DeepLTSC: Long-Tail Service Classification via Integrating Category Attentive Deep Neural Network and Feature Augmentation

Guobing Zou, Song Yang, Shengyu Duan, Bofeng Zhang, *Member, IEEE*, Yanglan Gan, and Yixin Chen

***Abstract*—With the explosive growth in the number and diversity of Web services, correlative research has been investigated on Web service classification, as it fundamentally promotes advanced service-oriented applications, such as service discovery, selection, composition and recommendation. However, conventional approaches are restricted to indiscriminatingly classify Web services, which can trigger many challenges. First, they have not made full advantage of the implicit relationships among multi-dimensional information of Web services, such as the increasing number of service categories. Thus, it leads to low effectiveness of learning and representing service features, failing to ensure the overall accuracy of service classification. Second, the imbalance of service distributions has been ignored, while it is observed that service categories reveal distinct long-tail characteristics. That results in low accuracy on service classification for those categories that contain fewer Web services. To handle the challenges of more effectively learning implicit service features across the service repository, and with a particular concentration on those tail categories that contain fewer Web services, we propose a novel framework called DeepLTSC to more accurately perform the task of Web service classification under long-tail distributions. In DeepLTSC, we first present an improved label attentive convolutional deep neural network (LACNN) with service categories, which can generate deep service features to improve the overall classification performance. Then, a proposed service feature augmentation model (SFA) together with focal loss function is integrated into DeepLTSC to further optimize service features, aiming to boost the classification accuracy on tail service categories. Extensive experiments are conducted on three large-scale real-world services datasets with different long-tail distributions. The results demonstrate that DeepLTSC significantly outperforms state-of-the-art approaches for Web service classification on both overall and tail categories.**

Manuscript received February 24, 2021; revised November 25, 2021; accepted January 24, 2022. Date of publication February 16, 2022; date of current version June 10, 2022. This work was supported by National Natural Science Foundation of China (No. 61772128, 62172088), and Shanghai Natural Science Foundation (No. 21ZR1400400). The associate editor coordinating the review of this article and approving it for publication was P. Bellavista. *(Corresponding author: Guobing Zou.)*

Guobing Zou, Song Yang, Shengyu Duan, and Bofeng Zhang are with the School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China (e-mail: gbzou@shu.edu.cn; samyangvictory@gmail.com; bfzhang@shu.edu.cn).

Yanglan Gan is with the School of Computer Science and Technology, Donghua University, Shanghai 201620, China (e-mail: ylgan@dhu.edu.cn).

Yixin Chen is with the Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO 63130 USA (e-mail: chen@cse.wustl.edu).

Digital Object Identifier 10.1109/TNSM.2022.3151764

***Index Terms*—Web services, long-tail classification, deep neural network, service category attention, service feature augmentation.**

I. Introduction

WITH the advances of service-oriented architecture (SOA) in software integration and applications, Web services are becoming popular and important building blocks for fast establishing next generation real-world applications. Driven by the benefits of Web service technologies, the category and number of Web services available on the Internet have been significantly increased in the past few years [1]. As of December 9, 2020, ProgrammableWeb,[1] as the largest online RESTful service repository, has registered over 23,800 APIs with nearly 500 diverse categories. The rapid proliferation of Web services further boosts service-oriented software design and development.

However, as the explosive growth in the category and number of Web services, it is placing a heavy burden on service providers. When a service vendor wants to release a new service in ProgrammableWeb, he has to choose its affiliated tag from a candidate pool of nearly 500 service categories, which is a very labor-intensive task for the registry of Web services. Service classification, as a fundamental way to facilitate a series of service-oriented tasks, has been widely applied for service discovery [2], service selection [3], [4], service composition [5], service recommendation [6], [7], [8] and service management [9]. Therefore, how to design an effective approach that can accurately classify Web services has become a critical research issue to be addressed.

In recent years, service classification has received many attentions and there are a lot of correlative studies [1], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19] which are dedicated to improving the accuracy of Web service classification. Existing approaches on service classification can be divided by their applied techniques. Earlier ontology and Web Services Description Language (WSDL) based service classification investigations [13] require strict formatting and manual domain annotation, which can no longer fit the trends as RESTful API services become the mainstream way in service registry and consumption. Inspired by the wide application of machine learning techniques in the field of text classification, researchers apply traditional machine learning algorithms

[1]https://www.programmableweb.com

such as LDA [1], SVM [16] and Native Bayes [17] in service classification. They regard service description as natural language text to extract service features and achieve a relatively high classification accuracy with very few service categories. Recently, deep learning technique is introduced for service classification [12], which can further boost classification accuracy with more service categories.

Although these existing approaches can assist and facilitate Web service classification, they still cannot reach the satisfaction of service providers as well as service consumers. The primary reason is that it has become a difficult and challenging research issue on how to accurately mine service features, as the category and number of services rapidly increase in real-world scenarios. The key challenges of current research on service classification are twofold. On one hand, along with the growth of service categories, the existing approaches cannot make full advantage of the implicit relationships among multi-dimensional information of Web services such as service category and name, which may cause low effectiveness of learning service features. Thus, it fails to ensure the overall accuracy of Web service classification. On the other hand, conventional approaches have not taken into account the imbalance of service distributions. It is observed from large-scale online service management platform that service categories reveal distinct long-tail characteristics. That is, they have paid less attention to Web services affiliated to tail categories, resulting in low classification accuracy on those categories that contain few Web services. Nevertheless, tail services have been widely used in real application scenarios, e.g., it is often necessary to apply multiple tail services when creating a mashup service [8]. In such case, conventional approaches still fall into indiscriminatingly classifying Web services, which not only reduces the classification accuracy of Web services on those tail categories, but also declines the performance of overall service classification, as the number of service categories rapidly increases. To further improve the service classification performance with the consideration of imbalanced service distributions, we mainly concentrate on handling the challenges of more effectively learning implicit service features on both across the service repository and particularly on those tail categories.

To address the above challenges, we propose a novel deep learning framework for long-tail service classification called DeepLTSC, which integrates label attentive convolutional deep neural network (LACNN) with service categories and service feature augmentation (SFA) model. Extending from the traditional convolutional neural network based models [11], [20], [21], LACNN first considers the implicit relationship among multi-dimensional information of Web services, which can provide auxiliary heuristics to intensify the significance of those words within service functionality descriptions and service name most related to service categories. Thus, implicit service feature learning from entire service categories can be effectively improved without any external information. Then, the learned service features are fed into SFA model through meta-learning, which leverages the high-quality features of large-scale Web services in head categories to further reinforce the feature learning for those small number of Web services in tail categories. In this way, DeepLTSC can simultaneously enhance the performance on the overall service classification and specially for those Web services affiliated to tail categories.

To evaluate the effectiveness of our proposed approach DeepLTSC, we conduct extensive experiments on three real-world datasets under long-tail distributions, which consist of a total number of 14,807 Web services and 180 service categories from ProgrammableWeb. The results show that DeepLTSC outperforms state-of-the-art benchmarking approaches in terms of multiple evaluation metrics. The main contributions of this paper are summarized as follows:

- We propose a novel framework DeepLTSC for Web service classification under long-tail distribution. It combines an improved convolutional deep neural network with service category attentive mechanism and service feature augmentation model as a whole, which can not only entirely improve the classification accuracy on all of service categories, but also more accurately classify those tail categories that contain few Web services.
- We propose a novel implicit service feature extraction model LACNN and a service feature augmentation model SFA, respectively. LACNN leverages the implicit relationship among service categories, service name and service functionality descriptions to ensure the utmost feature learning of Web services, while SFA based on meta-learning is designed to further improve feature quality of Web services specially on those tail categories.
- Extensive experiments are conducted on three real-world service datasets under different long-tail distributions. The results demonstrate that DeepLTSC receives superior performance on long-tail service classification.

II. Related Work

With the wide application of machine learning techniques, many researchers have investigated new approaches to classify Web services described in natural language. Wang *et al.* [16] proposed a hierarchical classification system to classify Web services based on support vector machine (SVM). It maps high-dimensional service features to low-dimensional ones through a feature selection model, which is further fed to classifying Web services by SVM. Liu *et al.* [17] proposed a semantic Web service classification approach based on naive bayes. It elaborates the concrete process of how to use the three stages of bayesian classification to classify Web services. However, these traditional machine learning-based service classification approaches can only effectively classify a small number of service categories, resulting in remarkable decrease on classification accuracy as the number of service categories increases. Moreover, there has been correlative studies on service classification using probabilistic topic model such as LDA. Liu *et al.* [1] proposed a service classification model that incorporates LDA and SVM for classification on few number of service categories.

With the advances of deep learning techniques, some researches have taken advantage of deep learning techniques for service classification and text mining. Yang *et al.* [12]

proposed a deep learning framework that can partially solve the issue on service classification accuracy. It applied 2-directional CNN to obtain local relations and 2-directional LSTM to retain global long-term dependencies of service descriptions. Tan *et al.* [18] proposed a dynamic embedding projection-gated convolutional neural network for multi-class text classification, which can control how much context information is incorporated into each specific position of a word-embedding matrix. Shi *et al.* [19] integrated bound optimization theory with variational Bayesian inference to fix overfitting problem in nonlinear soft sensor development. Although advanced deep neural networks have been applied to improve service classification performance, they have not taken into account the characteristics of Web services under long-tail distributions. In such case, it is difficult to ensure the classification accuracy of those Web services on tail categories.

In addition, non-functional characteristics of Web service, are also important for Web service selection and recommendation. Luo *et al.* [6] presents a biased non-negative latent factorization of tensors model for temporal pattern-aware QoS prediction with the consideration of temporal dynamics. Wu *et al.* [22] proposed a data-characteristic-aware latent factor model to predict highly accurate vacant QoS values. Moreover, Wu *et al.* [23] proposed a posterior-neighborhood-regularized LF model for QoS prediction considering information security, identity privacy, and commercial interests in real-world application scenarios. Meanwhile, some recent studies [24], [25] have focused on improving the efficiency of model training, which is beneficial to highlight the performance of QoS prediction. Luo *et al.* [24] investigated eight extended stochastic gradient descent algorithms and proposed eight novel latent factor based models, while they [25] proposed a generalized momentum method based non-negative latent factor model with fast learning performance.

Motivated by the above investigations, we aim at focusing on the issue of Web service classification under long-tail distributions. To solve this problem, we propose a novel deep learning based framework DeepLTSC that can not only raise the overall classification performance by leveraging the advanced label attention mechanism and deep neural network, but also significantly boost the accuracy of classifying Web services on tail categories by service feature augmentation.

