Ranking of Academic Papers
Final Report

Data Science Fundamentals CSE 519
Prof. Steven Skiena

December 7, 2018

1 Abstract

In this project we develop a metric to rank papers and researchers from various fields of research. The algorithm we use is a modified version of the page rank which considers important information such as the number of authors and the year of publication. In addition we also rank the conferences using the scores of the research papers. We then use the above information to rank the authors. We classify the research papers on the basis of sub-domains which were extracted from abstract . We compare our developed metric with h-index for each domain. We compute various centrality measures and devised a reach function to measure the influence of authors. In the end to fine tune our metric we use the LGBM machine learning model. Time series analysis helps us in understanding the influence of popularity on the increase in citation count (impact) of the paper.

2 Implementation

The dataset used for this project was obtained from DBLP [1].This dataset contains information such as the name of the research paper, its authors, the year of publication, abstract, the conference of publication and the list of papers cited by the paper.

2.1 Data Cleaning

The data was cleaned as follows: Fields like abstract and references had some rows containing NA. Imputing such fields is difficult given the conditions that papers in the references had string ids. All such rows were dropped. There were some rows where the conference field had null values. We treated all such values as a separate unknown conference and used it in the ranking algorithm. On scoring the conferences, we observed that that unknown conference had a really low score so this assumption was justified. In addition, we removed all the self references (author citing themselves) as keeping them would give an incorrect (biased) rank.

2.2 Modified Page Rank Algorithm

The citation graph is defined as comprising a set V of nodes, with each node N_i representing a research paper R_i and a set E of directed edges, with each edge E_{ij} directed from the citing node N_i to the cited node N_j. The edges defined above are :

- In-Links: All the links connecting a node with other nodes that have cited this node.
- Out-Links All the links from a node to other nodes that have been cited by this node.

The page rank algorithm mentioned below uses these inlinks and outlinks to iteratively calculate the score of each node. The stopping criterion is the flag value, which is set to true if the authority score does not change, else the score for the node is updated. [2]

The rank of a node is proportional to the ranks of all the nodes that point to it. If a node pointing to a node A also points to many other nodes, the score effect contributed by this node should be lower as opposed to a node that would only point to this node A. Therefore, this is factored in to the algorithm by dividing the inlink score by the number outlinks of the inlink. The use of damping factor in this algorithm is required to prevent the scores of research papers that do not have any inlinks from falling to zero. The default value of damping factor is set to 0.85 based on observations from several papers.

Algorithm 1 Page Rank

```
Require: Citation Network $Cite\_Net$, Paper Year $PYear$, Outlinks Count $OutC$, Paper Inlinks $InLinks$,
  Damping Factor $\theta$ Paper Rank PRank to 1.0 for each paper R
  while $True$ do
      $flag = True$
      for Each paper R in PRank do
          Current Score $Curr\_Score$= PRank[R]
          if R in PI then
              Inlinks List IL = InLinks[R]
              New Score NewScore= 0.0
              for Each link I in IL do
                  if I in PRank then
                      NewScore += $PRank[I]/OutC[I]$
                  end if
                  NewScore = $(1 - \theta) + \theta * NewScore$
                  if  Curr_Score is not equal to NewScore then
                      $flag = False$
                  end if
                  Updated Paper Rank $UPRank[R] = NewScore$
              end for
          end if
      end for
  end while
  if Flag = True then
      break
  end if
  Copy UPRank to PRank
  Clear UPRank
  Maximum Score MS = Maximum score in PRank
  for Each Paper R in PRank do
      PRank[R]=MS
  end for
```

Once we obtain all the scores for the papers, we divide the scores by the number of authors of that paper. This gives us a better scoring mechanism as the efforts of a single author are taken into consideration. For each paper we compute:

$$NewScore = \frac{NewScore}{Number\ of\ Authors\ of\ that\ paper}$$

Given that we have citations of papers and their year of publication, time independence was implemented by calculating the average number of citations per year and number of papers published for that year. This term is then incorporated into the algorithm - we divide NewScore by the average number of citations per paper per year.

