Course Recommendation Based on Semantic Similarity Analysis

Hualong Ma, Xiande Wang, Jianfeng Hou
Graduate Management Brigade
Defense Information Academy
Wuhan, China
e-mail: mahualong.cool@mail.com, 214159976@qq.com, 214159976@qq.com

Yunjun Lu
The fourth department
Defense Information Academy
Wuhan, China
e-mail: 33186792@qq.com

***Abstract*—For many students, course recommendation is a troublesome problem. To choose satisfying courses when they only have a little of information about the courses, they have to turn to common course schedule systems for help, but the result is disappointing. To solve the problem, the method of scoring similarity of courses and clustering courses based on course descriptions is presented. Different from the method, this paper applies semantic similarity analysis into course selection, realizes a course recommendation system. Each course description is first modeled as a document, and a clean-tokenize-TFIDF-Word2Vec-Doc2Vec pipeline is built to create vectors for each course from which cosine similarities will be calculated. The result is even better than the above method through evaluation.**

Keywords-course recommendation; semantic similarity; evaluation

I. INTRODUCTION

The typical approach for course schedule system is to search keywords in the course titles or pre-defined categories just like common course schedule systems. However, this method has its own limitations. Imagine a scenario: You plan to learn something about data science and you find a course, "Data mining". However, according to its description, you realize the course's targets, focus or other contents are not totally the same as you planned. You are wondering whether there are other courses similar to "Data mining" that might be more suitable for you. The question is, how can you find such courses when you only have information of the course "Data mining"?

To solve this problem, in 2009, University of California, Berkeley ISchool students Kentaro Suzuki and Hyunwoo Park, implented UC Berkeley course recommendation system by scoring similarity of courses and clustering courses based on course descriptions. Their method used EM algorithm to cluster courses among different departments and recommend courses based on clustered categories [1]-[3].

II. PROJECT GOALS

A. Original Goals

We plan to address the same goal as Kentaro Suzuki and Hyunwoo Park did. But, our project differs from their's in the way that we plan to apply semantic similarity analysis into course selection which would better serve students to find similar courses on semantic level. It does not simply help students to search courses from course titles and description, but recommends similar courses on topic level analyzed from these information. In our recommendation system, students could use a full course name to find similar courses, based on which they can compare and choose a more suitable one.

To evaluate and improve our project, we plan to deploy a trial version of our system to Information School Students to test our recommendation results. The participants could submit their feedback, including scores and reviews, about the course similarity and the course's content. The course similarity, furthermore, would be updated and adjusted according to these feedbacks.

Init workflow of our recommendation system consists of five modules, collection preprocessing, tokenizing, tf-idf calculation, Word2Vec modeling, and Doc2Vec calculation. Each course description is first modeled as a document. We plan to build a clean-tokenize-TFIDF-Word2Vec-Doc2Vec pipeline to create vectors for each course from which cosine similarities will be calculated.

- Preprocessing. Remove punctuation, symbols, and stopwords. Perform stemming. Find frequent word pairs to treat as bi-grams.
- Tokenizing. Convert the cleaned strings of words into a list of separate n-grams (1 and 2-grams, in this case)
- TF-IDF. Compile a vocabulary of all tokens occurring more than once in the corpus which are used to create a bag-of-words representation for each course. This is then used to build a TF-IDF weighted Document-Token feature matrix.
- Word2Vec Modeling. Train a Word2Vec skip-gram model using all sentences which occur throughout the entire course description document corpus. Word2Vec will yield a dense multi-dimensional vector for each word in the vocabulary. These word vectors will be treated as encoding the semantic 'meaning' of each word.
- Doc2Vec Matrix. Calculate course vectors as the weighted average of the word vectors for all vocabulary words appearing in a given document. The individual word vector weightings are based on the scaled TF-IDF weights for each word in a document.

