Attributed Graph Kernels Using the Jensen-Tsallis q-Differences

Lu Bai[1], Luca Rossi[2], Horst Bunke[3], and Edwin R. Hancock[1,⋆]

[1] Department of Computer Science, University of York
Deramore Lane, Heslington, York, YO10 5GH, UK
[2] School of Computer Science, University of Birmingham
Edgbaston, Birmingham, B15 2TT, UK
[3] Institute of Computer Science and Applied Mathematics
University of Bern, Switzerland

Abstract. We propose a family of attributed graph kernels based on mutual information measures, i.e., the Jensen-Tsallis (JT) q-differences (for $q \in [1, 2]$) between probability distributions over the graphs. To this end, we first assign a probability to each vertex of the graph through a continuous-time quantum walk (CTQW). We then adopt the tree-index approach [1] to strengthen the original vertex labels, and we show how the CTQW can induce a probability distribution over these strengthened labels. We show that our JT kernel (for $q = 1$) overcomes the shortcoming of discarding non-isomorphic substructures arising in the R-convolution kernels. Moreover, we prove that the proposed JT kernels generalize the Jensen-Shannon graph kernel [2] (for $q = 1$) and the classical subtree kernel [3] (for $q = 2$), respectively. Experimental evaluations demonstrate the effectiveness and efficiency of the JT kernels.

Keywords: Graph kernels, tree-index method, continuous-time quantum walk, Jensen-Tsallis q-differences.

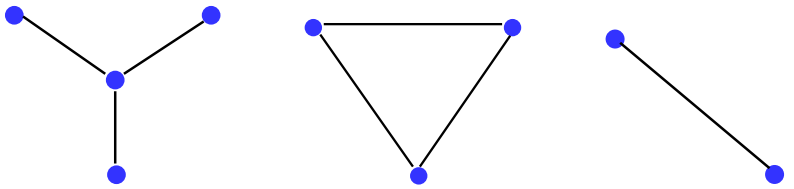
1 Introduction

There has recently been an increasing interest in evolving graph kernels into kernel machines (e.g., a Support Vector Machine) for graph classification [4, 5]. A graph kernel is usually defined in terms of a similarity measure between graphs. Most of the recently introduced graph kernels are in fact instances of the generic R-convolution kernel proposed by Haussler [6]. For a pair of graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, suppose $\{\mathcal{S}_{1;1}, \ldots, \mathcal{S}_{1;n_1}, \ldots, \mathcal{S}_{1;N_1}\}$ and $\{\mathcal{S}_{2;1}, \ldots, \mathcal{S}_{2;n_2}, \ldots, \mathcal{S}_{q;N_2}\}$ are the sets of the substructures of G_1 and G_2 respectively. An R-convolution kernel k_R between G_1 and G_2 can be defined as

$$k_R(G_1, G_2) = \sum_{n_1=1}^{N_1} \sum_{y=1}^{N_2} \delta(\mathcal{S}_{1;n_1}, \mathcal{S}_{2;n_2}), \tag{1}$$

⋆ Edwin R. Hancock is supported by a Royal Society Wolfson Research Merit Award.

T. Calders et al. (Eds.): ECML PKDD 2014, Part I, LNCS 8724, pp. 99–114, 2014.



(a) Graph G_1 (b) Graph G_2 (c) Graph G_3

Fig. 1. Example Graphs

where we have that

$$\delta(\mathcal{S}_{1;n_1}, \mathcal{S}_{2;n_2}) = \begin{cases} 1 \text{ if } \mathcal{S}_{1;n_1} \simeq \mathcal{S}_{2;n_2}, \\ 0 \text{ otherwise,} \end{cases} \tag{2}$$

δ is the Dirac kernel, i.e., it is 1 if the arguments are equal and 0 otherwise, and $\mathcal{S}_{1;n_1} \simeq \mathcal{S}_{2;n_2}$ indicates that $\mathcal{S}_{1;n_1}$ is isomorphic to $\mathcal{S}_{2;n_2}$. k_R is a positive definite kernel.

The existing R-convolution graph kernels can be categorized into three classes, namely the graph kernels based on comparing all pairs of a) walks (e.g., the random walk kernel [7]), b) paths (e.g., the shortest path kernel [8]), and c) restricted subgraph or subtree structures (e.g., the subtree or subgraph kernel [9–11]). Unfortunately, there are two main problems arising in the R-convolution kernels. First, Eq.(1) indicates that the R-convolution kernels only enumerate the pairs of isomorphic substructures. As a result, the substructures which are not isomorphic are discarded. For instance, for the three graphs shown in Fig.1 (a), (b) and (c), the pair of graphs G_1 and G_3 and the pair of graphs G_2 and G_3 both share three same pairs of isomorphic substructures (i.e., pairwise vertices connected by an edge). The R-convolution kernels only count the number of pairs of isomorphic substructures. As a result, the kernel value for G_1 and G_3 is the same as that for G_2 and G_3, though the graphs G_1 and G_2 are structurally different. Second, Eq.(2) indicates that the R-convolution kernels simply record whether two substructures are isomorphic. As a result, the kernels do not reflect any other potential information between these substructures. These drawbacks clearly limit the accuracy of the similarity measure (i.e., the kernel value) between a pair of graphs.

Recently, Bai and Hancock [2] have developed an alternative kernel, namely the Jensen-Shannon (JS) graph kernel, by measuring the Jensen-Shannon divergence (JSD) for a pair of graphs. The JSD is a dissimilarity measure between probability distributions in terms of the nonextensive entropy difference associated with them. The JSD between a pair of graphs is defined in terms of the difference between the entropy of a composite graph and the entropies of the individual graphs. Unlike the R-convolution kernels, the entropy associated with a probability distribution of an individual graph can be computed without decomposing the graph. As a result, the computation of the JS graph kernel avoids the burdensome computation of comparing all the substructure pairs. Unfortunately, the JS graph kernel only captures the global similarity between a pair of graphs, and thus lacks information on the interior topology of the graphs. Moreover, the required composite entropy is computed from a composite structure which does not reflect the correspondence information between the original graphs. Finally, this kernel is restricted to non-attributed graphs. As a summary of the existing kernels

(i.e., the R-convolution and JS graph kernels), it is fair to say that developing efficient and effective graph kernels still remains an open challenge.

To overcome the shortcomings of existing graph kernels, we aim to propose novel kernels for attributed graphs where we make use of the JT q-differences. In information theory, the JT q-difference [12] is a nonextensive measure between probability distributions over structure data. Moreover, the JT q-differences generalize the Hadamard-Schur (element wise) product (for $q = 0$), the classical Jensen-Shannon divergence (JSD) (for $q = 1$), and the inner product (for $q = 2$). To compute the JT q-differences between attributed graphs, we commence by performing a CTQW on each graph to assign each vertex a time-averaged probability. The reasons for using the CTQW are twofold (see details in Sec. 2.2). First, the CTQW reduces the tottering effect arising in classical random walks [13]. Second, the CTQW offers us a richer structure than the classical random walk. We then apply the tree-index (TI) method [1] on each graph to strengthen the vertex labels. At each iteration h, we compute the probability of a strengthened label by summing the probabilities of the vertices having the same label, and then obtain a probability distribution for each graph. With the probability distributions for a pair of attributed graphs to hand, the JT kernels between the graphs are computed in terms of the JT q-differences. As a result, our kernels reflect the similarity between the probability distributions over the global graphs, while also capturing the local correspondence information between the substructures.

