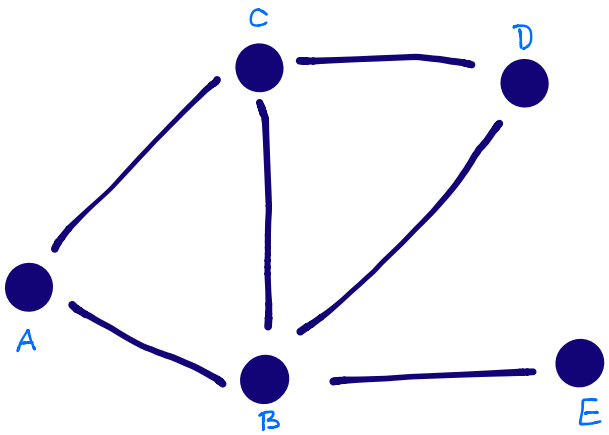
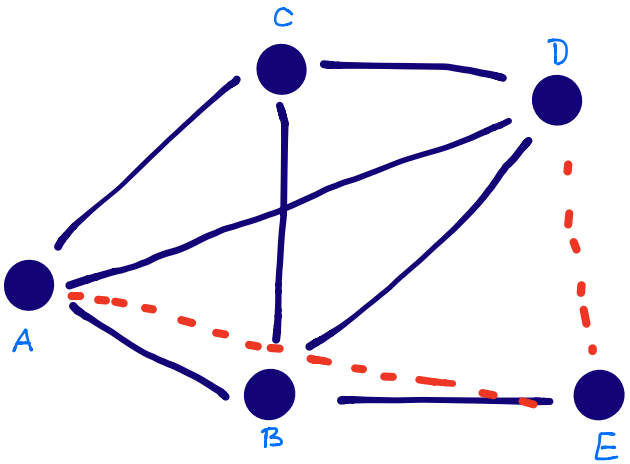
Link Prediction Algorithms

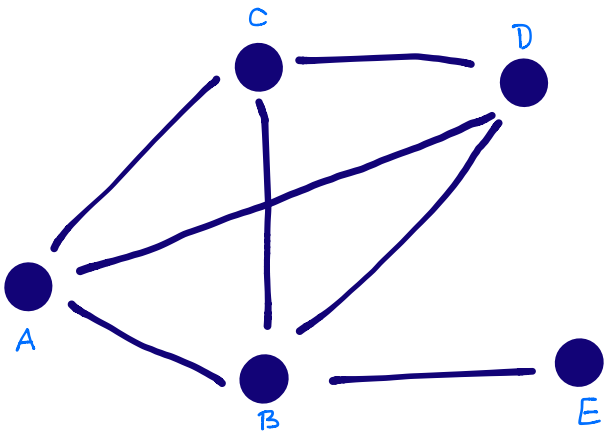
What will facebook look like tomorrow?

Ahmad Sadraei

May 2014







t_1

t_2

t_k

o

Network-evolution models

- Social Network Analysis
  - Link prediction
    - Supervised learning
      - binary classifier
    - Unsupervised (non-learning)
      - Node based topological Similarity
        - common neighbors
        - Jaccard Coef.
        - Adamic/Adar
        - Prefrential Attachment
      - Path based topological Similarity
        - katz
        - Hitting time
        - Rooted PageRank

Social Networks: structures whose nodes represent people or other entities embedded in a social context, and whose edges represent interaction, collaboration, or influence between entities

Applications:

friend recommendation: "you may know these people"

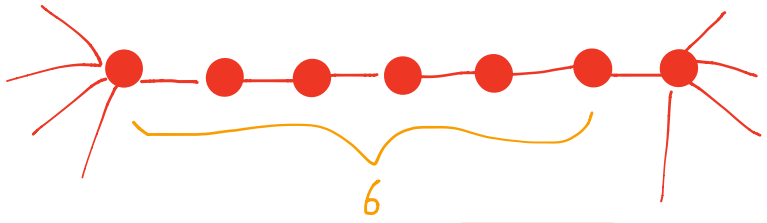
Product recommendation: "people who bought this item also bought"

Collaborator recommendation: Scientific citation networks

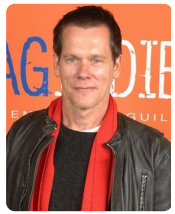
Molecular interactions: Protein-Protein interaction ...

Surveillance: NSA?

