Event Recommendations with Suggested Stays in Proximity using Page Rank

*Note: Sub-titles are not captured in Xplore and should not be used

1st Gaddam Rohit Varma
Software Engineering
Arizona State University
Tempe, USA
rgaddam2@asu.edu

2nd Divya Rao P
Software Engineering
Arizona State University
Tempe, USA
dpolinen@asu.edu

3rdPrem Preeti P
Software Engineering
Arizona State University
Tempe, USA
ppatnala@asu.edu

4th Sree Pradeep Kumar Relangi
Software Engineering
Arizona State University
Tempe, USA
srelang1@asu

***Abstract*—Event recommendations popularly have been done using heavy machine learning techniques like matrix factorization. However there have been a huge over head in computation and management of complex models. In this article we suggest a lightweight model using page rank algorithms to present an estimation of rank in importance. Data presented as directed graphs enabling link structures focusing on entities and their relations can be analyzed using Page Ranking algorithms. In addition to making recommendation of events on bases of the rankings computed the presented model makes suggestions on basis of location proximity as well.**

***Index Terms*—Event recommendations, Page Ranking algorithms**

I. Introduction

The exponential increase in usage of mobile devices and world wide web resulted in a social atmosphere responsible for innovative applications. Enormous amounts of data are being dumped into social networking sites such as Flickr, Facebook, Picasa, Twitter etc. It is now more essential than ever to structure this data. Semantic web is an unexplored domain which has lot of potential to make data more organized and more intelligent. Large amount of information about our entity of interest (Events) is readily available to be presented to the user in a structured way. In its complete form, data about a single entity may involve interesting attributes like popularity in friend circles, Location and other features. Several feasible applications can be engineered using these features of data available. Natural Language processing applications such as recommendation systems and question answering models are some of the examples. Intuitively speaking, A factor which could be ranked based on the importance of a chosen feature of our entity could be very potential in implementation a wide range of applications. Though there are various ways for sorting the importance of entities on basis of a certain feature, Page Rank algorithms present promising results in generating such an application.

Identify applicable funding agency here. If none, delete this.

II. Problem Definition

Thanks to the wide spread of social platforms there is substantial amount of reasonable data like global positioning. This data is retrieved from user data and is used to suggest him relevant options ordered in the ranking of importance. This order of importance is retrieved by applying Page Rank algorithm. An added suggestion considering the global positioning for accommodation in the near proximity to the event suggested is made in the suggested model. The http request from the user interface is directed to a back end procedure which has the capability to query the semantically arranged data sets in the database. JSON data is returned. We plan to do this by creating a graph from the user data such that each user's friends are mapped to each other's friends. Page rank formula is used to compute rank of each and every user which helps us prioritise and match events to users.

III. Motivation and Related Work

Pandey, Yogesh, and Srividya K. Bansal. [1] presents the use of semantic technologies to build a web application called Safety Check. They extracted data from various sources for emergency alerts, weather alerts, and contacts data. The extracted data is integrated using a semantic data model and transformed into semantic data. Semantic reasoning was done through rules and queries. They used a high-performance triple store, to store the data, and used a SPARQL compliant engine, to query RDF models. To retrieve instances of weather based on location, they used queries. Similarly, by querying, we can retrieve hotels and event cases in a specific area.

Arif Nurwidyantoro [2] discusses the identification of events and the methods used to identify them from social media streams. To estimate the time and location of an occurrence, a temporal and spatial model has been developed by Sakaki[7] by using tweets as sensor. They also analyze how information diffuse related to real time event. The outcome illustrates that different events have different patterns of diffusion. From this research, an earthquake reporting system as a early warning system is developed.

Searching events mostly are done by filtering by the GPS data, regions, cities, etc. Even though using the manual filter methods amount of data will be overloaded because of the enormous data we have in this data era. The author in [4] has dealt with the same problem while summarizing the entities of linked data. To solve the problem author presented the method of scoring all the entities connected to the target entity and considering the top K entities.We are using the same technique to filter the events recommended to users. The scoring will be applied to the user graph data where the connection is the directed edges between the users and users to events there are interested in.

