الجمهورية الجزائرية الديمقراطية الشعبية
People's Democratic Republic of Algeria

وزارة التعليم العالي و البحث العلمي
Ministry of Higher Education and Scientific Research

جامعة غرداية
University of Ghardaia

Registration No
/..../..../..../..../....

كــــليـــة العـــلـــــوم والتـكنولــوجيــــا
Faculty of Science and Technology

قســـــم الرياضيــــات والاعـــلام الآلــــــــي
Department of Mathematics and Computer Science

Presented to obtain the academic Master diploma in Computer Science

Specialty : Intelligent Systems for Knowledge Extraction

THEME

Deep Learning Context-aware Technique for Citation Recommendation

Presented by
Bouchra BOUKANOUN & Khira BEN RAHAL

Jury members

Mr.Slimane OULAD NAOUI	MCB	Univ.Ghardaia	President
Mrs.Nacera BRAHIM	MAA	Univ.Ghardaia	Examiner
Mr.Houssem Eddine DEGHA	MCB	Univ.Ghardaia	Supervisor

University Year: 2022/2023

ملخص

إن زيادة عدد المنشورات العلمية على الإنترنت تمثّل تحديًا كبيرًا أمام الباحثين والأكاديميين لاكتشاف المستندات العلمية ذات الصلة بكفاءة والبقاء على اطِّلاع دائم بآخر الأبحاث في مجالاتهم المختصة. كاستجابة لهذا التحدي، ظهرت أنظمة التوصية كتقنية معاصرة توفر توصيات شخصية مصممة خصيصًا لاهتمامات المستخدمين الخاصة بهم. من بين مختلف تقنيات توصية المستندات العلمية، لقد حظيت توصية الاقتباس المتناسبة بسياق النص بانتباه كبير. يهدف هذه النهج إلى توفير قوائم متناسقة من المستندات المرشحة ذات جودة عالية بناءً على تحليل سياقات الاقتباس. للتعامل مع تعقيدات هذه المهمة، استفدنا من مجموعة بيانات CiteULike-a، وهي مجموعة البيانات معروفة لتوصية المقالات العلمية، لتدريب وتقييم نموذجنا. تمت متابعة تنفيذنا بنهج استرجاع المعلومات، بما يتضمن نموذج استرجاع ونموذج تصنيف ونموذج ما بعد التصنيف. استخدمنا نموذج تمثيلات التشفير ثنائية الاتجاه من المحولات، وهو أحدث هندسة التعلم العميق، لتعلم التمثيلات للمستندات وسياقات الاقتباس على حد سواء، بالاضافة الى الشبكة العصبونية ذات البوابات وهي نوع من الشبكات العصبية المتكررة التي تتميز بقدرتها على نمذجة البيانات المتتالية . من خلال دمج العلاقات الاقتباسية، قمنا بتعزيز التمثيلات المتوزعة للمتجهات لالتقاط المعلومات الدلالية والسياقية الهامة. لتقييم أداء نموذجنا، استخدمنا مقاييس تقييم مختلفة، حيث كانت الدقة هي المقياس الأساسي. قمنا بتقييم درجات الدقة لقيم مختلفة لأفضل عدد مستندات، حيث تمثل هذه القيم عدد المستندات ذات الصلة المعتبرة في عملية الاسترجاع. كشفت تجاربنا عن اتجاهات مثيرة للاهتمام، حيث أشارت إلى أنه مع زيادة عدد أفضل المستندات ذات الصلة، تزداد درجة الدقة تدريجيًا. في الختام، أظهر نموذجنا لتوصية الاقتباس المتناسب مع السياق، الذي تم تنفيذه بنهج استرجاع المعلومات واستخدام نموذج تمثيلات التشفير ثنائية الاتجاه من المحولات، نتائج واعدة. من خلال دمج التقنيات المتقدمة وتحسين موارد الحوسبة، حققنا تحسينًا في الدقة لتوصية المستندات العلمية ذات الصلة حيث بلغت 60% . تساهم أبحاثنا في تقدم المعرفة في مجال استرجاع المعلومات للمستندات الأكاديمية، ويمكن أن تركز الأبحاث المستقبلية على استكشاف تحسينات إضافية ودمج ملاحظات المستخدم لتعزيز أداء النظام بشكل أكبر.

الكلمات الرئيسية : توصية الاقتباس ،الوعي السياقي، التعلم العميق

ABSTRACT

The ever-growing number of scientific publications on the Internet presents a significant challenge for researchers and academia to efficiently discover relevant scientific papers and stay up-to-date with the latest research in their respective fields.In response to this challenge, recommender systems have emerged as a contemporary technology that offers personalized recommendations tailored to users' specific interests.

Among the various techniques for scientific paper recommendation, context-aware citation recommendation has garnered considerable attention. This approach aims to provide users with curated lists of high-quality candidate papers based on the analysis of citation contexts. To address the complexities of this task, we leverage the CiteULike-a dataset,a well-known scholarly article recommendation system, to train and evaluate our model. Our implementation followed an information retrieval approach, comprising a retrieval model,ranking model,and post-ranking model. We utilize the BERT model based architecture, to learn representations of both papers and citation contexts.In addition to GRU ,to model sequential data. By incorporating citation relationships,we enhance the distributed vector representations to capture important semantic and contextual information. To evaluate the performance of our model,we employ various evaluation metrics, with accuracy being the primary measure.We assess the accuracy scores for different values of top-k papers,representing the number of relevant papers considered in the retrieval process. Our experiments revealed interesting trends,indicating that as the number of top relevant papers increased, the accuracy score exhibited a gradual increase. Our context-aware citation recommendation model, demonstrated promising results.By incorporating advanced techniques and optimizing computational resources,we achieve improved accuracy in recommending relevant scientific papers which estimated by 60 %. Our findings contribute to the advancement of knowledge in the field of information retrieval for scholarly papers, and future work could focus on exploring additional enhancements and incorporating user feedback to further enhance the system's performance.

***Keywords* :** Citation Recommendation, Context-Aware, Deep Learning.

RÉSUMÉ

Le nombre croissant de publications scientifiques sur Internet représente un défi important pour les chercheurs et le milieu universitaire afin de découvrir efficacement les articles scientifiques pertinents et de se maintenir à jour avec les dernières recherches dans leurs domaines respectifs. En réponse à ce défi, les systèmes de recommandation ont émergé en tant que technologie contemporaine offrant des recommandations personnalisées adaptées aux intérêts spécifiques des utilisateurs.

Parmi les différentes techniques de recommandation d'articles scientifiques, la recommandation de citations contextuelles a suscité une attention considérable. Cette approche vise à fournir aux utilisateurs des listes sélectionnées de candidats d'articles de haute qualité basées sur l'analyse des contextes de citation. Pour relever les défis de cette tâche, nous exploitons le jeu de données CiteULike-a, un système de recommandation d'articles scientifiques bien connu, pour entraîner et évaluer notre modèle. Notre mise en oeuvre suive une approche de recherche d'informations comprenant un modèle de recherche, un modèle de classement et un modèle de post-classement. Nous utilisons le modèle BERT pour apprendre les représentations des articles et des contextes de citation. En plus , GRU pour modéliser des données séquentielles. En incorporant les relations de citation, nous avons amélioré les représentations vectorielles distribuées pour capturer des informations sémantiques et contextuelles importantes. Pour évaluer les performances de notre modèle, nous utilisons différentes mesures d'évaluation, l'exactitude étant la mesure principale. Nous évaluons les scores d'exactitude pour différentes valeurs de top-k articles, représentant le nombre d'articles pertinents considérés dans le processus de recherche. Nos expériences ont révélé des tendances intéressantes, indiquant qu'à mesure que le nombre d'articles pertinents augmente, le score d'exactitude augmente de manière graduelle. Notre modèle de recommandation de citations contextuelles a donné des résultats prometteurs. En intégrant des techniques avancées et en optimisant les ressources informatiques, nous avons amélioré l'exactitude des recommandations d'articles scientifiques pertinents Estimé à 60%. Nos résultats contribuent à l'avancement des connaissances dans le domaine de la recherche d'informations pour les articles scientifiques, et des travaux futurs pourraient se concentrer sur l'exploration d'améliorations supplémentaires et l'intégration les commentaires d'utilisateurs pour améliorer davantage les performances du système.

Mots clés : Recommandation de citation, connaissance du contexte,apprentissage profond.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Acknowledgment

Praise be to Allah SWT, we express gratitude and appreciation for the strength and bravery bestowed upon us by Allah while pursuing our studies and completing this thesis.

We would like to begin by expressing our deep gratitude to our supervisor, Mr. DEGHA Houssem Eddine, for his unwavering support and guidance throughout this endeavor.

We are truly thankful for his patience and understanding, and it has been an absolute honor to collaborate with him on this thesis. Furthermore, we would like to extend our heartfelt appreciation to the members of the jury for their interest in our modest research.

We would also like to convey our sincere thanks to all the teachers who have imparted their knowledge and wisdom to us during our college years. Your dedication and hard work have been invaluable, and we truly appreciate the time and effort you have invested in our education.

Dedication

I dedicate this thesis to my parents, who have always been my greatest supporters, I am forever grateful for your sacrifice, and encouragement and I would not be where I am today without your guidance.

To my brother, to my sisters, to my family BEN RAHAL, GUELLIL and RAZZOUG, who have shared with me the joys and challenges of growing up, I thank you for your love and inspiration.

To my supervisor Dr DEGHA Houssem Eddine, who has challenged me to think and write correctly, I thank you for your guidance and patience.

To my teachers, who have imparted to me the knowledge and wisdom, I thank you for your passion, and expertise. You have kindled in me a love for learning and a desire to make a difference in the world.

To my adored partner Bouchra, who made the way so special. This work will prove our enduring love and friendship.

To my colleagues especially Maria, I thank you for your friendship. You have made this journey much more enjoyable and rewarding.

To everyone who wishes me the best and has helped me along the way.

This thesis is dedicated to all of you.

BENRAHAL Khira

Dedication

First and foremost, I thank Allah who gave me the stamina and courage so that I can realize my dreams with this diploma.

To my beloved parents, whose unwavering love, unwavering support, and endless sacrifices have been the cornerstone of my journey. Your belief in me, your encouragement, and your constant presence have been the driving force behind my thesis achievement, you were the sun that lights my way. Thank you for always being my pillars of strength, I am forever indebted to you for the immeasurable love and guidance you have bestowed upon me. Ask Allah to prolong your ages and keeps you crown upon my head.

To my dear brother and sister, Souad and Farouk and to all my maternal uncles,

BEN KOUMAR Kadour, Boualem, Nour Eddine e and Lamine rahimaho Allah for their precious moral support throughout my studies.

To my mother's parents my grandmother and grandfather Rahimaho Allah.

To Dr. Houssem Eddine DEGHA, for his immense assistance and motivation, which has given me the strength to present this work. I am grateful for the trust you placed in me by agreeing to supervise and guide me throughout this journey.

To my dear friends Khira and Chuireb Rym, BOUCHENAK Maria Louiza

To all my esteemed teachers. I can't describe how much impact they have had on my academic life.

Through your wise guidance and deep knowledge engagement, you learned a lot and grew as a student. It inspired me to strive for excellence and continuous development, not only to communicate knowledge but also to build self-confidence and foster a spirit of creativity,

Thank you all for being an exceptional educator and for making a lasting difference in my academic life.

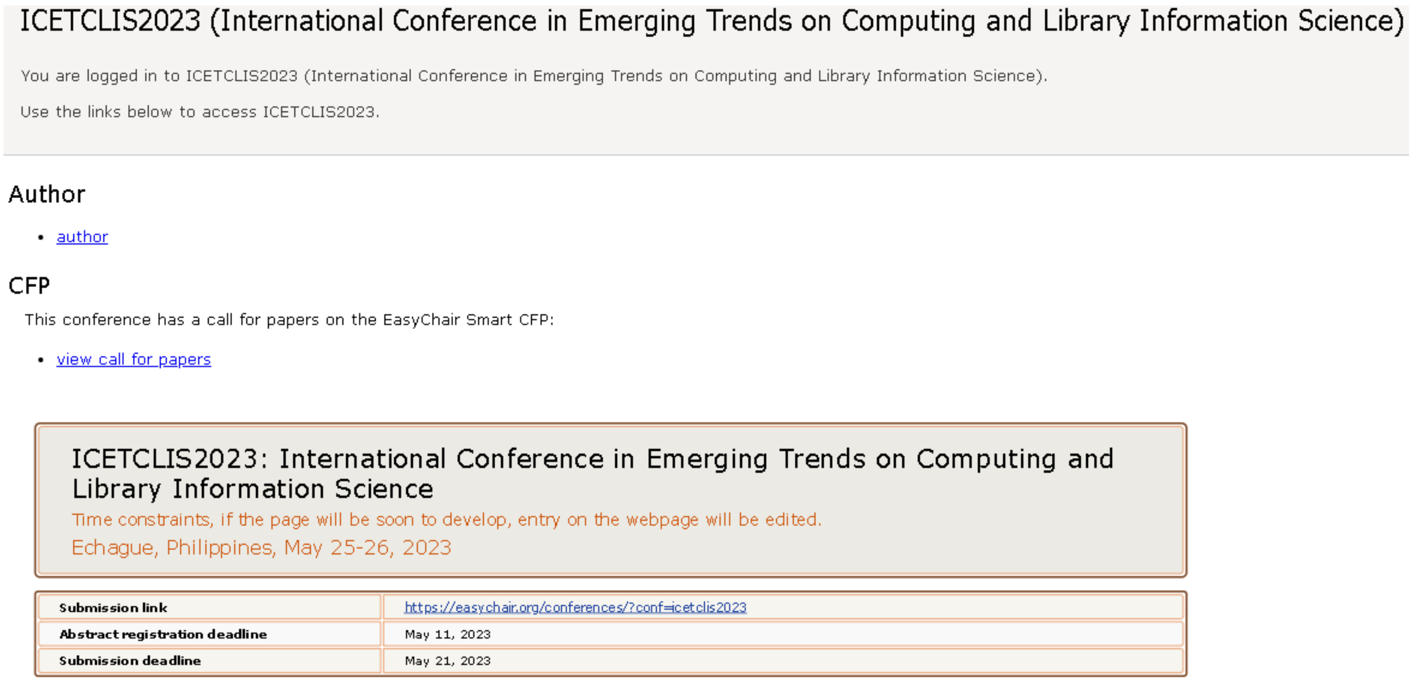
BOUKANOUN Bouchra

Our publications throughout the thesis

1st conference: ICETCLIS2023

ICETCLIS2023, International Conference in Emerging Trends on Computing and Library Information Science, The Hybrid International Conference scheduled for May 25-26, 2023, is hosted by the ISU College of Computing Studies, Information and Communication Technology, in collaboration with several institutions including the University of the East, Chung Yuan Christian University-Taiwan, National University-CCSIT, ISU Office of the Vice President for Research and Development, Extension and Training, and ISU University Professional Development Training Institute. The conference will be held both onsite at Isabela State University's Main Campus in Echague, Isabela, Philippines, as well as on a virtual conference platform for online participation. The conference aims to bring together researchers, academicians, industry professionals, and students from various disciplines to exchange knowledge and ideas related to knowledge technology management. It provides a platform for participants to present their research findings, share innovative solutions, and discuss emerging trends and challenges in the field.

- Organizing Committee
  - Dr. Lloyd Bareng
  - Dr. Christine Charmain F. San Jose
  - Dr. Isagani Angeles
  - Dean Ma. Teresa Boredor
  - Dean Avonn Nova
- Program Committee
  - Dr. Albert A. Vinluan
  - Dr. Jesusimo L. Dioses Jr.
  - Dr. Edward B. Panganiban
  - Dr. Betsie M. Dela Cruz
  - Dr. Sheila M. Geronimo
  - Dr. Rex Bringula
  - Dr. Mideth Abisado



ICETCLIS2023 (International Conference in Emerging Trends on Computing and Library Information Science)

You are logged in to ICETCLIS2023 (International Conference in Emerging Trends on Computing and Library Information Science).

Use the links below to access ICETCLIS2023.

Author

- author

CFP

This conference has a call for papers on the EasyChair Smart CFP:

- view call for papers

ICETCLIS2023: International Conference in Emerging Trends on Computing and Library Information Science

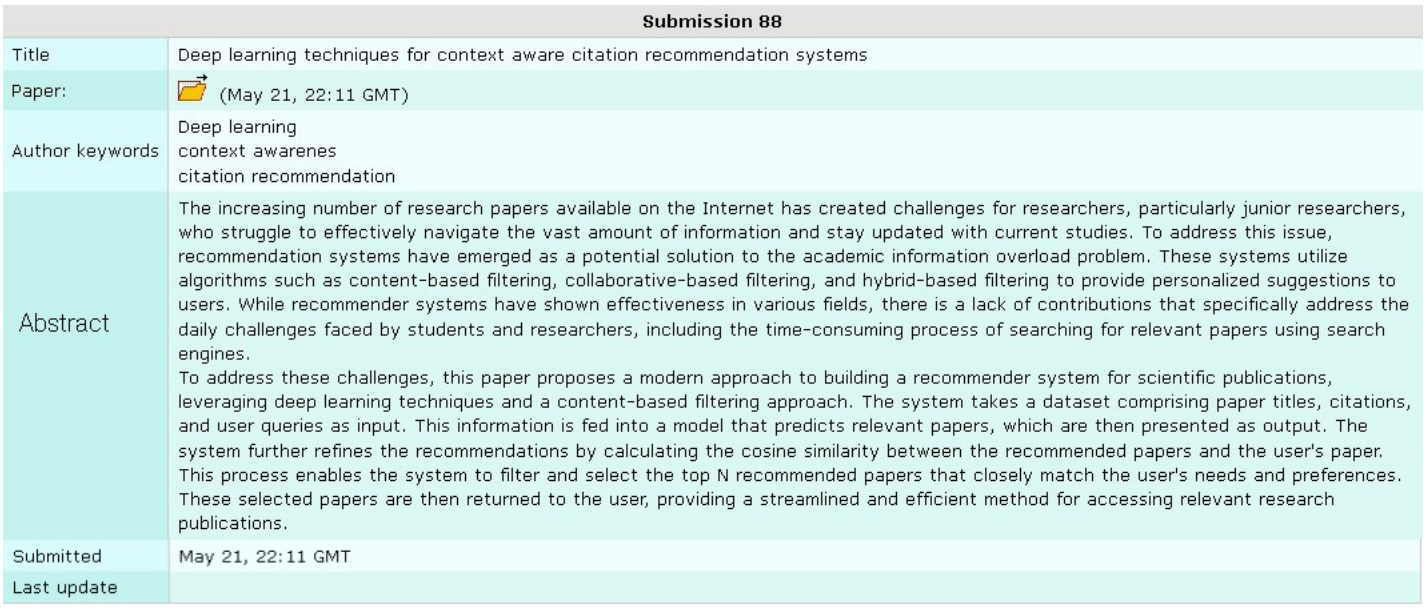
Time constraints, if the page will be soon to develop, entry on the webpage will be edited.

Echague, Philippines, May 25-26, 2023

Submission link	https://easychair.org/conferences/?conf=icetclis2023
Abstract registration deadline	May 11, 2023
Submission deadline	May 21, 2023

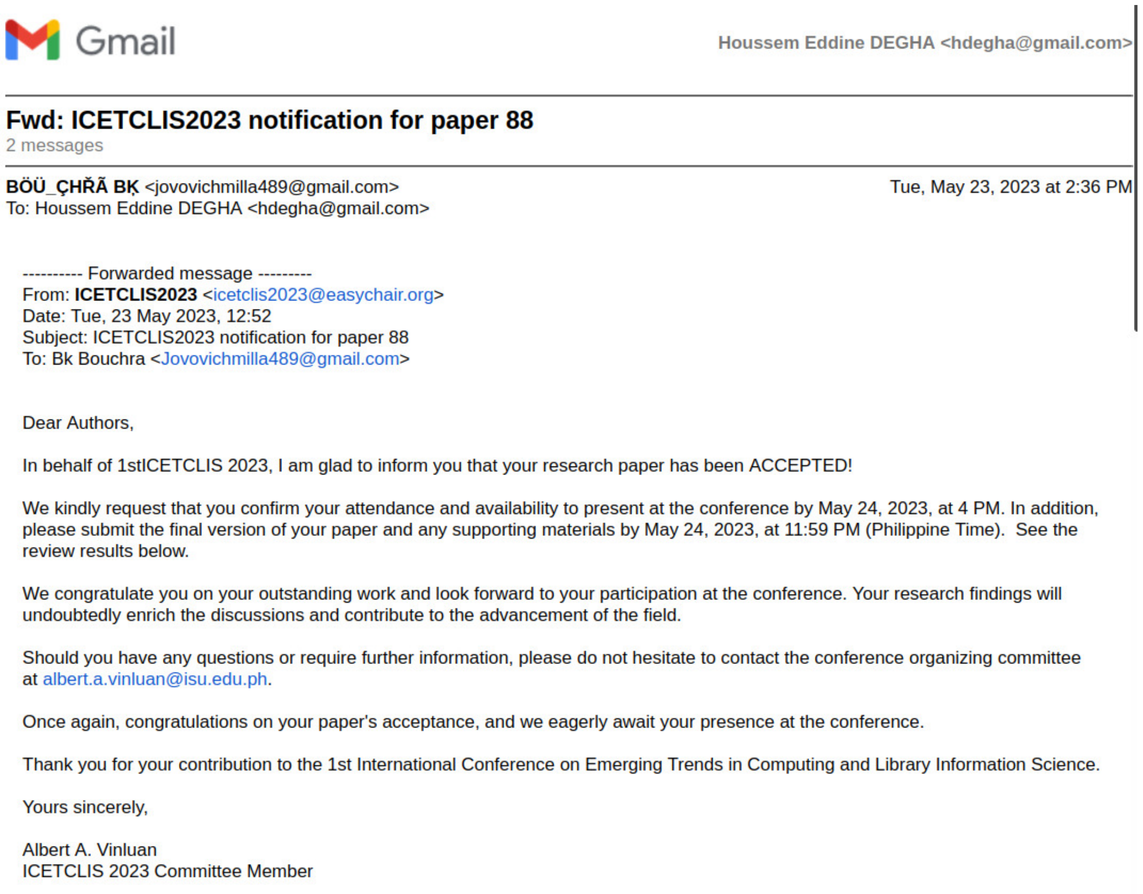
Figure :1 ICETCLIS2023 platform

- Our paper is entitled "Deep learning techniques for context aware citation recommendation systems" has been sent , accepted and presented online on ZOOM platform Figures(2,3,4).



Submission 88	
Title	Deep learning techniques for context aware citation recommendation systems
Paper:	(May 21, 22:11 GMT)
Author keywords	Deep learning context awarenes citation recommendation
Abstract	The increasing number of research papers available on the Internet has created challenges for researchers, particularly junior researchers, who struggle to effectively navigate the vast amount of information and stay updated with current studies. To address this issue, recommendation systems have emerged as a potential solution to the academic information overload problem. These systems utilize algorithms such as content-based filtering, collaborative-based filtering, and hybrid-based filtering to provide personalized suggestions to users. While recommender systems have shown effectiveness in various fields, there is a lack of contributions that specifically address the daily challenges faced by students and researchers, including the time-consuming process of searching for relevant papers using search engines. To address these challenges, this paper proposes a modern approach to building a recommender system for scientific publications, leveraging deep learning techniques and a content-based filtering approach. The system takes a dataset comprising paper titles, citations, and user queries as input. This information is fed into a model that predicts relevant papers, which are then presented as output. The system further refines the recommendations by calculating the cosine similarity between the recommended papers and the user's paper. This process enables the system to filter and select the top N recommended papers that closely match the user's needs and preferences. These selected papers are then returned to the user, providing a streamlined and efficient method for accessing relevant research publications.
Submitted	May 21, 22:11 GMT
Last update	

Figure :2 Information about the sended paper


Gmail

Houssem Eddine DEGHA <hdegha@gmail.com>

Fwd: ICETCLIS2023 notification for paper 88
2 messages

BÖÜ_ÇĦŘÃ BĶ <jovovichmilla489@gmail.com> Tue, May 23, 2023 at 2:36 PM
To: Houssem Eddine DEGHA <hdegha@gmail.com>

---------- Forwarded message ---------
From: **ICETCLIS2023** <icetclis2023@easychair.org>
Date: Tue, 23 May 2023, 12:52
Subject: ICETCLIS2023 notification for paper 88
To: Bk Bouchra <Jovovichmilla489@gmail.com>

Dear Authors,

In behalf of 1stICETCLIS 2023, I am glad to inform you that your research paper has been ACCEPTED!

We kindly request that you confirm your attendance and availability to present at the conference by May 24, 2023, at 4 PM. In addition, please submit the final version of your paper and any supporting materials by May 24, 2023, at 11:59 PM (Philippine Time). See the review results below.

We congratulate you on your outstanding work and look forward to your participation at the conference. Your research findings will undoubtedly enrich the discussions and contribute to the advancement of the field.

Should you have any questions or require further information, please do not hesitate to contact the conference organizing committee at albert.a.vinluan@isu.edu.ph.

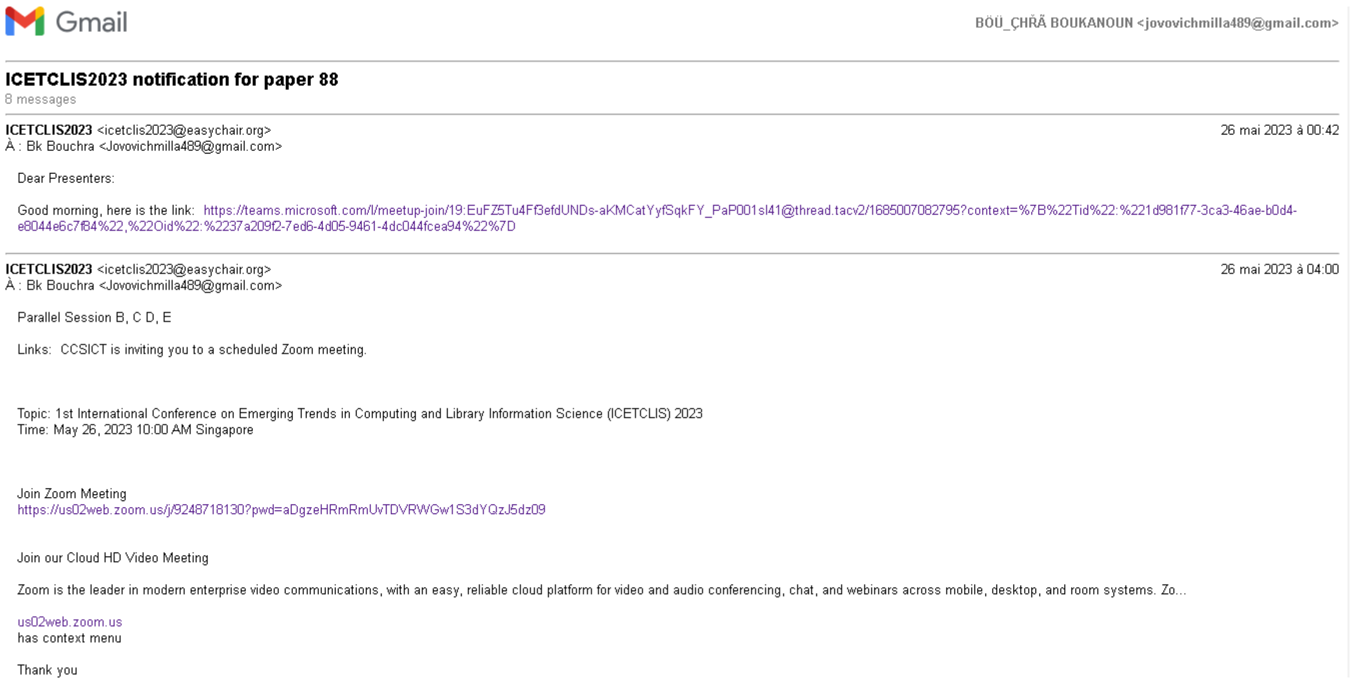
Once again, congratulations on your paper's acceptance, and we eagerly await your presence at the conference.

Thank you for your contribution to the 1st International Conference on Emerging Trends in Computing and Library Information Science.

Yours sincerely,

Albert A. Vinluan
ICETCLIS 2023 Committee Member

Figure :3 Acceptance of paper and invitation to present it


Gmail

BÖÜ_ÇĦŘÃ BOUKANOUN <jovovichmilla489@gmail.com>

ICETCLIS2023 notification for paper 88
8 messages

ICETCLIS2023 <icetclis2023@easychair.org> 26 mai 2023 à 00:42
À : Bk Bouchra <Jovovichmilla489@gmail.com>

Dear Presenters:

Good morning, here is the link: https://teams.microsoft.com/l/meetup-join/19:EuFZ5Tu4Ff3efdUNDs-aKMCatYyfSqkFY_PaP001sl41@thread.tacv2/1685007082795?context=%7B%22Tid%22:%221d981f77-3ca3-46ae-b0d4-e8044e6c7f84%22,%22Oid%22:%2237a209f2-7ed6-4d05-9461-4dc044fcea94%22%7D

ICETCLIS2023 <icetclis2023@easychair.org> 26 mai 2023 à 04:00
À : Bk Bouchra <Jovovichmilla489@gmail.com>

Parallel Session B, C D, E

Links: CCSICT is inviting you to a scheduled Zoom meeting.

Topic: 1st International Conference on Emerging Trends in Computing and Library Information Science (ICETCLIS) 2023
Time: May 26, 2023 10:00 AM Singapore

Join Zoom Meeting
https://us02web.zoom.us/j/9248718130?pwd=aDgzeHRmRmUvTDVRWGw1S3dYQzJ5dz09

Join our Cloud HD Video Meeting

Zoom is the leader in modern enterprise video communications, with an easy, reliable cloud platform for video and audio conferencing, chat, and webinars across mobile, desktop, and room systems. Zo...

us02web.zoom.us
has context menu

Thank you

Figure :4 Link for online paper presentation

INTRODUCTION

In recent years, the rapid increase in online scientific publications has presented a challenge for researchers and the academic community in finding the most relevant papers for their research. The traditional search process using scientific search engines can be time-consuming, as researchers need to read the titles and summaries of articles to determine their suitability and interest. This process often requires further searching for additional relevant articles, adding to the complexity and time investment.

To address this issue, Recommender Systems (RSs) have gained attention as a potential solution. RSs aim to alleviate information overload by suggesting items of potential interest based on user preferences and behavior. They have been successfully applied in various fields[7], including e-commerce, entertainment, and now, scholarly publications. Paper recommender systems specifically focus on recommending relevant scholarly publications[8] to make the process of finding information easier and more efficient. However, many existing approaches to recommender systems primarily focus on recommending the most relevant products to specific consumers, overlooking other important factors. They often fail to consider contextual information such as users' context, document's context, and environmental context[9].

In other words, traditional recommender systems typically deal with applications involving only two entities: people and objects, without placing them in a broader context. In the case of paper recommender systems, there has been a shift towards considering more detailed information beyond just a few keywords. These systems aim to leverage additional contextual information associated with scholarly publications to provide more accurate and personalized recommendations. Such contextual information may include the author's expertise, publication venue, citation network, and even the reader's research interests and background.

By incorporating contextual information, paper recommender systems can go beyond simple keyword matching and provide more tailored recommendations that align with the user's specific research needs and interests. This approach enhances the precision and the relevance of the recommendations, helping researchers navigate the vast land-

scape of scientific publications more effectively. Furthermore, advancements in Natural Language Processing (NLP) and Machine Learning (ML) techniques have facilitated the analysis and extraction of contextual information from scholarly publications. Techniques such as topic modeling, citation analysis, and entity recognition allow for a deeper understanding of the content, relationships, and relevance of papers, enabling more sophisticated recommendation algorithms. The utilization of contextual information in paper recommender systems has the potential to revolutionize the research process by saving researchers' time and effort.
By receiving targeted recommendations based on their research interests, expertise, and the broader scientific landscape, researchers can quickly identify relevant papers, discover new perspectives, and stay up-to-date with the latest developments in their field.
In conclusion, the increased availability of online scientific publications has posed challenges for researchers in finding relevant papers. Recommender Systems, have emerged as a promising solution to address information overload. By incorporating contextual information and leveraging advanced NLP and ML techniques, these systems aim to provide personalized and precise recommendations, easing the burden of searching for scholarly publications and facilitating more efficient research. In this thesis we propose a deep learning approach that uses content based filtering to retrieve the most relevant scientific papers so that why we organize our thesis as follow:
Chapter N#1 "Background," we provide a concise overview of the fields that are relevant to our work. This section aims to provide a contextual understanding of the areas in which our research is situated.
Chapter N#2 "Related works ",We discuss the most well-known relevant works as well as suggestions made by academics and authors in this field.
Chapter N#3 "Conception", we present the overall concept and approach behind the design of our system. This section provides a comprehensive explanation of the underlying principles, methodologies, and ideas that form the foundation of our proposed solution.

Chapter N#4 "Implementation", we delve into the detailed implementation of our proposed system for papers recommendation. Furthermore, we thoroughly analyze and discuss its performance based on existed metrics.Finally, we conclude with "General conclusion" .

CHAPTER 1

BACKGROUND

1.1 Introduction

In order to place the reader in the proper context, this chapter aims to provide some background information about the study area in which this work is situated. Initially, we explore the topic of citation recommendation systems and its various components. We start by defining recommendation systems, which are algorithms designed to suggest relevant items to users. Then, we define the concept of citations, citation recommender systems, which are specialized recommendation systems that provide researchers with suggestions for relevant citations. Next, we delve into the history of citation recommendation systems, tracing their development and evolution over time. We discuss the importance of these systems in research communication, highlighting how they facilitate the dissemination of knowledge and foster collaboration among researchers. Additionally, we outline the benefits of effective citation recommendation, including improved research quality, time-saving, and enhanced scholarly impact.

Moving on, we explore traditional approaches for citation recommendation. This include content-based filtering, collaborative filtering, hybrid approaches, and rule-based techniques. We discuss the strengths and limitations of these traditional methods in generating citation recommendations.

Following that, we delve into deep learning techniques and their application in recommendation systems. We introduce the concept of deep learning and explore how neural networks, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models, can be used for citation recommendation.

Furthermore, we discuss the role of context-awareness in recommendation systems. We explain the importance of understanding context in recommendations and explore different types of contextual information. We also delve into incorporating context in collaborative filtering and context-aware techniques in content-based recommendations. Additionally, we explore the use of contextual embeddings to enhance the performance of recommendation systems. Another important aspect we cover is the evaluation metrics for citation recommendation systems. We discuss accuracy metrics such as precision, recall, and F1-score, as well as ranking metrics like Mean Average Precision (MAP). Additionally, we explore novelty and diversity metrics that assess the novelty and diversity of recommended citations. We also discuss the challenges and considerations involved in evaluating citation recommendation systems.

1.2 Citation Recommendation systems

1.2.1 Definition of Recommender system

A recommendation system(RS), is a type of software application that delivers or proposes items to users based on their needs or preferences. In a conventional recommender system, users input their recommendations, which the system then combines and directs to the appropriate recipients. In certain cases, The main focus of some recommendation systems is on aggregating recommendations. In other situations, the system's value is primarily derived from its capability to facilitate accurate matches between those providing recommendations and those in search of recommendations [10].

A recommender system utilizes the information gathered from a user-item rating matrix in a specific domain. It analyzes and filters this data to extract the most relevant information. Based on this pertinent information, personalized recommendations are calculated for the user [11].

1.2.2 Definition of citation

In the context of recommender systems, a citation refers to the act of referencing or acknowledging the sources of information or research that are utilized or cited within the recommender system. It involves providing proper attribution to the authors or creators of the algorithms, techniques, datasets, or other relevant components incorporated into the recommender system. Citations in recommender systems serve the purpose of acknowledging and giving credit to the original authors, promoting transparency, and allowing users and researchers to explore the referenced sources for further understanding and evaluation.