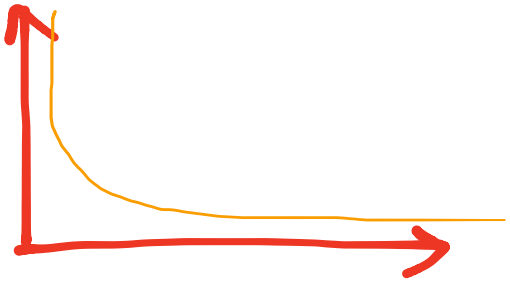
Small World Effect



- there are short paths between most pairs of nodes
- on average around 6 hops



Scale Free Effect



- node degree distribution follows a power law
- hubs have high degree, most of the rest have low

*Also the clustring effect

$G = \langle V, E \rangle$ ← Social Network

$e = \langle u, v \rangle \in E$ ← Interaction between u and v

$G[t_0, t_1]$ ← Given Subgraph as training set

$G[t_2, t_3]$ ← Infer new Edges/used for testing

$Score(u, v)$ ← Liklihood that u and v share an edge (Proximity or Similarity)

why it's a hard Problem

For our social network G(V, E), there are $V \times V - E$ possible edges to choose from, if we were picking a random edge to predict for our existing social network.

If G is dense, then $E \approx V^2$ - b where b is some constant between 1 and V. Thus, we have a constant number of edges to choose from, and O(1/c) probability of choosing correctly at random.

If G is sparse, then $E \approx V$. Thus, we have a V^2 edges to choose from, and $O(1/V^2)$ probability of choosing correctly at random. Unfortunately social networks are **sparse**, so picking at random is a terrible idea!

why it's a hard Problem

In the DBLP dataset, in the year 2000, the ratio of actual and possible link is as low as 2×10^{-5}. So, in a uniformly sampled dataset with one million training instances, we can expect only 20 positive instances.

Even worse, the ratio between the number of positive links and the number of possible links also slowly decreases over time, since the negative links grow quadratically whereas positive links grow only linearly with a new node.

ed

nonexistant edges

actual edges

Simple Algorithm

Algorithm 1 *Basic Experiment for Testing Heuristics*

Input: Observed network $G(V', E')$

1: $G''(V', E'') = G(V', E')$ - random edges
2: Score some or all of $V'^2 - E''$ edges using a heuristic method
3: E_{new} = pick k top ranked edges
4: Evaluate prediction method: effectiveness= $|E_{new} \cap (E' - E'')|$

Output: Effectiveness of the heuristic used

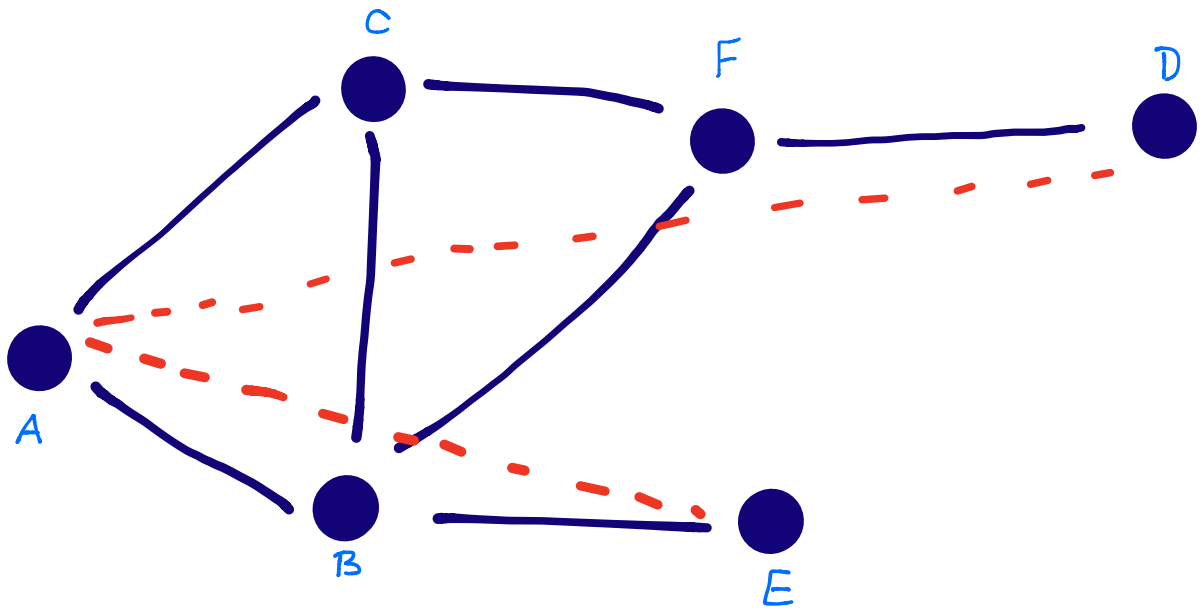
Graph Distance

negated

$$Score(x, y) = \sqrt{\text{Lenght of Shortest Path Between } x \text{ and } y}$$

