Annals of Library and Information Studies
Vol. 65, March 2018, pp. 77-82

A book recommendation system based on named entities

Tulasi Prasad Sariki[a] and Bharadwaja Kumar G[b]

[a]Assistant Professor [Senior], School of Computing Science and Engineering, Vellore Institute of Technology, Chennai, Email: tulasiprasad.sariki@vit.ac.in

[b]Associate Professor [Senior], School of Computing Science and Engineering, Vellore Institute of Technology, Chennai, Email: bharadwaja.kumar@vit.ac.in

Received: 25 October 2017; revised and accepted: 11 April 2018

Recommendation systems are extensively used for suggesting new items to users and play an important role in the discovery of relevant new items, be it books, movies or music. An effective recommendation system should provide heterogeneous results and should not be biased towards only the most popular items. Books are particularly well suited to content based filtering as they are now widely available in digital formats which can allow various text mining approaches to dig out content related information. This paper presents a framework to develop a content based recommendation system for books which can further be integrated with a collaborative filtering model. The proposed content based recommender will use the Named Entities as the basic criteria to rank books and give recommendations.

Keywords: Natural Language Processing; Recommendation systems; Named entity extraction; Similarity metrics; Text mining

Introduction

The growing acceptability of Web as a medium for electronic and business transactions has initiated the development of recommender systems for personalized recommendations. An important catalyst in this regard is the ease with which the web enables users to provide feedback about their likes or dislikes. Recommendation techniques are usually categorized as collaborative, content based, utility based, demographic and knowledge based[1]. Of these, the first two techniques have been widely utilized and a combination of these approaches, called the hybrid approach is also popular.

Recommendations systems play a vital role in making books visible which otherwise might be overlooked. This has tremendous commercial implications as the probability of a book being bought surely increases if it is recommended by a respected recommender. In case of recommender systems for books, many of the e-commerce sites and research works rely on the hybrid model[2]. This is an appropriate model as books are frequently bought items and hence a relatively large number of reviews and ratings can be quickly generated and user profiles can be created. Content based filtering for books is usually limited to using only the metadata available with the book such as genre, author's name, keywords supplied by the author or the publisher etc. Amazon has its own variant called item-to-item collaborative filtering[3] where it is based on the recommendation deduced from similarity of the items. Their algorithm is primitive and does not use the information that can be extracted from books in any way.

Due to increasing availability of books in digital form from various resources such as google books, it has become easy to extract useful information about books that augments content based filtering. Machines can now go through a large number of books and extract useful information.

In this paper, we look at named entities in a book that can provide an insight about the reader's possible choices of reading other related books. This notion applies to applies strongly to the biographical genre. This reasoning is based on the locality of reference of named entities. A user reading a biographical text would surely be interested in the major protagonists in

the book i.e. one or more lead characters which could be of interest to the reader. It can be logically considered that the reader might also be interested in reading other books in which some of the people in the current book figure prominently.

Review of literature

The proposed method relies heavily on extracting information about the most important protagonists figuring in a book. Popular tools available openly for named entity tagging include Stanford Named Entity Recognizer (SNER), Apache OpenNLP, and Alias-i LingPipe, Illinois Named Entity Tagger[4]. There are many other effective tools which are proprietary. Most of the open source tools mentioned here come with their own models which have been pre-trained on datasets such as Conference on Natural Language Learning and Message Understanding Conference datasets which employ text from newswire and a variety of other newspapers gathered over a period of time and annotated manually. The collected articles belong to a vast array of genres so that the data model performs well over general text.

The proposed system will work primarily with biographical texts so the NER tool chosen must perform well over a biographical domain. A comparison of these open source tools indicates that Stanford NER is currently best suited for the task. It scores consistently highly on both precision and recall over a data set of Wikipedia articles[5]. Stanford NER comes with 3 classifiers which can classify entities in up to 7 categories of which person, location and organization are the most important in the context of this paper. In terms of tagging people, Stanford NER was shown to have 93 % precision and 95% recall. Due to its high performance and easy availability, it emerges as the tool of choice.

Most non-fiction books tend to have larger paragraphs or are solitary documents and hence the method of finding the importance of entities should be suited to the length of the text. In a short paragraph, a person who is mentioned just once may play an important role in the context of the paragraph. Whereas in an entire book, a person who is mentioned once would not be that important in the context of the book. Research has shown that as many as 56% mentions tend to be singletons[6]. It can be reasonably argued that an entity's importance grows with the number of mentions it gets in a text and from personal experience I believe it holds up for biographies and other non-fictional writing.

