

Hybrid Personalized Book Recommender System Based on Big Data Framework

Fan Liu, Suriya Priya R. Asaithambi, Ramanathan Venkatraman

NUS-ISS, National University of Singapore, Singapore

liufan@nus.edu.sg, suria@nus.edu.sg, rvenkat@nus.edu.sg

Abstract—With the rapid growth of e-books and e-libraries, it becomes difficult for people to find right books from the flood of information. Personalized recommender system has been developed to help users to find appropriate books based on user's ratings and interests. Most of the existing book recommender systems are based on content-based or collaborative filtering algorithms. However, those recommender systems provide book recommendations only based on books' contents and users' personal interests. This paper proposes a hybrid personalized book recommender system which combines both book popularity and personalized interests to recommend books for users. Book popularity is identified by the collected tweets to indicate current reading trends. The book popularity is represented by Lucene score of each book based on the collected tweets. The proposed book recommender system combines model-based recommender algorithms such as item-based and Alternating Least Squares (ALS) with book popularity. A scalable architecture using big data framework is developed to ingest, store, process and analyze both batch and streaming data. Open-source big data tools such as Apache Solr, Apache Kafka, and Apache Spark MLlib are used to implement the proposed hybrid book recommender system. PySpark is used to implement data pre-processing and big data analysis. The ALS recommender algorithm is implemented by using Apache Spark MLlib. The results show that the accuracy of the proposed hybrid book recommender system is acceptable.

Keywords – Hybrid recommender algorithm, Big data framework, Collaborative filtering, Alternating Least Squares (ALS), PySpark

I. INTRODUCTION

Recommender systems are a subclass of information filtering techniques that attempt to predict “rating” or “preference” that a user would give to an item (such as movies, music, books, websites and news etc.) [1]. It becomes an important factor for the success of e-commerce and other IT industries nowadays. It gradually becomes popular in various application domains such as e-commerce recommender systems [2][3], e-learning recommender systems [4][5] and e-book recommender systems [6][7][8].

Three main algorithms are widely used to build recommender systems [25]: (1) collaborative filtering algorithm (2) content-based algorithm and (3) hybrid algorithm. Firstly, the collaborative filtering algorithm is the most widely used and the most matured technique. It is a process of filtering information or pattern based on the similarity between items. Secondly, content-based algorithm is based on the matching of user profile and some specific

characteristics of an item. Lastly, the hybrid algorithm combines the features of two or more recommender techniques to overcome the drawbacks of one recommender algorithm and obtain the advantages of different recommender techniques.

Recommender systems have been applied in various IT industries. Some companies such as Amazon, Netflix and Google are already utilizing the power of recommender systems in building and enhancing their new and existing systems.

Amazon has developed an item-to-item collaborative filtering algorithm to match each user's purchased and rated items to similar items, then combined those similar items into a recommendation list. This approach solved recommendation problems with the use of an algorithm different from the traditional recommender algorithms. The item-to-item collaborative filtering algorithm used in Amazon's online shop computations was scalable regardless of the number of customers or items in the product catalogue [9].

Netflix provides a subscription service model that offers personalized recommendations to help users to find shows and movies of users' interest. It developed a proprietary and complex recommender system to provide personalized recommendations to users based on metrics such as major actors or genre [10].

Recommender systems are also widely used in book recommendation [6]-[10]. Numerous book recommender systems have been developed to provide personalized recommendation services to help users to select the most appropriate books [6]-[8][10][11]. It can also recommend books to the librarians for book acquisition so that the library system can provide more effective recommendations based on readers' interests [24].

Currently, the rapid growth in data has resulted in a new era of information known as the era of big data. Every day, a large amount of data, both structured and unstructured are generated via the Internet. How to use those data to create a more innovative, more efficient and more effective book recommender system becomes a promising and challenge research direction. Recently, many researchers have begun to use the real-time social media data, such as posts and tweets that generated by Facebook and Twitter, to identify the popularity and current trends, as well as to make popularity-based recommendations [21].

This paper proposed a hybrid personalized book recommender system that combines different recommender algorithms such as item-based and Alternating Least Squares (ALS) as well as the books' popularity. Relevant tweets are collected to prioritize the book recommendation. The book popularity is represented by Lucene score [12] of each book based on the collected tweets. A scalable architecture using big data framework is developed to ingest, store, process and analyse both batch and streaming data. Open-source big data tools such as Apache Solr, Apache Kafka, and Apache Spark MLlib are used to implement the proposed hybrid book recommender system. We provide an overall architecture of the proposed big data solution to implement book recommendation.

