WS-HFS: A Heterogeneous Feature Selection Framework for Web Services Mining

[1]Liang Chen, [2]Qi Yu, [3]Philip S. Yu, [1]Jian Wu
[1]*College of Computer Science & Technology, Zhejiang University, Hangzhou, China*
[2]*College of Computing & Informaton Sciences, Rochester Institute of Technology, Rocehster, USA*
[3]*Department of Computer Science, University of Illinois at Chicago, Chicago, USA*
[1]*{cliang,wujian2000}@zju.edu.cn,* [2]*qi.yu@rit.edu,* [3]*psyu@cs.uic.edu*

***Abstract*—With the development of Service Computing and Big Data research, more and more heterogeneous data generated in the process of Service Computing attracts our attention. Combining correlated data sources may help improve the performance of a given task. For example, in service recommendation, one can combine (1) user profile data (e.g. genders, age, etc.), (2) user log data (e.g., clickthrough data, service invocation records, etc.), (3) QoS data (e.g. response time, cost, etc.), (4) service functional description (e.g., service name, WSDL document, etc.) and (5) service tagging data (i.e., tags annotated by users) to build a recommendation model. All these data sources provide informative but heterogeneous features. For instance, user profile and QoS data usually have nominal features reflecting users' background and services' qualities, log data provides term-based features about users' historical behaviors, and service functional description and tagging data have term-based features reflecting services' functionalities and users' collective opinions. Given multiple heterogeneous data sources, one important challenge is to find a unified feature subspace to capture the knowledge from all data sources. To handle this problem, in this paper, we propose a Heterogeneous Feature Selection framework, named as WS-HFS, in which the consensus and the weight of different sources are both considered. Moreover, we apply the proposed framework to Web service clustering as a case study, and compare it with the state of the art approaches. The comprehensive experiments based on real data demonstrate the effectiveness of WS-HFS.**

***Keywords*-Heterogeneous Feature; Web Service Mining; Clustering**

I. Introduction

With the explosive growth and population of Web services, more and more heterogeneous data is generated in the process of Service Computing (e.g., service invocation, service composition, etc.). For example, WebserviceX.Net[1] serves 6,000,000+ Web services transactions every day, in which massive data are generated, e.g., user profile data, service invocation records, etc. Web service search engines, such as Seekda![2] and Titan[3], allow users to annotate tags to Web services, which generates lots of tagging data. Current research on QoS prediction [14], [18] also provides many QoS information which could not be obtained in the previous research.

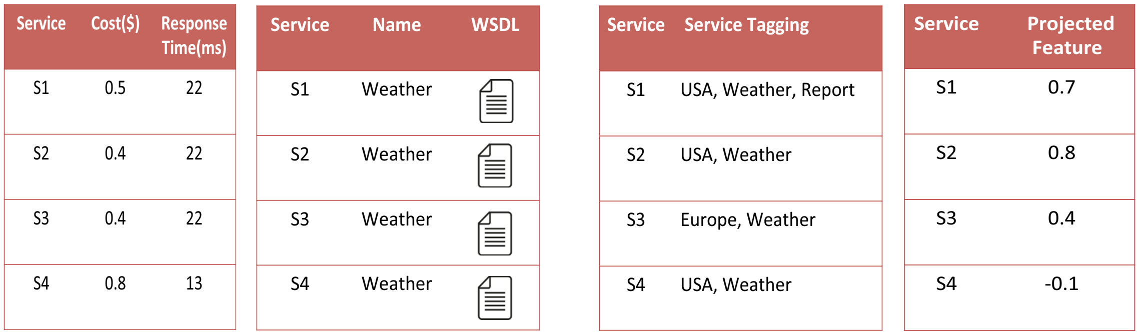
Combining correlated data sources may help improve the performance of a given Web service mining task. For example, in Web service substitution, a user may want to find a similar service to substitute an unavailable one by considering both the functional and non-functional properties. For this task, related data sources can be (1) QoS data (as shown in Fig.1(a)), (2) service functional description data (as shown in Fig.1(b)), and (3) service tagging data (as shown in Fig.1(c)). Each single data source may not be informative enough to build the accurate model, while the combination of them may provide comprehensive knowledge. For instance, in Fig.1(a), given that the weather forecasting service s_1 is unavailable, we have to select a similar one from the other three services to substitute s_1. For simplicity, the WSDL documents of the other three services are assumed to have the same similarity with the one of s_1. Note that it is difficult to obtain the answer just from any one of the data sources (Fig.1(a), Fig.1(b) or Fig.1(c)). However, if we combine these three data sources together, it can be observed that s_2 is similar to s_1 in all three data sources, thus s_2 could be selected to substitute s_1. In particular, it should be noted that s_1 and s_2 forecast the USA weather, while s_3 forecasts the Europe weather, hence s_2 is the better choice than s_3 even though they are both similar to s_1 in QoS and functional description.

