中国科学技术大学
University of Science and Technology of China

《人工智能数学原理与算法》
第4章：图神经网络

4.2 图表征学习

王翔

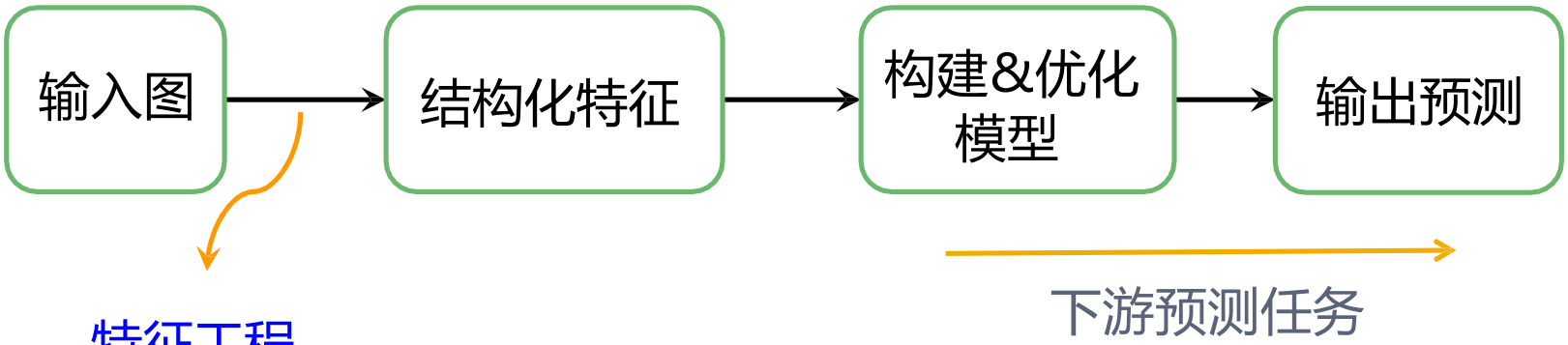
xiangwang@ustc.edu.cn

目录

回顾：面向图数据的传统机器学习

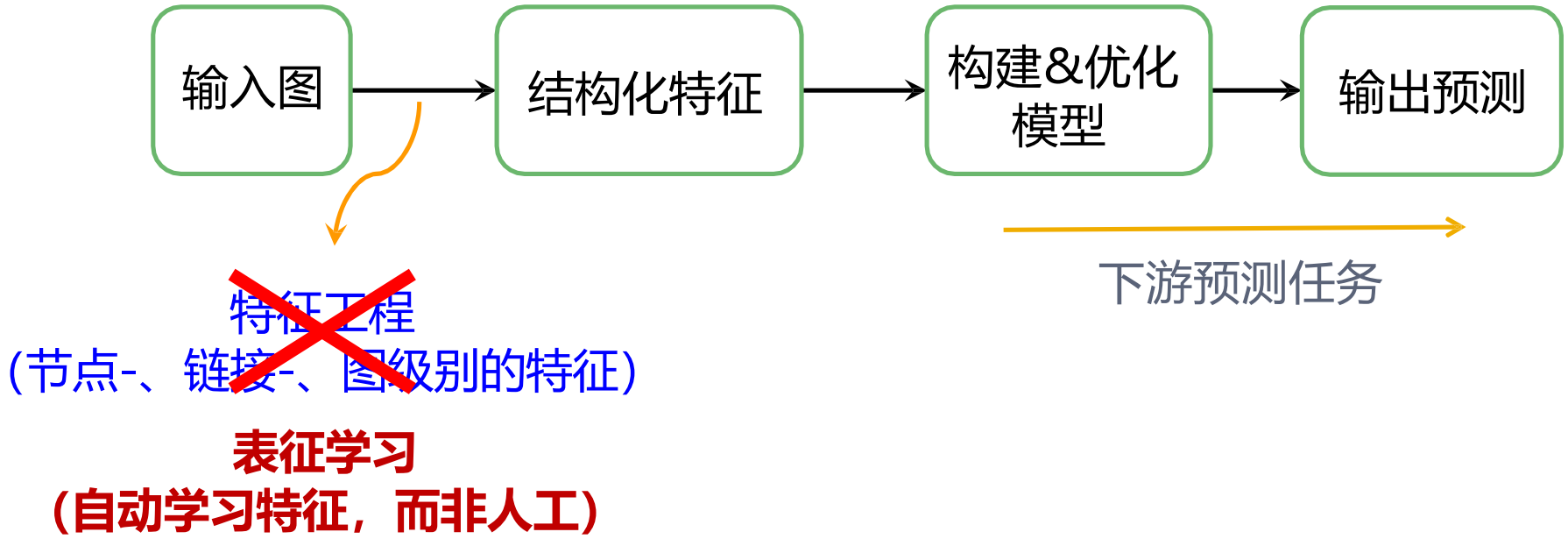
- 给定一个输入图，提取节点、边和图级特征，然后学习一个将特征映射到标签的模型（如 神经网络等）