Note that, in this paper, we only consider $q = 1$ or 2 for the JT q-differences. The reasons for this are twofold. First, we show that our JT kernel (for $q = 1$) not only generalizes the JS graph kernel [2], but it also overcomes the shortcomings of the JS graph kernel. We also show that the JT kernel overcomes the shortcoming of discarding non-isomorphic substructures that arises in R-convolution kernels. Finally, we show that our JT kernel (for $q = 2$) generalizes the R-convolution kernels based on subtrees. The remainder of this paper is organized as follows: Section 2 reviews the TI method and the CTQW, Section 3 gives the definition of our new kernels, Section 4 provides experimental evaluation and Section 5 concludes the work.

2 Preliminary Concepts

In this section, we review some preliminary concepts which will be used in this work. We commence by introducing a TI method for strengthening the vertex label. Finally, we show how to assign a probability to a vertex of a graph by performing the CTQW.

2.1 A TI Method for Strengthening Vertex Labels

In this subsection, we introduce the TI method of Dahm et al. [1] for strengthening the vertex label of a graph. Given an attributed graph $G(V, E)$, let the label of a vertex $v \in V$ be denoted as $f(v)$. Using the TI method, the new strengthened label for v at the iteration h is defined as

$$TI_h(v) = \begin{cases} f(v) & \text{if } h = 0, \\ \cup_u \{TI_{h-1}(u)\} & \text{otherwise,} \end{cases} \tag{3}$$

where $u \in V$ is adjacent to v. At each iteration h, the TI method takes the union of neighbouring vertex label lists from the last iteration as a new label list for v (the initial step is identical to listing). This creates an iteratively deeper list corresponding to a subtree rooted at v of height h. An example of how the TI method defined in Eq.(3) strengthens the vertex label is shown in Fig.2. In this example, the initialized vertex labels for vertices A to E are their corresponding vertex degrees, i.e., 1, 2, 3, 2 and 2 respectively. Using the TI method, the second iteration indicates the strengthened labels for vertices A to E as $\{\{1,3\}\}$, $\{\{2\},\{2,2,2\}\}$, $\{\{1,3\},\{2,3\},\{2,3\}\}$,$\{\{2,2,2\},\{2,3\}\}$, and $\{\{2,2,2\},\{2,3\}\}$ respectively.

Unfortunately, Fig.2 indicates that the above procedure clearly leads to a rapid explosion of the labels length. Moreover, strengthening a vertex label by only taking the union of the neighbouring label lists also ignores the original label information of the vertex. To overcome these problems, at each iteration h we propose to strengthen the label of a vertex as a new label list by taking the union of both the original vertex label and its neighbouring vertex labels. We use a Hash function to compress the strengthened label list into a new short label. The pseudocode of the re-defined TI algorithm is shown in Algorithm 1, where the neighbourhood of a vertex $v \in V$ is denoted as $\mathcal{N}(v) = \{u|(v,u) \in E\}$.

Algorithm 1. Vertex labels strengthening procedure

1: Initialization.

- Input an attributed graph $G(V,E)$.
- Set ***h=0***. For a vertex $v \in V$, assign the original label $f(v)$ as the initial label $\mathcal{L}_h(v)$.

2: Update the label for each vertex.

- Set ***h=h+1***. For each vertex $v \in G$, assign it a new strengthened label list as

$$\mathcal{L}_h(v) = \cup_{u \in \mathcal{N}(v)} \{\mathcal{L}_{h-1}(u), \mathcal{L}_{h-1}(v)\}. \tag{4}$$

  Note that, $\mathcal{L}_{h-1}(v)$ is at the end of the label list $\mathcal{L}_h(v)$, and $\mathcal{L}_{h-1}(u)$ is arranged as ascending order.

3: For each vertex, compress its strengthened label list into a new short label.

- Using the Hash function $\mathbf{H} : \mathcal{L} \to \Sigma$, compress the label list $\mathcal{L}_h(v)$ into a new short label for each vertex v as

$$\mathcal{L}_h(v) = \mathbf{H}(\mathcal{L}_h(v)). \tag{5}$$

4: Check h.

- Check h. Repeat steps 2, 3 and 4 until the iteration h achieves an expected value.

Note that, in step 4 we use the same function $\mathbf{H}$ for any graph. This guarantees that all the identical labels of different graphs are mapped into the same number.

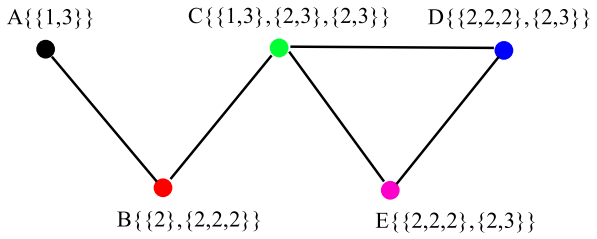


Fig. 2. Example of strengthened labels

2.2 Vertex Probabilities from The CTQW

In this subsection, we show how to assign a probability to a vertex of a graph by performing a CTQW. The CTQW is the quantum analogue of the classical continuous-time random walk (CTRW) [14]. Similarly to the CTRW, the state space of the CTQW is the vertex set V of a graph $G(V, E)$. However, unlike the CTRW, where the state vector is real-valued and the evolution is governed by a doubly stochastic matrix, the state vector of the CTQW is complex-valued and its evolution is governed by a time-varying unitary matrix. Hence the evolution of the CTQW is reversible, which implies that the CTQW is non-ergodic and does not possess a limiting distribution. As a result, the CTQW possesses a number of interesting properties not exhibited in the CTRW. One notable consequence of this is that the problem of tottering of the CTRW is naturally reduced [13]. Furthermore, the quantum walk has been shown to successfully capture the topological information in a graph structure [15–18]. Note also that, contrary to the CTRW and its non-backtracking counterparts [19–21], the CTQW is not dominated by the low frequency of the Laplacian spectrum, and thus is potentially able to discriminate better among different graph structures.

