Learning With Kernels: A Local Rademacher Complexity-Based Analysis With Application to Graph Kernels

Luca Oneto, *Member, IEEE*, Nicolò Navarin, *Member, IEEE*, Michele Donini, Sandro Ridella, *Member, IEEE*, Alessandro Sperduti, *Senior Member, IEEE*, Fabio Aiolli, and Davide Anguita, *Senior Member, IEEE*

***Abstract*— When dealing with kernel methods, one has to decide which kernel and which values for the hyperparameters to use. Resampling techniques can address this issue but these procedures are time-consuming. This problem is particularly challenging when dealing with structured data, in particular with graphs, since several kernels for graph data have been proposed in literature, but no clear relationship among them in terms of learning properties is defined. In these cases, exhaustive search seems to be the only reasonable approach. Recently, the global Rademacher complexity (RC) and local Rademacher complexity (LRC), two powerful measures of the complexity of a hypothesis space, have shown to be suited for studying kernels properties. In particular, the LRC is able to bound the generalization error of an hypothesis chosen in a space by disregarding those ones which will not be taken into account by any learning procedure because of their high error. In this paper, we show a new approach to efficiently bound the RC of the space induced by a kernel, since its exact computation is an NP-Hard problem. Then we show for the first time that RC can be used to estimate the accuracy and expressivity of different graph kernels under different parameter configurations. The authors' claims are supported by experimental results on several real-world graph data sets.**

***Index Terms*— Generalization performances, graph kernels, kernel methods, local Rademacher complexity (LRC), RC.**

I. Introduction

KERNEL methods are powerful learning algorithms that can be easily applied to every input domain [1] and are backed up from statistical learning theory (SLT) that offers strong theoretical guarantees on the output hypothesis [2]. These methods represent the solution in terms of pairwise similarity among the examples and do not work on an explicit representation of them [3]. The function used to compute the similarity has to be a kernel function.

In particular, kernel methods are a well-established solution for structured domains because they can be defined directly on structured data [4]. This relieves the user from the definition of a vectorial representation of the data, a time-consuming and task-specific operation. Given an input domain, e.g., a graph domain, the user is required to fix a specific kernel function and the corresponding values for its hyperparameters to adopt. The crucial importance of the kernel adopted for the performance of a kernel machine is well known, and researchers are investigating on the automatic learning of kernels, also known as kernel learning. For example, it is possible to combine multiple kernels with different parameter configurations by using the, so-called, multiple kernel learning (MKL) algorithms [1], [5]–[7].

In most cases, this task is performed using time-expensive parameter selection procedures that are often implemented via resampling methods [8], [9] like a K-fold cross validation (KCV) and Bootstrap. However, SLT provides powerful tools that can be exploited for this task up to some extent, or at least can be used for understanding the properties of different kernels [2], [6]. More in detail, a learning process can be described as the selection of an hypothesis in a fixed set, based on empirical observations [2]. Its asymptotic analysis has been thoroughly investigated in the past, through bounds on the generalization error [2], [10]. However, as the number of samples is limited in practice, finite samples analysis with global measures of the complexity of the hypotheses set was proposed, and represented a fundamental advance in the field [2], [11]–[13]. A further refinement consisted in exploiting local measures of complexity, which take into account only those models that well approximate the available data [14]–[16]. In this context, the Rademacher complexity (RC), its global RC (GRC) [12], and local RC (LRC) [14] versions represent state-of-the-art tools for measuring the complexity of an hypothesis space induced by a kernel function [6], [17], [18]. Unfortunately, their computation is not trivial [11]–[16], [19].

On the other side, when it comes to dealing with graphs, several instances of kernels have been presented in literature [20]. A recent advance in the field are fast kernels

Manuscript received August 14, 2016; revised May 26, 2017, September 25, 2017, and November 6, 2017; accepted November 6, 2017. *(Corresponding author: Luca Oneto.)*

L. Oneto and D. Anguita are with the Department of Informatics, Bioengineering, Robotics, and Systems Engineering, University of Genoa, I-16145 Genoa, Italy (e-mail: luca.oneto@unige.it; davide.anguita@unige.it).

N. Navarin, A. Sperduti, and F. Aiolli are with the Department of Mathematics, University of Padova, I-35121 Padua, Italy (e-mail: nnavarin@math.unipd.it; sperduti@math.unipd.it; aiolli@math.unipd.it).

M. Donini is with the Computational Statistics and Machine Learning Department, Istituto Italiano di Tecnologia, I-16163 Genoa, Italy (e-mail: michele.donini@iit.it).

S. Ridella is with Dipartimento di Ingegneria Navale, Elettrica, Elettronica e delle Telecomunicazioni, University of Genoa, I-16145 Genoa, Italy (e-mail: sandro.ridella@unige.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2017.2771830

(near-linear time) that allow for an explicit, sparse feature space representation that can be successfully applied to large graph data sets [21], [22]. Each kernel considers as features different small substructures of the original graph. Empirical comparisons among different kernels can be found in literature [23], [24] but, with few exceptions [25], no theoretical comparison is present. Moreover, usually kernels depend on one or more user-specified parameters that control the resulting computational complexity and change the induced hypothesis space. The selection of an appropriate kernel and values for its hyperparameters can be a critical phase for achieving satisfactory predictive performance on a specific task. In the context of graph kernels, the expressiveness of a kernel is defined as its ability to distinguish between nonisomorphic examples. In [26], it is shown that complete graph kernels (kernels that map each nonisomorphic graph in a different point in the feature space) are hard to compute. Thus, the kernels that we consider (and the ones that are used in practice) are not complete, but it is difficult to characterize their expressiveness, even in a relative way. If the nonzero features generated by different kernels are independent of each other, then it is easy to see that the more the nonzero features a kernel generates, the more it is able to discriminate among examples, and so the more it is expressive. However, this is not the case with structural features, where there are strong dependence relationships among them, i.e., a feature can be nonzero only if some specific features are nonzero as well. In this case, there is no easy way to assess how expressive a kernel is.

In this paper, we present a novel theoretical result on LRC. Specifically, we show a novel approach to efficiently estimate the LRC of the space induced by a kernel and how to relate this estimate to the generalization capability of a function chosen in the set of linear separators in this space. We then demonstrate the practical usefulness of the proposed result by exploiting it to empirically analyze the expressiveness and the predictive performance of different state-of-the-art graph kernels by means of GRC and LRC.

The paper is organized as follows. Section II introduces the learning framework. Section III recalls the state-of-the-art GRC and LRC generalization bounds. Section IV introduces the problem of learning with kernels. Section V shows how to effectively and efficiently estimate the GRC and LRC from the data. Section VI recalls the state-of-the-art kernels for graphs. Section VII shows how to exploit the results of Section V in order to measure the accuracy and the expressivity of different kernels (or the same kernel with different parameter configurations), with particular reference to Graph Kernels, on a series of real-world graph data sets.

II. Learning Framework

In this paper, we will deal with binary classification problems [2]: based on some random observations X of the input space $\mathcal{X}$, one has to estimate the associated label Y which belongs to the output space $\mathcal{Y} = \{-1, +1\}$ by choosing a suitable hypothesis $h : \mathcal{X} \to \widehat{\mathcal{Y}}$, in a set $\mathcal{H}$ of possible hypotheses. A learning algorithm selects $h \in \mathcal{H}$ by exploiting a set of labeled samples $\mathcal{D}_n : \{(X_1, Y_1), \cdots, (X_n, Y_n)\}$. The latter are sampled independent identically distributed (i.i.d.) according to the distribution μ over the Cartesian product between the input and output space $\mathcal{X} \times \mathcal{Y}$. The generalization error

$$L(h) = \mathbb{E}_{(X,Y)}\ell(h(X), Y) \tag{1}$$

associated with an hypothesis $h \in \mathcal{H}$ is defined through a loss function $\ell(h(X), Y) : \widehat{\mathcal{Y}} \times \mathcal{Y} \to [0, 1]$. As μ is unknown, $L(h)$ cannot be explicitly computed, thus we have to resort to its empirical estimator, namely, the empirical error

$$\widehat{L}_n(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(h(X_i), Y_i). \tag{2}$$

Note that $\widehat{L}_n(h)$ is a biased estimator of $L(h)$, since the data exploited for selecting the hypothesis and for computing the empirical error coincide. Since the purpose of any learning procedure is to select the hypothesis $h \in \mathcal{H}$ characterized by the smallest possible generalization error, a crucial issue is represented by the estimation of this bias.