Li et al.[7] analyzed book recommendation models in library setting. In this work, they reviewed major categories of book recommendation models, i.e. content-based filtering, collaborative filtering, hybrid approach and generalized profile association. Pera et al.[8] proposed a book recommendation system that is based on social interactions and personal interests to suggest books appealing to the users. Rana et al.[9] proposed a temporal based recommendation system using a counter for each item which gets updated with passage of time and thereby improving the whole recommendation process. Vaz et al.[10] proposed a work on book recommendation system through author ranking. Goodreads.com's[11] proposed item-based collaborative filtering (ICF) can be used to make good recommendations in a public library and assessed whether selecting books by author preferences can improve recommendations.

Based on our literature survey, the major works on book recommendations systems[7-11] have not used named entities for enhancing the existing recommender systems.

Methodology

The proposed system will take a book in plain text format as input and through a series of steps it will tag the entities mentioned in the book, build co-reference chains, extract the entities and co-references and rank them based on frequency. The output would contain the list of people in the book ranked according to their importance. Metrics used for judging the rank of a person will be discussed in the forthcoming sections. These ranks will be used to find similarity between books and to provide recommendations. The architecture of the proposed system can be seen in the Figure 1.

Named entity tagging

We collected few books related to the biographical genre from Project Gutenberg online book catalog. These books were fed to Stanford NER to tag named entities in the books. Stanford NER system provides an implementation of linear chain Conditional Random Field (CRF) sequence models[12]. As most books are very well formatted and undergo strict proofreading, no special pre-processing is required. It

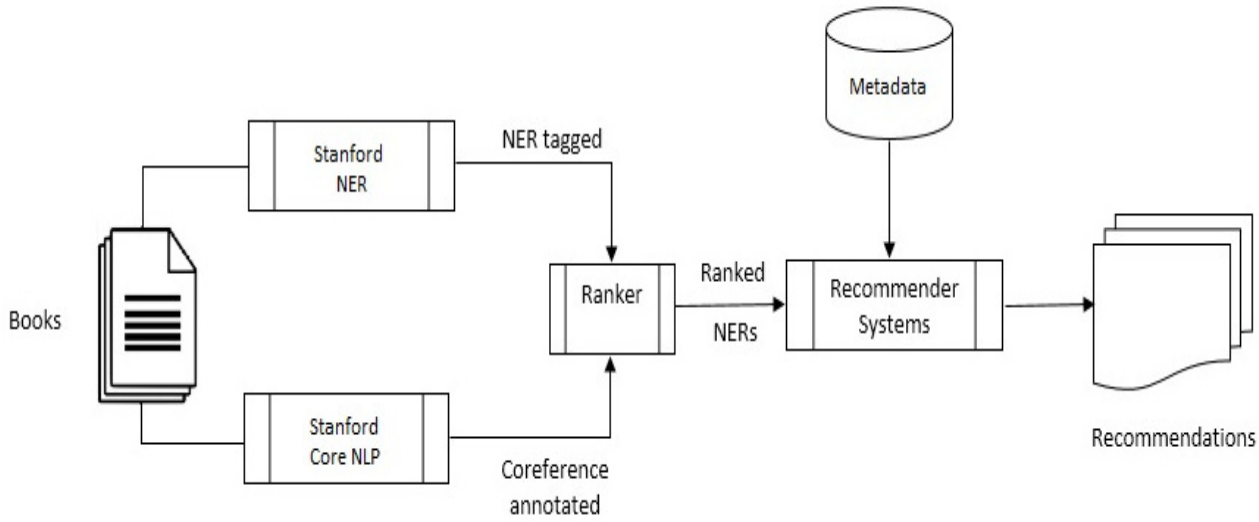

Fig. 1—System architecture

has well-engineered feature extractors for Named Entity Recognition in particular to the 3 classes such as person, organization and location. Output could be taken in plain text form, tab separated values or in-line XML. In our study, in-line XML was chosen as it gives a structured format easier for text processing in later stages. The running time of the system is proportional to the size of the input, usually tagging an entire book in under a minute. There is an option to train a new model for classification for better results but it needs a large amount of annotated data hence only the supplied models were used. As per the statistics shown in the cited research works, SNER is very accurate in this domain[5]. As the size of the text is quite large, some minimal errors in tagging were be overlooked. A manual perusal of the annotated text showed remarkable accuracy of tagging. For exact results, the input book needs to be manually tagged for comparison with the output which was avoided as it would have been time consuming.

