Content Based Cross-Domain Recommendation Using Linked Open Data

Lakshman Jayaratne
University of Colombo School of Computing
Colombo 07, Sri Lanka
klj@ucsc.cmb.ac.lk

Abstract— A recommender system, irrespective of the approach that has been used to implement it suffers from the cold-start situation. Not being able to predict items to a new user due to not having access to his previous preferences, and not being able to recommend a new item to users due to not having any prior ratings on the particular item is the two cold-start problems. Even though content-based recommender systems are immune to item cold-start problem, they are comparatively less used due to lack of up-to-date data sources that provide item features and also due to the high amount of pre-processing required when using existing data sources for retrieving meta-data.

In this paper we present a content-based cross domain recommendation system using Linked Open Data to address the issue of cold-start situation. The evaluation proves that this approach can be used as a solution to a cold-start situation and also the prevailing issue of content-based recommender systems which forced them to take the backseat will no longer be applicable when Linked Open Data is used.

Keywords—recommender systems, cold-start situation, content-based, Collaborative Filtering, Vector Space Model, Linked Open Data

I. INTRODUCTION

Individual users accessing World Wide Web frequently encounter scenarios which leverage Information Filtering. Amidst the pair approaches, a commonly used approach for information filtering is Recommendation Systems. The purpose of recommender systems is to provide new items to users based on their previous preferences. Personalization is a vital research area when it comes to recommender systems. Hence randomly suggesting a list of recommendations cannot be consider as a valid solution. Two individuals are never the same. Their choices, preferences differ from each other. Therefore when making recommendations, it is important to give importance to individual user's choices.

Recommender systems can be classify into two approaches. Content-based recommender systems that recommend items based on the similarity of the new item's features with users past preferred item's features and Collaborative Filtering systems that recommend the user a set of items preferred by similar users. In both the approaches in order to personalize recommendations, a system should have access to users' previous preferences in the target domain for it to determine which items are more relevant to a particular user. But there are several occasions when the system doesn't possess any such data. In the domain of recommender systems, this is called the new user cold-start problem. Such situations can benefited by using the knowledge about user's preferences in auxiliary domains to recommend items in a target domain. This approach is known as Cross-Domain Recommendation.

In addition to this, there is another cold-start issue which is called the new item problem. This situation arises when the recommender system is incapable of recommending a newly added item to a user because it has no history of ratings given by users. New item problem is limited to Collaborative Filtering approach as they only recommend an item based on the fact that whether a similar user has rated the item or not. Content-based recommender systems are immune to this issue.

Out of the two main recommendation approaches, comparatively content-based recommender systems are less used because it takes an additional time and cost to collect and pre-process meta-data of items. Even though some data sets provide meta-data either they are not structured in a universal way or either they are incomplete or not up-to-date. Hence if we are to use a content-based recommender system, then a proper data source should be available which contains the vast amount of up-to-date data which is universally represented.

Linked Open Data can be used as a solution to this since they provide data in a universally structured format using RDF and thus these data are machine-readable. Linked Open Data sources such as DBpedia and WikiData provides feature based information on domains such as music, movies, books, places, people etc. and hence we can use these data sources to solve the afore mentioned the issue in content-based recommender systems.

This paper presents an approach to overcome the cold-star situation. We try to propose a cross domain recommendation a system to overcome the cold-start problem in the target domain and we use a content-based approach to avoid the new item cold-start problem in Collaborative Filtering based approaches. We also attempt to show that how Linked Open Data sources can be used as an approach that minimizes pre-processing effort applied in content-based recommender systems.

DOI: 10.5176/2251-3043_5.3.373

II. Background

A. Linked Open Data

With the exponential growth of data available on the web which are of different formats and the handling of these data became beyond once capabilities, Tim-Berners-Lee [1] proposed a new concept of publishing data that follows a global set of rules. Along with this global way of data publishing came the linked data concept.

In its simplest form, Linked Data can be interpreted as data that is publish on the web which contains the following properties.

- Machine readable
- Contain meta-data
- Linked with data sets of other external domains

Within the aspect of linked data, structured data are published on the web. Since they are linked with each other ultimately it can be interpreted as a Web of Data.

B. Recommender Systems

Content-based recommender systems **[2]** make use of content wise details of items. It can be item attributes, feedback provided by users or tags associated with it. These systems only focus on an individual user as oppose to collaborative filtering. New items are suggested based on the fact that how similar the new item is in comparison to what he has preferred in the past. To be qualified as a newly recommended item, it should share some similar attributes with the user's pre-preferred items.

In Collaborative Filtering approach, once the user have rated or liked items, based on his list of preferences a set of similar users are chosen who have the same set of interests [3]. New items are recommend to the users based on the preferences made by similar users. The hypothesis upon which this approach is based on is that the similar users will prefer similar items.

The whole recommendation process is done in 3 phases. User profiling phase to identify similar users next is obtaining the union of items chosen by similar users and assigning a weight to items regarding the importance of the item within the given set and the 3rd phase where items are suggested to the users based on their importance. Finding similar users is mostly done by using an algorithm like the nearest neighbor algorithm. There are several issues in recommender systems

- New User

Collaborative Filtering propose recommendations based on finding similar users based on current user's preferences. But if the user is new to the system; he has no initial preferences so it is not possible to retrieve similar users, to provide recommendations in such situations [4]. Content based recommender systems suffers from the same issue since no information about a user's preference and ends up in not being able to create a user profile.