III. Problem Formulation

Definition 1 (Web Service): A Web service is defined as a three-tuple $\boldsymbol{s} = <\boldsymbol{L^s}, \boldsymbol{N^s}, \boldsymbol{D^s}>$, where $\boldsymbol{L^s}$ is the service category of $\boldsymbol{s}$. $\boldsymbol{N^s} = \{\boldsymbol{w_{n_1}}, \boldsymbol{w_{n_2}}, \ldots\}$ is the name of $\boldsymbol{s}$, which consists of several words provided from service publisher. $\boldsymbol{D^s} = \{\boldsymbol{w_{d_1}}, \boldsymbol{w_{d_2}}, \ldots\}$ contains a set of words that constitute the functionality description of $\boldsymbol{s}$.

Definition 2 (Web Service Repository): From the registration of service providers, a Web service repository is composed of a set of Web services, denoted as $\mathbb{S} = \{\boldsymbol{s_1}, \boldsymbol{s_2}, \ldots, \boldsymbol{s_n}\}$.

Given a Web service repository $\mathbb{S}$, a service category set $\mathbb{L} = \{\boldsymbol{L^{s_1}}, \boldsymbol{L^{s_2}}, \ldots, \boldsymbol{L^{s_n}}\}$, a service name set $\mathbb{N} = \{\boldsymbol{N^{s_1}}, \boldsymbol{N^{s_2}}, \ldots, \boldsymbol{N^{s_n}}\}$ and a service functionality description set $\mathbb{D} = \{\boldsymbol{D^{s_1}}, \boldsymbol{D^{s_2}} \ldots, \boldsymbol{D^{s_n}}\}$ can be derived from $\mathbb{S}$, respectively. Here, $\mathbb{L} = \{\boldsymbol{L^{s_1}}, \boldsymbol{L^{s_2}}, \ldots, \boldsymbol{L^{s_n}}\}$ can be further denoted as $\mathbb{L} = \{\boldsymbol{L^1}, \boldsymbol{L^2}, \ldots, \boldsymbol{L^m}\}$, where $\boldsymbol{m}$ is the number of the service categories among all of the Web services in $\mathbb{S}$. More specifically, we apply a mapping function $\boldsymbol{f}$ to recognize the category of a Web service, i.e., $\boldsymbol{L^s} = \boldsymbol{f}(\boldsymbol{s})$. In this way, for each service category $\boldsymbol{L^x} \in \mathbb{L}$, its corresponding Web services can be expressed as, $\mathbb{S}_{L^x} = \{\boldsymbol{s_k} | \boldsymbol{f}(\boldsymbol{s_k}) = \boldsymbol{L^x}\}$.

Definition 3 (Head and Tail Service Category): Given a Web service repository, where $\mathbb{S} = \{\boldsymbol{s_1}, \boldsymbol{s_2}, \ldots, \boldsymbol{s_n}\}$ and $\mathbb{L} = \{\boldsymbol{L^1}, \boldsymbol{L^2}, \ldots, \boldsymbol{L^m}\}$. By applying a threshold θ, $\mathbb{L}$ can be partitioned into two subsets, including a set of head service categories $\mathbb{HL} = \{\boldsymbol{L^{h1}}, \boldsymbol{L^{h2}}, \ldots\}$ and tail service categories $\mathbb{TL} = \{\boldsymbol{L^{t1}}, \boldsymbol{L^{t2}}, \ldots\}$.

With regard to a head service category $\boldsymbol{L^{hi}} \in \mathbb{HL}$, the number of Web services in $\mathbb{S}_{L^{hi}}$ is equal to or larger than θ, satisfying $|\mathbb{S}_{L^{hi}}| \geq \theta$; similarly, a tail service category $\boldsymbol{L^{tj}} \in \mathbb{TL}$ has the number of Web services that is less than θ, i.e., $|\mathbb{S}_{L^{tj}}| < \theta$. Generally, most Web services get involved in $\mathbb{HL}$, while only a small portion of Web services in $\mathbb{S}$ belong to $\mathbb{TL}$. That is, it shows obvious characteristics of long-tail distribution in a Web service repository.

Definition 4 (Long-tail Service Classification Problem, LTSC): Give a Web service repository $\mathbb{S}$, an LTSC problem can be defined as a five-tuple *LTSC* = $< \mathbb{S}_{\mathbb{HL}}, \mathbb{S}_{\mathbb{TL}}, \mathbb{L}, \hat{s}, L^{\hat{s}} >$, where

(1) $\mathbb{S}_{\mathbb{HL}}$ is a set of Web services from the head categories $\mathbb{HL} = \{\boldsymbol{L^{h1}}, \boldsymbol{L^{h2}}, \ldots\}$, each of which consists of large-scale number of services in long-tail distribution.
(2) $\mathbb{S}_{\mathbb{TL}}$ is a set of Web services from the tail categories $\mathbb{TL} = \{\boldsymbol{L^{t1}}, \boldsymbol{L^{t2}}, \ldots\}$, each of which consists of very few number of services in long-tail distribution.
(3) $\mathbb{L} = \{\boldsymbol{L^1}, \boldsymbol{L^2}, \ldots\} = \mathbb{HL} \cup \mathbb{TL}$ is the set of all the head and tail service categories from $\mathbb{S}$.
(4) $\hat{\boldsymbol{s}}$ is the Web service to be classified into $\mathbb{HL}$ or $\mathbb{TL}$.
(5) $\boldsymbol{L^{\hat{s}}}$ is the service category predicted by classification model.

IV. The Framework of DeepLTSC

The overall framework of DeepLTSC is illustrated in Fig. 1. It consists of three independent but correlative components, including service semantic representation, long-tail service feature extraction and service classification.

- In the component of service semantic representation, service descriptions and names provided by service providers are fed into a pre-trained word embedding model. After that, they are semantically represented as matrices, respectively. Moreover, service categories obtained from Web service repository are similarly represented as a set of matrices by the same embedding model. These three kinds of service semantic representations are simultaneously taken as inputs for extracting deep and long-tail service features.
- In the component of long-tail service feature extraction, an improved label attentive convolutional neural network with service category attention is trained to generate deep service features for classification task across all the service categories. Then, the generated deep service features

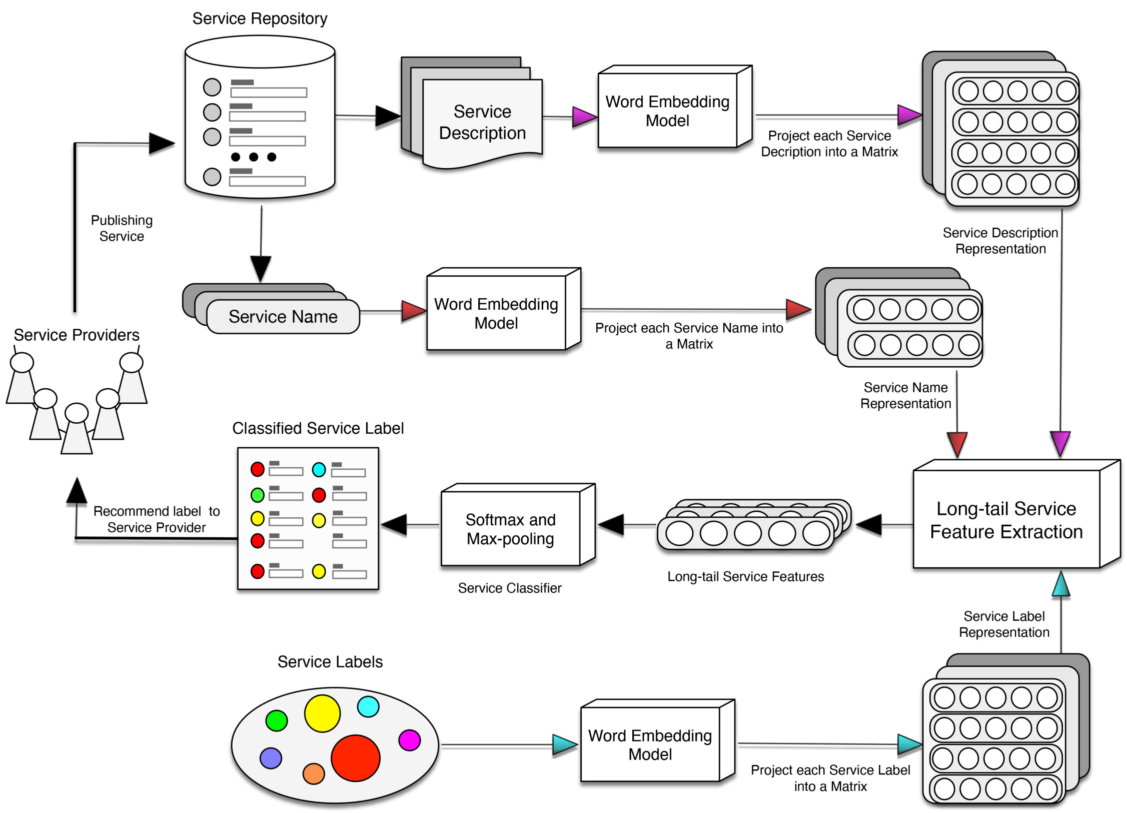

Service Repository
Service Description
Word Embedding Model
Project each Service Decription into a Matrix
Service Description Representation
Publishing Service
Service Name
Word Embedding Model
Project each Service Name into a Matrix
Service Name Representation
Service Providers
Classified Service Label
Long-tail Service Feature Extraction
Softmax and Max-pooling
Long-tail Service Features
Service Classifier
Recommend label to Service Provider
Service Label Representation
Service Labels
Word Embedding Model
Project each Service Label into a Matrix

Fig. 1. Overall framework of DeepLTSC for Web service classification under long-tail distribution.

are further fed into a trained service feature augmentation model that can extract long-tail service features beneficial to classification task for those Web services on tail categories.

- In the component of service classification, we apply softmax to normalize the extracted long-tail service feature, and then a service category can be identified through max-pooling. The recommended category can be used for service providers to effectively and efficiently perform online service registration.

