FMDB ***Transactions on Sustainable Computer Letters***

Building a Content-Based Book Recommendation System

Rajkumar Rajasekaran[1,*], A. Jayaram Reddy[2], J. Kamalakannan[3], K. Govinda[4]

[1,4]School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu, India.
[2,3]School of Computer Science Engineering and Information Systems, Vellore Institute of Technology, Vellore, Tamil Nadu, India.
vitrajkumar@gmail.com[1], ajayaramreddyreddy@vit.ac.in[2], jkamalakannan@vit.ac.in[3], kgovinda@vit.ac.in[4]

Abstract: Recommender systems can be described as algorithms that are designed with the aim of suggesting relevant and most-used items to users (movies, products, videos, books, electronics, etc.). The whole system is meant to follow the user/customer's browsing tactics and find out the most relevant. In the present world of e-commerce, where every product is available online, users, as well as businesses, need an algorithm that can suggest items based on their category and the buyer's preference. These algorithms can be made to suit any category of products. The algorithm implemented in this system will help readers all around the world find books relevant to their choice (preferred genre) and other users' ratings. The recommendation system has been built on a NoSQL Graph Database. The main advantage of using Neo4J is that it captures the relationship between the data and similarity between items on the basis of the type and the preference of the user and also records the behaviour of data. There are various factors that can affect the recommendation system; we have considered users' favourite genres and similar types of items. Since it is a graph database, it is easy for any person to analyze and visualize the relationships between different nodes and also manage huge amounts of data with ease.

Keywords: Book Recommendation System; NoSQL Graph Database; Huge Amount of Data; Neo4j; Querying Relational Database; World of E-Commerce; Category of Products; Property Graph Data Model.

Received on: 05/01/2023, **Revised on:** 02/03/2023, **Accepted on:** 21/04/2023, **Published on:** 05/05/2023

Cited by: R. Rajasekaran, A. J. Reddy, J. Kamalakannan, and K.Govinda, "Building a Content-Based Book Recommendation System," FMDB Transactions on Sustainable Computer Letters., vol. 1, no. 2, pp. 103 –114, 2023.

1. Introduction

When it comes to dealing with vast amounts of information on e-commerce platforms, recommendation algorithms play an extremely important role in the process. It offers ideas to users, which assist them in various decision-making processes, and it does so in a variety of ways [1]. The brisk expansion of the e-commerce business has resulted in an increased demand for the provision of suggestions by means of the selective filtering of datasets including products [2]. Users of e-commerce platforms previously had a tough time making judgments about which products to purchase due to the vast array of products that were available to them through the various e-commerce platforms before the development of recommender systems. Users are becoming increasingly overwhelmed by the number of product options, which in turn leads to poor decision-making. In recent years, the utilisation of recommender systems has proven to be beneficial in the management of issues related to information overload. The issue of having too much information is addressed by recommender systems, which do this by presenting the user with a fresh, previously unseen item that may be pertinent to the user.

The more conventional ways of purchasing books, such as going to a bookstore, have been rendered obsolete as a result of the meteoric rise in popularity of online shopping. In more recent times, books have shifted from being kept on bookshelves to being read on Kindle readers. Books that are offered for sale online are organised into sections according to the literary category

*Corresponding author.

they belong to, making it easier for customers to search for specific titles. In addition, due to the rapid development in the volume and amount of books, a process needs to be put into place in order to offer users with books that are pertinent to the choices they make. The point at which recommendation systems become useful is at this point. Users of these systems can receive book recommendations tailored to their preferred literary category and typical purchasing behaviour. According to the findings of many statistical studies, recommender systems have been instrumental in contributing to Amazon's revenue growth by 35 percent [3].

In recent years, a significant amount of effort has been put toward the discovery and development of new algorithms and techniques for recommender systems. On the other hand, its performance has been subpar, which is problematic considering that an effective recommendation system must be able to handle data in a timely and precise manner. Because the cost of joins in relational databases is not incurred when connecting data in a graph database, this type of database is more efficient and versatile than relational databases. When compared to accessing relational databases, the cost of querying graph databases is significantly lower.

For the purposes of this article, we shall construct a book recommendation system by making use of a graph database. This will be a content-based recommendation system that will present users with individualised recommendations based on the genres that they are most interested in listening to. The method will also take into account how popular the books are by determining their popularity based on the average ratings of the books.

Neo4j will make use of the graph database that was provided. Neo4j's low complexity, minimum dependencies, and support across multiple platforms are the primary selling points for using this system. Cypher, a straightforward declarative querying language that is reminiscent of SQL, is used, and it offers complete support for the fundamental NoSQL features, such as flexible schemas and scalability. Neo4j is a database that uses NoSQL, however it offers ACID transaction dependability despite being a NoSQL database.