- New Item

Collaborative Filtering systems focus on items that have been already preferred by a similar set of users. If a new item is added to the data set, none of the users have preferred it yet. Hence that item will not appear as a new recommendation in any recommend list [4].

Content-based systems have the upper hand on collaborative filtering systems as they will recommend any item to the user if it matches with the user profile unlike collaborative filtering systems which recommend only those items that are already preferred by similar users.

- Sparsity

Even though a particular item is highly rated but only amongst few of the users then such items will not be recommended to other users very often **[5]** also, if a small no of users persist unique preferences than most of the users then the recommendations provided to them will tend to be inaccurate.

- Over-Specialization

The main issue in is content-based recommender systems is that they tend to be over-specialized. An unexpected item or a novel item to the existing user profile will not be recommended at all **[2].**

C. Cross-Domain Recommender Systems

The main objective of a cross domain recommendation the system is to derive information from a source domain and to use that information in an appropriate manner, to provide recommendations for items in a target domain. Hence cross-domain recommendation approaches use transferring or combining the knowledge amongst domains to derive recommendations. Even though some large scale providers do use multiple domain concepts within their recommender systems, they do not combine the knowledge amongst the domains and hence ends up keeping their recommender systems as same as a single domain recommender systems. [6]

Cross-domain recommendation techniques are approached to solve several research problems that exist within recommendations. Cold-Start issue being one of them, the rest can be listed as enhancing accuracy, adding diversity and creating more pair user models **[7].**

D. Cold-Start Problem in Recommender Systems

When a recommender system encounters a new user, the system faces the difficulty in suggesting a recommendation to the user as no information about his preference history is available, to derive new predictions. Regardless of what the approach is all Recommender Systems face this problem. In addition to the cold-start problem, some Recommender Systems do not have enough amounts of data on existing users nor items to provide accurate recommendations. The need of additional sources of information is a must during these scenarios.

Hence some Recommender Systems comes up with solutions such as suggesting a random set of items to new users or suggesting a newly released set of items to the users [8]. But none of the above solutions guarantees whether the recommender system is going to end up suggesting a list that will actually be of any use to the users. Another approach to handle this situation is to aggregate additional information sources that contain data about the new user's preferences, provided that these information sources will actually be of use in suggesting a new set of items to the users. That is, the information sources should relate with each other in some manner.

III. Literature Survey

A. Addressing the Cold Start Situation

As a solution to the cold-start problem several researchers have observed how data in multiple source domains can be used to provide recommendations for a particular user in a target domain. Collaborative filtering can be easily used in this approach since it focuses only on the similar users in the source domain and provide the new user with target domain items which are already preferred by those similar users?

Another approach which is a modification of the basic collaborative filtering approach proposed by Larson et al. [9] by finding similar users and items based on tags. Where users have associated the source domain items with several tags and similar users are obtained by the tag similarities amongst them. Again using collaborative filtering theory, items from similar users' preferences are recommended to the new user. Taking this step another step forward Manuel et al. [10] have proposed a recommendation system where item based similarities amongst different domains are obtained by using the tag-based an approach where items that have retrieved same tags from users in separate domains are considered as equal and for a news user based on the tags he has associated with items in a source domain, target domain items that have been assigned with the similar tags are recommended.

Nevertheless the unavoidable item based cold-start the problem still prevails even though it solves user based cold-start problem. Another issue as addressed by Saheb et al. [11] is the importance of the amount of information that can be retrieved by the source domains where they have found out that cross domain the recommendation in cold-start context cannot be carried out properly when the number of data on the source domain is low. This situation limits the researcher from identifying what data are supposedly noisy and what aren't. Hence it is important to have a reasonable amount of user preferences in the source domain to achieve accurate predictions in the target domain.

B. Linked Open Data in Recommender Systems

An existing issue in the content-based recommender systems is to find a proper data set which provides item information (meta-data). Due to the lack of this information most of the content-based recommender systems are using social tags allocated by users [19]. But this process of converting user allocated tags into a knowledge source is not a simple task, as unless the users are pre-requested to allocate a pre-defined set of tags, they are free to allocate any word in the vocabulary which results in researchers ending up creating ontologies. Therefore most of the Recommender Systems are based on Collaborative Filtering.

Linked Open Data cloud provides different data sources on several domains as shown in Figure I. WikiData, DBpedia, LinkedMDB, MovieLens, and MusicBrianz are some of the repositories that contribute to the LOD concept.

- WikiData

WikiData is a free project where any user can collaborate to and retrieve data from. It is used to facilitate Wikipedia data and to expose its data in RDF format over several endpoints. It provides links connected to other RDF graphs and hence data from WikiData can be interlinked with other Linked Open Data sources [20]. It acts as a contributor to Wikipedia and provides all the information inherent to items and therefore contains a complete set of attributes as shown in Figure II.

C. Linked Open Data in Recommender Systems

In content-based recommendation systems, there is a separate module named Content Analyzer which is used to extract item descriptions and to represent them in a vector using their keywords [12]. To extract the relevant keywords, these Context Analyzers should engage in Natural Language Processing methodologies. The advantage in using LOD is that the information/keywords that had to be extracted by using a Content Analyzer, is already structured and that too in a universal standard manner. Therefore the number of pre-processing steps involved can be reduced and kept to a minimum. This results in having much efficient Recommender Systems.

