GRADUATE PROGRAMS INSTITUTE

USE OF DEEP LEARNING FOR RESEARCH PAPER RECOMMENDATION

Doniazad BEN SAYAH

MASTER'S THESIS

İstanbul, November 2023

USE OF DEEP LEARNING FOR RESEARCH PAPER RECOMMENDATION

Doniazad BEN SAYAH

MASTER'S THESIS

COMPUTER ENGINEERING DEPARTMENT

COMPUTER ENGINEERING MASTER'S PROGRAM WITH THESIS

MASTER'S THESIS ADVISOR

Assit. Prof. Özlem Feyza ERKAN

MASTER'S THESIS JURY MEMBERS

Assit. Prof. Burçin Kulahcioğlu

Assoc. Prof. Onur Cihan

PREFACE

I would like to express my gratitude and appreciation to everyone who has contributed to the successful completion of my dissertation. Firstly, I am immensely thankful to my advisor, Asst. Prof. Ö. Feyza Erkan, for her invaluable help, guidance, and mentorship throughout my graduate journey. Her support has been instrumental in enabling me to undertake this research and communicate its results effectively.

I am also deeply grateful to all the members of Beykoz University cannot miss the chance to express my heartfelt love and appreciation to my treasured family especially, you mean everything to me, and I want you to know that I have finally completed my master studies.

Inshallah, I look forward to dedicating more time to you in the future. Thank you for your unwavering love and support.

CONTENTS

ABSTRACT

USE OF DEEP LEARNING FOR RESEARCH PAPER RECOMMENDATION

In the field of scientific writing, it is crucial to be able to identify and choose relevant publications. However, due to the increasing number of papers being published, researchers face difficulties in manually navigating through the vast amount of literature. In response to the challenge of navigating the growing volume of academic literature, automated paper recommendation systems have emerged to assist researchers in finding relevant publications. These systems also address the need to establish connections between research outcomes and academic journal subjects for article submissions. Traditional text representation models have limitations in capturing semantic relationships between words and addressing the diverse range of topics across different journals, which affects classification and recommendation results.

In this study, a dataset sourced from Kaggle, comprising titles and abstracts of research articles, has been expanded to a total of 60,286 instances. This expansion includes 40,000 additional instances from the National Library of Medicine alongside the original 20,286 instances. These instances are categorized into four classes: physics, mathematics, statistics, and computer science. The research employs various feature extraction methods, including doc2vec, TF-IDF, and Bag of Words, and evaluates the performance of classical machine learning algorithms and deep learning models on the feature vectors. The resulting model functions as a recommendation system, offering content-based suggestions for authors seeking papers relevant to their research interests.

Keywords: content-based recommender systems, research paper recommendation, feature extraction, machine learning

Date: 1.11.2023

ÖZET

ARAŞTIRMA MAKALESİ ÖNERİSİ İÇİN DERİN ÖĞRENMENİN KULLANIMI

Bilimsel yazının alanında, ilgili yayınları tanımlayıp seçebilme yeteneği oldukça önemlidir. Ancak, giderek artan sayıda makalenin yayımlandığı bir ortamda, araştırmacılar, geniş bir bilimsel literatürü el ile gezme konusunda zorluklarla karşılaşırlar. Büyüyen akademik literatürün içinde gezinme sorununa cevap olarak, ilgili yayınları bulmada araştırmacılara yardımcı olmayı amaçlayan otomatik makale öneri sistemleri ortaya çıkmıştır. Bu sistemler aynı zamanda araştırma sonuçları ile akademik dergi konuları arasında bağlantı kurma ihtiyacını da ele almaktadır. Geleneksel metin temsil modelleri, kelimeler arasındaki anlamsal ilişkileri yakalama ve farklı dergilerin kapsadığı geniş konu yelpazesini ele alma konularında sınırlamalar taşımaktadır, bu da sınıflandırma ve öneri sonuçlarını etkilemektedir. Bu çalışmada, araştırma makalelerinin başlıklarını ve özetlerini içeren Kaggle veritabanından alınan bir veri kümesi kullanılmış ve bu veri kümesi toplamda 60.286 örneğe genişletilmiştir. Bu genişleme, orijinal 20.286 örneğin yanı sıra Ulusal Tıp Kütüphanesi'nden gelen 40.000 ek örneği içermektedir. Bu örnekler fizik, matematik, istatistik ve bilgisayar bilimleri olmak üzere dört farklı sınıfa ayrılmıştır. Araştırma, doc2vec, TF-IDF ve Kelime Çantası gibi çeşitli özellik çıkarma yöntemlerini kullanmakta ve elde edilen özellik vektörleri üzerinde klasik makine öğrenme algoritmaları ve derin öğrenme modellerinin performansını değerlendirmektedir. Sonuçta ortaya çıkan model, araştırma ilgi alanlarına uygun makaleler arayan yazarlar için içerik tabanlı öneriler sunmaktadır.

Anahtar Kelimeler: içerik tabanlı öneri sistemleri; araştırma makalesi önerisi; özellik çıkarımı; makine öğrenimi.

Tarih: 1.11.2023

ABBREVIATIONS

RS : Recommendation System
TF- IDF : Term Frequency-Inverse Document
SVM : Support Vector Machines
SVC : Support Vector Classifier
RF : Random Forest
LSTM : Long Short-Term Memory

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

Recommender Systems (RS) have become increasingly common in helping consumers make decisions and sort through extensive information and product spaces, boosting their engagement and happiness with online services (Jawaheer, Weller, Kostkova, 2014).In a study done by Isinkaye, Folajimi, & Ojokoh, 2015, RS usually concentrates on two tasks: (1) predict task, which inquires as to what the user is likely to choose for an item given a user and the item, and (2) recommendation task, which inquires as to the best-ranked list of n-items for the user's requirements.

Researchers in the fields of information retrieval, machine learning, and human computer interaction grew interested in RS (Ekstrand, Riedl, & Konstan, 2011). Numerous studies have been conducted in this field, primarily aimed at developing new algorithms for recommendations (Shani, & Gunawardana, 2011). Systems that make recommendations have been studied and utilized across various internet services. E-commerce is one of the most well-liked industries, where recommendation algorithms are employed to make product suggestions to customers based on their purchasing patterns, browsing habits, and demographics. This aid online merchants in boosting revenues and enhancing user experience. The entertainment industry is another one that frequently uses recommendation algorithms. According to users' viewing preferences or ratings, recommendation systems in this field make movies, songs, or TV series recommendations to them. This aids customized content delivery and user retention for streaming services like Netflix and Spotify.

Another area where recommendation algorithms are commonly employed is social media. Based on users' interests and prior behavior, recommendation systems make recommendations for users to follow and content to interact with. This enhances user engagement and retention on social media platforms like Facebook and Instagram. Another industry where recommendation systems are becoming more prominent is travel. In this area, recommendation algorithms make suggestions to users on locations, lodgings, and activities based on their past travel experiences, preferences, and spending capacity which aids in the user experience improvement and revenue growth of the travel industry.

Last but not least, recommendation systems are also utilized in education to suggest programs, books, research papers, and other materials (Wang, & Blei, 2011). Providing additional resources based on their academic background and intereststo scholars. This contributes in their academic success and the advancement of research outcomesat their schools. Deep Learning architectures and techniques, which have already promise in other difficult disciplines like computer vision, natural language processing (NLP), machine translation, and speech recognition, have seen a strong growth in RS research from Osindero, & Teh, 2006.

1.1 MOTIVATION

We encounter several issues while building a recommender system from scratch. Whatshould we do if the system does not have enough papers or articles? The problem leads to the cold start problem, which arises when a recommendation system cannot provideprecise recommendations for new users or goods with limited or no previous data, is one of the most frequent problems.

Previous research indicates that collaborative filtering and content-based filtering consistently emerge as common approaches in recommender system methodologies. While content-based recommender systems provide suggestions comparable to those the identicaluser has expressed interest in during earlier encounters with the system, collaborative techniques make recommendations based on what other users who share similar preferences have previously liked. As a result of matching an item's attributes with a user's preference profile, content-based approaches can recommend a product to them.

The cold start issue is a weakness shared by different types of recommender systems. Collaborative algorithms need accurate historical item ratings to make the best recommendations. However, in such areas, brand-new objects exist without any prior ratings, making collaborative techniques ineffective. Some types of cold start problem appears when we are going to recommend new products (Adomavicius & Tuzhilin, 2005). Content-based recommender systems have helped to some extent to mitigate thisproblem which involves pointing consumers in the direction of related products based on the properties of the item, such as title, keywords, abstract, or tags. This method works effectively for brand-new products that users still need to give a

rating to the representation of an article will then be solved so that a system can comprehend its area. The prerequisitefor assessing the degree of similarity between the two articles is that article attributes likearea and author are a technique to classify articles (Lam, Konstan, & Riedl 2003). However, each aspect of the article should be given a distinct amount of weight because each one affects the recommendation differently way. So, we have these inquiries:

- The features of articles that the recommender system can use
- Methods for determining similarity between two articles.

1.2 OBJECTIVE

The objective of this thesis project is to investigate recommender systems to utilize. There are many different types of RS, butnot all of them are appropriate for a given issue or circumstance. Our goal is to discovera fresh approach to enhance article classification based on the title, author name, which serves as a prerequisite for enhancing content-based recommender systems. Additionally, a methodological study for the experimental approach, setup, and metrics should be performed in such a waythat appropriate baseline methods and experimental designs are used.

