DOI: 10.15439/2023F8647
ISSN 2300-5963 ACSIS, Vol. 35

Emotion-Based Literature Books Recommender Systems

Elena-Ruxandra Luţan
0000-0001-5363-9930
Department of Computers and Information Technology
University of Craiova, 200585, Craiova, Romania
Email: elena.ruxandra.lutan@gmail.com

Costin Bădică
0000-0001-8480-9867
Department of Computers and Information Technology
University of Craiova, 200585, Craiova, Romania
Email: costin.badica@edu.ucv.ro

***Abstract*—In this paper we propose two book recommendation methods based on emotions extracted from user reviews, using content-based filtering and collaborative filtering. The methods were experimentally evaluated on our own dataset that we collected from *Goodreads* – a popular website with large database of books and readers reviews. We created an experimental setup where the recommendation algorithms for carrying out the evaluation using two proposed evaluation metrics: coverage and average recommendations similarity.**

I. Introduction

PEERS feedback from interactions mediated by social media plays an increasingly important role in how we choose to approach different aspects of our lives. The lack of personal experience when taking decisions often leads to seeking the experience of peers by means of recommendations. Such recommendations can emerge in various forms including word of mouth, surveys, printed or online reviews, thus actively supporting our day-to-day life [13].

The aim of a recommender system is to provide meaningful recommendations to the users based on the products which might interest them, the recommendations trustworthiness being a mandatory characteristic. The design of a recommender system varies depending on the nature of products for which recommendations will be issued [9].

In this paper, we are focusing on a specific category of products, literature books. We propose and evaluate two recommender systems that incorporate emotion information based on two different recommendation techniques: content-based filtering and collaborative filtering.

Content-based filtering refers to recommending products which are similar to the product that is being watched [2]. Our proposed content-based filtering recommendation algorithm must observe the user interaction with a book and identify similar books based on certain book characteristics, such as book title or book author(s).

Collaborative filtering aims to mine the most similar users with the user of interest and to observe their preferences. Then, these preferences can be used to make predictions about what the user of interest might enjoy [3]. This allows the recommender system to also focus on products that the user of interest has not yet interacted with.

We appreciate that the outcome of our research is both fascinating and useful, because our methods use social generated data for identifying the similarities between the books, in addition to general publisher details about a given book (book title, author or genre).

The experimental dataset was collected from a popular book-oriented website, *Goodreads*, using our own customized web scraper.

The paper is structured as follows. In Section II, we present related works. Section III describes our proposed book recommendation algorithms, using content-based filtering and collaborative filtering. In Section IV, we provide an overview of the dataset and then we discuss the experimental results. The last section presents our conclusions.

II. Related Works

Chhavi Rana and Sanjay Kumar Jain [12] propose a system which makes content-based book recommendations based on the user navigation pattern. The system analyzes user's behaviour and then it predicts the category of books that would interest the user using content-based filtering. The authors observe the lack of accuracy of content-based recommendations, as after a certain amount of time, the users will be recommended the same similar items. Therefore they introduce a temporal dimension, which means that user navigation and most visited links are periodically analyzed and revised when using the content-based approach to make recommendations.

In [8], Jessie Caridad Martin Sujo and Elisabet Golobardes i Ribe present a system which recommends the book that best suits the reader based on the semantics of his or her writing style. They use posts from Twitter social network in order to determine the psychological profile of the user. The authors use a database consisting in characters text, associated personality type and corresponding book. Their proposed method computes the similarity between the Twitter post text and the cases database in order to recommend the most suitable book to the user.

An Enhanced Personalized Book Recommender System (EPBRS) is described in [15]. The proposed system uses the a similarity function based on Euclidean distance in order to identify users with similar interests. The recommendations are done using collaborative filtering by considering the books

Topical area: Information Technology for Business and Society

preferred by similar users. A dataset of reviews, users and associated ratings from Amazon bookstore was used for experiments. The book ratings are considered as features when making the predictions.

In [16], authors propose a system which is able to provide replies to queries regarding products details. The answer that is returned to the query is actually a review available for the product, which contains the relevant details. For experiments, they use two neural models, a simple model (NNQA) and a Transformer-based model (BERTQA). These models are evaluated regarding their ability to find the relevant reviews.