The Property Graph Model is implemented in the Neo4j database, which is a multi-relational database. This indicates that it is capable of working with nodes, relationships, and the characteristics associated with them. Both the nodes and the relationships can be compared to the vertices and edges of a graph, respectively. In addition to that, it offers a browser-based interface for utilising Cypher to query databases. The returned findings are presented through the interface in the form of an interactive graph visualisation. It is able to define a colour and size for certain relationship labels as well as nodes in the graph.

2. Literature survey

Many researchers have proposed articles, algorithms and approaches to create recommender systems all over the world. All the recommender systems have been narrowed down into two types. The recommendation architecture that followed was initially filtered out using the content of the product/video/movie/book by having a complete text analysis on the description of the item. Occasionally, they combined the filtering results with ratings by users who were already using them [4]. This gave a very quick jump start to the recommendation system business, but the responsiveness and accuracy of the results did not catch up with the increase in dynamic user data collected by the system over time.

This gave birth to another recommendation architecture known as collaborative filtering. These researchers used a vector system and defined a user/customer through the vector of items (N-dimensional for distinct N items). A way to differentiate between the positive and negative ratings over the item which is purchased by any user is to set positive values for good ratings on the rating component of the vector and vice-versa. One major differentiation between the two techniques followed by recommender systems is that collaborative filtering's algorithm strives to make the less utilized/purchased item components more accessible and relevant to the user [5].

Collaborative filtering does not come with perfect results. The two commonly faced problems are described below:

Sparsity: With the increase in product/movie or item data in general, similarity computation becomes a costly process, and for such huge data, we obviously cannot rely on the index calculated on any sample. This also becomes a setback when we do not find any overlap between such products, thus rendering the degree of similarity undefined [6].

The cold-start problem: The whole architecture to give us the required recommendations depends upon the user's previous behaviour, the preference of items for each of the users, and the kind of items that the user tends to appreciate and give a positive rating for. Now, when a new user vector gets into the system without a history backing the user, the system will fail to provide us with the recommendations and, thus, the name cold start issue.

Recommender Systems have an impact on e-commerce sales in the following ways: [7]

Browsers-into-buyers: Online users visiting a website often browse through the content without purchasing any items. Recommender Systems take users' preferred choice while logging in and, by studying their behaviour, recommend items they might be willing to purchase [8].

Enhances cross-selling: The Recommender system helps improve cross-selling by increasing the number of suggested items to the user, which will make users more tempted to add more products to the cart or order. Thus, this helps an e-commerce site to gain more revenue [9].

Shopping enjoyment: It is one of the major reasons why users are attached to a certain e-commerce platform [10]. As more and more user data(user behaviours, preferences, ratings, shopping patterns, other user's interpersonal attributes like the size of clothes that are recommended also depends on the size we purchased prior to this order and feedback) are fed into the systems, the more effective they become in giving us relevant items according to the users. Customers are more likely to return to the sites that help them pick out their wardrobe according to their preferences [11].

In this paper, the users are given an option to enter their preferred book genre when they register to the system. Based on his favourite genre and the top-rated books in that genre, our recommendation engine will output relevant/recommended items to each user [12].

3. Neo4j Database

Neo4j is one of the most popular open-source graph databases on the market. The Query language used for manipulating the graph data is CQL (Cypher Query Language), which is similar to SQL [13].

Indexes: We can describe an Index as a kind of data structure that can be added for every node as an attribute. The primary reason for using Indexes is to increase the speed of all data retrieval and storage operations [14]. This is one of the reasons we opted for Neo4j to scale our data, and still, we can expect quick query results [15]. Whenever an index is created, neo4j creates the same redundant copy of the whole data in the data. We may add an index whenever we find some query is taking a lot of time. Any Node with a designated label can be assigned an index: CREATE INDEX ON: Book (book_name) [16].

Property Graph Data Model: A Property Graph is said to increase the readability of the graph and store important properties or related features with the node itself. Any Graph is said to be a property graph when the nodes and relationships have descriptive names and some other related attributes [17].

Adding UNIQUE constraints on the database is easier with naming/having properties associated with nodes and relationships, too.

ACID Rules: It is said that neo4j completely follows the ACID Property, thus giving us Consistency and familiarity in writing transactional queries [18].

4. Methodology

It is necessary to import the metadata into the Graph Database, also known as Neo4j, in order to develop a recommendation system. The user data is retrieved from a web page and then loaded into the graph at the same time. After the integrated graph has been constructed, recommendation queries are executed on it to obtain the best recommendations based on the category and ratings [19].

