Uncertainty and Robustness of Graph Embeddings

Aleksandar Bojchevski
Technical University of Munich, Germany

Graph Embedding Day 2018 - Lyon

Neglected aspects of graph embeddings

Capturing uncertainty

Robustness to noise

Robustness to adversarial attacks

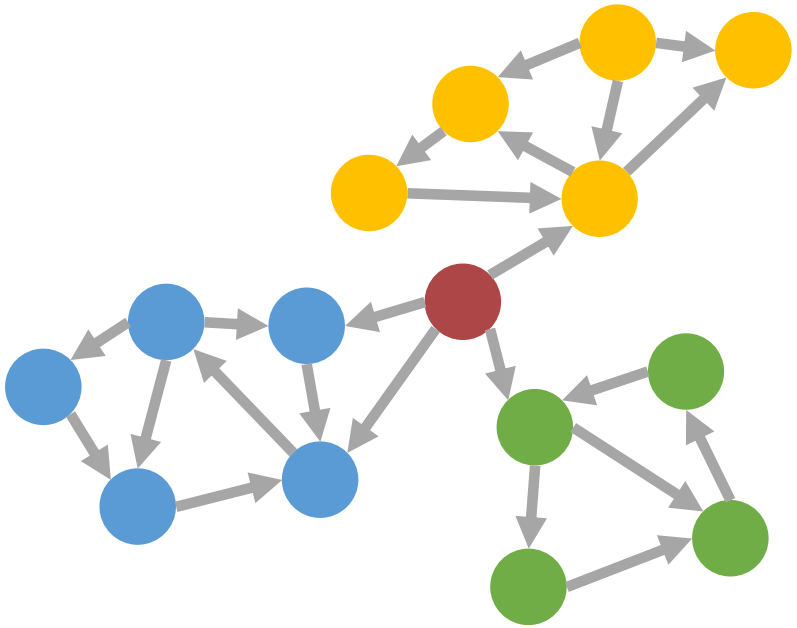
Neglected aspects of graph embeddings

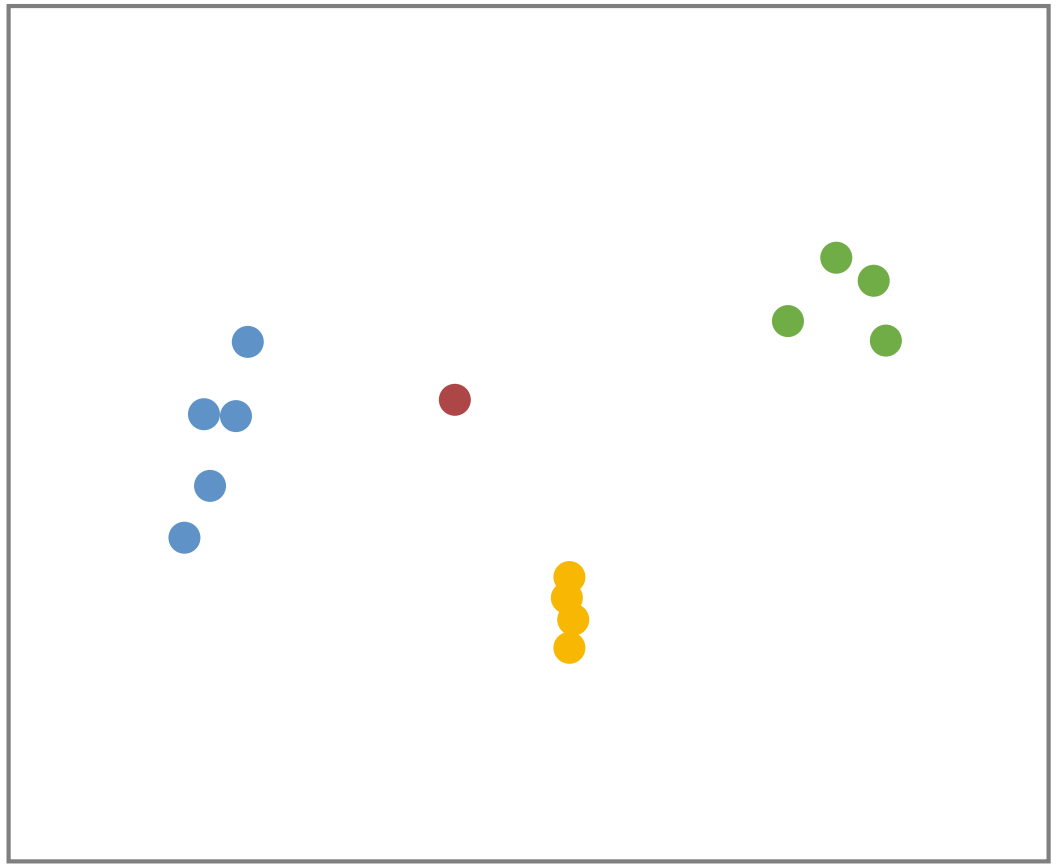
Capturing uncertainty

Robustness to noise

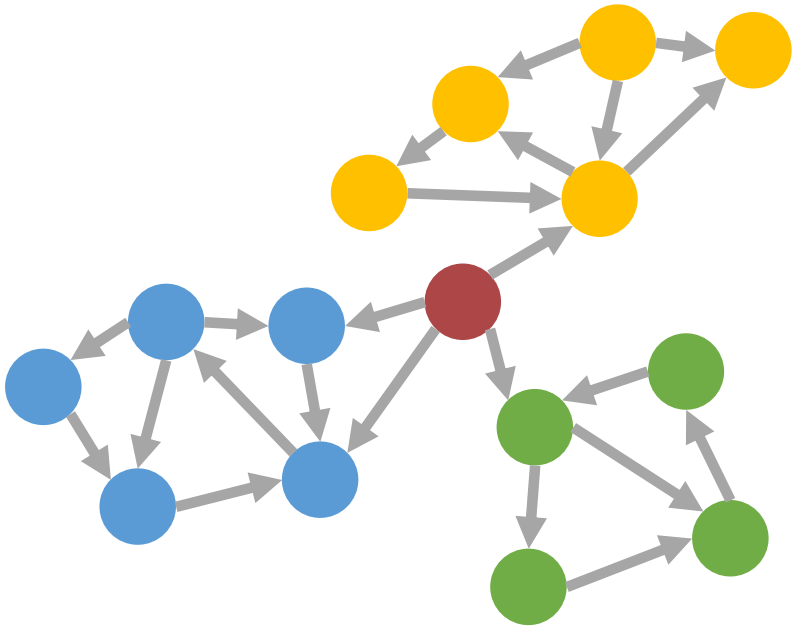
Robustness to adversarial attacks

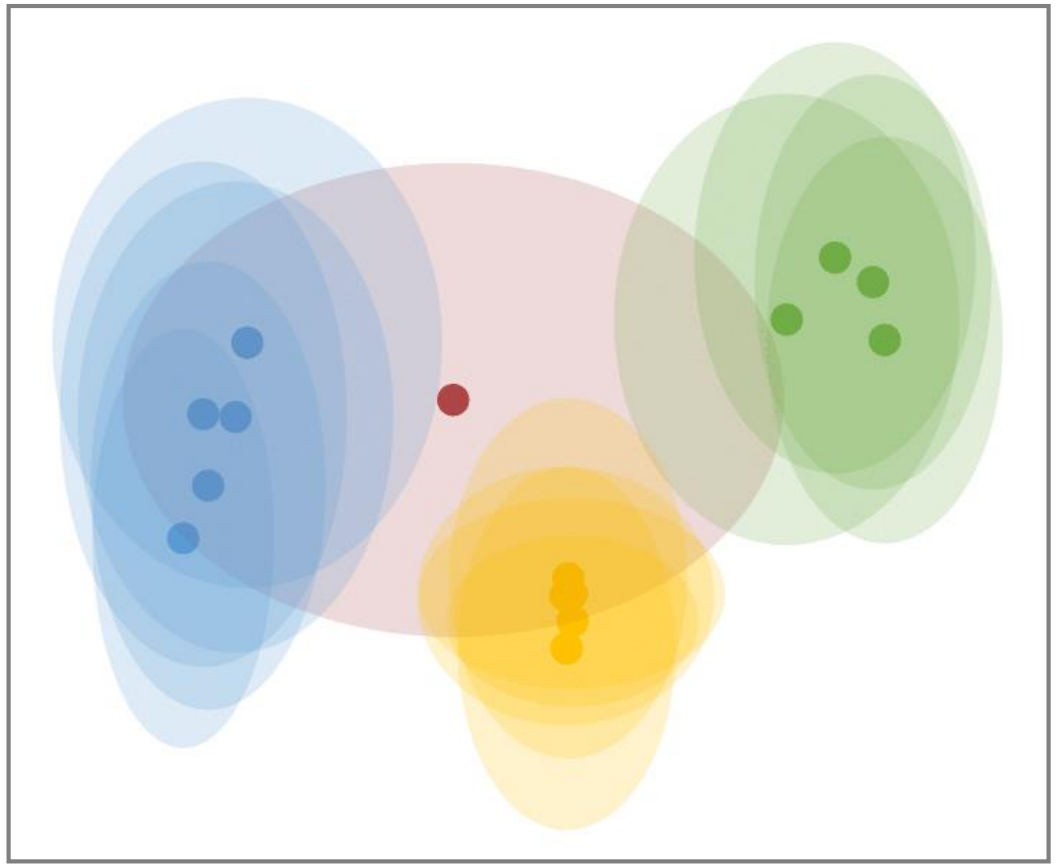
Nodes are points in a low-dimensional space