Anil Kumar and Sonal Chawla [6] make an analysis of the recommendation techniques which are most frequently used for book recommender systems. They also propose a new book recommender system based on Hybrid recommendation technique. The Hybrid recommender system works as follows: when the user searches for a book, the system computes the list of book recommendations using collaborative filtering on book ratings. Then the positive and negative user reviews for each book are identified such that the recommendation list will be sorted based on the number of positive reviews. The user is displayed the book recommendation list together with the details of the searched book.

Harsh Dubey and Suma Kamalesh Gandhimathi [4] propose a recommender system which uses Deep Learning GPT3 (Generative Pre-trained Transformer). The project refers to building an application which finds books that are similar to a certain book provided as input. On a web interface, the user must describe a book that he or she has enjoyed reading. OpenAI API module is used for generating the recommendations of books that are similar with the input book description, and the top 3 recommendations are displayed together with details about the books availability obtained using Google Books API.

III. System Design

Three main recommender system techniques can be identified: content-based filtering, collaborative filtering and a combination of both [2]. They differ in their data sources, as well as in how these data sources are interpreted, analyzed and processed for building the recommendations [9].

In this paper we propose two recommender system algorithms for literature book recommendation, corresponding to the two distinct techniques: content-based filtering and collaborative filtering. Both algorithms incorporate the emotion categorization of each book as an important feature for determining similarities between books.

The emotions are extracted from online book reviews and then used for creating an emotion-based categorization of books using the system we previously proposed in [7]. In total, there are 35 emotions considered: 'cheated', 'singled out', 'loved', 'attracted', 'sad', 'fearful', 'happy', 'angry', 'bored', 'esteemed', 'lustful', 'attached', 'independent', 'embarrassed', 'powerless', 'surprise', 'fearless', 'safe', 'adequate', 'belittled', 'hated', 'codependent', 'average', 'apathetic', 'obsessed', 'entitled', 'alone', 'focused', 'demoralized', 'derailed', 'anxious', 'ecstatic', 'free', 'lost', 'burdened'.

Field Name	**Field Description**
Book Id	The id which uniquely identifies the book
Book URL	The Goodreads URL of the book
Book Title	The title of the book
Book Series	The book series name
Book Author	The author(s) of the book
Book Overall Rating	The book rating, a number in interval [1, 5]
Book Ratings Number	The number of ratings available on Goodreads for the book
Book Reviews Number	The number of reviews available on Goodreads for the book
Book Full Description	The description of the book
Book Genres	Top 10 genres available for the book on Goodreads website
Book Pages	The number of pages of the book
Book Year	The year in which the book was published
Emotions	The book emotions computed using [7]

TABLE I
Book Entity Description

Field Name	**Field Description**
Review Id	The id which uniquely identifies the review
Review URL	The Goodreads URL of the review
Book Id	The id of the book for which the review is given
Author Id	The id of the user who wrote the review
Review Stars	The rating given by the review author, as integer in interval [1, 5]
Review Date	The date when the review was written
Review Tags	Review tags or keywords given by the review author
Review Content	The review (text) provided by the review author

TABLE II
Review Entity Description

The emotion extraction workflow takes as input the review, performs standard NLP text preprocessing techniques (tokenization, lower casing, removal of stop words) and determines the emotions present in the text by making word-matching with a list of adjectives and their corresponding emotion.

Our proposed recommendation algorithms were validated on the experimental dataset previously introduced in [7]. This data set was collected by us from Goodreads website using our own customized web scraper.

The dataset contains tabular data describing two entities, Book and Review, which are interrelated by a one-to-many relationship. For both entities, several parameters available on Goodreads website were extracted and captured as separate columns. They are described in Tables I and II.

A. Content-Based Filtering

Content-Based Filtering approach recommends items considering user preferences. The hypothesis of Content-Based Filtering is that users are usually more interested in those items that are similar to items they liked in the past [3].

We analyzed which fields of each book item can be used to better define its characteristics. We decided to use the Book Title, Book Series, Book Author, as well as the main emotions triggered by the book reading, which are computed during the extraction of sentiments from the book reviews.

The recommendation algorithm takes as input a review of a given user for a given book which is available in the database. The review consists of two components: a number in range $[1, 5]$ which represents a scaled value capturing how much the user liked the book (which will be referred as Review Stars) and the opinion of the user expressed in natural language text (which will be called Review Content).

