Social Media Analytics: Graph Essentials

Lecture:8

Garima Jindal

BITS Pilani
Pilani Campus

Link Prediction

Applications

Evaluation

link prediction methods

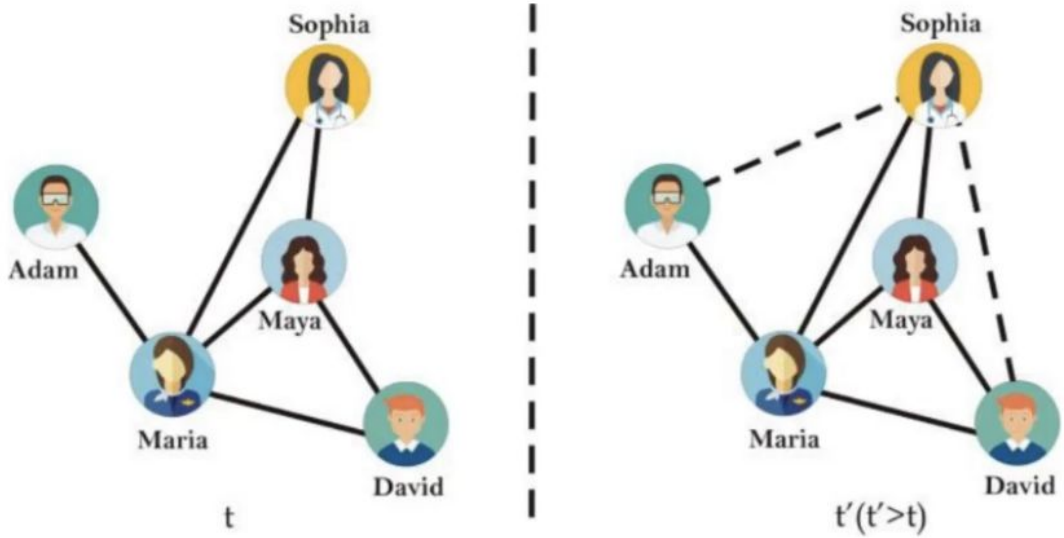
Local similarity based

Global similarity based

Link prediction

What is Link Prediction?

- The problem of predicting the existence of a link between two entities in a network
- Involve several research communities ranging from statistics and network science to machine learning and data mining
- Help in predicting the state of a dynamic network at future timestamp



https://www.nature.com/articles/s41598-019-57304-y

Application Areas

Online Social Networks

- Recommend friends to connect
- Suggest users/pages to follow

E-commerce

- Recommend products/services

Police/Military

- Identify hidden groups of terrorists
- Spot criminals in security related applications

Bioinformatics/Biology

- Predict protein-protein interactions
- Infer interactions between drugs and targets

Network Reconstruction

- Remove spurious edges
- Predict missing links
- Predict new links

Citation Networks

- Predict missing citations
- Predict future collaboration

Types of Link Prediction Problems

Missing Link Prediction

- Goal: Identify **edges that already exist** but are **not observed**
- Assumes the graph is **incomplete**
- Links are **static** (no time component)
- Common in:
  - Incomplete data collection
  - Noisy or hidden networks

Example:
Two users are already friends in reality, but the connection is missing in the dataset.

Types of Link Prediction Problems

. Future Link Prediction

- Goal: Predict **edges that will form in the future**
- Graph is **dynamic and evolving**
- Strongly depends on **temporal patterns**
- Common in:
  - Social networks
  - Recommendation systems
  - Citation networks

Example:
Predicting which users will become friends next month.

