L25: Graph Embeddings

Apr 16, 2025

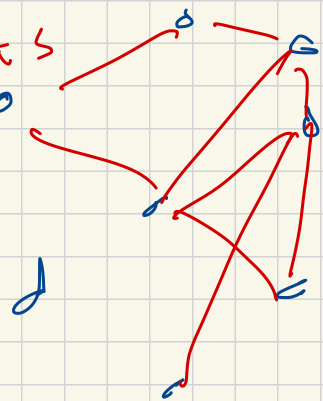
Data Mining

Jeff M. Phillips

Input Data Graph $\quad G = (V, E)$

vertices edges

Data objects $\simeq$ vertices $\rightarrow$ Data points



Information $\simeq$ edges

Before: Data $X = \{x_1 \dots x_n\} \subset \mathbb{R}^d$

Distance $D : X \times X \to \mathbb{R}$

$$D(x_1, x_2) = \| x_1 - x_2 \|$$

$$x_1 = (x_{11}, x_{12} \dots x_{1d})$$

1. Edges as encoding Similarity in Graph

adjacency matrix $A \in \{0,1\}^{n \times n}$ $\quad |V| = n$

$A_{ij} = 1$ iff $e = (v_i, v_j) \in E$

o.w. $A_{ij} = 0$

c MDS

Similarity Matrix $S = A$ $\quad \mathrm{Sim}(v_i, v_j) = A_{ij}$

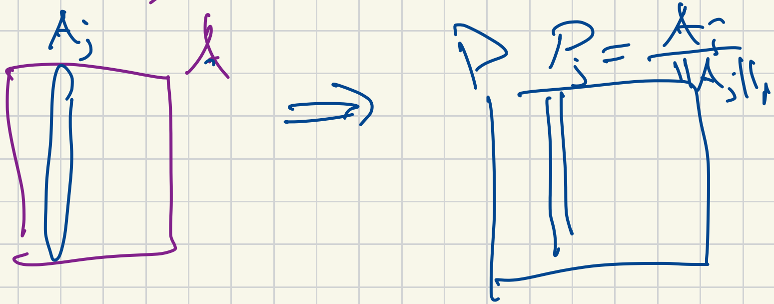
$S = X^T X \leftarrow \mathrm{eig}(S)$ $\quad X \in \mathbb{R}^{n \times d}$

let X_k best rank-k approx of X

$X_k \in \mathbb{R}^{n \times k}$ $\quad X_k = {}_n\underset{k}{[\;\;]}$ $\quad x_i \in \mathbb{R}^k$ embedding of vertex v_i

$X = U \Lambda^{\frac{1}{2}}$ $\quad [U, \Lambda] \leftarrow \mathrm{eig}(S)$

Adjacency $A \rightarrow L_1$-normalize $\Longrightarrow P$

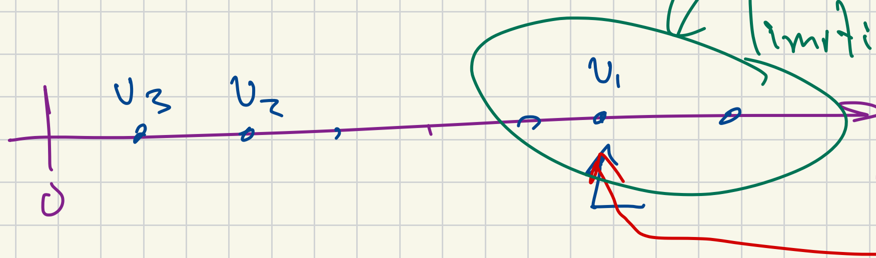


$P_j = \frac{A_j}{\|A_j\|_1}$

probability transition matrix

Let $U \Lambda U^T = P$ via eigs(P)

Consider top eigenvector v_1



leave prob

limiting distr of MC.

$v_1 = (v_{1,1}, v_{1,2}, \ldots, v_{1,d})$

Adjacency A as similarity S

use S as labeled input to

Distance Metric Learning.