4.1. Dataset

The dataset includes bibliographic information for 16,559 books that was taken from Freebase. The book ID, title, author, genre, and rating are all contained inside this metadata. Because the dataset has not been cleaned, it is possible that the quality of the suggestions will suffer as a result. Following the completion of the cleaning process, this dataset will be ready to be put into the graph database.

4.2. Data cleaning

The most significant issue that was discovered with these data was that it lacked genre value information. Due to the fact that a content-based recommendation system is now under consideration, genres are one of the essential values necessary for suggestions. There are several different approaches to dealing with missing values. If these were quantitative data, we could do something called statistical imputations to fill in the gaps where there are blanks in the dataset [20]-[24]. Because the genre column is a form of categorical data, we have utilised Python code to eliminate the entries that were found to be lacking the appropriate categories (Figures 1 and 2).

```
In [7]:  import numpy as np
         import pandas as pd

         book=pd.read_csv('Desktop/bookssum.csv');

         book.isnull().sum()

Out[7]:  Wikipedia ID           0
         Freebase ID            0
         Book title             0
         Book author         2382
         Publication date    5610
         Genres              3718
         Plot summary           0
         dtype: int64
```

Figure 1: Dataset before it was cleaned

```
In [10]:  import numpy as np
          import pandas as pd

          book=pd.read_csv('Desktop/bookssum.csv');

          book=book.dropna(axis=0)

          book.isnull().sum()

Out[10]:  Wikipedia ID        0
          Freebase ID         0
          Book title          0
          Book author         0
          Publication date    0
          Genres              0
          Plot summary        0
          dtype: int64
```

Figure 2: Dataset after the removal of NULL values

4.3. Books and Genre nodes

The book's metadata is loaded into Neo4j. Each book and genre is represented by a node in the graph, as shown in the figure 3. The book node has properties such as *book_id*, *title*, *author name*, *average rating and summary* of the book. The genre nodes have a property of *genre name (Figures 3 to 5)*.

```
using periodic commit load csv from 'file:///bookssum.csv' as row with row,
split(row[4],",") as genres unwind genres as genre
merge(b1:books{title:row[1],author:row[2],rating:row[5],summary:row[6]})
merge(b2:genre{genres:genre}) return b1,b2
```

Figure 3: Cypher query for creating nodes

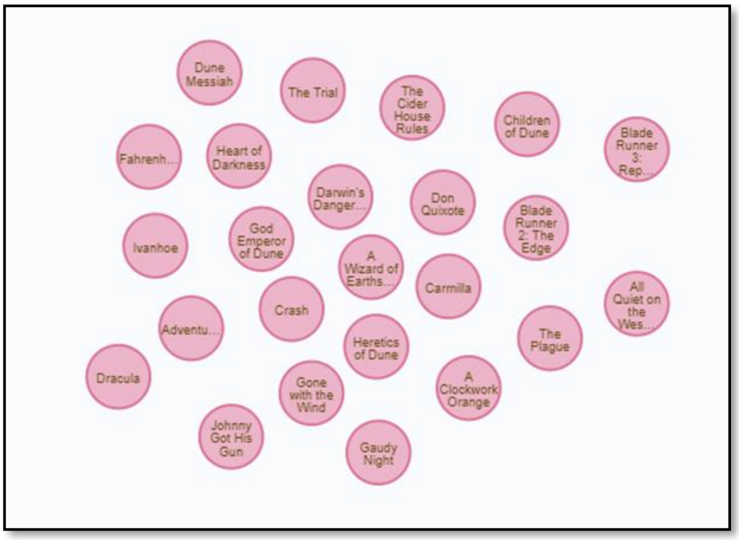


Figure 4: Nodes representing books

Figure 5: Nodes representing genres.

4.4. Relationships

Using the genre list in the dataset, the nodes have been split into books and genres. Now, a relationship is defined between these books and their respective genres, as depicted below (Figures 6 and 7).

```
$ create (b1:books{title:b1.title})-[r:BELONGS_TO]->
(b2:genre{genres:b2.genres}) return b1,r,b2
```

Figure 6: Cypher query to define relationships

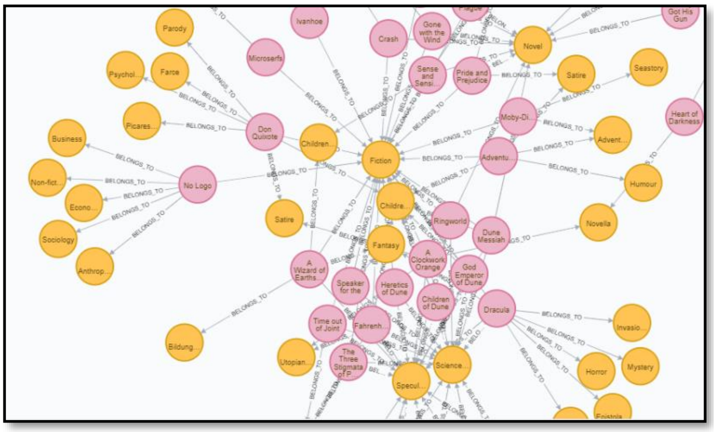


Figure 7: Graph in Neo4j showing the relationships

A. User Input

  a. After loading the metadata, the user data from the Flask web application is to be loaded into the database.
  b. The user data during the registration phase includes name, password and three favourite genre names.
  c. With this, we create the following entities in the graph:

    i. The "User" node has two properties: name and password.
    ii. Relationship "likes" between the user and the 3 Genre nodes already present in the database.

