Social Network Analysis: Ideas, Algorithms and Applications

S. Durga Bhavani
School of Computer and Information Sciences,
University of Hyderabad, Hyderabad.

Invited talk at: The Department of Computer Science, Pondicherry University, India

Networks Everywhere!

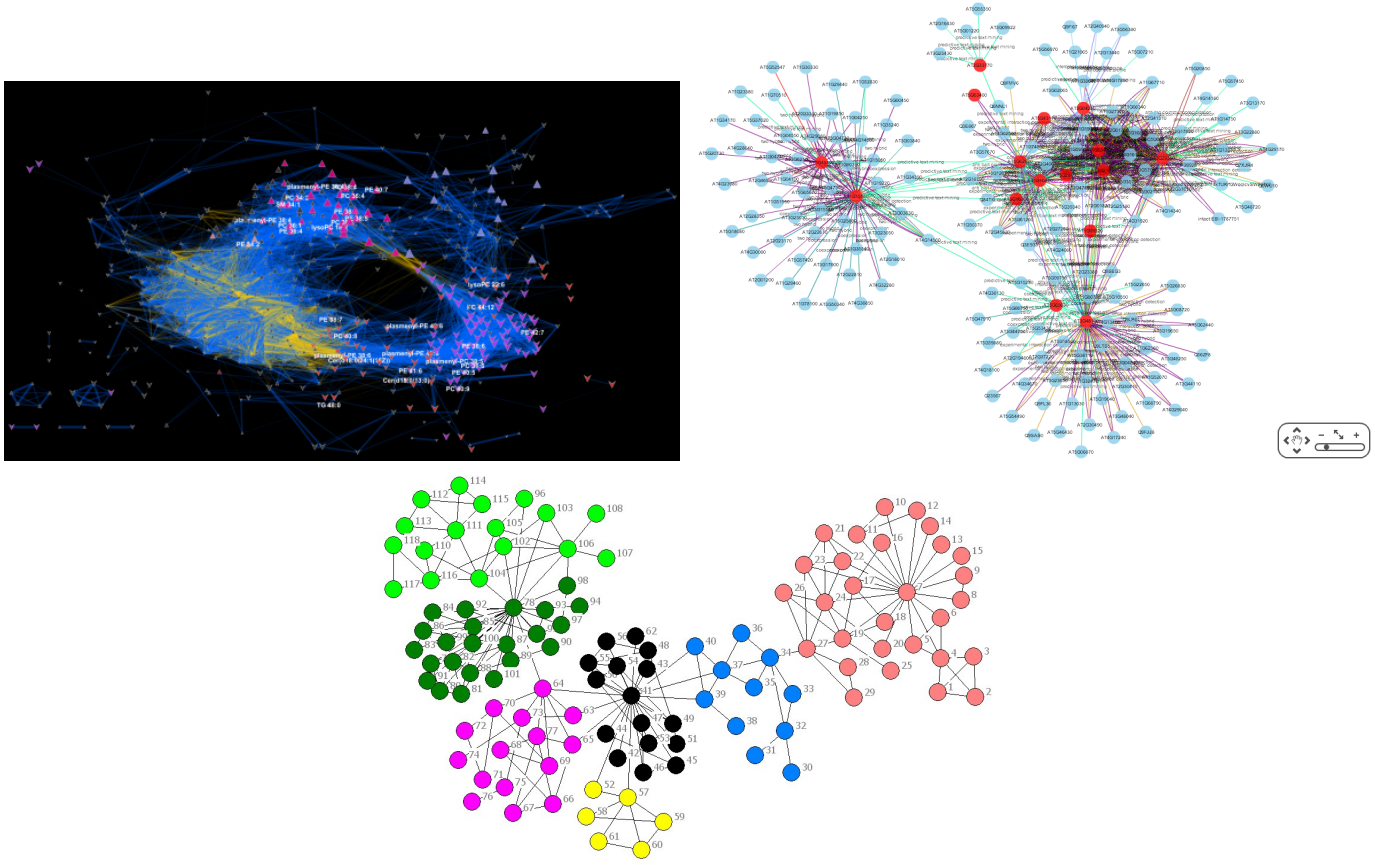


Figure: genotype network, PPI network and author collaboration network

Network

Networks are modeled as graphs $G = (V, E)$ where nodes in the network represent genes/ proteins/ authors and edges denote interactions between pairs of nodes.

- Networks are typically large directed/undirected weighted graphs
- Traditional graph theoretical analysis may not help since these are dynamic graphs and many a time probabilistic graphs.
- Combination of ideas from Physics for growth models; dynamical systems, probabilistic distributions and statistical analysis are applied.
- Hence Network Science is a highly interdisciplinary science.

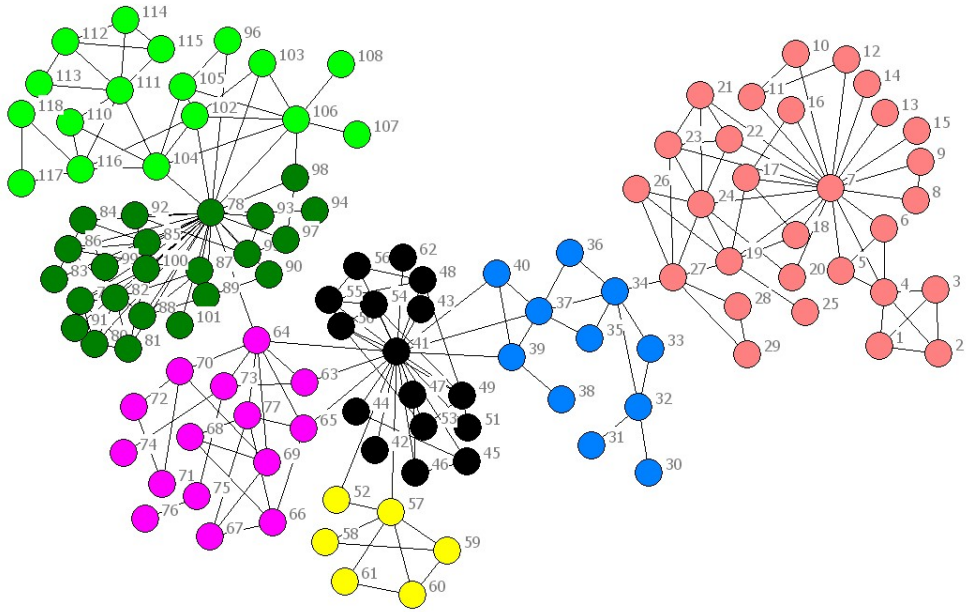
Contents Outline

- Ideas: Network Centrality Measures
- Ideas: Six degree separation, Preferential attachment,
- Network Models

Contents Outline

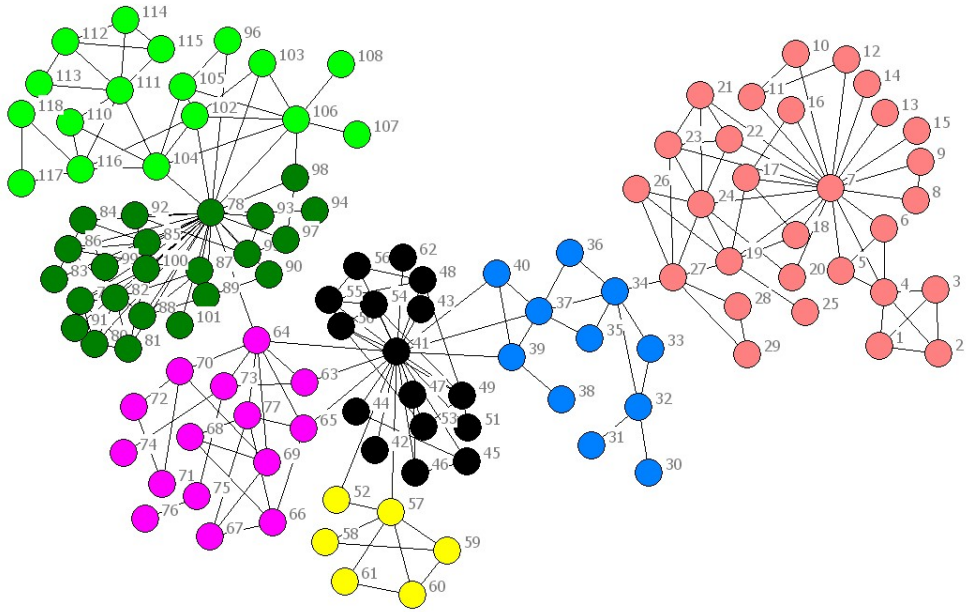
- Ideas: Network Centrality Measures
- Ideas: Six degree separation, Preferential attachment,
- Network Models
- Interesting Problems
  - Hub and authority node detection
  - Citation recommendation

Network Centrality Measures



- Degree Centrality: d_i is the number of neighbours of node i

Network Centrality Measures



- Degree Centrality: d_i is the number of neighbours of node i
- Clustering Coefficient: What is the probability that two of my friends are also friends?

$$CC(i) = \frac{2T_i}{d_i(d_i - 1)}$$

Where T_i is the number of triangles at the given node i.

Six-Degree Separation: It's a small world!

Stanley Milgram's famous social experiments.

- Milgram chose individuals from South-West: cities of Omaha, Nebraska, and Wichita, Kansas, to be the starting points and who have to send letters to reach people in the far east.

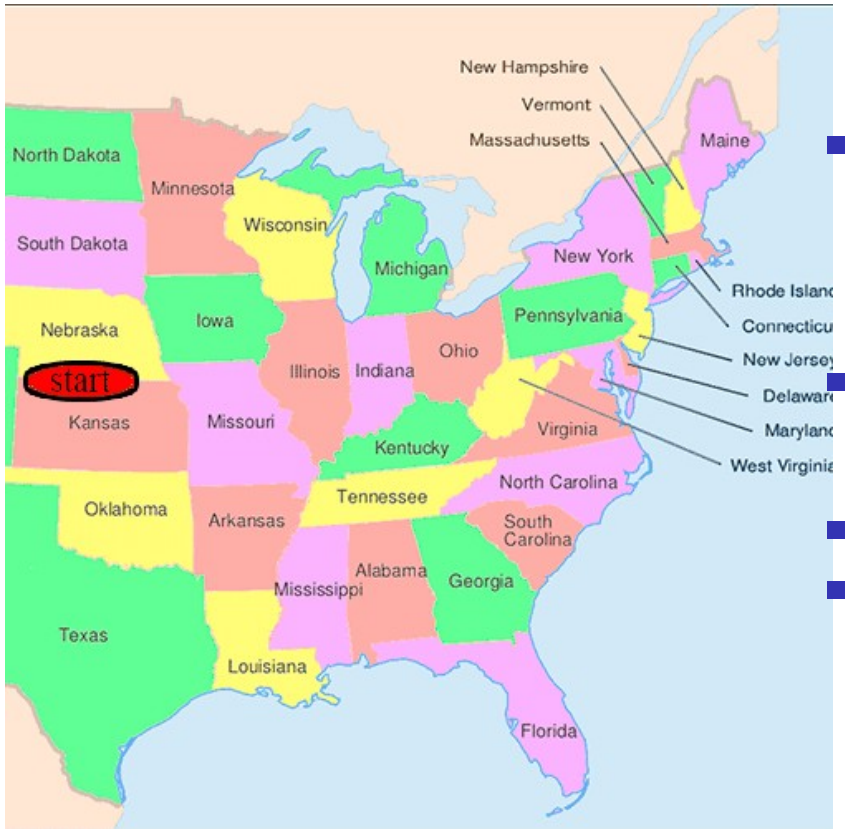
Six-Degree Separation: It's a small world!

Stanley Milgram's famous social experiments.

- Milgram chose individuals from South-West: cities of Omaha, Nebraska, and Wichita, Kansas, to be the starting points and who have to send letters to reach people in the far east.
- The letter can be posted to only people that you know who you think are likely to know the target address.
- Many letters never reached the destination!

Six-Degree Separation: It's a small world!

Stanley Milgram's famous social experiments.



- Milgram chose individuals from South-West: cities of Omaha, Nebraska, and Wichita, Kansas, to be the starting points and who have to send letters to reach people in the far east.
- The letter can be posted to only people that you know who you think are likely to know the target address.
- Many letters never reached the destination!
- In one experiment, 64 of the letters eventually did reach the target contact. Among these chains, the average path length found to be between $5\frac{1}{2}$ and 6.

Real-world Networks

The networks that occur in nature/real world are NOT Random networks.