Use S to learn Mahalanobis

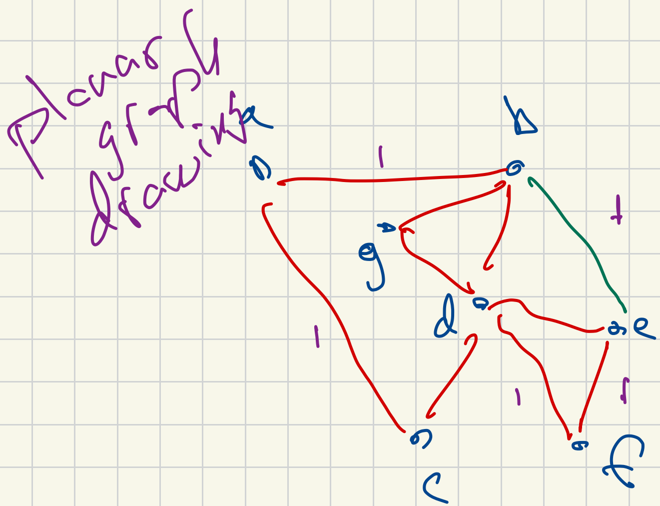
distance $d_M(x_i, x_j) = \sqrt{(x_i - x_j)^T M (x_i - x_j)}$

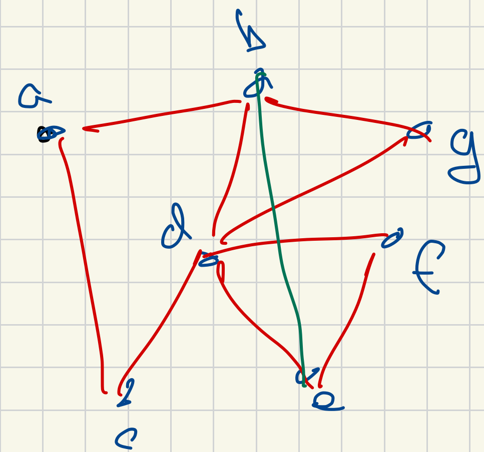
or $M = R^T R$

multiply $X R^{-1} \rightarrow$ Euclidean

Planar Graphs $G = (V, E)$

is a graph that can be drawn in the plane $\mathbb{R}^2$, w no crossing edges.





Laplacian

input $G = (V, E)$

Nice property of NCut

- adjacency A
- degree $D \in \mathbb{R}^{n \times n}$ diagonal $D_{ii} = \deg(v_i)$

$D_{i \neq j} = 0$

unnormalized Laplacian $L_0 = D - A$

Laplacian $L = I - D^{-1/2} A D^{-1/2} = D^{-1/2} L_0 D^{-1/2}$

Spectral Clustering

$L = U \Lambda U^T$ eigs

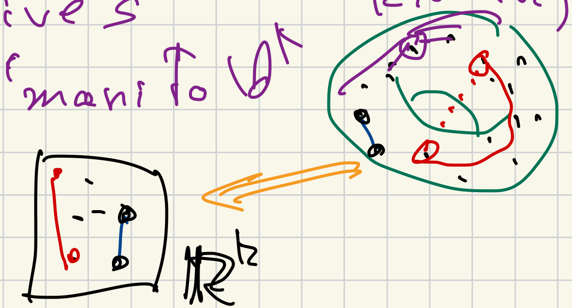
Step 1: Take the top k eigenvectors $U_k \Lambda^{\frac{1}{2}} \subset \mathbb{R}^k$ (2 … k+1)

Step 2: Partition U into clusters.

Laplacian Eigen Maps

for "manifold learning" / non-linear dimensionality reduction

Premise (circa 2005) data $X \subset \mathbb{R}^d$ lives on a "manifold"



$K \in \mathbb{R}^{n \times n}$ k-nearest-neighbor graph.