978-1-5386-0484-7/17/$31.00 ©2017 IEEE

After the above workflow, similarity value will be calculated to determine similarity among courses. We also plan to develop an online evaluation system to gain feedback of our course recommendations. The feedback will contain two parts, similarity scores and course reviews. The similarity scores will be weighted to adjust the courses similarity database, which will then affect rankings of recommended courses. On the other hand, when reviews feedback reaches a certain amount, we will make it as part of dataset to recalculate course similarities [4].

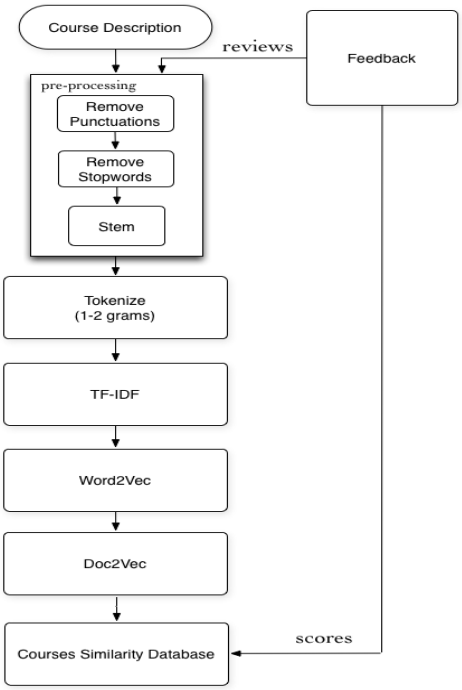

Figure 1. Workflow of our recommendation system.

Challenges:

There are more than 90 thousand courses existed in our university course schedule system, which will be too large for computing and modeling on a personal computer. So we plan to focus on courses from certain departments such as EECS, ISchool, Hass, and Mathematics, from which Information School students would probably choose courses.

B. Achievements

Based on our plan, we implement most of our original goals, including,

- Used LSI algorithm to analyze semantic similarity among 537 courses from seven departments and build similarity index file;
- Used D2V algorithm to analyze semantic similarity and build similarity index file;
- Used D2V+LSI algorithms to analyze semantic similarity and build similarity index file;
- Did A/B testing to compare the above three algorithms, and chose D2V+LSI as final algorithm;
- Tried Kmeans clustering method to do recommendation based on D2V+LSI matrix;
- Developed Web UI to help user find similar courses with our algorithm;

Deployed our Web application to ISchool students to gain feedback of recommendation;

C. Results/Evaluation

We deployed our web application to more than 20 Information School students, and searched more than 40 courses in the seven departments. Most of students are satisfied with our recommendation courses. Our algorithm succeeded to find similar courses among the 537 courses we analyzed. However, the ranking of our recommended courses should be adjusted according to the students feedback. For example, when student searched "Cyberlaw", the course "Supply Chain Management" had higher similarity than "Information Law and Policy", while in fact "Information Law and Policy" should be listed above "Supply Chain Management".

D. Future Work

- Enlarge the library of the courses:

Now, due to the consideration of computation time and capacity of computers, we only use courses from seven departments and 537 courses in all. In future work, we could try to add more courses from other departments or even from other universities to see how our project works.

- Add more content to the description of courses:

According to the feedback of users, we intend to add more useful information, including syllabus and reading articles, to the description of courses which now only contains course titles and course descriptions.

- Add user's feedback function:

During the project, we also realize that some descriptions are out of date or are not fully in the accordance to its content. Thus we intend to add feedback function to our project. The feedback contains two parts. The first one is user's reviews. These reviews, after filter, would be processed and added to the description of the courses for further re-calculation of the similarity. Another feedback is the score which users mark the recommendation courses. The score may be combined to the similarity number to impact the sequence of the courses.

- Improve our algorithm and attempt new algorithms

Due to time constraints, we were not able to try every our thoughts in this project. Thus in the future, we could try some new algorithms or new features to improve our current algorithm to yield a better result than we did now.

III. DESCRIPTION OF DATA

Our dataset is course description data that are crawled from UC Berkeley course schedule website (http://guide.berkeley.edu/courses/). We crawled all 9,205 courses and stored the data into mysql database for analyzing similarity and retrieving. The data that describes a course include category, code, title, unit, term, description, and other details such as professor and grade options.

