Fudan University

Social Network Mining

Final Project: Scholar Network Mining

Group Members:

Tianxiao Hu 14300240007

Haohan Zhang 14300240009

Hui Yang 14340240003

June 14, 2017

1 Project Overview

(1) **Datasets and Details**

1. **数据集**
DBLP（http://dblp.org/）是国际上最知名的收录计算机/IT 领域学术文献的网址，其收录的数据包括论文的标题、作者列表、发表时间、发表刊物/会议等信息。清华大学的学术搜索网站 ArnetMiner（https://cn.aminer.org/）基于 DBLP 和 ACM 等数据源收集了更丰富的文献数据，其中就有文献的引用数据集（https://cn.aminer.org/citation），其中包含了几十万篇计算机学术论文的相关数据，为 txt 格式，需自行解析。
2. **项目要求**
a）设计聚类算法或社区挖掘算法，对数据集中的所有论文进行聚类，用可视化工具展现出各簇/社区/领域（需注明相应的研究主题），并找出各领域中最有影响力的几个学者。
b）能对输入的任意一个学者，展现其自我中心网络（ego-network）的功能（参照 ArnetMiner 网站上的功能示例）。
c）加分项：利用 DBLP 和 ArnetMiner 提供的其它数据，对更多的学者间社会关系进行分析建模和预测，例如预测两个学者间的合作或引用关系，预测一个学者将来会在哪个刊物/会议上发表论文。

(2) **Group Members**

Tianxiao Hu	14300240007	Community Detection and Scholar Ranking
Haohan Zhang	14300240009	Ego-network Realization and Website Construction: Polar
Hui Yang	14340240003	Advisor-advisee Relationship Mining and Co-author Prediction

Tianxiao Hu:

- Used K Clique and Fast Unfolding to detect co-author communities[1].
- Adapted Number of Published Paper and Eigenvector Centrality to rank scholars.
- Made attempts to apply Word2vec model on DBLP dataset.

[1]https://github.com/TianxiaoHu/Polar

- Participated in making the slides used in presentation.

Haohan Zhang:

- Front-end: Designed and constructed Polar's website[2].
- Back-end: Realized back-end functions, provide datasets for the front-end.
- Human-computer Interaction: Realized the interaction and data transmission between back-end and front-end.

Hui Yang:

- Readings: Mining advisor-advisee relationships from research publication networks, written by Tang Jie, and reproduced the experiment using its model(TPFG).
- Read papers and wrote a review on algorithms about link predictions in co-authorship networks.

(3) **Data Collection and Preprocessing**
Firstly we got raw data from ArtnetMiner[3]. Afterwards we collected 7 fields of Computer Science and selected 3 or 4 top conferences for each field according to the University of Alberta's Computer Science Conference Rankings[4]. Detailed information about our dataset is shown in the table below.

Field	**Conferences**	**Papers**	**Authors**
Data Mining	ICDE, SIGMOD, KDD, ICDM	6985	11541
Distributed & Parallel Computing	PPOPP, PACT, IPDPS, ICPP	6568	11916
Computer Education	AIED, ITS, ICALT	4688	7958
Machine Learning	IJCAI, AAAI,ICML, NIPS	18628	21225
Networks, Communications & Performance	SIGCOMM, PERFORMANCE, INFOCOM, MOBICOM	8829	11954
Natural Language Processing	ACL, EACL, COLING, EMNLP	6885	7893
Operating Systems / Simulations	MASCOTS, SOSP, OSDI	1668	3649

Table 1: Dataset Details

[2]https://github.com/hh1680651/Social-Networking
[3]https://cn.aminer.org/citation
[4]http://webdocs.cs.ualberta.ca/ zaiane/htmldocs/ConfRanking.html

2 Co-author Community Detection

For Co-author Community Detection, we used papers in Data Mining, Distributed & Parallel Computing and Networks, Communications & Performance. We first calculated each author's papers in these three field and left out authors who has less than 10 papers. Finally we selected the field with the most papers as the author's field. The picture below shows an overview of authors' relationship.

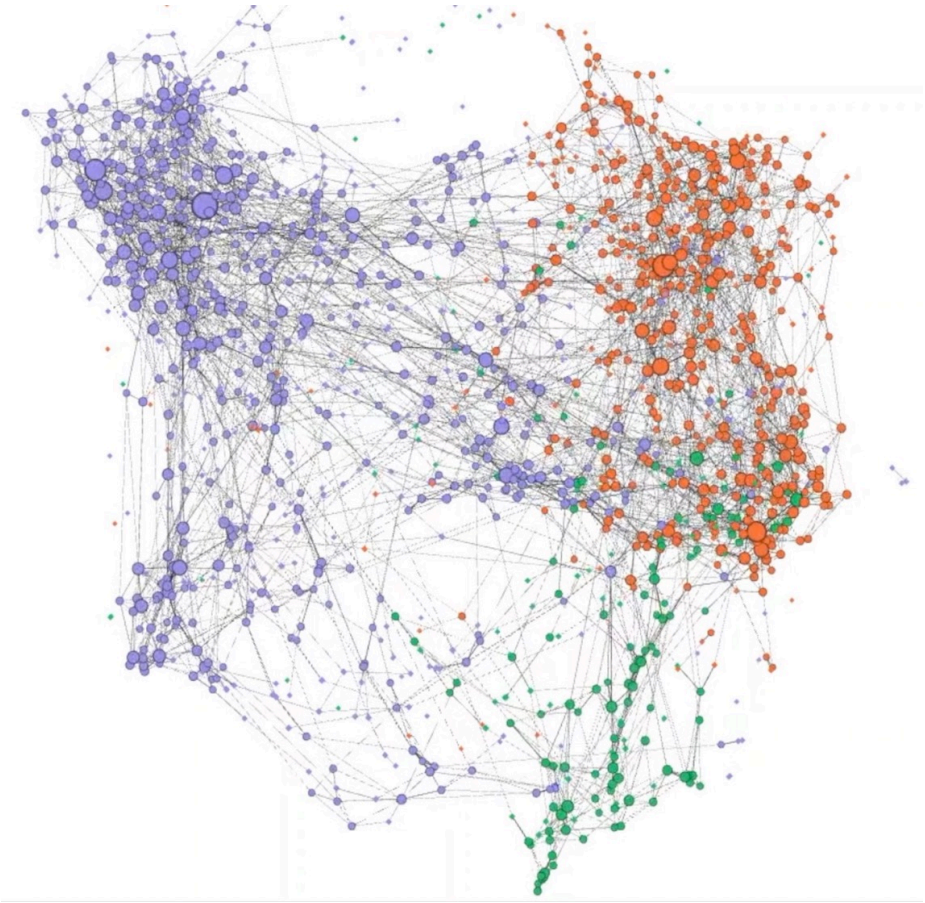


Figure 1: Overview of authors' relationship

Some scholars in Distributed & Parallel Computing and Networks and Communications & Performance collaborate many papers. There are also many scholars having published papers on conferences of both fields.

(1) **K Clique Algorithm**
K Clique Algorithm[1] can detect overlapping communities, which means a scholar can be detected into several co-author communities. After applying K Clique Algorithm on our dataset, 142 communities was found. However, we find only one community which has more than 30 numbers. A short statistic is shown below.

Community Larger than	**Amount**
30	1
20	3
10	10
5	39

Table 2: Overview of K Clique Algorithm

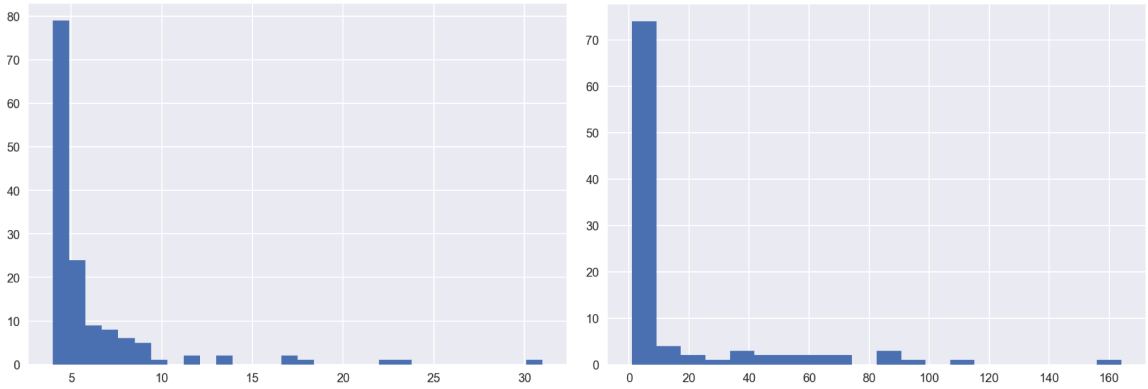
(2) **Fast Unfolding Algorithm**
Fast Unfolding Algorithm[2] is a heuristic method that is based on modularity optimization. After applying Fast Unfolding Algorithm on our dataset, 98 communities was found. A short statistic is shown below.