There are so many algorithms to score the entities in a graph model and these algorithms are prominent for information retrieval. The author in this [5], presented the performance of the page rank algorithm to estimate the importance of graph-structured data[]. The author in [5] performed the page rank algorithm used in [8] and weighted page rank used in [9] on the graph extracted from the links of the various wiki data and presented rank correlations. Based on the results presented in the paper, we are using the page rank algorithm to compute the score for events

Sheba Selvam, Ramadoss Balakrishnan, and Balasundaram Ramakrishnan [3] emphasized the task of social event detection using multimedia content gathered from the daily activity and uploads of users on different social media sites. This involves identifying a picture or any multimedia content as an event. Metadata like a photo ID or geo-location is used to identify each picture. The integral task is to create an ontology for all the events. Protege is the tool used to create the ontology. Reasoning is performed on the created ontology with the help of HermiT reasoner. To retrieve the metadata representing each event we make use of SPARQL queries. Then, we try to gather openly available information online about the extracted events and semantically link them together. SPARQL queries are used again to measure the performance of the extracted events using the semantically linked open data created previously. Additionally, the weather for the retrieved event is also included. This feature helps to reduce false positives in the task to a great extent.

Global Positioning System (GPS) enabled cameras on smartphones are of great help in this task. Events are being classified into two main categories - global and local. Local implies personal events such as birthdays and meetings while global means events that have a worldwide impact. While social events are events that are planned by people, attended by people, and the media illustrating the events are captured by people. An event can be identified using terms of metadata related to the event. Social photo collections can be created using contextual analysis and identifying them as events. This summarizes the process of social event detection.

IV. Approach and High-level System Design

Approach for this recommendation problem is via a page ranking algorithm. User data has 2 properties: hasFriend and hasPopularityScore where popularity score is computed by using the page rank algorithm. For computing popularity score, usersfriends.csv is pre-processed and represented as edges and vertices where the file consists of two rows namely : node 1 of type userid and node 2 of type userid which means userid in column node 1 hasFriend(functional property) of userid in column node 2. The edges will be sorted. These sorted edges will provide fast iterations which lead to faster results as we can read the edges from files instead of using an adjacency list for the same kind of operation. For each iteration, user score will be computed by the page rank formula. After certain number of iterations, each user gets a particular score. The recommendation purely depends on the score calculated by page rank algorithm where for each and every user, we will pick top k friends based on the score and recommend the events to the user where the friends are invited to.

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}$$

Figure 1 :Formula for rank calculation

The Page Rank value for a page u is dependent on the Page Rank values for each page v contained in the set Bu (the set containing all pages linking to page u), divided by the number L(v) of links from page v.

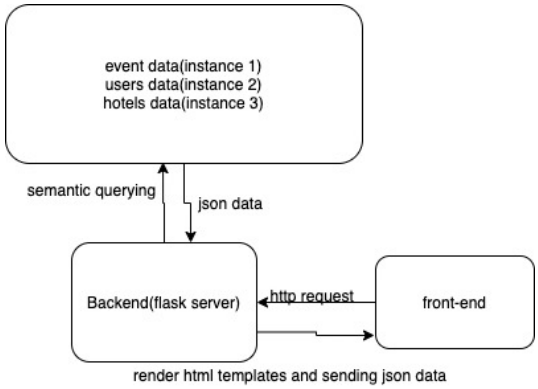

Figure 2 : Overall system design

The entire design includes three components : Database which consists the 3 RDF datasets which are events, users and hotels. We are planning to implement the back end part using python flask which is used to query from the databases for certain http requests which are requested by the user with the help of the front end web application and returns JSON data to front-end.

V. Data collection and Processing

We have collected different types of data from Kaggle. Kaggle is a website which allows users to publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers.

The first one we picked is the U.S. Airbnb Open Data. It is the Airbnb listings and metrics of regions in the U.S. Airbnb is an online marketplace which lets people rent out their properties or spare rooms to guests. The dataset has

seventeen columns of values namely : id, name, hostid, hostname, neighbourhoodgroup, neighbourhood, latitude, longitude, roomtype, price, minimumnights, numberofreviews, lastreview, reviewspermonth, calculatedhostlistingscount, availability365,city. We are only using a few columns which we think are most needed for successful completion of the web application we are planning to create. The columns or attributes which we are using are the id, latitude and longitude.