↙

Small World



Score(A, E) = -2 ✓

Score(A, D) = -3

desc order

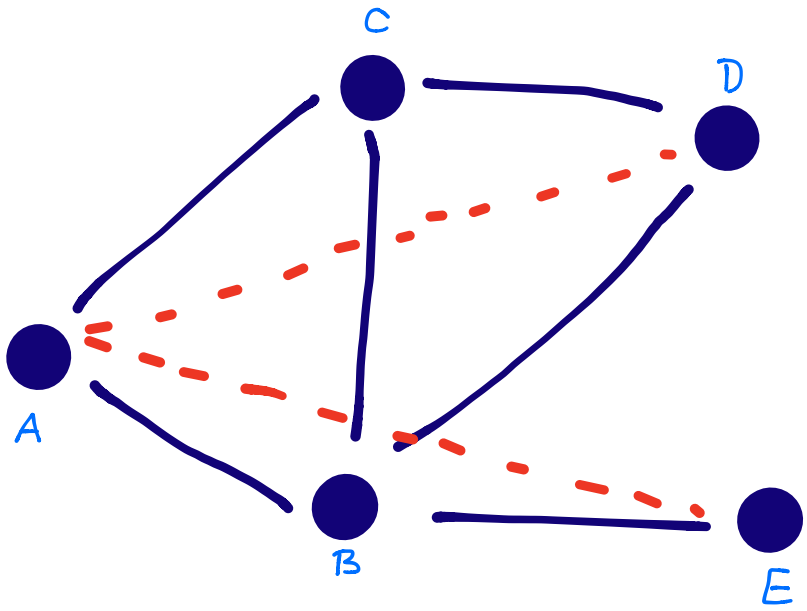
Common Neighbors *

$$Score(x, y) = |\Gamma(x) \cap \Gamma(y)|$$

Neighbors of x

* Triadic Closure

List Comparison : $O(V \cdot n \log n)$



$|\Gamma(A) \cap \Gamma(D)|$

B, C B, C

$S = 2$ ✓

$|\Gamma(A) \cap \Gamma(E)|$

B, C B

$S = 1$

Jaccard's Coefficient

Common friends

$$Score(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$$

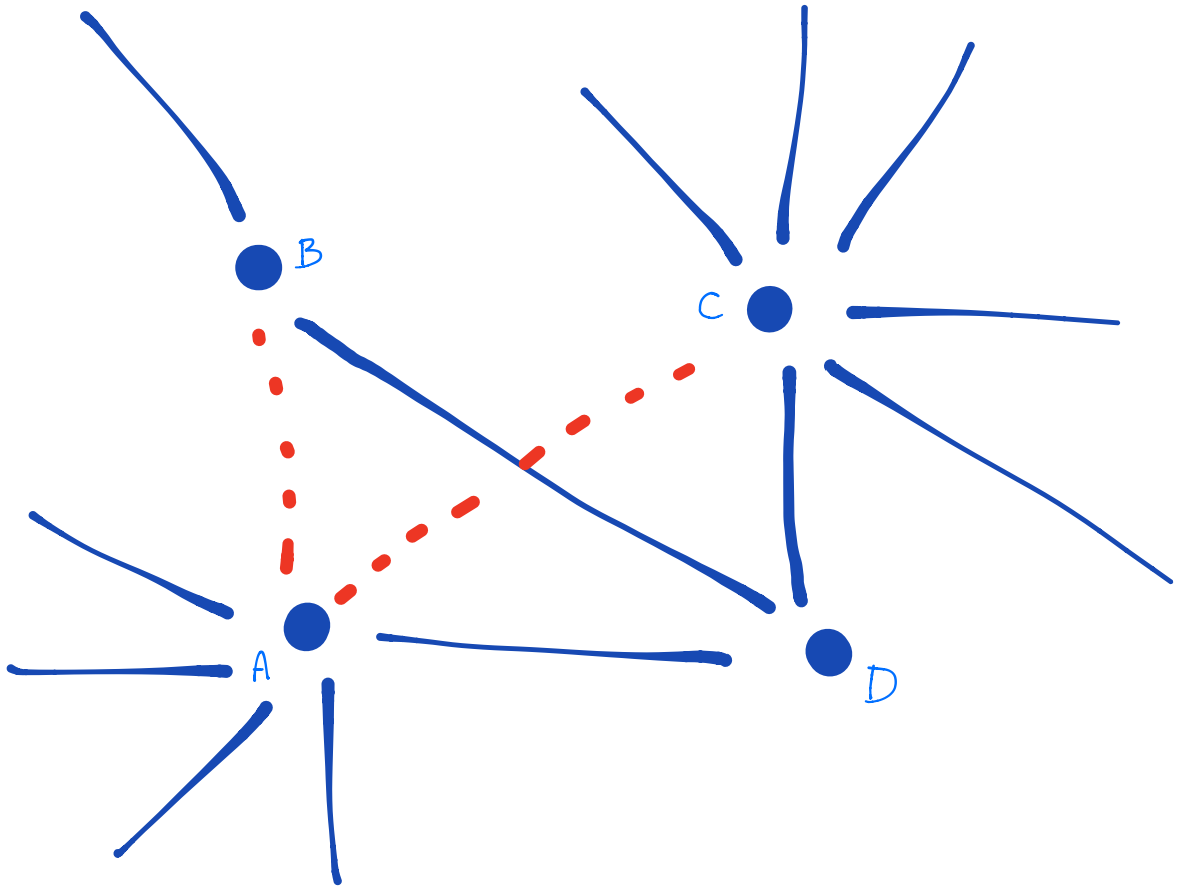
total friends

Solves the problem where:

two nodes could have many common neighbors because they have lots of neighbors, not because they are strongly related

