Ranking Papers using PageRank Algorithm (August 2023)

Anuj Rayamajhi[1], UG, Aayush Regmi[1], UG.
[1]Institute of Engineering Thapathali Campus, Tribhuvan University, Kathmandu, Nepal

ABSTRACT In an era where the deluge of digital information poses challenges for effectively navigating scholarly landscapes, this project sets out to harness the transformative potential of the PageRank algorithm. Originally designed to empower web searches by ranking pages based on importance, this project reimagines its application to revolutionize the evaluation and curation of research papers. By treating research papers as interconnected nodes within a vast academic network, with citations and references as the connecting threads, the algorithm offers a dynamic new approach to gauging scholarly influence. The project's goal is to unlock a more insightful and efficient means of uncovering impactful works, ultimately reshaping the way academia discovers and values knowledge.

KEYWORDS PageRank algorithm, Research paper ranking, Academic impact, Knowledge curation, Scholarly network, Citation analysis, Digital information, Web ranking, Research discovery, etc.

I. INTRODUCTION

In the digital age, where the flow of information is abundant and knowledge is readily accessible, the challenge of effectively navigating the vast sea of research materials has become both a blessing and a predicament. The digital landscape is teeming with an overwhelming assortment of research papers, spanning virtually every conceivable subject. Within this dynamic realm of academia, researchers, students, and educators alike find themselves confronted with an inexhaustible deluge of scholarly contributions. In this pursuit of invaluable insights, the need to efficiently rank and sift through these contributions has risen to the forefront of academic priorities.

Drawing inspiration from the algorithms that underpin search engines and web ranking, a pioneering concept has emerged - the notion of ranking research papers. This innovative approach seeks to adapt and apply the PageRank algorithm, originally devised to prioritize web pages based on their significance and influence, to the realm of academic research. The PageRank algorithm's ingenious framework offers a novel perspective on evaluating the worth of research papers, thereby equipping individuals with the means to unearth the most impactful and pertinent works within their spheres of interest.

The exponential growth of digital information repositories has led to an unprecedented influx of research papers, often resulting in an information overload. As a result, researchers, students, and educators face a daunting challenge: how to efficiently navigate through this veritable avalanche of knowledge to identify the most relevant and influential research. Traditional methods of sorting and categorizing papers based solely on publication dates or journal rankings have proven inadequate in effectively capturing the essence of a paper's impact and contribution.

To address this pressing need, a transformative approach was required. The introduction of the PageRank algorithm marked a pivotal moment in the quest for a solution. Originally designed by Larry Page and Sergey Brin as part of their groundbreaking work on Google's search engine, PageRank revolutionized how web pages were ranked based on their interconnectedness and importance. This concept of quantifying importance by examining a page's relationships rather than just its content laid the foundation for a paradigm shift that extended far beyond the realm of web search.

At its core, the PageRank algorithm centers on the premise that the significance of a web page is not solely determined by its content, but also by the links it receives from other important pages. The more incoming links a page has from reputable and influential sources, the higher its PageRank score, signifying its importance in the web's intricate network.

Transposing this notion to academia, the PageRank algorithm has been adapted to assess the impact and relevance of research papers. In this context, research papers are viewed as interconnected nodes in a vast academic network, linked by

citations and references. The more citations a paper garners from other highly cited papers, the greater its academic influence and relevance. By considering not just the intrinsic content of a paper, but also its relationships with other papers, the algorithm provides a nuanced perspective on the scholarly impact of each work.

The application of the PageRank algorithm in academia marks a significant departure from conventional methods of evaluating research papers. Rather than relying solely on journal prestige or author reputation, this approach takes into account the collective judgment of the academic community, reflected through citations. Papers that contribute substantively to a field and are frequently cited are accorded higher importance, much like web pages that receive numerous authoritative links.

The result is a more dynamic and democratic method of assessing the significance of research contributions. This approach empowers researchers, students, and educators to uncover the most influential and germane works within their domains of interest, fostering a deeper understanding of the evolving landscape of knowledge.

II. METHODOLOGY

A. THEORY

Finding relevant and valuable information efficiently is a challenge that has driven the evolution of search engines. At the heart of this evolution lies the PageRank, a revolutionary concept developed by Larry Page and Sergey Brin during their time at Stanford University in the late 1990s. Initially designed to rank web pages based on their significance and authority, PageRank not only transformed the way search engines organize and present search results but also laid the groundwork for understanding influence and importance within various interconnected systems, including the academic realm.