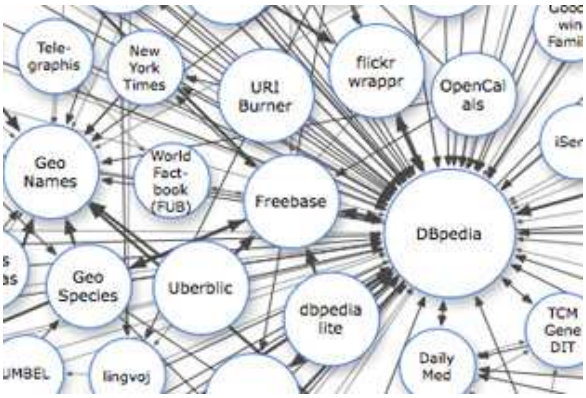

Fig. I. LOD Cloud

Mirizzi et al. [13] have described how LOD concept is used as the data set in a recommender system by using vector space model concept. The assumption that they have used is that if two movies share the same set of features (genre, actors, etc.) then they are similar. They represent movies that a user has preferred already in a 3D matrix where each slice stands for a particular property of the movie and the movie is the vector that includes each of these properties. Once the movies are represented in this manner, the degree of similarities between two movies can be obtained by their cosine similarity. A new the movie is suggested to a user based on the overall similarity that it has against the entire user profile. The researchers have used data sets from DBpedia, LinkedMDB and Freebase and via precision-recall matrices they have found that the proposed approach performs better than collaborative approaches and keyword based approaches as Linked Data Semantic Distance approach.

Heitmann et al. [14] have addressed a situation where Linked Open Data cloud can be used to provide recommendations. Rather than retrieving item based features from the LOD cloud, they have focused on obtaining user based information. They have argued that LOD data cloud-based communities such as Friend-of-a-friend (FOAF) and MySpace can be used to retrieve information about users of whom the recommender systems are unaware of. In the situation of a new user, the recommender system will find the FOAF page of the particular user and then find his preferences using RDF links. FOAF pages display RDF format based information of people. Hence Heitmann argues that a user's preferred music artist can be easily found by querying his FOAF page. If the user doesn't

have a FOAF page, then the recommender system will query for all the people who have the new user as a friend in their MySpace account. This information is added by default to each user's FOAF page. The system will then retrieve the preferences of the new user's friends and recommend the same set of items to the new user. Their recommender system consists of a reasonable amount of average precision and recall but the issue is the data sources they have used is limited to only a few amount of users.

property	object
instance of	television series
instance of	television anime
instance of	Japanese TV series
genre	Shōjo manga
genre	science fiction comics
genre	neo-noir
genre	comedy drama anime and manga
IMDb identifier	tt0213338
Commons category	Cowboy Bebop
original network	TV Tokyo

Fig. II. WikiData extract

When it comes to Cross-Domain recommendation using linked-data one of the existing research that has been carried out by Contador et al. is obtaining musicians given a particular place of interest [15]. In their approach, they have used graph based algorithms to obtain related musicians. Using DBpedia data-sets that are relevant to the two domains they have defined how these domains can be related to each other. Next, using that information, they have created a class network which is a directed acyclic graph (DAG). Next they have queried DBpedia to obtain the immediate results as per the DAG that matches to a given place of interest instance (i.e. - Given a name of a place of interest, the century it was built on). The results of the first query are used as inputs for the second query and the process is carried out until a result from the target instance (musicians) is obtained. Finally, after obtaining a set of target instances, they have assigned weights to the edges (relations) of the graph, based on user preference and presented the highest ranked instances of the target domain as the results. The experiments have given a positive result of over 80% precision accuracy for the top 5 recommendations of musicians.

D. Content-Based Cross Domain Recommender Systems

There are some domains that share some common features amongst each other. Movies can be related with Books by the script writer, genre. Movies and Music can be related by music composer, album. As content-based recommendation systems focus on features or attributes, they will first obtain how features of separate domains can be related. Then an individual user's preferred items in a source domain will be extracted [6]. A user profile will be generated based only on the values provided for features that can be inter-related. Ex: movie genres comedy, drama, etc. Based on the user profile, items from the target domain will be suggested to the user, which matches best with the user profile.

By taking a content-based approach on Cross Domain Recommendations, Ricci et al. have used music recommendations based on place of interest (POI). They have assigned particular tags that can be used to describe both songs and POIs [15]. So when a user selects a particular place, they compare the tags that are associated with it against the tags that are given to the list of songs in the dataset and ends up suggesting the songs that have the highest degree of similarity with the given set of tags, which means those that are co-related with the user's choice of POI.

IV. Methodology

The main purpose of this thesis is to find a solution to the cold-start situation by using a content-based cross-domain recommender system. The functional aspect of this recommender system will be as follows. The recommender system will use a new user's preferences in the movie domain and retrieve the necessary features of the movies by querying the Linked Open Data cloud. Then it creates a user profile that has the most significant set of movies belonging to the user. Then by analyzing against the user profile, it would find Books that matches the user profile and recommend the most similar items to the user thereby solving the cold-start situation.