III. Rademacher Complexity and Generalization Error

The LRC is one of the most powerful tools for estimating the discrepancy between the generalization error and the empirical error [14]–[16]. Before recalling the LRC-based bound on the generalization error of an hypothesis $h \in \mathcal{H}$, we recall the definition of empirical square error

$$\widehat{L}_n^2(h) = \frac{1}{n}\sum_{i=1}^{n}[\ell(h(X_i), Y_i)]^2 \tag{3}$$

and the definition of GRC

$$\widehat{R}_n(\mathcal{H}) = \mathbb{E}_{\boldsymbol{\sigma}} \sup_{h\in\mathcal{H}} \frac{2}{n}\sum_{i=1}^{n} \sigma_i \ell(h(X_i), Y_i) \tag{4}$$

where $\boldsymbol{\sigma}$ is a vector whose components $[\sigma_1|\cdots|\sigma_n]^T$ are n $\{\pm 1\}$-valued i.i.d. Rademacher random variables for which $\mathbb{P}\{\sigma_i = +1\} = \mathbb{P}\{\sigma_i = -1\} = (1/2)$.

Based on these definitions it is possible to recall the GRC-based bound on the generalization error of an hypothesis $h \in \mathcal{H}$ [11]–[13].

Theorem 1 [12]: Consider a $[0, 1]$*-bounded loss function, then* $\forall h \in \mathcal{H}$ *and with probability at least* $(1-2e^{-x})$*, we have that*

$$L(h) \le \widehat{L}_n(h) + \widehat{R}_n(\mathcal{H}) + 3\sqrt{\frac{x}{2n}}. \tag{5}$$

The constants involved in Theorem 1 can be improved [13] but this is out of the scope of this paper. Instead, the drawback of Theorem 1 is that it takes into account all the functions in $\mathcal{H}$, including the ones that will never be chosen during the learning phase; the latter in fact aims to select only those functions with small error. The LRC-based generalization bound [14]–[16] addresses exactly this issue by considering only those functions characterized by a small error.

Theorem 2 [14], [16]: Consider a $[0,1]$*-bounded loss function and the hypotheses space* $\mathcal{H}$*. Let us define the following subset of* $\mathcal{H}$:

$$\widehat{\mathcal{H}}_{r,\alpha,x,n}=\left\{h \,\middle|\, h \in \mathcal{H}, \widehat{L}_n^2(h) \le \frac{1}{\alpha^2}\left(3r+\sqrt{\frac{x}{2n}}\right)\right\} \quad (6)$$

and let $\widehat{r}_n$ *be the fixed point of the following subroot function*:

$$\widehat{\psi}_n(r) = \sup_{\alpha\in(0,1]} \alpha \widehat{R}_n(\widehat{\mathcal{H}}_{r,\alpha,x,n}) + \sqrt{\frac{2x}{n}}. \quad (7)$$

Then, $\forall h \in \mathcal{H}$ *and with probability at least* $(1-3e^{-x})$*, we have that*

$$L(h) \le \min_{K\in(1,+\infty)} \frac{K}{K-1}\widehat{L}_n(h) + K\widehat{r}_n + 2\sqrt{\frac{x}{2n}}. \quad (8)$$

Note that the bound of Theorem 2, contrarily to the conventional one of Theorem 1, takes into account only a subset of the models in $\mathcal{H}$, in particular the one with small square empirical error.

Even if the bounds of Theorems 1 and 2 take into account only empirical quantities, their computation is rather complex [12], [14], and it turns out to be NP-Hard [12], [14].

IV. Learning With Kernels

A kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a symmetric positive semi-definite function that corresponds to a dot product in a reproducing kernel Hilbert space (RKHS), i.e., there exists a $\phi : \mathcal{X} \to \mathcal{K} \subseteq \mathbb{R}^D$, where $\mathcal{K}$ is a Hilbert space (commonly referred to as *feature space*), such that $k(X_1, X_2) = \langle\phi(X_1), \phi(X_2)\rangle$ with $X_1, X_2 \in \mathcal{X}$. Note that the input space $\mathcal{X}$ can be any space.

The main problem when we deal with kernels involves the choice of one kernel, together with its values for the hyperparameters, with respect to another one. Fixing a kernel means fixing the representation for our task but, unfortunately, the user's choice of the values for the kernel hyperparameters usually is suboptimal and time-consuming. Selecting the wrong kernel is equivalent to selecting the wrong set of features to describe our task, and the performance of our learning algorithm is deeply effected by this inaccurate choice [27]. This is a fundamental issue which is also at the basis of the MKL problem [5]–[7], [18]. In MKL, one has to properly combine multiple kernels and decide which kernels must be disregarded and consequentially which features in the RKHS are more important, connecting Kernel Learning to the Feature Learning theory [28].

In this paper, we will consider as hypotheses space $\mathcal{H}$ the set of linear separators in the Hilbert space such that

$$h(X)=W\cdot\phi(X), \quad \text{s.t. } \|W\|^2 \le H^2 \le 1, \ \|\phi(X)\|^2 \le 1 \quad (9)$$

where $H \in (0, \infty]$. Consequently, we will face the problem of learning linear separators in the space induced by the kernel. Moreover, the linear loss function [29]

$$\ell_T(h(X), Y) = \frac{1 - Yh(X)}{2} \quad (10)$$

will be used. Note that, since we are dealing with binary classification problems, the Hard loss function $\ell_H(h(X), Y)$, counting the number of misclassified examples, can be easily upper bounded through the linear loss function

$$\ell_H(h(X), Y) = \frac{1 - Y\text{sign}[h(X)]}{2} \le 2\ell_T(h(X), Y). \quad (11)$$

A. Data Separation and Kernel Normalization

Given a training set, we consider the domain $\hat{\Gamma}$ of pairs of probability distributions $\boldsymbol{\gamma} \in \mathbb{R}_+^n$ defined over the sets of positive and negative examples. More formally

$$\hat{\Gamma} = \left\{\boldsymbol{\gamma} \in \mathbb{R}_+^n \,\middle|\, \sum_{i\in\oplus}\gamma_i = 1, \sum_{i\in\ominus}\gamma_i = 1\right\} \quad (12)$$

where $\oplus = \{i | i \in \{1,\cdots,n\}, Y_i = +1\}$ and $\ominus = \{i | i \in \{1,\cdots,n\}, Y_i = -1\}$. Note that any element $\boldsymbol{\gamma} \in \hat{\Gamma}$ corresponds to a pair of points, the first in the convex hull of positive training examples and the second in the convex hull of negative training examples.

Given two generic points in the convex hulls of positive and negative examples in feature space, specified by a vector $\boldsymbol{\gamma} \in \hat{\Gamma}$, then their squared distance can be computed by

$$\begin{aligned} D(\boldsymbol{\gamma}) &= \left\|\sum_{i\in\oplus}\gamma_i\phi(X_i) - \sum_{i\in\ominus}\gamma_i\phi(X_i)\right\|^2 \\ &= \left\|\sum_i Y_i\gamma_i\phi(X_i)\right\|^2 = \sum_{i=1}^n\sum_{j=1}^n Y_iY_j\gamma_i\gamma_j k(X_i, X_j). \quad (13)\end{aligned}$$

It follows that the minimum squared distance between the convex hulls of positive and negative examples can be computed by $D_{\min} = \min_{\boldsymbol{\gamma}\in\hat{\Gamma}} D(\boldsymbol{\gamma})$, and similarly the maximum squared distance between pairs of positive and negative examples can be obtained by $D_{\max} = \max_{\boldsymbol{\gamma}\in\hat{\Gamma}} D(\boldsymbol{\gamma})$.

Using well-known properties of symmetric positive semi-definite matrices [30], we can give a lower bound to the minimum distance as follows:

$$D_{\min} = \min_{\boldsymbol{\gamma}\in\hat{\Gamma}} D(\boldsymbol{\gamma}) \ge \frac{\lambda^{\min-1}}{c^{\min}} + \frac{\lambda^{\min}}{c^{\max}} \ge \frac{2}{n}\lambda^{\min} \quad (14)$$

where $\lambda^{\min-1} \ge \lambda^{\min}$ are the smallest eigenvalues of the matrix $\Phi\Phi^T$, where $\Phi = [\phi(X_1)|\cdots|\phi(X_n)]^T$, $c^{\min} = \min\{|\oplus|, |\ominus|\}$, and $c^{\max} = \max\{|\oplus|, |\ominus|\}$. This implies that, when the kernel matrix is not singular, then any random labeling can be shattered by an hyperplane in feature space as the distance between the convex hulls is strictly larger than zero, and this is true for any labeling.