Community Larger than	**Amount**
100	2
70	6
50	12
30	18
10	23

Table 3: Overview of Fast Unfolding Algorithm

We find only two community which has more than 100 numbers. One in Machine Learning field and another in Networks, Communications & Performance.

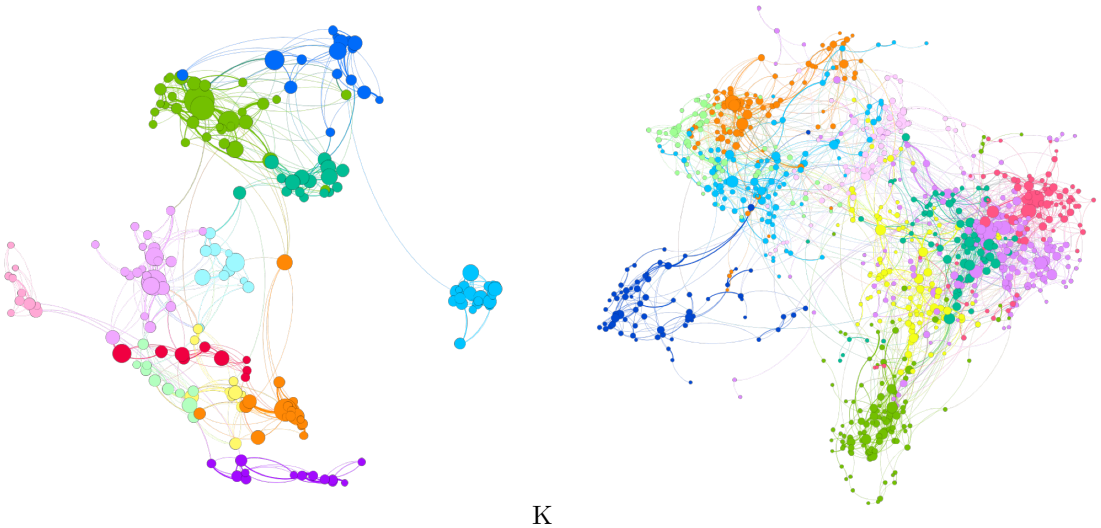
(3) **Comparison Between K Clique and Fast Unfolding**
K Clique Algorithm will detect more small-size communities while Fast Unfolding Algorithm less large-size communities. Communities distributions are shown below.



K Clique Fast Unfolding

Figure 2: Comparison of Community Distribution

We also visualized the results using Gephi about large communities detected in these two algorithm.



K

Clique Fast Unfolding

Figure 3: Comparison of Community Visualization

(4) **Attempts on Word2Vec Model**

We also adapted Word2Vec model[5] to co-author networks. We try to use word

[5]https://en.wikipedia.org/wiki/Word2vec

embedding on co-author network, gain a vector for each author and cluster them. However, we found the distribution messy and we couldn't gain convincing results from these high-dimension vectors.

(5) **Case Study: Michael I. Jordan**
We can find that in K Clique communities, authors are connected tighter. On the contrary, many small communities detected in K Clique Algorithm will be merged into a big community when using Fast Unfolding Algorithm. We will take Michael I. Jordan as an example.
Using K Clique Algorithm, Jordan is in several co-author communities. Three representatives are listed below.

Michael I. Jordan	**Michael I. Jordan**	**Michael I. Jordan**
Yann LeCun	**David M. Blei**	**Andrew Y. Ng**
Yoshua Bengio	Charles Kemp	Daphne Koller
Geoffrey E. Hinton	Naonori Ueda	Stuart J. Russell
Peter Dayan	Thomas L. Griffiths	Ronald Parr
...	...	...

Communities including Jordan: K Clique

However, Fast Unfolding Algorithm will detect a large community including all scholars above. The large community also includes other scholars(nearly all well-known scholars in Machine Learning field) such as Qiang Liu from Dartmouth College, Jun Zhu from Tsinghua University.

Michael I. Jordan
David M. Blei
Andrew Y. Ng
Yoshua Bengio
Geoffrey E. Hinton
Qiang Liu
Jun Zhu
Fei Sha
...

Community including Jordan: Fast Unfolding

3 Scholar Ranking

We use dataset of Machine Learning field to show how we rank scholars.

(1) **Published Paper**
We rank scholars by how many paper have been published. Michael I. Jordan has published 129 papers in top conferences and ranks first.

(2) **Eigenvector Centrality**
Eigenvector centrality (also called eigencentrality) is a measure of the influence of a node in a network. It assigns relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes.[6] The table below shows a contrast between two methods of ranking.

(3) **Pagerank**
PageRank (PR) is an algorithm used by Google Search to rank websites in their search engine results[7]. We also adapted Pagerank Algorithm to scholar network.

Published Paper	**Eigenvector Centrality**	**Pagerank**
Michael I. Jordan	Michael I. Jordan	Michael I. Jordan
Bernhard Schölkopf	Bernhard Schölkopf	Bernhard Schölkopf
Andrew Y. Ng	Tom M. Mitchell	Andrew Y. Ng
Zoubin Ghahramani	Andrew Y. Ng	Yoshua Bengio
Rong Jin	William W. Cohen	Qiang Yang
Geoffrey E. Hinton	Zoubin Ghahramani	Zoubin Ghahramani
Tuomas Sandholm	Partha Pratim Talukdar	Milind Tambe
Qiang Yang	Mehdi Samadi	Lawrence Carin
Yoshua Bengio	Abhinav Gupta	Satinder P.Singh
Lawrence Carin	Ni Lao	Eric P. Xing

Table 4: Three methods to rank Machine Learning scholars

[6]https://en.wikipedia.org/wiki/Centrality
[7]https://en.wikipedia.org/wiki/PageRank

4 Web-based Demo: Polar

(1) **Website for Display: Polar**
We developed a web-based app called Polar. Polar is the combination of **P**aper and Sch**olar**. This website is constructed for the presentation of scholar info, the overview of our datasets, also, the result of our network mining. This web is built with python Flask on the backend, html, bootstrap(css), echarts.js, D3.js on the front-end, self-wrote jQuery files on the human-computer interaction and front-back end communication.

(2) **Website Construction**
Front End
- css
  - bootstrap.min.css
  - sweetalert.css
- js
  - d3.min.js
  - echarts.min.js
  - jquery-3.1.1.min.js
- html

Back End:
- flask
- werkzeug

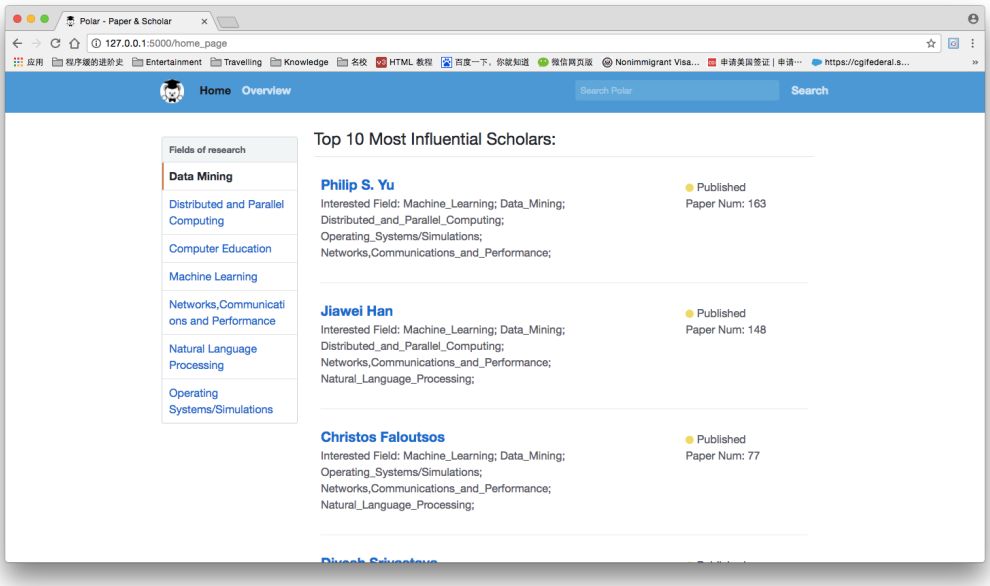
(3) **Website Functions**

(a) **View Top 10 influential scholars of each field of research (front page)** You can have a overview about the top 10 influential scholars of 7 fields we summarized on the front page and view their personal info via the href. For each field, you can see the content as follows:

- Title: author name - Link of title: Author's Personal Info Page.
- Sub title: Author's interested field.
- Right label: Number of published paper

(b) **Search for interested scholars or papers (navigator or search page)**

- The links would lead to search page includes: Navigator search block. Search section on search page.
- Input: Scholar name / paper name / key words ···