A. Domain Correlation

Items within two domains may inter-relate with each other. But this doesn't imply that all the features of both domains correlate. Since the research use Movies as the source domain and Books as the target domain, only the important correlated features within the two domains should be taken into account

When building relations between domains, attributes such as name, producer, and publisher are considered as noisy attributes. They provide no means about the correlations amongst the two domains. Hence there is no pointing in extracting such data and storing them as it will end up consuming both time and space efficiency of the system.

For the two domains, the following attributes will be considered as related due to the provided reasons.

- Genre

Movies and Books can both share the same genre which can be used to generate a similarity among the two of them. A user who prefers movies of 'Horror' genre tends to prefer books of the same genre as well.

- Based On

If a user likes the movie Barbie as Rapunzel, then it is possible for him to prefer Rapunzel book as well. Which movie is based on which book or vice-versa can be obtained by this attribute.

- Writer/Author

Movies and books can have a correlation amongst them via the Author property. If a user likes the movie, 'The Shaggy Dog' by Felix Salten it would be better to recommend him a book like 'Perri' which it authored by the same writer.

Winoto et al. [16] provides why the knowledge in one domain can be aggregated to recommend items in another domain using the above-mentioned properties. The evaluations

they've conducted proved that there exist a relationship amongst user's preferences in multiple domains when they share common attributes.

Based on the above-retrieved values we inter-relate the two domains as shown in Figure III. The highlighted attributes belong to the movie domain and the rest belong to the target domain.

B. Retrieve Item Features

DBpedia and WikiData contain data in RDF format. Since they provide feature wise information of items, it is important to query these data sources and obtain relevant features. RDF format presents data in triples. These triples act in the form of subject-predicate-object. For the users to query RDF triples, LOD data sources have made these data available at their end points and WikiData's end point is an SPARQL endpoint. Hence by using SPARQL we can retrieve the relevant data. The RDF format is capable of linking one subject-predicate-object triple with another.

To retrieve data, we have used Apache Jena API which is an open source Semantic Web framework for Java. It provides an API to extract data from and write to RDF graphs. Using Apache Jena and DBpedia SPARQL endpoint we retrieved per item data from WikiData.

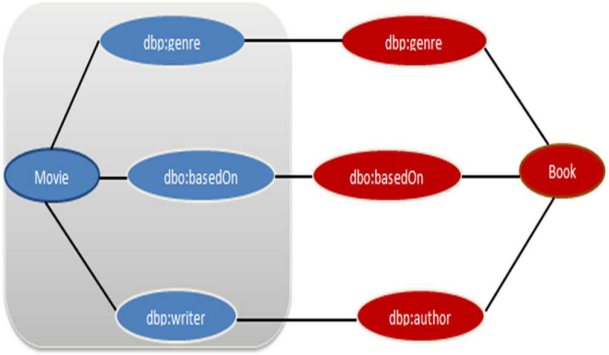

Fig. III. Domain Correlation

The data set provided by the ESWC 2015 Linked Open Data challenge [21], contains user preferences for movies and books which are mapped with their relevant DBpedia link. Due to the lack of data available in DBpedia for the Genre property for some of the items we focused on retrieving WikiData information. To do this, as shown in Figure IV the particular WikiData link of each of the item is retrieved by using DBpedia SameAs property which states the external links to a particular item.

When retrieving item genres, the main issue to overcome was genre sub categories. For Book domain, Wikidata genres are originated from 2 main classes, the literary genre and genre as shown in Figure V. Movie domain has genres originated from literary genre and _lm genre classes. Retrieving all the pair genres of each Book and Movie is a must. Hence while querying, we focused on all the possible genre subclasses.

In addition to the fact that genre is originated from the given base classes, genre property usually consists of transitive properties. As it is emphasized in Figure VI a book when we refer to its pair genre property will only highlight that it belongs to Vampire genre whereas we have to go one step further to know that it belongs to Horror genre as well. It is important to have the main genre class an item belongs to compare the item against a set of some other items.

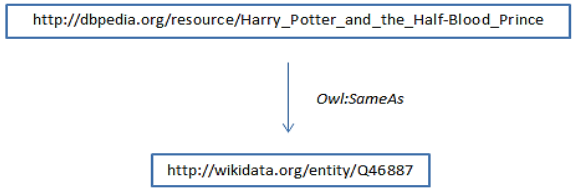

Fig. IV. DBpedia to WikiData Mapping

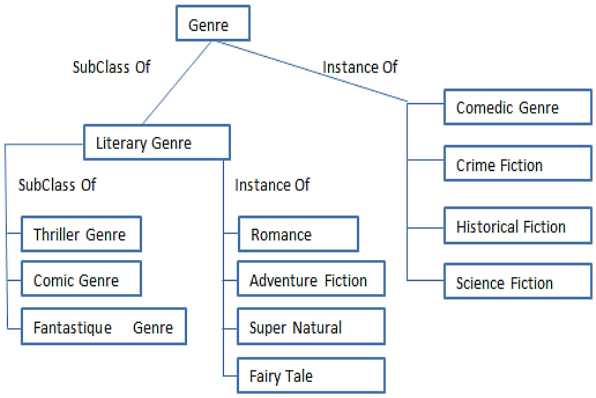

Fig. V. Genre Hierarchy

- Retrieve Target Domain Item Features

We assume that a Book recommender system is in posses with its book data set and hence it is not important to query for book features real time for each new user. Therefore as shown in Figure VII we will first extract the book features from LOD sources and store them so that recommendation process can be carried out relatively quickly.