The general idea of the algorithm is to use the input review in order to decide how much the user enjoyed the current book in order to recommend other relevant books to the user.

Algorithm 1 Content-Based Filtering Algorithm

```
1: if Review Stars $< 3$ then
2:     Extract the emotions from the Review Content
3:     if Review Content Emotions match Book Emotions $> threshold$ then
4:         Recommend a book which differs from the rated book
5:     else
6:         Recommend a book similar with the rated one
7:     end if
8: end if
9: if Review Stars $\geq 3$ and Review Stars $<5$ then
10:    Recommend a better book similar with the rated one
11: end if
12: if Review Stars $= 5$ then
13:    Recommend a book similar with the rated one
14: end if
```

The Review Stars is used to classify the level of satisfaction that the book provided to the user, as follows:

- The user did not like the book (Review Stars = 1 or 2).
- The user liked the book, but was not over-joyed (Review Stars = 3 or 4).
- The user loved the book (Review Stars = 5).

We detail each case of the algorithm used for Content-Based Filtering (Algorithm 1). The algorithm contains 3 main IF clauses that deal with each one of the possible three satisfaction levels extracted from the input review.

The first IF clause (line 1) refers to the case when the user did not like the book (Review Stars = 1 or 2). In this case, we need to know what caused this dissatisfaction. We will take into consideration the Review Content and extract the emotions. In order to decide what kind of books to recommend, we decided to compare the Review Content Emotions with the Book Emotions. In case they match with high value, we considers this indicates that the user did not like the overall idea of the book, the kind of emotions that the book made him or her feel. In this case, we will recommend a completely different book emotions-wise, because it is most likely that the user will prefer something different. If the Review Content Emotions and the Book Emotions do not match, we interpret this as indicating that the user did not perceive the book as expected; maybe he or she did not actually understand the meaning of the book. In this case, we will recommend a book that is similar with the current one, as we guess that the user is likely to enjoy a new book which provides emotions rather close with the ones present in current book.

The second IF clause (line 9) refers to the case when user liked the book, but was not over-joyed by it. The aim of the recommender system is to provide recommendations for products which are likely to offer the greatest experience. For this reason, we will recommend books which provide similar sentiments, but are higher in ratings than the current book.

The last IF clause (line 12) refers to the case when the user loved the book. In this case, we recommend to the user a book which is very similar to the current one, because he or she is likely to enjoy it as much.

When recommending new books, we include only books that the user has not seen, i.e. books to which the user has not yet given reviews.

By applying the content-based filtering algorithm (Algorithm 1) we obtain a list of books which are considered the user might enjoy, and the top 5 books are displayed to the user as recommendations.

B. Collaborative Filtering

Collaborative Filtering method aims to find similarities between users based on the user-item interaction [1]. The system divides the users into clusters by considering their past interactions and makes recommendations according to the preference of the cluster the user belongs [3].

Similarly to the Content Based Filtering method, the Collaborative Filtering algorithm takes as input a review of a user for a given book which is available in the database, with its two components, Review Stars and Review Content. Its pseudocode is presented as Algorithm 2.

Algorithm 2 Collaborative Filtering Algorithm

```
1: Compare the user of interest with all the users who provided reviews for the given book
2: if A similar user which matches $> threshold$ is found then
3:     The users are similar, recommend a book that the similar user liked
4: end if
```

So, we are interested in evaluating the similarity of the user of interest with other users from the database. In our model, the similarities between the user of interest and each of the users in database are computed based on the emotions that exist in their reviews given for the given book. Then we determine the 5 topmost users similar with the user of interest and we analyze their preferences. This means that we analyze their reviews to determine which other books these top 5 users rated.

We consider that a user liked a book if he or she provided 4 or 5 stars. Therefore, from all the books rated by the top 5 users, we will select only those which got 4 or 5 stars in the reviews. This will lead to a set of books which we consider the user of interest might enjoy.

In order to offer the greatest experience, we decided to filter the set of books according to the Book Overall Rating, assuming that higher rating means better book. Book Overall Rating is an attribute present for each book in our dataset and it represents the book rating as it is recorded on the *Goodreads*.