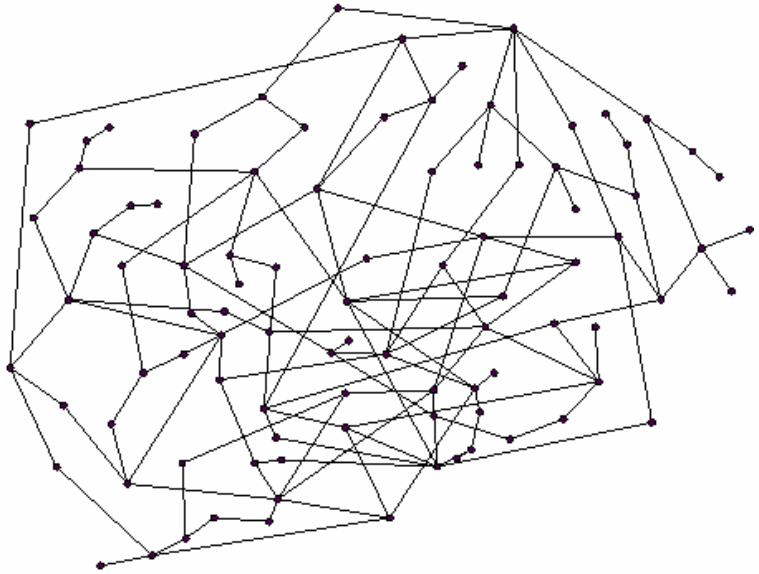
- Erdos-Renyi(ER) (Random Network)

- Barabasi-Albert Model (Real-world Network)

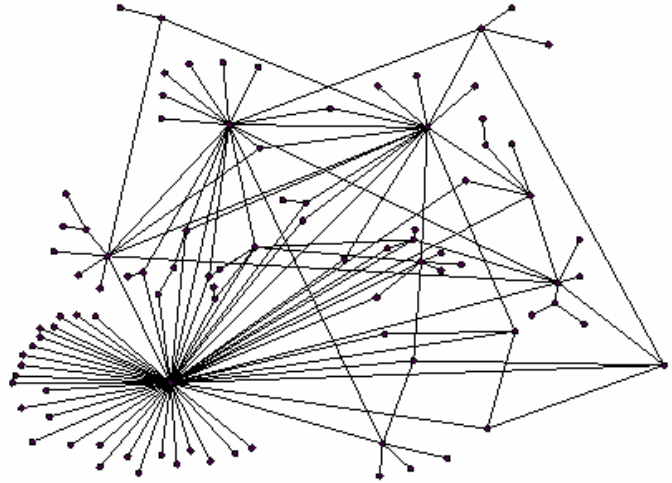
Real-world Networks

The networks that occur in nature/real world are NOT Random networks.

- Erdos-Renyi(ER) (Random Network)



- Barabasi-Albert Model (Real-world Network)



Pioneers of Network Models

Figure: Random Networks: Paul Erdos and Alfred Renyi

Pioneers of Network Models

Figure: Random Networks: Paul Erdos and Alfred Renyi

Figure: Preferential Attachment: A-L.Barabasi and Reka Albert

Idea: Preferential Attachment

Preferential attachment: A process in which a new node entering the graph does NOT connect randomly but shows a preference to attach to a node of a certain type (Eg. higher degree).

Idea: Preferential Attachment

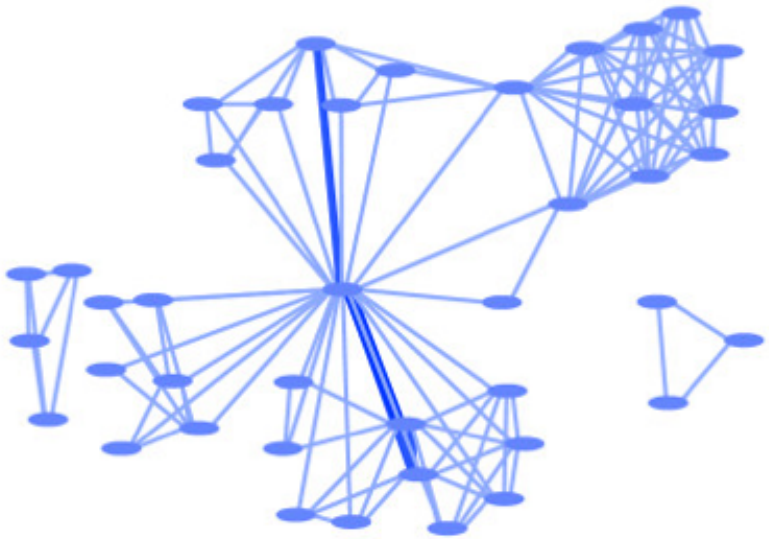
Preferential attachment: A process in which a new node entering the graph does NOT connect randomly but shows a preference to attach to a node of a certain type (Eg. higher degree).

- These kind of connections lead to the phenomenon 'Rich-gets-Richer'
- They show 'preferential attachment' creating 'hub' nodes and forming communities.

Idea: Preferential Attachment

Preferential attachment: A process in which a new node entering the graph does NOT connect randomly but shows a preference to attach to a node of a certain type (Eg. higher degree).

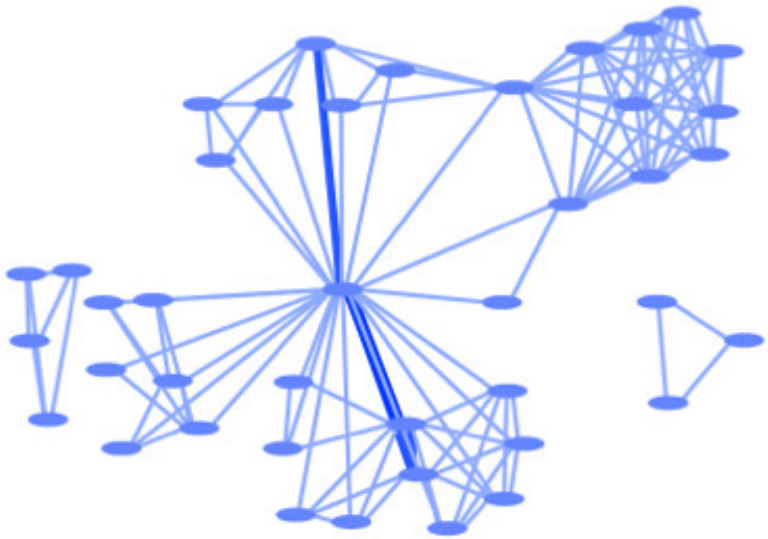
- These kind of connections lead to the phenomenon 'Rich-gets-Richer'
- They show 'preferential attachment' creating 'hub' nodes and forming communities.

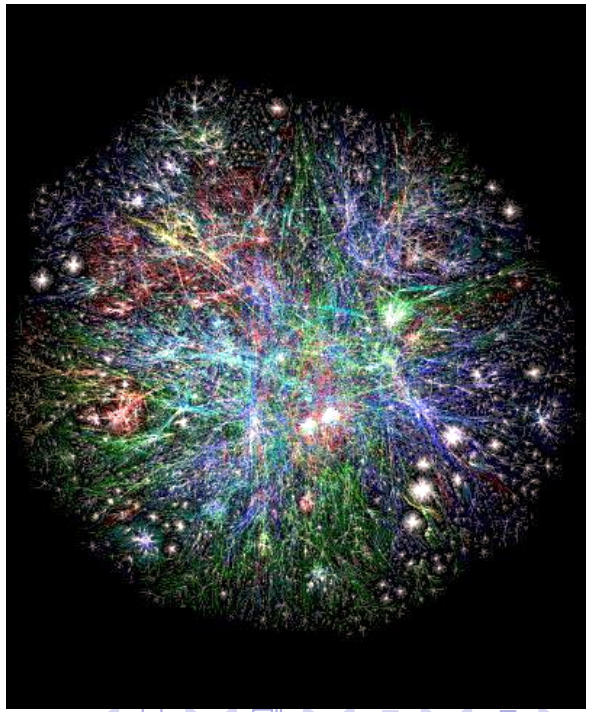


Idea: Preferential Attachment

Preferential attachment: A process in which a new node entering the graph does NOT connect randomly but shows a preference to attach to a node of a certain type (Eg. higher degree).

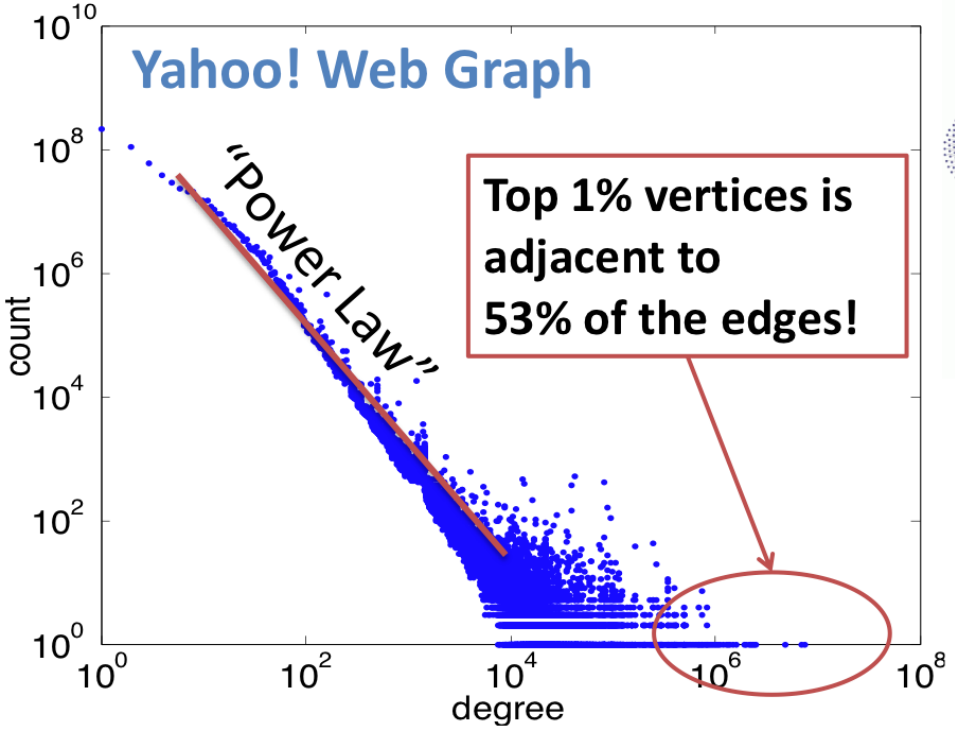
- These kind of connections lead to the phenomenon 'Rich-gets-Richer'
- They show 'preferential attachment' creating 'hub' nodes and forming communities.





The Power-law

- Natural Graphs:



Courtesy: Wikipedia

Real-world Networks

- The networks that occur in nature/real world are NOT Random networks.
- They show 'preferential attachment' creating 'hub' nodes and forming communities.
- Preferential attachment A process in which a new node entering the graph prefers to attach to a node of a certain type (Eg. higher degree).
- Barabasi-Albert proposed a network model that exhibits scale-free property:
  Degree distribution satisfies a Power Law
  $P(k) \sim k^{-\gamma}$ with exponent $2 < \gamma < \infty$.

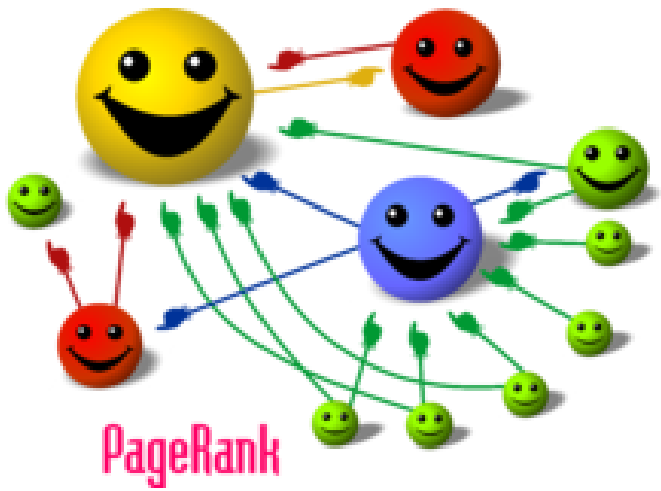
Summary Comparison

Type of Network	Degree Distribution	Average Path Length	Clustering Coefficient
Random Network	Poisson	Short	Low
Real-world	Power-law	Short	High
Regular	Uniform	Long	High

Now for some interesting problems on social networks!

Interesting Problems

Which are the 'important' nodes in a network?



Wikipedia: size of each face proportional to the total size of faces pointing to it

- PageRank or PR(A) can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web.

Web graph

WWW can be modeled as a graph in which each page on the internet is considered as a node and a page 1 is connected with a directed link to page 2 if there is a hyperlink in page 1 pointing to page 2.