V. Approach

A. Category Attentive Deep Service Feature Extraction

1) Service Category Attention: To initialize the representation of a Web service as inputs, each word $\boldsymbol{w}$ in service description or name is projected into a dense vector $\boldsymbol{r}(\boldsymbol{w}) \in \mathbb{R}^{d}$ through a trained word embedding model, where $\boldsymbol{d}$ is the dimension of the word representation. Similarly, each service category $\boldsymbol{l}$ is also projected into a dense vector in the same embedding space $\boldsymbol{r}(\boldsymbol{l}) \in \mathbb{R}^{d}$. Here, Bidirectional Encoder Representations from Transformers (BERT) [26] is applied as the word embedding model for service semantic representation. It is a deep language model based on bi-directional transformers, which is pre-trained in several large-scale natural language corpora and can adapt to different downstream tasks by parameter fine-tuning. Since BERT is state-of-the-art language model and receives the best performance on multiple natural language processing tasks, it is chosen for service embedding representation with initial feature matrix on service category, service name and service description, when extracting deep service feature by category attentive convolutional deep neural network. In addition, we align the length of every service description with a definite value L_{desc}. If the length of a service description is over L_{desc}, the extra words are pruned from the description. Conversely, zero-padding is applied for the scenario where the length of service description is less than L_{desc}. In the same way, the length of a service name is unified as L_{sname}. Thus, a Web service can be semantically represented as follows:

Definition 5 (Service Embedding Representation): Given a service $\boldsymbol{s} =< \boldsymbol{L^s}, \boldsymbol{N^s}, \boldsymbol{D^s} >$, it is correspondingly represented by a set of semantically embedding matrices, denoted as $< \boldsymbol{W^l}, \boldsymbol{W^n}, \boldsymbol{W^d} >$. $\boldsymbol{W^l}$ is an embedding matrix of service category, where $\boldsymbol{W^l} \in \mathbb{R}^{n_l \times d}$ and $\boldsymbol{n_l}$ is the total number of service categories; $\boldsymbol{W^n}$ is an embedding matrix of service name, where $\boldsymbol{W^n} \in \mathbb{R}^{n_k \times d}, \boldsymbol{n_k} = \boldsymbol{L_{sname}}$; $\boldsymbol{W^d}$ is an embedding matrix of service description, where $\boldsymbol{W^d} \in \mathbb{R}^{n_d \times d}, \boldsymbol{n_d} = \boldsymbol{L_{desc}}$.

To highlight the importance on those key words most related to the service category in service description or service name, label attention mechanism based on service categories is taken to reinforce their effects on service embedding representation. Let $\boldsymbol{S} = \boldsymbol{W^d}$ be the service embedding matrix of service description, the correlation matrix of service description and service category $\boldsymbol{CM} \in \mathbb{R}^{n_d \times n_l}$ is represented as in (1).

$$\boldsymbol{CM} = \frac{\boldsymbol{S}\boldsymbol{L}^{\mathrm{T}}}{\sqrt{\boldsymbol{d_k}}} = \boldsymbol{\omega_1} \oplus \boldsymbol{\omega_2} \oplus \cdots \oplus \boldsymbol{\omega_{n_d}} = \begin{bmatrix} \boldsymbol{\omega_1} \\ \boldsymbol{\omega_2} \\ \vdots \\ \boldsymbol{\omega_{n_d}} \end{bmatrix} \tag{1}$$

where $\oplus$ is concatenation operator that makes n_d row vectors formulate a matrix, $\boldsymbol{L^T}$ is the transpose of vector matrix on service categories $\boldsymbol{L}$, and d_k is the scaling factor, having the same as the dimension of word embedding. $\boldsymbol{CM}$ consists of n_d word-category correlation embeddings $\boldsymbol{\omega}$ on n_l dimension. In such case, max-pooling technique is applied for $\boldsymbol{CM}$ to attain the most related service category of each word as in (2).

$$\begin{aligned}&\boldsymbol{CV} = (\boldsymbol{m_1}, \boldsymbol{m_2}, \ldots, \boldsymbol{m_{n_d}})^{\mathbf{T}}\\&\boldsymbol{m_i} = \mathrm{Max}\Big(\boldsymbol{\omega_i} = \Big\{\boldsymbol{m_i^1}, \boldsymbol{m_i^2}, \ldots, \boldsymbol{m_i^{n_l}}\Big\}\Big), \boldsymbol{\omega_i} \in \boldsymbol{CM} \qquad (2)\end{aligned}$$

where $\boldsymbol{CV}$ is n_d dimensional vector. We further normalize $\boldsymbol{CV}$ by softmax function to generate service category attention score on service description $\boldsymbol{SA}$ as in (3).

$$\begin{aligned}&\boldsymbol{SA} = (\boldsymbol{SA_1}, \boldsymbol{SA_2}, \ldots, \boldsymbol{SA_{n_d}})\\&\boldsymbol{SA_i} = \frac{\exp(\boldsymbol{m_i})}{\sum_j \exp(\boldsymbol{m_j})}, \boldsymbol{m_i} \in \boldsymbol{CV} \qquad (3)\end{aligned}$$

By applying the distribution score in $\boldsymbol{SA}$ as importance weighting to service description embedding matrix $\boldsymbol{S}$, category attentive embedding matrix of service description $\boldsymbol{S_a^d}$ is calculated as in (4).

$$\boldsymbol{S_a^d} = \boldsymbol{SA} \cdot \boldsymbol{S} \qquad (4)$$

In the same way, service category attention score on each word in service name can be generated by repeating above steps. Thus, by applying the score distribution of as importance weighting to service name embedding matrix, we attain category attentive embedding matrix of service name $\boldsymbol{S_a^n}$.

2) Deep Service Feature Extraction: Based on the category attentive service embedding matrix, we further extract the implicit neighborhood features from the words that are near the most relevant words to service category, as they are more valuable to classification task than words in other locations in service description. Here, TextCNN model [20] is applied to extract the neighborhood features in $\boldsymbol{S_a^d}$. Let $\boldsymbol{x_i} \in \mathbb{R}^d$ be the *i*-th category attentive word vector, and thus $\boldsymbol{S_a^d}$ can be represented as in (5).

$$\boldsymbol{S_a^d} = [\boldsymbol{x_1} \oplus \boldsymbol{x_2} \oplus \cdots \oplus \boldsymbol{x_{n_d}}] \qquad (5)$$

where n_d denotes the number of words in service description. We first perform convolution operation with a $\boldsymbol{filter} \in \mathbb{R}^{h\times d}$, which uses a window of *h* words to acquire a new feature. The size of filter *h* and *d* is a hyperparameter of experiment and the dimension of word embedding, respectively. Given a window matrix of words $[\boldsymbol{x_i} \oplus \boldsymbol{x_{i+1}} \oplus \cdots \oplus \boldsymbol{x_{i+h-1}}]$, a new feature $\boldsymbol{c_i}$ produced by the filter can be represented as in (6).

$$\boldsymbol{c_i} = \boldsymbol{ReLU}(\boldsymbol{w} \cdot [\boldsymbol{x_i} \oplus \boldsymbol{x_{i+1}} \oplus \cdots \oplus \boldsymbol{x_{i+h-1}}] + \boldsymbol{b}) \qquad (6)$$

Then, after several rounds of convolutions by moving the filter window on category attentive service embedding matrix $\boldsymbol{S_a^d}$, which shifts from $\boldsymbol{x_{1\,:\,h}}$ to $\boldsymbol{x_{n_d-h+1\,:\,n_d}}$, a set of features can be obtained as in (7).

$$\boldsymbol{c} = (\boldsymbol{c_1}, \boldsymbol{c_2}, \ldots, \boldsymbol{c_{n-h+1}}) \qquad (7)$$

After that, max-overtime-pooling operation is used to get the maximum value $\hat{c} = \mathrm{Max}(c)$ as the neighborhood feature extracted by the filter. After leveraging a predefined number of filters with different size of sliding windows, category attentive service embedding matrix is transformed into a feature vector of service description $\boldsymbol{f_d}$. It is concatenated by a set of maximum neighborhood features as in (7), where *k* is the number of filters.

$$\boldsymbol{f_d} = (\boldsymbol{\hat{c_1}}, \boldsymbol{\hat{c_2}}, \ldots, \boldsymbol{\hat{c_k}}) \qquad (8)$$

As for the category attentive embedding matrix of service name, since it generally originates from very few words, it is difficult to extract implicit neighborhood features from service name. In such case, max-pooling is performed to extract the maximum value in each dimension from category attentive embedding matrix of service name, which is formulated as $\boldsymbol{f_n}$ and calculated as in (9).

$$\boldsymbol{f_n} = \textbf{max-pooling}(\boldsymbol{S_a^n}) \qquad (9)$$

Finally, we combine service description feature vector $\boldsymbol{f_d}$ with service name feature vector $\boldsymbol{f_n}$ to generate a comprehensive vector of deep service feature $\boldsymbol{V}$, which is attained by using hyperparameter r_n to adjust the proportion of $\boldsymbol{f_n}$ and $\boldsymbol{f_d}$. It is represented as in (10).

$$\boldsymbol{V} = \boldsymbol{f_d} + \boldsymbol{r_n} \cdot \boldsymbol{f_n}. \qquad (10)$$

B. Long-Tail Service Feature Augmentation

In order to apply the characteristics of service classifier learned from Web services on head categories to the tail service classification task, we present a meta-learning service feature augmentation model based on [27]. Meta-learning is expected to transform the capability of the trained model in certain domains into the tasks in other similar application domains, which indicates the trained model has the ability to learn how to learn. For the task of long-tail service classification, it can be divided into classifying Web services on the head categories and tail ones, respectively. According to the meta-learning theory, it can acquire effective classification knowledge from learning the implicit patterns on head service categories, which can be used to strengthen the learning ability and improve the classification performance on tail service categories. By utilizing a deep service feature generated from previous layer, SFA can further optimize service characteristics by differentiating the head and tail categories, and output an augmented long-tail service feature for more accurate classification. It consists of three components, including central feature representation of service categories, relevance distribution of service categories, and head-tail discrimination of service categories.