Similarly, we can give an upper bound to the maximum distance of training examples, which depends on the maximal eigenvalue, that is

$$D_{\max} = \max_{\boldsymbol{\gamma}\in\hat{\Gamma}} D(\boldsymbol{\gamma}) \le \lambda^{\max} \quad (15)$$

where $\lambda^{\max}$ is the larger eigenvalue of the matrix $\Phi\Phi^T$.

Then, the geometry of feature space is implicitly fixed by the kernel definition, modulo the difference between the maximum distance of training examples and the minimum distance between the convex hulls of positive and negative examples. In this sense, a normalization of the space has to be performed in order to compare two kernels: we divide by the difference between $\lambda^{\max}$ and $(2/n)\lambda^{\min}$. Consequently, the normalized kernel $\widetilde{\boldsymbol{\phi}}$ defined from $\boldsymbol{\phi}$ can be introduced as

$$\widetilde{\boldsymbol{\phi}} = \frac{\boldsymbol{\phi}}{\sqrt{\left(\lambda_{\max} - \frac{2}{n}\lambda_{\min}\right)}}. \tag{16}$$

An immediate consequence of this normalization is the invariance of the geometry with respect to multiplicative constants, i.e., the normalized kernel of $t \cdot \boldsymbol{\phi}$ is the same $\forall t \in \mathbb{R}, t > 0$.

Remark 1: The following bound for the 2-norm of $\widetilde{\boldsymbol{\phi}}(X_i)$ holds:

$$\|\widetilde{\boldsymbol{\phi}}(X_i)\|_2^2 = \frac{\boldsymbol{\phi}(X_i) \cdot \boldsymbol{\phi}(X_i)}{\lambda_{\max} - \frac{2}{n}\lambda_{\min}} \leq \frac{n\lambda_{\max}}{n-2} \leq \frac{n^2}{n-2}. \tag{17}$$

Consequently we are able to enclose the data projected by different kernels into an hypersphere of known radius.

V. Accuracy and Expressiveness of Kernels

In this section, we will show how to measure in polynomial time the expressivity of the hypothesis space defined by the sheaf of linear separators passing through the origin in the space induced by a kernel in polynomial time. The same will be done for the accuracy of any hypothesis chosen in this hypothesis space.

Let us also define the following vector $\boldsymbol{y} = [Y_1|\cdots|Y_n]^T$ and also the optimum of the following problem:

$$W^* : \arg\min_{h\in\mathcal{H}} \widehat{L}_n^2(h) = \arg\min_{\|W\|^2\leq H^2} \|\Phi W - \boldsymbol{y}\|^2. \tag{18}$$

By basic functional analysis, it is possible to prove the following lemma.

Lemma 3: For any $\|W\|^2 \leq H^2$, we have that

$$\|\Phi W - \boldsymbol{y}\|^2 - \|\Phi W^* - \boldsymbol{y}\|^2 - \|\Phi(W - W^*)\|^2 \geq 0. \tag{19}$$

Note that Lemma 3 has been proven under more general conditions in [31] and [32] but the constants involved in that result are much worse than the ones of Lemma 3.

Thanks to the result of Lemma 3 it is possible to prove the next lemma which will be useful later in this section.

Lemma 4: Let us consider an hypotheses space $\mathcal{H}$ such that $h(X) = W \cdot \phi(X)$ with $\|W\|^2 \leq H^2$ and $\|\phi(X)\|^2 \leq 1$, then

$$\left\{ W \middle| \|W\|^2 \leq H^2, \frac{1}{n}\sum_{i=1}^n \left[\frac{1 - YW\cdot\phi(X)}{2}\right]^2 \leq c_1 \right\} \subseteq \left\{ W \middle| \|W\|^2 \leq 4H^2, \|\Phi W\|^2 \leq c_2 \right\} \tag{20}$$

where $c_1 \in [0, \infty)$ and $c_2 = 4nc_1 - \|\Phi W^* - \boldsymbol{y}\|^2$.

The proof is reported in Appendix A.

Lemma 4 is a fundamental result which can be used to upper bound the LRC without facing the original NP-Hard problem. Before showing this result, we need to prove first how to upper bound the GRC.

Theorem 5: Let be given an hypotheses space $\mathcal{H}$ such that $h(X) = W \cdot \phi(X)$ and $\|W\|^2 \leq H^2$, then

$$\widehat{R}_n(\mathcal{H}) \leq \frac{H}{n}\sqrt{\sum_{i=1}^n \lambda_i} = \frac{H}{n}\sqrt{\sum_{i=1}^n Q_{i,i}} \tag{21}$$

where $\{\lambda_1, \cdots, \lambda_n\}$ are the eigenvalues, sorted in descending order, of the gram matrix $Q = \Phi\Phi^T$ and where $Q_{i,j} = k(X_i, X_j)$.

The proof is mainly based on the result of [12]. The bound of Theorem 5 is also tight since the following lemma can be proved:

Lemma 6: Consider an hypotheses space $\mathcal{H}$ such that $h(X) = W \cdot \phi(X)$ and $\|W\|^2 \leq H^2$, then

$$\widehat{R}_n(\mathcal{H}) \geq \frac{H}{\sqrt{2}n}\sqrt{\sum_{i=1}^n \lambda_i} = \frac{H}{\sqrt{2}n}\sqrt{\sum_{i=1}^n Q_{i,i}}. \tag{22}$$

The proof is mainly based on the Khinchin–Kahane inequality [33].

Theorem 5 allows to bound the GRC when a set of linear separators in the Hilbert space are exploited in polynomial time $O(n)$. Consequently, thanks to the results of Theorem 1, we can state the following corollary.

Corollary 7: Let us consider an hypotheses space $\mathcal{H}$ such that $h(X) = W \cdot \phi(X)$ and $\|W\|^2 \leq H^2$. Then $\forall h \in \mathcal{H}$, the following inequality holds with probability at least $(1-2e^{-x})$:

$$\mathbb{E}_{(X,Y)}\ell_H(h(X), Y) \leq \frac{1}{n}\sum_{i=1}^n \ell_T(h(X_i), Y_i) + \frac{H}{n}\sqrt{\sum_{i=1}^n \lambda_i} + 3\sqrt{\frac{x}{2n}}. \tag{23}$$

The proof can be easily obtained by combining Theorems 1 and 5 and the inequality of (11).

Before showing the LRC-based counterpart of Corollary 7, we still need the following last lemma.

Lemma 8: Given the previous definitions the following inequality holds:

$$\mathbb{E}_{\boldsymbol{\sigma}} \sup_{\{W | \|W\|^2 \leq 4H^2, \|\Phi W\|^2 \leq c_2\}} \sum_{i=1}^n \sigma_i W \cdot \phi(X_i) \leq 2H\sqrt{\sum_{i=1}^n \min\left[\frac{c_2}{4}, \lambda_i\right]}. \tag{24}$$

The proof is rather simple and the same result, but under more general conditions, can be found in [6], [14], and [34]. Also the bound of Lemma 8 is tight, in fact in [6], [14], and [34], it is proven the following, lemma:

Lemma 9 [14]: Given the previous definition the following inequality holds:

$$\mathbb{E}_{\boldsymbol{\sigma}} \sup_{\{W | \|W\|^2 \leq 4H^2, \|\Phi W\|^2 \leq c_2\}} \sum_{i=1}^n \sigma_i W \cdot \phi(X_i) \geq c_3 H\sqrt{\sum_{i=1}^n \min\left[\frac{c_2}{4}, \lambda_i\right]} \tag{25}$$

where c_3 is an absolute positive constant.

Note that Lemmas 8 and 9 allow to bound the LRC in polynomial time $O(n^{2.4})$ [35].

At this point, it is possible to prove the LRC-based counterpart of Corollary 7. Note that, at the best of our knowledge, the following result is novel.

Corollary 10: Let us consider an hypotheses space $\mathcal{H}$ such that $h(X) = W \cdot \phi(X)$ with $\|W\|^2 \leq H^2$ and define $\widehat{r}_n$ as the fixed point of the following subroot function:

$$\widehat{\psi}_n(r) = \sup_{\alpha\in(0,1]} \alpha\frac{2H}{n}\sqrt{\sum_{i=1}^n \min\left[\frac{c_2(\alpha, r)}{4}, \lambda_i\right]} + \sqrt{\frac{2x}{n}} \tag{26}$$

where $c_1(\alpha, r) = (1/\alpha^2)(3r+((x/2n))^{1/2})$ *and* $c_2(\alpha, r) = 4nc_1(\alpha, r) - \|\Phi W^* - \mathbf{y}\|^2$. *Then,* $\forall h \in \mathcal{H}$ *and with probability at least* $(1 - 3e^{-x})$, *we have that*

$$L(h) \leq \min_{K \in (1,+\infty)} \frac{K}{K-1} \widehat{L}_n(h) + K\widehat{r}_n + 2\sqrt{\frac{x}{2n}}. \quad (27)$$

The proof is reported in Appendix VIII-B.