$$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x)$$



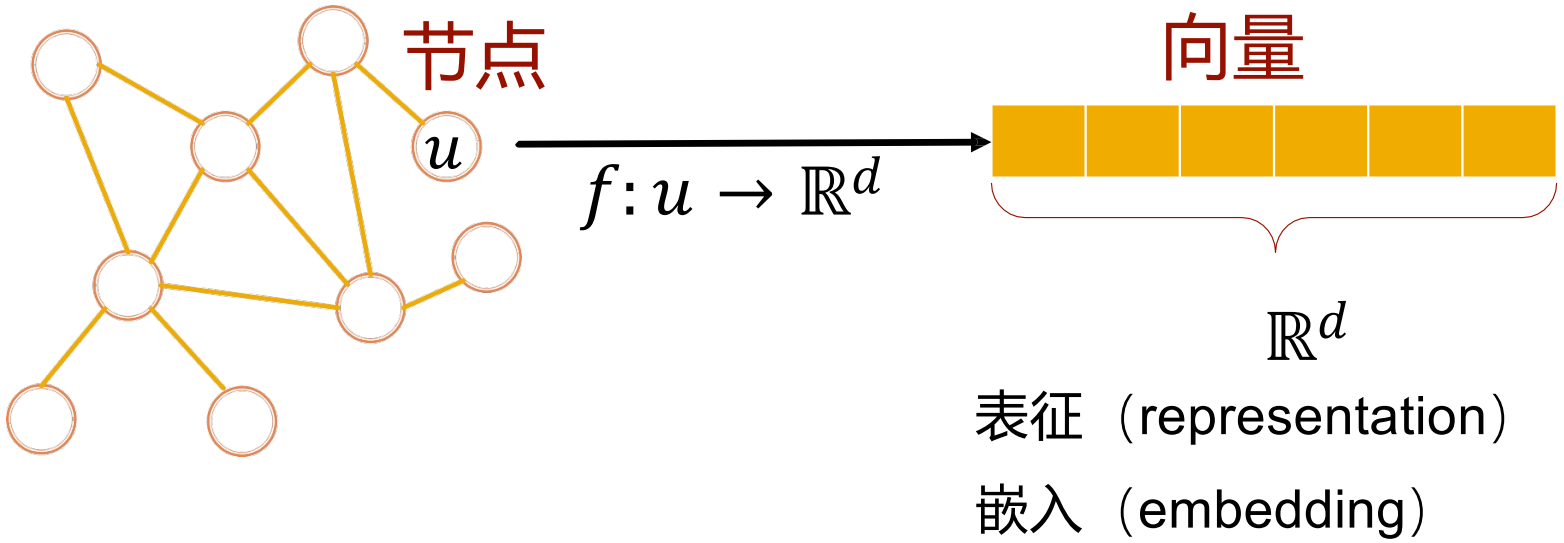
图表征学习

□ 图表征学习缓解了每次都需要进行特征工程的需求



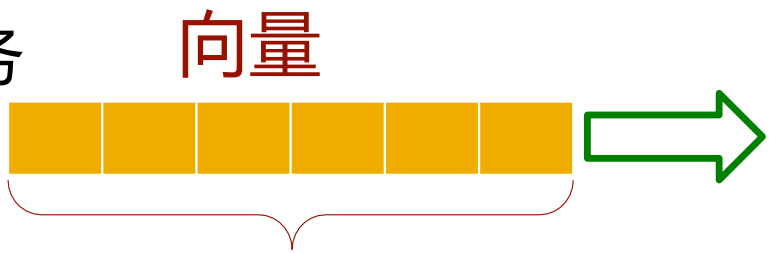
图表征学习

- **目标**：实现高效的、与任务无关的图特征自动学习！



图表征学习：节点嵌入

- **任务：将节点映射到一个嵌入空间（embedding space）**
  - 节点嵌入的相似性反映它们在图中的相似性
    - 例如：两个节点彼此接近（通过一条边相连）
  - 编码图结构信息
  - 可用于多种下游预测任务



特征表示（feature representation）

向量嵌入(embedding)

节点嵌入示例

- Zachary 空手道俱乐部网络中节点的二维嵌入表示

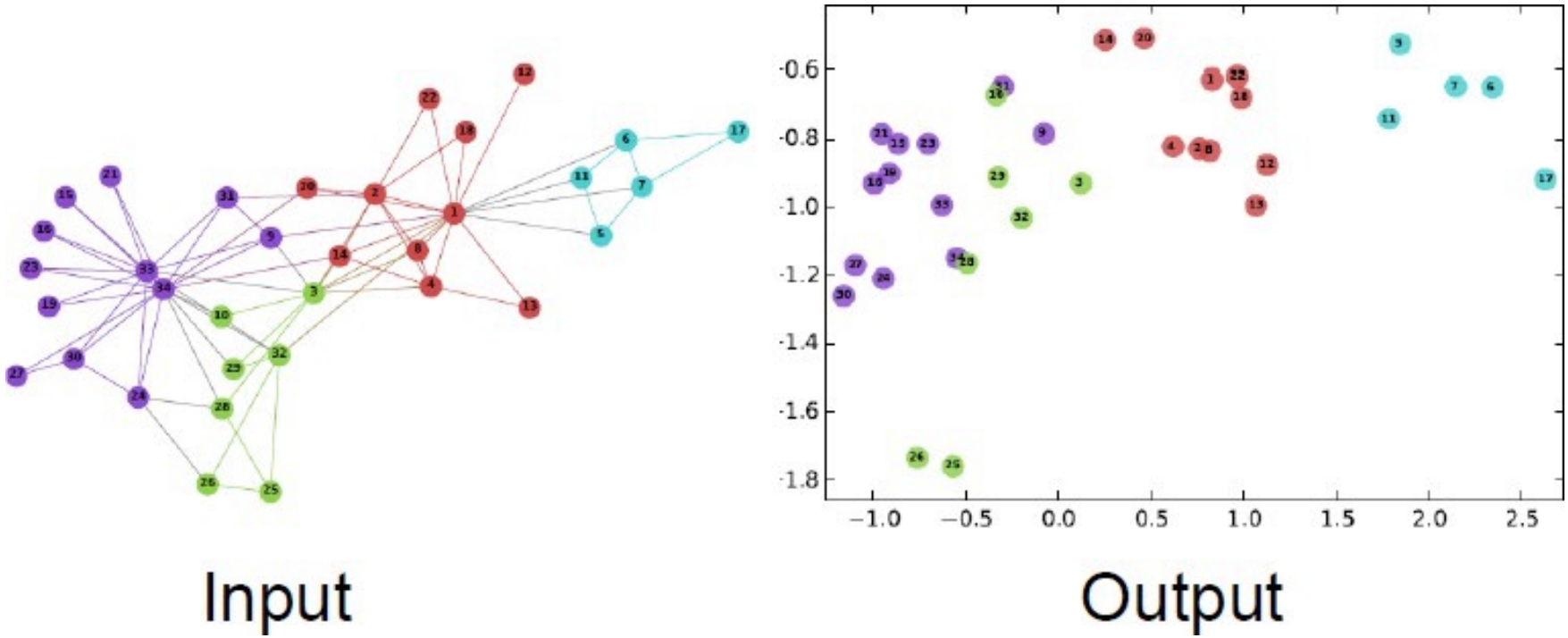
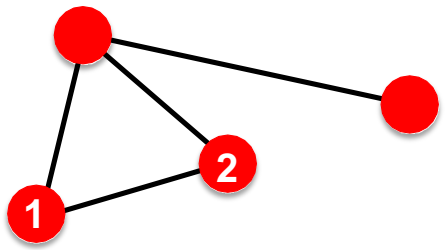