- Retrieve Source Domain Item Features

When addressing user preferences in the movie domain we will only take binary ratings into account. Hence we assume that either a user thoroughly prefers a movie or he doesn't like it at all. The notion of the difference in considering a movie as an excellent movie, a good movie, an average movie is not focused on this research. Hence this section will not focus on how many ratings are provided for a movie.

Under the assumption that it is not important for a Book recommender system to store details about other non-relevant domains, we extract user's preferred movie details from LOD sources in real-time as shown in Figure VIII.

Fig. VI. Genre Transitivity

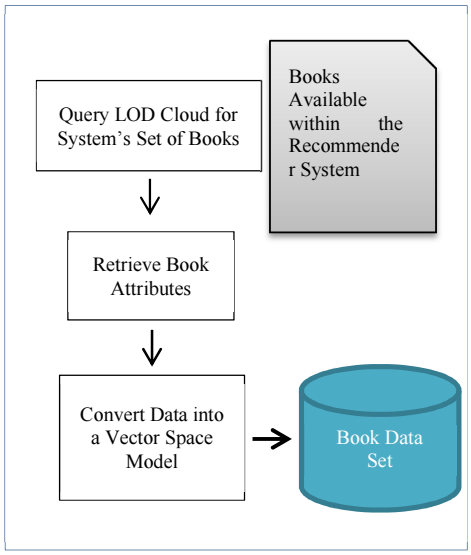

Fig. VII. Creating Book Data Set

Storing details of non-relevant domains will result in additional storage and since the Book recommender system is not aware of the fact that what would be the movies that are preferred by a user, it will have to store limitless amount of movie information which is not necessary. Another reason for designing the system in this way is to enhance the scalability in source domains. When data of the source domain is retrieved online rather than locally storing them, the system can easily extend the number of its source domains.

Users' initial preferences tend to be noisy. Hence relevant movies should be extracted from them. To do this, it is important to apply mathematical functions to them so that items can be selected based on some criteria, such as similarity or diversity. Hence we use vector space model to represent user's initial movie preferences

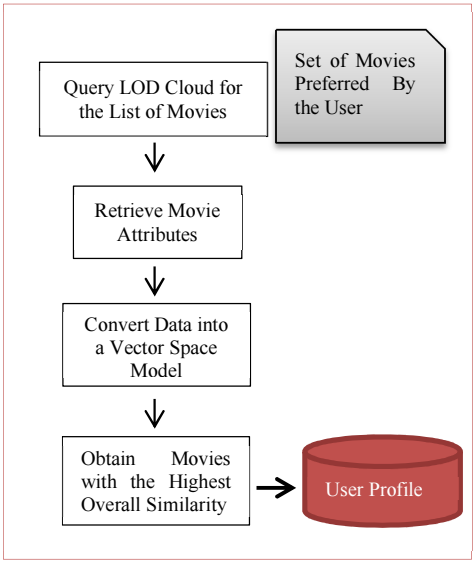

Fig. VIII. Creating User Profile

C. Creating User Profile

Vector Space Model (VSM) is used to represent documents in a multi-dimensional algebraic manner to apply mathematical functions to the document. It represents a document as a vector. The vector is capable of containing sub-vectors within it. Each attribute of the document is considered as an individual vector. In the context of the given problem of the thesis, an item (movie/book) is considered as a vector, and its attributes such as genre and author will be sub-vectors. Both genre and author vectors can contain multiple values since an item can be authored by several people as well as can belong to several genres. Each item is considered as a point in the vector space and it assumes that the most relevant or similar items are the nearest ones. To compare a Book with a Movie, their relevant sub vectors are compared with each other and similarity measured using Cosine Similarity and TF-IDF weights.

This model has been approached for recommender systems by [13] for a single domain content-based recommender system and they have proved that this method when compared to collaborative filtering approaches as well as the most frequently used Linked Data Semantic Distance approach that is used in content-based recommender systems when Linked Open Data sources are used. Therefore, we will be adapting the same Vector Space Model concept into our research.

A user may have preferred many items in the source domain which will result in too much information/items. Hence the user profile should consist of a reasonable amount of items which should represent the particular user. These items should stand for both similarity and diversity.

If the user's initial movie preference list only contains movies of thriller and horror genres, then it might not be ideal to suggest him items of some other genre such as comedy. In the context of this research, we do not focus on serendipity. Hence we assume that a user's preferred set of items reflects his personal choices and behavior pattern. Therefore the user profile should contain similar items while ensuring the fact that, it doesn't become too similar.

If the user has the majority of movies as Harry Potter movies within his user profile, then the recommender system will keep on recommending Harry Potter books to him. Therefore there would be 0% of novelty & diversity. Hence while maintaining the similarity; the recommender system should include some amount of diversity to the list of recommendations. The TF-IDF weighting scheme handles this by not giving the entire prominence to items that contain features common to majority of movies in the user profile.