Based on the results of Corollary 7 and 10, an important factor for understanding the generalization ability of a kernel is to study the GRC or LRC and the empirical error. The RC represents the expressiveness of the kernel and its complexity; this means that the higher it is the value, the more configurations of the labels can be learned by the kernel but, at the same time, the risk of over-fitting the data is larger. Therefore, one has to choose the kernel with the best tradeoff between accuracy on the data, namely, small empirical error, and expressiveness, namely, GRC or LRC.

Note that, even if in this paper we have always exploited the Euclidean norm $\|\cdot\|_2$, a desirable result would be to extend the derived properties to other norms such as the Manhattan one $\|\cdot\|_1$ or, more generally, the Lp-norm $\|\cdot\|_p$ [36]. Unfortunately such extension is not trivial. In fact, properties such as the one described in Lemma 4 cannot be easily extended since both upper and lower bounds to the size of the space of functions are needed. An example of such nontrivial extension is the one reported in [17] where a notion of LRC for MKL has been derived. These limitations are peculiar of LRC. Indeed, for GRC, it is trivial to prove that, given an hypotheses space $\mathcal{H}^a$ [such that $h(X) = W \cdot \phi(X)$, $\|W\|_p \leq H$, and $p \leq 2$], and another hypotheses space $\mathcal{H}^b$ [such that $h(X) = W \cdot \phi(X)$ with $\|W\|_2 \leq H$], then $\mathcal{R}_n(\mathcal{H}^a) \leq \mathcal{R}_n(\mathcal{H}^b)$.

VI. Graph Kernels

In this paper, we focus on the space of graphs. A *graph* is a 3-tuple $G = (V_G, E_G, L_G)$, where $V_G = \{1, \ldots, n_v\}$ is the set of *vertices* (or *nodes*), $E_G = \{(i, j)|i, j \in V_G\}$ is the set of *edges* (with $|E_G| = m$), and $L_G : V_G \to \Sigma$ is a function mapping each vertex to a discrete label in a fixed alphabet Σ. A graph is *undirected* if $(i, j) \in E_g \Rightarrow (j, i) \in E_G$, otherwise it is *directed*. ρ is the maximum out-degree (or number of outgoing edges) of a node in a graph, i.e., $\max_{i \in V} |\{(i, j) : (i, j) \in E\}|$. A *random walk* is a sequence of vertices $v_1 \cdots v_l$ where $(v_i, v_{i+1}) \in E_G$. A *path* is a random walk where all vertices are distinct from one another. A *cycle* is a walk where $v_1 = v_l$. A *tree* is a *directed acyclic graph* where one vertex has no incoming edges.

In [26], it is shown that complete graph kernels (kernels that map each nonisomorphic graph in a different point in the feature space) are hard to compute. Thus, efficient graph kernels that have been proposed in literature (and the ones that are used in practice) are not complete. Early works on graph kernels were based on random walks. The marginalized graph kernel considers common walks as features [37] (the work has been extended in order to make it more efficient and effective in [38]). Informally, this kernel is defined as the expected value of a kernel over all possible pairs of label sequences generated by random walks on two graphs. The worst case time complexity of the algorithm presented in [39] is $O(|V_G|^3)$. One of the main drawbacks of this kernel is the tottering problem, i.e., since vertices can appear multiple times in a random walk, a random walk of any length can involve only a very small subset of the vertices of a graph (moving backward and forward), inducing an artificially high similarity between two graphs. This phenomena can negatively influence the discriminative capabilities of the kernel. In [24], a variant of this kernel that considers random walks up to a fixed length p (thus limiting to a certain extent tottering) is presented. We will refer to this kernel as η-RW. An example of feature generated by this kernel for the graph in Fig. 1(a) is depicted in Fig. 1(c). Then [40] proposed to consider shortest paths instead of walks. The Shortest Path Kernel associates a feature to each pair of nodes of one graph. The value of the feature is the length of the shortest path between the corresponding nodes in the graph. The complexity of the kernel is $O(|V_G|^4)$. The *graphlet* kernel [41] counts all types of matching subgraphs of small size k (e.g., $k = 3, 4$ or 5). There are efficient schemes for computing this kernel, but they are applicable only on unlabeled graphs. For the labeled case, the computational complexity of this kernel is $O(|V_G|^k)$.

The Weisfeiler–Lehman (WL) graph kernels [24], [42] are based on the recursive WL color refinement procedure. The principal member of this family, the Fast Subtree WL kernel, in an efficient way ($O(|E_G|)$), maps a graph in a RKHS where each feature represents a subtree-walk pattern (subtrees where vertices can appear multiple times). The value associated with a feature is the frequency of the particular subtree-walk in the input graph. WL kernel is computed in an iterative fashion, that stops after η (user-specified parameter) iterations. The number of nonzero features associated with a graph is at most $|V_G|\eta$. Fig. 1(b) shows an example of a feature generated by this kernel.

The ordered decomposition DAGs (ODDs) kernel framework [25] considers as nonzero features in the RKHS the trees that appear as subtrees of the input graphs. It exploits the shortest path (up to length η) DAG decompositions starting from each node in the graph to generate DAG structures [Fig. 1(d)], and then extracts tree features from them. Each tree-feature is weighted as $f \cdot \omega^{\dim}$, where f is the frequency of the feature, dim its dimension (the number of vertices in the tree) and $\omega > 0$ a weighting parameter. The time complexity of

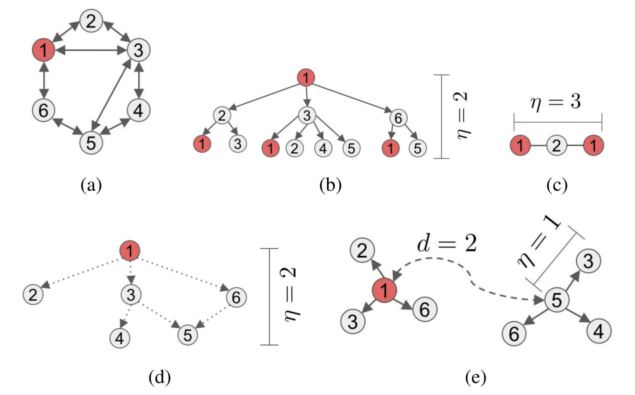

Fig. 1. Example of (a) graph and (b)–(e) features generated by different graph kernels.

the main representative of this family, ODD$_{\text{ST}}$, is, under mild conditions, $O(|V_G|\log|V_G|)$, and the number of generated features for a graph is at most $|V_G|\rho^\eta$.

The neighborhood subgraph pairwise distance kernel (NSPDK) [23] considers as features pairs of small-sized subgraphs (up to radius η) that appear in an input graph at a (shortest path) distance of at most d [see Fig. 1(e)]. The number of features generated by this kernel is $(\eta|V_G|\rho^d/2)$, that lies between $\eta|V_G|$ and $(\eta|V_G|^2/2)$. Note that WL is very close to a degenerate case of NSPDK, where $d = 0$. In Section VII, we will consider the η-RW, as a representative of the broad family of random walk graph kernels, and the ODDST, WL and NSPDK kernels, which are considered state-of-the-art from both the predictive power and computational complexity points of view.

VII. Experimental Results

In this section, we present an empirical comparative analysis among the estimated generalization error, the 10CV (the KCV error [8] with $k = 10$), the expressiveness score derived from the LRC of a kernel, and the error on the training set of the trained model.

The experiments have been performed on a total of six data sets. The data sets involve chemo and bioinformatics data: MUTAG [43], CAS,[1] CPDB [44], AIDS [45], NCI1, and NCI109 [46]. The data sets are composed by 188, 4337, 684, 1503, 4110, and 4127 examples, respectively. See [25] for more details. All the data sets involve chemical compounds represented as graphs and involve binary classification problems. The nodes are labeled according to the associated atom type and the edges represent the bonds.