First, central feature representation of service categories can be initialized by deep service features. Let $\boldsymbol{v_i}$ be the central feature of *i*-th service category, which is originally calculated by the mean of all the deep service features affiliated to that category. Thus, central feature representation of service categories can be represented as in (11).

$$\boldsymbol{C_s} = [\boldsymbol{v_1} \oplus \boldsymbol{v_2} \oplus \cdots \oplus \boldsymbol{v_{n_l}}], \boldsymbol{v_i} = \frac{\sum_{k=1}^{n} \boldsymbol{V_k}}{\boldsymbol{n}} \qquad (11)$$

where n_l is the number of service categories, *n* is the number of Web services involved in the *i*-th service category, and $\boldsymbol{V_k}$

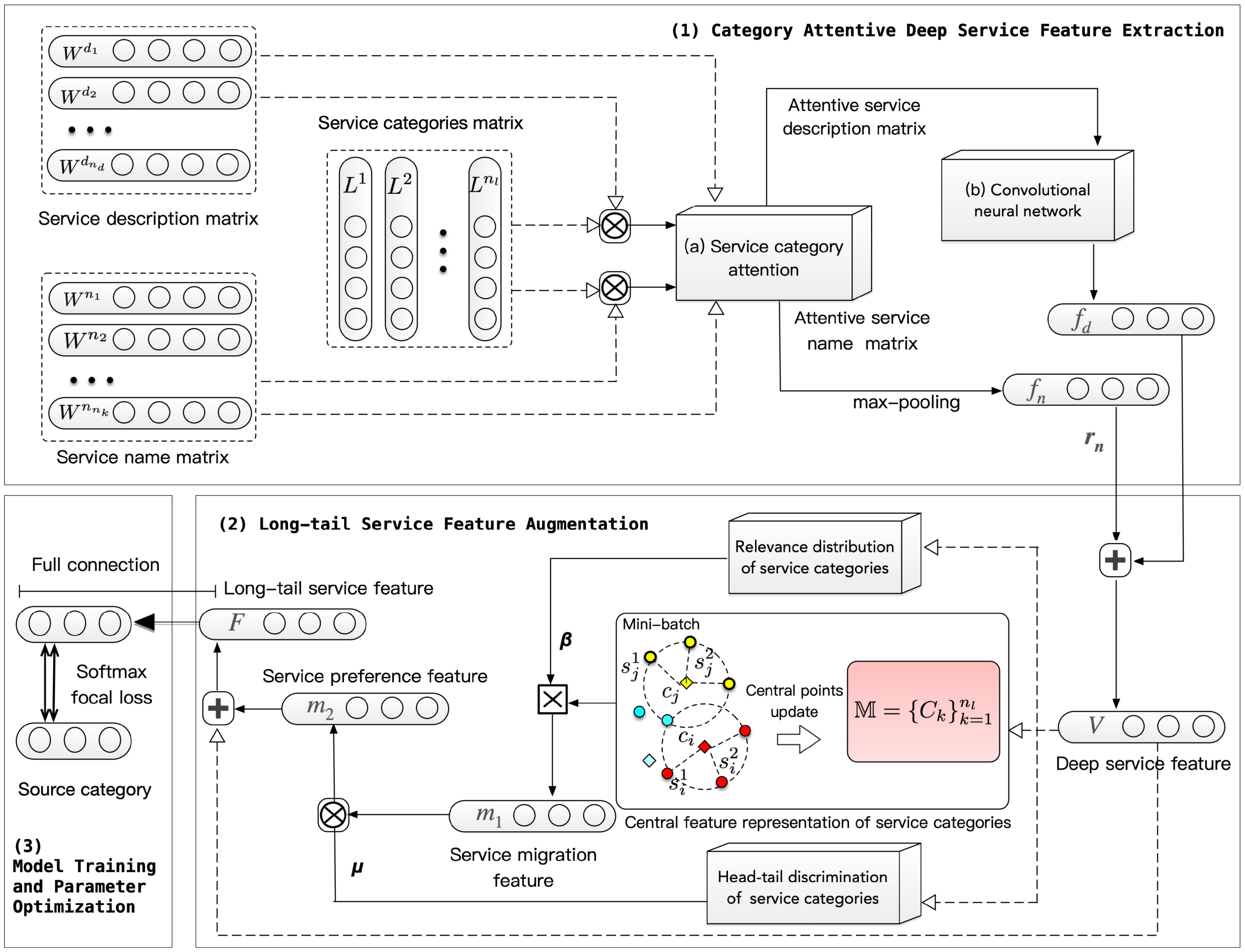

(1) Category Attentive Deep Service Feature Extraction
Service description matrix
Service categories matrix
Service name matrix
Attentive service description matrix
(b) Convolutional neural network
(a) Service category attention
Attentive service name matrix
max–pooling
r_n
(2) Long-tail Service Feature Augmentation
Full connection
Long–tail service feature
Relevance distribution of service categories
β
Mini–batch
Central points update
$\mathbb{M} = \{C_k\}_{k=1}^{n_l}$
Softmax focal loss
Service preference feature
Deep service feature
Source category
Central feature representation of service categories
Service migration feature
μ
Head-tail discrimination of service categories
(3) Model Training and Parameter Optimization

Fig. 2. Multi-layer training process of long-tail service feature extraction in DeepLTSC.

denotes a deep service feature affiliated to the *i*-th service category.

To effectively discriminate the representation of head and tail service categories, central feature of each service category is updated by model training as below. When a mini-batch Web services across multiple categories are used for model training, they are partitioned as a set of groups by their affiliated service categories and a central feature of each group is calculated by deep service features. The corresponding category representations in $\boldsymbol{C_s}$ are replaced by the newly calculated central features in mini-batch. Moreover, when calculating the central feature of a service group, we alternatively update deep service features, in order to minimize the distance between every deep servie feature and its corresponding central feature and also maximize the distance to central features of other service categories.

Then, given a deep service feature $\boldsymbol{V}$, we use the relevance distribution of service categories, which is a fully connected lightweight neural network to generate the correlation distribution of $\boldsymbol{V}$ on different service categories. We apply softmax function to output a normalized relevance vector of service categories $\boldsymbol{\beta} \in \mathbb{R}^{n_l}$, which is represented as in (12).

$$\boldsymbol{\beta} = \textbf{softmax}(\boldsymbol{w_1} \cdot \boldsymbol{V} + \boldsymbol{b_1}) \tag{12}$$

where w_1 and b_1 are parameters to be learned from model training. After acquiring $\boldsymbol{\beta}$, it is used as an importance factor multiplied with the central feature representation of service categories $\boldsymbol{C_s}$, which calculates a service migration feature $\boldsymbol{m_1}$ as in (13).

$$\boldsymbol{m_1} = \boldsymbol{\beta}^{\boldsymbol{T}} \boldsymbol{C_s}, \forall \boldsymbol{m_1^j} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n_l}} \boldsymbol{\beta_i^T} \boldsymbol{C_{s_i}^j} \tag{13}$$

Finally, to more favor the tail categories, we further train the head-tail discrimination of service categories, which is composed of a fully connected lightweight neural network. As a result, given a deep service feature, the output of head-tail discrimination of service categories is fed into *tanh* as activation function to derive an adjusting factor μ as in (14).

$$\boldsymbol{\mu} = \boldsymbol{tanh}(\boldsymbol{w_2} \cdot \boldsymbol{V} + \boldsymbol{b_2}) \tag{14}$$

where w_2 and b_2 are parameters to be learned from model training, $\boldsymbol{\mu}$ reflects the model preference on tail service categories. After receiving $\boldsymbol{\mu}$, we multiply service migration feature $\boldsymbol{m_1}$ with adjusting factor $\boldsymbol{\mu}$ to gain a service preference feature $\boldsymbol{m_2}$. In such way, long-tail service feature $\boldsymbol{F}$ is obtained by integrating the service preference feature $\boldsymbol{m_2}$ into the deep service feature $\boldsymbol{V}$ as in (15, where $\otimes$ means element-wise multiplication.

$$\boldsymbol{F} = \boldsymbol{V} + \boldsymbol{\mu} \otimes \boldsymbol{m_1}. \tag{15}$$

C. Model Training and Parameter Optimization

To boost the overall classification accuracy on both head and tail service categories, Large-Margin Softmax Loss [28] is used to maximize the distance of central features across multiple service categories and minimize the distance among deep service features within the same service category. Moreover, Focal Loss [29] is also used to train DeepLTSC, ensuring that it can receive better classification performance on tail service categories.

Since Large-Margin Softmax Loss is capable of differentiating learned features, it can effectively guarantee the compactness of inner class and intensify the separability of inter-classes by adding a parameter *m* to enlarge the margin of similar service categories. Thus, it is desired to make central feature representation of service categories and long-tail service features more sparse in feature space, which is beneficial to differentiating head and tail service categories. It is represented as in (16) and (17), respectively.

$$\boldsymbol{LM_i} = -\log\left(\frac{e^{\left\|W_{y_i}\right\|\|x_i\|\psi(\theta_{y_i})}}{e^{\left\|W_{y_i}\right\|\|x_i\|\psi(\theta_{y_i})} + \sum_{j\neq y_i} e^{\|W_j\|\|x_i\|\cos(\theta_j)}}\right) \tag{16}$$

$$\boldsymbol{\psi(\theta)} = \begin{cases} \cos(m\theta), & 0 \leq \theta \leq \frac{\pi}{m} \\ \mathcal{D}(\theta), & \frac{\pi}{m} < \theta \leq \pi \end{cases} \tag{17}$$

where LM_i is the training loss of the *i*-th service in training set, *e* is the natural logarithm, y_i is the original service category, *W* is the parameter to be learned, x_i is the feature of the *i*-th service, θ_{y_i} behaves the vector angle between the classification boundary and the *i*-th service feature, and $\|\|$ denotes modulo operation. $\psi(\theta)$ is a function to calculate the cosine value of θ and $\mathcal{D}(\theta)$ is a monotonically decreasing function.

Focal loss is an improved version of traditional cross-entropy loss function, which can boost the model training quality of our long-tail service classification problem. The primary reason is that, those tail categories with very few Web services are partially omitted in model learning, leading to the decline in tail service classification. To solve this issue, focal loss is applied to particularly favor those tail service categories in model training of DeepLTSC, which decreases the weighting of head services and focuses on learning to distinguish tail services. It is represented as in (18) and (19).

$$\mathrm{FL}(p_\mathrm{t}) = -\alpha(1-p_\mathrm{t})^\gamma \log(p_\mathrm{t}) \tag{18}$$

$$p_\mathrm{t} = \begin{cases} p, & \text{if } y = 1 \\ 1-p, & \text{otherwise} \end{cases} \tag{19}$$

where *p* is the output of the task layer, and given the original category *y*. Besides, α and γ are parameters of focal loss and they are used to adjust the attention ratio on tail service categories with different combinations of α and γ. Since the formal focal loss function is mainly appropriate for binary classification, we fix the function in the multi-class long-tail service classification task by scaling the outputs from the task layer, where the service category probabilities of each sample sum to 1, as in (20).

$$FL^{'} = \sum_{i=1}^{n_l} -\alpha(1-p_i)^\gamma y_i log(p_i) \tag{20}$$

Based on the above applied functions, the overall loss function of our model favored on tail service categories is represented as in (21), which acquires more sparse service central feature $\boldsymbol{C_s}$.

$$L = \sum_{n=1}^{N} FL^{'}(\boldsymbol{F_n}) + \lambda \cdot LM(\boldsymbol{F_n}, \boldsymbol{C_s}) \tag{21}$$

where $\boldsymbol{F_n}$ is the long-tail service feature extracted from the *n*-th service, and λ is hyperparameter used to adjust the ratio of training loss and central feature loss.

