Nodes' Evolution Diversity and Link Prediction in Social Networks

Presented by Zoykin Alexander

Tasks to solve

network evolution analysis

detecting anomalies in graph edges

link prediction

predicting new edges in a graph

Simple approach

- Collect features about nodes
- Train a model for similarity

Simple approach x2

$$CN(u,v) = \mid \Gamma(u) \cap \Gamma(v) \mid$$

$$JC(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|}$$

$$AA(u,v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{log(|\Gamma(w)|)}$$

$$RA(u,v) = \sum_{w \in \Gamma(u) \cap \Gamma(v)} \frac{1}{|\Gamma(w)|}$$

$$PA(u,v) = \mid \Gamma(u) \mid \times \mid \Gamma(v) \mid$$

$$AR(u,v) = \frac{2(ad-bc)}{(a+b)(b+d)+(a+c)(c+d)}$$

$$ND(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{\sqrt{|\Gamma(u)| \times |\Gamma(v)|}}$$

$$TN(u,v) = |\Gamma(u) \cup \Gamma(v)|$$

$$UD = \mid \Gamma(u) \mid$$

$$VD = \mid \Gamma(v) \mid$$

$$SC(u,v) = \begin{cases} 1 & \text{if u and v belong to the same community} \\ 0 & \text{otherwise} \end{cases}$$

- Γ - set of neighbours
- XGBoost with this features

Barabási–Albert model

Model tries to explain the laws of network evolution.

Nodes' degrees tend to be distributed with $P(k) \sim k^{-3}$

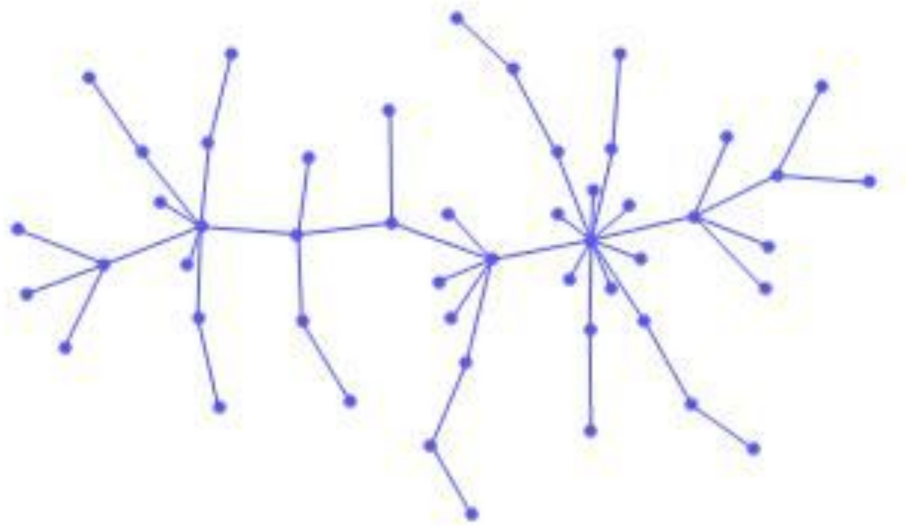
Why?

Preferential attachment

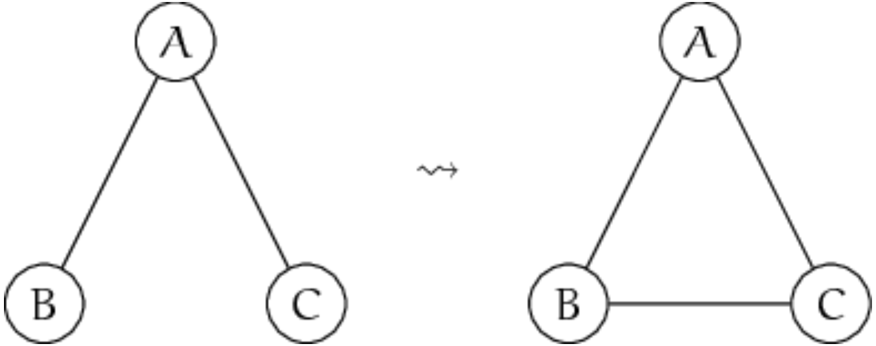
Connect new node to one of old ones, neighbour is more likely if it has more own neighbours.