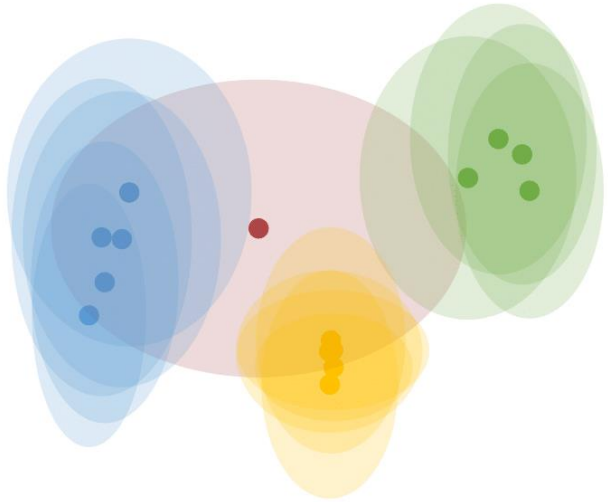
Nodes are distributions



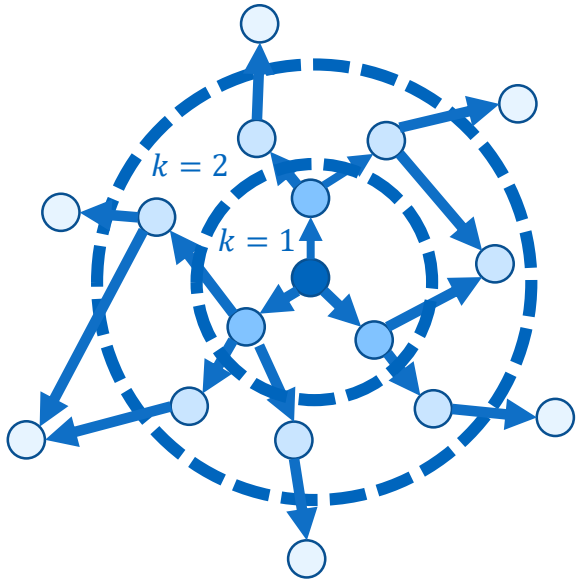


Graph2Gauss - 3 key modeling ideas

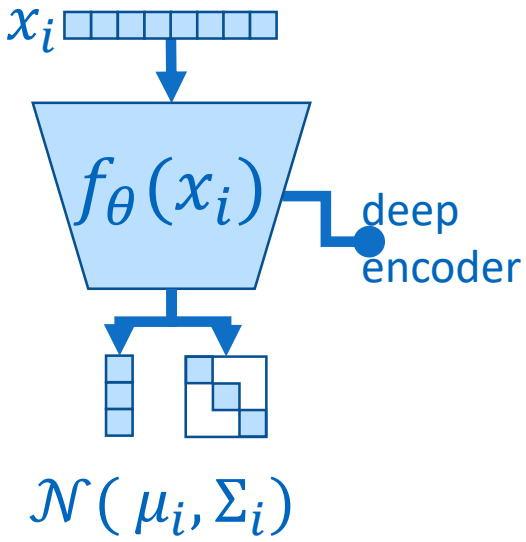
1. Uncertainty



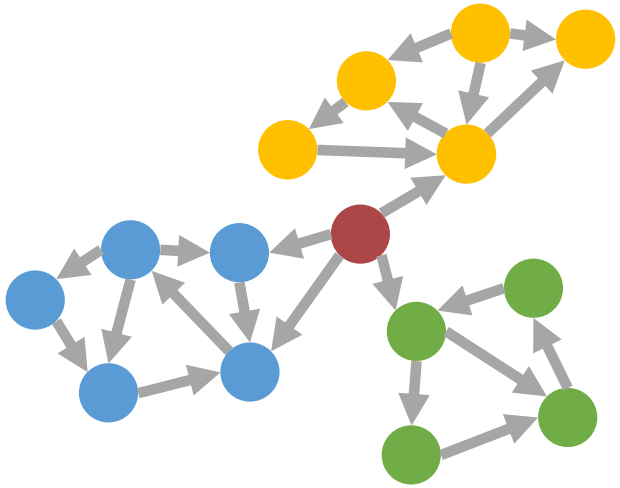
2. Personalized ranking



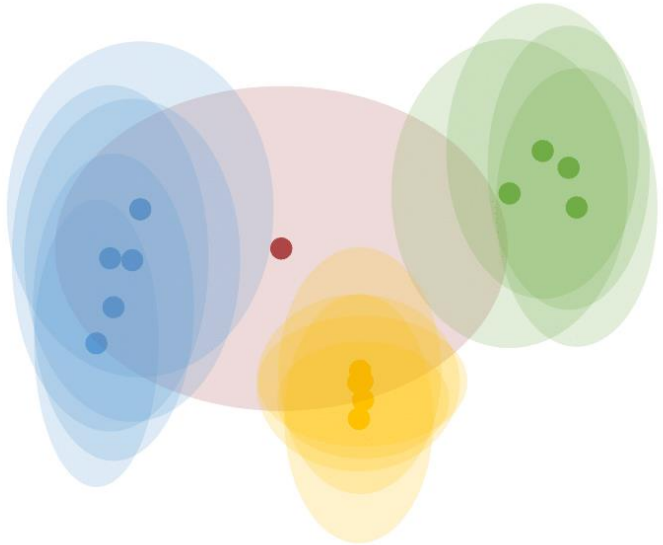
3. Inductiveness



Uncertainty



Embed nodes as (Gaussian) distributions

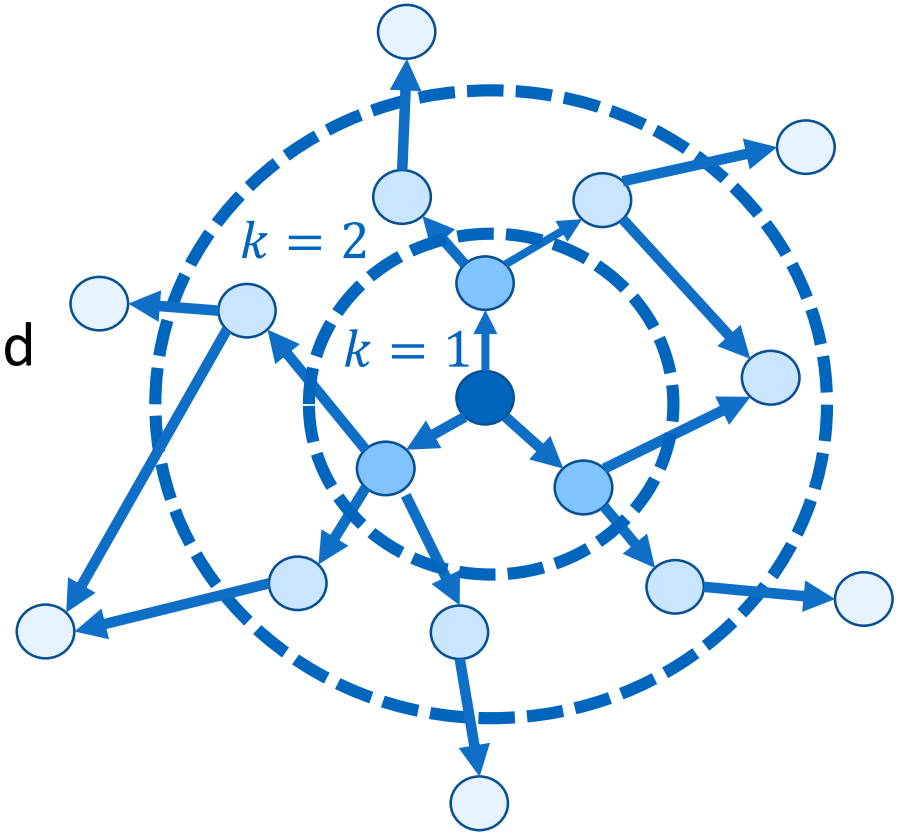


Sources of uncertainty:

- Conflicting structure and attributes
- Heterogenous neighborhood
- Noise, outliers, anomalies, ….

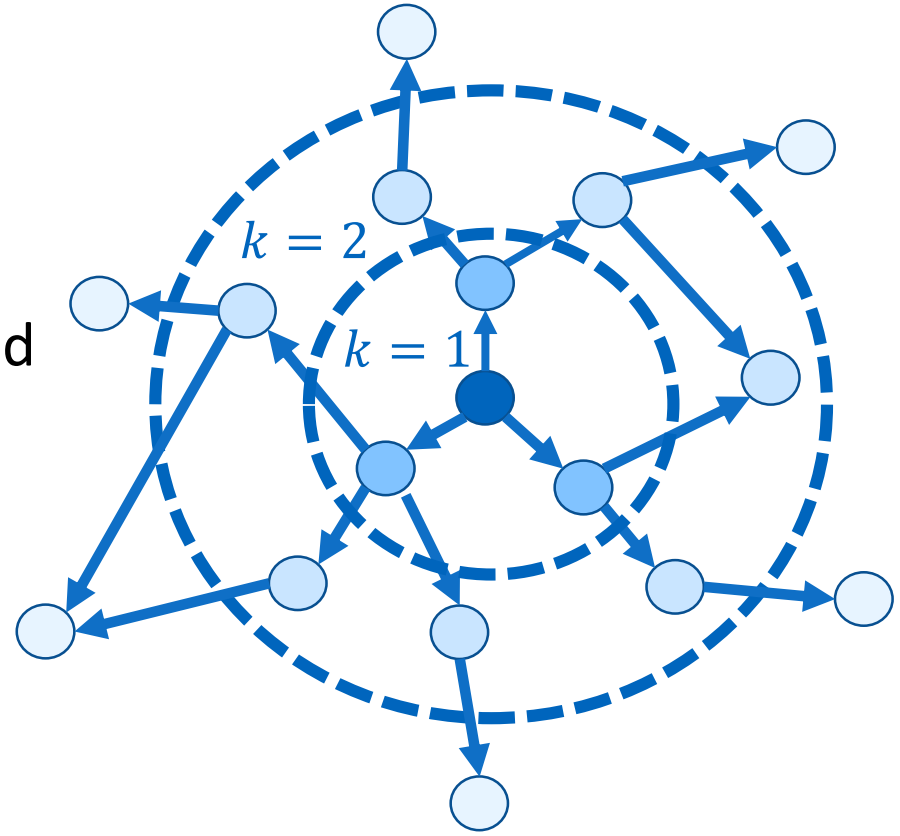
Personalized ranking

For each node i: nodes in its (k)-hop neighborhood should be closer to i compared to nodes in its $(k+1)$-hop neighborhood



Personalized ranking

For each node i: nodes in its (k)-hop neighborhood should be closer to i compared to nodes in its $(k+1)$-hop neighborhood

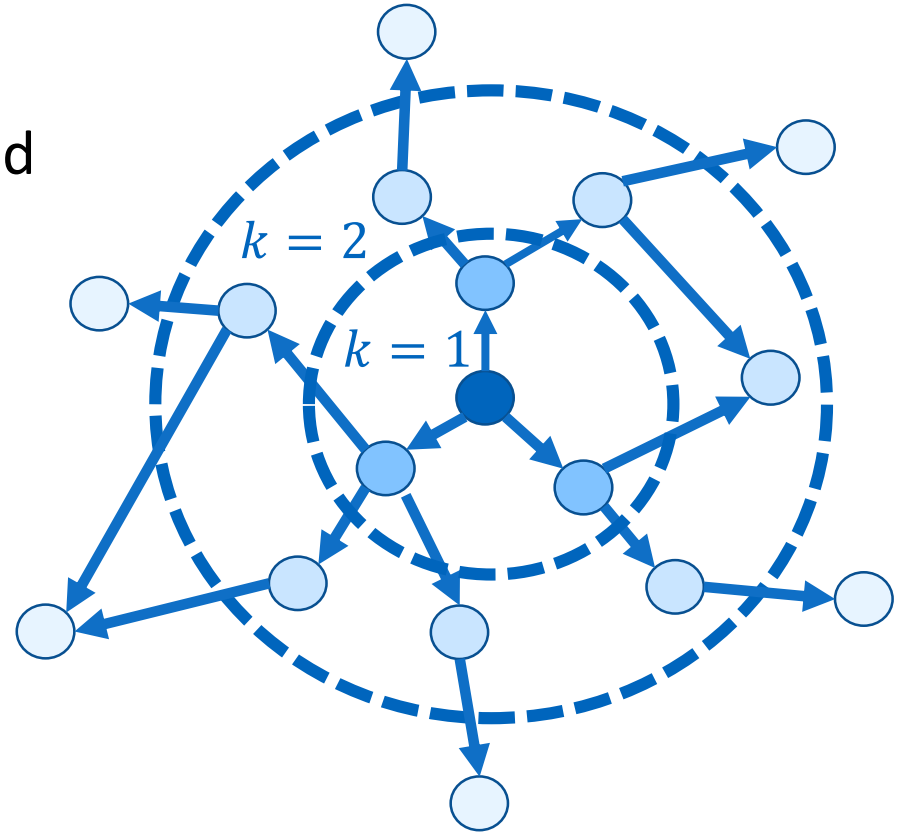


Personalized ranking

For each node i: nodes in its (k)-hop neighborhood should be closer to i compared to nodes in its $(k+1)$-hop neighborhood

Example: closer in terms of the KL Diveregence

KL is asymmetric ⇒ handles directed graphs



Personalized ranking

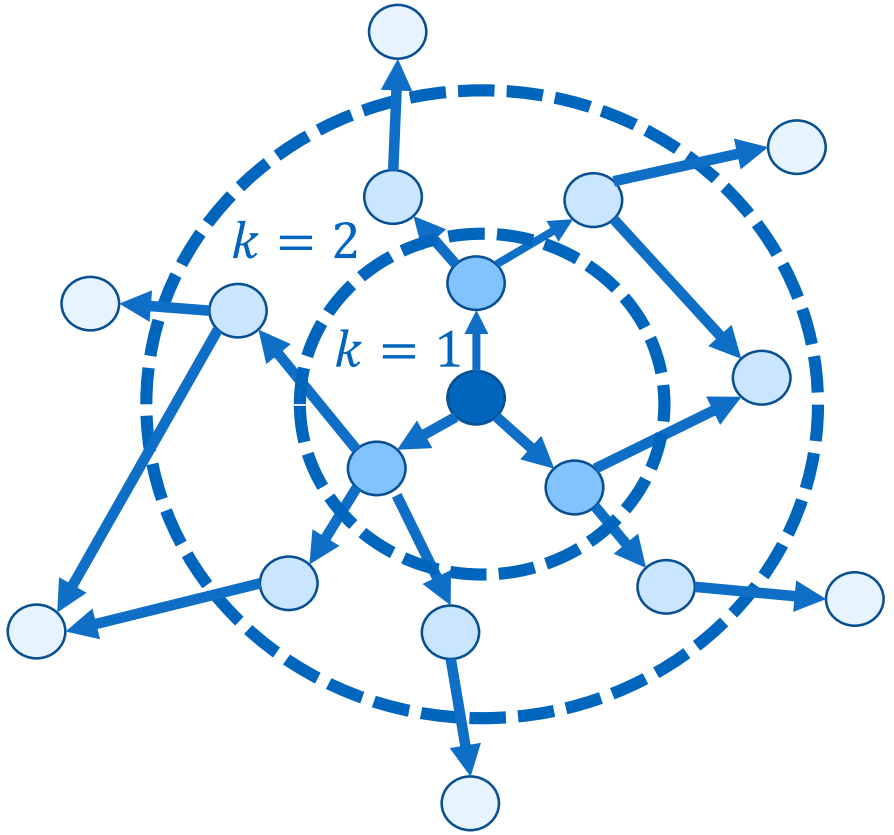
Personalized ranking implies

pairwise constraints for node i

$$\mathrm{D}_{KL}(\mathcal{N}_j||\mathcal{N}_i) < \mathrm{D}_{KL}\,(\mathcal{N}_{j'}||\mathcal{N}_i)$$
$$\forall j \in N_i^{(k)}, \forall j' \in N_i^{(k')}, \forall k < k'$$

↓

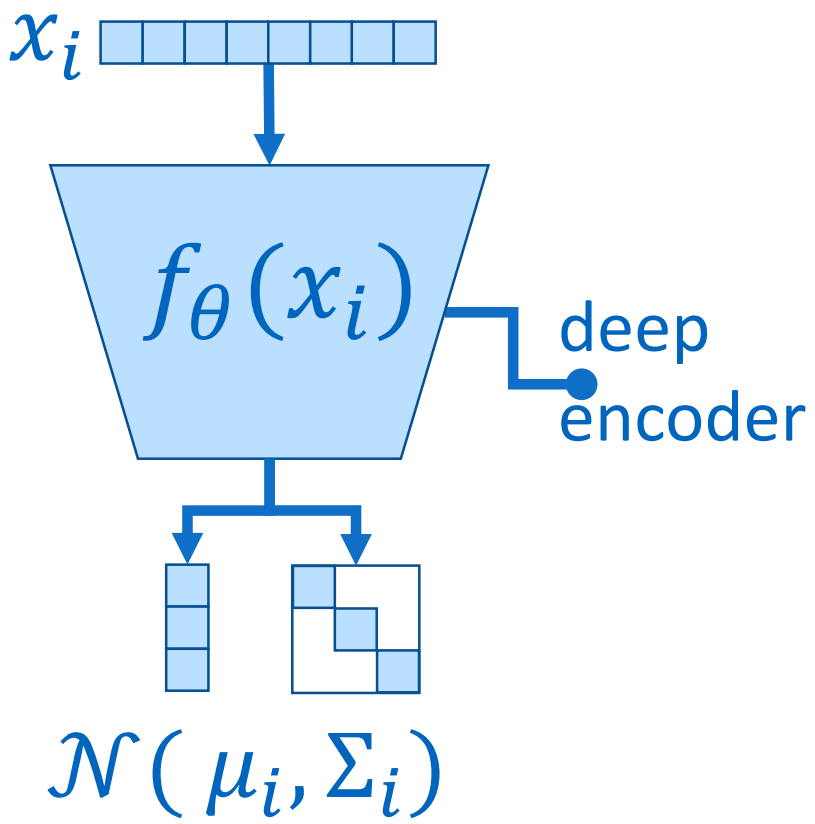
set of nodes in the k-hop neighborhood of node i