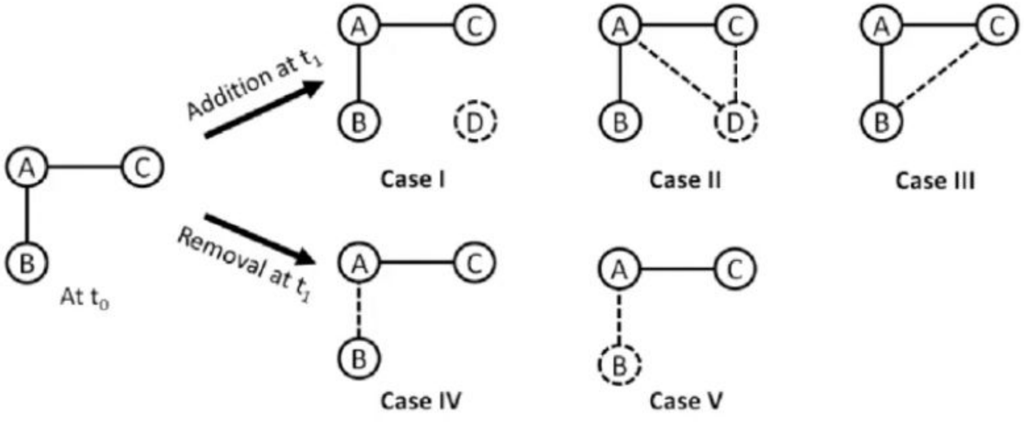
Key Differences Between Missing vs Future Link Prediction

Aspect	Missing Link Prediction	Future Link Prediction
Time	No time component	Time-aware
Graph Type	Static	Dynamic
Objective	Recover hidden links	Forecast new links
Data Assumption	Incomplete graph	Evolving graph
Difficulty	Structural ambiguity	Temporal uncertainty

Temporal Changes in a Network

$G_{t_0}(V, E)$: Topology at time $t = t_0$ $G_{t_1}(V', E')$: Topology at time $t = t_1$ $(t_1 > t_0)$

- Case I
  - New nodes are added, but they do not form any link
- Case II
  - New nodes join and form new connections
- Case III
  - No new nodes join, but some new edges are formed
- Case IV
  - Some existing edges are removed, but endpoints retained
- Case V
  - Some existing nodes and edges are removed



Note: Shall restrict discussions on case III only

Link Prediction: Problem Definition

Given following snapshots of a network $G_{t_0}(V,E)$ at time $t = t_0$ and $G_{t_i}(V,E')$ at time $t = t_i\ (> t_0)$, the set $E' \setminus E$ of edges joined the network during the time interval $[t_0, t_i]$. Then the task of link prediction is defined as the prediction of the edge set $E' \setminus E$ at time $t = t_0$

<u>Alternatively,</u>

The problem of link prediction can be coined as: the task of determining the likelihood that any two nodes that are not connected at time $t = t_0$, will be connected at time $t = t_i\ (t_i > t_0)$,

https://www.youtube.com/watch?v=OdlRxeHjYBA&t=6s

How difficult id this problem??

Link Prediction: Problem Setting & Challenges

Graph Characteristics

- Consider a graph with **V = 100 nodes**
- Maximum possible edges:

$$\binom{100}{2} = 4950$$

- However, real-world graphs are **sparse**
  - Typically:

$$O(E) = O(V)$$

  - So, number of actual edges ≈ **100**

Positive samples (existing edges) ≈ 100
Negative samples (non-existing edges) ≈ 4950-100 **= 4850**

This creates a **severe class imbalance**:

- Very **few positive linkse samples**
- **Huge number of negativ**

Link prediction is challenging not because of graph size, but due to **extreme sparsity and imbalance between positive and negative samples**.

Evaluating Link Prediction Methods: Train-Test Split

Case I (task of inferring missing links)

- Only a single snapshot $G_{t_i}(V, E)$ of the network at timestamp $t = t_i$
- Split E into disjoint sets E_{train} and E_{test}
- To obtain test set, delete edges from E and add them to E_{test}
- Deletion strategies:
  - Uniformly at random
  - Based on the degrees of their endpoints
  - Based on the degrees of their endpoints

Case II (task of predicting future links)

- At least two snapshots of the network: $G_{t_i}(V, E)$ at time $t = t_i$ and $G_{t_j}(V, E')$ at time $t = t_j\ (> t_i)$
- Set $E_{train} = E$ and $E_{test} = E' \setminus E$