The three data sources in Fig.1 provide informative but heterogeneous features. For instance, QoS data usually has nominal features reflecting services' qualities, service functional description and tagging data have term-based features reflecting services' functionalities and users' collective opinions. Given multiple heterogeneous data sources, one important challenge is how to combine them to collectively solve the problem, more specifically, how to find a unified feature subspace to capture the knowledge from all data sources. In particular, there are two points worth noting:

1) **How to find one unified feature subspace**. Given the heterogeneous features from multiple sources, it is not clear how to find one unified feature subspace. Further, the feature selection/reduction should be considered, because a straightforward join of features from different sources is not practical for real problem.
2) **How to weight different data sources**. For two

[1] WebserviceX.Net: http://www.webservicex.net/ws/default.aspx
[2] Seekda!: http://webservices.seekda.com/
[3] Titan: http://ccnt.zju.edu.cn:8080/

978-1-4673-7272-5/15 $31.00 © 2015 IEEE
DOI 10.1109/ICWS.2015.35



Service	Cost($)	Response Time(ms)
S1	0.5	22
S2	0.4	22
S3	0.4	22
S4	0.8	13

(a) QoS Data

Service	Name	WSDL
S1	Weather	📄
S2	Weather	📄
S3	Weather	📄
S4	Weather	📄

(b) Service Functional Description Data

Service	Service Tagging
S1	USA, Weather, Report
S2	USA, Weather
S3	Europe, Weather
S4	USA, Weather

(c) Service Tagging Data

Service	Projected Feature
S1	0.7
S2	0.8
S3	0.4
S4	-0.1

(d) A Unifined Feature Space

Figure 1: Related data sources in service substitution. Given service s_1 is unavailable, it is difficult to select the most suitable one for substitution by considering only one of the data sources (Fig.1(a), Fig.1(b) or Fig.1(c)). However, if we capture the knowledge from all three sources, it can be observed that s_2 is the most similar one to s_1. A unified feature subspace generated from this example is given in Fig.1(d)

different tasks, the effect of the same data source may be different. Furthermore, some data sources may contain substantial noise, thus it is desirable to reduce their effect.

To solve the above problems, we propose a Heterogeneous Feature Selection framework for Web service mining, named as WS-HFS. Given a certain task of Web service mining (e.g., service clustering, recommendation), the goal of WS-HFS is to capture the knowledge from multiple related data sources to handle it, the strategy is to project the features of multiple data sources onto a unified feature subspace, and the principle of finding unified feature subspace is to maximize the variance of the projected data. In particular, all feature spaces should be projected onto the same feature subspace consensually. For example, the heterogeneous features from Fig. 1(a) and Fig. 1(b) both have to agree on and be projected onto a common feature subspace as Fig. 1(d), such that their original data structures (e.g., data similarity) are preserved. Further, given multiple data sources, we propose a quadratic programming problem to identify sources that are more informative, and optimally weight them to learn a better projection. To evaluate the effectiveness of the proposed WS-HFS, we apply it to Web service clustering as a case study. The comprehensive experiments based on real-world Web services demonstrate the effectiveness of WS-HFS.

In particular, the main contributions of this paper can be summarized as follows:

1) We propose a novel Heterogeneous Feature Selection framework (WS-HFS) for Web service mining, in which the knowledge from multiple data sources can be captured and combined to solve problems.
2) We propose an approach to weight the effect of each data source, for the purpose of improving the accuracy of WS-HFS.
3) We crawl 15,968 real Web services to evaluate the effectiveness of WS-HFS framework and the proposed weight learning approach.

The rest of this paper is organized as follows. Section 2 gives an overview of the related work on Web service mining. Section 3 details the proposed WS-HFS framework, while Section 4 presents the weight learning approach that helps improving the performance of WS-HFS. Section 5 shows the comprehensive experimental results on Web service clustering, and Section 6 concludes the paper.

II. Related Work

Web service mining [6], which combines traditional service oriented computing (SOC) and state-of-the-art data mining techniques, attracts more and more attentions. QoS Data, WSDL documents, service invocation records are three main information sources for the research of Web service mining. In recent years, tagging data, which is annotated by users and provides meaningful descriptions, is utilized as another information source for Web service mining. Further, there is a trend that researchers begin to use multiple sources for Web service mining, rather than using only one source. In the following, we overview the state of the art research of the above four data sources for handling different tasks in Web service mining.

A. Web Service Clustering

Recently, Web service clustering [15], [13] has been demonstrated as an effective tool to boost the performance of Web service discovery. WSDL documents and tagging data are two main data sources for Web service clustering. Liu et al. and Elgazzar et al. propose to extract four kinds of information, i.e., *content, context, host name, and service name*, from the WSDL document to cluster Web services [5], [12]. SVD based and matrix factorization based approaches are adopted to achieve the co-clustering of services and operations in [16], [17]. Co-clustering exploits the duality relationship between services and operations to achieve better clustering quality than one-side clustering. In our prior work [2], we first proposed to utilize both WSDL documents and tags to cluster Web services by combining users' knowledge and service providers' knowledge. The LDA model is employed as another approach to utilize WSDL documents and tagging data [3].