The rest of the paper consists of the following sections. Section II describes the background of book recommender systems and introduces different recommendation techniques found in literature and their limitations. Section III conducts the exploratory data analysis on the book-crossing rating datasets. Section IV describes the details of the proposed hybrid book recommender systems. Section V describes the implementation of the proposed hybrid book recommender system using a big data framework and discusses the implementation results. Section VI provides conclusions as well as future improvements.

II. BACKGROUND AND RELATED WORK

A. Background

With the rapid growth of e-books and e-libraries, readers increase demands to find desired books from the flood of information. Recommender system plays an important role in information filtering and provides personalized suggestions for readers [6] [11]. Book recommender systems must recommend appropriate books that are of buyers' interests and current book reading trends. Many academic libraries have implemented personalized recommender systems to assist readers in finding relevant books and making better book selection [6]-[10].

B. Related Work

Numerous book recommender systems have been developed using different recommendation techniques [6]-[10]. Most of these systems use different information filtering techniques such as collaborative filtering [7] [9] or content-based filtering [11] for book recommendations.

Okon [7] designed and developed an improved recommendation model by using collaborative filtering algorithm as well as an efficient quick sort algorithm to make efficient book recommendation. This system was implemented by using a real-time, cloud-hosted NOSQL database. Amazon [9] recommended books based on the users' purchase patterns. It uses collaboration filtering algorithm to provide a personalized online store experience to its customers. Although collaborative filtering is a popular technique for book recommendation, the major drawbacks of a collaborative recommender are cold start [14] and sparsity [16] if the users or items are new to the system.

Content-based recommendation systems recommend items by matching item contents and features against user's previous reading choices and preferences [13]. Furthermore, content-based technique requires that items be represented in terms of features that are automatically extracted from these items, such as term frequency for representing text documents and genre for representing a movie [11]. Content-based recommenders,

on the other hand, suffer from overspecialization and limited content analysis. For example, it cannot differentiate between good books or bad books if both of them are using the same terminology. Another major drawback of content-based recommenders is that they ignore current users' ratings when recommending new items, and they have low accuracy in the book recommendation [6].

Different filtering techniques can be combined to develop hybrid book recommender systems. Using hybrid techniques, it can overcome the drawbacks of one recommendation technique and obtain advantages of different recommendation techniques [8] [10] [11].

Chandak [8] proposed a book recommender system that combines the features of collaborative filtering and content-based filtering in a hybrid way. It combines recommendations generated by two techniques. Furthermore, the cold start problem for new users and new items has been overcome by using demographic attributes to find user co-relationships and provide appropriate recommendations. Zafar [11] created a hybrid book recommender system that recommends relevant books based on book contents, item-item collaborative filtering and association rule mining.

However, those recommender systems did not consider the current reading trends when making book recommendations. How to use social media information to detect current reading trends is an interesting research direction. Another critical issue of current recommender systems is the scalability of algorithms with large amounts of data. Dealing with massive amounts of dynamic data generated by online interactions between users and online book platforms becomes difficult.

In this paper, we design and develop a hybrid personalized book recommender system that combines personalized recommender algorithms such as item-based filtering and ALS with book popularity. To identify current reading trends, we derive book popularity from tweets collected from social media platform such as Twitter. To implement the proposed hybrid book recommender system, we use open-source big data frameworks such as Apache Kafka, Apache Spark, Apache Spark MLlib and Apache Solr [18].

III. DATA SOURCE EXPLORATION

In this section, we will describe various data sources and conduct exploratory data analysis on the book crossing rating datasets in this section.

A. Data Sources

There are two data sources:

- Book Dataset – Book crossing rating data and book description data are collected from Kaggle book repository [16].
- Social Media Data – Tweets are collected via Twitter developer API.

The book dataset was retrieved from Kaggle book dataset repository [16]. This dataset consists of three tables named BX-Book-Rating.csv, BX-Users.csv and BX-Books.csv. It has 278,858 users (anonymized but with demographic information) who have given 1,149,780 ratings (including both explicit and implicit ratings) to 271,379 books. The description of the dataset is as follows:

- BX-Book-Rating.csv – it is a typical ratings data that has three attributes: user_ID, ISBN and Book_Rating. This data set is the main data source which is used to build the book recommender system. Sample data is shown in Table I.
- BX-Users.csv – it contains the users’ demographic information. It has three attributes: user-ID, Location and Age. Sample data is shown in Table II.
- BX-Books.csv – it contains the information of each book including ISBN, Book-Title, Book-Author, Year-of-Publication, Publisher, Image-URL-S, Image-URL-M and Image-URL-L. The last three columns are removed as it won’t be used in our recommender algorithms. Sample data is shown in Table III.

Another dataset is used to supplement the book-crossing rating dataset mentioned above. The book description dataset includes book information about book title, description, summary, and category [17]. It will be used to query the collected tweets to obtain the Lucene score for each book. Three columns are chosen: ISBN, Book-Title and description. Sample data is shown in Table IV.

