Personalized Service Recommendation Based on User Dynamic Preferences

Benjamin A. Kwapong(✉), Richard Anarfi, and Kenneth K. Fletcher

University of Massachusetts Boston, Boston, MA 02125, USA
{benjamin.kwapong001,richard.anarfi001,kenneth.fletcher}@umb.edu

Abstract. In order to personalize users' recommendations, it is essential to consider their personalized preferences on non-functional attributes during service recommendation. However, to increase recommendation accuracy, it is essential that recommendation systems include users' evolving preferences. It is not sufficient to only consider users' preferences at a point in time. Existing time-based recommendation systems either disregard rich and useful historical user invocation information, or rely on information from similar users, and thus, fail to thoroughly capture users' dynamic preferences for personalized recommendation. This work proposes a method to personalize users' recommendations based on their dynamic preferences on non-functional attributes. First, we compose a user's preference profile as a time series of his/her invocation preference and pre-invocation dependencies (i.e. the different services that were viewed prior to invoking the preferred service). We model a user's invocation preference as a combination of non-functional attribute values observed during service invocation, and topic distribution from WSDL of the invoked service using Hierarchical Dirichlet Process (HDP). Next, we employ long short-term memory recurrent neural networks (LSTM-RNN) to predict the user's future invocation preference. Finally, based on the predicted future invocation preference, we recommend service(s) to that user. To evaluate our proposed method, we perform experiments using real-world service dataset, WS-Dream.

Keywords: Service recommendation · User preference profile · LSTM-RNN · User preference evolution · Topic modeling · HDP

1 Introduction

Recommendation systems are attracting much attention because they provide users with prior knowledge of candidate choices to deal with information overload on the Web [1,2]. Most traditional personalized preference-based recommendation systems usually consider user's preferences at a point in time during service recommendation [1–6]. However, it is important to note that, typical user preferences change over time. It is therefore essential to incorporate users' dynamic preferences to provide accurate and timely personalized recommendations [7].

J. E. Ferreira et al. (Eds.): SCC 2019, LNCS 11515, pp. 77–91, 2019.
https://doi.org/10.1007/978-3-030-23554-3_6

For recommendation systems to provide personalized recommendations to users, they must provide a way to: (1) accurately capture and model user dynamic personalized preferences on non-functional attributes, (2) accurately predict user future preferences, and (3) recommend relevant service(s) that satisfies user dynamic preferences. Capturing user dynamic preferences on non-functional attributes is challenging. Some existing works, that have attempted to do so, have utilized either a session-based [8] or a time-series approach [3,7]. In these works, topic modeling such as Latent Dirichlet Allocation (LDA) [9] are used to mine topic distributions from service description documents for a user profile. However, LDA suffers from low efficiency, excessively long training time and low accuracy, especially in applications where the input document is relatively short.

There are some methods that use traditional recurrent neural networks (RNN) to infer user's future preferences for service recommendation [10,11]. These methods, however, suffer from vanishing and exploding gradient issue inherent with RNNs when user historical data in sequence gets larger [12]. Others use various linear predictors based on traditional statistical techniques, such as Autoregressive Moving Average and Autoregressive Integrated Moving Average [13,14], and Gaussian Process [7] algorithms. These time series regression approaches are very sensitive to outliers and depend on an unchanged cause and effect relationships which makes them unsuitable to predict user future dynamic preferences.

To address these limitations and provide accurate personalized recommendations, this work proposes a method that employs Hierarchical Dirichlet Process (HDP) [15,16] and long short-term memory RNN (LSTM-RNN) [12] to capture and predict user preferences for service recommendation. HDP and LSTM-RNN deals with the limitations introduced by LDA and regression methods respectively. The main contributions of this work are summarized as follows:

1. We model a user's invocation preference as a combination of a set of topic distribution, obtained from WSDL and non-functional attribute values, that were observed when that user invoked a service. We employ HDP to extract the topic distribution from the WSDL of the invoked service. In addition, we obtain topic distribution from all WSDL documents from the user's pre-invocation services (i.e. the different service(s) that were viewed prior to invoking the preferred service). This is to establish a relationship between user's intent and invoked service(s). We then aggregate invocation preferences at each timestamp to build a time series of that user's preference profile, which depicts the changes in his/her preferences.
2. Using the user's preference profile, we apply LSTM-RNN to learn and predict his/her future invocation preference, i.e. the topic distribution and non-functional attribute values of a prospective service. To recommend top-K services, we compute the similarity between the user's future non-functional attribute and candidate services. This similarity value is then used as weights in the weighted Jensen-Shannon Divergence [17] to compute the similarity

between user's future invocation preference and topic distribution of candidate services. Top-ranked services are then recommended to the user.
3. We perform a series of experiments using real-world services, WS-Dream [18], to evaluate and validate our proposed method.