Let p_i be probability of i-th node and k - number of neighbours

$$p_i = \frac{k_i}{\sum_j k_j}$$

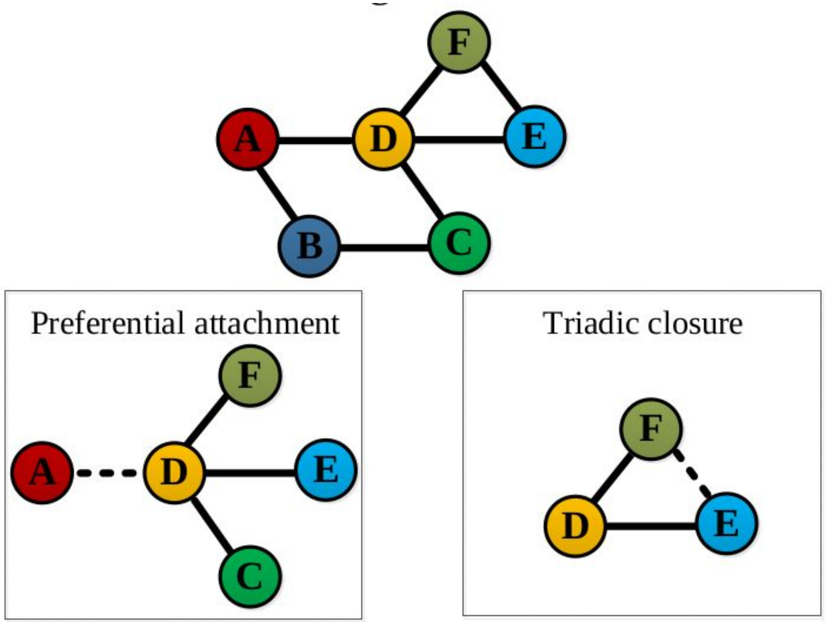


Triadic closure



Also - homophily, reciprocity and social balance

Sometimes it is better to use both



Nodes’ evolution diversity

Let’s have a separate prediction algorithm(from some given algorithm set) for every edge of every node of a graph. Let’s assume that graph should be full and find answers for all possible edges.

Network evolution analysis part

- Get a link prediction algorithm set
- Different algorithms(like preferential attachment) give different likelihoods for all edges in a graph.
- Get a Mt - matching set - algorithm to choose for every edge of every node

Network evolution analysis part

- Divide edges into train E^{tr} and test E^{pr}, $E = E^{tr} + E^{pr}$, $E^{tr} \cap E^{pr} = \varnothing$
- Predict edges E^{tr} based on Mt and E^{tr}

Evolution diversity evaluation

- egc - a measure to evaluate the extent to which the generation of an edge can be explained by a link prediction algorithm.
- $egc_{\langle i,j\rangle}(lp)$ $egc_{\langle j,i\rangle}(lp)$
- For a fixed algorithm, let’s compare likelihood value of an edge with another edges. Let’s say it is greater than n_0 of them and equal to n_1 of them. Than

$$egc_{\langle i,j\rangle}(lp) = \frac{n_0 + 0.5 \times n_1}{N}$$

- Select the best

ALGORITHM: *DNAA*

INPUT: $G = (V, E), \gamma, E,$ and ψ.

OUTPUT: E^{d}.

Step 1: Randomly select $\gamma|E|$ edges from E to constitute E^{pr}, and the other edges in E constitute E^{tr}.

Step 2: Calculate and rank the likelihood values of all the non-existent edges and the edges in E^{pr} by each $lp \in \psi$ based on E^{tr}.

For $i = 1$ to $N_V = |V|$:

  For $j = 1$ to $N_V = |V|$:

    If $i < j$ and $\langle i, j \rangle$ not in E^{tr}:

      Step 3: Determine $E_p(i)$ and $E_p(j)$.

      Step 4: $egc_{(i,j)} = max\{egc_{\langle i,j \rangle}(lp), egc_{\langle j,i \rangle}(lp)\}$, where $lp \in \psi$ (detailed in Section 4).

    End

  End

End

Step 5: Rank all the non-existent edges and the edges in E^{pr} in descending order of egc.

Step 6: Select the edges in the top $|E^{pr}|$ places as E^d.