Inductiveness

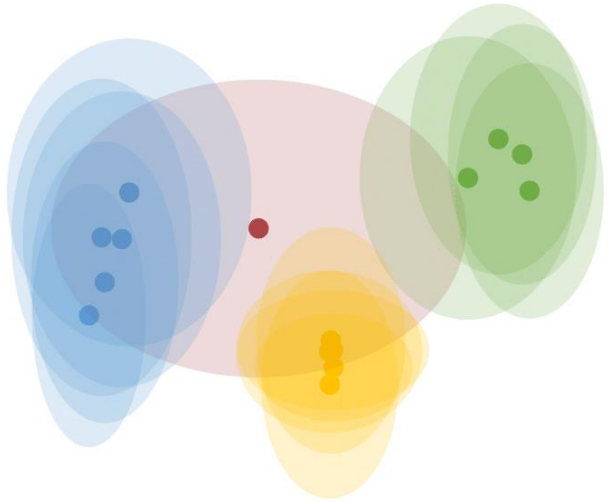
Generalize to unseen nodes by learning

a mapping from features to embeddings

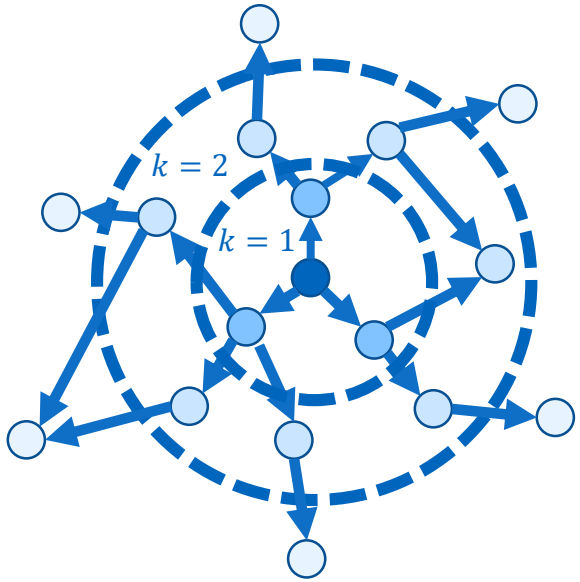


Graph2Gauss - 3 key modeling ideas

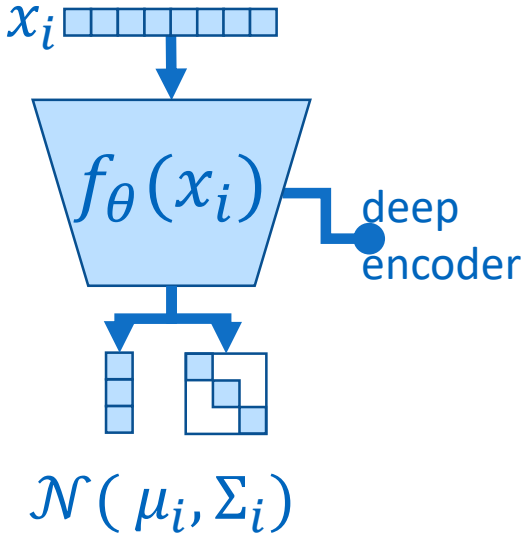
1. Uncertainty



2. Personalized ranking



3. Inductiveness



Learning with energy-based loss

$$E_{ij} = \mathrm{D}_{KL}(\mathcal{N}_j || \mathcal{N}_i) \qquad \mathcal{L} = \sum_{(i,j,j')} (E_{ij}^2 + \exp^{-E_{ij'}})$$

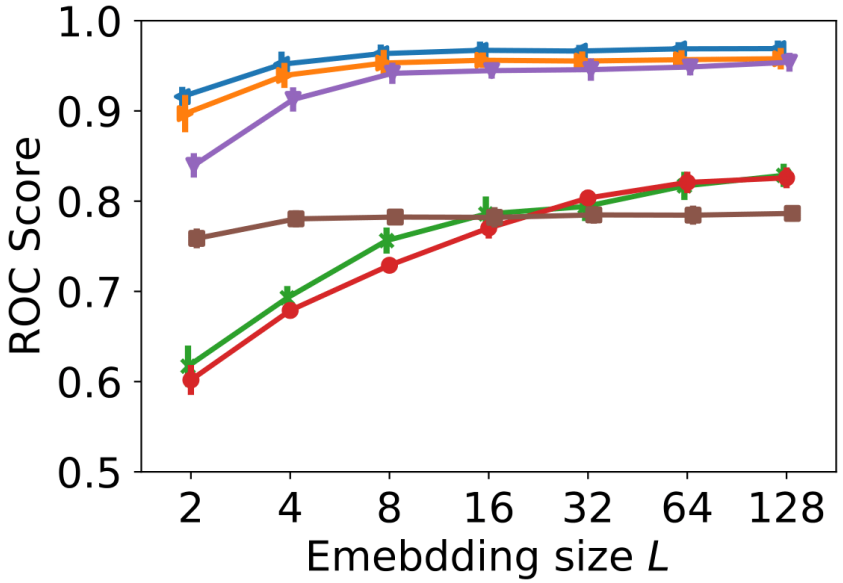
Closer nodes should have lower energy

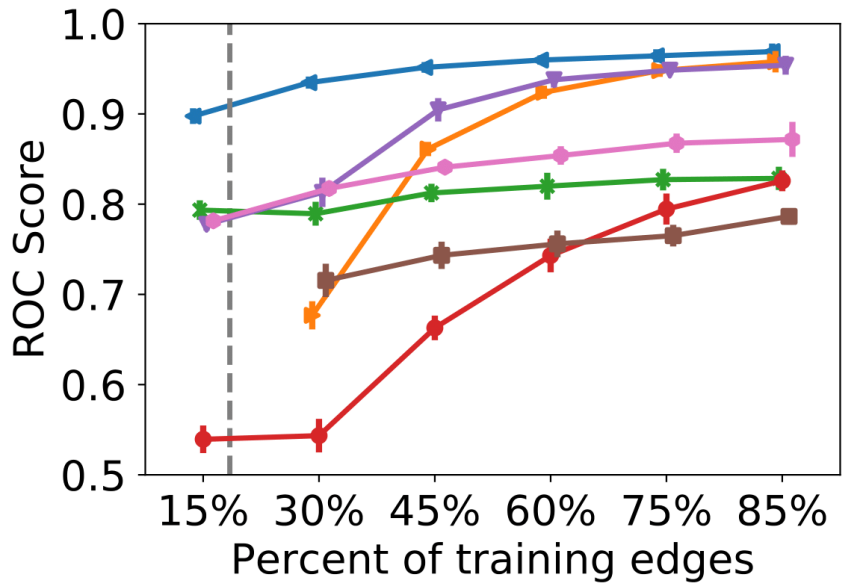
Naively: $O(N^3)$ complexity

Node-anchored sampling strategy:

- For each node same one another node from every neighborhood
- Less than 4.2% triplets seen to match performance
- Lower gradient variance

