Cross Domain Recommender For Conference Attendees

Kehao Xu
School of information sciences
University of Pittsburgh
Pittsburgh PA 15260, USA
kex3@pitt.edu

Abstract

This recommendation system is for people who are in the same conference. The system adopted Google custom search API to get information that uses to calculate a score for recommendation. The information is separated into five domains, Google Scholar, Researchgate, Twitter, Wiki and general search result. The score consists of two parts, content similarity and network similarity. User can view a recommendation list for a certain person and the dashboard that consist of what system get from Google search for everyone in the list.

1. Introduction

It is a challenge to do people recommendation task based on various uncertain data sources or lack of data. In this system, we try to build up a system that query user information from Google search. Based on the search result, the system can retrieve the user's information from multiple source, like Google Scholar, Researchgate. With certain and rich information, it will be easier to do people recommendation task.

To achieve this goal, the system will get the attendee list from Conference Navigator 3[1] (CN3) and use the name of attendee as a query to search in Google search. This search result will be processed to five different domains then modeling the processed data to get the final recommendation.

In this paper, we will introduce the system structure, the method to process the data, the model to make the recommendation, the interface of website and the evaluation of the recommendation.

2. Related Works

In cross-domain recommendation, the goal is to use various source domain information to recommend items in the target domains[2]. However, even there are many method to do the recommendation, how to get various source information is a problem. To solve this problem, we choose Google search do this job. People now is using and more relying on the Internet for many years. It makes the Internet a huge information source in various domain, and Google search is a good tool that can retrieve the related information from the Internet.

First of all, we need to decide what the target domains is and what the various source domains are. The target domains is to make the recommendation for the attendees in the same conference. The attendee list is got from the CN3[1] website. Various source domains are Google scholar, Researchgate, which are the most famous and complete academic source, Twitter, the most popular social tool, Wiki, the richest information website and general search result, which means the other results which are not that special but relative.

After retrieving the information for the system, there need to be some algorithms and models to make the recommendation. The score that used to make the recommendation is linear combined with content similarity and network similarity in the same weight. The content similarity uses the Vector Space Model[3] (VSM) to calculate the score. The data to be calculated is from the character data in those five domains. The network similarity uses the Jaccard's coefficient[4]. The data is from the network information like coauthor or friends in those five domains. All the sub scores in those two parts are also linear combined in the same weight.

To provide this cross domain recommendation system to users, we make the system a web application. The front end is based on the HTML5 and the interface is using the template from HTML5UP[5] . In a conference, the attendee may not contact with his computer all the time, so providing a website with a good display in smart phone is important. The HTML5 and this template are good at this.