TABLE I. BOOK RATING

User-ID	ISBN	Book-Rating
276725	034545104X	0
276726	0155061224	5
276727	0446520802	0

TABLE II. USERS

User-ID	Location	Age
276725	tyler, texas, usa	NaN
276726	seattle, washington, usa	NaN
276727	new south wales, australia	16.0

TABLE III. BOOKS

ISBN	Book-Title	Book-Author	Year-Of-Publication	Publisher
0195153448	Classical Mythology	Mark P. O. Morford	2002	Oxford University Press
0060973129	Decision in Normandy	Carlo D'Este	1991	Harper Perennia

TABLE IV. BOOK DESCRIPTION

ISBN	Book-Title	Description
184018907 X	Soldier Five : The Real Truth About The Bravo Two Zero Mission	SOLDIER FIVE is an elite soldier's explosive memoir of his time within the Special Air Service (SAS)...
1784183725	A Travel Guide to the Seven Kingdoms of Westeros	Bustling cities, ancient wonders and snow-peaked mountain ranges; the Seven Kingdoms of Westeros are..

The streaming tweets are extracted by using Twitter developer API. The tweet object provided by the Twitter developer API is a complex object which has more than 20 fields with nested fields inside.

Four fields are selected to calculate the Lucene score. The description of the four fields is as follows:

- created_at – time when Tweet was created.
- text – the tweet text
- extended_full_text – the “full_text” filed that contains the complete un-truncated Tweet message when longer than 140 characters.
- retweeted_full_text – contains the “full_text” if tweet is a retweet.

Before we build our hybrid book recommender system, we need to implement data pre-processing as the raw data contains lots of invalid and incomplete records. Relevant attributes are selected based on the algorithms’ requirements.

The following data preprocessing steps are implemented on the raw dataset:

- Convert data types
- Replace null values
- Replace invalid values
- Remove duplicate records
- Remove unnecessary columns
- Remove extra characters from ISBN attribute

B. Exploratory Data Analysis

After data pre-processing, the cleansed datasets are merged into a single dataset named as “combined book rating”. This is the primary data source used to build our hybrid book recommender system. Sample data from “combined book rating” dataset is shown in TABLE V.

Exploratory data analysis is conducted to understand the statistical distribution of the book dataset. The Top 15 number of ratings (exclude 0 ratings) for a book is shown in Figure 1. Book rating distribution is shown in Figure 2.

The recommendation accuracy is represented by RMSE (Root Mean Squared Error), which emphasizes on larger absolute errors. The lower the RMSE is, the more accurate is the recommendation.

RMSE (Root Mean Squared Error): RMSE is given by [23]:

$$RMSE = \sqrt{\frac{1}{n} \sum_{u,i} (p_{u,i} - r_{u,i})^2} \quad (1)$$

where $p_{u,i}$ is the predicted rating of user u on item i , $r_{u,i}$ is the actual rating of user u on item i .

IV. HYBRID PERSONALIZED BOOK RECOMMENDER SYSTEM

The proposed hybrid personalized book recommender system is described in details in this section. Firstly, we present a personalized book recommender system that combines features of collaborative filtering and ALS to recommend books for users. Secondly, we will calculate the Lucene score of each book based on the collected tweets to determine book popularity. Lastly, we combine the book popularity with a personalized recommender system to create a hybrid book recommender system.

TABLE V. COMBINED BOOK RATING

ISBN	Book-Title	Book-Author	Year-of-Publication	Publisher	User-ID	Book-Rating	Location	Age
0002005018	Clara Callan	Richard Bruce Wright	2001	HarperFlamingo Canada	8	5	timmins, ontario, canada	35
0887841740	The Middle Stories	Sheila Heti	2004	House of Anansi Press	8	5	timmins, ontario, canada	35

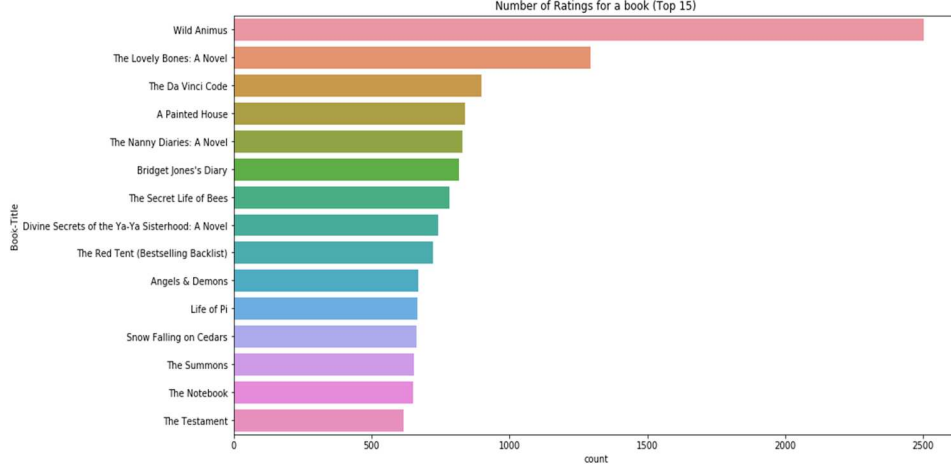


Figure 1. Number of ratings for a book (Top 15)