Using the Dirac notation, we define the basis state corresponding to the CTQW being at a vertex $u \in V$ as $|u\rangle$, where $|.\rangle$ denotes an orthonormal vector in a n-dimensional complex-valued Hilbert space $\mathcal{H}$. A general state of the CTQW is a complex linear combination of the basis states $|u\rangle$, i.e., an amplitude vector, such that the state of the CTQW at time t is

$$|\psi_t\rangle = \sum_{u \in V} \alpha_u(t)\,|u\rangle\,, \tag{6}$$

where the amplitudes $\alpha_u(t) \in \mathbb{C}$. The probability of the CTQW visiting a vertex $u \in V$ at time t is

$$\Pr(X^t = u) = \alpha_u(t)\alpha_u^*(t), \tag{7}$$

where $\alpha_u^*(t)$ is the complex conjugate of $\alpha_u(t)$. For all $u \in V$, $t \in \mathbb{R}^+$, we have $\sum_{u \in V} \alpha_u(t)\alpha_u^*(t) = 1$ and $\alpha_u(t)\alpha_u^*(t) \in [0, 1]$. Note that there is no restriction on the sign or phase of the amplitudes and this allows destructive and constructive interference to take place during the evolution of the walk. These interference patterns are responsible for the faster hitting times [13] and the ability of the CTQW to capture the presence of particular structural patterns in the graph [16]. Note also that when the quantum walk backtracks on an edge it does so with opposite phase, thus creating destructive inter fence which reduces the problem of tottering of classical random walks [13].

Let A be the adjacency matrix of G, then the degree matrix D is a diagonal matrix whose elements are given by $D(u,u) = d_u = \sum_{v \in V} A(u,v)$, where d_u is the degree of u. We compute the Laplacian matrix as $L = D - A$. The spectral decomposition $L = \Phi^\top \Lambda \Phi$ of the Laplacian matrix L is given by the diagonal matrix $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_{|V|})$ with the ordered eigenvalues as elements $(\lambda_1 < \lambda_2 < ... < \lambda_{|V|})$ and the matrix $\Phi = (\phi_1 | \phi_2 | ... | \phi_{|V|})$ with the corresponding ordered orthonormal eigenvectors as columns.

Given an initial state $|\psi_0\rangle$, the Schrödinger equation gives us the state of the walk at time t, i.e.,

$$|\psi_t\rangle = \Phi^\top e^{-i\Lambda t} \Phi |\psi_0\rangle . \tag{8}$$

In this work we propose to let the initial amplitude be proportional to $D(u,u)$, i.e.,

$$\alpha_u(0) = \sqrt{D(u,u) / \sum D(u,u)}. \tag{9}$$

Note that, on the other hand, choosing a uniform distribution over the vertices of G would result in the system remaining stationary, as the initial state vector would be an eigenvector of the Laplacian and thus a stationary state of the walk.

In quantum mechanics, a state $|\psi_t\rangle$ is called a pure state. In general, however, we deal with mixed states, i.e., a statistical ensemble of pure states $|\psi_t\rangle$, each with probability p_t. The density matrix associated with such a system is defined as $\rho = \sum_t p_t |\psi_t\rangle \langle\psi_t|$, where $|\psi_t\rangle \langle\psi_t|$ denotes the outer product between $|\psi_t\rangle$ and its conjugate transpose. Let $|\psi_t\rangle$ be the state corresponding to the CTQW that has evolved from $|\psi_0\rangle$ defined as in Eq.(9) until a time t. Using Eq.(8), we can define the time-averaged density matrix (i.e., mixed density matrix) ρ_G^T for G as

$$\rho_G^T = \frac{1}{T} \int_0^T \Phi^\top e^{-i\Lambda t} \Phi |\psi_0\rangle \langle\psi_0| \Phi^\top e^{i\Lambda t} \Phi \, \mathrm{d}t. \tag{10}$$

Let ϕ_{ra} and ϕ_{cb} denote the (ra)-th and (cb)-th elements of the matrix of eigenvectors Φ of the Laplacian matrix L. When we let $T \to \infty$, Rossi et al. [16] have shown that the (r,c)-th element of ρ_G^∞ can be computed as

$$\rho_G^\infty(r,c) = \sum_{\lambda \in \tilde{\Lambda}} \sum_{a \in B_\lambda} \sum_{b \in B_\lambda} \phi_{ra} \phi_{cb} \bar{\psi}_a \bar{\psi}_b, \tag{11}$$

where $\tilde{\Lambda}$ is the set of distinct eigenvalues of the Laplacian matrix L, and B_λ is a basis of the eigenspace associated with λ. Eq.(11) indicates that the mixed density matrix ρ_G^T relies on computing the eigendecomposition of G, and thus has time complexity $O(|V|^3)$.

The mixed density matrix ρ_G^∞ is a $|V| \times |V|$ matrix. Note that, for a graph G, the time-averaged probability of the CTQW to visit a vertex $v \in V$ at time $T \to \infty$ is

$$p_Q(v) = \rho_G^\infty(v,v). \tag{12}$$

Interestingly, despite the non-stationary behaviour of the CTQW, we have that as $T \to \infty$ the time-averaged probability converges. Also, Eq.(8) indicates that the evolution of the CTQW relies on the spectral decomposition of the graph Laplacian, which in turn encapsulates rich interior graph information. Thus, the probability $p_Q(v)$ reflects the interior topology information of the graph.

3 Graph Kernels from JT q-differences

In this section, we define a family of novel graph kernels from JT q-differences (for $q = 1$ and 2). The JT q-difference is a dissimilarity measure between probability distributions [12]. Consider the graphs $G_y(V_y, E_y)$ where $y \in Y = \{1, 2\}$ and $x \in X = \{l_1, \ldots, l_{|X|}\}$ is the label of a vertex $v_y \in V_y$. X is a label set which contains any possible vertex label. Let $\mathbf{P}_y = \{p_{y1}, \ldots, p_{y|X|}\}$, where $p_{yx} = P(X = x|Y = y)$ is the probability distribution over vertex labels of G_y. $\pi_y \in \pi = \{\pi_1, \pi_2\}$ is the weight associated with $\mathbf{P}_y$, such that $\pi_y \geq 0$ and $\sum_{y \in Y} \pi_y = 1$. $P(X|Y)$ is the joint probability distribution for the two variables X and Y. P_X and P_Y are two probability distributions over $x \in X$ and $y \in Y$ respectively. We define the JT q-differences (for q=$\{1,2\}$) between the probability distributions $\mathbf{P}_1$ and $\mathbf{P}_2$ (i.e., $\mathbf{P}_{y=1}$ and $\mathbf{P}_{y=2}$) as

$$JT_q^{\pi}(\mathbf{P}_1, \mathbf{P}_2) = S_q(X) - \int_{y \in Y} \pi_y^q S_q(X|Y) = S_q(X) - S_q(X|Y), \tag{13}$$

where $S_q(.)$ is the Tsallis entropy [22] and

$$S_q(X) = \frac{k}{q-1}(1 - \sum_{x \in X} 2P_X(x)^q), \tag{14}$$

and

$$S_q(X|Y) = S_q(X) + S_q(Y) - (q-1)S_q(X)S_q(Y). \tag{15}$$

By letting $k = 1$ [12], Eq.(14) can be simplified as

$$S_q(X) = -\sum_{x \in X} P_X(x)^q \ln_q P_X(x), \tag{16}$$

where $\ln_q(P_X(x)) = (P_X(x)^{(1-q)} - 1)/(1-q)$ is the q-logarithm function introduced by Tsallis [22]. Let $\pi_1 = \pi_2 = 1/2$. We define the JT q-difference for the pair of graphs G_1 and G_2 by re-writing Eq.(13) as

$$JT_q(G_1, G_2) = JT_q^{1/2,1/2}(\mathbf{P}_1, \mathbf{P}_2) = S_q(\frac{\mathbf{P}_1 + \mathbf{P}_2}{2}) - \frac{S_q(\mathbf{P}_1) + S_q(\mathbf{P}_2)}{2^q}. \tag{17}$$