3. Approach

3.1 System Structure

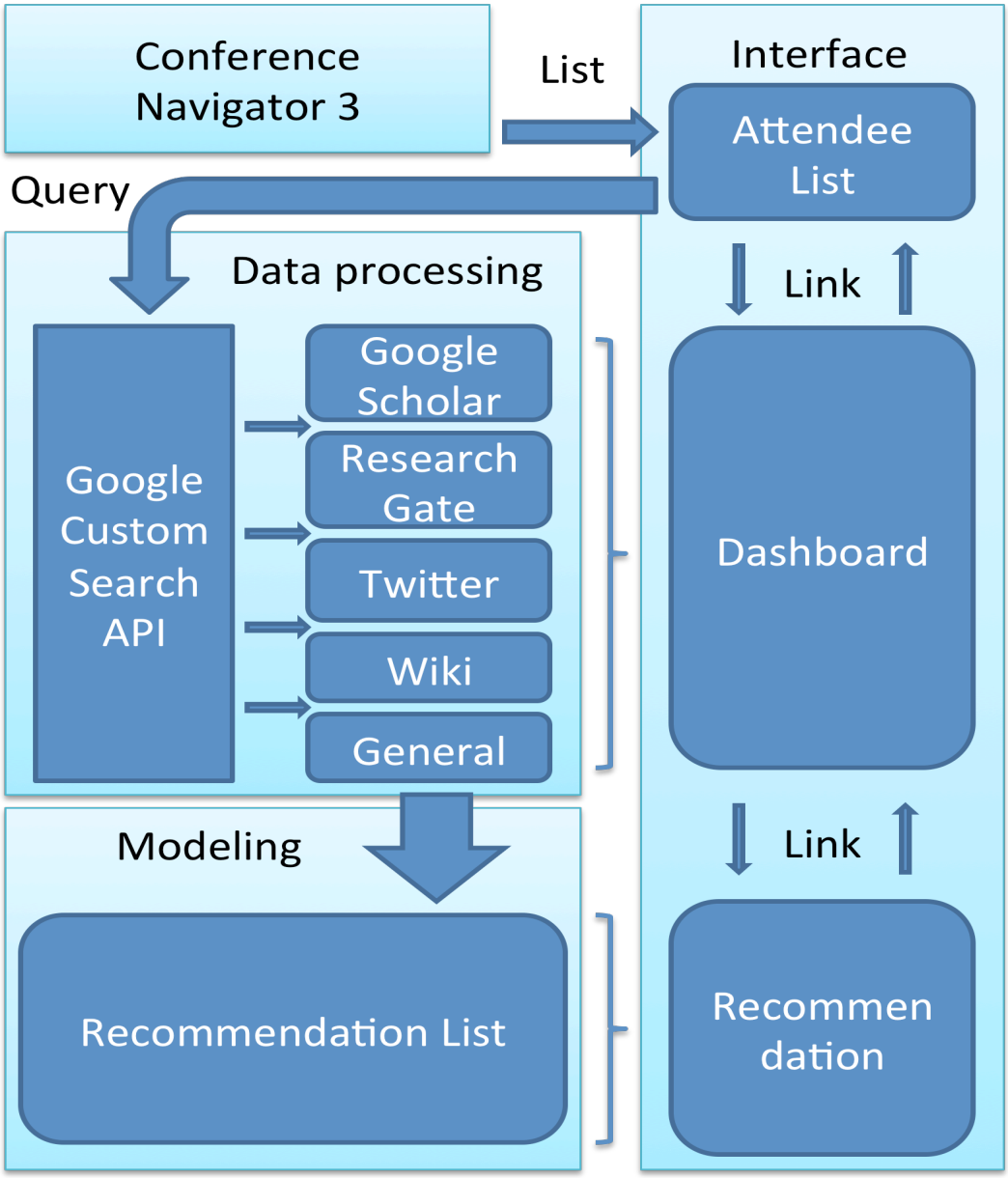

Figure 1 : The Structure Of System

To make recommendation, there should be a score to rate people. To get people been rated, it needs information that related to them. The system consists of three subsystems, the picture 1 is showing the whole structure of the system. Out of the system, there is CN3, which is the attendee list coming from. For the subsystems, the first is data processing, using the Google search to get the related information and separating and parsing the search result to five special domains, Google Scholar, Researchgate, Twitter, Wiki and general search result, the second is modeling, using the processed data to calculate the score between each person by using VSM and Jaccard's coefficient, the third is interface, show the recommendation list and dashboard to

user by website.

3.2 Design

3.2.1 Data process

The system uses Google search to get a top 100 result to get the original data. At start, it used a web spider to get the search result. However, after about 100 times request, Google will return 503 error and refuse the HTTP request because it regards the system as a robot. To solve this problem, the method changed to use the Google custom search API to get the search result.

The Google custom search API is called by html get function. The URL is https://www.googleapis.com/customsearch/v1?. The parameters are: 1) Num: means how many result return, maximize is 10; 2) Key and Cx: the authorize parameters; 3) Q: the query that you want to search; 4) Start: the index of the records that the result starts with. One response just contains maximize 10 records, so one collecting needs 10 requests. The response from the Google custom search API is in JSON format.