1.3 OUTLINE OF THE THESIS

This thesis is organized as follows. Subsequent to the introduction in Chapter 1, Chapter 2 explores a literature survey focusing on recommender systems. In the first part of Chapter 2, we explain the concept of the recommendation system. The second part is devoted to the discussion and comparison among different types of those systems such as collaborative, content-based, and hybrid approaches. In Chapter 3, the research methodology starts by presenting our dataset and explaining different techniques used in data prepossessing. We also introduce feature extraction methods like TF-IDF, BoW and Word2Vec then we utilize different machine learning and deep learning models. The selection of most suitable method for creating a recommender system is given in Chapter 3 as well. The results will be presented in Chapter 4. The final chapter concludes the thesis and provides some possible directions for further research.

2 LITERATURE SURVEY

The rapidly expanding landscape of publications has created an abundance of potentially relevant papers (Alzoghbi, Arrascue Ayala, Fischer, & Lausen, 2015) that serve various specific purposes, including finding related papers (Collins, & Beel, 2019), selecting materials for in-depth reading (Wang, Weng, & Wang, 2021) and conducting comprehensive literature searches to gain fresh insights and understand cutting-edge methodologies (Kang, Hou, Zhao, & Gan, 2021). Researchers dedicate considerable time to the task of seeking out relevant work, often relying on keyword-based search techniques, which demand a certain level of subject-specific knowledge (Bai, Wang, Lee, Yang, 2019; Lee, Lee, & Kim, 2013). This endeavor is further complicated by the frequent lack of explicit articulation of users' information requirements (Li, Chen, Pettit, &Rijke, 2019).

In the following sections, we explore the domain of recommender systems. Given that the primary goal of this research is to enhance the performance of recommender algorithms, it is crucial to comprehend the most widely used recommender systems, their approaches, benefits, drawbacks, and techniques for assessing prediction accuracy.

2.1 RECOMMENDER SYSTEMS

Recommendation systems (RS) refer to software tools and methods designed to offer users with content tailored to their preferences and needs (Burke, 2007; Mahmood & Ricci, 2009). These recommendations can pertain to a variety of contexts, from suggesting articles to read, friends to add to a social network, or books to purchase. As (Cacheda, Carneiro, Fernández, and Formoso , 2011) highlight, RSs generate personalized recommendations based on user profiles and item characteristics.

In today's era of information abundance, RSs have become vital tools for aiding users in making informed choices by offering novel and relevant solutions based on their immediate needs. These systems rely on vast datasets, encompassing user preferences, item catalogs, and historical interactions, all stored in dedicated databases. User feedback, both implicit and explicit, plays a crucial role in enhancing the quality of future recommendations (Ricci, et al., 2011). The term "item" is commonly used to refer to what is recommended to a user in the context of RSs. To

create effective RSs across various domains, understanding the characteristics of both "items" and "users" is essential. These systems find practical applications inwebsites like Amazon.com where the online store is customized for each customer (Hwang, Kuo, & Yu, 2008). Personalized recommendation systems provide tailored suggestions based on individual interests and preferences, while non-personalized systems are simpler in nature, often used by online periodicals and newspapers to recommend popular items to all users. The research efforts focus typically on gravitates toward personalized recommendations.

In personalized RSs, suggested items are presented in a ranked list, reflecting user tastes, choices, and constraints. To execute this process, RSs create user profiles, a critical step that can be achieved either overtly (such as user ratings) or covertly (like studying user interactions). This user profile-building process is fundamental to the success of RSs (Adomavicius and Tuzhilin, 2005).

The roots of recommender systems can be traced back to various fields, including information retrieval, cognitive science, forecasting theories, approximation theory, marketing, and management. However, the distinct field of recommender systems emerged in the mid-1990s, coinciding with the growing popularity of user profiles as a novel approach to generating recommendations (Goldberg, Nichols, Oki, & Terry, 1992; McSherry & Mironov, 2009).

Based on current knowledge, comprehensive literature reviews by (Rohani, & Ow, 2012), (Bai, et al, 2019) are the most thorough, particularly concentrate on on recommendation algorithms forscholarly papers. In recent years, recommender systems have gained significant attention in academia and education, with universities offering RS-specific courses, and conferences featuring workshops and tutorials on RS topics. Several books addressing RS methodologies have also been published (Jannach et al., 2010; Tiroshi, Kuflik, Kay, & Kummerfeld, 2012)

2.1.1 RS Techniques

Adomavicius and Tuzhilin (2005) classified recommender systems into three primarygroups after a review of prior work:

1. Content-based recommendations: These systems consider users' preferences frompast interactions when predicting new items.
2. Collaborative recommendations: Items are predicted based on the preferences ofusers with similar tastes and interests.
3. Hybrid approaches: These methods blend elements from both collaborative andcontent-based approaches.

A more recent classification, proposed by Ricci, et al. (2011), introduced three additional types of recommender systems were introduced:

I. Demographic suggestions: Recommendations are tailored based on various demographic profiles.
II. Knowledge-based recommendations: Recommendations are generated according tospecific knowledge domains.
III. Social network-based recommendations: User preferences and friends'recommendations play a significant role in this type of RS (Zhi, & Zou, 2019).

Collaborative Filtering

The initial Collaborative Filtering (CF) approach, as presented by Schafer et al. in 2001, recommends items to active users based on the preferences of similar users. The similarity between the prior ratings of two users is crucial in determining the similarity of their tastes, a concept Schafer et al. referred to as "people-to-people correlation." The CF method relies more on user feedback than on item characteristics. It takes into account the user's perspective and preferences when making recommendations, emphasizing the community aspect. CF algorithms are typically categorized into memory-based (user-based) and model-based (item-based) approaches (Cacheda et al., 2011).

Memory-based approaches use data from all users, items, and their ratings to make recommendations. These methods require collecting and storing all available data in

memory. Model-based approaches, on the other hand, search for related items rather than grouping similar users, addressing certain limitations of memory-based methods. The keydistinction lies in their use of items rather than users to predict ratings. Both approacheshave their advantages and disadvantages (Wang, De Vries, & Reinders, 2006).

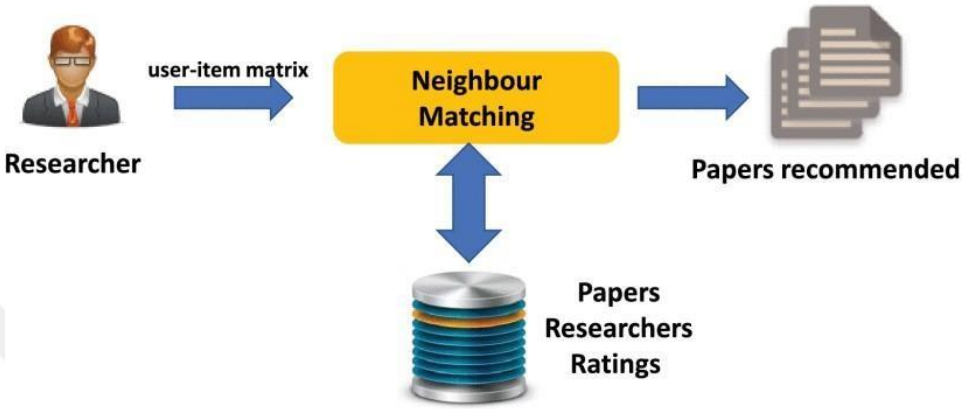

Figure 2.1 Collaborative Filtering System for Paper Recommendation

Model-based collaborative filtering techniques offer several advantages over memory-based approaches. They provide more comprehensive explanations for the recommendations, require less memory, and facilitate faster recommendation generation. They excel in scenarios where user profiles change gradually, rendering them more suitable for real-worlduse (Deshpande et al., 2004; Linden et al., 2003; Sarwar et al., 2001). Figure 2.1 depicts the general organization of collaborative filtering systems. (Barolli, Di Cicco, & Fonisto, 2022).

Content-Based Filtering

Content-Based Filtering has been a long-standing technology for generating personalized recommendations based on individual user interests, with a history spanning three decades (Herlocker, 2000). These systems create user models to represent preferences and interests by analyzing previously rated items (Mladenic, 1999). Content Based RSs aim to match these user preferences, obtained from a user profile, against item attributes to determine the user's level of interest in specific items. This approach has found applications in recommending various content, including websites, news articles, restaurants, television shows, and products. Although the specifics may differ, content-based recommendation systems typically involve

describing recommended items, constructing user profiles, and comparing items to these profiles for recommendations.

Architecture of Content-Based RSs

A high-level architecture, as suggested by Lops, 2011 comprises the primary components:

Content Analyzer: This component is responsible for extracting relevant information from items in a format suitable for further processing. It prepares analyzed data items for the Profile Learner and Filtering Component.

Profile Learner: The Profile Learner component is responsible for building user profiles. It typically employs machine learning techniques to generalize userpreferences and interests. For instance, in a web page recommender system, this component creates models by combining vectors of both positive and negative web page samples.

Filtering Component: This component compares user profile preferences with item attributes to suggest the most relevant items. This process results in the creationof a ranked list of items anticipated to be interesting for the user.

To offer further more clarity, the Content Analyzer utilizes information retrieval techniques to analyze item attributes obtained from various sources. This transforms initially unstructured data into a structured representation of items (Figure 2.2).

