2017 IEEE 24th International Conference on Web Services

Service2vec: A Vector Representation for Web Services

Yuqun Zhang*, Mengshi Zhang†, Xi Zheng‡, Dewayne E. Perry†
* *Department of Computer Science and Engineering, South University of Science and Technology of China*
Shenzhen, Guangdong, China
† *Department of Electrical and Computer Engineering, The University of Texas at Austin*
Austin, TX, USA
‡ *School of Information Technology, Deakin University*
Melbourne, Victoria, Australia
Email: zhangyq@sustc.edu.cn, {mengshi.zhang, perry}@utexas.edu, xi.zheng@deakin.edu.au

***Abstract*—Among the approaches that investigate the similarity between web services, hardly any concentrates on the impacts from contexts. In this paper we introduce service2vec which is an approach to represent web services as service embeddings based on a recent popular deep learning technique word2vec. Our approach composes and combines web services to be a document that is trained by the modeling technique of word2vec. As a result, each web service in the document is vectorized. By taking the advantage of word2vec, the resulting service embeddings of service2vec can be used to illustrate the contextual relations between web services. The experimental results suggest that service2vec can deliver contextual similarity between web services.**

I. Introduction

In recent years, topic modeling techniques, e.g., LDA, have been widely applied in the research domain of web services [1]–[5]. In this paper, we study one recent influential deep learning framework word2vec [6] [7], that can be used to vectorize words in documents and compute the contextual similarities between them. Particularly, with the vector representation of words, word2vec not only enables similar words to be close to each other, but also delivers multiple degrees of similarity. To reduce the computational complexity caused by hidden layers, word2vec develops new architectures, namely the Continuous Bag-of-Words Model and the Continuous Skip-gram Model, that cancels the unnecessary hidden layer and improves the training data efficiency.

Our work service2vec roots from the modeling framework of word2vec. Typically, web services are modeled as "words". The "documents" in service2vec are formed by composing and grouping web services, that can be vectorized after applying the techniques of word2vec. Compared with other approaches of similarity measurements for web services, the advantages of vectorizing web services are: *1)* providing contextual similarity measurements between any web service under different documents; *2)* providing a representation of web services in a similar manner to the famous formula *vector*("husband") = *vector*("king") - *vector*("queen") + *vector*("wife") in word2vec. A set of experiments are implemented to validate them.

The rest of the paper is organized as follows. Section II introduces the related work. Section III demonstrates service2vec, its associated features, and their efficacy. Section IV concludes this paper and introduces the future work.

II. Related Work

Some recent web service techniques are focused or based on the similarity of web services. Liu et al. [3] develop a framework that combines LDA and SVM for classifying web services where labeling training data could be automatically realized. Zhang et al. [4] propose a user-demand-service recommendation system by applying LDA and the Gibbs sampler.

Word2vec has been widely referenced after it was released. App2vec [8] is a typical application where mobile apps are mapped as words and their corresponding downloading sequence together are mapped into documents. Deepwalk [9] improves the training data coverage by applying random walk algorithms based on word2vec. Bai et al. [10] use word2vec to study sentiment computation and classification for Weibo (the Chinese version of Twitter), where a sentiment dictionary and an emotional dictionary are built by the word2vec framework to capture the emotion tendencies of Weibo messages. Chung et al. [11] propose Audio Word2vec to offer the vector representation of fixed dimensionality for variable-length audio segments to a good degree.

III. Service2vec

In this section, we present service2vec that references the modeling techniques of word2vec, e.g., mapping the concepts of words and documents. In service2vec, web services are analogized as words. The corpus units are composite services that are formed by any service composition method and used for composing documents.

978-1-5386-0752-7/17 $31.00 © 2017 IEEE
DOI 10.1109/ICWS.2017.114

A. Word2vec Recap

Many NLP techniques treat words as atomic units without considering considering the similarity between them. Though they are advanced in simplicity and robustness, it is argued that they are less effective when data is limited in certain domains. Word2vec [6], [7] is designed to compute the continuous vector representations from large data sets.