At its core, PageRank addresses a fundamental problem: how to sift through the seemingly infinite number of web pages and decide which ones deserve prominence in search results. Prior to PageRank, most search engines relied primarily on keyword frequency and placement to determine a web page's relevance. However, this method often resulted in poor user experiences, with less relevant or low-quality pages ranking higher than they deserved. PageRank introduced a novel approach that utilized the structure of the web itself as a key determinant of a page's importance.

The central idea of PageRank revolves around the concept of "link analysis." In essence, it posits that the importance of a web page can be inferred from the number and quality of links pointing to it from other pages. This is based on the premise that if a page is frequently linked to by other reputable pages, it is likely to be valuable and trustworthy. The concept is akin to how recommendations from friends or experts can influence our perception of the worthiness of something in the real world.

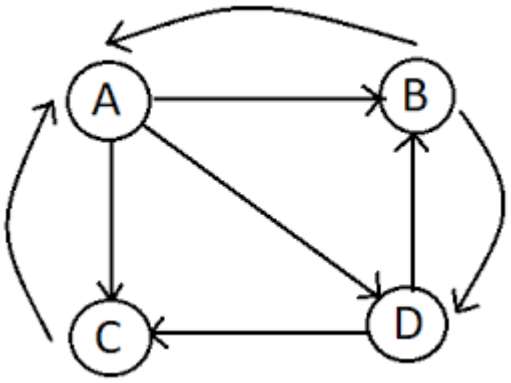

Figure 1 : A Simple Connection of Nodes

$$PR(A) = (1-d) + d * (\frac{PR(C)}{OUT(C)} + \frac{PR(B)}{OUT(B)})$$

This formula is the update rule for the node A. Here, PR(A) represents the page rank of node A, d is the damping factor and OUT(B) and OUT(C) represents the number of directed edges that start at B and C respectively. This formula can be generalized to the equation given below:

$$PR(A) = (1-d) + d * \sum_{t \in IN(A)} \frac{PR(t)}{OUT(t)}$$

Here, IN(A) represents the directed edges that end at A.

PageRank's mathematical formulation is elegant yet powerful. It treats the web as a vast interconnected graph, with web pages as nodes and hyperlinks as edges. Each link from one page to another can be seen as a "vote" or endorsement of the linked-to page's content. However, not all votes are equal. PageRank introduces the concept of "damping factor" – a probability that a user will follow a link – which adds a touch of realism to the model, recognizing that not everyone will click on every link they encounter.

The algorithm operates iteratively, with each web page being assigned an initial PageRank score. In each iteration, the PageRank score of a page is updated based on the sum of the PageRank scores of the pages linking to it, weighted by the importance of those linking pages. This recursive nature of the algorithm is what sets PageRank apart from simpler link-counting methods. It takes into account not only the direct links to a page but also the influence of pages that, in turn, are influential themselves. This propagation of influence through a network of links creates a dynamic system where the importance of a page can flow and accumulate from multiple sources.

While the application of PageRank to web ranking is well-known, its adaptation to the world of research paper ranking carries exciting implications. Imagine a vast network of

research papers, each citing and being cited by others. Just as web pages are linked, research papers are connected through citations, signifying their intellectual relationships. Applying PageRank to this network means that research papers with many citations from important and reputable papers will receive higher rankings. Additionally, the recursive nature of PageRank would ensure that papers cited by influential papers also receive a boost, accounting for indirect influence.

PageRank also introduces the concept of "eigenvector centrality," which measures a node's importance based on the importance of its neighbors. In the context of research paper ranking, this means that a paper's importance is not only influenced by the number of citations it has but also by the importance of the papers citing it. This mirrors the real-world academic landscape, where a paper's impact extends beyond its direct citations to influence the direction of subsequent research.

However, PageRank's adaptation to research paper ranking is not without challenges. The inherent differences between web pages and research papers require thoughtful adjustments. Notably, citation practices can vary widely across fields and may not always reflect the true importance of a paper. Additionally, the dynamic nature of academic research can lead to sudden shifts in influence, challenging the stability of ranking over time.

