Scholar Team Finder - Link Prediction Model for Identifying Scholars in Academic Social Networks

A Thesis

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Harsh Joshi

Dr. Prasad Calyam, Thesis Supervisor

May 2023

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled.

***Scholar Team Finder* - Link Prediction Model for Identifying Scholars in Academic Social Networks**

Presented by Harsh Joshi,

A candidate for the degree of Master Science

And hereby certify that, in their opinion, it is worthy of acceptance.

_____

Dr. Prasad Calyam

_____

Dr.Ilker Ersoy

_____

Dr. Filiz Bunyak

ACKNOWLEDGEMENT

When I began to attend the program of this master degree, I clearly remembered that it would be difficult, but interesting. Looking back to these two and half years for graduate studies, the facts proved that it was a hard but meaningful journey. In fact, I did not only learn a lot from courses, but also obtain good experiences, life meaning and unique memory with professors, students, lab members and friends who I made around these years.

Therefore, I would like to immensely thank Prof. Prasad Calyam for being my thesis advisor and mentor who provided me the best guides during my thesis research. Meanwhile, I also thank respectful Prof. Prasad Calyam for supplying me the opportunity to be a member of CERI Center and VIMAN Lab family and be part of his numerous, challenging research projects in cloud computing, machine learning and web development. I am extremely grateful for his constant guidance, insight, encouragement, and wisdom throughout my program. In fact, without his support, I cannot complete my thesis at all. In addition, I would like to express my gratitude and the most heartfelt thanks to Dr. Ilker Ersoy and Dr. Filiz Bunyak for reviewing my thesis, providing lots of insightful and valuable comments and suggestions as well as being part of my thesis committee.

I would like to thank Dr.Grant Scott and Dr. Tim Heathcoat for the guidance in my thesis and reviewing my thesis. In addition, I am also very grateful to my colleagues in my lab. When I was in trouble, their support and suggestions were truly insightful and useful for me to make it, especially for our common works, projects, and targets. Special thanks to my partner and friend, Xiyao Cheng. Finally, I would like to thank my parents, friends, relatives Without their support and accompany, I probably would not have started my master program in Mizzou and probably cannot insist on completing the graduate studies.

Abstract

Collaborations between scholars from multiple fields are becoming more common in research tasks. However, identifying suitable co-workers can be a challenging and time-consuming process. In response to this, the authors propose a novel model called ScholarTeamFinder, which utilizes a knowledge graph to identify collaborators within an academic social network (ASN) for multi-disciplinary research tasks. The model uses graph-based deep learning to learn node embeddings from the knowledge graph and recommends a scholar team. The approach uses semantic text features to improve the link prediction for identifying a suitable team.

The authors evaluate the ScholarTeamFinder using large ASN datasets, including the NSF award dataset of federal grant awards, scholars' publication data, and two other widely used datasets. They also propose a beam-search algorithm for scholar team prediction based on the model. The results show that the ScholarTeamFinder outperforms state-of-the-art baseline models by approximately 15% across different datasets.

The ScholarTeamFinder project aims to improve the collaboration process for researchers by providing recommendations for potential collaborations with related scholars. However, the model has not been trained with a knowledge graph, which limits its functionality, usability, and accuracy. Future work includes rebuilding the model with a knowledge graph to better represent relationships between scholars, venues, and publications, expanding the recommendations to include publications, venues, and other useful points, combining user queries with data models, and integrating expanding data models with a science gateway for ease of use.

To ensure the accuracy of the model, data collection would involve gathering scholarly publications, conference proceedings, and related data sources to create a comprehensive knowledge graph that can be integrated with the existing model. Additionally, collecting data on user queries and interactions with the model will help ensure that it is effectively meeting the needs of its users.

In summary, the ScholarTeamFinder model addresses the challenge of identifying suitable collaborators for multi-disciplinary research tasks. It utilizes a knowledge graph and graph-based deep learning to recommend a scholar team. The model outperforms baseline models by approximately 15% across different datasets, and future work includes expanding the recommendations and integrating data models with a science gateway for ease of use.

Index Terms—heterogeneous knowledge graph, scholar team recommender, deep learning, node embedding, link prediction.

Table of Contents

1. Introduction

The world has experienced a surge in the amount of published information, which has led to the development of advanced search engines and recommendation systems to filter relevant information efficiently and effectively. These tools are widely used by individuals and organizations alike to sort through the overwhelming amount of data available and find what they need quickly and accurately. In the academic world, this trend is particularly prevalent, where research tasks involve finding relevant multi-disciplinary information and collaborating with experts in different fields to create new knowledge that transcends disciplinary boundaries. However, identifying potential collaborators can be a challenging task, especially when the research problem is multi-disciplinary. One of the main challenges in identifying scholar collaborators for a given multi-disciplinary research problem is that scholar profiles evolve dynamically over several years. This evolution can be influenced by various factors, including publications or research grants. As a result, manually identifying potential collaborators for a research project can be a time-consuming and tedious process. Additionally, given the sheer volume of published research and the constantly changing nature of the academic landscape, it can be difficult to keep track of who the relevant experts are in each field. To address these challenges, researchers and academics have turned to advanced tools such as data mining and machine learning algorithms. These tools can automatically identify potential collaborators based on factors such as publication history, research interests, and academic affiliations.[1] This approach can save significant amounts of time and effort in identifying potential collaborators, allowing them to focus on the actual research process. Data mining algorithms are used to extract patterns and relationships from large datasets and machine learning algorithms are used to build predictive models based on the data. These algorithms can be used to analyze the publication history of scholars and identify those who have published on similar topics. They can also be used to analyze the research interests of scholars and identify those who have shown an interest in related topics. [12] Moreover, algorithms can be used to analyze the academic affiliations of scholars and identify those who are affiliated with institutions that are known for their expertise in related fields. In conclusion, the explosion of information has made it essential to rely on advanced search engines and recommendation systems to filter relevant information efficiently and effectively. In academia, this trend is particularly prevalent, where research tasks require finding relevant experts for multi-disciplinary knowledge creation. However, identifying potential collaborators is a challenging task that can be addressed using advanced tools such as data mining and machine learning algorithms. These tools can save researchers significant amounts of time and effort in identifying potential collaborators, allowing them to focus on the actual research process. Consequently, researchers are increasingly relying on these advanced tools of recommendation system to identify potential collaborators and streamline the research process efficiently and effectively.

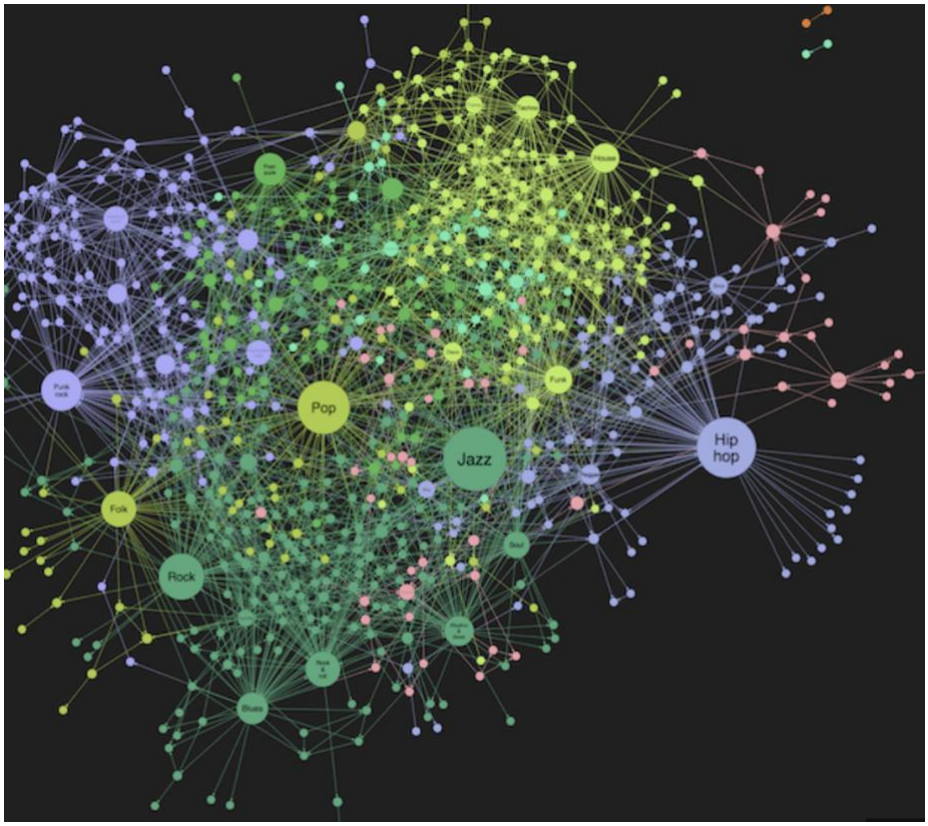


Fig.1 - Knowledge Graph Example

2. Motivation Towards Work

The field of research is rapidly advancing, thanks to the recent advancements in technology, which has led to an abundance of data. The availability of such data has inspired researchers to delve deeper into their work. However, this abundance of data also presents certain challenges for researchers. Despite the vast availability of data on the internet, researchers often struggle to find relevant resources due to the sheer volume and disorder of the data. This can be a time-consuming and frustrating process, especially for researchers who would prefer to focus on their work. Another challenge is the need for cross-disciplinary research, which has made it difficult for researchers to find collaborators who possess the necessary knowledge and expertise. Traditionally, scholars have been recommended for collaboration based on their research areas, keywords, and citations. This involves manual search and selection of scholars based on their research interests, expertise, and past collaborations. With the advent of machine learning models, several traditional methods have been

automated to recommend scholars for collaboration. Some of these traditional methods using machine learning models are:

Content-based recommendation: This method uses the similarity between research papers, keywords, and abstracts to recommend scholars for collaboration.

Collaborative filtering: This method uses the past collaborations and co-authorship networks of scholars to recommend potential collaborators.

Citation-based recommendation: This method uses the citation patterns of scholars to identify potential collaborators based on their citation patterns and research interests.

Social network analysis: This method uses the social network analysis to identify potential collaborators based on the network of co-authors and research collaborators.

Hybrid methods: These methods combine two or more of the above methods to provide more accurate and relevant recommendations for scholars. A knowledge graph is a type of knowledge representation that represents entities and the relationships between them. It is designed to capture the complex relationships between entities in a way that is easily understandable. Knowledge graphs have become an essential tool in the field of machine learning, particularly in recommendation systems. [8] Graph Neural Networks (GNNs) have recently gained attention due to their ability to analyze graph structural data. [2] Many websites and data models have used knowledge graphs to train models that can recommend resources for users. These models have proven to be effective in solving the information overload problem in areas such as e-commerce, entertainment, and social media. However, there are still some areas where these models could be improved, such as recommending collaborators or venues. One limitation of current models is that they rely on homogeneous graphs, which have the same kind of node. This can be limiting, particularly in areas where there are multiple types of entities. To overcome this limitation, researchers are exploring the use of heterogeneous graphs to train models. Although this is a more complicated and difficult approach, it is a promising one.
Another area of interest is Query Generation and Knowledge Graph Question Answering. [3] User query is an important consideration for recommendation systems, as it can help users find the content, they are looking for more efficiently. Query generation can also help users find deeper related information. Therefore, incorporating query generation into data models is a promising direction. Graphs and GNNs are promising tools for recommending resources to researchers, but there is still room for improvement. Heterogeneous graphs, Query Generation, and Knowledge Graph Question Answering are all promising directions for recommendation systems. Finally, creating a user-friendly interface is important for maximizing the potential of these tools. [10]

In recent studies, various approaches have been proposed to recommend individual scholars, with a focus on leveraging entity and relationship data in online academic communities, or Academic Social Networks (ASNs). For instance, one approach uses a knowledge graph-based model to recommend scholar-friends, while another approach proposes a personalized recommendation system using multi-dimensional features. [9]However, these prior works have limitations such as not including link labels as needed in a knowledge graph, relying on potentially inconsistent information from online academic community platforms, and not addressing the challenge of identifying scholar teams with multi-disciplinary expertise. [4] To address this, we define a scholar team as a research group consisting of two or more scholars from the same or different institutions, with collective expertise essential to solve a multi-disciplinary research problem. They aim to develop a system that recommends scholar teams, leveraging the strengths of knowledge graphs and addressing the limitations of prior models.

3. Research Aim

ScholarTeamFinder is a proposed model that aims to predict links between scholars with the help of knowledge graph embedding. To evaluate this model by comparing it with two state-of-the-art methods - deepwalk and metapath2vec. These models are trained using the same knowledge graph as ScholarTeamFinder. Deepwalk is a graph neural network that uses a randomized path traversing technique to learn the inner structure of the graph network. It has been applied in various fields with satisfactory performance. On the other hand, metapath2vec is a network embedding method that is suitable for a heterogeneous network. A scholar team that spans multiple fields, as illustrated in Fig. 2, there are different types of nodes, such as scholars, co-authors etc. , and different types of links between them, such as scholars working on the same proposals or publishing the same publications. The current practice of manually finding potential collaborators is limiting and does not allow for a data-driven approach to identifying relevant scholars. Academic Social Networks (ASNs) can provide access to information about hundreds of thousands of scholars and links between scholars, such as those who have already worked closely with each other or those within the same organization. However, there are cases where scholars do not inherently have commonalities, and tracking scholars and characterizing their latest research interests can be highly challenging, especially given the frequent changes in academic positions and interests. [11] The challenge of identifying suitable collaborators is a significant barrier to building effective and diverse scholar teams. While ASNs can provide some assistance in this regard, there is a need for more sophisticated data-driven approaches that can analyze the vast amounts of data available and identify potential collaborators based on their research interests, expertise, and collaborations.

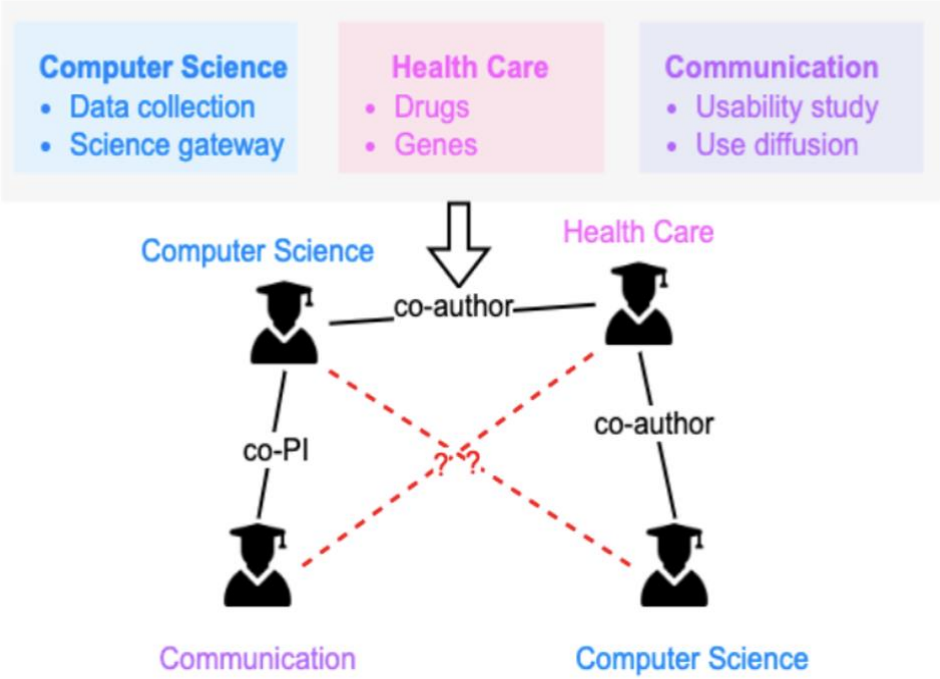


Fig. 2 - Example to show how a potential scholar team is identified to address a multi-disciplinary research problem.