The second one from which we picked data is called the Event Recommendation Engine Challenge.It has data which can predict what events our users will be interested in based on user actions, event metadata, and demographic information. There are six files in all: train.csv, test.csv, users.csv, userfriends.csv, events.csv, and eventattendees.csv. We have used users.csv, userfriends.csv, events.csv and eventattendees.csv.

users.csv contains demographic data about our some of our users (including all of the users appearing in the train and test files), and it has the following columns: userid, locale, birthyear, gender, joinedAt, location, and timezone. We are using userid which is the id of the user in our system.

userfriends.csv contains social data about this user, and contains two columns: user and friends. user is the user's id in our system, and friends is a space-delimited list of the user's friends' ids.

events.csv contains data about events in our system, and has 110 columns. Some of the useful columns are namely eventid, userid, starttime, city, state, zip, country, lat, and lng. We are making use of eventid which is the id of the event.City, state, and country represent more details about the location of the venue (if known). lat and lng are floats representing the latitude and longitude coordinates of the venue, rounded to three decimal places.

eventattendees.csv contains information about which users attended various events, and has the following columns: eventid, yes, maybe, invited, and no. eventid identifies the event. yes, maybe, invited, and no are space-delimited lists of user id's representing users who indicated that they were going, maybe going, invited to, or not going to the event.

VI. Ontology Description

We designed three ontologies for this project, which are users, events and hotels. Events ontology has the following classes: event, location, city, state and country. It has object properties atCity, atCountry, atLocation, atState, locationisIn. It has data properties hasEventID, hasLatitude, hasLongitude, hasLocationName, hasStartTime and isCategoryOf.

User ontology has only one class user .It has symmetric object property hasfriend which has user as domain and range. It has data properties hasID which has a user as domain and string as range. Other data property invitedTo which has a user as domain and string as range.Last data property hasPopularityScore which has a user as domain and string as range.

Hotel ontology has classes hotel and location. It has object property, haslocation which has a hotel as domain and location has range. It has a data property hasID which has the user as domain and string as range. Second data property hasLatitude which has location as domain and float as range. Second data property hasLatitude which has location as domain and float as range.

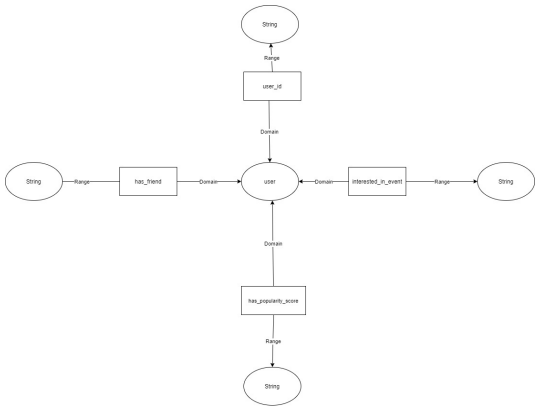

Figure 3a :Ontology Visualization for users

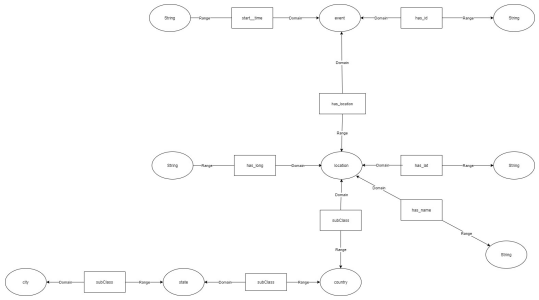

Figure 3b :Ontology Visualization for events

VII. Implementation plan

Implementation contains of 4 steps : Ontology Engineering, User's popularity score generator,Semantic querying and web application implementation. Ontology design and engineering is can be considered completed . Our future plan is that, the first teammate will work on the ranking algorithm for users using a page ranking algorithm to generate score for each user. The second teammate will be working on semantic querying of the data to extract desired data. The remaining two teammates will be working on routes and front-end templates of web application to create a user-friendly application.

VIII. Tasks to be completed

1) Creating a graph with user's friends data 2) Computing ranks for users from the graph using page rank algorithm. 3) Creating three instances of the three RDF datasets which we picked, in the remote servers. 4) Writing semantic queries for events of interest and hotels for users and friends of users. Query the nearest hotel for particular events. 5) Writing routes for different URL paths and processing the response from semantic queries. 6) Writing front-end templates for the web application.