  d. With the given username and password, the given user will be able to log in to the application.

B. Connection between HTML and Neo4j using Python

  a. The Python package py2neo v4.x is used for all the graph-related functions.
  b. The following command helps in establishing the connection between the application and the database :

```
graph = Graph (uri=" bolt://localhost:7687", user=neo4j, password=password)
```

  c. For creating a new node, say User node:

```
user = User () // User is a class created with 2 properties name and password user.name = self.username
    user.password = password
    graph.push(user)
```

  d. For creating relationships, let's say between User node and Genre Nodes :

```
user_genre = Relationship (user.__ogm__.node,'likes',given_genre.__ogm__.node)
graph.create (user_genre)
```

  e. For running direct queries on the database and retrieving recommendation details:

```
query='''
MATCH (b:books {title:b.title})-[r:BELONGS_TO]->(g:genre{genres:g.genres}),
(u:User{name:u.name})-[r1: likes]-> (g1:genre {genres:g.genres})
where u.name={username} and toFloat(b.rating) > 4 AND g.genres=g1.genres and not (u)-[:read]->(b)
return distinct b.title'''
result = graph.run(query,username=self.username)
df = DataFrame(result)
return df[0]
```

Neo4j graph database has its own querying language known as cypher Language now when cypher queries are executed on the neo4j browser or from any external source, like in our case, python (flask) neo4j returns data in a data structure named Cursor [25]-[27].

Cursors are defined through the indexes that neo4j uses for each node for faster operations [28]-[32]. There are many py2neo functions to access the cursor output values; one of them is to convert it into a Python data frame, and they access these values in the view part of the application [33].

4.5. User and favourite genre nodes

Once the input has been taken from the user, a graph is created in Neo4j that shows the relationship between the users and their favourite genres (Figures 8 to 12) [34].

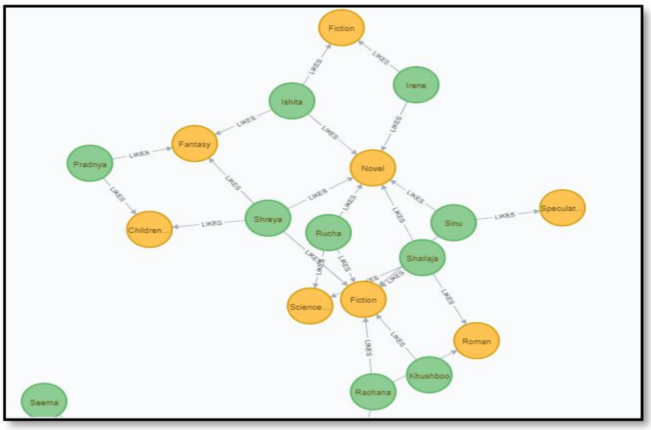


Figure 8: Graph in Neo4j representing users and their favourite genres

C. Recommendation queries

a. Matching using genres

```
MATCH (b:books{title:b.title})-[r:BELONGS_TO]->(g:genre{genres:g.genres}),
(u:user{name:u.name})-[r1:LIKES]->(g1:genre{genres:g.genres})
where g.genres=g1.genres return b
```

Figure 9: Matching using genres

b. Matching using genres and books read by users

```
MATCH (b:books{title:b.title})-[r:BELONGS_TO]->(g:genre{genres:g.genres}),
(u:user{name:u.name})-[r1:LIKES]->(g1:genre{genres:g.genres})
where u.name='Rucha' AND g.genres=g1.genres and not (u)-[:HAS_READ]->(b)
return distinct b.title
```

Figure 10: Matching using genres and books read by users

c. Matching using the highest rating

```
MATCH (b:books{title:b.title})-[r:BELONGS_TO]->(g:genre{genres:g.genres}),
(u:user{name:u.name})-[r1:LIKES]->(g1:genre{genres:g.genres})
where u.name='Rucha' and toFloat(b.rating) > 4 AND g.genres=g1.genres and not
(u)-[:HAS_READ]->(b)
return distinct b.title
```

