Overview on NLP techniques for content-based recommender systems for books

Melania Berbatova
Sofia University ”St. Kliment Ohridski”
melania.berbatova@uni-sofia.bg

Abstract

Recommender systems are an essential part of today’s largest websites. Without them, it would be hard for users to find the right products and content. One of the most popular methods for recommendations is content-based filtering. It relies on analysing product metadata, a great part of which is textual data. Despite their frequent use, there is still no standard procedure for developing and evaluating content-based recommenders. In this paper, we first examine current approaches for designing, training and evaluating recommender systems based on textual data for books recommendations for the GoodReads website. We examine critically existing methods and suggest how natural language techniques could be employed for the improvement of content-based recommenders.

Nomenclature

CBF	Content-based filtering
CF	Collaborative filtering
RS	Recommender systems

1 Introduction

Recommendation systems are engines that use algorithms leveraging the interaction between users to generate personalized recommendations. They provide users with recommendations for new content these users might be interested in (music, movies, books, etc).

Recommendation systems can be divided into three main types: Collaborative Filtering (CF), Content-based Filtering (CBF) and Hybrid systems. Collaborative filtering systems analyze users interactions with the items (e.g. through ratings, likes or clicks) to create recommendations. On the other hand, content-based systems use semantic information (frequently called metadata) about the items in the system. Hybrid systems are a combination of these two approaches. If compared to collaborative or content-based systems, hybrid ones usually exhibit higher recommendation accuracy. This is due to the fact that CF lacks information about domain dependencies, while CBF systems do not take into account users preferences.(Krasnoshchok and Lamo, 2014)

Collaborative filtering recommenders are systems that suggest recommendations based on users interactions (most commonly, ratings). A great deal of the most efficient collaborative filtering algorithms are based on the matrix factorization (MF). Matrix factorization algorithms work by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices. (Koren et al., 2009) This family of methods gained popularity around the Netflix prize challenge and showed state-of-art results for many RS tasks. However, in this paper we will focus on content-based methods, as they can benefit from natural language techniques and increase the accuracy of recommendations in a hybrid approach.

Content-based recommenders are a type of recommender systems that use item metadata (description, rating, products features, reviews, tags, genres) to find items, similar to those the user has enjoyed in the past. To generate recommendations, we use items that are most similar to the ones liked by a given user. In the context of books, main characteristics, and even whole book content can be present as metadata. As descriptions and reviews are purely natural language data, and categorical data such as tags and genres can also be represented in a way suitable for natural language processing, employing such techniques is crucial for the design of successful content-based recommenders.

In content-based recommendation systems, the features of products, e.g. the genre and the author of a book, are represented most often as a bag of words or a vector space model. Features of a book might refer to its title, summary, outline, whole text, or metadata, including the author, year of publication, publisher, genre, number of pages, etc.

Content-based recommenders use a variety of machine-learning algorithms, including Naive Bayes, support vector machines, decision trees, and kNN. As bag-of-words and vector representations can have hundreds or thousands of dimensions, techniques as Latent Dirichlet Allocation (LDA) are often adopted. The content may also require natural language processing (NLP) techniques to make use of semantic and syntactic characteristics.

A recommender system should not be designed without taking into consideration the nature of the items. Contrary to those of other text-based items, such as news and scientific papers, book preferences are highly influenced by characteristics specific to books, such as book size, readability level and writing style. Thes particularities motivate the designing of recommenders that perform both a syntactic and a semantic analysis of book texts. A promising way to enrich book metadata is the automatic genre identification. Users on community-based book websites can assign tags and organize books under custom-defined ”shelves”. These tags and shelves can serve as genres and can be used to indicate patterns in users' opinions.

2 Dataset

Goodbooks-10k is a compilation of 5,976,479 ratings for the most popular 10,000 books in the book website Goodreads, as well as book metadata for each book. The dataset is available online on the FastML website[1] . Data, in the form of ratings, books metadata, to-read tags, and user tags and shelves, is organised in 5 files. The distribution of ratings in this dataset is centered around 100 ratings per user, where the average rating per user is 4. The distribution of the number of ratings per user and the average rating per user seems to follow a multivariate normal distribution. (Greenquist et al., 2019)

[1]http://fastml.com/goodbooks-10k-a-new-dataset-for-book-recommendations/

Some previous research on the topic of books recommender systems relies on datasets such as Book-Crossing, LitRec (Vaz et al.), LibraryThing (Lu et al., 2010). The main advantage of the goodbooks-10k dataset over the above mentioned ones is the volume of data. As the number of records is close to 6 million and the data presented is diverse and consistent, it allows experimenting with different algorithms, including ones that are designed for big data.