Graph2Gauss is parameter/data efficient

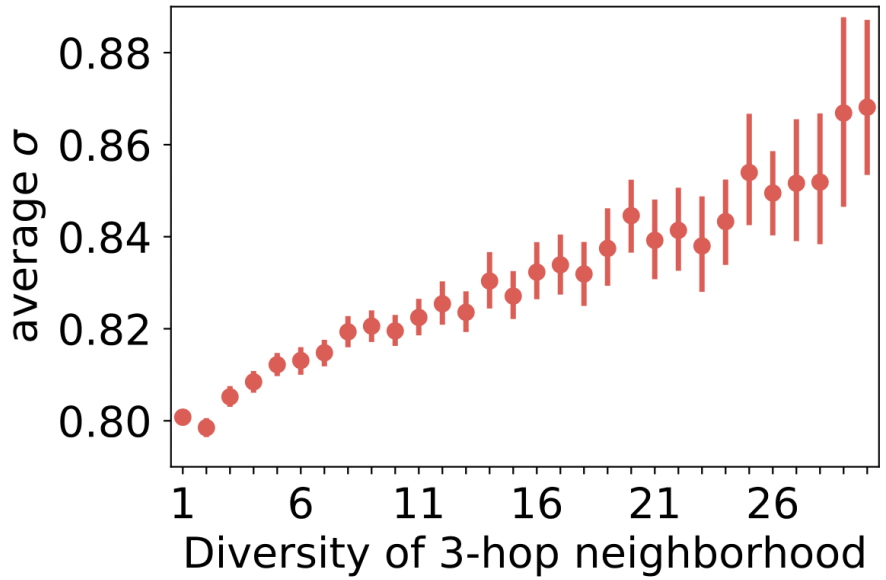




Graph2Gauss captures uncertainty

Uncertainty correlates with diversity

Diversity: number of distinct classes in a node's k-hop neighborhood



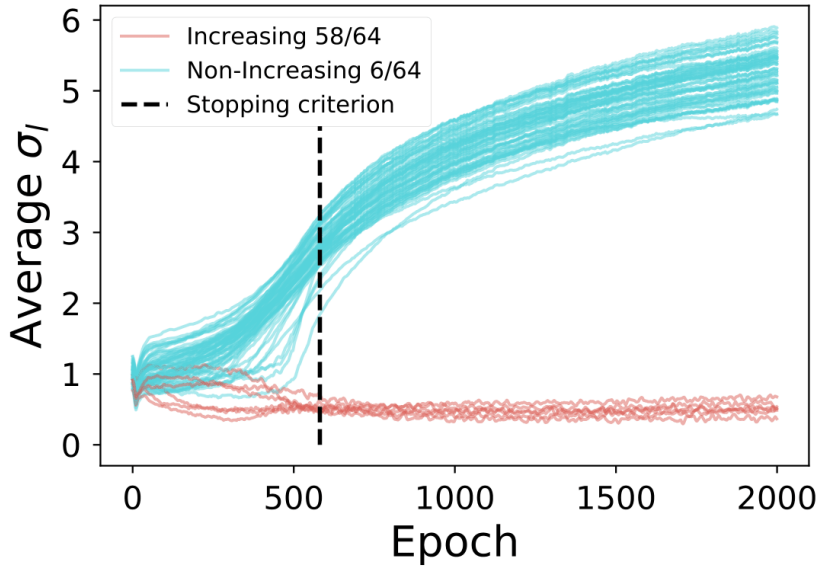
Graph2Gauss captures uncertainty

Uncertainty reveals the

intrinsic latent dimensionality of the graph

Detected latent dimensions

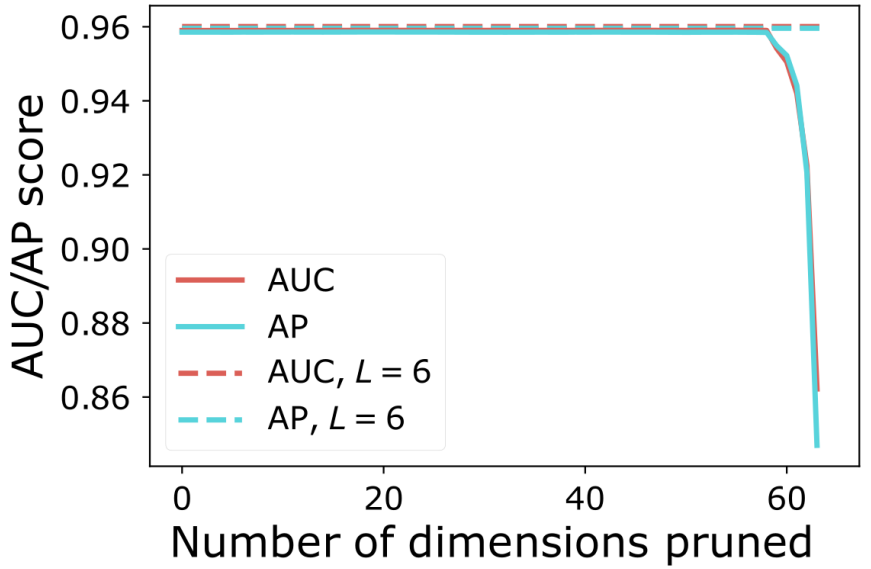
≈ number ground-truth communities



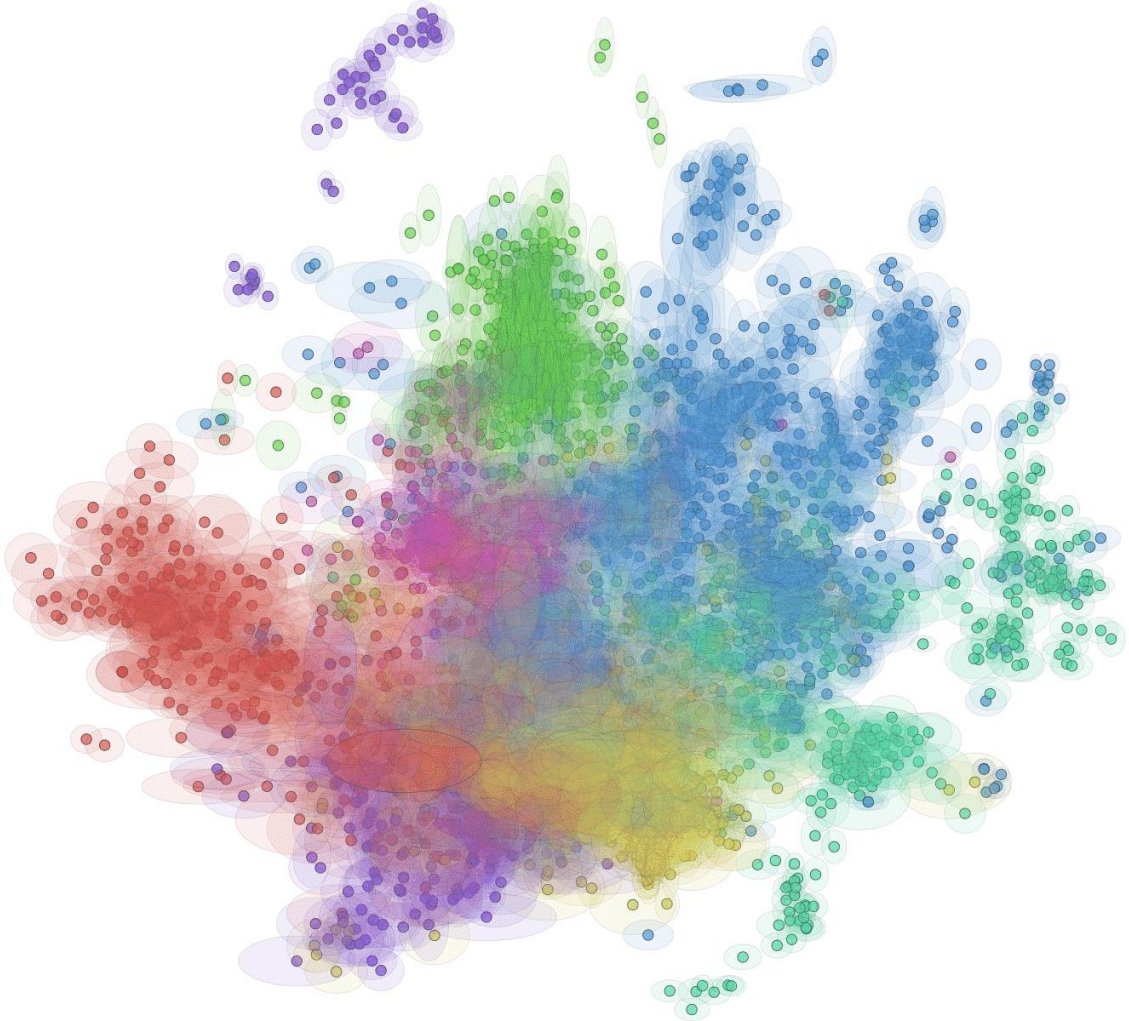
Uncertainty and link prediction

Prune dimensions with high uncertainty

Maintaining link prediction performance



Graph2Gauss is effective for visualization



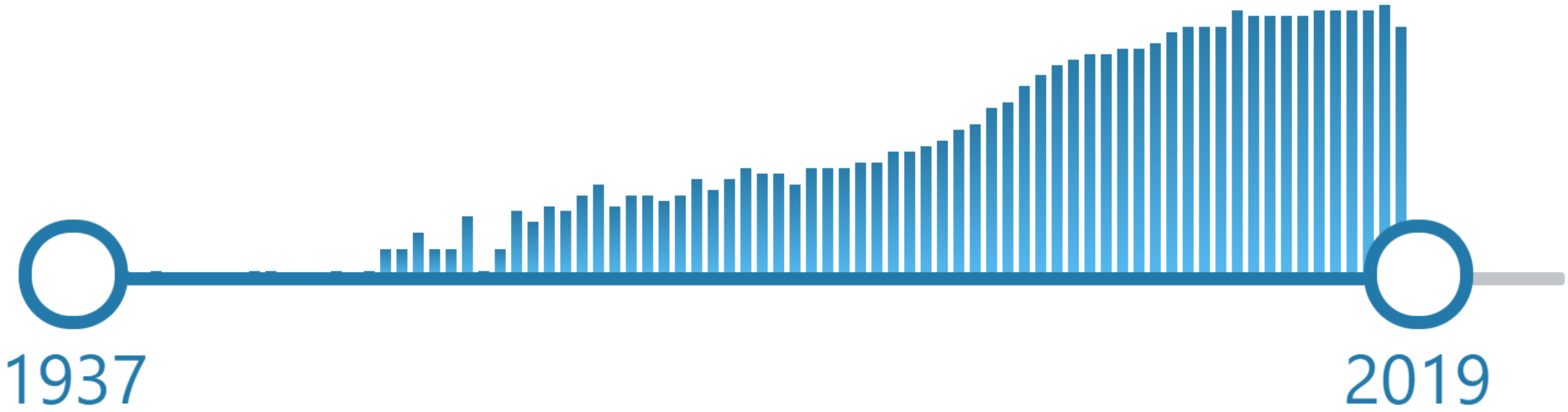
Neglected aspects of graph embeddings

Capturing uncertainty

Robustness to noise

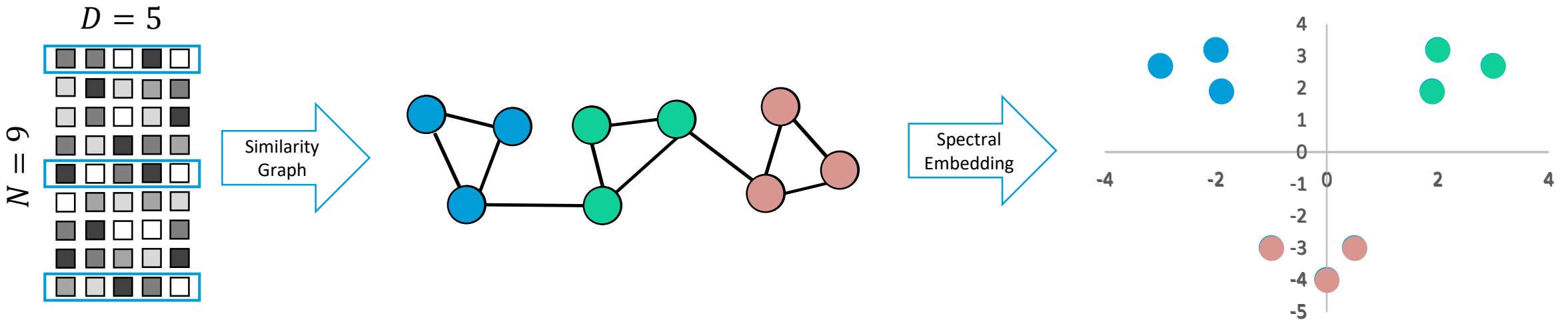
Robustness to adversarial attacks

Why spectral embedding



https://www.semanticscholar.org

What is spectral clustering

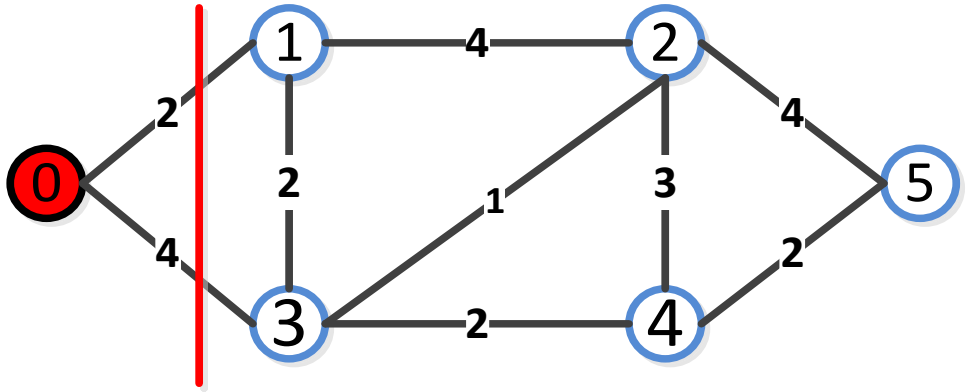


Graph clustering

- Maximize within-cluster edges
- Minimize between cluster edges

The minimum cut

Partition V into two sets C_1 and C_2, such that the sum of the inter-cluster edge weights $\text{cut}(C_1, C_2) = \sum_{v_1 \in C_1, v_2 \in C_2} w(v_1, v_2)$ is minimized



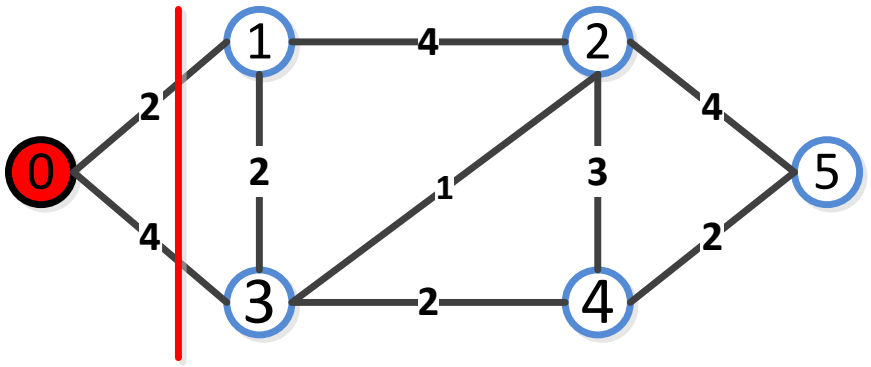
Drawbacks:

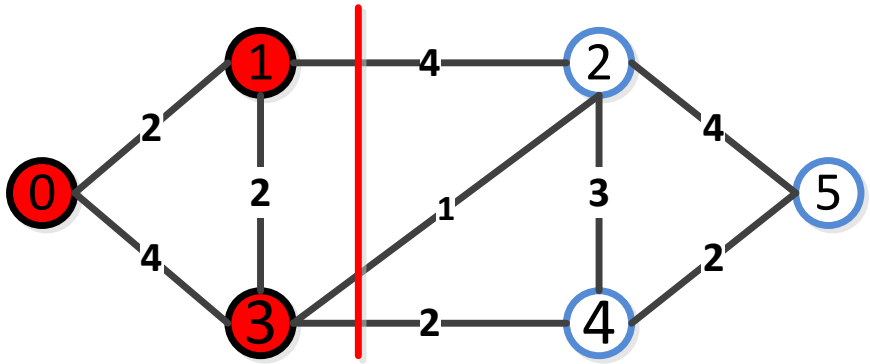
- Tends to cut small vertex sets from the rest of the graph
- Considers only inter-cluster edges, no intra-cluster edges

The normalized cut

Ratio Cut: Minimize $\frac{cut(C_1,C_2)}{|C_1|} + \frac{cut(C_2,C_1)}{|C_2|}$

Normalized Cut: Minimize $\frac{cut(C_1,C_2)}{\mathrm{vol}(C_1)} + \frac{cut(C_1,C_2)}{\mathrm{vol}(C_2)}$



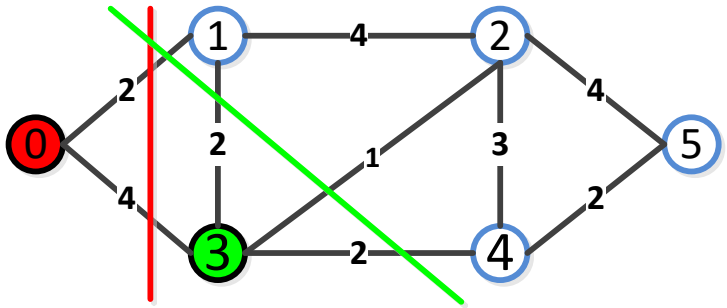


Multi-way graph partitioning

Generalization to $k \geq 2$ clusters

Partition V into disjoint clusters $C_1, \ldots, C_k$ such that

- Cut: $\min_{C_1,\ldots,C_k} \sum_{i=1}^{k} cut(C_i, V \backslash C_i)$
- Ratio Cut: $\min_{C_1,\ldots,C_k} \sum_{i=1}^{k} \frac{cut(C_i, V \backslash C_i)}{|C_i|}$
- Normalized Cut: $\min_{C_1,\ldots,C_k} \sum_{i=1}^{k} \frac{cut(C_i, V \backslash C_i)}{vol(C_i)}$



Minimum Cut for $k = 3$

Finding the optimal solution is NP-hard

How to compute an approximate solution efficiently?