Web graph

WWW can be modeled as a graph in which each page on the internet is considered as a node and a page 1 is connected with a directed link to page 2 if there is a hyperlink in page 1 pointing to page 2.

Jure Leskovec.

I am Professor of Computer Science at Stanford University.

My general research area is applied machine learning for large interconnected systems focusing on modeling complex, richly-labeled relational structures, graphs, and networks for systems at all scales, from interactions of proteins in a cell to interactions between humans in a society. Applications include commonsense reasoning, recommender systems, computational social science, and computational biology with an emphasis on drug discovery.

Home
Publications
Research group
Network data
Network software
Teaching
Bio
Press
Contact

Our group has several open research positions: **Undergraduate and Graduate as well as Postdoctoral**.

Follow @jure

What's new

- Organizing Stanford Graph Learning Workshop 2024. The workshop will bring together leaders from academia and industry to showcase recent advances in Machine Learning and AI. November 2024.
- We released PyTorch Frame: A PyTorch-based framework for deep learning over multi-modal tabular data. March 2024.
- We released RelBench: Relational Deep Learning Benchmark . An open benchmark for machine learning over relational databases. December 2023.
- We released PyG: The ultimate library for Graph Neural Networks. September 2021.
- Videos and slides from Stanford Graph Learning Workshop. Held at Stanford, September 2021.
- We released the Open Graph Benchmark---Large Scale Challenge and held KDD Cup 2021. Check the workshop slides and videos. August 2021.
- Tutorial on Meta-learning for Bridging Labeled and Unlabeled Data in Biomedicine. Held at ISMB 2021.
- Videos of my CS224W: Machine Learning with Graphs, which focuses on representation learning and graph neural networks. CS224W Syllabus.

Web graph

Stanford Graph Learning Workshop 2024

Stanford Data Science Affiliates Program

Register Now! Submit talk/poster!

The workshop will bring together leaders from academia and industry to showcase recent advances in Machine Learning and AI in Relational domains, Foundation Models, and Agents. The workshop will discuss methodological advancements, a wide range of applications to different domains, machine learning frameworks and practical challenges for large-scale training and deployment of AI models.

We are excited to announce a call for contributed talks and posters/demos at the upcoming Stanford Graph Learning Workshop. Submit your contribution!

Registration

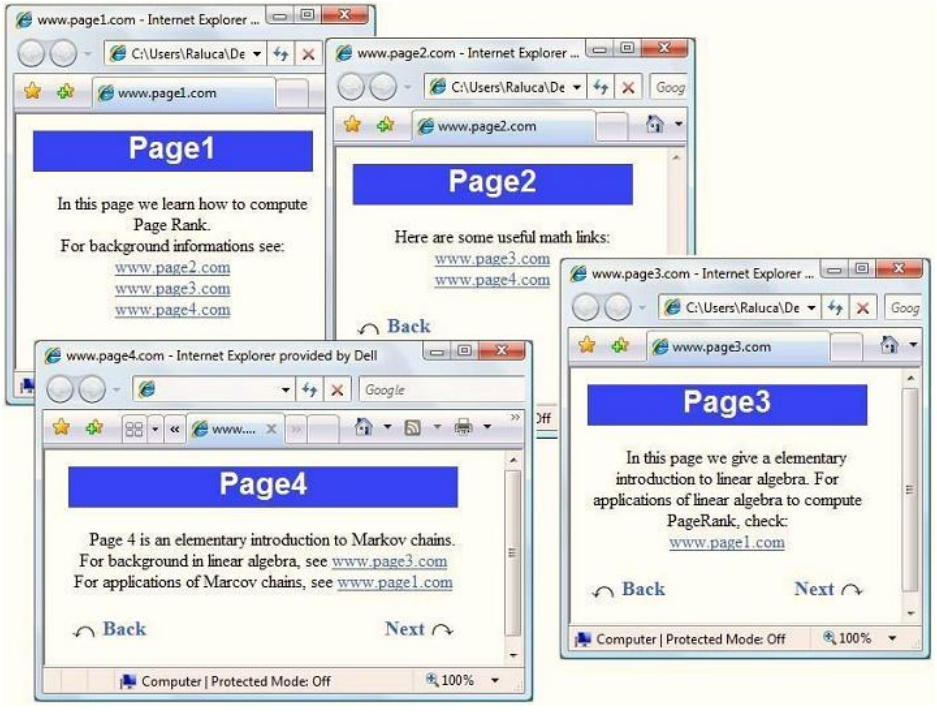
The Stanford Graph Learning Workshop will be held on Tuesday, Nov 5 2024, 09:00 - 18:00 Pacific Time.

The event will take place at Stanford University and will be live-streamed online. Free registrations are available. Register here.

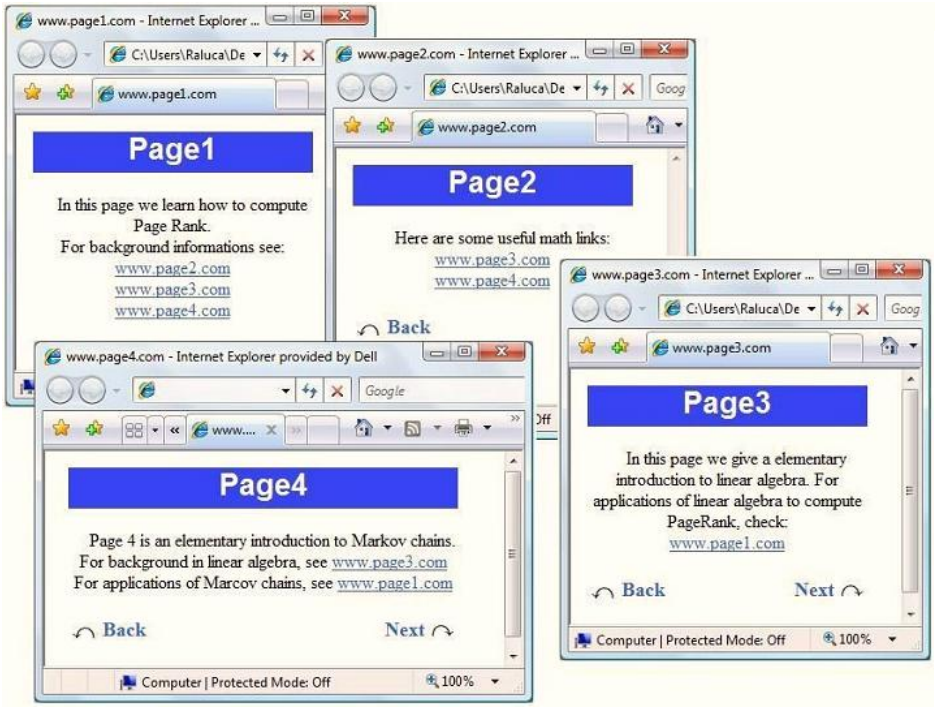
Schedule

08:00 - 09:00		Registration & Breakfast
09:00 - 09:10	Jure Leskovec, Stanford University	Welcome and Overview
09:10 - 09:30	Joshua Robinson, Isomorphic Labs & Stanford	Relational Deep Learning - Graph Representation Learning on Relational Databases
09:30 - 09:50	Matthias Fey & Akhiro Nitta, PyG & Kumo.AI	What's New in PyG
09:50 - 10:10	Rishabh Ranjan, Stanford University	RelBench: A Benchmark for Deep Learning on Relational

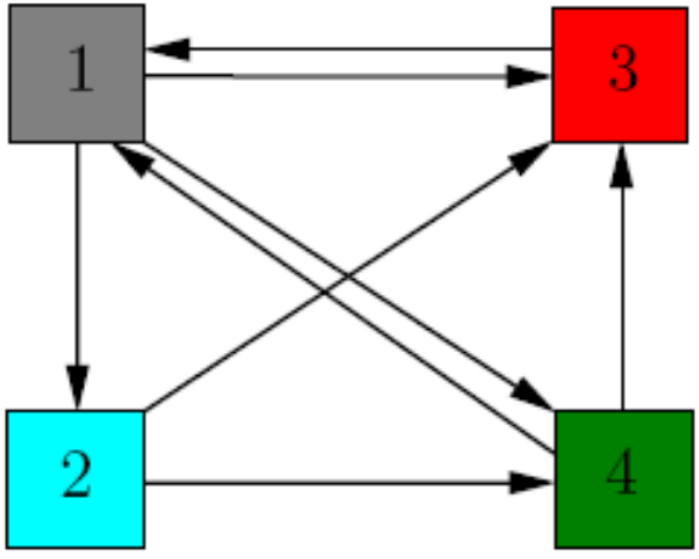
Web graph


www.page1.com - Internet Explorer ...
www.page1.com
Page1
In this page we learn how to compute Page Rank.
For background informations see:
www.page2.com
www.page3.com
www.page4.com
www.page2.com - Internet Explorer ...
www.page2.com
Page2
Here are some useful math links:
www.page3.com
www.page4.com
Back
www.page3.com - Internet Explorer ...
www.page3.com
Page3
In this page we give a elementary introduction to linear algebra. For applications of linear algebra to compute PageRank, check:
www.page1.com
Back
Next
Computer | Protected Mode: Off
100%
www.page4.com - Internet Explorer provided by Dell
Page4
Page 4 is an elementary introduction to Markov chains.
For background in linear algebra, see www.page3.com
For applications of Marcov chains, see www.page1.com
Back
Next
Computer | Protected Mode: Off
100%

Web graph

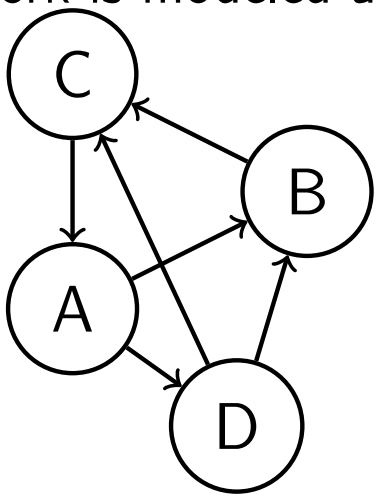


Courtesy: Slides from Prof. Remus, Cornell University



Graphs/ Networks as Matrices

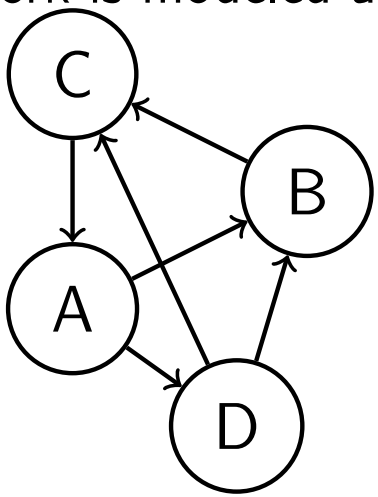
A network is modeled as a



graph.

Graphs/ Networks as Matrices

A network is modeled as a

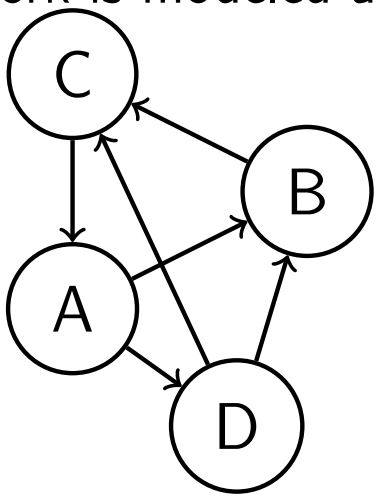


graph.

Form the adjacency matrix A of a graph.

Graphs/ Networks as Matrices

A network is modeled as a



graph.

Form the adjacency matrix A of a graph.

	A	B	C	D
A	0	1	0	1
B	0	0	1	0
C	1	0	0	0
D	0	1	1	0

Recall: Eigen Vector!

The eigen vector x of A satisfies the equation:

$$Ax = \lambda x$$

Whenever a vector x satisfies this equation, it is called the eigen vector corresponding to the eigen value λ.

Recall: Eigen Vector!

The eigen vector x of A satisfies the equation:

$$Ax = \lambda x$$

Whenever a vector x satisfies this equation, it is called the eigen vector corresponding to the eigen value λ. Note: Applying the linear transformation A to a vector x is equivalent to a scalar multiplication of x.

Recall: Eigen Vector!

The eigen vector x of A satisfies the equation:

$$Ax = \lambda x$$

Whenever a vector x satisfies this equation, it is called the eigen vector corresponding to the eigen value λ. Note: Applying the linear transformation A to a vector x is equivalent to a scalar multiplication of x.
The 'direction' of A is the same as that of vector x.