After getting search result, the system will process the search result one by one to separate them into five special domains, which are Google Scholar, Researchgate, Twitter, Wiki and general search result. The separating is based on the URL. The search results is not guaranteed as unique, for example, there could be more than one result from Google Scholar in one search results. The system just use the first matched result because Google supposed to return the result in order of relative.

Wiki	{“content”:””, (Other parameters are dynamic, it depends on what does the page contain, but all the things are String, no JSONArray) }
Google Scholar	{“name”:””, “position”:””, “interests”:[ ], “citations”:””, “h-index”:””, “i10-index”:””, “coauthors”:[ ], “titles”:[ ] }
Researchgate	{ “institution”:””, “name”:””, “skills”:[ ], “topics”:[ ], “publications”:[ ], “topcoauthors”:[ ] }
Twitter	{ “timelines”:[ ], “favorites”:[ ], “friends”:[ ] }
Other	[{"index":"", "title":"", "content":"" },.....]

Table 1 , JSON structure of parsed data

To parse the search result to the data that system can calculate, there is a parse class. For five different domains, there are five different parse method. Twitter data is using the Twitter API[6] to get accurate data. The system gets the Twitter name from the URL and by using the Twitter name, the Twitter API will return what the system need. Google Scholar, Researchgate and Wiki are parsed by the web spider which has special designs for their structures. Other search results are parsed by readability API[7] which can get the main text from a web page. The readability API is called by HTML get function. Using the URL of the web page as the parameter to get the main text. The structures of the JSON that stores the parsed data shows in table 1. The online version API just allow the free user to call 1000 times in one day. Finally the system

change to use a local version API to get the main text.

The whole data been processed can be cached in JSON format and store in the hard disk to make sure that if the calculate method change the original data is the same. At first, the system does not cache anything in hard disk. However, when the first time we run the system for a person's recommendation, it took more than 20 minutes to get the result. This long time will not meet the require of users. To provide a better user experiment, we decide to cache all of the data in hard disk. The structure of the storage is that the name of top folder is the conference number, in the folder, there are folders that named by the people who will attend this conference and their affiliation. In the person's folder, there are files that cached the Google search results, parsed data of five different domains and the web page which gets from the URL of each search result.

3.2.2 Model

After getting all the data in JSON format, the system will use the data to calculate the score between each attendee in the conference. The total score consists of content similarity and network similarity, the weights of each similarity are 0.5.

The content similarity is using the VSM, which uses cosine distance between two space vectors that are consist of the words from parsed data. The meta data that this part uses is the data in the table 1 expect the "coauthors" in Google Scholar, the "topcoauthors" in Researchgate and the "friends" in Twitter. Before the term is been put into the vector, it will be normalized, which means change all the characters to lowercase, and remove all the stop words. After processing, the terms will consist a vector that present the information of one person. Calculating the cosine distance could show the potential relation between those two persons.

The network similarity is using Jaccard's coefficient, which uses the number of the intersection of the sets of people that they know dived the number of the union of the sets of people that they know. Not all of the five domains have the network information, just Google Scholar, Twitter and Researchgate have those data, which are "coauthors", "topcoauthors" and "friends". This similarity will show the actual relation between those two persons.

Each domain with two parts will have a sub score, if this domain has the information for this domain, and it is very important to save all of the sub scores because the user can sort the recommendation list by any sub scores or total scores that he wants. The total score of content similarity, the total score of network similarity and the total score will be calculated at final. All the sub scores are linear combined and the weight of different parts is the same. When all of the scores have been calculated, the recommendation list is also created. The recommendation list will be stored in hard disk in JSON format to make the display faster.

3.2.3 Interface

In the system, there is much data that has been cached in hard disk, what and how to display to user is important. The system is using web page to display the recommendation and the information. There are three parts, attendee list, dashboard, recommendation. All of the

three parts can link to the other two parts.