D. Time Computational Complexity Analysis

The time computational complexity of DeepLTSC is mainly composed by service category attention, deep service feature extraction and service feature augmentation. First, the time consumption of service category attention is determined by the size of the service description matrix $\boldsymbol{W^d}$, service name matrix $\boldsymbol{W^k}$ and service category matrix $\boldsymbol{W^l}$. Taking $\boldsymbol{W^d}$ as an example, the calculation of service category attention consumes $O(n_l \cdot n_d \cdot d_k + n_d \cdot n_l + n_d + n_d \cdot d_k)$, which is reduced to $O(n_l \cdot n_d \cdot d_k)$. Then, the cost of extracting deep service feature by convolutional neural network is expressed by $O(\sum_{s_k=s_1}^{s_n} C_{out} \cdot ((n_d - s_k + 1) \cdot (s_k \cdot d_k) + (n_d - s_k + 1)))$, where s_i behaves a kernel size. It can be further reduced to $O(C_{out} \cdot n_d \cdot s_k \cdot d_k)$, where C_{out} is the size of output kernel and s_k is the size of convolution kernel. Finally, the time complexity of service feature augmentation is primarily influenced by the two fully connected neural networks, including relevance distribution of service categories and head-tail discrimination of service categories. Thus, it is calculated by $O((2 \cdot d_k - 1) \cdot n_l + n_l \cdot d_k + d_k)$, which can be represented as $O(d_k \cdot n_l)$.

From the above analysis, it is observed that the time computational complexity of DeepLTSC is $O(n_l \cdot n_d \cdot d_k) + O(C_{out} \cdot n_d \cdot s_k \cdot d_k) + O(d_k \cdot n_l)$, which depends mainly on the dimensionality of the word embedding, the length of the service description or service name, and the number of service categories to be classified. In real-world application scenarios, it generally satisfies that the dimensionality of word embedding is much larger than the length of service description and the number of service categories. Thus, the computational complexity of long-tail service classification can be performed in polynomial time that is highly correlative to the dimensionality of word embedding.

VI. Experiments

A. Experimental Datasets and Setup

To validate the performance of DeepLTSC, we have designed a Web crawler and obtained Web services from ProgrammableWeb until July 1, 2018. The crawled service repository as the experimental dataset contains 17,923 real-world Web services distributed in 384 service categories,

TABLE I
STATISTICS OF *PW*-120 DATASET

Item Name	Value	Item Name	Value
Number of Services	11775	Number of Categories	120
Head Services	10497	Number of Head Categories	30
Tail Services	1278	Number of Tail Categories	90

TABLE II
STATISTICS OF *PW*-150 DATASET

Item Name	Value	Item Name	Value
Number of Services	16425	Number of Categories	150
Head Services	10497	Number of Head Categories	30
Tail Services	5928	Number of Tail Categories	120

TABLE III
STATISTICS OF *PW*-180 DATASET

Item Name	Value	Item Name	Value
Number of Services	14807	Number of Categories	180
Head Services	8595	Number of Head Categories	25
Tail Services	6212	Number of Tail Categories	155

which is available on our Lab.[2] Besides, the number of services in each category is extremely uneven, e.g., the category *Tools* contains 887 Web services while *Solar* only has two Web services. In order to maintain characteristics of Web services under the long-tail distribution, we select those Web services in the top 200 service categories as the experimental dataset.

More specifically, we have selected 120, 150 and 180 service categories from the total number of 200 service categories to build three experimental datasets with different long-tail distributions, including *PW*-120, *PW*-150 and *PW*-180. The detailed statistics of these three experimental datasets are shown in Tables I, II, and III. In the experiments, we randomly divide each service dataset into training set, validation set and test set by the ratio of 6:2:2.

In the experiments, we use BERT as the word embedding model, for service description, name and category. Generally, there are two kinds of BERT models in application scenarios, including BERT-base (L = 12, H = 768, A = 12, total parameters = 110M) and BERT-large (L = 24, H = 1024, A = 16, total parameters = 340M), where the number of layers is denoted as L, the size of the hidden layer is denoted as H, and the number of self-attentive heads is A. Considering the characteristics of short text of the service description and reducing the time consumption of model training, we choose BERT-base model (L = 12, H = 768, A = 12) as the word embedding model for long-tail service classification.

B. Evaluation Metrics

We evaluate the classification performance of DeepLTSC by two widely-used evaluation metrics: classification accuracy and F$_1$-score. The overall classification accuracy of DeepLTSC is measured as in (22), where $\mathbb{L}$ is the set of all service categories, and *n* is the number of all service categories in the dataset.

$$ACC_O = \frac{\sum_{sc \in \mathbb{L}} ACC(sc)}{n} \tag{22}$$

To further evaluate the classification performance of DeepLTSC on head and tail service categories, we respectively measure ACC_H and ACC_T for the classification accuracy on head and tail service categories as in (23) and (24), where $\mathbb{HL}$ and $\mathbb{TL}$ are the set of all head and tail service categories, and n_h and n_t correspond to the number of head and tail service categories in the dataset.

$$ACC_H = \frac{\sum_{hc \in \mathbb{HL}} ACC(hc)}{n_h} \tag{23}$$

$$ACC_T = \frac{\sum_{tc \in \mathbb{TL}} ACC(tc)}{n_t} \tag{24}$$

Here, F$_1$-score is used to evaluate the classification balance of different service classifiers on head and tail categories. It is calculated as in (27), which is measured by Recall as in (25) and Precision as in (26), where *TP* and *FN* denote the number of services assigned to head and tail service categories, respectively.

$$Recall = \frac{TP}{TP + FN} \tag{25}$$

$$Precision = \frac{TP}{TP + FP} \tag{26}$$

$$F_1\text{-}score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{27}$$

In our experiments, all the four evaluation metrics are real numbers ranged in [0, 1]. Higher values indicate the higher classification performance.

C. Competing Methods

To demonstrate the effectiveness of DeepLTSC, we compare it with eight state-of-the-art service classification approaches and two our self-developed variants. They are detailedly described as below.

- *LSTM [30]:* It is an improved approach of Recurrent Neural Network (RNN). It extracts a service description as a word sequence, where long-term dependency relationships are captured for improving the service classification accuracy.
- *LEAM [31]:* It attempts to apply label in an attention framework for text classification. During the feature extraction, it measures the compatibility between text description and its corresponding label, which is further leveraged to produce an attention score for service description sequences.
- *Bi-LSTM [32]:* It is a variant of LSTM, considering bidirectional long-term dependency relationships of the text sequence. Thus, it can extract comprehensive semantic information for service classification.
- *GRU [33]:* It simplifies the neural structure of LSTM by reducing one cell gate, leading to better performance of service classification with lower time complexity.
- *FastText [34]:* It is proposed by Facebook to achieve an extraordinary service classification performance in terms of time consumption through linear feature combination.

[2]https://scdm-shu.github.io/papers/datasets/PW-Dataset.zip

TABLE IV
PERFORMANCE COMPARISONS OF WEB SERVICE CLASSIFICATION ON THREE LARGE-SCALE DATASETS WITH LONG-TAIL DISTRIBUTIONS AMONG COMPETING APPROACHES

Methods	**PW_120**				**PW_150**				**PW_180**			
	$\mathrm{ACC_O}$	$\mathrm{ACC_H}$	$\mathrm{ACC_T}$	$\mathrm{F_1}$-score	$\mathrm{ACC_O}$	$\mathrm{ACC_H}$	$\mathrm{ACC_T}$	$\mathrm{F_1}$-score	$\mathrm{ACC_O}$	$\mathrm{ACC_H}$	$\mathrm{ACC_T}$	$\mathrm{F_1}$-score
LSTM	0.409	0.593	0.069	0.141	0.442	0.552	0.129	0.207	0.394	0.529	0.083	0.137
LEAM	0.48	0.642	0.186	0.274	0.453	0.572	0.169	0.253	0.453	0.589	0.179	0.260
Bi-LSTM	0.499	0.619	0.257	0.330	0.458	0.560	0.226	0.309	0.458	0.567	0.204	0.271
GRU	0.525	0.645	0.295	0.362	0.490	0.569	0.274	0.347	0.477	0.581	0.249	0.306
FastText	0.516	0.614	0.306	0.376	0.496	0.587	0.279	0.354	0.501	0.593	0.280	0.335
TextRCNN	0.535	0.673	0.217	0.333	0.513	0.638	0.207	0.301	0.519	0.650	0.191	0.266
TextCNN	0.567	0.699	0.294	0.340	0.522	0.643	0.228	0.337	0.525	0.653	0.227	0.309
ServeNet	0.535	0.658	0.285	0.351	0.491	0.590	0.252	0.331	0.497	0.610	0.252	0.318
DeepLTSC (LACNN)	**0.603**	**0.700**	0.424	0.466	**0.581**	**0.669**	0.404	0.453	0.562	**0.650**	0.372	0.408
DeepLTSC (LACNN, SFA)	0.598	0.676	0.482	0.495	0.558	0.642	0.446	0.461	0.560	0.608	0.497	**0.462**
DeepLTSC (LACNN, SFA, Focal)	0.593	0.668	**0.496**	**0.503**	0.558	0.641	**0.454**	**0.464**	0.559	0.604	**0.501**	0.460

- *RCNN [35]:* By fully utilizing the advantages of CNN and recurrent neural network, it comprehensively extracts high-level semantic feature of service description, which increases the performance of service classification.
- *TextCNN [20]:* It adopts different size of convolution kernels to extract key points from different neighbor areas of service description sequences. Then, max-pooling technique is applied to attain the most representative high dimension feature for service classification.
- *ServeNet [12]:* It is state-of-the-art approach in service classification due to the combination of two deep neural networks. By integrating 2D convolutional operation and bi-directional LSTM for service feature extraction, it can receive better service classification effectiveness.
- *DeepLTSC (LACNN):* It is one of our self-developed variant approaches. Here, only category attentive CNN is applied to extract deep service feature for service classification.
- *DeepLTSC (LACNN, SFA):* It is another one of our self-developed variant approaches. Based on the previous variant, we first utilize LACNN to extract deep service feature that is fed into SFA model. During the model training, cross-entrofy function is used as loss function to learn and optimize the parameters.
- *DeepLTSC (LACNN, SFA, Focal):* It is our main proposed approach for service classification, we apply focal loss [29] as objective function to train DeepLTSC model for better classification performance on both overall Web services and those on tail categories.

