Unsupervised evaluation of multiple node ranks by reconstructing local structures

Emmanouil Krasanakis, Symeon Papadopoulos, Ioannis Kompatsiaris

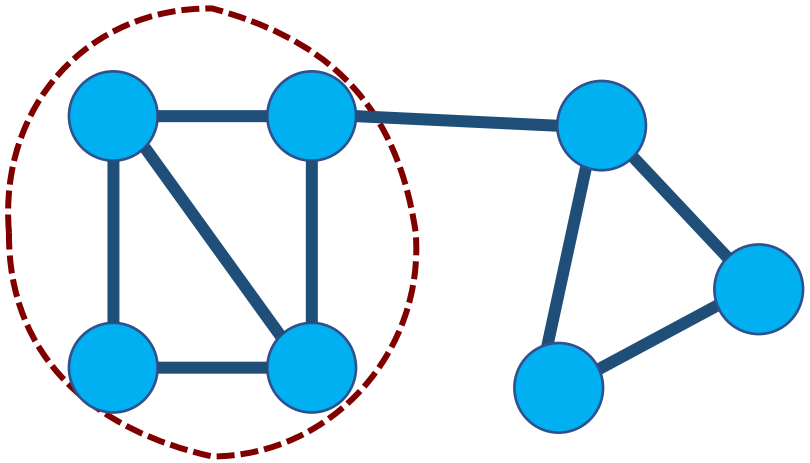
Remote meeting, 26 January 2022

CERTH
CENTRE FOR RESEARCH & TECHNOLOGY HELLAS

THINK TANK

Node ranking

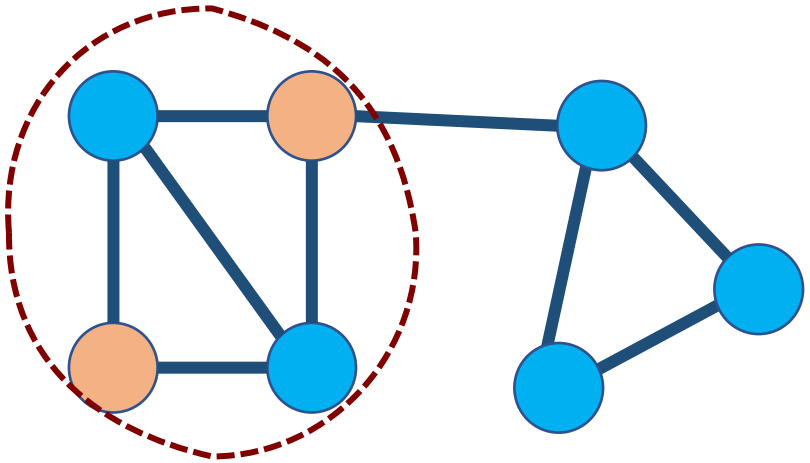
Network community $\boldsymbol{A}$
(e.g. structural, metadata group of common attribute value)



Node ranking

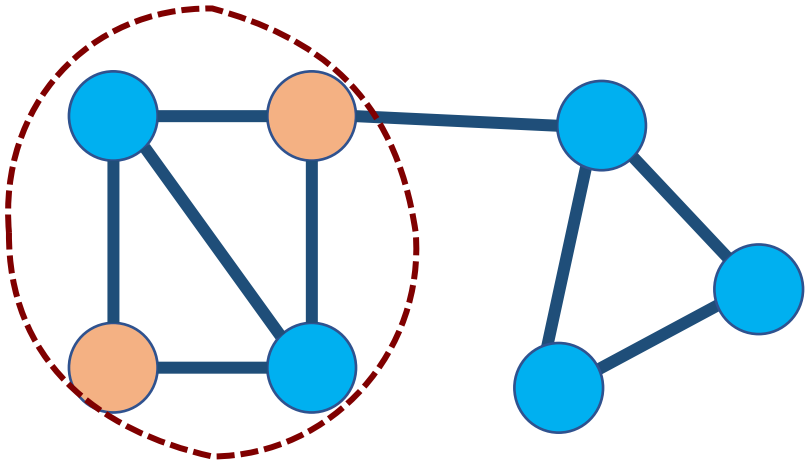
Network community $\boldsymbol{A}$
(e.g. structural, metadata group of common attribute value)

Seed set of known members S_A



Node ranking

Network community $\boldsymbol{A}$
(e.g. structural, metadata group of common attribute value)

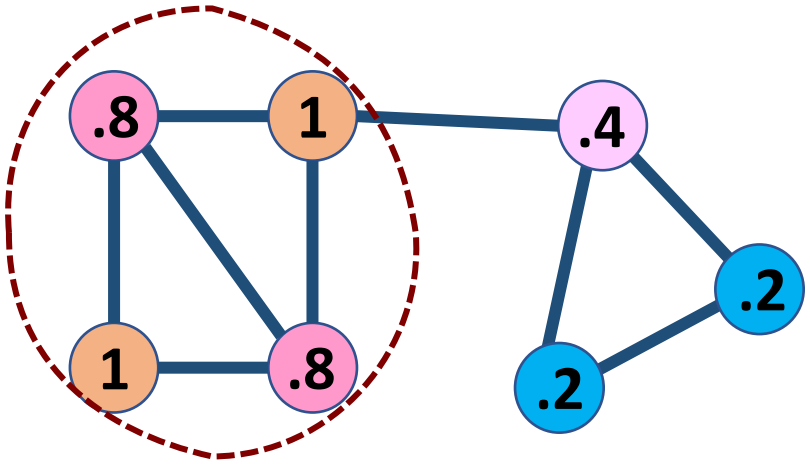


Seed set of known members S_A

Node ranking estimates a probability that other nodes $\in \boldsymbol{A}$

Node ranking

Network community $\boldsymbol{A}$
(e.g. structural, metadata group of common attribute value)



Seed set of known members S_A

Node ranking estimates a probability that other nodes $\in \boldsymbol{A}$
(e.g. for Personalized PageRank, Heat Kernels this takes some form of structural proximity to seeds)

Node ranking algorithms

Estimate probability that nodes are members of communities

- a type of recommender system
- do not necessarily need node features
  (if these are available decoupled GNNs also employ node ranking with effectively fuzzy seeds)

Node ranking algorithms

Estimate probability that nodes are members of communities

- a type of recommender system
- do not necessarily need node features
  (if these are available decoupled GNNs also employ node ranking with effectively fuzzy seeds)
- many algorithms and parameters to suit different ad-hoc assumptions of “proximity” to seeds
  (e.g. random walk with restart for PageRank, best assumptions differ from network to network)

Node ranking algorithms

Estimate probability that nodes are members of communities

- a type of recommender system
- do not necessarily need node features
  (if these are available decoupled GNNs also employ node ranking with effectively fuzzy seeds)
- many algorithms and parameters to suit different ad-hoc assumptions of “proximity” to seeds
  (e.g. random walk with restart for PageRank, best assumptions differ from network to network)

Problem: how to select which one to use?

Evaluation methodology

How to select best node ranking algorithm?
(best assumptions differ from network to network)

Evaluation methodology

How to select best node ranking algorithm?
(best assumptions differ from network to network)

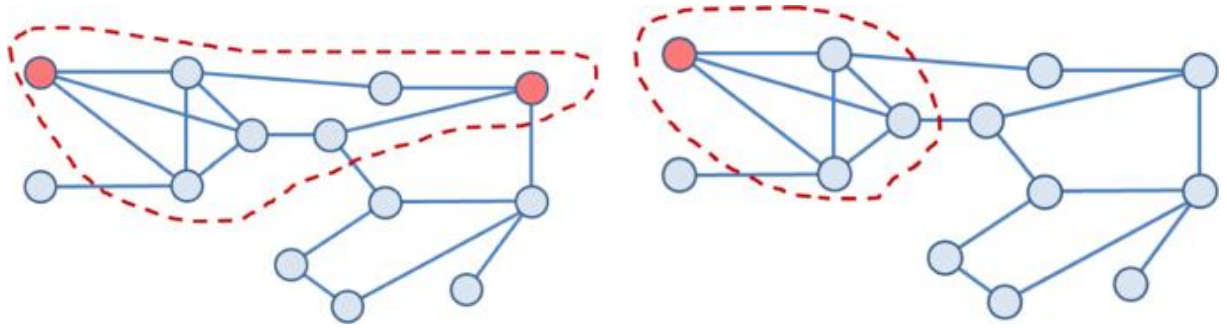
- many seeds → supervised evaluation
  withhold some examples and see if they get high ranks