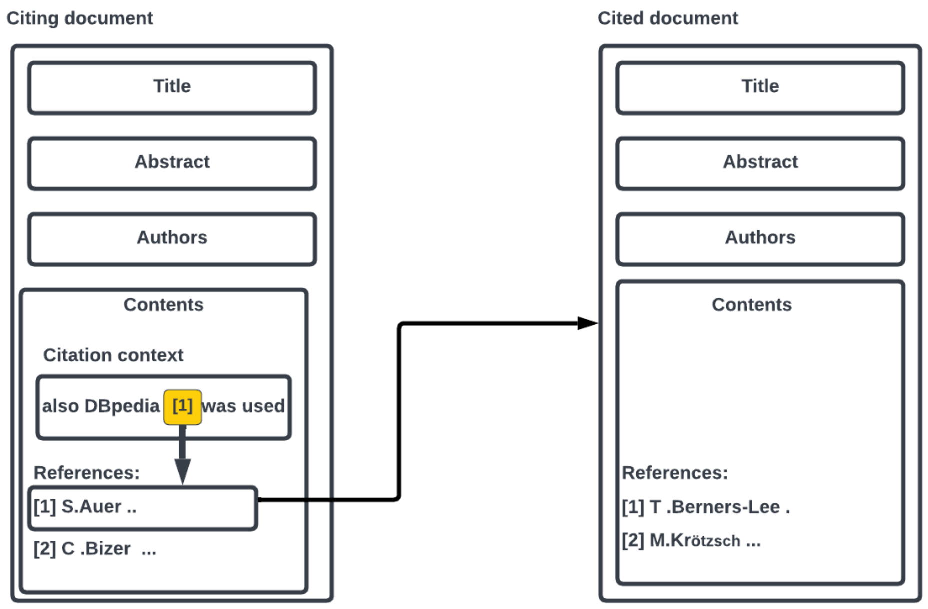


Figure :1.1 Visioning of a citation in a scientific paper[1]

1.2.3 Definition of citation Recommender system

A citation recommendation system is a specialized type of recommender system that suggests relevant academic or scholarly citations to users based on their research or writing needs. It assists users in finding and selecting appropriate references or sources to support their work, such as research papers, books, articles, or other scholarly materials. These systems utilize various techniques, such as text analysis, topic modeling, and semantic similarity, to analyze the user's input (such as the document or research topic) and provide tailored recommendations for citations that align with the user's requirements. The goal of a citation recommendation system is to enhance the research process by assisting users in discovering high-quality and relevant scholarly sources to support their academic work.

1.2.4 History

Recommendation systems (RS) are widely recognized as one of the most powerful tools in today's digital world, despite being a relatively new concept in research. However, the underlying concept of recommendations has been prevalent in society for a significant period.

Over the course of human evolution, our complex thinking abilities, language usage, and tool-making skills have developed over hundreds of thousands of years. Notably, the concept of recommendations can also be observed in various creatures, including humans and ants [12]. Drawing inspiration from the foraging behavior of ants, humans can develop algorithms for recommendation systems. Just like worker ants wandering randomly in search of food and leaving pheromone trails to guide other ants, Tapestry, the first recommendation engine created in 1992 at the Xerox Palo Alto Research Center, aimed to address the overwhelming amount of electronic mail in organizations. Electronic mail was the prevalent medium for information sharing among co-workers at that time, leading to difficulties in filtering out relevant messages based on individual tasks and preferences [13]. Tapestry was specifically designed to tackle this issue by filtering a continuous stream of electronic documents into a smaller, personalized selection that matched a user's interests. It organized and selected relevant documents to improve information management within the experimental mail system.

The field of recommender systems experienced a peak explosion in research when Amazon introduced its Collaborative Filtering method in the late 1990s, leading to a notable increase in their sales [14]. Amazon's success story also paved the way for the development of various approaches, including hybrid approaches that combine multiple methods. What sets Amazon's recommendation engine apart is its ability to adapt to the challenges posed by a growing customer base. Instead of focusing on individual customers and providing recommendations based solely on their past activities, Amazon began clustering customers with similar preferences. This approach ultimately yielded more accurate results [15].

After the successful period in the late 1990s, the industry witnessed a significant influx of funding for research in recommendation systems (RSs). One notable competition in the RS field was organized by Netflix, a renowned internet streaming media provider. In 2006, they launched the Netflix Prize, a competition offering a $1 million prize to the team that developed the best RS movie recommendation system. The winning team was announced in 2009 [14]. In 2010, YouTube also implemented its own RS on its website, recommending personalized sets of videos to users based on their activity on the platform [16]. With the rapid growth and advancement of Internet technology, the World Wide Web has become an enormous repository of billions of web pages, resulting in the problem of information overload for individuals. In addressing this challenge, recommendation systems have emerged as valuable tools that effectively filter, prioritize, and present relevant information. These systems have garnered increasing interest from both academia and industry, finding successful application in various fields such as e-commerce, movies, music, news, and e-learning [17]. In recent times, there has been a growing interest in the scientific community regarding the application of recommendation techniques. This field has gained significant attention due to the surge in the number of scholarly papers available on the web. The issue of recommending related scientific papers is known as the recommendation of a scientific article in the scientific community [18]. In digital repositories, the primary function of a recommender system is to offer recommendations for pertinent documents. The field of information science and communication technologies has witnessed notable advancements, resulting in a significant rise in electronic literature and the proliferation of scientific publications within digital libraries. Consequently, numerous academic digital libraries now house millions of digital objects encompassing various document types such as research papers, publications, journals, research projects, newspapers, magazine articles, and books. Online libraries play a vital role in supporting the scientific community by serving as repositories for storing and preserving valuable research data and findings. These digital libraries not only provide storage capabilities but also offer tools for organizing, searching, and retrieving the vast amount of content they contain. However, with the exponential growth of research articles in digital repositories, researchers are faced with the challenging task of locating relevant papers to study and cite in their own research. Typically, researchers rely on search engines like Google Scholar or Microsoft Academic, using specific keywords to retrieve relevant papers. They then manually review these papers to determine which ones are suitable for citation[19]. This process can be extremely time-consuming and demanding, particularly for junior researchers who may have limited experience in effectively searching for research articles.

1.2.5 Importance in Research communication

Recommender systems have emerged as valuable tools in research communication, offering significant benefits to researchers and the scientific community as a whole. The importance of recommender systems in research communication can be highlighted in several ways:

- **Enhancing Discoverability:** With the exponential growth of scholarly literature, it has become increasingly challenging for researchers to navigate and discover relevant research articles. Recommender systems address this issue by analyzing user preferences, citation patterns, and other relevant factors to suggest personalized and highly relevant research articles. By improving discoverability, recommender systems facilitate efficient literature exploration, enabling researchers to stay up-to-date with the latest advancements in their field.

- **Promoting Interdisciplinary Collaboration:** Research communication across different disciplines is crucial for fostering interdisciplinary collaboration and innovation. Recommender systems can play a pivotal role in bridging disciplinary boundaries by recommending relevant research articles from related fields. By exposing researchers to diverse perspectives and interdisciplinary research, recommender systems encourage cross-disciplinary collaboration and facilitate the emergence of novel research ideas.

- **Supporting Serendipitous Discovery:** Serendipity, the unexpected discovery of valuable information, is a vital aspect of research communication. Recommender systems can facilitate serendipitous discovery by suggesting articles that researchers may not have encountered otherwise. By broadening the scope of researchers' exposure to diverse research literature, recommender systems can stimulate creativity, spark new research directions, and contribute to breakthroughs in scientific knowledge.

- **Tailoring Recommendations for Researchers:** Each researcher has unique interests, preferences, and expertise. Recommender systems can leverage this information to provide personalized recommendations tailored to individual researchers' needs. By considering researchers' publication history, citation patterns, and collaboration networks, recommender systems can offer highly relevant and customized suggestions, assisting researchers in their exploration of new research avenues and fostering their professional development.
- **Facilitating Knowledge Exchange and Collaboration:** Recommender systems can facilitate knowledge exchange and collaboration within the scientific community. By recommending articles based on similarity, citation networks, and collaboration patterns, recommender systems can identify potential research collaborators, foster networking opportunities, and facilitate the sharing of expertise and resources among researchers. This promotes collaboration, accelerates scientific progress, and enhances the overall impact of research communication.

1.2.6 Benefits of Effective Citation Recommendation

Effective citation recommendation systems provide several benefits:

- **Improved Research Efficiency:** Citation recommendation systems help researchers save time and effort by suggesting relevant and appropriate references for their work. Instead of manually searching for relevant citations, researchers can rely on the recommendations to discover relevant sources more efficiently.
- **Enhanced Accuracy and Quality:** By suggesting appropriate citations, these systems can help improve the accuracy and quality of academic and scientific papers. Researchers can access a wider range of relevant and reliable sources, leading to more comprehensive and well-supported arguments.
- **Avoidance of Plagiarism:**Citation recommendation systems can help researchers avoid unintentional plagiarism by providing proper references for the sources they use. By suggesting appropriate citations, these systems ensure that researchers give credit to the original authors and avoid any potential ethical or legal issues associated with plagiarism.

- **Discovery of Relevant Literature:** Citation recommendations can introduce researchers to new and relevant literature in their field. These systems can suggest papers and articles that researchers might have otherwise missed, expanding their knowledge and helping them stay up-to-date with the latest research.
- **Increased Collaboration:**Effective citation recommendation systems can foster collaboration among researchers. By suggesting relevant work from other researchers, these systems facilitate the discovery of potential collaborators and encourage interdisciplinary connections.

1.2.7 Challenges in Citation Recommendation

While citation recommendation systems offer numerous benefits, they also face several challenges. Some of the key challenges include:

- **Ambiguity and Context:** Determining the appropriate citations for a given document can be challenging due to the ambiguity and context of the text. Understanding the intended meaning and context of a particular phrase or sentence requires deep comprehension of the content, which can be difficult for an algorithm to accurately capture.
- **Subjectivity and Diversity:** Different researchers may have varying preferences and perspectives when it comes to selecting citations.Recommendation systems must consider the subjective nature of citation choices and cater to the diverse needs of researchers from various disciplines and backgrounds.
- **Data Quality and Availability:** The quality and availability of citation data pose challenges for recommendation systems.Incomplete or inaccurate citation databases can limit the effectiveness of the recommendations.Moreover, not all papers are freely accessible, and access restrictions can hinder the availability of relevant citation information.
- **Multi-modal Recommendations:** Citation recommendation systems typically focus on text-based recommendations.However, research papers often include other forms of media, such as figures, tables, or datasets.Incorporating multi-modal recommendations that consider these additional elements poses a challenge for existing systems.

1.2.8 Role in Enhancing Scholarly Impact

Citation recommendation systems play a significant role in enhancing scholarly impact in several ways:

- **Improved Citations:** Citation recommendation systems can assist researchers in selecting appropriate and accurate citations for their work. By suggesting relevant and well-cited sources, these systems encourage researchers to cite influential and highly regarded papers, which can enhance the credibility and impact of their own work.

- **Strengthened Research Foundations:** Effective citation recommendations help researchers build a solid foundation for their research. By suggesting seminal or foundational papers in a specific field, these systems ensure that researchers are aware of and properly cite the key works that form the basis of their research. This strengthens the theoretical framework and intellectual lineage of the research, enhancing its scholarly impact.

- **Contextualization of Research:** Citation recommendation systems provide researchers with contextual information about how their work fits into the broader scholarly landscape. By suggesting related papers and prior works, these systems enable researchers to situate their research within the ongoing discourse, identify gaps, and contribute to existing knowledge in a meaningful way, thereby enhancing the impact of their research.

- **Facilitation of Reproducibility and Transparency:** Citation recommendations can include references to data sets, software, or methodologies used in research. This promotes reproducibility and transparency by providing researchers with the necessary information to verify and build upon previous work. Transparent and reproducible research practices contribute to the overall impact and trustworthiness of scholarly work.

1.3 Traditional Approaches for Citation Recommendation

Recommendation systems are capable of handling vast amounts of data and enable users to filter information based on relevance or personal preferences. These systems employ different filtering methods to offer recommendations and can be categorized into four main types: Content-based filtering (CBF) , Collaborative filtering(CF), Hybrid Filtering, Graph based Approaches as it mentioned in Figure1.2.

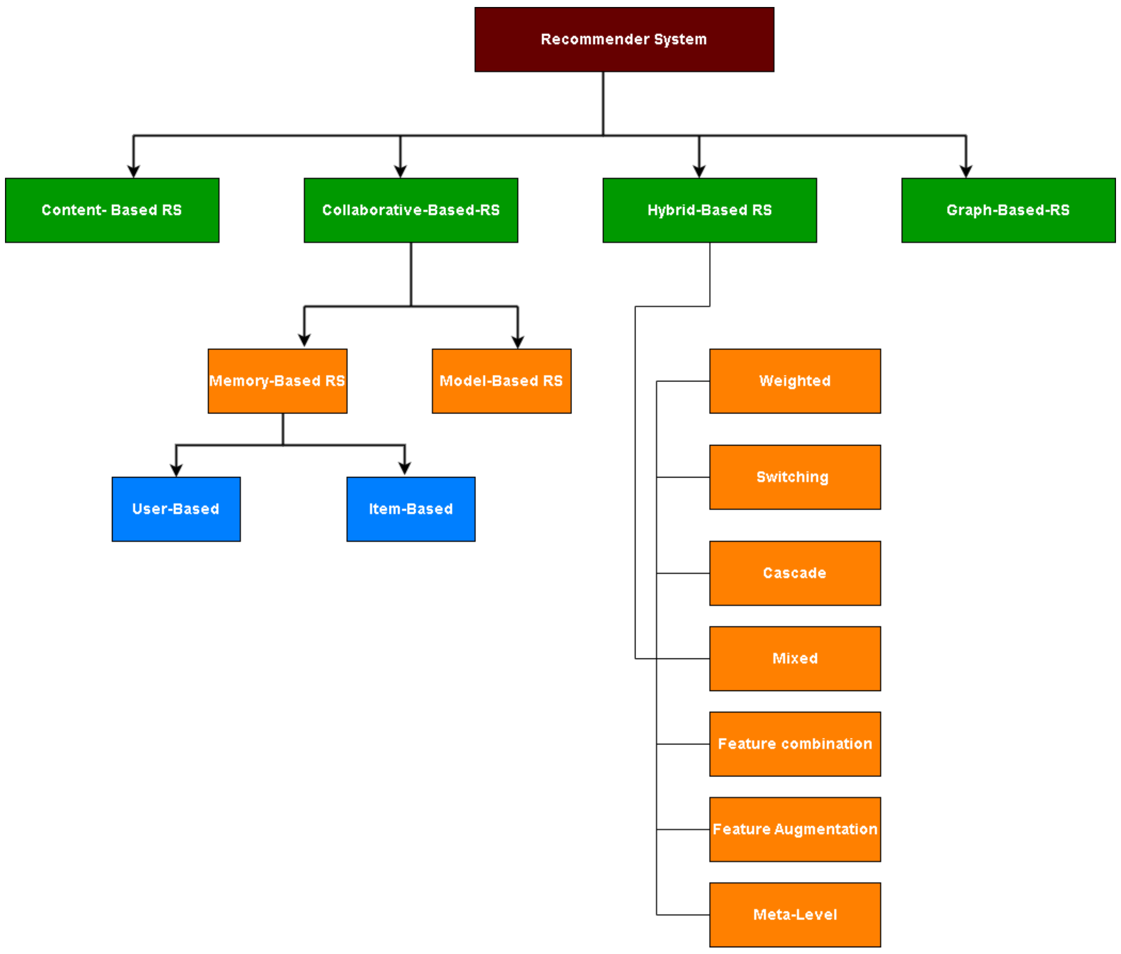


Figure :1.2 Recommendation systems approaches

1.3.1 Content based filtering

The content-based filtering approach in recommendation systems retrieves and recommends relevant documents by matching the features of document content with the user's profile. It operates by identifying papers that are similar to the ones a user has previously liked or positively rated based on their content.

The content-based filtering model is widely employed in recommender systems and serves as a fundamental approach in this field. It extensively utilizes text mining methods to analyze the content of papers. These methods involve content analysis to extract relevant papers and employ techniques from natural language processing (NLP), such as the well-known topic model called Latent Dirichlet Allocation (LDA). Information retrieval techniques, such as TF-IDF and bag-of-words, are also utilized. TF-IDF calculates the weighted importance of each word based on its frequency in the document and corpus, while the bag-of-words model does not consider grammar and word order, but rather focuses on word multiplicity. TF-IDF reflects the significance of a word within a specific document in a collection of documents or corpus [20].

According to [21], the main goal of the LDA algorithm is to analyze texts in natural language. Its purpose is to uncover the underlying topics present in the texts and represent them using probability distributions across words. Content-based systems have the benefits of being simple and efficient. However, they come with certain drawbacks. One challenge is the difficulty in accurately assessing the quality and style of the content in resources. Moreover, these systems may have limited ability to discover new resources that align with users' interests, as they tend to primarily recommend items that are similar to what users have already expressed interest in[22].

Basic structure of content-based systems

To enable effective comparison between items and user profiles, a content-based recommender system requires techniques that can generate efficient representations. In this regard, [23] propose a high-level architecture (Figure 1.3) that divides the recommendation process into three steps, with each step managed by a dedicated component:

❶ **Content Analyzer:**when the information is unstructured for instance, when an item is represented by text.This module intends to do pre-processing to extract the pertinent information, structure it, and represent it in a suitable target form such as a keyword vector.

❷ **Profile Learner:** This module gathers data indicative of the user's preferences and generalizes it In order to learn and create the user's profile. For this, machine learning techniques [24] can be applied. Neural networks, decision trees, and naive Bayes categorization are a few examples. These methods try to infer a user's profile based on information what they liked and not liked.

❸ **Filtering Component :** This module filters the pertinent items By comparing the user profile representation to the candidate items for recommendations.Utilizing similarity metrics between the item and the user profile, the relevance of the item is computed . The more closely the item resembles the "positive" profile and the less closely it resembles the "negative" profile, the more probable it is that the item will be recommended.

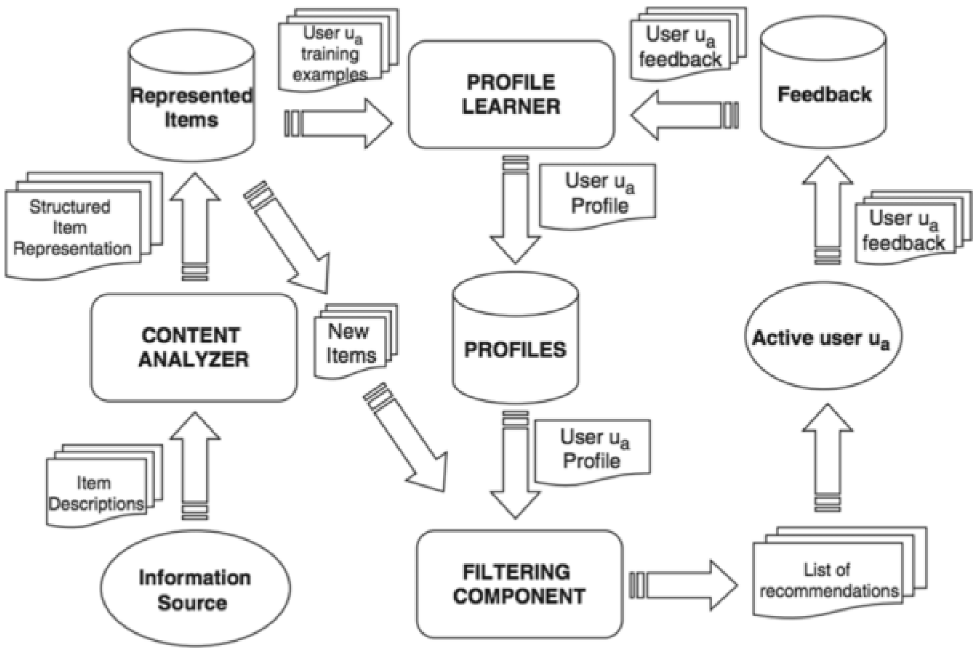


Figure :1.3 Architecture of a Content-based recommender system [2]

1.3.2 Collaborative based filtering

Collaborative Filtering(CF) has emerged as a highly effective technique in recommender systems, as highlighted by [25]. The fundamental idea behind collaborative filtering is that individuals who share similar interests in one domain are likely to have similar preferences for items or products in other domains as well [12].Collaborative filtering, which is a classical recommendation method, has found widespread application in various domains,particularly in e-commerce platforms like Amazon. Additionally,researchers have also employed this method for recommending scientific literature. Collaborative filtering methods play a crucial role in recommending relevant papers to users by considering not only their own interests but also the behavior of other users who have similar preferences. CF methods in scientific paper recommendation systems generate recommendations by discovering the connections between researchers who share common research interests.These connections are identified through factors such as the readership patterns and shared publications among researchers.By leveraging these correlations,CF approaches offer personalized recommendations for scientific papers [19].Collaborative filtering' process depicted in Figure 1.4.

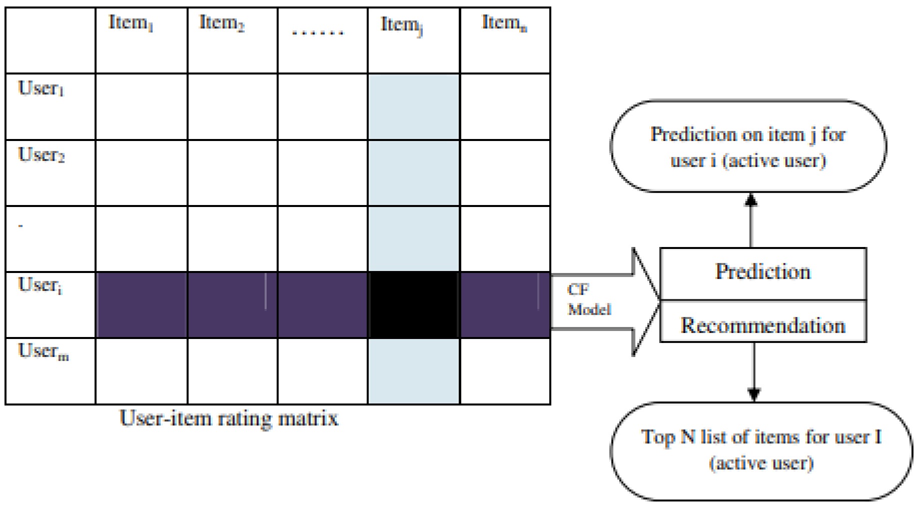


Figure :1.4 Collaborative filtering process [3]

Collaborative filtering models categorized into two types: memory-based CF and model-based CF.

✳ **Memory-Based CF:** The items that have been previously rated by a user play a significant role in finding similar neighbor who share similar preferences. Once a user's neighbors are identified, various algorithms can be employed to aggregate the preferences of these neighbors and generate recommendations. Memory-based collaborative filtering can be categorized into two techniques: User-Based CF and Item-Based CF.

- **User-Based technique:** This approach entails comparing the evaluation data of users for the same item in order to identify similarities between them. By leveraging the ratings of similar users, the system can provide recommendations for items that are likely to match a user's preferences [26]. Similarities are typically calculated using metrics like cosine similarity. However, this approach can encounter common challenges, such as data sparsity and cold-start issues. Data sparsity refers to situations where only a small portion of the available items in a database are rated by users, leading to limited data for accurate recommendations [3].On the other hand, the cold-start problem arises when a new item that hasn't been rated by any user needs to be recommended.

- **Item-based technique**In item-based collaborative filtering, the prediction of unknown ratings is calculated by considering items that are similar to the specific item for which the rating is being predicted [27]. This method is known to be more stable compared to user-based collaborative filtering and helps address the issues faced by the latter approach

✳ **Model-Based CF:** The core concept of the model-based approach is to develop a model by utilizing user or characteristics and rating information. This trained model is then employed to predict the potential rating of the target user [28]. Model-based collaborative filtering commonly leverages techniques such as matrix factorization and clustering algorithms to achieve accurate predictions. These methods are extensively utilized in model-based CF to enhance the effectiveness of recommendations.

1.3.3 Hybrid based filtering

Hybrid filtering approaches combine multiple recommendation techniques to create more effective recommendation systems for scientific papers, addressing the challenges and limitations of pure recommendation systems. These hybrid recommender systems typically integrate content-based filtering and collaborative filtering methods, leveraging the strengths of both techniques while mitigating their weaknesses. The combination of content-based filtering and collaborative filtering methods enhances the accuracy of the recommender system compared to using a single algorithm alone [29]. Hybrid recommender systems have been successfully applied in various domains such as movies, education, music, and web services. In the context of movie recommendations, there is a wide range of algorithms and solutions available [30]. The hybrid recommendation model can be categorized into seven types: weighted hybridization, switching hybridization, cascade hybridization, mixed hybridization, feature combination, feature augmentation, and meta-level, depending on the method used to combine filtering techniques[26].

- **Weighted Hybridization:** Weighted hybridization is a technique that combines the outcomes of different recommenders to generate a recommendation list or prediction. It integrates the scores obtained from each technique using a linear formula. P-tango, for instance, serves as an example of a weighted hybridized recommendation system [3]. The underlying principle of this method involves gradually adjusting the weights based on the level of agreement between the user's evaluation of an item and the evaluation predicted by the recommendation system.

This enables the system to adapt and provide more accurate recommendations over time.

- **Switching Hybridization:** The switching hybrid system utilize a set of criteria to switch between recommendation techniques related to the situation. The strengths and weaknesses of the recommenders that make up this system are taken into consideration which means thta this system is particularly responsive to them.
- **Cascade Hybridization :** Cascade hybrids employ a two-step process for generating recommendations. Initially, one approach is utilized to generate an initial set of recommended items. Subsequently, a second approach is employed to carefully select the most appropriate items from that initial set, resulting in a final recommendation [18].
- **Mixed Hybridization:** Mixed hybrids involve combining the recommendation outcomes of multiple recommendation techniques simultaneously, rather than providing a single recommendation per item. An instance of the mixed hybridization approach is the PTV system, as discussed by [31].
- **Feature-Combination :** Feature-combination hybridization scheme combines collaborative and content information, with a focus on effectively utilizing the collaborative information. The hybridization process yields a standalone and improved system that possesses knowledge about the both genres of information as described by [32].
- **Feature-Augmentation:** The technique utilizes the ratings and information generated by the previous recommender as input features in the primary recommendation system, contributing to the generation of the final predicted outcome.
- **Meta-level:** Meta-level hybrids leverage the internal model produced by one recommendation technique as input for another, resulting in a more comprehensive information model compared to relying solely on a single rating. Furthermore, Another advantage of meta-level hybrids is their ability to effectively overcome the issue of data sparsity commonly found in collaborative filtering techniques, as emphasized by [3].

1.3.4 Graph based filtering

According to [33], a graph consists of two sets: a set of nodes and a set of edges connecting those nodes. The construction of graphs is a central focus in graph-based approaches, which involve creating graphs from various sources such as citation networks and social networks. In these graphs, researchers and papers serve as nodes, and the relationships between researchers, researchers and papers, and papers and papers are represented as edges [29]. By utilizing walking algorithms on the graph, recommender systems can leverage information from diverse sources to suggest relevant papers to researchers. One of the key advantages of the graph-based approach is its ability to find suitable papers by integrating information from multiple sources.

1.3.5 Limitation of Traditional Approaches

While recommendation systems have proven to be valuable, there are several limitations associated with their techniques. Some common limitations include:

- **Sparsity Problem:**The sparsity problem is a significant challenge faced by recommender systems, and it significantly impacts the quality of recommendations.Typically, systems like MovieLens represent data in the form of a user-item matrix populated with ratings given to movies.As the number of users and items increases, the dimensions of the matrix grow, leading to increased sparsity.The main reason for data sparsity is that most users do not rate most items, resulting in sparse available ratings. Collaborative filtering techniques are particularly affected by this problem since they heavily rely on the rating matrix.Despite efforts by researchers to address this issue, as evident in studies [34], [35], [36],there is still a need for further research in this area to mitigate the impact of data sparsity on recommendation quality.
- **Cold Start problem:**When a recommendation system encounters a new user or item, it faces a problem known as the cold start problem.It encompasses three types of cold start problems: the new user problem, new item problem, and new system problem.These scenarios make it difficult to provide recommendations due to limited information available. In the case of a new user, there is minimal user data available, making it challenging to understand their preferences and generate personalized recommendations. Similarly, for a new item, there are typically no ratings or interaction data to rely on, which poses a challenge for collaborative filtering techniques that heavily depend on such information. However, content-based methods offer a solution in the case of new items since they don't rely on previous ratings information of other users.
- **Scalability:**Scalability refers to the ability of a system to effectively handle an increasing amount of information in a smooth and efficient manner.As the internet has witnessed an enormous growth in data,recommender systems are faced with the challenge of managing this explosion of information while meeting the growing demands of users. Many recommender system algorithms involve computations that become more complex as the number of users and items increases.In collaborative

filtering (CF), these computations can grow exponentially, leading to higher costs and potentially inaccurate results.To address the scalability problem, various methods have been proposed that rely on approximation mechanisms to speed up the recommendation process. However, while these methods can improve performance and computational efficiency, they often leads to reduced accuracy, as mentioned in [37].

- **Over Specialization Problem:** In certain cases, users may face an issue known as the over-specialization problem, where they are limited to receiving recommendations that closely resemble the items already known or defined in their profiles [36]. This restricts their ability to discover new items and explore alternative options. However, diversity in recommendations is considered a desirable feature in all recommendation systems. By utilizing genetic algorithms to address this problem, users can be presented with a diverse and extensive set of alternatives, expanding their range of choices.
- **Robustness of RSs:** RSs face a significant challenge in terms of their robustness against attacks, which is a crucial performance metric.In order to achieve certain benefits, attackers may employ various attack models, such as Push/Nuke Attacks, to create fabricated user profiles.These attacks aim to artificially boost or diminish the popularity of specific target items, respectively. The term used to describe these types of attacks collectively is "shilling attacks" or "profile injection attacks."[38]

1.4 Deep Learning Techniques In Recommendation Systems

1.4.1 Introduction to Deep Learning

Deep learning has emerged as a prominent field within machine learning, leveraging artificial neural networks to learn multiple layers of representations. This hierarchical approach allows for the definition of higher-level concepts based on lower-level ones [39]. The efficiency of training deep models was introduced by [40] demonstrated the capabilities of deep architectures in complex artificial intelligence tasks in 2009, leading to the growing popularity of deep learning in computer science. Today, deep learning approaches are at the forefront of solving problems in computer vision, natural language processing, and speech recognition [39]. Although the concept of neural networks and deep models has existed for over 50 years, it is in the last decade that the power of deep learning techniques has become evident. Several factors have contributed to the prominence of deep learning as the leading machine learning technique:

- Big Data: Deep learning models benefit from large amounts of data, as they can learn more accurate representations from a larger and more diverse dataset.
- Computational Power: The availability of powerful graphical processing units (GPUs) has provided the necessary computational resources to perform complex computations required by deep learning models. GPUs excel at parallel processing, significantly accelerating the training and inference processes.

1.4.2 Neural Networks for Recommendation Systems

Deep neural networks have demonstrated significant progress in various domains such as image processing, voice recognition, natural language processing, and recommendation systems.

There are two major categories for classifying deep neural-based recommender systems[41]:

1. Recommender systems that solely employ deep learning approaches for generating predictions.
2. Recommender systems that combine traditional recommendation techniques with deep learning approaches.

Deep learning approaches commonly used for recommender systems include Convolutional Neural Networks (CNNs), Multi-Layer Perceptrons (MLPs), Autoencoders (AEs), and Recurrent Neural Networks (RNNs).

- Convolutional Neural Networks (CNNs)are feed-forward neural network [42]are typically used in computer vision and natural language processing tasks. They employ convolutional layers to extract local features and pooling layers to generate concise representations. In recent years, CNNs have also been applied to recommender systems.

- Multi-Layer Perceptrons (MLPs) are feed-forward neural networks that consist of an input layer, one or more hidden layers, and an output layer. Non-linear activation functions are applied to all nodes except for the input nodes. MLPs use backpropagation for training and have been employed in recommender systems.

- Autoencoders (AEs) are unsupervised neural networks that learn to reconstruct the input data. They typically have three layers: an input layer, a bottleneck layer (which captures the compressed representation), and an output layer. AEs have gained popularity in recommender systems in recent years.

- Recurrent Neural Networks (RNNs) are utilized for learning sequential data[43]. While RNNs have been widely applied in natural language processing and speech recognition, they have also found use in recommender systems. RNNs are able to retain information about a sequence over time, and a well-known variant is the Long Short-Term Memory (LSTM) model.

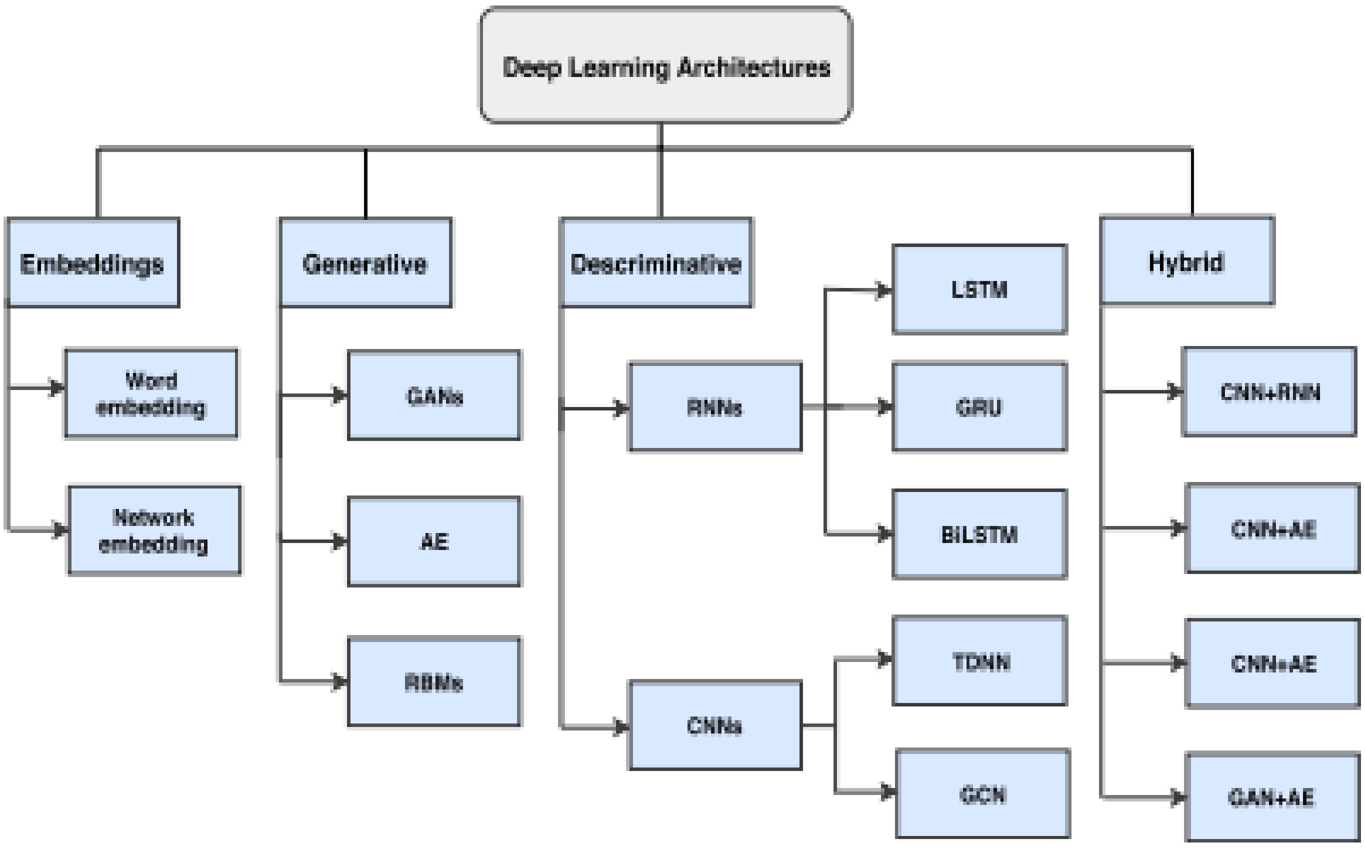


Figure :1.5 Deep learning techniques [4]

1.4.3 Convolutional Neural Network in Recommendations

CNNs (Convolutional Neural Networks) are neural networks that utilize convolution operations in at least one of their layers. While CNNs are commonly associated with tasks such as image recognition and object classification, they also offer benefits to recommender systems. In the context of recommender systems, CNNs are utilized for various purposes, including extracting latent factors from different types of data.

[44] employed CNNs to extract latent factors from audio data when these factors cannot be directly obtained from user feedback. By applying convolutional operations on the audio data, CNNs can capture relevant patterns and features that contribute to the recommendation process.

[45] utilized CNNs to extract latent factors from text data. Textual information, such as product descriptions or user reviews, can be processed using CNNs to derive meaningful representations that capture important semantic features. These latent factors can then be used for generating recommendations based on text data.

[46] focused on extracting visual features for generating visual interest profiles of users. CNNs were employed to analyze images and extract relevant visual features that reflect users' preferences. By understanding the visual aspects of user interests, more accurate recommendations can be generated.

[47] utilized CNNs to extract latent features from images and map them to the same latent space as user preferences. By aligning image features with user preferences, the recommender system can make personalized image recommendations that align with the user's interests. In the context of context-aware recommender systems, CNNs are utilized to extract semantic meanings from textual information. [48] leveraged CNNs to extract meaningful semantic representations from text, allowing for more qualified and contextually relevant recommendations.

