International Journal of Electronics Engineering (ISSN: 0973-7383)
Volume 11 • Issue 2 June - Dec 2019 pp. 105-108 www.csjournals.com

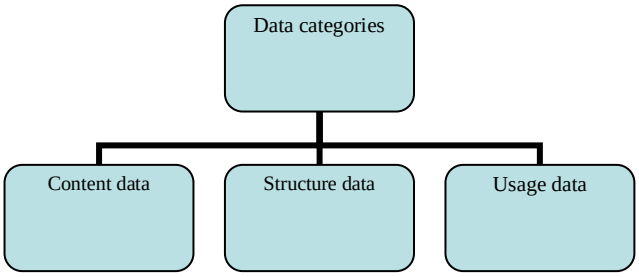
Solving a long-tail keywords problem in web-service on internet- A Review

Er. Meenakshi Sharma[1], Anjali[2], Dr. Satpal[3]
Department of Computer Science, Baba Mastnath University, Haryana , India
anjali87dagar@gmail.com[2]

Abstract: Web service recommendation can help address the unbalanced, long tail distribution of service usages, where only a small portion of web services are being known and intensively used and the large portion of other services are seldom noticed. The continuous growth in the size and use of the World Wide Web imposes new methods of design and development of online information services. Most Web structures are large and complicated and users often miss the goal of their inquiry, or receive ambiguous results when they try to navigate through them. However, a Web service recommender system is a multi-criteria decision-making system, in which users should evaluate the performance of a Web service based on more than one QoS property, rather than simply assign each candidate service a single-value ranking score during the random walk. To overcome the problem of severe sparsity of historical usage data and unsatisfactory quality of description content, we are focusing on Convolutional Neural Network approach of Deep Neural Networks to boost the performance of the network. Moreover existing recommendation system is time consuming, to eliminate this problem we will work on time series also.
Keywords—Webservices , CNN, sparsity, Qos

I. INTRODUCTION

The continuous growth in the size and use of the World Wide Web imposes new methods of design and development of online information services. Most Web structures are large and complicated and users often miss the goal of their inquiry, or receive ambiguous results when they try to navigate through them. Instead of this, e-business sector is quickly developing and Web marketplaces needs that do in advance the requirements of the clients are more evident than always [1]. Consequently, the obligation for expecting user necessities in order to progress the usability and user preservation of a Web site can be addressed by identifying it. Web facts are those that can be composed and used in the framework of Web personalization. These data are classified in four categories. figure of data categories here.



• Content data are accessible to the end-user appropriately organized. They can be modest text, pictures and organized data, such as data retrieved from databases.

• Structure data signify the method of content is systematized. They can be either entity of data utilized within a Web page such as XML tags,HTML or data entities used to put a Web site merged such as hyperlinks linking one page to another.

• Usage data characterize a Web site's procedure, like a visitor's IP address, period and access date, complete path retrieved, referrers' address and other features that can be included in a Web access log.

• User profile data offer information about the customers of a Web site. A user profile comprises demographic information i.e. name, country, age, education, marital status, interests etc. for each user of a Web site, and users' interests and preferences as well. Such data is developed over registration forms, questionnaires and also by analyzing Web usage logs.

Web services provide an efficient, convenient, and flexible way of delivering businesses, interacting with customers, and exchanging or sharing data on the Web. They also allow complicated and instant services accessible to ubiquitous mobile devices, such as smart phones and tablets. As a result, the number of online services and the size of their users have been dramatically increased over the past decade [2]. This makes service recommendation, which is to proactively suggest interesting web services to users, a promising solution to effectively advertising web services, facilitating service discovery, and improving user experiences. This is analogous to making push notifications or recommending related products to customer by e-commerce companies, such as Amazon, which have been demonstrated to be very effective in promoting products and improving customers' purchasing experiences. Moreover, web service recommendation can help address the unbalanced, long tail distribution of service usages, where only a small portion of web services are being known and intensively used and the large portion .