PageRank stands as a pioneering algorithm that reshaped how we think about ranking and importance in complex networks. Its application to web ranking revolutionized search engines, and its extension to research paper ranking offers a new perspective on assessing scholarly impact. By valuing the interconnectedness of information and the influence of key nodes within networks, PageRank continues to influence the way we navigate both the digital realm and the world of knowledge. While challenges remain in adapting PageRank to different contexts, its fundamental principles remain a guiding light in the quest to unravel the dynamics of influence and importance in our interconnected world.

B. MATHEMATICAL EXPLANATION

In their 1998 paper, Brin and Page gave the algorithm for finding the PageRank as:

$$PR(A) = (1-d) + d * \left[\frac{PR(T_1)}{L(T_1)} + \cdots + \frac{PR(T_{n)}}{L(T_n)}\right]$$

where they find the PageRank (PR) of page A, by taking the the PageRank of all pages that link to A, defined here as T, divided by the number of outgoing links on each page, defined as L. The parameter d is a dampening factor, which can be set from between 0 and 1, given as 0.85 in their paper. It is meant to simulate the number of links that a random surfer will follow before they go to a random, unlinked page.

Brin and Page (1998) also claimed that the PageRanks formed a probability distribution over all web pages, so that the sum of all of them would be 1. This is not the case with the algorithm they gave, so it must be modified as such:

$$PR(A) = \frac{(1-d)}{N} + d * \left[\frac{PR(T_1)}{L(T_1)} + \cdots + \frac{PR(T_{n)}}{L(T_n)}\right]$$

Where N is the total number of pages in the network. With this modification it then forms a probability distribution. By performing a number of iterations of the algorithm, the PageRanks of all pages in the network can be determined.

$$PR(a_i) = \frac{(1-d)}{N} + d * \sum_{a_j \in G(a_i)} \frac{PR(a_j)}{L(a_j)}$$

Where ai is a webpage, and a_j is a page with an outgoing link to a_i.

DANGLING NODES

Dangling nodes is a term used to refer to web pages that have no outgoing links. These pages are also sometimes called "dead-end pages" or "sink nodes." Dangling nodes are significant because they pose a challenge to the original PageRank formulation, as they interrupt the flow of PageRank scores through the network of pages.

In the PageRank algorithm, the calculation involves the transition probability matrix, where each element represents the probability of moving from one page to another through a hyperlink. When a page has no outgoing links, its corresponding column in the matrix contains all zeros, indicating that there's no way to move from that page to any other page in a single click.

This creates a problem because the algorithm would get "stuck" at these dangling nodes. The PageRank flow wouldn't distribute further, leading to a non-converging situation. In mathematical terms, the PageRank matrix becomes "stochastic irreducible," which means it's not guaranteed to converge to a stable solution.

To address this issue, adjustments are made to the original PageRank algorithm to ensure that the PageRank values can still flow through dangling nodes and across the entire network. One common approach is to redistribute the PageRank score of dangling nodes evenly among all pages. This way, even if the algorithm encounters a dangling node, the PageRank values continue to circulate through the network.

Here's how the adjustment works:

- Calculate the total PageRank score of all dangling nodes.
- Redistribute this total score evenly among all pages, including the dangling nodes themselves.

By doing this, the PageRank scores flow smoothly, and the algorithm can converge to a stable solution. This modification ensures that the algorithm doesn't get trapped at dangling

nodes, and it also reflects the principle that every page should have some nonzero probability of being reached from any other page. Handling dangling nodes is a critical aspect of refining the PageRank algorithm to make it more robust and applicable to real-world scenarios where web pages may have varying link structures.

C. SYSTEM BLOCK DIAGRAM

The system block diagram consists of multiple blocks which is responsible for their respective functionalities and which makes comprehending the process easier.

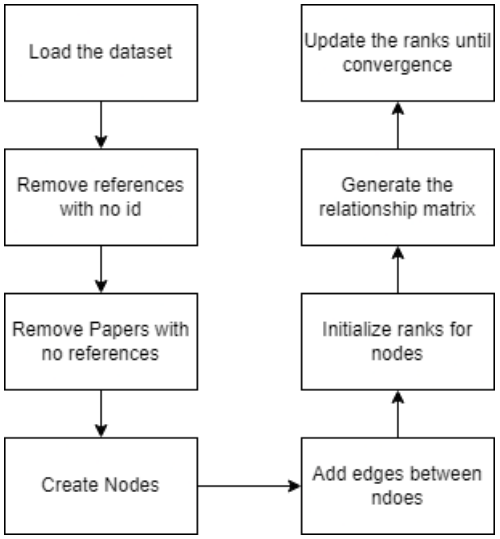

Figure 2: System Block Diagram

These are the stepwise operations that is performed to calculated the rank for the research papers.