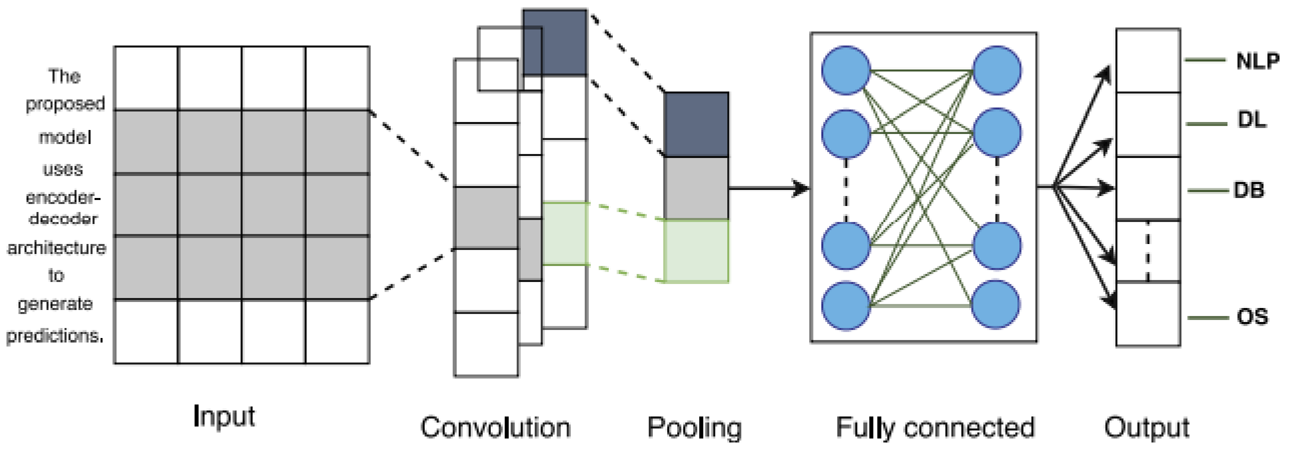

The proposed model uses encoder-decoder architecture to generate predictions.
Input
Convolution
Pooling
Fully connected
Output
NLP
DL
DB
OS

Figure :1.6 Convolutional neural network architecture [4]

1.4.4 Recurrent Neural Network in Recommendations

In e-commerce systems, a user's browsing history is an important factor that influences their purchase behavior. However, traditional recommender systems often overlook the current history and the order of user actions by creating user preferences at the beginning of a session. To address this limitation, RNNs (Recurrent Neural Networks) have been utilized in recommender systems to integrate the user's current viewing web page history and the sequential order of their actions, resulting in more accurate recommendations [49],[50],[51].
Wu et al[49] proposed a method that combines the results of an RNN with the output of a feed-forward neural network. This approach takes into account the correlations between users and items, leveraging the RNN to capture temporal patterns in user behaviors. By integrating the RNN representations with latent factors of user preferences, the model produces more precise predictions for personalized recommendations. Ko et al[52] also employed RNNs to capture the temporal and contextual aspects of user behaviors. By representing user actions and preferences using RNNs, they enhanced the accuracy of recommendations. The RNN representations were combined with latent factors of user preferences to create a comprehensive recommendation model.

RNNs have been utilized in recommendation systems not only to capture sequential user behaviors but also to represent the influence between users' and items' latent features and their coevolution over time [53]. This approach allows for non-linear representations of the complex relationships between users and items. In the work of Devooght and Bersini [54], RNNs were employed to integrate the evolution of user tastes into the recommendation process. They framed the recommendation problem as a sequence prediction task.

Analyzing studies on deep learning in recommender systems reveals several key findings. One notable finding is that RNNs (Recurrent Neural Networks) have demonstrated positive effects on the coverage of recommendations and short-term predictions, particularly when compared to conventional approaches like nearest-neighbor and matrix factorization-based methods. This success can be attributed to RNNs' ability to account for the evolution of users' tastes and the coevolution between user and item latent features [54] [53].RNNs are particularly well-suited for session-based recommender systems and the integration of users' implicit behaviors into their preferences

1.4.5 Transformer models in Recommendations

Transformers models initially introduced in natural language processing (NLP) tasks . BERT is one of these models, it has been adapted and applied to recommendation tasks

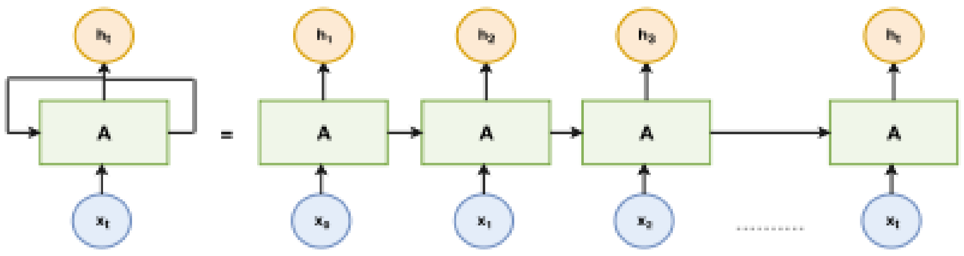


Figure :1.7 Reccurent neural network architecture [4]

due to their ability to capture complex patterns and dependencies in data.The BERT model undergoes pre-training with two distinct learning objectives, which compel the model to grasp semantic information at both the within and between sentences [55].

[56] introduced a context-aware model for citation recommendation that combined BERT [57]and a variation of GCN [58]. They utilized pre-trained BERT as a context encoder to generate textual embeddings, while the GCN model was employed to generate graph embedding. By combining these two components, the model can provide more accurate and relevant recommendations that consider both the content and the citation relationships between papers.

1.5 Context Awareness in Recommendation Systems

Researchers in a variety of disciplines, such as information retrieval, ubiquitous computing, marketing, and management, have acknowledged the value of contextual information. However, contextual data has not been extensively used in recommender system research. In some fields, information like time, location, and the company of other people might enhance the recommendation process. Users and items are the only two types of entities that traditional recommender systems work with. However, it might not be enough to take users and items into account for certain applications, such as recommender systems for travel. Integrating information about context is frequently crucial. For instance, to produce an appropriate recommendation, a system of recommendations for vacation hotels must consider the season. Similar to this, a mobile device-based tourist recommendation system may privilege recommending attractions and activities near the user.

1.5.1 Understanding Context in Recommenations

Context is a broad concept that is particularly hard to give an overall and practical definition.We quote the definition provided by [59], which is one of the most extensively admitted:

"Context can be defined as any information that can be utilized to describe the status of an entity.Entities include both users and apps themselves and any other people, places, or things that are thought to be important to the interaction between a user and an application". However, Zimmermann [60] has questioned this description as being too general and non-operational.

According to Zimmermann[60], "context is any information that can be utilized to describe the status of an entity .The elements used to describe this context information can be divided into five groups: action, individuality, location, time, and relations".

1.5.2 Types of Contextual information

Any information that describes entity's context falls into one of the five distinct types, as illustrated in Figure 1.8 .

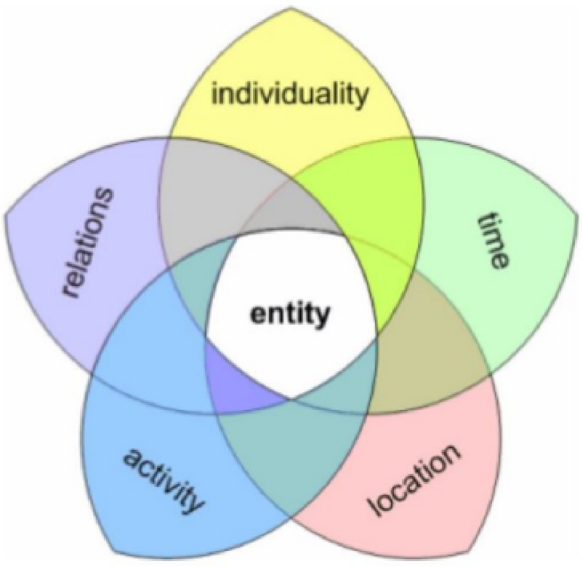


Figure :1.8 The five fundamental elements of the Context. [5]

- **individuality:** This category gives access to contextual information about the entity to which it is linked. This information includes everything that can be observed about an entity and especially its state. Entities can be categorized into active, to manipulate other entities, or passive, real or virtual, mobile or fixed.
- **relations:** this category of contextual information captures the relationships established between the given entity and another entity that may be a person, service or information.
- **activity:** This context covers the entitys present and future activities. In most scenarios of interactions with context sensitive systems, the entity is engaged in a task that determines the purpose of the activities performed
- **location:** location is one of the main parameters of a context-sensitive system. Physical objects move in ubiquitous environments. This category describes location models that classify the physical or virtual location (e.g. IP address) of an entity as well as other spatial information: speed, orientation, etc.

- **time:** time is an indispensable aspect for understanding and classifying context. This includes information about the time a process takes to run, the interval between two recurring events, etc.

1.5.3 Incorporating Context in Collaborative Filtering

In traditional recommender systems, there are several approaches used to cater to user preferences, including collaborative filtering [61][62], content-based [23], knowledge-based [63], and hybrid [64] methods. However, in the case of Context-Aware Recommender Systems (CARS), researchers have focused on integrating contextual information into these conventional recommendation approaches, with particular emphasis on collaborative filtering techniques.

Adomavicius and Tuzhilin [6] have identified three main approaches to incorporating contextual information in a recommender system as it shown in Figure 1.9, based on the stage at which the context is introduced. These approaches can be summarized as follows:

- **Contextual pre-filtering**
- **Contextual post-filtering**
- **Contextual modeling**

Let's briefly outline these three approaches.

- **Contextual pre-filtering** Incorporating context by pre-filtering or pre-processing allows choosing a subset of data that is momentous for context where the individual is located and limiting the recommendation process to this subset . Building a model for each context is implied by this. Let's use the example of a movie recommendation system that leverages the temporal context to demonstrate this strategy: if a user wishes to watch a movie during the weekend, only the weekends' movies are recommended as candidates. Candidates for recommendations are those that are available during the weekend, and only the ratings of users who have seen the films over the weekend are utilized for ratings anticipation. As the data set is decreased and can cause issues for score prediction if the system does not have enough data, the use of this a priori filtering has been criticized.

- **Contextual post-filtering** When using a contextual post-filtering strategy, the recommender system does not consider the contextual data while making recommendations . The results of the recommendation algorithms are edited a posteriori in order to reorganize the list of recommended items according to context. For instance, a recommendation system for tourist destinations may utilize the user's geographic position (location context) and might determine to call off a posteriori recommendations for locations that are too far away from the user's location .

- **Contextual modeling** The context modeling strategy involves incorporating contextual information right into the process of making recommendations for item score prediction . Tensor factorization techniques are suggested by [65] to incorporate context. For these methods, each genre of context is taken into account as a new dimension in addition to the first two dimensions that are traditionally utilized for items and users. The score is now is not regarded as a function with the two parameters : user and item but it is function with the parameters :item ,user and context.

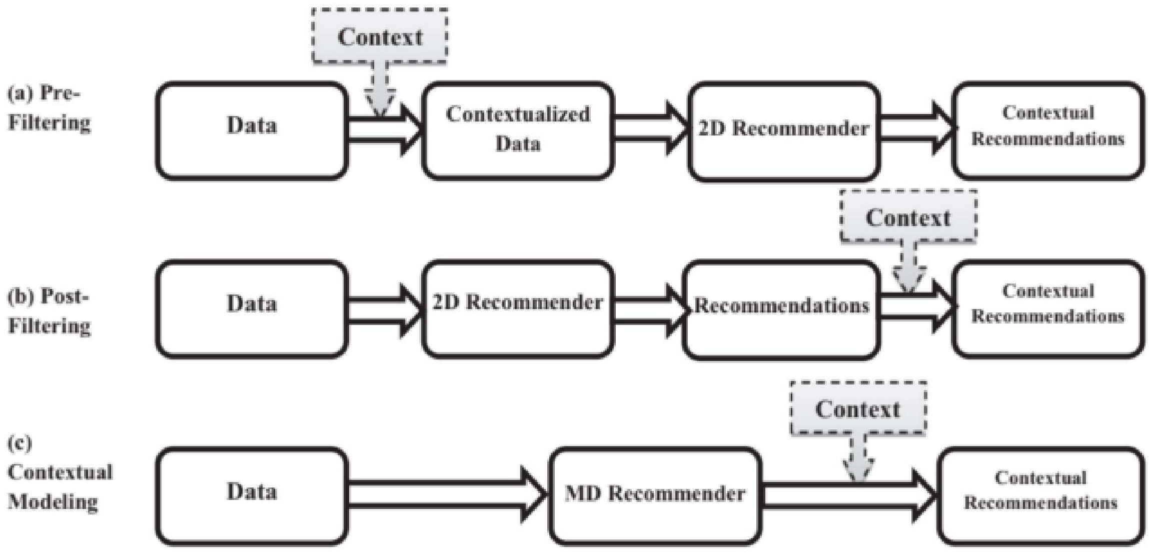


Figure :1.9 Context incorporation in Recommender system process [6]

1.5.4 Context modeling for recommender systems

Traditional recommendation systems are two-dimensional (2D) because they solely take into account the dimensions of the user and the item.The context is viewed as an extra information in the context-integrated recommendation system.We include the context dimension in addition to the user and the item, which will help the system's recommendation function better.Consequently, score functions in the form will be taken into account by a context-sensitive recommendation system.

$$R : UserxItemxContext \Rightarrow Rating$$

The rating function which symbolized with R is if we characterize contextual information using a set of contextual dimensions D,two of it are user and item ,the rest are contextual.

$$R : D1\text{x}............\text{x}Dn \Rightarrow Rating$$

As we can notice, contextual information might be of variety of aspects, included time and location. Additionally, each contextual side may have a sophisticated structure that reflects the nature of contextual information (like text, for example). Contextual information is often hierarchical and represented as a tree to handle this complexity.

1.5.5 Context Awareness

Overall, The term contextual awareness refers to the app's ability to discover and utilize contextual data, for instance the location of user and contiguous tools. The concept of context-awareness was initially put forth by Shilit [66], who described it as an application's ability to discover and respond to changing in the user's environment. According to Brown [67], they are "applications that may vary their behavior based on the user's situation", moreover to Dey [59] who defines a context-aware application as one that takes advantages of contextual information to provide the user with information and services that are pertinent for them, with pertinence decided by the user's labour.
A sundry of factors need to be considered in order to take an effective advantage of the context and set up a trusty way to develop context-aware services. number of challenges have arisen [68]:

- **Context representation:** Suggesting a high-level abstraction-based representation.
- **Context capture:** Getting the user's context's characteristics.
- **Context management:** Context management comprises addressing aspects that are not functional.
- **Context interpretation and reasoning:** The reasoning around the context is to draw context from the current context at a high semantical level of service adaptation Service adaption: Services must be operated and adjusted through productive scenarios.
- **Context reuse:** Utilizing contextual features to claiming for the expiration of their validity.

1.6 Evaluation Metrics for Citation Recomendation Systems

Within the field of recommendation systems (RSs), researchers and professionals are primarily concerned with user satisfaction and ensuring that the recommendations provided deliver maximum value to the users.It is important for RSs to go beyond suggesting more of the same and instead offer suggestions that are genuinely useful to the users.Therefore, researchers focus on evaluating various aspects of user interaction and consumption experience within the system.To address this challenge, researchers have started exploring evaluation concepts beyond traditional measures of predictive accuracy and machine learning techniques.
The performance of CRSs should be measured based on the value they bring to the users. Consequently, different evaluation concepts have emerged in the research community, including Precision,Recall,Mean Average Precision(MAP),coverage, novelty, diversity, and surprise of recommendations .These concepts have been thoroughly examined and assessed by researchers to understand their impact on the quality of recommendations.

1.6.1 Accuracy Metrics

There are several types of accuracy metrics commonly used in different domains such as citation recommendations systems .

- **Recall:** The recall metric is used to determine the percentage of relevant papers that are included in the top-k recommendations.A higher recall in lower top-k values indicates that the system is able to capture a greater proportion of relevant papers, resulting in more robust and comprehensive results[69].

$$Recall = \frac{1}{Q}\sum_{j=1}^{Q}\frac{R_p \cap T_p}{T_p}$$

  where Q represents the total number of target articles and R_p denotes the list of top-k recommendations produced against a target paper p.

  It can be represented also as follow:

$$Recall = \frac{|\text{Relevant papers recommended }|}{|\text{All relevant papers}|}$$

- **Precision:** The precision of a recommendation is defined as the ratio of the relevant papers recommended divided by the all the recommended papers.

$$Precision = \frac{|\text{Relevant papers recommended }|}{|\text{All recommended papers}|}$$

- **F1 score:** The F1 score is a metric that combines precision and recall by taking their harmonic mean. It serves as a summary of both precision and recall.[70]

$$F1 = 2.\frac{Precision * Recall}{Precision + Recall}$$

1.6.2 Ranking Metrics Metrics

Ranking metrics are commonly used to evaluate either the quality of the generated recommendations and assess how well they are ranked or ordered.These metrics focus on the position or order of relevant items within the recommendation list.

1.6.3 Mean Average Precision(MAP)

MAP : is a metric used to evaluate the quality of recommendations in terms of whether relevant papers are included in the top-K results or not.To calculate the Average Precision (AP) for a specific query paper, the precision is computed after each Ground Truth Positive (GTP) is retrieved. The AP is then determined by taking the mean of these precision values[69].

$$AP@K = \frac{1}{GTP}\sum_{i=1}^{K}\frac{TPseen}{i}$$

TPseen refers to the count of true positive items that have been encountered or seen up to position K within the recommendation list.

1.6.4 Diversity, Novelty an Serendipity Metrics

- **Diversity:** Ziegler et al.[71] introduced a metric for assessing diversity in recommendation lists by measuring the intra-list similarity, which is essentially a measure of dissimilarity between the items. Diversity can be define as the inclusion of dierent types of item set in recommendation for user which is dierent from their past preferences. The calculation of diversity often involves using an intra-list similarity measure [72].

$$diversity = \frac{1}{2}\sum_{i_j \in u}\sum_{i_k \in u} sim(i_j, i_k)$$

The term "sim(i_j, i_k)" represents the similarity measure between two items, i_j and i_k, that are commonly rated by a user, u.

- **Novelty:**Novelty is a key qualities of recommendation systems that contributes to their effectiveness and the addition of new items to their lists, both of which increase accuracy[72].

$$Novelty = \frac{U_x}{U_i}$$

  U_x is the item set which is unknown to user and $U_i$is the item set that is liked by the user U[73].

- **Serendipity** Serendipity refers to how surprising or pertinent recommendations are made for the user. Serendipity is computed as the difference of the probability of an item being recommended specifically for a user and the probability of that item is recommended for any other user[72].

$$Serendipity = \sum_u \frac{RS_u \cup E_u}{|E_u|}$$

  Where the recommendation generated for user "u" is denoted as RS_u and the item set of user "u" is represented as "E_u. The complete item set of user is denoted as |N|.

1.6.5 Challenges and Considerations in Evaluation Metrics for Citation Recommendation

When it comes to evaluating metrics in citation recommendation systems, there are several challenges and considerations to keep in mind. Here are some key ones:

- **Lack of ground truth:**Obtaining an accurate ground truth dataset for citation recommendation is a significant challenge. It requires manual annotation by experts, which is time-consuming and expensive. Additionally, different experts may have varying opinions on what constitutes a relevant citation, leading to inconsistencies in the ground truth labels.

- **Subjectivity and context:**Evaluating citation recommendations often involves subjective judgments, as the relevance of a citation can vary depending on the specific research context. Different researchers may have different preferences or interpretations of relevance, making it challenging to define a universally accepted evaluation metric .

- **Evaluation scale:**Determining the appropriate scale for evaluating citation recommendations is important. Binary evaluation metrics (e.g., precision, recall) may not capture the nuances of relevance, while continuous metrics (e.g., ranking-based metrics like mean average precision) require extensive manual judgments and may not reflect real-world user behavior.

- **Generalizability:**Citation recommendation systems often need to be evaluated on different datasets and research domains to assess their generalizability. Evaluating system performance across diverse domains and datasets is crucial to understanding their robustness and effectiveness beyond specific contexts

1.7 Conclusion

In this chapter, we delved into the topic of citation recommendation systems. We started by defining recommendation systems and then specifically focused on citation recommender systems. We explored the history of citation recommendation and discussed its importance in research communication. The benefits of effective citation recommendation are highlighted, along with the challenges involved in this task. Additionally, we discussed the role of citation recommendation in enhancing scholarly impact.

Next, we presented the traditional approaches used in citation recommendation, including content-based filtering, collaborative filtering, hybrid approaches, and graph-based techniques. We also acknowledged the limitations of these traditional methods.

We then shifted our focus to deep learning techniques in recommendation systems. We provided an introduction to deep learning and explored how neural networks, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models, can be applied to recommendation tasks. Context-awareness in recommendation systems is another aspect we covered. We explained the concept of context in recommendations, delved into different types of contextual information, and discussed how context can be incorporated into collaborative filtering. We also touched upon context modeling for recommender systems and the overall importance of context awareness. Evaluation metrics for citation recommendation systems are then discussed, including accuracy metrics like precision, recall, and F1-score, as well as ranking metrics such as mean average precision (MAP). Additionally, we considered novelty, diversity, and serendipity metrics and the challenges faced when evaluating citation recommendation systems.

CHAPTER 2

RELATED WORK

2.1 Introduction

Related Work provides a comprehensive overview of the existing literature and research efforts in the field of citation recommendation. This chapter serves as a foundation for the development of the proposed deep learning context-aware technique for citation recommendation. By examining prior work and understanding the strengths and limitations of existing approaches, we can identify research gaps and opportunities for innovation.

In this chapter, we classify and analyze related works into three distinct categories: data representation methods, methodologies and models, and personalization techniques. By categorizing and examining prior work within these categories, we aim to gain a comprehensive understanding of the advancements made, challenges faced, and opportunities for improvement in the domain of citation recommendation.

The objective of this chapter is to provide a systematic and structured review of the various approaches and techniques used in citation recommendation systems. By classifying related works into these three categories, we can examine different aspects of the research landscape and identify key contributions and trends.

The first category, data representation methods, focuses on how citation data is represented and transformed to extract meaningful features. The second category, methodologies and models, delves into the different approaches and algorithms employed in citation recommendation. The third category, personalization, focuses on tailoring citation recommendations to the specific needs and preferences of individual researchers as shown in Tables (2.1,2.2,2.3).

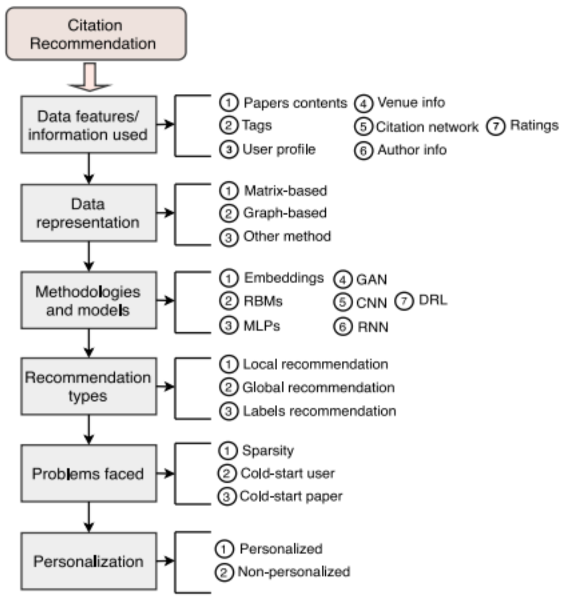


Figure :2.1 An overview of algorithms taxonomy [4]

2.2 Recommendation model classification

In this section, our initial focus is to investigate and classify 35 citation recommendation models that utilized deep neural networks. We classify these models based on three specific criteria: the representation of the data, the methodologies employed, and the type of personalization. The classification results are presented in Tables (2.1; 2.2; 2.3).
The purpose of this multilevel taxonomy is to thoroughly analyze the strengths and weaknesses of the examined models and identify trends within specific domains. Additionally, we provide a comprehensive overview of the contents presented in tables (2.1; 2.2; 2.3), offering detailed insights into the characteristics and details of each model.

2.2.1 Data representation methods

This section discusses a range of data representation techniques utilized in the studies under investigation.

- **Matrix-based:** is referred to algorithms that utilize matrix to represent data. Specifically, these algorithms employ a user-item rating matrix, where each entry represents the interaction between user i and item j. This approach is particularly relevant in the context of implicit feedback, where the focus is on whether a user has interacted with an item rather than explicit ratings. The elements within this matrix can take the value of 1 to indicate that a user has interacted with a specific

item, or it can be 0 if there was no interaction.
The model employs the items R_i^+ that have been interacted with by the user in order to propose a list of the top K relevant items from a pool of unseen items, denoted as R_i^-.

In this context, the Multi-model Adversarial Auto-encoder (MAAE) model developed by [74] utilizes a ratings matrix $x(0, 1)$ that represents implicit feedback based on the interactions between a document j and other nodes k.
In contrast to the conventional user-item rating matrix $[UI]$, the approach considered in this study involves treating research papers as users, with authors being associated with those papers. The rationale behind this approach is that an author can collaborate on multiple research works across different domains, but all authors associated with a specific paper should receive similar recommendations. Likewise, a research paper should receive a consistent set of recommendations for potential subject areas. It is worth noting that only a limited number of models in the literature have employed a matrix for data representation in this manner ([75], [76], [77], [78], [74], [79], [80], [81]).

Tableau :2.1 Classification of citation recommendation models Matrix based data representation [4]

		Data factors/Information used							Data representations			Methodologies adopted							Recommendation types			Problem faced		Personalization	
	Models	Paper contents	Tags/keywords	User/author profile	Venue information	Citation network	Social network	Ratings	Matrix-based	Graph-based	Hybrid	Embeddings	RBM	MLP	GAN	CNN	RNN	DRL	Local	Global	Tags/Labels	Sparsity	Cold Start	Non-Personalized	Personalized
1	MAAE[74]	✓	✓	✓	–	✓	–	✓	✓	–	–	–	–	–	✓	–	–	–	–	✓	✓	✓	–	–	✓
2	POLAR[78]	✓	–	✓	–	–	–	–	✓	–	–	–	–	–	–	✓	–	✓	–	✓	–	–	–	–	✓
3	ML-DTR[75]	✓	✓	✓	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	✓	–	✓	✓	–	✓
4	Paper2veC[81]	–	–	–	–	✓	–	–	✓	–	–	✓	–	–	–	–	–	–	–	✓	–	–	–	✓	–
5	VOPRec [79]	–	–	✓	–	✓	–	–	✓	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
6	TMR-PCR[82]	–	–	✓	–	✓	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	✓	–	–	–	–	✓
7	BNR[77]	✓		✓		✓			✓	✓		✓								✓					✓
8	HRM[80]	✓	✓	✓	✓		✓		✓					✓						✓		✓	✓		✓

- **Graph-based:** Utilizing k-partite graphs to investigate meaningful connections among nodes is a common approach. One such method, called Bibliographic Network Representation (BNR) as proposed by [77] , incorporates both the graph structure and content (such as authors, papers, and venues) to generate lower-dimensional representations of objects. The BNR model computes citation suggestions by evaluating the similarity between the representations of papers and authors. According to Table 2.2, specifically in the column labeled "Data representations" it is observed that 17 models have employed graphs to represent data.

- **Hybrid:** refers to a combination of different approaches, such as incorporating both graph-based and matrix-based methods or exploring alternative techniques.

Table :2.2 Classification of citation recommendation models Graph based data representation [4]

		Data factors/Information used							Data representations			Methodologies adopted							Recommendation types			Problem faced		Personalization	
	Models	Paper contents	Tags/keywords	User/author profile	Venue information	Citation network	Social network	Ratings	Matrix-based	Graph-based	Hybrid	Embeddings	RBM	MLP	GAN	CNN	RNN	DRL	Local	Global	Tags/Labels	Sparsity	Cold Start	Non-Personalized	Personalized
1	CIRec [83]	✓	✓	–	–	✓	–	–	–	✓	✓	✓	–	–	–	–	–	–	✓	–	–	–	–	–	✓
2	RBM-CS[84]	–	–	✓	–	✓	–	–	–	✓	–	–	✓	–	–	–	–	–	✓	–	–	–	–	–	✓
3	MMRQ[19]	✓	✓	✓	–	✓	–	–	–	✓	–	–	–	–	–	–	–	✓	–	✓	–	–	–	–	✓
4	HGRec[85]	✓	✓	✓	✓	✓	✓	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	✓	–	✓
5	DRDF-CR [86]	–	–	✓	–	✓	–	–	–	✓	–	–	–	✓	–	–	–	–	✓	–	–	–	–	–	✓
6	HRLHG[87]	✓	✓	✓	–	✓	–	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
7	SAR[88]	–	–	✓	–	✓	–	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
8	BERT-GCN[56]	✓	–	✓	–	✓	–	–	–	✓	–	–	–	–	–	✓	–	–	✓	–	–	–	–	–	✓
9	ASL[89]	✓	–	✓	–	✓	–	–	–	✓	✓	–	–	–	–	–	✓	–	✓	–	–	–	–	–	✓
10	WHIN-CSL[90]	✓	–	✓	✓	✓	✓	–	–	✓	–	✓	–	–	–	✓	✓	–	–	✓	–	–	–	–	✓
11	GAN-HBNR[76]	✓	–	✓	–	✓	✓	–	–	✓	–	–	–	–	✓	–	–	–	–	✓	–	✓	–	–	✓
12	PCCR[91]	✓	–	✓	–	✓	–	–	–	✓	–	–	–	–	–	–	✓	–	✓	–	–	–	–	–	✓
13	VOPRec [79]	–	–	✓	–	✓	–	–	✓	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
14	TMR-PCR[82]	–	–	✓	–	✓	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	✓	–	–	–	–	✓
15	BNR[77]	✓		✓		✓			✓	✓		✓								✓					✓
16	NREP[92]	–	–	✓	–	✓	–	–	–	✓	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
17	HIPRec[93]	–	✓	✓	✓	✓	–	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	✓	–	✓

The growing number of models ([94]; [90]; [89]; [95]; [96]; [97]; [98]; [92]; [99]; [100]) utilizing this hybrid representation signifies a notable trend emerging within the field.

Table :2.3 Classification of citation recommendation models Hybrid data representation [4]

		Data factors/Information used							Data representations			Methodologies adopted							Recommendation types			Problem faced		Personalization	
	Models	Paper contents	Tags/keywords	User/author profile	Venue information	Citation network	Social network	Ratings	Matrix-based	Graph-based	Hybrid	Embeddings	RBM	MLP	GAN	CNN	RNN	DRL	Local	Global	Tags/Labels	Sparsity	Cold Start	Non-Personalized	Personalized
1	CIRec [83]	✓	✓	–	–	✓	–	–	–	✓	✓	✓	–	–	–	–	–	–	✓	–	–	–	–	–	✓
2	RI-PR[94]	✓	✓	✓	–	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
3	DocCit2Vec[101]	✓	–	–	–	✓	–	–	–	–	✓	✓	–	–	–	–	–	–	✓	–	–	–	–	✓	–
4	NCN[95]	✓	–	–	–	✓	✓	–	–	–	✓	–	–	–	–	✓	✓	–	✓	–	–	–	–	–	✓
5	ASL[89]	✓	–	✓	–	✓	–	–	–	✓	✓	–	–	–	–	–	✓	–	✓	–	–	–	–	–	✓
6	NPM[96]	✓	✓	✓	–	–	–	–	–	–	✓	–	–	✓	–	–	–	–	–	✓	–	–	–	–	✓
7	p-CNN [99]	✓	–	✓	–	–	✓	–	–	–	✓	–	–	–	–	✓	–	–	✓	–	–	–	–	–	✓
8	VCGAN[100]	✓	–	✓	–	–	✓	–	–	–	✓	–	–	–	✓	–	–	–	–	✓	–	–	–	–	✓
9	CPR[102]	✓	–	✓	–	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
10	CPA-CE[97]	✓	–	–	–	✓	–	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
11	AED[98]	✓	–	–	–	✓	✓	–	–	–	✓	–	–	–	–	✓	✓	–	✓	–	–	–	–	–	✓
12	NREP[92]	–	–	✓	–	✓	–	–	–	✓	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓

2.2.2 Methodologies and models

In recent years, Artificial Neural Networks (ANN) have shown promising results in the citation recommendation domain [103]. Similar to Restricted Boltzmann Machine (RBMs) [104], embedding methods [105][106], Convolutional Neural Network (CNN) [107], and Recurrent Neural Network (RNN) [108], ANNs are utilized to learn complex mappings within the network. The various types of ANN approaches employed by citation recommendation systems are discussed in the "Methodologies and Models" column of Table (2.1; 2.2; 2.3).
Typically, an ANN consists of multiple layers, including an input layer, one or more hidden layers, and an output layer, with each layer comprising a set of neurons [107]. This part provides an overview of deep learning architectures that are closely relevant to the explored models, as illustrated in Figure 1.5.

- **Restricted Boltzmann Machine (RBM):** The individual components of the network are interconnected in a symmetrical manner, allowing them to make stochastic decisions regarding whether they should be activated or deactivated. Furthermore, these neurons can establish connections within the same layer. However, the learning algorithm of this type of network is relatively simple but can become slow when dealing with multiple layers. On the other hand, the Restricted Boltzmann Machine (RBM) [104] is a two-layer neural network consisting of only an input layer and a hidden layer.
  RBMs impose constraints on the connections between layers, making their implementation easier compared to Boltzmann Machines. These restrictions on the intra-layer connectivity in RBMs result in significantly improved learning efficiency when compared to Boltzmann Machines [109]. A basic depiction of an RBM network can be seen in Figure 2.2.

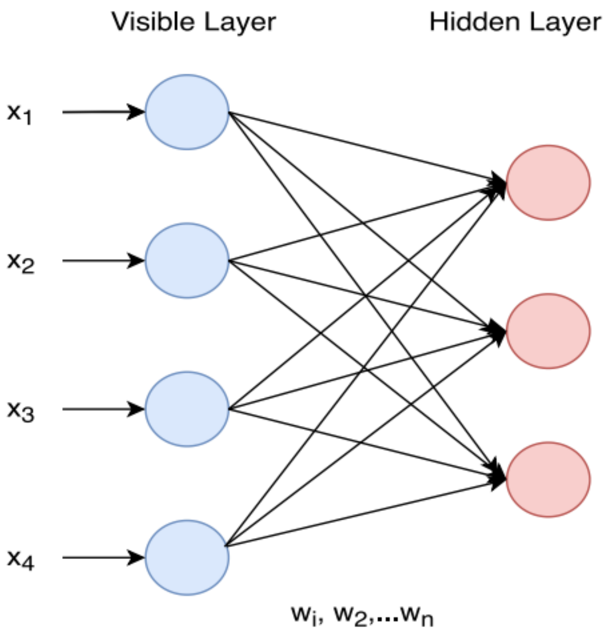


Figure :2.2 Architecture of Restricted Boltzmann machine [4].

The hidden layer H_i calculates the score for each node by multiplying the four inputs from the visible layer V_l with their corresponding weights. The products are then summed and combined with a bias term. Finally, an activation function is applied to pass the results to the output, expressed as $h = g(W \cdot v + b)$. The probability of observing a particular state of v and h is computed as follows:

$$p(v, h) = \frac{1}{Z} e^{-(a^T v + b^T h + h^T wv)}$$

The biases a and b represent the offsets or adjustments for the visible and hidden layers, respectively. The symbol Z represents the normalization function, and W represents the weight parameter connecting the visible layer V_l and the hidden layer H_l. The optimization process aims to optimize the following objective:

$$\arg\max \sum_{v \in V} \log P(v, h)$$

Unlike sparse Autoencoder and other networks, Restricted Boltzmann Machine (RBM) stands out for its fast performance due to its simple forward encoding process. [84] introduced a variant of RBM called RBM-CS, which consists of two layers. RBM-CS aims to learn the topic distribution of articles' contents and citation relationships. By incorporating a hidden topic layer, RBM-CS captures the topic distribution of papers and utilizes these relationships to generate a ranked list of top$-n$ papers.

- **Multi Layer Perceptron (MLP):** Considered as the most straightforward type of neural network, the feed-forward method consists of one or more hidden layers and

one output layer [41]. These layers incorporate activation functions that adjust the weights during training. When the hidden layer(s) receive input representations, they perform a mapping of inputs to outputs by applying non-linearity [107] as shown in Figure2.3. The output of each neuron is calculated using $h = g(W \cdot x_i + b)$, where W represents the weights between corresponding neurons of the input and hidden layers. The variables h and b denote the hidden layer neuron and the bias vector, respectively, while g refers to the activation function used, such as relu, tanh, sigmoid, and others.
Finally, the network's output is predicted by propagating the weighted sums from the hidden layer(s) to the output layer, which applies non-linearity to the transmitted value to generate the final prediction as follows:
$\hat{y} = g(W \cdot h + b)$

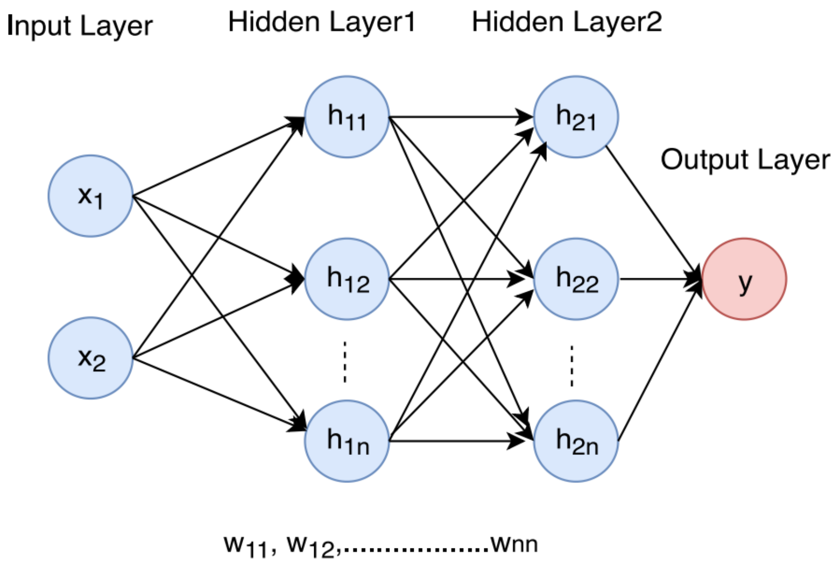


Figure :2.3 An example of MLP network [4].