As for the kernels, we considered the WL, ODD$_{\text{ST}}$, NSPDK and η-RW, detailed in Section VI. For each kernel, we defined the grid of hyperparameters as follows. For all the kernels, we set $\eta \in \{1, \ldots, 8\}$. Moreover, for NSPDK, we set $d \in \{1, \ldots, 8\}$, and for ODD$_{\text{ST}}$, we set $\lambda \in \{0.5, 0.7, 0.9, 1.0, 1.1, 1.2, 1.4, 1.6, 1.8\}$ Finally, we set the SVM C parameter in the following range $\{10^{-10}, \ldots, 10^7\}$.

In (5), the bound on the generalization error based on the GRC has been provided. Based on the results of the previous sections, it is possible to note that the GRC upper bound is proportional to $(1/n)(\sum_{i=1}^n \lambda_i)^{1/2}$. Instead the LRC is a function of a parameter $r \in [0, 1]$, that is the maximum empirical square loss of the considered hypothesis, and its upper bound is proportional to $(1/n)(\sum_{i=1}^n \min[nr, \lambda_i])^{1/2}$. We define here an intuitive way to derive an expressivity measure from the LRC curve. Fixed a kernel and its parameters, it is possible to obtain a plot similar to the one reported in Fig. 2. Obviously, the more slope the LRC curve has, the more expressive the kernel is, since for a fixed r, the higher the slope, the higher the LRC will be. For this reason, we defined the RC ratio (RCR) measure as the ratio between the area under the LRC curve and the GRC. We would like to point out that other measures can be defined in order to summarize the information in the plot. Our proposal is just a possibility that responds to our desiderata.

[1]http://www.cheminformatics.org/datasetsbursi

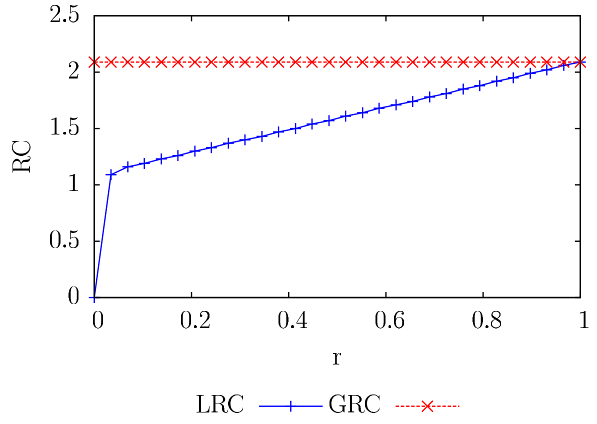

Fig. 2. LRC as a function of r and GRC, referred to the WL kernel with $\eta = 8$ on the CAS data set.

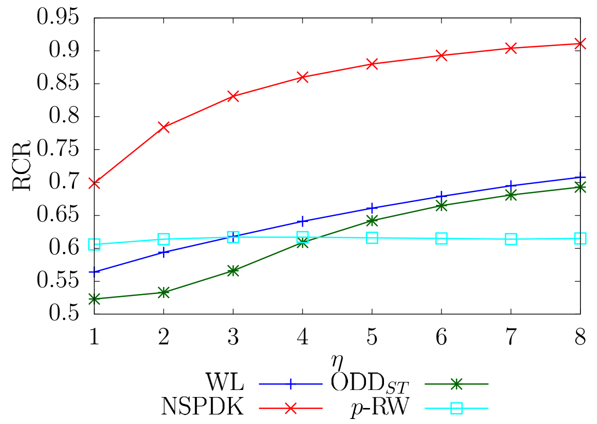

Fig. 3. RCR as a function of η for WL, NSPDK, and ODD$_{\text{ST}}$ kernels on AIDS data set. For NSPDK and ODD$_{\text{ST}}$, the reported value is an average among the other parameters' configurations.

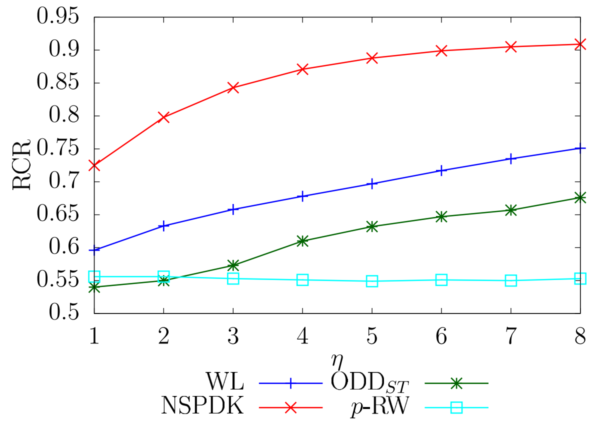

Fig. 4. RCR as a function of η for WL, NSPDK, and ODD$_{\text{ST}}$ kernels on CAS data set. For NSPDK and ODD$_{\text{ST}}$, the reported value is an average among the other parameters' configurations.

In the first set of experiments, we adopted the RCR measure to compare the expressiveness of different graph kernels. The purpose of these experiments is twofold. First, we would like to understand whether RCR measure is suited for analyzing the complexity of graph kernels. From the definitions of the considered graph kernels, it is reasonable to expect that, for

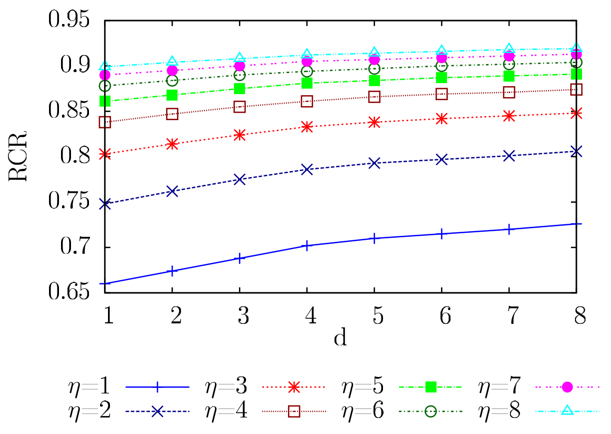

Fig. 5. RCR as a function of d parameter for NSPDK kernel on AIDS data set.

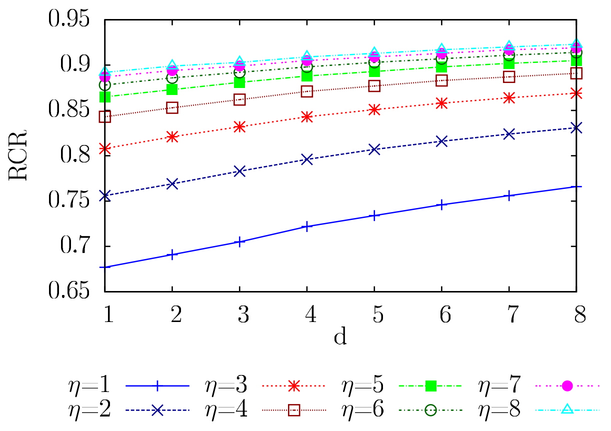

Fig. 6. RCR as a function of d parameter for NSPDK kernel on CAS data set.

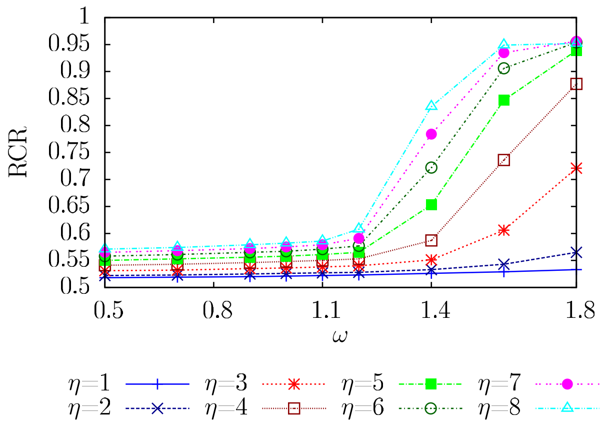

Fig. 7. RCR as a function of ω parameter for ODD$_{\text{ST}}$ kernel on AIDS data set.

all of them, incrementing the η parameter should result in an increased kernel complexity. This is true for NSPDK, WL and ODD$_{\text{ST}}$ kernels on all the considered data sets. We report the plots corresponding to AIDS and CAS data sets in Figs. 3 and 4, respectively. For the other data sets, the situation is similar and thus the plots are omitted. Note, however, that for η-RW kernel, the RCR measure slightly increases up to $\eta = 3$, and then remains more or less stable.