Evaluation methodology

How to select best node ranking algorithm?
(best assumptions differ from network to network)

- many seeds → supervised evaluation
  withhold some examples and see if they get high ranks

- few seeds → cannot do the same
  e.g., removing a red seed drastically changes the ideal community

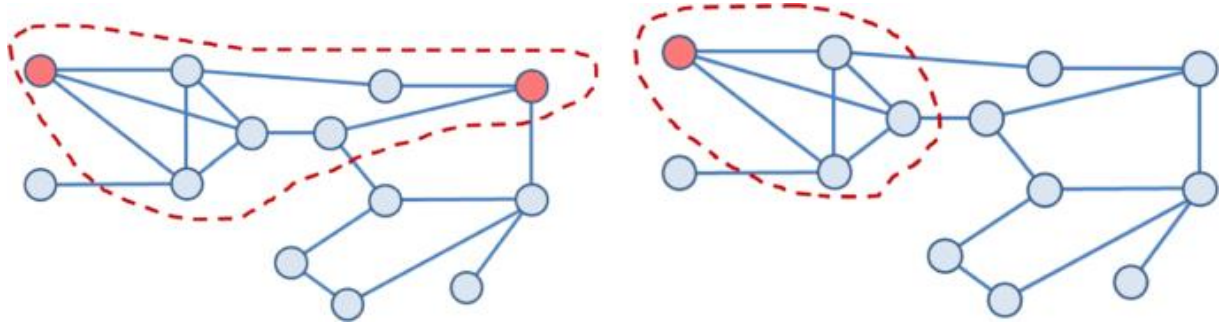


Evaluation methodology

How to select best node ranking algorithm?
(best assumptions differ from network to network)

- many seeds → supervised evaluation
  withhold some examples and see if they get high ranks

- few seeds → cannot do the same
  e.g., removing a red seed drastically changes the ideal community



Alternative: unsupervised evaluation

Unsupervised rank evaluation

Obvious take

- adapt unsupervised community quality measures to node ranking (e.g. $\text{conductance} = \frac{\text{outgoing}}{\text{internal edges}}$, $\text{density} = \frac{\text{internal edges}}{\text{nodes}(\text{nodes}-1)}$)

Unsupervised rank evaluation

Obvious take

- adapt unsupervised community quality measures to node ranking (e.g. $\text{conductance} = \frac{\text{outgoing}}{\text{internal edges}}$, $\text{density} = \frac{\text{internal edges}}{\text{nodes}(\text{nodes}-1)}$)
- **BUT** they make high-level structural assumptions of what's good (e.g. they favor small/local/dense communities)

Unsupervised rank evaluation

Obvious take

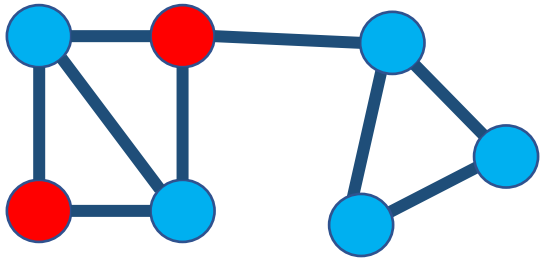
- adapt unsupervised community quality measures to node ranking
  (e.g. $\text{conductance} = \frac{\text{outgoing}}{\text{internal edges}}$, $\text{density} = \frac{\text{internal edges}}{\text{nodes}(\text{nodes}-1)}$)
- **BUT** they make high-level structural assumptions of what's good
  (e.g. they favor small/local/dense communities)

Proposal: transform node ranks to something evaluate-able
(evaluate local structure assumptions of how links are formed, *not* what communities should look like)

