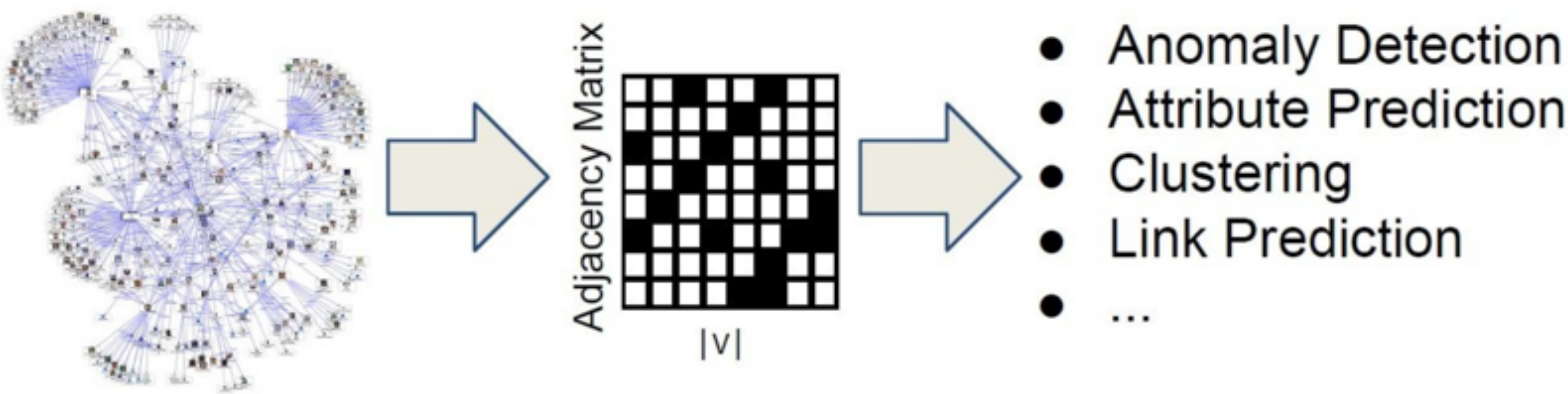
- Simplify data
- Denoising
- Optimize calculate



DeepWalk

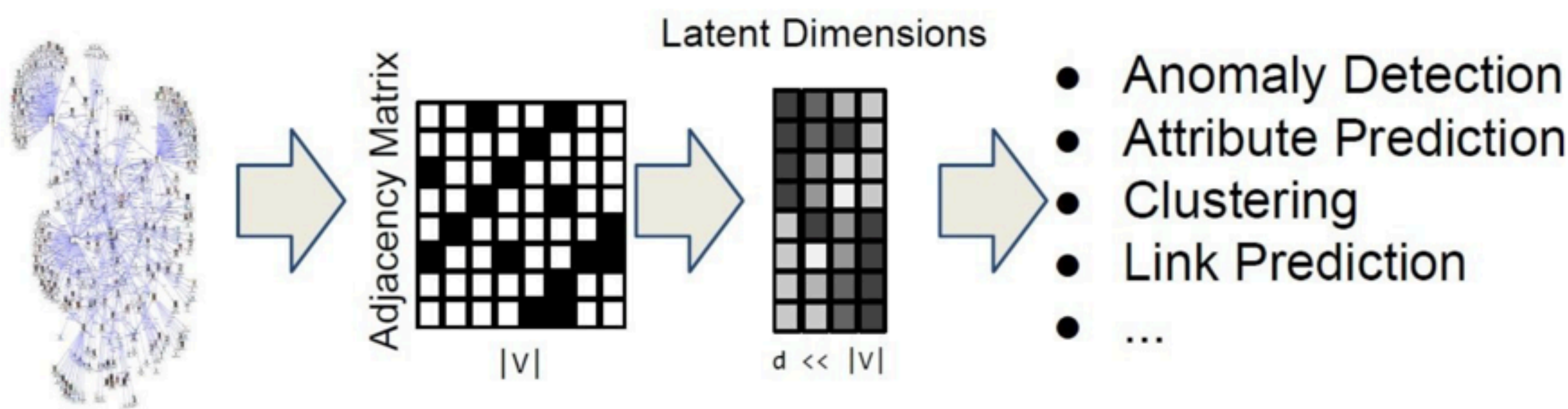
Introduction

- A first step in machine learning for graphs is to extract graph features:



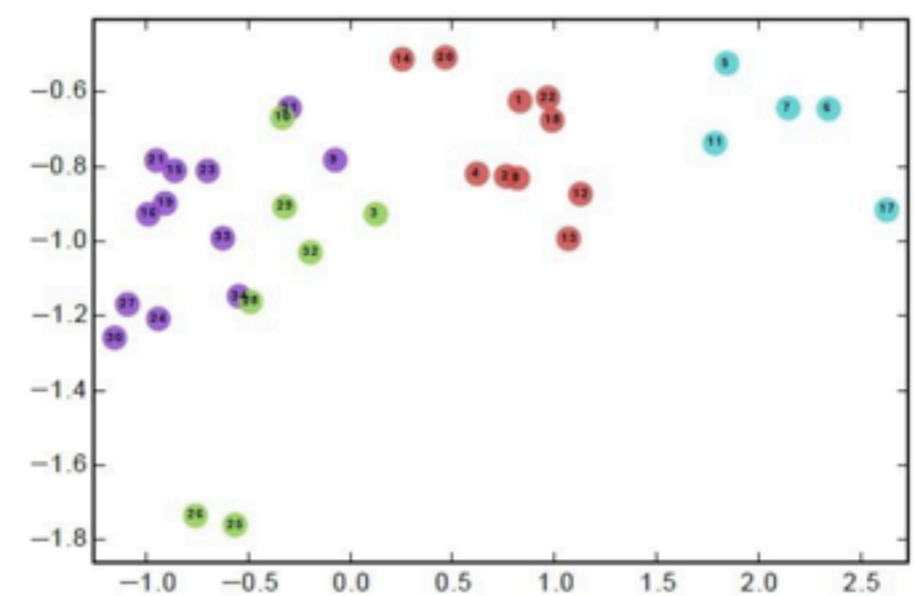
Introduction(cont.)

- We can also create features by transforming the graph into a lower dimensional latent representation.



Introduction(cont.)

(a) Input: Karate Graph



(b) Output: Representation

Problem definition

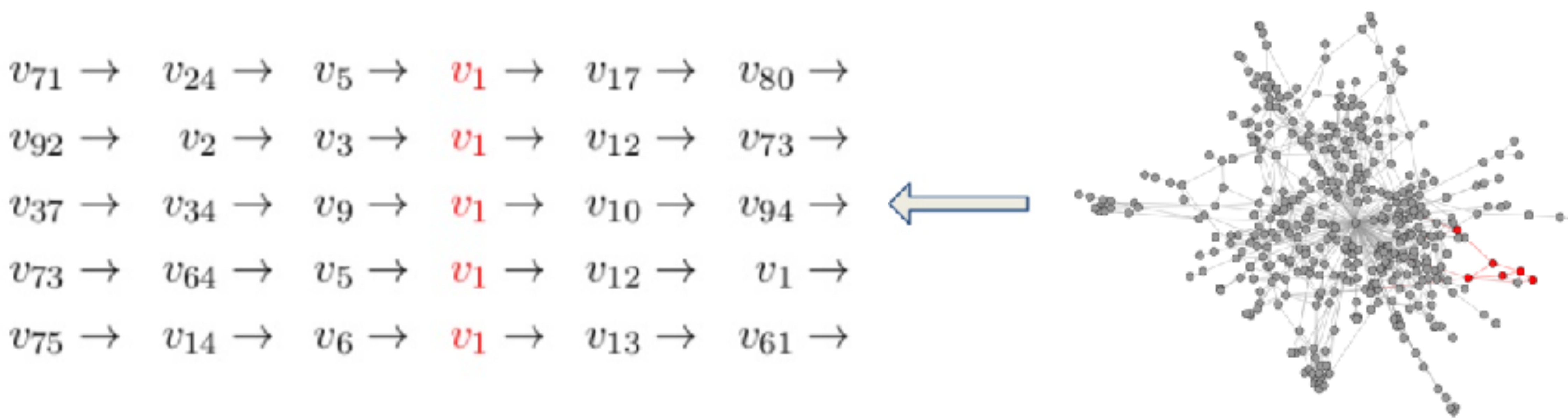
General: $G = (V,E)$,$E \subseteq (V x V)$