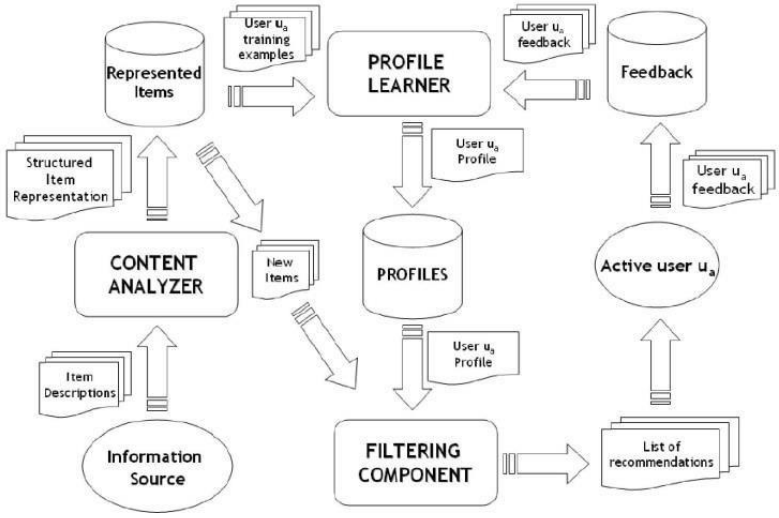

Figure 2.2 Content-Based Recommender High Level Architecture

Feedback Collection

A key consideration in Content-Based RSs is the collection of user feedback, which categorized into twotypes as explicit and implicit. In explicit feedback, users actively rate items using strategies like "LIKE/DISLIKE" (binary classification), numerical ratings, or text comments. Implicit feedback, on the other hand, is inferred from user interactions with the system over time. Explicit feedback can be obtained through various strategies, each with its distinct approach, such as binary classifications, numerical ratings, or text comments. For instance, Amazon.com and eBay.com text comments to gauge whether an item was influenced by previous purchases.

Hybrid Approach

In most cases, content-based Recommendation Systems RSs do not compete with CFapproaches. Alternately, they can both work together to create more potent hybrid treatment. Movie Lens (Jung, 2011), Video Recommender (Verhoeyen, Vriendt, & Vleeschauwer, 2012), and Group Lens (Miller, Riedl, & Konstan, 2003) are just a few ofthe successful studies that have been produced in this area. The advantages of recommending technique A are used to make up for technique B's drawbacks in hybrid recommender systems, which combine the two. As an illustration, collaborative filtering techniques that experience new-item issues are unable to determine the forecast for newitems in the absence of any rating records. In order to address this weakness and achieve better outcomes, content-based and collaborative methodologies might be combined (Ricci, et al., 2011).

The following categories can be used to categorize a hybrid method that combines content-based and collaborative recommender systems:

1. Putting collaborative and content-based recommender systems into place independently before integrating their recommendations. Incorporating some qualities depending on content into a collaborative approach.
2. Integrating some collaborative elements into a content-based approach.
3. Using a combination of content-based and collaborative methods to create an all-encompassing model.

Comparison

Each recommendation strategy, whether content-based or collaborative, has its own metrics and limitations. The effectiveness of these approaches varies depending on the dataset and the specific problem they are applied to. Content-based filtering may face challenges when extracting features from unstructured text data, while collaborative filtering might encounter difficulties with cold start problems when dealing with new users or items. Table 2.1 presentes detailed comparison (Adomavicius, & Tuzhilin, 2005).

Decision Trees and Rule Induction

To create a decision tree, decision tree learners like ID3 (Quinlan, 1986) iteratively divide training data, such as text documents, into subgroups until each subgroup only contains instances of a single class. A popular criterion for choosing the most informative features for the partition tests is expected information gain (Yang & Pedersen, 1997). A decision tree can simply depict and learn a profile of someone who prefers to eat at pricey French restaurants or low-cost Mexican restaurants. Decision trees have been extensively studied in use with structured data. The decision tree bias may not be optimal option forjobs requiring the classification of unstructured text (Pazzani & Billsus, 1997). Decisiontree inductive bias favors tiny trees with few tests as a result of the information-theoreticsplitting criteria utilized by decision tree learners. However, experimental evidence suggests that tasks for classifying text usually entail a significant number of pertinent elements (Joachims, 1998). Therefore, a decision tree's propensity to base classifications on the fewest number of tests possible may lead to sub optimal results in text categorization.

Nearest Neighbor Methods

The nearest-neighbor approach returns all features of the raining data in memory. This method finds the "nearest neighbor" or the k nearest neighbors by comparing a new item to all previously stored items using a similarity function. The class labels of the new item's closest neighbors can then be used to determine its class label ornumerical score. The nearest neighbor approach uses a similarity function that is dependent on the type of data. Typically, a Euclidean distance metric is employed whendealing with organized data. The cosine similarity metric is frequently applied

when employing the vector space model (Salton, 1989b). If related attributes of two samples have tiny values, the cosine similarity function won't have a significant value. Conversely, the Euclidean distance function treats a characteristic equaly whether it has a small or a large value in two samples

As a result, when two documents are about the same subject, it is accptable to use language to make them similar, not necessary cover not when they are not. Several text classification applications have used the cosine similarity function and the vector space method (Allan et al., 1998; Cohen & Hirsh, 1998; Yag, 1999). In order to build a model of the user's short-term interests, the Daily Learner system uses the closest neighbor method (Billsuset al., 2000).

Through this chapter we explored various advantages and limitations of recommendation systems and discussed about several additions that might be made in order to improve suggestion capabilities. In Table 2.1, a general overview of recommender systems is presented. The most popular methods and relevant research projects are given for each category to provide more insight. (Table 2.2, Rohani, & Ow, 2012).

Table 2.1 Comparison of Contented-Based and Collaborative Filtering

Recommendation algorithms	Advantages	Disadvantages
Content based	Recommendation result is intuitive and easy to interpret; No need for users? Access history data; No new item problem and no sparity problem; Supported by the mature technology of classification learning.	Limited by the features extraction models; New user problem; The training of classifier needs massive data; Poor scalability.
Collaboration filtering	No need for professional knowledge; Performance improving as the increasing of the user number; Automatic; Easy to find user? Is new interesting point; Complex unstructured item can be processed., music, video, etc.	Sparity problem; Poor scalability; New user and new item problem; The recommendation quality limited by the history data

Table 2.2 Classification of Recommender Systems Research

Recommendation Approach	**Recommendation Technique**
Collaborative	Commonly used Techniques • Nearest neighbor (cosine, correlation) • Memory-based (used based) • Model-based(item based) • Clustering • Graph theory • Bayesian networks • Artificial neural network • Linear regression • Probabilistic models Representative research examples • Resnich et al.(1994)
Content-Based	Commonly used Techniques • TF-IDF (information retrieval) • Clustering • Bayesian classifiers • Decision trees • Artificial neural networks
Hybrid	Commonly used Techniques • Linear combination of predicted ratings • Various voting schemas • Building one unifying model

3 METHODOLOGY

3.1 APPROACH

A content-based recommendation system can be tailored for academic collaboration in the field of research paper recommendations. This system aims to provide authors suggestions based on the characteristics and qualities of research articles. The first crucial step in this process involves gathering a comprehensive collection of research papers, followed by data cleaning and applying text normalization algorithms to ensure consistency and quality. Afterward, relevant features are extracted from the papers, employing various methods such as TF-IDF representation, bag of words, and word embeddings like Word2Vec. after the data is pre-processed and the features are extracted, it is time to train and evaluate different machine learning models, including deep learning models like neural networks and matrix factorization techniques. These models are subjected to proper evaluation metrics to assess their performance and accuracy. Through this rigorous evaluation, the model that demonstrates the highest accuracy is selected as the most suitable for the recommendation system (Pipeline Machine learning).

When a new paper is presented, it goes through the pipeline of the machine learning based recommendation system. This pipeline includes various steps such as datapreprocessing, feature extraction, and model training. Once the new paper undergo these steps and the necessary features have been extracted, the recommendation system utilizes the k-nearest Neighbors (KNN) algorithm. The KNN algorithm is applied to identify the most similar papers to the given author's profile or the specific paper of interest. By calculating the similarity between the embeddings or feature representations of the new paper and the existing papers in the dataset, the KNN algorithm identifies the'n' most similar papers. The value of 'n' can be adjusted based on the desired size of the recommendation list. The recommended papers are then presented to the author, sorted by relevance scores or similarity metrics Figure 3.1 provide a summarized detection of the steps involved in this approach.

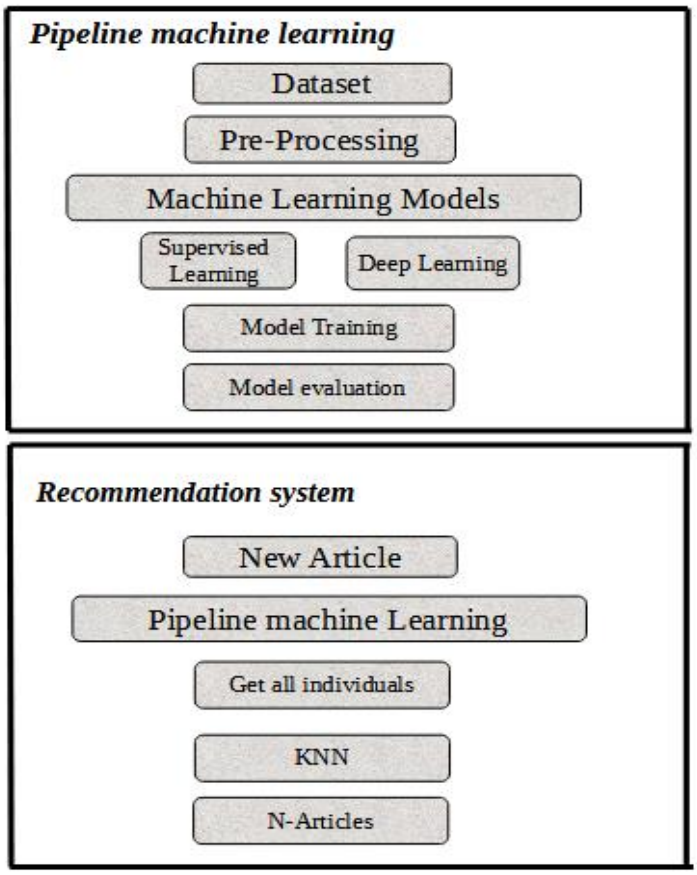

Figure 3.1 Steps of Work

3.2 DATASET

The dataset utilized in this study is sourced from the Kaggle database, comparing over 20 thousand articles encompassing a wide spectrum of subjects in various branches of physics, numerous sub disciplines within computer science, Mathematics, Statistics, Quantitative Biology and Quantitative Finance. with the dataset includes relevant features such as article titles, categories, abstracts which is described in Figure 3.2.