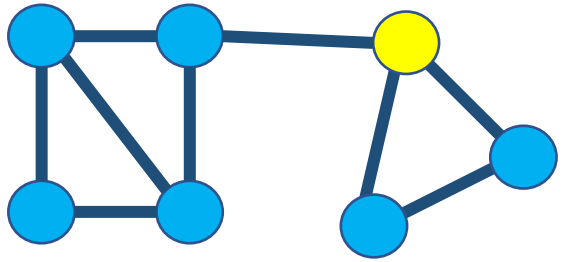
Node rank similarity - Idea

Consider community ranks as “topics” distributed across nodes

$\boldsymbol{S_A}$ for A

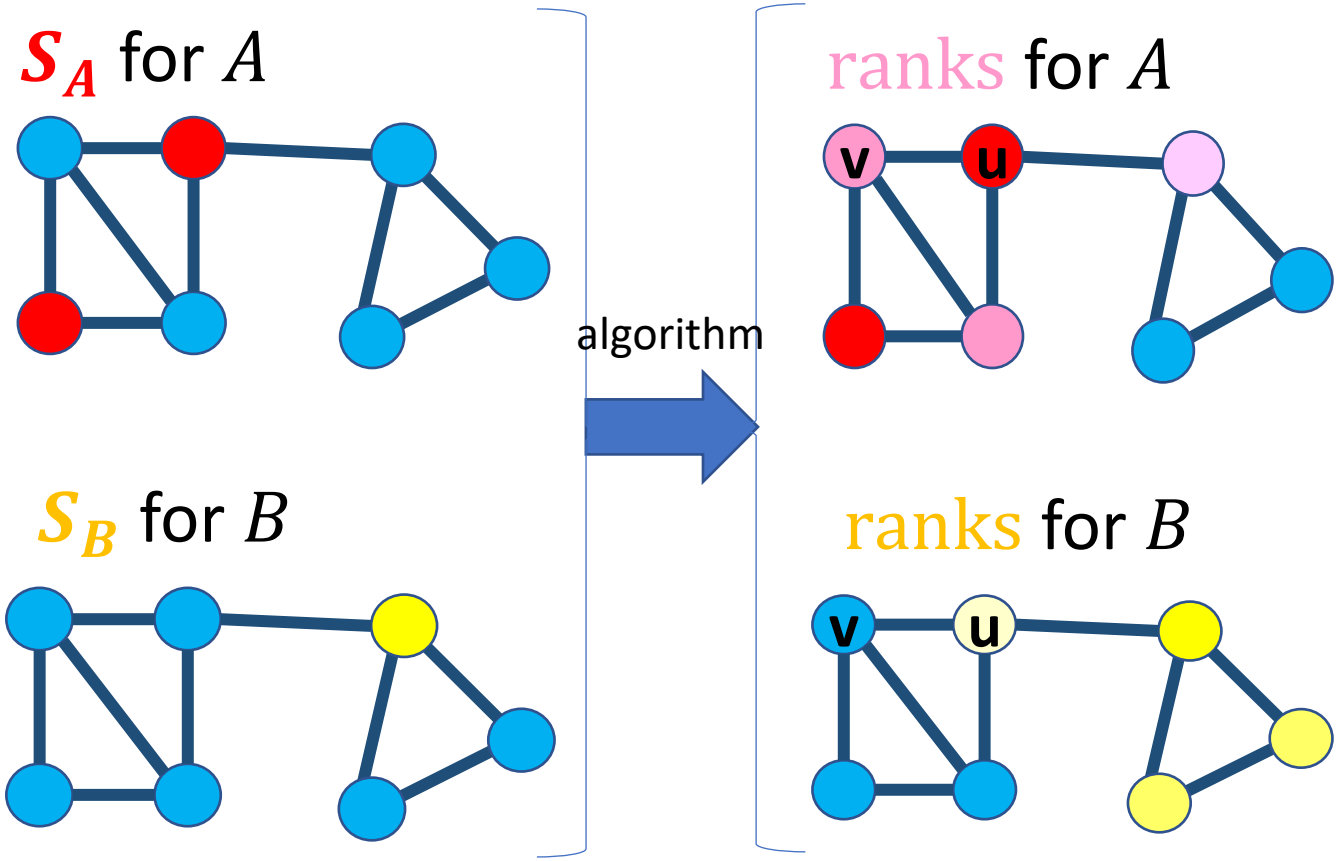


$\boldsymbol{S_B}$ for B



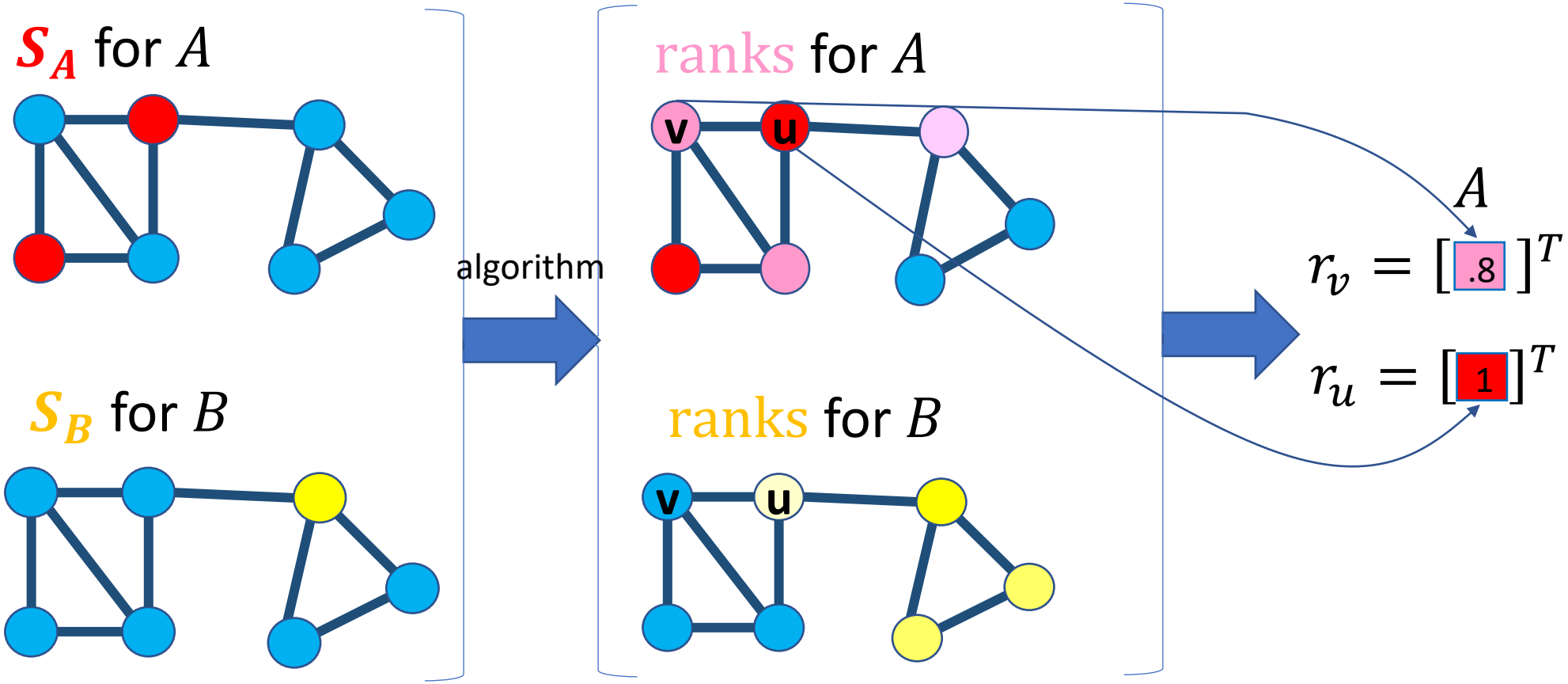
Node rank similarity - Idea

Consider community ranks as “topics” distributed across nodes



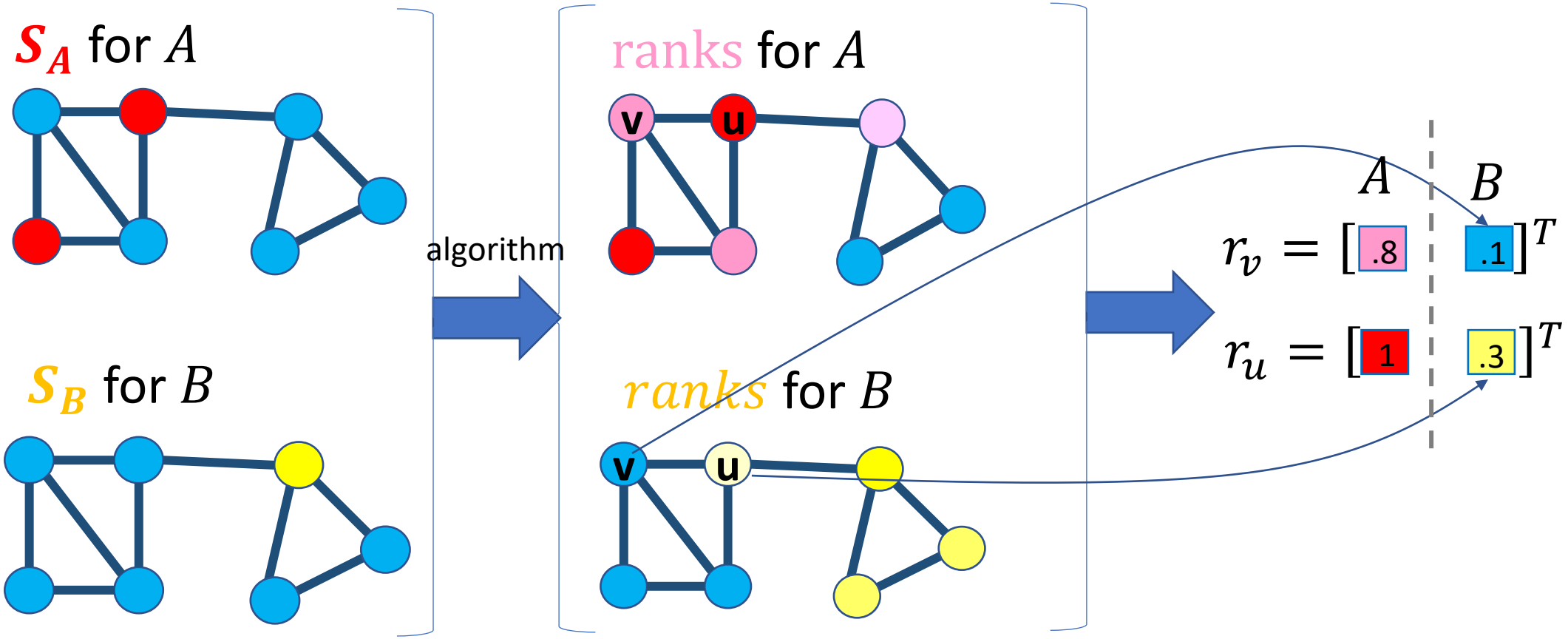
Node rank similarity - Idea

Consider community ranks as “topics” distributed across nodes



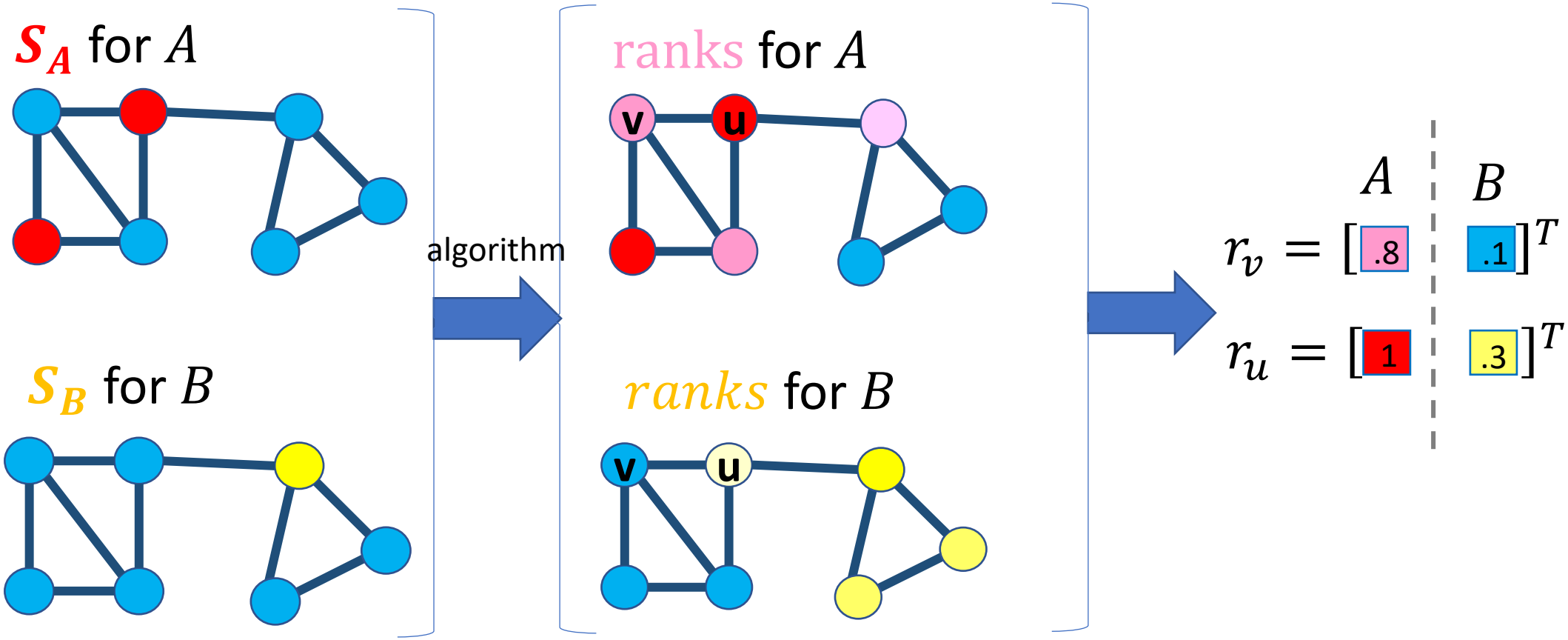
Node rank similarity - Idea

Consider community ranks as “topics” distributed across nodes



Node rank similarity - Idea

Consider community ranks as “topics” distributed across nodes



Node rank similarity - Overview

Consider community ranks as “topics” distributed across nodes

- r_v, r_u rank distributions for nodes v, u across communities
  (they are column vectors)

Node rank similarity - Overview

Consider community ranks as “topics” distributed across nodes

- r_v, r_u rank distributions for nodes v, u across communities
  (they are column vectors)
- estimate pairwise similarity $\widehat{M}[v,u] = r_v \cdot r_u$
  (dot product, perform L2 normalization first for cosine similarity)
  ⇒ in matrix form $\mathrm{R} = [r_v\ r_u\ \ldots]$ and $\widehat{M} = RR^T$

Node rank similarity - Overview

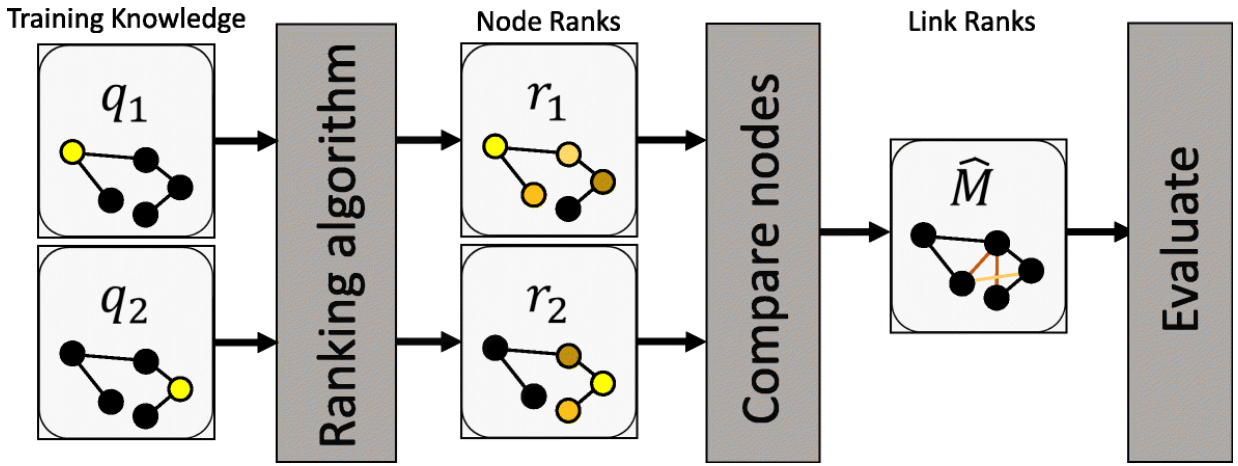
Consider community ranks as “topics” distributed across nodes

- r_v, r_u rank distributions for nodes v, u across communities
  (they are column vectors)
- estimate pairwise similarity $\widehat{M}[v,u] = r_v \cdot r_u$
  (dot product, perform L2 normalization first for cosine similarity)
  ⇒ in matrix form $\mathrm{R} = [r_v\ r_u\ \ldots]$ and $\widehat{M} = RR^T$
- for best algorithms, $\widehat{M}$ approximates well the real adjacency matrix M

Node rank similarity - Overview

