A Deep Hybrid Graph Kernel Through Deep Learning Networks

Lixin Cui*§, Lu Bai*§, Luca Rossi†, Yue Wang*, Yuhang Jiao*, and Edwin R. Hancock‡
*Central University of Finance and Economics, 39 South College Road, Beijing, China
†School of Engineering and Applied Science, Aston University, Birmingham, B4 7ET, UK
‡Department of Computer Science, University of York, York, YO10 5DD, UK
§*These authors are co-first authors*

***Abstract*—In this paper, we develop a new deep hybrid graph kernel. This is based on the depth-based matching kernel [1] and the Weisfeiler-Lehman subtree kernel [2], by jointly computing a basic deep kernel that simultaneously captures the relationship between the combined kernels through deep learning networks. Specifically, for a set of graphs under investigations, we commence by computing two kernel matrices using each of the separate kernels. With the two kernel matrices to hand, for each graph we use the kernel value between the graph and each of the training graphs as the graph characterisation vector. This vector can be seen as a kernel-based similarity embedding vector of the graph [3]. We use the embedding vectors of all graphs to train a deep autoencoder network, that is optimized using Stochastic Gradient Descent together with the Deep Belief Network for pretraining. The deep representation computed through the deep learning network captures the main relationship between the depth-based matching kernel and the Weisfeiler-Lehman subtree kernel. The resulting deep hybrid graph kernel is computed by summing the original kernels together with the dot product kernel between their deep representations. We show that the deep hybrid graph kernel not only captures the joint information between the associated depth-based matching and Weisfeiler-Lehman subtree kernels, but also reflects the information content over all graphs under investigations. Experimental evaluations demonstrate the effectiveness of the proposed kernel.**

I. Introduction

In recent years, graph kernels have been shown to provide powerful tools for the analysis of graphs [4]. One main advantage of using graph kernels is that they can better preserve the structural information of graphs by mapping them to points in a high dimensional Hilbert space.

Generally speaking, most state-of-the-art graph kernels are instances of the generic R-convolution kernel developed by Haussler [5], i.e., they compare pairs of substructures resulting from the input graphs decomposition. Thus, any graph decomposition approach can be employed to develop a new graph kernel, e.g., walks [6], paths [7], restricted subgraphs [8], or subtrees[9], [2], [10], [11]. Unfortunately, most R-convolution based kernels suffer from the drawback of ignoring the relative arrangement of correspondences between decomposed substructures. This problem occurs when an R-convolution kernel roughly identifies any pair of isomorphic substructures and does not consider locational arrangement of correspondences between the substructures.

To overcome the shortcoming of neglecting structural correspondence information of R-convolution kernels, Neuhaus and Bunke [12] introduce an aligned random walk kernel by identifying the alignments through graph edit-distances. Fröhlich et al. [13] developed a family of optimal assignment kernels based on aforehand aligned substructures. Bai et al. [14] developed a depth-based matching kernel by aligning vertices through depth-based representations [15]. All these kernels establish reliable structural correspondence information and reflect more precise kernel-based similarities between graphs. However, these graph kernels are all generally based on the analysis of one type of substructure, hence potentially missing some important structural information. This is because it is difficult to define a graph kernel that encapsulates any type of structural pattern. One way to overcome this problem is to compute a hybrid kernel by combining a number of alternative kernels with different properties. Specifically, with two existing kernel measures K_1 and K_2 to hand, a hybrid graph kernel $K_{\rm Hyb}$ [16] can be defined as

$$K_{\rm Hyb} = \mu K_1 + (1-\mu)K_2, \tag{1}$$

where $0 \leq \mu \leq 1$. The main objective here is to learn the rate μ which ensures that $K_{\rm Hyb}$ combines the similarity information of the original kernels K_1 and K_2 while achieving a higher performance than using the original kernels individually. Unfortunately, determining the optimal rate μ is not trivial and typically requires cross-validation methods to search hyperparameters space in a principles way. Furthermore, the optimal value of μ usually varies across different datasets, thus influencing the runtime of the learning process.