Graph Laplacian

Laplacian matrix $L = D - A$

- A = (weighted) adjacency matrix, D = degree matrix

Observation: For any vector f we have

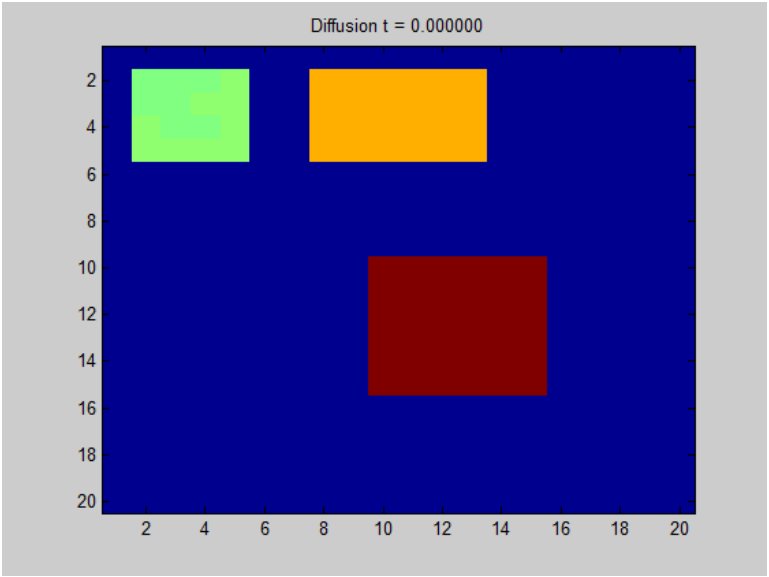
$$f^T \cdot L \cdot f = \frac{1}{2} \cdot \sum_{(u,v) \in E} W_{uv}(f_u - f_v)^2$$

Normalized Laplacian $L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$

Physical interpretation of the Laplacian (I)

Let f be a heat distribution over a graph with f_i = the heat at node v_i

The heat transferred between v_i and v_j is prop. to $(f_i - f_j)$ if $(i, j) \in E$



https://en.wikipedia.org/wiki/Laplacian_matrix#/media/File:Graph_Laplacian_Diffusion_Example.gif

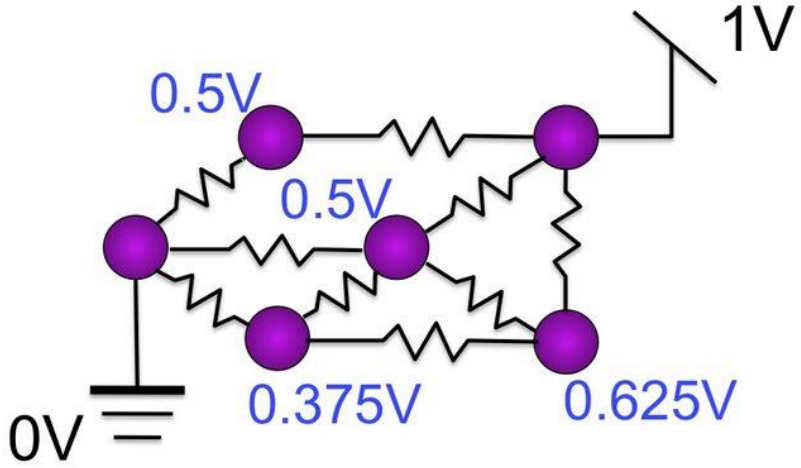
Physical interpretation of the Laplacian (I)

Graph is viewed as an electrical circuit with edges as wires (resistors)

Apply voltage at some nodes and measure induced voltage at other nodes

Induced voltages minimizes $\sum_{(u,v)\in E}(x_u - x_v$

We can find the voltage by minimizing $x^T Lx$



Properties of the Graph Laplacian

L is symmetric and positive semi-definite

The number of eigenvectors of L with eigenvalue 0 corresponds to the number of connected components

Algebraic connectivity of a graph is $\lambda_2(L)$

- The magnitude reflects how well connected the graph overall is

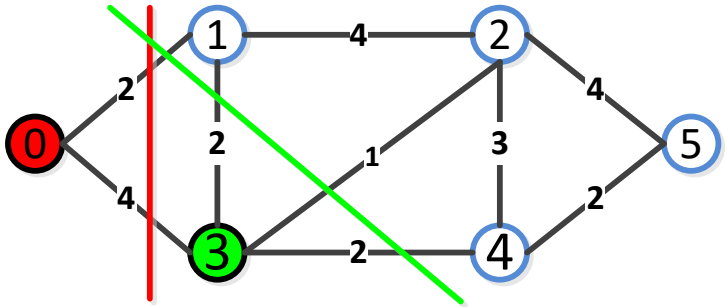
The spectrum of L encodes useful information about the graph

- Unfortunately, there exist co-spectral graphs

Minimum cut and the graph Laplacian

Define indicator vector: : $h_{C_k}[i] = \begin{cases} \frac{1}{\sqrt{|C_i|}} & if\ v_i \in C_k \\ 0 & else \end{cases}$

Let $\mathrm{H} = [h_{C_1}; h_{C_2}; \dots; h_{C_k}]$



Observations:

$H^T H = Id$ is orthonormal

$h_{C_i}^T \cdot L \cdot h_{c_i} = \frac{cut(C_i, V \setminus C_i)}{|C_i|}$ and $h_{C_i}^T \cdot L \cdot h_{c_i} = (H^T L H)_{ii}$

$$RatioCut(C_1, \dots, C_k) = \sum_{i=1}^{k} \frac{cut(C_i, V \setminus C_i)}{|C_i|} = \sum_{i=1}^{k} (H^T L H)_{ii} = trace(H^T L H)$$

Minimum cut and the graph Laplacian

Minimizing ratio-cut (normalized cut with L_{sym}) is equivalent to

$$\min_{C_1,\dots,C_k} trace(H^T L H) \text{ subject to } H^T H = Id$$

Constraint relaxation: allow arbitrary values for H

$$\min_{H \in R^{V \times K}} trace(H^T L H) \text{ subject to } H^T H = Id$$

Standard trace minimization problem

Optimal H = First K smallest eigenvectors of L

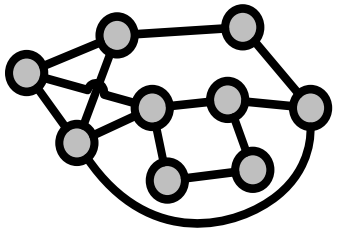
Spectral embedding: random walk view

$L_{rw} = D^{-1}L = I - D^{-1}A = I - P$ is the the random walk Laplacian

- λ is an eigenvalue of L_{rw} with eigenvector u if and only if λ is an eigenvalue of L_{sym} with eigenvector $w = D^{1/2}u$

Let $P(B|A) = P(X_1 \in B | X_0 \in A)$ be the probability of a random walker currently at any node in A to transition to any node in B, for $A \cap B = \emptyset$ and $A, B \subset V$.
Sample $X_0 \sim \pi$ from the stationary distribution $Ncut(A, \bar{A}) = P(\bar{A}|A) + P(A|\bar{A})$

Spectral Embedding



$$L(A) = D(A) - A$$

$$H^* = \arg \min_{H \in \mathbb{R}^{n \times d}} Trace(H^T \cdot L(A) \cdot H)$$
subject to $H^T \cdot D_A \cdot H = Id$

Input
Graph A

Graph Laplacian
$L(A)$

Trace minimization
Ratio/Normalized Cut

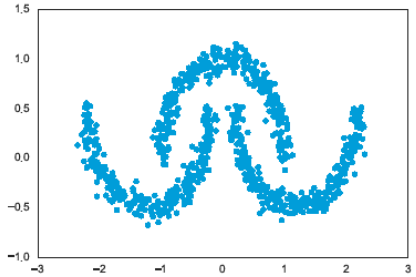
Output
Embedding H^*

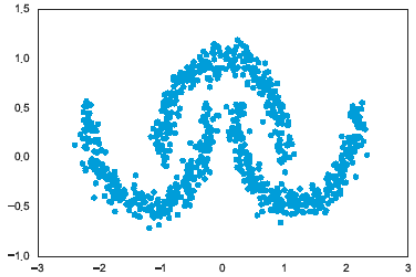
Finding the spectral embedding = Solving an optimization task

$H^* =$ k-first eigenvectors of $L(A)$

Problem: sensitive to noisy data

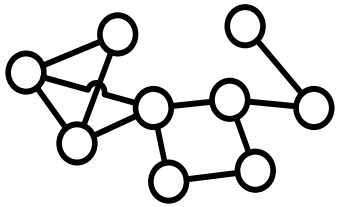
Clean

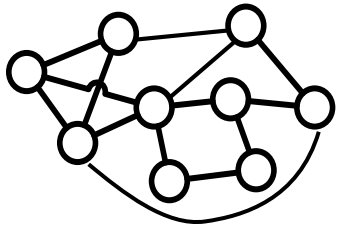




Noisy

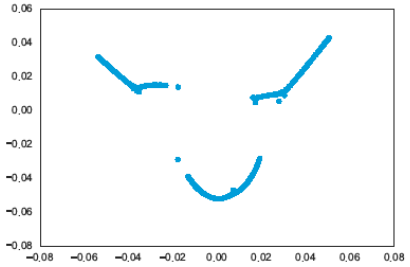
⇒

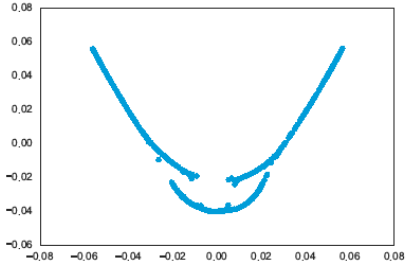




Spurious edges

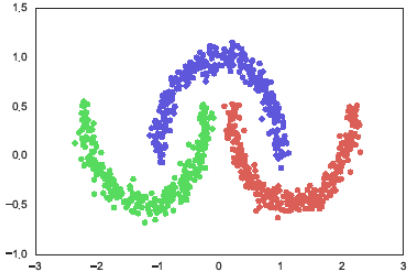
⇒

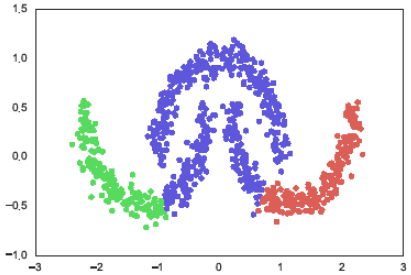




Distorted embedding

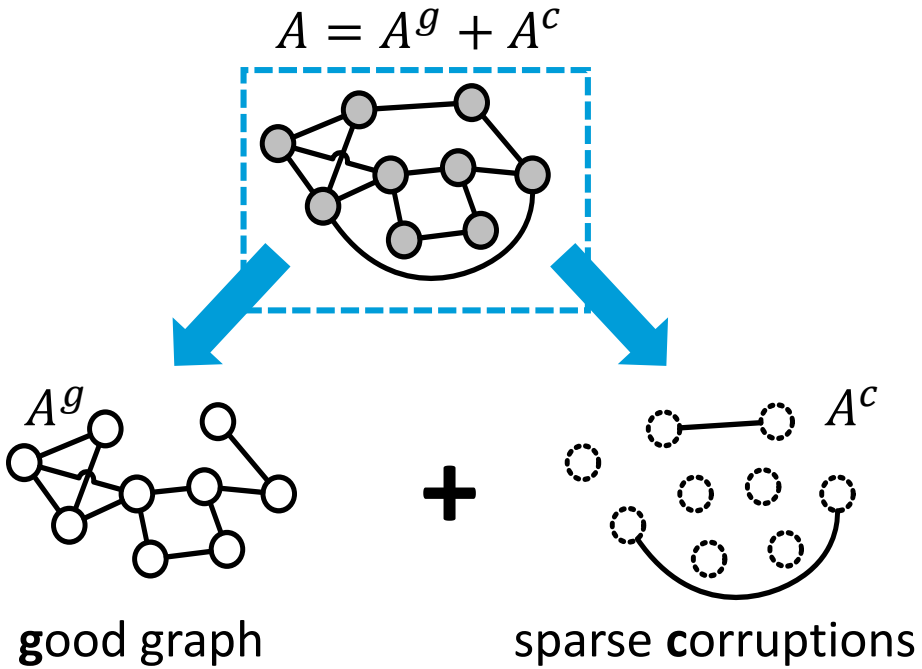
⇒





Wrong clustering

Robustness via Latent Decomposition



$$H^* = \arg\min_{H \in \mathbb{R}^{n \times d}} Trace(H^T \cdot L(A^g) \cdot H)$$
$$\text{subject to } H^T \cdot D(A^g) \cdot H = Id$$

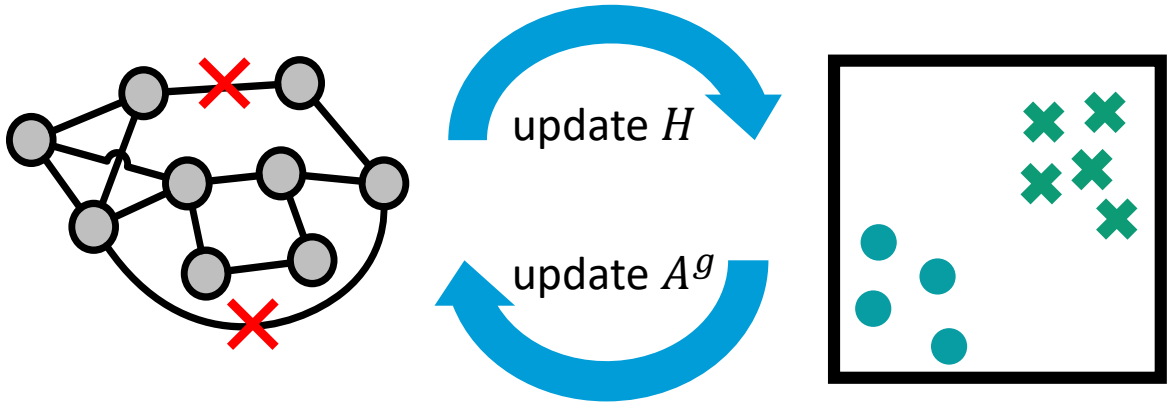
$$A^*, H^* = \arg\min_{\substack{H \in \mathbb{R}^{n \times d} \\ A^g \in (\mathbb{R}_{\geq 0})^{n \times n}}} Trace(H^T \cdot L(A^g) \cdot H)$$
$$\text{subject to } H^T \cdot D(A^g) \cdot H = Id$$
$$A = A^g + A^c$$
$$\|A^c\|_0 \leq 2\theta \quad \text{global}$$
$$\forall_i : \|a_i^c\|_0 \leq \omega_i \quad \text{local}$$