Front Page - Top 10 Most Influential Scholars of Each Field

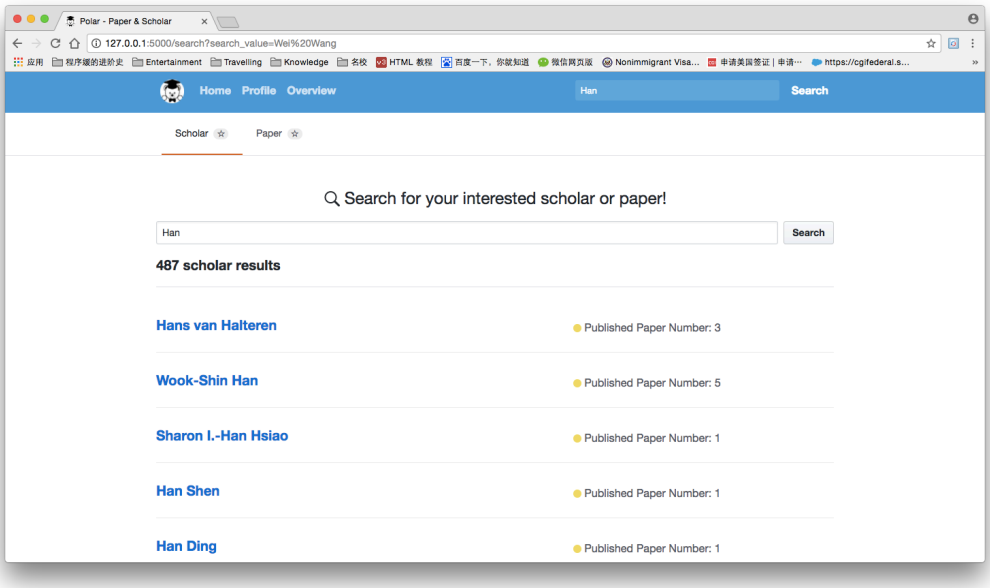
- Output: Search for scholars: List of scholars who's name contains the input value, with link to each Personal Info Page. Search for papers: List of papers who's title contains the input value.

(c) **Conference Publication Overview** On the page of Overview, you can see the statistical datas about publication numbers about each essential conferences for each field. When you move mouse on the conference name, you'll see the detailed datas. The Overview Table is presented as follow:

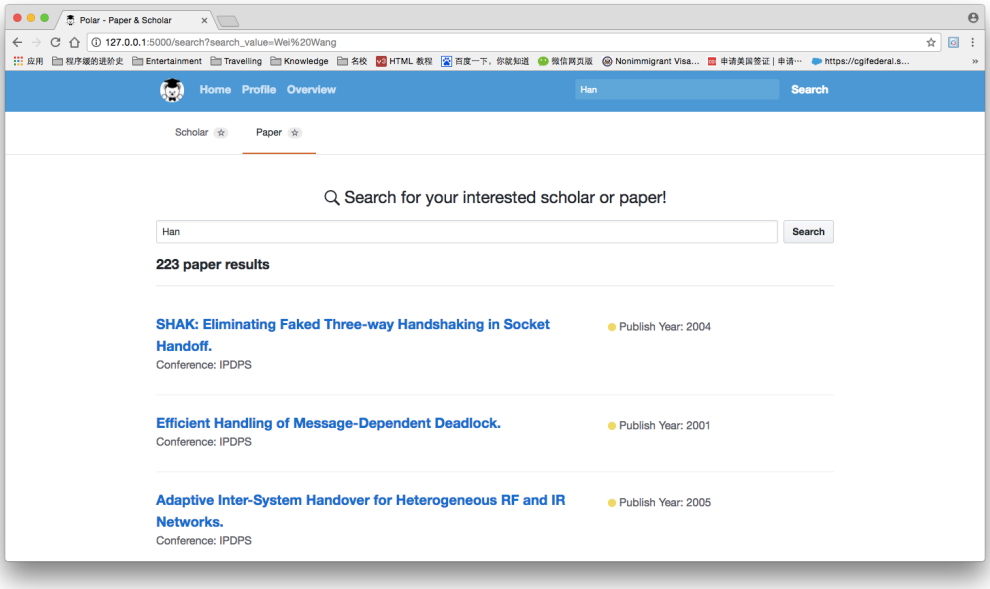
- X axis: year
- Y axis: Data: Conferences Color: Fields
- Data: Number of published data Color and circle size: growth as number increases.

(d) **View Personal Info (Profile Page)** On the profile page, you'll see three menu blocks on the left: Research Interests, which denoted the scholar's publication history; Ego Network: The succinct presentation about scholar's coauthors; Divergent Network: A divergent and colorful display with nodes of coauthors, and with links to their profile page respectively.

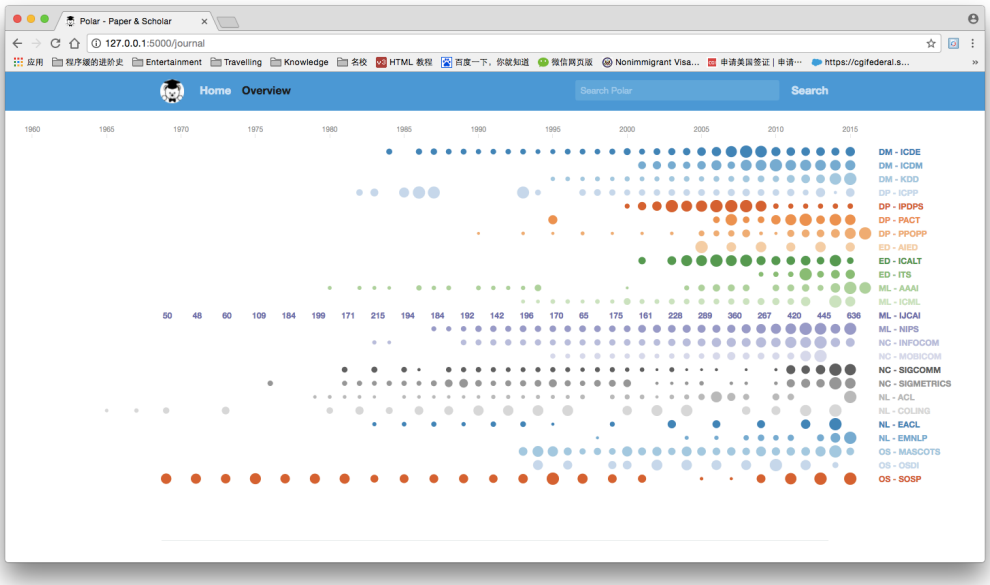
- Links which leads to this page: Front Page: Title of top 10 influential