Application: Dimensionality reduction

Answer: Dimensionality reduction using Principal Component Analysis (PCA)

Application: Dimensionality reduction

Answer: Dimensionality reduction using Principal Component Analysis (PCA)

- The idea is to eliminate 'features' which are not contributing to the image in a significant way.
- Compute Principal Components: Find the dominant eigen vectors of the Covariance matrix $\Sigma_X = E[(X - \mu_X)(X - \mu_X)^T]$

Application: Dimensionality reduction

Answer: Dimensionality reduction using Principal Component Analysis (PCA)

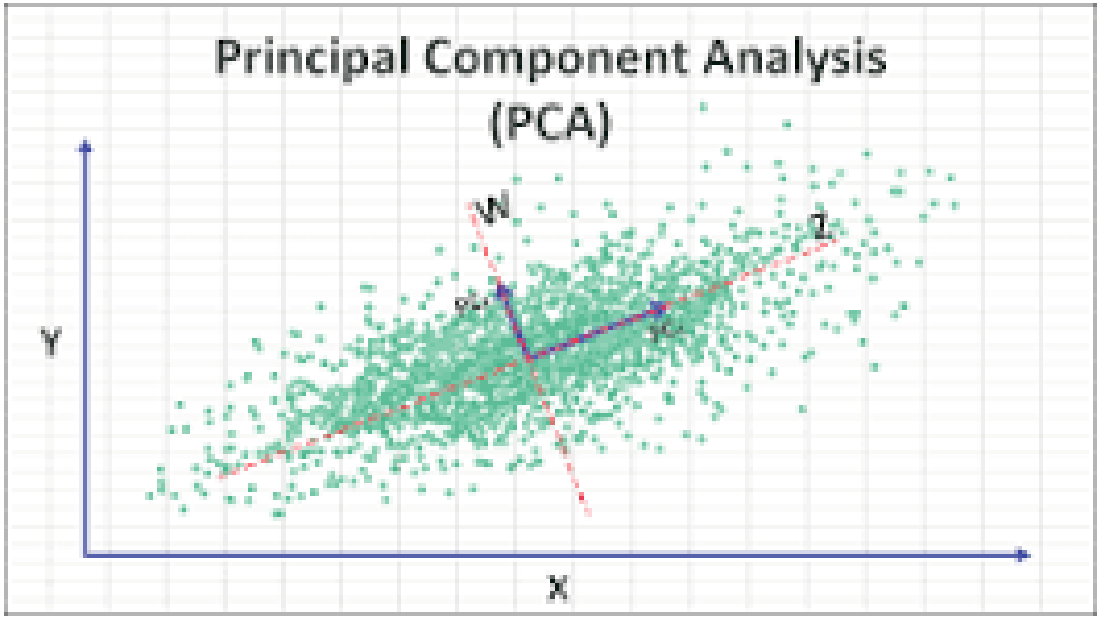
- The idea is to eliminate 'features' which are not contributing to the image in a significant way.
- Compute Principal Components: Find the dominant eigen vectors of the Covariance matrix $\Sigma_X = E[(X - \mu_X)(X - \mu_X)^T]$
- The eigen vector corresponding to the largest eigen value is called the principal component.

Application: Dimensionality reduction

Answer: Dimensionality reduction using Principal Component Analysis (PCA)

- The idea is to eliminate 'features' which are not contributing to the image in a significant way.
- Compute Principal Components: Find the dominant eigen vectors of the Covariance matrix $\Sigma_X = E[(X - \mu_X)(X - \mu_X)^T]$
- The eigen vector corresponding to the largest eigen value is called the principal component.
- The top eigen vectors derive the major 'directions' in which the image is concentrated.

Idea of PCA



Google Search Engine

How does google rank the results for our query?

- All the webpages are in a hyper-linked environment with each webpage pointing to other webpages.

Google Search Engine

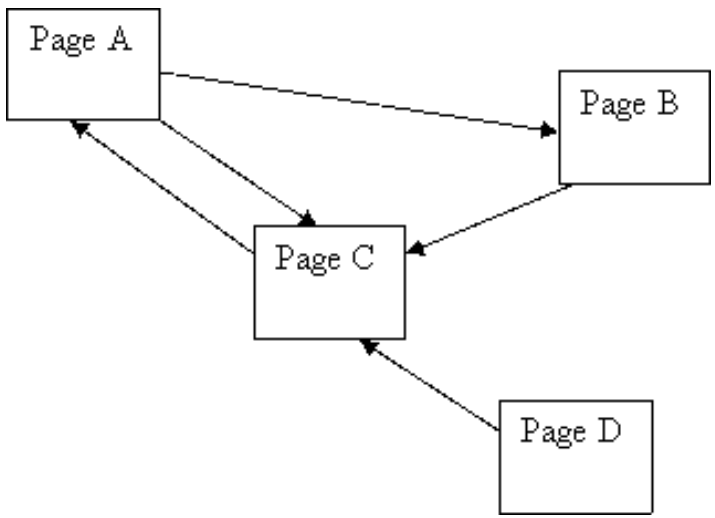
How does google rank the results for our query?

- All the webpages are in a hyper-linked environment with each webpage pointing to other webpages.
- This can be modeled as a network, with each webpage considered as a node. There is a directed edge between nodes A and B if A contains a hyperlink to page B.

Google Search Engine

How does google rank the results for our query?

- All the webpages are in a hyper-linked environment with each webpage pointing to other webpages.
- This can be modeled as a network, with each webpage considered as a node. There is a directed edge between nodes A and B if A contains a hyperlink to page B.



Google Search Engine

How does google rank the results for our query?

Figure: Larry Page and Sergei Brin

Idea! Start a Random walker traversing the network following the links! The more a node gets visited in the random walk, its importance grows more!

Power iteration method

- PageRank or r(A) can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web.
- PageRank extends this idea by not counting links from all pages equally, and by normalizing by the number of links on a page.

Recursive Updation

Given a web graph with N nodes, where the nodes are pages and edges are hyperlinks.

1. Assign each node an initial page rank $\frac{1}{N}$
2. Repeat until convergence $\sum_i |r_j(t+1) - r_j(t)| < \epsilon$
   - Calculate the page rank of each node as sum of the page ranks of the nodes that point to it

$$r_j(t+1) = \sum_{i \to j} \frac{r_i(t)}{d_i}$$

d_i is the outdegree of node i.

Recursive Updation

Given a web graph with N nodes, where the nodes are pages and edges are hyperlinks.

1. Assign each node an initial page rank $\frac{1}{N}$
2. Repeat until convergence $\sum_i |r_j(t+1) - r_j(t)| < \epsilon$
   - Calculate the page rank of each node as sum of the page ranks of the nodes that point to it

$$r_j(t+1) = \sum_{i \to j} \frac{r_i(t)}{d_i}$$

d_i is the outdegree of node i.

This idea is slightly modified to introduce randomness into the model.

Google Page Rank Algorithm

- The surfer randomly chooses one of its neighbours (hyperlinks) to visit with probability β

Google Page Rank Algorithm

- The surfer randomly chooses one of its neighbours (hyperlinks) to visit with probability β
- And with probability $(1 - \beta)$, jump to any node in the network.
- Idea : Recursive Updation of Ranks

Google Page Rank Algorithm

- The surfer randomly chooses one of its neighbours (hyperlinks) to visit with probability β
- And with probability $(1 - \beta)$, jump to any node in the network.
- Idea : Recursive Updation of Ranks
- Initialization: $r(j) = 1/N$ for all the nodes j in the network of size N.

Google Page Rank Algorithm

- The surfer randomly chooses one of its neighbours (hyperlinks) to visit with probability β
- And with probability $(1-\beta)$, jump to any node in the network.
- Idea : Recursive Updation of Ranks
- Initialization: $r(j) = 1/N$ for all the nodes j in the network of size N.

$$r(j) = \frac{1-\beta}{N} + \beta \left(\sum_{i \to j} \frac{r(i)}{d_i} \right)$$

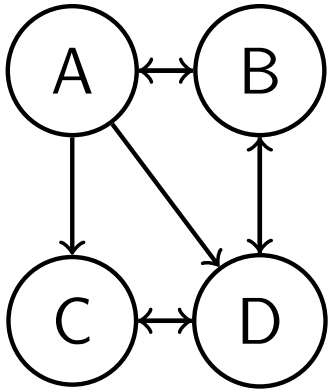
Google Page Rank Algorithm

- The surfer randomly chooses one of its neighbours (hyperlinks) to visit with probability β
- And with probability $(1-\beta)$, jump to any node in the network.
- Idea : Recursive Updation of Ranks
- Initialization: $r(j) = 1/N$ for all the nodes j in the network of size N.

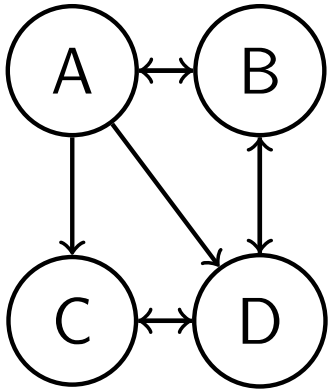
$$r(j) = \frac{1-\beta}{N} + \beta \left(\sum_{i \to j} \frac{r(i)}{d_i} \right)$$

- Solution of the equation corresponds to the principal eigenvector of the normalized link matrix of the web.

Example



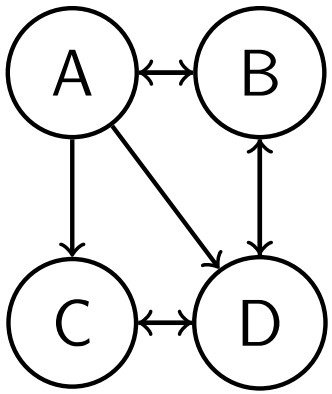
Example



Link Matrix L

A	0	1/3	1/3	1/3
B	1/2	0	0	1/2
C	0	0	0	1
D	0	1/2	1/2	0

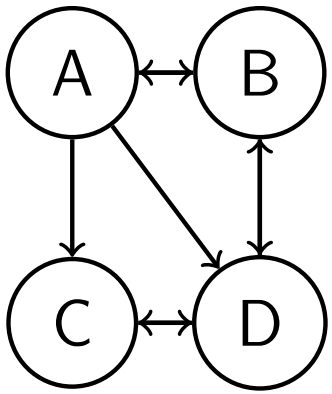
Example



Link Matrix L				
A	0	1/3	1/3	1/3
B	1/2	0	0	1/2
C	0	0	0	1
D	0	1/2	1/2	0

Using the update equation, the stochastic matrix is built.

Example

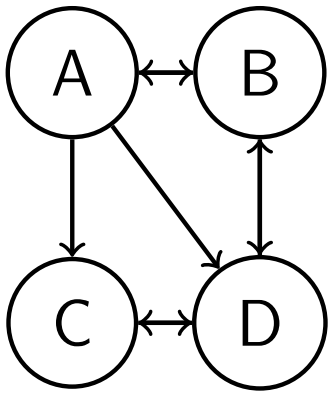


Link Matrix L

A	0	1/3	1/3	1/3
B	1/2	0	0	1/2
C	0	0	0	1
D	0	1/2	1/2	0

Using the update equation, the stochastic matrix is built. Find the eigen vector solution using an iterative method which converges.