With the vector representation of words, word2vec can realize contextual similarity between words, that is, for target word, not only the top K most similar words can be presented, but also the pairwise similarity can be identified. For instance, given the word embeddings of a pair {"France", "Paris"} and the word "Italy", the word embedding of "Rome" should be derived ideally by $vector(``Paris") - vector(``France") + vector(``Italy") = vector(``Rome")$. Similarly, the word embedding of "Berlin" can be derived when the word embedding of "Germany" is given. So is the word "Tallahassee" for "Florida".

B. Motivation

While there are a lot of existing service composition/recommendation methods, e.g, bag-of-words [12], vectorization [13], it is argued that they do not take contextual information into account and therefore cannot capture the services that function similarly under the same contexts [8]. Some examples are as follows

Motivating Example: Assume user A uses sequential services {"Zillows", "Walmart Supermarket", "Italian Restaurant", "In-Network Hospital", "College Football Schedules"}. It might indicate that user A would like to find a possible house to purchase/rent with walmart supermarket, Italian restaurant, and in-network hospital nearby. At last he might want to reserve some college football tickets for the coming games. On the other hand, user B experiences sequential services {"College Football Schedules", "Italian Restaurant", "Walmart Supermarket", "In-Network Hospital", "Zillows"}, that might indicate that user B wants to arrange a set of activities to fulfill on weekends and at last he just wants to check some housing prices information for fun. It is possible that same set of composite services could represent different user behaviors.

With contextual information, the examples above could be better captured and understood.

C. Solutions

The contextual information mentioned above can be provided by service2vec. In this paper, we use the concepts of words and documents from word2vec. In particular, words are mapped to be the names of web services. Composite services are generated by applying service network generation methods [14]. A service network is a multi-layer direct graph that can be used for delineating the input/output transitions between web services. Usually in a service network, web services are modeled as nodes, and input/output transitions as edges. In general, building a service network is equivalent to building and aggregating service compositions by matching all the input/output transitions between web services. Figure 1 is an example of service network, where blue dots represent web services and orange dots represent input/output parameters. In this paper, a composite service refers to a full path of web services that cannot be extended with further starting or ending web services, e.g., $\{w_1, w_5, w_{10}, w_{14}\}$.

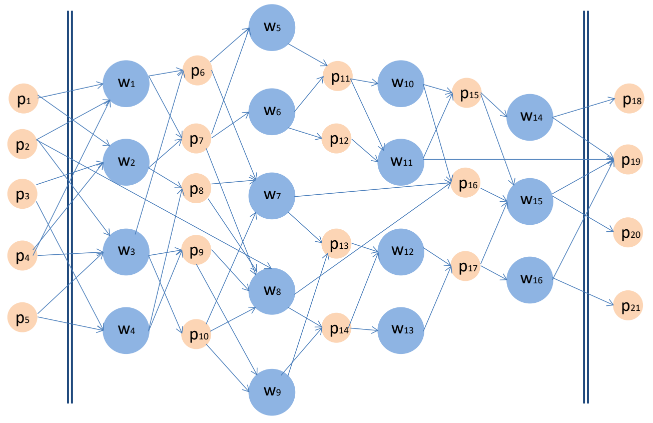

Figure 1: A service network example

Documents are then made by aggregating all the composite services.

Using service networks for generating web services can obtain the following advantages:

1) Generating service networks enables unified logics to assign input/output relations for a full/large coverage of a web service data set. In this way, even the web services that are not much semantically relevant could be possibly composed to generate a resourceful context.

2) The generation of service networks can be simply manipulated such that composite services that are similar to one another can be placed close to each other in a document. For instance, when generating a service network, a full path is generated by recursively matching the input/output transitions between web services and concatenating them. This recursion can result in aggregating full paths that are most similar to each other. For instance, in Figure 1, starting from w_1, $\{w_1, w_5\}$ and $\{w_1, w_7\}$ are captured and stored adjacently. Then taking $\{w_1, w_5\}$, $\{w_1, w_5, w_{10}\}$ and $\{w_1, w_5, w_{11}\}$ are captured and stored adjacently. So are $\{w_1, w_6, w_{10}\}$ and $\{w_1, w_6, w_{11}\}$ after taking $\{w_1, w_6\}$. In the end, $\{w_1, w_5, w_{10}, w_{14}\}$, $\{w_1, w_5, w_{11}, w_{14}\}$, $\{w_1, w_6, w_{10}, w_{14}\}$, and so forth are