The rest of the paper is outlined as follows. In Sect. 2 we discuss some of the notable and significant service recommendation works based on user dynamic preference. We present our proposed LSTM-RNN method in detail in Sect. 3 followed by our experiments, evaluations and analysis in Sect. 4. Finally, we conclude our paper and discuss some of the open ended challenges as a part of our future work in Sect. 5.

2 Related Work

To the best of our knowledge, this is the first recommendation technique that models user invocation preference to include non-functional attributes and topic distribution from WSDL documents extracted from user invocation and pre-invocation services history. However, the idea of personalized service recommendation and the importance of user intent has been exploited in some research areas. Wu et al. [19] employs a deep recurrent neural network approach to exploit current viewing history of the user to improve recommendation accuracy. The main difference between their work and our proposed method is that they failed to consider a user's pre-invocation services history during service recommendation. In addition, their method employed collaborative filtering approach, which primarily rely on information from similar users/items, thus failing to build personalized models for individual users with rich past information [7]. Other related personalized recommendation models [20,21] also employ collaborative filtering to make their recommendations.

Our use of topic modeling from user service invocations and pre-invocation services is similar to the work of Liu [7] and Uetsuji et al. [22]. In her work, Liu [7] proposed a method that uses LDA to model user's preferences and then applied the Gaussian Process to predict user's future preference. Uetsuji et al. [22] considered capturing user's intent in effecting service recommendation by using a topic tracking model. A customer's behavior was generated in a two-step probabilistic process; in the first step a topic was selected according to a topic probability distribution representing topic selection tendency. Then, in the next step the user's activity is determined according to activity probability distribution linked to the selected topic in the first step. They however trained their model using a probabilistic expectation maximization algorithm, which estimates parameters in statistical models. The importance of user intent in recommendation systems was further highlighted by Bhattacharya et al. [4]. They use tensor factorization techniques to encode user activity and then create an intent score. A combination of the intent score and contextual information produces recommendation scores. These recommendation scores are ranked through filtering and collaborative recommendation techniques. The methods in this work largely focus on probabilistic statistical models and therefore also have limitations of statistical regression mentioned in Sect. 1.

Another key consideration for service recommender systems is the concept of user preference evolution in user preference modeling. This essential because user preferences over time are rather dynamic than static. This renders traditional recommender systems, which considers user preferences at a point in time, less accurate as time unfolds and user preferences evolve. Zhou et al. [17], in their work, highlighted on the lack of automatic adaptiveness of the traditional user modeling systems to the dynamic nature of user preferences. They approached this problem by analyzing the characteristics of memory through ZGrapher. They then employed a Forgetting and Re-energizing User Preference (FRUP) algorithm to trace the user preferences. These preferences are divided into long-term, medium-term and short-term for the accurate description of memory patterns in different scenarios. In our work, however, we do not categorize the preferences into short or long term based on the premise that user preferences are dynamic. A short term preference could easily become a long term one and vice versa based on user preference evolution. Our model automatically captures these user dynamics by learning from the user's past invocations and pre-invocation dependencies.

Aside our difference in approach, the use of RNN has been exploited in other related service recommendation works [10,11,23]. These researchers, have leveraged the strength of RNN in handling a sequence of input data to fully capture user intent, patterns and behavior to accurately model the user profiles. Xia et al. [10] explores the provision of an explainable recommendation based on the sequential check-in data of the user. In their work, they make use of sequential check-in data to capture users' life pattern and intent, to describe the user's personal preference. They then employ a RNN, which they qualify as attention-based, to make a series of recommendations instead of simply showing top-N recommendations. RNNs have been applied to recommendation systems in the movie industry Chu et al. [11]. Here, they highlight the inability for collaborative filtering techniques to handle user changing habits. They subsequently built a prediction model based on RNN to handle the temporal factor of user interests. Their model treats a user's recent ratings or behavior as a sequence with each hidden layer modeling a user's rating or behavior in order. Li et al. [23] proposed a method for automatic Hashtag recommendation of new tweets. They used a skip-gram model to generate distributed word representations and then initially applied a convolutional neural network to learn semantic sentence vectors. After this, they made use of the sentence vectors to train a long short-term memory recurrent neural network (LSTM-RNN) and used the produced tweet vectors as features to classify hashtags.