The aim of this paper is to develop a new deep hybrid graph kernel based on the depth-based matching kernel [1] and the Weisfeiler-Lehman subtree kernel [2], by jointly computing a basic deep kernel that simultaneously captures the relationship between the combined kernels through deep learning networks. The reasons for using these two kernels are twofold. First, the depth-based matching kernel can identify reliable structural correspondence information between substructures by aligning their vertices. Second, the Weisfeiler-Lehman subtree kernel can reflect the intrinsic structure information through the subtrees rooted at each vertex with increasing heights. Thus, these graph kernels are not only powerful tools for graph classification problems, but also have different theoretical advantages. In this work, we are interested to define a new hybrid graph kernel that possesses the advantages of both the depth-based matching kernel and the Weisfeiler-Lehman subtree kernel. Specifically, for a set of graphs under investigations, we commence by computing two kernel matrices using the depth-based matching kernel and the Weisfeiler-Lehman subtree kernel respectively. With the two kernel matrices to hand, for each graph we use

the kernel value between the graph and each of the training graphs as the graph characterisation vector. This vector can be seen as a kernel-based similarity embedding vector of the graph [3]. We use the embedding vectors of all graphs to train a deep autoencoder network, that is optimized using Stochastic Gradient Descent together with a Deep Belief Network for pretraining. Based on [17], the deep autoencoder network can minimize the reconstruction error of the output and input embedding vectors that encapsulate the joint kernel-based information of the combined graph kernels, the deep network can jointly capture the salient information between the kernels in a highly non-liner latent space. As a result, unlike the traditional hybrid kernel that needs learning of an associated rate μ to reflect the relationship between the combined kernels, we can directly capture the main relationship between the combined kernels through the deep autoencoder network. Furthermore, we compute the deep representation of each graph through the deep network. The resulting deep hybrid graph kernel is computed by summing the matching kernel, the subtree kernel and the dot product kernel between the deep representations of the graphs. We show that the deep hybrid graph kernel not only captures the joint information between the associated combined depth-based matching and Weisfeiler-Lehman subtree kernels, but also reflect the information content over all graphs under investigations. Experimental evaluations demonstrate the effectiveness of the proposed kernel.

The remainder of this paper is organized as follows. Section II illustrates the preliminary concepts that will be used in this work. Section III gives the definition of the deep hybrid graph kernel. Section IV provides our experimental evaluation. Finally, Section V concludes this work.

II. Preliminary Concepts

In this section, we will introduce two state-of-the-art graph kernels that will be used to compute the deep hybrid graph kernel. We commence by reviewing the concept of the depth-based matching kernel. Finally, we review the concept of the Weisfeiler-Lehman subtree kernel.

A. The Depth-based Matching Kernel

The depth-based matching kernel is a vertex alignment kernel based on aligning the depth-based representations rooted at vertices between graphs [1]. Unlike most existing R-convolution kernels that ignore the structural correspondence information between substructures [5], this kernel can be seen as an aligned subgraph kernel that incorporates locational correspondences between pairs of isomorphic subgraphs.

Specifically, assume a graph $G(V,E)$ with V as the vertex set and E as the edge set. We commence by computing the shortest path matrix S_G for G, and each element $S_G(v,u)$ represents the shortest path length between vertices $v \in V$ and $u \in V$. The K-layer neighborhood vertex set N_v^K for a vertex $v \in V$ is determined by

$$N_v^K = \{u \in V \mid S_G(v,u) \leq K\}, \tag{2}$$

For the graph $G(V,E)$, the K-layer expansion subgraph $\mathcal{G}_v^K(\mathcal{V}_v^K;\mathcal{E}_v^K)$ rooted at vertex v is

$$\begin{cases} \mathcal{V}_v^K &= \{u \in N_v^K\}; \\ \mathcal{E}_v^K &= \{u,w \in N_v^K,\ (u,w) \in E\}. \end{cases} \tag{3}$$

Note that, if K is equal to or greater than the longest shortest path length L from v to the remaining vertices, the K-layer expansion subgraph rooted at v is the global structure of $G(V,E)$. For the graph $G(V,E)$, we compute the K-layer depth-based representation $DB_G^h(v)$ around $v \in V$ as the point of v, and

$$DB_G^K(v) = [H_S(\mathcal{G}_v^1), \cdots, H_S(\mathcal{G}_v^k), \cdots, H_S(\mathcal{G}_v^K)]^\top, \tag{4}$$

where $(k \leq K)$ and $H_S(\mathcal{G}_v^K)$ is the Shannon entropy of the subgraph $\mathcal{G}_v^K$ associated with the steady state random walk [18].