Search Page - Search for Scholar



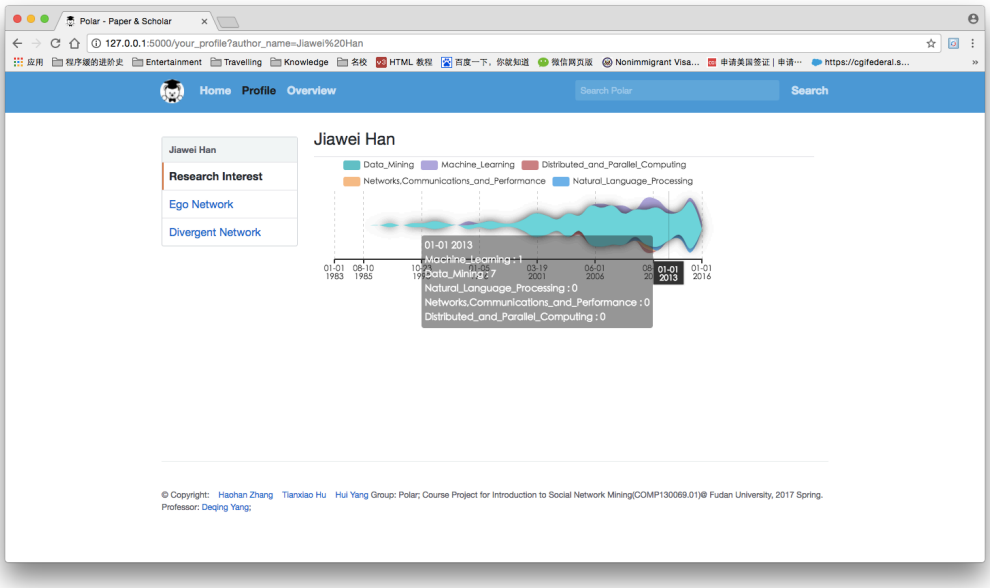
Search Page - Search for Paper



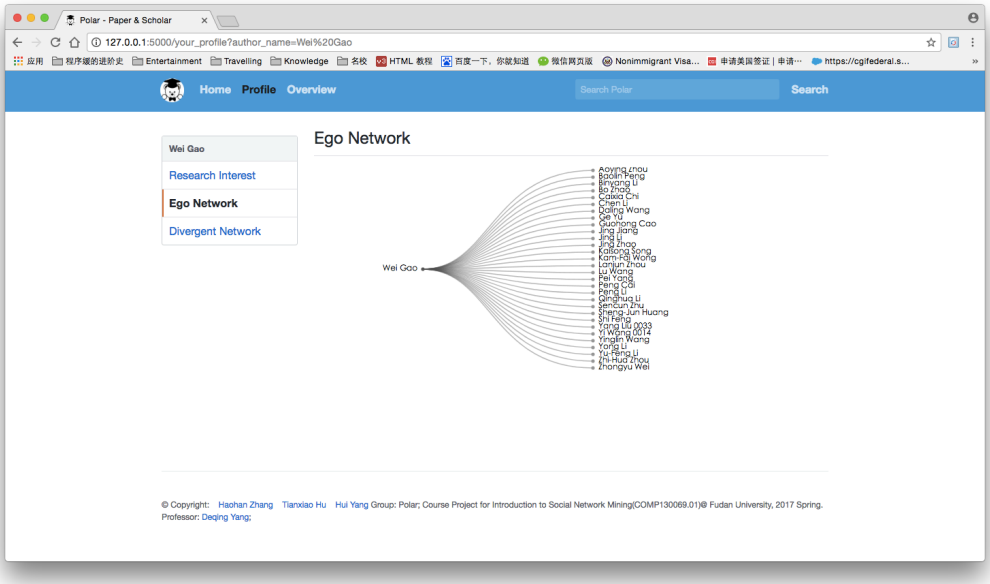
Overview Page - Publish Num of Conferences for Each Field

author of each field. Search Page: Title of scholar search results.

- **Interest Fields** In this section, you can see the publication number grouped by research field and presented with timeline. You can see one's specific field's details via clicking off the legend of the other fields. Or you can see the details via tooltip, which would be triggered by moving mouse on the dynamic graph.
  - Input: Author Name
  - Output: A dynamic river-flow graph, whit X axis denotes time, Y axis denotes num of papers published of a specific research field. Legend the data with field names.
  - Applying tool: Echarts
- **Ego Network** In this section, you'll see an explicit and succinct display about the author's neighbors: those who worked as coauthors with him/her.
  - Input: Author Name
  - Output: A network with a single node on left part (this.author), several nodes on the right part (this.author.coauthors).
  - Applying tool D3

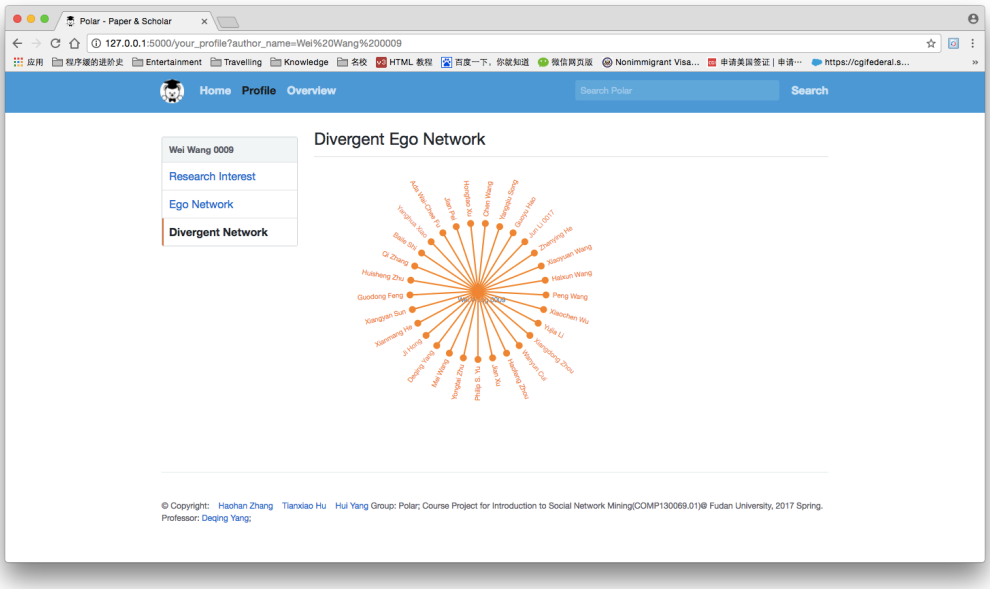


Profile Page - Publish History of Jiawei Han



Profile Page - Succinct Ego Network of Wei Gao

- **Divergent Ego Network** In this section, you'll see an divergent and colorful display about the author's neighbors: those who worked as coauthors with him/her. And you can click on the nodes of coauthors, it would lead you to their personal info page.
  - Input: Author Name
  - Output: A network with a single node on its center (this.author), several nodes on the ridge (this.author.coauthors), the nodes on the outside layer (coauthors) have href which links to the coauthor's Personal Info Page
  - Applying tool: D3



Profile Page - Divergent Ego Network of Prof. Wei Wang

5 Advisor-advisee Relationship Mining

Information network contains abundant knowledge which is often hidden. For example, in an author-paper network, the advisor-advisee relationship is hidden in the coauthor relations. If we can discover these hidden relationships, it will be beneficial to some interesting applications such as expert recommending.
Tang Jie[3] has put forward a efficient method for mining advisor-advisee relationship in large scholar network. We reproduce the experiment using TPFG model. The algorithm includes two stages.

(1) **Preprocessing**
The purpose of preprocessing is to generate the candidate graph H' and reduce the search space by removing the links which are not advisor-advisee relationships. The process can be seen in figure below.

First, an author-paper heterogeneous network is built as an input. Then this network should be transformed into a homogeneous network containing only by authors. This homogeneous network includes a virtual node a_0, which will be the root of an advising tree. Each edge connects authors a_i and a_j if and only if they have publications together, and there are two vectors associated with the edge, PubYearVector py_{ij} and PubNumVector pn_{ij} indicating the year they have publications and the number of coauthored papers they have at that time.
We denote the author a_i's advisor as a_{y_i}, where y_i is a hidden variable. And if a_i's advisor is a_j, we use $[st_{ij}, ed_{ij}]$ to represent the time interval advising relationship lasts. Then we need to construct a sub-graph by removing some edges and make the remaining edges directed from advisee to potential advisor. We need to create a potential link from a_i to a_j if a_j has a longer publication history than a_i and compute Kulczynski and Imbalance Ratio measure for the coauthored publications at different time t. Kulczynski and Imbalance are defined as:

$$kulc_{ij}^t = \frac{leq \sum_{py_{ij}^k \le t} pn_{ij}^k}{2} \left(\frac{1}{\sum_{py_i^k \le t} pn_i^k} + \frac{1}{\sum_{py_j^k \le t} pn_j^k}\right)$$

$$IR_{ij}^t = \frac{\sum_{py_j^k \le t} pn_j^k - \sum_{py_i^k \le t} pn_i^k}{\sum_{py_i^k \le t} pn_i^k + \sum_{py_j^k \le t} pn_j^k - \sum_{py_{ij}^k \le t} pn_{ij}^k}$$