We compute the similarity between items i and j using the adjusted cosine similarity:

$$sim(i, j) = \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{u \in U} (r_{u,j} - \bar{r}_u)^2}} \quad (2)$$

where $r_{u,i}$ is the rating of user u on item i , $r_{u,j}$ is the rating of user u on item j , $\bar{r}_u$ is the mean rating of user u , U is the set of all users. Each individual rating is within a scale from 0 to 10. Rating 0 indicates that the user has not yet rated this book.

Once we have the similarities between the items i and j , we can predict user u 's rating for an item i using:

$$Predicted\ rating(u, i) = \frac{\sum_{j \in J} sim(i, j) * (r_{u,j})}{\sum_{j \in J} |sim(i, j)|} \quad (3)$$

where J is the set of K similar items to i (among items that user u has rated).

B. Alternating Least Squares(ALS) Algorithm

Item-based is faster than user-based algorithms. There are usually far fewer items than users and the similarities between items change much less frequently than between users hence all item-item distances can be pre-computed. However, as item-based approach relies on calculating the similarities between each book, there will be scenarios where this is not possible since there are insufficient overlapping ratings. For example, for some users, the model is unable to make any or a few recommendations (e.g., less than 5 books). In this case, the rating matrix R is a sparse matrix because items do not receive ratings from a large number of users which leads to R containing many missing values.

To address this issue, ALS approach is used to supplement the item-based approach. One solution to the sparse matrix

A. Item-based Collaborative Filtering

Our proposed personalized book recommender algorithm uses an item-based technique rather than a user-based technique. Since the number of users (40553) is more than the number of books (5647) in the cleansed dataset, user-based collaborative filtering is more likely to have the sparsity problem than the item-based filtering.

One critical step in the item-based collaborative filtering algorithm is to calculate the similarity between items and then to select the most similar items. Here, we use the adjusted cosine-based similarity [15], as cosine-based similarity the magnitude which is more suitable for our dataset.

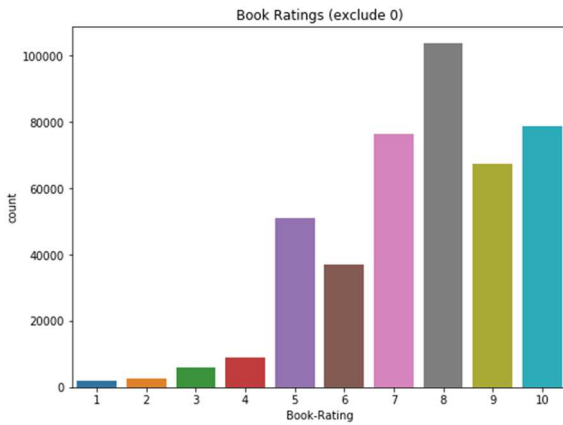


Figure 2. Book rating distribution

problem is ALS matrix factorization [19]. Because the ALS algorithm generates a full matrix, it is always able to make recommendations. The details of the ALS algorithm is described as follows [19].

There are two k dimensional vectors which are referred to as “factor”.

- x_u is k dimensional vectors summarizing’s every user u .
- y_i is k dimensional vectors summarizing’s every item i .

Let,

$$r_{ui} \approx x_u^T Y_i \quad (4)$$

$$x_u = x_1, x_2, \dots, x_n \in R^k \quad (5)$$

$$y_i = y_1, y_2, \dots, y_n \in R^k \quad (6)$$

Equation (1) can be transformed to an optimization problem to find:

$$\text{argmin}_{r_{ui}} (r_{ui} - x_u^T Y_i)^2 + \lambda (\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2) \quad (7)$$

Here, λ is the regularization factor, which is used to address the overfitting issue, is referred as weighted- λ -regularization. The default value of λ is 1. The details of ALS algorithm can be found in [19] [20].