IX. Implementation

The main objective of this project is to recommend various events to each user that he/she might be interested in going to.

This is achieved by using various ranking algorithms and one such ranking algorithm that we used in our implementation is Page Rank algorithm.

The recommendation system uses the user friend's graph data and user interested events graph data to figure out the scores for each individual user node and also for each event node. The scores for each user and events nodes are derived using the page rank algorithm. For the user data, the score for a user node is derived based on its own importance and its relations to other user nodes. The user nodes are connected to other user nodes via edges named "has friend" indicating that the 2 nodes have their own importance on one another. Each connecting edge has both outward and inward arrows that help in figuring out the importance of one node over the other. The inward arrow on an edge arriving from a different node to the current node indicates the importance of the current node. Whereas the outward arrow on a particular edge extending from the current node to a different node indicates the importance of the connected node in correspondence to the other nodes it is connected to. The importance score of an edge connected to a different node is measured as

(importance score of the node)/(total no.of nodes connected to it)

The self importance score of a particular node is calculated as a sum of all the scores of the edges connected to that node from other nodes. The high importance score of a particular user node will also affect the importance score of other user nodes connected to it. Initially all the user nodes receive an importance score of "1" and after multiple iterations through the graph node the importance score of each node converges to a value that does not change by a huge value. Using the above method the importance score of each user node is calculated.

Below is an example image of 3 nodes and their edge connections that will help in easily understanding how the score for each edge is calculated and how the importance of each node influences the other nodes.

For the event recommendation system, we select a user node from the user data with computed importance scores. We then figure out all the relations (friend) this selected user has with other user nodes from the user data and extract the computed importance score for every related node. We then sort the extracted information based on the importance score of each related user node. From the sorted data we select the user id's of top 5 users based on their importance score. The reason for selecting only top 5 users is later discussed in the challenge section of the document. Generally, a user might be interested in many events and since we cannot recommend all the events that the selected 5 friends are interested in. We derived a formula that can be used to figure out the event score that would help in determining which event is to be recommended for the selected user. The derived formula is an inspiration of the page rank algorithm.

In our implementation of the formula, the events are categorized into 3 categories namely - "Interested", "Maybe", "Invited" and each of the categories have their own individual weights that would be used in deriving the event score. The weights for each category are as follows and are not experimentally obtained but rather selected by us for the recommendation system, for "Invited" - we assigned a value of 5, for "Maybe" - we assigned a value of 4 and for "Interested" - we assigned a value of 9. Using the Interested events graph data we can figure out the events that the selected user might be linked with and find the importance of the event in the user perception by using the formula

[(user score) * (Weight of the Event category)] / (total no. of linked events)]

Weight of the event category helps us in determining the importance of the event. We can then rule out the events that the user is invited to or is maybe interested in attending it. This way the events that a user is interested in going to have high importance. Each event can be linked to multiple users under multiple categories. The importance of the event with respect to other users is also calculated using the above formula. The event score for that particular event is then calculated by summing all the individual event importance values with respect to the linked users. We now extract all the event information that the selected top 5 friends of a user are linked to. We then calculate the event score for every event that the selected 5 friends are linked to and then select the top 10 events based on the event scores.

USE CASES FOR TESTING

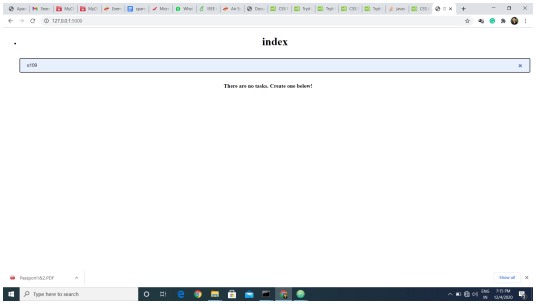

Figure 4a :Searching for a user

The search bar will predict and is intuitive for searching for a user and the results are displayed in the following figure. As the user starts typing the results are auto populated according to the entered string.If the query is successful we can see the results as in figure 4b.