III Related Work

A Web service recommender system selects a Web service with optimal QoS performance from a group of service

candidates and recommends it to a service user. Since the QoS performance of Web services is generally not predetermined, neighborhood-based CF algorithms, which mainly include user-based [1, 2], item-based [3, 4] and hybrid [5, 6] approaches, are employed to make QoS prediction. In addition, to achieve better prediction performance, other researchers incorporate context information into the basic CF methods. The most widely discussed context information is location [7, 8]. They hold the opinion that the geographically close users or services may have similar QoS experience. Thus, more historical QoS values from geographically close neighbours can be obtained for more accurate QoS prediction. However, to the best of our knowledge, none of the existing neighbourhood based CF methods for Web service recommendation take into consideration the service invocation time information, which is another important context factor affecting QoS. This greatly limits the QoS prediction accuracy, since the QoS performance of Web services is highly related to invocation time due to some time-varying factors (e.g., service status, network condition, etc.). Another drawback of neighbo rhood-based CF is that it is vulnerable to data sparsity. The indirect similarity inference approach [9] focuses on inferring indirect relationship between any two given users to alleviate data sparsity. However, this method cannot automatically recognize indirect similar neighbours for a given user. Other researchers employ random walk algorithms for automatic discovery of related objects. Yildirim et al. [10] perform random walk on an item graph (extracted by treating items as nodes and item similarities as edges) to find more related items, and assign each item a stationary visiting probability as its ranking score. This approach can alleviate data sparsity to some extent, but does not take user information into consideration. In [11– 12, 13], random walk is performed on a hybrid graph, which is extracted by treating both users and items as nodes and the user-item interactions as edges. Similarly, items with higher stationary visiting probabilities are more likely to be recommended to target users. These methods based on random walk are applicable to one-dimensional rating systems. However, a Web service recommender system is a multi-criteria decision-making system, in which users should evaluate the performance of a Web service based on more than one QoS property, rather than simply assign each candidate service a single-value ranking score during the random walk.

ii. Problem formulation

As long-tail services are playing an increasingly important role in web API economy, how to recommend long-tail web services effectively is becoming a key issue. However, very scarce work has focused this problem, and traditional web service recommendation methods perform poorly on the long-tail side.

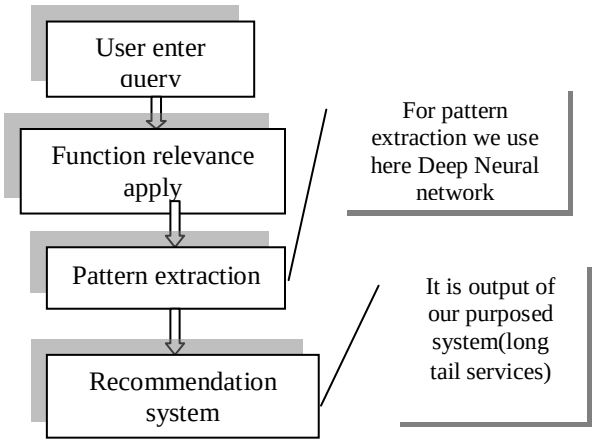
Along with lack of consideration of time information, another problem that we faced is to handle data sparsity. The number of users and services on the Internet is very large. Even very active users may invoke only a few Web services and even very popular Web services may be invoked by only a few users. This problem commonly referred to as the sparsity problem. In literature, Stacked Denoising Auto encoders (SDAE) have been widely discussed to alleviate data sparsity [14]. Its basic idea is to learn the patterns of developers' preference instead of modeling individual services. However in past [14] only the historical usage data was assumed as sparsity.This is the issue of considerations, in our research we will be going togive stress on more parameters such as: QoS, user profiles and social connection betweenservices. Moreover to boost the output performance we will be going to investigate more sophisticated deeplearning models such as convolutional neural networks(CNN) for Classification.

So, motivated bythe existing Stacked Denoising Autoencoders (SDAE) algorithm and the features of Webservices, we propose convolutional neural networksto handle data sparsity for Web service recommendation and to enhance performance.The key contribution of our work isone-fold:

A convolutional neural network is employed to handle data sparsity, by performing personalized random walk on both the user graph and the service graph, to discover more indirect similar users and services. Then, the user and service random walk results are combined together for final social connection between services, user profiles and QoS prediction.

IV METHODOLOGY

Today all know long tail services is a big problems to solve the problem here we propose methodology as following below in figure .



First of all, user enter query. After getting user query, some basic pre-processing tasks has to be performed. Function relevance is applied to perform long-tail services. Pattern extraction is the next step to integrate with preference. For the extraction purpose we use novel technique named as Deep Neural Network which overcome the problem of existing techniques. Further output of this phase is passed to recommendation system and after that sorting is performed from database and create ranked list of long tail services. This completes the process of recommendation system.