Co-reference resolution

The same book in plain text is also fed to Stanford CoreNLP[13] and the output is an annotated text file complete with dependency trees, token offsets, lemmas and co-references. Depending on the annotation options and the size of the input, CoreNLP can take a lot of time to run. It sets up a pipeline consisting of a tokenizer followed by a sentence splitter, lemmatizer, part of speech tagger, parser, named entity tagger and a deterministic co-reference resolver. The most time consuming parts of the pipeline are the parser and the co-reference resolver. By tweaking the properties of these components, it was found that the run time can be significantly reduced (from over an hour to 20 minutes) by using proper end of line delimiters and limiting the valid size of a sentence. It was found that limiting the sentence length to 50 had both good accuracy and run time.

To further reduce run time and memory usage, the co-reference distance can be shortened. Co-reference distance is the distance in sentences for which the system will look for direct and indirect mentions of an item. A value of 10 sentences was found to be acceptable. Co-references are of particular importance in biographical or autobiographical books as a person will mostly be referred indirectly with an alias or a pronoun. Resolving the referents can greatly help in assessing the importance of a person in the books context. The output can be taken in text, XML or serial format. Here, text format is chosen as the output. Although NER is built into CoreNLP, the standalone version is used in parallel to speed up the process.

Entity extraction

The output of SNER is then sent to regular expression parser developed by us to extract only the entities which are related to persons. The white space

between the first name and last name is removed and replaced by a special character. This is important because now the entire name will be considered as one word and will allow the use of hash maps to do a word count. After the word count, the special characters are removed and the output now contains the names of people and the number of times their name appeared in the book. One immediate observation is that there would be a lot of solitary first name and last name mentions which can lead to ambiguity if multiple people in the book share a common first or last name.

One way to solve this is to follow the sequence of names from the beginning of the book and use cataphora resolution. Although this approach is not suitable for all purposes but owing to the structure of the English language and the use of correct grammar in books, this approach does show significant results but keeping in mind the general nature of the proposed system, it has not been used. Instead, cataphora resolution is done with help of the coreference chains generated as output by CoreNLP.

Reading co-reference chains

The output file from CoreNLP contains co-reference chains in text from which follow the pattern:

(a, b, [c, d]) -> (e, f, [g, h]) that is: "ANT" ->"CON"

where "***ANT***" refers to the antecedent and "***CON***" refers to the consequent in the co-reference chain, ***a*** and ***e*** are digits which represent the sentence number of the antecedent and the consequent respectively, ***b*** and ***f*** refer to the beginning token number of the antecedent and consequent restrictively. The ***c*** and ***d*** represent the starting and ending token offset for the antecedent and similar to g and h for the consequent. Due to the consistency of this pattern, a regular expression was used to extract only these chains and stored in a separate file for post-processing. The co-reference chains generated here contain a lot of mentions which do not contain any entity name so they can be pruned to keep the size down. The algorithm shown in Table 1 is used to prune unnecessary chains and the algorithm shown in Table 2 is used to resolve indirect referents.

Ranking metrics

The importance of an entity cannot be correctly measured by just the number of its occurrences, although counting the number of indirect references does give a better score but more factors need to be considered. The dispersion of an entity's referents can give an idea to the scope of the role it plays in the book. An entity which is mentioned very frequently and also is widely dispersed throughout the book is bound to be more important than an entity mentioned very frequently but very close together and in only certain parts of the book. The criteria to classify something as dispersed or concentrated, frequent or infrequent has to be decided differently based on each books length and type. As such, this problem presents itself to be treated as a fuzzy problem.

The position of the entity in the text can also give clues about its importance. A person mentioned in the title could have more prominence. Entities which come together frequently along with other important entities should also be noted. Once a sufficiently large number of books have been processed by the system

Table 1—Algorithm for pruning co-reference chains

Algorithm 1— Pruning Co-reference chains

```
for All Last Names in entity list do
    if (ANT== Last Name or CON==Last Name then
        Add the chain to final list Remove chain from current list
    else
        continue
for All First Names in entity list do
    if (ANT== First Name or CON==First Name then
        Add the chain to final list Remove chain from current list
    else
        Remove chain from current list
```

Table 2—Algorithm used to resolve indirect referents

Algorithm 2—Resolving Co-references

```
for All Names in list with Last Name==NULL do
    if (ANT == First Name && CON contains First Name) then
                replace words d-c in sentence a with CON Frequency(Name) +1

    else

        continue
```

tf-idf[14] can be used to further refine the importance of an entity in a book. The text-rank[15] algorithm may also be used to rank entities and a hybrid approach using all these metrics should be good enough for the systems purposes.