Image from: Perozzi et al. DeepWalk: Online Learning of Social Representations. *KDD 2014.*

目录

设置

- **假设：假设我们有一个（无向）图 G：**
  - V是节点集合
  - A是链接矩阵（假设为二值矩阵）
  - 为简化起见：不使用节点特征或其他附加信息

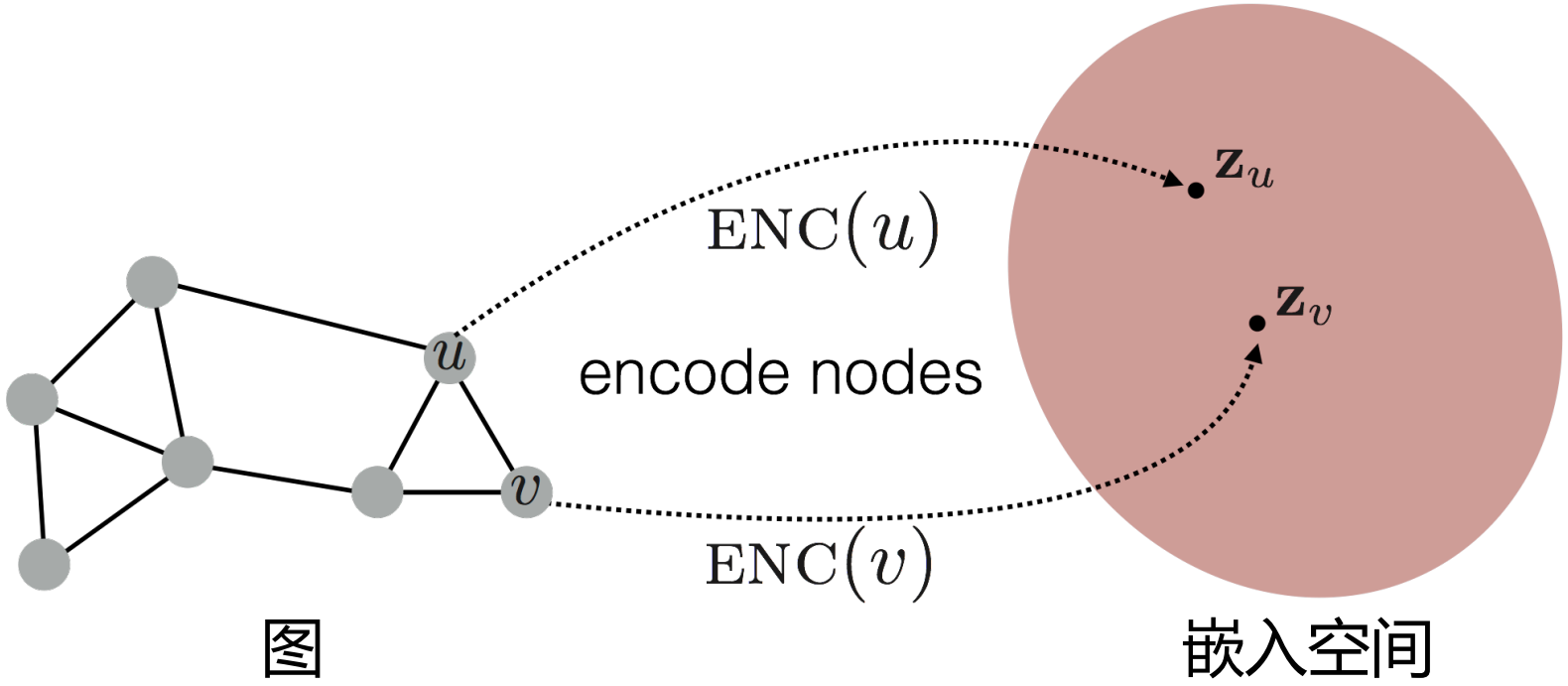


V: {1, 2, 3, 4}

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

节点嵌入

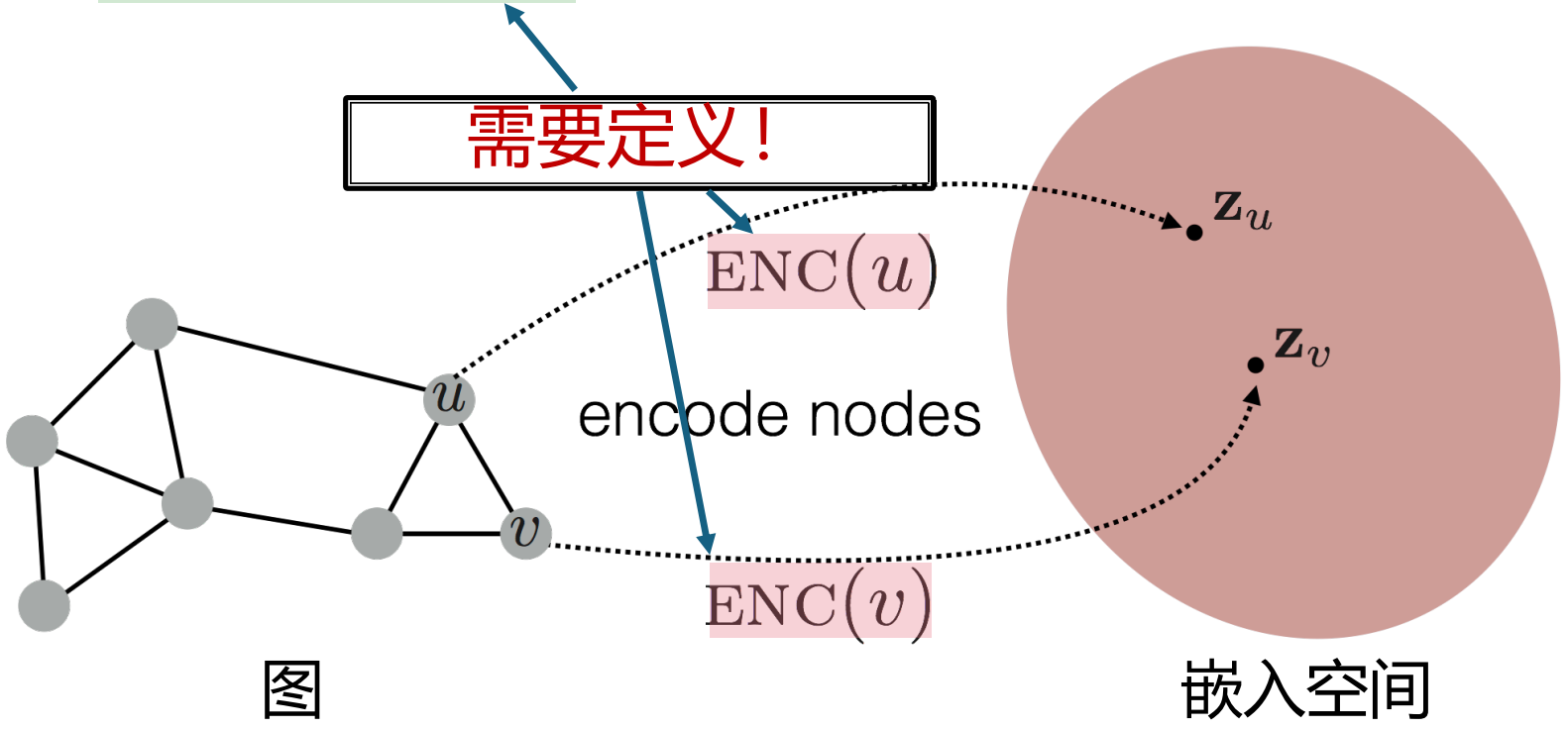
- **目标**：目标是对节点进行编码，使得**嵌入空间中的相似性**（例如点积）能够近似反映**图中的相似性**



节点嵌入

目标： $\text{similarity}(u, v) \approx z_v^T z_u$

图上相似度　　嵌入空间相似度



节点嵌入的学习

1. **编码器Encoder将节点映射到嵌入表示**

$$\mathrm{ENC}(\mathbf{v}) = \boldsymbol{z_v} \in \boldsymbol{R^d}$$

2. **定义一个节点相似性函数（即，度量图中的相似性）**
3. **解码器Decoder从嵌入中恢复相似性得分**
4. **优化编码器的参数，使得：**

$$\mathbf{similarity}(\mathbf{u},\mathbf{v}) \quad \approx \quad \mathrm{DEC}(\boldsymbol{z_u}, \boldsymbol{z_v}) = \boldsymbol{z_v^T z_u}$$