Note that Furuichi [23] has defined the Tsallis MI as $I_q(X;Y) = S_q(X) - S_q(X|Y) = I_q(Y;X)$, which is a nonextensive measure. Thus, for the graphs G_1 and G_2, the JT q-difference $JT_q(G_1, G_2)$ not only reflects the dissimilarity between their probability distributions $\mathbf{P_1}$ and $\mathbf{P_2}$, but also measures the MI (i.e., the mutual dependence) between the graphs and their probability distributions. In other words, $JT_q(G_1, G_2)$ reflects the dissimilarity over the global graphs. By contrast, the R-convolution kernels measure the (dis)similarity in terms of the substructures.

3.1 The JT Graph Kernel

In this subsection, we define a family of JT graph kernels using the JT q-difference (for $q = 1$ and 2). For a graph $G(V, E)$, we commence by performing a CTQW (for $T = \infty$)

on G, and we associate with each vertex $v \in V$ the time-averaged probability $p_Q(v)$. At each iteration h, we strengthen the vertex labels using Algorithm 1. The probability of the label $x \in X$ for G at iteration h is $p_x^h = \sum_{v \in V} p_Q(v)$, where each vertex v satisfies $\mathcal{L}_h(v) = x$. We thus obtain a probability distribution over the vertex labels of G as $\mathbf{P^h} = \{p_1^h, \ldots, p_{|X|}^h\}$. For a pair of graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, we compute the probability distributions $\mathbf{P_1^h} = \{p_{11}^h, \ldots, p_{1|X|}^h\}$ and $\mathbf{P_2^h} = \{p_{21}^h, \ldots, p_{2|X|}^h\}$ over the vertex labels through the CTQW. Based on Eq.(17), the JT q-difference for G_1 and G_2 at iteration h is defined as

$$JT_q^h(G_1, G_2) = JT_q^{1/2,1/2}(\mathbf{P_1^h}, \mathbf{P_2^h}). \tag{18}$$

Definition (Jensen-Tsallis Graph Kernel). The kernel $k_{JT}^{(q,H)} : G_1 \times G_2 \longrightarrow R^+$ for the graphs G_1 and G_2 is

$$k_{JT}^{(q,H)}(G_1, G_2) = \sum_{h=0}^{H} \exp\{-\lambda JT_q^h(G_1, G_2)\} = \sum_{h=0}^{H} \exp\{-\lambda JT_q^{1/2,1/2}(\mathbf{P_1^h}, \mathbf{P_2^h})\}, \tag{19}$$

where H is the largest number of TI iterations, $q = 1$ or 2, and $0 < \lambda \leq 1$ is a decay factor. Here, λ is used to ensure that the large value dose not tend to dominate the kernel value. Based on our experimental evaluation, the different values of λ do not influence the performance of our JT kernels. Thus, in this work we decide to set λ to 1. □

Lemma. *The JT graph kernel is positive definite (**pd**).*

Proof. This follows the definition in [24]. In fact, if a similarity or dissimilarity measure $s_G(G_1, G_2)$ between a pair of graphs G_1 and G_2 is symmetrical, then a diffusion kernel $k_s = \exp(\lambda s_G(G_1, G_2))$ or $k_s = \exp(-\lambda s_G(G_1, G_2))$ associated with the similarity or dissimilarity measure $s_G(G_1, G_2)$ is **pd**. As a result, the JT kernel $k_{JT}^{(q,H)}$ is the sum of several **pd** kernels in terms of the exponentiated JT q-difference, and is also **pd**. □

Reisen and Bunke [24] observed that in a diffusion kernel the exponentiation enhances the (dis)similarity between the graphs. Thus, the kernel $k_{JT}^{(q,H)}$ enhances the similarity measure between the graphs.

3.2 Relation to State of the Art Graph Kernels

Proposition 1. When $q = 2$, the JT graph kernel generalizes the R-convolution kernels based on subtrees. □

Proof. We verify this proposition by revealing the relationship between the JT kernel and the classical subtree kernel [3]. For a graph $G(V, E)$, the strengthened label $\mathcal{L}_h(v)$, which is defined in Eq.(5), corresponds to a subtree of height h rooted at a vertex $v \in V$. For a pair of graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, let $v_1 \in V_1$ and $v_2 \in V_2$ denote a pair of vertices. If $\mathcal{L}_h(v_1) = \mathcal{L}_h(v_2)$, the subtrees of height h rooted at v_1 and v_2 are isomorphic. Thus, a subtree kernel k_{st}^H for G_1 and G_2 can be defined as

$$k_{st}^H(G_1, G_2) = \sum_{h=0}^{H} k_{st}^h(G_1, G_2) = \sum_{h=0}^{H} \sum_{v_1 \in V_1} \sum_{v_2 \in V_2} \delta\{\mathcal{L}_h(v_1), \mathcal{L}_h(v_2)\}, \tag{20}$$

where H is the largest iteration h for Algorithm 1. δ is a Dirac kernel, that is, it is 1 when its arguments are equal and 0 otherwise. Clearly, the subtree kernel k_{st}^H counts the number of all the pairwise isomorphic subtrees identified by the TI method. k_{st}^h is a base subtree kernel counting the number of pairwise isomorphic subtrees of height h.

As a result, the base subtree kernel k_{st}^h can be defined by an inner product $\langle \cdot, \cdot \rangle$, i.e., a linear kernel. For the graph G, let $n^h(x)$ denote the number of vertices having the same label $x \in X$ at iteration h. Thus G can be represented by the feature vector $FV^h = \{n^h(l_1), \ldots, n^h(l_{|X|})\}^\top$. Given $FV_1^h = \{n_1^h(l_1), \ldots, n_1^h(l_{|X|})\}^\top$ and $FV_2^h = \{n_2^h(l_1), \ldots, n_2^h(l_{|X|})\}^\top$, the kernel k_{st}^h between G_1 and G_2 at iteration h can be defined as

$$k_{st}^h(G_1, G_2) = \langle FV_1^h, FV_2^h \rangle. \tag{21}$$

Martin et al. [12] have observed that the JT q-difference is related to the inner product when $q = 2$, i.e., $JT_2^{1/2,1/2}(\mathbf{P_1^h}, \mathbf{P_2^h}) = 1/2 - 1/2\langle \mathbf{P_1^h}, \mathbf{P_2^h} \rangle$. Simply, we have $JT_2^{1/2,1/2}(\mathbf{P_1^h}, \mathbf{P_2^h}) = -\langle \mathbf{P_1^h}, \mathbf{P_2^h} \rangle$. As a result, the JT kernel $k_{JT}^{(2,H)}$ ($\lambda = 1$) can be re-written as

$$k_{JT}^{(2,H)}(G_1, G_2) = \sum_{h=0}^{H} \exp\{\langle \mathbf{P_1^h}, \mathbf{P_2^h} \rangle\}, \tag{22}$$