When a new review is given, the first step is to compute the emotions present in the review. The content of the review is pre-processed by removing unnecessary text from the review, tokenizing the review text into words and removing the stop words. Then we extract the emotions from the pre-processed text, using our own Emotions Extraction Algorithm introduced in [7]. Our emotion model is based on a list of maximum 35 emotions and their weights.

Following, we extract from the reviews dataset the set of reviews available for the rated book. This subset will be used with the purpose of finding similar users with the user who provided the review. Two users are considered similar if they provided review for the same book and the emotions which are available in their reviews match at least 50%.

The next step of the collaborative filtering algorithm is to identify the books that similar users liked in order to recommend them to the user of interest. In order to make recommendations, for each of the matching users we identify the rated books which received more than 3 stars (as we assume that the similar users liked these books) and were not yet rated by the user of interest, and we add them to the list of recommendations.

At this stage, we have obtained a list of recommendations which can be provided to the user of interest. Initially, we considered the default ordering of this list according to how were the books appended to the list. According to this ordering, the books preferred by the most similar users are located as topmost entries of this list. However, after a deeper analysis, we realized that this might not be the best possible ordering, because we would rely only on the most similar users in order to make recommendations, and this would restrict too much the space of possible recommendations. Therefore, we decided to define a better way to order the recommendations such that to not rely only on the preferences of the single topmost similar user. Consequently, we considered that a possibility is to order the recommendations list by the Book Overall Rating value, before providing the top recommendations to the user.

If the recommendation list does not contain the minimum number of 5 recommendations, the list is completed by adding the books with the highest rating available in the dataset.

IV. Experiments and Discussions

A. Dataset Preparation

The data set was pre-processed before the application of the recommendation algorithm. The aim of the pre-processing is to compute the books similarity matrix that contains the similarity value for each pair of books in the dataset.

We did not use all the fields of a book entity (see Table I) for our recommendation algorithms. Therefore we selected only those book features which are relevant and we combined them into a single text field. We consider to be relevant the following fields: Book Title, Book Series, Book Author and Emotions. The resulting string is stored into the books dataset as an additional column named "Combined Features".

Then we converted each "Combined features" field of a book into a vector of token counts. We applied this processing for each book of the dataset, thus obtaining a matrix T of token counters with elements natural numbers. The total number of tokens is equal to the size of the vocabulary that is found by analyzing the "Combined Features" field of each book. So, if there are n books and the size of the vocabulary is m the resulting matrix of token counts will have size $n \times m$. The count matrix was created using *CountVectorizer* class of *Scikit-learn* library available in Python [10].

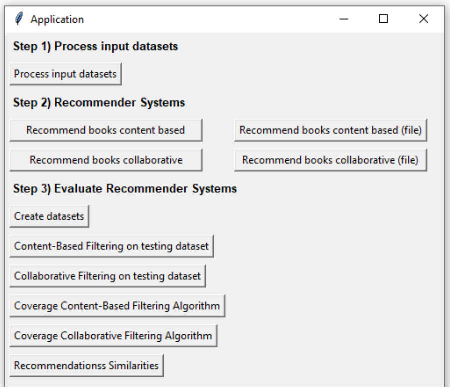

Application
Step 1) Process input datasets
Process input datasets
Step 2) Recommender Systems
Recommend books content based
Recommend books content based (file)
Recommend books collaborative
Recommend books collaborative (file)
Step 3) Evaluate Recommender Systems
Create datasets
Content-Based Filtering on testing dataset
Collaborative Filtering on testing dataset
Coverage Content-Based Filtering Algorithm
Coverage Collaborative Filtering Algorithm
Recommendationss Similarities

Fig. 1. Application Main Panel

Each row i of matrix T is a vector of counters describing book i. The similarity of two books i and j can be determined by applying a similarity measure to the vectors represented by rows i and j of T. In our implementation we have used the cosine similarity measure. If there are n books then the similarity matrix is a squared and symmetric matrix S of size n with real values in interval $[0, 1]$. We determined the books similarity matrix and we saved it into variable *cosine_similarity_matrix* [5].

For each input book $1 \leq i \leq n$, the books that are most similar with it can be determined by examining the row i of matrix S consisting of elements $S_{i,j}$ for all $1 \leq j \leq n$ of higher value.