Evaluating Link Prediction Methods: Positive-Negative Samples

- Initial Network: $G_{t_i}(V, E)$ of the network at timestamp $t = t_i$
- Set of all possible edges in the network: $U = 2^E$
- Obtain E_{train} and E_{test} following either of the cases mentioned in earlier
- Set of edges not formed till timestamp $t = t_i$: $L = U \setminus E_{train}$
- Convert the problem of link prediction into a binary classification problem
  - Edges in E_{test} form the positive samples
  - Edges in set $L \setminus E_{test}$ form the negative samples
- Positive samples are expected to have higher probability than negative samples

Evaluating Link Prediction Methods: Confusion Matrix

Actual → / Predicted ↓	Link formed	Link not formed
Link formed	True Positive (TP)	False Positive (FP)
Link not formed	False Negative (FN)	True Negative (TN)

- **True Positive (TP):** the count of how many times the model predicted a link to be formed, and it actually forms
- **True Negative (TN):** the count of how many times the model predicts that a link will not form, and it actually does not form

False Positive (FP): the count of how many times the model predicts a link to be formed; however, it actually does not form

False Negative (FN): the count of how many times the model predicts a link will not form; however, it actually forms (opposite case of FP)

Question

Actual → / Predicted ↓	Link formed	Link not formed
Link formed	True Positive (TP)	False Positive (FP)
Link not formed	False Negative (FN)	True Negative (TN)

- **Actual non-edges = TN + FP**
- **Predicted edges = TP + FP**

- **Actual non-edges = TN + FP**
- **Predicted edges = TP + FP**

Evaluating Link Prediction Methods: Confusion Matrix

- Accuracy (ACC): ratio of the total number of correct predictions to the total number of predictions

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision (P): out of all the links that are predicted by the model as positive, how many does actually belong to the positive samples

$$P = \frac{TP}{TP + FP}$$

- Recall (R): out of all the links that are actually positive, how many are predicted as positive by the model

$$R = \frac{TP}{TP + FN}$$

Evaluating Link Prediction Methods: Confusion Matrix

- True Negative Rate (TNR)/specificity: Out of all the links that are actually negative, how many are predicted by the model to be negative

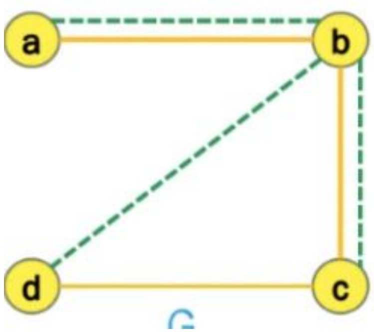
$$TNR = \frac{TN}{TN + FP}$$

- False Positive Rate (FPR)/false alarm ratio/fallout rate: Out of all the negative samples, how many are wrongly predicted to belong to positive class instead

$$FPR = \frac{FP}{FP + TN}$$

Question

Confusion Matrix: Illustration



To find confusion matrix related metrics for the prediction of network G in the figure

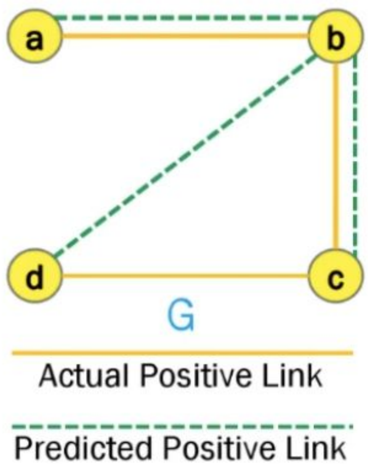
Actual positive links = $\{(a,b),(b,c),(c,d)\}$, Actual negative links = $\{(a,d),(a,c),(b,d)\}$

Pred. positive links = $\{(a,b),(b,c),(b,d)\}$, Pred. negative links = $\{(a,d),(a,c),(c,d)\}$

Find

1. TP, TN, FP, FN

Confusion Matrix: Illustration



To find confusion matrix related metrics for the prediction of network G in the figure