TABLE VI. LUCENE SCORE

ISBN	Score
014034294X	250.1378
60931418	364.5198
60008776	914.1927
385335555	238.1046

C. Lucene Score Calculation

Tweets from Twitter’s public timeline are collected and analyzed to identify the book popularity. Popular books are those that are mentioned frequently in tweets. Apache Solr [18] is used to calculate the Lucene score. Apache Solr is comprised of three main components: an indexer, a Lucene index and a server [21]. To determine the popularity of a book, relevant tweets are collected from Twitter using Twitter Developer API and passed on as queries to the Apache Solr server. The server retrieves the book’s description associated with the tweet content along with a weight that indicates the similarity between the query and the book’s description for each gathered tweet. These books are sorted based on the total weight of their tweets. As a result, the books are ranked according to their popularity (indicated by Lucene score). The more popular the book is, the higher is the Lucene score.

D. Hybrid Book Recommender System

The proposed hybrid book recommender system is shown in Figure 3. The system consists of three modules:

- Model-based book recommender algorithm. It recommends N number of books to a user based on the item-based algorithm and the ALS algorithm.

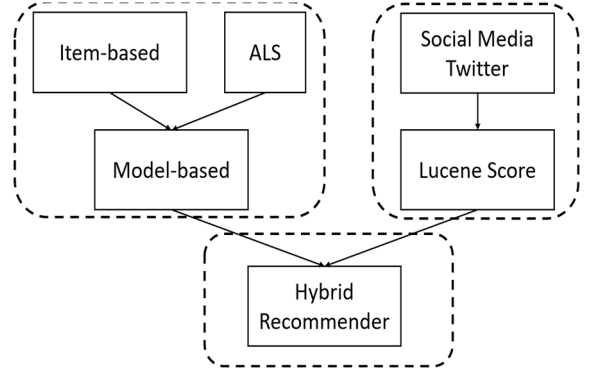


Figure 3. Hybrid recommender system

- Lucene score calculation. It is used to prioritize books that are deemed more popular based on the relevant Twitter tweets.
- Hybrid Recommender. Item-based algorithm recommendations will always come first, followed by the ALS recommendations. The book popularity will be used to sort the recommendations for each algorithm. In order to combine both book popularity and recommender algorithms, the combined book popularity will be calculated as follows:

$$\text{combined ranking} = \text{predicted rating} * \text{Lucene Score} \quad (8)$$

The recommendations for each model will be sorted based on the *combined ranking* which is calculated by (8).

V. BIG DATA IMPLEMENTATION

A. Overall Architecture

The proposed hybrid book recommender system uses a big data framework and Figure 4 depicts its overall architecture. It consists of five major layers: (1) data ingestion (2) data pre-processing (3) data storage (4) data analytics and (5) hybrid recommender system. The following is a detailed description of each layer.

- Data Ingestion. Twitter streaming data is ingested using Apache Kafka. Tweepy, an easy-to-use Python library for accessing the Twitter API, allows us to connect the Twitter API and listen for tweets. Google Colab is used to upload book dataset including book crossing rating and book description files into Google Drive.
- Data Pre-processing. Apache Spark Streaming is used to clean the tweets. Apache Spark Stream is subscribed to the Kafka’s Twitter topic. The original data format is JSON format, we need to define new schema and extract the useful fields by using Spark Streaming. The processed and cleansed tweets will be

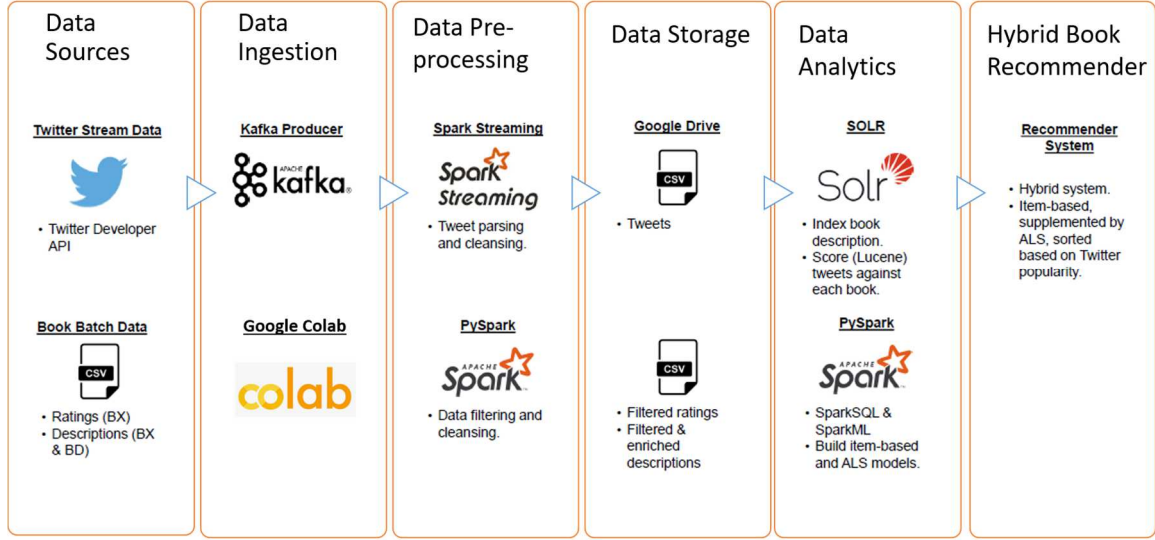


Figure 4. Overall Architecture of Big Data Implementation

exported and save as CSVs. PySpark is used to cleanse, filter and integrate data.