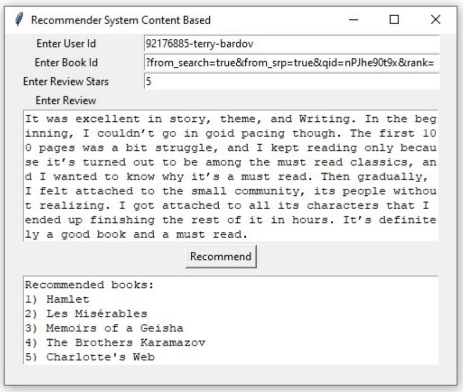
B. Application Interface

In order to simplify the use of the proposed system, we developed a convenient application interface using *Tkinter* Python Library, which can be seen in Figure 1.

The button "Process input dataset" from Step 1 refers to processing the reviews and books dataset by extracting the emotions from reviews and categorizing the books emotions based using our approach previously introduced in [7].

Step 2 refers to applying one of the recommendation algorithms. For both recommendation algorithms, we have created two approaches: the first that takes into consideration only one review manually inserted by the user, and the second that takes as input a list of reviews provided in a *CSV* file. From the main console of the application, the user can choose which function to execute, by using the corresponding button on Step 2.

Using the first approach (a manually inserted review), another panel will appear on the screen (Figure 2). The user has to insert using the keyboard the following information: the user id, the book id, the number of stars and the review content. If the Recommender System would be used in a real setting, the information about the user id and book id would be automatically collected from the context (current



Enter User Id: 92176885-terry-bardov
Enter Book Id: ?from_search=true&from_srp=true&qid=nPJhe90t9x&rank=
Enter Review Stars: 5
Enter Review:

```
It was excellent in story, theme, and Writing. In the beg
inning, I couldn’t go in goid pacing though. The first 10
0 pages was a bit struggle, and I kept reading only becau
se it’s turned out to be among the must read classics, an
d I wanted to know why it’s a must read. Then gradually,
I felt attached to the small community, its people withou
t realizing. I got attached to all its characters that I
ended up finishing the rest of it in hours. It’s definite
ly a good book and a must read.
```

```
Recommended books:
1) Hamlet
2) Les Misérables
3) Memoirs of a Geisha
4) The Brothers Karamazov
5) Charlotte's Web
```

Fig. 2. Application Panel for insertion of review details and results obtained using Content-Based Filtering Recommendations Algorithm

book selection and authentication information), but since our project is focused on the experimental evaluation of our algorithms (not on the actual graphical user interface of a Web-based deployed system), this information needs to be manually inserted by the user.

After filling in all the required fields, the user must press the button “Recommend” thus triggering the recommendation process. The Recommender System processes the input fields, applies the selected recommendation algorithm and displays the top 5 recommendations in the lower part of the panel.

The second implementation approach uses a list of reviews given in an input CSV file and applies the selected recommendation algorithm to each given review, rather than using a single review which was provided by the user as input.

Since using the second approach the results cannot be displayed in the same way as for the first approach, we had to find a meaningful way to display the output recommendations. We decided to store the results as a list inside an output text file. For readability, we also added to this file the information about the processed review, respectively Author Id, Book Id, Review Stars, Review Content and Emotions, together with the recommendations themselves.

C. Experimental Results

The dataset consists in 78 books and 6566 associated reviews, collected from *Goodreads* website. For majority of books (71 books) the reviews dataset contains 90 reviews, while for 7 books there are less than 90 reviews available.

These reviews were written by a total of 2658 users. 1755 users have written only 1 review, 795 users have written between 2 and 10 reviews, while 108 users have written more than 10 reviews (between 11 and 74 reviews).

The experimental setup is configured on Step 3 of the application workflow: Evaluate Recommender Systems (Figure 1). Firstly, the training and testing datasets have to be defined.

We have split the *Goodreads* dataset of 6566 reviews as 80% for training and 20% for testing. As different number of reviews are contained in the data set for each separate book, training - testing split was done for each book. Following this splitting procedure, the training dataset contains 5267 reviews and the testing dataset contains 1299 reviews.

The training reviews dataset was used for defining the book emotions feature, which means that the emotions were extracted from the review content and are attached to the book using our procedure previously introduced in [7].