Jointly learn decomposition & embedding

Decomposition steered by the underlying embedding / clustering

Solution: Alternating optimization

$$A^*, H^* = \arg \min_{\substack{H \in \mathbb{R}^{n \times d} \\ A^g \in (\mathbb{R}_{\geq 0})^{n \times n}}} Trace(H^T \cdot L(A^g) \cdot H)$$



Update H, Given $A^g / A^c \rightarrow Easy$

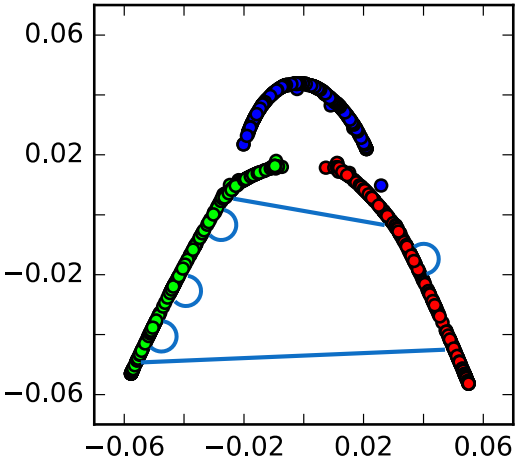
- Trace minimization problem
- Solution for H are the k first generalized eigenvectors of $L(A^g)$

Solution: Alternating optimization

$$A^*, H^* = \arg \min_{\substack{H \in \mathbb{R}^{n \times d} \\ A^g \in (\mathbb{R}_{\geq 0})^{n \times n}}} Trace(H^T \cdot L(A^g) \cdot H)$$

$$f\left([a^c_{uv}]_{u,v \in E}\right) = \sum_{u,v \in E} a^c_{uv} \left(\underbrace{\|\boldsymbol{h}_u - \boldsymbol{h}_v\|_2^2}_{\substack{nodes\ far\ away\ in \\ the\ embedding\ space}} - \underbrace{\left\|\sqrt{\lambda \circ \boldsymbol{h}_u}\right\|_2 - \left\|\sqrt{\lambda \circ \boldsymbol{h}_v}\right\|_2}_{\substack{prefers\ edges\ close \\ to\ the\ origin}} \right)$$

subject to $\|\cdot\|_0$ constraints



Update A^g / A^c, Given $H \rightarrow (NP)\ Hard$

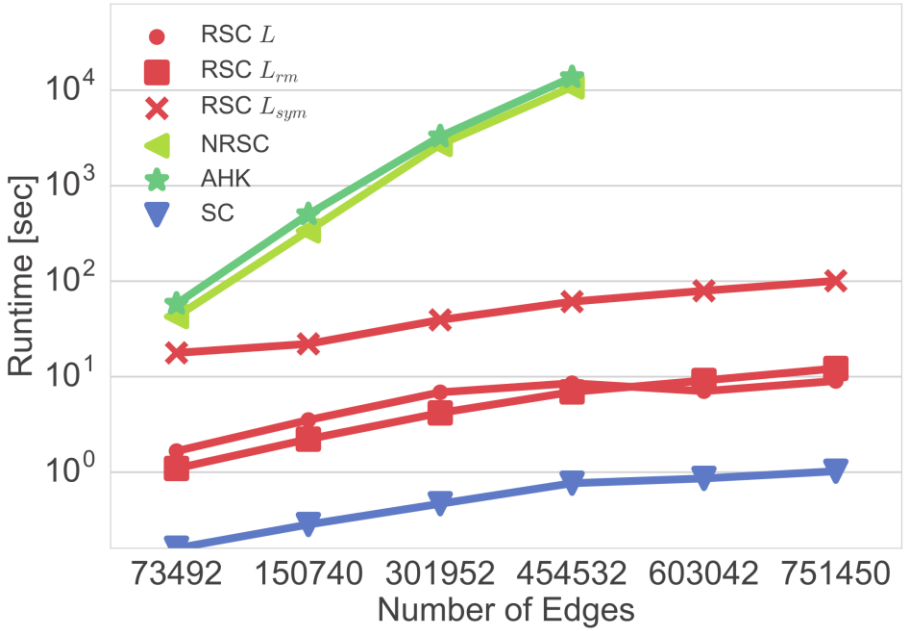
- Express eigenvalues of A^g_{new} in closed form
- A^g_{new} that minimizes the trace equivalent to maximizing f

Solution: Alternating optimization

Equivalent to Multidimensional Knapsack problem

- Greedy approximation
- Best possible approximation ratio of $\frac{1}{\sqrt{N+1}}$

Efficient solution in O(#edges)



jupyter Robust Spectral Clustering Last Checkpoint: 7 hours ago (autosaved)

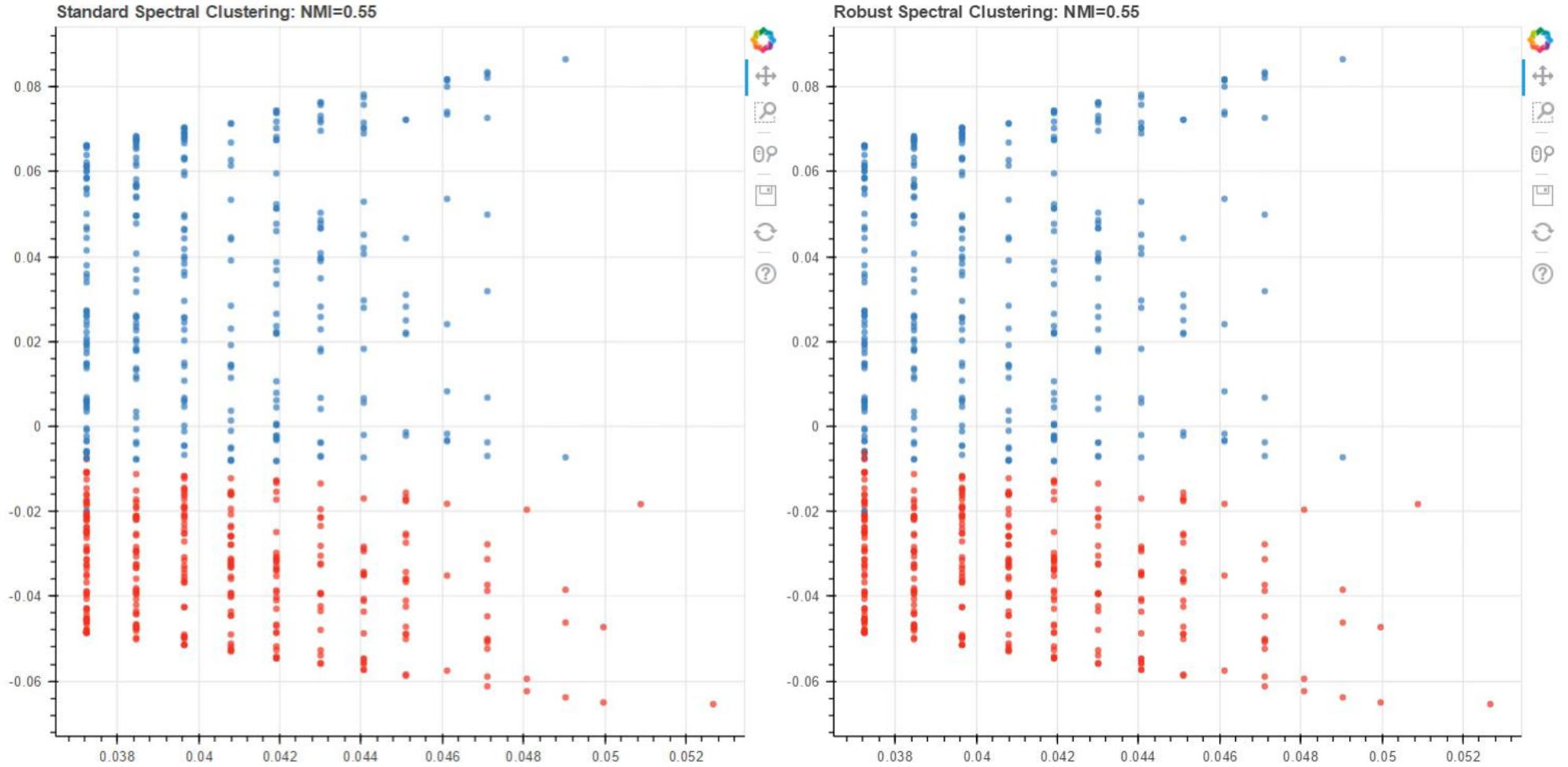
Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python [Root]

Code

theta 0



Conclusion

Spectral embedding is sensitive to noisy data

Robustness via latent decomposition

$$\underbrace{A}_{\substack{original\\graph}} = \underbrace{A^g}_{\substack{good\\graph}} + \underbrace{A^c}_{\substack{sparse\\corruptions}}$$

Removed corrupted edges ⇒ increased discrimination

Neglected aspects of graph embeddings

Capturing uncertainty

Robustness to noise

Robustness to adversarial attacks

Adversarial attacks on graph embeddings

(Spectral) Embeddings are not robust to noise / but we can remedy that

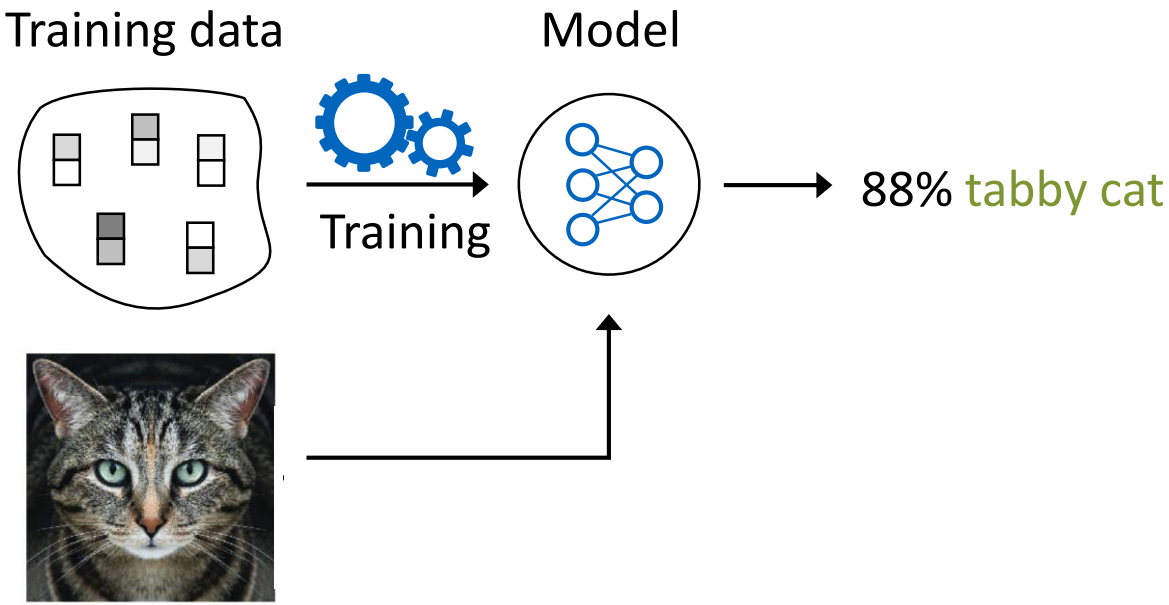
Are graph embeddings robust to adversarial attacks?

In domains where graph embeddings are used (e.g. the Web)

adversaries are common and false data is easy to inject

Adversarial attacks in the image domain

Image of a tabby cat correctly classified

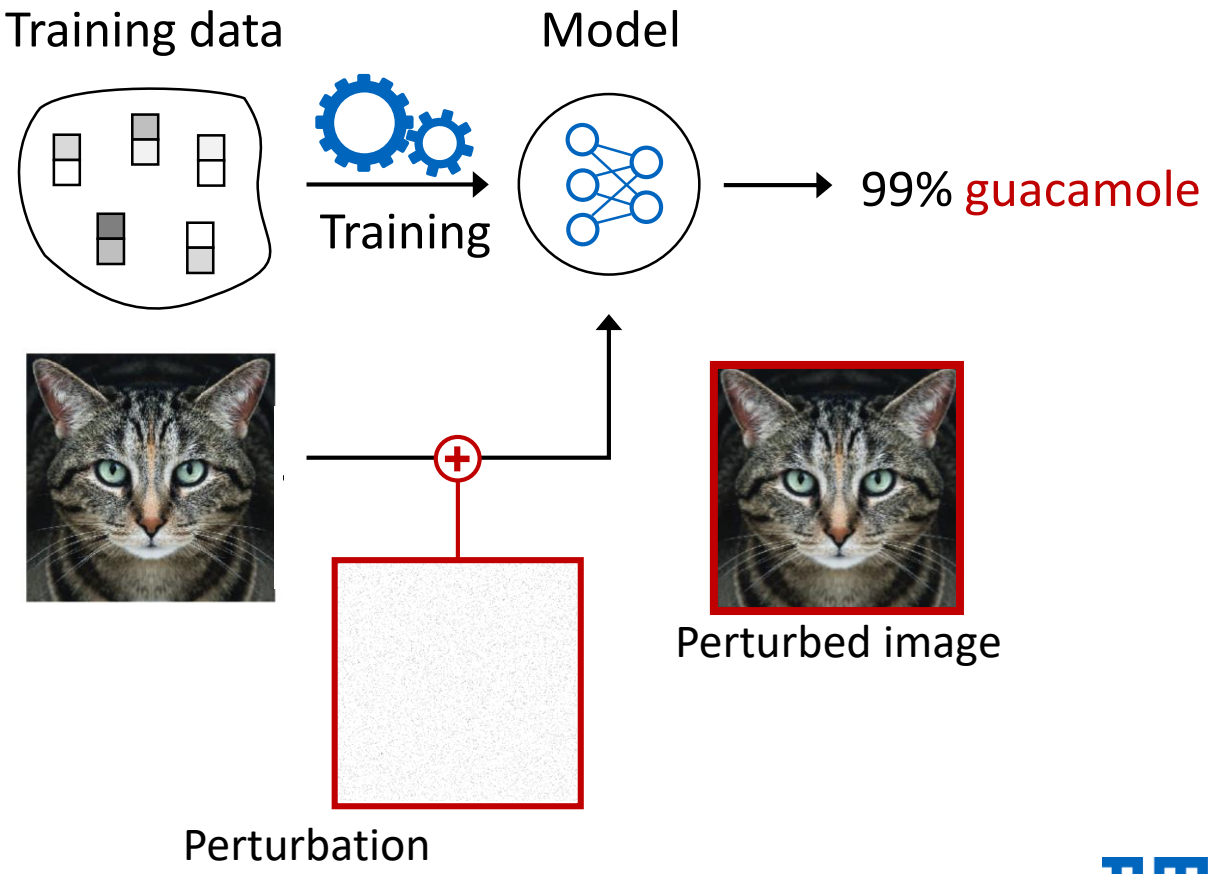


Adversarial attacks in the image domain

Image of a tabby cat correctly classified

Add **imperceptible** perturbation

Model classifies the cat as guacamole



The relational nature of the data might

Improve Robustness	Cause Cascading Failures
embeddings are computed jointly rather than in isolation	perturbations in one part of the graph can propagate to the rest

