University of Michigan Campus Event Search Engine

Muhan Yuan, Ruihan Wang, Xiaojie Liu

Introduction

As one of the leading academic institutions in the world, University of Michigan attracts students and faculties from all over the world. Its rich academic and extracurricular activities makes the campus a great place for students to broaden their horizons and strengthen their skills. As a large public university, University of Michigan currently has a 20,965 acres campus, enrolling a total of 44,718 students with 6,771 academic staff and 18,986 administrative staff[1] as of 2016. With such a large body of students and faculty, there are hundreds of campus events going on every day, which makes it extremely difficult for people to keep track of specific event. "Happening @ Michigan", a website maintained by Campus Information and Technology Service, provides a naive event search calendar, which allows user to search campus events. However, this simple search mechanism does not provide sufficient search results as desired, nor does it support complicated query search. We want to build an advanced search engine that supports complicated query search that better fits user needs. This search engine enables user to search for Umich events based on key phrases, location, time, etc.

Compared to an ideal search engine, the current search engine cannot satisfy many search requests by users, nor it is able to provide accurate or relevant search results. Meanwhile, the current search engine does not support advanced query search or synonyms search. To better improve user experience and to provide more accurate search result, we believe it is crucial to build such search engine that allow students and faculties on campus better locate events with details.

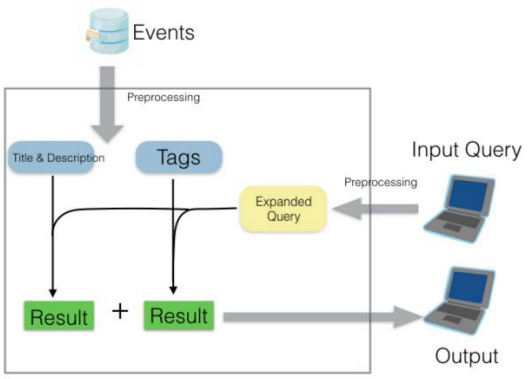
It is important to build such search engine since it will be a convenient tool for users to search events given the fact there are such a large quantity of campus events going on every day. Our target users are students and faculty of Umich who run on busy schedules, it will be a handy tool for them to keep track of campus events going on historically or in the future. Providing an advanced event search engine will enable users to find relevant events as long as they have a search query, no matter it's a related title, or a tag of the event.

This event search engine we built utilized basic information retrieval techniques since this search engine requires to process both search query as well as original events data. It is an information retrieval problem since it involves obtaining information relevant to an information need from a collection of massive information resources, that is to say, to retrieval a series of events relevant to search query from all the events data. To solve this problem, it requires scientific data pre-processing and text analysing and retrieving, as well as search query processing. These activities and processes all require advanced information retrieval techniques.

Method

The basic idea of building a event search engine is ranking the relevancy of the content of each event to the query. To achieve this, figure out a method to extract the content of events from the information we have and give a clear definition to the relevancy.

[1] "UniversityofMichigan—EnrollmentOverview", retrieved October 28, 2016



Although real-time database should be used for event searching, we chose to use a static dataset given the data collection efficiency. To building a dataset containing information need for events search, we collected 9713 events from Happening @ Michigan Event Feed in JSON format, including the following information: event ID, tags, description, locations, etc. The data will be separated into two parts (past and future) every time it is loaded according to the current time and the user would be able to choose from full dataset or part of it.

At this stage, the information we used for event searching includes the events' tags, titles and description. In most cases, since the tag is a single word or a phrase containing less than 3 common words, no special preprocessing is needed except splitting the phrases and turning them into lowercase.

For title and description, the content is more complex. First step is stripping punctuation in descriptions and converting them to lowercase. Then, titles and descriptions may use different form of one word, which could potentially hinder the match queries and document. So, using "textblob" package, we performed stemming and lemmatization to find the base form of a word.