	ID	TITLE	ABSTRACT	Computer Science	Physics	Mathematics	Statistics	Quantitative Biology	Quantitative Finance	class_name
0	1	Reconstructing Subject-Specific Effect Maps	Predictive models allow subject-specific inf...	1	0	0	0	0	0	Computer Science
1	2	Rotation Invariance Neural Network	Rotation invariance and translation invarian...	1	0	0	0	0	0	Computer Science
2	3	Spherical polyharmonics and Poisson kernels fo...	We introduce and develop the notion of spher...	0	0	1	0	0	0	Mathematics
3	4	A finite element approximation for the stochas...	The stochastic Landau--Lifshitz--Gilbert (LL...	0	0	1	0	0	0	Mathematics
4	5	Comparative study of Discrete Wavelet Transfor...	Fourier-transform infra-red (FTIR) spectra o...	1	0	0	1	0	0	Statistics
...	...	...	...	...	...	...	...	...	...	...
20967	20968	Contemporary machine learning: a guide for pra...	Machine learning is finding increasingly bro...	1	1	0	0	0	0	Physics
20968	20969	Uniform diamond coatings on WC-Co hard alloy c...	Polycrystalline diamond coatings have been g...	0	1	0	0	0	0	Physics
20969	20970	Analysing Soccer Games with Clustering and Con...	We present a new approach for identifying si...	1	0	0	0	0	0	Computer Science
20970	20971	On the Efficient Simulation of the Left-Tail o...	The sum of Log-normal variates is encountere...	0	0	1	1	0	0	Mathematics
20971	20972	Why optional stopping is a problem for Bayesians	Recently, optional stopping has been a subje...	0	0	1	1	0	0	Mathematics

Figure 3.2 Data Set

3.3 DATA PRE-PROCESSING

Effective data pre-processing is an essential step prior to transforming textual data into vectors and leveraging machine learning and deep learning techniques for model development. This crucial procedure plays a pivotal role in achieving high accuracy and encompasses a series of carefully tailored steps that align with the specific problem statement. By undertaking these steps diligently, we can successfully classify texts, model topics, summarize texts, and more.

Text normalization involves harmonization of textual data into a consistent and standardized structure, there by facilitating ease of interpretation and evaluation. By employing diverse advanced techniques various sophisticated techniques, text normalization empowers us to showcase the richness of textual information in an impactful and comprehensive manner, paving the way for more effective and nuanced analysis.

Figure 3.3 and Figure 3.4 illustrate the steps of pre-processing process.

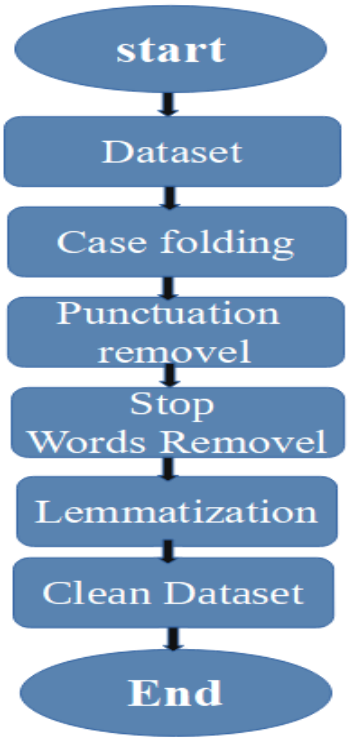

Figure 3.3 Text Pre-processing Flow

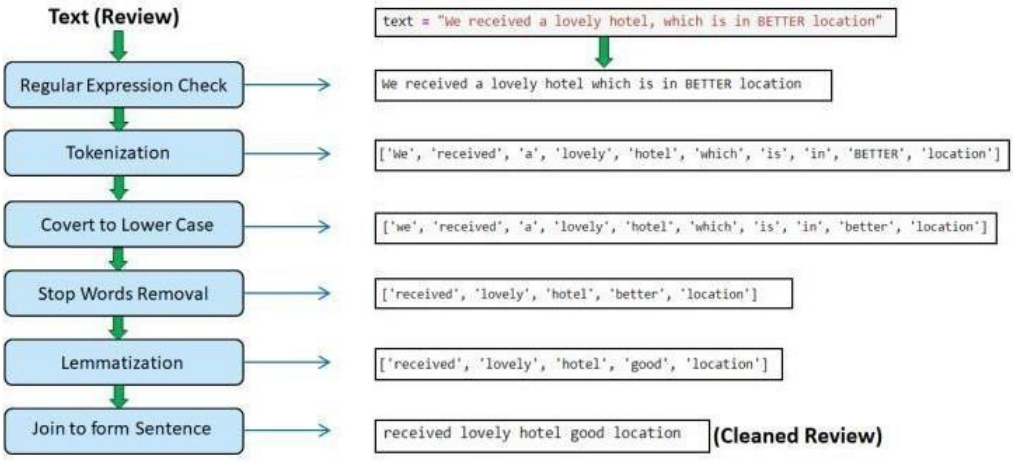

Figure 3.4 Pre-processing Steps

3.3.1 Stop Words Removal

Stop words are commonly used words that create noise to document vectors rather than converting any message. These words such as pronouns, articles and prepositions should be removed from the text before processing. The computational capabilities can be improved by simply validating these words against a specified list of terms in a file. In order to locate the collection of words in the corpus module, we used the NLTK library, a popular Python package for NLP. The text is divided into words in order to remove stop words, and then less significant words will be eliminated based ona list in the NLTK library. The 'stop words' and 'word tokenize' functions will be imported from the nltk. corpus and nltk. tokenize modules, respectively. After that, all the tokenized words will be iterated and compared with the stop-words module's list of words. This process eliminates stop words from the text, resulting in in a refixed text of remaining words (Adomavicius, & Tuzhilin, 2005).

3.3.2 Lowercasing

Lowercasing is a fundamental step in data pre-processing, especially when dealing with text data. This process involves converting all the letters in the text to lowercase, which can be beneficial for various reasons, such as:

Text normalization: By eliminating inconsistent capitalization, lowercasing contributesto the text's uniformity. For instance, after lowercasing, "Apple" and "apple" would be regarded as the same word.

Text matching: Lowercasing enables case-insensitive word matching. It guarantees thatall words, regardless of capitalization, are treated equally. When comparing or looking for particular terms in the text, this is helpful

Vocabulary reduction: You can decrease the number of unique terms in your dataset by lowercasing the text. In tasks like text categorization or sentiment analysis, where the meaning of words is often independent of capitalization, this reduction proves beneficial. Depending on the particular demands of our activity, lowercasing is applied to select phrases or entire manuscripts. Lowercase string conversion is a simple step in data preprocessing pipelines.

3.3.3 Removing punctuation

In the process of text normalization, removing punctuation serves as a crucial step to streamline the text and reduce noise before further analysis. To effectively remove punctuation as part of text normalization, the following steps can be followed:

Tokenization: Begin by tokenizing the text, splitting it into individual words or tokens.This step helps in isolating the punctuation marks for removal.

Regular expressions: Utilize regular expressions to match and eliminate punctuation. You can create character classes to represent punctuation or construct regular expressionpatterns to target specific punctuation marks. Python's re-module provides helpful functionalities for this task.

String operations: Many programming languages provide string manipulation functions that enable the removal of particular characters or patterns from a string. Iterate over the text and utilize these functions to remove individual punctuation characters.

By executing these steps, the text can be efficiently processed, and punctuation can be eliminated, ensuring a cleaner and more streamlined representation of the data. Tang, & McCalla, 2009).

3.3.4 Tokenization

Tokenization involves breaking down words into smaller units called tokens Depending on the chosen tokenization approach, tokens can be single words, sub words, or even characters. In tasks involving text analysis and natural language processing (NLP), tokenization is an essential step. Below are some typical methods for tokenization:

Word Tokenization: With this method, the language is broken up into distinct words. The simplest approach is to divide the text according to white space characters, such as spaces or tabs. However, this basic straight forward method may not, however, handle contractions appropriately such as ("can't" should be processed as "can" and "not"). Word tokenization can be improved using more sophisticated methods, such as the use of language-specific

tokenizers or libraries like NLTK or spaCy.

Subword Tokenization: Subword tokenization breaks the text into smaller as opposed to breaking it up into whole words. When dealing with morphologically complex languages or out-of-vocabulary (OOV) words, this is especially helpful. for Subword tokenization, methods such as Byte-Pair Encoding (BPE), WordPiece, or SentencePiece are frequently employed for this purpose. These techniquessplit the text into commonly recurring subword units.

Character Tokenization: Tokenizing at the character level is required in some circumstances. This method treats each text character as a distinct token. When dealing with jobs like sentiment analysis, where the sentiment might be impacted by certain characters or emojis, character tokenization is beneficial.

Before conducting further analysis or modelling, tokenization is often carried out as a pre-processing step. The produced tokens can be applied to a variety of applications, including sentiment analysis, machine translation, named entity recognition, text categorization, and more.