3 Proposed Personalized Service Recommendation Based on User Dynamic Preferences

This section first discusses an overview of our proposed method and subsequently describes the main modules (engines) that drives our method. Figure 1 shows an overview of our proposed method. The three main engines are:

1. User preference profile engine: Based on the Hierarchical Dirichlet Process (HDP) [15], this engine is responsible for creating user preference profile, which is a time series of the user's invocation;
2. LSTM-RNN prediction engine: Takes user preference profile as an input and based on a transformation function, predicts its future invocation preference; and
3. Service ranking engine: Ranks services by computing similarity between future invocation preference and candidate services. The similarity function is based on weighted Jensen-Shannon Divergence [17]. Top-ranked services are then recommended to the user.

Details of each of these engines are described in the sections that follow.

3.1 Problem Definition

Let $U = \{u_1, u_2, ..., u_e\}$ be a set of users and $S = \{s_1, s_2, ..., s_f\}$ be a set of services. For each $s \in S$, there is a set of non-functional attributes, Q, that describe the service s. When a user $u \in U$ invokes a service $s \in S$, we record the user's invocation preference as a tuple:

$$I_s^u =< \Lambda, \breve{Q}, \Omega > \tag{1}$$

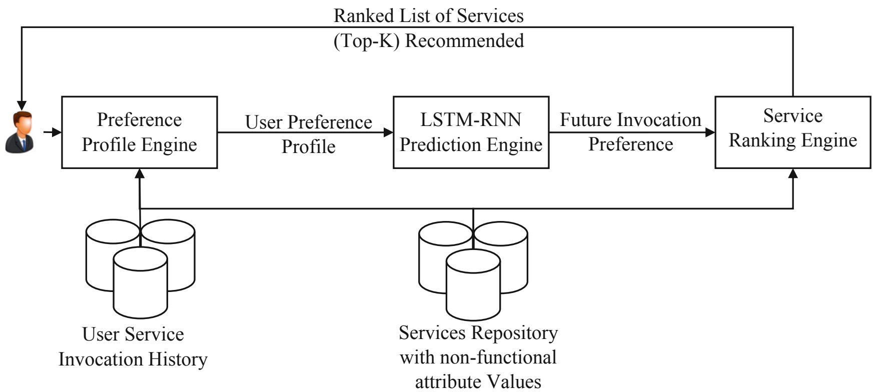


Fig. 1. Overview of the proposed LSTM-RNN method for personalized service recommendation based on user dynamic preferences

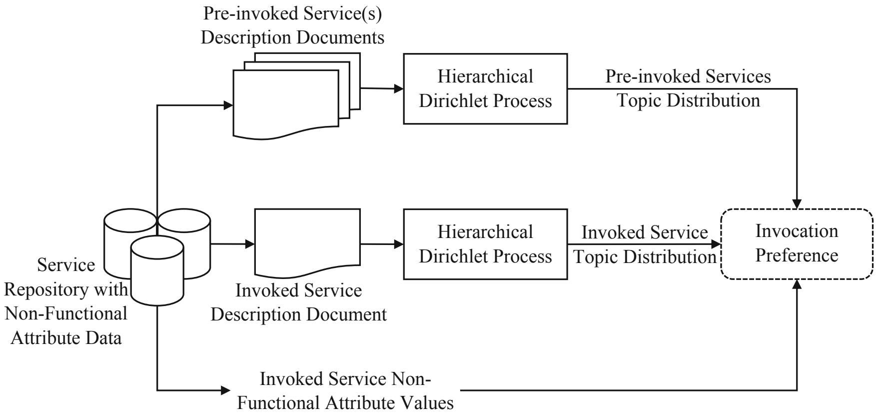


Fig. 2. User invocation preference process

where $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_n\}$ is the topic distribution extracted from the WSDL document of s, $\breve{Q} = \{\breve{q}_1, \breve{q}_2, ..., \breve{q}_m\}$ is the set of non-functional attribute values observed when s was invoked, and $\Omega = \{\omega_1, \omega_2, ..., \omega_k\}$ is the set of topic distribution recognized from $u's$ activities prior to invoking s. We model a user $u's$ preference profile P_u as a time series of his/her invocation preferences as:

$$P_u = \{I^u_{\bar{S}}(t), t = 0, 1, 2, ...\} \tag{2}$$

where $\bar{S} \subset S$.

Given a user $u's$ preference profile, P_u, we predict the user's future invocation preference on a probable service $\tilde{s} \in S$, using LSTM-RNN as:

$$f(P_u) :\longrightarrow \hat{I}^u_{\tilde{s}} \tag{3}$$

where $\hat{I}^u_{\tilde{s}}$, is the $u's$ future invocation preference i.e. the topic distribution and non-functional attribute values of that user's probable service.