In this context, the symbol $\hat{y}$ represents the output, and g is used to denote the non-linear activation function. These activation functions play a crucial role in propagating information to the subsequent layers of the network.

In the field of citation recommendation, Multi-Layer Perceptrons (MLPs) have demonstrated promising outcomes in capturing the semantic representations of research papers and generating high-quality results. One notable example is the NNRank model introduced by [110] which utilizes a three-layer feed-forward neural architecture to encode research papers into a lower-dimensional space based on their content. The model then identifies the K nearest neighbors for a given query paper and applies another discriminative model to re-rank the papers, distinguishing between observed and unseen papers. During training, the model is optimized to predict a high cosine similarity for the pair (d_q , d^+) and a low cosine similarity for the pair (d_q , d^-) by employing the per-instance triplet loss as defined in the context:

Loss = $\max(\alpha + s(d_q; d^-) - s(d_q; d^+), 0)$ The term $s(d_i, d_j)$ represents the cosine similarity between document embeddings cos $-\text{sim}(e_{d_i}, e_{d_j)}$

Where e_{d_i}, e_{d_j} are the embeddings of documents d_i, d_j, respectively. The hyper-parameter α is used in the model as a tuning parameter.

The model demonstrates superior performance compared to other models, particularly in scenarios where metadata such as author information, venue information, and key-phrases are not accessible. Additionally, [96] leverage the semantic representations of references and their contexts to generate recommendations. They train a multi-layer neural network to learn the probability of references given the contexts, enabling the computation of a score for each paper d_j using the following formula:

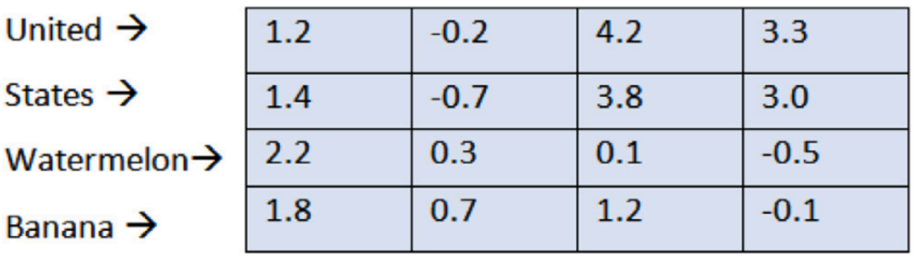
$$p(d_i|q) = \sum_{j=1}^{q} p(d_i|w_j)p(w_j|q)$$

In this context, $p(w_j|q$ represents the probability of an article given a query q. In contrast, [86] proposed a co-citation model called DRDF-CR, which combines text and citation graphs to learn multi-vector representations. Each vector captures different discourse facets of a paper, enabling context-aware recommendations. To classify sections within an article, the model utilizes the fastText library [111]. It computes vector representations for each section by averaging the vectors of all words present in the section. Facet representations (such as objective, method, and result) are computed by averaging the corresponding facets for each paper. Additionally, the citation graph is enhanced by incorporating a discourse facet for each citation edge. Finally, the model updates facet and section vectors using the LINE [112].

Similarly, [80] presented a model called Hybrid Reranking Model (HRM) that utilizes a two-layer feedforward neural network to generate predictions. The model takes article features as input and produces a prediction score for each article in the output layer. The HRM model sends weekly article recommendations to registered users. It leverages both article content and user behavior to re-rank the suggested articles provided by ScienceDirect. Specifically, it incorporates various content-based measures that are derived from different aspects such as space, tags, and author similarity. Additionally, the model employs joint matrix factorization to link the articles that the user has browsed with the user's click behavior. A pairwise learning approach is used to further refine the final list of recommendations.

- **Embedding methods:** Discrete variables are transformed into continuous vector representations through a process known as encoding. These learned embeddings offer several advantages over traditional encoding methods like one-hot encoding. They find diverse applications, including identifying nearest neighbors to enable recommendations based on user or group interests, serving as input for other neural networks, and facilitating the visualization of concepts and relationships across different categories. Embedding techniques can be broadly categorized into word embedding [106] and graph embedding methodsg [113].
  In natural language processing (NLP), word embedding is commonly used to represent phrases and employ feature learning methods that encode words into vectors. These methods have proven effective in capturing both syntactic and semantic information in words. Among the popular models in this domain are BERT [57], word2vec [106], and doc2vec [114], which are utilized to embed users, items, documents, and locations [115] into a latent space for various applications.

  For example, when the word 'United' is inputted into an embedding model, it is highly probable that the model will generate a representation similar to related words like 'States', rather than unrelated words like 'Banana' or 'Watermelon'. This is achieved by analyzing the contexts in which these words appear and generating their vector representations, as shown in Figure 2.4. As a result, the 4-dimensional representation uncovers words that occur in similar contexts and exhibit similar representations in the embedding space.



United →	1.2	-0.2	4.2	3.3
States →	1.4	-0.7	3.8	3.0
Watermelon→	2.2	0.3	0.1	-0.5
Banana →	1.8	0.7	1.2	-0.1

Figure :2.4 A 4-dimensional vector representation of words[4].

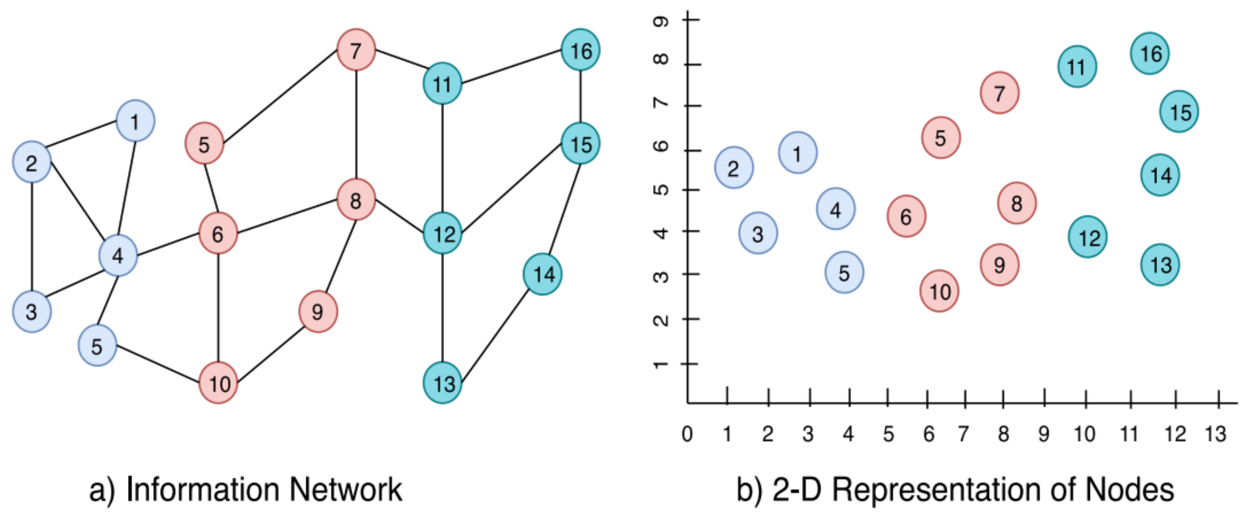


Figure :2.5 An Illustration of Network representation learning [4]

In contrast, graph embedding techniques transform nodes into a lower-dimensional space by leveraging the graph's topological structure and other relevant information such as node contents and attributes. These representations enable the capture of node relationships by measuring distances in the embedded space, as illustrated in Figure 2.5. Notably, closely connected nodes are positioned closer to each other in the latent space. For example, node 7 exhibits first-order proximity with node 11, indicating their close proximity in the embedded space. Similarly, nodes 12 and 14, while not directly connected, share common neighbors and maintain second-order proximity, leading to their proximity in the embedding. Various learning methods have been introduced in the literature, including LINEE [112], DeepWalk [116], HINE [117], and Node2vec [105]. It is important to note that these approaches can be classified as homogeneous or heterogeneous graph embeddings. Homogeneous network embedding models such as LINE and DeepWalk focus on a single type of nodes and relations, making them unsuitable for handling heterogeneity. On the other hand, heterogeneous network embedding models like HINE and Node2vec explore multiple types of nodes and relations when generating vector representations. For a more comprehensive discussion on novel network embedding approaches, refer to [113].

In the context of citation recommendations, the aforementioned embedding techniques are utilized either to measure similarity between nodes or as input for other methods, typically supervised learning approaches, that generate the recommendations. In line with this, [83] introduced a weighted heterogeneous network embedding model known as Tendency Random Walk (CIRec). This model fo-

cuses on capturing the relationships between papers and their references through a weighted version of random walk. Specifically, the citation tendency is computed by considering the connections between papers and their citations based on shared features like authors and terms. The skip-gram method is then employed to obtain vector representations of the research papers, optimizing the following objective function:
$L = \sum_{v_i \in V} \log Pr(N_s(v_i)|v_j)$

Where V represents all the nodes in the graph, and N_s contains the neighborhood information within the same sliding window as v_i. The model utilizes cosine similarity to measure the similarity between papers in the embedding space, which is then used to generate recommendations. On the other hand, [83] introduced WHIN-CSL, a model that leverages semantic relations and attribute values to generate reference recommendations. It constructs a weighted heterogeneous information network (HIN) consisting of nodes representing papers and authors, and considers four different types of relations: writing, semantic linking, citing, and co-authorship. After Node2vec [105] is applied to obtain vector representations of the nodes, and similarity between vertices is computed using a linear combination method, such as:

$$P_r(cp_c|tp_t) = w_1\mu_1(cp_c|tp_t) + w_2\mu_2(cp_c|tp_t) + (1 - w_1 - w_2)\mu_3(cp_c|tp_t)$$

Where $P_r(cp_c|tp_t)$ represents the conditional probability between the target paper tp_t and the candidate paper $\mu_1(cp_c|tp_t)$, $\mu_2(cp_c|tp_t)$ and $\mu_3(cp_c|tp_t)$ represent the similarity measures for abstract-abstract, paper-paper vertex, and paper-author vertex, respectively. The weights $w1$ and $w2$, with $w1 + w2 < 1$, are used to adjust the contribution of each relation. By considering these factors, the model generates top$-n$ relevant reference recommendations.

- **Convolutional Neural Network (CNN):** these models are similar to MLP (Multi-Layer Perceptron) and offer advantages such as reduced pre-processing requirements and efficient performance on large-scale networks with low memory usage during training. Their architecture includes an input layer, a convolutional layer, a sub-sampling layer for pooling, a fully connected layer, and an output layer, as shown in Figure 2.6. This architecture produces a feature map f^k, which is calculated using the following process:
$f_i^{(l)} = \tanh((W^l * f_i^{(l-1)}) + b^l)$

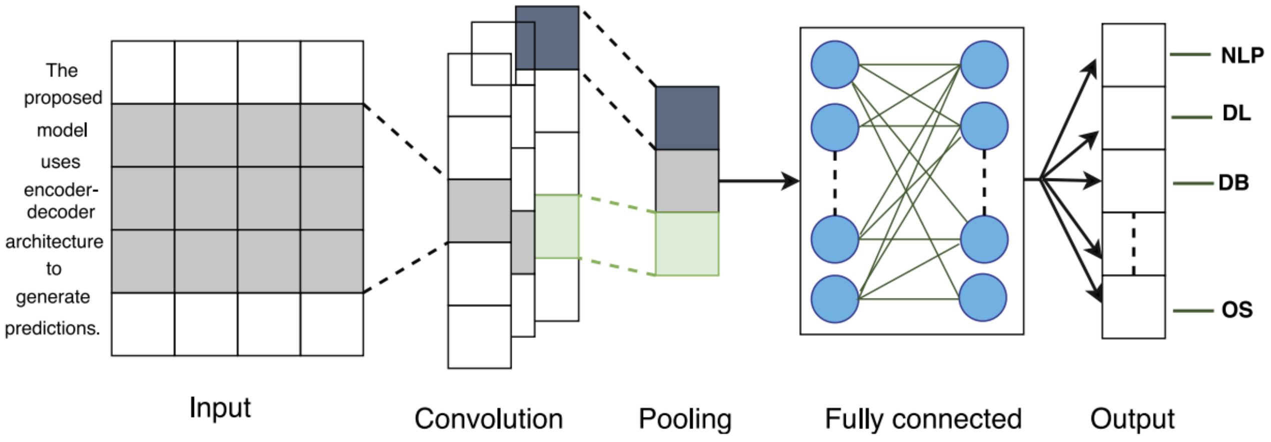

The proposed model uses encoder-decoder architecture to generate predictions.
Input
Convolution
Pooling
Fully connected
Output
NLP
DL
DB
OS

Figure :2.6 Convolutional neural network [4]
:

In the given context, the activation function used is the hyperbolic tangent function *tanh*. The symbol W represents the parameters specific to the corresponding layer, and $f_i^{(l-1)}$ represents the segment of the layer for convolution at a particular location i. Furthermore, the max-pooling layer performs downsampling of text or images by using a sliding window, as described below:
$f_i^{(l)} = \max\{f_t^{(l-1)}, f_t^{(l-1)}(i+1)\}$
By adopting this methodology, the computational burden is significantly reduced as each layer becomes smaller. This iterative process is performed on multiple layers, allowing each layer to learn valuable features. Once these features are learned, the CNN network functions as a classifier. The penultimate layer calculates the probabilities associated with each class for the classification of items/papers. Ultimately, the final layer of the network serves as a classification layer, generating the classification output.

CNN models have been performing really well in the areas of NLP and recommender systems in recent years. They are more effective than multi-layer perceptrons because they can reduce the number of neurons through pooling operations. The shared-weights architecture also helps to reduce the overall number of parameters, making them faster and less complex. Because of these advantages, CNNs have been showing promising results in capturing the meaning and context of citations for citation recommendation tasks. One specific model, called the personalized convolutional neural network (p-CNN), was introduced by [99]. This model learns unique features about an author and uses them to calculate how relevant a cited article is to the context. It uses a discriminative training strategy to minimize a specific objective function as following:

$$Loss(\theta) = \max\{0, 1 + s(cc, D_j^-) - s(cc, D^+)\}$$

In he given equation $s(cc, D)$ represents the similarity score between context c and document D, which is obtained using multi-layer perceptions in the fully connected layer. POLAR [78] proposed an attention-based CNN model to generate citation recommendations. The model utilizes an attention matrix to calculate the importance of a term based on local and global weights for text similarity. The attention matrix and matching matrix are then fed into the CNN network to identify various levels of textual similarities.

In addition [56] put forward a context-aware model that employs a combination of the BERT [57] model and a modified version of the Graph Convolution Network (GCN) [58]. In order to produce textual embedding, the pre-trained BERT functions as a context encoder, while the GCN model is used to create graph embedding. This approach enables the model to utilize both textual content and citation network to generate context-aware citation recommendations.

- **Recurrent Neural Network (RNN):** is similar to Convolutional Neural Networks (CNNs) in their structure, but differ in their ability to retain previously learned information and apply it to future inputs. RNNs utilize a directed graph with sequential connections between nodes, allowing for the persistence of information over time. In contrast, traditional neural networks lack this ability and are thus limited in their decision-making and event prediction capabilities. RNNs address this limitation by incorporating memory into their architecture, as illustrated in Figure 1.7 [109]. The diagram depicts an input x_t being passed through a layer of feed-forward neural network A, with the resulting output value being represented by h_t. Loops are used to maintain information and pass it from one step to the next in the network, with each participating network passing a message to its succeeding network. Overall, RNNs consist of multiple copies of the same graph, with each copy contributing to the retention and utilization of previous information.
  Recurrent Neural Networks (RNNs) are highly efficient in handling sequential data

and can remember and identify patterns across different time periods. RNNs comprise an input, an output, and hidden units that are essential in storing information. Unlike Multilayer Perceptrons (MLPs), RNNs incorporate a directional loop that stores previous information and applies it to the output. Moreover, RNNs utilize a dynamic user uut and paper vpt attributes that are learned from Long-Short Term Memory (LSTM), and corresponding static attributes uu and vp obtained through Matrix Factorization. Based on this, predictions for a paper can be computed as follows:

$\hat{\mathrm{r}}_{up|t} = f(u_{ut}, v_{pt}, u_u, v_p)$

In order to produce precise predictions, the process aims to reduce errors by aligning the anticipated score with the actual score. Several types of RNNs exist and are chosen based on the specific requirements of the application. However, the most frequently implemented methods are LSTM (Long Short-Term Memory) [108] and GRU (Gated Recurrent Unit) [108], as cited by Goodfellow et al. in 2016. LSTM is a type of time recurrent neural network that is well-suited for forecasting significant events that occur over longer intervals.

When dealing with long sequences, traditional RNNs can experience the short-term memory problem. This means that if we want to process a lengthy textual paragraph to generate predictions, traditional RNNs may miss important information. Another issue that RNNs face is the vanishing gradient problem, where gradients shrink as they backpropagate through time, leading to small values that do not contribute much to learning. As a result, layers that encounter this problem stop the learning process, and RNNs experience the short-term memory problem. To address this issue, LSTM integrates both short and long-term memories through a subtle gate control. The main difference between these two approaches is that LSTM adds a "cell" to determine whether the information it possesses is useful or not. A cell memory contains three cells, including an input, a forget, and an output gate, as depicted in Figure 2.7. Mathematically, these cells are represented in Equation [118].

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$
$$f_t = \sigma(W_f x_t + U_o h_{t-1} + b_f)$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$
$$\tilde{c}_t = \tanh(\sigma(W_c x_t + U_c h_{t-1} + b_c)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}$$
$$h_t = o_t \odot \tanh(c_t)$$

where σ and $\odot$ denotes the sigmoid function and the element-wise multiplication, respectively.

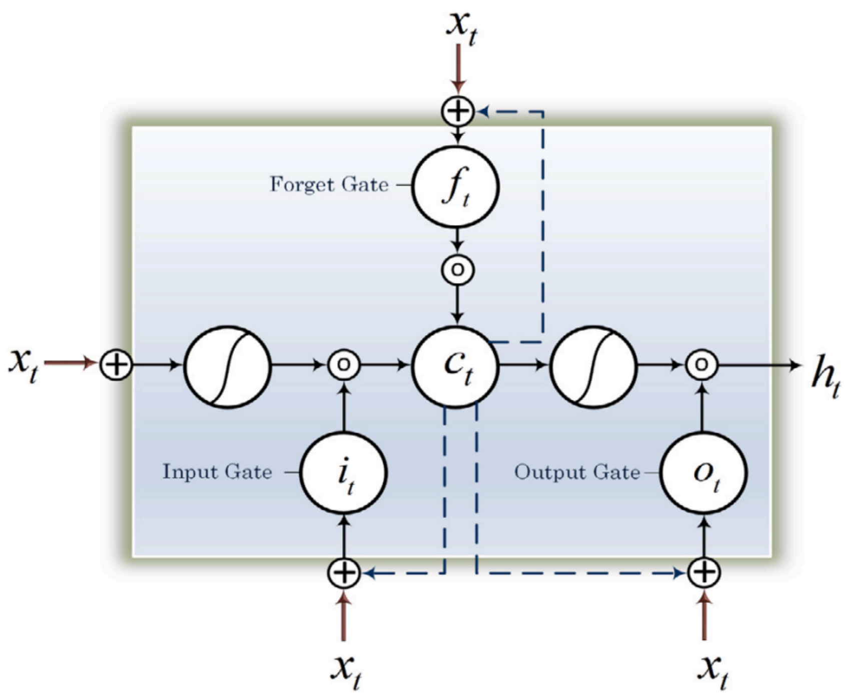


Figure :2.7 Long short term memory block[4]
:

In the context of LSTM (Long Short-Term Memory) networks, various symbols are used to represent specific components. The symbol t is used to denote a specific time step. The input gate is denoted by i_t, while the forget gate is represented by f_t. The output is represented by $\odot_t$, the cell state by c_t, and the hidden state by h_t. Additionally, the parameters of the LSTM are represented by W, U, and b. The cell state retains relevant information for transfer to the next level, while the forget gate ensures that only useful information is retained during training.
On the other hand, GRUs use the hidden state to transmit information instead of relying on the cell state. Moreover, GRUs incorporate two novel gates - the reset gate and the update gate. The update gate functions akin to the forget gate in an LSTM network, whereas the reset gate determines which prior information to disregard. As a result of utilizing fewer tensor operations compared to LSTMs, GRUs exhibit greater speed.
At the end, CACR [91] utilized LSTM to acquire a distributed representation of articles context and manuscripts. The model then selects the most relevant articles based on the scores of relevance between the two. Similarly, MLDTR [75], a CF-based model, produces paper recommendations by generating the latent representation of text sequence using GRUs. Another RNN-based model [119] generates personalized recommendations by discovering the latent semantic features of scientific papers and considering users'feedback. The study investigates the impact of using word2vec and LSTM to extract the semantic representation of the content of

papers.

- **Generative Adversarial Network (GAN):** In the field of artificial intelligence, Generative Adversarial Networks (GANs) are composed of two neural networks: a Discriminator and a Generator, as described by [120]. The Generator produces fake data such as images or text, and attempts to deceive the Discriminator, which in turn tries to differentiate between real and fake samples. During the training process, these two models compete against each other, repeating their respective steps multiple times to achieve their goals. GANs have been developed as a min-max game, with the aim of minimizing the reward of the Discriminator while maximizing the probability of the Discriminator making an error. The Discriminator attempts to minimize its loss by estimating the probability that the sample it received is from the probe set and not the generated one, as shown in Figure 2.8.

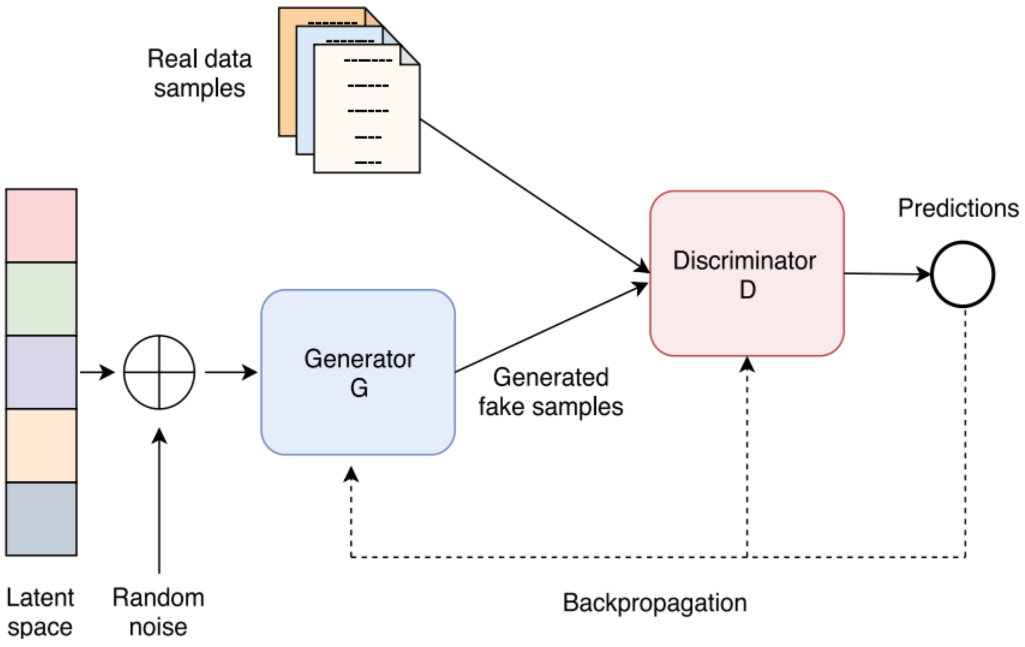


Figure :2.8 Framework of generative adversarial network[4]
:

In a professional tone, assume that the user's relevance distribution is denoted by $(P_{true} = d|q_n, r)$ represents the users relevance distribution, and the Generator $(P_\theta = d|q_n, r)$ aims to estimate the actual relevance distribution. On the other hand, the Discriminator $(f_\phi = q|d)$ distinguishes between relevant and non-relevant items/papers. The primary objective function is expressed as follows:
$J^{G^*,D^*} = \min_\theta \max_\phi \sum_{n=1}^{N} \left(E_{d\sim p_{true}(d|q_n,r)}[\log D(d|q_n)] + E_{z\sim p_\theta(d|q_n,r)}[1 - \log D((d|q_n))]\right)$
Where $D(d|q_n) = \sigma(f_\phi(q_n|d))$ the symbol σ represents the sigmoid function,

while θ is the parameter used for generative retrieval and ϕ is utilized for discriminative retrieval. To learn both parameters, gradient descent is employed.
In order to achieve this objective, [100] developed VCGAN, a system that utilizes Generative Adversarial Network to produce personalized citation recommendations. The system leverages both the network and content associated with nodes to create content-based graph representations. It also examines the co-authorship network. The resulting representations are combined to form the node feature vectors, which are then utilized to generate citation recommendations for a given manuscript query:
$s(q, p_j) = \frac{q \cdot p_j}{\|q\| \cdot \| \|p_j\| \|}$
The variables q and p_j denote the query and candidate papers, respectively. Additionally, there exists a comparable approach for learning in heterogeneous bibliographic networks, known as GAN-HBNR [76]. The variables q and p_j denote the query and candidate papers, respectively. In this model, personalized recommendations are generated by utilizing the network structure, authors, papers, and query manuscript to develop representations. Doc2vec embedding approach is used to acquire the content representation of each vertex [114]. To incorporate the network structure and vertex content, the model utilizes a Denoising Autoencoder (DAE) network. The DAE network includes a feed-forward generator network, $G(z)$, which takes a vector $z \in \mathbb{R}^{h_g}$ as input and produces a generated vector. Additionally, the discriminator network, $D(x)$, takes vectors $z \in \mathbb{R}^{m+n}$ and produces an energy estimate $E \in \mathbb{R}$. Once the network representations are learned, the model uses a similarity computation method between vectors to determine top-ranked papers:
$\vec{V}_q = \vec{V}_{PR}\vec{V}_{qt}^T + \vec{V}_{AR}\vec{V}_{qa}^T$
The vectors $\vec{V}_{PR}$, $\vec{V}_{qt}$ and $vecV_{qa}$ represent the training papers, manuscript text, and manuscript author, respectively. Additionally, [74] proposed a technique called MAAE, which combines generative adversarial networks with auto-encoders to generate recommendations for citations and labels.

- **Deep Reinforcement Learning (DRL):** in the field of Machine Learning (ML), Reinforcement Learning (RL) is a technique where an agent takes action within an environment at a specific point in time (t). The agent receives two responses from the environment: a reward that quantifies the action taken by the agent at that time step, and a state where the environment changes in response to the agent's action. This process is repeated until a terminal state is reached. Deep Reinforcement Learning (DRL) integrates RL with deep learning models to tackle more intricate challenges in Natural Language Processing (NLP) and Information Retrieval (IR). Citation recommendation models utilize Deep Reinforcement Learning (DRL) to assign scores to potential candidate citations for a target user. DRL incorporates gradient descent operations to estimate values that take into account the changing preferences of a researcher.

In order to generate citation recommendations, TMR-PCR [82] combines researchers, venues, and papers in a three-layered graph using a mutually reinforcing approach. To personalize the recommendations, the model incorporates both the researcher's information and the query's textual information into a random walk process. By utilizing a three-layered interactive clustering method, the model addresses the computational complexity challenges associated with random walk methods when applied to large graphs.

Likewise, MMRQ [19] utilizes a multi-layered graph containing entities such as authors, papers, and keywords. The model employs multi-layer reinforcement rules within the graph to generate citation recommendations that are tailored to the specific query.

- **hybrid:** to enhance the effectiveness of citation recommendations, various models have integrated multiple deep learning networks, such as CNN and RNN, CNN and AE, GAN and AE, among others. During our survey, we found that only a few models utilized hybrid deep learning approaches to generate citation recommendations, as shown in Table 2.4, in the column labeled "Methodologies and models".

Table :2.4 Classification of citation recommendation models[4].

		Data factors/Information used							Data representations			Methodologies adopted							Recommendation types			Problem faced		Personalization	
	Models	Paper contents	Tags/keywords	User/author profile	Venue information	Citation network	Social network	Ratings	Matrix-based	Graph-based	Hybrid	Embeddings	RBM	MLP	GAN	CNN	RNN	DRL	Local	Global	Tags/Labels	Sparsity	Cold Start	Non-Personalized	Personalized
1	CIRec [83]	✓	✓	–	–	✓	–	–	–	✓	✓	✓	–	–	–	–	–	–	✓	–	–	–	–	–	✓
2	RBM-CS[84]	–	–	✓	–	✓	–	–	–	✓	–	–	✓	–	–	–	–	–	✓	–	–	–	–	–	✓
3	MMRQ[19]	✓	✓	✓	–	✓	–	–	–	✓	–	–	–	–	–	–	–	✓	–	✓	–	–	–	–	✓
4	MAAE[74]	✓	✓	✓	–	✓	–	✓	✓	–	–	–	–	–	✓	–	–	–	–	✓	✓	✓	–	–	✓
5	HGRec[85]	✓	✓	✓	✓	✓	✓	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	✓	–	✓
6	POLAR[78]	✓	–	✓	–	–	–	–	✓	–	–	–	–	–	–	✓	–	✓	–	✓	–	–	–	–	✓
7	RI-PR[94]	✓	✓	✓	–	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
8	DocCit2Vec[101]	✓	–	–	–	✓	–	–	–	–	✓	✓	–	–	–	–	–	–	✓	–	–	–	–	✓	–
9	DRDF-CR [86]	–	–	✓	–	✓	–	–	–	✓	–	–	–	✓	–	–	–	–	✓	–	–	–	–	–	✓
10	HRLHG[87]	✓	✓	✓	–	✓	–	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
11	ML-DTR[75]	✓	✓	✓	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	✓	–	✓	✓	–	✓
12	SAR[88]	–	–	✓	–	✓	–	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
13	BERT-GCN[56]	✓	–	✓	–	✓	–	–	–	✓	–	–	–	–	–	✓	–	–	✓	–	–	–	–	–	✓
14	NCN[95]	✓	–	–	–	✓	✓	–	–	–	✓	–	–	–	–	✓	✓	–	✓	–	–	–	–	–	✓
15	5 PPR-DL[119]	✓	–	✓	–	–	–	–	–	–	–	–	–	–	–	–	✓	–	–	✓	–	–	–	–	✓
16	ASL[89]	✓	–	✓	–	✓	–	–	–	✓	✓	–	–	–	–	–	✓	–	✓	–	–	–	–	–	✓
17	CITEWERTs(a)[121]	✓	–	–	✓	–	–	–	–	–	–	–	–	–	–	✓	✓	–	✓	–	–	–	–	✓	–
18	CITEWERTs(b)[122]	✓	–	–	✓	–	–	–	–	–	–	–	–	–	–	✓	✓	–	✓	–	–	–	–	✓	–
19	WHIN-CSL[90]	✓	–	✓	✓	✓	✓	–	–	✓	–	✓	–	–	–	✓	✓	–	–	✓	–	–	–	–	✓
20	NNRank[110]	✓	–	✓	✓	–	✓	–	–	–	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
21	GAN-HBNR[76]	✓	–	✓	–	✓	✓	–	–	✓	–	–	–	–	✓	–	–	–	–	✓	–	✓	–	–	✓
22	PCCR[91]	✓	–	✓	–	✓	–	–	–	✓	–	–	–	–	–	–	✓	–	✓	–	–	–	–	–	✓
23	NPM[96]	✓	✓	✓	–	–	–	–	–	–	✓	–	–	✓	–	–	–	–	–	✓	–	–	–	–	✓
24	Paper2veC[81]	–	–	–	–	✓	–	–	✓	–	–	✓	–	–	–	–	–	–	–	✓	–	–	–	✓	–
25	VOPRec [79]	–	–	✓	–	✓	–	–	✓	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
26	p-CNN [99]	✓	–	✓	–	–	✓	–	–	–	✓	–	–	–	–	✓	–	–	✓	–	–	–	–	–	✓
27	VCGAN[100]	✓	–	✓	–	–	✓	–	–	–	✓	–	–	–	✓	–	–	–	–	✓	–	–	–	–	✓
28	TMR-PCR[82]	–	–	✓	–	✓	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	✓	–	–	–	–	✓
29	CPR[102]	✓	–	✓	–	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
30	BNR[77]	✓		✓		✓			✓	✓		✓								✓					✓
31	HRM[80]	✓	✓	✓	✓		✓		✓					✓						✓		✓	✓		✓
32	CPA-CE[97]	✓	–	–	–	✓	–	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
33	AED[98]	✓	–	–	–	✓	✓	–	–	–	✓	–	–	–	–	✓	✓	–	✓	–	–	–	–	–	✓
34	NREP[92]	–	–	✓	–	✓	–	–	–	✓	✓	✓	–	–	–	–	–	–	–	✓	–	–	–	–	✓
35	HIPRec[93]	–	✓	✓	✓	✓	–	–	–	✓	–	✓	–	–	–	–	–	–	–	✓	–	–	✓	–	✓

In a similar vein, [95] employ an encoder-decoder (ED) approach to understand the semantic connections between citation contexts and relevant cited papers by considering author relationships. The encoder component transforms a given citation context into a vector representation using a modified version of CNN known as the time delay neural network. The GRU decoder component utilizes an attention mechanism and author networks to decode the encoded representation produced by the encoder. Refer to Figure2.9 for a visual representation of this process.

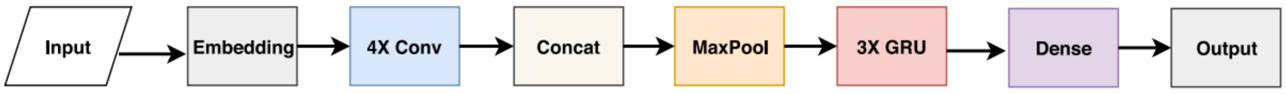


Figure :2.9 Architecture of CNRN model[4]

Similarly, the attention-based encoder-decoder (AED) model developed by [98] generates citation recommendations that are sensitive to the context. This model employs a time delay neural network (TDNN) as the encoder network and a recurrent neural network (RNN) as the decoder network. To capture the semantic relationships between citation contexts and research papers, the AED model incorporates an attention mechanism that takes into account the author and venue information. By leveraging these components, the model is able to provide context-aware citation recommendations.

2.2.3 Personalization

This section pertains to the various types of recommendations provided to a specific user, including personalized, non- personalized, and group-based recommendations, which are detailed in Table 2.4. Personalized models use a user's profile information and browsing history to generate recommendations. Conversely, non-personalized models create recommendations based on popular or highly-rated items, resulting in all users receiving similar recommendations, regardless of individual research interests. Upon reviewing several studies, it was found that many models have produced personalized recommendations.

2.3 Datasets and metrics for evaluation

In order to assess the effectiveness of different models, researchers have utilized diverse sets of data and performance measures. With the emergence of new recommendation algorithms, it is crucial to identify suitable datasets and metrics that can be employed for evaluating experimental outcomes. Presented below is a summary of the datasets and metrics employed by the models under scrutiny.

2.3.1 Datasets

This section presents an analysis of the most popular datasets, as shown in Tables **??**. The findings indicate that the ACL anthology dataset is the most frequently used dataset, with 13 mentions. This is due to the dataset's ability to provide access to more comprehensive information about papers, authors, venues, citation relations, and content. Additionally, a considerable number of articles utilize self-collected datasets from various online repositories and libraries, such as Springer, ScienceDirect, ACM, and IEEE. Finally, it is worth noting that two datasets, CiteUlike and RARD II, offer user ratings information, making them suitable for adoption in CF-based models [75].

2.3.2 Evaluation metrics

In this section, we present a comprehensive list of the most commonly used evaluation metrics in the literature, which include Precision, Recall, F-Measure, Root Mean Square Error (RMSE), Coverage, normalized Discounted Cumulative Gain (nDCG), and Accuracy. Table2.5 shows that the majority of models use multiple metrics for evaluation. Specifically, six models perform a thorough evaluation using four metrics, eleven models use three metrics, while eight models use two metrics. On the other hand, the remaining nine models use only one metric. It is important to note that using more metrics leads to a more thorough analysis of the model's performance, and the use of multiple metrics enhances our understanding of the model's impact.