Prefrential Attachment*

$$Score(x,y) = |\Gamma(x)| \cdot |\Gamma(y)|$$



$AB < AC$

* Scale free network effect AKA "the rich are getting richer"

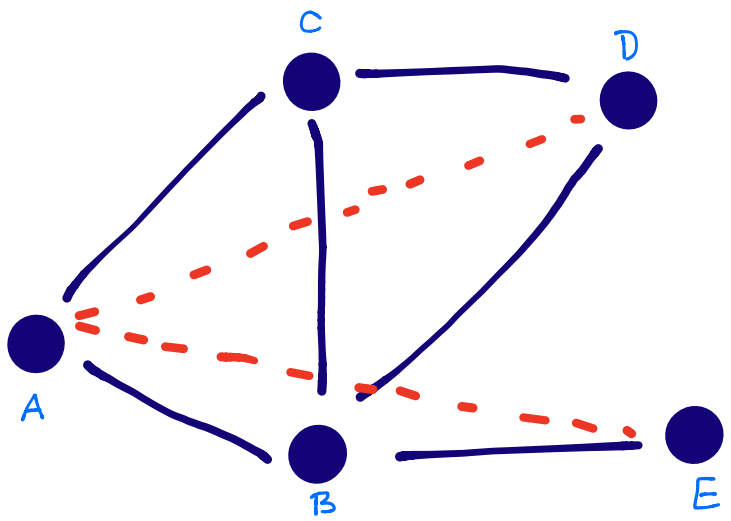
Katz

$$Score(x,y) = \sum_{l=1}^{\infty} \beta^l \cdot |Path^l_{x,y}|$$

exponentially damped by lenght

Set of all lenght l Paths from x to y

BFS: (n^3)



# of Hops

$Path^{2}_{A,D} = 2$ $Path^{3}_{A,D} = 2$

$S = \frac{1}{2} \cdot 2 + \frac{1}{4} \cdot 2 + \cdots$

Damping Factor

$Path^{2}_{A,E} = 1$ $Path^{3}_{A,E} = 1$

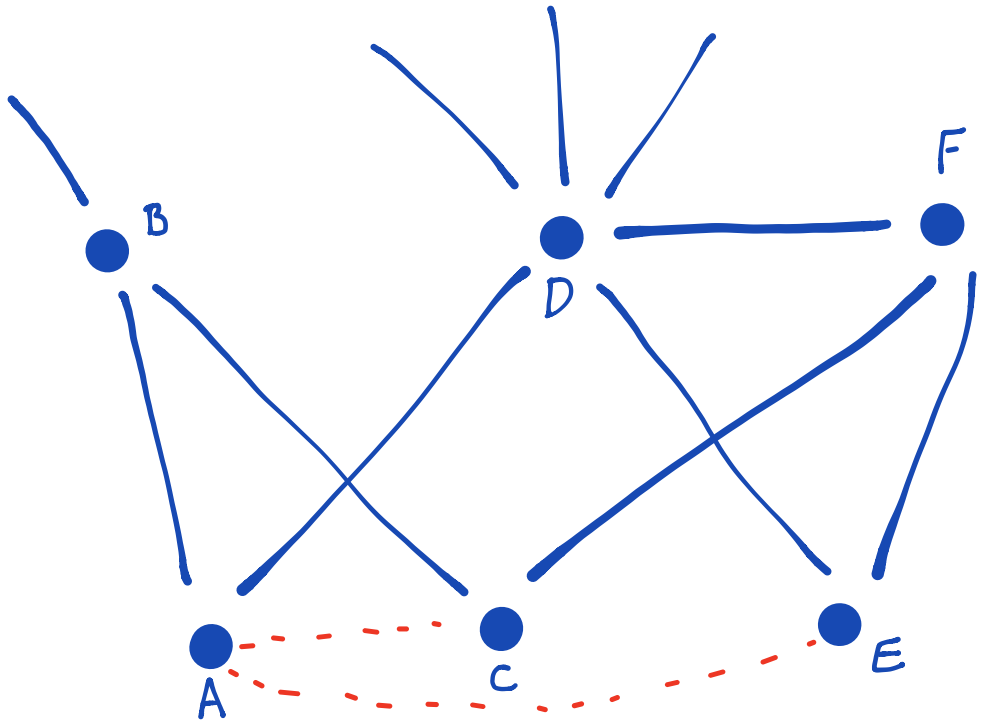
$S = \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 1 + \cdots$