Given S, Q and $\hat{I}^u_{\tilde{s}}$, the service top-K recommendation process can be modeled as a ranking in terms of the similarity between the user's future invocation preference and candidate services [24], so that for any two services S_i and S_j and a similarity function, Sim, the following is true.

$$S_i \succ S_j \Longleftrightarrow Sim(\hat{I}^u_{\tilde{s}}, S_i) \leq Sim(\hat{I}^u_{\tilde{s}}, S_j) \tag{4}$$

3.2 User Preference Profile Model

We model a user preference profile as a time series of that user's invocation preference. A user's invocation preference is constructed from the topic distribution obtained from the WSDL document together with the non-functional attribute values of the invoked service. We also include the topic distribution of all services a user visits prior to invoking the preferred service. Figure 2 describes the user invocation preference process. When a user u invokes a service s, we associate the service invocation WSDL document, W_s and non-functional attribute, $\breve{Q}$. Using this document as a corpus, we employ HDP to extract topics such that each word in the document has the probability of being assigned to a topic and each W_s is associated to a topic distribution Λ. We then append $\breve{Q}$ to Λ_s. We complete $u's$ invocation preference I by also adding the topic distribution of all $u's$ service interactions, Ω. To build $u's$ preference profile, we sort all invocation preferences by timestamp, in a chronological order and model each invocation preference with time as a time series (see Fig. 3).

User preference profile involves topic distribution modeling. Lately, probabilistic topic models such as Latent Dirichlet Allocation (LDA) [9], have been applied to extract and represent users' preference in different application scenarios [7]. LDA has been applied successfully to identify topics in documents and discover implicit semantic correlation among those documents. However, it suffers from low efficiency, excessively long training time and low accuracy, especially in applications where the input document is relatively short (e.g. WSDL document). Due to these limitations, we employ HDP for our topic modeling.

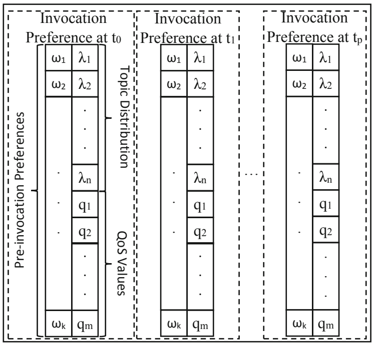


Fig. 3. User preference profile

HDP, an extension of LDA, is a multi-layer form of the Dirichlet Process (DP), designed to address cases in topic document modeling where the number of topic terms is not known in advance. For each document, a mixture of topics are drawn from a Dirichlet distribution, and then each word in the document is treated as an independent draw from that mixture [15]. Figure 4 shows a graphical model formalism of HDP. The global measure, G_0 is distributed as a Dirichlet Process (DP) with concentration parameter γ and base probability measure H:

$$G_0 \mid \gamma, H \sim DP(\gamma, H) \tag{5}$$

and the random measures G_j are conditionally independent given G_0, with distributions given by a Dirichlet Process with base probability measure G_0:

$$G_j \mid \alpha_0, G_0 \sim DP(\alpha_0, G_0) \tag{6}$$

The hyperparameters of the Hierarchical Dirichlet Process consist of the baseline probability measure H, and the concentration parameters γ and α_0. The baseline H provides the prior distribution for the topic of the i^{th} word in the j^{th} WSDL document, θ_{ji}. For each j let $\theta_{j1}, \theta_{j2}, ...$ be independent and identically distributed random topics distributed as G_j. Each θ_{ji} is a topic corresponding to a single observation x_{ji}. The likelihood is given by:

$$\theta_{ji} \mid G_j \sim G_j$$

$$x_{ji} \mid \theta_{ji} \sim F(\theta_{ji}) \tag{7}$$

which is the Hierarchical Dirichlet Process mixture model [15].

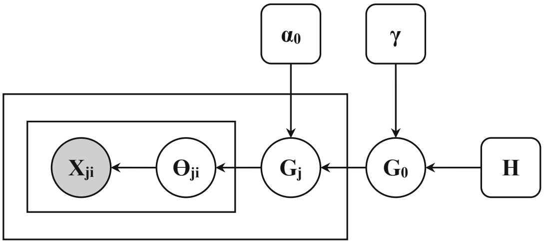


Fig. 4. A Hierarchical Dirichlet Process mixture model. In the graphical model formalism, each node in the graph is associated with a random variable, where shading denotes an observed variable. Rectangles denote replication of the model within the rectangle [15].