B. Web Service Composition

QoS data and service invocation records are two main data sources for Web service composition. Canfora et al. first proposed to use genetic algorithms for QoS-aware service composition [1]. Liu et al. propose to enhance the performance of service composition by taking advantage of historical QoS records, rather than using the tentative QoS values advertised by the service provider [11]. Recently, Chen et al. propose to utilize both QoS data and service invocation records to improve the performance of service composition by using Bayes theorem and A^* algorithm [4]. In particular, Bayes theorem is employed to find the candidates with high possibility, while A^* algorithm is used to reduce the search space.

C. Web Service Recommendation

QoS-aware Web service recommendation is consistently a hot research topic, in which QoS data is utilized for service recommendation [18]. Recently, service invocation records and WSDL documents are employed as complemental sources for recommendation. Kang et al. propose to employ both service invocation records and QoS data for QoS-aware recommendation, by extracting users functional interests and QoS preferences from the invocation records [9]. Liang et al. propose to utilize both WSDL documents and QoS data for QoS-aware recommendation, by employing matrix factorization and topic model [10].

From the overview of the state of the art research, it can be observed that the usage of multiple heterogeneous data sources for Web service mining will be the trend in the near future. However, the usage of multiple sources in the above research is quite straightforward, which can not totally capture the knowledge from multiple sources. In the following of this paper, WS-HFS framework and its technique details will be introduced, and the comprehensive experiment results will demonstrate its effectiveness.

III. WS-HFS Framework

In this section, we first describe the architecture of the proposed WS-HFS framework for Web service mining, and then introduce the details of the heterogeneous feature selection approach.

A. Framework Overview

Figure 2 shows the architecture of the proposed WS-HFS framework, which consists of two major components: data preprocessing and service mining. In the first component, according to the given service mining task, related data sources are selected and the corresponding feature spaces are constructed. Using Fig. 1 as an example, QoS data source, service functional description data source, and service tagging data source are the related data sources. As for the feature space construction, for instance, each feature in the QoS data source represents a property of service quality (e.g., cost, response time), and each feature in the service functional description data source represents a word in the service name or in the WSDL document. After that, WS-HFS is employed to generate a unified feature subspace, in which the knowledge from all related data sources are captured and combined. Finally, according to the given task, the features in the unified feature subspace are utilized for task-oriented computation, and the result will be returned to the user. For example, given a Web service clustering task, the features in the unified feature space are utilized for service similarity computation, and the clustering result based on the service similarities will be returned to the user.

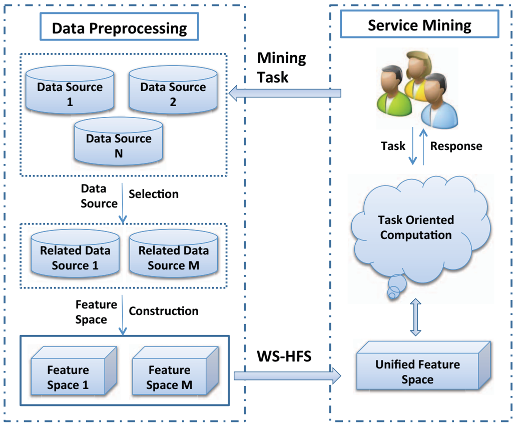


Figure 2: WS-HFS Framework for Web Service Mining

B. Heterogeneous Feature Selection

As discussed in Section 3.1, the data will be described in heterogeneous feature spaces from multiple sources. In this paper, we use column vector $x_i^j \in \mathbb{R}^{d_j}$ to denote the i-th data in the j-th data source (or feature space) whose dimension (i.e., number of features) is d_j. Using Fig.1(a) as an example, given QoS data source is the first data source, s_1 can be described as x_1^1=(0.5,22), and the dimension is 2. As for the text-based data source, such as WSDL document, each feature represents one word and the feature value reflects the presence or absence of the word in the document. In the matrix form, we denote $\mathbf{X}^j \in \mathbb{R}^{d_j \times m}$ as the set of data in the j-th feature space where m is the sample size (e.g., number of services in Fig.1).

WS-HFS is not simply joining features from multiple data sources together. Actually, joining features together for service mining is not practical in real applications. In such a case, given p related data sources, the dimension goes up to $\sum_j^p d_j$, which is approximately p times of the original dimension. In a linear classification model, it means the number of variables increases by p times, which may easily encounter the under-fitting problem. Further, some data sources may contain substantial noise that hurt the service mining performance. Thus, in some ways, heterogeneous feature selection is a kind of dimensionality reduction.