1. **Load the dataset:** The dataset was scattered over various files. The data from these different files were load and compiled into a single file.
2. **Removing references:** There were many reference papers on the dataset, that were missing arXiv id. In this study, the arXiv was used to uniquely identify the research papers. So, the reference papers missing the arXiv id were removed from the dataset.
3. **Removing papers:** After removing the reference papers, there were many papers with no reference papers left. So, next up all these papers were removed.
4. **Create Nodes:** Using the remaining paper, a graph was created. Every node represents a research paper. These nodes contain every information about their corresponding research papers.
5. **Add edges**: Directed edges were added from the paper to their references. The edges start from the original papers and were directed towards their references. The subset of paper was created in such a way that each of the nodes have at least one or more edges ending or starting on them.
6. **Initializing ranks**: After the graph has been created. The ranks are initialized for each of the node. As PageRank is an iterative process, it converges no matter of the initialization. So, all of the ranks are initialized to 1.
7. **Generation of relationship matrix:** After the ranks are initialized, a relationship matrix is generated. As its name suggests, the relationship matrix represents the relation between different nodes.

$$r_{ij} = \begin{cases} \frac{1}{N_i}, if\ j\ is\ a\ reference\ of\ i \\ 0, othersise \end{cases}$$

Where, r_{ij} is the value of i^{th} column and j^{th} row, and N_i is the total number of references of i^{th} paper.

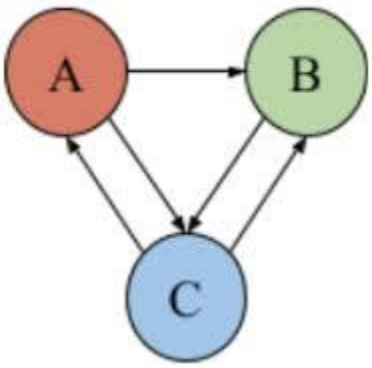

Figure 3: Sample Nodes Connection

For the above graph, the relationship matrix will be:

$$R = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Where, the 1^{st} column represents the outward link between node A and nodes B and C. Our ward link meaning, the directed edge begins from A and ends at other nodes. The value of r_{13} is 1 as there is a directed edge that starts at node A and ends at node C. Similarly, the 2^{nd} and 3^{rd} columns represent the outward links of node B and node C respectively. While the columns represent the outward links, the rows represent the inward link. The value at r_{32} is 1 as there is a directed graph that starts at node C and ends at node A. Here, the diagonal values are 0, but they are not necessarily 0. They might be one if there is a that edge begins and end at the same node.

In our case the diagonal values are always zeros, as a research paper cannot reference itself.

8. **Updating ranks**: After the ranks and the relationship matrix is ready. The ranks are updated iteratively until, the ranks converge and attain a single value.

III. Exploratory Data Analysis

This study implements the PageRank algorithm to rank research papers that are available on arXiv. arXiv is a preprint repository and distribution platform for research papers across various fields of science, mathematics, computer science, and related disciplines. The dataset that is used for this study is known as the unarXiv dataset. The unarXive data set contains

- One million papers in plain text
- 63 million citation contexts
- 39 million reference strings
- A citation network of 16 million connections

The data is generated from all LaTeX sources on arXiv from 1991–2020/07.

A small subset of the original UnarXiv dataset is used for this study. The subset contained about 19241 research papers, which was further narrowed down to 1798 after removing the papers that were not present on arXiv or had no ids to uniquely identify them.

The reducted dataset that is used in this study contained about 1136 research papers with total number of 1527 reference links. The reference count may seem quite low, this is because many of the reference papers were not present on arXiv, so, they were removed to avoid confusion.