3.3 LSTM-RNN Model for User Preference Prediction

In this section, we discuss our proposed LSTM-RNN model. A recurrent neural network (RNN) is a type of artificial neural network (ANN) designed to recognize patterns in sequences of data [11]. Unlike traditional Neural Networks, which assume that all inputs and outputs are independent of each other, RNNs make use of sequential information. RNNs use their internal state to capture information that has been previously calculated, based on which the next item in the sequence is predicted. RNNs use back propagation algorithm [11], applied for every time stamp and this is commonly known as back propagation through time (BPTT). BPTT, however, introduces vanishing gradient and exploding gradient issues in RNN, when the number of items in the sequence gets large (long term dependencies) [12]. These limitations can be resolved by Long Short-Term Memory networks (LSTM), which we will employ in this work. LSTM are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter and Schmidhuber [12] and were refined and popularized by many people in following work [25–28]. An LSTM is composed of a cell, an input gate, an output gate and a forget gate. The major component is the cell state ("memory") which runs through the entire chain with occasional information updates from the input(add) and forget(remove) gates. An LSTM network computes a mapping from an input sequence $x = (x_1, ..., x_T)$ to an output sequence $y = (y_1, ..., y_T)$ by calculating the network unit activations using the following equations iteratively from $t = 1\ to\ T$ [29]:

$$i_t = \sigma(W_{ix}x_t + W_{im}m_{t-1} + W_{ic}c_t + b_i) \tag{8}$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1} + W_{fc}c_{t-1} + b_f) \tag{9}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cm}m_{t-1} + b_c) \tag{10}$$

$$o_t = \sigma(w_{ox}x_t + W_{om}m_{t-1} + W_{oc}c_t + b_o) \tag{11}$$

$$m_t = o_t \odot h(c_t) \tag{12}$$

$$y_t = \phi(W_{ym}m_t + b_y) \tag{13}$$

Fig. 5. Encoder-decoder LSTM architecture

- f: forget gate's activation vector
- i: input gate's activation vector
- o: output gate's activation vector
- h: output vector of the LSTM unit
- g: cell input activation function, generally tanh
- h: cell output activation functions, generally tanh
- c: cell activation vector
- W: weight matrices parameters
- b: bias vector parameters
- $\odot$: element-wise product of the vectors
- σ: the logistic sigmoid function
- ϕ: the network output activation function

We model our predicting function by using the encoder-decoder LSTM architecture [23], which is comprised of two models: the first is to read the input sequence and encode it into a fixed-length vector, and the second for decoding the fixed-length vector and outputting the predicted sequence. Figure 5 shows a simplified diagram of an encoder-decoder LSTM. Our prediction model takes as input, the invocation preference at the various timestamps $(t_0, t_1, t_2, ..., t_n)$ and predicts as output, the future user invocation preference at time t_{n+1}.

In this work, we use one LSTM to implement the encoder model and another LSTM for the decoder model. The encoder learns the relationship between the steps in the input sequence and develop an internal representation of those relationships. The decoder then transforms the learned internal representation of the input sequence into the correct output sequence. Figure 6 shows a basic representation of our sequence to sequence model with encoder-decoder LSTM.

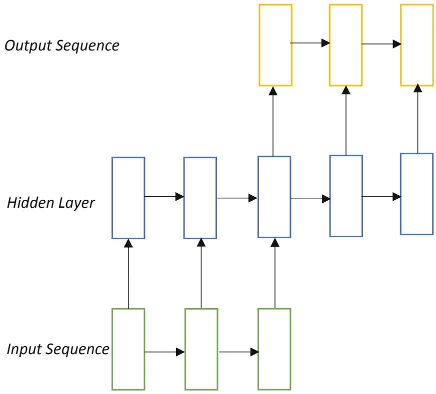


Fig. 6. Many-to-many RNN

3.4 Service Recommendation

To recommend a service to a user u, we calculate the weighed Jensen-Shannon divergence [30] between u's future invocation preference and the topic distribution of each candidate service using non-functional attribute values as weights. Given two normalized distributions $t_j = \{{t_j}^1, {t_j}^2, .., {t_j}^K\}$ and $r_j = \{{r_j}^1, {r_j}^2, .., {r_j}^K\}$, where K is the number of the bins in each histogram. Then the Jensen-Shannon Divergence (JSD) between t_j and r_j can be defined as:

$$JSD(r_j \| t_j) = \frac{1}{2} KLD(r_j \| m_j) + \frac{1}{2} KLD(t_j \| m_j) \tag{14}$$

$$KLD(r_j \| m_j) = \sum_{k=1}^{K} r_j^k log \frac{r_j^k}{m_j^k} \tag{15}$$

where $m_j = \frac{1}{2}(r_j + t_j)$.

For a JSD, it is alway necessary to select the hypothesis that produces smaller differences between the ideal and predicted distributions. Hence, the residual distribution is weighed using non-functional attribute values generated in a standard Gaussian function $g = \{g^1, g^2, ..., g^K\}$ $(\mu = 0, \sigma 2 = 1)$ to generate the Gaussian-weighted JSD (GJSD) [30]. The Gaussian weight function reinforces the influence of JSD for data points. GJSD is formulated as:

$$GJSD(r_j \| t_j) = \frac{1}{2} GKLD(r_j \| m_j) + \frac{1}{2} GKLD(t_j \| m_j) \tag{16}$$

$$GKLD(r_j \| m_j) = \sum_{k=1}^{K} g^k r_j^k log \frac{r_j^k}{m_j^k} \tag{17}$$

4 Experiments and Results

This section describes the experiments we conducted we conducted to evaluate and validate our proposed LSTM-RNN method for personalized service recommendation based on user dynamic preference. We also discuss our results.