Table :2.5 Evaluation metrics.[4]

No.	Models	Precision	Recall	RMSE	Coverage	nDCG	MRR	F-Measure	Accuracy	Novelty
1	CIRec [83]	✓	✓	–	–	✓	–	–	–	–
2	RBM-CS[84]	✓	–	–	–	–	✓	–	–	–
3	MMRQ[19]	–	✓	–	–	✓	–	–	–	–
4	MAAE[74]	–	–	–	–	–	✓	–	–	–
5	HGRec[85]	✓	✓	–	–	–	–	✓	–	–
6	POLAR[78]	–	–	–	–	✓	–	–	–	–
7	RI-PR[94]	✓	✓	–	–	–	–	✓	✓	–
8	DocCit2Vec[101]	✓	✓	–	–	✓	–	–	–	–
9	DRDF-CR [86]	–	–	–	–	✓	–	–	–	–
10	HRLHG[87]	✓	–	–	–	✓	✓	–	–	–
11	ML-DTR[75]	–	✓	–	–	–	–	–	–	–
12	SAR[88]	–	✓	–	–	✓	–	–	–	–
13	BERT-GCN[56]	✓	✓	–	–	–	✓	–	–	–
14	NCN[95]	✓	✓	–	–	✓	✓	–	–	–
15	PPR-DL[119]	–	✓	–	–	✓	–	–	–	–
16	ASL[89]	✓	✓	–	–	✓	–	–	–	–
17	CITEWERTs(a)[121]	✓	✓	–	–	–	–	✓	✓	–
18	CITEWERTs(b)[122]	✓	✓	–	–	–	–	✓	✓	–
19	WHIN-CSL[90]	–	✓	–	–	✓	–	–	–	–
20	NNRank[110]	–	✓	–	–	✓	–	–	–	–
21	GAN-HBNR[76]	✓	✓	–	–	–	✓	–	–	–
22	PCCR[91]	✓	✓	–	–	–	✓	–	–	–
23	NPM[96]	✓	–	–	–	✓	✓	–	–	–
24	Paper2veC[81]	–	–	–	–	–	–	–	–	✓
25	VOPRec [79]	✓	✓	✓	–	✓	–	–	–	–
26	p-CNN [99]	✓	✓	–	–	–	–	–	–	–
27	VCGAN[100]	–	✓	–	–	✓	–	–	–	–
28	TMR-PCR[82]	✓	✓	–	–	–	✓	–	–	–
29	CPR[102]	–	–	–	–	✓	–	–	–	–
30	BNR[77]	✓	–	–	–	–	✓	–	–	–
31	HRM[80]	✓	–	–	–	–	–	–	–	–
32	CPA-CE[97]	–	–	–	–	✓	–	–	–	–
33	AED[98]	✓	✓	–	–	✓	–	–	–	–
34	NREP[92]	✓	✓	–	–	✓	–	–	–	–
35	HIPRec[93]	–	–	–	–	–	–	–	✓	–

2.4 Conclusion

In conclusion, this chapter provided a comprehensive overview of the related work in the field of context-aware citation recommendation using deep learning techniques. The studies and approaches discussed in this chapter shed light on the advancements made in the area of citation recommendation systems and highlight the significance of incorporating contextual information for improved recommendation accuracy and relevance.

The review of literature revealed that traditional citation recommendation methods often suffer from limitations such as the reliance on simple features, lack of consideration for contextual factors, and inability to capture the semantic relationships between papers. To address these challenges, researchers have turned to deep learning techniques, which have shown great potential in capturing complex patterns and representations in citation networks.

Despite the progress made in context-aware citation recommendation using deep learning, there are still several challenges and open research questions that need to be addressed. These include the development of more robust and scalable models, the integration of additional contextual factors, the consideration of temporal dynamics in citation networks, and the exploration of interpretability and explainability in recommendation results.

CHAPTER 3 CONCEPTION

3.1 Introduction

The goal of this chapter is to outline the conception of the proposed citation recommendation system. The subsequent sections delve into the system design and architecture, system components and functionality, and the presentation of a sequence diagram.
The chapter aims to provide a comprehensive understanding of the inner workings of the proposed system. By discussing the design and architecture, the system's components and their respective functionalities will be explored, shedding light on the specific algorithms and techniques employed in each component. Furthermore, the sequence diagram will visually illustrate the flow of information and operations within the system.

3.2 System Design and Architecture

The system design and architecture of the proposed system are crucial in transforming a large dataset of thousands of papers into meaningful results through the stages of preprocessing, retrieval, ranking, and post-ranking. In this section, we provide an overview of the system's design principles and the components involved in each stage as it depicted in Figure3.1.

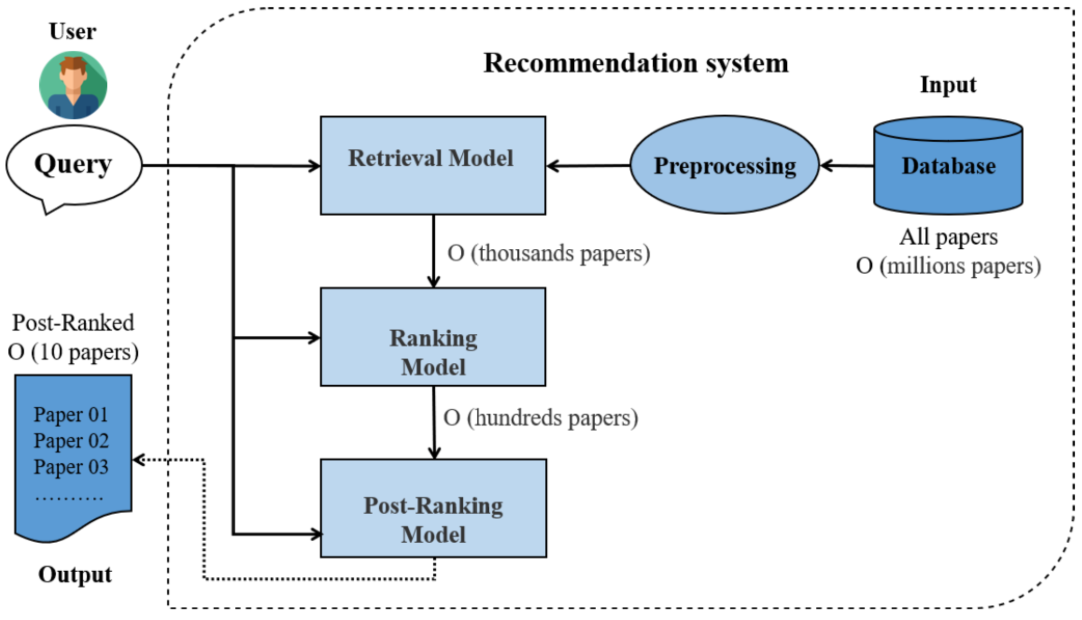


Figure :3.1 System Design and Architecture

3.2.1 Preprocessing Stage

The preprocessing stage is the initial step in the system pipeline, where the dataset of thousands of papers is processed and transformed into a more manageable format. This stage involves various tasks such as data cleaning, normalization, and feature extraction. The preprocessing tasks aim to improve the quality of the dataset, remove irrelevant or redundant information, and extract essential features that will be used in subsequent stages.

3.2.2 Retrieval Stage

The retrieval stage focuses on selecting a subset of papers from the preprocessed dataset that are most relevant to a given query or user context. This stage utilizes retrieval techniques to match the query with the available papers based on their textual content, metadata, or other relevant factors. Various retrieval algorithms, such as vector space models or neural networks, may be employed to determine the similarity between the query and the papers in the dataset. The output of this stage is a reduced set of thousands of papers that are deemed relevant to the query.

3.2.3 Ranking Stage

The ranking stage aims to further refine the retrieved set of papers by assigning a relevance score or rank to each paper. This stage leverages ranking algorithms that take into account various factors such as paper quality, citation count, author reputation, or user feedback. The ranking algorithms analyze the characteristics of each paper and assign a score to determine its relative importance or relevance within the retrieved set. As a result, the ranking stage produces a ranked list to establish the most relevant hundreds of papers based on their scores.

3.2.4 Post-Ranking Stage

The post-ranking stage focuses on enhancing the ranked list of papers by applying additional post-processing techniques. This stage may involve techniques such as result diversification, clustering, or user-specific filtering. Result diversification aims to provide a diverse set of papers to cater to different user preferences or avoid redundancy. Clustering techniques can group similar papers together to improve the organization and presentation of the results. User-specific filtering can further personalize the ranked list based on a user's specific interests, previous interactions, or feedback. The output of the post-ranking stage is a refined and personalized set of papers that better aligns with the user's needs.

3.2.5 Query Stage

In addition to the data flow through the preprocessing, retrieval, ranking, and post-ranking stages, the system also accommodates user queries. Users input their specific query, such as a research topic or keywords, to receive personalized and relevant citation recommendations based on their specific needs.

The system design and architecture presented here provide an overview of the transformation process from the dataset of millions of papers through the preprocessing, retrieval, ranking, and post-ranking stages. The specific implementation details, algorithms, and technologies used may vary based on the specific requirements and characteristics of the dataset and the target application.

3.3 System Components and Functionality

Our deep learning context-aware citation recommendation system consists of several key components that work together to provide efficient and accurate recommendations. Each component plays a vital role in the overall functionality of the system. Here we explore the main components and their respective functionalities:

3.3.1 Retrieval Description and Role

The retrieval stage is a fundamental component in our system. It plays a crucial role in retrieving a subset of relevant papers from a vast dataset based on the user's query. This stage focuses on efficiently narrowing down the search space and identifying papers that are most likely to be of interest to the user.

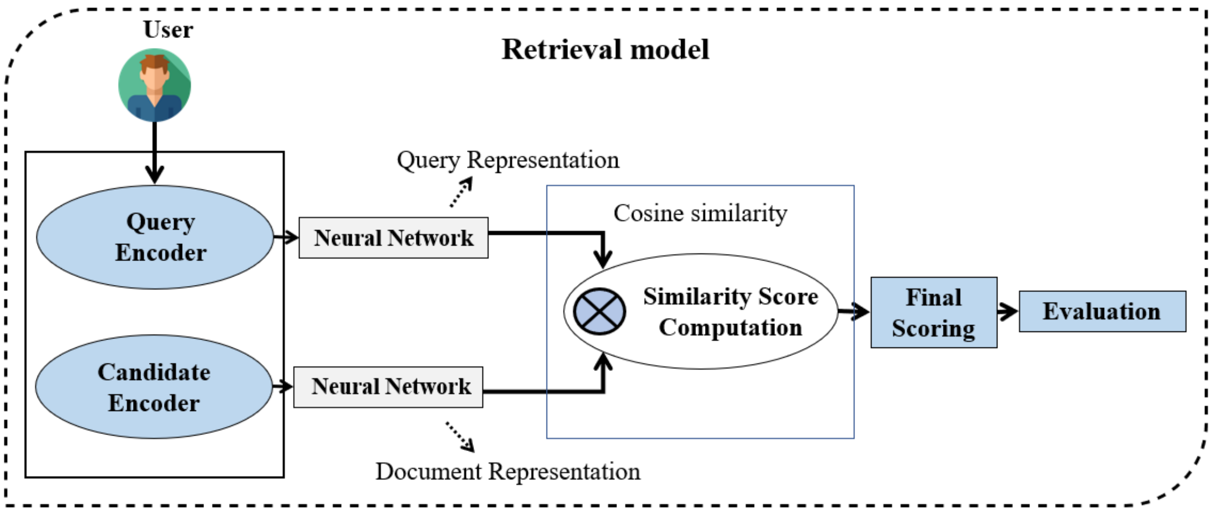


Figure :3.2 Internal Retrieval Model Architecture

- **Query Encoder:** this component is responsible for encoding the user query, which represents the information the user provides to search for relevant papers. It transforms the query into a numerical representation that can be processed by the retrieval model using embedding technique.
- **Candidate Encoder:** the candidate encoder takes the features of the candidate papers, such as titles and abstracts, and encodes them into numerical representations by applying embedding technique. These representations capture the important characteristics of the candidate papers.
- **Similarity Score Computation:** the similarity score computation calculates the similarity between the encoded query and the encoded candidate papers. It measures the relevance or similarity between the user query and the candidate papers, allowing for retrieval of the most relevant papers.
- **Evaluation:** is an essential aspect of any retrieval model to assess its performance and effectiveness. Various evaluation metrics are used to measure the quality of the retrieval results precision, recall, F1-score, mean average precision (MAP).

3.3.2 Ranking Description and Role

The ranking stage is a critical component in our system. It follows the retrieval stage and focuses on ranking the retrieved candidate papers based on their relevance to the user's query. The ranking stage plays a vital role in determining the order in which the recommended papers are presented to the user.

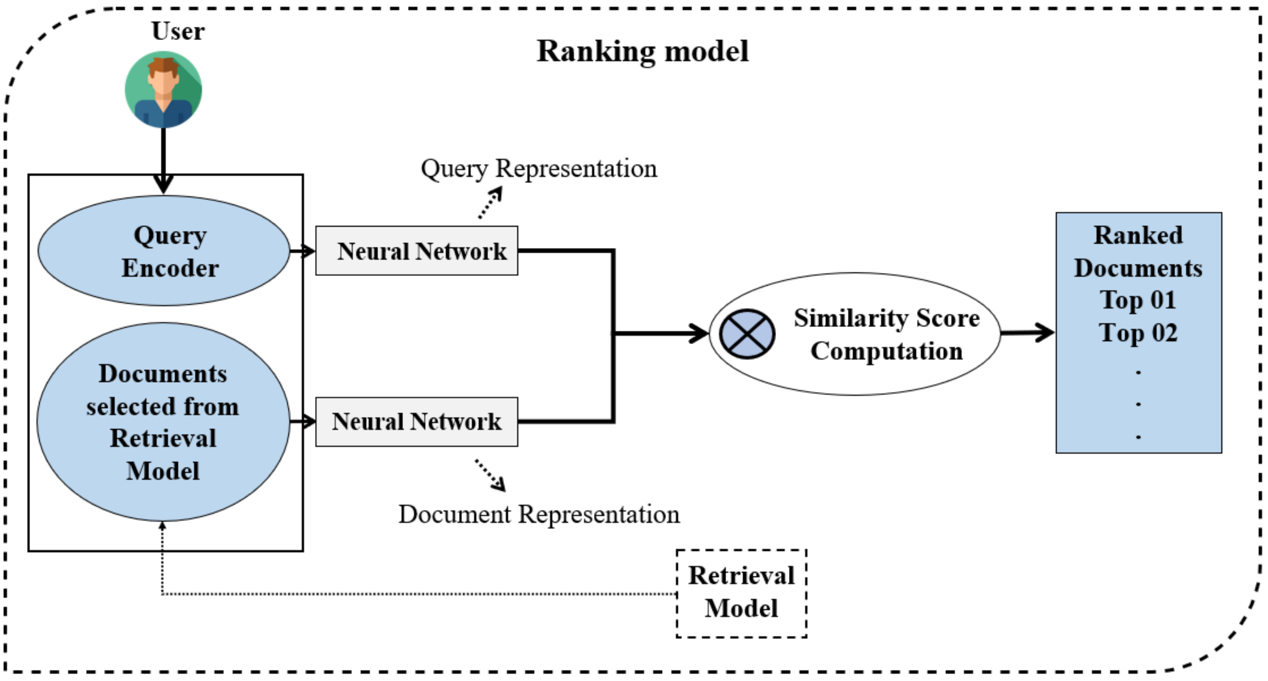


Figure :3.3 Internal Ranking Model Architecture

Our ranking model follows a straightforward architecture that efficiently generates recommendations based on user queries and a collection of documents. The process begins with the user submitting a query, specifying their information need or topic of interest. Simultaneously, the retrieval model selects a relevant set of documents from the entire collection based on various retrieval techniques.

Once the query and documents are transformed into numerical representations, the ranking model computes a similarity score between the query and each document in the collection. This similarity score serves as a measure of relevance, indicating how well each document aligns with the user's query. The computation of the similarity score is clculated by cosine similarity.

Finally, based on the computed similarity scores, the ranking model arranges the collection of documents in descending order, placing the most relevant documents at the top of the ranking list. This ranking enables users to quickly identify and access the most pertinent documents that address their information need.

3.3.3 Post-ranking Description and Role

The post-ranking stage is a crucial component in our system. It follows the initial ranking stage and focuses on further refining the ranking of recommended papers based on additional factors and considerations. The post-ranking stage plays a vital role in fine-tuning the recommendations and improving their relevance to the user's needs.

3.4 Sequence diagram

Sequence diagram provides a visual representation of the flow and interactions between different components or actors in a system. It illustrates the chronological order and dependencies of various operations involved in the citation recommendation process. It showcases the sequence of messages, actions, and responses among the system components, highlighting the dynamic behavior of the system. By presenting a step-by-step depiction of the system's functionality, the sequence diagram offers a clear understanding of how different components collaborate and exchange information to accomplish the citation recommendation task.

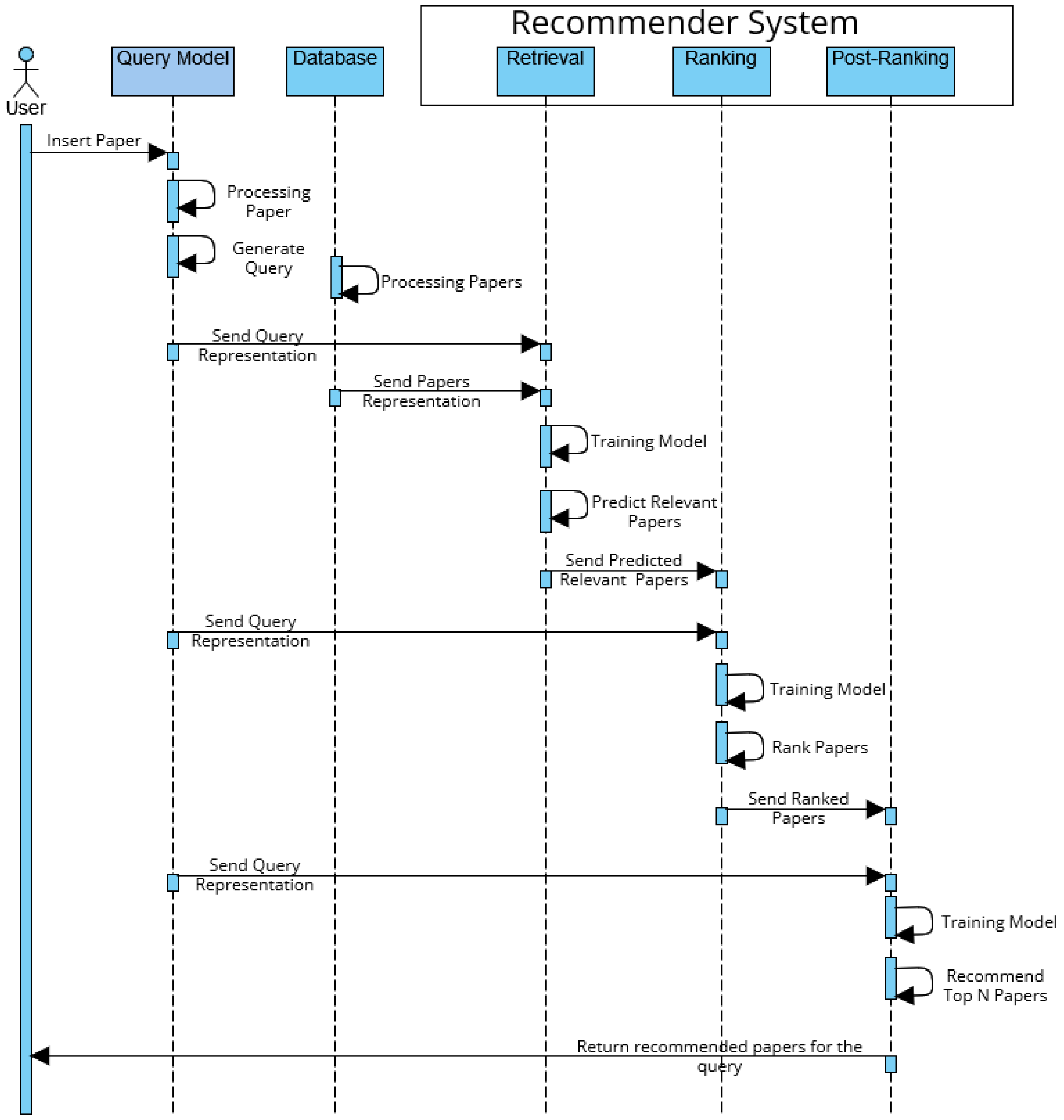


Figure :3.4 Sequence Diagram: Interaction Flow of the Proposed System Components

The sequence diagram 3.4illustrates the flow of actions in the system for inserting and processing papers, generating queries, retrieving relevant papers, ranking them, and recommending top k papers to the user.

- The process starts with the user inserting a paper into the system.
- The query model receives the inserted paper and processes it to generate a query.
- The retrieval model takes the query representation and the representations of papers as input and predicts the relevant papers.
- Once the papers are predicted to be relevant, they are sent along with the generated query representation to the rank model.
- The rank model, which has been trained, ranks the papers based on their relevance to the query.
- The post-rank model receives the ranked papers and the generated query representation.
- The post-rank model, recommends the top k papers by considering their ranking.
- Finally, the system sends the top k recommended papers to the user.

This sequence diagram demonstrates the flow of data and actions within the system, showing how papers are processed, queries are generated, papers are retrieved and ranked, and ultimately, top k papers are recommended to the user.

3.5 Conclusion

In this chapter, we have discussed the system design and architecture, including various stages such as the preprocessing stage, retrieval stage, ranking stage, post-ranking stage, and query stage. The preprocessing stage involves preparing the data, while the retrieval stage focuses on finding relevant information based on user queries. The ranking stage assigns ranks to the retrieved items, and the post-ranking stage further refines the ranked information. The query stage handles user queries and provides appropriate responses. We have also described in detail the system components and their functionalities, including the retrieval component responsible for retrieving information, the ranking component for assigning ranks, and the post-ranking component for refining results. Finally, a sequence diagram have been provided to visualize the interactions and flow of messages between the system components.

CHAPTER 4
IMPLEMENTATION AND EVALUATION

4.1 Introduction

In this chapter, we will delve into the details of the implementation process, including the design choices, programming languages, libraries, and frameworks employed. Additionally, we will discuss the evaluation methodology, metrics used, and the results obtained from the experiments conducted on real-world datasets.
The implementation and evaluation chapter plays a crucial role in assessing the effectiveness and performance of the proposed deep learning context-aware technique for citation recommendation. This chapter focuses on the practical implementation of the developed system, including the selection of appropriate tools, technologies, and frameworks. It involves translating the theoretical concepts and algorithms.
Furthermore, the evaluation aspect of this section aims to measure the system's performance, effectiveness, and efficiency.

4.2 Development environment

The development environment section provides an overview of the tools, technologies, and resources used to implement our system.
It encompasses the hardware and software setup that facilitated the development and experimentation process.
In this section, we will discuss the hardware infrastructure employed, including the specifications of the computer systems used for training and evaluating the models. This may include details such as the processor, memory, and storage capacity, which are crucial for executing resource-intensive deep learning tasks.

4.2.1 Hard-ware environment

- 1^{st} computer:
  PC:Acer computer
  Processor: Intel Core i3 5th generation
  Hard disk:hdd
  RAM: 8GB
  Storage: 466GB
  Operating System: Windows 8

4.2.2 Soft-ware environment

- **Python:** is a high-level programming language known for its simplicity, readability, and versatility. It was created by Guido van Rossum and first released in 1991. Python emphasizes code readability and uses a clean syntax, making it easy to learn and understand for both beginners and experienced programmers.

- **Tensorflow:** TensorFlow is an open-source machine learning framework developed by Google. It provides a comprehensive ecosystem of tools, libraries, and resources for building and deploying machine learning models. TensorFlow is designed to efficiently handle large-scale numerical computations and is particularly well-suited for training and deploying deep neural networks. It offers a flexible and intuitive interface for constructing computational graphs and executing operations on various hardware platforms, including CPUs and GPUs. TensorFlow has gained popularity for its ability to simplify the development of complex machine learning models.

- **Keras:** is an open-source high-level neural networks library written in Python. It is built on top of TensorFlow, Theano, or CNTK, providing a user-friendly interface for building and training deep learning models. Keras offers a modular and intuitive API that allows developers to quickly prototype and experiment with various neural network architectures. It supports both convolutional and recurrent neural networks. Keras abstracts away the complexities of low-level implementation details, making it easy for beginners to get started with deep learning. It also provides a range of pre-trained models and utilities for common tasks such as natural language processing. Overall, Keras is known for its simplicity, flexibility, and ease of use.

4.3 Dataset Collection and Acquisition

In this section, the emphasis is on discussing the methods and techniques used to collect and acquire the dataset, ensuring its quality and relevance to the project's objectives. It may involve data collection from online sources, data scraping, data extraction from databases, or collaborations with organizations or institutions that provide access to specific datasets.

4.3.1 Selecting the sources of research papers

A variety of data sets for the topic citation recommendation systems have been collected in order to train and evaluate models created for this topic. Datasets used for citation recommendation systems consist of two parts, collection of scientific articles and information about citations between this articles.
For this topic we find several available datasets like DBLP, RefSeer, citeUlike-a and many others. Difference between datasets is: some datasets include full article text while others include just abstract of article, another distinction: while some datasets list only the articles cited, others include the textual context for those citations.
As result of these differences, some datasets are only suitable for global CR tasks (datasets without context), whereas others can be used for both global and local CR tasks. Figure 4.1 below shows the sundary datasets known in citation recommendation task.

Table :4.1 Common Data sets for Citation recommendation specifications[4]

Data sets	Papers	Users / Authors	Citation-relations	Citation-context	Tags	Key-phrases	Venues	Title	Abstract	Full text	Year info	Ratings	Release year
ACL-ARC[1]	23766	18862	✓	-	-	-	373	✓	✓	✓	-	-	2016
DBLP[2]	4,107,340	318,406	✓	-	-	-	23709	✓	✓	-	✓	-	2019
RefSeer[3]	855,735	-	✓	✓	-	✓	-	-	✓	✓	✓	-	2015
Amnier[4]	3,680,007	212,567	✓	-	-	-	12770	✓	✓	-	✓	-	2017
PubMed[5]	789	212,312	✓	-	✓	✓	2319	✓	✓	✓	-	-	2015
OpenCorpus[6]	6,090,000	8,030,000	✓	-	-	✓	23672	✓	✓	-	✓	-	2018
Citeulike-a[7]	16980	5551	✓	-	✓	-	-	✓	✓	-	-	✓	2013

The citeulike-a dataset was carefully chosen as the primary dataset for our citation recommendation system. This dataset played a pivotal role in training and evaluating the models developed in our research. Citeulike-a offers a rich collection of scientific articles along with comprehensive information about the citations between these articles. By leveraging this dataset, we were able to capture the interconnectedness of scholarly works and extract valuable insights for effective citation recommendation.
By utilizing the citeulike dataset, we aimed to ensure the representativeness of our research findings within the scholarly domain. The dataset encompasses a wide range of research areas and covers a substantial volume of articles, allowing us to achieve a broader perspective and overcome potential biases associated with narrower datasets. Moreover, the richness and depth of the CiteULike-a dataset empower our models to capture intricate relationships and semantic connections between research papers, enabling more precise and context-aware citation recommendations.
In summary, the CiteULike-a dataset serves as the foundation of our citation recommendation system, providing the necessary data to train and evaluate our models. Its comprehensive coverage, contextual information, and representation of various research domains make it a valuable resource in advancing context-aware citation recommendation.

[1](https://acl-arc.comp.nus.edu.sg)
[2](https://dblp.dagstuhl.de)
[3](https://citeseerx.ist.psu.edu)
[4](https://www.aminer.cn/data)
[5](https://pubmed.ncbi.nlm.nih.gov)
[6](https://www.semanticscholar.org/product/api)
[7](https://github.com/js05212/citeulike-a)

4.3.2 Defining the criteria for inclusion in the dataset

When defining the criteria for inclusion in the dataset, we toke several considerations into account to ensure the dataset's quality and relevance for citation recommendation research. The following criteria were established:

- **Research Paper Relevance:** only research papers directly related to the target domain or field of study were included in the dataset. Papers from diverse disciplines were considered to capture a wide range of topics.In the CiteULike-a dataset, research paper relevance played a crucial role in the selection process. The goal was to compile a dataset that encompasses a diverse range of scholarly articles while ensuring their alignment with specific research areas.

- **Language:** papers written in a specific language or languages were chosen to maintain consistency and facilitate analysis. The CiteULike-a dataset was written in English.

- **Abstract Screening:** the abstracts of the papers in citeulike-a were reviewed to assess their alignment with the selected topic(s). Papers with abstracts that demonstrated relevance to the research area were given priority.

- **Expert Knowledge:** the expertise of domain experts or researchers familiar with the topic(s) was utilized to ensure the inclusion of papers that are highly relevant and significant within the field.

- **Publication Type:** CiteULike-a dataset provides a diversity in types of publications, such as conference papers and journal articles, were considered to provide a comprehensive representation of scholarly communication.

- **Availability:** the citeulike-a dataset is publicly available for research purposes. It can be accessed and downloaded from the official citeulike website or other data repositories where it has been made available. The dataset is often provided in a structured format, such as CSV or XML, making it easier for researchers to process and analyze the data.

By employing these criteria, the CiteULike-a dataset was carefully curated to include research papers that are pertinent to the chosen research area, enabling the development and evaluation of citation recommendation models that effectively cater to the specific domain.

4.3.3 Value of dataset

- The dataset has been utilized to create and assess recommender systems for scholarly papers. The data can be utilized to train machine learning models that can make personalized recommendations based on a interests of users, inclinations.
- A free service that assists you in storing, organizing, and sharing the scholarly papers you are reading.
- It enables users to include their academic reference library in their online profile on the CiteULike website.
- CiteULike allows users to engender their own collections of articles.
- Can be utilized to create and test information science and machine learning models. For case, analysts may utilize it to prepare models to foresee the affect of scholarly distributions or recognize the foremost significant articles for a specific inquire about address.

To recap , the ensuing Table 4.2 includes all previously mentioned information:

Table :4.2 Specification Table

Subject	Computer Science.
Type of data	Table.
Data were acquired	The process of obtaining the CiteULike-a dataset included web scraping and data processing methods.Web scraping: The CiteULike site was crawled and significant information was mined from the website's HTML code. This included writing code to scoop out paper titles, authors, abstracts, publication venues, and other metadata from the site. Besides the processing methods involved data cleansing by getting rid of duplicate information and correcting errors.
Data description	The CiteULike-A dataset is a portion of the CiteULike dataset, which is extracted from the CiteULike platform. CiteULike-A particularly alludes to a version of the dataset that has been preprocessed and made accessible for use in research and analysis. The CiteULike-A incorporates 16980 papers. Each paper (document) includes metadata such as title, users, abstract, paper_ID, publication year, as well as citations. The CiteULike-A dataset has been utilized in different research studies such as citation analysis.
Data accessibility	Citeulike-a dataset is available publicly Press here.

4.3.4 Ensuring that dataset is representative for the field of study

According to the previous subsection, "Defining the criteria for inclusion in the dataset," we are able to ensure the quality and relevance of the CiteULike-a dataset for citation recommendation research. So in brief we can confidently state that the CiteULike-a dataset is representative of the field of study.

4.3.5 Data Collecting and Format

CiteULike-a dataset was collected from CiteULike and Google Scholar using web scrapping techniques to mine a significant informations such as document title, abstract, users,publication venues and other informations . It was released as portion of the RecSys Challenge 2013 a competition to develop algorithms for recommending scientific articles to users based on their past lecture behavior. CiteULike-a dataset contains seven files among them raw-data.csv ,users.dat,citations.dat which we adopted in this memorandum.

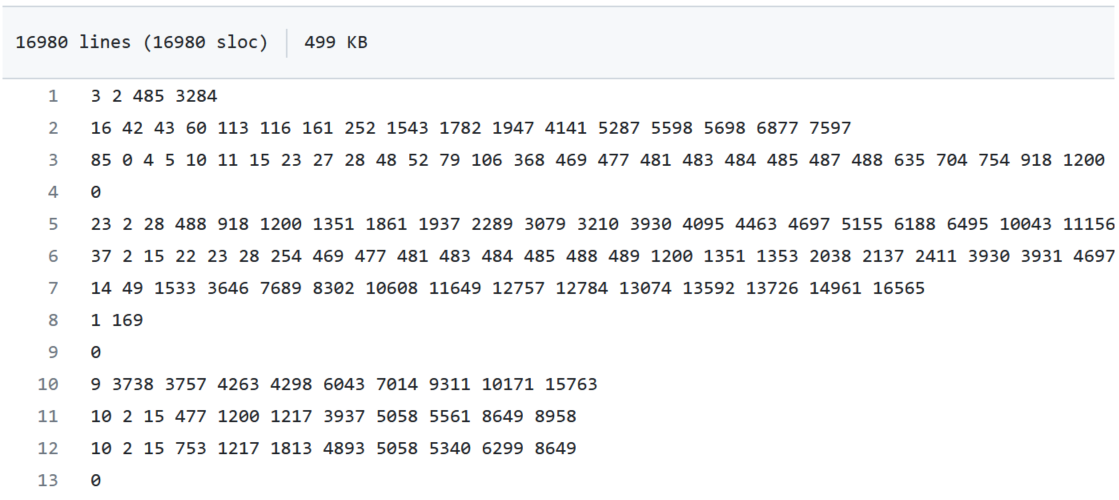
Some statistics are represented as follow:

Tableau :4.3 Total number of each attribute in citeULike-a

Entity	Totaling number
tags	46391
Items	16980
Users	5551
Citations	44709
User-item pairs	204987

Dataset CiteULike-a for CTRSR is collection of 16980 articles or documents.