web services	threshold = 0.37 (1054 edges)	threshold = 0.38 (875 edges)	threshold = 0.4 (598 edges)	threshold = 0.42 (561 edges)
get map of address	max price liquid service	title media recommended price quality service	price whiskey coffee Ziko service	get population density of location
	geopolitical entity corporation profession service	title media price quality service	missile government organization funding service	meat taxed price physical quantity service
	get traffic information	municipal unit skilled occupation time duration service	country sports position service	gazetteer lookup location
	title film price quality service	city weather system service	title comedy film taxed price quality service	real-time geocoding
	google static maps API	care organization investigating service	shopping mall price purchaseable item range service	country drought service
find nearby Wikipedia articles	title comedy film taxed price quality service	user romantic novel price service	query parser location	publication publisher service
	title video media MM service	search raw address	apple price service	get location of address Yahoo maps
	municipal unit weather system service	title obtainable video media service	title saving service	book search service
	query parser location	weapon missile funding I-raq service	lemon fruit price service	title science fiction film max price quality service
	search formatted address	publication number book author service	title obtainable video media service	university lecturer in a-cademia current semester service

Table I: An example of top-5 similar web services with documents generated under different thresholds

stored adjacently in order in a document. In this way, we could simulate the underlying patterns that in regular documents, words in the same context are usually similar to one another.

3) The existing methods to build service network are mostly semantic-based, that is strongly related with the thresholds to determine the similarity level of network components, e.g., WordNet [14]. By adjusting the thresholds, one can obtain different edges based on the same set of nodes in service networks. In other words, we could easily have different documents with the same set of words for evaluating service2vec under different circumstances.

D. Evaluation

A data set OWLS-TC4 [15] is applied to evaluate the efficacy of service2vec. This data set collects OWL-S files of 1,083 web services that scale over medicare, food, travel, etc. An OWL-S file includes modules ServiceProfile, ServiceGrounding, and ServiceModel, where ServiceProfile encloses input/output information that are used to build service network upon.

To determine whether web services can be connected via matching input/output transitions between them, the model Sequence Matcher [16] is used in our experiments. In Sequence Matcher, thresholds are used to determine whether two words/terms match one another. In particular, an output parameter of web service A matches an input parameter of web service B when their similarity score is larger than the preset similarity threshold.

Table I demonstrates an example of top-5 similar web services with two web services "get map of address" and "find nearby Wikipedia articles" under multiple documents generated by different similarity thresholds of the Sequence Matcher model (the names of web services are adjusted for better understanding), where the number of the generated edges are displayed together with the similarity thresholds. Smaller similarity thresholds indicates larger chances that input/output parameters can match each other and therefore result in more service transition and larger-sized documents. It can be discovered that under different documents, the top-5 recommendations appear to be diverse. For instance, for "find nearby Wikipedia articles" with the threshold 0.38, "user romantic novel price service" and "publication number book author service" are recommended because they are contextually similar to "articles". On the other hand, "search raw address" is recommended for being similar to "nearby", and "title obtainable video media service" may be similar to "Wikipedia".

Table II illustates an example of presenting web services in a similar manner with *vector*("wife") = *vector*("queen") - *vector*("king") + *vector*("husband") in word2vec. In our experiments, web service "get map of address" is mapped as "king", "google geocoding API" is mapped as "queen", "hospital investigating service" is mapped as "husband", and Table II displays the top-5 most possible web services that can be mapped as "wife". Note that under the threshold 0.37, the fact that "hospital diagnostic process time measure service" makes a reasonable "wife" implies that with larger-sized documents, it is more possible to find contextually analogy of web services.