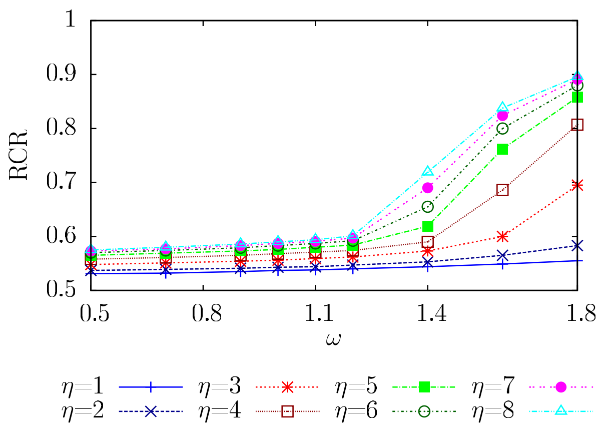

Fig. 8. RCR as a function of ω parameter for ODD$_{\text{ST}}$ kernel on CAS data set.

TABLE I

CROSS CORRELATION BETWEEN π AND THE 10CV ERROR OF THE MODEL TRAINED WITH DIFFERENT KERNELS, DIFFERENT CONFIGURATIONS OF THE HYPERPARAMETERS, AND DIFFERENT DATA SETS. MISSING VALUES (-) INDICATE THAT THE KERNEL MATRIX COMPUTATION DID NOT FINISH IN 24 h

Kernel	Dataset	Correlation
NSPDK	AIDS	0.995
	CAS	0.991
	CPDB	0.993
	MUTAG	0.997
	NCI1	0.987
	NCI109	0.994
ODD$_{ST}$	AIDS	0.987
	CAS	0.987
	CPDB	0.996
	MUTAG	0.994
	NCI1	0.990
	NCI109	0.990
WL	AIDS	0.996
	CAS	0.993
	CPDB	0.996
	MUTAG	0.996
	NCI1	0.990
	NCI109	0.992
η-RW	AIDS	0.998
	CAS	0.998
	CPDB	0.998
	MUTAG	0.999
	NCI1	-
	NCI109	-

This is because of the tottering problem we mentioned in Section VI, where introducing more complex features actually does not augment the kernel complexity. We can argue that this happens because, increasing the η parameter, more and more of the generated features contain repeated vertices. These features are noisy, thus may negatively impact the similarity measure induced by the kernel.

This is a known problem with kernels based on random walks, that is also confirmed by the classification performance of this kernel being the poorest among the considered kernels on all the data sets we tested (results are not reported here). Thus in this case, the RCR measure successfully captures also the weak points of this kernel.

As for the NSPDK and ODD$_{\text{ST}}$ kernels, both of them depend on a second parameter, d and ω, respectively. It is interesting to point out that the growth in complexity is

TABLE II

PERFORMANCE OF RC FOR KERNEL PARAMETER SELECTION AGAINST THE 10CV FOR DIFFERENT DATA SETS AND KERNELS. Δ IS THE PERFORMANCE GAP (IN PERCENTAGE) BETWEEN THE MODEL BUILT WITH SET OF HYPERPARAMETERS $\Theta_{\text{CV}}^{\mathcal{C}}$ AND THE ONE BUILT WITH $\Theta_{\text{CV}}^{\mathcal{C}^{\epsilon}}$ FOR DIFFERENT VALUES OF ϵ

Dataset	Kernel	ϵ	Δ	c^{ϵ}	$100\, c^{\epsilon}/c$
AIDS	ODD_{ST}	0.0	4.0	1	0.1
		0.001	4.0	2	0.2
		0.01	3.3	7	0.6
		0.05	2.5	62	5.1
		0.1	2.1	148	12.2
		0.2	0.5	198	16.3
		0.3	0.0	261	21.5
	NSPDK	0.0	9.8	1	0.1
		0.001	9.8	2	0.2
		0.01	9.1	3	0.2
		0.05	5.7	14	1.2
		0.1	4.0	39	3.2
		0.2	0.4	104	8.6
		0.4	0.2	485	40.0
		0.5	0.0	600	49.5
	WL	0.0	4.8	1	0.7
		0.001	4.8	1	0.7
		0.01	3.7	3	2.0
		0.05	2.3	9	5.9
		0.1	1.5	22	14.5
		0.2	0.2	46	30.3
		0.3	0.0	55	36.2
	η-RW	0.0	1.7	1	0.6
		0.001	1.7	2	1.2
		0.01	1.7	20	11.8
		0.05	0.8	40	23.5
		0.1	0.0	93	54.7
CAS	ODD_{ST}	0.0	13.5	1	0.1
		0.001	12.9	2	0.2
		0.01	1.9	10	0.8
		0.05	0.7	91	7.5
		0.1	0.3	109	9.0
		0.2	0.0	154	12.7
	NSPDK	0.0	5.4	1	0.1
		0.001	5.4	2	0.2
		0.01	5.3	4	0.3
		0.05	1.2	25	2.1
		0.1	0.1	62	5.1
		0.2	0.0	177	14.6
	WL	0.0	2.9	1	0.7
		0.001	2.3	2	1.3
		0.01	1.7	4	2.6
		0.05	0.6	13	8.6
		0.1	0.0	32	21.1
	η-RW	0.0	0.0	1	0.6
NCI1	ODD_{ST}	0.0	1.3	1	0.1
		0.001	1.3	1	0.1
		0.01	0.9	10	0.7
		0.05	0.4	86	6.3
		0.1	0.0	179	13.1
	NSPDK	0.0	0.0	3	0.2
	WL	0.0	1.9	1	0.7
		0.0001	0.9	5	3.4
		0.01	0.9	5	3.4
		0.05	0.0	20	13.4
MUTAG	ODD_{ST}	0.0	12.2	5	0.4
		0.001	12.2	5	0.4
		0.01	12.2	10	0.8
		0.05	8.4	20	1.6
		0.1	4.8	38	3.1
		0.2	3.8	106	8.7
		0.3	0.0	192	15.7
	NSPDK	0.0	8.5	5	0.5
		0.001	8.5	5	0.5
		0.01	8.1	20	1.8
		0.05	1.3	139	12.8
		0.1	0.0	301	27.7
	WL	0.0	8.9	1	0.7
		0.001	8.9	1	0.7
		0.01	8.9	2	1.5
		0.05	0.0	6	4.4
	η-RW	0.0	3.5	1	0.6
		0.001	3.5	1	0.6
		0.01	3.1	5	2.9
		0.05	0.0	136	80.0
CPDB	ODD_{ST}	0.0	2.5	1	0.1
		0.001	2.5	1	0.1
		0.01	0.5	13	1.0
		0.05	0.0	134	9.8
	NSPDK	0.0	7.0	1	0.1
		0.001	6.7	2	0.2
		0.01	3.1	4	0.3
		0.05	0.6	18	1.5
		0.1	0.1	41	3.4
		0.2	0.0	56	4.6
	WL	0.0	1.8	1	0.7
		0.001	0.8	2	1.3
		0.01	0.4	4	2.6
		0.05	0.0	11	7.2
	η-RW	0.0	0.1	1	0.6
		0.001	0.0	2	1.2
NCI109	ODD_{ST}	0.0	17.1	1	0.1
		0.001	17.1	3	0.2
		0.01	17.1	5	0.4
		0.05	13.5	21	1.7
		0.1	10.1	33	2.7
		0.2	5.4	50	4.1
		0.5	0.2	172	14.1
		0.7	0.0	253	20.7
	NSPDK	0.0	7.4	1	0.1
		0.001	5.6	4	0.4
		0.01	5.4	15	1.4
		0.05	3.8	91	8.4
		0.1	1.9	197	18.1
		0.2	0.1	327	30.1
		0.3	0.0	355	32.6
	WL	0.0	5.2	1	0.7
		0.001	5.2	1	0.7
		0.01	5.2	1	0.7
		0.05	1.3	6	4.4
		0.1	0.0	30	22.1

different in the two kernels. Figs. 5 and 6 show the situation for NSPDK on AIDS and CAS data sets, while Figs. 7 and 8 are referred to ODD_{ST}. For NSPDK, fixed an η, the growth in complexity is linear in d, while for ODD_{ST} different η values show different shapes for the complexity curves as a function of ω.