Consider community ranks as “topics” distributed across nodes

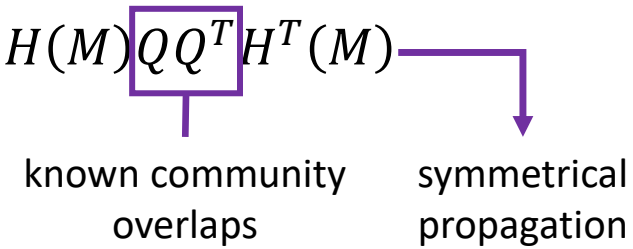
- r_v, r_u rank distributions for nodes v, u across communities
  (they are column vectors)
- estimate pairwise similarity $\widehat{M}[v,u] = r_v \cdot r_u$
  (dot product, perform L2 normalization first for cosine similarity)
  ⇒ in matrix form $\mathrm{R} = [r_v\ r_u\ \ldots]$ and $\widehat{M} = RR^T$
- for best algorithms, $\widehat{M}$ approximates well the real adjacency matrix M



Node rank similarity - Explanation

For node ranking based on filters $H(\cdot)$

- $R = H(M)Q$ where $Q[v, A] = \{1 \text{ if } v \in S_A, 0 \text{ otherwise}\}$ and M is the adjacency matrix
- $\widehat{M} = RR^T = H(M)QQ^T H^T(M)$



- $\widehat{M}$ can predict M only if both known community overlaps and filter propagation are accurate

LinkAUC

$$\text{LinkAUC} = \text{AUC}\left(M, \widehat{M}\right)$$

- local structure: homophily (similar neighbors)
- for no community overlap $\text{LinkAUC} \in O(\sum_A w_A d_A)$, where rank density $d_A = \frac{E_A[\text{internal edges}]}{E_A[nodes(nodes-1)]}$
- community inter-relations enrich this

Emmanouil Krasanakis, Symeon Papadopoulos, and Yiannis Kompatsiaris. "LinkAUC: Unsupervised evaluation of multiple network node ranks using link prediction." *International Conference on Complex Networks and Their Applications*. Springer, Cham, 2019.

HopAUC

$$\text{HopAUC} = \text{AUC}\left(M + M^2 \geq 1, \widehat{M}\right)$$

- local structure: either homophily *or* structural equivalence (similar neighbors-of-neighbors)
- adds second-order proximity to the assessment
- basically LinkAUC but nodes in the extended neighborhood up to 2 hops away at matched

Emmanouil Krasanakis, Symeon Papadopoulos, and Yiannis Kompatsiaris. "Unsupervised evaluation of multiple node ranks by reconstructing local structures." *Applied Network Science* 5.1 (2020): 1-32.