Actual positive links = $\{(a,b),(b,c),(c,d)\}$, Actual negative links = $\{(a,d),(a,c),(b,d)\}$

Pred. positive links = $\{(a,b),(b,c),(b,d)\}$, Pred. negative links = $\{(a,d),(a,c),(c,d)\}$

$TP = 2,\quad TN = 2,\quad FP = 1,\quad FN = 1$

$ACC = \frac{2+2}{2+2+1+1} = \mathbf{0.67}\quad P = \frac{2}{2+1} = \mathbf{0.67},\quad R = \frac{2}{2+1} = \mathbf{0.67}$

$TNR = \frac{2}{2+1} = \mathbf{0.67}\quad FPR = \frac{1}{1+2} = \mathbf{0.33}$

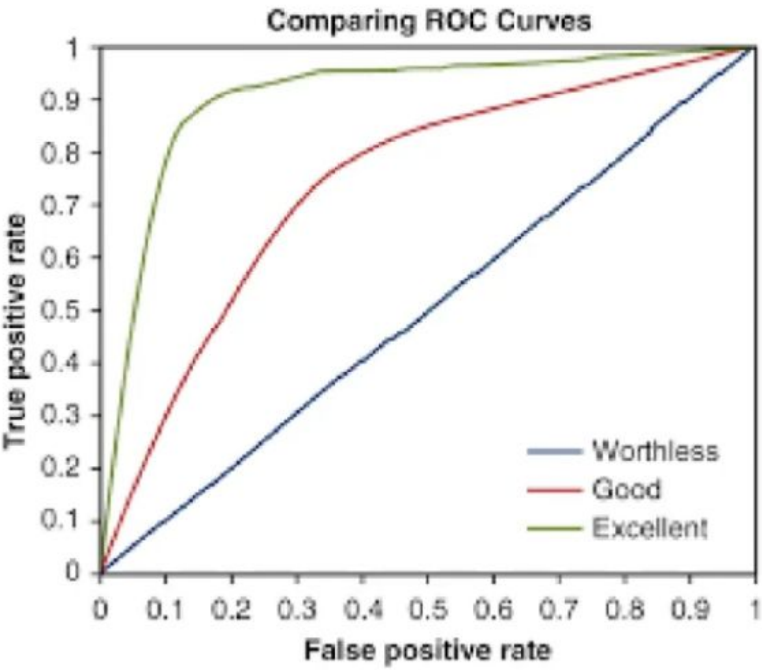
Evaluating Link Prediction Methods: Mere Accuracy Is Not Enough

- A typical example network that consists of
  - 9 actual negative edges
  - 1 actual positive edges
- A typical prediction model:
  - the model predicted 10 negative edges in the network
- *How Good is the Prediction Model???*

- $TP = 0,\ TN = 9,\ FN = 1,\ FP = 0$
- $\boldsymbol{ACC = 90\%}$

- However, the model failed at its desired task of predicting a rare positive edge in the network!!!