TF can be represented as $tf_{t,d}$ is the frequency of a particular term *t* within a given document *d*. The equation for TF weight is as follows. In our approach, this is the number of times an attribute value (genre/writer) occurs within a single item (movie). For a given movie, a genre or the writer's name can only occur once or not occur at all. Hence it will be either 0 or 1.

$$w_{t,d} = \begin{cases} 1 + \log_{10} \mathrm{tf}_{t,d}, & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Document Frequency (DF) gives the number of documents that contain a particular term *t* and is represented as df_t. Inverse Document Frequency (IDF) on the other hand reduce the prominence of highly used terms and gives an important to less frequently used items as well.

$$\mathrm{idf}_t = \log_{10}(N/\mathrm{df}_t) \quad (2)$$

TF-IDF weight is the product of both TF and IDF weights and provides the term specific weight of the system and this value is used in obtaining the Cosine similarity.

$$w_{t,d} = \log(1+\mathrm{tf}_{t,d}) \times \log_{10}(N/\mathrm{df}_t) \quad (3)$$

Cosine similarity provides means of calculating the similarity between two vectors. We can leverage it to compute the similarity between two movies based on a particular feature *p*. As we use TF-IDF weights in calculating the Cosine similarity, the user profile will contain a reasonable amount of similarity as well as diversity and thereby by solving the overspecialization issue.

$$sim^p(\vec{m}_i, \vec{m}_j) = \frac{\vec{m}_i \bullet \vec{m}_j}{|\vec{m}_i||\vec{m}_j|} = \frac{\vec{m}_i}{|\vec{m}_i|} \bullet \frac{\vec{m}_j}{|\vec{m}_j|} = \frac{\sum_{n=1}^{t} w_{n,i,p} \cdot w_{n,j,p}}{\sqrt{\sum_{n=1}^{t} w_{n,i,p}^2} \cdot \sqrt{\sum_{n=1}^{t} w_{n,j,p}^2}} \quad (4)$$

$\vec{m}_i$ - *item i*,

$w_{n,i,p}$ - *tf-idf weight of item i based on attribute p*

$\vec{m}_j$ - *item j*,

$w_{n,j,p}$ - *tf-idf weight of item j based on attribute p*

By comparing the similarity of every movie in the user's the initial preference list with the rest of the movies in the same list, we can find out what are the movies that give the highest similarity value with the TF-IDF weighting scheme. While creating the user profile, it is necessary to compare both the genre vector as well as the author vector of each a movie on the list of preferred movies. Out of the items in the user's initial preference list, top 10 items will be added into the user profile. Therefore the movies in the user profile contain a reasonable amount of similarity as well as diversity amongst them.

The following equation calculates the overall similarity value for an item $\vec{m}_i$ against the entire profile u [13]. Here, α_p is the weight allocated to the attribute p based on its significance in the user profile.

$$\bar{r}(u, m_i) = \frac{\sum_{m_j \in profile(u)} v_j \cdot \frac{\sum_p \alpha_p \cdot sim^p(m_j, m_i)}{\sum_p \alpha_p}}{|profile(u)|} \quad (5)$$

D. Measuring Book Similarity

To provide the most relevant list of recommendations compared with the profile of the user, we use the same Vector Space Model that is representing the user profile. Applying the Vector Space Model to compute the most similar items available in the user profile is usually done for the items in the same domain. But, since we only used the attributes that are common to both the domains it is possible to compute the similarity between the movie profile of the user and the newly selected items in the book domain.

To determine whether a new book is highly related with the existing user profile, its similarity value should be obtained based on the entire user profile. Hence user profile is compared against every single Book that is stored in the Book data set. For the comparing purpose, a similarity metric is used.

To obtain the book similarity against the user profile we use Jaccard Similarity index. This similarity measure only focus on attribute wise similarity between items and not TF-IDF weighted similarity.

$$Jaccard\ Similarity = \frac{Recommended\ Items \cap Preferred\ Items}{Recommended\ Items \cup Preferred\ Items} \quad (6)$$

E. Top-N Recommendation

After obtaining the similarity value of every book against the user profile, the books will be sorted in descending order on their similarity and the top N books in the list would be selected as the recommended books to the users and will be added to the recommendation list as shown in Figure IX.

V. Evaluation

For the purpose of evaluating the system, we used the data set provided by ESWC 2015 linked open data challenge [21]. This data set was collected via Facebook which contains user's preferences in respective domains. Hence the dataset contains binary preferences of users. To limit the collaborative approaches to address on the cold-start problem, they have limited the amount of information given to the target domain.

Here we have used those users who have preferences in both book domains and movie domains. Table I. shows an overview of the dataset. The book data and movie data both are mapped to the relevant DBpedia link where we additionally mapped them to the WikiData link as well to gather more data.

When it comes to cross-domain recommender systems in cold-start situations, since there are 0 items of the target domain available for the recommendation task, training data set is the user's source domain preferences, and testing set is the target domain items. This evaluation approach is named as leave all users out and especially used for evaluating cross domain recommender systems in the cold-start situation. [7].

The evaluation is done based on accuracy, novelty and diversity matrices. The following evaluation matrices were used for the purpose of evaluating the system. Precision, Recall, F1 Score to measure the performance of the recommender system and Entropy-based novelty to show that items are recommended to users despite of their popularity and aggregate diversity measures to show that the system provides a good coverage all over.