Example

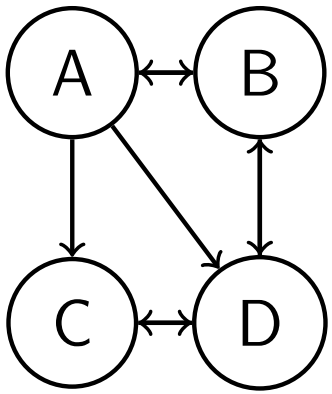


Link Matrix L				
A	0	1/3	1/3	1/3
B	1/2	0	0	1/2
C	0	0	0	1
D	0	1/2	1/2	0

Using the update equation, the stochastic matrix is built. Find the eigen vector solution using an iterative method which converges.

r(A)	r(B)	r(C)	r(D)
0.125	0.2083	0.2083	0.4583

Example



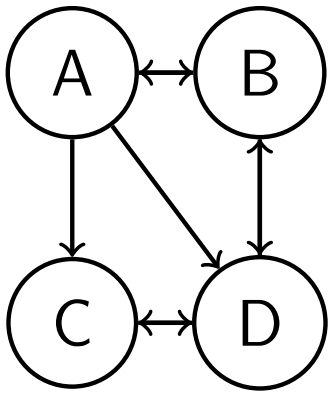
Link Matrix L

A	0	1/3	1/3	1/3
B	1/2	0	0	1/2
C	0	0	0	1
D	0	1/2	1/2	0

Using the update equation, the stochastic matrix is built. Find the eigen vector solution using an iterative method which converges.

r(A)	r(B)	r(C)	r(D)
0.125	0.2083	0.2083	0.4583
0.1354	0.2118	0.2118	0.4410

Example

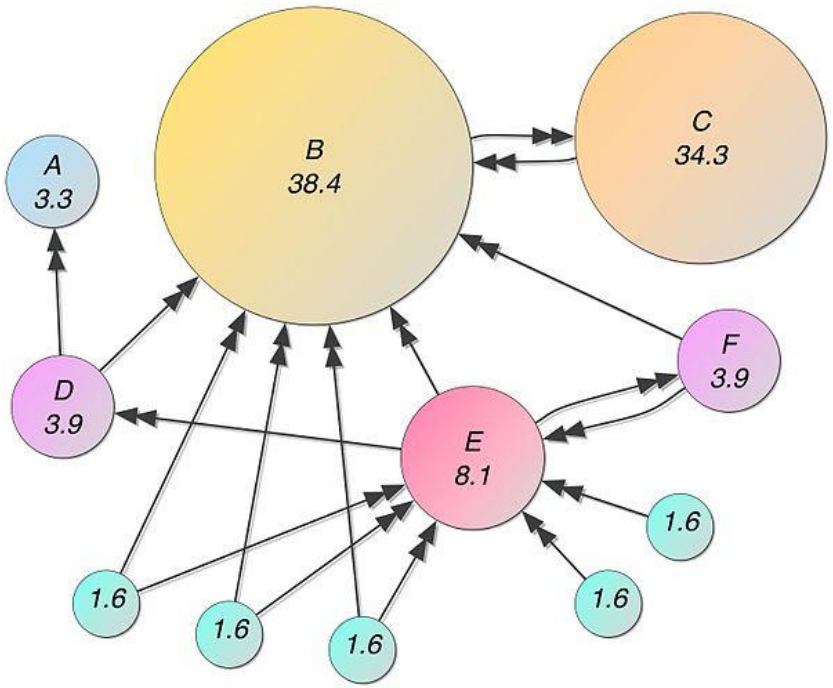


Link Matrix L				
A	0	1/3	1/3	1/3
B	1/2	0	0	1/2
C	0	0	0	1
D	0	1/2	1/2	0

Using the update equation, the stochastic matrix is built. Find the eigen vector solution using an iterative method which converges.

r(A)	r(B)	r(C)	r(D)
0.125	0.2083	0.2083	0.4583
0.1354	0.2118	0.2118	0.4410
0.1084	0.2611	0.2611	0.4410
..			..
0.1200	0.2400	0.2400	0.4200

Example from Wikipedia



Kleinberg's Hub and Authority Node detection

Kleinberg's HITS algorithm

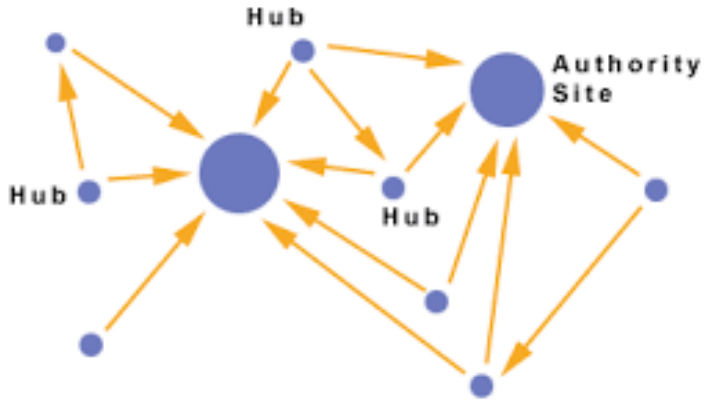
Jon Kleinberg's proposed HITS (Hyperlink Induced Topic Search) algorithm which assigns two scores to nodes in a directed graph.

- Authority node: one that has many in-links

Kleinberg's HITS algorithm

Jon Kleinberg's proposed HITS (Hyperlink Induced Topic Search) algorithm which assigns two scores to nodes in a directed graph.

- Authority node: one that has many in-links
- Hub node: one that has many out-links

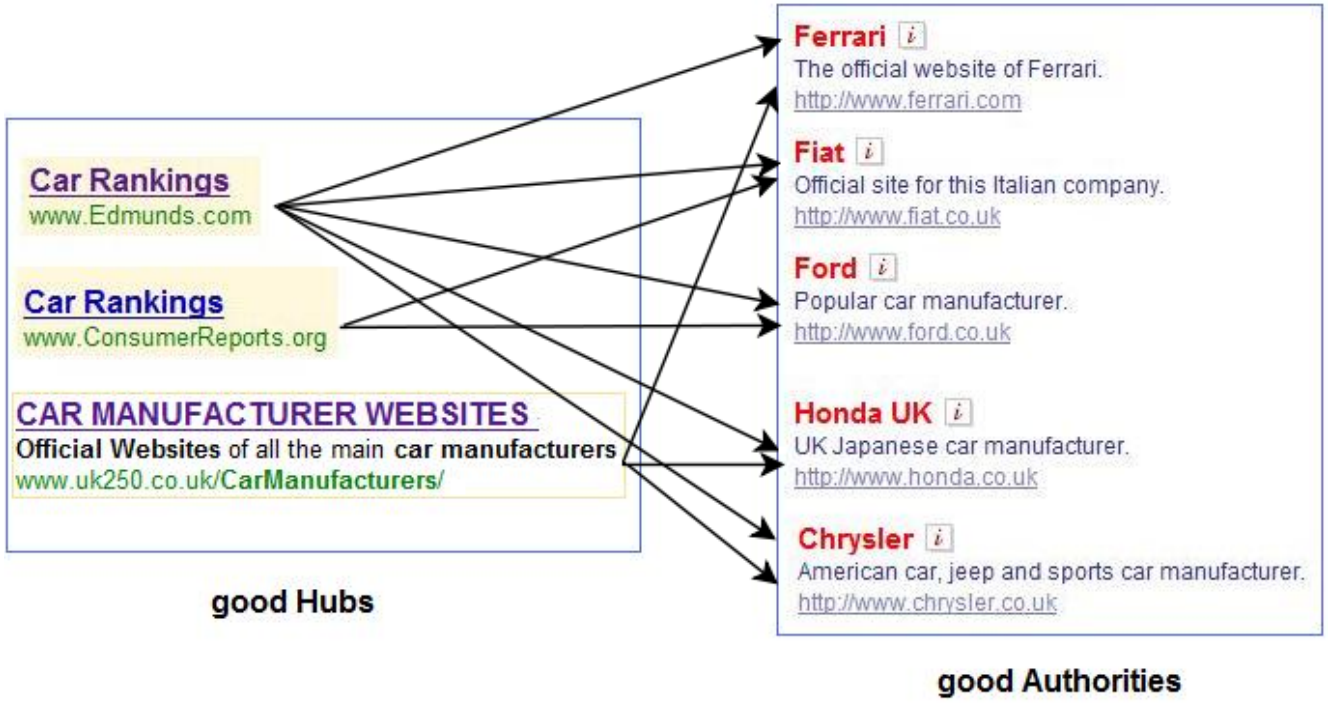


Hub and Authority nodes: Idea

- A good hub links to many good authorities
- A good authority is linked from many good hubs

Hub and Authority nodes: Idea

- A good hub links to many good authorities
- A good authority is linked from many good hubs



Acknowledgement: This picture is from Prof. Leskovec's slides.

Hub and Authority Scores

- These weights are defined recursively.
- A higher authority(hub) weight occurs if the page is pointed to by pages with high hub (authority) weights and vice-versa.

HITS Algorithm Each page j has 2 scores: authority score a_j and hub score h_j.

Initialize: $a_j(0) = 1/\sqrt{N}, h_j(0) = 1/\sqrt{N}$.

Repeat the following updation steps for all the nodes i until convergence:

1. $$a_j(t+1) = \sum_{i \to j} h_i(t)$$
2. $$h_j(t+1) = \sum_{j \to i} a_i(t)$$
3. Normalize the scores.

Recursive updates

The algorithm basically recursively updates the two operations

$$a(t+1) = (A^T.A).a(t)$$

Recursive updates

The algorithm basically recursively updates the two operations

$$a(t+1) = (A^T.A).a(t)$$

$$h(t+1) = (A.A^T).h(t)$$

with $a(0)$ and $h(0)$ initialized as unit vectors.

- Once again 'eigen vector' equations!!

Recursive updates

The algorithm basically recursively updates the two operations

$$a(t+1) = (A^T.A).a(t)$$

$$h(t+1) = (A.A^T).h(t)$$

with $a(0)$ and $h(0)$ initialized as unit vectors.

- Once again 'eigen vector' equations!!
- The authority weight vector a is the probabilistic eigenvector corresponding to the largest eigenvalue of $A^T A$.

Recursive updates

The algorithm basically recursively updates the two operations

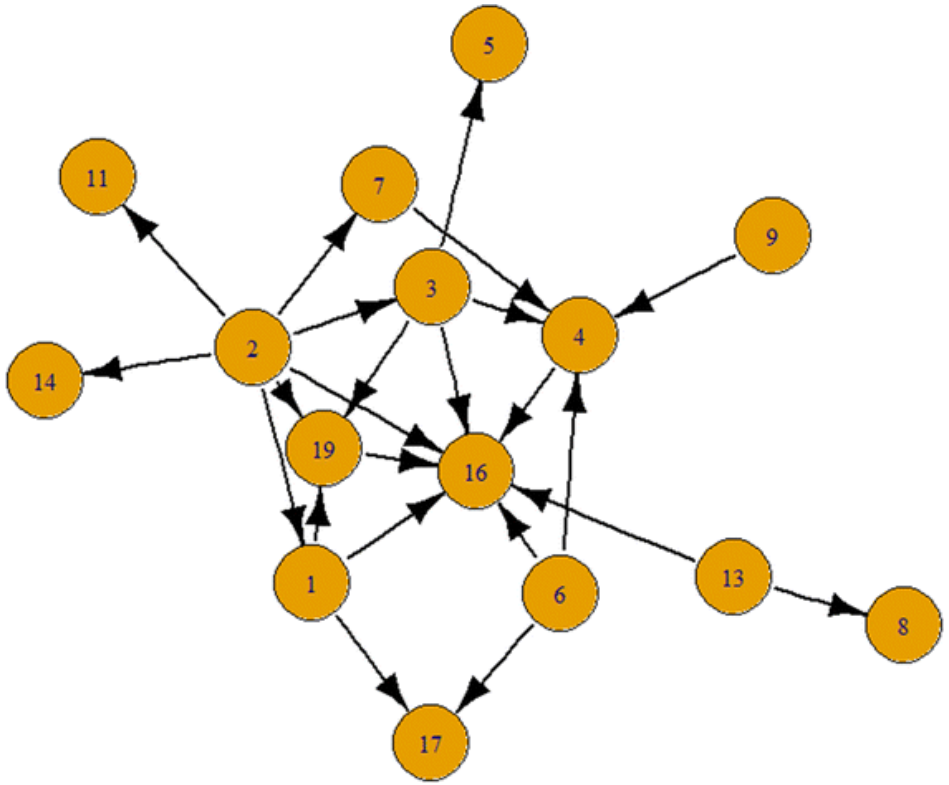
$$a(t+1) = (A^T.A).a(t)$$

$$h(t+1) = (A.A^T).h(t)$$

with $a(0)$ and $h(0)$ initialized as unit vectors.

- Once again 'eigen vector' equations!!
- The authority weight vector a is the probabilistic eigenvector corresponding to the largest eigenvalue of $A^T A$.
- The hub weights h of the nodes are given by the probabilistic eigenvector of the largest eigenvalue of AA^T.

Now we present some of our work extending HITS algorithm to citation networks.