The home page displays the attendee list. The background of the whole web page is hazy, so that users will focus on the function on the web page. At the start of opening the home page, there is an animation that the title which is the name of the system will first come out and there will be two lines easing into the upward and downward side of the title. This animation will draw users' attention to the page and the title. After this animation, other object will come out. There is a selector under the tile for users to select which conference do they attend. Under the selector, there is a hyperlink text tell users to "Get the Authors List". This hyperlink has an underscore with it to let users know that this is clickable. After you click the hyperlink, the page will be scrolled down to second section that contains the list of attendees. At the top of the list, there is a paragraph tells users the name of the conference. Under this paragraph, there is a text to tell users how to use this page. The list has three section, name, affiliations, recommendation. The names are all hyperlinks that link to the dashboard page for this attendee. The recommendations are all hyperlinks that link to the recommendation page for the attendee.

The dashboard page displays the information that the system gets from Google search. There are five sections, which according to the special domain when processing the data, Google scholar, Researchgate, Twitter, Wiki, other search result. The data from Google scholar, Researchgate and Twitter is formatted and easy to display by list, so those three section are displayed by list. The data of Wiki is not that formatted because the wiki is made by users. To make the page neater, this section nest the wiki web page. The data of other search result is too much to display all of it in the web page, so that there are just the titles of each search result and the titles are also hyperlinks to their result website. At the top of the page, there is a paragraph tells users who is the dashboard for. There is also a hyperlink to the recommendation page for this person. All the data in each sections are displayed in a two column list. To make the elements in the list can be distinguished with others, the alpha of the background color will be increased one by one. It can make the border obvious without using a line.

The recommendation page display the information of the recommendation that made by the system. The recommendation list is ordered by the total score between this user and others. However, the list is not only display the final total score. It also display the sub scores of each domains. It can allow users to sort the list by their decision. The list also support search function to find the certain person in the list.

4. Scenario and Evaluation

The first page is the attendee list page. Users can select which conference they are attending (Figure 2). Here we select "UMAP 2015" as an example. After selecting, users can click the "Get the Attendee List" to get the attendee list(Figure 3). In the attendee list, users can click the name in the "The Name" column to go to the dashboard page of this attendee. The page of dashboard contains five domains' information(Figure 4). For the general search result part, all the title that listed can be click and link to the search result page. If some of the domain is absent in Google Search result, the web page will show link figure 5. To go to the recommendation page, users can click the list at the end of each record in attendee list or click the "Get the

recommendation for..." in the dashboard page. In recommendation page, users can click the column name to sort the recommendation list by the column they clicked(Figure 5). Users also can click the name in the "Name" column to go to the dashboard page of this attendee.