- **Data Storage.** Both the cleansed tweets and book rating files are saved in Google Drive for further use.
- **Data Analytics.** This layer will implement two major tasks: Lucene score calculation and recommender system model building. Apache Solr provides a user-friendly GUI for query test as well as a REST API for query calls. Each books' Lucene scores are saved as a CSV and uploaded onto Google Drive. The item-based collaborative filtering algorithm is combined with a model derived from the ALS algorithm in the hybrid recommender system. Because the Apache Spark MLlib does not provide a ready-made item-based model, we use an item-based algorithm from [22] and make necessary modification to some of the functions to implement the item-based recommender algorithm. The ALS model is implemented by using Apache Spark MLlib.
- **Hybrid Recommender System.** We create a hybrid recommender system by combining the item-based model, the ALS model and the books' popularity scores (Lucene scores). The use of both models can supplement the shortcomings of each individual model. For example, the full matrix generated by ALS partially mitigates data sparsity issues that cause the item-based model to fail to make recommendations. Books' popularity is determined by the Lucene scores which are used to prioritize the book recommendations.

B. Experimental Results

We have created a hybrid personalized book recommender system based on users' interests, ratings and book popularity. We conducted a series of experiments to assess the efficacy of the proposed hybrid book recommender system.

The book description dataset which includes the ISBN and description of each book is uploaded into Apache Solr. The Lucene score of each book is summed up and saved as a CSV file. Sample data of the CSV file is shown in TABLE VI. . The

total dataset is split into a training and test set in a 4:1 ratio. The training set contains 114640 data points while the testing set contains 28589 data points. The training dataset is used to train item-based and ALS models to predict users' ratings in the test dataset. The prediction is evaluated by comparing it to the actual prediction in the test dataset.

Statistical accuracy metrics compare predicted ratings to actual user ratings to determine the accuracy of a filtering technique. Two measure metrics RMSE and MAE are used to evaluate the performance of the recommender algorithm.

MAE (Mean Absolute Error): MAE is the most popular and commonly used measurement metrics to evaluate prediction model. It is a measure of deviation of recommendation from user's specific value. Its computation formula is as follows [23]:

$$MAE = \frac{1}{N} \sum_{u,i} |p_{u,i} - r_{u,i}| \quad (9)$$

where $p_{u,i}$ the predicted rating from user u on item i , $r_{u,i}$ is the actual rating from user u on item i , N is the total number of ratings on the item set. The lower the MAE , the better the recommendation algorithm predicts user's ratings.

We select User-ID = 46671 at random to demonstrate the proposed hybrid book recommender algorithm. The following steps describes the workflow of our hybrid recommender system which combines the item-based, the ALS model and the book popularity derived from the collected tweets.

Step1: Both item-based and ALS models will recommend N number of books for a user. E.g., if $N = 5$, the item-based model will generate 5 recommendations and the ALS will generate 5 recommendations, giving a total of 10 recommendations (assuming there are no duplicated recommendations from these two models).

Step2: Retrieve Lucene score for each recommended book.

Step3: Calculate the *combined ranking* for each recommended book by using (8).

Step4: Sort recommendations based on the following rules:

- Item-based recommendations will always come first, followed by ALS recommendations.
- Then the recommendations from each model will be sorted based on the *combined ranking*.
- Drop duplicate recommendations in the ALS model if it is already recommended by the item-based model.

The RMSE for the item-based model is 1.7670 and the MAE is 1.1331, whereas the RMSE for the ALS model is 2.0047 and the MAE is 1.6421. The item-based model performs slightly better than the ALS. The 5 books recommended by item-based model combined with the book popularity is shown in TABLE VII. The 5 books recommended by ALS model combined with book popularity is shown in TABLE VIII. The final recommendations from our hybrid personalized recommender system are shown in TABLE IX. We can see that there is no duplicated recommendations. Item-based recommendations will always be preferred over ALS recommendations because the RMSE and MAE of the item-based model are lower than those of the ALS model.