4. Related Work

a. Preliminary Research

The ScholarTeamFinder project is focused on proposing a recommendation model that provides recommendations of related scholars to researchers. The purpose of this model is to enable researchers to identify potential collaborations and partnerships, facilitating and enhancing research activities. The current ScholarTeamFinder model, which is a deep generative model, has achieved good performance when compared to state-of-the-art baseline models such as Boost and DNN. However, this model is still not trained with a knowledge graph. This presents an opportunity to extend and improve the current model by incorporating a knowledge graph.

The process of enhancing the ScholarTeamFinder model through the incorporation of a knowledge graph and data integration is shown in Figure 2. The proposed model seeks to address current limitations in the model and improve its functionality, usability, and accuracy. Data collection for this research would involve gathering scholarly publications, conference proceedings, and related data sources to create a comprehensive knowledge

graph that can be integrated with the existing model. This would involve gathering data on scholarly publications, conferences, and other relevant scholarly activities.

Overall, the proposed enhancements to the ScholarTeamFinder model have the potential to greatly enhance its usefulness for researchers. By incorporating a knowledge graph and expanding the dimensions of the recommendations, the model can provide more comprehensive and accurate recommendations to researchers. The incorporation of user queries and integration with a science gateway and chatbot would further enhance the model's functionality and usability. Collecting relevant data sources and conducting user studies would be important steps in the preliminary research process for developing and enhancing the ScholarTeamFinder model.

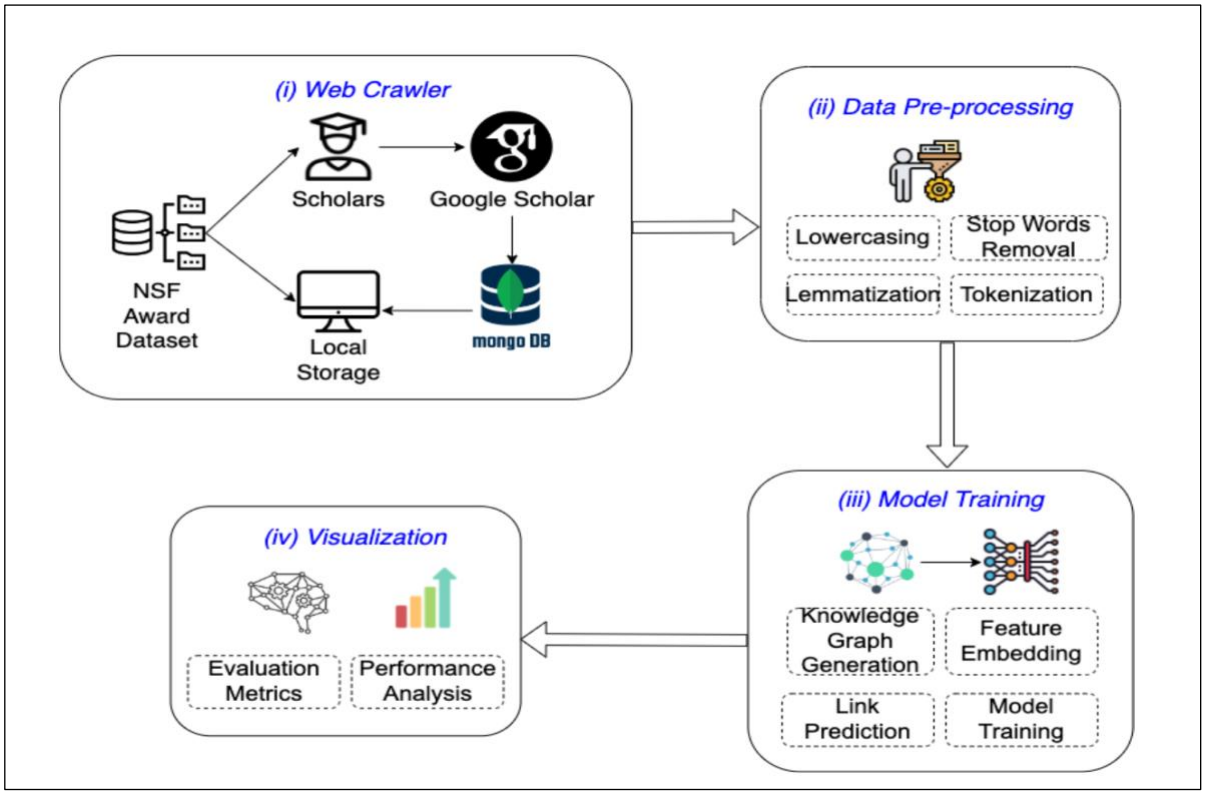


Fig. 3 - Scholar Team Finder Research Process

b. Representation Learning

Representation learning is a crucial aspect of machine learning on graphs, which finds applications in diverse areas such as drug design and social network recommendations [15]. In the training of machine learning models, creating a knowledge graph, or finding a way to represent it is a challenging task. The quality of the representation method and the structured knowledge graph can significantly impact the final model's performance. Representation learning is beneficial for researchers as it can extract relevant information from speech, text, and figures, thereby saving time in the feature engineering process.

Recent research focuses on representation learning in the context of knowledge graphs. Most studies concentrate on homogeneous knowledge graphs, such as DeepWalk [16], Line [17], and node2vec [18]. These models process specific types of nodes within the knowledge graph. However, with the vast amount of information available and its diversity, a homogeneous knowledge graph may not satisfy the actual network's requirements. Recent works, therefore, focus on heterogeneous knowledge graph methods. For instance, the authors of [19] designed a deep embedding algorithm for networked data, which captures complex interactions between heterogeneous data in a network. Similarly, the authors of [20] proposed a unified framework to solve the problem of embedding learning for an Attributed Multiplex Heterogeneous Network. In addition, [21] proposed two scalable representation learning models, metapath2vec and metapath2vec++, which can efficiently embed graph features and perform well in many data mining tasks.

Although these existing models can efficiently extract features from knowledge graphs, they do not consider the nodes' text features. To overcome this limitation, we extend the metapath2vec model by adding semantic features to embed our knowledge graph to perform scholar team recommendations.

c. Graph-based Recommendation

In recent years, graph-based recommendation systems have become increasingly popular due to their ability to model complex relationships between entities in various domains. Graph Neural Networks (GNNs) have played a vital role in advancing the field of recommendation systems by enabling the efficient learning of representations from structured data.GNNs are a class of deep learning models that operate directly on graphs, which are networks composed of nodes and edges. Nodes represent entities, and edges represent the relationships between them. GNNs are particularly effective at learning representations for structured data because they can incorporate both the structural information of the graph and the attributes of the nodes and edges. One area where GNNs

have been applied successfully is in generating recommendations. For example, in intent recommendation, the goal is to recommend the next item to a user based on their previous interactions. The authors of [25] propose a metapath-guided heterogeneous GNN model to learn the embedding of objects in intent recommendation.

The model adds the complex objects and rich interactions in intent recommendation as a Heterogeneous Information Network. Similarly, in [26], the HGNR model that uses a Graph Convolutional Network to learn the embedding of users and items based on a heterogeneous graph. In multi-relational recommendation, the goal is to predict various types of user behaviors, such as whether they will like or dislike an item, purchase it, or click on an ad. The authors of [24] propose a GHCF model for multi-relational recommendation, which can discover the relationship between users or users and items and shows multi-task ability to predict various types of user behaviors in one model. In scholar recommendation, the goal is to recommend scholars based on their expertise and research interests. Many researchers have applied graph-based recommendation methods to scholar recommendation. In [2], the authors propose a heterogeneous network-based approach to recommending scholar-friends with online academic communities. In [27], the ISRMACD model to provide recommendation service for scholars with low influence in academic social networks. In [28], the authors present a framework that can provide co-authorship strength, author contribution, and scholar search. In [29], the deep generative model to learn the scholars' representations using their publication data.
To address this limitation, the ScholarTeamFinder model, which is designed to recommend scholars to help build a high-quality research team. The model goes beyond prior state-of-the-art models by focusing on recommending a team of scholars rather than a single scholar. The model uses a graph-based approach to capture the complex relationships between scholars and their research interests. The ScholarTeamFinder model is based on the idea of knowledge graphs, which are networks that represent entities and the relationships between them. The model uses a knowledge graph to represent the scholarly data, where the nodes represent scholars, and the edges represent the relationships between them. The model is designed to learn the embeddings of the nodes in the knowledge graph, which capture the latent features of the scholars and their research interests. The model consists of two main components: the knowledge graph construction and the scholar team recommendation. In the knowledge graph construction phase, the model constructs a knowledge graph from the scholarly data by extracting the co-authorship and research topic information. The model then uses the extracted information to build a graph where the nodes represent scholars, and the edges represent co-authorship and research topic relationships.

5. Tools and Technologies Involved

a. Python

In the ScholarTeamFinder project, Python is used extensively to extract data from various sources such as academic journals, grant databases, and other scholarly sources. Python libraries such as Beautiful Soup and Scrapy are used to extract the data, while Pandas and NumPy are used for data manipulation and analysis. Python is also used for machine learning tasks in the ScholarTeamFinder project. The recommendation model, ScholarFinder, is a deep generative model that was built using machine leaning models.

Additionally, Python is used for data cleaning and filtering. Data cleaning involves removing unwanted characters, fixing spelling errors, and formatting data in a consistent manner. Data filtering involves selecting only relevant data and removing duplicates. Python provides powerful tools for these tasks, including regular expressions, string manipulation functions, and Pandas data filtering functions.

b. Database

The ScholarTeamFinder project required a database to store and manage large amounts of unstructured data. A NoSQL database was chosen as the data was very unstructured and inconsistent, and traditional SQL databases like PostgreSQL or MySQL would not be suitable. MongoDB was chosen as the NoSQL database because it provides fast extraction and insertion of data, is easy to set up, and is free to use initially.

MongoDB is a document-based database that stores data in JSON-like documents. It is highly scalable and can handle large amounts of unstructured data with ease. In the ScholarTeamFinder project, MongoDB was used to store scholar information, NSF award data, and other relevant data. MongoDB's document-based architecture allowed for easy querying and filtering of data, making it easier to extract meaningful insights from the data.

c. Cloud Server

The ScholarTeamFinder project required a cloud server to run Python scripts continuously for data extraction. This was necessary because the data was very large, and the extraction process could take several hours or even days to complete. Additionally, IP blocking was a significant issue in web scraping, and a cloud server could provide a new IP address after every restart. Amazon Web Services (AWS) Elastic Compute Cloud (EC2) instance was used to run the Python scripts continuously. AWS EC2 provides scalable computing capacity in the cloud, allowing users to run applications and services on virtual machines. The instance was configured to run the Python scripts continuously and restart automatically in case of failures. EC2 also provides an easy way to scale up or down the computing capacity as needed, making it ideal for handling large amounts of data.

d. VS Code Editor

VS Code was used to write and edit Python scripts, manage Git repositories, and debug the code. It also provides extensions for Python, MongoDB, and other libraries used in the project, making it easier to integrate with other tools.

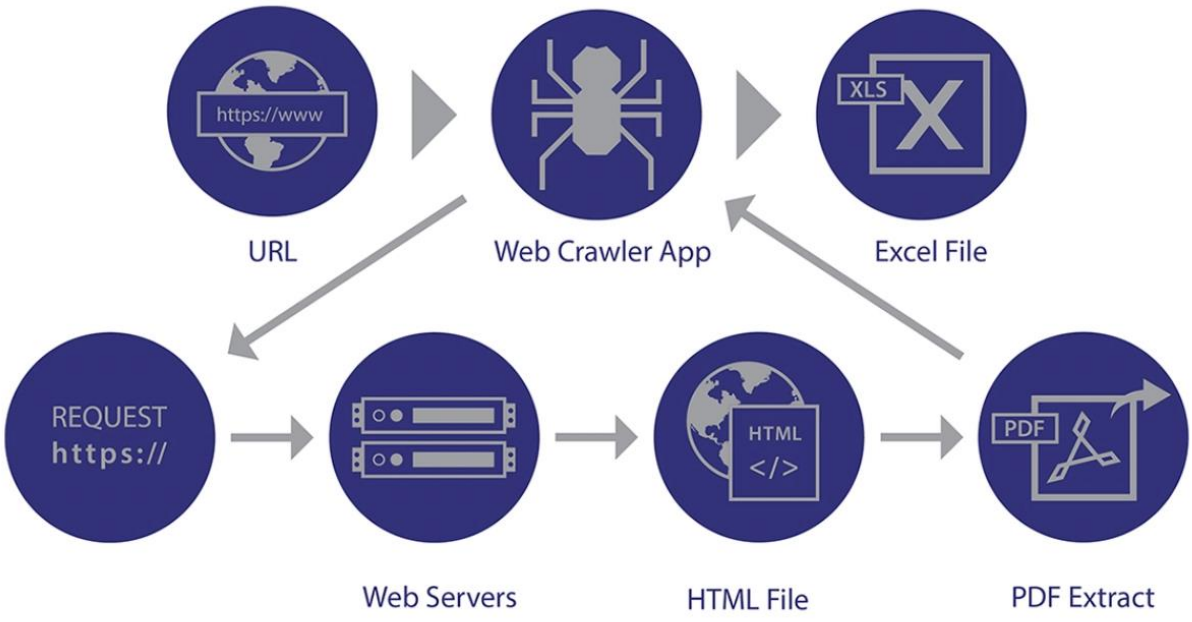


Fig.4- Process of Data Processing and Extraction from different sources with the help of python , vocode , AWS servers

e. Machine Learning Models Used and Score Used

DeepWalk: It is a graph embedding technique that learns low-dimensional representations of nodes in a network by modeling random walks on the network. DeepWalk generates node embeddings by treating random walks as sentences and using a language model to learn the embedding space. DeepWalk can be applied to both homogeneous and heterogeneous networks, and it has been shown to perform well on several real-world applications, such as social network analysis, recommendation systems, and bioinformatics [39].

Metapath2vec: It is a graph embedding technique that learns embeddings of nodes and edges in heterogeneous networks, considering the structural diversity of the network.

Metapath2vec generates node and edge embeddings by using a skip-gram model to predict the context of a given node or edge, considering the sequence of meta-paths connecting nodes. Meta-paths are paths that describe the structural semantics of the network, for example, a path connecting authors and papers in a bibliographic network. Metapath2vec has been shown to outperform other state-of-the-art embedding methods on several real-world applications, such as recommendation systems and gene function prediction[40].

ScholarTeamFinder Model: It is a link prediction model that uses collaborative relationships between scholars to predict potential collaborations between them, based on their learned embeddings. ScholarTeamFinder generates embeddings of scholars by considering their co-authorship, publication history, and topic similarity. The model predicts links between scholars by measuring the similarity of their embeddings in the learned embedding space. ScholarTeamFinder has been shown to perform well on the task of predicting potential collaborations between scholars.

HR@K: HR@K (Hit Rate at K) computes the proportion of test cases where the true positive item is ranked among the top K recommendations. For example, if we set K=10, HR@10 measures the percentage of test cases where the true positive item is among the top 10 recommendations. HR@50 and HR@100 measure the same for K=50 and K=100, respectively. The higher the HR@K score, the better the recommendation system's performance is. For instance, an HR@10 score of 0.5 means that, on average, half of the test cases have the true positive item among the top 10 recommendations.