3 Related work

In their paper ”A survey of book recommender systems”, Alharthi et al. (2017) present a detailed survey on different approaches to book recommendation, compiled from over 30 papers up to 2017. These publications report results from CBF, CF and other methods obtained on the Book-Crossing, LitRec, LibraryThing, INEX, and Amazon reviews datasets. Only LitRec (Vaz et al.) dataset uses data from GoodReads.

In the current study, we will focus on models for GoodReads built on the goodbooks-10k dataset. Recent publications written on this dataset mostly deal with collaborative filtering. Out of 11 unique papers in English on recommender systems retrieved by Google Scholar when search is performed for goodbooks-10k (Le, 2019; Kula, 2017; Recommendation; Greenquist et al., 2019; Zhang et al., 2019, 2018; Paudel et al., 2018; Khanom et al., 2019; Kouris et al., 2018; Yang et al., 2018; Hiranandani et al., 2019), 10 examine algorithms for Collaborative filtering, two (Le, 2019; Greenquist et al., 2019) implement hybrid systems, and only one (Le, 2019) implement a simple content-based recommender. We will examine the content-based systems or components of hybrid systems, developed on goodbooks-10k, and will compare them to systems using another dataset for GoodReads - LitRec.

3.1 Overview

An overview of published content-based approaches for GoodReads is shown in Table 1.

In their bachelor thesis, Le (2019) implement simple collaborative, content-based and hybrid systems for book recommendations. Their content-based recommender uses only ratings data and leaves aside books metadata. They achieve a best score of 0.842 of root-mean-square error (RMSE) for FunkSVD algorithm.

Authors	Dataset	Features	Evaluation metrics	Algorithms	Dataset creation
Le (2019)	goodbooks-10k	ratings	MAP, CC, MPS, MNS, MDS	cosine similarity	test set - of 5-star ratings
Greenquist et al. (2019)	goodbooks-10k	tf-idf vectors	RMSE	cosine similarity	5+ ratings per user
(Alharthi and Inkpen, 2019)	Litrec	linguistic and stylometry features	precision@10, recall@10	kNN	10+ rating per user

Table 1: Overview of recent published papers

Greenquist et al. (2019) implement a CBF/CF hybrid system. To gather more information, they merge goodbooks-10k data with Amazon reviews data. For books representations, they use tf-idf vectors of the books descriptions, tags, and shelves. Authors report using book descriptions in their content-based approach, but it is unclear how they obtained the descriptions, as the latter are not present in the goodbooks-10k dataset.

In addition to published ones, there are many other approaches to the goodbook-10k dataset, implemented and shared by community members on platforms such as Kaggle.com. On Kaggles goodbook-10k dataset page, there are 31 shared kernels[2]. Some of them contain demonstrations on the development of content-based systems, including ones that use tags information. It can be seen that, as mentioned in (Greenquist et al., 2019), tags are turned into tf-idf vectors, and cosine similarity is used for determining the books that are the most similar to a given one.

Alharthi and Inkpen (2019) use the Litrec dataset to develop a book recommendation system based on the linguistic features of the books. Litrec dataset has ratings of 1,927 users of 3,710 literary books and contains the complete text of books tagged with part-of-speech labels.

The content-based systems that Alharthi and Inkpen develop are based on the analysis of lexical, character-based, syntactic, characterization and style features of the books texts. Feature sets are learned from book texts converted into a numerical value using one-hot encoding.

For linguistics analysis, the authors use Linguistic Inquiry and Word Count (LIWC) (Tausczik and Pennebaker, 2010), which is a popular resource that focuses on grammatical, content and psychological word categories. Using LIWC 2015 dictionary, Alharthi and Inkpen compute 94 categories, such as: percent of latinate words, function words, affect words, social words, perpetual processes etc.

Other text measurements are computed by using GutenTags built-in tagger (Brooke et al., 2015), which uses a stylistic lexicon to calculate stylistic aspects usually considered when analyzing English literature. The six styles are colloquial (informal) vs. literary, concrete vs. abstract and subjective vs. objective. In addition, they use the fiction-aware named entity recognizer LitNER to identify number of characters and number of locations mentioned in a book. Finally, a text readability measurement is introduced, by calculating the Flesch reading ease score. All 120 features are used for finding most similar books using k-nearest neighbours (kNN) and Extreme Trees (ET) algorithms, and are tested against CBF baselines: LDA, LSI, VSM and Doc2vec. Both kNN and ET achieved higher scores than the baselines in both precision@10 (0.36 and 0.37, respectively) and recall@10 (0.17 for both).