$$NewScore = (1 - \theta) + \theta * (\frac{NewScore}{A\_YCiteCount[Y]})$$

```
for Each paper R in PYear do
    Get Year of R: Y=PYear[R]
    Sum of outlinks for R: Year Citation Count YCitecount[Y] += OutC[R]
    Increase Year Paper count: YPaperCount[Y] += 1
end for
```

We rank conferences based on the fact that the rank of a conference depends on the quality of papers it publishes. Cumulative score for a conference is the sum of scores of the papers divided by the number of papers published in it .

$$Conference\ Score = \sum \frac{Paper\ Score}{Number\ of\ papers}$$

The rank of an author is a combination of the score of the papers he publishes as well as the conferences he publishes them in. This depends on the fact that publishing papers in a good conference is difficult and should contribute more to the score. The conference score is treated as a weight to the respective research papers and then the mean score is calculated as follows [3]:

$$Author\ Score = \sum \frac{Paper\ Score\ *\ Conference\ Score}{Number\ of\ papers\ published\ by\ the\ author}$$

2.3 Topic Modeling

One of the most important components of the project involves finding the domain of the paper. This is essential because papers from different domains have varied importance of citation counts. For example papers in chemistry, physics and mathematics are difficult to publish and take a long period of time to become relevant, compared to papers in the computer science domain. So classifying the papers into separate domains is required to rank the papers and hence the authors. In our current dataset, we observed that the data is related only to papers in the field of Computer Science. We aim to classify the papers according to their sub-domains.

2.3.1 Approach 1:

For obtaining keywords/topics from the text in order to classify the domain of the paper we use topic modeling. It is an unsupervised learning approach that looks for repeating patterns of co-occurring terms. We used Latent Dirichlet Allocation (LDA) for topic modeling method which is one the most popular method for finding topic clusters. [4]

LDA assumes documents are produced from a mixture of topics. Those topics then generate words based on their probability distribution. Given a dataset of documents, LDA backtracks and tries to figure out what topics would create those documents in the first place. In general, LDA is a matrix factorization technique. It converts document-term matrix into two lower dimensional matrices document-topic and topic-words matrix. We run this algorithm until we come across a stable distribution of the topics.

- We have used abstract of papers as documents for training LDA model. The keywords for each topic were generic and hence not useful in classification of papers.
- We have used title of papers as documents for training LDA model. It didn't perform any better.
- We have used title and conference venue of papers as documents for training LDA model. This didn't work either.

2.3.2 Approach 2:

As the above LDA method failed to give us the domain as it produced common words. We then shifted to a semi-supervised technique called the guidedLDA which works on the same principle of LDA but now we have seed values (topic categories and a bunch keywords that define this topic). The algorithm gives an extra boost to these seed words to converge to a corresponding topic. This worked a lot better than LDA but was still not up to the mark.

2.3.3 Approach 3:

We identified the most important sub topics from online journals and articles from wikipedia. After dividing into 8 major subtopics (Computer Vision, Operating System, Computer Networks, Theory, Artificial Intelligence, Computer Architecture, Bio-informatics, Databases) we identified the most important keywords in each subfield, which helped us distinguish the articles based on their abstracts. For each subtopic, first we found out the most important 5 keywords in the field. Then, for each keyword, we looked up for synonyms of the word and picked the best synonyms along with the keyword to represent the subtopic.

We use these keywords for Topic modeling. We check the abstract of each paper and generate a score based on the keywords of each subtopic. The subtopic receiving the highest score is assigned to the paper. This approach worked the best and we were able to classify the papers in various subdomains with very low error rate.