AUC Score: The area Under the Curve score is a metric used to evaluate the performance of a binary classifier by calculating the area under the receiver operating characteristic (ROC) curve. The ROC curve plots the true positive rate against the false positive rate at various thresholds, and the AUC score represents the area under the curve. The AUC score ranges from 0.5 (random classifier) to 1 (perfect classifier). The AUC score is a commonly used metric in binary classification problems, including link prediction in network analysis.

NDCG@K (Normalized Discounted Cumulative Gain at K): Normalized Discounted Cumulative Gain at K (NDCG@K) is a ranking-based evaluation measure that considers the relevance and position of the recommended items in a list. It assigns higher scores to relevant items that are ranked higher in the list and discounts the scores of items that are ranked lower in the list. The measure is commonly used in information retrieval and recommender systems. To compute NDCG@K, the relevance of each recommended item is first determined. This can be done using a binary relevance measure, where an item is considered relevant if it meets a certain threshold of relevance, or using a graded relevance measure, where the relevance of an item is assigned a score between 0 and 1.

6. Data Collection and Techniques

a. Data Collection

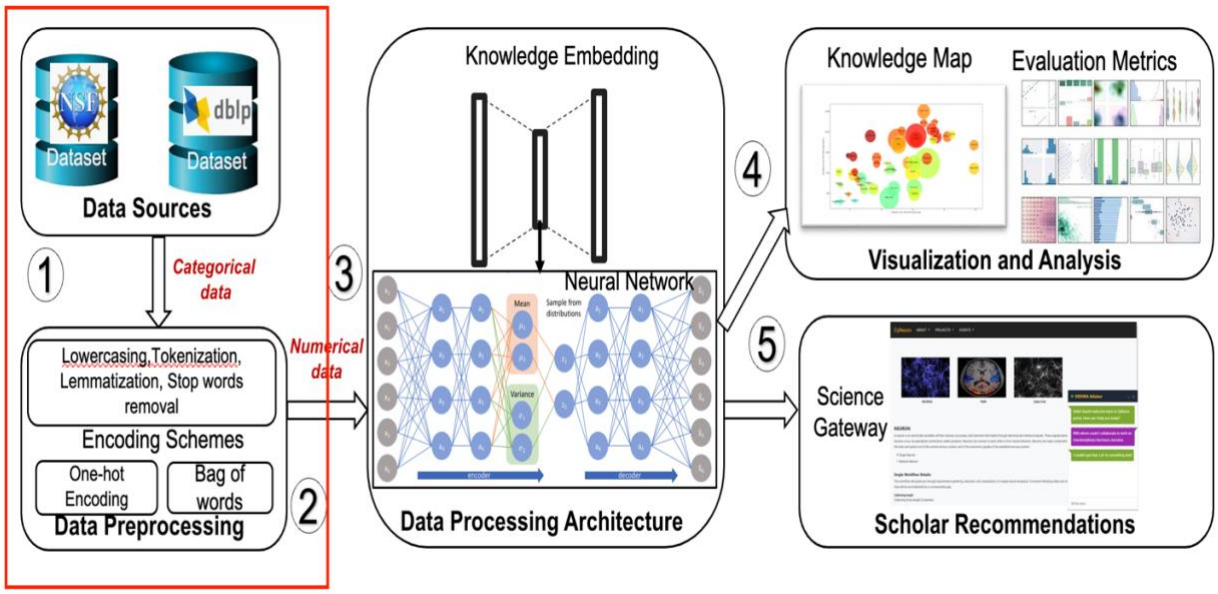


Fig.5 - ScholarTeamFinder Architechcutre

Google Scholar Data :

The Scholar Info component of the project contained all the relevant information related to the scholars, such as their publications, citations, h-index, name, email, etc. This information was collected from various sources, such as academic databases, research papers, and academic profiles of scholars. The Scholar Info was the backbone of the project, as it helped in the recommendation of related scholars to researchers for potential collaborations. The Scholar Info component was created by scraping the data from various academic databases, such as Google Scholar, Web of Science, and Scopus, and was stored in a database.
Example data shown in Fig.7.

Key Columns Name from Google Scholar Data –

'email', 'institute', 'name', 'scholar_id', 'status', 'affiliation', 'citedby', 'cites_per_year', 'coauthors', 'email_domain', 'google_scholar_id', 'hindex', 'homepage', 'i10index', 'interests', 'journal', 'journal_data', 'publications', 'div_id', 'div_name'

Data Source Link - Data Link

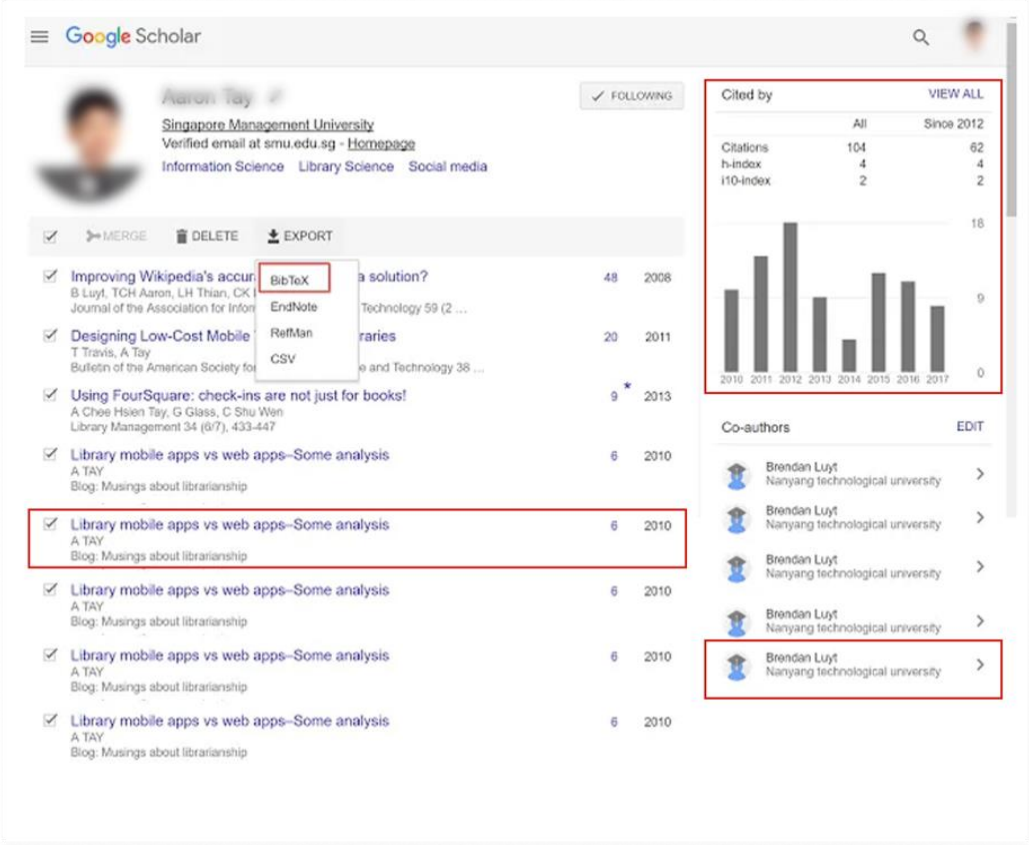


Fig.6 - Google Scholar data Demo from Google Website

Steps Involved in Scraping Scholar Data from Google –

1. First, we extracted the scholar names, emails, and university information from the NSF grant award data.
2. Google Scholar does not provide any official API or direct link to download scholar data, so we needed to write a Python script to fetch the data manually.
3. We initially tried to use web scraping libraries such as Selenium and Beautiful Soup, but we encountered issues with Google blocking robot access to the site.
4. To overcome this, we developed our own custom Python data scraping API that could fetch the data from Google Scholar. The API was hosted on an AWS EC2 instance and was set to run continuously to collect data.

5. We discovered that Google was blocking our server IP address, which was slowing down the data collection process. To overcome this issue, we used multiple servers and databases to distribute the scraping process and increase the speed of data collection.
6. We implemented a time limit for fetching data for each scholar and verified that the scholar's email domain matched their name and university information to ensure that we were collecting data for the correct scholar.
7. After fetching the raw data, we inserted it into a MongoDB database. MongoDB was used because it is a document-oriented NoSQL database that can store unstructured data as is, without the need for table creation or predefined schemas.
8. We continued to fine-tune our scraping process to overcome IP blocking issues and improve the speed of data collection.
9. After scraping, our database contained a total of 100,000 data records for scholars.

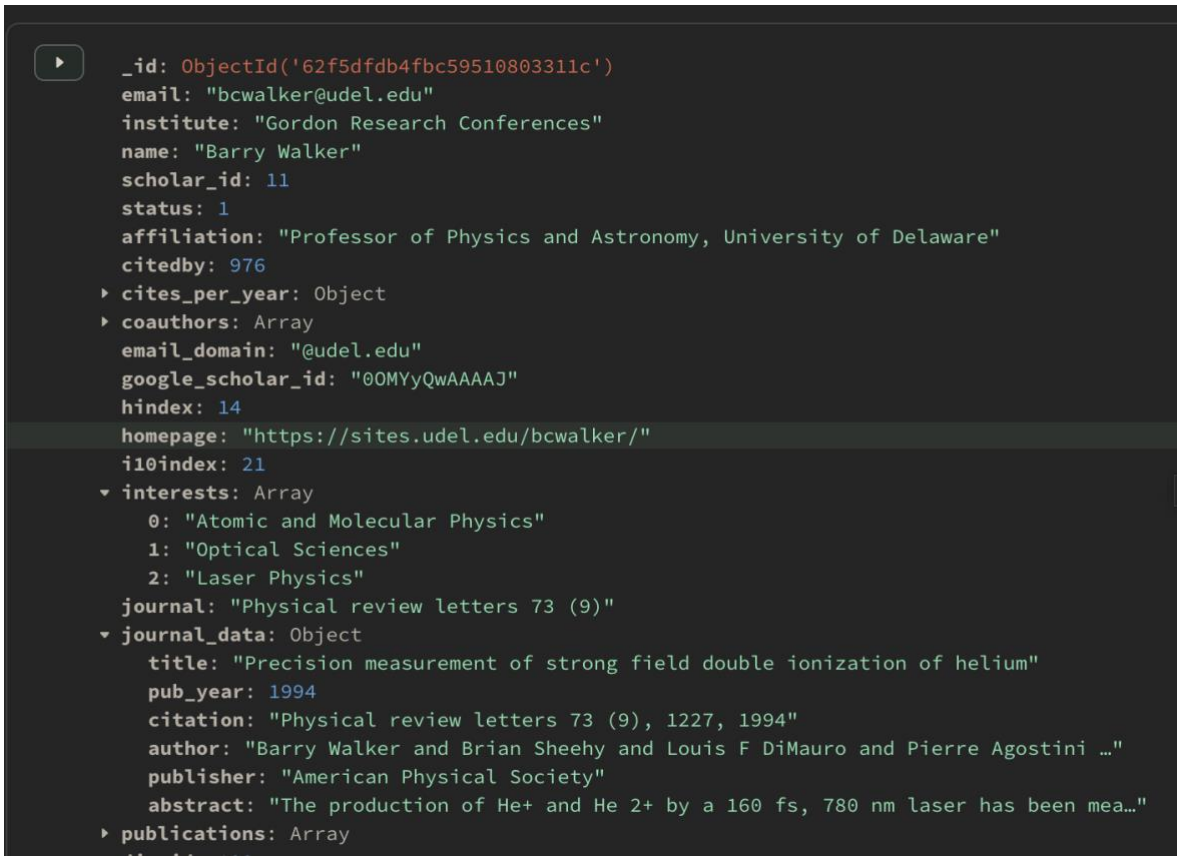


Fig.7 - Google Scholar Data stored in database

NSF Award Data :

The NSF Award Data component of the project contained information collected from the National Science Foundation (NSF) about scholar names, email addresses, and grant awards data. This information was used to identify the scholars who had received NSF grants and to recommend them to researchers for potential collaborations. The NSF Award Data component was created by scraping the data from the NSF website and was stored in a separate database.
Example data shown in Fig.8.

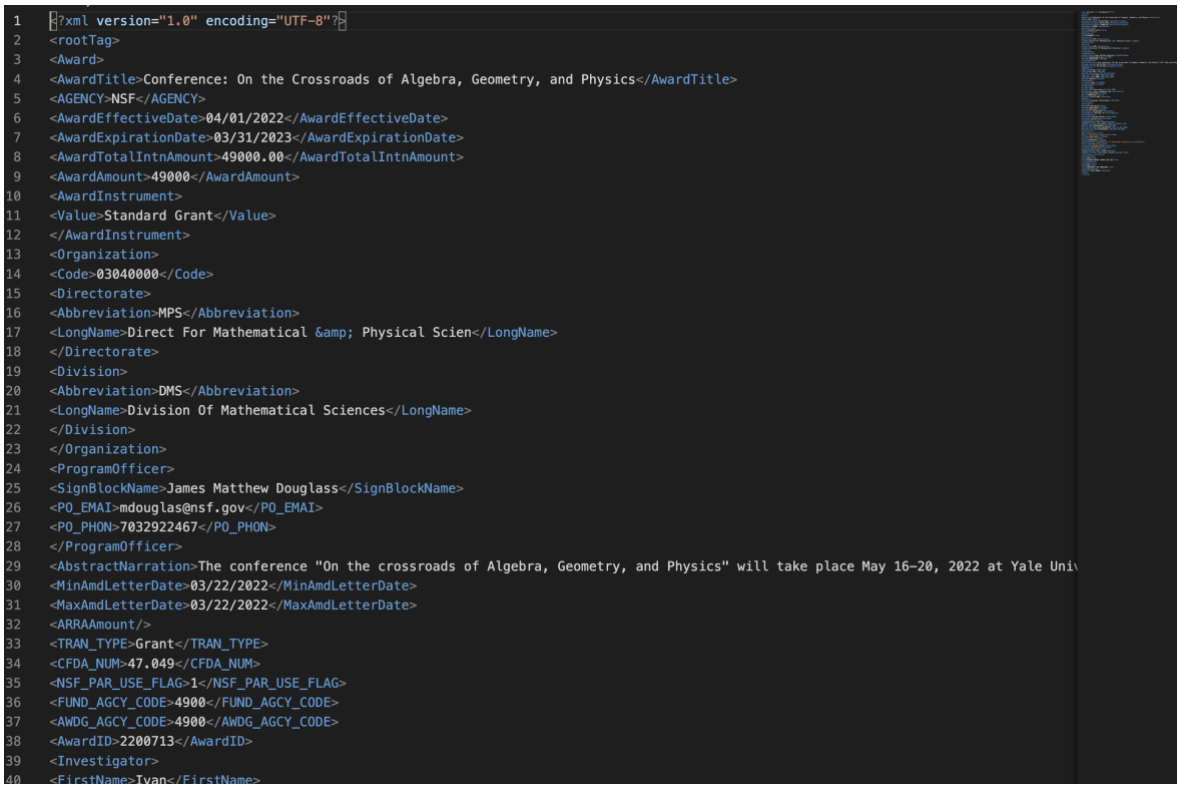


Fig.8 - NSF Raw Data in XML Files

Key Columns Name from NSF Award Data –

'awardid', 'division', 'email', 'name', 'div_id', 'div_name', 'grant' … Many More

Data Source Link - [Data Link]

Steps Involved in NSF Data Extraction -

1. The first step in the process was to download data from the NSF award site for the past 10 years. This data contained information on grant recipients, award amounts, institutions, and other relevant details.
2. The data was downloaded in multiple folders and in XML files, which were then extracted from each folder.
   Next, the XML files were parsed, and the relevant data ('awardid', 'division', 'email', 'name', 'div_id', 'div_name', 'grant') was extracted and stored into the database. This involved using an XML parser to extract information such as grant number, recipient name, institution name, award amount, and other details.
3. After storing the data in the database, the total volume of data was more than 2 million rows. This database contained all the relevant information related to NSF grant recipients and was used to identify scholars who had received NSF grants for potential collaborations.
4.