In this paper, we model heterogeneous feature selection as an unsupervised dimensionality reduction problem, and propose WS-HFS to project heterogeneous features from multiple sources onto a unified feature subspace. The principle of WS-HFS is to maximize the variance of the projected data to select the principal features. In addition, we introduce an important constraint, which forces all heterogeneous feature spaces to be projected onto the same subspace. Similar to common dimensionality reduction approaches, the data are normalized to zero mean, and the aim of WS-HFS is to find the orthogonal linear projection bias u^j for all feature spaces ($j = 1, 2, ..., p$). Thus, the general objective function can be described as follows:

$$\max_{\mathbf{u}} \Omega(\mathbf{u}) + \alpha\Psi(\mathbf{u}), \tag{1}$$

where $\Omega(\mathbf{u})$ means the variance of the projected data, which depends on the projection bias $\mathbf{u} = [u^1, u^2, ..., u^p]$. The second term $\Psi(\mathbf{u})$ is employed to reflect the constraint that forces all feature spaces to be projected on the same subspace. And the parameter α is introduced to control how strongly we want the data to be projected onto a unified subspace. And the detailed objective function can be written as follows:

$$\begin{aligned} \max_{u^1,u^2,...,u^p} & \sum_{j=1}^{p} w_j \sum_{i=1}^{m} (x_i^{j^T} u^j)^2 \\ & - \alpha \sum_{j=1}^{p}\sum_{k=1}^{p}\sum_{i=1}^{m} ||(x_i^{j^T} u^j) - (x_i^{k^T} u^k)||^2 \\ & s.t. ||u^j|| = 1, j = 1, 2..., p \end{aligned} \tag{2}$$

where $x_i^{j^T} u^j$ means the projection length of x_i^j on u^j, w_j means the weight of j-th data source (or feature space), and $||(x_i^{j^T} u^j) - (x_i^{k^T} u^k)||$ measures the "agreement" of projections from different data sources on the unified feature subspace. Ideally, we should give higher weights to the data sources that are more informative, however all weights are first set to be equal in this section. A data source weight learning approach will be introduced in the next section. To solve the optimization problem in Eq.2, we first derive an equivalent optimization problem. The second term in Eq.2 can be transformed as follows:

$$\begin{aligned} & - ||(x_i^{j^T} u^j) - (x_i^{k^T} u^k)||^2 \\ & = -(x_i^{j^T} u^j)^T (x_i^{j^T} u^j) - (x_i^{k^T} u^k)^T (x_i^{k^T} u^k) \\ & + 2(x_i^{j^T} u^j)^T (x_i^{k^T} u^k) \end{aligned} \tag{3}$$

Note that the first two terms in Eq.3 can be incorporated into the first term in Eq.2, and the last term is controlled by the parameter α. Thus, maximizing Eq.2 is equivalent to maximize the following equation:

$$\begin{aligned} \max_{u^1,u^2,...,u^p} & \sum_{j=1}^{p} w_j \sum_{i=1}^{m} (x_i^{j^T} u^j)^2 + \\ & \alpha \sum_{j=1}^{p}\sum_{k=1}^{p}\sum_{i=1}^{m} (x_i^{j^T} u^j)^T (x_i^{k^T} u^k) \\ & s.t. ||u^j|| = 1, j = 1, 2..., p \end{aligned} \tag{4}$$

And the optimization problem in Eq.4 can be written in a matrix form as follows:

$$\begin{aligned} \max_{u^1,u^2,...,u^p} \sum_{j=1}^{p} w_j u^{j^T} \mathbf{X}^j \mathbf{X}^{j^T} u^j + \alpha \sum_{j=1}^{p}\sum_{k=1}^{p} u^{j^T} \mathbf{X}^j \mathbf{X}^{k^T} u^k \\ s.t. ||u^j|| = 1, j = 1, 2..., p \end{aligned} \tag{5}$$

As discussed above, the weights of data sources are set to be equal, i.e., $w_j = \frac{1}{p}$. Thus, the left unknown variables in Eq. 5 are just the projection biases $u^1, u^2, ..., u^p$. To solve the optimization problem in Eq.5, we further transform it into a more compact form:

$$\max_{u} \mathbf{u}^T \mathbf{Z} \mathbf{u}, \quad s.t. ||u^j|| = 1, j = 1, 2..., p, \tag{6}$$

where $\mathbf{u} = [u^1, u^2, ..., u^p]$, and $\mathbf{Z}$ is defined as follows:

$$\mathbf{Z} = \begin{bmatrix} w_1 \mathbf{X}^1 \mathbf{X}^{1^T} & \alpha \mathbf{X}^1 \mathbf{X}^{2^T} & ... & \alpha \mathbf{X}^1 \mathbf{X}^{p^T} \\ \alpha \mathbf{X}^2 \mathbf{X}^{1^T} & w_2 \mathbf{X}^2 \mathbf{X}^{2^T} & ... & \alpha \mathbf{X}^2 \mathbf{X}^{p^T} \\ ... & ... & ... & ... \\ \alpha \mathbf{X}^p \mathbf{X}^{1^T} & \alpha \mathbf{X}^p \mathbf{X}^{2^T} & ... & w_p \mathbf{X}^p \mathbf{X}^{p^T} \end{bmatrix} \tag{7}$$

Then the optimization problem in Eq.6 is now a standard eigenvalue problem, where $\mathbf{u}$ are the eigenvectors that correspond to the top-p largest eigenvalues of $\mathbf{Z}$. Due to the space limitation, we do not show the details of the eigenvalue solution. After solving Eq.6, the projection bias $\mathbf{u} = [u^1, u^2, ..., u^p]$ can be obtained, and the unified feature subspace can be generated by projecting the heterogeneous feature spaces according to $\mathbf{u}$.