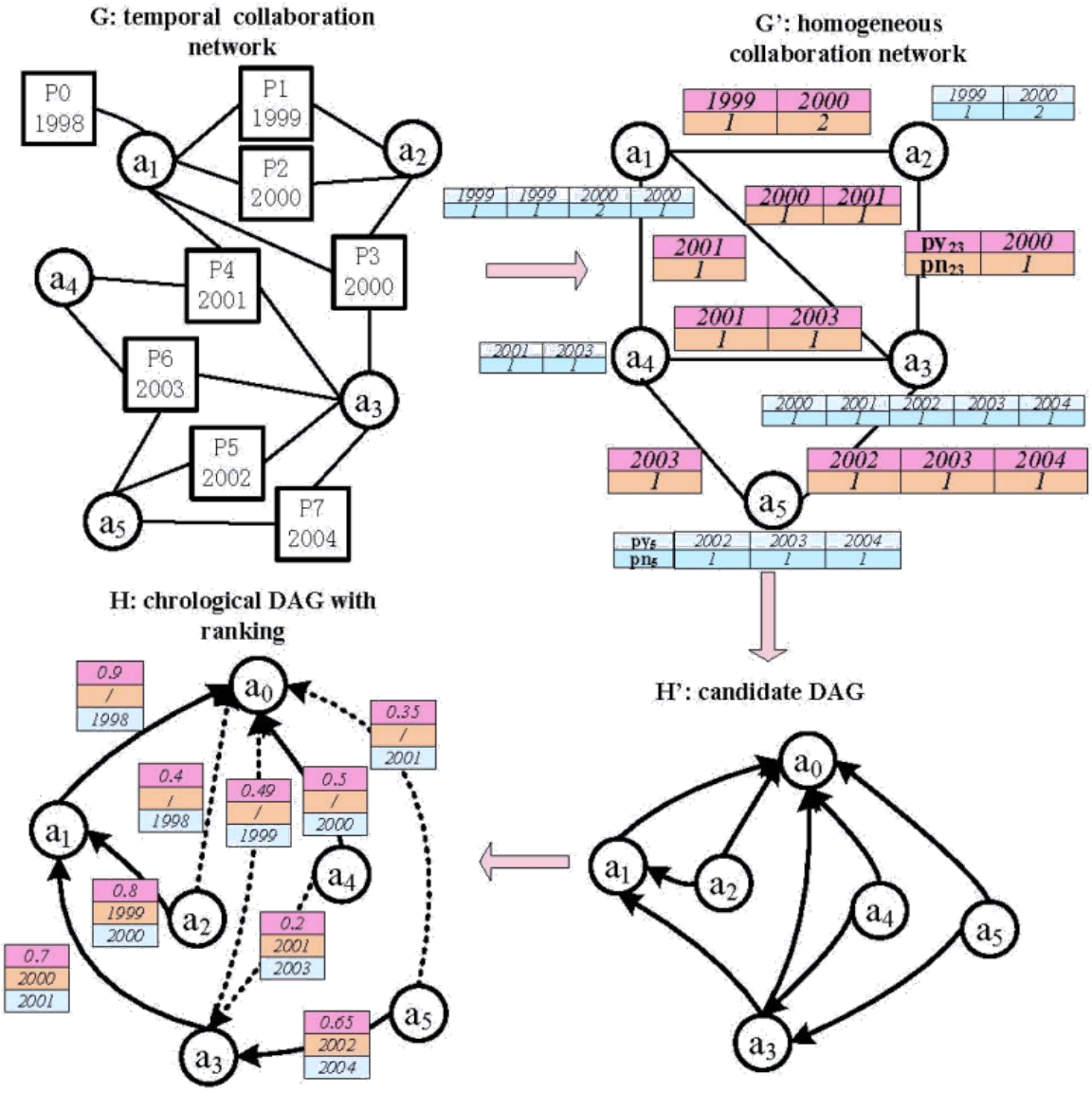


Figure 4: Example of graph transformation

The Kulczynski measure reflects the correlation of the two authors' publications. Here we further incorporate the time factor, to calculate the measure year by year, and check whether there is an increase in the sequence$\{kulc_{ij}^{t}\}$. For IR, we calculate the sequences in the same way. IR is used to measure

the imbalance of the occurrence of a_j given a_i and the occurrence of a_i given a_j. The intuition is that the advisor has more publications than the advisee during the advising time. Then we have the following rule.
Author a_j is not considered to be a_i's advisor if one of the following conditions holds:

a. $IR_{ij}^t < 0$ in the sequence $\{IR_{ij}^t\}$t during the collaboration period of a_i and a_j
b. there is no increase in the sequence during the collaboration period
c. the collaboration period of a_i and a_j lasts only for one year,
d. $py_j^1 + 1 > py_{ij}^1$

We sum average Kul and IR as a rough likelihood, which is defined as:

$$l_{ij} = \frac{\sum_{st_{ij} \le t \le ed_{ij}} (kulc_{ij}^t + IR_{ij}^t)}{2(ed_{ij} - st_{ij} + 1)}$$

In addition, we estimate the starting time and ending time of the advising. The starting time is estimated as the time they begin to collaborate. The ending time is estimated as either the time point when the Kulczynski measure starts to decrease, or the year making the largest difference between the Kulczynski measure before and after it.

(2) **TPFG Model**
In this stage, we transform the candidate graph into a factor graph, see figure below:

The graph is composed of two kinds of nodes: variable nodes and function nodes. Each variable node corresponds to a function node. The factor graph reflects the dependency of the variables. A set of variables are correlated if they are neighbors of the same function node e.g., $y1, y2, y3$ with $f1(y1, y2, y3)$. We can see that two hidden variables are correlated if their corresponding author nodes are linked by an edge on the candidate graph, which means there is a potential advising relationship between them.
After constructing the factor graph, we need to infer the marginal maximal joint probability on it. In general, the sum-product algorithm is used to compute marginal function on a factor graph based on message passing. It performs exact inference without any cycles. However, we can see that, time-constrained

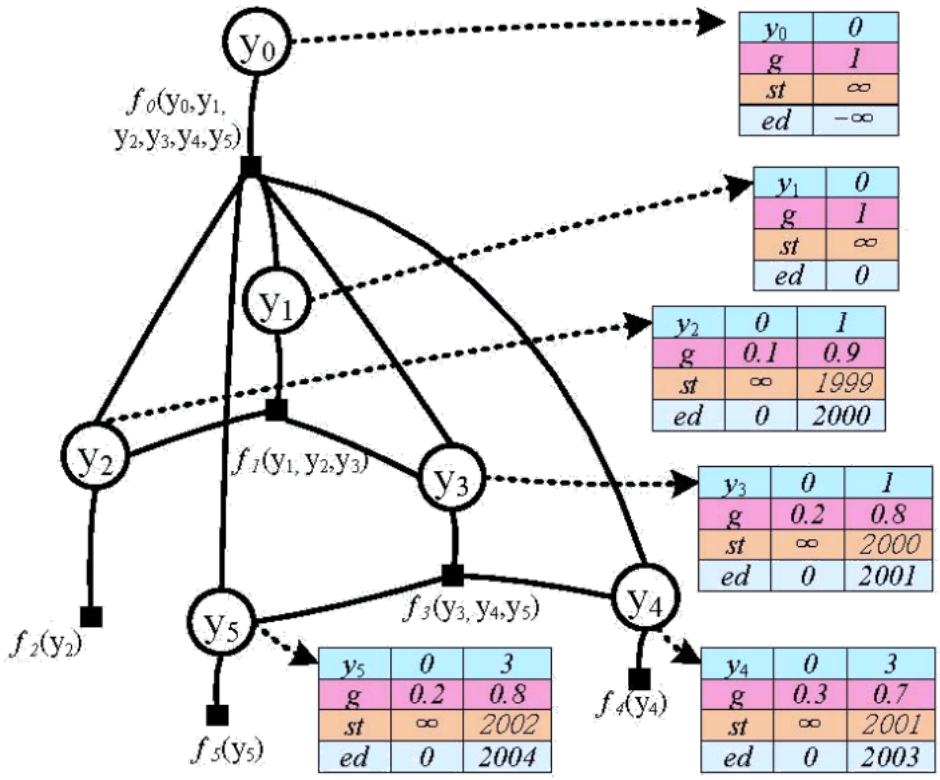


Figure 5: Time-constrained probabilistic factor graph(TPFG)

probabilistic factor graph we construct contains cycles, so we need to change the sum-product algorithm. The new algorithm needs two phases, see figure below:

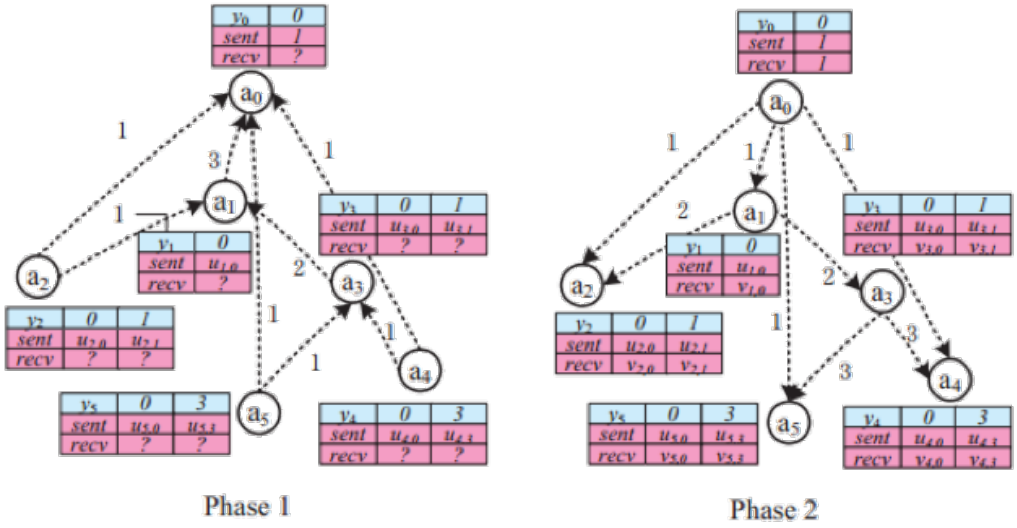


Figure 6: The 2-phase message passing schema

In the first phase, the messages $sent_i$ which passed from one to their ascendants are generated in a similar order as before. In the second, messages returned from ascendants $recv_i$ are stored in each node. After the two phases, each node collects the two vectors to generate the final ranking score. The derived rules are as follows.