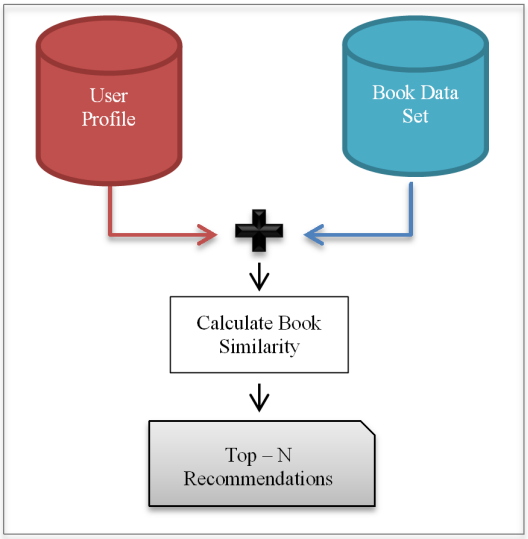

Fig. IX. List of Recommended Books

A. Entropy Based Novelty

$$EBN_u@N = -\sum_{i \epsilon L_u} p_i . \log_2 p_i \quad (8)$$

where $p_i = \frac{|\{u \epsilon U \mid i \text{ is relevant to } u\}|}{|U|}$

Property	Data
No. of Users (have preferences in both domains)	250
Average no. of movies per user	23
Average no. of books per user	8
Total no. of movies	5389
Total no. of books	3225

TABLE I. DATASET ANALYSIS

An issue within collaborative recommender systems is that without combining with a hybrid model they tend to recommend popular items to users and hence some items will never be encountered by them [18]. Therefore they tend to suffer from item-cold start problem. Entropy-based novelty is a measure that gives an insight on the ability of how successful/unsuccessful the recommender system is in recommending less popular items. A low value indicates that system provides less popular items instead of being biased on the popularity. We obtained an average EBN@20 score of 1.1682.

To observe the quality of the entropy based novelty measure of the recommender system, we assumed that the system would be recommending items that are preferred by at least three users. A scenario based on our assumption will provide an EBN@20 score of minimum 1.5314. The average EBN value that our system actually scored is less than this value. Hence on average our system suggests items that have a popularity of fewer than three users. As per the data set, the number of users who have preferred a single book varies in the range of 0 to 62. Hence it can be presumed that our approach is not biased on item popularity.

B. Performance

Precision measures the capability of the recommender system to provide items that are of users' actual choice. Recall measures the probability of a relevant item being recommended.

Two values are always measured with along with the number of items recommended. Hence Precision and Recall values are obtained at the length n of the recommended list. These values are mostly used when the data set contains binary preferences. The two values have an inverse relationship amongst each other. When the recall value increases precision decreases. Hence F1 score is used to combine both to a single value so that comparison becomes much easier. [22]

It acts as an accuracy measure for the system. We have measured precision@N recall@N and F1_Score@N when N=15 and N=20. We evaluated the system using two similarity matrices Cosine similarity and Jaccard similarity where Cosine Similarity decreases with an average F1@20 value of 0.02768.

Table II shows the overall precision, recall and F1 values obtained for the entire set of users

Precision value tends to become low in this situation due to the fact that the provided dataset on average has around 8 items in the target domain (Even if the recommender system suggests the correct eight items precision value would still be 0.4 when N=20). On average the system recommends 0-2 items that have been preferred by the user.

VI. CONCLUSION

The issue of overcoming the cold-start problem in recommendation systems to provide recommendations to a new user was addressed in the thesis. Since we used a content-based approach the item based cold-start problem was implicitly handled because content-based recommender systems do not rely on item popularity or item ratings. Content-based recommender systems mainly focus on item's with features only.

We used Linked Open Data as our data source. Content-based recommender systems are less used due to the reason of not having enough data sources. We observed that LOD cloud provides up-to-date meta-data of several domains that are universally structured and hence retrieving LOD data doesn't need any steps that complicate the performance of the recommender system. We showed that it is as simple as executing a single SPARQL query. Moreover, LOD data is available at the real time hence there is no need to store source domain item details within the system itself.

	N=15	N=20
Precision@N	0.02477	0.02889
Recall@N	0.06263	0.07231
F1 Score@N	0.03550	0.04129

TABLE II. PERFORMANCE MEASURES

In our research, we used Cross-Domain recommendation to overcome user based cold-start problem since the recommender system can provide a user an item of the target domain by using a source domain where the two domains have

overlapping features. After evaluation, the results showed that the approach we used to solve the cold-start problem using content-based Cross-domain recommendation indeed provides a reasonable the amount of accuracy while recommending novel items to users and thus solving both item and user based cold-start problem.

In this paper, we used attributes that directly bears relationships with both domains. (i.e. Genre, Author and BasedOn). In future, we can use DBpedia pair property to retrieve more details about items. By aggregating natural language processing techniques along with the proposed approach important keywords or bag of words can be retrieved from both the domains' items which will improve the accuracy further more. In addition to this DBpedia term, Subject can also be used along with NLP techniques to determine which category the item belongs to and hence use another feature while comparing the item similarity.

REFERENCES

[1] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data-the story so far", *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, pp.205-227, 2009.