4.1 Experimental Setup and Dataset Description

Our experiments were performed on WS-Dream dataset [18], a real-world web service quality of service performance dataset. The data set contains about 2 million web service invocation records of 5,825 web services with about 339 users. It contains the response time and throughput values for all invoked services. We visited all the WSDL addresses in the dataset and out of the 5,825 web services, 3,544 were found to be valid. Therefore, we conducted our experiments with these valid services.

From the 3,544 valid services, we identified 294 different groups of services based on their similar WSDL addresses. For each group of n valid services, we chose the first $1..n-1$ and designated those as pre-invocation dependencies while the n^{th} service was designated as the invoked service. Based on this, we used HDP to generate topic distributions and user profiles (timestamps of invocation preferences). We subsequently split the 294 different groups of services into training, testing and validating sets for experiments.

4.2 Performance Metric

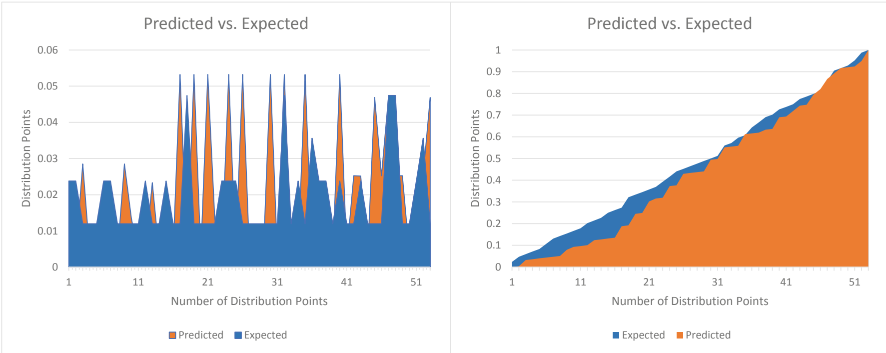
To quantitatively assess the overall performance of our LSTM-RNN model, Mean Square Error (MSE) was used to estimate the prediction accuracy. MSE is a scale dependent metric which quantifies the difference between the predicted values and the actual values of the quantity being predicted by computing the average sum of squared errors:

$$MSE = \frac{1}{N} \sum_{N=1}^{N} (y_i - \hat{y_i})^2 \tag{18}$$

where y_i is the observed value, $\hat{y_i}$ is the predicted value and N represents the total number of predictions.

4.3 Results

We conducted our experiment with two RNN based encoder-decoder models of LSTM and Gated Recurrent Unit (GRU). The experiments were ran over 25 epochs for both models with 256 memory cells each for encoder and decoder models. We employed *rmsprop* as optimizer and used *categorical crossentropy* as the loss function for our LSTM and GRU models. Figures 7a and b show an overlay distribution and a cumulative ogive of predicted and expected respectively. From these two figures, we observe that our trained model is sensitive to