$K = U \Delta U^T \leftarrow$ eigs

$Z \subset \mathbb{R}^k \Rightarrow Z = U_{1:k} \Delta_{1:k}^{1/2}$

around 2015

word embeddings
word2vec, GloVe

string of $\boxed{\text{text}}$ in large corpus ...

$\underbrace{\text{- - - -}}_{\text{context}}$ $\boxed{\text{text}}$ $\underbrace{\text{- - - - - -}}$

↳ $v_{text} \in \mathbb{R}^{300}$

if two words

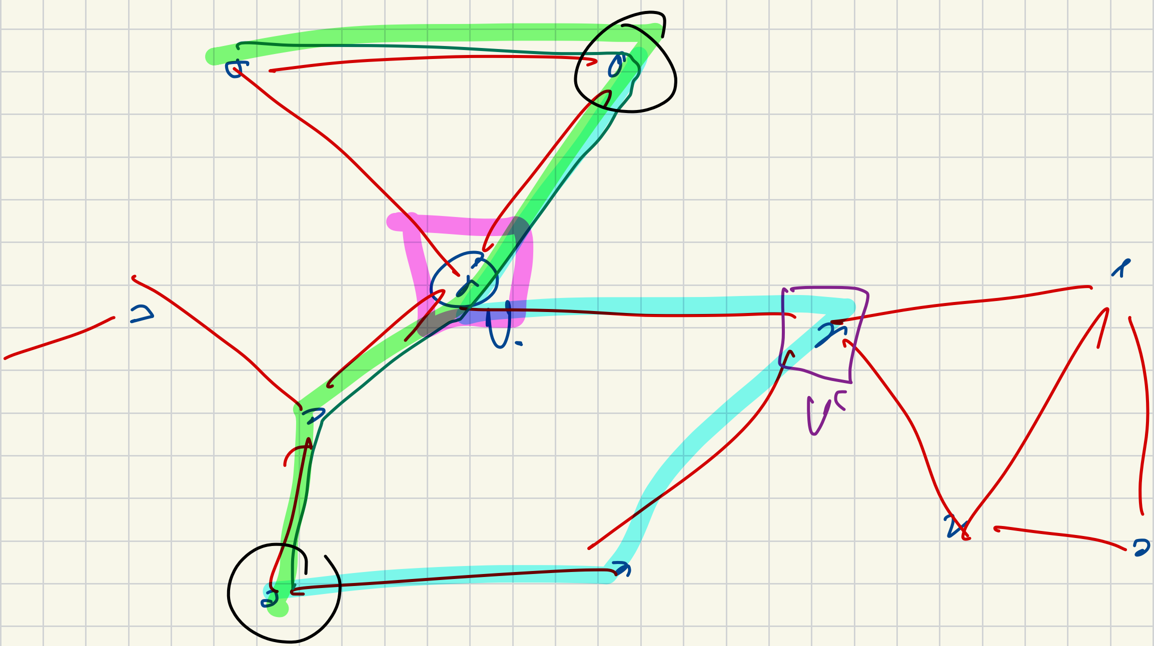
$\langle v_{text}, v_{word} \rangle$
large

if "text" "word"
had similar
context.

DeepWalk (Stonybrook)
node2vec (Stanford)

DeepWalk

Random Walk



context similarity matrix

context length L

scaling factor

$$C = Z \sum_{j \in L}' \frac{T-j}{|E|} D_A P^j$$

$\in \mathbb{R}^{n \times n}$

degree scaling

probability transition matrix

to power j

C_{ij} = probability node v_j is in the context (size L) of node v_i

Instead of eigen-embedding

$\mapsto$ Stochastic Gradient Descent.

loss function

if large $\downarrow$ then this large

$$L(X;Y) = -\sum_{v_i \in X} \sum_{v_j \in Y} C_{ij} \log\left(\frac{\exp(\langle x_i, y_j \rangle)}{\sum_{i=1}^{n} \exp(\langle x_i, y_j \rangle)}\right)$$

embedding

words in context

Warnings about embeddings

"especially language/words"

Learn representation from data