$$sent_{ij} = \log l_{ij} + \sum_{k \in Y^{-1}} \max_{st_{kx} > ed_{ij} or x \neq i} sent_{kx}$$

$$recv_{ij} = \max_{j' \in Y_j, edjj' < st_{ij}} (recv_{jj'} + \log_{jj'} + \sum_{k \in Y_j^{-1}, k \neq j} \max_{x \in Y_k, st_{kx} > ed_{jj'} or x \neq j} sent_{kx})$$

$$+ \sum_{x \in Y_i} \max_{j' \in Yx} (\sum_{k \in Y_x^{-1}, k \neq i} \max_{x \in Y_k, st_{kx} > ed_{xj'} or x' \neq x} sent_{kx'})$$

Experiment Setup

- **Data Sets**: we use DBLP data as the dynamic collaboration data set G to infer the advisor-advisee. It consists of 654,628 authors and 1,076,946 publications with time from 1970 to 2008. To test the accuracy of the discovered advisor-advisee relationships, we adopt three data sets: One

is manually labeled by looking into the home page of the advisors, and the other two are crawled from the Mathematics Genealogy project and AI Genealogy project. We refer to them as MAN, MathGP and AIGP respectively. They only partially cover the authors in DBLP. We further separate MAN into three sub data sets: Teacher, PhD and Colleague. Teacher contains all kinds of advisor-advisee pairs, while PhD only contains graduated PhDs pairing with their advisors. Colleague contains colleague pairs which are negative samples for advisor-advisee relationship.

- **Methods**. We compare the proposed TPFG with the following baseline methods:
  Independent Maxima (IndMAX):
  It computes the maximal local likelihood for each variable independently.
  SVM:
  It is a supervised approach and requires labeled pairs, both positive and negative, as training data.
  RULE:

  For each author, from all the collaborators that satisfy $py_{j_1} < py_{i_1}$, choose the one with most coauthored papers.

Experiment Results
We try different rules one by one to construct the corresponding candidate graph, compute the ranking score with our algorithm, and compare the accuracy on some labeled data. The accuracy is compared through ROC curves. Further details are shown in figure below.

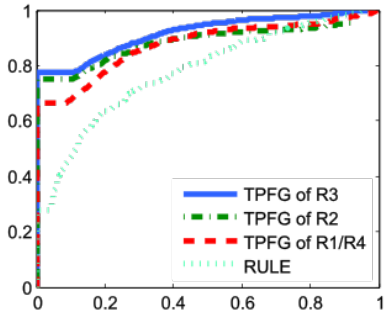


Figure 7: TPFG with different rules

From figure above we can see that R2/R3 has the highest suitability on the tested data. R1 and R4 both lead to a slightly worse curve and their curves overlap. It is observed that TPFG is not sensitive to those rules. For example, if we choose R2, or even R1/R4 other than R3, the worst AOC value 0.88 is not degraded drastically from the optimal choice 0.91. It indicates that our network modeling approach is robust in handling inaccurate local features.
We use different methods to infer advisor-advisee relationships. The results are shown in the table below.

Dataset	**rule**	**SVM**	**IndMAX** Empirical parameter	**IndMAX** optimized parameter	**TPFG** Empirical parameter	**TPFG** optimized parameter
TEST1	69.9%	73.4%	75.2%	78.9%	80.2%	84.4%
TEST2	69.8%	74.6%	74.6%	79.0%	81.5%	84.3%
TEST3	80.6%	86.7%	83.1%	90.9%	88.8%	91.3%

Table 5: Accuracy of prediction by $P@(2,\theta) : \frac{T}{T+F}$
TRAIN1=Colleague(491)+PHD(100)
TEST1=Teacher(257)+MathGP(1909)+Colleague(2166)
TRAIN2=TRAIN3=Teacher(257)+Colleague(2166)
TEST2=PHD(100)+MathGP(1909)+Colleague(4351)
TEST3=AIGP(666)+Colleague(459)
IndMAX, TPFG: left - θ = 3rd quartile of r_{ij}; right -trained
P@(2,θ) means to fetch top 2 potential advisors of ai and check whether
a_j is one of them while $r_{ij} > ri_0$ or $rij > \theta$, where θ is a threshold such as 0.5.

Although in this work we define our model as an unsupervised learning approach, it can also work with supervised learning. We can also optimize the parameter θ in the P@k, according to certain criteria such as achieving best information gain on the training data. Then we use the trained parameters to do predictions on test data. Table 1 shows the improvement by utilizing the training data. After training, TPFG can reach an accuracy of 84% to 91%. And it can achieve comparable or even better accuracy compared with a supervised method.

6 Co-author Prediction

Thanks to recent advances in network science theory, many real world systems can be modeled as a complex network. One of the crucial tasks in complex networks is link prediction which aims to exploit dependencies between the node pairs. Link prediction problem has been well investigated in recent years. At the beginning, the majority of the proposed link predictors assume that the network is homogeneous. Then meta-paths based link predictors are used on heterogeneous network.
We will introduce the two link prediction techniques on homogeneous networks and one technique on heterogeneous network.

(1) **Homogeneous Network**
Generally, homogeneous link predictors can be divided into four categories[4]. We will introduce the first and second categories in detail, the third and the forth briefly.

- **Based on the topology of the network**
  Experiment Setting
  Suppose that there is a social network $G =< V, E >$ in which each edge represents an interaction between u and v that took place at a particular time t(e). We record multiple interactions between u and v as parallel edges, with potentially different timestamps. For two times t <t', let G[t, t'] denote the subgraph of G consisting of all edges with a timestamp between t and t'. We choose four times t0<t0'<t1<t1' and give an algorithm access to the network G[t0, t0']; it must then output a list of edges not present in G[t0, t0'] that are predicted to appear in the network G[t1,t1']. We refer to [t0,t0'] as the training interval and [t1, t1'] as the test interval.
  We denote the subgraph on the training interval by $G_{collab} :=< A, E_{old} >$ and use E_{new} to denote the set of edges <u, v> such that$u, v \in A$and u, v coauthor an article during the test interval, but not the training interval—these are the new interactions we are seeking to predict. In the co-author network, we can identify which authors are active throughout the entire period on the basis of the number of articles published and not on the number of coauthors. Thus, here we define the set Core to consist of all authors who have written at least $\kappa_{training} := 3$ articles during the training period and at least $\kappa_{test} := 3$ articles during the test period.
  Each link predictor p that we consider outputs a ranked list L_p of pairs in $A * A - E_{old}$; these are predicted new collaborations, in decreasing order

of confidence. For our evaluation, we focus on the set Core, so we define $E^*_{new} := E_{new} \cap (Core \times Core)$ and $n := |E^*_{new}|$. Our performance measure for Predictor p is then determined as follows: From the ranked list L_p, we take the first n pairs that are in $Core \times Core$, and determine the size of the intersection of this set of pairs with the set E^*_{new}.

Methods for Link Prediction

All the methods assign a connection weight $score(x, y)$ to pairs of nodes $< x, y >$, based on the input graph G_{collab}, and then produce a ranked list in decreasing order of score(x, y). Thus, they can be viewed as computing a measure of proximity or "smilarity" between nodes x and y, relative to the network topology. Table below summarizes most of these measures.

Methods Based on Node Neighborhoods:

For a node x, let $\Gamma(x)$ denote the set of neighbors of x in G_{collab} . A number of approaches are based on the idea that two nodes x and y are more likely to form a link in the future if their sets of neighborhoods $\Gamma(x)$ and $\Gamma(y)$ have large overlap; this approach follows the natural intuition that such nodes x and y represent authors who have many colleagues in common and hence who are more likely to come into contact themselves.

Methods Based on the Ensemble of All Paths:

A number of methods refine the notion of shortest-path distance by implicitly considering the ensemble of all paths between two nodes.

- **Based on the two-class classification problem**

  In this part, we refer to the method proposed by Al Hasan et al[6].