The testing reviews dataset contains those reviews based on which the system will provide recommendations in order to perform the experimental evaluation of our proposed recommendation algorithms. Each entry in the testing dataset can be seen as a new review that is currently added by a user who expects to receive book recommendations.

Let us define the following parameters that are used for the rigorous definition of our proposed evaluation metrics:

- *Recommendation space* R refers to the total number of possible recommendations, i.e. the total number of books available in the books dataset (in our case 78).
- *User input space* U refers to the total number of user inputs u. A user input is a new review added for a certain book from the dataset.

$$u = (book, review), where\ book \in R$$

- *Test space* T refers to the subset of the input space $T \subset R$ used for experimental evaluation.
- A *recommendation* $f_i(u)$ refers to the output recommendation obtained when applying recommendation algorithm i. The output is a set of 5 books $r_i \in R$. $i = 1$ denotes the Content-based Filtering Algorithm, while $i = 2$ denotes the Collaborative Filtering Algorithm.

$$f_i \colon U \rightarrow R^5$$

$$f_i(u) = (r_1, r_2, r_3, r_4, r_5)$$

- *Total number of unique recommendations* $TNUR_i$ refers to the amount of unique books from the dataset which are returned as recommendations by algorithm i. In our case, $TNUR_1$ refers to the books returned as recommendations by Content-based Filtering algorithm and $TNUR_2$ refers to the books returned as recommendations by Collaborative Filtering algorithm.

$$TNUR_i = \bigcup_{u \in U} \{f_i(u)\}$$

- *Recommendations similarity* s refers to the similarity between recommendations $f_1(u)$ and $f_2(u)$ provided for the same user input u using the Content-based Filtering algorithm, respectively Collaborative Filtering algorithm.

$$s \colon R^5 \times R^5 \rightarrow [0, 1]$$

s is determined using Jaccard index.

$$s(f_1(u), f_2(u)) = \frac{|f_1(u) \cap f_2(u)|}{|f_1(u) \cup f_2(u)|}$$

Considering that each of the two algorithms provides a list of 5 recommendations, it follows:

$$s(f_1(u), f_2(u)) = \frac{|f_1(u) \cap f_2(u)|}{10 - |f_1(u) \cap f_2(u)|}$$

We propose two performance measures for evaluating our recommendation algorithms, as follows:

- *Coverage* C_i determines the proportion of books from R that the system was able to recommend using recommendation algorithm *i*.

$$C_i = \frac{|TNUR_i|}{|R|}$$

- *Average Recommendations Similarity* ARS is the average of the similarity between recommendations provided using Content-based Filtering and Collaborative Filtering algorithms.

$$ARS = \frac{1}{|T|} \sum_{u \in T} s(f_1(u), f_2(u))$$

For the Content-based Filtering algorithm, we have obtained a coverage of 96.153%, which means that that when suggesting book recommendations for the 1299 testing reviews, 75 books from the dataset were recommended.

The same coverage 96.153% is obtained for Collaborative Filtering algorithm (just simple coincidence).

Note that even if we obtained equal coverage values for both recommendation algorithms, the books which are not recommended by these algorithms are different. For the Content-Based Filtering algorithm the 3 not-recommended books from the dataset were: index 24 ("The Handmaid's Tale" by Margaret Atwood), 54 ("Charlie and the Chocolate Factory" by Roald Dahl) and 77 ("The Story of My Life" by Helen Keller), while for the Collaborative Filtering algorithm the books from the dataset which were not recommended are: index 63 ("The Good Earth by Pearl S. Buck | Summary & Study Guide"), 64 ("Sidekick to Mockingjay by Suzanne Collins" by Katherine R. Miller) and 70 ("The Road by Cormac McCarthy | Summary & Study Guide").

The *Average Recommendations Similarity* between the books recommendations received using the two algorithms is 5.71%. This rather low value was somehow expected. It shows that applying both recommendations algorithms on the same user input review generates rather different recommendations. In total, for our 1299 input reviews, 6495 recommendations were obtained using Content Based Filtering and 6495 were obtained using Collaborative Filtering, as both recommendations algorithms provide to the user the top 5 recommendations. Out of the 6495, only 602 were identical.