(a) Distribution overlay (b) Cumulative distribution ogive

Fig. 7. Comparison between predicted and expected distributions

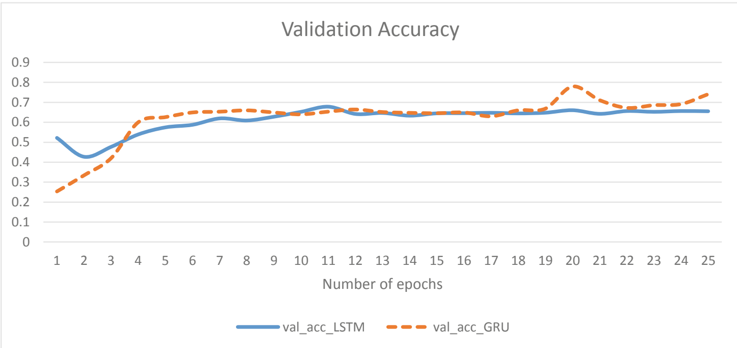


Fig. 8. Validation accuracy over 25 epochs for LSTM and GRU

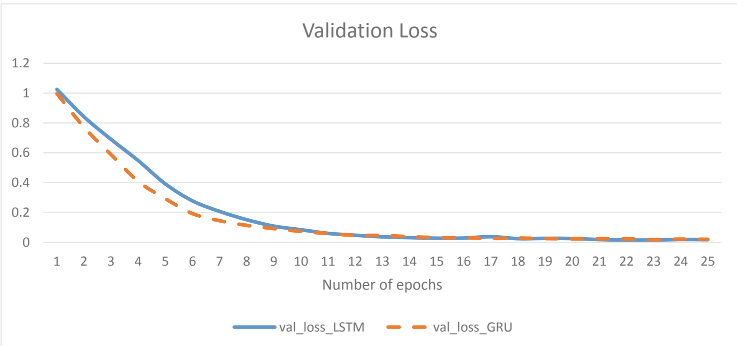


Fig. 9. Validation loss over 25 epochs for LSTM and GRU

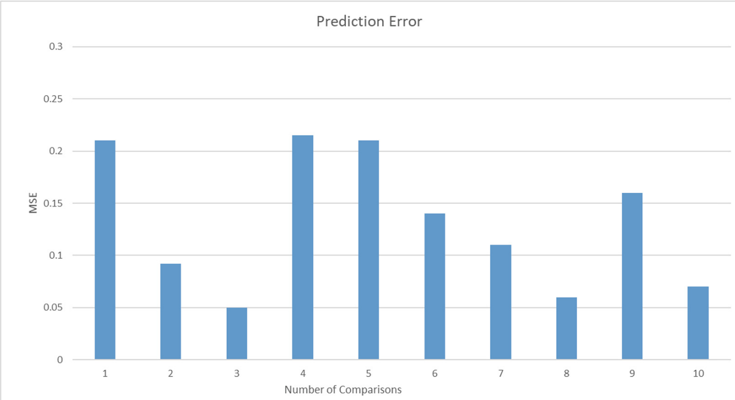


Fig. 10. MSE of predicted and expected distributions

some distribution data points in our test set. This, we attribute to the dataset on which we trained our model with. The graphs of validation accuracy and validation loss of the LSTM and GRU are shown in Figs. 8 and 9 respectively. It is evident from Fig. 8 that the performances of LSTMs and GRUs for our experiment were comparable. This comes as no surprise as both LSTMs and GRUs use memory to ensure that the gradient can pass across many time steps without vanishing or exploding. We also checked the accuracy of our prediction. For this experiment, the predicted distribution from our model is compared to the expected distribution and the MSE recorded. Figure 10 shows different error rates that we obtained for the 10 predictions we validated.

5 Conclusion and Future Work

Giving personalized recommendations is a very essential task for business and individuals alike. However, to increase recommendation accuracy, it is essential that recommendation systems include users' evolving preferences. It is not sufficient to only consider users' preferences at a point in time because user preferences change with time. In addition, users leave behind rich and useful historical invocation information, that could be employed to improve recommendation accuracy. In this work, we have proposed a method to personalize users' recommendations based on their dynamic preferences on non-functional attributes. Our proposed model creates a user preference profile as a time series of his/her invocation preference and pre-invocation dependencies (i.e. the different services that were viewed prior to invoking the preferred service). In our work, we also modeled a user's invocation preference as a combination of non-functional attribute values observed during service invocation, and topic distribution from WSDL of the invoked service using Hierarchical Dirichlet Process (HDP). We employed long short-term memory recurrent neural networks (LSTM-RNN) to predict the user's future invocation preference to recommend service(s) to that user. To evaluate our proposed method, we have performed experiments with WS-Dream dataset and results from our experiments were very promising.

As future work, we will expand our model in a new hypothesis and run several experiments with a couple of relevant datasets. Specifically, we would like to explore the advantages of including attention mechanisms in our LSTM model.