IV. Results:

After the steps mentioned above were followed a rank for each of the nodes were obtained:

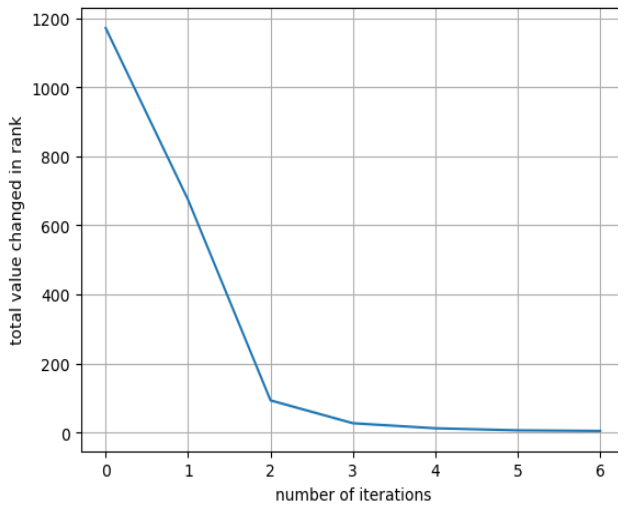

Figure 2 : Change in PageRank values per iteration

The plot above represents the total change ranks after iterations. It is clearly visible that the change in ranks is very less after 5th iteration. So, we can say that the ranks converge after 5 iterations.

The ranks of the 5 papers along, their name and their citation counts are present in the table below:

TABLE I
PAPER AND THEIR PAGERANKS

Paper Title	Rank	Citation Count
Cosmological bounce from a deformed Heisenberg algebra	3.68556507	6
Correlations, Risk and Crisis: From Physiology to Finance	3.63631826	3
A Generalization of the Goldberg-Sachs Theorem and its Consequences	3.52599236	5
Weyl Tensor Classification in Four-dimensional Manifolds of All Signatures	3.51608559	5
Incompressible limit of the compressible magnetohydrodynamic equations with vanishing viscosity coefficients	3.42032616	7

From the table we can see, that having greater number of citations doesn't necessarily mean that it will have a higher rank. In fact, the top 10 papers that had the most citations were not even in the top 5 in terms of rank. This proves the effectiveness of PageRank algorithm in ranking the papers in terms of their importance rather than in terms of number of citation counts. The maximum rank as shown in the table belongs to Cosmological bounce from a deformed Heisenberg algebra with a value of 3.68556507. The dataset contained many papers that had no inward links so many of the papers and as a damping factor was set to 0.85 the minimum rank was 0.15 and there were quite a few papers with that rank.

V. DISCUSSION AND ANALYSIS

The implementation of PageRank for ranking research papers within the UnarXiv dataset elicits a multifaceted discussion. The algorithm's capacity to unearth latent connections among papers offers a comprehensive view of academic influence, addressing the long tail of research and fostering interdisciplinarity. However, the algorithm's uniform treatment of citations raises questions about its adaptability to diverse disciplines, potentially necessitating domain-specific refinements. While PageRank's dynamic nature aligns with evolving research trends, ethical considerations regarding reinforcement of existing biases remain. Future research avenues involve hybrid models merging citation-based metrics with network analysis and exploring semantic and co-authorship dimensions to enhance the algorithm's holistic assessment of scholarly impact.

The application of the PageRank algorithm to rank research papers within the UnarXiv dataset has yielded promising results and provided valuable insights into the world of academic literature. By adapting the PageRank framework, we have harnessed the inherent interconnectedness of scholarly works to create a novel approach for assessing the significance and impact of research papers.

The journey of employing PageRank to rank research papers has unveiled the power of network analysis in the academic domain. The algorithm's ability to consider not only direct citations but also the influence propagated through interconnected references has led to a more nuanced evaluation of scholarly influence. Through iterative calculations and convergence, we have successfully identified influential papers that might have otherwise been overshadowed by the limitations of traditional metrics.

The findings derived from applying PageRank to the UnarXiv dataset underscore the potential of this approach to enhance knowledge discovery and streamline information retrieval in the academic landscape. The methodology has opened avenues for identifying seminal works, influential authors, and emerging trends, enabling researchers, educators, and students to navigate the vast sea of academic papers more efficiently and effectively.

However, it is essential to acknowledge the challenges encountered during this endeavor. Adapting PageRank for research paper ranking required careful consideration of domain-specific factors, such as citation practices and the evolving nature of academic research. The heterogeneity of research fields within the UnarXiv dataset posed certain limitations, reminding us that context matters when applying algorithmic methodologies.

Looking ahead, the fusion of PageRank and research paper ranking holds immense promise for further refining the way we gauge the impact of scholarly contributions. Future research could delve deeper into optimizing the algorithm for specific academic disciplines, considering variations in citation practices and domain-specific characteristics. Additionally, exploring ways to incorporate factors beyond citations, such as co-authorship networks and semantic analysis, could lead to a more comprehensive and holistic assessment of research influence.