Tools and Techniques: For the implementation purpose we will be used following tools:

Python: Python is an open source, high level programming scripting language. It is designed to be highly readable. In

machine learning python is a great language for many reasons as for web development, data analysis, artificial intelligence, and scientific computing it is highly recommended because of its clear, short and easy syntax. Extreme easy text manipulation make ample of web development and documentation. No need to declare variable, type, methods in source code which make code short and flexible which other languages such as Java, C are lack off.

Keras: Keras is a high-level neural networks API, written in Python and capable of running on top of either TensorFlow or Theano. It was developed with a focus on enabling fast experimentation. It is neural network Library and used as a front end.

Tensor Flow: It is a google library which is used for the numerical computations. It is used at the back end.

We need following system requirements for Python:

Operating system	**Linux**
RAM	**4GB**
Processor	**Intel Core I3**

v. Conclusion

Web services provide an efficient, convenient, and flexible way of delivering businesses, interacting with customers, and exchanging or sharing data on the Web. They also allow complicated and instant services accessible to ubiquitous mobile devices, such as smart phones and tablets. As a result, the number of online services and the size of their users have been dramatically increased over the past decade. As long-tail services are playing an increasingly important role in web API economy, how to recommend long-tail web services effectively is becoming a key issue. However, very scarce work has focused this problem, and traditional web service recommendation methods perform poorly on the long-tail side. To overcome the problem of severe sparsity of historical usage data and unsatisfactory quality of description content, we are focusing on Convolutional Neural Network approach to boost the performance of the network and reduce time consumption.

vi. References

J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence. Morgan Kaufmann Publishers Inc., 1998, pp. 43– 52.

1. R. Jin, J. Y. Chai, and L. Si, "An automatic weighting scheme for collaborative filtering," in Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2004, pp. 337–344.
2. 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.
3. [8] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in Proceedings of the 10th international conference on World Wide Web. ACM, 2001, pp. 285–295.
4. Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Wsrec: A collaborative filtering based web service recommender system," in IEEE International Conference on Web Services (ICWS'09). IEEE, 2009, pp. 437–444.
5. Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Qos-aware web service recommendation by collaborative filtering," Services Computing, IEEE Transactions on, vol. 4, no. 2, pp. 140–152, 2011. [11] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for qos-based service recommendation," in Web Services (ICWS), 2012 IEEE 19th International Conference on. IEEE, 2012, pp. 202–209.
6. Y. Shen, J. Zhu, X. Wang, L. Cai, X. Yang, and B. Zhou, "Geographic location-based network-aware qos prediction for service composition," in Web Services (ICWS), 2013 IEEE 20th International Conference on. IEEE, 2013, pp. 66–74.
7. M. Papagelis, D. Plexousakis, and T. Kutsuras, "Alleviating the sparsity problem of collaborative filtering using trust inferences," in Trust management. Springer, 2005, pp. 224–239.
8. H. Yildirim and M. S. Krishnamoorthy, "A random walk method for alleviating the sparsity problem in collaborative filtering," in Proceedings of the 2008 ACM conference on Recommender systems. ACM, 2008, pp. 131–138.
9. S. Shang, S. R. Kulkarni, P. W. Cuff, and P. Hui, "A randomwalk based model incorporating social information for recommendations," in Machine Learning for Signal Processing (MLSP), 2012 IEEE International Workshop on. IEEE, 2012, pp. 1–6.
10. Z. Zhang, D. D. Zeng, A. Abbasi, J. Peng, and X. Zheng, "A random walk model for item recommendation in social tagging systems," ACM Transactions on Management Information Systems (TMIS), vol. 4, no. 2, p. 8, 2013.
11. Y. Zhou, L. Liu, C.-S. Perng, A. Sailer, I. Silva-Lepe, and Z. Su, "Ranking services by service network structure and service attributes," in Web Services (ICWS'13), 2013 IEEE 20th International Conference on. IEEE, 2013, pp. 26–33.
12. M. Jiang, P. Cui, F. Wang, Q. Yang, W. Zhu, and S. Yang, "Social recommendation across multiple relational domains," in Proceedings of the 21st ACM international conference on Information and

knowledge management. ACM, 2012, pp. 1422–1431.
13. Bai, Bing, et al. "DLTSR: A Deep Learning Framework for Recommendation of Long-tail Web Services." IEEE Transactions on Services Computing (2017).
14. Hinton, Geoffrey, et al. "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups." IEEE Signal Processing Magazine 29.6 (2012): 82-97.