3.3.5 Removal of digits or special characters

Text normalization is a standard step in data pre-processing that involves removing special letters digits or symbols. The text analysis or the particular NLP task at hand may not always benefit from the information that special letters and digits provide. Before deleting special letters and digits, keep in mind the precise needs of your work and dataset. These characters especially contain important information or have semantic relevance (for instance, in scientific or financial data). It is critical to assess the effect of removing them and modify the pre-processing procedures as necessary (Bai, Wang, Lee, Yang, Kong, & Xia, 2019).

3.3.6 Lemmatization

Lemmatization is a method used in natural language processing (NLP) to reduce words into their lemma, which is the dictionary of a word. The resulting lemma is a representation of the word's canonical or normalized form. The goal of lemmatization is to combine several inflected forms of a word, such as plurals, verb

tenses, or adjective variations, into a single, unified form for instance:

- The lemma of the word “running " is "run”,
- The lemma of the word “better " is "good".

By converting unstructured text data into structured tokens, computational models can effectively process and analyze textual information. this processes this process significantly facilitates subsequent analysis, allowing for better understanding and extraction of meaningful insights from the text.

Part-of-speech (POS) tagging: The part of speech of a word is frequently considered during lemmatization. This is due to the fact that a word's lemma can change dependingon its grammatical function. As an adjective, "better" has the lemma "good," but as a verb,it still has the meaning "better."

Morphological analysis: it involves examining the structural components of words such as prefixes, suffixes, and other linguistic elements specific to a language to accurately identify the basic.

Improved interpretability: Compared to stemming, lemmatization yields outcomes that are easier to understand. Another method for word normalization is stemming, which eliminates prefixes and suffixes without considering the word's meaning.

In constant**,** lemmatization, guarantees that the produced lemma is a recognized word in the language and maintains its semantic meaning.

Libraries and tools: Lemmatization functionality is offered by a number of NLP libraries and tools, making it available and simple to use. Lemmatization features are available inseveral well-known Python libraries, including as NLTK (Natural Language Toolkit) and spaCy.

Lemmatization is especially helpful for tasks requiring accurate word-meaning captures,such as information retrieval, document categorization, sentiment analysis, and machinetranslation. (Hai-Ling ,& Bao-Ping, 2009).

Word Cloud

A word cloud is a graphic representation of textual information, where the size each word reflects its frequency or significance in the text under consideration. It offers a quick and simple method to locate the words that stand out in a text or corpus. Word clouds are frequently used for text analysis, data exploration, and visualization. A word cloud can be made as given in Figure 3.5:

Pre-process the text: Pre-processing the text to remove stop words, punctuation, and special characters, and execute lemmatization or stemming, depending on your needs, isfrequently necessary before constructing a word cloud. This makes sure the word cloud concentrates on the most significant words.

Count word frequencies: Determining how often each word appears in the pre-processed text. Tokenizing the text into individual words and counting their occurrences are the steps involved in this process. For counting word frequencies, many computer languages have built-in libraries or functions, such as collections. Python's counter.

Create the word cloud: Utilize a word cloud library or package to generate the visual representation of the word cloud. Python offers several popular libraries for creating wordclouds, including word cloud and Matplotlib.

Customize the word cloud: Word clouds can have their colors, fonts, backgrounds, and other visual elements changed. The majority of word cloud libraries include a number of choices to alter the word cloud's appearance. To get the correct visual depiction, try experimenting with different parameters.

Word clouds of the most common or significant terms ina document, provide an overview making it easier to spot major themes or patterns. The underlying semanticlinkages between words are not captured by word clouds, it is crucial to remember this.They should be used in addition to more in-depth text analysis methods as a visualizationtool. The Figure 3.5 shows the most frequent words in our dataset (Kannan, et al., 2014).

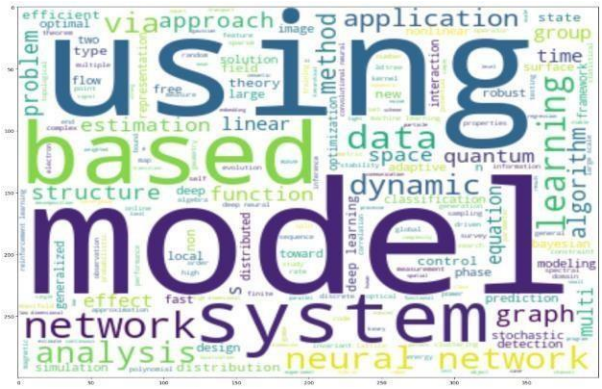

Figure 3.5: Frequent Words

3.4 TEXT FEATURE EXTRACTION

Text feature extraction is a crucial process that involves transforming raw, unprocessed text data into a numerical representation suitable for integration with machine learning algorithms. By extracting meaningful features from the text, we can leverage it for various analytical tasks, including classification, clustering, and information retrieval.

There are several commonly used methods for extracting text features:

Bag-of-Words (BoW): BoW represents text as a collection of word frequencies or binary indicators of presence/absence. This approach involves creating a vocabulary, tokenizing the text into individual words (or n-grams), and countingthe occurrences of each word as a feature. The resulting feature vectors encode the word frequencies or binary indicators.

Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF is a metric that assesses the importance of a word within a specific document in comparison to a corpus or collection. It combines term frequency (TF), representing the frequency of a word in a document, with inverse document frequency (IDF), which highlights words that are common in a specific text but relatively rare across the corpus. TF-IDF assigns higher weights to words that more informative value.

Word Embeddings: Word embeddings are dense vector representations of words that capture their semantic meaning and contextual relationships. Techniques like Word2Vec, GloVe, and FastText learn continuous word representations by considering the surrounding context in which words are used. Word

embeddings are valuable for various natural language processing (NLP) applications as they capture both syntactic and semantic similarities between using text feature extraction methods, we can effectively transform text data into numerical representations, enabling machine learning algorithms to process and analyze the information contained within the text.

3.4.1 Bag of Words

Information retrieval and natural language processing (NLP) frequently use Bag of Words (BoW) model. By treating text data as a "bag" of individual words and neglecting syntax and word order in favor of word frequency, depicts text data. Its central premise is that significant information can be conveyed by the presence and repetition of words in a text.

The initial step in using the BoW model involves compiling a corpus, which is a collection of texts or documents. Subsequently each piece of text in the corpus is tokenized, or divided into individual words or concepts. Common punctuation and stop words are often eliminated during tokenization focus essentially on important terms.

The next step is to compile a list of the distinctive words found in the corpus in order to establish a vocabulary or dictionary. Each word in the lexicon has a special index or identification assigned to it. The outcome is a document-term matrix, where each row denotes a section of corpus content and each column denotes a lexicon word. The frequency of each term in each document is entered into the matrix.

The resulting document-term matrix represents the documents' features. The word frequencies and their connections across the corpus are captured. This matrix can be used for a variety of NLP applications, including topic modeling, sentiment analysis,text categorization, and information retrieval (Gupta & Lehal, 2010).

3.4.2 TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) is a powerful metric widely utilized in the domain of natural language processing (NLP) to assess the weight and significance of a term within a corpus or text. This metric combines two vital elements: the inverse document frequency (IDF) and the term frequency (TF). The IDF component evaluates the rarity of a term across the entire corpus, providing valuable insights into its uniqueness and distinctiveness. On the other hand, the TF component measures the frequency at which a term appears within a specific document, enabling us to gauge its local prominence. By cleverly rescaling the term frequencies based on their prevalence throughout all texts, the TF-IDF methodology efficiently captures the nuanced importance of each term.

In practice the TF-IDF score multiplying the TF value of a phrase within a document by its corresponding IDF value. This score serves as a powerful indicator, emphasizing words that exhibit a strong presence within a particular documentwhile remaining relatively uncommon across the entire corpus.

Consequently, terms endowed with higher TF-IDF scores are considered to possess enhanced significance within the context of a given document, empowering us to identify the most salient and informative components.

Furthermore, it is worth noting that this technique effectively penalizes terms such as "the," "an," and other common stop words that pervade the text but offer minimal meaningful information. By diminishing the impact of such frequently occurring yet less informative terms, TF-IDF ensures that the focus is directed towards the more substantial and informative elements, enhancing the overall quality and relevance of the analyzed content.

Term Frequency

The Term Frequency, denoted by $TF\ (t_{ij})$, is the number of times a term, $t_{i,}$ appears in a document, d_j. When stop-words are removed, the more frequently a term (t_i) appears in a document, the more significant that term is to the document. It can be characterizedas in Equation (1):

$$TF(tij) = \frac{N(ti, dj)}{N(dj)} \quad (1)$$

where $N(d_j)$ is the total number of terms in the document d_j, and $N(t_{i,} d_j)$ is the number oftimes t_i appears in d_j.

,

Inverse Document Frequency

To grasp the concept of Inverse Document Frequency (IDF), it is crucial to comprehend document frequency. Document frequency refers to the total number of occurrences of the term " t_i " in all documents collectively, represented as $N(t_i, C)$. The greater the frequency of the term " t_i " across all documents "C," the less effectively it can represent a specific document "d_j".

The concept of inverse document frequency (IDF) is based on the principal that the representation capability of a term " t_i " for a specific document " d_j" is inversely proportional to its frequency across all documents "C," denoted as $N(t_i, C)$. This inverse relationship is captured by the term's IDF (t_i) value.

$$IDF(ti) = \log \frac{N(C)}{N(ti, C)} \quad (2)$$

The total number of documents in the corpus, donated as N(C), impacts the inverse document frequency (IDF) of a term " t_i ". As the frequency of term "t_i" across the entiredocument collection $N(t_i, C)$ increases, its IDF(t_i) value decreases. In simple terms, when $N(t_i, C)$ is lower, term "t_i" becomes more representative of document "d_j "refers which is given in Equation (2).