V. Conclusions

In this contribution, we presented two different algorithms for making valuable book recommendations considering also the emotions extracted from online book reviews submitted by book readers. The proposed recommendations algorithms are based on content-based filtering and collaborative filtering.

The emotions present in online book reviews are used to create an emotion-based categorization of books. Then the emotion categorization is considered as an additional book feature when computing the similarity between two different books, together with the book title and the book author(s).

We created an experimental setup using a books and reviews dataset that we collected from *Goodreads* website using our customized web scraper. We divided the reviews dataset into two groups: training and testing. The training consists in extracting the emotions present in the reviews and using them to categorize the books, while the testing refers to giving the reviews one by one as input to our system and receiving recommendations.

We proposed two performance metrics *Coverage* and *Average Recommendations Similarity*. Our experimental evaluation shows a good books dataset coverage on both recommendation algorithms, as almost all books from the dataset are given as recommendations for all the possible user inputs. On the other hand, the *Average Recommendations Similarity* metric provides low similarity values of recommendations generated using Content-based Filtering and Collaborative Filtering. This is expected, considering the different nature of the recommendation methods involved.

References

[1] Aggarwal, C.: Recommender Systems The Textbook (2016) Springer International Publishing

[2] Agrawal, R.: How to Build a Book Recommendation System (2021) https://www.analyticsvidhya.com/blog/2021/06/build-book-recommendation-system-unsupervised-learning-project/

[3] Dey, V.: Collaborative Filtering vs Content-Based Filtering for Recommender Systems (2021) https://analyticsindiamag.com/collaborative-filtering-vs-content-based-filtering-for-recommender-systems/. Last accessed 10 Feb 2023

[4] Dubey, H., Gandhimathi, S. K.: Book Recommendation System Using Deep Learning (GPT3) International Research Journal of Engineering and Technology (IRJET), vol. 9(5) (2022)

[5] Karbhari, V.:What is a cosine similarity matrix? (2020) https://medium.com/acing-ai/what-is-cosine-similarity-matrix-f0819e674ad1. Last accessed 10 Feb 2023

[6] Kumar, A., Chawla, S.: Framework for Hybrid Book Recommender System based on Opinion Mining. International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Vol.8(4) (2019) 10.35940/ijrte.D7518.118419

[7] Luţan, E.-R., Bădică, C.: Emotion-Based Literature Book Classification Using Online Reviews. Electronics 2022, 11, 3412. https://doi.org/10.3390/electronics11203412

[8] Martín, J.,Ribé, E.: BRAIN L: A book recommender system (2023) 10.48550/arXiv.2302.00653

[9] Melville, P., Vikas, S.: Recommender systems. Encyclopedia of machine learning 1 pp. 829-838 (2010)

[10] Movie Recommendation Model Using Cosine_Similarity and CountVectorizer: Scikit-Learn (2019) https://regenerativetoday.com/movie-recommendation-model-using-cosine_similarity-and-countvectorizer-scikit-learn/ Last accessed 31 Mar 2023

[11] Polignano, M., Narducci, F. de Gemmis, M. Semeraro, G.: Towards Emotion-aware Recommender Systems: an Affective Coherence Model based on Emotion-driven Behaviors. Expert Systems with Applications 2021, 170, 114382, https://doi.org/10.1016/j.eswa.2020.114382

[12] Rana, C., Jain, S. K.: Building a Book Recommender system using time based content filtering. WSEAS Transactions on Computers 11.2 (2012): 27-33.

[13] Resnick, P., Hal R. V.: Recommender systems. Communications of the ACM 40.3 pp. 56-58 (1997)

[14] Roy, D., Dutta, M.: A systematic review and research perspective on recommender systems. Journal of Big Data 9, 59 (2022). https://doi.org/10.1186/s40537-022-00592-5

[15] Usman, A., Roko, A., Muhammad, A.B. Almu, A.: Enhancing Personalized Book Recommender System. Int. J. Advanced Networking and Applications, vol.14(03), pp. 5486–5492 (2022)

[16] Zhang, S., Lau, J. H., Zhang, X. J., Chan, J., Paris, C.: Discovering Relevant Reviews for Answering Product-Related Queries. 2019 IEEE International Conference on Data Mining (ICDM) 10.1109/ICDM.2019.00192