Figure 11: Matching using the highest rating

Algorithm

a. Start
b. Input **User** nodes as users, **Genre** nodes as genre, and **Book** nodes as books.
c. Input belongs_to relationship as belongs, read relationship as read and ikes Relationship as likes
d. Declare result_nodes
e. Input user_name node for a particular user
f. Input user_name's three favourite genres as fav_genres
g. Repeat the following steps for all three fav_genres
    a. tempNodes ← return books where book-(belongs_TO)→fav_genres is true
    b. tempNodes ← return temp_nodes where not temp_node –(:read)→user_name
    c. tempNodes ← return temp_nodes where temp_node rating > 4
h. Push tempNodes in result_nodes
i. Return temp_nodes
j. End

4.6. Data Model

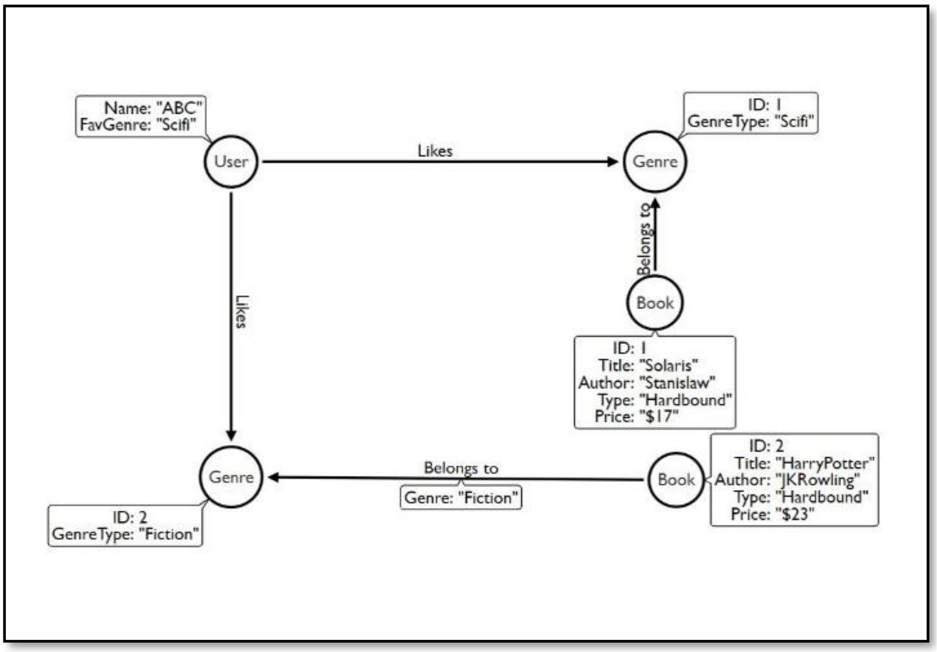


Figure 12: Data Model Diagram for Recommendation System

5. Result

The methodology designed to implement a book recommendation system gives recommendations based on the user's favourite genres and top-rated books. The final graph in neo4j is shown in below figure 13:

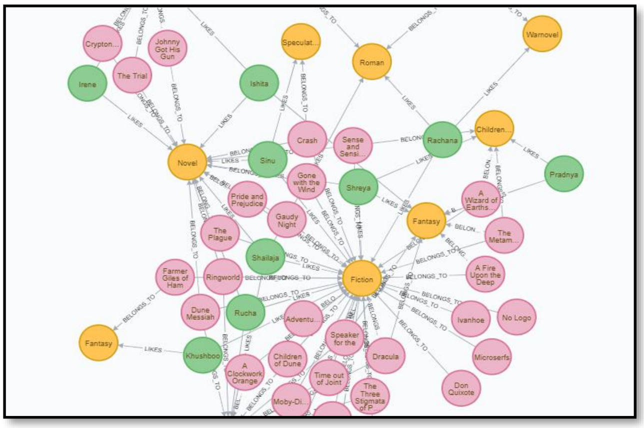


Figure 13: Graph containing all the nodes

Recommendation Queries are fired on the above graph to generate personalized, highly recommended books for the user (Figures 14 and 15).