  Experiment Setting

  Consider a social network G=<V, E> in which each edge $e =< u, v > \in E$ represents an co-author relationship between u and v at a particular time t. To predict a link, the paper partitions the range of publication years into two non-overlapping sub-ranges. The first sub-range is selected as training years and the later one as the testing years. Then, they prepare the classification dataset, by choosing those author pairs, that appeared in the training years, but did not publish any papers together in those years. Each such pair either represents a positive example or a negative example, depending on whether those author pairs published at least one paper in the testing years or not. Classification model of link prediction problem needs to predict this link by successfully distinguishing the positive classes from the dataset. Thus, link prediction problem can be posed as a binary classification problem, that can be solved by employing effective features

graph distance	(negated) length of shortest path between x and y
common neighbors	$\lvert\Gamma(x) \cap \Gamma(y)\rvert$
Jaccard's coefficient	$\dfrac{\lvert\Gamma(x) \cap \Gamma(y)\rvert}{\lvert\Gamma(x) \cup \Gamma(y)\rvert}$
Adamic/Adar	$\sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log\lvert\Gamma(z)\rvert}$
preferential attachment	$\lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert$
Katz$_\beta$	$\sum_{\ell=1}^{\infty} \beta^\ell \cdot \lvert\mathsf{paths}_{x,y}^{\langle\ell\rangle}\rvert$ where $\mathsf{paths}_{x,y}^{\langle\ell\rangle} :=$ {paths of length exactly ℓ from x to y} weighted: $\mathsf{paths}_{x,y}^{\langle 1\rangle} :=$ number of collaborations between x, y. unweighted: $\mathsf{paths}_{x,y}^{\langle 1\rangle} := 1$ iff x and y collaborate.
hitting time stationary-normed commute time stationary-normed	$-H_{x,y}$ $-H_{x,y} \cdot \pi_y$ $-(H_{x,y} + H_{y,x})$ $-H_{x,y} \cdot \pi_y + H_{y,x} \cdot \pi_x)$ where $H_{x,y} :=$ expected time for random walk from x to reach y $\pi_y :=$ stationary-distribution weight of y (proportion of time the random walk is at node y)
rooted PageRank$_\alpha$	stationary distribution weight of y under the following random walk: with probability α, jump to x. with probability $1 - \alpha$, go to a random neighbor of current node.
SimRank$_\gamma$	$\begin{cases} 1 & \text{if } x = y \\ \gamma \cdot \dfrac{\sum_{a \in \Gamma(x)} \sum_{b \in \Gamma(y)} \mathsf{score}(a,b)}{\lvert\Gamma(x)\rvert \cdot \lvert\Gamma(y)\rvert} & \text{otherwise} \end{cases}$

Methods for Link Prediction[5]

in a supervised learning framework.

– Data
This paper still uses DBLP as an example. They used 15 years of data, from 1990 to 2004. First 11 years were used as training and

the last 4 years as testing. Pairs of authors that represent positive class or negative class were chosen randomly from the list of pairs that qualify. Then they constructed the feature vector for each pair of authors.

- Feature set
  Choosing an appropriate feature set is the most critical part of any machine learning algorithm. For link prediction, they should choose features that represent some form of proximity between the pair of vertices that represent a data point. For DBLP, they choose features as follow:
  * Sum of Papers.
    The value of this feature is calculated by adding the number of papers that the pair of authors published in the training years. Since, all authors did not appear in all the training years, they normalized the paper count of each author by the years they appeared in. The choice of this feature comes from the fact that authors having higher paper count are more prolific. If either (or both) of the authors is (are) prolific, the probability is higher that this pair will coauthor compared to the probability for the case of any random pair of authors.
  * Sum of Neighbors
    This feature represents the social connectivity of the pair of authors, by adding the number of neighbors they have. Here, neighborhood is obtained from the co-author relationship information. A more accurate measure would consider the weighted sum of neighbors, where the weights represent the number of publication that a node has with that specific neighbor. They considered all the weights to be 1. This feature is intended to embed the fact that a more connected person is more likely to establish new coauthor links.
  * Shortest Distance
    This feature is one of the most significant features in link prediction. They use smallest hop count as the shortest distance between two nodes.
  * Second Shortest Distance
    This feature is the distance calculated from another shortest path that has no common edge with the first shortest path.

– Classification Algorithms
There are a lot of classification algorithms for supervised learning. In this research, they experiment with seven different classification algorithms. The algorithms that they use are SVM (two different kernels), Decision Tree, Multilayer Perceptron, K-Nearest Neighbors (different variations of distance measure), Naive Bayes, RBF Network and Bagging. Then they compared the performance of the above classifiers using different performance metrics like accuracy, precision-recall, F-value, squared-error etc.

Results and Discussions
Table below shows the performance comparison for different classification algorithms on DBLP datasets. In this datasets, counts of positive class and the negative class were almost the same. So, a baseline classifier would have an accuracy around 50% by classifying all the testing data points to be equal to 1 or 0, whereas all the models that we tried reached an accuracy above 80%. This indicates that the features that we had selected have good discriminating ability.

Classification model	Acuracy	Presicion	Recall	F-value	Squared Error
Decision Tree	82.56	87.70	79.50	83.40	0.3569
SVM(Linear Kernel)	83.04	85.88	82.92	84.37	0.1818
SVM(RBF Kernel)	83.18	87.66	80.93	84.16	0.1760
K Nearest Neighbors	82.42	85.10	82.52	83.79	0.2354
Multilayer Perceptron	82.73	87.70	80.20	83.70	0.3481
RBF Network	78.49	78.90	83.40	81.10	0.4041
Naive Bayes	81.24	87.60	76.90	81.90	0.4073
Bagging	82.13	86.70	80.00	83.22	0.3509

Table 6: Performance of different classification algorithms for BIOBASE database

On accuracy metrics, SVM with RBF kernel performed the best. other popular classifiers, like decision tree, k-nearest neighbors and multilayer perceptron also have similar performances, usually 0.5% to 1% less accurate than SVM. Such a small difference is not statistically significant, so no conclusion can be drawn from the accuracy metric about the most suited algorithm for the link prediction.
In the same tables, they also list Precision, Recall and F-value for the positive class. F-value is the harmonic mean of precision-recall that is

sometimes considered a better performance measure for a classification model in comparison to accuracy, especially if the population of the classes are biased in the training dataset. Considering the F-value metric, the rank of the classifiers don't really change, indicating that all the models have similar precision-recall behavior. Comparing the precision and recall columns, they find that most of the classifiers have precision value significantly higher than the recall value for the positive class. This indicates that their models have more false negatives than false positives.
This paper's another objective is to compare the features to judge their relative strength in a link prediction task. They run several algorithms for this. Table below provide a comparison of several features by showing their rank of importance as obtained by different algorithms. Last column shows an average rank that is rounded to the nearest integer.

Attribute	Information Gain	Gain Ratio	Chi-Square Attribute Eval.	SVM feature evaluator	Avg. Rank
Sum of Papers	4	4	4	2	4
Sum of Neighbors	3	3	3	4	3
Shortest Distance	1	1	1	1	1
2nd Shortest Distance	2	2	2	3	2

Table 7: Rank of different Attributes using different algorithms for DBLP dataset

From the results shown in table above, we can know that, shortest distance is the best feature in DBLP dataset.
Conclusion
From the results of experiment, we can see that, the link prediction problem can be handled effectively by modeling it as a classification problem.

- **Based on Probabilistic methods**
  The main idea is to optimize a defined target function in order to establish a parametric model that can best fit the observed data. Then, the posterior probabilities are obtained by defining a conditional probability model over the learned parameters.
- **Based on Linear algebraic methods**
  Accordingly, several graph kernels and dimensionality reduction methods are employed to solve the link prediction problem. These methods learn a function F that works directly on the graph adjacency or the graph Laplacian matrix.

(2) **Heterogeneous Network**

Most of the existing link prediction studies are designed for homogeneous networks, in which only one type of objects exists in the network. However, in the real world, most of the networks are heterogeneous. So if we use the topological features between objects in a heterogeneous network to do some prediction, it will be more accurate.

In this part, a model, called meta path-based relationship prediction model, is proposed to solve the problem of co-author relationship prediction in the heterogeneous bibliographic network. This new method is proposed by Yizhou Sun, et al[7].

First, meta path-based topological features are systematically extracted from the network. Then, a supervised model is used to learn the best weights associated with different topological features in deciding the co-author relationships. This paper uses the DBLP network as an example.

Meta path-based topological feature definition

As the heterogeneous networks are more complex, there are different types of objects and multi-typed relations. The nodes who are neighbors could belong to different types, paths between two nodes could follow different meta paths and indicate different relations. Thus, the paper defines a more complex strategy to generate topological features to distinguish paths with different meanings.

The paper defines the meta path over the network schema firstly. Based on the DBLP network, they extract all the meta paths within a length constraint, say 4, starting and ending with author type A(author). You can see in the table below.

After the semantic meaning of the meta path is defined, the next step is to give measures on these meta paths. In the paper, four measures are proposed along the lines of topological features in homogeneous networks. They are path count, normalized path count, random walk, and symmetric random walk, which are defined as follows.