Evaluating Link Prediction Methods: AUC-ROC



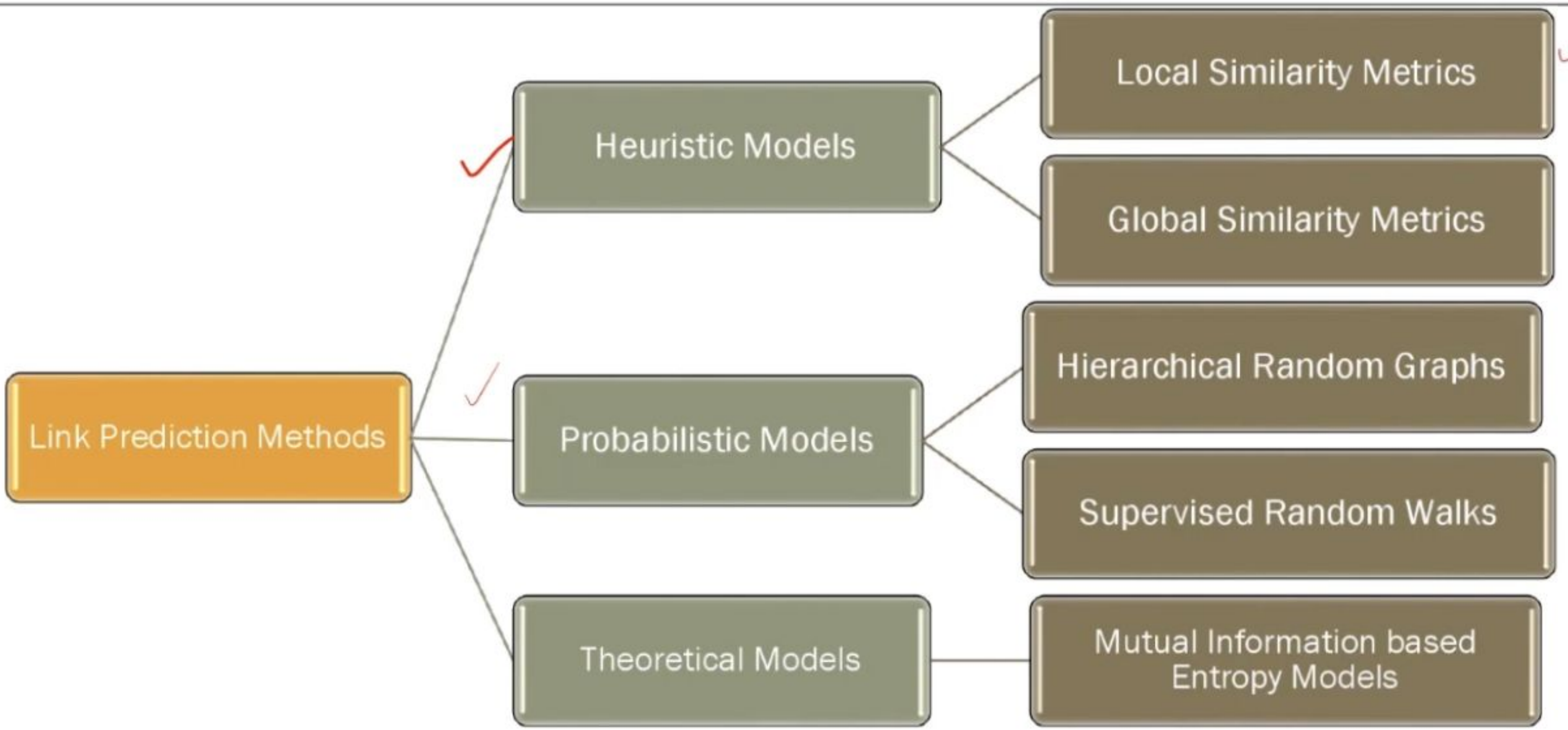
http://gim.unmc.edu/dxtests/roc3.htm

- Area Under the Receiver Operating Characteristic (AUC-ROC) curve
- Area mapped under the plot comparing the true positives with the false positives
- Score lies in the range of [0, 1]
- Determines how strong or weak the prediction model is compared to a random model

Evaluating Link Prediction: Some Unique Problems

- Temporal dynamics of the network
  - temporal changes hard to obtain from the real-world data
  - difficult to fit in a binary-classification scenario
  - a complicated mixture of addition and deletion of nodes and edges: complicates the comparison of the network instances
- Directionality of the edges
  - confusion matrix and AUC scores to be calculated accordingly
- Sign and weight of links
  - All the edges are not equally important
- Class imbalance
  - Size of negative samples is often much larger than that of the positive samples