$G_L = (V,E,X, Y)$, $X \in R$ ^(|V|*S) $Y \in R$ ^(|V|*y)

Our goal : $X_E \in R$ ^(|V|*d)

Random walk

- Local exploration is easy to parallelize.
- Random walks make it possible to accommodate small changes in the graph structure without the need for global recomputation.



Algorithm

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

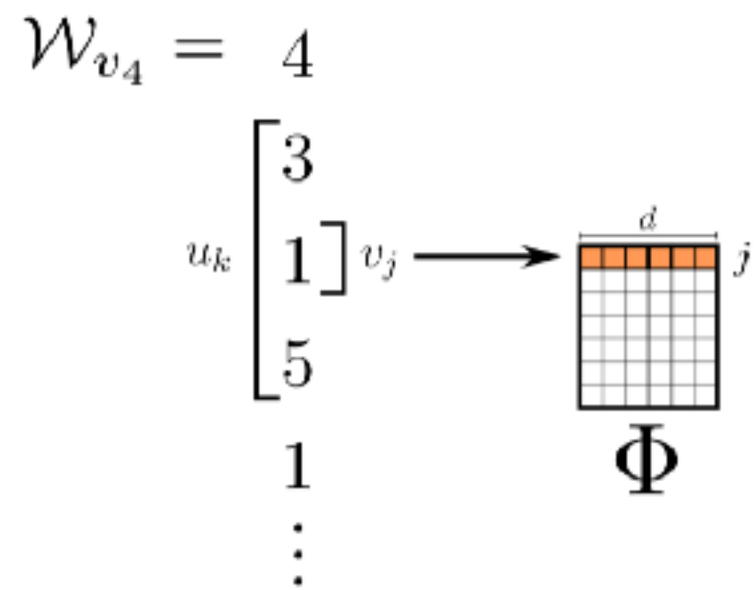
Input: graph $G(V,E)$
window size w
embedding size d
walks per vertex γ
walk length t
Output: matrix of vertex representations $\Phi \in \mathbb{R}^{|V|\times d}$

Initialization: Sample Φ from $\mathcal{U}^{|V|\times d}$
Build a binary Tree T from V
for $i = 0$ to γ **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