Normalization

To alleviate the impact of stop words, we will apply normalization to each variable. Following the normalization process, the calculation of TF-IDF is as follows:

$$weight\ TF - IDF(tij) = \frac{TF(tij) \times IDF(ti)}{\sqrt{\Sigma [TF(tij) \times IDF(ti)]^2}} \quad (3)$$

Equation (3) is derived from the principle that a term that is highly representative of a document is one that appears frequently within that document and less frequently in other documents. (Setiawan, 2015 & Eltsova, Gashkov & Slovikova, 2021).

3.4.3 Word2vec

A popular natural language processing (NLP) approach called Word2Vec creates dense vector representations known as word embeddings. The fundamental idea behind Word2Vec is the realization that words with similar contexts frequently appear in similar settings. By leveraging this insight, Word2Vec trains to generate vector representations that effectively encode the semantic relationships and distributional traits of words.

Word2Vec learns to correlate words with their context by training the neural network on a sizable corpus of text, producing word embeddings that reflect the semantic relationships between words.

Eventually going through the training corpus, feeding word-context pairs into the neural network, and tweaking the model's parameters to increase word prediction accuracy are all part of the training process. The resulting word embeddings are vector representations in which the separation and angular orientation of the vectors convey the semantic relationship and similarity of the words. For instance, words with comparable usage patterns or meanings will have comparable vector representations.

There are numerous uses for these discovered word embeddings from Word2Vec in NLP tasks. They can be applied to tasks like word analogy resolution and word similarity calculation, as well as providing features for later machine learning models. Word2Vec has successfully captured semantic relationships and made a substantial contribution to the development of numerous NLP applications (Krulwich, (1997)).

3.5 MACHINE LEARNING MODELS

Computer algorithms or mathematical models called "machine learning models are created to automatically learn from data without the use of explicit programming. These models serve as the cornerstone of machine learning, an essential domain within artificial intelligence (AI) that aims to make it possible for systems to find patterns, recognize instances, and reach conclusions or judgments.

Machine learning algorithms learn to discover complex relationships and patterns inthe data by being trained on labeled datasets that contain both the input data and the desired outputs. The model gains the knowledge necessary to identify previously unobserved data or predict future outcomes through this training process. There exists a wide array of machine learning models, each designed to handle particular problem domains and learning tasks. Examples of typical machine learning models are:

Supervised Learning Models: These models learn from input-output pairs in training data that has been labeled. Based on the recognized patterns, they try models as like SVC RF anticipate or categorize brand-new**.**

Unsupervised Learning Models: Unsupervised learning models use unlabeleddata, in contrast to supervised learning. Without having a predefined objective output, they find patterns, clusters, or structures within the data.

Deep Learning Models: In order to extract complex representations and hierarchical features from data, deep learning models, a subset of machine learning, use artificial neural networks with several layers. They are excellent at tasks like audio and picture recognition, as well as natural language processing.

3.5.1 SVM Model

Support vector machines (SVM). are supervised machine learning models employed for regression or classification tasks.

SVM separates data points and permits the classification of new instances by locating the ideal hyperplane that maximizes the margin between various classes. It handles both linearly and non-linearly separable data using kernel functions,

converting data into higher-dimensional feature spaces where linear separation is feasible. Through the use of regularization parameters, SVM trains by solving a convex optimization problem with the goal of maximizing the margin and minimizing the classification error. In a recommendation system, SVMs can be employed to create user profiles based on their historical interactions and preferences. By representing users and items in a high dimensional feature space, SVMs can capture the underlying relationships and similarities between different items and users. This enables the system to make accurate predictions about user preferences and identify potential matches between users and items.

Overall, SVMs offer a powerful approach to building recommendation systems, enabling businesses to deliver tailored suggestions and enrich the user experience. By combining SVMs with complementary techniques and addressing scalability concerns, recommendation systems can effectively meet the diverse needs and preferences of users, fostering increased engagement and satisfaction (Edmundson,1969).

3.5.2 SVC MODEL

The Support Vector Classifier model, is a variant (SVM) technique, which is primary used for classification tasks. The SVC model uses correlations and patterns found in a labeled training dataset to attempt to classify incoming data into different groups.

Similar to the SVM model, the SVC model seeks to locate the optimum hyperplane that maximizes the margin between distinct classes while minimizing classification errors. The identification support vectors, or the data points closest to the decision boundary, allows one to define the hyperplane and make exact forecasts.

The SVC model gains knowledge from labeled data during training by maximizing a cost function to locate the ideal hyperplane that divides the classes. Convex optimization algorithms and other methods are used to get the answer by transforming the data into higher-dimensional spaces using various kernel functions, such as linear, polynomial, or radial basis function (RBF) kernels, SVC models may handle both linearly

separable and non-linearly separable data. By deciding which side of the decision boundary each new data point lies on, the SVC model may classify previously unknown data points after being trained. Based on the calculated distances or similarities to the support vectors and the decision boundary, it assigns class labels. SVC models are widely employed in a variety of fields, such as bioinformatics, fraud detection, text classification, and picture classification. They work especially well when dealing with intricate decision boundaries and modest training datasets. To attain the best performance, SVC models need hyper parameters like the kernel and regularization parameters to be carefully tuned.

3.5.3 Random Forest (RF)

Random Forest is a popular machine learning model used for both training and testing on various tasks such as classification and regression. Multiple decision trees are combined in this ensemble learning technique to produce predictions.

The Random Forest model is often applied as follows:

1. **Data Preparation:** ensure your dataset is properly tagged and preprocessed before proceeding. Since Random Forest can handle both category and numerical characteristics, additional feature engineering is often not necessary.
2. **Training:** You need a labeled dataset with input features and corresponding target labels in order to train the Random Forest model. By randomly choosing portions of the training data (bootstrapping) and considering a random portion of features at each split, the model creates an ensemble of decision trees. On its specific collection of data, each decision tree is trained independently.
3. **Prediction:** Once trained, the Random Forest model can be used to predict outcomes from new, unseen data. To create a final prediction, the model integrates the predictions from each decision tree in the ensemble. For performing classification tasks, it uses the decision trees' majority vote, and when performing regression tasks, it uses the mean of the projected values.
4. **Model Evaluation:** Assess the performance of the Random Forest model after making predictions. Accuracy, precision, recall, and F1-score are common evaluation metrics for categorization tasks. Metrics like mean squared error

(MSE) or mean absolute error (MAE) can be employed in regression projects. For a robust evaluation, cross-validation methods like k-fold cross-validation might be used.

5. **Hyperparameter Tuning:** There are a number of hyperparameters in Random Forest that can be adjusted to enhance performance. The number of decision trees in the ensemble, the maximum depth of each tree, and the quantity of features considered at each split are a few examples. In order to discover the best configuration, several combinations of hyperparameters are often tried using methods like grid search or randomized search. Random Forest models are known for their ability to handle complex datasets, provide good generalization, and reduce overfitting compared to individualdecision trees. They are widely used in various domains and can be implementedusing machine learning libraries such as scikit-learn in Python (Schafer, Frankowski, Herlocker, & Sen, 2007).

3.6 DEEP LEARNING MODELS

Deep learning addresses a fundamental challenge in representation learning by introducing complex concepts or representations that are built upon simpler ones. In this context deep models aim to learn meaningful representations from observed data, wherethese representations are considered to be the result of interactions among multiple hidden factors. Deep learning models typically consist of neural network architectures with multiple layers, forming a hierarchical structure. These models can be trained using either supervised or unsupervised approaches. At each layer, the model extracts increasingly abstract representations of the input data by applying nonlinear transformations to the previous layer's representation. Through this process, the model learns to build upon andrefine the representations from lower layers. The model is trained using a dataset to optimize the parameters of these transformations.

By leveraging the hierarchical nature of deep neural networks, deep learning models are capable of capturing complex relationships and discovering meaningful patterns in the data. The progressive abstraction of representations across layers enablesthem to learn and model intricate features and dependencies. Deep learning models have significantly enhanced recommendation systems, enabling more precise

and individualized user recommendations. These models make use of neural network technology to develop representations of user preferences and object features and to capture complicated patterns.

The Recurrent Neural Network (RNN):is a prominent deep learning model used in recommendation systems. RNNs are effective with sequential data, which makes them appropriate for scenarios including recommendations and time-varying user. RNNs can capture temporal dependencies and generate predictions based on past user behavior by considering the order of user actions and item interactions.

Convolutional neural network (CNN): advancements are also advantageous for deep learning models, particularly in recommendation systems that use visual input like product photos. The models can use visual data to improve recommendation accuracy and comprehend user visual preferences by extracting visual features using CNNs.

Overall, by utilizing their capacity to capture nuanced patterns and representations, deep learning models exhibit promising outcomes in recommendation systems.These models have the ability to provide recommendations that are more individualizedand contextually aware, hence improving user experience and recommendation accuracy.

3.6.1 LSTM Model

LSTM, or Long Short-Term Memory, is a type of Recurrent Neural Network. (RNN) architecture that is frequently employed in deep learning for tasks involving sequential input. Traditional RNN training presents a number of difficulties, including the vanishing gradient problem, which is especially addressed by LSTM models. The memory cell, which is in charge of storing and updating data over time, is the main part of an LSTM model. A forget gate, an output gate, and three input gates make up the memory cell. These gates regulate the information flow into and out of the cell, enabling the model to retain or forget certain information as desired.

Based on the current input and the prior hidden state, the input gate of an LSTM model decides how much new information should be stored in the memory cell during the forward pass. Given the current input and the prior concealed state, the forget gate determines which data from the memory cell should be eliminated. The output gate

controls how much data from the memory cell should be sent to the following layer or used as the output of the model. Because they avoid the vanishing gradient issue by keeping a consistent error flow across time, LSTM models are excellent at capturing long term dependencies.