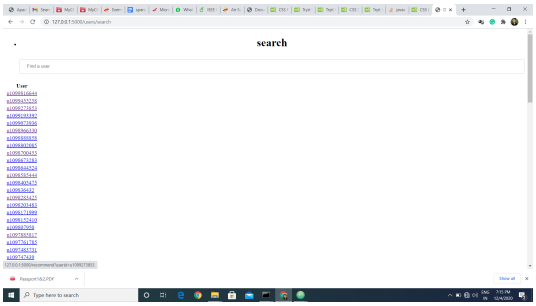

Figure 4b :The list of Users

The hrefs to all the related users are displayed for us to choose.

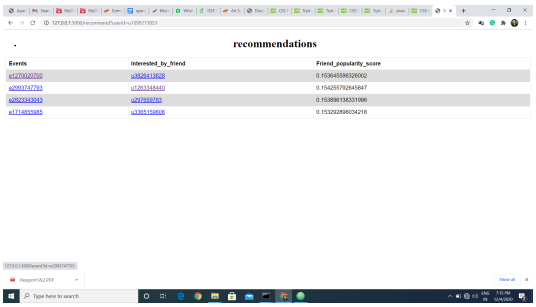

Figure 4c :Screen when any user from the list is selected

We can see the event recommendations for the selected user along with which friend of his is interested and the data is sorted in descending order which means the friend with the most popularity score is listed first and the event that he is attending is given the higher priority.

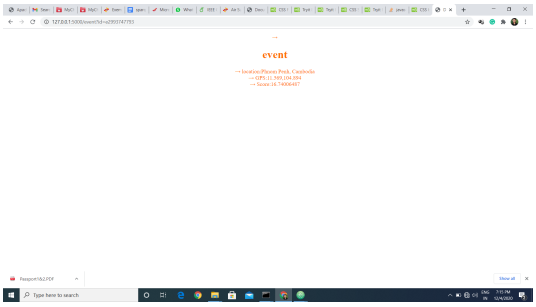

Figure 4d :event details

When an event on the page displayed in figure 4c is clicked we land on the page where event details are displayed. The location and gps of the event and the score given to the event are displayed in this page.

More details on how these features are integrated and implemented together :

Data Mapping and Integration:

As a part of data processing we already removed all the values in latitude and longitude which are null. From users we have used the entire data and then used the page ranking algorithm on each user as follows - Each user is connected to any other user in the data set if they have a friend. An edge between two users represent that they both are friends. We initialised the popularity score for users with no edges or no friends to be zero. The popularity score of users who have incoming edges will be the sum of all incoming edges. The formula in figure 1 shows us the formula to calculate the popularity score.

Every user has an initial popularity score and certain number of friends. So the contribution that user does to her/his friend's popularity score is (her/his popularity score) / number of friends. The calculation is done and each user's popularity score is updated through fifty cycles. In the initial cycles there will be a difference in the popularity scores after each cycle.But after certain number of cycles it reaches a threshold and the values become consistent and will not get affected anymore after the cycle completes. The consistent scores of each user is the final popularity score of that particular user.

We used Celfie plugin which is a desktop plugin for importing data we had pre-processed and stored in spreadsheet into OWL ontologies in Protege to generate instances.

Cloud Servers: Cloud servers have become a popular means to deploy Linked data. Amazon web services are utilized in this application to operate on Fuseki server to create two instances of EC2. One of the instances was used to process user data and the other was utilized to process event data.

Construction of Application: The application abides by the REST architecture. A clear and intuitive to and fro communication between the server and the client have been established.

Backend: We query the information residing in the Fuseki servers with the help of SPARQL queries. The results will be in a json format and the information we gather here is used as the input to our application in the front end.

Frontend: We used Flask and Jinja templates to build the front-end of our application. We presented a user-friendly GUI in which we built a user search on the basis of the parts of their name. The search is assisted by an auto suggestive algorithm making the usability of the model smooth.There are three main screens. The first screen to focus on is a screen that displays the users.

On selecting the intended user we render all the events in ranked order on the next screen . An event chosen by the user will direct him/her to a detailed description of the event including measures like the event score and event location.Event recommendations for that particular user are based on the events her/his friends are attending, though not all the friends but only five friends who have more popularity score than the others in his list of friends.