TABLE VII. ITEM-BASED RECOMMENDATION RESULTS

<i>ISBN</i>	<i>rating</i>	<i>score</i>	<i>combined ranking</i>
0345422384	7.0	1320	9240
0440237009	7.0	1243	8701
0451183665	8.0	1039	8312
0684847957	7.0	924	6468
0425083837	7.0	909	6363

TABLE VIII. ALS RECOMMENDATION RESULTS

<i>ISBN</i>	<i>rating</i>	<i>score</i>	<i>combined ranking</i>
0394800389	8.6	871	7490.6
0743454529	8.4	260	2184
0451456734	8.3	215	1784.5
0836213319	8.6	1	8.6
0394823370	8.4	1	8.4

TABLE IX. HYBRID RECOMMENDATION RESULTS

<i>ISBN</i>	<i>rating</i>	<i>score</i>	<i>rating score</i> *	<i>source</i>
0345422384	7.0	1320	9240	Item-based
0440237009	7.0	1243	8701	Item-based
0451183665	8.0	1039	8312	Item-based
0684847957	7.0	924	6468	Item-based
0425083837	7.0	909	6363	Item-based
0394800389	8.6	871	7490.6	ALS
0743454529	8.4	260	2184	ALS

<i>ISBN</i>	<i>rating</i>	<i>score</i>	<i>rating score</i> *	<i>source</i>
0451456734	8.3	215	1784.5	ALS
0836213319	8.6	1	8.6	ALS
0394823370	8.4	1	8.4	ALS

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed a hybrid book recommender system to make personalized recommendations for users. We used the item-based algorithm with the ALS to solve the sparsity problem and address the over-fitting issue. Given the book popularity, we used the Twitter Developer API to collect streaming tweets in order to prioritize the book recommendations. We calculated the Lucene score of a book by querying the collected tweets against its description. This gave us a numerical representation of how “popular” the book is right now. The higher the score, the more popular the book, and thus trends are incorporated into the book recommendations. As a result, a hybrid book recommender algorithm was proposed, which combined model-based recommender algorithms with the book popularity. Our proposed system was implemented using a big data framework. Apache Kafka and Apache Spark Streaming were used to ingest and process Twitter streaming data. PySpark was used to implement data pre-processing and data analytics. The ALS recommender algorithm was implemented using Apache Spark MLlib. The results showed that the accuracy of the proposed hybrid recommender system was acceptable and it assisted readers in finding appropriate books among a flood of information. In the future, we intend to improve our model by incorporating book bibliography data such as publisher, author and category as well as users’ demographic data. Regarding the big data framework, we intend to deploy the proposed hybrid recommender system on cloud platforms such as AWS Cloud or Google Cloud Platform.

ACKNOWLEDGMENT

The authors acknowledge and thank Twitter for allowing the use of Twitter API and public tweets with specific taglines and dates. The authors would like to thank the student team (Benjamin Ng Jet Hon, Ng Zhi Wen and Malur Seshadri Shridhar) for their assistance with data collection, data cleaning and model building.

REFERENCES

- [1] L. Sharma and A. Gera, “A survey of recommendation system: Research challenges,” *International Journal of Engineering Trends and Technology (IJETT)*, May 2013, vol. 4, no.5, pp.1989-1992.
- [2] J.B. Schafer, J. Konstan and J. Riedl, “E-commerce recommendation applications,” in: R. Kohavi, F. Provost (Eds.)
- [3] Z. Huang, W. Chung and H. Chen, “A graph model for e-commerce recommender systems,” *Journal of the American Society for Information Science and Technology*, vol. 55, 2004, pp.259-274.
- [4] O.R. Zaiane, “Building a recommender agent for e-learning systems,” *Proceedings of 2002 International Conference on Computers in Education*, vol. 51, 2002, pp. 55-59.
- [5] J. Lu, “A personalized e-learning material recommender system,” *Proceedings of the 2nd International Conference on Information Technology and Applications*, Harbin, China, CDROM, 2004.
- [6] D. Sarma, T. Mittra, and M.S. Hossain, “Personalized Book Recommendation System using Machine Learning Algorithm,” In (IJACSA) *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 1, 2021.