Unfortunately, in goodbooks-10k, the book content is not available, and it would be extremely hard to gather and process this data. Therefore introducing stylometry and content features, as the ones mentioned above, would be impossible. The only suitable way of incorporating linguistic features would be by analyzing tags or by scraping books descriptions and reviews available at GoodReads.

[2]https://www.kaggle.com/zygmunt/goodbooks-10k/kernels

3.2 Critiques on content-based recommenders

The main critique with regards to the systems developed on goodreads-10k is the lack of usage of textual data, such as tags available. Even in the cases where tags are used, no attention has been paid to the fact that most of the tags follow a hierarchical structure (eg, ”biographical”, ”biographical-fiction”, ”biographical-memoir”) and some have similar or equal meaning (e.g. ”ya-dystopian”, ”young-adult-dystopian”, ”teen-dystopian”, ”dystopian”, ”antiutopian”, ”utopia-dystopia”).

In order to deal with hierarchy, ambiguity and synonyms, some data normalization and natural language processing techniques could be adopted. Tf-idf vectorization is a common technique in text processing; however, it is arguable whether it is the most suitable way for vectorizing tags information, as their distribution does not necessarily follow the one of words in natural text.

Another observation is that the systems developed do not take advantage of the recent advances in machine learning algorithms, especially deep learning, despite the large volume of the data available.

3.3 General Critiques

One general critique on the observed publications is the lack of standardization in dataset preparation and of evaluation metrics usage. Firstly, some redefine a ”like” as having a rating of at least 3 stars, while others dont provide a clear definition of a like. Secondly, some drop users having less than 5 ratings, others - less than 10, and yet others seem not to take out any users. And lastly, since the initial dataset does not come with established training and test sets, the way different researchers have performed the train/test split seem to diverge. All these three factors result in different datasets used by the researchers, and therefore, lead to non-comparable results.

Another factor that makes the results non-comparable is the difference in evaluation metrics used. As it can be seen in Table 1, there is a huge variety of metrics, such as those that measure rating predictions (root mean squared error (RMSE)), ranking metrics (precision@k, recall@K, mean average precision (MAP)), metrics for coverage (catalog coverage (CC)), metrics for personalization, diversity and novelty. Without suitable, unified metrics, it would be impossible to credibly prove that the use of NLP techniques will significantly improve RS performance on the current task.

Since general dataset preparation and the choice of the evaluation metrics is a considerable topic, not central for this research proposal, we would like to leave it for future research.

4 Experiments

There are 5,976,479 ratings in our dataset. We chose the standard split of 80% percent of randomly sampled ratings for training data and 20% for test data, resulting in 4,781,183 train ratings and 1,195,296 test ratings. Because of the large volume of data, we preferred this method over multiple fold cross validation, as the cross validation would have slowed the work of the predictive algorithms.

As shown in Figure 1, the higher ratings significantly outnumber the lower ones. Therefore, we chose to define a like as a rating of 4 or more stars, instead of 3 or more, as defined in previous research. We estimated that this would lead to more balanced data and a better approximation of readers perception, which was further proved by our experiments.

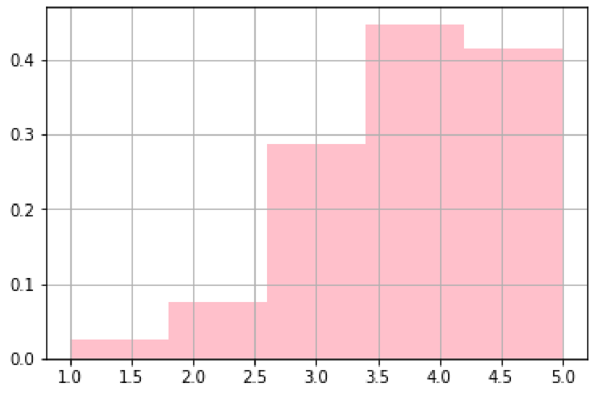

Figure 1: Distribution of ratings in the dataset

We used the python library TensorRec [3], which is a recommendation system library based on the deep learning library tensorflow, and employs the power of tensors and computational graphs. Recommender systems designed with TensorRec can be customized by choosing representation graphs, prediction graphs and loss graphs. Representation graphs are used for choosing the algorithms for

[3]https://github.com/jfkirk/tensorrec

latent representations (embeddings) of users and items. Prediction graphs are used for computing recommendation scores from the latent representations of users and items. Loss functions are the algorithms that define how loss is calculated from a set of recommendations.