LinkCC

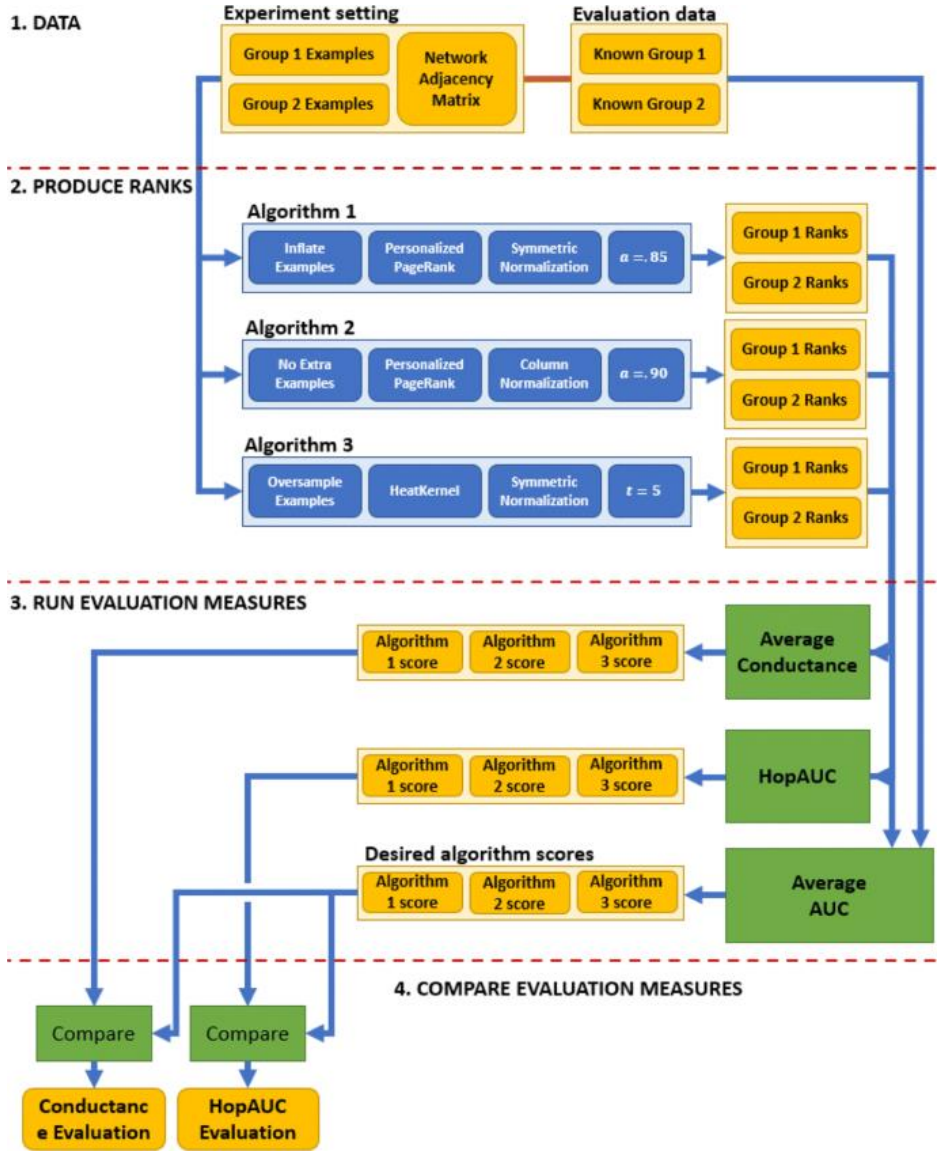
$$\text{LinkCC} = \frac{\text{tr}(M\widehat{M}M)}{\text{tr}(M^3)}$$

- local structure: both homophily and structural equivalence (rare to satisfy both)
- assesses triadic closure with an approximate of estimated over the true clustering coefficient
- not well-performing, but a different example of assessing edges instead of communities

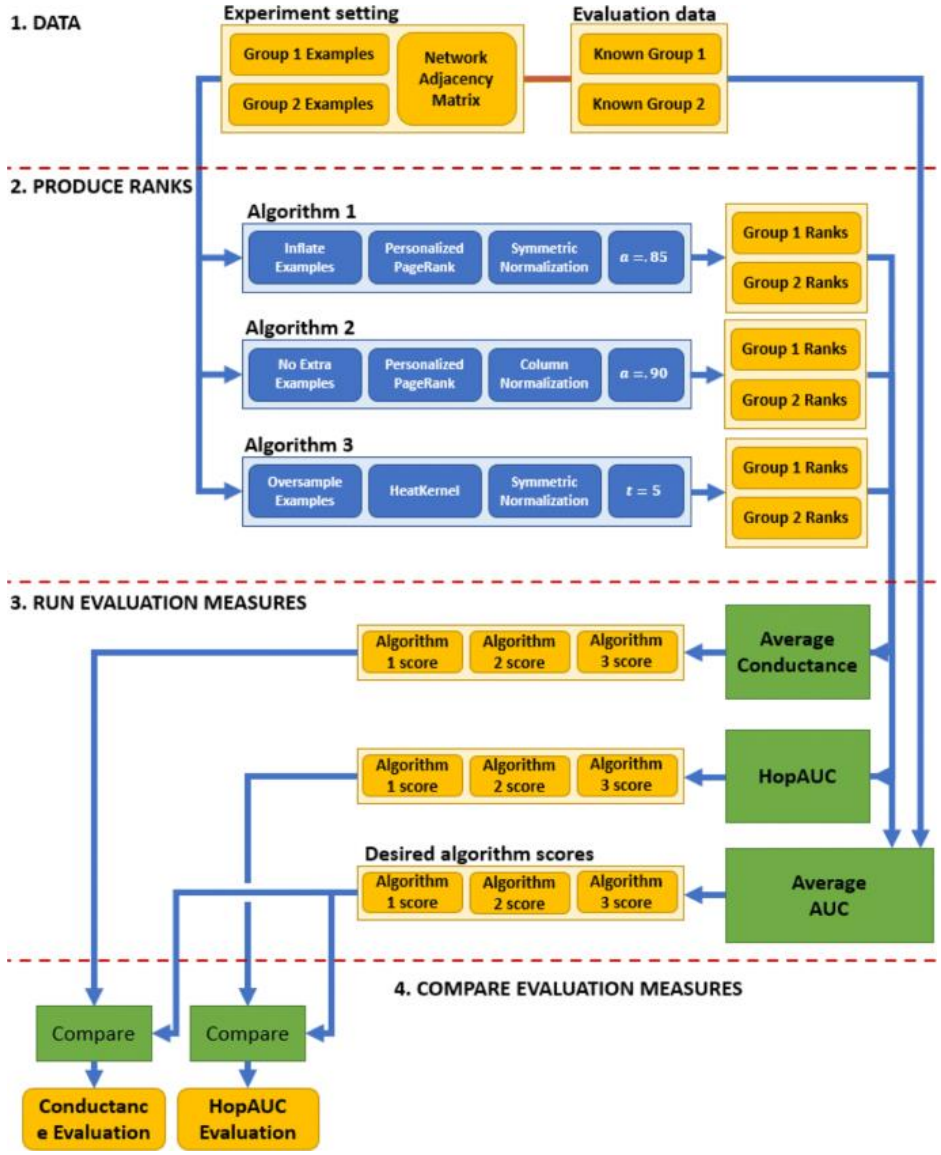
Emmanouil Krasanakis, Symeon Papadopoulos, and Yiannis Kompatsiaris. "Unsupervised evaluation of multiple node ranks by reconstructing local structures." *Applied Network Science* 5.1 (2020): 1-32.