D. Experiment Results and Analyses

In this section, extensive experiments on three large-scale real-world service datasets with long-tail distributions are conducted to validate the effectiveness of DeepLTSC.

1) Long-Tail Service Classification Comparison: Table IV shows the performance comparisons of Web service classification on three large-scale real-world service datasets with long-tail distributions among competing approaches. It is observed from the experiment results, DeepLTSC (LACNN, SFA, Focal) achieves the best service classification performance across four evaluation metrics, when comparing with eight state-of-the-art methods. When our approach obtains significant improvement over competing ones on those tail categories, it also receives better classification accuracy on head categories. The main reason is that DeepLTSC (LACNN, SFA, Focal) takes full advantage of category attentive mechanism that keeps track of the most relevant words to the category from service description and service name, which can boost the classification accuracy on overall service categories. Besides, compared with the conventional approaches, the classification accuracy on tail categories of DeepLTSC (LACNN, SFA, Focal) does not sharply fluctuate as the increasing number of service categories, which demonstrates the effectiveness of our approach for tail service categories classification in different long-tail distributions. The underlying reason is that DeepLTSC (LACNN, SFA, Focal) leverages service feature augmentation model to minimize the impact of low-quality and confusing service descriptions from tail categories, which is beneficial to preferably extracting service features on tail categories.

More specifically, taking the service dataset *PW*-120 as an example, ServeNet as the best one among the existing approaches, DeepLTSC (LACNN, SFA, Focal) outperforms it by 10.84% and 74.04% in terms of classification accuracy on overall and tail categories, respectively. Therefore, it confirms that our proposed approach can more effectively handle the task of service classification on those tail categories than traditional ones. Moreover, DeepLTSC (LACNN, SFA, Focal) receives superior performance on service classification by 43.30% on F_1-score compared with ServeNet, which demonstrates that it can better ensure the balance of service classification on head and tail categories than those conventional approaches.

To evaluate the effectiveness of different components in our approach, we remove service feature augmentation model from DeepLTSC (LACNN, SFA, Focal) and obtain a self-developed variant one named DeepLTSC (LACNN), which applies service category attention mechanism and convolutional neural network to generate deep service features for performing classification task. As shown in Table IV, DeepLTSC (LACNN, SFA, Focal) is superior to its variant DeepLTSC (LACNN) on ACC_T with an advantage of 16.98%, 12.38% and 34.68% in *PW*-120, *PW*-150 and *PW*-180, respectively. That is, DeepLTSC (LACNN, SFA, Focal) has an average 21.35% improvement on classification accuracy of tail categories, compared with its variation DeepLTSC (LACNN). Therefore, service feature augmentation based on meta-learning is effective for classifying those Web services on tail categories. From the experiment results, we also find that SFA negatively affects service classification performance on head categories. The reason is that unlike those tail categories that utilize valid knowledge from Web services on head categories by SFA to improve service classification accuracy, most of head categories include large number of Web services that are already sufficient for model training and fitting. Therefore, the learned low quality knowledge from few Web services on tail categories is more like noise and harmful to service classification on head categories.

To further validate the effectiveness of focal loss, we compare DeepLTSC (LACNN, SFA, Focal) with our another self-developed variant named DeepLTSC (LACNN, SFA), where traditional cross-entropy loss is applied instead of focal loss function. From the experiment results, the overall classification performance of DeepLTSC (LACNN, SFA, Focal) is better than that of DeepLTSC (LACNN, SFA), demonstrating that focal loss function is more applicable for long-tail service classification than the traditional cross-entropy loss function.

From the above results and analyses, we conclude that by leveraging category attentive mechanism and service feature augmentation with focal loss, DeepLTSC outperforms the competing methods for long-tail service classification on multiple evaluation metrics.

2) Analysis of Scatter Diagrams: In order to visually demonstrate the superiority of DeepLTSC, we transform the extracted service features from different competing approaches to 2-dimensional embeddings by t-SNE [36]. In the experiments, the transformation of service features is performed on 15 service categories in *PW*-120, which contains 12 head and 3 tail service categories.

The experiment results of the generated scatter diagrams are shown in Fig. 3, including four competing approaches LEAM, RCNN, TextCNN, ServeNet and two our self-developed approaches DeepLTSC (LACNN, SFA, Focal) and DeepLTSC (LACNN). By visualizing the results of DeepLTSC and the above competing approaches, it can illustrate the rationality on the overall structure and validate the effectiveness of each part of DeepLTSC. The reason of why we choose these approaches can be explained as follows. Since the same label attention mechanism is applied in LEAM model, it is compared with LACNN to verify the rationality of combining label attention mechanism and convolutional neural network. Simultaneously, to exclude the possibility that the effectiveness of LACNN mainly comes from convolutional neural network, we compare it with two classical implementations of convolutional neural networks without label attention in the field of text classification, including TextCNN and RCNN. In such case, we can further reveal the functionality of service category attention during long-tail service classification. Furthermore, ServeNet is current service classification model with the best performance, which is compared with DeepLTSC to demonstrate the effectiveness and applicability of our proposed approach in real-world scenarios. To verify the impact of SFA on long-tail service classification, we also compare our two self-developed variants, including DeepLTSC (LACNN) with only LACNN model and DeepLTSC (LACNN, SFA, Focal) with both LACNN and SFA models.

As illustrated in the scatter diagrams, our proposed approach DeepLTSC (LACNN, SFA, Focal) achieves the best 2-D visualization results among different service categories that have relatively far distances with each other in Fig. 3(f). Specifically, DeepLTSC (LACNN, SFA, Focal) enables the transformed 2-D service features to be more concentrated within the same service category, which reduces the possibility of those Web services affiliated to tail categories being mixed with head services. Thus, DeepLTSC (LACNN, SFA, Focal) can help service classifier easily recognize the significantly different features for more accurate long-tail service classification. Moreover, the 2-D visualization result of Fig. 3(e) is better than that of Fig. 3(a), (b), (c), (d) by smaller distances among intra-classes and larger distances among inter-classes, which demonstrates the rationality and effectiveness of our proposed deep service feature extraction model LACNN in DeepLTSC.

From the experiment results of our self-developed variant approach DeepLTSC (LACNN) in Fig. 3(e), it is observed that 2-D service features transformed from the same category also keep more concentrated than those generated by the conventional approaches, although the distances among different categories are much closer than those by DeepLTSC (LACNN, SFA, Focal) in Fig. 3(f). This phenomenon can be explained by the applicability of category attentive mechanism that makes the service features from the same category more similar, leading to closer gathering by shorter distances. Thus, it can help service classifier more easily distinguish tail services from head ones. Taking tail category *Medicine* as an example, 2-D service features from that category colored with orange points are mixed with those from head categories by conventional approaches as shown in Fig. 3(a)–(d). Conversely, they are closely gathered together as shown in Fig. 3(e), where all the orange points from *Medicine* are circled for better visualization.

To validate the effectiveness of service feature augmentation model in tail category classification, DeepLTSC (LACNN, SFA, Focal), by fully combining the advantages of LACNN

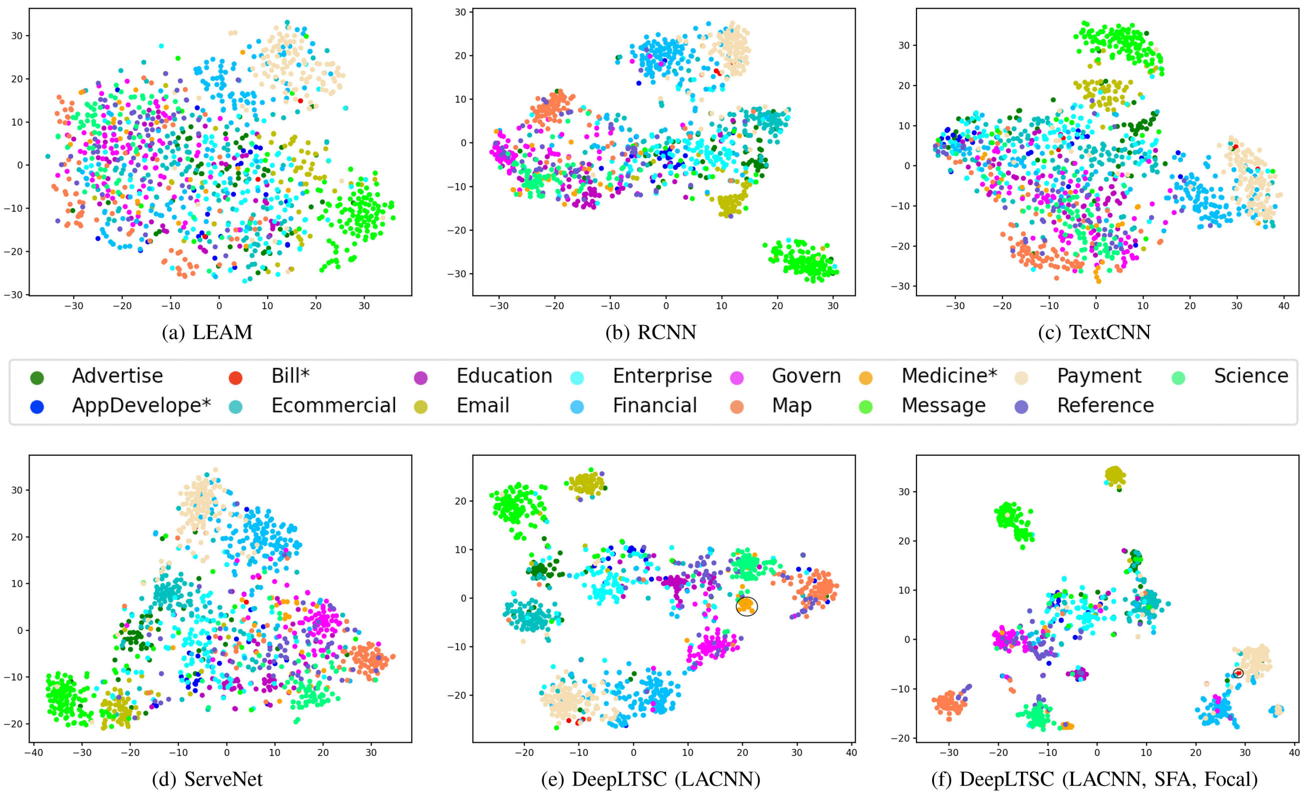


Fig. 3. 2-dimensional service classification scatter diagrams where Web service features are extracted by LEAM (a), RCNN (b), TextCNN (c), ServeNet (d), DeepLTSC (LACNN) (e), and DeepLTSC (LACNN, SFA, Focal) (f), respectively. Each point represents a Web service, and those points sharing the same color represent that their corresponding Web services are classified into the same service category. Among the selected service categories, the three tail categories are marked with * in the legend.