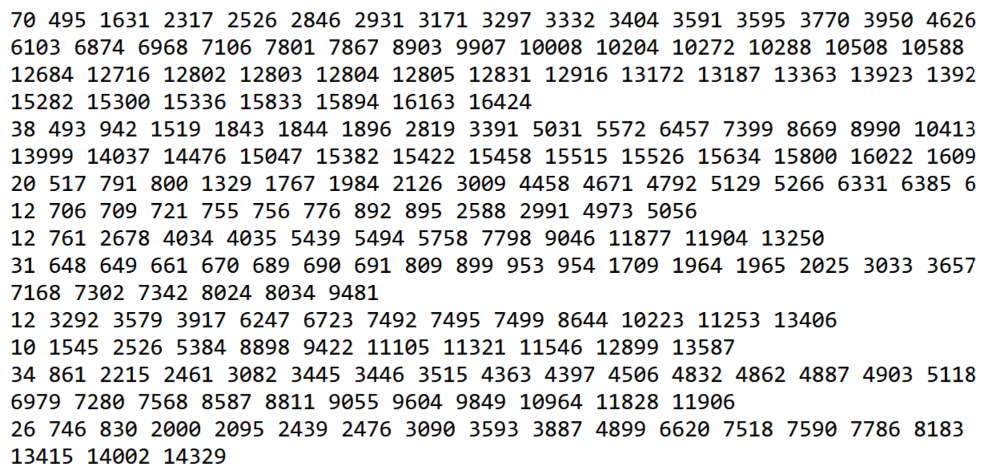
- **citation.dat:** each row in citation.dat corresponds to an edge connected to a node(article). For instance row 1: 3 2 485 3284 that indicates there is 3 edges connected article number 0 (taking into consideration that indexing starts from zero) their ID's are respectively 2, 485 and 3284. In another sense, the article that defined by "The metabolic world of Escherichia coli is not small" as title is connected to "Community structure in social and biological networks", "Exploring complex networks", "Reconstruction of metabolic networks from genome data and analysis of their global structure for various organisms".



```
16980 lines (16980 sloc)  |  499 KB

3 2 485 3284
16 42 43 60 113 116 161 252 1543 1782 1947 4141 5287 5598 5698 6877 7597
85 0 4 5 10 11 15 23 27 28 48 52 79 106 368 469 477 481 483 484 485 487 488 635 704 754 918 1200
0
23 2 28 488 918 1200 1351 1861 1937 2289 3079 3210 3930 4095 4463 4697 5155 6188 6495 10043 11156
37 2 15 22 23 28 254 469 477 481 483 484 485 488 489 1200 1351 1353 2038 2137 2411 3930 3931 4697
14 49 1533 3646 7689 8302 10608 11649 12757 12784 13074 13592 13726 14961 16565
1 169
0
9 3738 3757 4263 4298 6043 7014 9311 10171 15763
10 2 15 477 1200 1217 3937 5058 5561 8649 8958
10 2 15 753 1217 1813 4893 5058 5340 6299 8649
0
```

Figure :4.1 1st examples in row in citation.dat

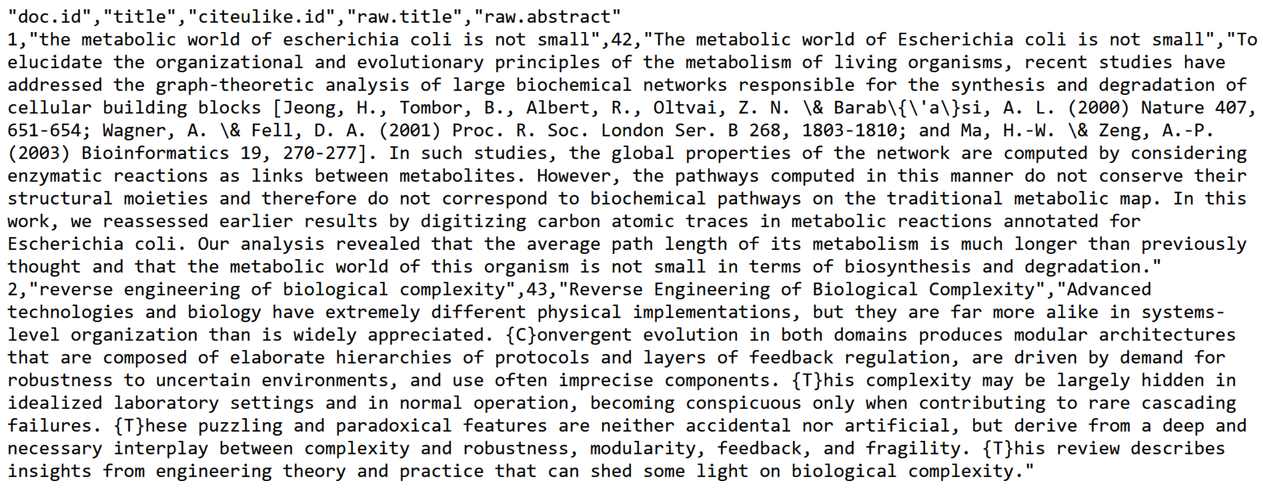
- **users.dat:** each row in users.dat corresponds to total number of papers that each user may have access and their ID's.For example raw 1: 70 495 1631 2317 2526 2846 2931 3171 3297 3332...means that user_1 have 70 papers that accessed to read their ID's are consecutively 495,1631,2317,.....



```
70 495 1631 2317 2526 2846 2931 3171 3297 3332 3404 3591 3595 3770 3950 4626
6103 6874 6968 7106 7801 7867 8903 9907 10008 10204 10272 10288 10508 10588
12684 12716 12802 12803 12804 12805 12831 12916 13172 13187 13363 13923 1392
15282 15300 15336 15833 15894 16163 16424
38 493 942 1519 1843 1844 1896 2819 3391 5031 5572 6457 7399 8669 8990 10413
13999 14037 14476 15047 15382 15422 15458 15515 15526 15634 15800 16022 1609
20 517 791 800 1329 1767 1984 2126 3009 4458 4671 4792 5129 5266 6331 6385 6
12 706 709 721 755 756 776 892 895 2588 2991 4973 5056
12 761 2678 4034 4035 5439 5494 5758 7798 9046 11877 11904 13250
31 648 649 661 670 689 690 691 809 899 953 954 1709 1964 1965 2025 3033 3657
7168 7302 7342 8024 8034 9481
12 3292 3579 3917 6247 6723 7492 7495 7499 8644 10223 11253 13406
10 1545 2526 5384 8898 9422 11105 11321 11546 12899 13587
34 861 2215 2461 3082 3445 3446 3515 4363 4397 4506 4832 4862 4887 4903 5118
6979 7280 7568 8587 8811 9055 9604 9849 10964 11828 11906
26 746 830 2000 2095 2439 2476 3090 3593 3887 4899 6620 7518 7590 7786 8183
13415 14002 14329
```

Figure :4.2 1st examples in row in users.dat

- **row in raw_data.csv:** each row in raw_data.csv represents an article. And each article defined by it's ID, Title, Abstract and citeulike.id.

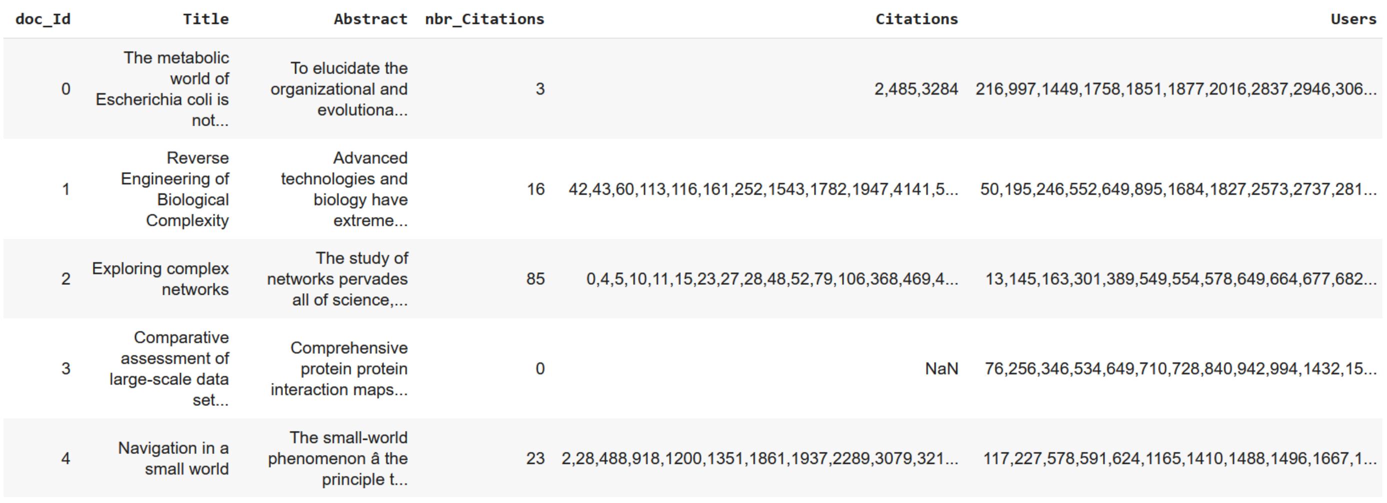


```
"doc.id","title","citeulike.id","raw.title","raw.abstract"
1,"the metabolic world of escherichia coli is not small",42,"The metabolic world of Escherichia coli is not small","To
elucidate the organizational and evolutionary principles of the metabolism of living organisms, recent studies have
addressed the graph-theoretic analysis of large biochemical networks responsible for the synthesis and degradation of
cellular building blocks [Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N. \& Barab\{\'a\}si, A. L. (2000) Nature 407,
651-654; Wagner, A. \& Fell, D. A. (2001) Proc. R. Soc. London Ser. B 268, 1803-1810; and Ma, H.-W. \& Zeng, A.-P.
(2003) Bioinformatics 19, 270-277]. In such studies, the global properties of the network are computed by considering
enzymatic reactions as links between metabolites. However, the pathways computed in this manner do not conserve their
structural moieties and therefore do not correspond to biochemical pathways on the traditional metabolic map. In this
work, we reassessed earlier results by digitizing carbon atomic traces in metabolic reactions annotated for
Escherichia coli. Our analysis revealed that the average path length of its metabolism is much longer than previously
thought and that the metabolic world of this organism is not small in terms of biosynthesis and degradation."
2,"reverse engineering of biological complexity",43,"Reverse Engineering of Biological Complexity","Advanced
technologies and biology have extremely different physical implementations, but they are far more alike in systems-
level organization than is widely appreciated. {C}onvergent evolution in both domains produces modular architectures
that are composed of elaborate hierarchies of protocols and layers of feedback regulation, are driven by demand for
robustness to uncertain environments, and use often imprecise components. {T}his complexity may be largely hidden in
idealized laboratory settings and in normal operation, becoming conspicuous only when contributing to rare cascading
failures. {T}hese puzzling and paradoxical features are neither accidental nor artificial, but derive from a deep and
necessary interplay between complexity and robustness, modularity, feedback, and fragility. {T}his review describes
insights from engineering theory and practice that can shed some light on biological complexity."
```

Figure :4.3 1st examples raw_data.csv

4.3.6 Organizing the dataset into a usable format

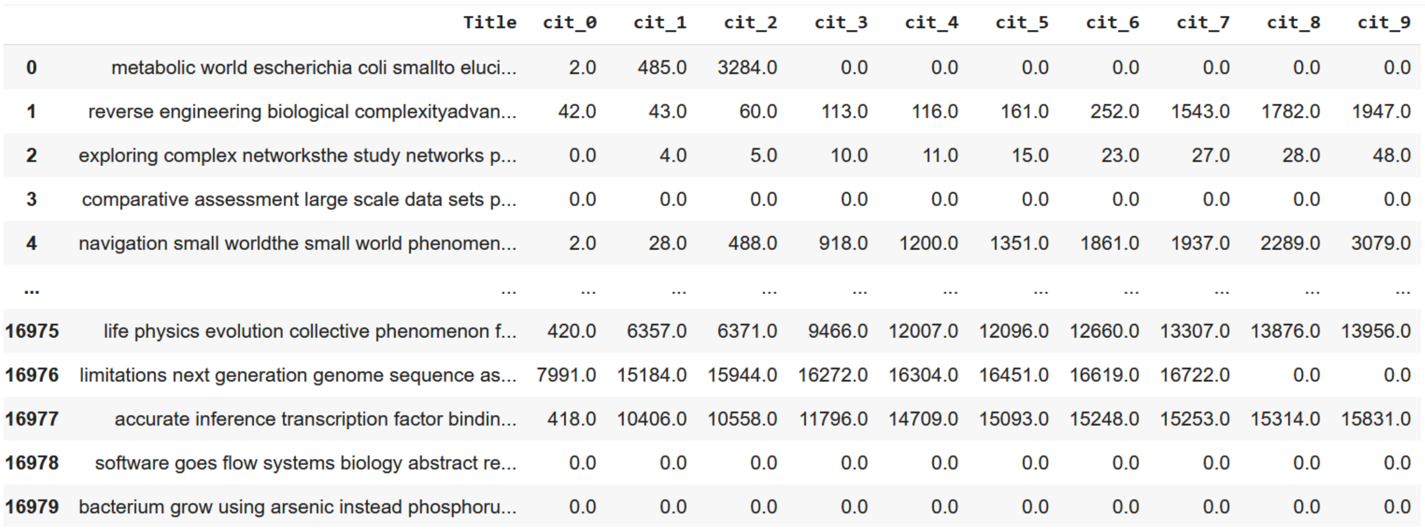
We selected the three files, raw-data.csv, users.dat, and citations.dat, from the CiteULike-a dataset available at the specified GitHub repository. These files contain the necessary information about research papers, users, and citations within the dataset.
We integrated the information from these three files by merging the relevant fields. This integration ensured that all essential information was captured in the final dataset. Where we concatinate the 3 files by ID's.
To facilitate data handling and compatibility with various analysis tools and programming languages, we transformed the dataset into a tabular format, typically using CSV (Comma-Separated Values) format. This format allows for easy storage, retrieval, and manipulation of the dataset.
After meticulous organization and careful processing, the format of final dataset as it is shown in the figure bellow. The result is a comprehensive and refined collection of data that encompasses a wealth of valuable information. The dataset now stands as a testament to our commitment to providing reliable and accurate data for our system.



doc_Id	Title	Abstract	nbr_Citations	Citations	Users
0	The metabolic world of Escherichia coli is not...	To elucidate the organizational and evolutiona...	3	2,485,3284	216,997,1449,1758,1851,1877,2016,2837,2946,306...
1	Reverse Engineering of Biological Complexity	Advanced technologies and biology have extreme...	16	42,43,60,113,116,161,252,1543,1782,1947,4141,5...	50,195,246,552,649,895,1684,1827,2573,2737,281...
2	Exploring complex networks	The study of networks pervades all of science,...	85	0,4,5,10,11,15,23,27,28,48,52,79,106,368,469,4...	13,145,163,301,389,549,554,578,649,664,677,682...
3	Comparative assessment of large-scale data set...	Comprehensive protein protein interaction maps...	0	NaN	76,256,346,534,649,710,728,840,942,994,1432,15...
4	Navigation in a small world	The small-world phenomenon â the principle t...	23	2,28,488,918,1200,1351,1861,1937,2289,3079,321...	117,227,578,591,624,1165,1410,1488,1496,1667,1...

Figure :4.4 Head of the CiteULike-a dataset

The final dataset is presented below, where each row represents a research paper and its corresponding features. The title of the paper represents the concatenation of its title and abstract, providing a comprehensive identifier and the other columns, each one represent the citations for the corresponding paper.



	Title	cit_0	cit_1	cit_2	cit_3	cit_4	cit_5	cit_6	cit_7	cit_8	cit_9
0	metabolic world escherichia coli smallto eluci...	2.0	485.0	3284.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	reverse engineering biological complexityadvan...	42.0	43.0	60.0	113.0	116.0	161.0	252.0	1543.0	1782.0	1947.0
2	exploring complex networksthe study networks p...	0.0	4.0	5.0	10.0	11.0	15.0	23.0	27.0	28.0	48.0
3	comparative assessment large scale data sets p...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	navigation small worldthe small world phenomen...	2.0	28.0	488.0	918.0	1200.0	1351.0	1861.0	1937.0	2289.0	3079.0
...	...	...	...	...	...	...	...	...	...	...	...
16975	life physics evolution collective phenomenon f...	420.0	6357.0	6371.0	9466.0	12007.0	12096.0	12660.0	13307.0	13876.0	13956.0
16976	limitations next generation genome sequence as...	7991.0	15184.0	15944.0	16272.0	16304.0	16451.0	16619.0	16722.0	0.0	0.0
16977	accurate inference transcription factor bindin...	418.0	10406.0	10558.0	11796.0	14709.0	15093.0	15248.0	15253.0	15314.0	15831.0
16978	software goes flow systems biology abstract re...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16979	bacterium grow using arsenic instead phosphoru...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure :4.5 Final dataset

4.4 Dataset Preprocessing

In order to ensure the integrity and relevance of the dataset used for this study, a thorough cleaning process was conducted to remove duplicates and irrelevant papers as it depicted in Figure 4.6. This subsection outlines the steps taken to clean the dataset, ensuring the accuracy and reliability of the data used for analysis.

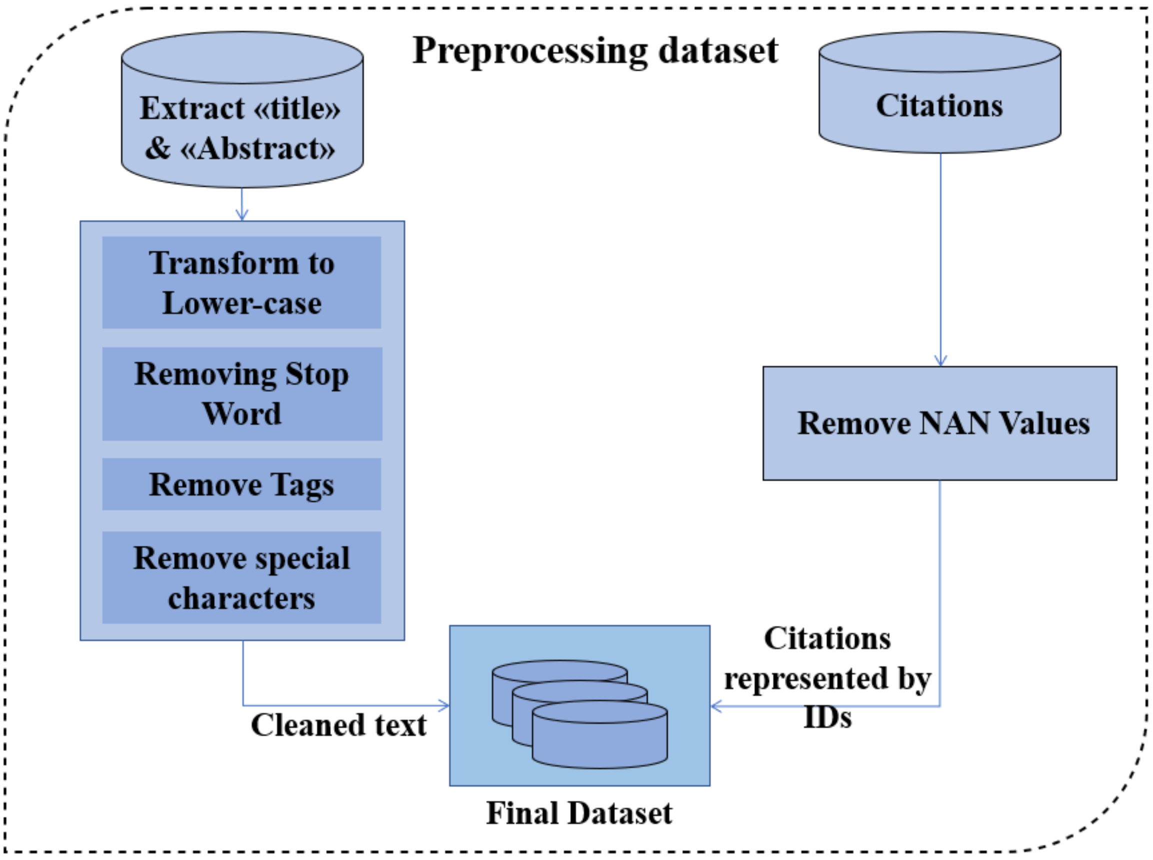


Figure :4.6 Dataset Preprocessing Pipeline

4.4.1 Removing Duplicate Papers

Duplicate papers can introduce bias and lead to skewed results in the analysis. To address this issue, a duplicate removal process was conducted. Initially, a similarity measure was applied to identify potential duplicates based on the textual content of the papers. This measure compared the similarity between titles and abstracts of different papers. Once potential duplicates were identified, manual inspection and verification were performed to validate and retain the most relevant version of the paper. By removing duplicate papers, the dataset's quality and integrity were enhanced, ensuring each paper represented a unique contribution.

4.4.2 Lowercase Conversion

To maintain consistency and facilitate text analysis, all text data in the dataset was converted to lowercase. This step helped eliminate any potential discrepancies arising from inconsistent capitalization in titles and abstracts. By converting all text to lowercase, the dataset became standardized and easier to process.

4.4.3 Removing Tags

Tags, such as HTML or XML tags, are commonly found in scraped or downloaded datasets. These tags provide formatting information but are irrelevant for the analysis. Therefore, a tag removal process was implemented to strip off any tags present in the dataset. This step ensured that only meaningful content remained for further cleaning and analysis.

4.4.4 Removing Special Character and Digit

Special characters, such as punctuation marks and symbols, do not typically contribute significant meaning to the text analysis. Therefore, a special character and digit removal process was applied to eliminate them from the dataset. This step involved systematically scanning each document and removing any special characters or digits encountered. The resulting dataset contained only alphabetic characters and words, which were more suitable for subsequent analysis.

4.4.5 Removing Stop Word

Stop words are commonly occurring words in a language, such as "and," "the," or "in," that do not provide substantial meaning in the context of the analysis. These words can be noise and hinder the accuracy of text analysis algorithms. Hence, a stop word removal process was implemented to eliminate stop words from the dataset. A predefined list of stop words specific to the research domain was used to identify and remove such words. This step enhanced the quality of the dataset by eliminating noise and reducing the dimensionality of the text data.

4.4.6 Visualization of some features

Data visualization is a powerful technique for transforming complex information into visual formats such as maps and charts. It serves the purpose of aiding human comprehension and extracting insights from data. By presenting data visually, it becomes easier to identify patterns, trends, and outliers, especially when dealing with large datasets.
In the context of our study, we have utilized data visualization techniques to explore and understand the dataset. As shown in Figure 4.4, we observed that certain features in the dataset are categorical, meaning they have non-numeric values. To effectively model these features, we employed the embedding technique, to transform the categorical data into numerical representations.
By transforming the title and abstract fields into numerical data through the embedding technique using "bert-base-nli-mean-tokens" model , we can now establish relationships between the previously mentioned features. With the numerical representations, it becomes easier to model and analyze the relationships between variables. This allows us to explore how each feature is related to the others and gain a deeper understanding of the dataset.

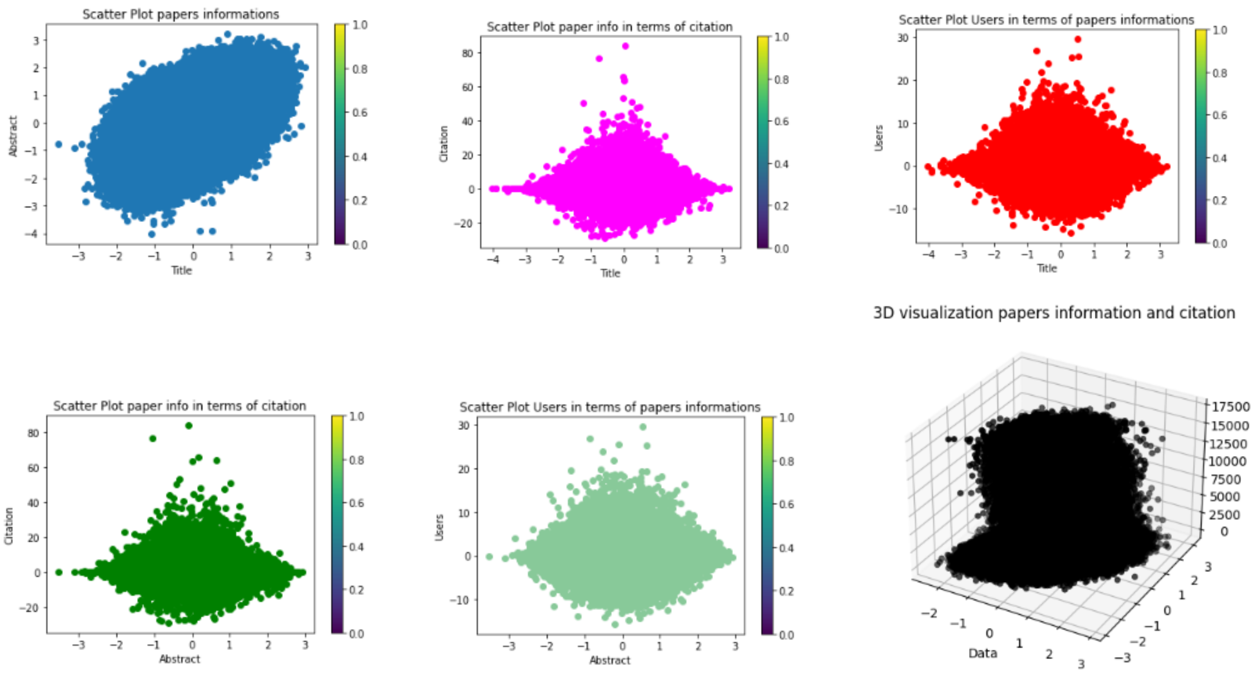


Figure :4.7 Visualization of sundry features in terms of each others

Figure 4.7 provides a visual representation of the correlations between the previously mentioned features. It demonstrates that these features exhibit a significant level of

correlation with each other. This finding is valuable as it indicates that changes in one feature are associated with changes in other features, suggesting potential interdependencies within the system under consideration.
The correlation among the features serves as a beneficial tool for gaining a better understanding of the underlying system. By examining the strength and direction of the correlations, we can identify relationships and dependencies between variables. This knowledge enables us to make more informed interpretations and predictions based on the observed patterns.

4.4.7 Checking for consistency and accuracy in the data

To ensure the reliability and accuracy of the data used in our study, we conducted thorough checks for consistency and accuracy. Data visualization played a crucial role in exploring and understanding the CiteULike-a dataset, as it allowed us to transform complex information into visual formats such as maps and charts. By presenting the data visually, we were able to identify patterns, trends, and outliers, especially when dealing with large datasets.
In addition to leveraging data visualization techniques, we also performed thorough checks for consistency and accuracy within the CiteULike-a dataset. These checks ensured that the dataset was reliable and of high quality. By validating the consistency of data elements, such publication titles, abstracts, citataions and users, we verified the coherence and conformity of the dataset.
In summary, the CiteULike-a dataset underwent rigorous checks for consistency and accuracy to ensure the reliability of our study's findings. By employing data visualization techniques, we explored the dataset, transformed categorical features into numerical representations, and identified significant correlations among the features. These checks, along with the validation and cleaning processes, allowed us to maintain the integrity of the CiteULike-a dataset and generate meaningful insights from the data.

```python
x=abstract
y=title
plt.scatter(x, y)
plt.title("Scatter Plot papers informations")
# Setting the X and Y labels
plt.xlabel('Title')
plt.ylabel('Abstract')
plt.colorbar( orientation="vertical")
plt.show()
```

The code snippet provided performs a scatter plot visualization of the dataset, where the X-axis represents the "Abstract" and the Y-axis represents the "Title" of the papers. Here's an explanation of the code:

- In this code, the "abstract" variable represents the abstract information of the papers, while the "title" variable represents the title information. The plt.scatter() function is used to create a scatter plot, where each point represents a paper. The X-axis represents the abstracts, and the Y-axis represents the titles of the papers.
- The plt.title() function sets the title of the plot to "Scatter Plot of Papers Information". The plt.xlabel() and plt.ylabel() functions set the labels for the X and Y axes, respectively, as "Abstract" and "Title".
- Additionally, the plt.colorbar() function adds a colorbar to the plot, which provides additional information related to the points plotted. The colorbar helps in interpreting any additional variable associated with the papers, such as the relevance or importance of the papers.
- Finally, the plt.show() function is called to display the scatter plot on the screen.

The given code below snippet performs a 3D visualization of the dataset, incorporating the "abstract" and "title" information along with the "citation" data.
Here's an explanation of the code:

```python
ax=plt.axes(projection='3d')
ax.scatter(abstract,title,citation,cmap='viridis',linewidth=0.5,color = 'black')
ax.set_title('3D visualization papers information and citation')
ax.set_xlabel('Data')
```

- Overall, this code snippet visualizes the dataset in a 3D space, incorporating the abstract, title, and citation information. Each data point is represented by a marker in the plot, with the position in the 3D space determined by the values of the abstract, title, and citation variables.

4.5 Dataset Preprocessing Code

In this section, we provide an overview of the code used for preprocessing the dataset. The code outlined below describes the specific steps and techniques employed to clean and transform the dataset into a suitable format for further processing.

Raw-data

```
df=pd.read_csv('/content/drive/MyDrive/raw-data.csv',encoding='utf-8', encoding_errors='
    replace')
df=pd.DataFrame(df)
df=df.drop(['doc.id', 'title','citeulike.id'], axis=1)
df=df.rename(columns={"raw.title": "Title", "raw.abstract": "Abstract"})
df['Title'] = df['Title'] + df['Abstract']
df=pd.DataFrame(df['Title'])
```

Explanation:

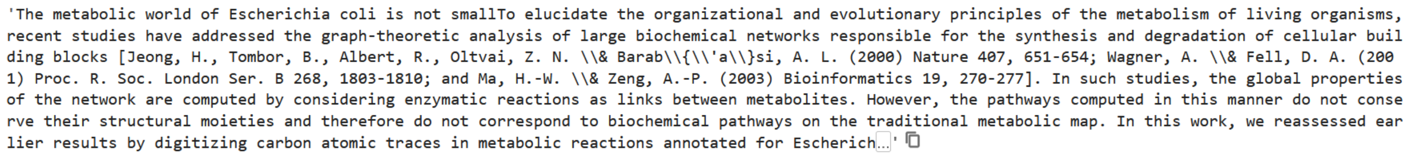
- The code starts by reading the contents of the CSV file. The encoding parameter is set to 'utf-8' to handle the file's encoding, and encoding_errors is set to 'replace' to handle any encoding errors encountered during reading. The resulting data is stored in the variable df.
- The next line converts the df variable into a DataFrame explicitly.
- The code then drops three columns from the DataFrame using the drop() function. This operation removes these columns from the dataset.
- The next line renames two columns in the DataFrame. The column 'raw.title' is renamed to 'Title', and 'raw.abstract' is renamed to 'Abstract'. This step provides more meaningful column names for these attributes.
- The code concatenates the values of the 'Title' and 'Abstract' columns together using the + operator and assigns the result back to the 'Title' column. This concatenation merges the text content of the Title and Abstract fields into a single column, facilitating subsequent processing if needed.
- Finally, the code creates a new DataFrame named df containing only the 'Title' column by assigning df['Title'] to df. This operation discards all other columns, resulting in a DataFrame that solely contains the combined Title and Abstract information.

```python
import re
def pre_process(text):
    # lowercase
    text=text.lower()

    #remove tags
    text=re.sub("</?.*?>"," <> ",text)

    # remove special characters and digits
    text=re.sub("(\\d|\\W)+"," ",text)
    return text
df['Title'] = df['Title'].apply(lambda x:pre_process(x))
```

The provided code defines a function named pre_process that performs text preprocessing operations on a given input text as it is explained:

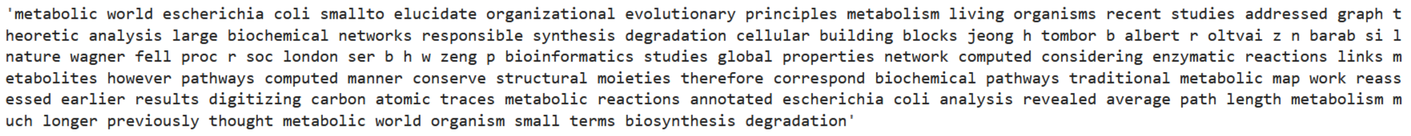
- The code begins by importing the re module, which provides support for regular expressions in Python. This module is necessary to perform pattern-based string operations.
- The code defines a function named pre_process that takes a single parameter, text, representing the input text to be preprocessed.
- The first preprocessing step is to convert the entire input text to lowercase. This is done using the lower() method, which converts all uppercase characters to lowercase.
- The code uses a regular expression pattern to remove any HTML tags present in the text. The re.sub() function is used with the pattern "</?.*?>" to match and replace any occurrences of HTML tags with a space. This effectively removes the tags from the text.
- The next line of code removes special characters and digits from the text. The re.sub() function replaces these matches with a space, effectively removing them from the text.
- The next line, the function returns the preprocessed text.
- Finally, the code applies the pre_process function to each element of the 'Title' column in the DataFrame df and replaces the original values with the preprocessed versions. This step ensures that the text in the 'Title' column is cleaned and normalized according to the defined preprocessing rules before further analysis or processing.

Here, we see the difference between the original text and the cleaned one.



```
'The metabolic world of Escherichia coli is not smallTo elucidate the organizational and evolutionary principles of the metabolism of living organisms,
recent studies have addressed the graph-theoretic analysis of large biochemical networks responsible for the synthesis and degradation of cellular buil
ding blocks [Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N. \\& Barab\\{\\'a\\}si, A. L. (2000) Nature 407, 651-654; Wagner, A. \\& Fell, D. A. (200
1) Proc. R. Soc. London Ser. B 268, 1803-1810; and Ma, H.-W. \\& Zeng, A.-P. (2003) Bioinformatics 19, 270-277]. In such studies, the global properties
of the network are computed by considering enzymatic reactions as links between metabolites. However, the pathways computed in this manner do not conse
rve their structural moieties and therefore do not correspond to biochemical pathways on the traditional metabolic map. In this work, we reassessed ear
lier results by digitizing carbon atomic traces in metabolic reactions annotated for Escherich...'
```

Figure :4.8 Original text



```
'metabolic world escherichia coli smallto elucidate organizational evolutionary principles metabolism living organisms recent studies addressed graph t
heoretic analysis large biochemical networks responsible synthesis degradation cellular building blocks jeong h tombor b albert r oltvai z n barab si l
nature wagner fell proc r soc london ser b h w zeng p bioinformatics studies global properties network computed considering enzymatic reactions links m
etabolites however pathways computed manner conserve structural moieties therefore correspond biochemical pathways traditional metabolic map work reass
essed earlier results digitizing carbon atomic traces metabolic reactions annotated escherichia coli analysis revealed average path length metabolism m
uch longer previously thought metabolic world organism small terms biosynthesis degradation'
```

Figure :4.9 Cleaned text

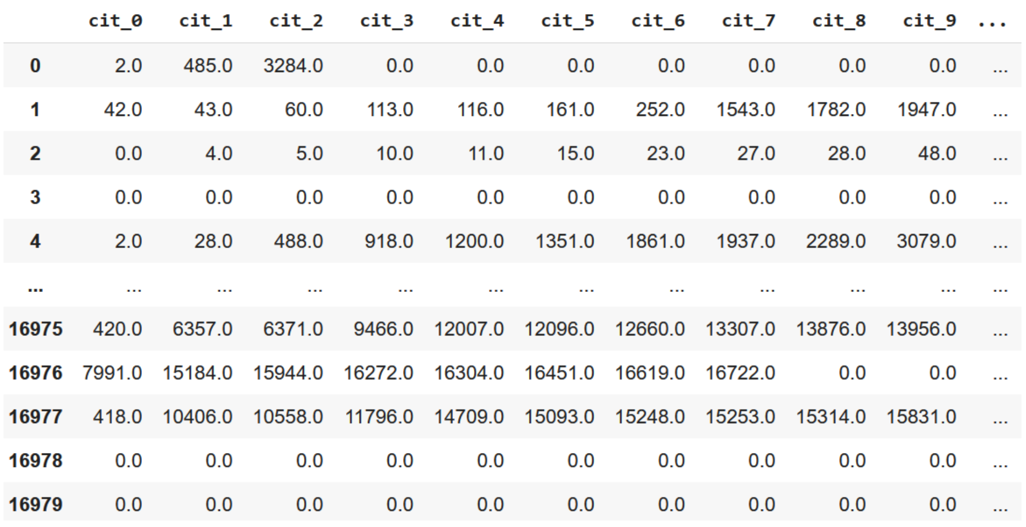
Citation data

```python
import pandas as pd
citation = pd.read_csv('/content/drive/MyDrive/citations.csv',header=None)
citation=citation.drop(citation.columns[0],axis=1)
citation=citation.fillna(0)
citation=citation.rename(columns={citation.columns[i]:'cit_'+ str(i) for i in range(0,193)})
```

The explanation of each step:

- The second line reads a CSV file called "citations.csv" located at "/content/-drive/MyDrive/" and assigns the resulting DataFrame to the variable "citation". The "header=None" parameter indicates that the CSV file doesn't have a header row, so pandas will assign default column names.
- The third line drops the first column from the DataFrame "citation". The citation.columns[0] retrieves the label of the first column (number of citations), and the drop() function is used to remove it. The axis=1 parameter specifies that the operation should be performed along the columns.
- The fourth line fills any missing values (NaN) in the "citation" DataFrame with zeros. The fillna() function is used to replace missing values with a specified value, in this case, 0.
- The last line renames the columns of the "citation" DataFrame. It uses a dictionary comprehension to iterate over the column indices (0 to 192) and assigns new column names to each column. The new column names follow the pattern "cit_0", "cit_1", "cit_2", and so on, where the index is appended to the string "cit_".

The result of executing the provided code is depicted in Figure 4.10:



	cit_0	cit_1	cit_2	cit_3	cit_4	cit_5	cit_6	cit_7	cit_8	cit_9	...
0	2.0	485.0	3284.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
1	42.0	43.0	60.0	113.0	116.0	161.0	252.0	1543.0	1782.0	1947.0	...
2	0.0	4.0	5.0	10.0	11.0	15.0	23.0	27.0	28.0	48.0	...
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
4	2.0	28.0	488.0	918.0	1200.0	1351.0	1861.0	1937.0	2289.0	3079.0	...
...	...	...	...	...	...	...	...	...	...	...	...
16975	420.0	6357.0	6371.0	9466.0	12007.0	12096.0	12660.0	13307.0	13876.0	13956.0	...
16976	7991.0	15184.0	15944.0	16272.0	16304.0	16451.0	16619.0	16722.0	0.0	0.0	...
16977	418.0	10406.0	10558.0	11796.0	14709.0	15093.0	15248.0	15253.0	15314.0	15831.0	...
16978	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...
16979	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...