Evaluation of evaluation measures

Evaluation of evaluation measures

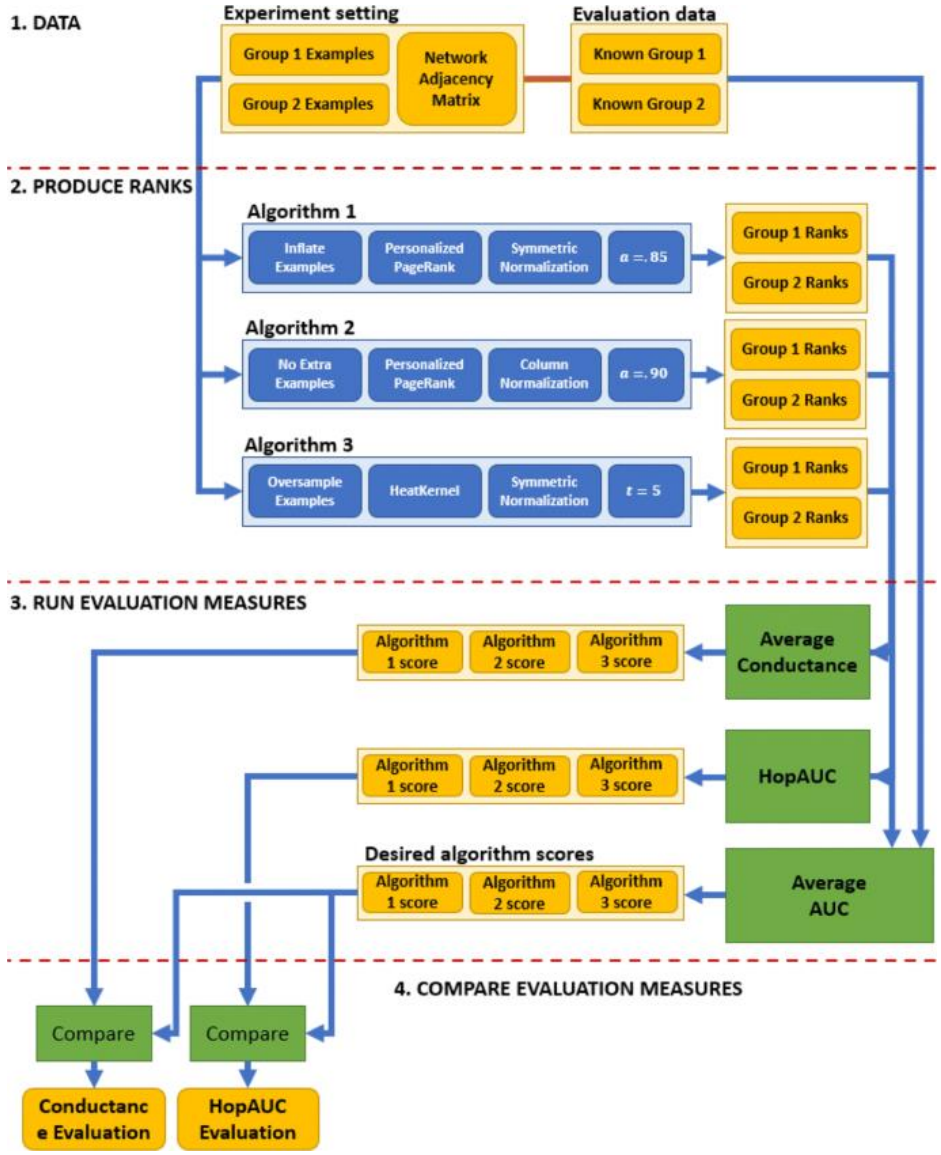


Evaluation of evaluation measures



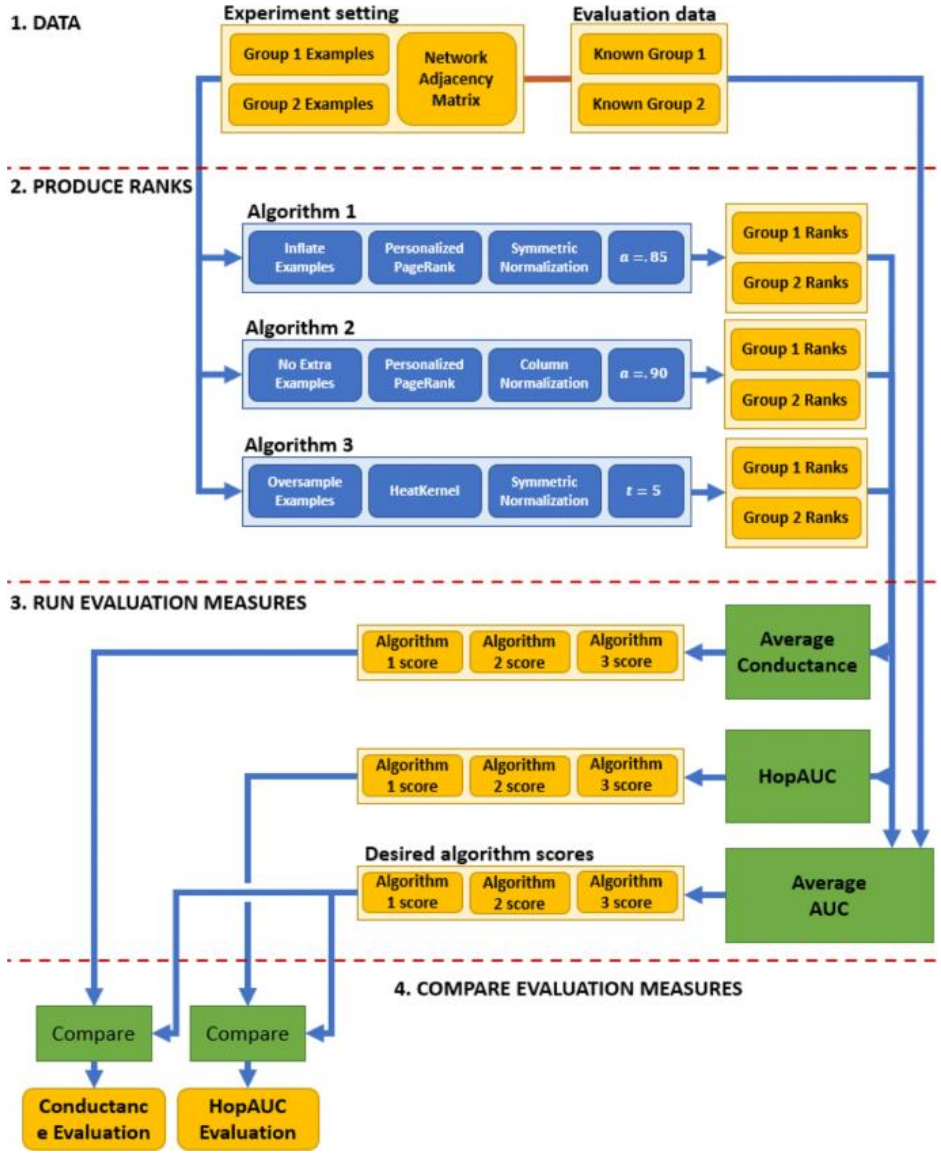
- 25 experiment settings (different fractions of seeds in 5 networks)

Evaluation of evaluation measures



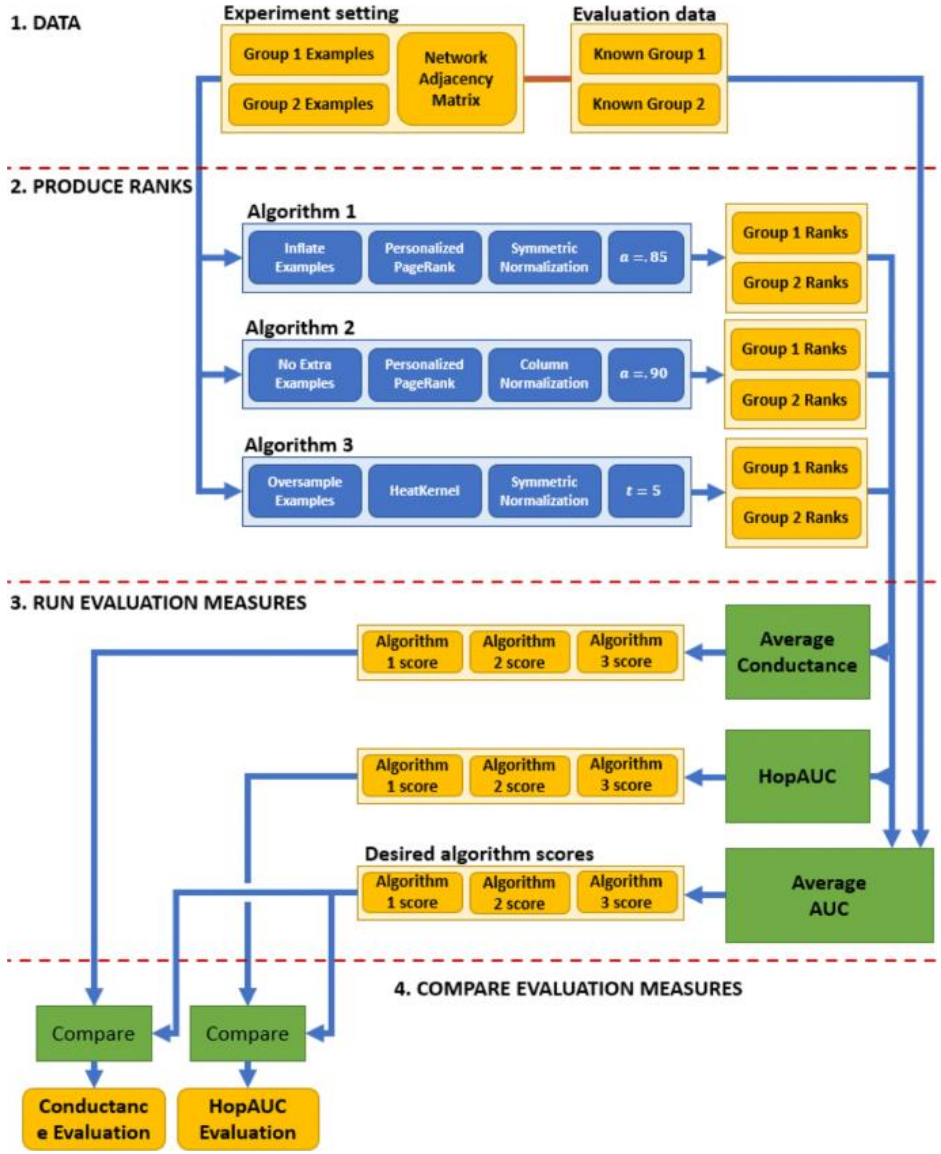
- 25 experiment settings (different fractions of seeds in 5 networks)
- 72 node ranking algorithms (variations of 3 base ones)