VI. CONCLUSION

As we conclude this exploration into the utilization of PageRank for ranking research papers within the UnarXiv dataset, it is evident that this fusion of algorithmic innovation and academic scholarship has the potential to reshape how we navigate, evaluate, and contribute to the ever-evolving landscape of scientific knowledge. By bridging the realms of network analysis and scholarly impact assessment, we continue to unlock new dimensions of discovery and insight in the pursuit of knowledge.

VII. REFERENCES

[1] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Stanford InfoLab.

[2] Ma, N., Guan, J., & Zhao, Y. (2008). Bringing PageRank to the citation analysis. Information Processing & Management, 44(2), 800-810.

[3] Fiala, D., Rousselot, F., & Ježek, K. (2008). PageRank for bibliographic networks. Scientometrics, 76(1), 135-158.

[4] Sayyadi, H., & Getoor, L. (2009). Futurerank: Ranking scientific articles by predicting their future PageRank. In SDM (pp. 533-544).

[5] Yan, E., Ding, Y., & Sugimoto, C. R. (2011). P-rank: An indicator measuring prestige in heterogeneous scholarly networks. Journal of the American Society for Information Science and Technology, 62(3), 467-477.

[6] Walker, D., Xie, H., Yan, K. K., & Maslov, S. (2007). Ranking scientific publications using a model of network traffic. Journal of Statistical Mechanics: Theory and Experiment, 2007(06), P06010.

ANUJ RAYAMAJHI is an ambitious individual currently in the final year of his Bachelor's degree program in Computer Engineering at the esteemed Institute of Engineering Thapathali Campus. With a keen interest in various fields such as Artificial Intelligence, Machine Learning, Data Science, Software Development, and Engineering, Anuj possesses a diverse range of skills and a thirst for knowledge. He constantly seeks out opportunities to enhance his understanding of these subjects through research, online courses, and practical experience.

Apart from his academic pursuits, Anuj has a passion for music, sports, and computer games. In his leisure time, he enjoys exploring different genres of music, engaging in sports activities to stay active, and immersing himself in the virtual worlds of computer games.

With a strong foundation in computer engineering and a multifaceted interest in emerging technologies, Anuj is driven to make significant contributions in the fields of AI, machine learning, and data science. He strives to stay up-to-date with the latest advancements in these domains, continuously expanding his knowledge and honing his skills. Anuj's dedication, enthusiasm, and well-rounded interests make him a promising individual poised to excel in the ever-evolving field of technology

AAYUSH REGMI is a dynamic individual currently pursuing his Bachelor's degree in Computer Engineering at the renowned Institute of Engineering Thapathali Campus. With a strong inclination towards technology, Aayush has developed a keen interest in a wide range of fields, including Artificial Intelligence, Machine Learning, Data Science, Web Development, Quantum Computing, and more. His diverse expertise reflects his commitment to exploring various facets of computer science and pushing the boundaries of innovation.

In addition to his academic pursuits, Aayush finds solace and joy in his hobbies. As an avid sports enthusiast, he actively engages in cricket and football, relishing the thrill of competition and teamwork. Aayush also has a creative side and enjoys playing musical instruments, finding harmony in the melodies he creates.

With an ever-curious mind and a passion for learning, Aayush strives to stay ahead in the rapidly evolving world of technology. He is particularly intrigued by the emerging field of Quantum Computing and its potential to revolutionize the computing landscape. Aayush's dedication, coupled with his diverse skill set, positions him to make significant contributions in web development, quantum computing, and other innovative domains.

Driven by a relentless pursuit of knowledge and a desire to create a positive impact, Aayush is determined to shape the future through his expertise in computer engineering. His enthusiasm, adaptability, and holistic interests make him a promising individual poised to excel in the dynamic and ever-expanding field of technology.