The actual data we use to analyze similarity is course title and description since those two data is more informative when analyzing semantic similarity. In our demo,, due to time efficiency and hardware capacity, we used courses from

7 departments: CS, EECS, MBA, INFO, DATA SCIENCE, STATISTICS, MATHS.

IV. DESCRIPTION OF ALGORITHMS

What algorithm do we use in this project?

In this project, we tried several algorithms including, tf-idf, word2vec, Latent semantic analysis (LSI) and Doc2vec (D2V). In these four algorithms, tf-idf and word2vec are used to pre-possess the corpus while LSI, D2V are used to generate the course similarity.

As shown in Figure 2, below is the algorithm flow in this project:

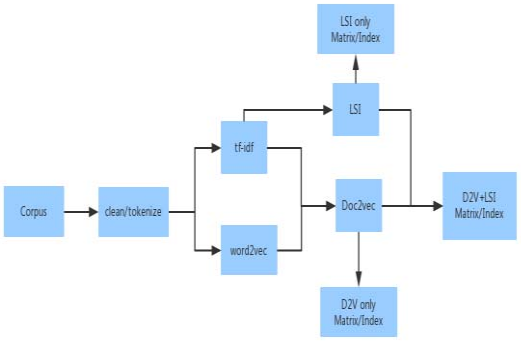

Figure 2. The algorithm flow in this project.

Why do we choose LSI and Doc2vec for similarity?

Our project goal is to present the course recommendation according to course's similarity, which based on the relationship between article and article (course description + titles). LSI could present the relationship between word and article while Doc2vec could calculate the direct relationship between article and article. Thus we chose these 2 algorithms and their combination to generate 3 similarities at first. After further comparison, we chose D2V+LSI as our final one.

How are these algorithms implemented?

In these four algorithms, we utilize Gensim to build tf-idf, word2vec and LSI models. For Doc2vec, although Gensim also provides a model, we found it is too complicated to use and not fit for our purpose, we built a simple function to implement our own Doc2vec.

How do we build Doc2vec?

Our goal is to build a vector matrix that is similar to word2vec but is based on one article not one word. After searching similar documentations/projects and referring to KenDooley's project, Better Restaurant Recommendations via Word2Vec (https://github.com/KenDooley/Tast.io) , our thought to simplyfy the function is:

Each article has its weight which is the sum of its each word's tf-idf weight

According to the ratio of one word's weight to the article's weight, we could get each word's importance in this article (percentage). The more important a word is, the more influence it has to the article's vector.

We multiply each word's vector in word2vec and its importance. Then the sum of all words' result would be the vector of this article.

Below is the Figure 3 to present how the function works.

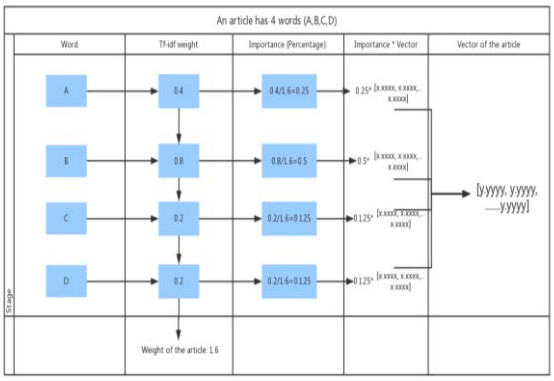

Figure 3. The workflow of the function.

Why do we choose D2V+LSI?

After generated three similarities, we use A/B testing methods to compare the results:

In the comparison, we found:

In LSI similarity, its result is easier to be extreme and is sensitive to the words the description used. If 2 courses use different structure or different words but are discussing the same topic, the LSI result would regard them as not relevant.

In D2V similarity, its result are usually high and seems courses are quite similar to each other. The reason could be usually the description is quite short and the numbers of description is not very large. Thus only a few words in one description would be used to calculate D2V. After the percentage (importance), their vector result would not be so different to each other's.