[2] P. Lops, M. De Gemmis, and G. Semeraro, Content-based recommender systems: State of the art and trends", in *Recommender systems handbook*,Springer,2011,pp.73-105.

[3] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms",*ACM Transactions on Information Systems (TOIS)*, vol. 22, no.1,pp.143-177,2004.

[4] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques", *Advances in artificial intelligence*, vol. 2009, p. 4, 2009.

[5] I. Fernandez-Tobias, I. Cantador, M. Kaminskas, and F. Ricci, "A generic semantic-based framework for cross-domain recommendation", in *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, ACM, 2011, pp. 25-32.

[6] I. Fernandez-Tobias, I. Cantador, M. Kaminskas, and F. Ricci, "Cross- domain recommender systems: A survey of the state of the art", in *Spanish Conference on Information Retrieval*, 2012.

[7] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, *Recommender Systems Handbook*, 1st. New York, NY, USA: Springer-Verlag New York,Inc.,2010, isbn: 0387858199, 9780387858197

[8] I. Cantador, I. Fernandez-Tobias, S. Berkovsky, and P. Cremonesi, "Cross domain recommender systems"

[9] Y. Shi, M. Larson, and A. Hanjalic, "Tags as bridges between domains:

Improving recommendation with tag-induced cross-domain collaborative filtering", in *User Modeling, Adaption and Personalization*, Springer, 2011, pp. 305-316.

[10] M. Enrich, M. Braunhofer, and F. Ricci, "Cold-start management with crossdomain collaborative filtering and tags", in *E-Commerce and Web Technologies*, Springer, 2013, pp. 101-112.

[11] S. Sahebi and P. Brusilovsky, "Cross-domain collaborative recommendation in a cold-start context: The impact of user profile size on the quality of recommendation", *in User Modeling, Adaptation, and Personalization*, Springer, 2013, pp. 289-295.

[12] V. C. Ostuni, G. Gentile, T. Di Noia, R. Mirizzi, D. Romito, and E. Di

Sciascio, "Mobile movie recommendations with linked data", in *Availability, reliability, and security in information systems and HCI*, Springer, 2013, pp. 400-415.

[13] T. Di Noia, R. Mirizzi, V. C. Ostuni, D. Romito, and M. Zanker, Linked open data to support content-based recommender systems", in *Proceedings of the 8th International Conference on Semantic Systems*, ACM, 2012, pp. 1-8.

[14] B. Heitmann and C. Hayes, "Using linked data to build open, collaborative recommender systems.", in *AAAI Spring Symposium: Linked Data Meets Articial Intelligence*, 2010, pp. 76-81.

[15] M. Kaminskas, I. Fernandez-Tobas, F. Ricci, and I. Cantador,

"Knowledgebased music retrieval for places of interest", in *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, ACM, 2012, pp. 19-24.

[16] P. Winoto and T. Tang, "If you like the devil wears prada the book, will you also enjoy the devil wears prada the movie? a study of cross-domain

recommendations", *New Generation Computing*, vol. 26, no. 3, pp. 209-225, 2008.

[17] G. Adomavicius and Y. Kwon, "Improving aggregate recommendation diversity using ranking-based techniques", *Knowledge and Data Engineering, IEEE Transactions on*, vol. 24, no. 5, pp. 896-911, 2012.

[18] T. Di Noia and V. C. Ostuni, "Recommender systems and linked open data", in *Reasoning Web. Web Logic Rules*, Springer, 2015, pp. 88-113.

[19] M. Kaminskas and F. Ricci, 'Location-adapted music recommendation using tags', in *User Modeling, Adaption and Personalization*, Springer, 2011, pp. 183--194.

[20] Wikidata.org, "Wikidata:Introduction - Wikidata", 2016. [Online]. Available: https://www.wikidata.org/wiki/Wikidata:Introduction. [Accessed: 27- Jan- 2016].

[21] Sisinflab.poliba.it, "2nd Linked Open Data-enabled Recommender Systems Challenge", 2016. [Online]. Available: http://sisinflab.poliba.it/events/lod-recsys-challenge-2015/. [Accessed: 27- Jan- 2016].

[22] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems", ACM Transactions on Information Systems (TOIS), vol. 22, no. 1, pp. 5-53, 2004.

Dr. Lakshman Jayaratne - (Ph.D (UWS), B.Sc.(SL), MACS, MCS(SL), and MIEEE) obtained his B.Sc (Hons) in Computer Science from the University of Colombo (UCSC), Sri Lanka in 1992. He obtained his PhD degree in Information Technology in 2006 from the University of Western Sydney, Sydney, Australia. He is working as a Senior Lecturer at the UCSC, University of Colombo. He was the President of the IEEE Chapter of Sri Lankan in 2012. He has wide experience in actively engaging in IT consultancies for public and private sector organizations in Sri Lanka. He was worked as a Research Advisor to Ministry of Defense, Sri Lanka. He Awarded in Recognition of Excellence in Research in the year 2013 and 2015 at Postgraduate Convocation of University of Colombo, Sri Lanka. His research interest includes Multimedia Information Management, Multimedia Databases, Intelligent Human-Web Interaction, Web Information Management and Retrieval, and Web Search Optimization. Also his research interest includes Audio Music Monitoring for Radio Broadcasting and Computational Approach to Train on Music Notations for Visually Impaired in Sri Lanka.