- Path count
  Path count measures the number of path instances between two objects following a given meta path, denoted as PC_R, where R is the relation denoted by the meta path.
- Normalized path count
  Normalized path count is to discount the number of paths between two objects in the network by their overall connectivity, and is defined as:

Meta Path	Semantic Meaning of the Relation
$A-P-A$	a_i and a_j are coauthors (the target relation)
$A-P\rightarrow P-A$	a_i cites a_j
$A-P\leftarrow P-A$	a_i is cited by a_j
$A-P-V-P-A$	a_i and a_j publish in the same venues
$A-P-A-P-A$	a_i and a_j are co-authors of the same authors
$A-P-T-P-A$	a_i and a_j write the same topics
$A-P\rightarrow P\rightarrow P-A$	a_i cites papers that cite a_j
$A-P\leftarrow P\leftarrow P-A$	a_i is cited by papers that are cited by a_j
$A-P\rightarrow P\leftarrow P-A$	a_i and a_j cite the same papers
$A-P\leftarrow P\rightarrow P-A$	a_i and a_j are cited by the same papers

Meta Paths under length 4 between authors in the DBLP network

$$NPC_R(a_i,a_j)=\frac{PC_R(a_i,a_j)+PC_{R-1}(a_j,a_i)}{PC_R(a_i,\cdot)+PC_R(\cdot,a_j)}$$

where R^{-1} denotes the inverse relation of R, $PC_R(a_i,\cdot)$ denotes the total number of paths following R starting with a_i, and $PC_R(\cdot,a_j)$ denotes the total number of paths following R ending with a_j. $PC_R(a_i,\cdot)$ and $PC_R(\cdot,a_j)$ can be viewed as degrees of a_i and a_j in the network respective to R and R^{-1}.

- Random walk.
  Random walk measure along a meta path is defined as:

$$RW_R(a_i,a_j)=\frac{PC_R(a_i,a_j)}{PC_R(a_i,\cdot)}$$

- Symmetric random walk
  Symmetric random walk considers the random walk from two directions along the meta path, and defined as:

$$SRW_R(a_i,a_j)=RW_R(a_i,a_j)+RW_{R-1}(a_j,a_i)$$

The co-authorship prediction model

This model is used to predict the probability of co-authorship between two authors based on topological feature between them. In this paper, they choose

the logistic regression model as the prediction model. For each training pair of authors $< a_{i_1}, a_{i_2} >$ let x_i be the (d+1)-dimensional vector including constant 1 and d topologcial features between them, and y_i be the label of whether they will be co-authors in the future ($y_i = 1$ if they will be co-authors, and otherwise 0), which follows binomial distribution with probability p_i. The probability p_i is modeled as follows:

$$p_i = \frac{e^{x_i \beta}}{e^{x_i \beta} + 1}$$

where β is the $d + 1$ coefficient weights associated with the constant and each topological feature. We then use standard MLE(Maximum Likelihood Estimation) to derive $\hat{\beta}$ that maximizes the likelihood of all the training pairs:

$$L = \Pi_i P_i^{y_i} (1 - p_i)^{(1-y_i)}$$

Experiment

- Experiment Setting
  The DBLP bibliographic network is used for experiment in this paper. According to the publication year, the time is divided into three time intervals: T0 =[1989; 1995], T1 = [1996; 2002], and T2 = [2003; 2009]. For the training stage, we use T0 as the past time interval, and T1 as the future time interval, which is denoted as T0-T1 time framework. For the test stage, we consider the same time framework T0-T1 for most of the studies, and consider T1-T2 time framework for the model generality test.
  Let an author pair be a_i is called the source author, and a_j is the target author. In all, then four labeled datasets are set:
  - the highly productive source authors with 2-hop target authors (denoted as HP2hop)
  - the highly productive source authors with 3-hop target authors (denoted as HP3hop)
  - the less productive source authors with 2-hop target authors (denoted as LP2hop)
  - the less productive source authors with 3-hop target authors (denoted as LP3hop)

  The highly productive authors are those who have published no less than 16 papers in the past time interval; and less productive authors are those

with between 5 and 15 publications. To evaluate the prediction accuracy, two measures are used. The first measure is the classification accuracy rate (accuracy) for binary prediction under the cut-off score as 0.5, and the second one is the area under ROC curve (AUC).

- Overall Accuracy
  Firstly, the paper compares the heterogeneous topological features with the homogeneous ones. For the heterogeneous topological features, path count measure is used; for homogeneous topological features, they use (1) the number of common co-authors, (2) the rooted PageRank with restart probability $\alpha = 0.2$ for the co-author sub-network, and (3) the number of paths between two authors of length no longer than 4, disregarding their different meta paths (denoted as homogeneous PC). The rooted PageRank measure is only calculated for the HP3hop dataset, due to its inefficiency in calculation for large number of authors. The comparison results are summarized below.

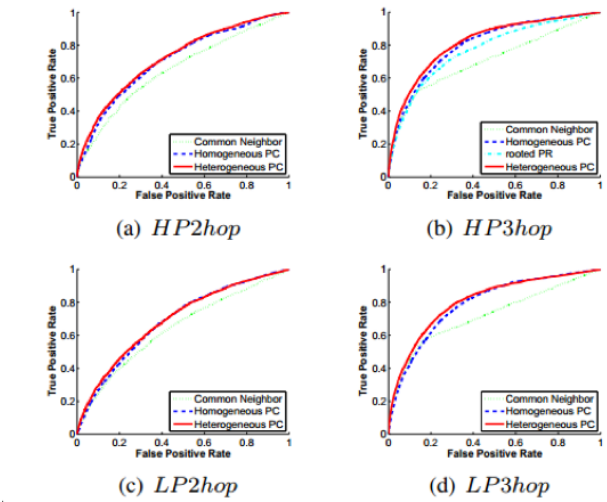


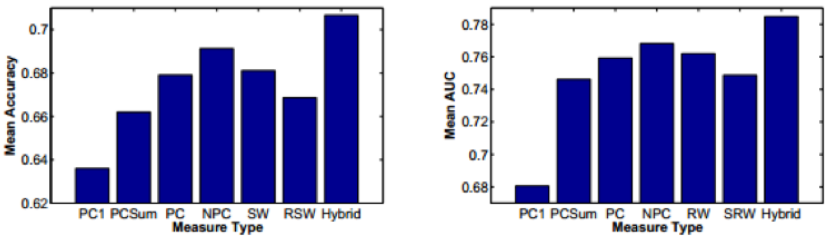
Figure 8: Homogeneous Features vs. Heterogeneous PC Feature

We can see that the heterogeneous topological feature beats the homogeneous ones in all the four datasets, which validates the necessity to consider the different meta paths separately in heterogeneous networks.
Then they compare different measures proposed for heterogeneous topological features. (1) the path count (PC), (2) the normalized path count

Dataset	Topological features	Accuracy	AUC
$HP2hop$	common neighbor	0.6053	0.6537
	homogeneous PC	0.6433	0.7098
	heterogeneous PC	**0.6545**	**0.7230**
$HP3hop$	common neighbor	0.6589	0.7078
	homogeneous PC	0.6990	0.7998
	rooted PageRank	0.6433	0.7098
	heterogeneous PC	**0.7173**	**0.8158**
$LP2hop$	common neighbor	0.5995	0.6415
	homogeneous PC	0.6154	0.6868
	heterogeneous PC	**0.6300**	**0.6935**
$LP3hop$	common neighbor	0.6804	0.7195
	homogeneous PC	0.6901	0.7883
	heterogeneous PC	**0.7147**	**0.8046**

Homogeneous Features vs. Heterogeneous PC Feature

(NPC), (3) the random walk (RW), (4) the symmetric random walk (SRW), and (5) the hybrid features of (1)-(4)(hybrid). The results for average accuracy over 4 datasets can be seen in figure below.



(a) Mean accuracy (b) Mean AUC

Figure 9: Average Accuracy over 4 Datasets for Different Features

It turns out that:

– All the heterogeneous features beat the homogeneous features (common neighbor is denoted as PC1, and homogeneous PC is denoted as PCSum).

- The normalized path count beats all the other three individual measures.
- The hybrid feature produces the best prediction accuracy.

Conclusion
Experiments on the DBLP bibliographic network show that by considering heterogeneous topological features, the relationship prediction accuracy can be significantly improved, and the model using hybrid features gives the best overall performance.

References

[1] Gergely Palla, Imre Derenyi, Illes Farkas1, and Tamas Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 2005.

[2] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks, 2008.

[3] Chi Wang, Jiawei Han, Yuntao Jia, Jie Tang, Duo Zhang, Yintao Yu, and Jingyi Guo. Mining advisor-advisee relationships from research publication networks. In *Proceedings of the Sixteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010.

[4] Shakibian H, Charkari N M, and Jalili S. A multilayered approach for link prediction in heterogeneous complex networks. *Journal of Computational Science*, 2016.

[5] Liben-Nowell D and Kleinberg J. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology*, 2007.

[6] Al Hasan M, Chaoji V, and Salem S. Link prediction using supervised learning. *workshop on link analysis, counter-terrorism and security*, 2006.

[7] Sun Y, Barber R, and Gupta M. Co-author relationship prediction in heterogeneous bibliographic networks. *Advances in Social Networks Analysis and Mining*, 2011.