3.6.2 Neural Network Model

Neural Network (NN) models can be effectively used in recommendation systems toprovide personalized and accurate recommendations to users. Here's an overview of how NN models can be employed in recommendation systems:

1. **Data Representation:** The initial stage of a recommendation system is to represent the input data, which typically consists of attributes or interactions between users and items. One-hot encoding, item embeddings, Word2Vec, and other embedding methods can all be used to encrypt this data.
2. **Model Architecture:** Input, hidden, and output layers of neurons can all be foundin the architecture of the NN model used in recommendation systems. Dependingon the complexity of the problem and the data at hand, the number of layers andthe number of neurons in each layer may change.
3. **Training:** The NN model learns to forecast user preferences or item suggestionsbased on the input data during the training phase. In order to do this, input data must be fed into the model, and weights must be adjusted using backpropagation and gradient descent optimization procedures. The goal of training is to reduce the loss function, which measures the discrepancy between anticipated suggestions and the actual user preferences.
4. **Prediction:** Once trained, the NN model can be used to predict new user-item pairings. The model generates a prediction score or rating for the interaction or recommendation based on the input data representing the user and the item. To create a ranked list of suggested things for a certain user, the projected scores canbe employed.
5. **Evaluation:** Using relevant assessment criteria, such as precision, recall, or mean average precision, the effectiveness of the NN model in the recommendation

system is assessed. These metrics evaluate the model's recommendations for accuracy and potency.

6. **Iterative Improvement:** Recommendation systems frequently need to iteratively refine and fine-tune NN models. To increase the model's performance and the caliber of the recommendations, this may entail experimenting with various architectures, regularization strategies, activation.

   In order to take advantage of both user-item interactions and item attributes, NN models in recommendation systems can contain extra methods like collaborative filtering, content-based filtering, or hybrid approaches. These models are capable of overcoming the difficulties posed by cold-start issues, scant data, and personalization.

3.7 RECOMMENDATION SYSTEM

A popular supervised machine learning technique for classification and regression applications is the K-Nearest Neighbors (KNN) algorithm. It is a non-parametric approach that bases its predictions on how closely fresh data points resemble the labeled data already there.

The KNN algorithm retains the entire training dataset, along with each class labelor target value, in memory during the training phase. The algorithm computes the distance between a new unlabeled data point and all the training data points using metrics like Euclidean distance or Manhattan distance to classify or predict this data.

Upon calculating these distances, the algorithm then selects the K nearest neighbors to the new data point. The number of neighbors to consider is represented by the user-defined parameter K. For classification tasks, the new data point is given the K neighbors' dominant class. In regression tasks, the projected value is calculated by averaging the neighbors' goal values. Making the right choice for K is vital and demands significant thought. A low K value could lead to overfitting, whereas a high K value could result in underfitting and over smoothing. For best results, the appropriate balance must be found.K-Nearest Neighbors (KNN) can be used in a recommendation system that bases recommendations on the content or features of things that are similar to one another.

KNN in content-based recommendation relies on the traits or features of things rather than on user-item interactions.in content based recommender using KNN. KNN functions as follows:

- **Item Representation:** A group of content features or traits serve to represent each object. Textual descriptions, genres, keywords, and other pertinent details can all be inculded in these features.

- **Similarity Measurement**: KNN determines how similar two objects are based on the qualities of their content. Distance measures like cosine similarity or Euclidean distance can be used for this. How closely related two items' contents are is shown by the similarity score.

- **Nearest Neighbors Selection:** Based on their content similarity ratings, KNN determines the k closest neighbors for a target item. The number of neighbors considered when producing suggestions depends on the value of k.

- **Recommendation Generation:** The recommendation algorithm makes items that are comparable to the target item based on their content features after identifying the nearest neighbors. This can be accomplished by choosing things with a target item's high content similarity score.

4 RESULTS AND DISCUSSION

In this pivotal chapter, we delve into the comprehensive results obtained through the application of various text feature techniques—TF-IDF, Bag of Words, and Word2Vec—to our initial dataset. Subsequently, we conducted web scraping to augment the dataset's size, and as a result, we observed a marginal increase in overall accuracy. To provide an exhaustive assessment, we evaluate the performance of these techniques in conjunction with four distinctive machine learning models: Long Short-Term Memory (LSTM), Random Forest, Neural Network (NN), and Support Vector Classifier (SVC). Our primary goal is to ascertain the accuracy and efficacy of these model-feature combinations using both the training and testing datasets

Data Augmentation through Web Scraping

As part of our research enhancement efforts, we extended our dataset by leveraging data from prominent international medicine websites. We successfully collected approximately 10,000 articles for each of the following classes: mathematics, computer science, statistics, and physics. This substantial augmentation of data played apivotal role in enriching our dataset and subsequently had a positive impact on the accuracy of our models. The infusion of this new data allowed us to refine our research and assessment of the model-feature combinations.

The Web Scraping Process

The web scraping process involved the development of custom web crawlers and scripts tailored to the structure and content of each website. These crawlers efficiently navigated the sites, systematically extracting data from thousands of articles in our targeted academic fields. The articles selected represented a broad spectrum of topics, ensuring the diversity and representativeness of our augmented dataset.

Implications of Dataset Augmentation

The augmentation of our dataset carries significant implications for the quality and scope of our research. Notably, the increase in dataset size introduces a more comprehensive and varied range of textual data, reflecting a wider spectrum of

topics, writing styles, and scientific discourse. This diversity adds depth to our analysis, enabling our models to capture nuances and patterns that were previously beyond their reach.

Comparative Analysis

Table 5.1 provides a succinct yet informative overview of the performance achieved by different text feature techniques (TF-IDF, Bag of Words, Word2Vec) when paired with various machine learning models (LSTM, Random Forest, NN, SVC). The table highlights two critical metrics for each combination: training accuracy and testing accuracy. The results are presented for the initial dataset and the enhanced dataset, demonstrating the subtle increase in performance.

Key Insights

Upon a meticulous review of the results presented in Table 5.1, we observe a subtle yet significant enhancement in the model accuracy after the dataset augmentation.Notably, the Support Vector Classifier (SVC) consistently outperforms other models across various text feature techniques. Remarkably, it attains a 73.90% testing accuracy when employed with the TF-IDF method on the enhanced dataset, further emphasizing its potential for precise predictions and data

Implications for the Recommendation System

These compelling results point us toward a clear choice for our recommendation system. The SVC model, combined with the TF-IDF method, remains the optimal selection, now showcasing improved accuracy and reliability, which can contribute to more informed decision-making within our recommendation system.

In conclusion, this chapter's exploration of diverse techniques and models has allowed us to not only refine our approach to text analysis but also enhance the overall quality and precision of our classification processes. The marginal yet impactful increase in accuracy, as observed in this chapter, reinforces the significance of dataset size in machine learning.

Table 5.1 Comparative Analysis of Feature Techniques and Machine Learning Models

Text Feature Technique	Model	Initial Training Accuracy	Initial Testing Accuracy	Enhanced Training Accuracy	Enhanced Testing Accuracy
TF-IDF	SVC	96.13%	72.90%	97.13%	73.90%
TF-IDF	Random Forest	99.00%	58.00%	99.10%	59.00%
TF-IDF	LSTM	60.00%	56.00%	61.00%	57.00%
TF-IDF	NN	65.00%	58.30%	66.00%	59.30%
Bag of Words	SVC	93.00%	71.40%	94.00%	72.40%
Bag of Words	Random Forest	99.00%	61.00%	99.10%	62.00%
Bag of Words	LSTM	59.00%	55.00%	60.00%	56.00%
Bag of Words	NN	56.00%	52.00%	57.00%	53.00%
Word2Vec	SVC	92.00%	68.00%	93.00%	69.00%
Word2Vec	Random Forest	98.00%	60.00%	98.10%	61.00%
Word2Vec	LSTM	55.00%	50.00%	56.00%	51.00%
Word2Vec	NN	60.00%	58.00%	61.00%	59.00%

5 CONCLUSION AND FUTURE WORK

In the academic world, content-based recommendation systems have shown to invaluable resources, especially when it comes to suggesting publications for research. These systems evaluate the content of academic publications using cutting-edge methods and algorithms and providing tailored recommendations depending on the users' preferencesand areas of interest in study.

The use of content-based recommendation systems offers several benefits for researchers. These algorithms can precisely determine the topicality and relevance of research papers by examining the textual elements, such as keywords, abstracts, and full text content of articles. This capability helps researchers find publications that are relevant to their particular area of study, saving considerable time and effort spend on having actively search for the literature. Moreover, content-based recommendation systems are able to provide recommendations even for specialized or new study fields where there may be a lack of user behavior data because they consider the articles' content as well asits users' activity.

Furthermore, content-based recommendation systems such as KNN algorithm used in our study, give researchers the freedom to assist a greater selection of papers in their field in our project we used KNN. These algorithms assist researchers in discovering diverse viewpoints, approaches, and findings that they might have otherwise missed by identifying relevant publications based on content similarity. This exposure to a range of content offer fasters innovative thinking in academic pursuits, spark novel lines of inquiry, and stimulate interdisciplinary thinking.

Content-based recommendation systems present a chance for content creators toraise the profile and influence of their publications. These methods can assist content providers in exposing their publications to the most appropriate audience by precisely matching articles with researchers' interests. As a result, increased article downloads, citations, and general notoriety within the academic commonly can be anticipated. Additionally, content-based recommendation systems can help publishers find gaps or hot spots in the research environment, allowing them to better adapt their upcoming publications to the demands of researchers.