Link Prediction Methods



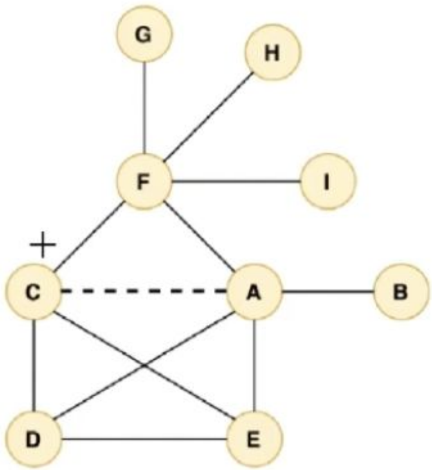
Link Prediction: Heuristic Models

- Irrespective of the techniques used, the underlying idea of link prediction is
  - to successfully connect nodes that share some similarities, but are not linked as of now
  - closer/similar two nodes are, the more likely they are to be in agreement, and more likely they are to interact
- Similarity between the nodes can be derived using a combination of properties
  - Level of nodes
  - Level of edges
  - Level of metadata to the nodes
- Heuristic measures of structural similarity
  - Local Heuristics
  - Global Heuristics
  - Quasi-local Heuristics

Link Prediction: Local Heuristic

- $G(V, E)$: an undirected dynamic network
- Three nodes $x, y, z \in V$ such that, at the current time instance
  - $(x, z) \in E$, $(y, z) \in E$
  - $(x, y) \notin E$
  - To decide the formation of the link (x, y) in near future
- Some local structural similarity base heuristic for the above
  - Common Neighbourhood
  - Jaccard Similarity
  - Preferential Attachment
  - Adamic Adar
  - Salton Index
  - Hub Promoted Index

Local Heuristic: Common Neighborhood



G_1

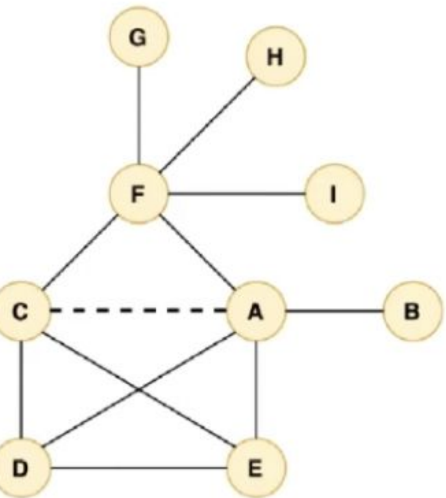
- Triadic closure property
  - By virtue of the common friend z, x and y are highly likely to be friends in future
- Common Neighborhood score between two randomly selected nodes x and y

$$S_{CN}(x,y) = |\Gamma(x) \cap \Gamma(y)|$$

Where $\Gamma(v)$: Neighbourhood set of node v

- Higher the number of common neighbours, more likely the node will be linked in future
- Example: In network G_1, $S_{CN}(A,C) = |\{B,D,E,F\} \cap \{D,E,F\}| = \mathbf{3}$

Local Heuristic: Jaccard Similarity



G_1

- Normalized version of common neighborhood score
- Jaccard Similarity score between two randomly selected nodes x and y

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

- The ratio of the number of common neighbors and the number of all neighbors of these two nodes
- Example: In network G_1, $S_J(A,C) = \frac{|\{B,D,E,F\} \cap \{D,E,F\}|}{|\{B,D,E,F\} \cup \{D,E,F\}|} = \frac{3}{4} = \mathbf{0.75}$

What is the Salton Index?

The **Salton Index** (also called **Cosine Similarity**) measures **similarity between two sets or vectors**.

It's commonly used in:

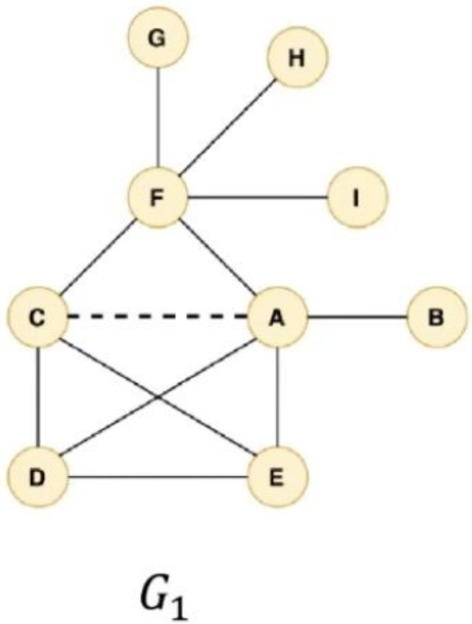
- information retrieval
- text mining
- recommender systems
- network / graph analysis (node similarity)