For a pair of graphs $G_p(V_p,E_p)$ and $G_q(V_q,E_q)$, we compute an affinity matrix R between pairwise vertices of G_p and G_q for vertex matching. Specifically, for each pair of vertices $v_i \in V_p$ and $u_j \in V_p$, we compute the Euclidean distance between their K-layer depth-based representations $DB_{G_p}^K(v_i)$ and $DB_{G_q}^K(u_j)$ as the element $R_{pq}(i,j)$ of R_{pq}, i.e.,

$$R_{pq}(i,j) = \|DB_{G_p}^K(v_i) - DB_{G_q}^K(u_j)]\|_2, \tag{5}$$

where $i \in \{1,2,\cdots,|V_p|\}$, $j \in \{1,2,\cdots,|V_q|\}$ and R is a $|V_p| \times |V_q|$ matrix. If $R_{pq}(i,j)$ is the smallest element both in row i and in column j, we say that there is a one-to-one correspondence between the vertices $v_i \in V_p$ $u_j \in V_q$. The state of correspondence between the vertices of G_p and G_q is recorded in a correspondence matrix $C_{pq} \in \{0,1\}^{|V_p||V_q|}$, and each element $C_{pq}(i,j)$ satisfies

$$C_{pq}(i,j) = \begin{cases} 1 & \text{if } R(i,j) \text{ is the smallest element} \\ & \text{both in row } i \text{ and in column } j; \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Eq.(6) indicates that if $C_{pq}(i,j) = 1$, the vertices v_i and v_j are matched.

For a pair of graphs $G_p(V_p,E_p)$ and $G_q(V_q,E_q)$, we compute the depth-based matching kernel $k_{\mathrm{DB}}^{(h)}(G_p,G_q)$ by counting the number of aligned vertex pairs associated with the correspondence matrix C_{pq}, i.e.,

$$k_{\mathrm{DB}}^{(h)}(G_p,G_q) = \sum_{i=1}^{|V_p|}\sum_{j=1}^{|V_q|} C_{pq}(i,j). \tag{7}$$

B. The Weisfeiler-Lehman Subtree Kernel

In this subsection, we review the concept of the Weisfeiler-Lehman subtree kernel. This kernel is based on counting the number of the isomorphic subtree pairs, as identified by the Weisfeiler-Lehman algorithm [19].

Specifically, for a sample graph $G(V,E)$ and a vertex $v \in V$, we denote the neighbourhood vertices of v as $\mathcal{N}(v) = \{u|(v,u) \in E\}$. For each iteration m where $m > 1$, the Weisfeiler-Lehman algorithm strengthens the current label $\mathcal{L}_{\mathrm{WL}}^{m-1}(v)$ of each vertex $v \in V$ as a new label $\mathcal{L}_{\mathrm{WL}}^m(v)$ by taking the union of the current labels of vertex v and its neighbourhood vertices in $\mathcal{N}(v)$, i.e.,

$$\mathcal{L}_{\mathrm{WL}}^m(v) = \bigcup_{u \in \mathcal{N}(v)} \{\mathcal{L}_{\mathrm{WL}}^{m-1}(v), \mathcal{L}_{\mathrm{WL}}^{m-1}(u)\}, \tag{8}$$

Note that, when $m = 1$ the current label $\mathcal{L}^0_{\mathrm{WL}}(v)$ of v is its initial vertex label. Based on the definition in [19], for each iteration m the new label $\mathcal{L}^M_{\mathrm{WL}}(v)$ of v corresponds to a specific subtree structure of height m rooted at v. Furthermore, for a pair of graphs $G_p(V_p, E_p)$ and $G_q(V_q, E_q)$, if the new updated vertex labels of $v_p \in V_p$ and $v_q \in V_q$ at the m-th iteration are identical, the subtrees corresponded by these new labels are isomorphic. Thus, the Weisfeiler-Lehman subtree kernel $k^{(M)}_{\mathrm{WL}}(G_p, G_q)$, that counts the pairs of isomorphic subtrees [2], can be defined by counting the number of identical updated vertex labels at each iteration m, i.e.,

$$k^{(M)}_{\mathrm{WL}}(G_p, G_q) = \sum_{m=0}^{M} \sum_{v_p \in V_p} \sum_{v_q \in V_q} \delta\{\mathcal{L}^m_{\mathrm{WL}}(v_p), \mathcal{L}^m_{\mathrm{WL}}(v_q)\}, \tag{9}$$

where

$$\delta(\mathcal{L}^m_{\mathrm{WL}}(v_p), \mathcal{L}^m_{\mathrm{WL}}(v_q) = \begin{cases} 1 & \text{if } \mathcal{L}^m_{\mathrm{WL}}(v_p) = \mathcal{L}^m_{\mathrm{WL}}(v_q), \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

III. The Deep Hybrid Graph Kernel

In this section, we introduce a framework for computing a new deep hybrid graph kernel. We commence by reviewing the concept of deep autoencoder network [17]. Finally, we show how to compute the deep hybrid graph kernel based on the depth-based graph kernel and the Weisfeiler-Lehman subtree kernel through the deep network.