图上相似度　　嵌入空间相似度

“浅层”编码

- 最简单的编码方法：编码器仅是一个嵌入查找表

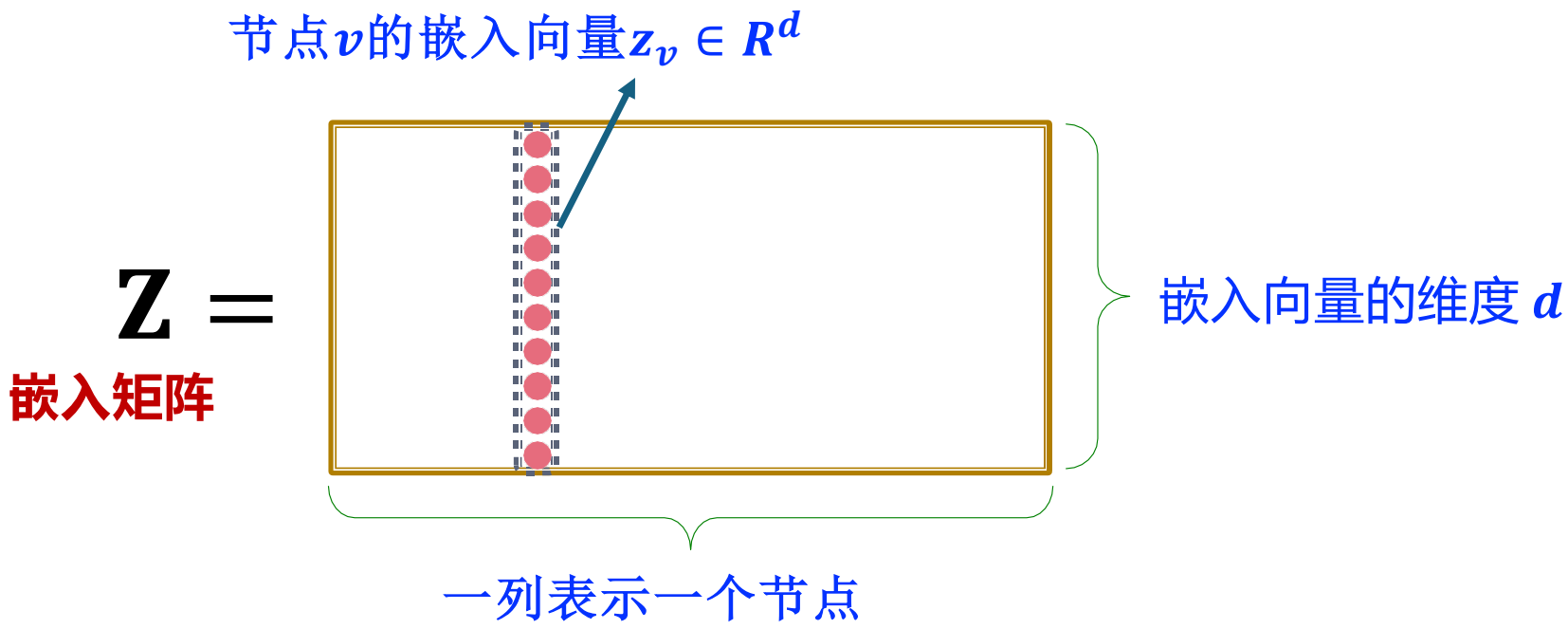
$$\mathrm{ENC}(\mathbf{v}) = \boldsymbol{z}_{\boldsymbol{v}} = \mathbf{Z} \cdot \boldsymbol{v}$$

$\boldsymbol{Z} \in \boldsymbol{R}^{\boldsymbol{d}\times|\boldsymbol{V}|}$ 矩阵：每一列是一个节点的嵌入表示（即我们要学习/优化的内容）

$\boldsymbol{v} \in \mathbb{I}^{|\boldsymbol{V}|}$ 指示向量：除了表示节点v的那一列为 1 外，其余全为 0。

“浅层”编码

- 最简单的编码方法：编码器仅是一个嵌入查找表



“浅层”编码

- **最简单的编码方法：编码器仅是一个嵌入查找表**

每个节点被分配一个唯一的嵌入向量

（即我们直接优化每个节点的嵌入表示）

代表方法包括：DeepWalk、node2vec

框架总结

- Encoder+Decoder框架
  - 浅层编码器：嵌入查找
  - 需要优化的参数：矩阵 Z，包含所有节点 $v \in V$的嵌入向量 z_v
  - 我们将在图神经网络（GNNs）中介绍深层编码器
  - 解码器：基于节点相似性进行计算

  - 目标：最大化**相似节点**对 (u, v)的内积$z_v^T z_u$

如何定义节点相似性?

- 不同方法的关键在于它们如何定义节点相似性

- 两个节点是否应具有相似的嵌入，取决于它们是否……
  - 相互连接?
  - 共享相邻节点?
  - 拥有相似的“结构角色”?

- 接下来我们将学习一种基于随机游走（random walks）的节点相似性定义，并介绍如何针对该相似性度量来优化节点嵌入。

节点嵌入的注意事项

- **这是一种无监督/自监督的节点嵌入学习方式。**
  - 我们不使用节点标签，也不使用节点特征。
  - 其目标是直接估计每个节点在嵌入空间中的一组坐标（即嵌入），使得图结构的某些特征（由解码器 DEC 捕捉）得以保留。