Finally, when using LSI + D2V, it present the most reasonable result compared to above two methods according to the review of participants, as shown in Figure 4.

Some other algorithms we tried:

When we intend to present our recommendation to our users, one question happens: how many courses shall we present? How much shall we set as the threshold of similarity?

At the beginning, we tried to set the threshold as a fixed figure. However, as a result, the number of the recommendation courses varied a lot.

Thus we tried to use Kmeans to cluster the courses according to the matrix of D2V + LSI. With these clusters, we could find a range of the courses and then list them according to the similarity.

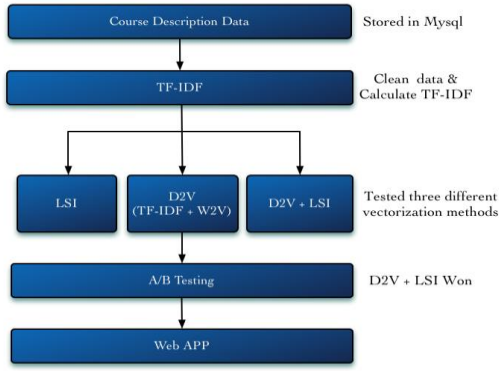

Figure 4. Algorithms implemented.

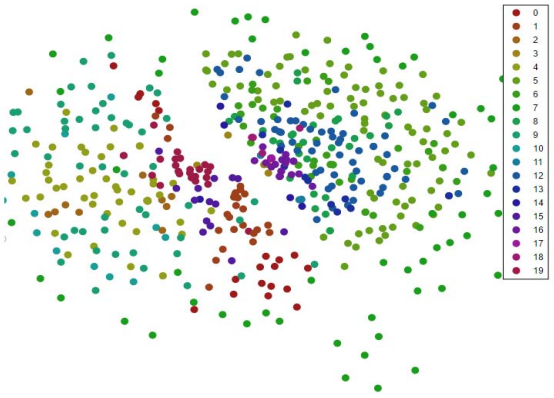

Figure 5. A part of the visualization of the clusters.

However, it does not work very well and even contradict our main purpose of the similarity [5]:

Kmeans would not balance the number of courses in each cluster. Some clusters might have only 3-4 courses while other cluster would have 50+ courses. It still has the issue that the number of the courses varied a lot.

Cluster doesn't equal to similarity. Some courses have very high similarity would be in different cluster while some courses in the same cluster just have a very low similarity.

Below is a part of the visualization of the clusters, as shown in Figure 5.

Thus, right now, we just present 10 courses with the highest similarity. To further working on this issue, we consider to use user's reviews to eliminate the unnecessary courses in the result.

V. CONCLUSION

Course recommendation is worth researching. Using LSI + D2V to resolve the course recommendation can present the most reasonable result compared to using above two methods alone according to the review of participants. To further our research, the next step is to determine the number of courses we shall present, and the threold of similarity according to real facts.

REFERENCES

[1] Dietmar Jannach, Markus Zanker, Alexander Felfernig, etc. Recommender systems: an introduction. Cambridge University Press. 2011:5-12.

[2] Bertolazzi P, Bock ME, Guerra C (2013) On the functional and structural characterization of hubs in protein-protein interaction networks.Biotechnol Adv 31(2):274–286.

[3] Guzzi PH, Mina M (2012) Investigating bias in semantic similarity measures for analysis of protein interactions. In: Proceedings of 1st international workshop on pattern recognition in proteomics, structural biology and bioinformatics, pp 71–80.

[4] Cannataro M, Guzzi PH, Veltri P. Protein Interaction Data: technologies, databases and algorithms, ACM Comput Sur , 2010, vol. 43 ,pg. 1-36.

[5] Harispe S, Sanchez D, Ranwez S, Janaqi S, Montmain J (2013) A framework for unifying ontology-based semantic similarity measures: a study in the biomedical domain. J Biomed Inf.