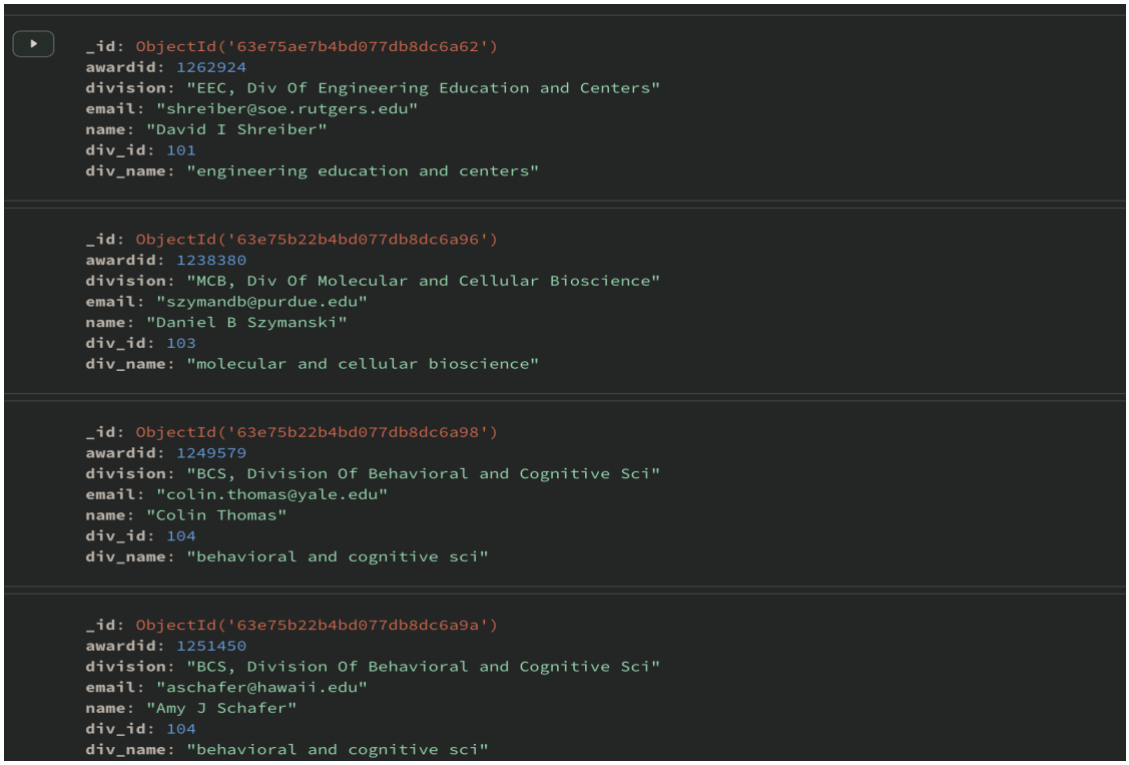


Fig.9 - NSF Data Columns stored into database

APS Data :

The APS (American Physical Society) is a non-profit academic organization that promotes the development of research in physics through academic journals, scientific conferences, and exhibitions. The experiment collected publication information from 18 core physics journals and stored the data in JSON format, making it easy for researchers to extract and analyze the data.
Example data shown in Fig.10.

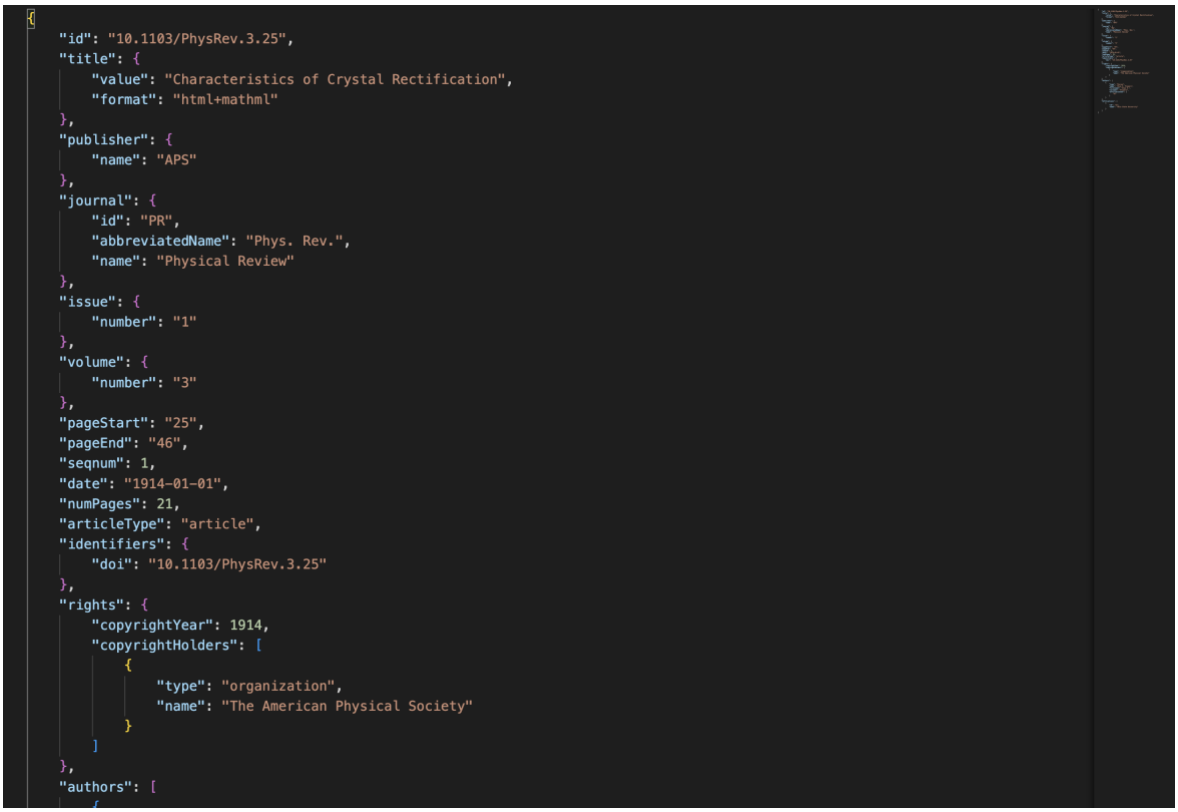


Fig.10 - APS Raw Data Columns in XML Files

Key Columns Name from APS Data –

'id', 'title', 'publisher', 'journal', 'issue', 'volume', 'pageStart', 'pageEnd', 'seqnum', 'date', 'numPages', 'articleType', 'identifiers', 'rights', 'authors', 'affiliations'

APS Data Link - Data Link

Steps Involved in APS Data Extraction –

1. Imported the required libraries, such as json, pymongo, and os.
2. Set up a connection to the MongoDB server using the MongoClient class.
3. Created a new collection or select an existing one within the database.
4. Used the os library to loop through all the JSON files in the specified directory.
5. Load the contents of each JSON file into a Python object using the json.load() method.
6. Iterate over the data and insert each record into the MongoDB collection using the insert_one() method.

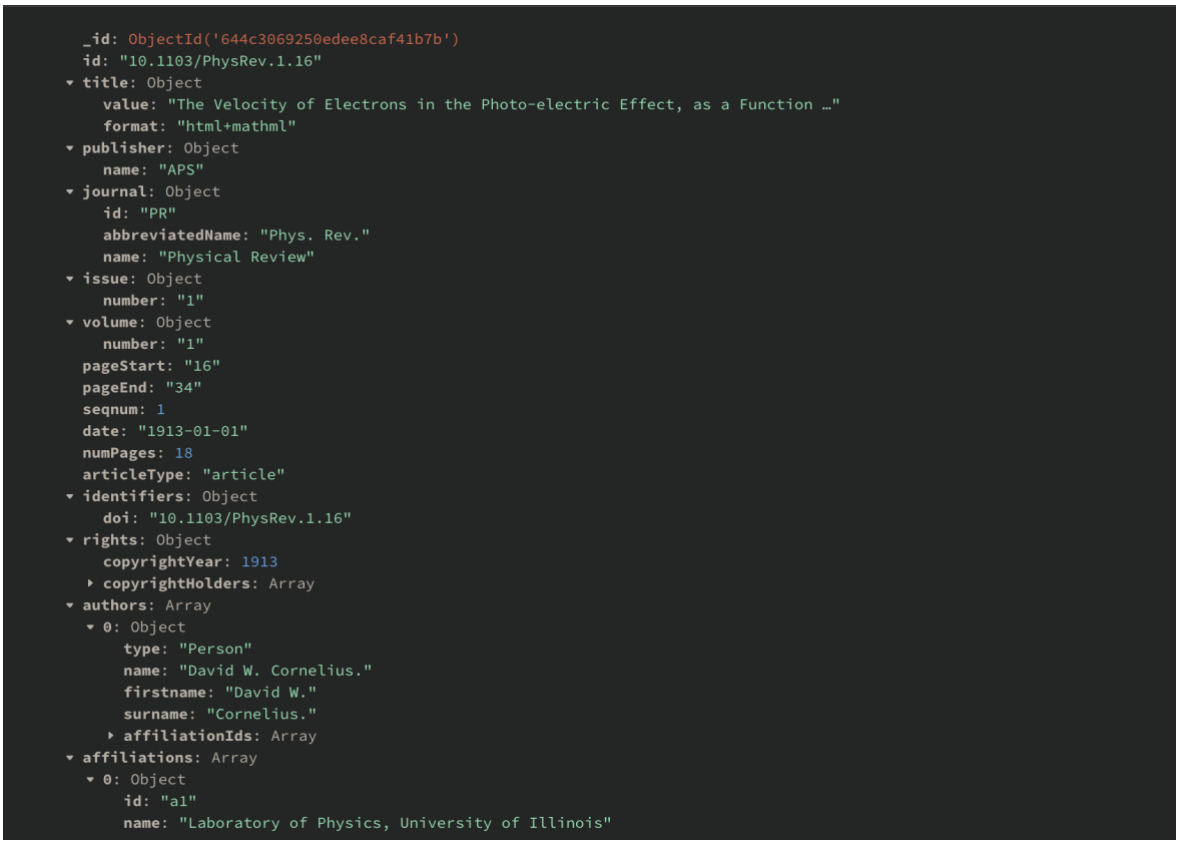


Fig.11 - APS Data Columns stored into database

Scholat Data :

SCHOLAT is an emerging vertical ASN (Academic Social Network) system that is designed and built specifically for researchers in China. The primary objective of SCHOLAT is to enhance collaboration and social interactions focused on scholarly and learning discourses among the community of scholars. Besides social networking capabilities, SCHOLAT

incorporates various modules to encourage collaborative and interactive discussions, such as chat, email, events, and news posts. The experimenters collected data from SCHOLAT, including 10,755 scholars and 202,249 collaboration relations between them.
In this I have used a chain-rule to expand the equation used to identify the ideal candidates, considering the first degree of connection for performance purposes. They used the FAISS library for efficient similarity search and clustering of dense vectors to retrieve the most similar scholars efficiently. They used a beam-search algorithm, a greedy algorithm commonly used in natural language processing or machine translation, to find a sub-optimal solution for the output sequence.

Key Columns Name from Scholat Data –

'scholarid', 'co-workerid'

Scholat Data Link – **Data Link**

	A	B	C
1	scholar_id	coworker_id	source
2	0	68	scholat
3	0	79	scholat
4	1	70	scholat
5	1	71	scholat
6	1	62	scholat
7	1	0	scholat
8	1	65	scholat
9	1	73	scholat
10	1	79	scholat
11	1	76	scholat
12	1	61	scholat
13	1	68	scholat
14	1	60	scholat
15	1	69	scholat
16	2	3	scholat

Fig.12 - Raw Scholat Data Columns in CSV Files

Steps Involved in Scholat Data Extraction –

Scholat dataset was obtained from the respective websites in the form of CSV files. The CSV files were then processed to extract relevant information such as scholar names, emails, and IDs using Python programming language. The extracted data was stored in a MongoDB database for further analysis and processing. The APS dataset contained information about scholars in the field of physics and related disciplines, while the Scholat dataset contained information about scholars from various fields of study. The combined value of the data

from both datasets was around 100,000 entries, making it a valuable resource for identifying potential collaborators in academic research.

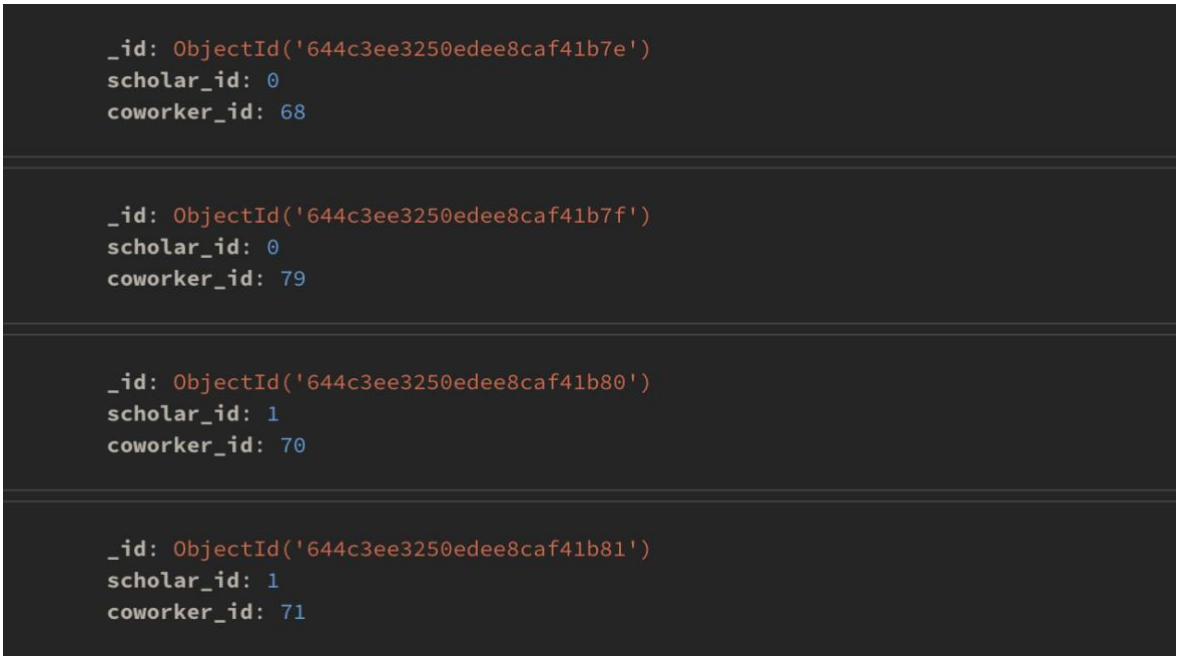


Fig.13 - Scholat Data Columns stored into database

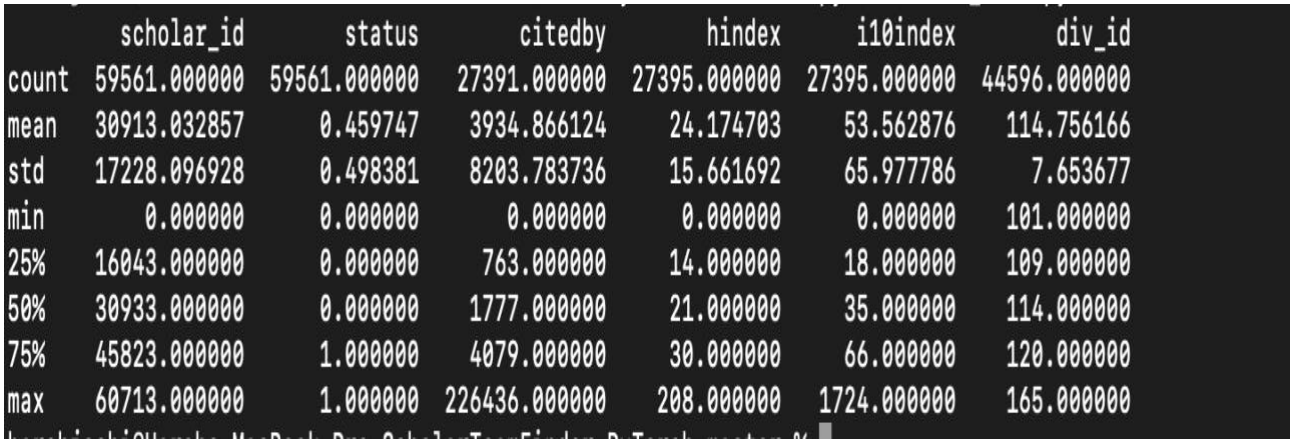
b. Data Carpentry

I scraped the data from various sources including the NSF website, Google Scholar, APS, and Scholat. The data was in different formats such as CSV, XML, and JSON, which I loaded into Python using appropriate libraries. To integrate the data, I consolidated fields and divisions based on their similarity and relevance to our research question. For example, I combined different fields related to research interests and divided them into broader categories such as physics, chemistry, and biology. I matched author names with the email ids which is unique for every scholar and matched the university names using a combination of techniques such as string-matching algorithms, fuzzy matching. I also used domain knowledge and contextual information such as co-authors, affiliations, and publications to ensure accurate matching.