Evaluation of evaluation measures



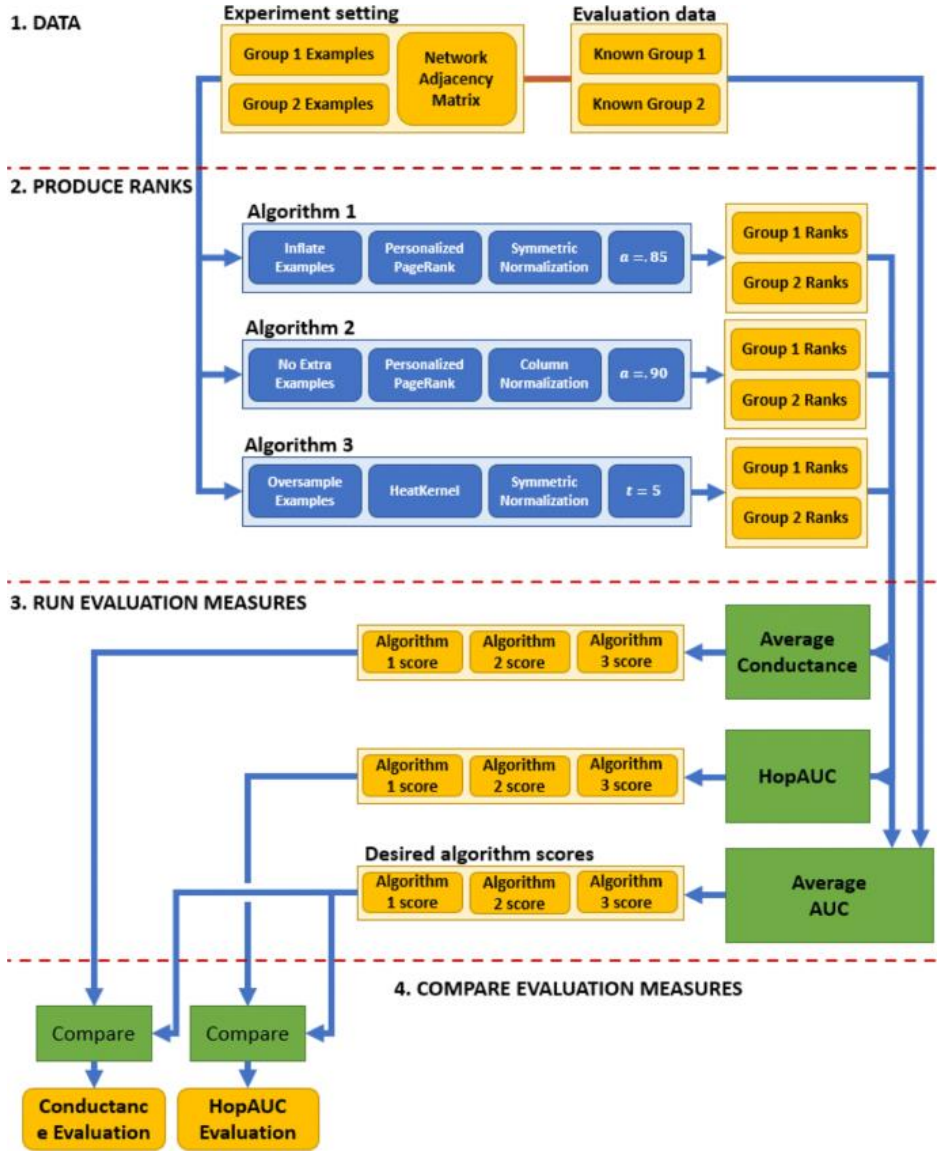
- 25 experiment settings (different fractions of seeds in 5 networks)
- 72 node ranking algorithms (variations of 3 base ones)
- 6 unsupervised measures (including ours)

Evaluation of evaluation measures



- 25 experiment settings (different fractions of seeds in 5 networks)
- 72 node ranking algorithms (variations of 3 base ones)
- 6 unsupervised measures (including ours)
- Compare unsupervised measures to 2 supervised ones: AUC and NDCG averaged across communities

Evaluation of evaluation measures



- 25 experiment settings (different fractions of seeds in 5 networks)

×

- 72 node ranking algorithms (variations of 3 base ones)

×

- 6 unsupervised measures (including ours)

×

- Compare unsupervised measures to 2 supervised ones: AUC and NDCG averaged across communities

$= 21{,}600$ algorithm evaluation results!

Reminder: we want to evaluate the unsupervised measures

Summary of results 1/3

Best Spearman correlation with supervised measures?

- for many seeds ($\gg 1\%$ of communities) unsupervised measures do not correlate with supervised ones
  (that's fine, because we can do supervised evaluation then)
- correlations are not too strong
  (that's fine too, given that even the supervised measures sometimes also fail to positively correlate between themselves)
- highest-correlated measure tends to be among local structure ones
- HopAUC best overall, but fails in the one of the two datasets where LinkCC performs well

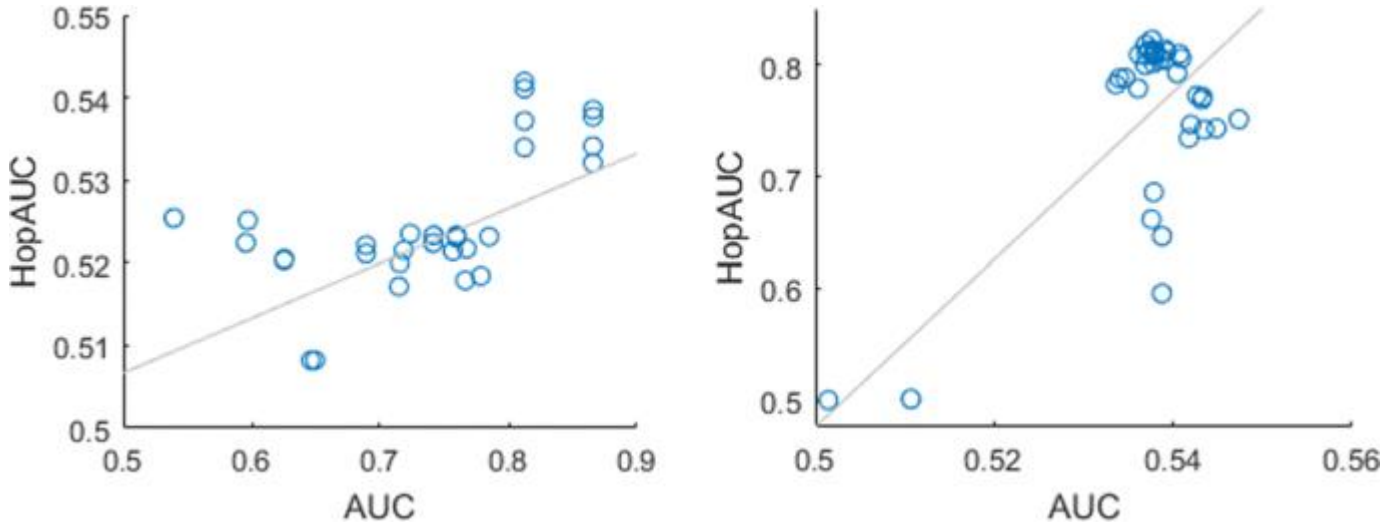
Verdict: better *not* use any unsupervised measure for pairwise comparison of node ranking algorithms, especially if there are enough seeds for supervised evaluation. If no other option, use HopAUC on the blind.