We decided to work with recall@k as an evaluation metric. Recall@k shows the number of user's liked test items that were included in the top k of the predicted rankings, divided by k. Similar to Alharthi and Inkpen (2019), we chose k to be 10. We preferred recall@k than precision@k, because if the user has rated less than 10 books, precision@10 for their prediction cannot be 1. Finally, we preferred racall@k over RMSE and other metrics that measure the ratings prediction error, as for the design of the current recommender system, the exact score predicted is not as important as ranking the liked items higher than the not liked.

So far we have experimented with training simple CF and CBF systems. For our experiments, we used linear embeddings and weighted margin-rank batch (WMRB) (Liu and Natarajan, 2017) loss graph, which led to significantly better results on recall@10 than if optimizing for RMSE. For collaborative filtering we achieved 5.5% recall@10.

For the content-based approach, we used the year of publication and one-hot-encoded language as metadata features. We obtained results of 0.98% recall@10. The problem that we faced with content-based approach, was that we could not train algorithms with bigger feature sets, as there were memory errors when we ran our experiments both locally and in popular cloud services, such as CodeLab[4] and Kaggle[5]. This prevented us from evaluating the methods proposed in the current paper. The scores achieved by both methods with different parameters are shown in Table 2. As expected, approaches using ratings of 3 stars as a ”like” showed worse results. We expect to improve results by several percents when we succeed in using more metadata features and we combine the designed CF and CBF in a hybrid system.

5 Suggested NLP improvements

5.1 Using tags information

As we explained in the previous section, user tags information can be better utilized for the recommenders. The tags of a book show its genres (e.g. ”young-adult”, ”fiction”, ”biography”), readers intents (”to-read”, ”to-buy”, ”to-be-finished”), books features (”printed”, ”books-in-spanish”), awards (”printz-award”), authors (”oscar-wilde), etc. and many of the tags have similar or equal meanings. Every book is characterized by its tags and the number of occurrences of every tag. We dispose of almost 1 million tracks of `((book_id, tag_id, count)`, or an average of 100 tags per book. The total number of defined tags is 34,251. Around 1000 of the tags are from languages different from English, such as Arabic, Persian, Russian, Greek etc. The set of tags per book is similar to textual data, except that there is no sequence between the tags.

We propose using a bag-of-words word model of tags, as being more suitable than a tf-idf-based one. Firstly, available tags need to be split into tokens and cleaned. Foreign language are around 3 percent of all, so excluding them is not expected to lead to significant loss of information. After cleaning and bag-of-words vectorization, we can extract a variety or features. As Alharthi and Inkpen (2019) mention, we can use linguistic resources as Linguistic Inquiry and Word Count, or alternatively, predefined dictionaries of book genres (such as the ones available on Wikipedia[6] and YourDictionary[7]) to design features of books genres and vocabulary style. The count of how many times the tokens of a book tags fall into a category can be used as a fuzzy representation of to what degree the given book belongs to a category. Alternatively, we can use the bag-of-words representations of tags together with an unsupervised dimensionality reduction algorithm, as latent semantic analysis (LSA), to represent books.

Another approach is to use the power of word embeddings. Embeddings are used for transforming a word into a vector from a vector space with a fixed dimensionality, in a way that words occurring in similar contexts are represented by similar vectors. Current pre-trained word embeddings, such as word2vec (Mikolov et al., 2013), ELMo (Peters et al., 2018) and BERT (Devlin et al., 2018) have proved to raise performance on many natural language processing tasks, including text classification (Lai et al., 2015). We can use the weighted average of the tags tokens of a book as representation of the book.

[4]https://codelabs.developers.google.com/

[5]https://www.kaggle.com/kernels

[6]https://en.wikipedia.org/wiki/List_of_writing _genres

[7]https://reference.yourdictionary.com/books-literature/different-types-of-books.html

Algorithm	Parameters	Recall@10
CF	Like = 3+ rating	4.69%
CF	Like = 4+ rating	5.52%
CBF	Like = 3+ rating	0.83%
CBF	Like = 4+ rating	0.98%

Table 2: Results

5.2 Data enrichment

In addition to a better exploitation of the available data, we can also gather new natural language data and extract additional features from it. As mentioned by Greenquist et al. (2019), it would be useful to work with book descriptions. We can easily scrape these descriptions from GoodReads website using the books IDs, or, if unavailable in certain cases, we can scrape them from Amazon.com book pages. From descriptions we can extract features as sentiment and distribution of adjectives, adverbs, nouns, and verbs; or we can represent descriptions as tf-idf vectors.