A. Deep Autoencoder Networks

In this subsection, we review the deep autoencoder, which is an example of unsupervised deep learning network [17] and consists of an encoder network and a decoder network. On the one hand, the encoder network comprises a number of non-linear functions that can transform the original input data into a lower dimensional deep representation [20], [21]. On the other hand, the decoder network comprises a number of non-linear functions that can reconstruct the original input data based on the deep representations. It has been proven that these deep representations are able to capture the manifold structure of the input data embedded in a highly non-linear space. In other words, the deep representation enhances the linear separability of the original input data [17].

Fig.1 shows the deep autoencoder network architecture that will be used in this work. The parameters of the network are shown in Table.I. Let x_i be an input. For each t-th layer, we compute the hidden representation y^t_i as

$$y^t_i = \begin{cases} \sigma(W^{(1)} x_i + b^{(1)}) & \text{if } t = 1; \\ \sigma(W^{(t)} y^{t-1}_i + b^{(t)}) & \text{if } t = 2, ..., T. \end{cases} \tag{11}$$

After T steps, we transform the input data x_i into $y^{(T)}_i$ through the encoder network. Then, after another T steps, we reverse the calculation process of the encoder network and reconstruct x_i as output $\hat{x}_i$ based on $y^{(T)}_i$ through the decoder network. $y^{(T)}_i$ is the resulting deep representation of the input data x_i. The objective of the deep autocoder network is to minimize the reconstruction error of output $\hat{x}_i$ and input x_i, and the loss function is defined as

$$L = \sum_{i=1}^{n} \|\hat{x}_i - x_i\|_2^2 \tag{12}$$

To optimize the deep autocoder network, we first use the Deep Belief Network [17] to pretrain its parameters to avoid trapping in local optimum in the parameter space. Then, the deep network is optimized by means of the Stochastic Gradient Descent method, where the gradients can be conveniently obtained by applying the chain rule to backpropagate error derivatives first through the decoder network and then through the encoder network, i.e., back-propagate $\frac{\partial L}{\partial \theta}$ to update θ^k.

As [20] stated, minimizing the reconstruction error can smoothly capture the manifold structure of the original data and thus capture the main characteristics of the data.

B. The Deep Hybrid Graph Kernel

In this subsection, we propose a new deep hybrid graph kernel based on the depth-based graph kernel [1] and the Weisfeiler-Lehman subtree kernel [2] through the deep autoencoder network. The reason of using these two state-of-the-art graphs is twofold. First, the Weisfeiler-Lehman subtree kernel can not only accommodate vertex labels but also capture rich intrinsic structure information through the expanding subtree rooted at each vertex. Second, the depth-based matching kernel can establish reliable structural correspondence information between graphs that is ignored by the Weisfeiler-Lehman subtree kernel. In other words, while these two graph kernels have different characteristics with different advantages, the resulting hybrid kernel will combine these advantages.

Assume we have a set of graphs $\mathbf{G} = \{G_1, \cdots, G_i, \cdots, G_N\}$. We commence by computing the kernel matrices K_{DB} and K_{WL} using the depth-based matching kernel and the Weisfeiler-Lehman subtree kernel, respectively. Each kernel matrix $K_{\cdot}$ is a $N \times N$ matrix, and each element $K_{\cdot}(i, j)$ indicates the similarity between a pair of graphs $G_i \in \mathbf{G}$ and $G_j \in \mathbf{G}$ in terms of a graph kernel measure. For each graph $G_i \in \mathbf{G}$, we compute the characteristics vector V_i of G_i associated with the i-th rows of the kernel matrices K_{DB} and K_{WL}, i.e.,

$$\begin{aligned} V_i = [&K_{\mathrm{DB}}(i, 1), \cdots, K_{\mathrm{DB}}(i, M), \\ &K_{\mathrm{WL}}(i, 1), \cdots, K_{\mathrm{WL}}(i, M)], \end{aligned} \tag{13}$$

where M denotes the number of training graphs. Based on the definition in [3], V_i can be seen as a vectorial representation of G_i based on the similarity between G_i and a set of prototype graphs. Here, we use the graph kernels as the similarity measures and the training graphs in $\mathbf{G}$ as the prototype graphs. As a result, this joint kernel-based similarity embedding vectors encapsulate rich structural information from both the depth-based matching kernel and the Weisfeiler-Lehman subtree kernel.