Steps Involved -

1. Load the JSON file into a Python environment using the appropriate library, such as "json".
2. Explore the structure of the JSON file and its keys using the "keys()" method.

3. Flatten nested JSON structures into a tabular format, such as a Pandas Data Frame, using the "json_normalize()" function.
4. Clean the data by removing null values, duplicates, and irrelevant columns.
5. Rename columns and convert data types as necessary using functions like "rename()" and "astype()".
6. Perform exploratory data analysis by summarizing the data using descriptive statistics, visualizations, and grouping operations. This can be done using functions like "describe()", "value_counts()", and plotting libraries such as Matplotlib or Seaborn.
7. Load the CSV file into a Python environment using the appropriate library, such as "pandas".Explore the structure of the CSV file and its columns using the "head()" method.
8. Clean the data by removing null values, duplicates, and irrelevant columns.
9. Rename columns and convert data types as necessary using functions like "rename()" and "astype()".
10. Handle missing values by imputing or removing them using functions like "fillna()" or "dropna()".
11. Store the cleaned and transformed data back into a MongoDB collection.
12. Store any removed data into another collection, as it may be useful for future analysis.
13. Remove data with null or incorrect values of important columns like interest, grant, publications, etc.
14. Perform exploratory data analysis by summarizing the data using descriptive statistics, visualizations, and grouping operations. This can be done using functions like "describe()", "value_counts()", and plotting libraries such as Matplotlib or Seaborn.



	scholar_id	status	citedby	hindex	i10index	div_id
count	59561.000000	59561.000000	27391.000000	27395.000000	27395.000000	44596.000000
mean	30913.032857	0.459747	3934.866124	24.174703	53.562876	114.756166
std	17228.096928	0.498381	8203.783736	15.661692	65.977786	7.653677
min	0.000000	0.000000	0.000000	0.000000	0.000000	101.000000
25%	16043.000000	0.000000	763.000000	14.000000	18.000000	109.000000
50%	30933.000000	0.000000	1777.000000	21.000000	35.000000	114.000000
75%	45823.000000	1.000000	4079.000000	30.000000	66.000000	120.000000
max	60713.000000	1.000000	226436.000000	208.000000	1724.000000	165.000000

Fig.14 - Summary of data while performing actions

Here are some benefits of data carpentry and cleaning in building machine learning models:

Increased accuracy: By cleaning and transforming the data, it is easier to identify and remove outliers, inconsistencies, and missing values, which can lead to more accurate models.

Improved data quality: Data carpentry helps in improving the quality of data by removing unwanted data, handling missing data, and standardizing data formats. This ensures that the data used in building the model is of good quality and free from errors.

Enhanced model performance: By removing irrelevant or noisy data, cleaning data can improve the performance of the machine learning model, as the model can focus on the most important data.

Better understanding of the data: Exploratory data analysis during data carpentry helps in understanding the data better and identifying patterns or trends that can be used to build better models.

Time and cost savings: Cleaning and transforming the data can save time and money in the long run, as it can prevent errors and improve the accuracy of the model, which can reduce the need for rework and iterations.

Fig.15 - Data Cleaning Benefit's

c. Data Visualization

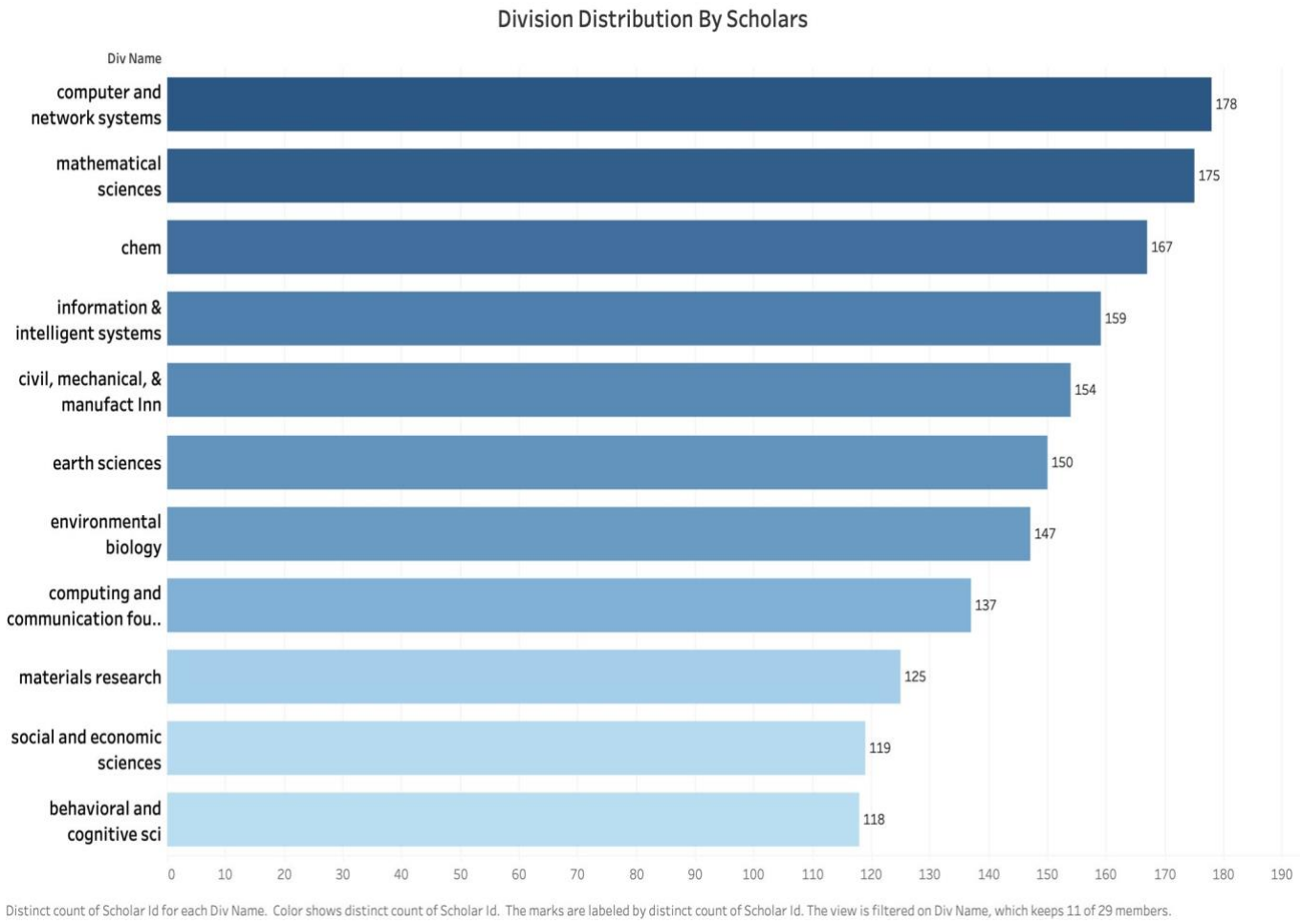


Fig.16 - Division Distribution of Scholars for year 2021

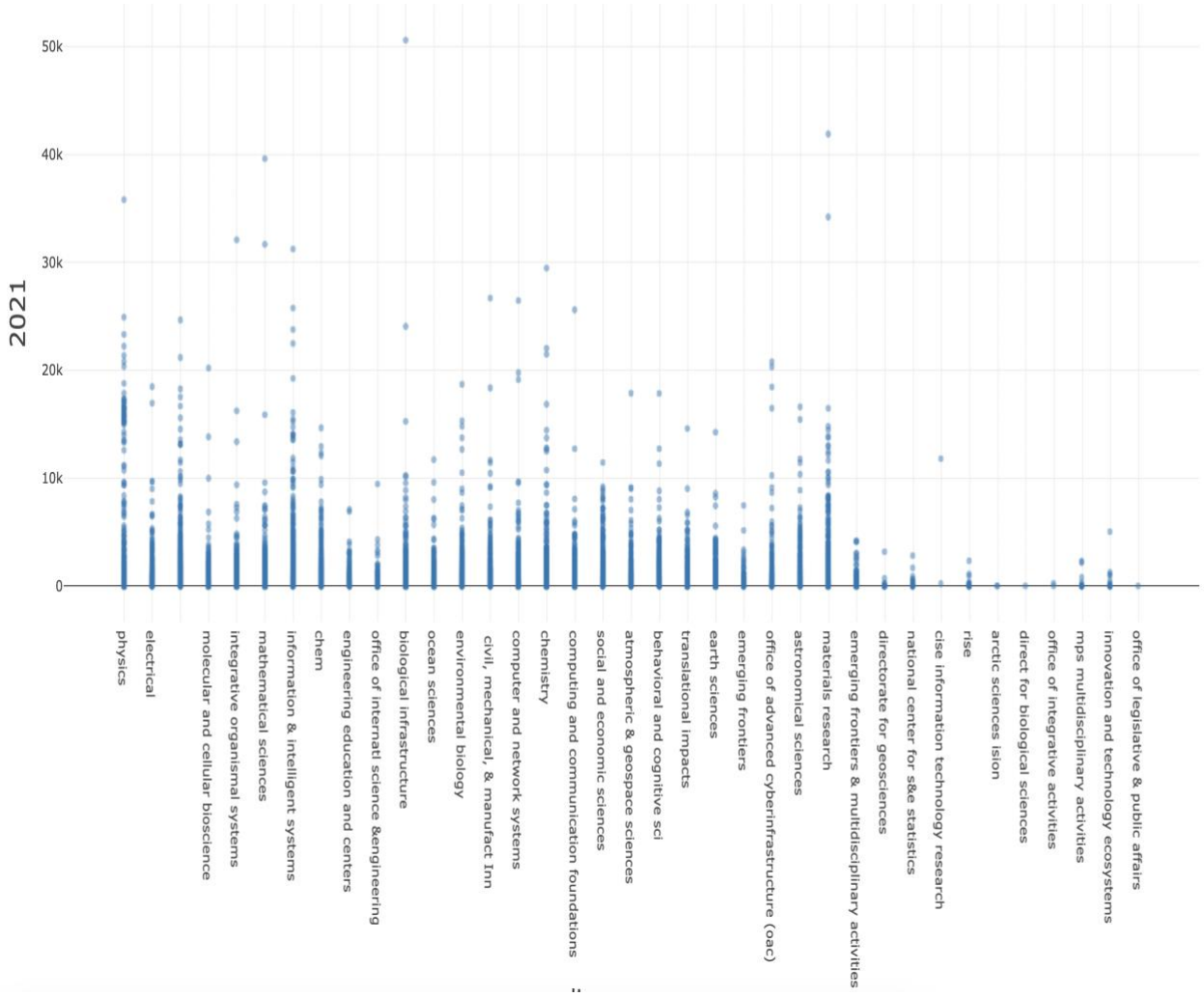


Fig.17 - Division Distribution of Scholars Citation for the year 2021

d. Knowledge Graph Construction

The Knowledge Graph Construction component of the project involves creating a graph-based model that can represent the relationships between scholars, venues, publications, and other related data. This component involves creating a machine learning model that can recommend related scholars to researchers based on their research interests and past collaborations. The Knowledge Graph Construction component involves the use of machine learning algorithms such as deep learning and natural language processing to build a recommendation system. This involves creating a graph-based model that can represent the relationships between scholars, venues, publications, and other related data. This component

will involve creating a machine learning model that can recommend related scholars to researchers based on their research interests and past collaborations.

Phase 1: Extracting facts from Free Text
To begin the process of constructing a knowledge graph, data is extracted from free text, unstructured data sources, and semi-structured data sources. This raw data is then processed to extract entities, relations, and attributes that define them. If data is already structured, it is fused with information from third-party knowledge bases. Various natural language processing techniques are applied to the fused knowledge and processed data, including co-reference resolution, named entity resolution, and entity disambiguation.

Phase 2: Formulating triples from extracted facts
Once pre-processing is complete, an ontology extraction process categorizes the extracted entities and relations under their respective ontologies. Facts are then refined and stored as triples in the knowledge base.

Phase 3: Constructing the knowledge graph with new links and confidences.
To construct the knowledge graph from the knowledge base, statistical relational learning (SRL) is applied to the triples. This process computes a confidence for each fact to identify how far those facts hold true. Missing links are then identified using this confidence, and newly inferred relational links are formed.