After inputting a search query, python "autocorrect" package is used to check if there are any spelling error. The user will be asked to choose between the raw query and the auto correction result if change has been made. To match the query with relevant events that might not have the exact word in query, we tried to expand the query by finding the synonyms of words in query using "wordnet" in "nltk.corpus" package and form a new query that contains all relevant words. We assume the words in the original query should more precisely represent the user's meaning. Thus, words in the original query will repeat N times and N equals to 1/4 of the total amount of potential synonyms in order to manually put more weight on the those words. In order to match the query to stemmed titles and descriptions, we performed stemming and lemmatization to new query as well.

We assume the tag information most accurately reflects contents of the event, then the title, and finally the description. For events that contains several tags, the rarer tags are considered to be more important for the events. So, we first match the tags of events with stemmed words in expanded search query. If the tags of events can match with search query, the events will be considered relevant.

Tag> Title> Description

Events have rarer tags match with query words will get higher rank in the result.

Then, we merged the title and description of one event together as an overall event description. In order to put more emphasis on the title, we manually repeated the title 3 times in the new description. When vectorizing the feature, we tried different n-grams range and found (1,2) works best for events, since phrases like "human right" cannot be properly detected if only single words are taken into consideration. Also, in order to put different weight to features in the description, we used the tf-idf transform. Similarly, query is processed with the same vectorizer.

$$tf - idf = tf(t,d) * idf(t)$$
$$idf(t) = log(\frac{1+nd}{1+df(d,t)}) + 1$$

The last step is ranking the cosine similarity between query vector and other event vectors. Events at the top of the list will be added to the result if not in it, the top 20 of all results will be presented to the user.

Results

Our event search engine delivers optimistic results based on common evaluation criteria. In the test cases we run, our event search engine shows better results compared to the existing one using the same search query. Taking the search query "big data" for instance, 6 out of 10 returned results for this search query from our event search engine are big data related events, however, only 2 out of 10 returned results from existing search engine are big data related events. For this specific search query, our event search engine performs much better that the existing one. Meanwhile, since our event search engine has the feature of processing typed in search query, such as auto-correcting typo, if user uses a search query that contains typo, our event search engine is able to deliver desired search results, which existing one is not able to do so. For detailed comparison, see comparison later.

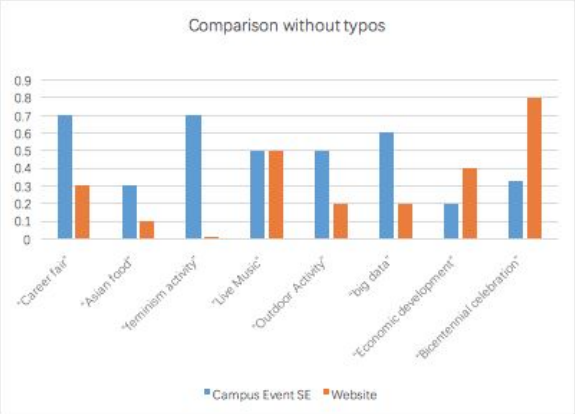
To have a better overview of the performance of our search engine, we chose 8 queries to evaluate the search results of our search engine as well as the website. The results are evaluated by the precision which is defined by the following formula based on the first 10 responses:

$$\text{precision} = \frac{number\ of\ relevant\ results}{number\ of\ total\ results}$$

Here are the results of selected common queries we are using to test the performance of our search engine and the website.

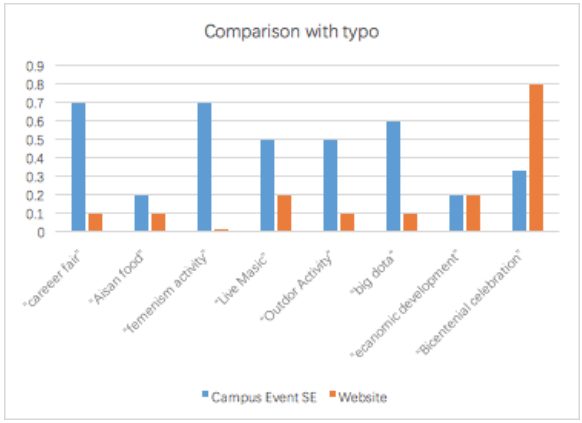
Queries	Precisions	
	Campus Event SE	Website
"Career fair"	0.7	0.3
"Asian food"	0.3	0.1
"feminism activity"	0.7	0
"Live Music"	0.5	0.5
"Outdoor Activity"	0.5	0.2
"big data"	0.6	0.2