For a graph G, the probability of a label $x \in X$ at the iteration h is $p_x^h = \sum_{v \in V} p_Q(v)$, where each vertex v satisfies $\mathcal{L}_h(v) = x$ and $p_Q(v)$ is the time-averaged probability of a CTQW visiting v. Given G and its associated feature vector FV^h with elements $n^h(x)$, if we compute the probability of a label x instead of computing the frequency $n^h(x)$ of x (i.e., we compute the probability for a class of isomorphic subtrees which correspond to the label x, instead of computing the number of these subtrees), we re-write FV^h as $FV_p^h = \mathbf{P}^h$. For the graphs G_1 and G_2, we thus have

$$k_{JT}^{(2,H)}(G_1, G_2) = \sum_{h=0}^{H} \exp\{\langle FV_{p1}^h, FV_{p2}^h \rangle\} = \sum_{h=0}^{H} \exp\{k_{st}^h(G_1, G_2)\}. \tag{23}$$

As a result, the kernel $k_{JT}^{(2;H)}$ can be seen as a generalization of the subtree kernel k_{st}^H where we assign a probability to each class of isomorphic subtrees and then exponentiate the base subtree kernel k_{st}^h at each iteration h. □

Discussions. Prop. 1 and its proof make two interesting observations on the JT kernel $k_{JT}^{(2,H)}$. First, for each pair of graphs the kernel $k_{JT}^{(2,H)}$ computes the probability of a vertex label by summing the probabilities of the vertices having the same label. The probabilities of these vertices are computed by means of a CTQW, whose propagation depends on the interior connections of the graph and thus reflects its interior topology information. For a pair of graphs, the kernel measures the similarity between their pairwise probabilities in terms of matching labels, where the labels correspond to classes of isomorphic subtrees in the graphs. The kernel thus reflects more information among these subtrees rather than only the isomorphism. Second, the $k_{JT}^{(2,H)}$ exponentiates the base subtree kernel k_{st}^h (i.e., the similarity measure between pairwise subtrees of height

h). Thus, the kernel $k_{JT}^{(2,H)}$ enhances the similarity measure between the graphs by exponentiating the kernel k_{st}^{h}.

Unfortunately, the JT kernel $k_{JT}^{(2,H)}$ also suffers from the drawback of discarding non-isomorphic subtrees arising in the subtree kernel. This can be observed from Eq.(23). For a pair of graphs, if a class of isomorphic subtrees are only contained in one graph, the probability for a label corresponding to these subtrees in the other graph is 0. As a result, Eq.(23) cannot reflect the similarity between pairwise probabilities of the labels for the graphs. Below, we will show how the kernel $k_{JT}^{(1,H)}$ solves this problem.

Proposition 2. When $q = 1$, the JT graph kernel generalizes the JS graph kernel. □

Proof. We verify the proposition by revealing the relationship between the JT kernel and the JS graph kernel [2]. For a graph $G(V, E)$ and its degree matrix D, the probability of a classical steady state random walk (CSSRW) visiting a vertex $v \in V$ is

$$p_v = \frac{D(v,v)}{\sum_v D(v,v)} = \frac{\sum_{u\in V} A(u,v)}{\sum_{u\in V}\sum_{v\in V} A(u,v)}. \tag{24}$$

Thus, we can associate to the vertices of G the distribution $\mathbf{P^c} = \{p_1^c, \ldots, p_{|V|}^c\}$. For a pair of graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, we compute the probability distributions $\mathbf{P_1^c} = \{p_{11}^c, \ldots, p_{1|V|}^c\}$ and $\mathbf{P^c} = \{p_{21}^c, \ldots, p_{2|V|}^c\}$ associated with the CSSRW. The JS graph kernel for G_1 and G_2 is defined as

$$k_{JS}(G_1, G_2) = \log 2 - D_{JS}(\mathbf{P_1^c}, \mathbf{P_2^c}), \tag{25}$$

where $D_{JS}(G_1, G_2)$ is the JSD and is defined as

$$D_{JS}(\mathbf{P_1^c}, \mathbf{P_2^c}) = H_S(\mathbf{P_U^c}) - \frac{H_S(\mathbf{P_1^c}) + H_S(\mathbf{P_2^c})}{2}. \tag{26}$$

Here, $H_S(\mathbf{P_1^c})$ is a Shannon entropy associated with the probability distribution $\mathbf{P_1^c}$ and is defined as

$$H_S(\mathbf{P_1^c}) = -\sum_{v\in V} p_{1v}^c \log p_{1v}^c. \tag{27}$$

Moreover, $H_S(\mathbf{P_U^c})$ is the Shannon entropy of the probability distribution from an union graph of G_1 and G_2 (i.e., the composite graph of G_1 and G_2). The union graph can be either a product graph or a disjoint union graph of G_1 and G_2 [2, 25].

For the JT kernel $k_{JT}^{(1,H)}$, Martin et al. [12] have observed that the JT q-difference is related to the JSD when $q = 1$, i.e.,$JT_1^{1/2,1/2}(\mathbf{P_1^h}, \mathbf{P_2^h}) = D_{JS}(\mathbf{P_1^h}, \mathbf{P_2^h})$, where

$$D_{JS} = H_S(\frac{\mathbf{P_1^h} + \mathbf{P_2^h}}{2}) - \frac{H_S(\mathbf{P_1^h}) + H_S(\mathbf{P_2^h})}{2}. \tag{28}$$

Thus, the JT kernel $k_{JT}^{(1,H)}$ ($\lambda = 1$) can be re-written as

$$\begin{aligned} k_{JT}^{(1,H)}(G_1, G_2) &= \sum_{h=0}^{H} \exp\{-D_{JS}(\mathbf{P_1^h}, \mathbf{P_2^h})\} \\ &= \sum_{h=0}^{H} \exp\{H_S(\mathbf{P_1^h}) + H_S(\mathbf{P_2^h}) - H_S(\frac{\mathbf{P_1^h} + \mathbf{P_2^h}}{2})\}. \end{aligned} \tag{29}$$

Here, $H_S(\frac{\mathbf{P_1^h}+\mathbf{P_2^h}}{2})$ is the Shannon entropy of the composite probability distribution $\frac{\mathbf{P_1^h}+\mathbf{P_2^h}}{2}$, and is defined as

$$H_S(\frac{\mathbf{P_1^h}+\mathbf{P_2^h}}{2}) = -\sum_{x=1}^{|X|}\frac{p_{1x}^h+p_{2x}^h}{2}\log\frac{p_{1x}^h+p_{2x}^h}{2}. \quad (30)$$