Figure :4.10 Final result of citations.csv

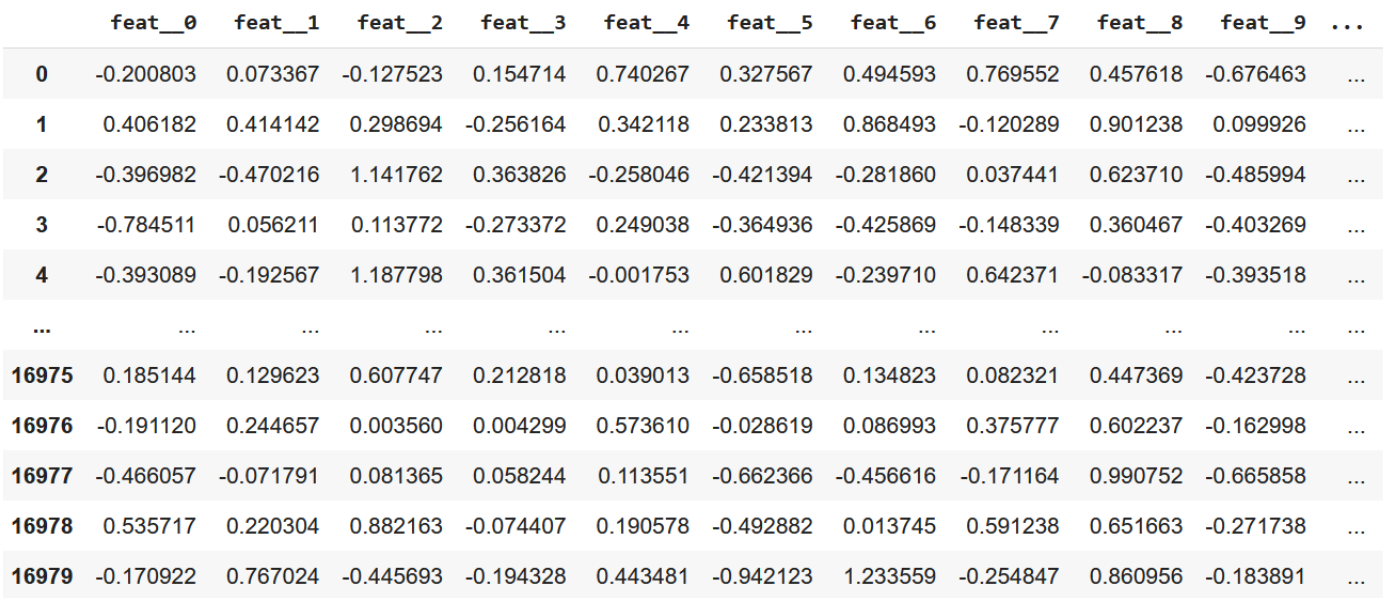
Embedding files

```
abstract=pd.read_csv('/content/drive/MyDrive/Embeddingg Abstract.csv')
abstract=abstract.rename(columns={abstract.columns[i]:'feat_'+ str(i) for i in range(0,768)
    })
title=pd.read_csv('/content/drive/MyDrive/Embeddingg Title.csv')
title=title.rename(columns={title.columns[i]:'feat__'+ str(i) for i in range(0,768)})
```

The explanation of the code above:

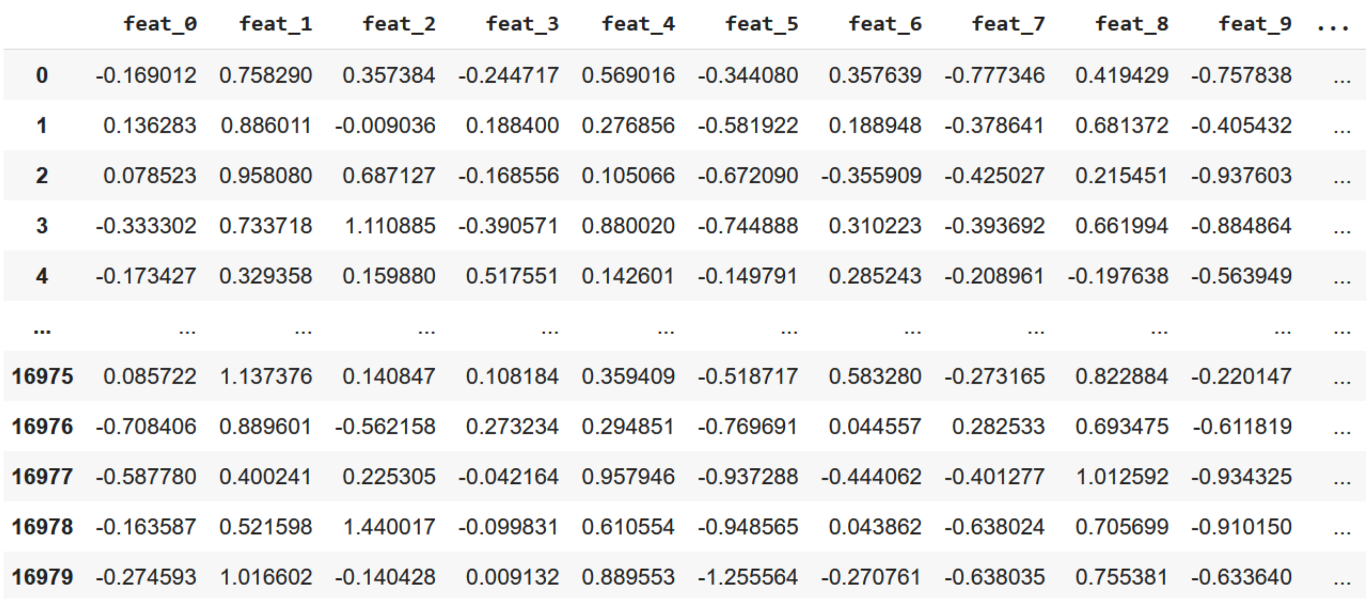
- The given code loads two CSV files, "Embeddingg Abstract.csv" and "Embeddingg Title.csv," containing abstract and title embeddings, respectively.
- In the first line, the "abstract" dataset is loaded using the pd.read_csv() function, and then the column names are renamed using a dictionary comprehension. Each column is renamed as "feat_" followed by the index number ranging from 0 to 767.
- Similarly, in the next lines, the "title" dataset is loaded and its column names are also renamed using a dictionary comprehension. Here, each column is renamed as "feat__" followed by the index number ranging from 0 to 767.

The result of executing the provided code is:



	feat__0	feat__1	feat__2	feat__3	feat__4	feat__5	feat__6	feat__7	feat__8	feat__9	...
0	-0.200803	0.073367	-0.127523	0.154714	0.740267	0.327567	0.494593	0.769552	0.457618	-0.676463	...
1	0.406182	0.414142	0.298694	-0.256164	0.342118	0.233813	0.868493	-0.120289	0.901238	0.099926	...
2	-0.396982	-0.470216	1.141762	0.363826	-0.258046	-0.421394	-0.281860	0.037441	0.623710	-0.485994	...
3	-0.784511	0.056211	0.113772	-0.273372	0.249038	-0.364936	-0.425869	-0.148339	0.360467	-0.403269	...
4	-0.393089	-0.192567	1.187798	0.361504	-0.001753	0.601829	-0.239710	0.642371	-0.083317	-0.393518	...
...	...	...	...	...	...	...	...	...	...	...	...
16975	0.185144	0.129623	0.607747	0.212818	0.039013	-0.658518	0.134823	0.082321	0.447369	-0.423728	...
16976	-0.191120	0.244657	0.003560	0.004299	0.573610	-0.028619	0.086993	0.375777	0.602237	-0.162998	...
16977	-0.466057	-0.071791	0.081365	0.058244	0.113551	-0.662366	-0.456616	-0.171164	0.990752	-0.665858	...
16978	0.535717	0.220304	0.882163	-0.074407	0.190578	-0.492882	0.013745	0.591238	0.651663	-0.271738	...
16979	-0.170922	0.767024	-0.445693	-0.194328	0.443481	-0.942123	1.233559	-0.254847	0.860956	-0.183891	...

Figure :4.11 Embeded Title



	feat_0	feat_1	feat_2	feat_3	feat_4	feat_5	feat_6	feat_7	feat_8	feat_9	...
0	-0.169012	0.758290	0.357384	-0.244717	0.569016	-0.344080	0.357639	-0.777346	0.419429	-0.757838	...
1	0.136283	0.886011	-0.009036	0.188400	0.276856	-0.581922	0.188948	-0.378641	0.681372	-0.405432	...
2	0.078523	0.958080	0.687127	-0.168556	0.105066	-0.672090	-0.355909	-0.425027	0.215451	-0.937603	...
3	-0.333302	0.733718	1.110885	-0.390571	0.880020	-0.744888	0.310223	-0.393692	0.661994	-0.884864	...
4	-0.173427	0.329358	0.159880	0.517551	0.142601	-0.149791	0.285243	-0.208961	-0.197638	-0.563949	...
...	...	...	...	...	...	...	...	...	...	...	...
16975	0.085722	1.137376	0.140847	0.108184	0.359409	-0.518717	0.583280	-0.273165	0.822884	-0.220147	...
16976	-0.708406	0.889601	-0.562158	0.273234	0.294851	-0.769691	0.044557	0.282533	0.693475	-0.611819	...
16977	-0.587780	0.400241	0.225305	-0.042164	0.957946	-0.937288	-0.444062	-0.401277	1.012592	-0.934325	...
16978	-0.163587	0.521598	1.440017	-0.099831	0.610554	-0.948565	0.043862	-0.638024	0.705699	-0.910150	...
16979	-0.274593	1.016602	-0.140428	0.009132	0.889553	-1.255564	-0.270761	-0.638035	0.755381	-0.633640	...

Figure :4.12 Embeded Abstract

4.6 Experimental Results and Discussion

This section provides a comprehensive analysis and interpretation of the results obtained, offering insights into the effectiveness, accuracy, and overall performance of the developed model.
We begin by describing the experimental setup, the evaluation metrics employed, and any preprocessing or parameter settings applied during the experiments. This ensures transparency and reproducibility of the results.
Next,we present the quantitative evaluation results obtained from the experiments and we present the qualitative evaluation findings.Furthermore, we compare our results with existing state-of-the-art citation recommendation models or approaches.This comparative analysis helps to improve the performance of our model against others in the field.The experimental results and their implications are then discussed in detail,addressing the research questions,objectives, or hypotheses stated in the earlier sections of the thesis.Finally, we identify the limitations of our study,acknowledging any potential biases,constraints,or areas for future improvement.We provide recommendations for future research directions based on the insights gained from the experimental results and discussions.By presenting and discussing the experimental results in a structured and comprehensive manner,this section aims to provide a clear understanding of the performance and effectiveness of the developed citation recommendation system,contributing to the knowledge and advancement of the field.

4.6.1 Model selection

In the process of developing our system, we carefully studied the selection of models to integrate them. The two main components of our model selection process were the retrieval model based on artificial neural network(ANN) and the incorporation of BERT model and Gated Recurrent Unit(GRU). The retrieval model served as the foundation for our recommendation system. We opted for an embedding-based retrieval model, leveraging the power of embeddings to represent the papers in a high-dimensional space. By transforming textual data into numerical representations, we could effectively capture the semantic meaning and similarities between papers. The retrieval model played a crucial role in retrieving a set of potentially relevant papers based on user queries.

4.6.2 Model training & testing

In the "Model training" subsection, we focus on the process of training our model. Once the dataset is prepared, we proceed with the model training phase. The code implementation involves setting up the model architecture, defining loss functions, selecting optimization techniques, and specifying the evaluation metrics. Through this subsection, we provide a comprehensive overview of the code required for training our model as it is explained below.

```python
columns = ['Title'] +['cit_{}'.format(i) for i in range(10)]
dataset = tf.data.Dataset.from_tensor_slices(tuple([tf.cast(df[col].values, tf.string) for
    col in columns]))

def rename(x0,x1,x2,x3,x4,x5,x6,x7,x8,x9,x10):
    y = {}
    y["Title"] = x0
    y['cit_0'] = x1
    y['cit_1'] = x2
    y['cit_2'] = x3
    y['cit_3'] = x4
    y['cit_4'] = x5
    y['cit_5'] = x6
    y['cit_6'] = x7
    y['cit_7'] = x8
    y['cit_8'] = x9
    y['cit_9'] = x10
    return y
dataset = dataset.map(rename)

Tset = dataset.map(lambda x: {
    "Title": x["Title"],
    "cit_0": x["cit_0"],
    "cit_1": x["cit_1"],
    "cit_2": x["cit_2"],
    "cit_3": x["cit_3"],
    'cit_4' : x['cit_4'],
    'cit_5' : x['cit_5'],
    'cit_6' : x['cit_6'],
    'cit_7':x['cit_7'] ,
    'cit_8':  x['cit_8'] ,
    'cit_9': x['cit_9']

})
TCset = Tset.map(lambda x: x["Title"])
```

- The first line creates a list of column names. The list starts with the string 'Title' and is followed by a list comprehension that generates column names in the format 'cit_0', 'cit_1', ... The resulting list will have 11 elements. The second line creates a tf.data.Dataset object using the from_tensor_slices method. It takes as input a tuple of tensors created from the columns of the DataFrame. In breif it consists of converting the values from specific columns of a pandas DataFrame into TensorFlow string tensors.

- The function called rename takes in multiple arguments (x0, x1, x2, ..., x10). The purpose of this function is to rename the input arguments and create a dictionary (y) with specific keys. After defining the rename function, the code proceeds to apply it to the dataset using the map method. The map function applies the rename function to each element of the dataset, transforming the elements accordingly.

- Another map function is applied to the Tset dataset. This time, a lambda function is used to extract only the values corresponding to the key "Title" from each element x in the dataset.

```
tf.random.set_seed(42)
shuffled = dataset.shuffle(1000, seed=42, reshuffle_each_iteration=False)

train = shuffled.take(10000)
test = shuffled.skip(10000).take(6980)

T_titles = df.Title.values
TC_citations0 = df.cit_0.values
TC_citations1 = df.cit_1.values
TC_citations2 = df.cit_2.values
TC_citations3 = df.cit_3.values
TC_citations4 = df.cit_4.values
TC_citations5 = df.cit_5.values
TC_citations6 = df.cit_6.values
TC_citations7 = df.cit_7.values
TC_citations8 = df.cit_8.values
TC_citations9 = df.cit_9.values
unique_T_titles = np.unique(list(T_titles))
unique_TC_citations0 = np.unique(list(TC_citations0))
unique_TC_citations1 = np.unique(list(TC_citations1))
unique_TC_citations2 = np.unique(list(TC_citations2))
unique_TC_citations3 = np.unique(list(TC_citations3))
unique_TC_citations4 = np.unique(list(TC_citations4))
unique_TC_citations5 = np.unique(list(TC_citations5))
unique_TC_citations6 = np.unique(list(TC_citations6))
unique_TC_citations7 = np.unique(list(TC_citations7))
unique_TC_citations8 = np.unique(list(TC_citations8))
unique_TC_citations9 = np.unique(list(TC_citations9))
```

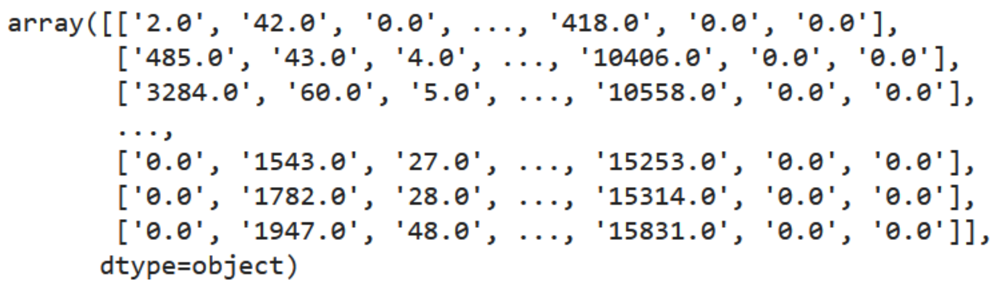
- The first line sets the random seed to ensure reproducibility.By setting the seed to a specific value (in this case, 42). The shuffle function is applied to the dataset to randomly shuffle its elements. The argument 1000 indicates that a buffer of size 1000 will be used to shuffle the elements. The reshuffle_each_iteration=False option ensures that the same order is maintained across multiple iterations if the dataset is iterated multiple times. The take function is used to create a new dataset (train) that contains the first 10000 elements from the shuffled dataset. The skip function is used to skip the first 10000 elements of the shuffled dataset. Then, the take function is applied to select the next 6980 elements. This subset represents the testing data.

- The code segment extracts the values from specific columns of a DataFrame and creates variables to store them. It also computes the unique values for each column and stores them in separate variables prefixed with "unique_".

```python
T_titles = df.Title.values
TC_citations0 = df.iloc[:,1].values
for i in range(1):
  TC_citations1 = df.iloc[:,i+2].values
  TC_citations2 = df.iloc[:,i+3].values
  TC_citations3 = df.iloc[:,i+4].values
  TC_citations4 = df.iloc[:,i+5].values
  TC_citations5 = df.iloc[:,i+6].values
  TC_citations6 = df.iloc[:,i+7].values
  TC_citations7 = df.iloc[:,i+8].values
  TC_citations8 = df.iloc[:,i+9].values
  TC_citations9= df.iloc[:,i+10].values
  TC_citations9 = [TC_citations0,TC_citations1,TC_citations2,TC_citations3,TC_citations4,
    TC_citations5,TC_citations6,TC_citations7,TC_citations8,TC_citations9]
  TC_citations = np.array(TC_citations9)
TC_citations

unique_T_titles = np.unique(list(T_titles))
unique_TC_citations = np.unique(list(TC_citations))
```

- This code segment extracts values from specific columns of a DataFrame using integer-based indexing. It retrieves the values from different columns and stores them in separate variables. Finally, it combines the variables into a list and converts the list into a NumPy array (TC_citations). The resulting array contains the extracted values from the DataFrame columns.

- The two last lines extract the unique values from the variables T_titles (representing the "Title" column values) and TC_citations (representing the "cit_" column values) by converting them into lists and applying the np.unique() function. The resulting unique values are stored in the variables unique_T_titles and unique_TC_citations, respectively.



```
array([['2.0', '42.0', '0.0', ..., '418.0', '0.0', '0.0'],
       ['485.0', '43.0', '4.0', ..., '10406.0', '0.0', '0.0'],
       ['3284.0', '60.0', '5.0', ..., '10558.0', '0.0', '0.0'],
       ...,
       ['0.0', '1543.0', '27.0', ..., '15253.0', '0.0', '0.0'],
       ['0.0', '1782.0', '28.0', ..., '15314.0', '0.0', '0.0'],
       ['0.0', '1947.0', '48.0', ..., '15831.0', '0.0', '0.0']],
      dtype=object)
```

Figure :4.13 Final format of TC_citations

Citation sub-model construction

```

embedding_dimension = 320
cit_model = tf.keras.Sequential([
  tf.keras.layers.StringLookup(
      vocabulary=unique_TC_citations, mask_token=None),
  tf.keras.layers.Embedding(len(unique_TC_citations) + 1, embedding_dimension),
  tf.keras.layers.Reshape((-1, embedding_dimension)),
  tf.keras.layers.GRU(64),
  tf.keras.layers.Dense(320, activation='relu'),  # Additional dense layer
  tf.keras.layers.Dense(320, activation='relu'),  # Additional dense layer
])
```

The explanation of code is:

- **StringLookup Layer:**This layer is responsible for mapping unique_TC _citations vocabulary . It takes each citation string as input and converts it into an integer index.
- **Embedding Layer:**This layer maps the integer indices to dense vectors of size embedding_dimension (320 in this case). It learns the representation of each citation based on the provided vocabulary.
- **Reshape Layer :**This layer reshapes the output of the embedding layer from a 3D tensor to a 2D tensor. The resulting shape will be (-1, embedding_dimension), where -1 represents the batch size and embedding_dimension is the size of each citation embedding.
- **GRU Layer:** This layer implements a Gated Recurrent Unit (GRU) with 64 units. The GRU processes the sequence of citation embeddings and captures temporal dependencies.

- **Dense Layer:**Citation sub model consists of two Dense layers (line 9 line 10).These layers has 320 units and uses the ReLU activation function. It adds an additional non-linear transformation to the GRU output.

Paper sub-model construction

```
1  embedding_dimension = 320
2  paper_model = tf.keras.Sequential([
3    tf.keras.layers.StringLookup(
4        vocabulary=unique_T_titles, mask_token=None),
5    tf.keras.layers.Embedding(len(unique_T_titles) + 1, embedding_dimension),
6    tf.keras.layers.Reshape((-1, embedding_dimension)),
7    tf.keras.layers.GRU(64),
8    tf.keras.layers.Dense(320, activation='relu'),  # Additional dense layer
9    tf.keras.layers.Dense(320, activation='relu'),  # Additional dense layer
10 ])
```

The explanation of code is:

- StringLookup Layer:This layer is responsible for mapping unique_T_titles vocabulary . It takes each title string as input and converts it into an integer index.
- **Embedding Layer:**This layer maps the integer indices to dense vectors of size embedding_dimension (320 in this case). It learns the representation of each title based on the provided vocabulary.
- **Reshape Layer :**This layer reshapes the output of the embedding layer from a 3D tensor to a 2D tensor. The resulting shape will be (-1, embedding_dimension), where -1 represents the batch size and embedding_dimension is the size of each title embedding.
- **GRU Layer:** This layer implements a Gated Recurrent Unit (GRU) with 64 units. The GRU processes the sequence of title embeddings and captures temporal dependencies.
- **Dense Layer:**Paper sub model consists of Dense layers (line 9 line 10)These layers has 320 units and uses the ReLU activation function. It adds an additional non-linear transformation to the GRU output.

BERT model Incorporation

```python
from transformers import TFBertModel
from typing import Dict, Text

bert_model = TFBertModel.from_pretrained('bert-base-uncased')

class paperModel(tfrs.Model):
  def __init__(self, cit_model, paper_model):
    super().__init__()
    self.paper_model: tf.keras.Model = paper_model
    self.cit_model: tf.keras.Model = cit_model
    self.bert_model: tf.keras.Model = bert_model
    self.task: tf.keras.layers.Layer = task
  def compute_loss(self, features: Dict[Text, tf.Tensor], training=False) -> tf.Tensor:
    cit_embeddings = self.cit_model(features["cit_0"])
    positive_paper_embeddings = self.paper_model(features["Title"])
    return self.task(cit_embeddings, positive_paper_embeddings)
```

The explanation of code is:

- The first line imports the TFBertModel class from the Transformers library. It allows you to use the pre-trained BERT model for natural language processing tasks. Then it initializes a BERT model using the pre-trained 'bert-base-uncased' model. Then we import the Dict and Text types from the typing module. These types are used to specify the input features and labels in the subsequent code.

- Then we define a custom model class called paperModel, which extends the tfrs.Model class provided by TensorFlow Recommenders. It encapsulates the logic and structure of the recommendation model. Then it is initialized by taking two parameters: cit_model and paper_model, which represent the citation model and the paper model, respectively. Then we set the paper_model as an attribute of the paperModel class. After we set the bert_model as an attribute of the paperModel class. Then we set the task as an attribute of the paperModel class. The next is to define the compute_loss method in the paperModel class. After we retrieves the embeddings of the citation features by passing the "cit_0" feature from the input features dictionary and the "Title" feature from the input features dictionary through the paper_model.

- Finally, we calculate and return the loss. It compares the citation embeddings with the positive paper embeddings to compute the loss.

In summary, the code defines a custom paperModel class that combines a citation model (cit_model), a paper model (paper_model), and a BERT model (bert_model). The model computes the loss by passing the citation and paper features through the respective models and comparing them using a retrieval task. The compute_loss method encapsulates this computation.

```python
class NoBaseClasspaperModel(tf.keras.Model):
    def __init__(self, cit_model, paper_model, task):
        super(NoBaseClasspaperModel, self).__init__()
        self.cit_model = cit_model
        self.paper_model = paper_model
        self.bert_model: tf.keras.Model = bert_model
        self.task = task
    def call(self, features: Dict[Text, tf.Tensor]) -> tf.Tensor:
        cit_embeddings = self.dropout_cit(self.cit_model(features["cit_0"]))
        positive_paper_embeddings = self.dropout_paper(self.paper_model(features["Title"]))
        return self.task(cit_embeddings, positive_paper_embeddings)

    def train_step(self, features: Dict[Text, tf.Tensor]) -> Dict[Text, tf.Tensor]:
        with tf.GradientTape() as tape:
            predictions = self.call(features)
            loss = self.task.loss_fn(features, predictions)  # Assuming a custom loss function in the task
            regularization_loss = tf.reduce_sum(self.losses)
            total_loss = loss + regularization_loss

        gradients = tape.gradient(total_loss, self.trainable_variables)
        self.optimizer.apply_gradients(zip(gradients, self.trainable_variables))

        metrics = {"loss": loss, "regularization_loss": regularization_loss, "total_loss": total_loss}
        for metric in self.metrics:
            metric.update_state(loss)
            metrics[metric.name] = metric.result()

        return metrics

    def test_step(self, features: Dict[Text, tf.Tensor]) -> Dict[Text, tf.Tensor]:
        predictions = self.call(features)
        loss = self.task.loss_fn(features, predictions)  # Assuming a custom loss function in the task
        regularization_loss = tf.reduce_sum(self.losses)
        total_loss = loss + regularization_loss

        metrics = {"loss": loss, "regularization_loss": regularization_loss, "total_loss": total_loss}
        for metric in self.metrics:
            metric.update_state(loss)
            metrics[metric.name] = metric.result()

        return metrics
```

The explanation of code is :

- The __init__ method initializes the NoBaseClasspaperModel class by assigning the provided cit_model, paper_model, bert_model, and task to respective instance variables. These variables can then be utilized within other methods of the class.

- The call method defines the forward pass of the NoBaseClasspaperModel class. It applies dropout layers to the embeddings obtained from the citation and paper models, and then passes these embeddings to the retrieval task. The method returns the output tensor computed by the retrieval task.
- The train_step method defines a single training step for the NoBaseClasspaperModel class. It computes predictions, loss, regularization loss, and gradients, applies the gradients to update the trainable variables, and computes and returns the metrics for the training step.
- The test_step method defines a single testing step for the NoBaseClasspaperModel class. It computes predictions, loss, regularization loss, and the total loss. It also updates the metrics' states and returns a dictionary containing the computed metrics for the testing step.

```
model = paperModel(cit_model, paper_model)
model.compile(optimizer=tf.keras.optimizers.RMSprop(learning_rate=0.003))

cached_train = train.shuffle(500).batch(500).cache()
cached_test = test.batch(100).cache()

history=model.fit(cached_train, epochs=10)
```

The explanation of the code above is :

- It creates an instance of the paperModel class and assigns it to the variable model. The paperModel class is defined earlier in the code, and it takes two arguments: cit_model and paper_model.Then it configures the model for training by specifying the optimizer to be used.
  In this case, the RMSprop optimizer is used with a learning rate of 0.003. The compile method prepares the model for training by setting the loss function, optimizer, and any additional metrics.

- It takes the train dataset and performs three operations on it:

  – It shuffles the examples in the dataset with a buffer size of 500, which means it randomly rearranges the examples.
  – It groups the examples into batches, with each batch containing 500 examples. This is done to facilitate more efficient processing during training.

    - It caches the dataset in memory, which means the dataset will be stored in memory for faster access during training.

- It takes the test dataset and performs two operations on it:
    - It groups the examples in the dataset into batches, with each batch containing 100 examples.
    - It caches the dataset in memory.
- Finally, the training dataset, using cached_train that we created earlier and epochs=37. In this case, the model will be trained for Thirty-Seven epochs, meaning it will go through the entire training dataset 37 times.

In summary:
The first two lines of code create an instance of the paperModel class and configure it for training using the RMSprop optimizer with a learning rate of 0.003.
The second two lines of code create cached versions of the training and test datasets, with the training dataset being shuffled, batched, and cached, and the test dataset being batched and cached. Caching the datasets in memory improves training performance by reducing the time required to load and preprocess the data for each training iteration.
By running this code, the model will be trained on the cached training dataset for 37 epochs, and the history object will store the training metrics that can be used for analysis and evaluation.

4.6.3 Model evaluation and Reporting Results

After training our model, we get to the crucial step of model evaluation. This step involves assessing the performance and effectiveness of the trained model. Through a rigorous evaluation process, we analyze the model's predictive capabilities, assess its accuracy and other relevant metrics. By examining the model's performance on test data, we gain valuable insights into its strengths, limitations, and overall effectiveness. The results and findings obtained is summarized in the table below:

Tableau :4.4 Results of training model

Epochs	Top 1	Top 5	Top 10	Top 50	Top 100	Loss
1	0.0000e+00	2.0000e-04	3.0000e-04	0.0012	0.0030	3107.3367
2	0.0050	0.0065	0.0068	0.0087	0.0118	3104.1413
3	4.0000e-04	0.0018	0.0029	0.0090	0.0159	3025.4610
4	0.0283	0.0343	0.0369	0.0526	0.0641	2802.5468
5	0.0556	0.0637	0.0703	0.0973	0.1166	2705.2627
6	0.0971	0.1074	0.1126	0.1294	0.1429	2618.6160
7	0.1170	0.1286	0.1345	0.1608	0.1792	2499.2172
8	0.1032	0.1165	0.1238	0.1539	0.1835	2356.1027
9	0.1511	0.1812	0.1957	0.2483	0.2837	2236.4857
10	0.1769	0.2055	0.2209	0.2747	0.3170	2176.7205
11	0.1887	0.2167	0.2306	0.2911	0.3390	2057.6299
12	0.1944	0.2226	0.2367	0.3033	0.3547	2041.7897
13	0.1786	0.2070	0.2223	0.3000	0.3684	1925.7477
14	0.2019	0.2304	0.2504	0.3360	0.4135	1939.2665
15	0.1985	0.2310	0.2521	0.3457	0.4394	1854.8943
16	0.2191	0.2537	0.2728	0.3733	0.4680	1803.3240
17	0.1473	0.2010	0.2242	0.3294	0.4295	1802.8569
--	--	--	--	--	--	--
32	0.2646	0.3254	0.3708	0.6219	0.7311	1288.2220
33	0.2859	0.3446	0.3939	0.6527	0.7600	1265.0565
34	0.2814	0.3492	0.4053	0.6724	0.7633	1261.7922
35	0.2351	0.3214	0.3881	0.6733	0.7489	1204.0600
36	0.2379	0.3327	0.4029	0.6728	0.7492	1172.7148
37	0.2213	0.3142	0.3874	0.6659	0.7331	1167.6898

4.6.4 Presentation of Experimental Findings

In this subsection, we present the experimental findings of our study. The goal of our experiments was to evaluate the performance and effectiveness of our proposed model in the context of citation recommendation. We conducted a series of experiments using the trained model and the CiteULike-a dataset to assess its performance and provide insights into its capabilities.

Accuracy

```
import matplotlib.pyplot as plt
plt.plot(history.history['factorized_top_k/top_1_categorical_accuracy'])
plt.plot(history.history['factorized_top_k/top_5_categorical_accuracy'])
plt.plot(history.history['factorized_top_k/top_10_categorical_accuracy'])
plt.plot(history.history['factorized_top_k/top_50_categorical_accuracy'])
plt.plot(history.history['factorized_top_k/top_100_categorical_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epochs')
plt.legend(['Top_1', 'Top_5','Top_10','Top_50','Top_100'], loc='lower right')
plt.show()
```

The results of executing this code is:

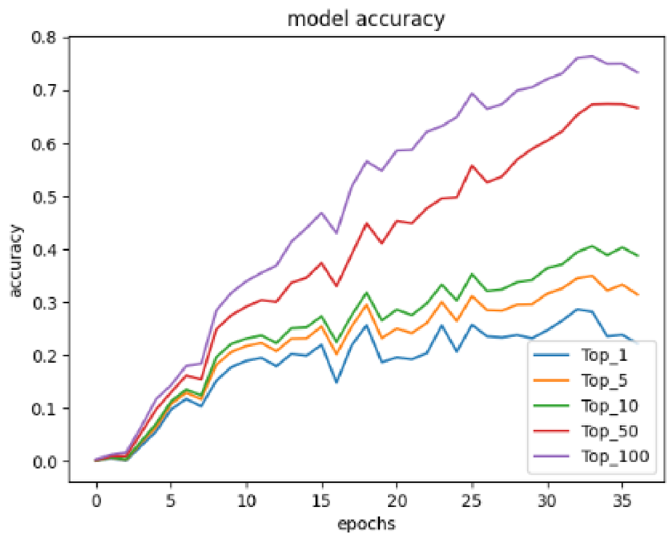


Figure :4.14 Performance of the Proposed Model Over Training Iterations

The figure 4.14 illustrates the accuracy of the model as a function of epochs, considering different values of k for the top relevant papers. It demonstrates the performance of the system in recommending highly relevant papers based on the trained model. The accuracy metric provides insights into the effectiveness of the ranking algorithm and its ability to retrieve top-k papers that align with user preferences and query relevance.

Loss

```
import matplotlib.pyplot as plt
# summarize history for accuracy
plt.title('model loss')
plt.ylabel('Lossy')
plt.xlabel('epochs')
plt.legend(['Loss'], loc='lower right')
plt.show
```

This code generates a line plot showing the values of the regularization loss and overall loss over the course of training a model. The plot helps visualize the progression of the losses and can provide insights into the model's performance and convergence.

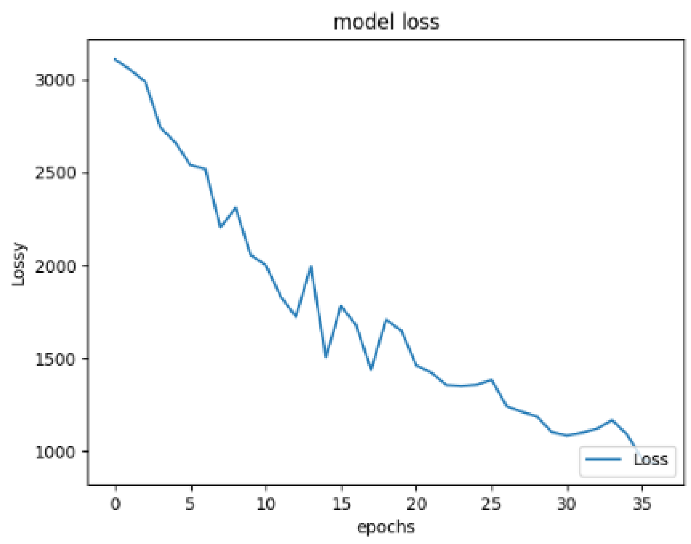


Figure :4.15 Training Loss over Epochs

The generated plot 4.15 illustrates the reduction of loss over time during the training process of our model. The horizontal axis represents the number of epochs, which refers to the number of times the model has iterated through the entire training dataset. The vertical axis represents the loss, which is a measure of the deviation between the predicted values and the actual values.

As the plot shows, the loss gradually decreases as the number of epochs increases. This indicates that the model is learning and making more accurate predictions over time. The decreasing trend of the loss indicates that the model is becoming more proficient at capturing patterns and relationships within the data.

Visualize results using TensorBoard

In the field of machine learning, assessing and enhancing the performance of models often requires the ability to quantify their progress. TensorBoard serves as a valuable instrument for obtaining the necessary measurements and visual representations throughout the machine learning process. It facilitates the monitoring of key metrics such as loss and accuracy, offers insights into the structure of the model through graph visualizations, allows for embedding projections into lower-dimensional spaces, and provides numerous additional functionalities to aid in analysis and improvement.

```
log_dir = "logs/fit/" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir, histogram_freq=1)

```

```
load_ext tensorboard
tensorboard --logdir logs/fit
```

Explanation of the code above is:

- In the first line we created a log directory path by concatenating the "logs/fit/" string with the current date and time in the format of "YYYYMMDD-HHMMSS". This ensures that each run of the training process has a unique log directory.
- The second line initializes a TensorBoard callback using the tf.keras.callbacks. TensorBoard class. It takes the log_dir parameter, which specifies the directory where the logged data will be stored. Additionally, histogram_freq=1 indicates that histograms should be computed and logged for each epoch during training.
- load_ext tensorboad permits to load the TensorBoard extension, allowing to use the tensorboard magic command to launch and display TensorBoard within the notebook interface.
- In the last line we started TensorBoard and display the visualizations within the notebook interface which allows to explore and analyze the logged data.