IV. Data Source Weight Learning

In the process of heterogeneous feature selection, the information contained in the features from different data sources are inherent different. Ideally, the data sources that are more informative should be given higher weights, which will not only help obtaining the more important information, but also reducing the impact of noise. In this section, we propose a data source weight learning approach to obtain optimal weights.

Before introducing the process of weight learning, we first introduce a $m \times m$ similarity matrix $\mathbf{C}$ (m is the sample size) :

$$\mathbf{C}(i,j) = \begin{cases} 1 & \text{if the } i\text{-th data and } j\text{-th data are similar} \\ 0 & \text{no preference} \\ -1 & \text{if the } i\text{-th data and } j\text{-th data are dissimilar} \end{cases}$$

It should be noted that "similar" "dissimilar" in similarity matrix $\mathbf{C}$ both reflect the relationship between i-th data and j-th data. The general idea of weight learning is to first use PCA[4][8] to obtain projections $\mathbf{\Phi}^k = \mathbf{X}^{k^T} u^k$ individually for each data source where u^k is given by PCA, and then obtain the optimal weights by finding which projection feature spaces $\mathbf{\Phi}^k$ can better satisfy the similarity matrix $\mathbf{C}$. Further, we define a relationship similarity matrix for the k-th data source as follows:

$$\mathbf{S}^k(i,j) = |\mathbf{C}(i,j)| \times (\mathbf{\Phi}^k(i)\mathbf{\Phi}^k(j)^T), \tag{8}$$

where $\mathbf{\Phi}^k(i)$ means the i-th data's vector in the k-th source's projection feature space, and $\mathbf{\Phi}^k(i)\mathbf{\Phi}^k(j)^T$ computes the similarity between i-th data and j-th data in the k-th source's projection feature space. $|\mathbf{C}(i,j)|$=1 if and only if the i-th data and the j-th data have relationship, regardless it is "similar" or "dissimilar". And if there is no relationship between the i-th data and the j-th data, $\mathbf{S}^k(i,j)$=0. Thus, we call $\mathbf{S}^k$ as relationship similarity matrix. Moreover, since $\mathbf{S}^k$ has zero mean as in PCA, $\mathbf{S}^k(i,j) > 0$ if and only if the i-th data and the j-th data are similar, and $\mathbf{S}^k(i,j) < 0$ iff the i-th data and the j-th data are dissimilar. Ideally, the relationship similarity matrix should have the same value as the similarity matrix. Thus, given relationship similarity matrix $\mathbf{S}^k$ from all sources where $k = 1, 2, ...p$, the objective function is to find a linear combination that could best satisfies the similarity matrix $\mathbf{C}$:

$$\begin{gathered} \min_{w_1, w_2, \ldots, w_p} ||(\sum_{k=1}^{p} w_k \mathbf{S}^k) - \mathbf{C}||_F^2 \\ s.t. \sum_{k=1}^{p} w_k = 1, w_k \geq 0, \end{gathered} \tag{9}$$

where $|| * ||_F^2$ is the Frobenius norm. By solving Eq.9, the optimal weights are obtained, which can then be used in Eq.6 to improve the performance of feature selection.

V. Experiments

In this section, we apply the proposed WS-HFS framework to handle the task of Web service clustering. State of the art approaches will be applied to compare with WS-HFS, and the performance of the proposed weight learning approach will also be evaluated.

A. Experiment Setup

To evaluate the performance of Web service clustering approaches, we crawl 15,361 real Web services from the Internet. For each Web service, we get the data of service name, service provider, WSDL document, and tag. For the convenience of comparison, we employ two data sources, i.e., WSDL documents and tagging data, to cluster Web services. In particular, we publicize the crawled dataset, which can be downloaded via http://www.zjujason.com.

As the manual creation of ground truth is an expensive process, we randomly select 228 Web services from the dataset we crawled to evaluate the performance of Web service clustering. We perform a manual classification of these 228 Web services to serve as the ground truth for the clustering approaches. Specifically, we distinguish the following categories: "Weather", "Stock", "SMS", "Finance", "Tourism", and "Email". There are 33 Web services in "Weather" category, 25 Web services in "Stock" category, 37 Web services in "SMS" category, 33 Web services in "Finance" category, 56 Web services in "Tourism" category, 24 Web services in "Email" category. 20 Web services are randomly selected from other categories as noise in our experiment. Limited by space, we don't show the detailed information of these Web services. For WSDL documents and tagging data source, each feature represents one word and the feature value reflects the presence or absence of the word in the document or tag. In particular, the widely used k-means [7] is chosen as the clustering algorithm due to its efficiency.