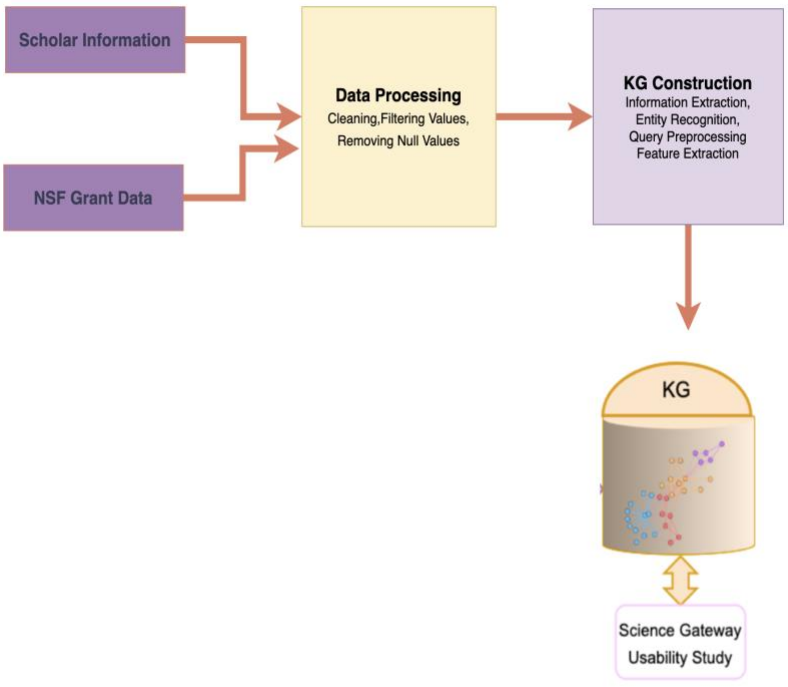


Fig.18 - Knowledge Graph Building Process

7. Scholar Team Finder Methodologies

a. Knowledge Graph Building

To create a knowledge graph, the first step involves collecting the last ten years' worth of NSF award data which includes information such as proposal title, abstract, which organization it belongs to, PI and Co-PI information, among others. From this dataset, information on 60,714 scholars is obtained and used to collect their publications and research interests with a web crawler. With this information, a heterogeneous knowledge graph is built with various types of entity nodes including "Scholar", "Proposal", "Division, Org", "Institution", "Place", "Publication", "Journal" and "Publish Year". There are also various types of relationship edges including "writes", "is written", "is supported by", "includes", "belongs to", "collaborates with", "is published by", and "is published at". The graph edges represent the relationships between the entity nodes. Scholar-related nodes include their affiliated institutions and location, as well as their collaboration relationships. Features for specific nodes such as the text of proposal title and abstract for "Proposal" nodes and research interests for "Scholar" nodes are also added. This knowledge graph enables models to gain insights about potential connections between scholars and can be analyzed for collaboration information on many scholars and their related entity nodes. For example, if two scholars have publications that are published in the same journal and have the same interests, there may be a possible link between them that could lead to a potential collaboration.

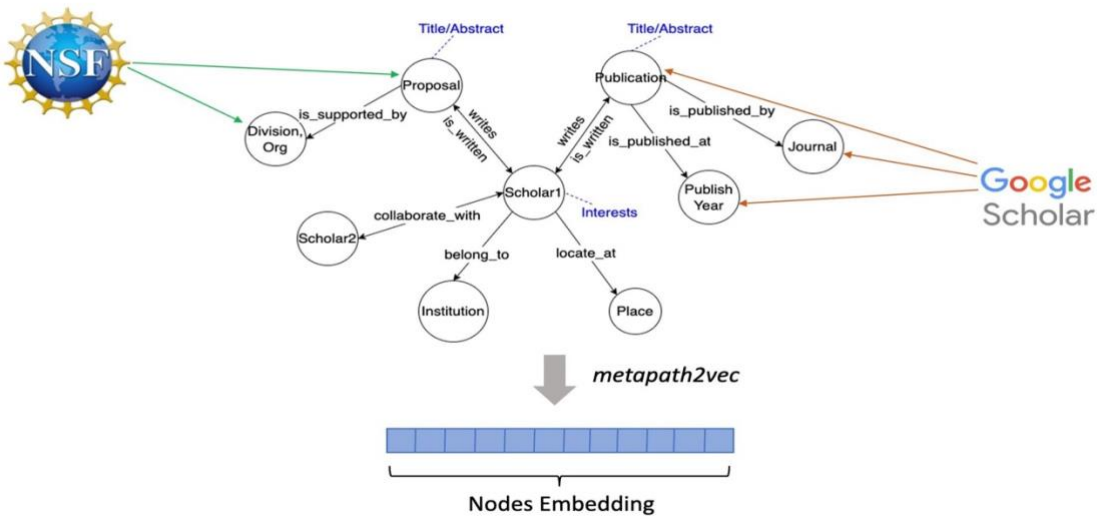


Fig. 19 - The data schema of the proposed academic knowledge graph with connections of properties associated with the entities (Title/Abstract, Interests) of the knowledge graph

b. Scholar Team Types in ASNs

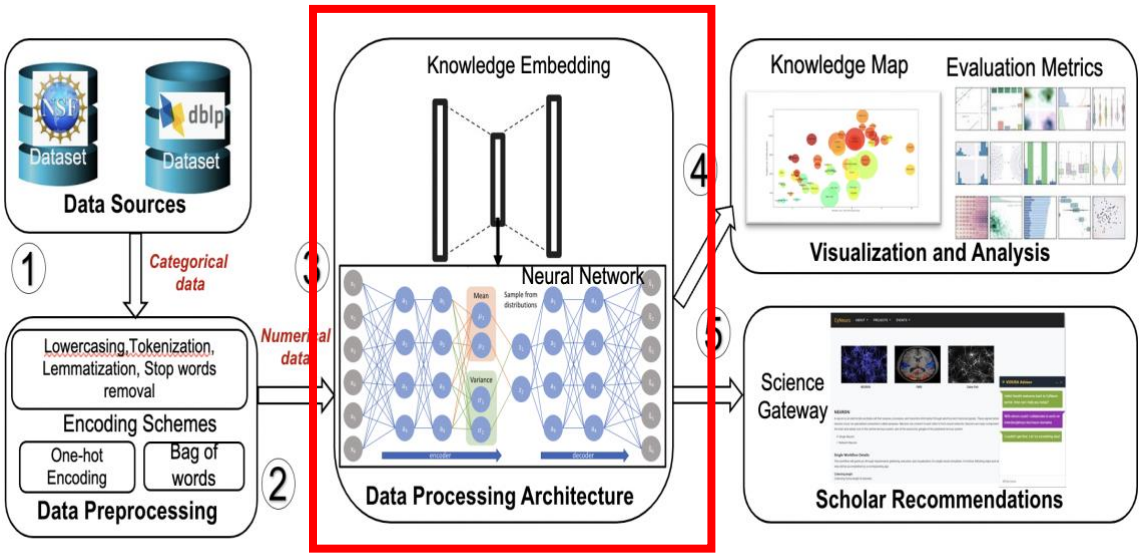


Fig. 20 – Scholar Team Finder Architecture

In this section, we will discuss the concept of scholar teams and how ScholarTeamFinder can assist scholars in forming their own teams. The approach uses link prediction to assess the likelihood of a connection between two scholars, including past collaborations. The application extends link prediction for predicting scholar teams, allowing scholars to locate high-quality teams to collaborate on multi-disciplinary research projects. Two types of scholar teams can be defined and considered:

Finding an existing scholar team:
An increasing number of research problems requires knowledge from multiple fields to identify a solution. As a result, scholars must find collaborators in different fields to form their research teams. In this model, we can use the knowledge graph and deep learning algorithms to locate the scholar who is most related to the target scholar, allowing us to identify the collaborators of that scholar through the "collaborate with" edge in the graph. By using this approach, we can provide recommendations for existing scholar teams.

Generating a new scholar team by identifying related scholars:
Unlike finding an existing scholar team, our model can also help scholars identify several related scholars to create their own team. Our model can predict the links between scholars based on their research interests. Once scholars receive information about related scholars

in their specific fields, they can use this information to form their own scholar team for their research project.

In summary, ScholarTeamFinder can assist scholars in building scholar teams in two ways: by recommending existing teams based on collaborations and by identifying related scholars to form new teams. By using this approach, scholars can collaborate with researchers in other fields and work on interdisciplinary research projects, improving the quality of their research outcomes.

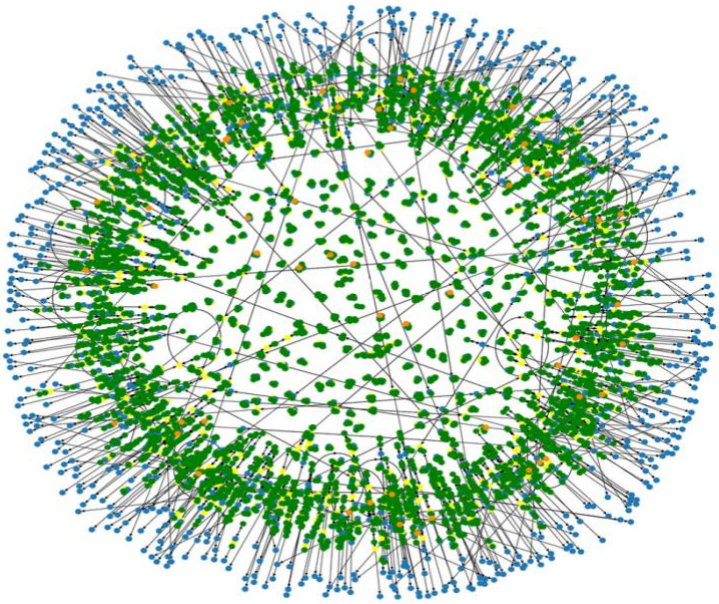


Fig. 21 - An exemplar knowledge graph of 2000 scholar nodes.

c. Feature Encoding in the Knowledge Graph

The original methpath2vec model only used node ID information to create low-dimensional representations of high-dimensional embeddings. In other words, the model initialized embeddings randomly and then used a Skip-gram based method to learn from scratch. In contrast, our work successfully trained the model using semantic node features. Specifically, we assigned semantic meaning to certain entities, such as "Proposal" or "Scholar". To achieve this, we used the Sentence Bert model to map the text of proposals or research interests into vector representation, which we then used as embedding weights to initialize nodes. We also used Sentence Bert to encode the "Proposal" and "Scholar" nodes. This approach allowed similar proposals or scholars with similar interests to be naturally close to

each other, thereby improving the model's performance. The methpath2vec model was limited in that it only considered node ID information when creating embeddings. This meant that important semantic information was not being considered, and the model could potentially miss out on important connections between nodes. In contrast, our approach leveraged semantic node features to better capture the meaning and relationships between entities in the knowledge graph.

To achieve this, we first identified certain entities that we wanted to assign semantic meaning to, such as "Proposal" or "Scholar". We then used the Sentence Bert model, which is a powerful language representation model, to encode the text of proposals or research interests into vector representation. These vector representations were then used as embedding weights to initialize the corresponding nodes in the knowledge graph.

d. Link Prediction Model

The process of predicting links between scholars in our model involves extracting the collaborative relationships between them as shown in Figure 5. The process of predicting links between scholars in our model involves extracting the collaborative relationships between them as shown in Figure 5. Link prediction is a crucial task in knowledge graph analysis, which involves predicting the likelihood of a link between two entities. In the context of a knowledge graph, this task involves predicting the existence of a relationship between two entities based on the graph's structure. One popular approach for link prediction is Metapath2vec, which involves generating node sequences using a specified metapath and using them to learn node embeddings. Metapaths are sequences of node types that define a particular type of path in the graph. For example, in a co-authorship network, a metapath could be "Author-Paper-Author," which describes a path where two authors are connected by co-authoring a paper.

To generate node sequences, a node sequence generator is trained to produce sequences of nodes that follow the specified metapath. The node sequences are then used as input to a neural network-based model to learn node embeddings. Node embeddings are low-dimensional vector representations of nodes that capture the relationships between nodes in the graph. The node embeddings are then used as features in a machine learning model for link prediction. The machine learning model is trained to predict the existence of a link between two entities based on their node embeddings. The model can be trained using various machine learning algorithms, such as logistic regression, random forests, or neural networks.

Overall, the link prediction process involves defining a metapath, generating node sequences, learning node embeddings, and training a machine learning model using the node embeddings. This approach can be used to make link predictions in various types of knowledge graphs, including social networks, citation networks, and biological networks.

Node sequence generator:

To apply metapath2vec to a knowledge graph, we need to generate sequences of nodes that capture the desired semantic relationships. One common approach is to use a random walk algorithm, which starts at a given node and then traverses the graph by randomly selecting one of its neighbors at each step. By repeating this process multiple times, we can generate a set of node sequences that represent the structural context of each node in the graph.

Node embedding:

After generating the node sequences, we can apply the metapath2vec algorithm to learn an embedding for each node in the graph. The goal of the embedding is to represent each node as a dense vector in a high-dimensional space, such that nodes that occur in similar contexts are closer together in the embedding space. The embedding can then be used as input to downstream machine learning models, such as a link prediction model.

Model training:

The final step in building a link prediction model with metapath2vec is to train a machine learning model on the learned node embeddings. The input to the model is typically a pair of node embeddings, representing the two nodes we want to predict a link between. The output of the model is a probability score, indicating the likelihood that the two nodes are connected in the graph. One common approach is to use logistic regression or a neural network to predict the link probability based on the node embeddings. The model is trained on a set of positive and negative examples of links in the graph, and the goal is to optimize the model parameters to maximize the accuracy of the link predictions.

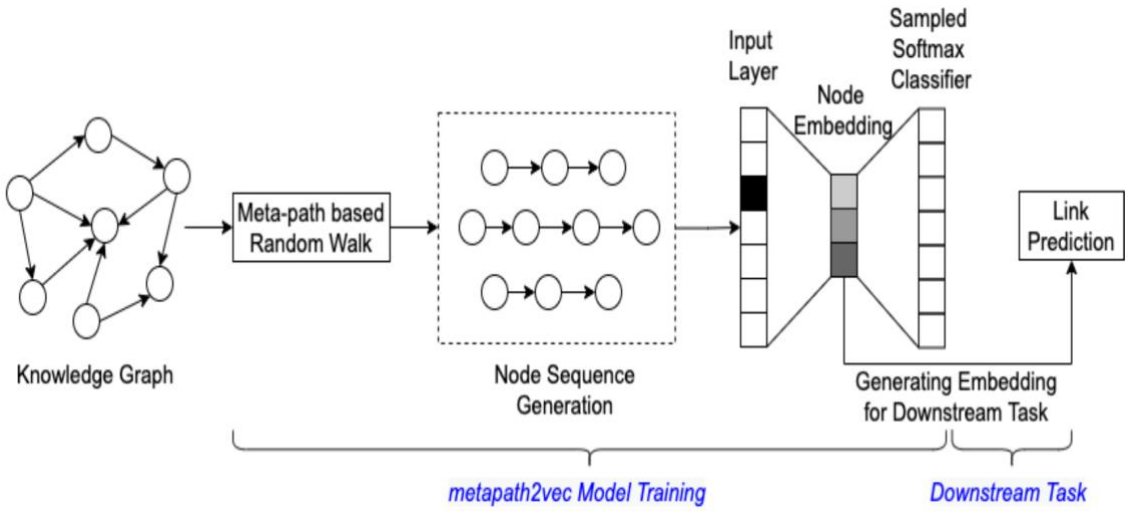


Fig. 22: ScholarTeamFinder model architecture based on knowledge graph.

e. Scholar Team Prediction

A method to predict the probability of whether two scholars have collaborated or could be connected, using link prediction. They extend this approach to predict scholar teams for multi-disciplinary research projects. The model is an improvement on the methpath2vec model and uses Sentence Bert to map semantic node features such as proposals or research interests into vector representations. This helps to improve the model's performance by allowing similar scholars to be closer to each other. To retrieve the most similar scholars efficiently, the authors use the FAISS library for efficient similarity search and clustering of dense vectors. They then use a beam-search algorithm to find a scholar team based on a research problem topic. The algorithm is a greedy algorithm that finds a sub-optimal solution of an output sequence, and it is commonly used in natural language processing or machine translation. The goal of the algorithm is to identify K ideal candidates to work on the research problem, given a target scholar and a research problem. This can be mathematically formulated as --

$$\arg\max_{c} \sum_{i=1}^{K} logp(c_i|s,t)$$

where ci represents the ideal candidate and I range from 1 to K. To expand this equation using a chain-rule, considering only the first degree of connection for the purpose of performance. To extract the collaborative relationships between scholars, the authors use the collaborate relationship between scholars. For example, if one scholar is the PI of a proposal and another scholar is the co-PI of the same proposal or they published one publication together, then they have a collaborative relationship with each other. This helps to get positive samples. For the negative samples, they randomly sample from the negative candidate pools. For every sample, the prediction whether there is a possible connection between scholar S1 and scholar S2 by computing the cosine similarity score with Equation 2. The score helps to find the similarity score between two scholars, and in turn, they can find whether there is a possible link between them. The proposed method can help scholars find an existing scholar team or generate a new scholar team through identification of related scholars, thus aiding multi-disciplinary research projects.