Attack possibilities

General attack

Goal: decrease the overall quality of the embeddings

Actions:

- Add/remove (flip) an edge
- Add/remove a node
- …

Targeted attack

Goal: attack a specific node or a specific downstream task

Examples:

- Misclassify a target node t
- Increase/decrease the similarity of a set of node pairs $\mathcal{T} \subset V \times V$

Attack model formally

Adjacency matrix of the graph after the attacker modified some entries

$$\hat{A}^* = \arg \max_{\hat{A} \in \{0,1\}^{N \times N}} \mathcal{L}(\hat{A}, Z^*)$$

$$Z^* = \min_{Z} \mathcal{L}(\hat{A}, Z) \; subj.to \; \left\| \hat{A} - A \right\|_0 = 2f$$

Optimal embedding from the to be optimized graph $\hat{A}$

The attacker's budget

Attack model formally

Adjacency matrix of the graph after the attacker modified some entries

$$\hat{A}^* = \arg\max_{\hat{A}\in\{0,1\}^{N\times N}} \mathcal{L}(\hat{A}, Z^*)$$

General attack

$$Z^* = \min_{Z} \mathcal{L}(\hat{A}, Z) \; subj.to \; \left\|\hat{A} - A\right\|_0 = 2f$$

Optimal embedding from the to be optimized graph $\hat{A}$

The attacker's budget

Attack model formally

Adjacency matrix of the graph after the attacker modified some entries

$$\hat{A}^* = \arg \max_{\hat{A} \in \{0,1\}^{N \times N}} \mathcal{L}_{atck}(\hat{A}, Z^*)$$

Targeted attack

$$Z^* = \min_{Z} \mathcal{L}(\hat{A}, Z) \; subj.to \; \left\| \hat{A} - A \right\|_0 = 2f$$

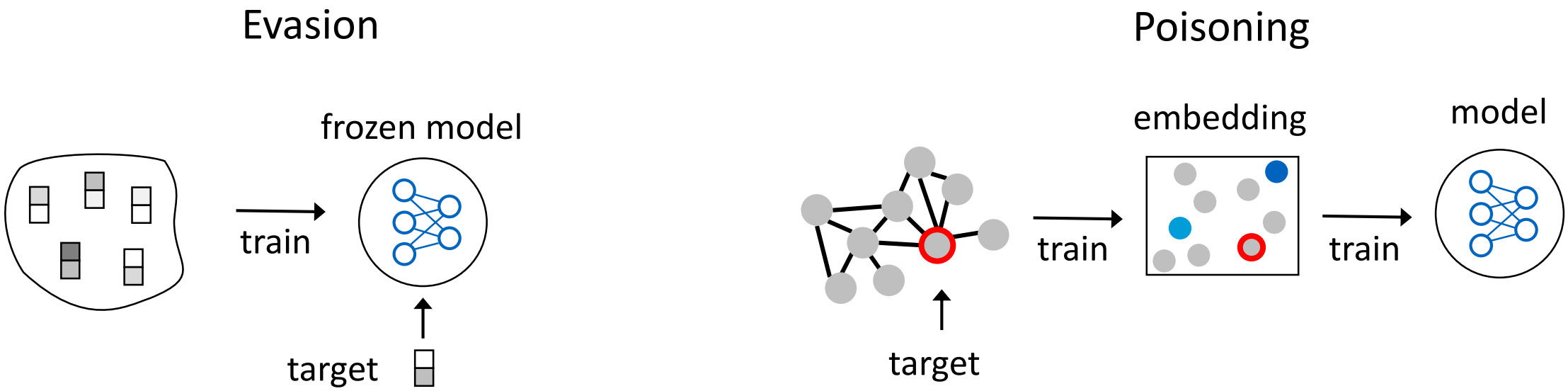
Optimal embedding from the to be optimized graph $\hat{A}$

The attacker's budget

Challenges

Discrete and Combinatorial Bi-level optimization problem

Transductive learning ⇒ network poisoning setting



Random-walk based embeddings

RW-based embeddings solve:

$$Z^* = \min_Z \mathcal{L}(\{r_1, r_2, \dots\}, Z) \text{ with } r_i = RW_l(A)$$

$Z^* \in \mathbb{R}^{N \times K}$: learned embedding

RW_l: e stochastic procedure that generates RWs of length l

$\mathcal{L}$: model-specific loss e.g. skip-gram with negative sampling (SGSN)

Challenge: RW sampling precludes gradient based optimization

Example: DeepWalk

DeepWalk is equivalent* to factorizing

$$\widetilde{M} = \log \max(M, 1)$$

Shifted Positive Pointwise Mutual Information (PPMI) Matrix

$$M = \frac{vol(A)}{T \cdot b} S$$

b negative samples

$$S = \left(\sum_{r=1}^{T} P^r\right) D^{-1}$$

window size T

$$P = D^{-1}A$$

transition matrix

with Z^* obtained by the SVD of $\widetilde{M} = U\Sigma V^T$ using the top K largest singular values/vectors i.e. $Z^* = U_K \Sigma_K^{1/2}$

Example: DeepWalk

Equivalent to $\min_{\widetilde{M}_K} ||\widetilde{M} - \widetilde{M}_K||_F^2$

The loss using the optimal embedding is $\mathcal{L}_{DW_1}(A, Z^*) = \sqrt{\sum_{p=K+1}^{|V|} \sigma_p^2}$, where

$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{|V|}$ are the singular values of $\widetilde{M}(A)$ ordered decreasingly

Idea: Given a perturbation ΔA, find the change in the singular values of $\widetilde{M}(A + \Delta A)$

Example: DeepWalk

$\widetilde{M} = \log\max(M, 1)$ $\quad M = \frac{vol(A)}{T \cdot b} S$ $\quad S = (\sum_{r=1}^{T} P^r) D^{-1}$

Linearization: ignore the $\log(\cdot)$ and $\max(\cdot,\ 1)$

Scalars $vol(A)$, T, b can be also ignore

Rewrite $\mathcal{L}_{DW_1}(A, Z^*) = \sqrt{\sum_{p=K+1}^{|V|} |\lambda_p|^2}$

Thus, find a change in the spectrum of S after the attacker perturbed the graph ΔA

Spectrum of S

Compute the generalized spectrum (generalized eigenvalues/vectors) of A

i.e. compute and U, Λ that solve $Au = \lambda Du$

Rewrite $S = (\sum_{r=1}^{T} P^r)D^{-1}$ as $S = U\,(\sum_{r=1}^{T} \Lambda^r)U^T$

simple function of the generalized eigenvalues λ_i of the graph

The task is now to find the change in generalized eigenvalues λ_p of the adjacency matrix A given a perturbation ΔA

Eigenvalue perturbation theory

Given U, Λ that solve $Au = \lambda Du'$ and a small perturbation $\Delta A, \Delta D$

Find U', Λ' that solve $(A + \Delta A)u' = \lambda'(D + \Delta D)u'$

First order approximation:

$$\lambda'_p = \lambda_p + u_p^T(\Delta A + \lambda_p \Delta D)u_p$$

for small ΔA and ΔD higher order terms become negligible

For a single edge flip

ΔA is a matrix with only 2 non-zero elements for a single edge flip (i, j)
namely $\Delta A_{ij} = \Delta A_{ji} = 1 - 2\mathrm{A}_{ij} \coloneqq \Delta \mathrm{w}_{ij}$

Similarly, $\Delta \mathrm{D}$ has only two non-zero elements on the diagonal

Then we can approximate the generalized eigenvalues of $\mathrm{A} + \Delta A$ in closed-form computable in O(1) time:

$$\lambda'_p = \lambda_p + \Delta \mathrm{w}_{ij}\left(2u_{pi} \cdot u_{pj} - \lambda_p(u_{pi}^2 + u_{pj}^2)\right)$$

Connecting it all together

1. DeepWalk is equivalent to a SVD of $\widetilde{M} = \log \max\left(\frac{vol(A)}{T \cdot b} S, 1\right)$
2. The loss can be computed from the singular values / the spectrum of S
3. The spectrum of S can be easily computed from the generalized spectrum of A
4. For any given edge flip (i, j) we can compute in O(1) the spectrum of $A + \Delta A$

Overall algorithm

However,

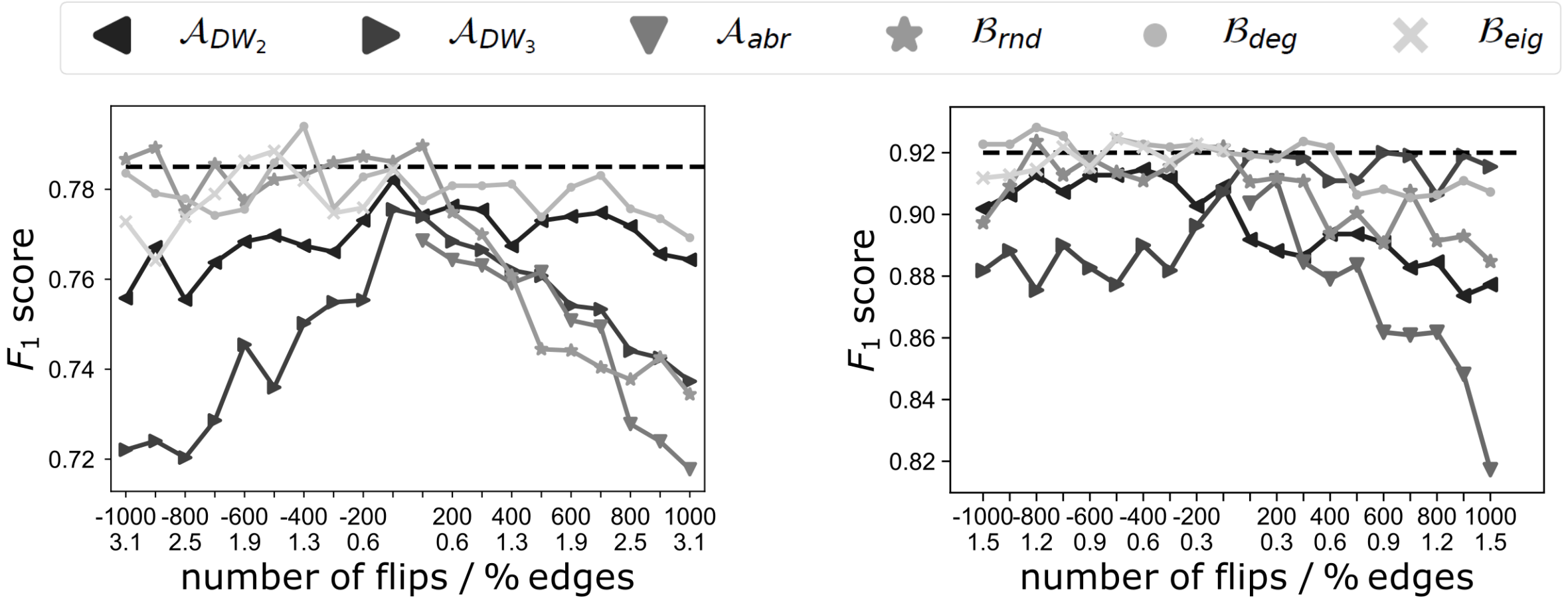
$$\hat{A}^* = \arg\max_{\hat{A}\in\{0,1\}^{N\times N}} \mathcal{L}_{closed-form} \quad subj.to \left\|\hat{A} - A\right\|_0 = 2f$$

is still hard to optimize – $\binom{N^2}{f}$ ways to choose the flips

Greedy solution:

1. For each edge (i, j) calculate its impact on the loss if flipped
2. Pick the top f edges

General attack (node classification proxy)