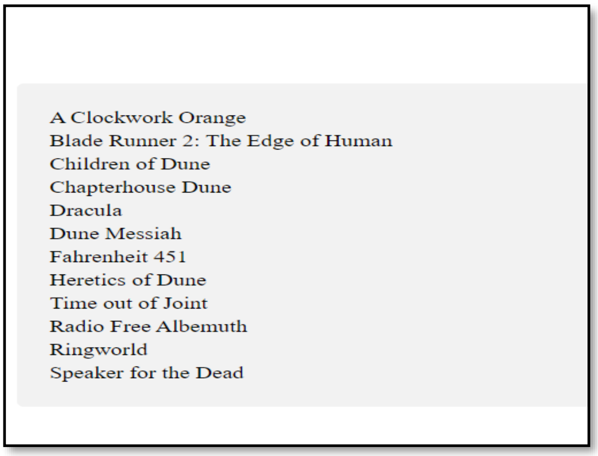
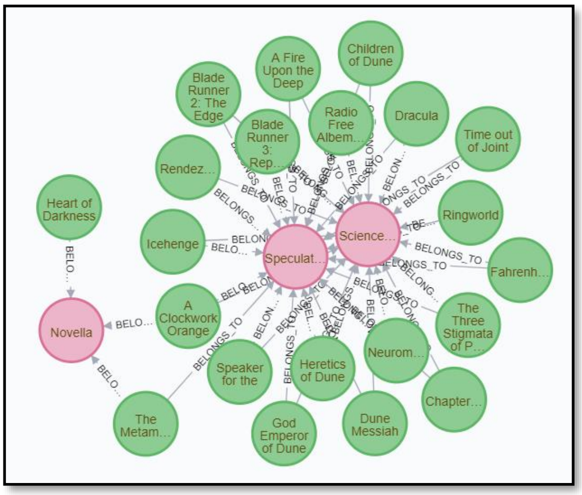


Figure 14: Recommendation output on web-application

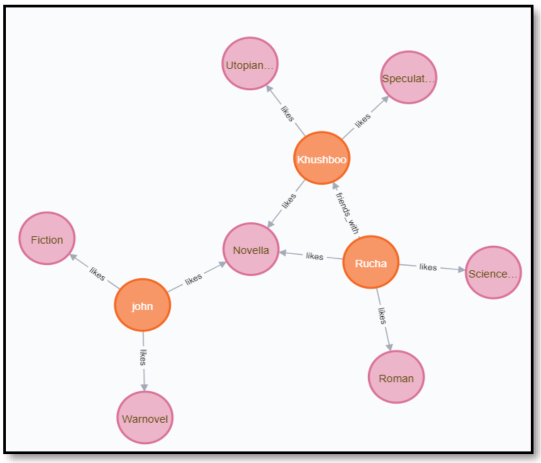
a. belongs_to -This relationship is between genre and books and is created while loading the metadata.



(a)

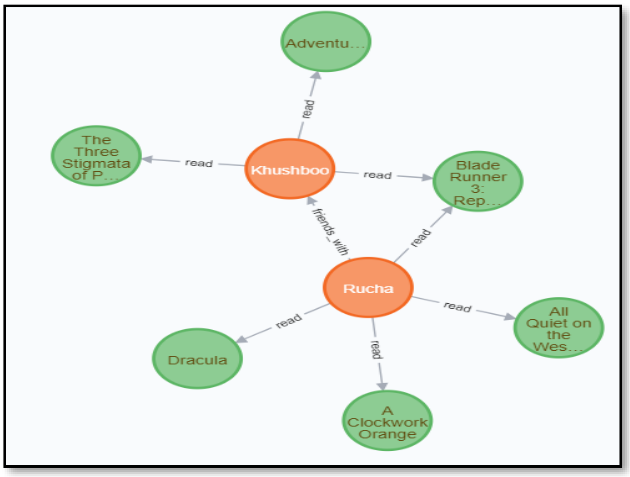
b. Likes – The relationship is defined between a User node and their three favourite genre nodes.
c. Read– The relationship is created between a user node and a book node.

The problem with most of the recommendation systems is that even after the product is purchased for the product recommendation system, the product is displayed as a recommendation to the user [35]. We resolved this issue by implementing a recommendation query that checks for all the books that have a 'read' relationship with the given user.



(b)

d. friends_with– This relationship is defined between two users. The aim of creating this was to have recommendations of the connected friends based on the common genre, such that we can define friend recommendations for every user. This feature is proposed as the future scope of our recommendation system.



(c)

Figure 15: Defining the relationships

6. Conclusion

The primary objective of this project was to construct from the ground up an advanced recommendation system that is both dynamic and effective. A system that is capable of giving out quick responses despite the vast volume of data being processed. In order to fix the problem, the recommender engine continues to give the user the same recommendation even after the purchase of the item has been successfully completed. In our case, we have taken measures to ensure that the user will not be given recommendations for books that they have previously read. The flask application is what's responsible for storing the user data in the database. The entirety of the graph will consist of four distinct varieties of relationships between the three Node Labels of User, Genre, and Books. The primary advantage of utilising Neo4J is that it records the behaviour of data in addition to capturing the relationship between the data and the similarity between items on the basis of the type and the preference of the user. In addition, Neo4J captures the similarity between items. The recommendation engine is susceptible to being influenced by a wide range of elements; among these are the users' preferred genres and the kinds of goods that are comparable to those genres. Because it is a graph database, it is simple for anyone to examine and visualise the relationships that exist between the various nodes, and it also makes it simple to handle a very large quantity of data.