“Economic development”	0.2	0.4
“Bicentennial celebration”	0.33	0.8



With Typo:

Queries	Precisions	
	Campus Event SE	Website
“careeer fair”	0.7	0.1
“Aisan food”	0.2	0.1
“femenism activity”	0.7	0
“Live Masic”	0.5	0.2
“Outdor Activity”	0.5	0.1
“big dota”	0.6	0.1
“ecanomic development”	0.2	0.2
“Bicentenial celebration”	0.33	0.8



From the table and charts given above, noticed that our search engine usually had a better precision over the website, especially when searching queries with small typos or direct matched tags.

We found there are several shortcomings of our results as we can see. First of all, as a typical problem for all search engines, the search result accuracy can be further improved. It is difficult to quantify the accuracy of the search results as we do not have access to ground truth for each search. From the test cases above, we noticed there are two cases where the result is less satisfactory: one is the existence of an important adjective. Another is some errors happen in stemming process. In the first case, the result will be dominated by the less important noun, and in the second cases, the vectorizer cannot even detect the keyword. Secondly, the ranking algorithm we adopted in our search engine is based on several assumptions, which can be modified if assumptions are changed. We can argue this ranking algorithm might not be the best ranking algorithm for a search engine, since we cannot perform user search log analysis. And it is challenging to optimize ranking algorithm and maximize user satisfaction even with search log data. Thirdly, our event search engine does not support advanced customized search. In an ideal event search engine, it would be helpful if it

supports customized search such as search event within a given period of time, or type, or location. Lastly, given the fact our current event search engine only runs on event data for one year (approximately 10,000 data entries), the search speed will get slower if the database we use increases in estimation. This shortcoming might not be too critical at current stage, however, as database increases, the performance of the search engine is extremely important for user experience.

Discussion

Overall, our Campus Event Search Engine takes a lot improvement over the current search website of the relevance between the results and queries. Performing tag recognition before query calculation, also, expanding the query set by adding synonyms and adding auto-correct function to handle unexpected typo in the query helped us enhance the relevance of the results effectively. Our final deliverable is a campus event search website which allows users to type queries and select parameters like time period from the user interface and get the results after data processing. All events in the result can redirect users to the webpage of each specific event if user wants to know more. The current event search website is unable to handle user typos and some terms with multiple expressions.

In the future, we will try to enhance the query expansion process. For example, do some filtering over query's synonyms set. The reason is that a word usually owns multiple meanings and parts of speech. Not every explanation makes positive contribution to the relevance of the results. Also, find a dynamic mechanism to determine the weight of the term in each query. In the current version, in order to avoid the negative effect of irrelevant words in the synonyms expansion, we set a fixed weights of the original term manually. Obviously, this is not the best choice, so we have to find a method to set the weights of the term wisely. Plus, we need a larger, dynamic, and more completed dataset. All the current work is based on a static dataset which retrieved from UM Event API of a specific period. If we combine our system with a live data source, the performance of our search engine could be better.

Reference

[1] Chum, Ondrej, James Philbin, and Andrew Zisserman. "Near Duplicate Image Detection: min-Hash and tf-idf Weighting." BMVC. Vol. 810. 2008.
[2] Ramos, Juan. "Using tf-idf to determine word relevance in document queries." Proceedings of the first instructional conference on machine learning. 2003.
[3] Paice, Chris D. "An evaluation method for stemming algorithms." Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval. Springer-Verlag New York, Inc., 1994.
[4] Xu, Jinxi, and W. Bruce Croft. "Corpus-based stemming using cooccurrence of word variants." ACM Transactions on Information Systems (TOIS) 16.1 (1998): 61-81.
[5] Diaz, Fernando. Autocorrelation and regularization of query-based information retrieval scores. University of Massachusetts Amherst, 2008.
[6] Hersh, William, Susan Price, and Larry Donohoe. "Assessing thesaurus-based query expansion using the UMLS Metathesaurus." Proceedings of the AMIA Symposium. American Medical Informatics Association, 2000.