Adamic/Adar

$$\text{Score}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log|\Gamma(z)|}$$

Frequency of z

Weighting rare features more heavily



$\Gamma(A) \cap \Gamma(C) = B$

$\frac{1}{\log(\Gamma(B))} = \frac{1}{\log 3} = 2.09$

$\Gamma(A) \cap \Gamma(E) = D$

$\frac{1}{\log(\Gamma(D))} = \frac{1}{\lg 6} = 1.2$

Hitting time

$$\text{Score}(x,y) = -H_{x,y}$$

expected time/step for random walk from x to reach y

Rooted PageRank → minimizes dependence on nodes far away from x and y

$$Score(x,y) = -H_{x,y} \cdot \pi_y$$

π_y: SDW of y
Proportion of time the random walk passes node y

- Prob = α
  - ↳ jump to X
- Prob = $1-\alpha$
  - ↳ go to random neighbor of current node

Friends-measure: When looking at two vertices in a social network, we can assume that the more connections their neighborhoods have with each other, the higher the chances are that the two vertices are connected. We accept the logic of this statement and define the *Friends-measure* as the number of connections between u and v neighborhoods. The formal definitions of *Friends-measure* is: Let be $G = \langle V, E \rangle$ and $u, v \in V$.

$$Friends - measure(u, v) = \sum_{x \in \Gamma(u)} \sum_{y \in \Gamma(v)} \delta(x, y) \tag{14}$$

where we define the function $\delta(x, y)$ as:

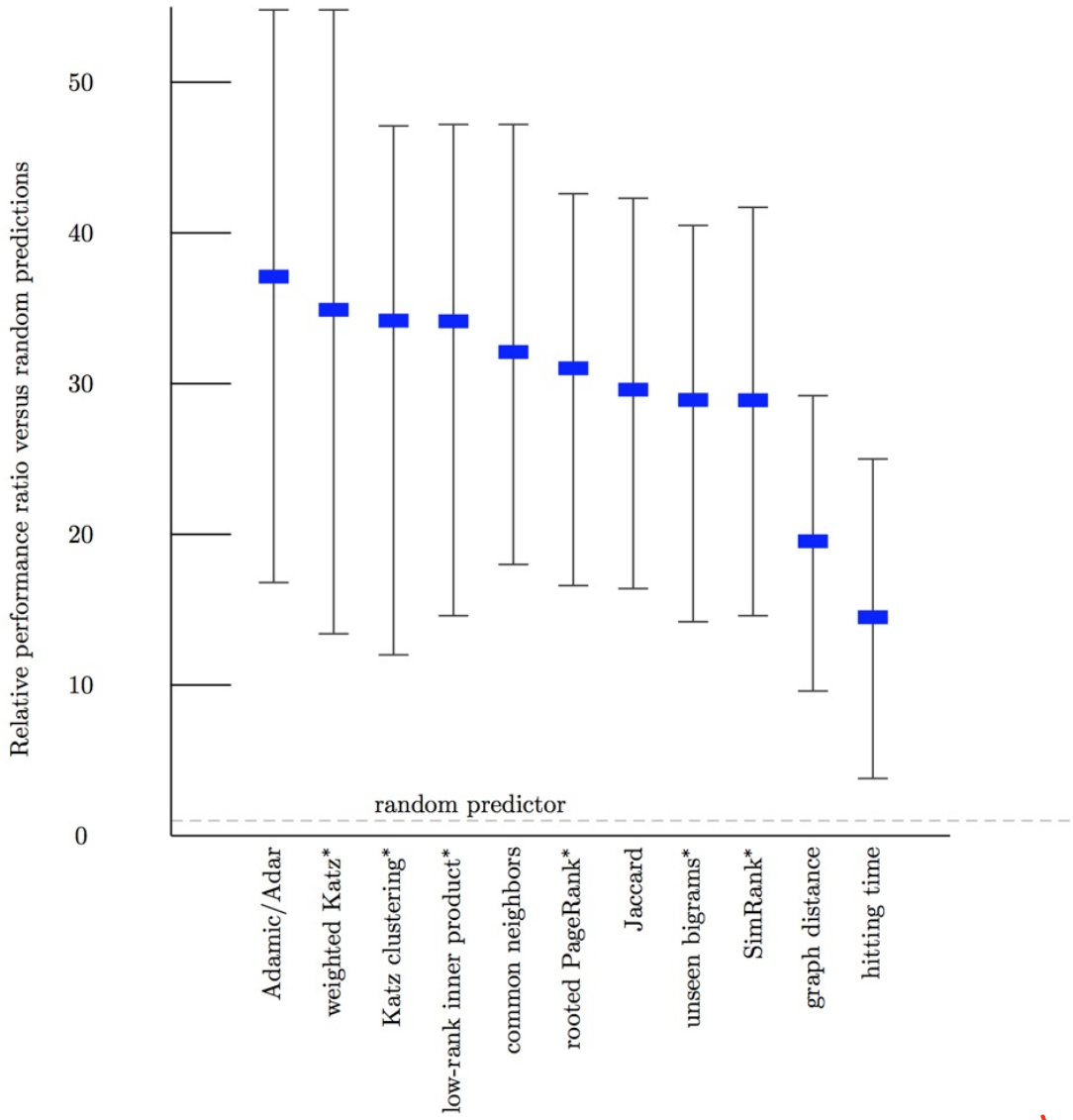
$$\delta(x, y) = \begin{cases} 1 & \text{if } x = y \text{ or } (x, y) \in E \text{ or } (y, x) \in E \\ 0 & \text{otherwise} \end{cases}$$