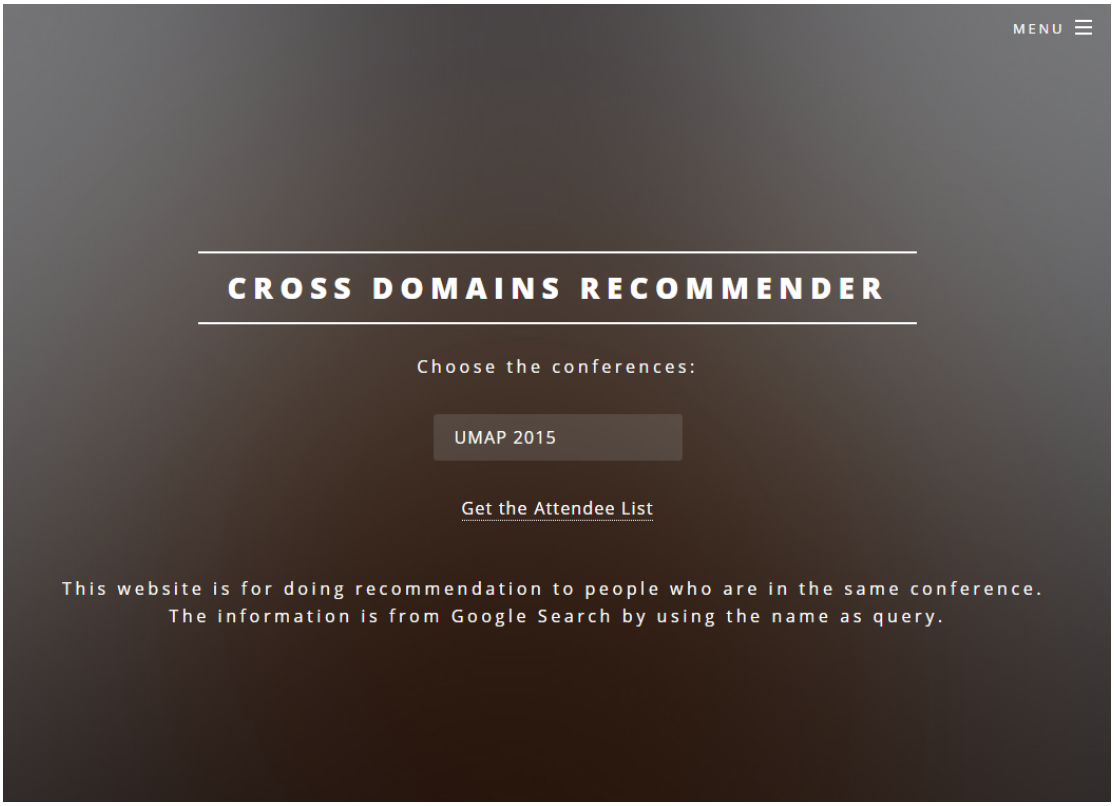

Figure 2: Scenario 1 : the start page of Cross Domain Recommender

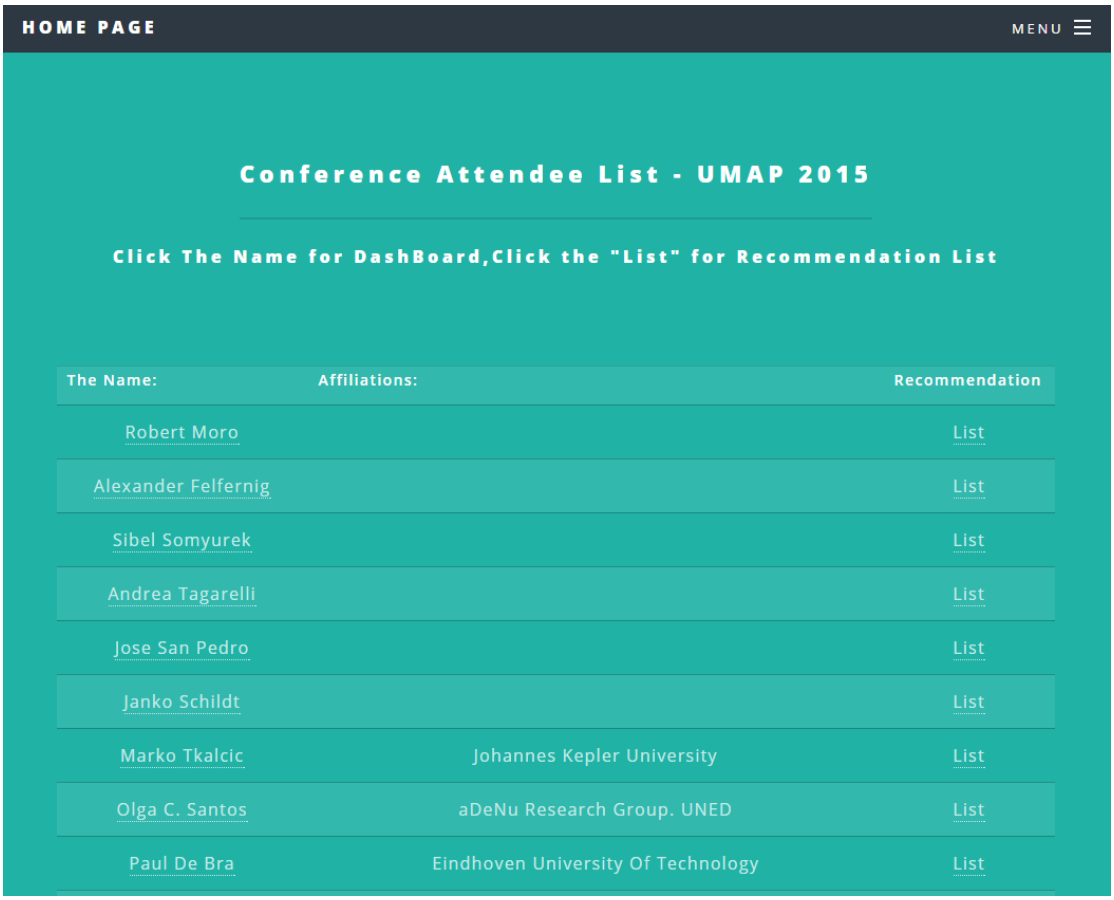

Figure 3: Scenario 2 : The attendee list

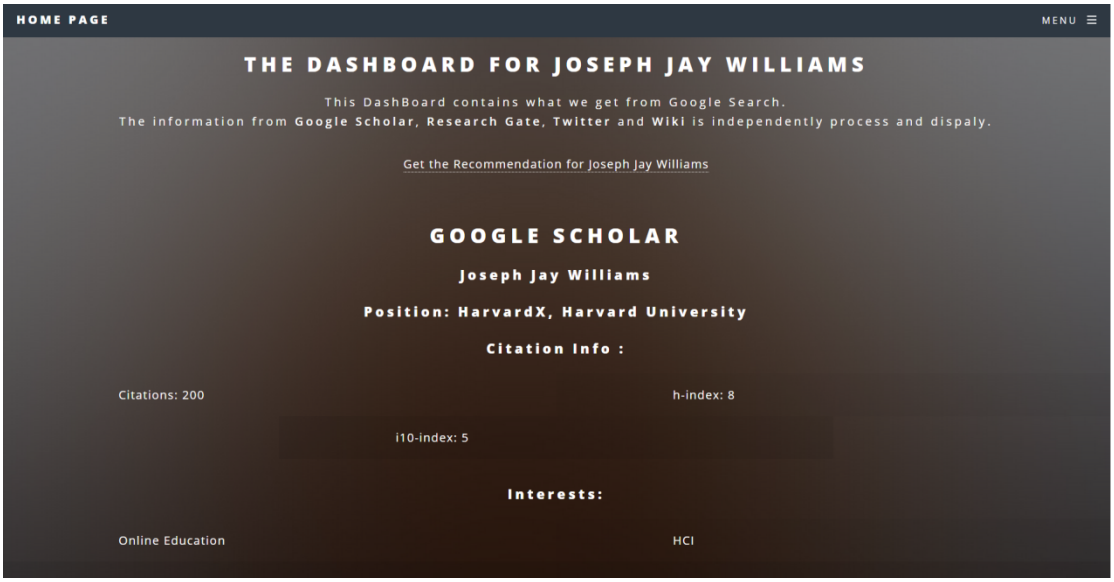

Figure 4: Scenario 3 : The dashboard

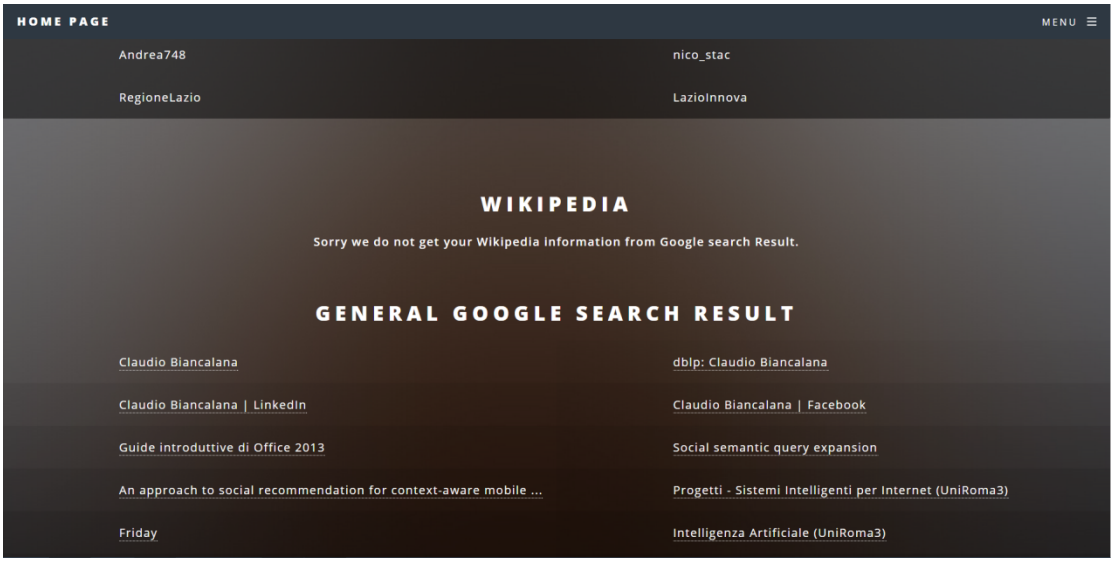

Figure 5: Scenario 4 : Absent information

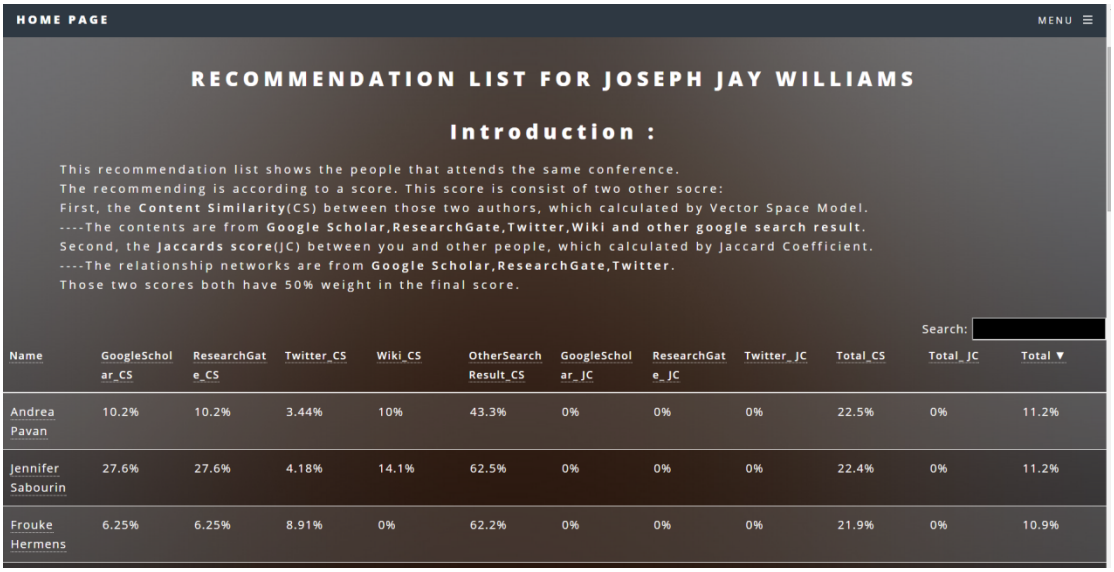

Figure 6: Scenario 5 : Recommendation list

To evaluate the recommendation, we choose one attendee's recommendation and analyzing the attendees that the system recommend to him. It can be seen that in this recommendation list ,which sorted by the total score, "Giuseppe Sansonetti" has the highest total score(Figure 7). The reason that this attendee is recommended to the user is that the scores in Google Scholar and Researchgate are very high. After reading the information in the dashboards, we find that they are coauthors in some paper and also have some common coauthors. However, if the user want to get the recommendation by one specific domain like Twitter, he can sort the list then get the new recommendation. It can be seen that after sorting the attendees with lower total score come to the top of recommendation list(Figure 8). We find that the second attendee who does not have any academic common point with the user has a very high score in Twitter CS score which means that they may have some common interesting or topics. The users who want to find some friend are also able to get a good recommendation.