Generating recommendations

The system has so far produced a file containing the entities mention in the input book, ranked according to their importance using the metrics such as tf-idf and text-rank algorithm discussed above. A metadata store needs to be maintained which would contain entity ranks of all previously processed books. Now the system will look for books whose entity ranks are closest to the current book and generate recommendations. In future, we would like to merge the proposed system with collaborative filtering method to produce hybrid system that could generate effective recommendations based on both content and other user's behaviour.

Experimental results

Currently the system is able to take any text file as an input and tag entities and generate co-reference chains. An entity count is performed without resolving indirect mentions. This leads to the direct problem of mention fragmentations. The resulting list will treat single name mentions as separate mentions. For example, if a book's protagonist John Doe is mentioned 13 times by his full name but also as John 23 times, the output will list John Doe as a separate entity and John as a separate entity. After co-references are resolved, some of this can be corrected as some solitary John mentions will be mapped to John Doe.

Calculating the dispersion rating of an entity is currently a work in progress. Finding the most efficient ranking method is also being researched by the authors. The system was tested using seven books and the output shows some clear patterns. Here, we provide an example which compares our system output with the popular book recommendation sites. The book chosen for testing purpose is "How Google works". We have observed that the recommendations generated by *Amazon* and *whatshouldireadnext.com*, also contains the similar output as our system output with the book entitled "*Zero to One: Notes on Startups, or How to Build the Future*".

The system can be integrated with sentiment analysis and a summary can be generated which lists out which entities were mentioned in a positive or negative way in the book. The system can be used to generate character lists for book and augment sections of popular sites like shelfari and good reads. It can also be used to create an open source and more general alternative to Amazon's Xray, which provides brief summaries of people and organizations mentioned in a book but Xray requires input from the authors and publishers whereas the proposed system ranks entities automatically.

Conclusion

The system developed so far shows promising results about how NER can be added as another dimension to develop book recommender system. The results obtained show that there is significant correlation between recommended books by our system to that of shown by other popular book recommender systems and sites. The authors are hopeful of finding more quantifiable results for further research.

References

1. Bobadilla J, Ortega F, Hernando A and Gutierrez A, Recommender systems survey, *Knowledge-Based Systems*, 46 (2013) 109-132.

2. Burke R, Hybrid recommender systems: Survey and experiments, *User-Modeling and User-Aadapted Interaction*, 12 (4) (2002) 331–370.
3. Linden G, Smith B and York J, Amazon.com recommendations: Item-to-item collaborative filtering, *IEEE Internet Computing*, 7 (1) (2003) 76–80.
4. Ritter A, Clark S, Mausam and Etzioni O, Named entity recognition in Tweets: an experimental study, In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2011, pp. 1524-1534.
5. Atdag S and Labatut V, A comparison of named entity recognition tools applied to biographical texts, In *Proceedings of 2nd International Conference the Systems and Computer Science (ICSCS)*, IEEE, 2013, pp. 228–233.
6. Recasens M, Marneffe MC and Potts C, The life and death of discourse entities: Identifying singleton mentions, In *HLT-NAACL*, 2013, pp. 627–633.
7. Li H, Gu Y and Koul S, Review of digital library book recommendation models, Available at http://dx.doi.org/10.2139/ssrn.1513415
8. Pera M S, Nicole C and Ng Y K, Personalized book recommendations created by using social media data, In *Chiu D K W et al, Web Information Systems Engineering – WISE 2010 Workshops. WISE 2010. Lecture Notes in Computer Science*, 6724 (2011) 390-403.
9. Rana C and Jain S K, Building a book recommender system using time based content filtering, *WSEAS Transactions on Computers*, 11 (2) (2012) 27-33.
10. Vaz P C, Matos D M, Martins B and Calado P, Improving a hybrid literary book recommendation system through author ranking, In *Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries*. ACM, 2012.
11. http://www.goodreads.com
12. Finkel J R, Grenager T and Manning C, Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005, pp. 363–370.
13. Manning C D, Surdeanu M, Bauer J, Finkel J, Bethard S J and McClosky D, The Stanford CoreNLP natural language processing toolkit, In *Proceedings of 52nd Annual Meeting of the ACL: System Demonstrations*, 2014, pp. 55–60.
14. Ramos J, Using tf-idf to determine word relevance in document queries, In *Proceedings of the First Instructional Conference on Machine Learning*, 2003.
15. Mihalcea R and Tarau P, Textrank: Bringing order into texts, Association for Computational Linguistics, 2004.