5.3 Alternative approaches

An alternative algorithmic approach to the ones discussed so far is to think of the recommendation task as a classification problem. For every user we can try to predict whether they ”like” or ”don’t like” certain set of books, based on a training set of labeled books. In this setup, we can use books’ metadata together with classical approaches for text classification, as SVM or Naive Bayes, or incorporate deep learning algorithms such as recurrent neural networks (RNNs) and long-short term memory (LSTM). If final predictions come with score for the labels given, we can sort the books in descending order of the scores for ”like”, take top 10 books in the sorted list and again measure recall@k. The downside of this approach is that it would not work well for users with too few ratings, as there will not be enough training data.

6 Conclusion

Many natural language processing techniques, such as extracting lexical, syntactic and stylometric features or word embeddings, can be used in content-based filtering for the recommendation of books. CBF systems employing these techniques can be used separately or in a hybrid with collaborative filtering. However, these techniques should be accompanied with standardized methods for dataset creation and result evaluation, so the results obtained are comparable to those from similar research.

References

Haifa Alharthi and Diana Inkpen. 2019. Study of linguistic features incorporated in a literary book recommender system. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 1027–1034. ACM.

Julian Brooke, Adam Hammond, and Graeme Hirst. 2015. Gutentag: an nlp-driven tool for digital humanities research in the project gutenberg corpus. In *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*, pages 42–47.

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

Nicholas Greenquist, Doruk Kilitcioglu, and Anasse Bari. 2019. Gkb: A predictive analytics framework to generate online product recommendations. In *2019 IEEE 4th International Conference on Big Data Analytics (ICBDA)*, pages 414–419. IEEE.

Gaurush Hiranandani, Raghav Somani, Oluwasanmi Koyejo, and Sreangsu Acharyya. 2019. Clustered monotone transforms for rating factorization. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 132–140. ACM.

Aniqa Zaida Khanom, Sheikh Mastura Farzana, Tahsinur Rahman, and Iftekharul Mobin. 2019. Bookception: A proposed framework for an artificially intelligent recommendation platform. In *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, pages 253–257. ACM.

Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37.

Panagiotis Kouris, Iraklis Varlamis, Georgios Alexandridis, and Andreas Stafylopatis. 2018. A versatile package recommendation framework aiming at preference score maximization. *Evolving Systems*, pages 1–19.

Oleksandr Krasnoshchok and Yngve Lamo. 2014. Extended content-boosted matrix factorization algorithm for recommender systems. *Procedia Computer Science*, 35:417–426.

Maciej Kula. 2017. Mixture-of-tastes models for representing users with diverse interests. *arXiv preprint arXiv:1711.08379*.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Hieu Le. 2019. Building and evaluating recommender systems.

Kuan Liu and Prem Natarajan. 2017. Wmrb: Learning to rank in a scalable batch training approach. *arXiv preprint arXiv:1711.04015*.

Caimei Lu, Jung-ran Park, and Xiaohua Hu. 2010. User tags versus expert-assigned subject terms: A comparison of librarything tags and library of congress subject headings. *Journal of information science*, 36(6):763–779.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Bibek Paudel, Sandro Luck, and Abraham Bernstein. 2018. Loss aversion in recommender systems: Utilizing negative user preference to improve recommendation quality. *arXiv preprint arXiv:1812.11422*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Collaborative-Filtering Recommendation. Building information systems using collaborative-filtering recommendation techniques. In *Advanced Information Systems Engineering Workshops: CAiSE 2019 International Workshops, Rome, Italy, June 3–7, 2019, Proceedings*, page 214. Springer.

Yla R Tausczik and James W Pennebaker. 2010. The psychological meaning of words: Liwc and computerized text analysis methods. *Journal of language and social psychology*, 29(1):24–54.

Paula Cristina Vaz, Ricardo Ribeiro, and David Martins de Matos. Litrec vs. movielens.

Fan Yang, Ninghao Liu, Suhang Wang, and Xia Hu. 2018. Towards interpretation of recommender systems with sorted explanation paths. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 667–676. IEEE.

Shuai Zhang, Lina Yao, Yi Tay, Xiwei Xu, Xiang Zhang, and Liming Zhu. 2018. Metric factorization: Recommendation beyond matrix factorization. *arXiv preprint arXiv:1802.04606*.

Xuejian Zhang, Zhongying Zhao, Chao Li, Yong Zhang, and Jianli Zhao. 2019. An interpretable and scalable recommendation method based on network embedding. *IEEE Access*, 7:9384–9394.