- **这些嵌入是与任务无关的：**
  - 它们不是为了某个特定任务而训练的，但可以用于各种任务。

目录

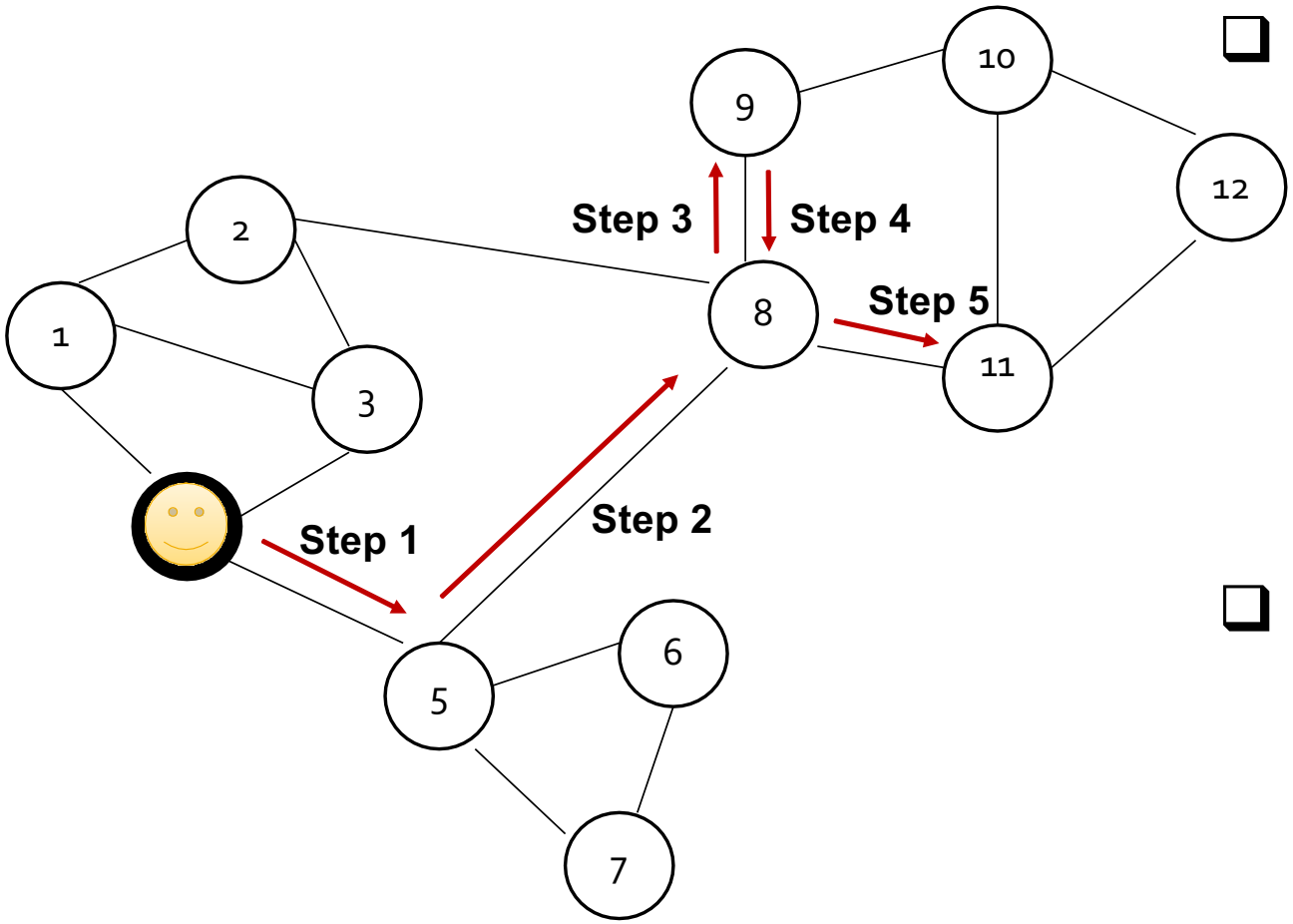
数学表示

- **向量 z_u：**
  - **节点u的嵌入（即我们希望学习的内容）**
- **概率$\mathrm{P}(v|z_u)$：**
  - 从节点u出发进行随机游走，访问到节点v的（预测）概率 。

- **用于生成预测概率的非线性函数：**
  - Softmax函数：将一个包含K个实数的向量（模型预测）转换为K个概率，这些概率的总和为 1：$S(\boldsymbol{z})[i] = \frac{e^{z[i]}}{\sum_{j=1}^{K} e^{z[j]}}$
  - Sigmoid函数：S型函数将实数值映射到 (0, 1) 的范围内

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

随机游走（Random Walk）



- 给定一个图和一个起始节点，我们随机选择它的一个邻居并移动过去；然后再从当前节点随机选择一个邻居并移动过去，如此反复。
- 以这种方式访问的一系列（随机）节点构成了图上的一次随机游走。

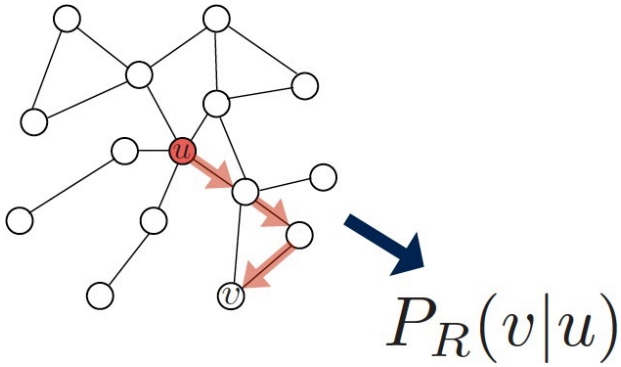
基于随机游走的嵌入

$$z_v^T z_u \approx$$

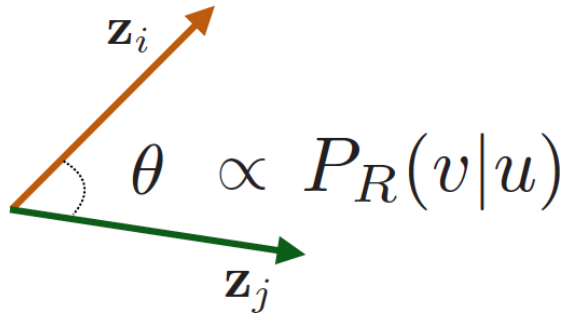
节点u和节点v在图上的随机游走中共同出现的概率

基于随机游走的嵌入

- 核心思想：如果从节点 u开始的随机游走以较高概率访问到节点v，则说明u和v相似（反映了高阶多跳的结构信息）。



- 优化嵌入，使其能够编码这些随机游走统计信息。
- 嵌入空间中的相似性（此处：点积 = cos(θ)）编码了随机游走中的“相似性”。



目标函数

- 给定一个图输入 $G = (V, E)$
- 目标是学习一个映射 $f: u \rightarrow R^d$

- 优化：对数似然估计

$$\arg\max_{z} \sum_{u \in V} \log \mathrm{P}(N_{\mathrm{R}}(u) | \mathbf{z}_u)$$

- $\boldsymbol{N_R(u)}$表示通过随机游走策略$\boldsymbol{R}$得到的节点$\boldsymbol{u}$的邻居集合

目标函数

- 相等地，通过最小化随机游走邻域$N_R(u)$的负对数似然，来优化节点嵌入z_u

$$\arg\min_z \mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

- Softmax函数：我们希望节点v在所有节点n中与节点u最为相似。

$$P(v|\mathbf{z}_u) = \frac{\exp(\mathbf{z}_u^{\mathrm{T}} \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^{\mathrm{T}} \mathbf{z}_n)}$$

目标函数

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log\left(\frac{\exp(\mathbf{z}_u^{\mathrm{T}} \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^{\mathrm{T}} \mathbf{z}_n)}\right)$$