Plot Accuracy for each top K, Loss , BERT histogram and GRU using TensorBoard:

Accuracy for each top K

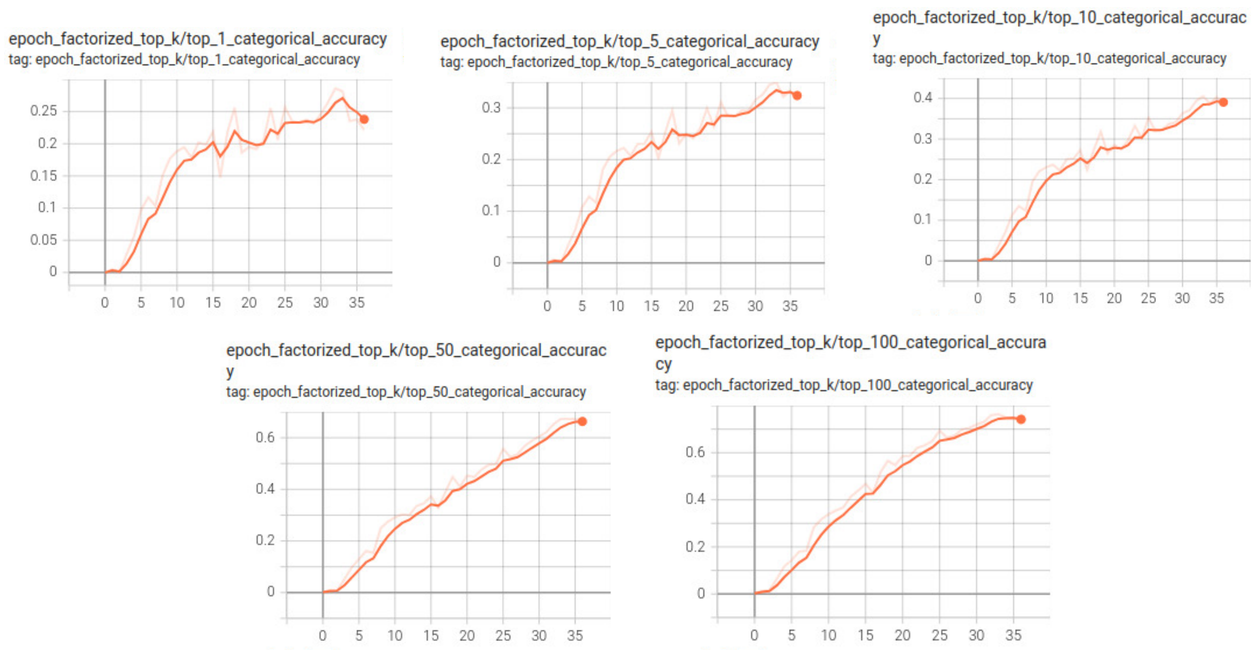


Figure :4.16 Model performance for each top K over Training epochs

Loss

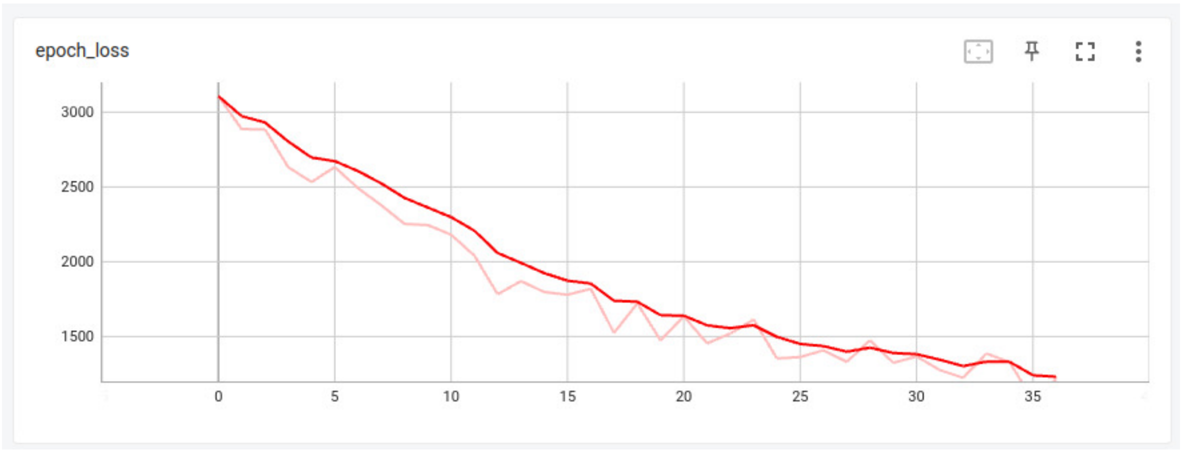


Figure :4.17 Training Loss over Epochs

BERT

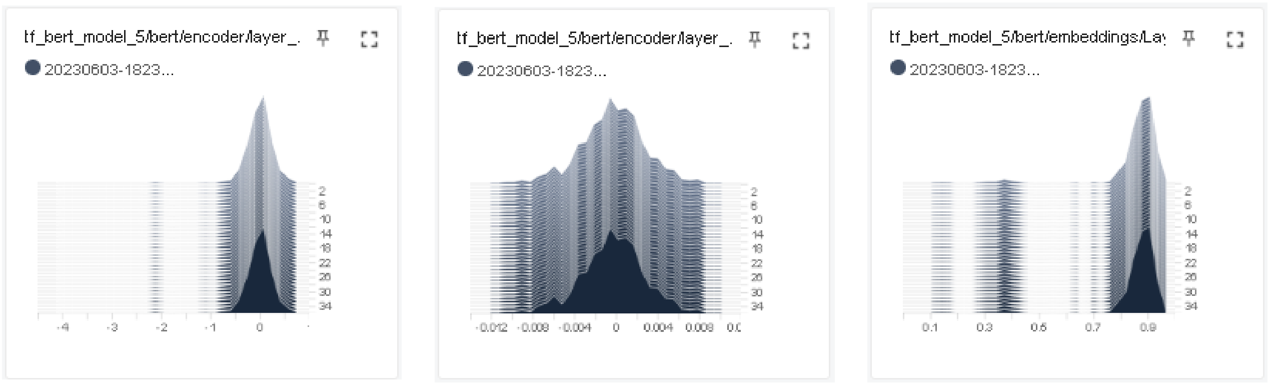


Figure :4.18 Bert performance

- First plot(Left side): The histogram of the values of the "beta_0" parameter in the Layer Normalization operation applied to the output of the attention mechanism in the first layer of the BERT model.
  The "beta_0" parameter is part of the Layer Normalization operation and represents the learned bias term. It is added to the normalized outputs to shift and scale the values, providing flexibility to the model to adapt the activations according to the task requirements
- Second plot (Middle):The histogram of the values of the "bias_0" parameter in the key component of the self-attention mechanism in the first layer of a BERT

model.The self-attention mechanism allows the model to attend to different words or positions within an input sequence. It consists of three components: key, query, and value. Each component has its associated weights and biases. The "key" component is responsible for providing the key vectors that help determine the relevance of different positions in the input sequence.
The "bias_0" parameter specifically refers to a bias term associated with the key component in the self-attention mechanism of the first layer. Bias terms are added to introduce additional flexibility to the model's computations. The bias term in this context affects the calculations performed on the key vectors during the self-attention operation.

- Third plot (Right side): The histogram of the values of the "gamma_0" parameter in the Layer Normalization operation applied to the embeddings of a BERT model.the embeddings layer is responsible for converting input tokens into continuous vector representations that capture semantic meaning.
  These embeddings are further processed and transformed by subsequent layers in the model to perform various natural language processing tasks.
  "gamma_0" specifically refers to the scaling parameter associated with the Layer Normalization operation applied to the embeddings layer. The "gamma_0" parameter is used to scale the normalized embeddings, allowing the model to control the importance or magnitude of the normalized values.

GRU(Gated Recurrent Unit)

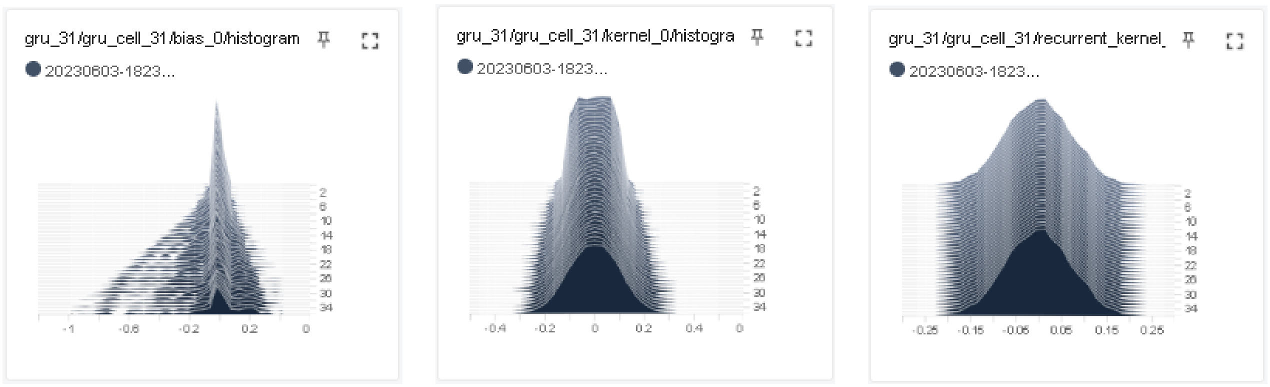


Figure :4.19 GRU layer performance

- First plot(Left side):In a GRU, there are different cells that make up the network. Each cell has various parameters, including biases, which are learnable weights that allow the network to adjust its behavior during training. These biases associated with the 64st GRU cell in the network help the GRU model to capture and process the input data effectively.The histogram of this bias parameter shows the distribution of its values. It provides insights into how the values of the bias parameter are spread across different ranges. Analyzing the histogram can help understand the impact of the bias on the GRU cell's activation and influence on the overall behavior of the neural network.

- Second plot(Middle): Represents The histogram of the weight matrix associated with the 64st GRU cell in the network. It is denoted as "kernel_0" .The histogram of this weight parameter shows the distribution of its values. It provides insights into how the weight values are spread across different ranges and can give an understanding of the importance and impact of each weight on the GRU cell's computation.

- Third plot(Right):Represents the histogram the of weight matrix associated with the recurrent connection of the 64st GRU cell in the network. It is denoted as "recurrent_kernel_0" .It provides information about the distribution of weight values.By examining this distribution, we can gain insights into the range of values and understand the significance of each weight in influencing the computation of the GRU cell.

Figures(4.16,4.17,4.18,4.19) demonstrates and proves to us that our proposed model PaperRec-BERTGRU has good performance,as a result of leveraging strengths in both the BERT and GRU architecture.In other word, our context-aware citation recommendation model,PaperRec-BERTGRU that relies on the retrieval framework has benefited from BERT's contextual understanding and the sequential modeling capabilities of GRU to capture the context of the paper and effectively recommend citations that align with the papers content thus earn correct predictions and perform better. In summary , BERT and GRU integration enhances the model's ability to understand and represent the semantic meaning of paper and citations. Thus, PaperRec-BERTGRU offers a strong solution to the context-aware citation recommendation, demonstrating its effectiveness in enhancing the search process.

4.6.5 Analysis and Interpretation of Results

To assess the effectiveness of our information retrieval approach,we employed a range of evaluation metrics,with the accuracy score being the primary measure of performance.The accuracy score represents the percentage of correctly retrieved relevant papers out of the total number of relevant papers in the dataset.
Our experiments revealed that the accuracy score varied depending on the number of top relevant papers considered in the retrieval process.By analyzing the results, we observed interesting trends and patterns.
When considering a smaller number of top relevant papers (e.g., k=5), the accuracy

score tended to be relatively low.This suggests that the retrieval system successfully identified and ranked the most relevant papers among the retrieved set.However, as the number of top relevant papers increased (e.g., k=10, 50),the accuracy score exhibited a gradual increase.
This decrease in accuracy can be attributed to various factors.As more papers are considered in the retrieval process,the system encounters a higher number of potentially relevant but less closely related papers.This introduces a greater possibility of including false positives in the retrieved set, resulting in a lower accuracy score.

4.6.6 Addressing Research Questions and Hypotheses

In this part, we aim to address the research questions and hypotheses that were formulated in the earlier sections. The primary objective of our study is to investigate the impact of information retrieval techniques on the performance of our proposed model. We have designed a series of experiments to evaluate the effectiveness of different retrieval methods in retrieving relevant papers from large scholarly databases. By analyzing the results, we can determine whether our model achieves a good accuracy and improved performance when incorporating advanced information retrieval techniques. Furthermore, we will assess the impact of various parameters, such as the number of top relevant papers considered and the choice of similarity measures, on the overall performance of the model. Through a rigorous analysis of the experimental findings, we aim to gain insights into the effectiveness and limitations of our approach, ultimately contributing to the advancement of knowledge in the field of information retrieval for scholarly papers.

4.7 Limitations and Challenges

While our system demonstrates promising results and contributes to the field of citation recommendation, it is important to acknowledge its limitations and the challenges we encountered during its development.

4.7.1 Data Limitations

The data used in this study is sourced from CiteULike, a well-known scholarly article recommendation system. While CiteULike offers valuable data for research purposes, it is important to acknowledge certain limitations associated with the dataset. These limitations include:

- **Limited Coverage:** CiteULike's dataset covers a specific range of academic disciplines and research domains. Consequently, the findings of this study may not be representative of other fields or areas of study that are not well-represented in the CiteULike-a dataset.
- **Limited Data Size:** The size of the dataset used in this study is relatively not large, with approximately 16,000 samples. While this provides a foundation for analysis, it is important to consider that a larger dataset may provide more robust and comprehensive results.

4.7.2 Challenges in Model Implementation and Execution

- **Computational Resources:** The successful execution of the recommendation system models often relies on adequate computational resources, including processing power, memory, and storage. Large-scale training and inference processes can be computationally intensive, requiring sufficient resources to achieve reasonable performance and response times.
- **Real-time Recommendations:** Deploying the recommendation system to provide real-time recommendations introduces additional challenges. Efficiently processing incoming queries, retrieving relevant papers, and generating timely recommendations require low-latency and scalable solutions. Ensuring the system can handle a high volume of concurrent requests and maintain responsiveness is critical in real-world scenarios.

4.8 Conclusion

In this chapter, we presented the implementation details of our citation recommendation system. We discussed the dataset used for training and evaluation, the selection of models, and the experimental results obtained. The results demonstrated the effectiveness of our system in providing relevant and accurate citation recommendations to users.
Through the implementation process, we have developed a robust system that leverages retrieval models and deep learning techniques to generate meaningful recommendations. The integration of various components, such as the retrieval model for candidate selection, the ranking model for relevance scoring, has contributed to the overall performance and effectiveness of the system.
Our implemented citation recommendation system demonstrates promising results in providing accurate relevant recommendations. The system's performance can be further improved by exploring advanced techniques, optimizing computational resources, and incorporating user feedback. By addressing these areas of future work, we can continue to enhance the system.

CONCLUSION

In the recent few years, there has been a remarkable increase in the quantity of scholarly publications, reflecting the expanding scope of research. As a result, researchers face an increasingly challenging and time-consuming task of staying abreast of past and current findings in their respective research areas.

To address these challenges, various approaches to paper recommendation have been proposed, as discussed in Chapter 1. However, recommending a specific number of papers from a vast collection of thousands poses a significant challenge. To tackle this issue, we developed our own recommendation model PaperRec-BERTGRU utilizing the BERT model and Gated reccurent unit(GRU) and information retrieval techniques.

Our recommender system employs an innovative and intelligent mechanism to improve recommendations and deliver scientific papers that are highly tailored to users' specific requirements. The system utilizes a content-based filtering technique, which compares the similarity between users' papers and the content of papers present in the dataset.

This system is particularly beneficial for PhD students and novice researchers, as it facilitates the discovery of the most relevant papers in their respective fields of study.

The system encompasses four primary components: Query Modelling Module, Retrieval Module, Ranking Module, and Database Module.
Python programming language and various libraries were employed in the development of our system. This system holds considerable significance for researchers seeking to save time when referencing pertinent papers for their research endeavors.
The most significant limitation of the current system is its failure to consider the user profile in the recommendation mechanism. In this case, the system lacks information about user preferences, resulting in the inability to provide relevant recommendations. Additionally, the system does not fully utilize the benefits of the collaborative-based filtering approach. To address these limitations, we plan to upgrade the system by incorporating advanced deep learning and natural language processing techniques.
Our project experience provided us with a valuable chance to apply the knowledge

we gained during our university studies and enhance our understanding of artificial intelligence techniques. It also allowed us to explore the functioning of recommender systems, along with their benefits and the challenges they pose. However, it is important to acknowledge that there is room for improvement in the resulting system, as indicated by the mentioned perspectives and measures.

BIBLIOGRAPHY

[1] Michael Färber and Adam Jatowt. Citation recommendation: approaches and datasets. *International Journal on Digital Libraries*, 21(4):375--405, 2020.

[2] Christopher James Carter Ramona Statache Svenja Adolphs Claire OMalley Tom Rodden Ansgar Koene, Elvira Perez and Derek McAuley. Ethics of personalized information filtering. *In International Conference on Internet Science*, pages 123-132, Springer,2015.

[3] Folasade Olubusola Isinkaye, Yetunde O Folajimi, and Bolande Adefowoke Ojokoh. Recommendation systems: Principles, methods and evaluation. *Egyptian informatics journal*, 16(3):261--273, 2015.

[4] Zafar Ali, Pavlos Kefalas, Khan Muhammad, Bahadar Ali, and Muhammad Imran. Deep learning in citation recommendation models survey. *Expert Systems with Applications*, 162:113790, 2020.

[5] Mohamed Anis Dhuieb, Florent Laroche, and Alain Bernard. Un compagnon virtuel d'aide à la décision: Une structuration multi-échelle de la connaissance in-extenso d'entreprise. In *21ème Congrès Français de Mécanique*, 2013.

[6] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217--253. Springer, 2010.

[7] Francesco Ricci, Lior Rokach, and Bracha Shapira. Recommender systems: introduction and challenges. *Recommender systems handbook*, pages 1-34, Springer,2015.

[8] Maya Hristakeva, Daniel Kershaw, Marco Rossetti, Petr Knoth, Benjamin Pettit, Saúl Vargas, and Kris Jack. Building recommender systems for scholarly information. In *Proceedings of the 1st workshop on scholarly web mining*, pages 25--32, 2017.

[9] Zohreh Dehghani Champiri, Seyed Reza Shahamiri, and Siti Salwah Binti Salim. A systematic review of scholar context-aware recommender systems. *Expert Systems with Applications*, 42(3):1743--1758, 2015.

[10] Afef Selmi, Zaki Brahmi, and M Gammoudi. Multi-agent recommender system: State of the art. In *Proc. of the 16th international conference on information and communications security*, 2014.

[11] Zeshan Fayyaz, Mahsa Ebrahimian, Dina Nawara, Ahmed Ibrahim, and Rasha Kashef. Recommendation systems: Algorithms, challenges, metrics, and business opportunities. *applied sciences*, 10(21):7748, 2020.

[12] Richa Sharma and Rahul Singh. Evolution of recommender systems from ancient times to modern era: a survey. *Indian Journal of Science and Technology*, 9(20), 2016.

[13] Jan Boehmer, Yumi Jung, and Rick Wash. e-commerce recommender systems. *The International Encyclopedia of Digital Communication and Society*, 9999(9999):1--8, 2015.

[14] Nunung Nurul Qomariyah. *Pairwise preferences learning for recommender systems*. PhD thesis, University of York, 2018.

[15] K. Verma. (4 july 2021) ,evolution of a recommendation engine. muvi one. retrieved may 8, 2022,from https://www.muvi.com/blogs/evolution-of-arecommendation-engine.html.

[16] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 293--296, 2010.

[17] Nazmus Sakib, Rodina Binti Ahmad, and Khalid Haruna. A collaborative approach toward scientific paper recommendation using citation context. *IEEE Access*, 8:51246--51255, 2020.

[18] Mladen Borovič, Marko Ferme, Janez Brezovnik, Sandi Majninger, Klemen Kac, and Milan Ojsteršek. Document recommendations and feedback collection analysis within the slovenian open-access infrastructure. *Information*, 11(11):497.doi:10.3390/info11110497, 2020.

[19] Dejun Mu, Lantian Guo, Xiaoyan Cai, and Fei Hao. Query-focused personalized citation recommendation with mutually reinforced ranking. *IEEE Access*, 6:3107--3119, 2017.

[20] Joseph Lilleberg, Yun Zhu, and Yanqing Zhang. Support vector machines and word2vec for text classification with semantic features. In *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 136--140. IEEE, 2015.

[21] Maha Amami, Gabriella Pasi, Fabio Stella, and Rim Faiz. An lda-based approach to scientific paper recommendation. In *Natural Language Processing and Information Systems: 21st International Conference on Applications of Natural Language to Information Systems, NLDB 2016, Salford, UK, June 22-24, 2016, Proceedings 21*, pages 200--210. Springer, 2016.

[22] Lianhuan Li, Zheng Zhang, and Shaoda Zhang. Hybrid algorithm based on content and collaborative filtering in recommendation system optimization and simulation. *Scientific Programming*, 2021:1--11, 2021.

[23] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. *Recommender systems handbook*, pages 73--105, 2011.

[24] Ryszard Stanislaw Michalski, Jaime Guillermo Carbonell, and Tom M Mitchell. *Machine learning :an artificial intelligence approach.* Springer Science & Business Media, 2013.

[25] Khalid Haruna, Maizatul Akmar Ismail, Damiasih Damiasih, Joko Sutopo, and Tutut Herawan. A collaborative approach for research paper recommender system. *PloS one*, 12(10):e0184516, 2017.

[26] Hyeyoung Ko, Suyeon Lee, Yoonseo Park, and Anna Choi. A survey of recommendation systems: recommendation models, techniques, and application fields. *Electronics*, 11(1):141, 2022.

[27] Jamilu Maaruf Musa and Xu Zhihong. Item based collaborative filtering approach in movie recommendation system using different similarity measures. In *Proceedings of the 2020 6th International Conference on Computer and Technology Applications*, pages 31--34, 2020.

[28] Guangxia Xu, Zhijing Tang, Chuang Ma, Yanbing Liu, and Mahmoud Daneshmand. A collaborative filtering recommendation algorithm based on user confidence and time context. *Journal of Electrical and Computer Engineering*, 2019,1-12.

[29] Xiaomei Bai, Mengyang Wang, Ivan Lee, Zhuo Yang, Xiangjie Kong, and Feng Xia. Scientific paper recommendation: A survey. *Ieee Access*, 1-1.doi:10.1109/access.2018.2890388.

[30] Erion Çano and Maurizio Morisio. Hybrid recommender systems: A systematic literature review. ArXiv, abs/1901.03888,2017.

[31] Lipi Shah, Hetal Gaudani, and Prem Balani. Survey on recommendation system. *International Journal of Computer Applications*, 137(7):43--49, 2016.

[32] Andreu Vall, Matthias Dorfer, Hamid Eghbal-Zadeh, Markus Schedl, Keki Burjorjee, and Gerhard Widmer. Feature-combination hybrid recommender systems for automated music playlist continuation. *User Modeling and User-Adapted Interaction*, 29:527--572, 2019.

[33] Suman Saha, Junbin Gao, and Richard Gerlach. A survey of the application of graph-based approaches in stock market analysis and prediction, . https://doi.org/10.1007/s41060-021- 00306-9. 2022.

[34] Chrsistian Desrosiers and George Karypis. Solving the sparsity problem: collaborative filtering via indirect similarities. 2008.

[35] Jia Zhou and Tiejian Luo. A novel approach to solve the sparsity problem in collaborative filtering. In *2010 International Conference on Networking, Sensing and Control (ICNSC)*, pages 165--170. IEEE, 2010.

[36] Yibo Chen, Chanle Wu, Ming Xie, and Xiaojun Guo. Solving the sparsity problem in recommender systems using association retrieval. *J. Comput.*, 6(9):1896--1902, 2011.

[37] Manos Papagelis, Ioannis Rousidis, Dimitris Plexousakis, and Elias Theoharopoulos. Incremental collaborative filtering for highly-scalable recommendation algorithms. In *Foundations of Intelligent Systems: 15th International Symposium, ISMIS 2005, Saratoga Springs, NY, USA, May 25-28, 2005. Proceedings 15*, pages 553--561. Springer, 2005.

[38] Ghazaleh Aghili, Mehdi Shajari, Shahram Khadivi, and Mohammad Amin Morid. Using genre interest of users to detect profile injection attacks in movie recommender systems. In *2011 10th International Conference on Machine Learning and Applications and Workshops*, volume 1, pages 49--52. IEEE, 2011.

[39] Yu D Deng L. Deep learning: methods and applications. *Found Trends Signal Process*, 7(3-4):197387, 2014,https://doi.org/10.1561/2000000039.

[40] Teh YW Hinton GE, Osindero S. A fast learning algorithm for deep belief nets. neural comput. *Found Trends Signal Process*, 18(7):15271554, 2006,https://doi.org/10.1561/2000000039.

[41] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, 52(1):1--38, 2019.

[42] Donghyun Kim, Chanyoung Park, Jinoh Oh, Sungyoung Lee, and Hwanjo Yu. Convolutional matrix factorization for document context-aware recommendation. In *Proceedings of the 10th ACM conference on recommender systems*, pages 233--240, 2016.

[43] Robin Devooght and Hugues Bersini. Collaborative filtering with recurrent neural networks. *arXiv preprint arXiv:1608.07400*, 2016.

[44] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. *Advances in neural information processing systems*, 26, 2013.

[45] Xiaoxuan Shen, Baolin Yi, Zhaoli Zhang, Jiangbo Shu, and Hai Liu. Automatic recommendation technology for learning resources with convolutional neural network. In *2016 international symposium on educational technology (ISET)*, pages 30--34. IEEE, 2016.

[46] Jiang Zhou, Rami Albatal, and Cathal Gurrin. Applying visual user interest profiles for recommendation and personalisation. In *MultiMedia Modeling: 22nd International Conference, MMM 2016, Miami, FL, USA, January 4-6, 2016, Proceedings, Part II 22*, pages 361--366. Springer, 2016.

[47] Chenyi Lei, Dong Liu, Weiping Li, Zheng-Jun Zha, and Houqiang Li. Comparative deep learning of hybrid representations for image recommendations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2545--2553, 2016.

[48] Hao Wu, Zhengxin Zhang, Kun Yue, Binbin Zhang, and Ruichao Zhu. Content embedding regularized matrix factorization for recommender systems. In *2017 IEEE International Congress on Big Data (BigData Congress)*, pages 209--215. IEEE, 2017.

[49] Sai Wu, Weichao Ren, Chengchao Yu, Gang Chen, Dongxiang Zhang, and Jingbo Zhu. Personal recommendation using deep recurrent neural networks in netease. In *2016 IEEE 32nd international conference on data engineering (ICDE)*, pages 1218--1229. IEEE, 2016.

[50] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[51] Yong Kiam Tan, Xinxing Xu, and Yong Liu. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st workshop on deep learning for recommender systems,Boston, MA, USA*, pages 17--22, 2016.

[52] Young-Jun Ko, Lucas Maystre, and Matthias Grossglauser. Collaborative recurrent neural networks for dynamic recommender systems. In *Asian Conference on Machine Learning*, pages 366--381. PMLR, 2016.

[53] Trivedi R Song L Dai H, Wang Y. Deep coevolutionary network: embedding user and item features for recommendation.

[54] Robin Devooght and Hugues Bersini. Long and short-term recommendations with recurrent neural networks. In *Proceedings of the 25th conference on user modeling, adaptation and personalization*, pages 13--21, 2017.

[55] Kenton Lee Jacob Devlin, Ming-Wei Chang and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *In Proceedings of NAACL*, page 41714186, 2019.

[56] Chanwoo Jeong, Sion Jang, Eunjeong Park, and Sungchul Choi. A context-aware citation recommendation model with bert and graph convolutional networks. *Scientometrics*, 124:1907--1922, 2020.

[57] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[58] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[59] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *Handheld and Ubiquitous Computing: First International Symposium, HUC99 Karlsruhe, Germany, September 27--29, 1999 Proceedings 1*, pages 304--307. Springer, 1999.

[60] Andreas Zimmermann, Andreas Lorenz, and Reinhard Oppermann. An operational definition of context. In *Modeling and Using Context: 6th International and Interdisciplinary Conference, CONTEXT 2007, Roskilde, Denmark, August 20-24, 2007. Proceedings 6*, pages 558--571. Springer, 2007.

[61] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175--186, 1994.

[62] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56--58, 1997.

[63] Robin Burke. Knowledge-based recommender systems. *Encyclopedia of library and information systems*, 69(Supplement 32):175--186, 2000.

[64] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12:331--370, 2002.

[65] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79--86, 2010.

[66] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *1994 first workshop on mobile computing systems and applications*, pages 85--90. IEEE, 1994.

[67] Peter J Brown, John D Bovey, and Xian Chen. Context-aware applications: from the laboratory to the marketplace. *IEEE personal communications*, 4(5):58--64, 1997.

[68] Hatim Guermah, Tarik Fissaa, Hatim Hafiddi, Mahmoud Nassar, and Abdelaziz Kriouile. An ontology oriented architecture for context aware services adaptation. *arXiv preprint arXiv:1404.3280*, 2014.

[69] Zafar Ali, Guilin Qi, Khan Muhammad, Bahadar Ali, and Waheed Ahmed Abro. Paper recommendation based on heterogeneous network embedding. *Knowledge-Based Systems*, 210:106438, 2020.

[70] Theodoros Vasiloudis. Extending recommendation algorithms bymodeling user context, 2014.

[71] Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, pages 22--32, 2005.

[72] Naina Yadav, Rajesh Kumar Mundotiya, Anil Kumar Singh, and Sukomal Pal. Diversity in recommendation system: A cluster based approach. In *doi:10.1007/978-3-030-49336-3_12*.

[73] Liang Zhang. The definition of novelty in recommendation system. *Journal of Engineering Science & Technology Review*, 6(3), 2013.

[74] Lukas Galke, Florian Mai, Iacopo Vagliano, and Ansgar Scherp. Multi-modal adversarial autoencoders for recommendations of citations and subject labels. In

Proceedings of the 26th conference on user modeling, adaptation and personalization, pages 197--205, 2018.

[75] Trapit Bansal, David Belanger, and Andrew McCallum. Ask the gru: Multitask learning for deep text recommendations. In *proceedings of the 10th ACM Conference on Recommender Systems*, pages 107--114, 2016.

[76] Xiaoyan Cai, Junwei Han, and Libin Yang. Generative adversarial network based heterogeneous bibliographic network representation for personalized citation recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[77] Xiaoyan Cai, Yu Zheng, Libin Yang, Tao Dai, and Lantian Guo. Bibliographic network representation based personalized citation recommendation. *IEEE Access*, 7:457--467, 2018.

[78] Zhengxiao Du, Jie Tang, and Yuhui Ding. Polar: Attention-based cnn for one-shot personalized article recommendation. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10--14, 2018, Proceedings, Part II 18*, pages 675--690. Springer, 2019.

[79] Xiangjie Kong, Mengyi Mao, Wei Wang, Jiaying Liu, and Bo Xu. Voprec: Vector representation learning of papers with text information and structural identity for recommendation. *IEEE Transactions on Emerging Topics in Computing*, 9(1):226--237, 2018.

[80] Xinyi Li, Yifan Chen, Benjamin Pettit, and Maarten De Rijke. Personalised reranking of paper recommendations using paper content and user behavior. *ACM Transactions on Information Systems (TOIS)*, 37(3):1--23, 2019.

[81] Han Tian and Hankz Hankui Zhuo. Paper2vec: Citation-context based document distributed representation for scholar recommendation. *arXiv preprint arXiv:1703.06587*, 2017.

[82] Xiaoyan Cai, Junwei Han, Wenjie Li, Renxian Zhang, Shirui Pan, and Libin Yang. A three-layered mutually reinforced model for personalized citation recommendation. *IEEE transactions on neural networks and learning systems*, 29(12):6026--6037, 2018.

[83] Xi Chen, Huan-jing Zhao, Shu Zhao, Jie Chen, and Yan-ping Zhang. Citation recommendation based on citation tendency. *Scientometrics*, 121(2):937--956, 2019.

[84] Jie Tang and Jing Zhang. A discriminative approach to topic-based citation recommendation. In *Advances in Knowledge Discovery and Data Mining: 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 Proceedings 13*, pages 572--579. Springer, 2009.

[85] Xiao Ma and Ranran Wang. Personalized scientific paper recommendation based on heterogeneous graph representation. *IEEE Access*, 7:79887--79894, 2019.doi:10.1109/access.2019.2923293.

[86] Yuta Kobayashi, Masashi Shimbo, and Yuji Matsumoto. Citation recommendation using distributed representation of discourse facets in scientific articles. In *Proceedings of the 18th ACM/IEEE on joint conference on digital libraries*, pages 243--251, 2018.

[87] Zhuoren Jiang, Yue Yin, Liangcai Gao, Yao Lu, and Xiaozhong Liu. Cross-language citation recommendation via hierarchical representation learning on heterogeneous graph. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 635--644, 2018.

[88] Shashank Gupta and Vasudeva Varma. Scientific article recommendation by using distributed representations of text and graph. In *Proceedings of the 26th international conference on world wide web companion*, pages 1267--1268, 2017.

[89] Tao Dai, Li Zhu, Yaxiong Wang, and Kathleen M Carley. Attentive stacked denoising autoencoder with bi-lstm for personalized context-aware citation recommendation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:553--568, 2019.

[90] Jie Chen, Yang Liu, Shu Zhao, and Yanping Zhang. Citation recommendation based on weighted heterogeneous information network containing semantic linking. In *2019 IEEE international conference on multimedia and expo (ICME)*, pages 31--36. IEEE, 2019.

[91] Libin Yang, Yu Zheng, Xiaoyan Cai, Hang Dai, Dejun Mu, Lantian Guo, and Tao Dai. A lstm based model for personalized context-aware citation recommendation. *IEEE access*, 6:59618--59627, 2018.

[92] Zhang-Z. Cai X. Guo Yang, L. Citation recommendation as edge prediction in heterogeneous bibliographic network: a network representation approach. *IEEE Access*, 23232-23239(7), 2019.

[93] Zhang-Y. Zeng J. Ma, X. Newly published scientific papers recommendation in heterogeneous information networks. *Mobile Networks and Applications*, 69-79(24), 2019.

[94] Betül Bulut, Esra Gündoğan, Buket Kaya, Reda Alhajj, and Mehmet Kaya. Users research interests based paper recommendation system: A deep learning approach. *Putting social media and networking data in practice for education, planning, prediction and recommendation*, pages 117--130, 2020.

[95] Travis Ebesu and Yi Fang. Neural citation network for context-aware citation recommendation. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 1093--1096, 2017.

[96] Wenyi Huang, Zhaohui Wu, Chen Liang, Prasenjit Mitra, and C Giles. A neural probabilistic model for context based citation recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[97] Anita Khadka and Petr Knoth. Using citation-context to reduce topic drifting on pure citation-based recommendation. In *Proceedings of the 12th ACM conference on recommender systems*, pages 362--366, 2018.

[98] Libin Yang, Zeqing Zhang, Xiaoyan Cai, and Tao Dai. Attention-based personalized encoder-decoder model for local citation recommendation. *Computational Intelligence and Neuroscience*, 2019, 2019.

[99] Jun Yin and Xiaoming Li. Personalized citation recommendation via convolutional neural networks. In *Web and Big Data: First International Joint Conference, APWeb-WAIM 2017, Beijing, China, July 7--9, 2017, Proceedings, Part II 1*, pages 285--293. Springer, 2017.

[100] Ye Zhang, Libin Yang, Xiaoyan Cai, and Hang Dai. A novel personalized citation recommendation approach based on gan. In *Foundations of Intelligent Systems: 24th International Symposium, ISMIS 2018, Limassol, Cyprus, October 29--31, 2018, Proceedings 24*, pages 268--278. Springer, 2018.

[101] Yang Zhang and Qiang Ma. Citation recommendations considering content and structural context embedding. In *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 1--7. IEEE, 2020.

[102] Ritu Sharma, Dinesh Gopalani, and Yogesh Meena. Concept-based approach for research paper recommendation. In *Pattern Recognition and Machine Intelligence: 7th International Conference, PReMI 2017, Kolkata, India, December 5-8, 2017, Proceedings 7*, pages 687--692. Springer, 2017.

[103] Zeynep Batmaz, Ali Yurekli, Alper Bilge, and Cihan Kaleli. A review on deep learning for recommender systems: challenges and remedies. *Artificial Intelligence Review*, 52:1--37, 2019.

[104] Michael Irwin Jordan, Terrence Joseph Sejnowski, and Tomaso A Poggio. *Learning and relearning in boltzmann machines*. 2001.

[105] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855--864, 2016.

[106] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *In Proceedings of the 26th international conference on neural information processing systems*, vol,2:3111--3119. Red Hook, NY, USA., 2013.

[107] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234:11--26, 2017.

[108] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[109] Ruihui Mu. A survey of recommender systems based on deep learning. *Ieee Access*, 6:69009--69022, 2018.

[110] Chandra Bhagavatula, Sergey Feldman, Russell Power, and Waleed Ammar. Content-based citation recommendation. *arXiv preprint arXiv:1802.08301*, 2018.

[111] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135--146, 2017.

[112] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067--1077, 2015.

[113] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *Transactions on knowledge and data engineering(TKDE)*, (5):833--852, 2019.

[114] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *In Proceedings of the 31st international conference on machine learning*, volume 32, pages 1188--1196, 2014.

[115] Giannis Christoforidis, Pavlos Kefalas, Apostolos Papadopoulos, and Yannis Manolopoulos. Recommendation of points-of-interest using graph embeddings. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 31--40. IEEE, 2018.

[116] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701--710, 2014.

[117] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S Yu Philip. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 31(2):357--370, 2018.

[118] Waheed Ahmed Abro, Guilin Qi, Huan Gao, Muhammad Asif Khan, and Zafar Ali. Multi-turn intent determination for goal-oriented dialogue systems. In *2019 international joint conference on neural networks (IJCNN)*, pages 1--8. IEEE, 2019.

[119] Hebatallah A Mohamed Hassan. Personalized research paper recommendation using deep learning. In *Proceedings of the 25th conference on user modeling, adaptation and personalization*, pages 327--330, 2017.

[120] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *in advances in neural information processing systems*, pages 2672--2680, 2014.

[121] Michael Färber, Alexander Thiemann, and Adam Jatowt. To cite, or not to cite? detecting citation contexts in text. In *Advances in Information Retrieval: 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings 40*, pages 598--603. Springer, 2018.

[122] Michael Färber, Alexander Thiemann, and Adam Jatowt. Citewerts: A system combining cite-worthiness with citation recommendation. In *Advances in Information Retrieval: 40th European Conference on IR Research, ECIR 2018, Grenoble, France, March 26-29, 2018, Proceedings 40*, pages 815--819. Springer, 2018.