X. SPARQL Querying

We used SPARQL queries for these major parts - 1. for searching users 2.for searching recommended events 3.for getting event details

While performing the user search operation, a FILTER is applied on the names in the date received. This data is then ORDER BY the ranking of the entry. We also made use of regex to accomplish that.

XI. Evaluating our Application

The running time of the application is mostly comprised of the score calculation. The popularity scores need to be calculated for the users as well as the events.Apart from that there is searching and populating the users list and generating the details of an event when an event is recommended to any user.

XII. Challenges Faced

The data set to discover if two users are friends was very huge. There were 13 million edges to go through and check which all users are friends of which all users. So we cut down a major part of it by considering only the top five friends of a particular user. Calculating the popularity score for events and figuring out the formula to do so was a challenge.

XIII. Future Scope

This recommendation feature can be further extended and can be made to provide hotel details in and around the event which we are attending. The location and GPS of the event are already known to us so we have to make use of that and be able to gather the information about hotels nearby and suggest them to all the users attending the event. The popularity score is calculated based on only these three options "Interested", "Maybe" and "Invited". There is need for the option of "Not Interested" also for providing more accurate recommendations.

XIV. Conclusion

We have learned a lot about how the ontologies that are made in the real world and how to make use of the semantic web technologies for the data in the real time. We also gained knowledge about development of the web application and fuseki server. It gave us an overall perspective how to create an end to end software web application.

References

[1] Pandey, Yogesh, and Srividya K. Bansal. "A Semantic Safety Check System for Emergency Management." Open Journal of Semantic Web (OJSW) 4.1 (2017): 35-50.

[2] A. Nurwidyantoro and E. Winarko, "Event detection in social media: A survey,"International Conference on ICT for Smart Society, Jakarta, 2013, pp. 1-5, doi: 10.1109/ICTSS.2013.6588106.

[3] Sheba Selvam, Ramadoss Balakrishnan, and Balasundaram Ramakrishnan, "Social Event Detection–A Systematic Approach using Ontology and Linked Open Data with Significance to Semantic Links,"The International Arab Journal of Information Technology, Vol. 15, No. 4, July 2018.

[4] Thalhammer, A., Lasierra, N., Rettinger, A.: LinkSUM: using link analysis to summarize entity data. In: Bozzon, A., Cudré-Mauroux, P., Pautasso, C. (eds.) ICWE 2016. LNCS, vol. 9671, pp. 244–261. Springer, Heidelberg (2016). doi: 10.1007/978-3-319-38791-814

[5] Thalhammer A., Rettinger A. (2016) PageRank on Wikipedia: Towards General Importance Scores for Entities. In: Sack H., Rizzo G., Steinmetz N., Mladenić D., Auer S., Lange C. (eds) The Semantic Web. ESWC 2016. Lecture Notes in Computer Science, vol 9989. Springer, Cham. https://doi-org.ezproxy1.lib.asu.edu/10.1007/978-3-319-47602-541

[6] Andreas Thalhammer. "https://github.com/athalhammer/danker"

[7] T. Sakaki, M. Okazaki and Y. Matsuo, "Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors," in Proceedings of the 19th International Conference on World Wide Web, 2010.

[8]Brin, S., Page, L.: The Anatomy of a large-scale hypertextual web search engine. In: Proceedings of the Seventh International Conference on World Wide Web 7, pp. 107–117. Elsevier Science Publishers B. V, Amsterdam (1998).

[9]Baeza-Yates, R., Davis E.: Web page ranking using link attributes. In: Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers and Amp; Posters, WWW Alt. 2004, pp. 328–329. ACM, New York (2004)

[10] Allan J., Papka R., and Lavrenko V., "On-line New Event Detection and Tracking," in Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, pp. 37-45, 1998.

[11]https://www.google.com/search?q=what+is+kaggleandoq=what+is+kaggleandaqs=chrome..69i57j0l7.3132j1j7 andsourceid=chromeandie=UTF-8

[12]https://www.kaggle.com/kritikseth/us-airbnb-open-data

[13]https://www.kaggle.com/c/event-recommendation-engine-challenge

[14]https://home.bt.com/lifestyle/travel/travel-advice/what-is-airbnb-11363981595930