Corollary: *avoid* unsupervised optimization of node ranking algorithm parameters.

Summary of results 2/3

What is the problem?

- unsupervised measures can be confused by small algorithm differences
- e.g., evaluation of HopAUC vs AUC on two random settings:



Observation: algorithms identified as good seem to be actually good (top-right)

Summary of results 3/3

Can we still find near-best algorithms?

Measure: number of experiment settings supervised measure gap between selected and best algorithms is gap
(e.g., number of settings it holds $\mathrm{AUC}\left[\mathrm{argmax}_{algorithm}(\mathrm{LinkAUC[algorithm]})\right] \geq \max \mathrm{AUC} - gap$)

- $gap \in \{5\%, 7\%\}$ for AUC
- $gap \in \{9\%, 13\%\}$ for NDCG

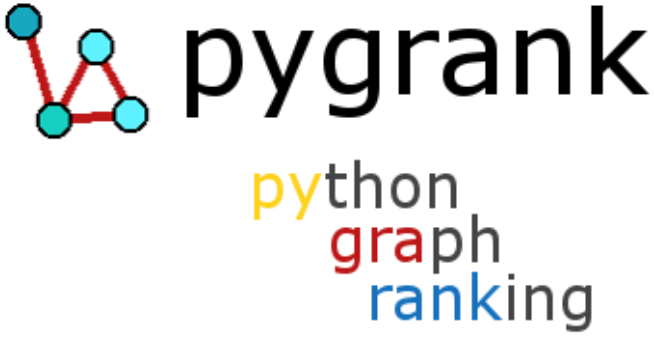
- HopAUC is the best (20/25, 23/25 for respective gaps, these numbers happen to be the same for AUC and NDCG)
- followed by Modularity (18/25, 22/25 – close, but importantly this sometimes fails for few seeds)
- LinkAUC (17/25, 19/25)
- other measures are much worse compared to HopAUC (30% or more relative reduction)

Verdict: to find good node ranking algorithms with unsupervised measures, experiment with *many* different ones and select the best. Do *not* make pairwise comparisons.

Takeaways

To use unsupervised evaluation measures, check for network dynamics:

- if you suspect strong homophily use LinkAUC
- if you additionally suspect structural equivalence to play a role use HopAUC (best on the blind)
- if you know communities are structurally separable use modularity (our adaptation to node ranking)



Python package implementing node ranking algorithms and measures

- https://github.com/MKLab-ITI/pygrank
- implements 7 of ours and more than 10 other related research
- modular algorithm construction, backends (*numpy, tensorflow, pytorch*), automatic tuning, easy experiments
- ongoing development (we welcome new ranking algorithms and evaluation measures)

pip install --upgrade pygrank

Hands-on 1/3

```
import pygrank as pg
loader = pg.load_datasets_multiple_communities(["EUCore"], max_group_number=3) # 3 com/ties
_, graph, communities = next(loader)  # loads the next dataset
```

Hands-on 1/3

```
import pygrank as pg
loader = pg.load_datasets_multiple_communities(["EUCore"], max_group_number=3) # 3 com/ties
_, graph, communities = next(loader)  # loads the next dataset

train, test = pg.split(communities, 0.05)  # 5% of community members are known
algorithms = pg.create_variations(pg.create_demo_filters(), pg.Normalize).values()
```

Hands-on 1/3

```
import pygrank as pg
loader = pg.load_datasets_multiple_communities(["EUCore"], max_group_number=3) # 3 com/ties
_, graph, communities = next(loader)  # loads the next dataset

train, test = pg.split(communities, 0.05)  # 5% of community members are known
algorithms = pg.create_variations(pg.create_demo_filters(), pg.Normalize).values()

sup = pg.AlgorithmSelection(algorithms, measure=pg.AUC)
mod = pg.AlgorithmSelection(algorithms, fraction_of_training=1,
                            measure=pg.Modularity().as_supervised_method())
```

Hands-on 1/3

```
import pygrank as pg
loader = pg.load_datasets_multiple_communities(["EUCore"], max_group_number=3) # 3 com/ties
_, graph, communities = next(loader)  # loads the next dataset

train, test = pg.split(communities, 0.05)  # 5% of community members are known
algorithms = pg.create_variations(pg.create_demo_filters(), pg.Normalize).values()

sup = pg.AlgorithmSelection(algorithms, measure=pg.AUC)
mod = pg.AlgorithmSelection(algorithms, fraction_of_training=1,
                            measure=pg.Modularity().as_supervised_method())

link = None
best_evaluation = 0
linkAUC = pg.LinkAssessment(graph, similarity="cos", hops=1)  # emails are homophilous
for algorithm in algorithms:  # no automatic algorithm selection (yet) for multi-com/ties
    evaluation = linkAUC.evaluate({community: algorithm(graph, seeds)
                                   for community, seeds in train.items()})
    if evaluation > best_evaluation:
        best_evaluation = evaluation
        link = algorithm
```

Hands-on 2/3

```
sup_evals = list()
mod_evals = list()
link_evals = list()
for seeds, members in zip(train.values(), test.values()):
    measure = pg.AUC(members, exclude=seeds)  # test ignores seeds
    sup_evals.append(measure(sup(graph, seeds)))
    mod_evals.append(measure(mod(graph, seeds)))
    link_evals.append(measure(link(graph, seeds)))
```

Hands-on 2/3

```
sup_evals = list()
mod_evals = list()
link_evals = list()
for seeds, members in zip(train.values(), test.values()):
    measure = pg.AUC(members, exclude=seeds)  # test ignores seeds
    sup_evals.append(measure(sup(graph, seeds)))
    mod_evals.append(measure(mod(graph, seeds)))
    link_evals.append(measure(link(graph, seeds)))

print(sup.cite())
print("Supervised", sum(sup_evals) / len(sup_evals))
print("Modularity", sum(mod_evals) / len(mod_evals))
print("LinkAUC", sum(link_evals) / len(link_evals))
```

Hands-on 3/3

REQUIRED CITATION: Please visit the url https://snap.stanford.edu/data/email-Eu-core.html for instructions on how to cite the dataset eucore in your research

selected the best among the following algorithms that optimizes AUC while withholding 0.200 of nodes for validation: \\

- personalized PageRank \cite{page1999pagerank} with restart probability 0.15 and max normalization \\
- personalized PageRank \cite{page1999pagerank} with restart probability 0.099 and max normalization \\
- personalized PageRank \cite{page1999pagerank} with restart probability 0.05 and max normalization \\
- HeatKernel \cite{chung2007heat} with emphasis on 3-hop distances and max normalization \\
- HeatKernel \cite{chung2007heat} with emphasis on 5-hop distances and max normalization \\
- HeatKernel \cite{chung2007heat} with emphasis on 7-hop distances and max normalization \\

Supervised 0.825

Modularity 0.859

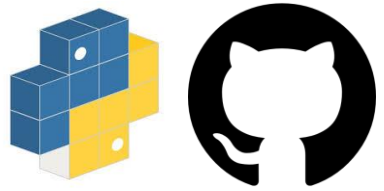
LinkAUC 0.859

Unsupervised algorithm selection is better for few seeds!

Thank you for your attention!

pygrank
python graph ranking

pip install --upgrade pygrank
github.com/MKLab-ITI/pygrank



Emmanouil Krasanakis
Symeon Papadopoulos
Yiannis Kompatsiaris

maniospas@iti.gr
papadop@iti.gr
ikom@iti.gr

FuturePulse

HELIOS