As a result, the JT kernel $k_{JT}^{(1,H)}$ and the JS graph kernel k_{JS} are both related to the JSD. Through Eq.(25), Eq.(26) and Eq.(29), we observe that the kernel $k_{JT}^{(1,H)}$ generalizes the kernel k_{JS} by three computational steps, i.e., a) assigning each vertex a probability using the CTQW instead of using the CSSRW, b) computing the probability distribution over the vertex labels at each iteration h instead of computing the probability distribution associated with the CSSRW, and c) computing the composite Shannon entropy using the composite probability distribution $\frac{\mathbf{P_1^h}+\mathbf{P_2^h}}{2}$ instead of using the probability distribution $\mathbf{P_U^c}$ from the union graph. Moreover, the JT kernel $k_{JT}^{(1,H)}$ also exponentiates the JSD measure for graphs at each iteration h. □

Discussions. Compared to the JS graph kernel k_{JS}, Prop. 2 and its proof reveal four advantages for the JT kernel $k_{JT}^{(1,H)}$. First, from Eq.(30) we observe that there is correspondence between pairwise discrete probabilities p_{1x}^h and p_{2x}^h through the same label x. As a result, the kernel $k_{JT}^{(1,H)}$ overcomes the shortcoming of lacking correspondence information between pairwise discrete probabilities that arises in the JS graph kernel k_{JS}. Second, since a pair of identical strengthened labels for a pair of vertices correspond to a pair of isomorphic subtrees rooted at the vertices. The kernel $k_{JT}^{(1,H)}$ encapsulates correspondence information between pairwise isomorphic substructures. By contrast, the JS graph kernel k_{JS} does not reflect substructure correspondence. Third, the kernel $k_{JT}^{(1,H)}$ overcomes the restriction on non-attributed graphs for the JS graph kernel k_{JS}. Fourth, similar to the kernel $k_{JT}^{(2,H)}$, the kernel $k_{JT}^{(1,H)}$ also enhances the similarity measure for graphs by exponentiating the JSD (i.e.,the dissimilarity measure between graphs). Furthermore, the evolution of the CTQW relies on the topology information of graphs. Thus, the kernel $k_{JT}^{(1,H)}$ also reflects rich interior topology information of graphs, relying on the CTQW. By contrast, the probability distribution associated with the CSSRW (i.e., the vertex degree distribution) only reflects limited topology information, because the vertex degree of a graph is structurally simple and reflects limited topology information.

Note finally that, through Eq.(29) and Eq.(30) we also observe that all the probabilities of the different labels are used to compute the entropies. We have known that each label corresponds to a subtree. All the strengthened labels of a graph will be used to compute the Shannon entropy. As a result, unlike existing R-convolution kernels which count the number of pairwise isomorphic substructures, the kernel $k_{JT}^{(1,H)}$ incorporates all the identified subtrees into the computation. The kernel $k_{JT}^{(1,H)}$ thus overcomes the shortcoming of discarding non-isomorphic substructures. In other words, the JT kernel $k_{JT}^{(1,H)}$ may distinguish different classes of graphs better than the R-convolution kernels.

3.3 Computational Analysis

For N graphs (each graph has n vertices) and their label set X, computing the $N \times N$ kernel matrix using the JT kernels $k_{JT}^{(1,H)}$ and $k_{JT}^{(2,H)}$ requires time complexity $O(HN^2 n^2 + HN^3 n + Nn^3)$ and $O(HN^2 n^2 + Nn^3)$, respectively. The reasons are explained as follows. a) For both of the kernels $k_{JT}^{(1,H)}$ and $k_{JT}^{(2,H)}$, computing the compressed strengthened labels for a graph at each iteration h $(0 \leq h \leq H)$ needs to visit all the n^2 entries of the adjacency matrix, and thus requires time complexity $O(Hn^2)$ for all the H iterations. b) For both of the kernels $k_{JT}^{(1,H)}$ and $k_{JT}^{(2,H)}$, computing the probabilities for all the vertices of a graph using the CTQW requires time complexity $O(n^3)$, because the CTQW relies on the eigen-decomposition of the graph Laplacian. Computing the probability distribution for a graph requires time complexity $O(HNn^2)$ (for the worst case, i.e., each vertex label for the N graphs at all the H iterations are all different and there thus are NHn different labels in X), because it needs to visit all the HNn entries in X for the n vertices. c) For the kernel $k_{JT}^{(1,H)}$, computing the $N \times N$ kernel matrix requires time complexity $O(HN^3 n)$, because the Tsallis entropy $S_q(.)$ for each pair of graphs requires time complexity $O(HNn)$. On the other hand, for the kernel $k_{JT}^{(2,H)}$, computing the $N \times N$ kernel matrix only requires time complexity $O(HN^2)$, because Eqs.(22) and (23) indicate that $k_{JT}^{(2,H)}$ can directly compute the kernel value for a pair of graphs by computing the inner product of their probability distributions. In other words, $k_{JT}^{(2,H)}$ does not need to compute the Tsallis entropy for each pair of graphs. As a result, the complete time complexities for the JT kernels $k_{JT}^{(1,H)}$ and $k_{JT}^{(2,H)}$ are $O(HN^2 n^2 + HN^3 n + Nn^3)$ and $O(HN^2 n^2 + Nn^3)$, respectively.

4 Experimental Results

In this section, we empirically evaluate the performance of our JT kernels on standard attributed graphs from bioinformatics. Furthermore, we also compare our new kernels with several state of the art graph kernels.

Table 1. Information of the Graph-based Datasets

Datasets	MUTAG	NCI1	NCI109	ENZYMES	PPIs	PTC(MR)
Max # vertices	28	111	111	126	238	109
Min # vertices	10	3	4	2	3	2
Mean # vertices	17.93	29.87	29.68	32.63	109.63	25.60
# graphs	188	4110	4127	600	219	344
# classes	2	2	2	6	5	2

4.1 Graph Datasets

We evaluate our kernels on standard graph datasets. These datasets include: MUTAG, NCI1, NCI109, ENZYMES, PPIs and PTC(MR). More details are shown in Table.1.
MUTAG: The MUTAG dataset consists of graphs representing 188 chemical compounds, and aims to predict whether each compound possesses mutagenicity.

NCI1 and NCI109: The NCI1 and NCI109 datasets consist of graphs representing two balanced subsets of datasets of chemical compounds screened for activity against non-small cell lung cancer and ovarian cancer cell lines respectively. There are 4110 and 4127 graphs in NCI1 and NCI109 respectively.
ENZYMES: The ENZYMES dataset consists of graphs representing protein tertiary structures consisting of 600 enzymes from the BRENDA enzyme. The task is to correctly assign each enzyme to one of the 6 EC top-level.
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). Note that, unlike the experiment in [26] that only uses the *Proteobacteria*40 and the *Acidobacteria*46 PPIs as the testing graphs, we use all the PPIs as the testing graphs in this paper. As a result, the experimental results for some kernels are different on the PPIs dataset.
PTC: The PTC (The Predictive Toxicology Challenge) dataset records the carcinogenicity of several hundred chemical compounds for Male Rats (MR), Female Rats (FR), Male Mice (MM) and Female Mice (FM). These graphs are very small (i.e.,$20-30$ vertices, and $25-40$ edges) and sparse. We select the graphs of MR for evaluation.