2.4 Machine Learning Model

We have used a Random Forest Regressor and Light Gradient Boosting machine learning models to fine tune our proposed metric based on some known facts/derived ones about the paper (features). The features used are as follows-

- **Conference score:** It is calculated as the sum of paper scores (calculated using modified page-rank) divided by the number of papers in a conference venue.
- **Number of References:** The number of references of a research paper.
- **Age of the paper:** The time since the paper has been published, i.e (2017 - year of publication).
- **h-index:** Average of the h-index of each of the authors
- **i-index:** Average of the i-index of each of the authors
- **Author score:** Average of the author scores (calculated using modified page-rank).
- **Topic:** The topics have been label encoded.

The target values are the paper scores obtained from modified page rank algorithm. The rmse value for Random Forest was 0.0026 and LGBM was 0.0025 which suggests that both the models are a good fit and predicts the score accurately. The predictions from this model gives us the improved new metric scores.

2.5 Ranking of papers

We generated top 10 papers based on the page rank algorithm and modified page rank algorithm. From Table 1 and 2 we can observe that most of the papers are common in top 10. The one that catches our eye is "Very Deep Convolutional Networks for Large-Scale Image Recognition" published in 2015 has come up in the top 10 list of modified page rank algorithm which wasn't as well ranked by page rank. The modification of removing time based citation bias was indeed useful.

- Ranking based on Page Rank

Table 1: Top 10 papers based on PageRank algorithm

Paper Title	**Score**	**Year**
A Temporal Logic of Nested Calls and Returns	1.0	2004
Perceived usefulness, perceived ease of use, and user acceptance of information technology	0.805	1989
A method for obtaining digital signatures and public-key cryptosystems	0.700	1978
Data clustering: a review	0.523	1999
Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces	0.474	1997
Time, clocks, and the ordering of events in a distributed system	0.473	1978
Term-weighting approaches in automatic text retrieval	0.439	1988
SURF: speeded up robust features	0.432	2006
Handlers of Algebraic Effects	0.416	2009
A density-based algorithm for discovering clusters in large spatial databases with noise	0.413	1996

- Ranking based on modified pagerank algorithm

Table 2: Top 10 papers using Modified Page Rank

Paper Title	**Score**	**Year**
Perceived usefulness, perceived ease of use, and user acceptance of information technology	1.0	1989
A method for obtaining digital signatures and public-key cryptosystems	0.873	1978
Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces	0.602	1997
Time, clocks, and the ordering of events in a distributed system	0.6019	1978
Very Deep Convolutional Networks for Large-Scale Image Recognition	0.562	2015
Term-weighting approaches in automatic text retrieval	0.556	1988
A density-based algorithm for discovering clusters in large spatial databases with noise	0.523	1996
Computer architecture: a quantitative approach	0.516	1990
A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting	0.458	1997
A Temporal Logic of Nested Calls and Returns	0.406	2004

2.6 Google Scholar Scraper

To retrieve various indices such as the h-index, i10-index along with the most common domains of the authors, we have written a script to scrape the google scholar website to retrieve the required metrics. We have utilized BeautifulSoup library in python for implementing the scraper. Google Scholar website has rate limits for making scripted calls to the website. Hence we are retrieving metrics for only the top 100 authors for each type of internal metric that we calculated. Since Google Scholar does not have metrics for few of the authors present in our dataset, we have to improve our scraper as a part of our next steps to scrape metrics to scrape their metrics from other websites such as Scopus.

2.7 Reach Function

To quantify the influence or "reach" of a paper, we used different centrality measures. [5]

- **Betweenness Centrality:**
  Betweenness centrality of a node v is the sum of the fraction of all-pairs shortest paths that pass through v. Vertices with high betweenness may have considerable influence within a network as they connect clusters with high edge density.

  $$c_B(v) = \sum \frac{\sigma(s,t|v)}{\sigma(s,t)} \quad s,t \in v$$

  where V is the set of nodes, $\sigma(s,t)$ is the number of shortest (s,t)-paths, and $\sigma(s,t|v)$ is the number of those paths passing through some node v other than s,t. If s=t, $\sigma(s,t) = 1$, and if $v \in s,t \quad \sigma(s,t|v) = 0$

- **Degree Centrality:**
  The degree centrality for a node v is the fraction of nodes it is connected to. Generally, vertices with higher degree or more connections are more central to the structure and tend to have a greater influence on others. In case of a directed graph, as in our case, we use in-degree centrality and out-degree centrality.