8. Experimental Setup and Evaluation Results

a. Model Performance for Different Datasets

The performance of the ScholarTeamFinder model was evaluated on different types of links, specifically scholar-proposal links, and scholar-publication links, using data extracted from the NSF award dataset. The data consisted of information about 17,952 scholars and their related publications, and the links between the publications and the scholars. The experiment results, shown in Table II, revealed that the ScholarTeamFinder model performed better on scholar-proposal links compared to scholar-publication links. However, when considering the HR@K score, the model on scholar-publication links showed better performance. The reason for this difference in performance is that the number of scholar-proposal data is larger than the number of scholar-publication data, but the number of scholars who work on one proposal data is less. This means that the embedding vectors used in the scholar-proposal model cannot provide more features about the collaboration among scholars, and the model cannot hit many targets in top K samples. On the other hand, the publication data includes more information about scholars' collaboration, such as scholars working on the same publication, and the model on scholar-publication data can hit more targets. It is important to note that while the scholar-publication model showed better HR@K performance, the scholar-proposal model still had a better overall performance in terms of AUC score. This indicates that the scholar-proposal model is more effective for link prediction between scholars, despite its limitations in capturing collaboration , we can obtain accurate and reliable link prediction models for large knowledge graphs.

The model evaluates the performance of the classifier using 5-fold cross-validation and calculates the area under the receiver operating characteristic curve (AUC) as a metric of performance. First, the necessary libraries are imported, including NumPy, pandas, scikit-learn, and matplotlib. The data is loaded from a file named ‘scholar_emb_64_aps.npy’ which contains the scholarly embeddings of authors. To obtain these values, a 5-fold cross-validation approach was used, which involves dividing the dataset into five subsets, or "folds," and training the model on four of the folds while evaluating its performance on the remaining fold. This process is repeated five times, with each fold serving as the test set once.

The code every time randomly choose ~3000 scholars to do predict. The classifier predicts the probability of a link between authors, and the ROC curve is calculated using the scikit-learn roc_curve function. The area under the curve (AUC) is calculated using the auc function, and the AUC score for each fold is stored in a list.

After all folds have been processed, the mean AUC score and standard deviation are calculated and printed to the console. The mean ROC curve is also calculated and plotted using the mean_fpr and mean_tpr arrays. The classifier curve is also plotted to show the baseline performance.

Model	Dataset	AUC-1	AUC-2	AUC-3	AUC-4	AUC-5	SD
Deepwalk	APS	64.1%	67.1%	68.8%	65.9%	65.3%	± 1.11%
	Scholar	46.6%	47.1%	46.2%	45.8%	46.9%	± 0.59%
Metapath2Vec	APS	71.9%	73.7%	72.6%	72.1%	72.9%	± 0.58%
	Scholar	80.2%	86.2%	83.4%	82.7%	83.3%	± 1.20%
ScholarTeam Finder	APS	65.8%	65.8%	66.1%	65.7%	65.7%	± 0.19%
	Scholar	98.0%	97.5%	97.1%	97.5%	97.6%	± 0.14%

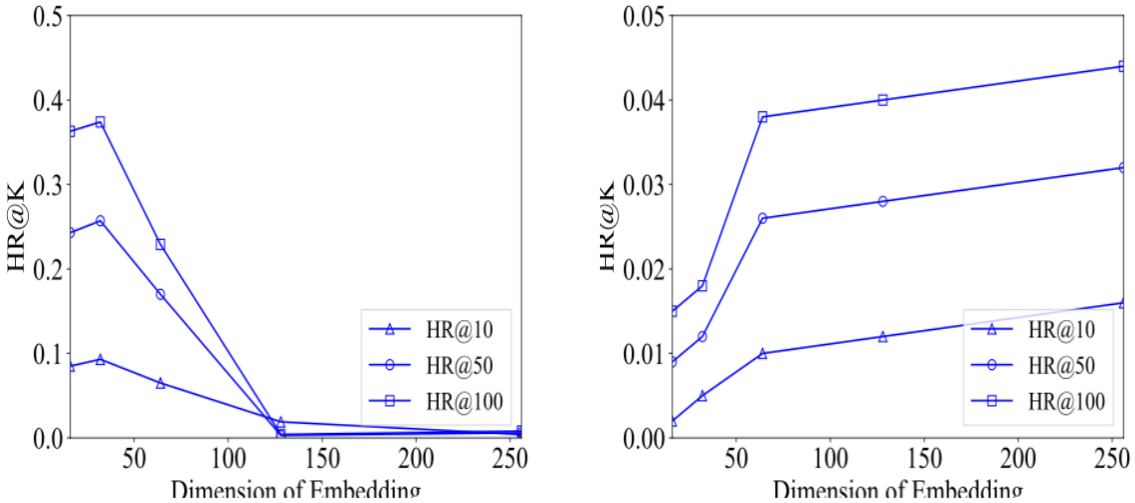
TABLE 1: Results of our proposed ScholarTeamFinder model comparison with 5-fold cross-validation models in terms of AUC Scores

To investigate the influence of embedding dimensions on ScholarTeamFinder's performance, the embedding dimension for both the scholar-proposal and scholar-publication links. The results of this sensitivity analysis are shown in Fig. 22 and Table 1. From Fig. 22, we can observe that the HR@K scores are boosted for the scholar-proposal link with the increase in dimension. On the other hand, for the scholar-publication link, the HR@K scores decrease as the dimension of embedding increases. The reason for this is that embeddings with a larger dimension can encode more useful information, leading to better performance for the scholar-publication link. However, for the scholar-proposal link, when the dimension of embedding is more than 32, the model might overfit, resulting in a decrease in performance. In addition to the sensitivity analysis, we also provided an application case study to demonstrate the efficiency of the ScholarTeamFinder model in real-world scenarios. In this case study, one scholar was selected, and the actual links from the proposal and publication datasets were compared with the prediction results from the model. The results of the comparison are shown in Fig. 12, where green rectangles represent scholars, purple rectangles indicate institutions, and blue and yellow edges between scholars have attributes and probabilities. The attributes of the edges include proposals and publications, indicating that two scholars work on one proposal or publication and have links between them. The probability of the edges represents the cosine similarity score for two or more scholars. The sensitivity analysis and application case study provide evidence of the efficiency and usefulness of ScholarTeamFinder. The model's ability to predict potential collaborators based on scholars' publication and proposal data has the potential to accelerate and improve scientific research by facilitating collaboration and enabling the formation of interdisciplinary research teams. However, it is important to note that the model's

performance is limited by the quality and completeness of the data used to train the model. Therefore, future work should focus on improving the quality and availability of scholarly data to further enhance the model's performance.

Dataset	**Model**	**HR@10**	**HR@50**	**HR@100**
	deepwalk	0.000	0.000	0.000
APS	Metapath2vec	0.003	0.005	0.007
	ScholarTeamFinder	0.171	0.242	.268
	deepwalk	0.000	0.000	0.000
SCHOLAT	Metapath2vec	0.000	0.000	0.000
	ScholarTeamFinder	0.000	0.000	0.000
	deepwalk	0.0001	0.0004	0.0012
NSF Proposal	Metapath2vec	0.004	0.006	0.008
	ScholarTeamFinder	0.093	.257	0.374

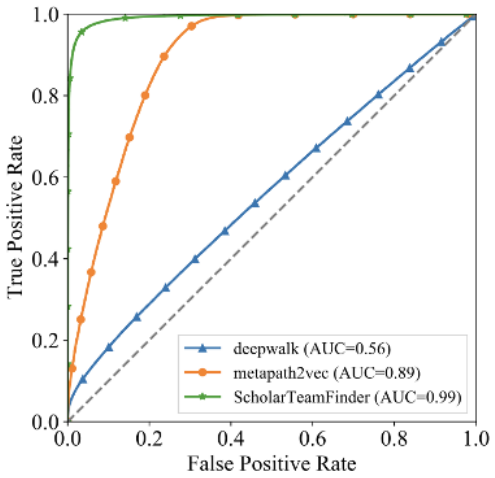
TABLE 2: Results of our proposed ScholarTeamFinder model comparison with state-of-the-art deepwalk and metapath2vec models in terms of HR@K.



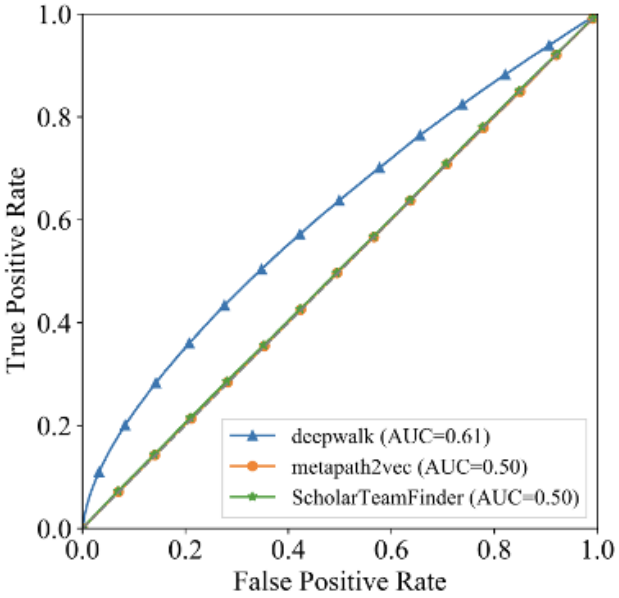
(a) Scholar-Proposal **(b) Scholar-Publication**

Fig. 23: HR@K scores comparison for different dimensions of embedding.

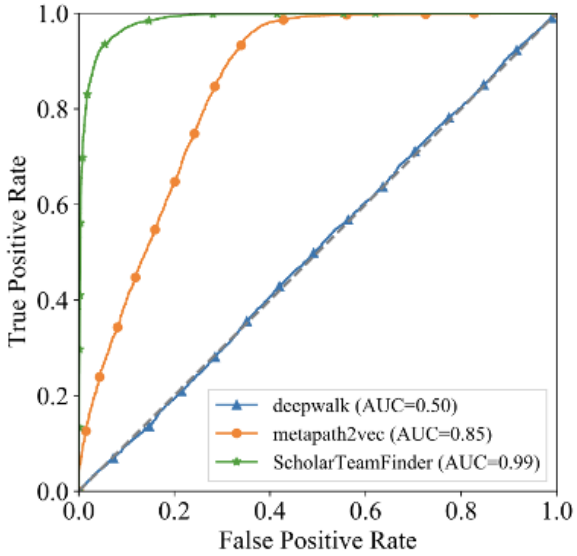
b. Data Visualization of ROC Curves for Different Datasets -



(a) APS dataset.



(b) Scholat dataset.



(c) NSF awards dataset.

Fig. 24 (a,b,c) : Models comparison using ROC curves on different datasets.

c. New Modal Results –

This table provides the evaluation results for three different models, DeepWalk, Metapath2Vec, and ScholarTeamFinder, on two different datasets, APS and Scholar. Each model and dataset combination reports four evaluation metrics: AUC, NDCG@10, NDCG@50, and NDCG@100.
Last week, I focused on getting the NDCG@K results for our model evaluation. This involved implementing the necessary code to calculate the NDCG@K score for our recommendations. NDCG is a measure of ranking quality that considers both relevance and position of the recommended items. It is a popular evaluation metric in recommendation systems. By computing the NDCG@K score, we can measure the effectiveness of our model in recommending relevant scholars to users. I am still working on analyzing and interpreting these new results and comparing them with the previously obtained AUC scores. This will help to gain a better understanding of the strengths and weaknesses of our model and identify areas for further improvement.
Among the models, ScholarTeamFinder has the highest AUC score and NDCG@100 score on both datasets, while Node2vec has the lowest AUC score and NDCG@100 score on both datasets. Overall, the ScholarTeamFinder model seems to perform better on both datasets based on the evaluation metrics used.

Choosing NDCG Over HR@K :

HR@K and NDCG are both evaluation metrics used to measure the performance of recommendation systems. HR@K stands for Hit Rate at K, and it measures the percentage of correct recommendations among the top K items recommended. NDCG stands for Normalized Discounted Cumulative Gain, which considers the position of the recommended item and discounts the score of the recommended items that are farther from the top. The main difference between HR@K and NDCG is that HR@K only considers whether the recommended item is among the top K, while NDCG takes into account the ranking of the recommended items. HR@K is a simpler metric to calculate, but it does not account for the ranking position of the recommended items. In this model NDCG is useful because it provides a more comprehensive evaluation of the performance of our recommendation system, considering not only whether the recommended items are among the top K but also their ranking position. NDCG also discounts the score of items that are farther from the top, reflecting the fact that users are more likely to look at the top-recommended items. Also, NDCG@k can evaluate the quality of the ranking by considering both the relevance and the order of the retrieved items. And HR@K is a binary metric. Our prediction results are based on the similarity scores, so it is a ranking list, we think NDCG@K should be reasonable for this result.

Additionally, these measures besides AUC because AUC only provides a measure of the overall ranking of the recommended items, but it does not provide information about the ranking position of each item. HR@K and NDCG provide a more detailed evaluation of the performance of the recommendation system by looking at the ranking position of the recommended items. This can help us identify potential issues with the recommendation system, such as items that are consistently ranked low.

Modal	**Dataset**	**AUC**	**NDCG@10**	**NDCG@50**	**NDCG@100**
Deepwalk	*APS*	62.96%	0.0605	0.0630	0.0631
	Scholar	51.2%	0.2636	0.2636	0.2636
Metapath2Vec	*APS*	57.5%	0.0964	0.0998	0.0999
	Scholar	50.8%	0.3267	0.3267	0.3267
ScholarTeamFinder	*APS*	70.7%	0.3185	0.3205	0.3206
	Scholar	84.1%	0.3316	0.3316	0.3316

TABLE 3: Results of our proposed ScholarTeamFinder model new evaluation results.

d. Parameter Sensitivity Analysis and Case Study

The ScholarTeamFinder model is a novel approach to predicting potential collaborations between scholars based on existing links in a knowledge graph. In Fig. 24, the center scholar is shown to be associated with other scholars with varying probabilities in terms of working on the same publications and proposals. These existing links in the knowledge graph are used as positive examples to train the model, which can then predict whether there is a possibility of collaboration between two scholars. The blue dotted line paths in Fig. 24 illustrate the links predicted by the ScholarTeamFinder model. Through manual checking of research interests, it is possible to identify potential collaborations from different perspectives. This example demonstrates how the ScholarTeamFinder model works and proves its efficiency. By following the first hop prediction results, it is possible to find scholars who have collaborated with the first scholar predicted, such as scholar 18643 and scholar 27455 in Fig. 24. Through computing the probabilities of the center scholar with the second hop prediction results, it is possible to identify possible links, thus proving the existence of research teams with ScholarTeamFinder.