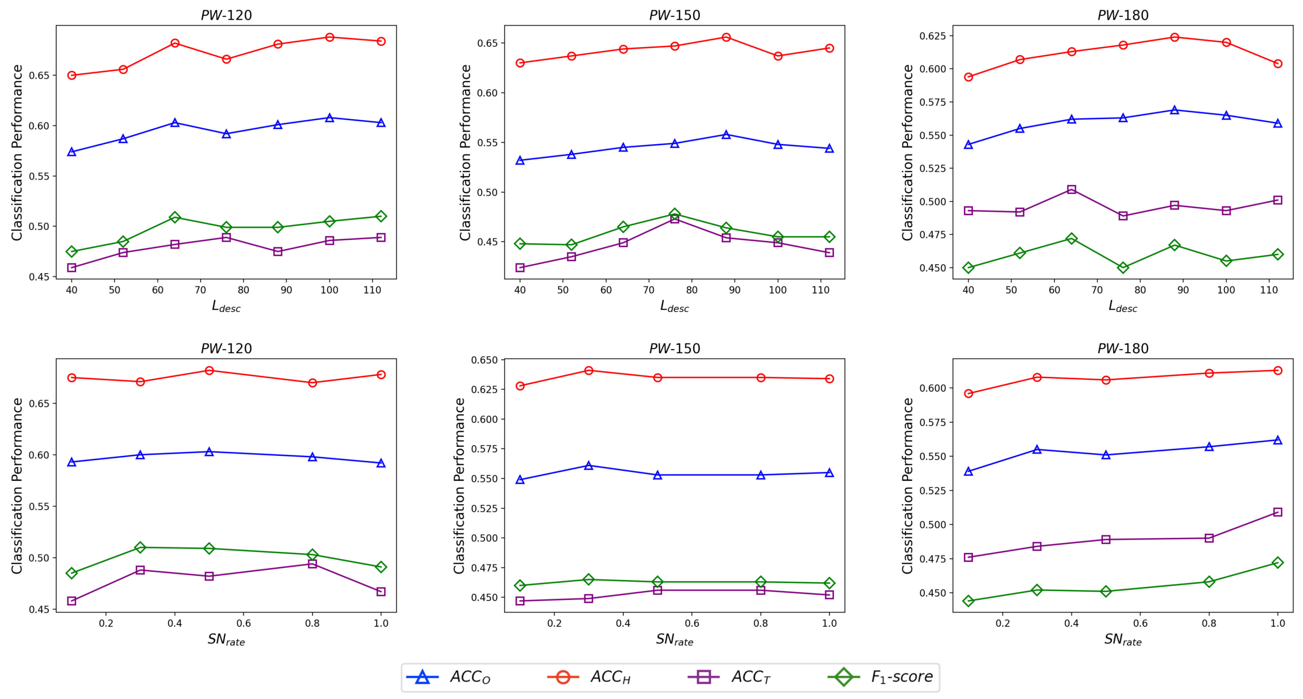


Fig. 4. Classification performance of DeepLTSC in three datasets with the changes of L_{desc} and SN_{rate}.

and SFA models, can extract service features from different categories that have greater inter-class distance but smaller intra-class distance. That can avoid the drawback of our variant approach DeepLTSC (LACNN) and is more conductive to service classification on tail categories. Taking tail category *Bill* as an example, service features extracted from category *Bill* colored with red points have much clearer boundary against that from category *Payment* colored with wheat

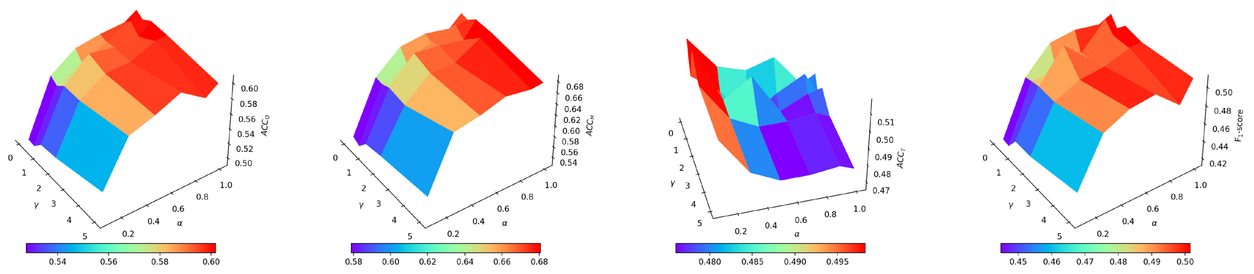


Fig. 5. Classification performance of DeepLTSC on multiple evaluation metrics with different combinations of α and γ.

points, which are circled and shown in bottom right corner of Fig. 3(f). Therefore, our main approach DeepLTSC (LACNN, SFA, Focal) enables to extract differentiated service features that can be effectively recognized for Web service classification among both head and tail categories, while our variant approach DeepLTSC (LACNN) without integrating SFA model still easily mix those Web services with the Web services from head categories due to their highly semantic similarity.

3) Performance Impact of Hyperparameters: In the experiments, classification performance of our approach DeepLTSC is mainly influenced by three groups of hyperparameters. (1) The length of service description L_{desc} can heavily impact the amount of semantic information extracted by the long-tail module of service feature extraction. (2) During the extraction of deep service features by category attentive mechanism and deep neural network, the proportion of service name SN_{rate} can significantly affect the classification performance. (3) During the extraction of long-tail service features, focal loss function is applied for model training, where the hyperparameters α and γ can also influence the performance of service classification.

Fig. 4 illustrates the service classification performance of DeepLTSC (LACNN, SFA, Focal) on three datasets with the changes of L_{desc} and SN_{rate}. On one hand, unlike ordinary short text, there exists basically structured mode in a service description. It first describes the service functionality, and then interpret how to invoke a Web service. However, the performance of long-tail service classification mainly depends on functionality description, rather than the interpretation on service invocation. To validate the influence of service description length, we adjust the changes with different settings of L_{desc} on three service datasets. As shown in the upper part of Fig. 4, the experiment results demonstrate that L_{desc} indeed impacts service classification performance, and it has different values to receive the best classification performance on three service datasets with different long-tail distributions. Therefore, an appropriate parameter value of L_{desc} could be tuned for a target service dataset in real-world application scenario, where it can best adapt to the long-tail service distribution and boost the accuracy of Web service classification.

On the other hand, we apply service name as heuristic information to fuse more service domain knowledge for better classification performance. However, the features extracted from service name that contains more sparse semantic information than that from service description, which may also harm service classification performance. To attain the best performance, a certain percentage features of service name can be tuned and integrated into the features of service description. As shown in the lower part of Fig. 4, along with the changes of SN_{rate}, classification accuracy behaves more relatively sharp fluctuations on tail categories than head ones. The reason is that Web services from head categories have enough service descriptions to provide sufficient semantic information, so that heuristic features extracted from service names play a less important role in deep service feature extraction on head categories than that on tail ones. Specially, the results indicate that the effectiveness of heuristic feature from service name appears an increasing trend on tail services, as the number of service categories becomes biger and biger.

Fig. 5 illustrates the classification performance of DeepLTSC (LACNN, SFA, Focal) on multiple evaluation metrics with different parameters of α and γ. In the service dataset *PW*-120, we conduct experiments to test the performance impact on service classification. As shown in Fig. 5, to make a trade-off between head and tail service categories, the combination of $\alpha = 0.7$ and $\gamma = 0.5$ can be set to attain an overall optimal performance on Web service classification using focal loss as objective function.

E. Discussion

1) Experimental Results Summary: In order to verify the effectiveness of DeepLTSC, we compare it with eight state-of-the-art competing approaches and two our self-developed variants. The experimental results show that DeepLTSC achieves the best classification accuracy on multiple evaluation metrics under different long-tail distributions. Meanwhile, extensive experiments are conducted to verify the functionality of each component in DeepLTSC, including service category attention, deep service feature extraction, and service feature augmentation, by visualizing the differences on scatter diagrams among competing approaches as well as our self-developed ones. Moreover, performance impact of hyperparameters is performed for the verification of the classification performance of DeepLTSC. The experiments show that the length of service description, the ratio of features of service name, and the parameter settings of focal loss play an important role in long-tail service classification performance on DeepLTSC.

2) Threats to Validity: In the experiments, the long-tail service classification performance of DeepLTSC is significantly affected by the tuning and optimization of hyperparameters, including L_{desc}, SN_{rate}, α and γ. However, it is difficult to find an effective scheme to theoretically guide the settings of these hyperparameters. Currently, finding the optimal combination on hyperparameters mainly depends on the iterative comparison of multiple rounds of experiments. Thus, when faced with different long-tail distributions of service datasets, it is expected to reassign suitable hyperparameter values to ensure the effectiveness of DeepLTSC. Additionally, service descriptions provided by different developers may be more irregular, and service names may be less relevant to their corresponding Web services, which would potentially deteriorates the classification performance of DeepLTSC in real-world application scenarios.

3) The Advantages and Disadvantages of DeepLTSC: When performing the task of long-tail service classification, the advantages of our proposed approach DeepLTSC are twofold. First, we extract implicit service features by integrating multi-dimensional information of service description, service name and service category where LACNN is proposed to generate deep service features. The experimental results demonstrate that LACNN can significantly improve the overall performance on service classification. Second, to further enhance the service feature representation on tail categories, SFA model based on meta-learning is proposed to boost the classification accuracy of Web services on tail categories. The experimental results validate the effectiveness of SFA on tail services.

Inevitably, DeepLTSC also has some shortcomings. First, the depth of the model is greatly increased due to the application of BERT model, convolutional neural network, and SFA, which requires DeepLTSC longer training time to converge. Moreover, uniformly pruning all the service descriptions potentially cannot adapt to all the Web services, since they are published by different service vendors with diverse length of functionality descriptions. Theoretically, the best way to eliminate the influence of the redundant service descriptions is to individually identify the useless part of each Web service description. To solve the issue, a promising way is to provide individually self-adaptive pruning strategy for a Web service, where personalized length for each Web service can be intelligently detected to further improve the quality of long-tail service feature extraction.