  The in-degree centrality for a node v is the fraction of nodes its incoming edges are connected to. The out-degree centrality for a node v is the fraction of nodes its outgoing edges are connected to.

- **Eigenvector Centrality:** Eigenvector centrality computes the centrality for a node based on the centrality of its neighbors. The eigenvector centrality for node i is the i-th element of the vector x defined by the below equation. This measure was already calculated as part of the page-rank algorithm.

  $$Ax = \lambda x$$

  where A is the adjacency matrix of the graph G with eigenvalue λ.

- **Katz Centrality:** This is a modification of eigenvector centrality.

  $$x_i = \alpha \sum_j A_{ij} x_j + \beta$$

where A is the adjacency matrix of the graph G with eigenvalues λ.

The parameter β controls the initial centrality and $\alpha < \frac{1}{\lambda_{max}}$. It calculates the relative influence of a node by measuring the number of the first degree neighbors and also all other nodes in the network that connect to the node under consideration through these immediate neighbors.

Extra weight can be provided to immediate neighbors through the parameter β. Connections made with distant neighbors are penalized by an attenuation factor α which should be strictly less than the inverse largest eigenvalue of the adjacency matrix in order for the Katz centrality to be computed correctly.

- **Local reach centrality:** Local reaching centrality of a node in a directed graph is the proportion of other nodes reachable from that node or the proportion of the graph that is reachable from the neighbors of the node.

Table 3: Centrality values for top 10 authors

Author	**Eigenvector Centrality**	**In-Degree Centrality**
Vittorio Ferrari	0.1224	0.0096
Andrew Zisserman	0.1223	0.0123
Tinne Tuytelaars	0.1019	0.0076
Antonio Torralba	0.0805	0.0091
Cordelia Schmid	0.0794	0.0043
Rob Fergus	0.0771	0.0047
Jitendra Malik	0.0769	0.0032
Marcin Eichner	0.0749	0.0034
Daphne Koller	0.0720	0.0037
David A. Forsyth	0.0704	0.0044

The table 3 contains some of the centrality measures calculated for the top ten authors according to the eigenvector-centrality are listed and as expected their In-Degree centralities are also among the highest. From the centrality values we can infer that "Tinne Tuytelaars" is cited by authors who are important (high centrality) as compared to "Antonio Torralba" who is cited by many not so influential authors. Calculating other centrality measures was consuming a lot of time so we couldn't compute them.

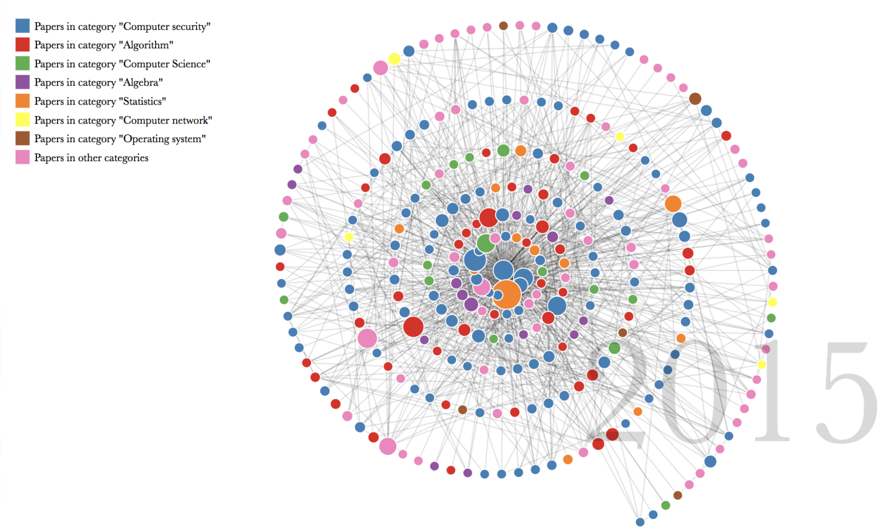
Papers in category "Computer security"
Papers in category "Algorithm"
Papers in category "Computer Science"
Papers in category "Algebra"
Papers in category "Statistics"
Papers in category "Computer network"
Papers in category "Operating system"
Papers in other categories
2015