Skip-gram

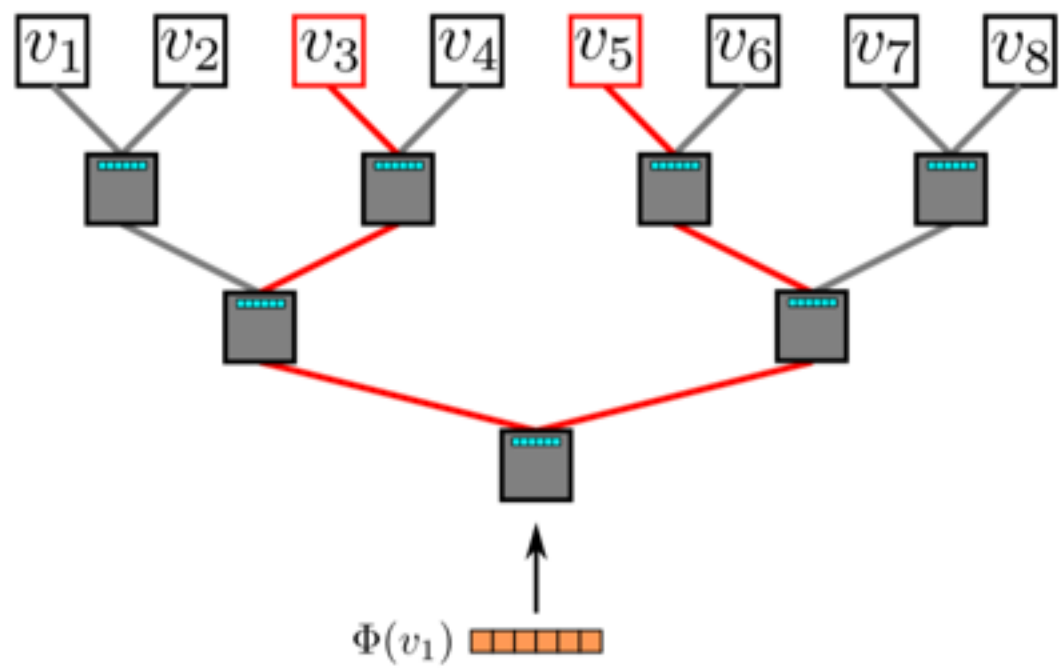
Algorithm 2 SkipGram(Φ, $\mathcal{W}_{v_i}$, w)

```
for each v_j ∈ W_{v_i} do
    for each u_k ∈ W_{v_i}[j − w : j + w] do
        J(Φ) = − log Pr(u_k | Φ(v_j))
        Φ = Φ − α ∗ ∂J/∂Φ
    end for
end for
```



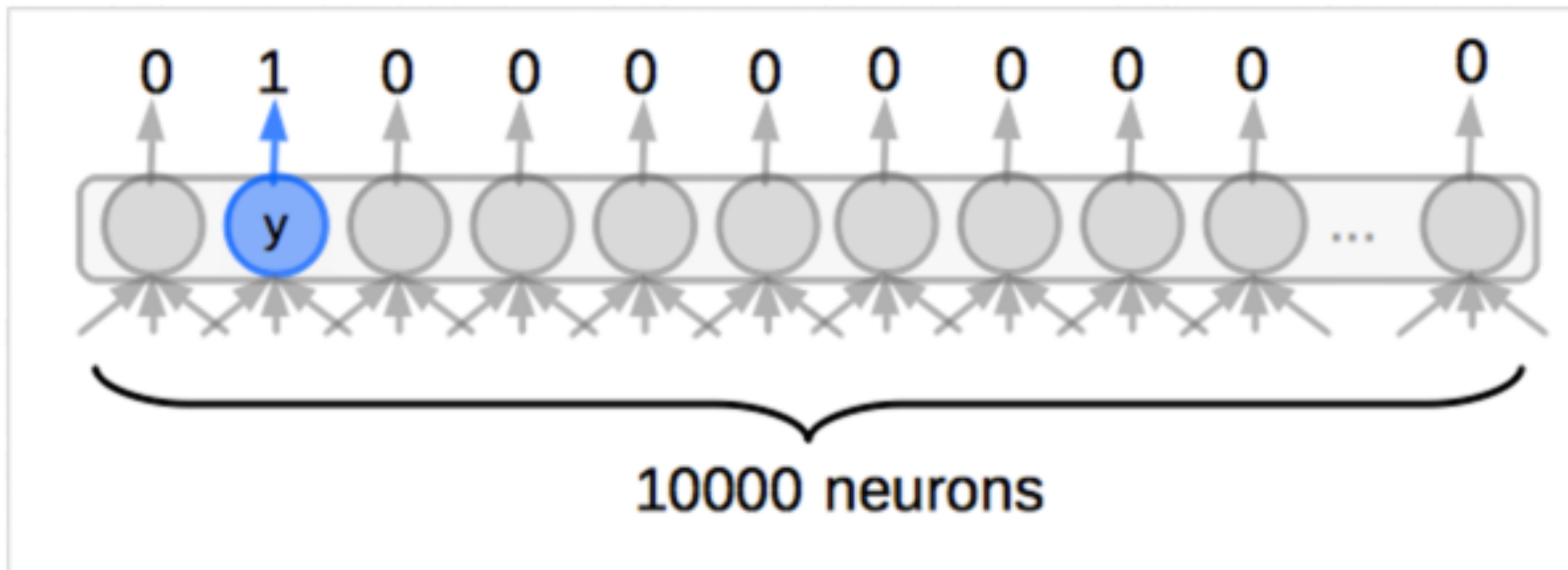
(b) Representation mapping.