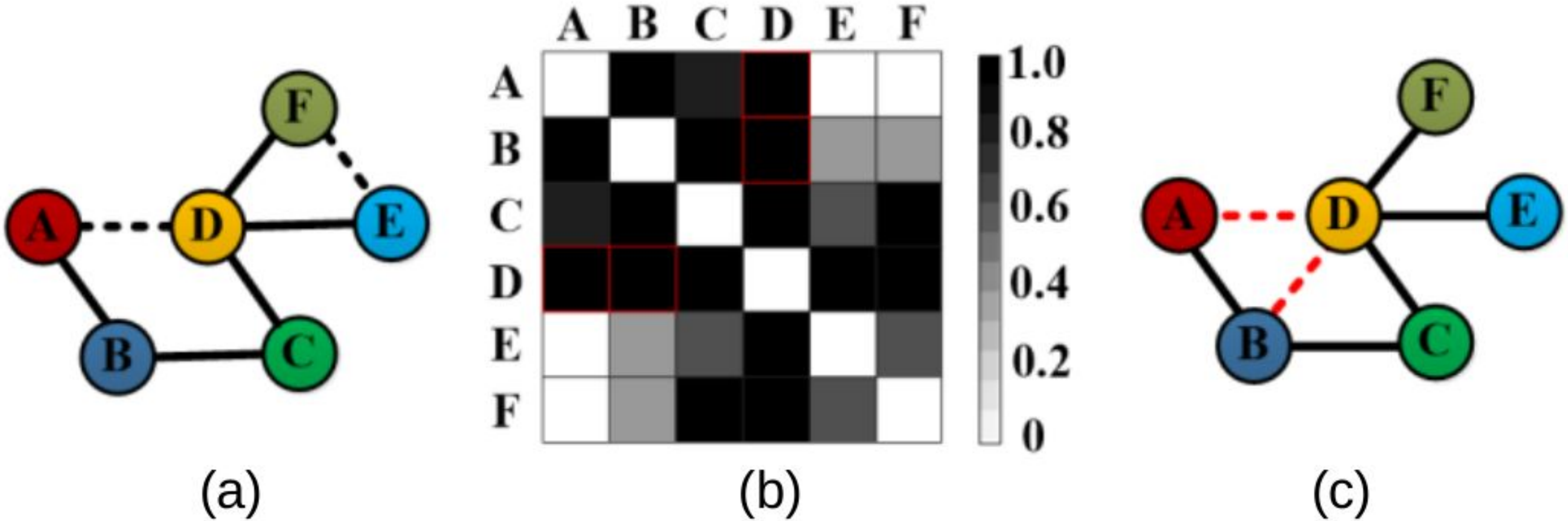


Fig. 3. Evaluating the performance of *DNAA*: (a) the probe set $E^{pr} = \{(A, D), (F, E)\}$ (black dashed edges), (b) the egc values of unobserved edges, (c) predicted social network (red dashed edges are the outcome).

TABLE 3

AUC VALUES OF DIFFERENT LINK PREDICTION ALGORITHMS FOR 10 REAL-WORLD NETWORKS

AUC	**Jazz**	**Wikivote**	**Facebook**	**SciNet**	**Enron**	**Epinions**	**AgrCol**	**Slashdot**	**Twitter**	**BrightKite**
PA	0.774	***0.985***	0.724	0.834	***0.893***	0.768	0.845	***0.915***	0.816	0.623
ACT	0.801	0.754	0.729	0.491	0.732	***0.894***	0.812	0.823	0.867	0.791
Lop	0.924	***0.983***	0.749	***0.892***	0.737	0.846	***0.885***	***0.901***	0.896	0.734
Katz	0.419	0.498	***0.938***	0.876	0.621	0.756	0.724	0.811	0.652	0.725
CN	0.955	0.852	0.571	0.892	0.867	0.823	0.831	0.827	0.834	0.723
AA	0.962	0.853	0.571	0.793	0.867	0.834	0.834	0.845	0.845	0.824
RA	***0.969***	0.853	0.571	0.891	0.866	0.857	0.823	0.834	0.839	0.813
HPI	0.942	0.843	0.572	***0.894***	0.859	0.814	0.811	0.832	0.728	0.811
HSM	0.912	0.894	0.612	0.875	0.820	***0.913***	0.849	0.891	***0.915***	***0.834***
SPM	0.794	0.867	0.634	0.889	***0.891***	0.891	***0.886***	0.876	0.920	0.813
SBM	*0.855*	*0.832*	*0.657*	*0.852*	*0.842*	*0.812*	*0.834*	*0.866*	*0.837*	***0.857***
LM	**0.973**	*0.932*	***0.857***	*0.876*	*0.863*	*0.889*	*0.871*	*0.887*	***0.937***	*0.821*
DNAA	0.985	0.991	0.972	0.922	0.919	0.932	0.901	0.947	0.989	0.899

Used literature

1. https://hackernoon.com/link-prediction-in-large-scale-networks-f836fcb05c88 - an article describing simple link prediction algos
2. https://sci-hub.tw/10.1109/TKDE.2017.2728527 - sci-hub link to original article
3. https://pdfs.semanticscholar.org/2d9c/af713b2dc3abf09fd94991ad2d1587d3e12c.pdf - article about triadic closure graph construction
4. https://arxiv.org/abs/cond-mat/0106096 - Barabasi-Albert model