With the kernel-based similarity embedding vectors of all graphs in $\mathbf{G}$ to hand, we use these embedding vectors as input data to train the deep autoencoder network introduced in Section III-A. Based on Eq.(11) we obtain a set of N deep representation vectors of all graphs in $\mathbf{G}$ as

$$\mathbf{DRs} = \{y^{(K)}_1, \cdots, y^{(K)}_i, \ldots, y^{(K)}_N\}, \tag{14}$$

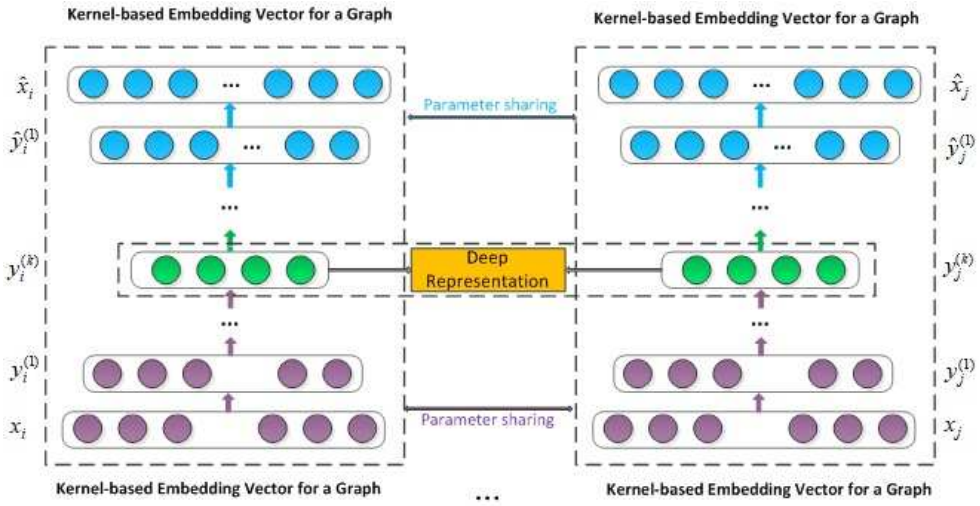

Fig. 1. The Architecture of the Deep Autorencode Network.

TABLE I. TERMS AND NOTATIONS

Symbol	Definition
n	**number of vertexes**
K	**number of layers**
$X = \{x_i\}_{i=1}^n, \hat{X} = \{\hat{x}_i\}_{i=1}^n$	**the input data and reconstructed data**
$Y^k = \{y_i^k\}_{i=1}^n$	**the k-th layer hidden representation**
$W^k, \hat{W}^k$	**the k-th layer weight matrix**
$b^k, \hat{b}^k$	**the k-th layer biases**
$\theta^{(k)} = \{W^k, \hat{W}^k, b^k, \hat{b}^k\}$	**the overall parameters**

where $y_i^{(K)}$ is the deep representation of graph $G_i \in \mathbf{G}$. The deep representation $\mathrm{y}_i^{(K)}$ of each graph G_i can effectively capture the manifold structure of the graph space through the the autorencoder network.

With the deep representation $y_i^{(K)} \in \mathbf{DRs}$ of each graph $G_i \in \mathbf{G}$ to hand, we compute a deep graph kernel k for each pair of graphs $G_i \in \mathbf{G}$ and $G_j \in \mathbf{G}$ as

$$k_{\mathrm{DP}} = \langle y_i^{(K)}, y_j^{(K)} \rangle, \tag{15}$$

i.e., the dot product between the deep representations $y_i^{(K)}$ and $y_j^{(K)}$ of G_i and G_j. The resulting deep hybrid graph kernel based on the depth-based matching kernel and the Weisfeiler-Lehman subtree kernel is defined as

$$k_{\mathrm{DHK}} = k_{\mathrm{DP}} + k_{\mathrm{WL}}^{(M)} + k_{\mathrm{DB}}^{(K)}, \tag{16}$$

i.e., the deep hybrid kernel consists of three basic kernel measures. Specifically, $k_{\mathrm{WL}}^{(M)}$ reflects the intrinsic structure information through the subtrees rooted at each vertex with increasing heights that vary from 1 to M. $k_{\mathrm{DB}}^{(K)}$ is based on the structural correspondence information between substructures identified through the K-layer depth-based representations rooted at each vertex. On the other hand, the deep graph kernel k_{DP} directly encapsulates the joint kernel-based similarity information of the kernels $k_{\mathrm{WL}}^{(M)}$ and $k_{\mathrm{DB}}^{(K)}$. This is because the deep representations for k_{DP} are computed through the deep autoencoder network, that minimizes the reconstruction error of the output and input kernel-based similarity graph embedding vectors associated with $k_{\mathrm{WL}}^{(M)}$ and $k_{\mathrm{DB}}^{(K)}$. As a result, this is turn allows us to avoid the problem of determining the optimal mixing between the input kernels (as in the hybrid reproducing kernel [16]) by encapsulating this information in k_{DP}.