Example: Citation Network

oogle Scholar

Jure Leskovec

FOLLOW

Professor of Computer Science, Stanford University
Verified email at cs.stanford.edu - Homepage

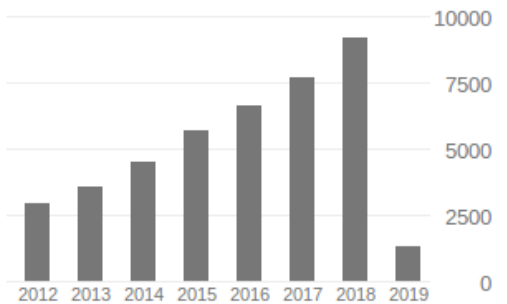
Data mining Social Network Analysis Information Networks

GET MY OWN PROFILE

TITLE	CITED BY	YEAR
Graphs over time: densification laws, shrinking diameters and possible explanations J Leskovec, J Kleinberg, C Faloutsos Proceedings of the eleventh ACM SIGKDD international conference on Knowledge …	2115	2005
The dynamics of viral marketing J Leskovec, LA Adamic, BA Huberman ACM Transactions on the Web (TWEB) 1 (1), 5	2109	2007
Friendship and mobility: user movement in location-based social networks E Cho, SA Myers, J Leskovec Proceedings of the 17th ACM SIGKDD international conference on Knowledge …	1959	2011
Graph evolution: Densification and shrinking diameters J Leskovec, J Kleinberg, C Faloutsos ACM Transactions on Knowledge Discovery from Data (TKDD) 1 (1), 2	1766	2007
Cost-effective outbreak detection in networks J Leskovec, A Krause, C Guestrin, C Faloutsos, J VanBriesen, N Glance Proceedings of the 13th ACM SIGKDD international conference on Knowledge …	1764	2007
Meme-tracking and the dynamics of the news cycle J Leskovec, L Backstrom, J Kleinberg Proceedings of the 15th ACM SIGKDD international conference on Knowledge …	1487	2009
{SNAP Datasets}:{Stanford} Large Network Dataset Collection	1392	2015

Cited by VIEW ALL

	All	Since 2014
Citations	47247	35371
h-index	87	81
i10-index	186	177



Example: Citations

About 3,560 results (**0.07** sec)

Vital nodes identification in complex networks — **[PDF]** sciencedirect.com
L Lü, D Chen, XL Ren, QM Zhang, YC Zhang, T Zhou - Physics reports, 2016 - Elsevier
Real networks exhibit heterogeneous nature with nodes playing far different roles in structure and function. To identify vital nodes is thus very significant, allowing us to control …
☆ Save 99 Cite Cited by 1419 Related articles All 6 versions Web of Science: 1114

Network representation learning: A survey — **[PDF]** arxiv.org
D Zhang, J Yin, X Zhu, C Zhang - IEEE transactions on Big Data, 2018 - ieeexplore.ieee.org
With the widespread use of information technologies, information networks are becoming increasingly popular to capture complex relationships across various disciplines, such as …
☆ Save 99 Cite Cited by 865 Related articles All 8 versions Web of Science: 468

Structural deep network embedding — **[PDF]** acm.org
D Wang, P Cui, W Zhu - Proceedings of the 22nd ACM SIGKDD …, 2016 - dl.acm.org
Network embedding is an important method to learn low-dimensional representations of vertexes in networks, aiming to capture and preserve the network structure. Almost all the …
☆ Save 99 Cite Cited by 3480 Related articles All 9 versions

[HTML] Graph embedding techniques, applications, and performance: A survey — **[HTML]** sciencedirect.com
P Goyal, E Ferrara - Knowledge-Based Systems, 2018 - Elsevier
Graphs, such as social networks, word co-occurrence networks, and communication networks, occur naturally in various real-world applications. Analyzing them yields insight …
☆ Save 99 Cite Cited by 2288 Related articles All 14 versions Web of Science: 1183

A survey on network embedding — **[PDF]** arxiv.org
P Cui, X Wang, J Pei, W Zhu - IEEE transactions on knowledge …, 2018 - ieeexplore.ieee.org

Interesting problem: Citation recommendation

- We now focus on the citation recommendation problem.
- We present two of our recent approaches to solve the problem:
  - Modified HITS algorithm
  - Graph neural networks (GNN) approach

1. Kammari, M., Bhavani, S.D. Citation recommendation using modified HITS algorithm, **Computing**, Volume 106, pages 2239–2259, 2024, Springer.
2. Monachary Kammari and Durga Bhavani S. Relevant article recommenda- tion by learning heterogeneous network embedding using GNN. In Proceed- ings of the 8th International Conference on Data Science and Management of Data **CODS-COMAD'24**, pages 201–209, New York, NY, USA, 2025. ACM.

Acknowledgement: **A big thanks to Monachary Kammari for sharing the following slides!**

Modeling Heterogeneous Network

Heterogeneous Network

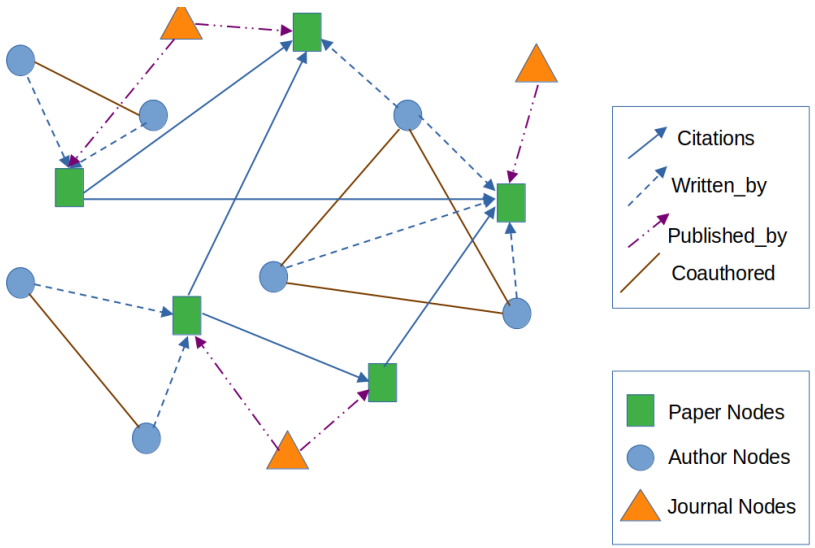
Given $G = (V, E)$ a heterogeneous network, where $V = \{V_1 \cup V_2 ... \cup V_l\}$ are set of different types of nodes and $E = \{E_1 \cup E_2 \cup E_3 ... \cup E_m\}$ are set of different types of edges.

Modeling Heterogeneous Network

Heterogeneous Network

Given $G = (V, E)$ a heterogeneous network, where $V = \{V_1 \cup V_2 ... \cup V_l\}$ are set of different types of nodes and $E = \{E_1 \cup E_2 \cup E_3 ... \cup E_m\}$ are set of different types of edges.

- There are different types of nodes and different types of interactions among these nodes.
- Model the heterogeneous network as defined below:



Example: Bibliographic Information Network

Let $G = (V, E)$ be a heterogeneous bibliographic network, where V is the set of nodes and E is the set of edges.

- Node set V consists of three types of nodes, namely papers, authors, and journals.

  $$V = \{V_p \cup V_a \cup V_j\}$$

  where V_p, V_a, and V_j are the sets of paper nodes, author nodes, and journal nodes respectively.

- Edge set E consists of four types of edges, namely cite, writes, coauthored, and published_in.

  $$E = \{E_{cit} \cup E_{aut} \cup E_{coa} \cup E_{pub}\}$$

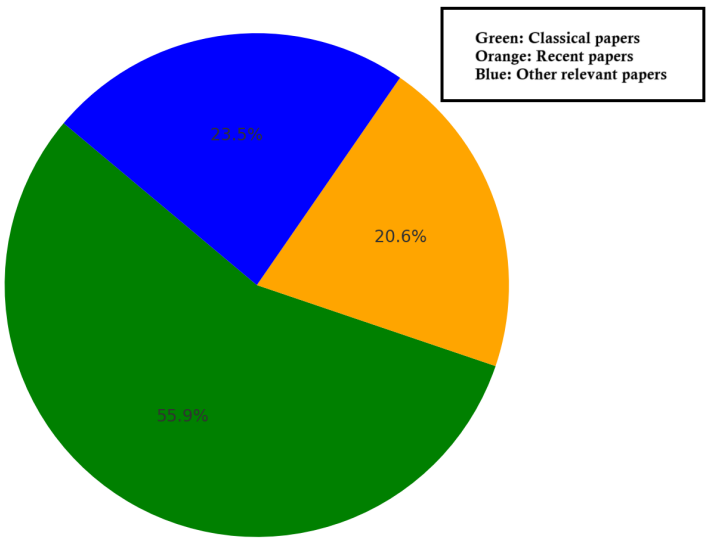
  where E_{cit} is a set of directed edges representing *citation* relationships; E_{aut} denotes *authorship* relations; E_{coa} is the set of *co-authorship* relations; and E_{pub} is the set of *published_in* relations.

Citation Recommendation Problem

Problem Statement

Given a problem query q (a set of keywords), and a set of seed papers B (possibly given by an expert), the citation recommendation problem is to predict the possible citations of the problem query q. That is, to predict the literature that is needed to be studied to address the problem q.

Motivation:



Idea for Modified HITS algorithm

- We propose node and edge weight measures based on the heterogeneous neighbourhoods.
- Initialize hub and authority scores as the proposed node heuristics.
- Run the HITS update equations using the node and edge weights.

Proposed Approach

- Assume that based on the research problem q, a set of seed papers B are given by an expert to the research student as a starting point to carry out the research.

Proposed Approach

- Assume that based on the research problem q, a set of seed papers B are given by an expert to the research student as a starting point to carry out the research.

CR-HITS Method

- **Step I: Pre-processing: Assignment of scores to the nodes and edges** Scores are assigned to different types of nodes based on the proposed node heuristics.
- **Step II: Extraction of the subnetwork** $G^{'}$ we extract the $k - hop$ neighborhood($k = 2$) from the network G around seed nodes B to form the heterogeneous subnetwork $G^{'}$.
- **Step III: Modified HITS algorithms** Apply proposed CR-HITS algorithm on sub-network $G^{'}$, which iteratively updates the hub and authority scores to converge.
- **Step IV: Return final recommendations** We recommend the top-k papers from the subnetwork $G^{'}$ based on the final hub scores given by the CR-HITS algorithm.

Extraction of the sub-network G' from base paper set

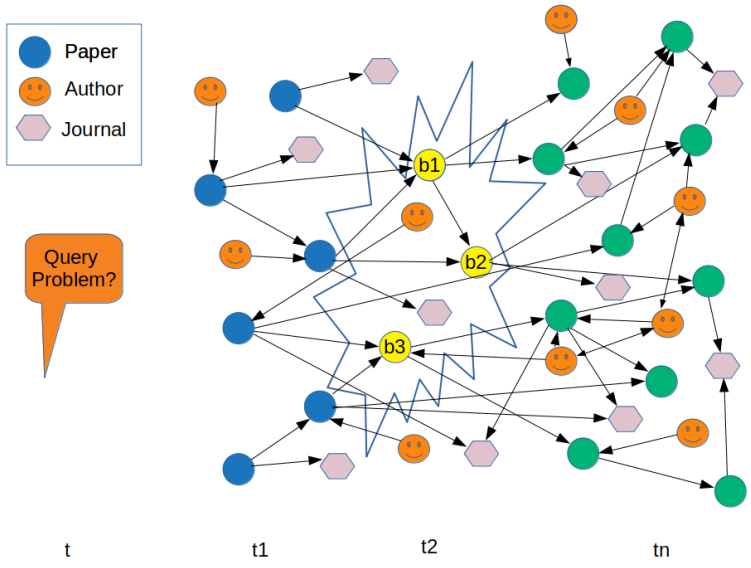


Figure: Extraction of Sub-network; b_1, b_2, b_3 are base papers

- Given a seed set, which consists of papers, authors, and venues, we construct a sub-network by adding the 2-hop neighborhood.
- Direction of the edges between the paper and author nodes will be a design decision and is chosen as the author to paper same for venues as well.

Scoring functions

1. Score for paper node: A paper i is given a high score if it acquires many citations and if it is published recently.

$$Score_P(i) = \log(\#citations(i)) * \beta^{(t-t_i)} \tag{1}$$

2. Score for author node: Author score $Score_A$ is simply the sum of the average number of citations of each paper over the time period of its publications.

$$Score_A(i) = \log\left(\frac{\sum_{l \in pubs(i)}\left(\frac{\#citations(l)}{(t-t_l)}\right)}{\#pubs(i)}\right) \tag{2}$$

where $pubs(i)$ is a set of papers written by author i, t is the current year, and t_l denotes the year of publication l.