4.2 Experiments on Graph Classification

We evaluate the performance of our JT kernels $k_{JT}^{(q,H)}$ for $q=1$ (JT1) and $q=2$ (JT2). Moreover, we also compare our kernels with several alternative state of the art graph kernels. These graph kernels for comparison include: 1) the Jensen-Shannon graph kernel (JSGK) associated with the CSSRW [2], 2) the unaligned and aligned quantum Jensen-Shannon kernels (QJSK and QJSKA) [26], 3) the Weisfeiler-Lehman subtree kernel (WLSK) [9], 4) the shortest path graph kernel (SPGK) [8], 5) the graphlet count graph kernel with graphlet of size 3 (GCGK3) [11], 6) the backtrackeless random walk kernel using the Ihara zeta function based cycles (BRWK) [19], 7) the random walk graph kernel (RWGK) [3].

For each kernel, we compute the kernel matrix on each graph dataset. We perform 10-fold cross-validation using the C-Support Vector Machine (C-SVM) Classification to compute the classification accuracies, using LIBSVM. We use nine samples for training and one for testing. All the C-SVMs were performed along with their parameters optimized on each dataset. We report the average classification accuracies ($\pm$ standard error) and the runtime for each kernel in Table 2. The runtime is measured under Matlab R2011a running on a 2.5GHz Intel 2-Core processor (i.e. i5-3210m). Note that, both our JT kernel and the WLSK kernel are related to a TI method. In this work, we set the parameter H (i.e., the maximum number of TI iteration) to 10, i.e., we vary h from 1 to 10 for both our kernel and the WLSK kernel. The reasons of setting $H=10$ are twofold. First, for most of the datasets, the strengthened vertex labels of the graphs tend

to be all different after $h = 10$. In other words, after $h = 10$, there are nearly no isomorphic subtrees, i.e., we achieve maximum discrimination. Second, in our experiments we observe that the classification performance tends to be more stable after $h = 10$. As a result, for each dataset we compute 10 different kernel matrices for both our kernel and the WLSK kernel. The classification accuracy is then the average accuracy over the 10 kernel matrices. Note that, the experimental results (for the WLSK kernel) on some datasets are different from those in [27], since the authors of [27] set $H = 3$. Finally, recall that our JT kernel, and the WLSK and SPGK kernels are all able to handle attributed graphs. However, the graphs in the PPIs dataset are unattributed graphs, thus we decided to use the vertex degree as a vertex label.

Table 2. Classification Accuracy (In $\%$ $\pm$ Standard Error) and Runtime for Various Kernels

Datasets	MUTAG	NCI1	NCI109	ENZYMES	PPIs	PTC(MR)
JT1	85.10 ± 0.64	**86.35** ± 0.12	**87.00** ± 0.15	**57.41** ± 0.53	87.28 ± 0.61	**60.16** ± 0.50
JT2	**85.50** ± 0.55	85.32 ± 0.14	85.79 ± 0.13	56.41 ± 0.42	**88.47** ± 0.47	58.50 ± 0.39
JSGK	83.11 ± 0.80	62.50 ± 0.33	63.00 ± 0.35	20.81 ± 0.29	34.57 ± 0.54	57.29 ± 0.41
QJSK	82.72 ± 0.44	69.09 ± 0.20	70.17 ± 0.23	36.58 ± 0.46	65.61 ± 0.77	56.70 ± 0.49
QJSKA	82.83 ± 0.50	–	–	24.31 ± 0.27	61.09 ± 0.98	57.39 ± 0.46
WLSK	82.88 ± 0.57	84.77 ± 0.13	84.49 ± 0.13	52.75 ± 0.44	88.09 ± 0.41	58.26 ± 0.47
SPGK	83.38 ± 0.81	74.21 ± 0.30	73.89 ± 0.28	41.30 ± 0.68	59.04 ± 0.44	55.52 ± 0.46
GCGK3	82.04 ± 0.39	63.72 ± 0.12	62.33 ± 0.13	24.87 ± 0.22	46.61 ± 0.47	55.41 ± 0.59
BRWK	77.50 ± 0.75	60.34 ± 0.17	59.89 ± 0.15	20.56 ± 0.35	–	53.97 ± 0.31
RWGK	80.77 ± 0.72	–	–	22.37 ± 0.35	41.29 ± 0.89	55.91 ± 0.37

Datasets	MUTAG	NCI1	NCI109	ENZYMES	PPIs	PTC(MR)
JT1	14”	7h21′	7h24′	11′30”	3′20”	1′10”
JT2	3”	10′50”	10′55”	30”	1′43”	8”
JSGK	1”	1”	1”	1”	1”	1”
QJSK	20”	2h55′	2h55′	4′23”	3′24”	1′46”
QJSKA	1′30”	$>$ 1 day	$>$ 1 day	$1h10'$	$1h54'$	16′40”
WLSK	3”	2′31”	2′37”	20”	20”	9”
SPGK	1”	16”	16”	4”	22”	1”
GCGK3	1”	5”	5”	2”	4”	1”
BRWK	11”	6′49”	6′49”	3′5”	$>$ 1 day	29”
RWGK	14”	$>$ 1 day	$>$ 1 day	9′52”	4′26”	2′35”

Results and Discussions. In terms of classification accuracy, our JT kernels overcome the alternative kernels on most of the datasets. Only the accuracy of the WLSK kernel on the PPIs dataset is a little higher than our kernel $k_{JT}^{(2,H)}$. The reasons of the effectiveness of our kernels are explained as follows. a) Compared to the JSGK kernel, our JT kernels overcome the restriction on non-attributed graphs. Moreover, our JT kernels also overcome the shortcoming of lacking correspondence information. Finally, since each strengthened label of the JT kernels correspondences to a subtree, the JT kernels reflect richer interior topology information than the JSGK kernel. By contrast, the JSGK kernel only reflects limited information in terms of the vertex degree distribution. b) Compared to the QJSK and QJSKA kernels, our JT kernels also overcome the restriction on non-attributed graphs arising in the two quantum kernels. Moreover, Bai et al. [26] show that the QJSK kernel requires the computation of an additional mixed state where the system has equal probability of being in each of the two original quantum states. Unless this quantum kernel takes into account the correspondences between the vertices of the two graphs, it can be shown that this kernel is not permutation invariant. While

our JT kernels are not only permutation invariant but also reflect the correspondence information between pairwise probabilities computed from the CTQW. c) Compared to the BRWK, RWGK and GCGK3 kernels, our JT kernels reflect richer topology information in terms of the subtrees identified by the TI method (i.e.,the strengthened labels). The reason for this is that the subtree based strategy can overcomes the shortcoming of structurally simple problem arising in the path and walk based strategies. Moreover, the CTQW required for our kernels also possess more interesting properties than the BRWK and RWGK based on the classical random walk. d) Compared to the WLSK kernel, our JT kernel $k_{JT}^{(2,H)}$ generalizes the subtree based kernel and reflects richer information. Moreover, our JT kernel $k_{JT}^{(1,H)}$ overcomes the shortcoming of discarding non-isomorphic subtrees arising in the WLSK kernel. Finally, the performance of the kernel $k_{JT}^{(1,H)}$ is a little better than that of the kernel $k_{JT}^{(2,H)}$. The reason for this is that the kernel $k_{JT}^{(2,H)}$ also suffers from the problem of discarding non-isomorphic substructures, while the kernel $k_{JT}^{(1,H)}$ can overcome this shortcoming. e) In terms of the runtime, our JT kernel $k_{JT}^{(2,H)}$ is fast and is competitive to the fast subtree kernel WLSK. Furthermore, the computational efficiency of our JT kernel $k_{JT}^{(1,H)}$ is obviously slower, but it can still finish the computation in a polynomial time on any dataset. By contrast, some kernels cannot finish the computation on some datasets in one day.