Figure 1: Example visualization showing the influence of the author "Leonard M Adleman" in different domains over the years

The above plot was created using eigenfactor, to visualize the publications of a particular author (in this case Leonard M. Adleman, up to the year 2015) and his influence in various domains or sub-domains, which are represented using different colors.

Algorithm 2 Reach function Algorithm (Global)

```
“Reach" contains the reach of an author calculated by the maximum no. of fields an author is referenced with a paper.
“ReachAverage" contains the reach of an author calculated by the average of the different reaches of each of their paper.
“Inlinks" contains the papers which made references to the current paper.
for Each row $\in$ dataset do
    max_reach = 0
    domains = empty_list
    for Each key in inlinks[id] do
        domains.append(papers_topic[k])
    end for
    current_reach = len(dict(Counter(domains)))
    for each author in row['authors'] do
        if reach[author] < current_reach then
            reach[author] = current_reach
        end if
        reach_avg[author].append(current_reach)
    end for
end for
```

Using Algorithm: 2, we calculate the **global reach** of the authors, i.e., their influence in various domains. For calculating **local reach**, we have used our modified page rank algorithm which was run for papers in each domain.

2.8 Validation

We compared the results obtained by modified page rank (our metric) with h-index [6] and number of citations. The sub-domains of the authors and the reach is calculated for each as using the reach algorithm. Few important observations from Table 4.
Note: Top 100 researchers from multiple domains has been shown in the jupyter notebook along with their h-index.

- The h-index has a better correlation with number of citations than our proposed metric in general.
- The author "Leo Breiman" has a very low h-index but is still ranked high by our metric because most of his papers are written only by him. We have penalized the papers which have many authors.
- "Leo Breiman" was mis-classified as belonging to the topic of Theory whereas most of his influential work has been in the field of Artificial Intelligence.
- "Judea Pearl" has less citations as compared to others but he has a high h-index because most of his papers have good number of citations (not just few highly cited papers). He is also cited by highly renowned authors so our calculated metric has ranked him higher.
- We can see that authors from the Computer Vision field have less average reach which means that they are generally cited by people in the same domain, i.e., they have lesser influence in other domains.
- Our metric in general gave a higher rank to authors whose influence is spread across multiple domains and not just in one domain.
- The google scholar h-index has higher values than our dataset h-index as it doesn't have all the data but there is clear correlation.

Table 4: Top 10 authors as per our metric

Name	Citations	GS h-index	h-index	Metric Rank	Citation Rank	Sub-Domain	Max Reach	Average Reach
David E. Goldberg	113375	104	73	1	2	Theory	8	3
David G. Lowe	77912	50	50	2	13	Computer Vision	3	1
Anil K. Jain	121850	179	144	3	1	Computer Vision	7	2
Lotfi A. Zadeh	44605	110	50	4	81	AI	7	4
Judea Pearl	29087	96	67	5	289	Theory	8	4
Leo Breiman	47250	51	8	6	67	AI	6	5
Adi Shamir	55714	56	63	7	43	Theory	8	5
Ronald R. Yager	41128	116	87	8	107	AI	8	2
Leslie Lamport	38292	74	56	9	132	OS	8	3
David L. Donoho	62357	61	50	10	32	Computer Vision	**8**	4

3 Results

3.1 Time Series Analysis:

For giving an unbiased rank to a paper, it is essential to identify if a paper has gained citations because of the popularity of the domain at that period of time or was actually influential. To identify this we devised an algorithm to find the increase in impact against time.