Targeted attack

To target node v we need the change in its embedding Z_v^*

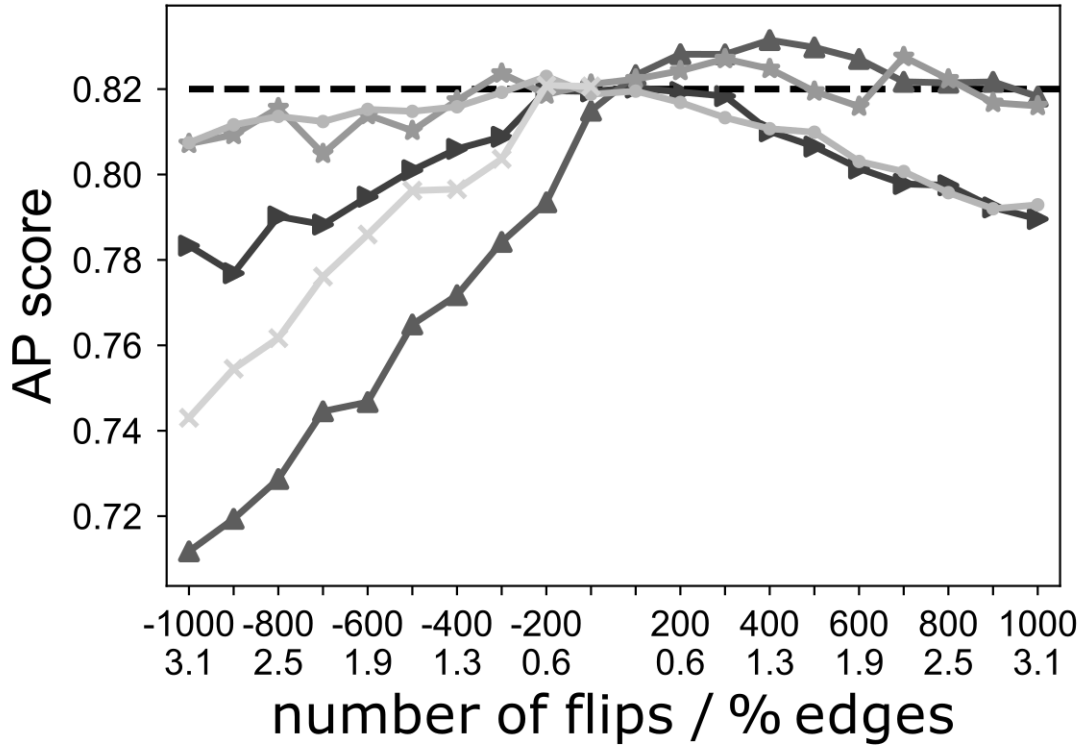
That is we need the change in eigenvectors

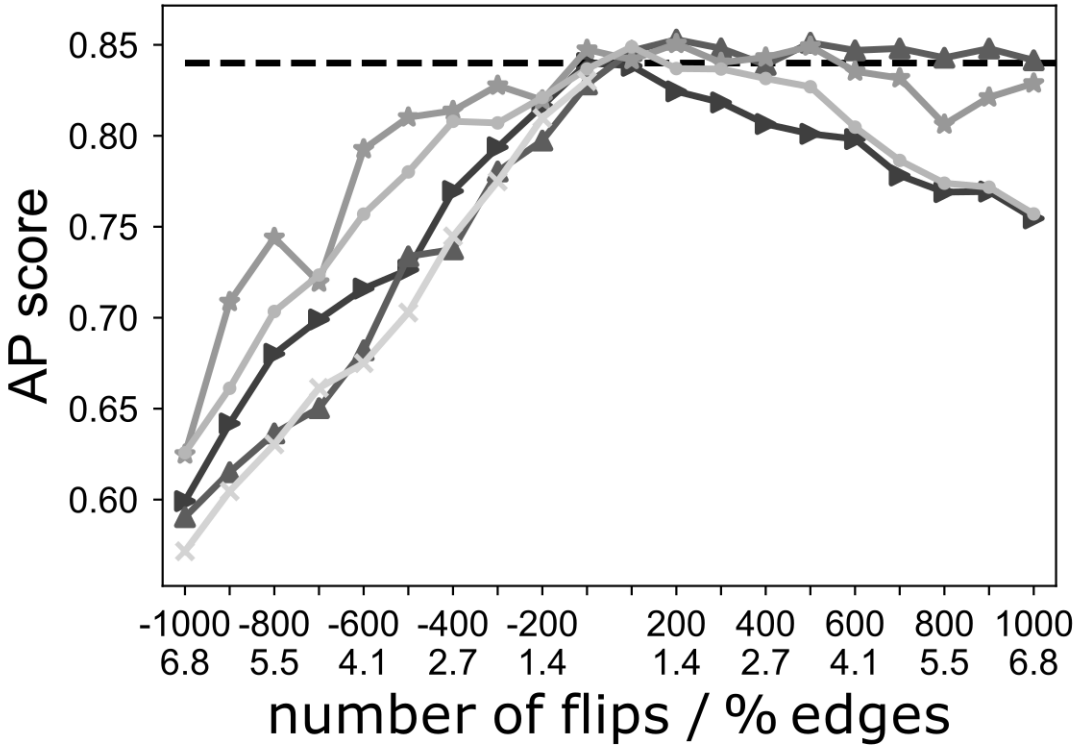
Apply eigenvalue perturbation again to approximate the top K eigenvectors

For a given edge flip (i, j) we get:

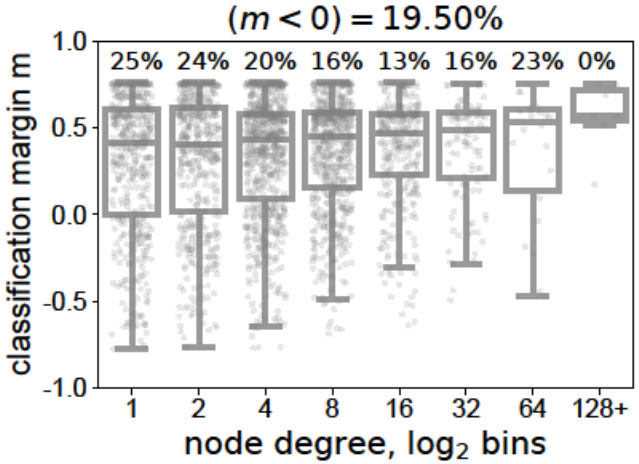
$$u'_p = u_p - \Delta w_{ij}(A - \lambda D)^+ \left(-\Delta\lambda_p u_p \circ d + E_i\left(u_{pj} - \lambda_p u_{pi}\right) + E_j\left(u_{pi} - \lambda_p u_{pj}\right)\right)$$

Targeted attack: Link prediction

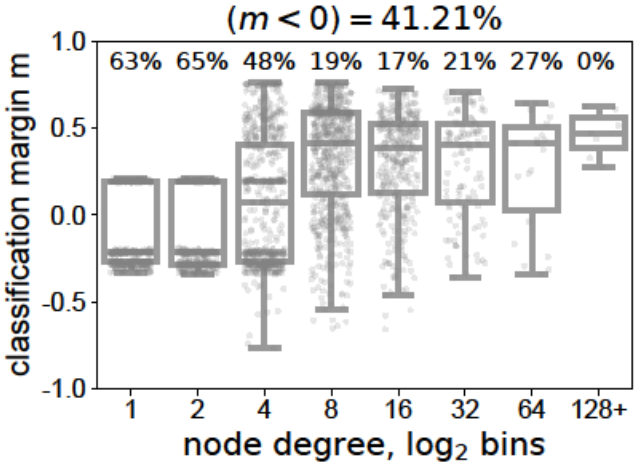




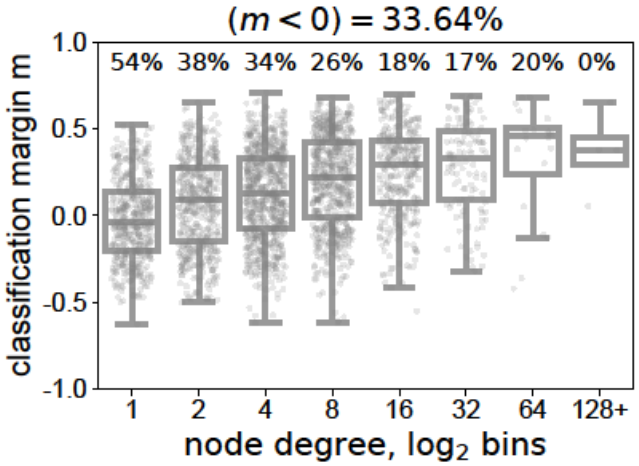
Targeted attack: Node classification



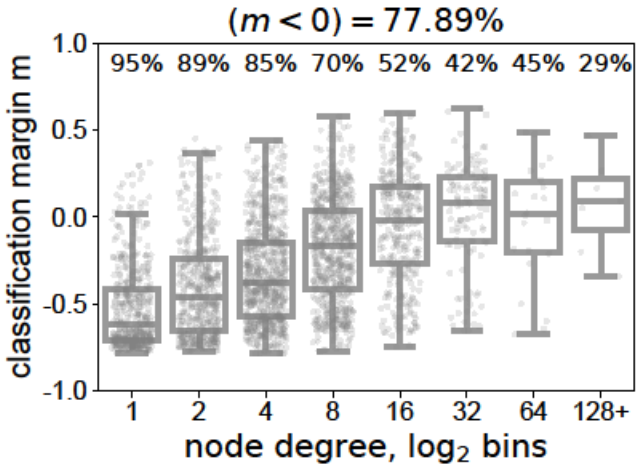
Before attack



Degree attack

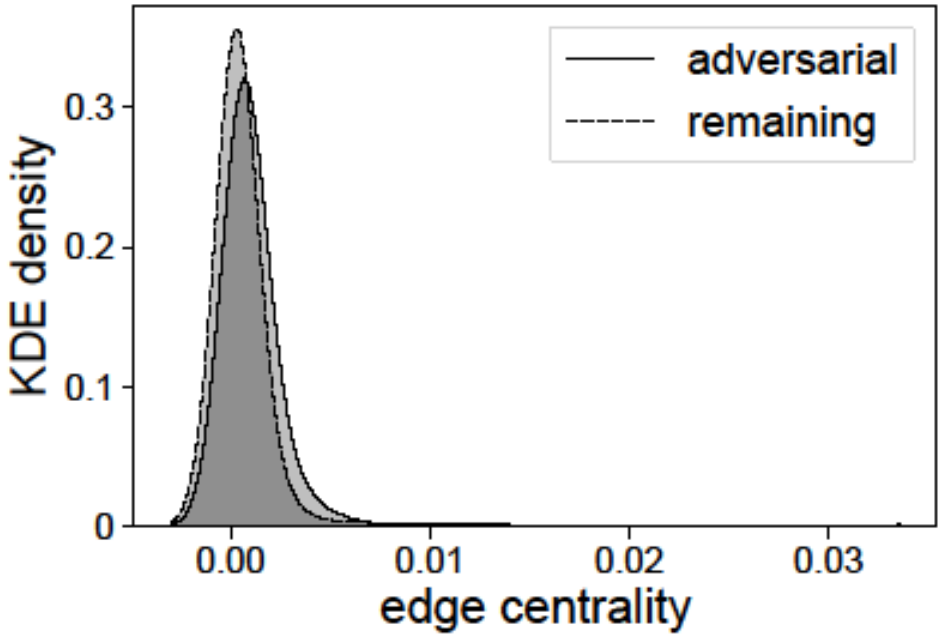


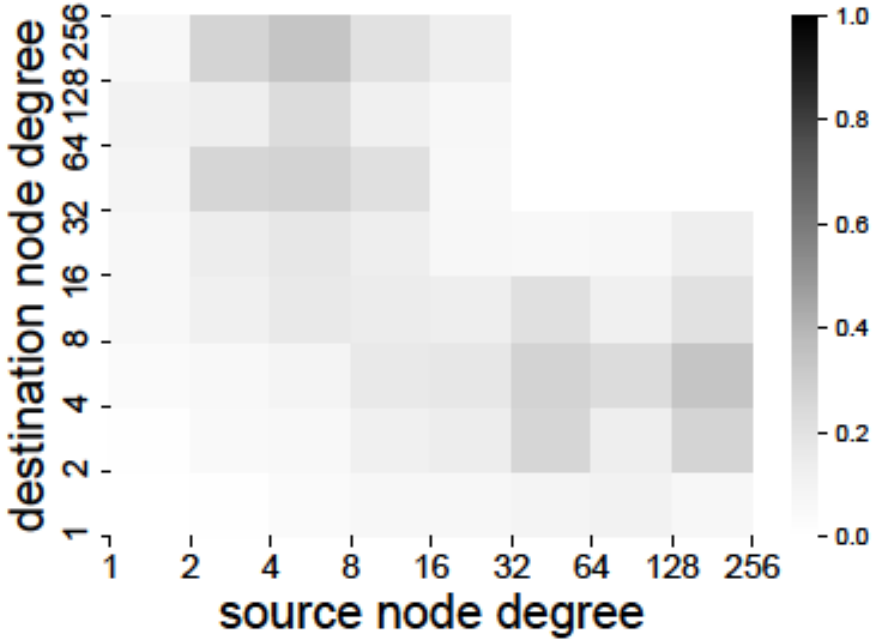
Random attack



Our attack

Analysis of adversarial edges





Transferability

	DW (SVD)	DW (SGNS)	n2v	Spect. Embd.	Label Prop.	GCN
$f = 250\ (0.8\%)$	-3.59	-2.37	-2.04	-2.11	-5.78	-3.34
$f = 500\ (1.6\%)$	-4.62	-3.97	-3.48	-4.57	-8.95	-2.33
$f = 250\ (1.7\%)$	-7.59	-5.73	-6.45	-3.58	-4.99	-2.21
$f = 500\ (3.4\%)$	-9.68	-11.47	-10.24	-4.57	-6.27	-8.61

Conclusion

Node embeddings are vulnerable to adversarial attacks

Poisoning has negative effect on the embeddings quality and the downstream tasks

Attacks are transferable – they generalize to many models

Important aspects of graph embeddings

Capturing uncertainty

Robustness to noise

Robustness to adversarial attacks