Acknowledgement: Our friends who patiently supported us during the research are likewise appreciated.

Data Availability Statement: This study's data, graphics, and code are accessible from the associated author upon reasonable request.

Funding Statement: No funding was received to conduct the research.

Conflicts of Interest Statement: Authors collaborate on this work and agree on its points, issues, and discoveries.

Ethics and Consent Statement: All authors agree to make this paper available for reading, teaching, and learning.

References

1. F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, “Recommendation systems: Principles, methods and evaluation,” Egypt. Inform. J., vol. 16, no. 3, pp. 261–273, 2015.
2. F. Ricci, L. Rokach, B. Shapira, and B. Paul, Kantor “Recommender Systems Handbook. Springer, 2010.
3. G. Linden, B. Smith, and J. York, “Amazon.com recommendations:item-to-item collaborative filtering,”IEEE Internet Computing, vol. 7,no. 1, pp. 76–80, 2003.
4. M. Balabanovic and Y. Shoham, “Fab: content-basedcollaborative recommendation,” Communications of the ACM, vol. 40, no. 3, pp. 66–72, 1997.
5. J. S. Breese, D. Heckerman, and C. Kadie, “Empirical analysis of predictive algorithms for collaborative filtering,” Proc. 14th Conf. Uncertainty in Artificial Intelligence, MorganKaufmann, pp. 43-52, 1998.
6. M. Grčar, D. Mladenič, B. Fortuna, and M. Grobelnik, “Data sparsity issues in the collaborative filtering framework,” in Advances in Web Mining and Web Usage Analysis, Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 58-76.
7. Z. Ying, C. Caixia, G. Wen, and L. Xiaogang, “Impact of recommender systems on unplanned purchase behaviours in e-commerce,” in 2018 5th International Conference on Industrial Engineering and Applications (ICIEA), 2018.
8. A. I. Hariadi and D. Nurjanah, “Hybrid attribute and personality based recommender system for book recommendation,” in 2017 International Conference on Data and Software Engineering (ICoDSE), 2017.
9. A. Virk and R. Rani, “Efficient approach for social recommendations using graphs on Neo4j,” in 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), 2018.
10. M. Salehi and I. N. Kmalabadi, “A hybrid attribute–based recommender system for E–learning material recommendation,” IERI Procedia, vol. 2, pp. 565–570, 2012.
11. A. Ldbyani and M. H. A. Al-Abyadh, “Relationship between Dark Triad, Mental Health, and Subjective Well-being Moderated by Mindfulness: A Study on Atheists and Muslim Students,” Islamic Guidance and Counseling Journal, vol. 5, no. 1, pp. 71–87, 2022.
12. D. K. Sharma, B. Singh, M. Raja, R. Regin, and S. S. Rajest, “An Efficient Python Approach for Simulation of Poisson Distribution,” in 2021 7th International Conference on Advanced Computing and Communication Systems, 2021.
13. D. K. Sharma, B. Singh, R. Regin, R. Steffi, and M. K. Chakravarthi, “Efficient Classification for Neural Machines Interpretations based on Mathematical models,” in 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), 2021.
14. D. K. Sharma, N. A. Jalil, R. Regin, S. S. Rajest, R. K. Tummala, and Thangadurai, “Predicting network congestion with machine learning,” in 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), 2021.
15. D. Uike, S. Agarwalla, V. Bansal, M. K. Chakravarthi, R. Singh and P. Singh, "Investigating the Role of Block Chain to Secure Identity in IoT for Industrial Automation," 2022 11th International Conference on System Modeling & Advancement in Research Trends (SMART), Moradabad, India, pp. 837-841, 2022.
16. F. Arslan, B. Singh, D. K. Sharma, R. Regin, R. Steffi, and S. Suman Rajest, “Optimization technique approach to resolve food sustainability problems,” in 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), 2021.
17. G. A. Ogunmola, B. Singh, D. K. Sharma, R. Regin, S. S. Rajest, and N. Singh, “Involvement of distance measure in assessing and resolving efficiency environmental obstacles,” in 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), 2021.