Algorithm 3 Time Series Algorithm

```
“OurMetricScores" contains the paper scores according to our metric
“OutLinks" contains the papers referred by each paper
for Each each paper $p \in$ Research Papers do
    Obtain list of papers C which have cited p
    for Each $c \in C$ do
        year = Year[c]
        score[year] += OurMetricScores[c]/OutLinks[c]
        count[year] = count[year] + 1
        result[year] = score[year]/count[year]
    end for
end for
```

In left graph of figure 2 we can clearly see the rapid increase in metric score from year 2012 because of the gain in popularity of "Artificial Intelligence" field whereas the right plot shows that the metric score saturates after the year of 2014 and the there has been a slight increase in metric score each year from 1990 (no rapid increase or boom). So we need to penalize paper scores of years (2012-present) in AI domain by weighing the paper scores with some fraction of 1 (e.g 0.8).

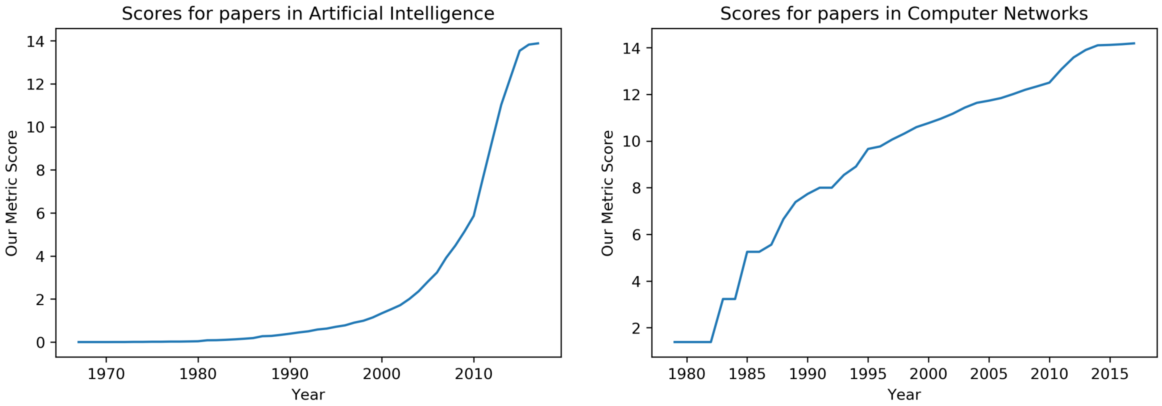

Figure 2: Comparison of metric score against time for AI and CN

3.2 Popularity of papers from arXiv:

From our data-set we looked at a popular field in this time frame “Artificial Intelligence" and picked the papers that gaining citations at a very high rate (received high rank using our metric). We observe that on an average there is increase in 60-80 citations every year (for popular papers). Refer to Table 5.

Table 5: Top papers in the field of Artificial Intelligence after 2016

Paper Title	**Year**	**Citations**
MLlib: machine learning in apache spark	2016	260
Dropout as a Bayesian approximation: representing model uncertainty in deep learning	2016	243
Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks	2016	204
LSTM: A Search Space Odyssey	2016	203
Fake It Till You Make It: Reputation Competition and Yelp Review Fraud	2016	156
The journal coverage of Web of Science and Scopus: a comparative analysis	2016	147
Fusing audio, visual and textual clues for sentiment analysis from multimodal content	2016	143
HCP: A Flexible CNN Framework for Multi-Label Image Classification	2016	130
On the Global and Linear Convergence of the Generalized Alternating Direction Method of Multipliers	2016	116
What Makes for Effective Detection Proposals	2016	116

The following papers in Table 6 are in the field of “Artificial Intelligence" published in year 2016 and gaining citations at a similar rate so they are expected to be popular in the coming years.