VII. Conclusion and FutureWork

In this paper, we focus on the issue of long-tail Web service classification by deep learning and feature augmentation techniques called DeepLTSC. Service category attention and convolutional neural network are integrated together to capture deep service feature for better overall service classification across both head and tail service categories. Moreover, service feature augmentation model based on meta-learning is applied for further mining long-tail service feature, which can specially boost the classification accuracy of Web services on tail categories. Extensive experiments have been conducted on three real-world large scale service datasets. The experimental results demonstrate that DeepLTSC can receive superior performance on Web service classification compared with state-of-the-art as well as our self-developed variant approaches in multiple evaluation metrics.

In the future, we plan to explore more advanced classification algorithms by combining prior knowledge such as composability invocation relationships among Web services to further attain more explicitly differentiate service features.

References

[1] X. Liu, S. Agarwal, C. Ding, and Q. Yu, "An LDA-SVM active learning framework for Web service classification," in *Proc. IEEE Int. Conf. Web Serv. (ICWS)*, 2016, pp. 49–56.

[2] T. Liang, L. Chen, J. Wu, G. Xu, and Z. Wu, "SMS: A framework for service discovery by incorporating social media information," *IEEE Trans. Services Comput.*, vol. 12, no. 3, pp. 384–397, May/Jun. 2019.

[3] M. Tang, Z. Zheng, G. Kang, J. Liu, Y. Yang, and T. Zhang, "Collaborative Web service quality prediction via exploiting matrix factorization and network map," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 1, pp. 126–137, Mar. 2016.

[4] G. Zou, J. Chen, Q. He, K.-C. Li, B. Zhang, and Y. Gan, "NDMF: Neighborhood-integrated deep matrix factorization for service QoS prediction," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 4, pp. 2717–2730, Dec. 2020.

[5] B. Cheng, S. Zhao, J. Qian, Z. Zhai, and J. Chen, "Lightweight service mashup middleware with REST style architecture for iot applications," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 3, pp. 1063–1075, Sep. 2018.

[6] X. Luo, H. Wu, H. Yuan, and M. Zhou, "Temporal pattern-aware QoS prediction via biased non-negative latent factorization of tensors," *IEEE Trans. Cybern.*, vol. 50, no. 5, pp. 1798–1809, May 2020.

[7] S. Li, J. Wen, F. Luo, M. Gao, J. Zeng, and Z. Y. Dong, "A new QoS-aware Web service recommendation system based on contextual feature recognition at server-side," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 2, pp. 332–342, Jun. 2017.

[8] B. Bai, Y. Fan, W. Tan, and J. Zhang, "DLTSR: A deep learning framework for recommendations of long-tail Web services," *IEEE Trans. Services Comput.*, vol. 13, no. 1, pp. 73–85, Jan./Feb. 2020.

[9] M. Trevisan, I. Drago, M. Mellia, H. H. Song, and M. Baldi, "Awesome: Big data for automatic Web service management in SDN," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 1, pp. 13–26, Mar. 2018.

[10] P. Rodríguez-Mier, C. Pedrinaci, M. Lama, and M. Mucientes, "An integrated semantic Web service discovery and composition framework," *IEEE Trans. Services Comput.*, vol. 9, no. 4, pp. 537–550, Jul./Aug. 2016.

[11] C. Ieracitano, A. Paviglianiti, M. Campolo, A. Hussain, E. Pasero, and F. C. Morabito, "A novel automatic classification system based on hybrid unsupervised and supervised machine learning for electrospun nanofibers," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 1, pp. 64–76, Jan. 2021.

[12] Y. Yang, W. Ke, W. Wang, and Y. Zhao, "Deep learning for Web Services classification," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, 2019, pp. 440–442.

[13] G. M. Kapitsaki, "Annotating Web service sections with combined classification," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, 2014, pp. 622–629.

[14] T. Liang, L. Chen, J. Wu, and A. Bouguettaya, "Exploiting heterogeneous information for tag recommendation in API management," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, 2016, pp. 436–443.

[15] W. Shi, X. Liu, and Q. Yu, "Correlation-aware multi-label active learning for Web service tag recommendation," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, 2017, pp. 229–236.

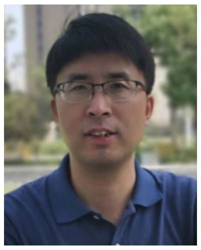
[16] H. Wang, Y. Shi, X. Zhou, Q. Zhou, S. Shao, and A. Bouguettaya, "Web service classification using support vector machine," in *Proc. IEEE Int. Conf. Tools Artif. Intell. (ICTAI)*, 2010, pp. 3–6.

[17] J. Liu, Z. Tian, P. Liu, J. Jiang, and Z. Li, "An approach of semantic Web service classification based on naive bayes," in *Proc. IEEE Int. Conf. Services Comput. (SCC)*, 2016, pp. 356–362.

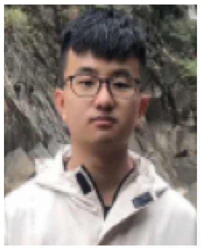
[18] Z. Tan, J. Chen, Q. Kang, M. Zhou, A. Abusorrah, and K. Sedraoui, "Dynamic embedding projection-gated convolutional neural networks for text classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 8, 2021, doi: 10.1109/TNNLS.2020.3036192.

[19] X. Shi, Q. Kang, J. An, and M. Zhou, "Novel L1 regularized extreme learning machine for soft-sensing of an industrial process," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1009–1017, Feb. 2022.

[20] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, 2014, pp. 1746–1751.
[21] J. Banzi, I. Bulugu, and Z. Ye, "Learning a deep predictive coding network for a semi-supervised 3D-hand pose estimation," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 5, pp. 1371–1379, Sep. 2020.
[22] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and X. Wu, "A data-characteristic-aware latent factor model for Web services QoS prediction," *IEEE Trans. Knowl. Data Eng.*, early access, Aug. 5, 2020, doi: 10.1109/TKDE.2020.3014302.
[23] D. Wu, Q. He, X. Luo, M. Shang, Y. He, and G. Wang, "A posterior-neighborhood-regularized latent factor model for highly accurate Web service QoS prediction," *IEEE Trans. Services Comput.*, early access, Dec. 24, 2019, doi: 10.1109/TSC.2019.2961895.
[24] X. Luo, D. Wang, M. Zhou, and H. Yuan, "Latent factor-based recommenders relying on extended stochastic gradient descent algorithms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 2, pp. 916–926, Feb. 2021.
[25] X. Luo, Z. Liu, S. Li, M. Shang, and Z. Wang, "A fast non-negative latent factor model based on generalized momentum method," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 610–620, Jan. 2021.
[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguist. Human Lang. Technol. (NAACL-HLT)*, 2019, pp. 4171–4186.
[27] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-scale long-tailed recognition in an open world," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 2537–2546.
[28] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 507–516.
[29] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 2980–2988.
[30] A. Graves, N. Jaitly, and A.-R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, 2013, pp. 273–278.
[31] G. Wang *et al.*, "Joint embedding of words and labels for text classification," in *Proc. Annu. Meeting Assoc. Comput. Linguist. (ACL)*, 2018, pp. 2321–2331.
[32] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling," 2016, *arXiv:1611.06639*.
[33] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.
[34] A. Joulin, E. Grave, and P. B. T. Mikolov, "Bag of tricks for efficient text classification," in *Proc. Conf. Eur. Chapter Assoc. Comput. Linguist. (EACL)*, 2017, pp. 427–431.
[35] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2015, pp. 2267–2273.
[36] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, 2008.



Guobing Zou received the Ph.D. degree in computer science from Tongji University, Shanghai, China, 2012. He is an Associate Professor and the Dean of the Department of Computer Science and Technology, Shanghai University, China. He has worked as a Visiting Scholar with the Department of Computer Science and Engineering, Washington University in St. Louis, USA, from 2009 to 2011. He has published more than 80 papers on premier international journals and conferences, including IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, IEEE International Conference on Web Services, International Conference on Service-Oriented Computing, and IEEE International Conference on Services Computing. His current research interests mainly focus on services computing, edge computing, data mining and intelligent algorithms, and recommender systems.



Song Yang received the bachelor's degree in computer science and technology from Shanghai University, China, 2019, where he is currently pursuing the master's degree with the School of Computer Engineering and Science. As the key member, he led a Research and Development Group to successfully design and implement a service-oriented big data analytics and visualization platform that has widely applied in environmental protection agency. He has published a paper on *International Journal of Computational Science and Engineering*. His research interests include service classification, deep learning, and natural language processing.

Shengyu Duan received the first B.Eng. degree in telecommunication engineering from the Huazhong University of Science and Technology, China, the second B.Eng. degree in electronic and electrical engineering from the University of Birmingham, U.K. in 2013, and the M.Sc. and Ph.D. degrees from the University of Southampton, U.K., in 2014 and 2019, respectively. He is currently working as an Assistant Professor with the School of Computer Engineering and Science, Shanghai University, China. His research expertise and interests include design for machine learning, IC reliability, hardware security, and hardware acceleration.

Bofeng Zhang (Member, IEEE) received the Ph.D. degree from Northwestern Polytechnic University in 1997, China. He is a Full Professor with the School of Computer Engineering and Science, Shanghai University. He experienced a Postdoctoral Research with Zhejiang University, China, from 1997 to 1999. He worked as a Visiting Professor with the University of Aizu, Japan, from 2006 to 2007. He worked as a Visiting Scholar with Purdue University, USA, from 2013 to 2014. He has published more than 150 papers on international journals and conferences. His research interests include service recommendation, intelligent human–computer interaction, and data mining.

Yanglan Gan received the Ph.D. degree in computer science from Tongji University, Shanghai, China, 2012. She is an Associate Professor with the School of Computer Science and Technology, Donghua University, Shanghai. She has published more than 50 papers on premier international journals and conferences, including *Bioinformatics*, *BMC Bioinformatics*, IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, IEEE International Conference on Web Services, International Conference on Service-Oriented Computing, *Neurocomputing*, and *Knowledge-Based Systems*. Her research interests include bioinformatics, service computing, and data mining.

Yixin Chen received the Ph.D. degree in computer science from the University of Illinois at Urbana Champaign in 2005. He is currently a Full Professor of Computer Science with Washington University in St. Louis, St. Louis, MO, USA. He has published more than 100 papers on premier international journals and conferences, including *Artificial Intelligence*, *Journal of Artificial Intelligence Research*, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IJCAI, AAAI, ICML, and KDD. His research interests include intelligent algorithms, data mining, machine learning, and big data analytics. He won the Best Paper Award at AAAI and a Best Paper Nomination at KDD. He is an Associate Editor for the *ACM Transactions on Intelligent Systems and Technology*, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and *Journal of Artificial Intelligence Research*.