One can notice that in undirected networks, the *Friends-measure* is a private case of the *Katz-measure* where $\beta = 1$, $l_{min} = 2$, and $l_{max} = 3$.

Same-community: Let be $V = \coprod_{C' \in C} \coprod_{u \in C'} u$, where C is the set of all disjoint communities created from G by the Louvain method [4]. We say that $u, v \in V$ are in the same community if $\exists C' \in C$ where $u, v \in C'$. The formal definition of *same-community* feature is:

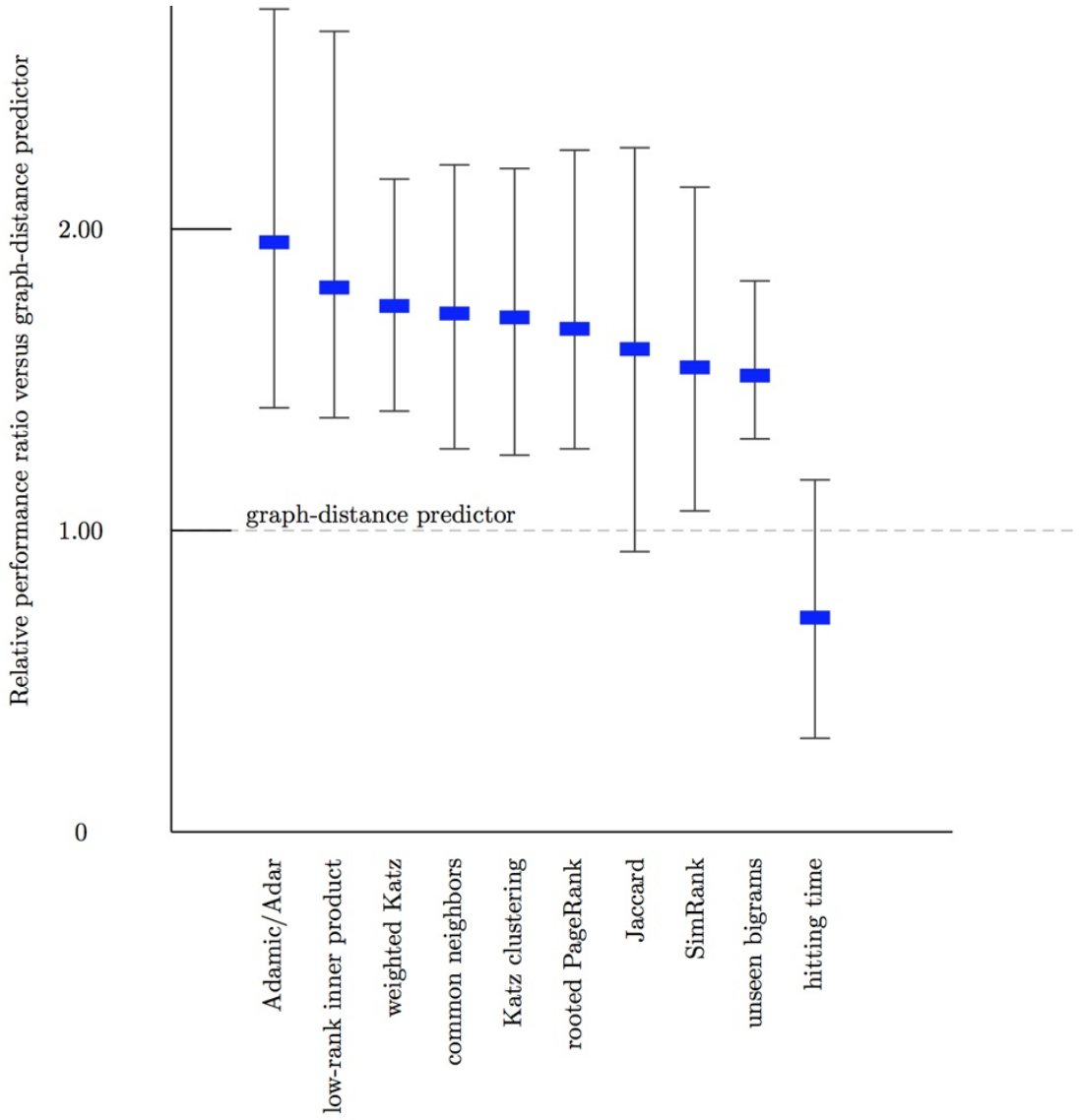
$$same\text{-}community(u, v) = \begin{cases} 1 & \text{if } \exists C' \in C \text{ where } u, v \in C' \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