In the second set of experiments, we try to understand whether the proposed RCR, and in general the approximated computation of the LRC presented in Corollary 10, can be useful in the process of kernel/parameter selection for kernels for structured data. For this reason, we computed for each kernel/parameter configuration Θ the following measures: the RCR measure on the LRC curve of the kernel, namely, $\text{RCR}(\Theta)$, the error on the training set, namely, $\text{Err}_{\text{Tr}}(\Theta)$, and the 10CV error $\text{Err}_{\text{CV}}(\Theta)$. We then computed the sum of the RCR measure and the error on the training set. We want to understand whether it is possible to predict the 10CV using only measures derived from the training set. Such a relationship is very interesting for different reasons. First, this is a nondisruptive method for including the LRC, and hence notions from SLT, in the model selection process. Moreover, if the adopted data sets are large, analyzing the results incrementally, it is possible to avoid the expensive resampling procedures for the least promising parameter configurations. Table I reports the normalized cross correlation measure [47] among the sum of the RCR measure and the error on the training set, namely $\pi(\Theta) = \text{RCR}(\Theta) + \text{Err}_{\text{Tr}}(\Theta)$, and the generalization error estimated via 10CV, that we refer as $\text{Err}_{\text{CV}}(\Theta)$, that is the measure we would like to approximate. From the table emerges that π is strongly correlated with the 10CV error. From this correlation relationship, we can argue that it is possible to estimate the relative performances of two kernels (in the sense of their generalization error), that is to give a fairly accurate prediction of which one will have the better predictive ability, comparing their π.

Note that the results of Sections IV and V are quite general and can be exploited, analogously to what has been done in this section, for the selection of kernels and kernel hyperparameters of any kind. In fact, in this paper, we did not make any assumption on the kernel type.

A. Toward RC for Parameter Selection

In this section, we show some results regarding the use of RC, in terms of the previously introduced π measure, for selecting the hyperparameters of a graph kernel.

In Section VII, we presented the correlation between π and the generalization error estimated via 10CV. This correlation can be obviously exploited to discover a set of good configurations of the graph kernel hyperparameters.

In order to show this result, let us define some quantities. Let us fix the considered data set and kernel, for ease of notation. Let $\mathcal{C}$ be the set of all the considered hyperparameter configurations, with $|\mathcal{C}| = c$. Let $\Theta^{\mathcal{C}}_{\text{CV}}$ be the set of hyperparameters corresponding to the minimum 10CV error, i.e., $\Theta^{\mathcal{C}}_{\text{CV}} = \arg\min_{\theta\in\mathcal{C}} \text{Err}_{\text{CV}}(\Theta)$. Note that $\Theta^{\mathcal{C}}_{\text{CV}}$ may be not unique. In that case, we can just randomly pick one of the configurations with minimum value. Let $\pi^*_{\mathcal{C}} = \min_{\Theta\in\mathcal{C}} \pi(\Theta)$.

Let us also consider the set $\mathcal{C}^{\epsilon} \subseteq \mathcal{C}$ of hyperparameters configurations that have π value close to $\pi^*_{\mathcal{C}}$, i.e., $\mathcal{C}^{\epsilon} = \{\Theta \in \mathcal{C} | \pi(\Theta) \leq (1+\epsilon)\pi^*_{\mathcal{C}}\}$, with $|\mathcal{C}^{\epsilon}| = c^{\epsilon}$. In such a set of hyperparameters configurations, we indicate with $\Theta^{\mathcal{C}^{\epsilon}}_{\text{CV}}$ the one corresponding to the minimum 10CV error, i.e., $\Theta^{\mathcal{C}^{\epsilon}}_{\text{CV}} = \arg\min_{\Theta\in\mathcal{C}^{\epsilon}} \text{Err}_{\text{CV}}(\Theta)$.

In Table II, we report, for each data set and graph kernel and for different values of ϵ, the performance gap (in percentage) $\Delta = (\text{Err}_{\text{CV}}(\Theta^{\mathcal{C}^{\epsilon}}_{\text{CV}}) - \text{Err}_{\text{CV}}(\Theta^{\mathcal{C}}_{\text{CV}})) \cdot 100$, between the model built with set of hyperparameters Θ_{CV} and the one built with $\Theta^{\mathcal{C}^{\epsilon}}_{\text{CV}}$.

Based on Table II, it is possible to observe the following.

1) Even when $\epsilon = 0$ in some cases (in particular with the η-RW kernel), Δ is quite small.
2) As expected, the larger ϵ is the larger c^{ϵ} becomes; consequently, the larger is the percentage of configurations that have π value ϵ-close to $\pi^*_{\mathcal{C}}$ [i.e., $100\,(c^{\epsilon}/c)$], the smaller Δ becomes.
3) The results show that the proposed methodology is able to identify (and rank) a small subset of possible optimal hyperparameters candidates.

TABLE III

PERFORMANCE AND TIME REQUIREMENTS OF RC FOR KERNEL PARAMETER SELECTION AGAINST THE 10CV FOR THE AIDS DATA SET. Δ IS THE PERFORMANCE GAP (IN PERCENTAGE) BETWEEN THE MODEL BUILT WITH SET OF HYPERPARAMETERS $\Theta^{\mathcal{C}}_{\text{CV}}$ AND THE ONE BUILT WITH $\Theta^{\mathcal{C}^{\epsilon}}_{\text{CV}}$ FOR DIFFERENT VALUES OF ϵ

Kernel	Method	ϵ	Δ	c^{ϵ}	$100\, c^{\epsilon}/c$	Time (s)
ODD_{ST}	Proposed	0.0	4.0	1	0.1	**18615**
		0.001	4.0	2	0.1	**18660**
		0.01	3.3	7	0.5	**18870**
		0.05	2.5	62	4.5	**21097**
		0.1	2.1	162	11.8	**25245**
		0.2	0.5	226	16.5	**28142**
		0.3	0.0	292	21.3	**31317**
	10CV	–	–	1368	100	209422
NSPDK	Proposed	0.0	10.1	1	0.1	**7462**
		0.001	10.0	2	0.2	**7492**
		0.01	9.2	3	0.2	**7526**
		0.05	5.9	15	1.2	**8055**
		0.1	4.7	40	3.3	**9197**
		0.2	0.5	106	8.7	**12464**
		0.3	0.5	248	20.4	**52319**
		0.4	0.1	478	39.3	**61456**
		0.5	0.0	602	49.5	**66326**
	10CV	–	–	1216	100	82783
WL	Proposed	0.0	4.8	1	0.7	**3974**
		0.001	4.8	1	0.7	**3974**
		0.01	3.7	3	2.0	**4065**
		0.05	2.3	9	5.9	**4340**
		0.1	1.5	22	14.5	**4814**
		0.2	0.2	46	30.3	**5788**
		0.3	0.0	55	36.2	**7902**
	10CV	–	–	152	100	57058
η-RW	Proposed	0.0	1.7	1	0.6	**6149**
		0.001	1.7	2	1.2	**13178**
		0.01	1.7	20	11.8	**54316**
		0.05	0.8	40	23.5	82095
		0.1	0.0	93	54.7	84074
	10CV	–	–	170	100	81366

4) By considering less than the 25% of the possible hyperparameters configurations, it is possible to reach a $\Delta < 1.0$.

Finally, it is worth to point out that the percentage of configurations $100\,(c^{\epsilon}/c)$ needed to obtain a $\Delta = 0$ is in turn an indicator of the robustness of the considered kernel with respect to its hyperparameters. Moreover, on the biggest data sets (NCI1 and NCI109), $100(c^{\epsilon}/c)$ is generally small, showing that our proposed complexity measure π becomes more consistent with the increase of available data.

B. Computational Times

In the previous sections, we considered 10CV as performance evaluation procedure. However, one can argue that 10CV is too computational expensive and a faster alternative should be adopted, for example, the 5CV. In fact, given a set of N kernel matrices (a matrix for each hyperparameter configuration, each one of size $n \times n$) to consider the set of values for the hyperparameter C of SVM, a performance estimation procedure, the KCV, and an ϵ value, with our approach we have to carry out the following:

1) Compute π for each kernel matrix, with a computational complexity of $O(Nn^{2.3})$.
2) Sort such a list, with a complexity of $O(N \log N)$.