Finally, note that, to eliminate difference in scales resulting when combining the different graph kernels, we suggest to transform the basic deep kernel k_{DP}, the depth-based matching kernel $k_{\mathrm{DB}}^{(K)}$ and the Weisfeiler-Lehman subtree kernel $k_{\mathrm{WL}}^{(M)}$ in Eq.(13), Eq.(15) and Eq.(16) into their normalized versions. For instance, for a sample kernel $k(i,j)$, its normalized version $k_{\mathrm{nm}}(i,j)$ is defined as

$$k_{\mathrm{nm}} = \frac{k(i,j)}{\sqrt{k(i,i)k(j,j)}}. \tag{17}$$

The resulting normalized kernel value is therefore bounded between 0 and 1.

C. Discussions

The proposed kernel has a number of advantages. First, unlike most state-of-the-art graph kernels [8], [22], [9], [23], [24], [14], [25], [26] that only reflect the similarity information between pairs of graphs, the deep hybrid graph kernel k_{DHK} can encapsulate similarity information between a larger number of graphs. This is because the associated basic deep graph kernel k_{DP} of k_{DHK} is computed as the dot product between the deep representations of the similarity embedding vectors of all graphs. The embedding vector of each graph is computed by measuring the kernel-based similarity between a graph and each prototype graph, and we use all training graphs as prototype graphs. As a result, the deep representation of each graph embedding vector encapsulates the information between

the graph and all the training graphs. Second, unlike other hybrid graph kernels (e.g., the hybrid reproducing kernel [16]) that need to compute the weighted coefficients associated with each combined graph kernel to reflect the relationship between these combined kernels, our deep hybrid graph kernel can smoothly capture the joint relationship between the combined kernels through the deep representations computed through the autoencoder networks. Furthermore, since the deep autoencoder network can significantly capture the manifold structure of the graph embedding space, the resulting deep hybrid graph kernel can reflect richer graph structural information. The above observations suggest the theoretical effectiveness of the proposed deep hybrid graph kernel.

IV. Experimental Results

A. Datasets

MUTAG: The MUTAG dataset consists of graphs representing 188 chemical compounds, and here the goal is to predict whether each compound possesses mutagenicity [27]. The maximum, minimum and average number of vertices are 28, 10 and 17.93 respectively.

PPIs: The PPIs dataset consists of protein-protein interaction networks (PPIs). The graphs describe the interaction relationships between histidine kinase in different species of bacteria. Histidine kinase is a key protein in the development of signal transduction. If two proteins have direct (physical) or indirect (functional) association, they are connected by an edge. There are 219 PPIs in this dataset and they are collected from 5 different kinds of bacteria (i.e., a) *Aquifex*4 and *thermotoga*4 PPIs from *Aquifex aelicus* and *Thermotoga maritima*, b) *Gram-Positive*52 PPIs from *Staphylococcus aureus*, c) *Cyanobacteria*73 PPIs from *Anabaena variabilis*, d) *Proteobacteria*40 PPIs from *Acidovorax avenae*, and e) *Acidobacteria*46 PPIs). The number of maximum, minimum and average vertices for the PPIs dataset are 128, 3 and 109.63 respectively.

Reeb: The SHREC 3D Shape database consists of 15 classes and 20 individuals per class, that is 300 shapes [28]. This is a standard benchmark in 3D shape recognition. From the SHREC 3D Shape database, we establish the Reeb graph datasets through the mapping function: ERG barycenter, i.e., the distance from the center of mass/barycenter. The number of maximum, minimum and average vertices for the Reeb dataset are 220, 41 and 95.42 respectively.

Shock: The Shock dataset consists of graphs from the Shock 2D shape database. Each graph is a skeletal-based representation of the differential structure of the boundary of a 2D shape. There are 150 graphs divided into 10 classes. Each class contains 15 graphs. The number of maximum, minimum and average vertices for the Shock dataset are 33, 4 and 13.16 respectively.

B. Evaluation on Graph Classification

We evaluate the performance of our kernels on several standard graph datasets, including MUTAG, Reeb, PPIs and Shock.