5 Conclusions

In this paper, we develop a family of JT kernels for attributed graphs. For a graph, we use a TI method to strengthen the vertex labels and then compute the probability distribution over the vertex labels through the CTQW. For a pair of graphs, the JT kernels are computed by measuring the JT q-difference between their probability distributions. We show that the JT kernels not only generalize some state of the art kernels but also overcome the shortcomings arising in these kernels. The experiments demonstrate the effectiveness and efficiency of the JT kernels. Our future work is to investigate the relationship between state of the art kernels and our JT kernel with other q values (e.g., $q = 0$). Furthermore, we are also interested in extending this work on financial analysis.

Acknowledgments. We thank Dr. Chaoyan Wang for the insights of future extension in financial analysis.

References

1. Dahm, N., Bunke, H., Caelli, T., Gao, Y.: A Unified Framework for Strengthening Topological Node Features and Its Application to Subgraph Isomorphism Detection. In: Kropatsch, W.G., Artner, N.M., Haxhimusa, Y., Jiang, X. (eds.) GbRPR 2013. LNCS, vol. 7877, pp. 11–20. Springer, Heidelberg (2013)
2. Bai, L., Hancock, E.R.: Graph Kernels from the Jensen-Shannon Divergence. Journal of Mathematical Imaging and Vision 47(1-2), 60–69 (2013)
3. Gärtner, T., Flach, P.A., Wrobel, S.: On Graph Kernels: Hardness Results and Efficient Alternatives. In: Schölkopf, B., Warmuth, M.K. (eds.) COLT/Kernel 2003. LNCS (LNAI), vol. 2777, pp. 129–143. Springer, Heidelberg (2003)

4. Bunke, H., Riesen, K.: A Family of Novel Graph Kernels for Structural Pattern Recognition. In: Rueda, L., Mery, D., Kittler, J. (eds.) CIARP 2007. LNCS, vol. 4756, pp. 20–31. Springer, Heidelberg (2007)
5. Jebara, T., Kondor, R.I., Howard, A.: Probability Product Kernels. Journal of Machine Learning Research 5, 819–844 (2004)
6. Haussler, D.: Convolution Kernels on Discrete Structures. Technical Report UCS-CRL-99-10 (1999)
7. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized Kernels Between Labeled Graphs. In: Proc. ICML, pp. 321–328 (2003)
8. Borgwardt, K.M., Kriegel, H.P.: Shortest-path Kernels on Graphs. In: Proc. ICDM, pp. 74–81 (2005)
9. Shervashidze, N., Borgwardt, K.M.: Fast Subtree Kernels on Graphs. In: Proc. NIPS, pp. 1660–1668 (2009)
10. Costa, F., Grave, K.D.: Fast Neighborhood Subgraph Pairwise Distance Kernel. In: Proc. ICML, pp. 255–262 (2010)
11. Shervashidze, N., Vishwanathan, S.V.N., Petri, T., Mehlhorn, K., Borgwardt, K.M.: Efficient Graphlet Kernels for Large Graph Comparison. Journal of Machine Learning Research 5, 488–495 (2009)
12. Martins, A.F.T., Smith, N.A., Xing, E.P., Aguiar, P.M.Q., Figueiredo, M.A.T.: Nonextensive Information Theoretic Kernels on Measures. Journal of Machine Learning Research 10, 935–975 (2009)
13. Julia, K.: Quantum Random Walks: An Introductory Overview. Contemporary Physics 44(4), 307–327 (2003)
14. Farhi, E., Gutmann, S.: Quantum Computation and Decision Trees. Physical Review A 58, 915 (1998)
15. Aubry, M., Schlickewei, U., Cremers, D.: The Wave Kernel Signature: A Quantum Mechanical Approach to Shape Analysis. In: Proc. ICCV Workshops, pp. 1626–1633 (2011)
16. Rossi, L., Tosello, A., Hancock, E.R., Wilson, R.C.: Characterizing Graph Symmetries through Quantum Jensen-Shannon Divergence. Physical Review E 88(3-1), 032806 (2013)
17. Suau, P., Hancock, E.R., Escolano, F.: Graph Characteristics from the Schrödinger Operator. In: Kropatsch, W.G., Artner, N.M., Haxhimusa, Y., Jiang, X. (eds.) GbRPR 2013. LNCS, vol. 7877, pp. 172–181. Springer, Heidelberg (2013)
18. Cottrell, S., Hillery, M.: Finding Structural Anomalies in Star Graphs Using Quantum Walks. Physical Review Letters 112(3), 030501 (2014)
19. Aziz, F., Wilson, R.C., Hancock, E.R.: Backtrackless Walks on A Graph. IEEE Transactions on Neural Networks and Learning System 24(6), 977–989 (2013)
20. Mahé, P., Ueda, N., Akutsu, T., Perret, J., Vert, J.: Extensions of marginalized graph kernels. In: Proc. ICML (2004)
21. Alon, N., Benjamini, I., Lubetzky, E., Sodin, S.: Non-backtracking Random Walks Mix Faster. Communications in Contemporary Mathematics 9(4), 585–603 (2007)
22. Tsallis, C.: Possible Generalization of Boltzman-Gibbs Statistacs. J. Stats. Physics 52, 479–487 (1988)
23. Furuichi, S.: Information Theoretical Properities of Tsallis Entropies. Journal of Math. Physics 47, 2 (2006)
24. Riesen, K., Bunke, H.: Graph Classification and Clustering based on Vector Space Embedding. World Scientific Publishing (2010)
25. Bai, L., Hancock, E.R., Ren, P.: Jensen-Shannon Graph Kernel using Information Functionals. In: Proc. ICPR, pp. 2877–2880 (2012)
26. Bai, L., Rossi, L., Torsello, A., Hancock, E.R.: A Quantum Jensen-Shannon Kernel for Unattributed Graphs. To appear in Pattern Recognition (2014)
27. Kriege, N., Mutzel, P.: Subgraph Matching Kernels for Attributed Graphs. In: Proc. ICML (2012)