B. Evaluation Metric

To evaluate the performance of Web service clustering, we introduce four metrics: Precision, Recall, Purity, and NMI (Normalized Mutual Information), which are widely adopted in the information retrieval community. Given a set of labeled classes $\mathbb{C} = \{c_1, c_2, ...c_j\}$ and the clustered results $\mathbb{W} = \{w_1, w_2, ..., w_k\}$, Precision and Recall are defined as the following shows:

$$Precision_{w_i} = \frac{succ(w_i)}{succ(w_i) + mispl(w_i)} \tag{10}$$

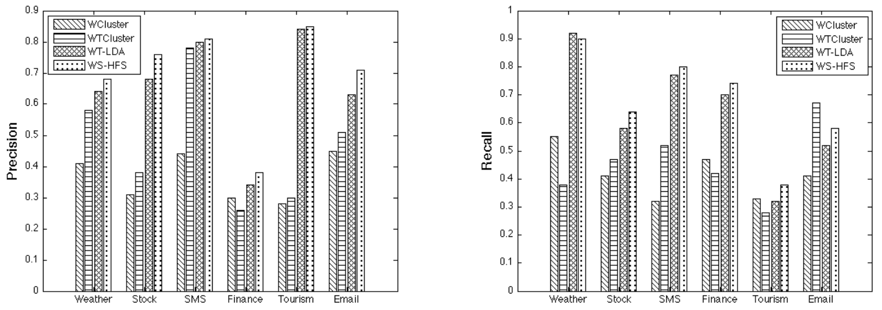
$$Recall_{w_i} = \frac{succ(w_i)}{succ(w_i) + missed(w_i)}, \tag{11}$$

where $succ(w_i)$ is the number of services successfully placed into cluster w_i, $mispl(w_i)$ is the number of services that are incorrectly placed into cluster w_i, and $missed(w_i)$ is the number of services that should be placed into w_i but are placed into another cluster.

The metric Purity evaluates the purity of the clustering result, and is defined as follows:

$$Purity(\mathbb{W}, \mathbb{C}) = \frac{1}{N} \sum_k \max_j |w_k \cap c_j|, \tag{12}$$

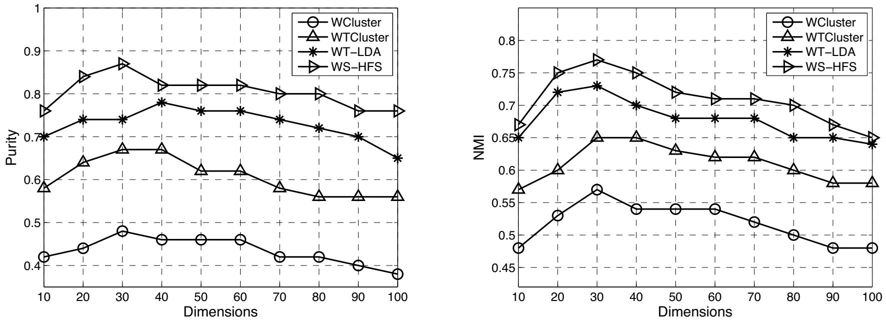
[4] Principal Component Analysis (PCA): A widely used approach in dimensionality reduction.



(a) Precision

(b) Recall

Figure 3: Web Service Clustering Performance Comparison via Precision and Recall



(a) Purity

(b) NMI

Figure 4: Web Service Clustering Performance Comparison via Purity and NMI

where N means the number of all services, and $|w_k \cap c_j|$ is the number of services that both belong to w_k and c_j. Bad clusterings have purity values close to 0, a perfect clustering has a purity of 1. And the metric NMI (Normalized Mutual Information) evaluates the clustering performance in the way of information entropy theory. The definition of NMI is as follows:

$$NMI(\mathbb{W}, \mathbb{C}) = \frac{I(\mathbb{W}, \mathbb{C})}{[H(\mathbb{W}) + H(\mathbb{C})]/2} \tag{13}$$

$$I(\mathbb{W}, \mathbb{C}) = \sum_k \sum_j \frac{|w_k \cap c_j|}{N} \log \frac{N|w_k \cap c_j|}{|w_k||c_j|}, \tag{14}$$

$$H(\mathbb{W}) = \sum k \frac{|w_k|}{N} \log \frac{|w_k|}{N}, H(\mathbb{C}) = \sum j \frac{|c_j|}{N} \log \frac{|c_k|}{N}, \tag{15}$$

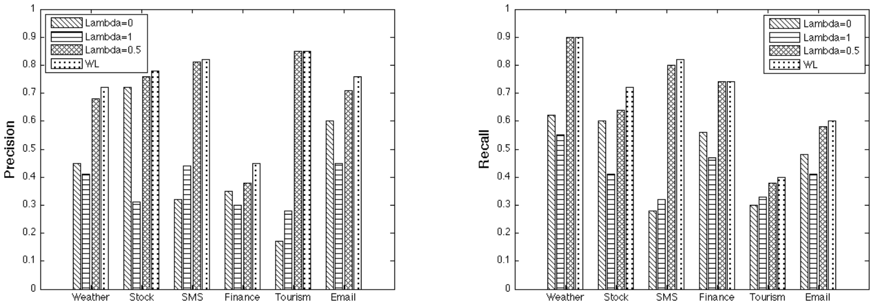
where $I(\mathbb{W}, \mathbb{C})$ is the mutual information between $\mathbb{W}$ and $\mathbb{C}$, while $H(\mathbb{W})$ and $H(\mathbb{C})$ are the information entropy for normalization. The value of NMI is in the range of (0,1), the higher means better.