In addition to identifying potential collaborations between scholars, the ScholarTeamFinder model also considers the institution that scholars come from. Scholars from the same

institution may have a higher potential for collaboration, as seen in the cases of scholar 36896 and scholar 9244. This finding is consistent with real-world observations and suggests that institution information can serve as an important feature for future recommendations. Moreover, the model can identify higher similarity scores with the link of co-authors compared to the link of co-PIs. This result is likely since publications provide more specific information about scholars' research, while proposals only provide a scope. Therefore, the model can capture more useful information from publications data, which in turn can help scholars to expand their existing academic links and find more potential collaborators from other institutions.

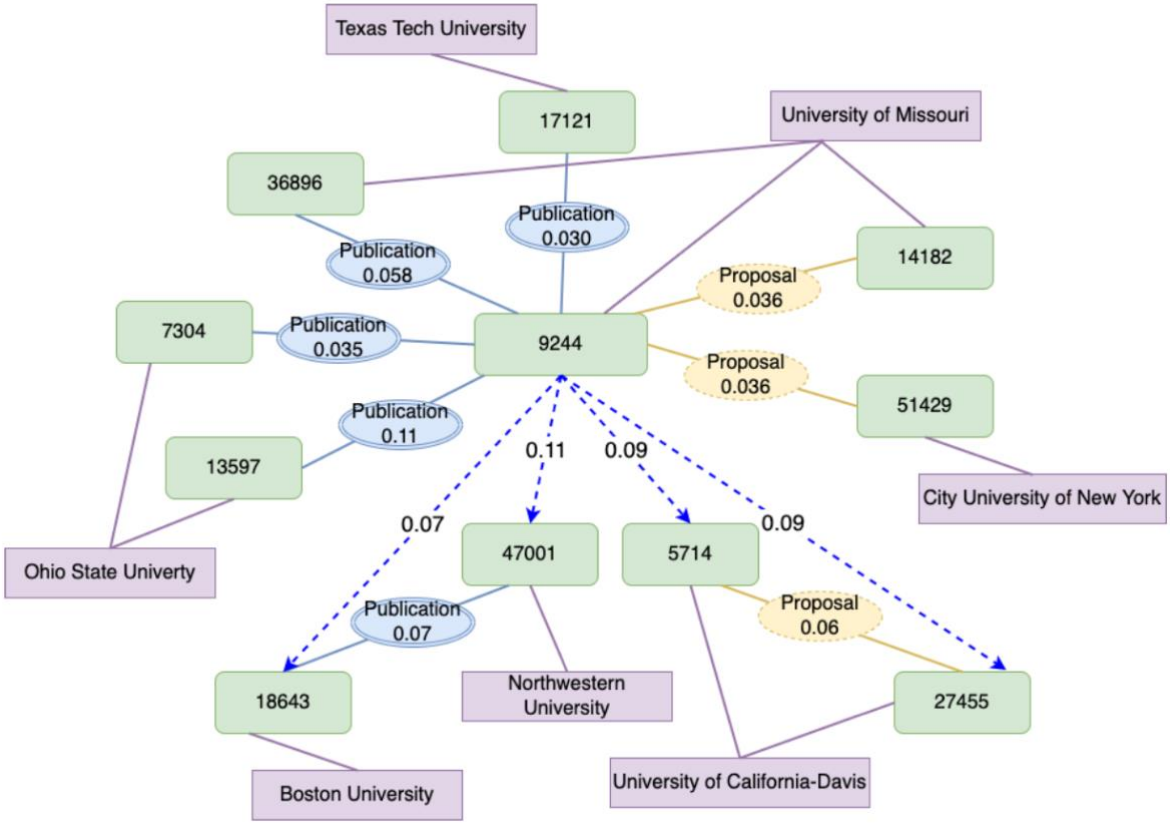


Fig. 25: Exemplar visualization of ScholarTeamFinder model output showing potential collaborators for a given scholar.

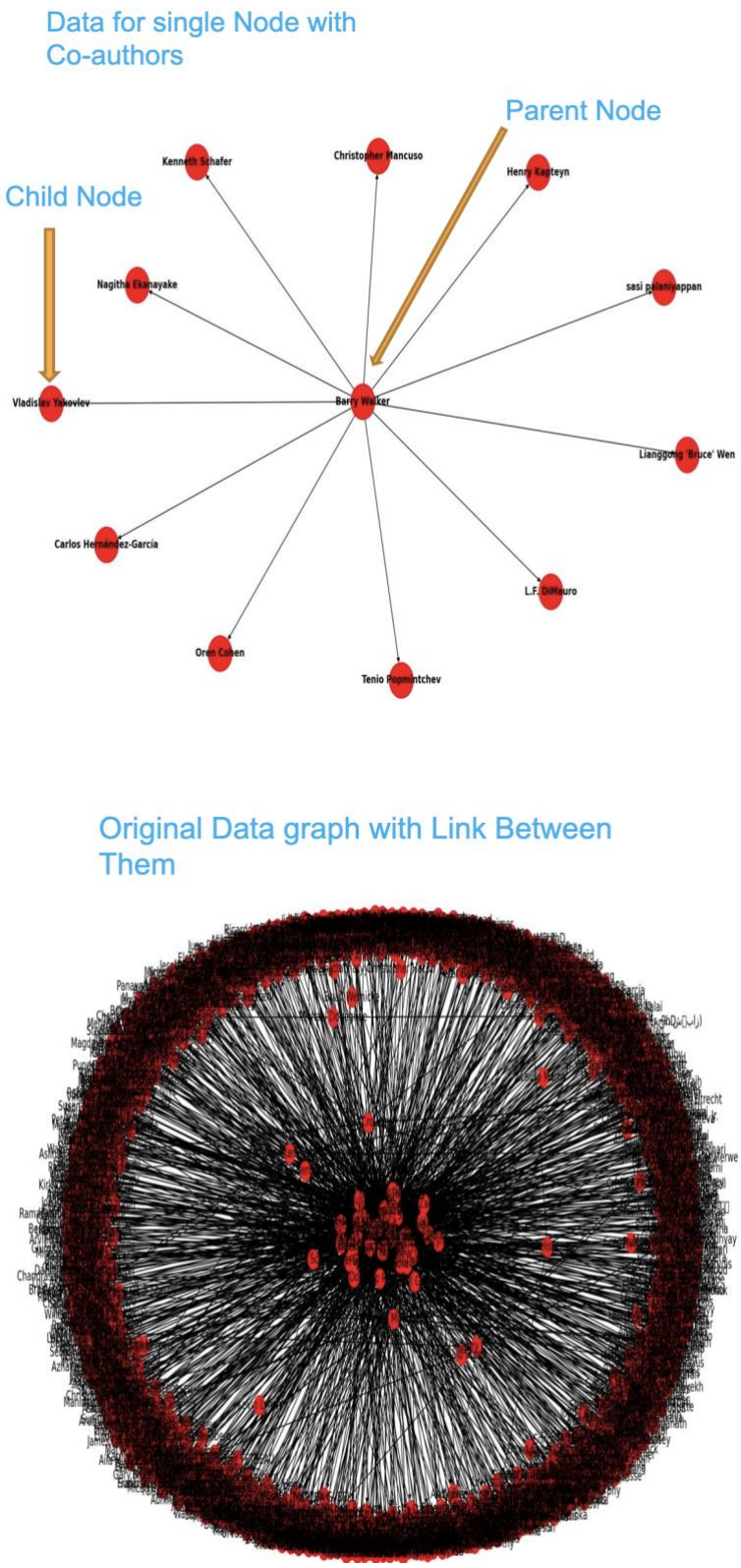


Fig. 26: visualization of ScholarTeamFinder model output of multimode graph and single node graph.

9. Conclusion

In conclusion, the ScholarTeamFinder model is a powerful tool for scholars looking to expand their research network and find potential collaborators. By leveraging existing links in a knowledge graph and analyzing publication and institution data, the model can predict the probability of collaborations between scholars. The embedding method used in the model efficiently extracts information from the knowledge graph and improves the model's performance. The beam search algorithm for finding a scholar team based on research interests of identified scholars further enhances the model's usefulness.

The ScholarTeamFinder model has numerous applications in the real world. It can be used in research organizations to identify potential collaborations within and outside the organization. Universities and other academic institutions can use it to facilitate interdisciplinary research collaborations between faculty members from different departments. Funding agencies can also use it to identify potential grant proposals that involve collaborations between researchers from different institutions.

The positive points of the ScholarTeamFinder model are its ability to identify potential collaborations and expand the research network of scholars, the efficiency of the embedding method used in the model, and the effectiveness of the beam search algorithm in finding a scholar team based on research interests. The model can help scholars save time and effort by providing them with a list of potential collaborators that match their research interests. It also has the potential to increase the impact of research by facilitating collaborations between researchers from different disciplines and institutions. Overall, the ScholarTeamFinder model is a valuable resource for scholars looking to expand their research network and find potential collaborators in their field.

10.References

1. [1] F. Xia, W. Wang, T. M. Bekele, and H. Liu, “Big scholarly data: A survey,” *IEEE Transactions on Big Data*, vol. 3, no. 1, pp. 18–35, 2017.
2. [2] Chong Chen et al. “Graph Heterogeneous Multi-Relational Recommendation”. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 35. 5. 2021, pp. 3958–3966.
3. [3] Ziyang Chen et al. “Temporal knowledge graph question answering via subgraph reasoning”. In: Knowledge-Based Systems (2022), p. 109134.
4. [4] Cheng Deng et al. “GAKG: A Multimodal Geoscience Academic Knowledge Graph”. In: Pro- ceedings of the 30th ACM International Conference on Information & Knowledge Management. 2021, pp. 4445–4454.
5. [5] “Nsf awards info,” https://www.nsf.gov/awardsearch, accessed: 2019-10-
6. [6] “Aps dataset,” https://www.aps.org/index.cfm, accessed: 2022-09-26.
7. [7] “Scholat dataset,” https://www.scholat.com/research/opendata/,

   accessed: 2022-09-26.

8. [8] Y. Xu, D. Zhou, and J. Ma, “Scholar-friend recommendation in online academic communities: an approach based on heterogeneous network,”

   Decision Support Systems, vol. 119, pp. 1–13, 2019.

9. [9] H. Jin, P. Zhang, H. Dong, M. Shao, and Y. Zhu, “Personalized scholar recommendation based on multi-dimensional features,” *Applied*

   Sciences, vol. 11, no. 18, p. 8664, 2021.

10. [10] Ruijie Wang et al. “Acekg: A large-scale knowledge graph for academic data mining”. In: Pro- ceedings of the 27th ACM international conference on information and knowledge management. 2018, pp. 1487–1490.
11. [11] Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. “Improving multi-hop question answering over knowledge graphs using knowledge base embeddings”. In: Proceedings of the 58th annual meeting of the association for computational linguistics. 2020, pp. 4498–4507.
12. [12] Xinyao Shen et al. “Diversified query generation guided by knowledge graph”. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. 2022, pp. 897– 907.
13. [13] S. Boussaadi, H. Aliane, O. Abdeldjalil, D. Houari, and M. Djoumagh, “Recommender systems based on detection community in academic so- cial

network," in 2020 International Multi-Conference on:"Organization of Knowledge and Advanced Technologies"(OCTA). IEEE, 2020, pp.

14. [14] S. Zhao, R. Peng, M. Zhang, and L. Tan, "Heterorwr: a novel algorithm for top-k co-author recommendation with fusion of citation networks," IEICE Transactions on Information and Systems, vol. 103, no. 1, pp. 71–84, 2020.
15. [15] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," arXiv preprint arXiv:1709.05584, 2017.
16. [16] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 701–710.
17. [17] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in Proceedings of the 24th international conference on world wide web, 2015, pp. 1067–1077.
18. [18] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855– 864.
19. [19] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, "Heterogeneous network embedding via deep architectures," in Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 119–128.
20. [20] Y.Cen,X.Zou,J.Zhang,H.Yang,J.Zhou,andJ.Tang,"Representation learning for attributed multiplex heterogeneous network," in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1358–1368.
21. [21] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, 2017, pp. 135–144.
22. [22] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," Advances in neural information processing systems, vol. 30, 2017.
23. [23] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," stat, vol. 1050, p. 20, 2017.
24. [24] C.Chen,W.Ma,M.Zhang,Z.Wang,X.He,C.Wang,Y.Liu,andS.Ma, "Graph heterogeneous multi-relational recommendation," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 5, 2021, pp. 3958–3966.
25. [25] S. Fan, J. Zhu, X. Han, C. Shi, L. Hu, B. Ma, and Y. Li, "Metapath- guided heterogeneous graph neural network for intent recommendation," in Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 2478–2486.

26. [26] S. Liu, I. Ounis, C. Macdonald, and Z. Meng, “A heterogeneous graph neural model for cold-start recommendation,” in Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, 2020, pp. 2029–2032.
27. [27] C. LI, Y. TANG, Z. XIAO, and T. LI, “Influential scholar recom- mendation model in academic social network,” Journal of Computer Applications, vol. 40, no. 9, p. 2594, 2020.
28. [28] W. Zhao, Z. Zou, Z. Wei, W. Gong, C. Yan, and A. K. Luhach, “Coauthorship network mining for scholar communication and collab- oration path recommendation,” Security and Communication Networks, vol. 2021, 2021.
29. [29] Y. Zhang, S. S. Sivarathri, and P. Calyam, “Scholarfinder: knowledge embedding based recommendations using a deep generative model,” in 2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService). IEEE, 2020, pp. 88–95.
30. [30] L. Lu¨and T. Zhou, “Link prediction in complex networks: A survey,” Physica A: statistical mechanics and its applications, vol. 390, no. 6, pp. 1150–1170, 2011.
31. [31] Y. Dong, J. Tang, S. Wu, J. Tian, N. V. Chawla, J. Rao, and H. Cao, “Link prediction and recommendation across heterogeneous social net- works,” in 2012 IEEE 12th International conference on data mining. IEEE, 2012, pp. 181–190.
32. [32] D. Davis, R. Lichtenwalter, and N. V. Chawla, “Multi-relational link prediction in heterogeneous information networks,” in 2011 Interna- tional Conference on Advances in Social Networks Analysis and Mining. IEEE, 2011, pp. 281–288.
33. [33] M. Lu, X. Wei, D. Ye, and Y. Dai, “A unified link prediction framework for predicting arbitrary relations in heterogeneous academic networks,” IEEE Access, vol. 7, pp. 124 967–124 987, 2019.
34. [34] N.Kanakaris,N.Giarelis,I.Siachos,andN.Karacapilidis,“Shalliwork with them? a knowledge graph-based approach for predicting future research collaborations,” Entropy, vol. 23, no. 6, p. 664, 2021.
35. [35] N.ReimersandI.Gurevych,“Sentence-bert:Sentenceembeddingsusing siamese bert-networks,” arXiv preprint arXiv:1908.10084, 2019.
36. [36] A. Graves, “Sequence transduction with recurrent neural networks,” arXiv preprint arXiv:1211.3711, 2012.
37. [37] J. Johnson, M. Douze, and H. Je´gou, “Billion-scale similarity search with GPUs,” IEEE Transactions on Big Data, vol. 7, no. 3, pp. 535– 547, 2019.
38. [38] Y.ZhangandX.Cheng,“Scholarteamfinder,”2022.[Online].Available: https://www.kaggle.com/dsv/4154382
39. [39] Perozzi, B., Al-Rfou, R., & Skiena, S. (2014). DeepWalk: Online Learning of Social Representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 701-710). ACM.

40. [40] Dong, Y., Chawla, N. V., & Swami, A. (2017). metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 135–144). https://doi.org/10.1145/3097983.3098036