- [7] E. U. Okon, B. O. Eke and P.O. Asagba, "An improved online book recommender system using collaborative filtering algorithm," *International Journal of Computer Applications*, vol. 179, no. 46, 2018, pp. 41-48.
- [8] M. Chandak, S. Girase and D. Mukhopadhyay, "Introducing hybrid technique for optimization of book recommender system," *Procedia Computer Science*, vol.45, 2015, pp.23-31.
- [9] G. Linden, B. Smith and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no.1, 2003, pp.76-80.
- [10] M. A. Vaidya and S. Shinde, "Hybrid Book Recommendation system," *International Research Journal of Engineering and Technology (IRJET)*, vol 6, No. 7, 2019, pp.3569-3577.
- [11] Z. Ali, S. Khuro and I. Ullah, "A hybrid book recommender system based on table of contents (toc) and association rule mining," In *Proceedings of the 10th International Conference on Informatics and Systems*, May 2016, pp. 68-74.
- [12] S. Langer and J. Beel, Apache Lucene as content-based-filtering recommender system: 3 lessons learned. *arXiv preprint arXiv:1703.08855*, 2017.
- [13] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, no. 3, 1997, pp.66-72.
- [14] J. Qi, S. Liu, Y. Song and X. Liu, "Research on personalized book recommendation model for new readers," In *2018 3rd International Conference on Information Systems Engineering (ICISE)*, IEEE, May 2018, pp. 78-81.
- [15] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-based collaborative filtering recommendation algorithms," In *Proceedings of the 10th international conference on World Wide Web*, April 2001, pp. 285-295.
- [16] R. Bhatia, Book-crossing (version 3). Retrieved from <https://www.kaggle.com/datasets/ruchi798/bookcrossing-dataset>, April 2011.
- [17] P. Simakis, Book Depository Dataset Data set. Retrieved from <https://www.kaggle.com/datasets/sp1thas/book-depository-dataset/metadata?select=dataset.csv>, 2020.
- [18] Apache SOLR website, <http://lucene.apache.org/solr/>
- [19] S. Gosh, N. Nahar, M. A. Wahab, M. Biswas, M.S. Hossain and K. Andersson, "Recommendation system for e-commerce using alternating least squares (ALS) on apache spark," In *International Conference on Intelligent Computing & Optimization*, pp. 880-893. Springer, Cham, December 2020.
- [20] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D.B. Tsai, M. Amde, S. Owen and D. Xin, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, vol. 17, no.1, pp.1235-1241, 2016.
- [21] N. Jonnalagedda and S. Gauch, "Personalized news recommendation using twitter," In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, IEEE, vol. 3, pp. 21-25, IEEE, November 2013.
- [22] <https://github.com/evancasey/spark-knn-recommender/tree/master/algorithms>.
- [23] F. O. Isinkaye, Y. O. Folajimi and B. A. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian informatics journal*, vol. 16, no. 3, 2015, pp.261-273.
- [24] C. Sirikayon, P. Thusaranon and P. Pongtawevirat, 2018, May. "A collaborative filtering based library book recommendation system," In *2018 5th International Conference on Business and Industrial Research (ICBIR)*, IEEE, May 2018 pp. 106-109.
- [25] G. Adomavicius and A. Tuzhilin, "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, vol. 6, 2005, pp. 734-749.



Fan Liu received the B. S. degree in engineering and the M.S. degree in automation from Northwestern Polytechnical University, Xi'An, China, in 2003 and 2006 respectively. She received the Ph. D degree in Electrical & Electronic Engineering from Nanyang Technological University, Singapore in 2013. In 2015, she worked as a data scientist in Government Technology, Singapore. She is a lecturer at ISS, National University of Singapore since 2019. Her current research interests include big data analytics, machine learning and artificial intelligence. She has authored/co-authored more than a dozen publications in the area of artificial intelligence and data analytics. Dr. Liu serves as a member of Singapore Computer Society (SCS). Dr. Liu is a certified NVIDIA deep learning institute instructor and ambassador.



Suriya Priya Asaithambi is working as Member at Institute of System Science, National University of Singapore. She designed & customised courses for industry professionals and MTech students in emerging topics. She consults in the areas of Data Engineering and Software Engineering. Her twenty-four yearlong teaching and consulting portfolio includes (1) Evaluating various reference architecture and choosing between hybrid design alternatives (2) Building customised technical stacks that satisfy given business needs) using combination of cloud native managed services, full-stack frameworks, libraries, programming languages, API end points and related testing tools). And (3) Building analytics solutions to provide operational and strategic business insights {using mathematical, data ETL pipelines, machine learning algorithms and data visualisation libraries}. She holds a PhD from School of Computing, National University of Singapore. She is a certified professional in TOGAF, GCCC and IBM products. Her current research emphasis is on cloud computing and data engineering.



Dr. Ramanathan Venkatraman is working as Member, Advanced Technology Application Practice at National University of Singapore. He has served industry and academia for more than 38 years and has a wide spectrum of experience in the fields of IT, business process engineering and optimization techniques. His current research focuses in evolving decision models for business problems and more recently, he has been contributing to frontiers of knowledge by devising innovative architectural models for ICT in domains such as Distributed Systems Architecture, Big Data and Enterprise Cloud Computing. Dr Venkatraman has a strong practice approach having worked in large scale IT projects for organisations across Asia, US, Europe and NZ. He has published more than 30 research papers in leading journals and reputed technical conference proceedings. Apart from research and consulting, he also teaches advanced technical courses for Master's program at NUS and has been a key architect in setting up innovative software engineering and business analytics curriculum in the fast changing IT education scenario.