C. Performance of Web Service Clustering

In this section, we apply the proposed WS-HFS framework to cluster Web services, and compare the performance of four Web service clustering approaches, including three state-of-the-art clustering approaches and the proposed WS-HFS. The details of these algorithms are given below:

1) ***WCluster***. In this approach, Web services are clustered according to the semantic WSDL-level similarity between Web services. This approach has been adopted in some related works [2], [5], [12].
2) ***WTCluster***. In this approach, both WSDL documents and the tagging data are employed to cluster the Web services according to the composite semantic similarity [2].
3) ***WT-LDA***. In this approach, LDA model is employed to cluster Web services based on WSDL documents and tagging data [3].
4) ***WS-HFS***. In this approach, Heterogeneous features projected in the unified subspace are utilized to compute the service similarity for the purpose of clustering. In particular, weight learning is not employed.

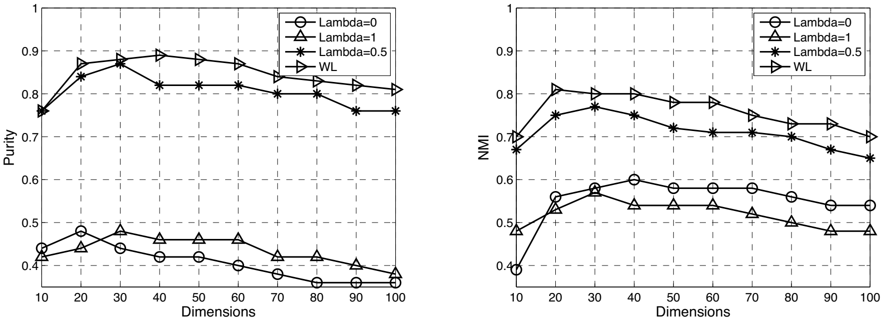
Figure 3 shows the performance comparison of above 4 Web service clustering approaches in terms of precision and recall. From Fig.3, it can be observed that the performance of WCluster is the worst in most cases. This is because only WSDL documents are utilized in WCluster. Among the other



(a) Precision

(b) Recall

Figure 5: Impact of Source Weight in terms of Precision and Recall



(a) Purity

(b) NMI

Figure 6: Impact of Source Weight in terms of Purity and NMI

three approaches which utilize both WSDL documents and tagging data, the proposed WS-HFS is the best, while the WTCluster is the worst. In WTCluster, it simply extracts the keywords from WSDL documents as the selected features, which is quite naive and straightforward. Compared with WTCluster, the feature selection strategy of the proposed WS-HFS considers the connections among heterogeneous data sources, which eventually improves the performance of clustering.

Figure 4 shows the clustering performance comparison in terms of purity and NMI. From Fig. 4(a), it can be observed that the purity value of each approach first rises then falls with the increase of the dimension (i.e., number of feature). This is because when the number of selected features is too small, the information is not enough to make accurate clustering decision. While the number of selected features is large, there are noise features which impact the performance of clustering. Moreover, the proposed WS-HFS outperforms the other three approaches in all cases. Similarly, the trend of NMI value in Fig.4(b) first rises then falls, and the performance of WS-HFS is still the best in terms of NMI.

D. Evaluation of Weight Learning

In this section, we evaluate the performance of weight learning by comparing the performance of WS-HFS with different weight settings. Since only two data sources are utilized in the experiment, i.e., WSDL documents and tagging data, a parameter λ is employed to reflect the weight of WSDL documents and the weight of tagging data is set to $(1-\lambda)$. In particular, we implement four versions of WS-HFS:

1) $\boldsymbol{\lambda = 0}$, where only tagging data are employed for Web service clustering.
2) $\boldsymbol{\lambda = 1}$, where only WSDL documents are employ for Web service clustering.
3) $\boldsymbol{\lambda = 0.5}$, where the weights of WSDL documents and tagging data are equal.
4) **weight learning (WL)**, where weight learning presented in Section 4 is employed, and an optimal λ is obtained for Web service clustering.

Figure 5 shows the clustering performance of WS-HFS with different weight settings in terms of precision and recall. It can be observed that WS-HFS with $\lambda = 0.5$ and weight learning outperform those with $\lambda = 0$ or $\lambda = 1$. This is because only WSDL documents are employed when

$\lambda = 1$, and only tagging data are employed when $\lambda = 0$. Further, the WS-HFS with an optimal weight outperforms the one with $\lambda = 0.5$ in most cases, since more informative data sources are given higher weights through weight learning.