APPENDIX A: CODE

```
import numpy as np
import json
import os


# getting all the jsonl files path
path = '.\data'
data_path  = []
for file in os.listdir(path):
    new_path = os.path.join(path, file)
    for data_file in os.listdir(new_path):
        if data_file.endswith('jsonl'):
            data_path.append(os.path.join(n
ew_path, data_file))


# creating a dictionary that maps paper_id
to paper_name and another dictionary that
maps paper to its references
paper_names = {}
paper_references = {}

for file_path in data_path[:2000]:
    file = open(file_path)
    for j, line in
enumerate(file.readlines()):

        data = json.loads(line)

        paper_name = '
'.join(data['metadata']['title'].split())
        paper_id = data['paper_id']

        paper_names[paper_id] = paper_name

        # reference_ids = [id for id in
[i['ids']['arxiv_id'] for i in
data['bib_entries'].values()] if i]

        ref_id = []
        for references in
data['bib_entries'].values():

            if 'ids' in references.keys():
                #
print(references['bib_entry_raw'],
references['ids']['arxiv_id'],
references['ids']['arxiv_id'])
                id =
references['ids']['arxiv_id']
                ref_id.append(id)

                # try:
                #     float(id)
                # except:
                #     continue

        ref_id = [id for id in ref_id if
id]

        if len(ref_id) > 0:
            paper_references[paper_id] =
ref_id


len(paper_references)


# count the number of references. Only
those references present in paper_names are
counted
ref_count = {}
not_in_dict = {}
for ref_list in paper_references.values():
    for ref in ref_list:
        if ref in ref_count:
            ref_count[ref] += 1
        elif ref in paper_names.keys():
            ref_count[ref] = 1
        else:
            if ref in not_in_dict:
                not_in_dict[ref] += 1
            else:
                not_in_dict[ref] = 1


# removing the references other than those
present in paper_names
for k, v in paper_references.items():
    paper_references[k] = [value for value
in v if value in ref_count.keys()]
```

```
# removing the papers with 0 references
new_ref_dict = {k:v for k,v in
paper_references.items() if len(v) > 0}
```

```
count = 0
for k, v in ref_count.items():
    count += v
print(count)
count = 0
for k, v in paper_references.items():
    count += len(v)
print(count)
count = 0
for k, v in new_ref_dict.items():
    count += len(v)
print(count)
```

```
len(paper_names), len(new_ref_dict),
len(ref_count)
```

```
# creating the a new list of papers those
are present in the paper and reference list
new_paper_names = {}
for key, value in paper_names.items():
    if key in new_ref_dict.keys():
        new_paper_names[key] = value
    else:
        for refs in new_ref_dict.values():
            if key in refs:
                new_paper_names[key] =
value
                break

len(new_paper_names)
```

```
# checking if all the references are int
the new list
for k,v in ref_count.items():
    if k not in new_paper_names.keys():
        print(k)
```

```
# checking if all the original papers are
on the new list
for k, v in new_ref_dict.items():
    if k not in new_paper_names.keys():
        print(k)
```

```
# checking papers that are present just as
a reference
for k, v in new_ref_dict.items():
    for refs in v:
        if ref not in new_ref_dict.keys():
            print(refs)
```

```
# checking if original papers are present
as a reference
for k, v in new_ref_dict.items():
    if k in ref_count.keys():
        print(k)
```

```
class Node:
    node_count = 0
    index_look_up_dict = {}

    @staticmethod
    def reset():
        Node.node_count = 0
        Node.index_look_up_dict = {}

    def __init__(self, paper_id,
paper_name):
        self.node_index = Node.node_count
        self.paper_id = paper_id
        self.paper_name = paper_name
        Node.node_count += 1
        self.rank = 0
        self.out_edges = []
        self.in_edges = []
        Node.index_look_up_dict[paper_id] =
self.node_index


    def add_edge(self, edge, out=True):
        if out:
            self.out_edges.append(edge)
```

```
        else:
            self.in_edges.append(edge)

class Edge:
    edge_count = 0

    @staticmethod
    def reset():
        Edge.edge_count = 0

    def __init__(self, node_from: Node,
node_to: Node):
        self.node_from = node_from
        self.node_to = node_to
        self.weight = 0
        self.edge_index = Edge.edge_count
        Edge.edge_count += 1

        node_from.add_edge(edge=self,
out=True)
        node_to.add_edge(edge=self,