IV. Conclusions and Future Work

In this paper we develop a framework namely service2vec that applies a recent deep learning modeling technique

threshold = 0.37	threshold = 0.38	threshold = 0.4	threshold = 0.42
google geocoding API	google geocoding API	google geocoding API	google geocoding API
hospital diagnostic process time measure service	grocery store food quantity service	national government scholarship quantity duration service	lenthu rent car service
author book recommended price service	title video media tax free price quality service	title media recommended price quality service	title vhs dvd service
title film action comedy service	get distance between cities worldwide	calculator distance spherical law of cosines	isbn book author service
shopping mall calendar date price camera service	get location of address	hospital investigating address service	get address of location

Table II: An example of top-5 similar candidates for web service analogy

word2vec. By carrying the advantages of word2vec, service2vec can deliver contextual similarity measurements among web services under documenting service compositions. In addition, it can provide pairwise analogies.

In service2vec at this point, web services are solely modeled as their names of the service description files and mapped as "words". It could be extended in the future by including more components of service description files. Moreover, service networks are applied to generate documents for simplicity. Future work could be conducted such that more real-world service composition methods are used to generate different types of documents. Those real-world service composition methods can be examined whether they result in the similar service composition or not.

References

[1] Xi Chen, Zibin Zheng, Qi Yu, and Michael R. Lyu. Web service recommendation via exploiting location and qos information. *IEEE Transactions on Parallel and Distributed Systems*, 25(7):1913–1924, 2013.

[2] Qi Xie, Shenglin Zhao, Zibin Zheng, Jieming Zhu, and Michael R. Lyu. Asymmetric correlation regularized matrix factorization for web service recommendation. In *IEEE International Conference on Web Services*, pages 204–211, 2016.

[3] Xumin Liu, Shaleen Agarwal, Chen Ding, and Qi Yu. An lda-svm active learning framework for web service classification. In *IEEE International Conference on Web Services*, pages 49–56, 2016.

[4] Yanmei Zhang, Tingpei Lei, and Yan Wang. A service recommendation algorithm based on modeling of implicit demands. In *IEEE International Conference on Web Services*, pages 17–24, 2016.

[5] Shanshan Feng, Jian Cao, Jie Wang, and Jing He. Group recommendations based on comprehensive latent relationship discovery. In *IEEE International Conference on Web Services*, pages 9–16, 2016.

[6] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *Computer Science*, 2013.

[7] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[8] Qiang Ma, S. Muthukrishnan, and Wil Simpson. App2vec: Vector modeling of mobile apps and applications. In *Ieee/acm International Conference on Advances in Social Networks Analysis and Mining*, pages 599–606, 2016.

[9] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. *Eprint Arxiv*, pages 701–710, 2014.

[10] Xue Bai, Fu Chen, and Shaobin Zhan. A study on sentiment computing and classification of sina weibo with word2vec. In *IEEE International Congress on Big Data*, pages 358–363, 2014.

[11] Yu An Chung, Chao Chung Wu, Chia Hao Shen, Hung Yi Lee, and Lin Shan Lee. Audio word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder. In *INTERSPEECH*, pages 765–769, 2016.

[12] Hengshu Zhu, Huanhuan Cao, Enhong Chen, Hui Xiong, and Jilei Tian. Exploiting enriched contextual information for mobile app classification. In *ACM International Conference on Information and Knowledge Management*, pages 1617–1621, 2012.

[13] Bin Liu, Deguang Kong, Lei Cen, Neil Zhenqiang Gong, Hongxia Jin, and Hui Xiong. Personalized mobile app recommendation: Reconciling app functionality and user privacy preference. In *Eighth Acm International Conference on Web Search and Data Mining*, pages 315–324, 2015.

[14] Shuo Wang, Zhongjie Wang, and Xiaofei Xu. Mining bilateral patterns as priori knowledge for efficient service composition. In *IEEE International Conference on Web Services*, pages 65–72, 2016.

[15] Matthias Klusch, Patrick Kapahnke, Benedikt Fries, Mahboob Alam Khalid, and Martin Vasileski. Owls-tc4. http://projects.semwebcentral.org/projects/owls-tc/.

[16] Sequence matcher. https://docs.python.org/2/library/difflib.html.