Figure 6 shows the clustering performance comparison in terms of purity and NMI. Similar to Figure 5, WS-HFS with an optimal weight and $\lambda = 0.5$ largely outperform the other two approaches, which demonstrates the importance of heterogeneous data sources integration in Web service mining. From Figure 6, it can also be observed the impact of dimensionality (number of features) to the performance of clustering. Similar to the trend in Figure 4, the performances of different approaches first rise then fall with the increase of dimension. Furthermore, WS-HFS with an optimal weight outperforms the other three approaches in all cases in terms of purity and NMI.

VI. Conclusion

With the explosive growth and popularity of Web services, more and more heterogeneous data are generated in the process of Service Computing (e.g., service invocation, service composition, etc.). A single data source may not be informative enough for the complex tasks in Web service mining. Usage of multiple data sources for Web service mining will be the trend in the near future.

In this paper, we propose a heterogeneous feature selection framework for Web service mining, referred to as WS-HFS, which captures the knowledge from multiple sources by projecting multiple feature spaces onto a unified feature subspace. Moreover, a data source weight learning approach is proposed to improve the performance of WS-HFS, by computing optimal weights for different data sources. Extensive experiments conducted over real Web services demonstrate the effectiveness of the proposed WS-HFS framework and weight learning approach.

VII. Acknowledgments

This research was partially supported by the National Technology Support Program under the grant of 2011BAH16B04, the Natural Science Foundation of China under the grant of No. 61173176, National High-Tech Research and Development Plan of China under the Grant No. 2013AA01A604.

References

[1] G. Canfora, M. D. Penta, R. Esposito, and M. L. Villani. An approach for qos-aware service composition based on genetic algorithms. *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 1069–1075, 2005.

[2] L. Chen, L. Hu, Z. Zheng, and J. Wu. Wtcluster: Utilizing tags for web services clustering. *International Conference on Service Oriented Computing*, pages 204–218, 2011.

[3] L. Chen, Y. Wang, Q. Yu, Z. Zheng, and J. Wu. Wt-lda: User tagging augmented lda for web service clustering. *International Conference on Service Oriented Computing*, pages 162–176, 2013.

[4] L. Chen, J. Wu, H. Jian, H. Deng, and Z. Wu. Instant recommendation for web services composition. *IEEE Transactions on Service Computing*, Preprint, 2013.

[5] K. Elgazzar, A. E. Hassan, and P. Martin. Clustering wsdl documents to bootstrap the discovery of web services. *International Conference on Web Services*, pages 147–154, 2009.

[6] Z. George and B. Athman. Web service mining. *Springer*, 2010.

[7] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[8] J. Ian. *Principal component analysis*. John Wiley & Sons, online, 2005.

[9] G. Kang, J. Liu, M. Tang, X. Liu, B. Cao, and Y. Xu. Awsr: Active web service recommendation based on usage history. *IEEE International Conference on Web Services*, pages 186–193, 2012.

[10] T. Liang, L. Ji, L. Chen, J. Wu, and Z. Wu. Collaborative qos prediction via matrix factorization and topic model. *International Conference on Service-Oriented Computing and Applications*, pages 282–289, 2013.

[11] W. Lin, W. Dou, X. Luo, and C. Jinjun. A history record-based service optimization method for qos-aware service composition. *IEEE International Conference on Web Services*, pages 666–673, 2011.

[12] W. Liu and W. Wong. Discovering homogeneous service communities through web service clustering. *Service-Oriented Computing: Agents, Semantics, and Engineering*, pages 69–82, 2008.

[13] C. Platzer, F. Rosenberg, and S. Dustdar. Web service clustering using multidimensional angles as proximity measures. *ACM Transactions on Internet Technology*, 9(3):1–26, 2009.

[14] J. Wu, L. Chen, Y. Feng, Z. Zheng, M. Zhou, and Z. Wu. Predicting quality of service for selection by neighborhood-based collaborative filtering. *IEEE Transactions on System, Man, and Cybernetics, Part A*, 43(2):428–439, 2013.

[15] H. Yang, J. Chen, X. Meng, and B. Qiu. Dynamically traveling web service clustering based on spatial and temporal aspects. *Lecture Notes in Computer Science*, 4802:348–357, 2007.

[16] Q. Yu. Place semantics into context: Service community discovery from the wsdl corpus. In *International Conference on Service Oriented Computing*, pages 188–203, 2011.

[17] Q. Yu and M. Rege. On service community learning: A co-clustering approach. In *Internatonal Conference on Web Services*, pages 283–290, 2010.

[18] Z. Zheng, H. Ma, M. R. Lyu, and I. King. QoS-aware Web service recommendation by collaborative filtering. *IEEE Transactions on Service Computing*, 4(2):140–152, 2011.