3. Score for journal node: Score for a journal i is based on the scores $Score_P$ of the top α% of the papers (P_α) published in the journal i.

$$Score_J(i) = \frac{\sum_{l \in P_\alpha} Score_P(l)}{\mid P_\alpha \mid} \tag{3}$$

Modifed HITS algorithm

for i *in* G' **do**
  $auth[i] = score[i]$; // initialize the authority score
  $hub[i] = score[i]$; // initialize the hub score
  $auth_t[i] = score[i]$; // initialize the authority score
  $hub_t[i] = score[i]$; // initialize the hub score
end
$counter \leftarrow 0$; // initializing the counter
while $counter \leq max\_iterations$ **do**
  for i *in* G' **do**
    $auth[i] = \sum_{j \in N_{in}(i)} hub_t[j] * weight(j, i)$; // updating authority score
    $hub[i] = \sum_{j \in N_{out}(i)} auth_t[j] * weight(i, j)$; // updating hub score
  end
  for i *in* G' **do**
    $auth[i] = \frac{auth[i]}{max_{k \in V}(auth[k])}$; // Normalizing authority score
    $hub[i] = \frac{hub[i]}{max_{k \in V}(hub[k])}$; // Normalizing hub score
  end
  for i *in* G' **do**
    if $| hub[i] - hub_t[i] | \geq \gamma$ *or* $| auth[i] - auth_t[i] | \geq \gamma$ **then**
      go to next iteration ; // non convergence
    end
  end
  if *convergence or* $counter = max\_iterations$ **then**
    return $hub, auth$
  end
  for i *in* G' **do**
    $hub_t[i] = hub[i]$; // assigning updated scores
    $auth_t[i] = auth[i]$; // assigning updated scores
  end
  Return: top N **nodes from descending order of hub scores**

Experimental Setup

- We conduct experimentation on randomly selected papers from the network as the query paper P_T.
- For each query paper q from P_T, we remove q from the network and select three seed papers P_s from the bibliography R_q of paper q.
- We extract subnetwork $G^{'}$ from the heterogeneous bibliography network G as explained.
- Two variants of the proposed algorithm, CR-HITS (that uses both node scores and edge weights) and CR-NHITS (uses only node scores) algorithms are implemented on the subnetwork $G^{'}$.

Experimental Setup

- We conduct experimentation on randomly selected papers from the network as the query paper P_T.
- For each query paper q from P_T, we remove q from the network and select three seed papers P_s from the bibliography R_q of paper q.
- We extract subnetwork $G^{'}$ from the heterogeneous bibliography network G as explained.
- Two variants of the proposed algorithm, CR-HITS (that uses both node scores and edge weights) and CR-NHITS (uses only node scores) algorithms are implemented on the subnetwork $G^{'}$.
- The ACM (version-9) and DBLP(version-11) datasets are downloaded from aminer website [Tang et al., 2008].

S.No	Type	ACM dataset	DBLP dataset
1	#Paper nodes	528158	3501133
2	#Author nodes	21573	245204
3	#Journal nodes	787	16209
4	#Citation edges	2470021	25022314

Table: Details of the Datasets

Baseline Methods

- **Doc2vec** [Le and Mikolov, 2014] is a method that works on the representation learning technique applied to the content of the research papers.
- **Node2vec** [Han et al., 2017] is a semi-supervised algorithm to learn node representations of a network.
- **TriDNR** [Cheng et al., 2017] is a representation learning method using bibliographic network structure and paper vertex content to learn paper network representation.
- **BNR** [Cai et al., 2018] is a representation learning technique that utilizes network proximity and content of relevant articles to recommend related papers.
- **NNRank** [Bhagavatula et al., 2018] is a neural network model that uses candidate articles and query manuscripts to generate vector representations.
- **PR-HNE** [Ali et al., 2020] is a network representation learning model that jointly learns from six information networks by encoding them into latent space.
- **CR-HBNE** [Ali et al., 2022] is a heterogeneous network embedding model that learns semantic relations and contextual information

Metrics

Metrics

$$MAP = \frac{1}{\mid T_p \mid} \sum_{p_i \in T_p} \frac{1}{\mid R_{p_i} \mid} \sum_{p_k \in R_{p_i}, rank(p_k) \neq 0} \frac{q(p_k)+1}{q(p_k)}$$

$$MRR = \frac{1}{\mid T_p \mid} \sum_{p_i \in T_p} \frac{1}{rank(p_{first})}$$

$$r@N = \frac{1}{\mid T_p \mid} \sum_{p_i \in T_p} \frac{\mid R_{p_i} \cap B@N \mid}{|B@N|}$$

$$rank(p_k) = \begin{cases} \text{position of } p_k \text{ in } B@N & \text{if } p_k \in B@N \\ 0 & \text{if } p_k \notin B@N \end{cases}$$

$q(p_k)$ is #papers $\in B@N$ that rank higher than p_k. R is the bibliography, and $B@N$ denotes top-N recommendations.

Results

We randomly select 300 nodes as query papers and the average results are tabulated.

Dataset	Model	MAP	MRR	r@20	r@40	r@60	r@80	r@100
ACM	HITS	0.49	0.26	0.34	0.48	0.57	0.61	0.66
	CR-NHITS	0.59	0.30	0.42	0.59	0.68	0.70	0.72
	CR-HITS	**0.62**	**0.33**	**0.46**	**0.62**	**0.71**	**0.73**	**0.76**
DBLP	HITS	0.51	0.27	0.33	0.47	0.59	0.60	0.64
	CR-NHITS	0.63	0.32	0.43	0.62	0.72	0.74	0.77
	CR-HITS	**0.66**	**0.35**	**0.47**	**0.64**	**0.74**	**0.75**	**0.79**

Table: The values of Mean average precision(MAP), Mean reciprocal ranking(MRR), and recall@N(r@N) obtained for the proposed methods of CR-NHITS and CR-HITS are compared with the basic HITS algorithm.

Results

Model	MAP	MRR	r@20	r@40	r@60	r@80	r@100
CR-HITS	**0.66**	0.35	0.47	0.64	0.74	0.75	0.79
CR-NHITS	0.63	0.32	0.43	0.62	0.72	0.74	0.77
HITS	0.51	0.27	0.33	0.47	0.59	0.60	0.64
CR-HBNE	0.49	–	**0.63**	**0.72**	**0.78**	**0.83**	**0.85**
PR-HNE	0.56	**0.63**	0.58	0.65	0.72	0.76	0.80
NNRank	0.52	0.57	0.52	0.57	0.63	0.67	0.71
BNR	0.26	0.29	0.53	0.63	0.66	0.68	0.72
TriDNR	0.25	0.27	0.51	0.62	0.65	0.67	0.71
Doc2Vec	0.24	0.26	0.50	0.61	0.64	0.65	0.70
Node2Vec	0.23	0.25	0.49	0.60	0.63	0.64	0.69

Table: The values of Mean average precision(MAP), Mean reciprocal ranking(MRR), and recall@N(r@N) obtained for the proposed methods of CR-NHITS and CR-HITS are compared with the baseline methods.

Info used by the algorithms

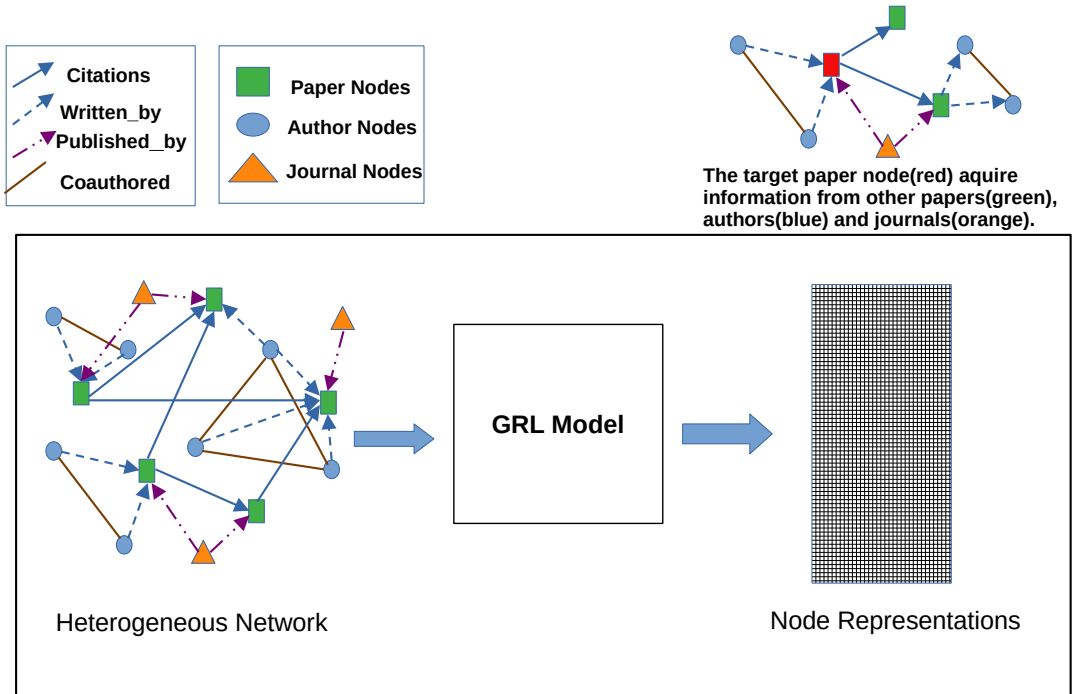
Model	Info. Used	Method
CR-HBNE [Ali et al., 2022]	Authors, Journals, Keywords Topics, Abstracts, Field of study, Title and Time period	Network embedding
PR-HNE [Ali et al., 2020]	Authors, Journals, Keywords Topics and Field of study	Network embedding
NNRank [Bhagavatula et al., 2018]	Authors, Journals, Document Topics and Keywords	Neural network model
CR-NHITS	Id's of Authors,Journals, Time Period	CR-NHITS algorithm
CR-HITS	Id's of Authors,Journals, Time Period	CR-HITS algorithm

Table: Additional information other than bibliography used by the models from the DBLP dataset

The Graph Neural Networks (GNN) approach to citation recommendation.

Problem Statement

Given a heterogeneous graph $G(V, E)$ where $V : \{v_1 \cup v_2 \cup \ldots \cup v_k\}$ consists different node types and edges $E : \{E_1 \cup E_2 \cup \ldots \cup E_l\}$ and $E_i \subseteq v \times v$ consists different edge types, the **Graph Representation Learning problem** is to learn a function $f : V \rightarrow \mathbf{R}^d$ (where $d << N$) mapping a node to its corresponding representation such that it preserves structural information.



Proposed Approach: CRGNN

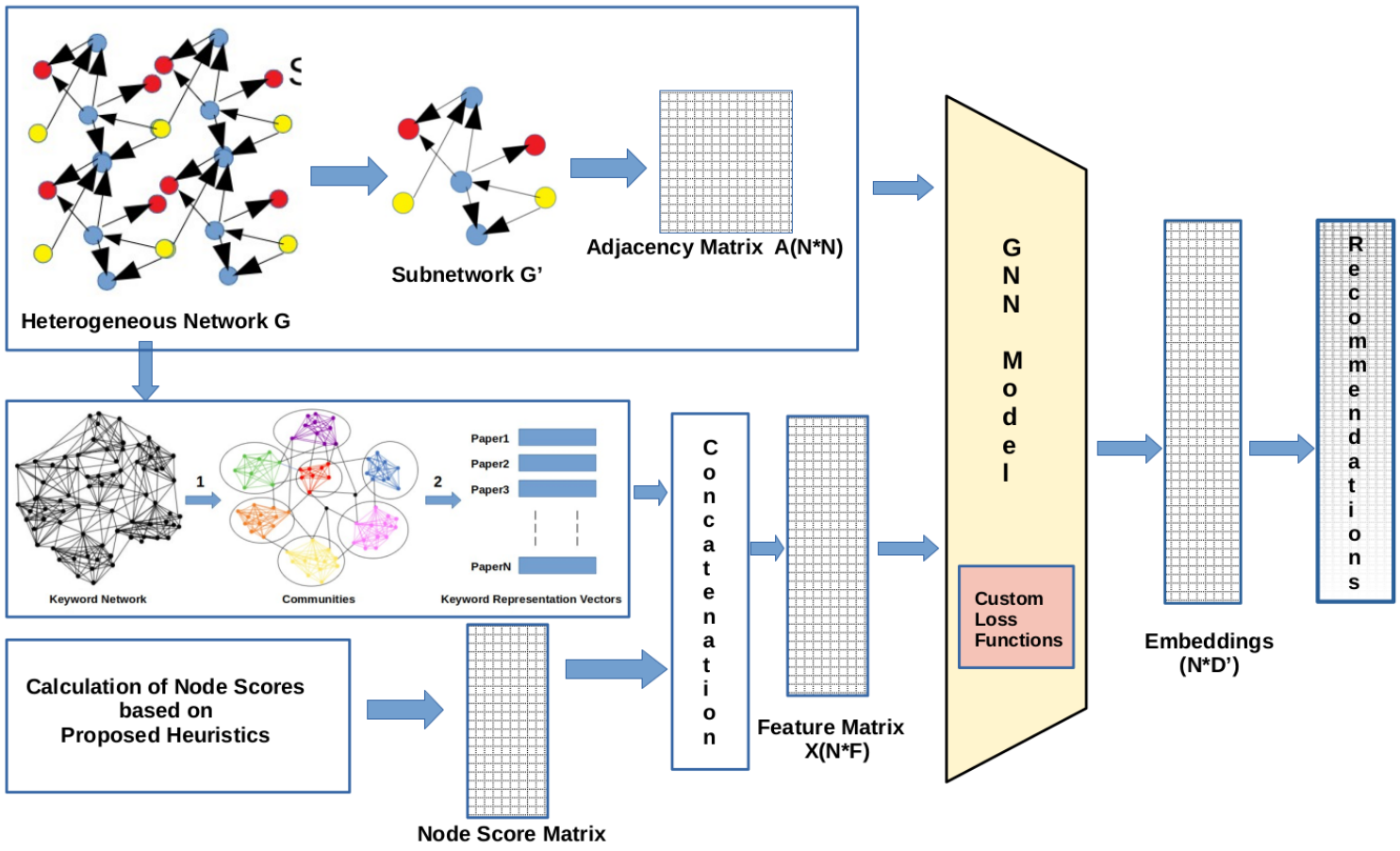


Figure: CRGNN model architecture

Feature Matrix

- The Feature matrix of the subgraph is formed by stacking the node features row-wise.
- For a paper p_a node features are its node score, age of the paper, author score, journal score, and keyword embedding.