out=False)


class Graph:

    def __init__(self):
        self.node_list = []
        self.edge_list = []
        Node.reset()
        Edge.reset()

    def create_node(self, paper_id,
paper_name):

        '''
        creates new node on the graph, does
not check if the node already exists or not

        paper_id: arxiv id of the paper,
        paper_name: name of the paper

        returns the created node
        '''
        node = Node(paper_id, paper_name)
        self.node_list.append(node)
        return node

    def add_edge(self, from_paper_id,
to_paper_id):
        '''
        add edge from one paper to another
        indicates that one paper is
reference of another paper

        from_paper_id: the id of the
original paper
        to_paper_id: id of the reference
paper



        returns the edge
        '''

        from_node_index =
Node.index_look_up_dict[from_paper_id]
        to_node_index =
Node.index_look_up_dict[to_paper_id]

        to_node =
self.node_list[to_node_index]
        from_node =
self.node_list[from_node_index]

        edge = Edge(from_node, to_node)
        self.edge_list.append(edge)
        return edge

    def print_edges(self):
        for edge in self.edge_list:
            print(f"from:
{edge.node_from.node_index}, to:
{edge.node_to.node_index}")

    def assign_ranks(self, ranks):
        for index, rank in
enumerate(ranks):
            self.node_list[index].rank =
rank

    def get_nodes(self, sorted=True,
ascending=True):
```

```
        temp = []

        for node in self.node_list:
            temp.append([node.paper_id,
node.paper_name, node.rank])

        temp.sort(key=lambda x: x[2],
reverse=not ascending)

        return temp

    def get_node_rank(self, paper_id):
        paper_index =
Node.index_look_up_dict[paper_id]

        node = self.node_list[paper_index]
        return [node.paper_id,
node.paper_name, node.rank]


class PageRank:
    def __init__(self, graph: Graph):
        self.graph = graph

        # setting the initial rank of all
the nodes to 1
        self.rank_matrix =
np.ones((len(graph.node_list), 1))

        # getting the update relationship
        self.relationship_matrix =
np.zeros((len(graph.node_list),
len(graph.node_list)))
        for edge in graph.edge_list:
            i = edge.node_from.node_index
            j = edge.node_to.node_index
            self.relationship_matrix[j, i]
= 1
        self.relationship_matrix =
self.relationship_matrix /
(self.relationship_matrix.sum(axis=0) +
0.0001)

    def update_rank(self, iter, d=0.85):
        self.rank_cache = []
        self.change_cache = []
        for i in range(iter):
            self.rank_cache.append(self.ran
k_matrix)
            self.rank_matrix = (1 - d) + d
* np.matmul(self.relationship_matrix,
self.rank_matrix)
            self.change_cache.append(abs((s
elf.rank_cache[-1] -
self.rank_matrix)).sum())

        self.change_cache =
np.array(self.change_cache)
        self.rank_cache =
np.array(self.rank_cache)
        graph.assign_ranks(self.rank_matrix
)

    def get_nodes(self, ascending=True):
        return
graph.get_nodes(ascending=ascending)

    def get_rank(self, paper_id):
        return
graph.get_node_rank(paper_id)

# creating graph and adding every paper as
a node
graph = Graph()
for paper_id, paper_name in
new_paper_names.items():
    graph.create_node(paper_id, paper_name)

for from_id, references in
new_ref_dict.items():
    for to_id in references:
        graph.add_edge(from_id, to_id)


pg = PageRank(graph)
pg.relationship_matrix.sum()



pg.update_rank(100)
```

```
index = 8
print(abs((pg.rank_cache[index] -
pg.rank_cache[index+1])).sum())
```

```
import matplotlib.pyplot as plt
%matplotlib inline
```

```
plt.plot(pg.change_cache[:7])
plt.grid()
plt.ylabel("total value changed in rank")
plt.xlabel('number of iterations')
plt.show()
```

```
pg.get_nodes(ascending=False)
```

['0805.1178',
 'Cosmological bounce from a deformed Heisenberg algebra',
 array([3.68556507])],
['0905.0129',
 'Correlations, Risk and Crisis: From Physiology to Finance',
 array([3.63631826])],
['1205.4666',
 'A Generalization of the Goldberg-Sachs Theorem and its Consequences',
 array([3.52599236])],
['1204.5133',
 'Weyl Tensor Classification in Four-dimensional Manifolds of All Signatures',
 array([3.51608559])],
['0905.3937',
 'Incompressible limit of the compressible magnetohydrodynamic equations with vanishing viscosity coefficients',
 array([3.42032616])],
['0812.3755',
 'Modification of Heisenberg uncertainty relations in non-commutative Snyder space-time geometry',
 array([3.40990462])],