Table 6: Papers on arXiv that should be popular

Paper Title	**Year**	**Citations**
Development and validation of deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs	2016	719
Xception: Deep learning with depthwise separable convolutions	2017	502
Youtube-8m: A large-scale video classification benchmark	2016	250
DeepStack: Expert-Level Artificial Intelligence in No-Limit Poker	2017	136
Deep reinforcement learning: An overview	2017	134

3.3 Plots

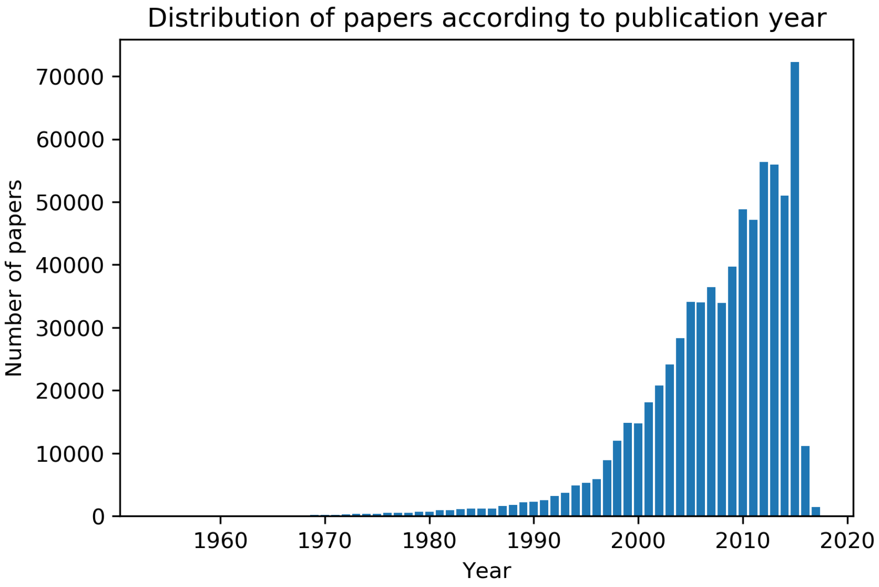

Figure 3

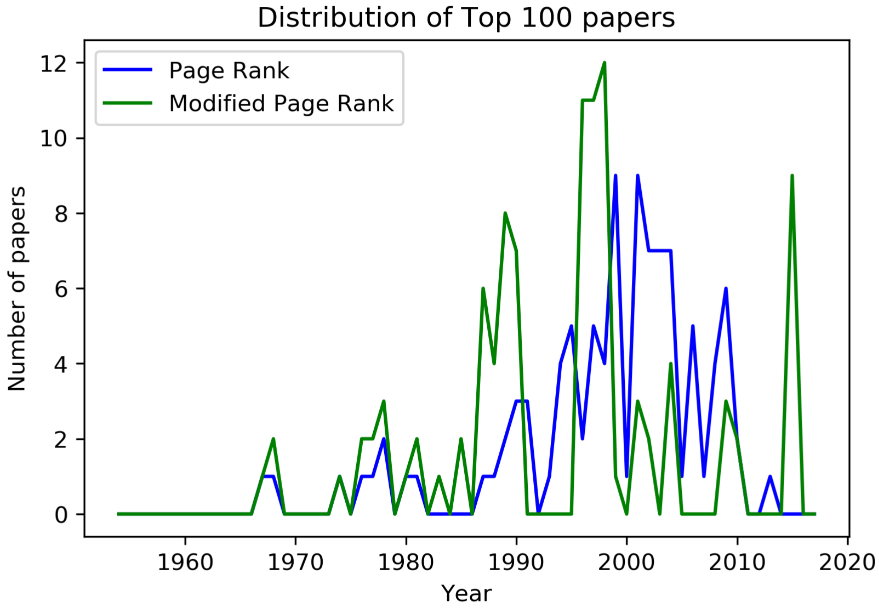

Figure 4

We expect the top 100 papers obtained by the modified page rank algorithm to include most of the new papers, compared to the page rank algorithm. Surprisingly, according to Figure 4 we see that both the algorithms produce papers belonging to the time period 1995-2010. One of the reasons is that most of the papers in the dataset belonged to that time frame. Refer Figure 3. By looking at the list of papers we observed that most of the recent papers actually went up by many ranks in modified page rank algorithm. Unfortunately, this distribution fails to capture such an increase.