References

1. Fletcher, K.K., Liu, X.F.: A collaborative filtering method for personalized preference-based service recommendation. In: Proceedings of the 2015 IEEE International Conference on Web Services, pp. 400–407, June 2015
2. Fletcher, K.K.: A method for dealing with data sparsity and cold-start limitations in service recommendation using personalized preferences. In: 2017 IEEE International Conference on Cognitive Computing (ICCC), pp. 72–79, June 2017
3. Cao, B., Liu, X., Rahman, M.M., Li, B., Liu, J., Tang, M.: Integrated content and network-based service clustering and web APIs recommendation for mashup development. IEEE Trans. Serv. Comput. **PP**(99), 1 (2017)
4. Bhattacharya, B., Burhanuddin, I., Sancheti, A., Satya, K.: Intent-aware contextual recommendation system. In: 2017 IEEE International Conference on Data Mining Workshops (ICDMW), pp. 1–8, November 2017
5. Zheng, Z., Ma, H., Lyu, M.R., King, I.: Qos-aware web service recommendation by collaborative filtering. IEEE Trans. Serv. Comput. **4**(2), 140–152 (2011)
6. Chen, X., Zheng, Z., Liu, X., Huang, Z., Sun, H.: Personalized QoS-aware web service recommendation and visualization. IEEE Trans. Serv. Comput. **6**(1), 35–47 (2013)
7. Liu, X.: Modeling users' dynamic preference for personalized recommendation. In: Proceedings of the 24th International Conference on Artificial Intelligence. IJCAI 2015, pp. 1785–1791. AAAI Press (2015)
8. Quadrana, M., Karatzoglou, A., Hidasi, B., Cremonesi, P.: Personalizing session-based recommendations with hierarchical recurrent neural networks. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, pp. 130–137. ACM, New York (2017)
9. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. J. Mach. Learn. Res. **3**, 993–1022 (2003)
10. Xia, B., Li, Y., Li, Q., Li, T.: Attention-based recurrent neural network for location recommendation. In: 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 1–6, November 2017
11. Chu, Y., Huang, F., Wang, H., Li, G., Song, X.: Short-term recommendation with recurrent neural networks. In: 2017 IEEE International Conference on Mechatronics and Automation (ICMA), pp. 927–932, August 2017
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
13. Azzouni, A., Pujolle, G.: A long short-term memory recurrent neural network framework for network traffic matrix prediction. CoRR abs/1705.05690 (2017)
14. Dai, J., Li, J.: VBR MPEG video traffic dynamic prediction based on the modeling and forecast of time series. In: 2009 Fifth International Joint Conference on INC, IMS and IDC, pp. 1752–1757, August 2009

15. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical Dirichlet processes. J. Am. Stat. Assoc. **101**, 1566–1581 (2004)
16. Fletcher, K.K.: A quality-based web API selection for mashup development using affinity propagation. In: Ferreira, J.E., Spanoudakis, G., Ma, Y., Zhang, L.-J. (eds.) SCC 2018. LNCS, vol. 10969, pp. 153–165. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-94376-3_10
17. Zhou, B., Zhang, B., Liu, Y., Xing, K.: User model evolution algorithm: Forgetting and reenergizing user preference. In: 2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing, pp. 444–447, October 2011
18. Zheng, Z., Ma, H., Lyu, M.R., King, I.: WSRec: a collaborative filtering based web service recommender system. In: 2009 IEEE International Conference on Web Services, pp. 437–444, July 2009
19. Wu, S., Ren, W., Yu, C., Chen, G., Zhang, D., Zhu, J.: Personal recommendation using deep recurrent neural networks in NetEase. In: 2016 IEEE 32nd International Conference on Data Engineering (ICDE), pp. 1218–1229, May 2016
20. Qing-ji, T., Hao, W., Cong, W., Qi, G.: A personalized hybrid recommendation strategy based on user behaviors and its application. In: 2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC), pp. 181–186, December 2017
21. Xing, L., Ma, Q., Chen, S.: A novel personalized recommendation model based on location computing. In: 2017 Chinese Automation Congress (CAC), pp. 3355–3359, October 2017
22. Uetsuji, K., Yanagimoto, H., Yoshioka, M.: User intent estimation from access logs with topic model. In: 19th International Conference on Knowledge Based and Intelligent Information and Engineering Systems, pp. 141–149 (2015)
23. Li, J., Xu, H., He, X., Deng, J., Sun, X.: Tweet modeling with LSTM recurrent neural networks for hashtag recommendation. In: 2016 International Joint Conference on Neural Networks (IJCNN), pp. 1570–1577, July 2016
24. Fletcher, K.: A method for aggregating ranked services for personal preference based selection. Int. J. Web Serv. Res. (IJWSR) **16**(2), 1–23 (2019)
25. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with LSTM. In: 1999 Ninth International Conference on Artificial Neural Networks ICANN 1999 (Conf. Publ. No. 470), vol. 2, pp. 850–855 (1999)
26. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM networks. In: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, vol. 4, pp. 2047–2052, July 2005
27. Wollmer, M., Eyben, F., Keshet, J., Graves, A., Schuller, B., Rigoll, G.: Robust discriminative keyword spotting for emotionally colored spontaneous speech using bidirectional LSTM networks. In: 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3949–3952, April 2009
28. Graves, A., Jaitly, N., Mohamed, A.R.: Hybrid speech recognition with deep bidirectional LSTM. In: 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, pp. 273–278, December 2013
29. Sak, H., Senior, A., Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, 14–18 September 2014, pp. 338–342, September 2014
30. Zhou, K., Varadarajan, K.M., Zillich, M., Vincze, M.: Gaussian-weighted Jensen-Shannon divergence as a robust fitness function for multi-model fitting. Mach. Vis. Appl. **24**(6), 1107–1119 (2013)