Name	GoogleScholar_CS	ResearchGate_CS	Twitter_CS	Wiki_CS	OtherSearchResult_CS	GoogleScholar_JC	ResearchGate_JC	Twitter_JC	Total_CS	Total_JC	Total ▼
Giuseppe Sansonetti	17.4%	17.4%	0%	0%	49.8%	0%	42.8%	0%	20.3%	14.2%	17.2%
Christoph Trattner	26.8%	26.8%	27.7%	0%	21.2%	0%	0%	0%	25.5%	0%	12.7%
Geert-Jan Houben	26.2%	26.2%	25.2%	0%	26.8%	0%	0%	0%	24.8%	0%	12.4%
Ivan Cantador	24.4%	24.4%	22.9%	0%	33.6%	0%	0%	0%	22.8%	0%	11.4%
David Hauger	0%	0%	30.4%	0%	28.8%	0%	0%	0%	22.3%	0%	11.1%
Antonino Lo Bue	0%	0%	27.9%	0%	70.5%	0%	0%	5.26%	19.6%	1.75%	10.7%
Dhavalkumar Thakker	23.3%	23.3%	23.6%	0%	32.1%	0%	0%	0%	21.3%	0%	10.6%

Figure 7 : The list that sorted by total score

Name	GoogleScholar_CS	ResearchGate_CS	Twitter_CS ▼	Wiki_CS	OtherSearchResult_CS	GoogleScholar_JC	ResearchGate_JC	Twitter_JC	Total_CS	Total_JC	Total
Victor Codina	15.5%	15.5%	31.1%	0%	12.1%	0%	0%	5.26%	11.7%	1.75%	6.75%
David Hauger	0%	0%	30.4%	0%	28.8%	0%	0%	0%	22.3%	0%	11.1%
Eoin O'Dell	0%	0%	29.8%	0%	25.2%	0%	0%	0%	11%	0%	5.5%
Giovanna Petrone	0%	0%	29.5%	0%	50.5%	0%	0%	0%	16.7%	0%	8.36%

Figure 8 : The list that sorted by Twitter CS score

5. Conclusion

In this project, we provide a system that can do a cross domain recommendation for the attendees in the same conference. We achieve this function by using Google custom search API, Twitter API, Readability API, VSM, Jaccard's coefficient and the spider that writes by ourselves. The recommendation system is displayed in a user friendly interface. It is also useful according to the evaluation we do based on an attendee's recommendation list.

6.Reference

1.PARRA, Denis, et al. Conference Navigator 3: An online social conference support system. In: UMAP Workshops. 2012. p. 1-4.

2. Sahebi, Shaghayegh, and Trevor Walker. "Content-Based Cross-Domain Recommendations Using Segmented Models." CBRecSys 2014 (2014): 57.

3. Bo, Yu, and Qi Luo. "Personalized web information recommendation algorithm based on support vector machine." In Intelligent Pervasive Computing, 2007. IPC. The 2007 International Conference on, pp. 487-490. IEEE, 2007.

4. Liben‐Nowell, David, and Jon Kleinberg. "The link‐prediction problem for social networks." Journal of the American society for information science and technology 58, no. 7 (2007): 1019-1031.

5. HTML5UP, http://html5up.net/

6. Twitter API, https://dev.twitter.com/overview/documentation
7. Readability API, https://www.readability.com/developers/api