Experimental Setup: We evaluate the performance of the deep hybrid graph kernel (DHGK) on graph classification problems. We compare our kernel with several alternative state-of-the-art graph kernels. These graph kernels include 1) the depth-based matching kernel (DBMK) [1], 2) the Weisfeiler-Lehman subtree kernel (WLSK) [2], 3) the shortest path graph kernel (SPGK) [7], 4) the graphlet count graph kernel with graphlet of size 4 (GCGK) [29], and 5) the hybrid reproducing graph kernel (HRGK) [16]. For the DHGK kernel, we set the parameters $K = 10$ (i.e., the greatest subgraph layer associated with the combined DBMK kernel) and $M = 10$ (i.e., the greatest subtree height associated with the combined WLSK kernel), because of the good performance of the kernels with these parameters based on the works in [1], [2]. Similarly, for the independent DBMK and WLSK kernels, we set both their own associated parameters M and K as 10. Moreover, for the associated multi-layer deep autoencoder network of the proposed HRGK kernel, we set the dimension of each layer is 600, 400, 200 and 100, i.e., the associated encoder and decoder network both have 4 layer learning structures. The classification performance of each kernel is evaluated using 10-fold cross-validation and a C-Support Vector Machine (C-SVM). Moreover, we employ the LIBSVM library [30]. For each fold, we choose the parameters of each kernel together with the C parameter of the C-SVM by cross-validation on the training data. For each kernel and dataset, we repeat the whole experiment 10 times and we compute the average classification accuracy and standard error. The average classification accuracies ($\pm$ standard error) are shown in Table II.

Results and Discussions: In terms of classification accuracy, we can see that the proposed DHGK kernel can outperform the alternative graph kernels, with the exception of the WLSK kernel on the PPIs dataset. However, the proposed DHGK kernel is still competitive to the WLSK kernel on the PPIis dataset. The effectiveness of the proposed DHGK kernel are threefold. First, unlike the remaining graph kernels that only reflect the information between pairwise graphs, the proposed DHGK kernel can reflect comprehensive information over all graphs under investigations. This is because the associated deep basic kernel for the DHGK kernel is computed through a kernel-based embedding vector, and this embedding vector of each graph is computed based on measuring the kernel value between the graph and each of the training graphs. Second, unlike the HRGK kernel, which is also a kind of hybrid graph kernel, only the proposed DHGK kernel can jointly capture the relationship between the combined kernels through the deep autoencoder network. Furthermore, since the deep network can capture the manifold structure of kernel-based graph embedding space, the proposed DHGK kernel can reflect richer graph characteristics. Third, the proposed DHGK kernel can simultaneously encapsulate the joint information and the independent information from the DBMK and WLSK kernels, thus leading to a significant improvement over the original kernels.

V. Conclusions

In this paper, we have developed a new deep hybrid graph kernel based on the depth-based matching kernel and the Weisfeiler- Lehman subtree kernel. Our main idea is based on jointly computing a basic deep kernel that simultaneously captures the relationship between the combined kernels through a deep autoencoder network. We have shown that the deep hybrid graph kernel not only captures the joint information

TABLE II. CLASSIFICATION ACCURACY (IN % ± STANDARD ERROR).

Datasets	MUTAG	Reeb	PPIs	Shock
DHGK	**88.05** ± .61	**69.73** ± .51	87.61 ± .87	**42.13** ± .55
DBMK	85.27 ± .69	69.40 ± .56	83.74 ± .73	35.20 ± .62
WLSK	82.05 ± .57	58.53 ± .53	**88.07** ± .41	36.40 ± 1.00
SPGK	83.38 ± .81	55.73 ± .44	59.04 ± .44	37.88 ± .93
GCGK	82.04 ± .39	23.40 ± .60	46.61 ± .47	26.93 ± .63
HRGK	84.35 ± .71	32.71 ± .51	53.61 ± .23	26.93 ± .63

between the associated depth-based matching and Weisfeiler-Lehman subtree kernels, but also reflects the similarity information between each graph and the set of training graphs. Experimental evaluations demonstrate the effectiveness of the proposed kernel. Overall, the proposed work provides us a new way of defining new hybrid graph kernel. Our future work will investigate different hybrid kernels based on other state-of-the-art graph kernels.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant no. 61503422 and 61602535), and the Open Projects Program of National Laboratory of Pattern Recognition. Lu Bai is supported by the program for innovation research in Central University of Finance and Economics.).

REFERENCES

[1] L. Bai, P. Ren, X. Bai, and E. R. Hancock, "A graph kernel from the depth-based representation," in *Proceedings of S+SSPR*, 2014, pp. 1–11.

[2] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels," *Journal of Machine Learning Research*, vol. 12, pp. 2539–2561, 2011.

[3] E. Z. Borzeshi, M. Piccardi, K. Riesen, and H. Bunke, "Discriminative prototype selection methods for graph embedding," *Pattern Recognition*, vol. 46, no. 6, pp. 1648–1657, 2013.

[4] M. Neuhaus and H. Bunke, "Bridging the gap between graph edit distance and kernel machines," *World Scientific*, 2007.