对所有节点 u 求和

对从 u 出发的随机游走中出现的节点 v 求和

计算节点 u 和 v 在随机游走中共现的预测概率

- **优化随机游走嵌入 = 寻找使损失函数 $\mathcal{L}$ 最小的嵌入向量 z_u**

优化算法：随机梯度下降

- 得到目标函数后，我们如何对其进行优化（最小化）？

$$\arg\min_{z} \mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

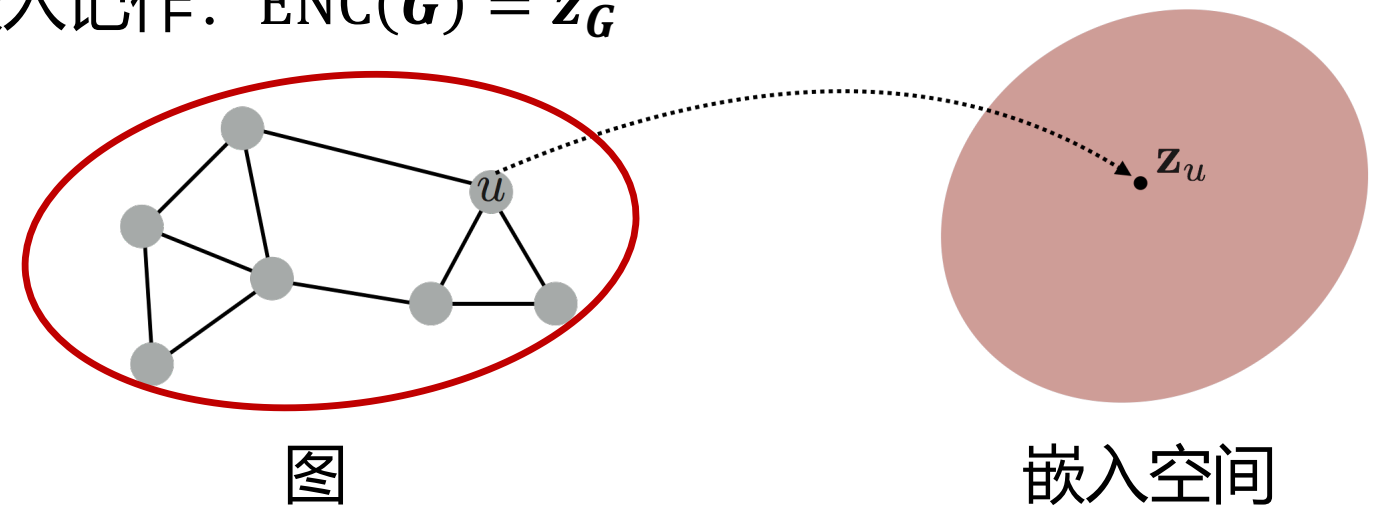
- 梯度下降 $z_u \leftarrow z_u - \eta \frac{\partial \mathcal{L}}{\partial z_u}$.

- 随机梯度下降 $z_v \leftarrow z_v - \eta \frac{\partial \mathcal{L}^{(u)}}{\partial z_v}$.

目录

图嵌入

- **目标：**希望对一个子图或整个图G进行嵌入表示。
- 图嵌入记作：$\mathrm{ENC}(G) = z_G$



- 任务：
  - 分类有毒与无毒分子
  - 识别异常图结构

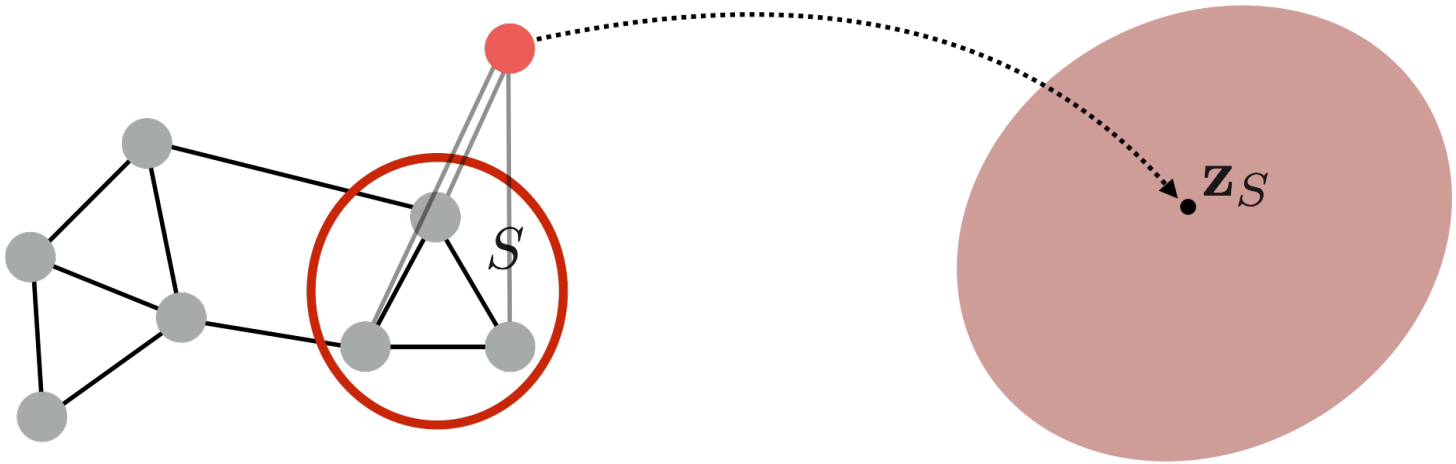
图嵌入方法1

- **简单有效的方法：**。
  - 在（子）图G上运行标准的节点嵌入方法
  - 然后对图G中的节点嵌入进行求和（或求平均）

$$\boldsymbol{z}_G = \sum_{v \in G} \boldsymbol{z}_v$$

图嵌入方法2

- 引入一个“虚拟节点”来表示（子）图，并运行标准的节点嵌入方法



如何使用嵌入?