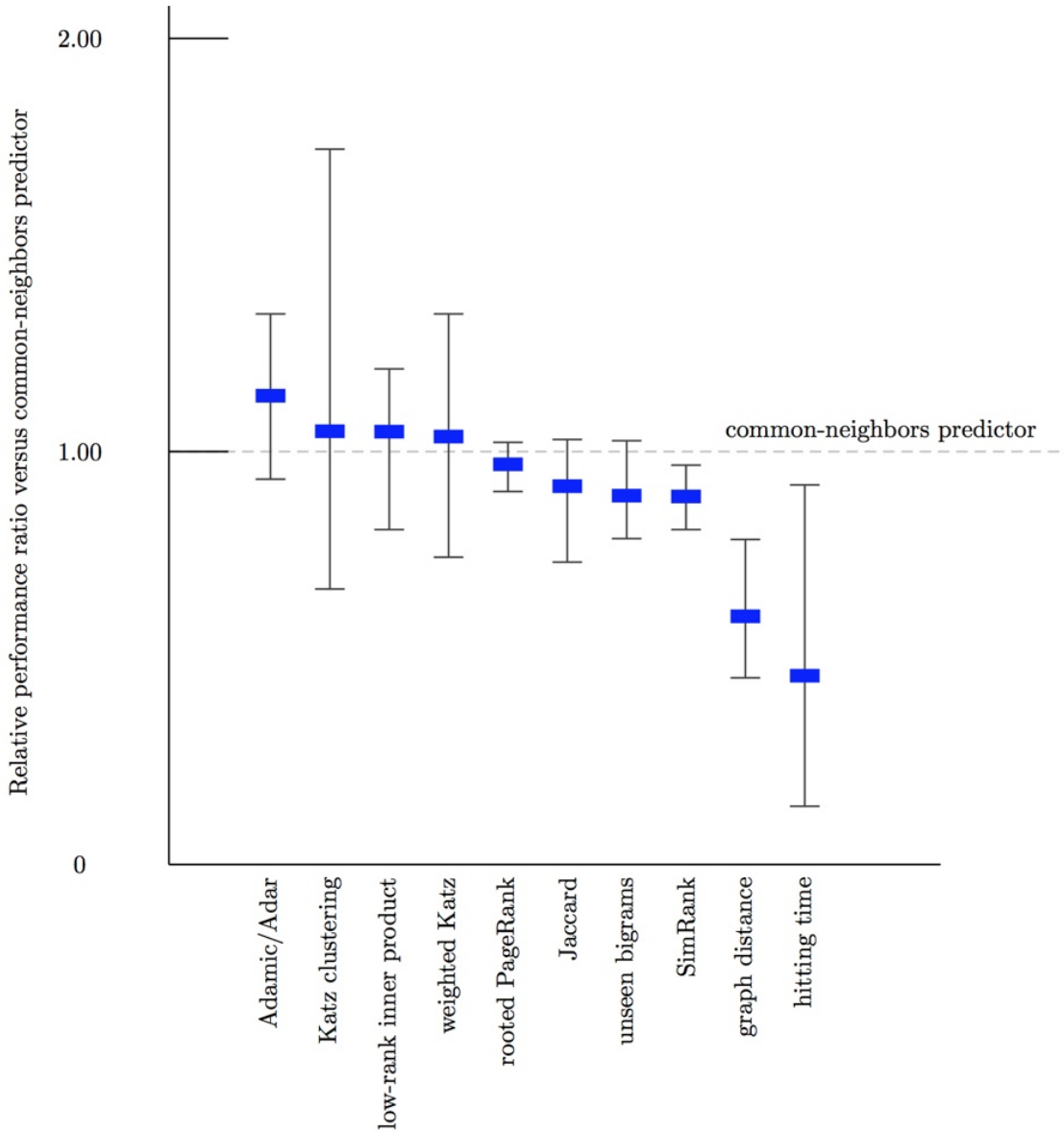
In case u and v were chosen at random, the Same-community feature can be used as an easy-to-compute rule of thumb for predicting existence of link between u and v[11].