[5] D. Haussler, "Convolution kernels on discrete structures," in *Technical Report UCS-CRL-99-10*, Santa Cruz, CA, USA, 1999.

[6] T. Gärtner, P. Flach, and S. Wrobel, "On graph kernels: hardness results and efficient alternatives," in *Proceedings of COLT*, 2003, pp. 129–143.

[7] K. M. Borgwardt and H.-P. Kriegel, "Shortest-path kernels on graphs," in *Proceedings of the IEEE International Conference on Data Mining*, 2005, pp. 74–81.

[8] N. Kriege and P. Mutzel, "Subgraph matching kernels for attributed graphs," in *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

[9] F. Costa and K. D. Grave, "Fast neighborhood subgraph pairwise distance kernel," in *Proceedings of ICML*, 2010, pp. 255–262.

[10] A. Feragen, N. Kasenburg, J. Petersen, M. de Bruijne, and K. M. Borgwardt, "Scalable kernels for graphs with continuous attributes," in *Proceedings of NIPS*, 2013, pp. 216–224.

[11] M. Neumann, N. Patricia, R. Garnett, and K. Kersting, "Efficient graph kernels by randomization," in *Proceedings of ECML-PKDD*, 2012, pp. 378–393.

[12] M. Neuhaus and H. Bunke, "Edit distance-based kernel functions for structural pattern classification," *Pattern Recognition*, vol. 39, no. 10, pp. 1852–1863, 2006.

[13] H. Fröhlich, J. K. Wegner, F. Sieker, and A. Zell, "Optimal assignment kernels for attributed molecular graphs," in *Proceedings of ICML*, 2005, pp. 225–232.

[14] L. Bai, L. Rossi, Z. Zhang, and E. R. Hancock, "An aligned subtree kernel for weighted graphs," in *Proceedings of ICML*, 2015, pp. 30–39.

[15] L. Bai and E. R. Hancock, "Depth-based complexity traces of graphs," *Pattern Recognition*, vol. 47, no. 3, pp. 1172–1186, 2014.

[16] L. Xu, X. Jiang, L. Bai, J. Xiao, and B. Luo, "A hybrid reproducing graph kernel based on information entropy," *Pattern Recognition*, vol. 73, pp. 89–98, 2018.

[17] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[18] L. Bai and E. R. Hancock, "Graph kernels from the jensen-shannon divergence," *Journal of Mathematical Imaging and Vision*, vol. 47, no. 1-2, pp. 60–69, 2013.

[19] B. Weisfeiler and A. Lehman, "A reduction of a graph to a canonical form and an algebra arising during this reduction," *Nauchno-Technicheskaya Informatsia*, vol. Ser.2, no. 9, 1968.

[20] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of KDD*, 2016, pp. 1225–1234.

[21] C. Hong, J. Yu, J. Wan, D. Tao, and M. Wang, "Multimodal deep autoencoder for human pose recovery," *IEEE Trans. Image Processing*, vol. 24, no. 12, pp. 5659–5670, 2015.

[22] H. Kashima, K. Tsuda, and A. Inokuchi, "Marginalized kernels between labeled graphs," in *Proceedings of ICML*, 2003, pp. 321–328.

[23] F. R. Bach, "Graph kernels between point clouds," in *Proceedings of ICML*, 2008, pp. 25–32.

[24] Z. Harchaoui and F. Bach, "Image classification with segmentation graph kernels," in *Proceedings of CVPR*, 2007.

[25] L. Bai, P. Ren, and E. R. Hancock, "A hypergraph kernel from isomorphism tests," in *22nd International Conference on Pattern Recognition, ICPR 2014, Stockholm, Sweden, August 24-28, 2014*, 2014, pp. 3880–3885.

[26] L. Bai and E. R. Hancock, "Fast depth-based subgraph kernels for unattributed graphs," *Pattern Recognition*, vol. 50, pp. 233–245, 2016.

[27] A. Debnath, R. L. de Compadre, G. Debnath, A. Shusterman, and C. Hansch, "Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds, correlation with molecular orbital energies and hydrophobicity," *J. of Med. Chem.*, vol. 34, pp. 786–979, 1991.

[28] S. Biasotti, S. Marini, M. Mortara, G. Patanè, M. Spagnuolo, and B. Falcidieno, "3d shape matching through topological structures," in *Proceedings of DGCI*, 2003, pp. 194–203.

[29] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt, "Efficient graphlet kernels for large graph comparison," *Journal of Machine Learning Research*, vol. 5, pp. 488–495, 2009.

[30] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm*, 2011.