18. H. A. H. Abdel Azeem and M. H. A. Al-Abyadh, “Self-compassion: the influences on the university students’ life satisfaction during the COVID-19 outbreak,” Int. J. Hum. Rights Healthc., vol. ahead-of-print, no. ahead-of-print, 2021.
19. K. Sharma, B. Singh, E. Herman, R. Regine, S. S. Rajest, and V. P. Mishra, “Maximum information measure policies in reinforcement learning with deep energy-based model,” in 2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), 2021.
20. K. Venkitaraman and V. S. R. Kosuru, “Hybrid Deep Learning Mechanism for Charging Control and Management of Electric Vehicles”,” EJECE, vol. 7, pp. 38–46, 2023.
21. M. H. A. Al-Abyadh and H. A. H. Abdel Azeem, “Academic achievement: Influences of university students’ self-management and perceived self-efficacy,” J. Intell., vol. 10, no. 3, p. 55, 2022.
22. N. A. S. Al-Abrat and M. H. A. Alabyad, “The Extent of Awareness of Faculty Members at Al-bayda University About the Concept of Educational Technology and Their Attitudes Towards It,” in New Trends in Information and Communications Technology Applications. NTICT 2021, vol. 1511, A. M. Al-Bakry, Ed. Cham: Springer, 2021.
23. Q. An, W. C. Hong, X. S. Xu, Y. Zhang, and K. Kolletar-Zhu, “How education level influences internet security knowledge, behaviour, and attitude: A comparison among undergraduates, postgraduates and working graduates,” International Journal of Information Security, vol. 22, no. 2, pp. 305–317, 2022.
24. S. Sonnad, M. Sathe, D. K. Basha, V. Bansal, R. Singh and D. P. Singh, "The Integration of Connectivity and System Integrity Approaches using Internet of Things (IoT) for Enhancing Network Security," 2022 5th International Conference on Contemporary Computing and Informatics (IC3I), Uttar Pradesh, India, pp. 362-366, 2022.
25. V. Bansal, S. Pandey, S. K. Shukla, D. Singh, S. A. Rathod and J. L. A. Gonzáles, "A Frame Work of Security Attacks, Issues Classifications and Configuration Strategy for IoT Networks for the Successful Implementation," 2022 5th International Conference on Contemporary Computing and Informatics, Uttar Pradesh, India, pp. 1336-1339, 2022.
26. V. Gunturu, V. Bansal, M. Sathe, A. Kumar, A. Gehlot and B. Pant, "Wireless Communications Implementation Using Blockchain as Well as Distributed Type of IOT," 2023 International Conference on Artificial Intelligence and Smart Communication (AISC), Greater Noida, India, pp. 979-982, 2023.
27. V. S. R. Kosuru and A. K. Venkitaraman, “Developing a Deep Q-Learning and Neural Network Framework for Trajectory Planning”,” EJENG, vol. 7, pp. 148–157, 2022.
28. V. Suthar, V. Bansal, C. S. Reddy, J. L. A. Gonzáles, D. Singh and D. P. Singh, "Machine Learning Adoption in Blockchain-Based Smart Applications," 2022 5th International Conference on Contemporary Computing and Informatics (IC3I), Uttar Pradesh, India, pp. 372-378, 2022.
29. W. C. H. Hong, “Improving English as a foreign language learners’ writing using a minimal grammar approach of teaching dependent clauses: A case study of Macao secondary school students,” in Innovative Approaches in Teaching English Writing to Chinese Speakers, B. L. Reynolds & M. F. Teng, Eds. De Gruyter Mouton, pp. 67–90, 2021.
30. W. C. H. Hong, “Macao Secondary School EFL Teachers' Perspectives on Written Corrective Feedback: Rationales and Constraints,” Journal of Educational Technology and Innovation, vol. 4, no. 4, pp. 1–13, 2021.
31. W. C. H. Hong, “The Effect of Absence of Explicit Knowledge on ESL/EFL Stress-Placement Accuracy: A quasi-experiment,” Asian EFL Journal, vol. 20, no. 2, pp. 262–279, 2018.
32. W. C. H. Hong, “The impact of ChatGPT on foreign language teaching and learning: Opportunities in education and research,” Journal of Educational Technology and Innovation, vol. 5, no. 1, pp. 37–45, 2023.
33. W. C. Hong, C. Y. Chi, J. Liu, Y. F. Zhang, V. N.-L. Lei, and X. S. Xu, “The influence of social education level on cybersecurity awareness and behaviour: A comparative study of university students and working graduates,” Education and Information Technologies, vol. 28, no. 1, pp. 439–470, 2022.
34. X. Lian, W. C. Hong, X. Xu, K.-Z. Kimberly, and Z. Wang, “The influence of picture book design on visual attention of children with autism: A pilot study,” International Journal of Developmental Disabilities, pp. 1–11, 2022.
35. X. Xu, W. C. Hong, Y. Zhang, H. Jiang, and J. Liu, “Learning paths design in personal learning environments: The impact on postgraduates’ cognitive achievements and satisfaction,” Innovations in Education and Teaching International, pp. 1–16, 2023.