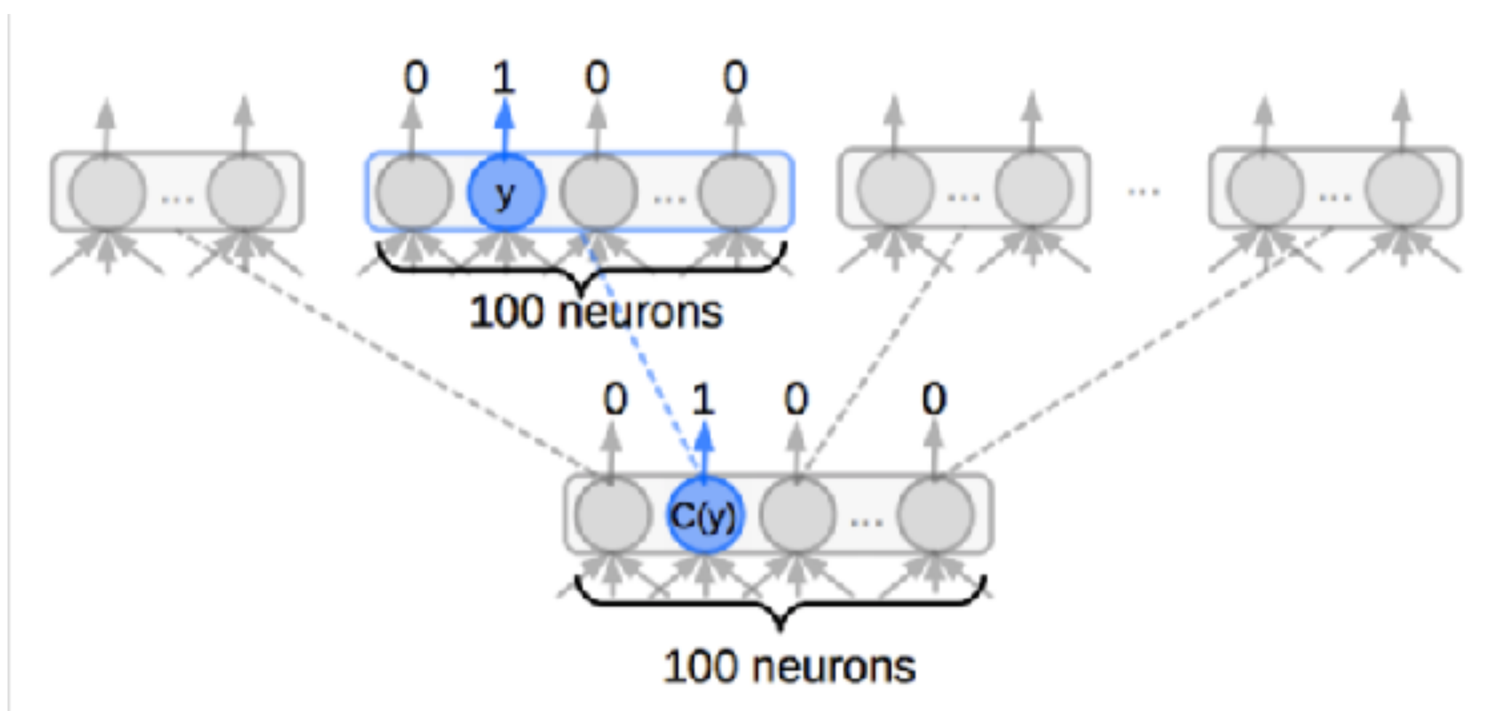
Hierarchical Softmax



(c) Hierarchical Softmax.

Hierarchical Softmax





Thanks!

–George Tsui