Co-authorship Network from arXiv

-David Liben-Nowell


Relative performance ratio versus graph-distance predictor
2.00
1.00
0
graph-distance predictor
Adamic/Adar
low-rank inner product
weighted Katz
common neighbors
Katz clustering
rooted PageRank
Jaccard
SimRank
unseen bigrams
hitting time


Relative performance ratio versus common-neighbors predictor
2.00
1.00
0
common-neighbors predictor
Adamic/Adar
Katz clustering
low-rank inner product
weighted Katz
rooted PageRank
Jaccard
unseen bigrams
SimRank
graph distance
hitting time

predictor		astro-ph	cond-mat	gr-qc	hep-ph	hep-th
probability that a random prediction is correct		0.475%	0.147%	0.341%	0.207%	0.153%
graph distance (all distance-two pairs)		*9.6*	*25.3*	*21.4*	*12.2*	*29.2*
common neighbors		***18.0***	***41.1***	***27.2***	***27.0***	***47.2***
preferential attachment		4.7	6.1	7.6	*15.2*	7.5
Adamic/Adar		*16.8*	***54.8***	***30.1***	***33.3***	***50.5***
Jaccard		*16.4*	***42.3***	19.9	***27.7***	*41.7*
SimRank	$\gamma = 0.8$	*14.6*	*39.3*	*22.8*	*26.1*	*41.7*
hitting time		6.5	23.8	*25.0*	3.8	13.4
hitting time, stationary-distribution normed		5.3	23.8	11.0	11.3	21.3
commute time		5.2	15.5	***33.1***	*17.1*	23.4
commute time, stationary-distribution normed		5.3	16.1	11.0	11.3	16.3
rooted PageRank	$\alpha = 0.01$	*10.8*	*28.0*	***33.1***	*18.7*	*29.2*
	$\alpha = 0.05$	*13.8*	*39.9*	***35.3***	*24.6*	*41.3*
	$\alpha = 0.15$	*16.6*	***41.1***	***27.2***	***27.6***	*42.6*
	$\alpha = 0.30$	*17.1*	***42.3***	*25.0*	***29.9***	*46.8*
	$\alpha = 0.50$	*16.8*	***41.1***	*24.3*	***30.7***	*46.8*
Katz (weighted)	$\beta = 0.05$	3.0	21.4	19.9	2.4	12.9
	$\beta = 0.005$	*13.4*	***54.8***	***30.1***	*24.0*	***52.2***
	$\beta = 0.0005$	*14.5*	***54.2***	***30.1***	***32.6***	***51.8***
Katz (unweighted)	$\beta = 0.05$	*10.9*	***41.7***	***37.5***	*18.7*	***48.0***
	$\beta = 0.005$	*16.8*	***41.7***	***37.5***	*24.2*	***49.7***
	$\beta = 0.0005$	*16.8*	***41.7***	***37.5***	*24.9*	***49.7***

Figure 3-3: *Performance of the basic predictors on the link-prediction task defined in Section 3.2. See Sections 3.3.1, 3.3.2, and 3.3.3 for definitions of these predictors. For each predictor and each arXiv section, the displayed number specifies the factor improvement over random prediction. Two predictors in particular are used as baselines for comparison: graph distance and common neighbors. Italicized entries have performance at least as good as the graph-distance predictor; bold entries are at least as good as the common-neighbors predictor. See also Figure 3-4.*

Network-evolution models

- Social Network Analysis
  - Link prediction
    - Supervised learning
      - binary classifier
    - Unsupervised (non-learning)
      - Node based topological Similarity
        - common neighbors
        - Jaccard Coef.
        - Adamic/Adar
        - Prefrential Attachment

        Performs the best but does not Scale well
      - Path based topological Similarity
        - katz
        - Hitting time
        - Rooted PageRank ✓

thank you


People you may know
See All
I know all these people....
I also hate them.