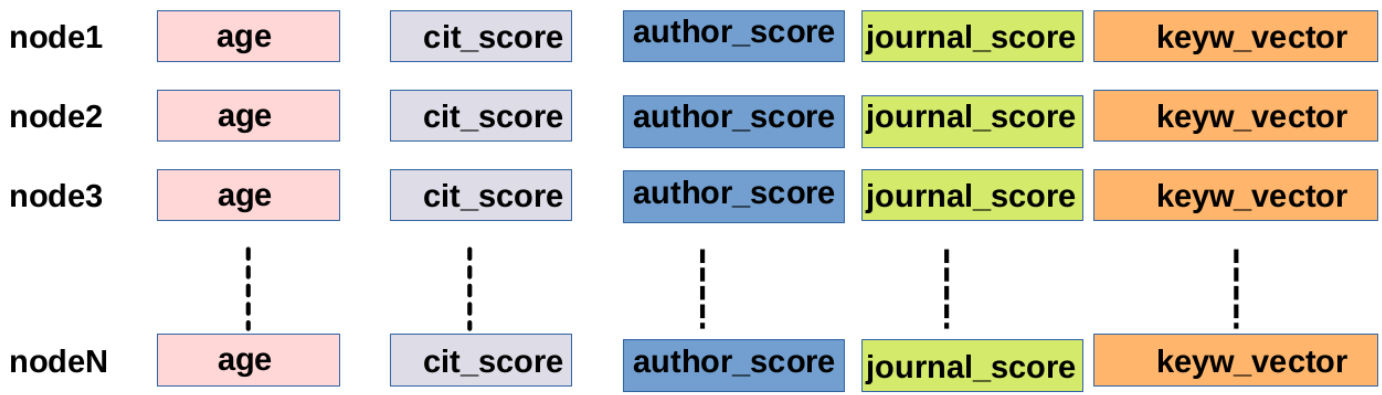


Figure: Feature Matrix

Learning Node Embeddings using GNN

- We have adapted three popular GNN models used for graph representation learning.
  - Graph Convolution Network(GCN)
  - Graph Attention Network(GAT)
  - Graph Autoencoder(GAE)

Learning Node Embeddings using GNN

- We have adapted three popular GNN models used for graph representation learning.
  - Graph Convolution Network(GCN)
  - Graph Attention Network(GAT)
  - Graph Autoencoder(GAE)
- We have **proposed a distance-based custom loss function** to learn effective node embeddings.

Distance-based custom loss function

The idea is to get each node embedding close to one of the seed node embeddings S. This results in gathering the nodes closer to one of the seed embeddings over the training period.

$$Loss = \sum_{i \in V} \min_{j \in S} \left(distance(i, j) \right) \tag{4}$$

where S is set of seed nodes and $distance(i, j)$ is the euclidean distance between embeddings of nodes i and j.

Final Recommendation

- The goal of the article recommendation problem is to recommend papers of high quality and relevance.
- For every paper p from g, the maximum of the cosine similarity values obtained with the seed set is considered as the rank score for the paper.
- Then the papers are sorted by their rank score, and top-k papers are recommended as relevant articles to the given query paper.

Experimental Setup

- We have done experiments on two real-world datasets DBLP V10, and DBLP V11 datasets.
- The DBLP stands for Digital Bibliography and Library Project, which contains papers from various domains such as computer science, mathematics, economics, etc.

Experimental Setup

- We have done experiments on two real-world datasets DBLP V10, and DBLP V11 datasets.
- The DBLP stands for Digital Bibliography and Library Project, which contains papers from various domains such as computer science, mathematics, economics, etc.
- Six variants of the proposed approach, namely,
  - Feature vectors without keyword representations
    1. CR-GCN
    2. CR-GAT
    3. CR-GAE
  - Feature vectors with keyword representations
    1. CR-GCNK
    2. CR-GATK
    3. CR-GAEK
- To begin with, we consider a set of 300 query papers randomly selected from the dataset for experimentation.
- For each query, three papers from its bibliography are selected as the seed set papers.

Experimental Setup

- We have done experiments on two real-world datasets DBLP V10, and DBLP V11 datasets.
- The DBLP stands for Digital Bibliography and Library Project, which contains papers from various domains such as computer science, mathematics, economics, etc.
- Six variants of the proposed approach, namely,
  - Feature vectors without keyword representations
    1. CR-GCN
    2. CR-GAT
    3. CR-GAE
  - Feature vectors with keyword representations
    1. CR-GCNK
    2. CR-GATK
    3. CR-GAEK
- To begin with, we consider a set of 300 query papers randomly selected from the dataset for experimentation.
- For each query, three papers from its bibliography are selected as the seed set papers.

Information	**DBLP V10**	**DBLP V11**
Papers	30,79,307	35,01,133

Parameter Settings & Baseline Methods

Model	I	H	O	E	γ
CR-GCN	4	10	8	400	0.02
CR-GCNK	10	10	8	400	0.02
CR-GAT	4	10	8	200	0.01
CR-GATK	10	10	8	200	0.01
CR-GAE	4	10	8	500	0.02
CR-GAEK	10	10	8	500	0.02

Table: Parameters of the proposed models; I is the input dimension, H is the hidden dimension, O is the output dimension, E is the number of epochs, γ is the learning rate.

Parameter Settings & Baseline Methods

Model	I	H	O	E	γ
CR-GCN	4	10	8	400	0.02
CR-GCNK	10	10	8	400	0.02
CR-GAT	4	10	8	200	0.01
CR-GATK	10	10	8	200	0.01
CR-GAE	4	10	8	500	0.02
CR-GAEK	10	10	8	500	0.02

Table: Parameters of the proposed models; I is the input dimension, H is the hidden dimension, O is the output dimension, E is the number of epochs, γ is the learning rate.

Analysis using MAP, MRR, R@N

Model	Info	MAP	MRR	R@20	R@40	R@60
CR-GCNK	1-4,8	0.59	0.31	0.44	0.56	0.59
CR-GATK	1-4,8	0.60	0.33	0.46	0.59	0.62
CR-GAEK	1-4,8	0.58	0.30	0.43	0.57	0.59
CR-HITS[Kammari and Bhavani, 2023]	1-3,8	**0.66**	0.35	0.47	0.65	0.74
CR-NHITS[Kammari and Bhavani, 2023]	1-3,8	0.63	0.32	0.43	0.62	0.72
HITS[Kleinberg, 1999]	1-3,8	0.51	0.27	0.33	0.47	0.59
CR-HBNE[Ali et al., 2022]	1-6,8	0.49	-	**0.63**	**0.72**	**0.78**
PR-HNE[Ali et al., 2020]	1-6,8	0.56	**0.63**	0.58	0.65	0.72
NNRank[Bhagavatula et al., 2018]	1-3,6-7	0.52	0.57	0.52	0.57	0.63
BNR[Cai et al., 2018]	1-5	0.26	0.29	0.53	0.63	0.66
Doc2Vec[Le and Mikolov, 2014]	1-3,7	0.24	0.26	0.50	0.61	0.64
Node2Vec[Grover and Leskovec, 2016]	1-3,5	0.23	0.25	0.49	0.60	0.63

Table: Results on the DBLP V11; Info is the information used{1. paper Id, 2. author Id's, 3. journal Id 4. keywords, 5. title, 6. abstract, 7. document, 8. year of publication}

Quality Metric

- The idea behind using these metrics is to match the original references of the query paper.
- We argue that matching the bibliography is not meaningful, as not every reference is topically relevant to the actual paper.
- Instead, we propose a quality metric *qs* as given in Equation 5 based on topical relevance, citations, and recency.

$$qs(p) = \alpha(js_{keyw}) + \beta(cs(p)) + \gamma(recency(p)) \tag{5}$$

Quality Metric

- The idea behind using these metrics is to match the original references of the query paper.
- We argue that matching the bibliography is not meaningful, as not every reference is topically relevant to the actual paper.
- Instead, we propose a quality metric *qs* as given in Equation 5 based on topical relevance, citations, and recency.

$$qs(p) = \alpha(js_{keyw}) + \beta(cs(p)) + \gamma(recency(p)) \tag{5}$$

Dataset	**DBLP V11 Dataset**				**DBLP V10 Dataset**			
Model	js_{keyw}	**age** ↓	**cs**	**qs**	js_{keyw}	**age**↓	**cs**	**qs**
CR-GCN	3.26	3.6	4.41	2.77	2.94	3.8	2.57	2.01
CR-GCNK	3.93	3.4	5.32	3.33	3.12	3.7	**2.72**	2.12
CR-GAT	3.94	3.6	5.13	3.26	3.08	3.4	2.79	2.14
CR-GATK	**4.02**	3.5	**5.37**	**3.37**	**3.26**	**3.1**	2.21	**2.31**
CR-GAE	3.2	3.7	4.31	2.71	2.91	3.5	2.62	2.02
CR-GAEK	3.4	**3.1**	4.94	3.02	3.01	3.2	2.71	2.09
CR-HITS[Kammari and Bhavani, 2023]	*2.98*	*9.8*	*1.64*	*1.65*	*2.32*	*8.3*	*2.24*	*1.63*
Original Refs	2.21	6.9	2.51	1.70	2.1	7.2	1.98	1.47

Table: Results in terms of quality metrics on the DBLP V11 and DBLP V10

Conclusion

- We have presented mainly two approaches to solve the article recommendation problem: Modified HITS algorithm and using Graph Neural Networks.
- In both the approaches, we proposed node and edge weights to incorporate the preferential mechanism while aggregating the information from the neighbourhood.
- We proposed a relevance metric and compared the results of the proposed models with baseline models.
- We extensively experimented with DBLP V10 and DBLP V11 datasets and compared the results with baseline methods.
- To the best of our knowledge, this is the first work that looks at the problem of article recommendation from the perspective of relevance.
- The results reflect that the proposed models outperformed SOTA methods regarding relevance.

References I

Ali, Z., Qi, G., Muhammad, K., Ali, B., and Abro, W. A. (2020).

Paper recommendation based on heterogeneous network embedding.
Knowledge-Based Systems, 210:106438.

Ali, Z., Qi, G., Muhammad, K., Bhattacharyya, S., Ullah, I., and Abro, W. (2022).

Citation recommendation employing heterogeneous bibliographic network embedding.
Neural Computing and Applications, 34(13):10229–10242.

Bhagavatula, C., Feldman, S., Power, R., and Ammar, W. (2018).

Content-based citation recommendation.
In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 238–251. Association for Computational Linguistics.

Cai, X., Zheng, Y., Yang, L., Dai, T., and Guo, L. (2018).

Bibliographic network representation based personalized citation recommendation.
IEEE Access, 7:457–467.

Cheng, G., Zhou, P., and Han, J. (2017).

Duplex metric learning for image set classification.
IEEE Transactions on Image Processing, 27(1):281–292.

Grover, A. and Leskovec, J. (2016).

node2vec: Scalable feature learning for networks.
In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864.

Han, J., Cheng, G., Li, Z., and Zhang, D. (2017).

A unified metric learning-based framework for co-saliency detection.
IEEE Transactions on Circuits and Systems for Video Technology, 28(10):2473–2483.

References II

Kammari, M. and Bhavani, S. D. (2023).

Citation recommendation using modified hits algorithm.
Computing, pages 1–21.

Kleinberg, J. M. (1999).

Hubs, authorities, and communities.
ACM computing surveys (CSUR), 31(4es):5–es.

Le, Q. and Mikolov, T. (2014).

Distributed representations of sentences and documents.
In *International conference on machine learning*, pages 1188–1196. PMLR.

Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., and Su, Z. (2008).

Arnetminer: extraction and mining of academic social networks.
In ACM, editor, *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, pages 990–998. ACM.