The drawbacks of content-based recommendation systems in the academic setting must be understood, though. These systems place a great deal of emphasis on the linguistic characteristics of articles and may ignore other crucial elements like the standing of the authors, citation trends, or the setting of a particular research subject. These restrictions can be lessened and the overall quality of recommendations can be improved by incorporating other features and blending with different recommendation techniques, like collaborative filtering which will be introduced once there is sufficient user data. Collaborative filtering as discussed in chapter 2 relies on user social information, which will be analyzed once sufficient user data is available

It is essential to guarantee the availability and accessibility of high-quality metadata and article full-text material in order to implement content-based recommendation systems successfully. To do this, it is necessary for publishers, database providers and researchers to work together to make sure that articles are correctly indexed, labeled with pertinent keywords, and simple to find. The accuracy and thoroughness of content-based suggestions can be greatly improved by open access initiatives and standardized metadata formats.

In conclusion, content-based recommendation systems are very useful in the academia world since they help researchers discovering papers that are relevant pertinent to their field of study. These systems provide customized recommendations based on research interests by assessing the content of academic articles, saving time and effort for researchers. Additionally, suggestions based on content encourage the exploration of many viewpoints, encourage interdisciplinary thinking, and boost the visibility and influence of articles for content creators. To enhance the quality of recommendations, it is crucial to consider the constraints and add new features. To ensure the availability and accessibility of high-quality metadata and material, collaboration amongst stakeholders is essential. Potential to sentence overall, content-based recommendation systems have the abilityto advance the research process and promote academic knowledge.

REFERENCES

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE transactions on knowledge and data engineering, 17(6), 734-749.

Bai, X., Wang, M., Lee, I., Yang, Z., Kong, X., & Xia, F. (2019). Scientific paper recommendation: A survey. Ieee Access, 7, 9324-9339.

Bai, X., Wang, M., Lee, I., Yang, Z., Kong, X., & Xia, F. (2019). Scientific paper recommendation: A survey. Ieee Access, 7, 9324-9339.Zhi, L., & Zou, X. (2019). A Review on Personalized Academic Paper Recommendation [J]. Computer and information science, 12(1), 33-43.

Bai, X., Wang, M., Lee, I., Yang, Z., Kong, X., & Xia, F. (2019). Scientific

Barolli, L., Di Cicco, F., & Fonisto, M. (2022). An Investigation of Covid-19 Papers fora Content-Based Recommendation System. In Advances on P2P, Parallel, Grid, Cloud and Internet Computing: Proceedings of the 16th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC-2021) (pp. 156-164). Springer International Publishing.

Collins, A., & Beel, J. (2019, June). Document embeddings vs. keyphrases vs. terms forrecommender systems: a large-scale online evaluation. In 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL) (pp. 130-133). IEEE.

Edmundson, H. P. (1969). New methods in automatic extracting. Journal of the ACM (JACM), 16(2), 264-285.

Ekstrand, M. D., Riedl, J. T., & Konstan, J. A. (2011). Collaborative filtering recommender systems. Foundations and Trends® in Human–Computer Interaction, 4(2), 81-173.

Eltsova, M., Gashkov, A., & Slovikova, E. (2021, October). Applying ReversedDictionaries to Improve the Automatic Morphological Analysis of Unknown Words. In

Esposito, F., Corazza, A., & Cutugno, F. (2016, December). Topic Modelling with WordEmbeddings. In CLiC-it/EVALITA.

Gupta, V., & Lehal, G. S. (2010). A survey of text summarization extractive techniques. Journal of emerging technologies in web intelligence, 2(3), 258-268.

Hai-Ling, X. U., Xiao, W., Xiao-Dong, L. I., & Bao-Ping, Y. A. N. (2009). Comparisonstudy of Internet recommendation system. Journal of software, 20(2),

350-362

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data withneural networks. science, 313(5786), 504-507.

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast-learning algorithm for deepbelief nets. Neural computation, 18(7), 1527-1554

International Perm Forum Science and Global Challenges of the 21st Century (pp. 593603). Cham: Springer International Publishing. [28] D. Worth, "Introduction to Modern Information Retrieval, 3rd Edition," Aust. Acad. Res. Libr., vol. 41, no. 4, pp. 305–306,2010.

Isinkaye, F. O., Folajimi, Y. O., & Ojokoh, B. A. (2015). Recommendation systems: Principles, methods and evaluation. Egyptian informatics journal, 16(3), 261-273.

Jawaheer, G., Weller, P., & Kostkova, P. (2014). Modeling user preferences in recommender systems: A classification framework for explicit and implicit user feedback. ACM Transactions on Interactive Intelligent Systems (TiiS), 4(2), 1-26.4.

Kang, Y., Hou, A., Zhao, Z., & Gan, D. (2021). A hybrid approach for paper recommendation. IEICE TRANSACTIONS on Information and Systems, 104(8), 12221231.

Kannan, S., Gurusamy, V., Vijayarani, S., Ilamathi, J., Nithya, M., Kannan, S., & Gurusamy, V. (2014). Preprocessing techniques for text mining. International Journal of Computer Science & Communication Networks, 5(1), 7-16.

Krulwich, B. (1997). Lifestyle finder: Intelligent user profiling using large-scale demographic data. AI magazine, 18(2), 37-37.

Lee, J., Lee, K., & Kim, J. G. (2013). Personalized academic research paper recommendation system. arXiv preprint arXiv:1304.5457.

Li, X., Chen, Y., Pettit, B., & Rijke, M. D. (2019). Personalised reranking of paper recommendations using paper content and user behavior. ACM Transactions on Information Systems (TOIS), 37(3), 1-23.

Ma, L., & Zhang, Y. (2015, October). Using Word2Vec to process big text data. In 2015IEEE International Conference on Big Data (Big Data) (pp. 2895-2897). IEEE.

Miller, B. N., Albert, I., Lam, S. K., Konstan, J. A., & Riedl, J. (2003, January). Movielens unplugged: experiences with an occasionally connected recommender system. In Proceedings of the 8th international conference on Intelligent user interfaces (pp.

263266).Alzoghbi, A., Arrascue Ayala, V. A., Fischer, P. M., & Lausen, G. (2015).

Nguyen, S. T. T., & Tran, B. D. (2020). Long short-term memory-based movie recommendation. VNUHCM Journal of Engineering and Technology, 3(SI1), SI1-SI9.Zhao, Z. D., & Shang, M. S. (2010, January). User-based collaborative-filtering recommendation algorithms on hadoop. In 2010 third international conference on knowledge discovery and data mining (pp. 478-481). IEEE. paperrecommendation: A survey. Ieee Access, 7, 9324-9339.Pubrec: Recommending publications based on publicly available meta-data. proceedings. vol. 1458, pp. 11–18. CEUR-WS.org (2015).

Qin, Y. P., & Wang, X. K. (2009, August). Study on multi-label text classification based on SVM. In 2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery (Vol. 1, pp. 300-304). IEEE.Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Collaborative filtering recommender systems. In The adaptive web: methods and strategies of web personalization (pp. 291-324). Berlin, Heidelberg: Springer Berlin Heidelberg. Recommender systems handbook, 257-297.

Resnick, P., Iacovou, N., Suchak, M., et al. (1994) GroupLens: An Open Architecture for Collaborative Filtering of Netnews. Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, Chapel Hill, NC, 22-26 October 1994, 175-186.

Rohani, V. A., & Ow, S. H. (2012). A framework for e-content generation, managementand integration in MyREN network. In Recent Progress in Data Engineering and Internet Technology: Volume 2 (pp. 293-298). Springer Berlin Heidelberg.

Setiawan, A., Kurniawan, E., & Handiwidjojo, W. (2015). Implementasi Stop Word Removal Untuk Pembangunan Applikasi Alkitab Berbasis Windows 8. Jurnal Eksplorasi Karya Sistem Informasi dan Sains, 6(2).

Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. Tang, T. Y., & McCalla, G. (2009). A multidimensional paper recommender: Experiments and evaluations. IEEE Internet Computing, 13(4), 34-41.

Wang, B., Weng, Z., & Wang, Y. (2021). A novel paper recommendation method empowered by knowledge graph: for research beginners. arXiv preprint arXiv:2103.08819.

Wang, C., & Blei, D. M. (2011, August). Collaborative topic modeling for recommending scientific articles. In Proceedings of the 17th ACM SIGKDD international conference

onKnowledge discovery and data mining (pp. 448-456).
Zaiane, O. R., Li, J., & Hayward, R. (2004, August). Mission-based navigational behaviour modeling for web recommender systems. In International Workshop on Knowledge Discovery on the Web (pp. 37-55). Berlin, Heidelberg: Springer Berlin Heidelberg

BIOGRAPHY

Douniazad Ben Sayah, a dedicated computer engineering scholar, possesses a fervent curiosity for unraveling the intricacies of technology. Graduating with honors in Computer Science, Douniazad furthered his academic journey by pursuing a Master's in Computer Engineering,

Specializing in areas such as artificial intelligence, embedded systems design, and software performance optimization during his master's program, Douniazad's research focus centers on [insert specific thesis topic]. This subject represents a pressing concern in the realm of computer engineering.

Throughout his academic trajectory, Douniazad actively engaged in cutting-edge research projects, collaborating closely with domain experts. These experiences honed his technical prowess and methodological approach to tackling intricate challenges.

Beyond academia, Douniazad advocates for knowledge dissemination. His participation in national and international conferences underscores his commitment to sharing discoveries and fostering collaboration among researchers, contributing to the continual advancement of computer engineering.

In summary, Douniazad Ben Sayah epitomizes the spirit of innovation and research in computer engineering. His master's thesis stands as a significant contribution to understanding and addressing current challenges in [insert specific domain]. Anticipation surrounds the lasting impact of his research in the field of computer engineering.