Formula

For two sets (or vectors) **A** and **B**:

$$\text{Salton}(A, B) = \frac{|A \cap B|}{\sqrt{|A| \cdot |B|}}$$

Local Heuristic: Preferential Attachment

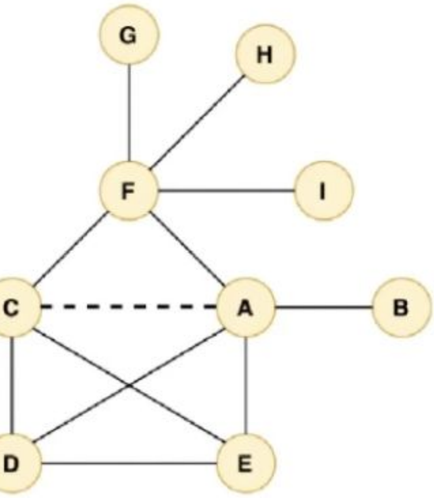


- Derived from the concept of preferential attachment of scale-free networks
- Likelihood of a node x to obtain a new edge is proportional to k_x, the degree of the node
- Preferential Attachment score between two randomly selected nodes x and y

$$S_{PA}(x, y) = k_x \times k_y$$

- Future interaction between them depends on the existing degree of the individual nodes
- Example: In network G_1, $S_{PA}(A, C) = k_A \times k_C = 4 \times 3 = \mathbf{12}$

Local Heuristic: Adamic Adar



G_1

- Primary Objective: shift focus towards rare events
- Assigns higher weights to less-connected nodes
- Adamic Adar metric between two randomly selected nodes x and y

$$S_{AA}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log k_z}$$

- Resource Allocation Index is variant of the above metric

$$S_{RA}(x,y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{k_z}$$

- Example: In network G_1, $S_{AA}(A,C) = \frac{1}{\log 3} + \frac{1}{\log 3} + \frac{1}{\log 5} = \mathbf{5.62}$

Local Heuristic: Salton Index and Others

- Salton Index (Commonly used metric to measure the similarity between a pair of documents or embeddings in a vector space) between two randomly selected nodes x and y

$$S_{SI}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{k_x \times k_y}}$$

- Hub Promoted Index (used to assign high scores to links adjacent to hubs) between two randomly selected nodes x and y

$$S_{SI}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{min(k_x, k_y)}$$

- Hub Depressed Index (used to assign low scores to links adjacent to hubs) between two randomly selected nodes x and y

$$S_{HDI}(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{max(k_x, k_y)}$$

Global Heuristic: Katz Score

- ❑Inspired by Katz centrality
- ❑Takes into account the influence by neighbors beyond 1-hop
- ❑However, longer the path length, less likely the end nodes influence each other
- ❑Between two random nodes x and y, Katz score is given by

$$S_{KZ}(x, y) = \sum_{p=1}^{\infty} \alpha^p \cdot A_{x,y}^p$$

Here, $A_{x,y}^p$: number of paths of length p that exists between x and y

and α: damping factor that reduces the impact of longer paths

Global Heuristic: Hitting Time

- Based on random surfing model. A random surfer
  - a) starts at node x
  - b) moves to a neighbor of x chosen uniformly at random
  - c) repeats step (a) till it reaches y
- Hitting time (HT_{xy}): Expected number of steps it takes for a random surfer starting at x to reach y
- The Hitting Time score between nodes x and y is given by

$$S_{HT}(x,y) = -HT_{xy}$$

- Smaller the hitting time between two nodes, closer in proximity the nodes, therefore higher the chances of their interaction in future
- The Normalized Hitting Time score between nodes x and y is given by

$$S_{HT}^{Norm}(x,y) = -HT_{xy} \cdot \pi_y$$

Here π: stationary distribution of PageRank for the network

Questions?

BITS Pilani

Pilani Campus