- **节点分类**：根据 z_v预测节点v的标签
- **链接预测**：根据(z_v, z_u)预测是否存在边(v, u)
  - 在进行节点对建模时，可以对嵌入进行如下操作：
  - 拼接： $f(z_v, z_u) = g([z_v, z_u])$
  - Hadamard积： $f(z_v, z_u) = g(z_v * z_u)$ （对应元素相乘）
  - 求和/平均： $f(z_v, z_u) = g(z_v + z_u)$
  - 距离： $f(z_v, z_u) = g(\|z_v - z_u\|_2)$
- **图分类**：通过聚合节点嵌入或使用虚拟节点得到图嵌入z_G，再基于图嵌入z_G进行标签预测。

总结

图表征学习：无需特征工程、用于下游任务的节点和图嵌入学习方法。

- **编码器-解码器框架**：
  - 编码器：嵌入查找
  - 解码器：基于嵌入预测得分，以匹配节点相似性

- **节点相似性度量**：随机游走

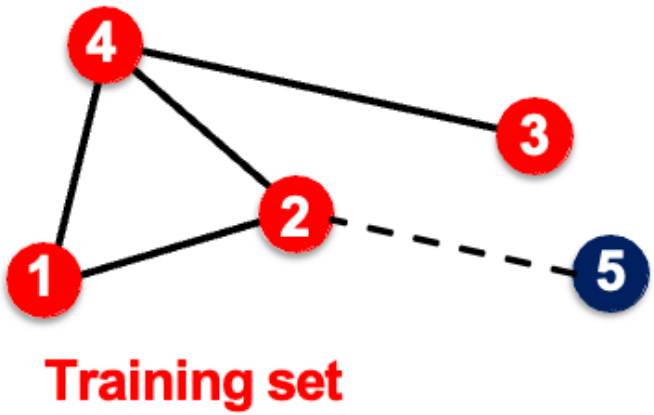
- **图嵌入扩展**：通过聚合节点嵌入获得图嵌入

局限性1

基于随机游走的节点嵌入方法存在以下局限性：

口 传导式（transductive）而非归纳式（inductive）：

- 无法为训练集中未出现的节点生成嵌入；
- 无法应用于新的图或动态变化的图。

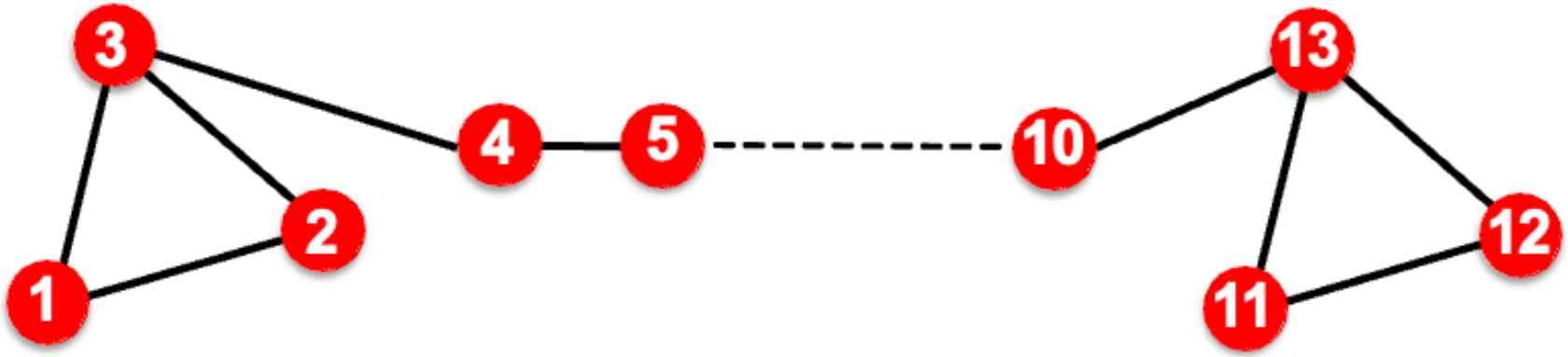


示例：在测试阶段新增一个节点 5（如社交网络中的新用户），

无法为其计算嵌入，必须重新计算所有节点的嵌入。

局限性2

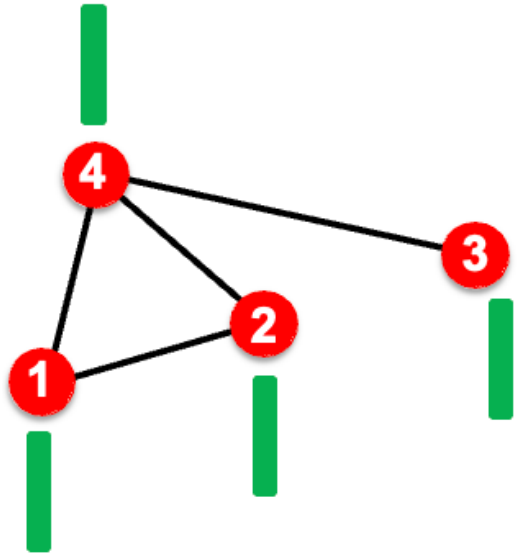
- **无法捕捉结构相似性（structural similarity）**：



- 节点 1 和节点 11 在结构上是相似的
  - 都属于一个三角结构，度为 2，……
- 然而，它们的嵌入却截然不同。
  - 从节点 1 出发的随机游走不太可能访问到节点 11。

局限性3

- **无法利用节点、边和图的特征信息**：



- 特征向量
  - 例如：在蛋白质-蛋白质相互作用图中，蛋白质的属性向量

- **解决上述局限的方法：图神经网络**