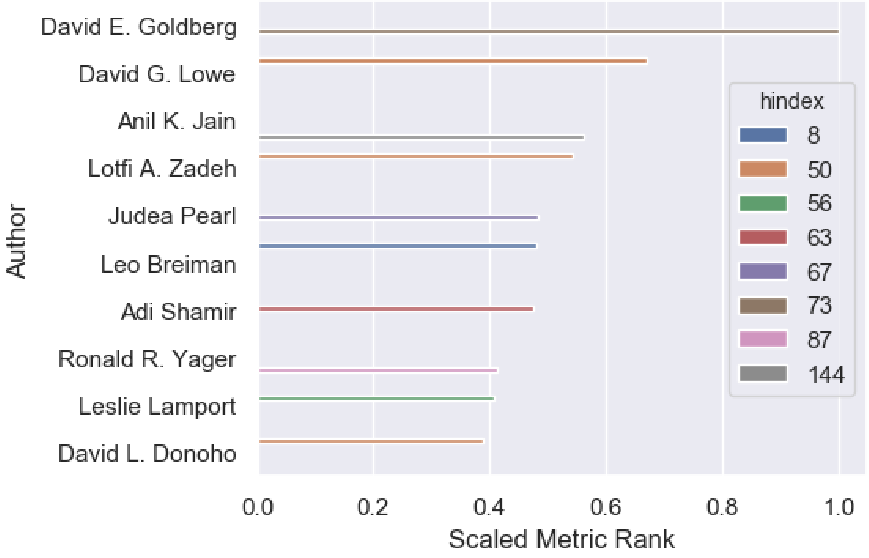

Figure 5: Comparison between internal h-index and modified page rank

Figure 5 is a plot between our modified page rank and the calculated h-index from our dataset on the top 10 authors of the modified page rank. For correlation, authors with higher h-index, like Ronald R. Yager (h-index of 87) should appear higher up but it is observed that even authors with h-indexes of 8(in color blue) received a higher modified page rank score. As the modified page rank algorithm takes into consideration from where did an author receive his citations, hence the authors having a higher modified page rank with a less h-index means that the author is cited by other highly ranked authors. Modified page rank takes into account the quality of the citations as well.

4 Conclusion

- Before this project we had an assumption that google scholar's h-index was the best metric for ranking researchers as it is used widely. During the course of this project, we learned about various metrics and techniques for ranking the research papers and authors.
- By developing our own metric we were able to factor in some of the disadvantages of h-index and come up with good ranking metric.
- In the future, an unsupervised learning algorithm can be devised which would rank all the papers using the features mentioned in the Machine Learning algorithm section.

References

[1] Citation network dataset. https://aminer.org/citation.

[2] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[3] A. P. Singh, K. Shubhankar, and V. Pudi, "An efficient algorithm for ranking research papers based on citation network," in *Data Mining and Optimization (DMO), 2011 3rd Conference on.* IEEE, 2011, pp. 88–95.

[4] "Topic modeling." [Online]. Available: https://www.analyticsvidhya.com/blog/2016/08/beginners-guide-to-topic-modeling-in-python/

[5] "Centrality." [Online]. Available: https://networkx.github.io/documentation/stable/reference/algorithms/centrality.html

[6] J. E. Hirsch, "An index to quantify an individual's scientific research output," *Proceedings of the National academy of Sciences.*