TABLE IV

PERFORMANCE AND TIME REQUIREMENTS OF RC FOR KERNEL PARAMETER SELECTION AGAINST THE 5CV FOR THE AIDS DATA SET. Δ IS THE PERFORMANCE GAP (IN PERCENTAGE) BETWEEN THE MODEL BUILT WITH SET OF HYPERPARAMETERS $\Theta_{\text{CV}}^{\mathcal{C}}$ AND THE ONE BUILT WITH $\Theta_{\text{CV}}^{\mathcal{C}^\epsilon}$ FOR DIFFERENT VALUES OF ϵ

Kernel	Method	ϵ	Δ	c^ϵ	$100\,{c^\epsilon}/{c}$	Time (s)
ODD_{ST}	Proposed	0.0	4.4	1	0.1	**18599**
		0.001	4.4	2	0.1	**18615**
		0.01	4.2	13	1.0	**18795**
		0.05	2.6	63	4.6	**19665**
		0.1	2.4	165	12.1	**21561**
		0.2	0.7	226	16.5	**22731**
		0.3	0.0	296	21.6	**24097**
	5CV	–	–	1368	100	86910
NSPDK	Proposed	0.0	10.1	1	0.1	**7432**
		0.001	10.0	2	0.2	**7454**
		0.01	9.2	3	0.2	**7476**
		0.05	5.9	15	1.2	**7693**
		0.1	4.7	40	3.3	**8258**
		0.2	0.5	106	8.7	**9683**
		0.3	0.5	248	20.4	**19767**
		0.4	0.1	478	39.3	**23866**
		0.5	0.0	602	49.5	**26084**
	5CV	–	–	1216	100	29421
WL	Proposed	0.0	6.0	1	0.7	**3988**
		0.001	6.0	1	0.7	**3988**
		0.01	5.3	3	2.0	**4107**
		0.05	2.8	9	5.9	**4220**
		0.1	1.6	22	14.5	**4830**
		0.2	0.2	45	29.6	**5215**
		0.3	0.0	55	36.2	26032
	5CV	–	–	152	100	23745
η-RW	Proposed	0.0	2.3	1	0.6	**6019**
		0.001	2.3	5	2.9	**9225**
		0.01	2.2	20	11.8	**27567**
		0.05	1.1	40	23.5	37445
		0.1	0.2	51	30.0	37652
		0.2	0.0	170	100.0	39747
	5CV	–	–	170	100	34088

3) Compute KCV for all the M configurations in $\mathcal{C}^\epsilon$, with a complexity of $O(MKn^{2.3})$.
4) Return the configuration with lower KCV error.

Therefore, the overall complexity of our procedure is $O((N+MK)n^{2.3})$. In Tables III and IV, we report, for our proposed method, the computational time required for performing such a procedure given the corresponding ϵ value. On the other hand, for 5CV and 10CV, we report the computational time required for standard cross validation, that has a complexity of $O(NKn^{2.3})$. Moreover, in general, different values of K lead to slightly different optimal parameter configurations [48].

Indeed, by comparing Tables III and IV, it is possible to notice that the best configuration for the η-RW kernel is discovered with an $\epsilon = 0.2$ in the case of $5CV$ and $\epsilon = 0.1$ in the case of $10CV$ (note that the order in which the different parameter configurations are explored by our proposed method is independent of the adopted performance estimation procedure). In order to compare the computational requirements of the different methods, because of space constraints, we decided to analyze the AIDS data set, that is one of the data sets where our proposed method takes more time in order to find the best parameter configuration. The computational times in Tables III and IV are computed as follows.

In Table III, we reported in bold the computational times relative to the ϵ values where our proposed method is faster than the 10CV. It is possible to notice that for the ODD_{ST} and WL kernel, the speedup is considerable (one order of magnitude). For the $NSPDK$ kernel, the speedup is lower (roughly 25%). Finally, for the η-RW kernel, while our proposed method is not convenient for obtaining the optimal configuration, with $\epsilon = 0$, it is possible to obtain a reasonably accurate configuration (i.e., $\Delta = 1.7$), at a fraction of the time required for 10CV. Note also that our method is more effective as the number of considered hyperparameters increases. The WL and η-RW kernels have a lower number of total hyperparameters combinations (i.e., only two hyperparameters have to be validated); thus for these two kernels, the proposed approach may be less useful than for other kernels.

Let us now focus on the 5CV results in Table IV. Obviously, the running times for 5CV are lower with respect to the previously reported ones relative to 10CV. As for our proposed method, the scenario is similar, being it faster than 5CV when using ODD_{ST} and $NSPDK$ kernels. For the WL kernel, our method is able to recover an almost-optimal hyperparameters configuration (i.e., $\Delta = 0.2$) in almost $(1/5)$ of the computational time required from 5CV. Finally, for the η-RW kernel, we have to allow a higher error ($\Delta = 2.3$) for our method in order to be faster than 5CV.

VIII. CONCLUSION

In this paper, we proposed a new method for bounding the LRC of the class of linear separators in a RKHS, starting from a kernel matrix. For the first time, we use the notion of LRC for studying the properties of three state-of-the-art graph kernels. Our analysis confirms some empirically known expressivity properties and support them with an adequate theoretical background. Moreover, we showed that the 10CV error on six real-world graph data sets is correlated with the sum of the training error and the RCR, an expressivity measure defined in this paper based on the LRC notion of complexity. The correlation between these two functions is a desirable property. In fact, based on this correlation, an analysis of the error on the training set and the RCR would be enough to select the optimal kernel/parameter configuration. We are still not able to fully reach such a result, but the experiments show that the proposed measure has promising properties, and thus can be exploited in the future research for the design of efficient model/parameter selection procedures.

APPENDIX
PROOFS

In this Appendix, we report the proofs of the results derived in the paper.

A. *Proof of Lemma 4*

Proof: In order to prove our statement, let us reformulate the first term of the inequality

$$\left\{ W \,\middle|\, \|W\|^2 \le H^2, \frac{1}{n}\sum_{i=1}^{n}\left[\frac{1-YW\cdot\phi(X)}{2}\right]^2 \le c_1 \right\}$$
$$\equiv \{W \,\big|\, \|W\|^2 \le H^2, \|\Phi W - \boldsymbol{y}\|^2 \le 4nc_1\}. \quad (28)$$

Let us add and subtract the same quantity in the above constraint and let us exploit Lemma 3

$$\begin{aligned}
&\{W \,\big|\, \|W\|^2 \leq H^2, \|\Phi W - \boldsymbol{y}\|^2 \leq 4nc_1\} \\
&\equiv \{W \,\big|\, \|W\|^2 \leq H^2, \|\Phi W - \boldsymbol{y}\|^2 - \|\Phi W^* - \boldsymbol{y}\|^2 \\
&\leq 4nc_1 - \|\Phi W^* - \boldsymbol{y}\|^2\} \\
&\subseteq \{W \,\big|\, \|W\|^2 \leq H^2, \|\Phi(W - W^*)\|^2 \leq 4nc_1 - \|\Phi W^* - \boldsymbol{y}\|^2\} \\
&\subseteq \{W \,\big|\, \|W\|^2 \leq 4H^2, \|\Phi W\|^2 \leq 4nc_1 - \|\Phi W^* - \boldsymbol{y}\|^2\}. \quad (29)
\end{aligned}$$

This concludes our proof. ■

B. Proof of Corollary 10

Proof: In order to prove the statement, we just have to combine Theorem 2 with Lemmas 4 and 8. In particular, by considering Theorem 2, we have that

$$\widehat{R}_n(\widehat{\mathcal{H}}_{r,\alpha,x,n}) = \widehat{R}_n\big(\big\{h \,\big|\, h \in \mathcal{H}, \widehat{L}_n^2(h) \leq c_1(\alpha, r)\big\}\big). \quad (30)$$

Then, by considering Lemma 4, we have that

$$\widehat{R}_n(\widehat{\mathcal{H}}_{r,\alpha,x,n}) \leq \mathbb{E}_{\boldsymbol{\sigma}} \sup_{\{W \mid \|W\|^2 \leq 4H^2, \|\Phi W\|^2 \leq c_2(\alpha,r)\}} \frac{1}{n}\sum_{i=1}^n \sigma_i W \cdot \phi(X_i). \quad (31)$$

Finally by exploiting Lemma 8, we get

$$\widehat{R}_n(\widehat{\mathcal{H}}_{r,\alpha,x,n}) \leq \frac{2H}{n}\sqrt{\sum_{i=1}^n \min\left[\frac{c_2(\alpha, r)}{4}, \lambda_i\right]}. \quad (32)$$

By plugging the last result again in Theorem 2, we get our proof. ■

References

[1] B. Schölkopf and A. J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2002.

[2] V. N. Vapnik, *Statistical Learning Theory*. Hoboken, NJ, USA: Wiley, 1998.

[3] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *Ann. Statist.*, vol. 36, no. 3, pp. 1171–1220, 2008.

[4] G. Bakir, T. Hofman, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan, *Predicting Structured Data*. Cambridge, MA, USA: MIT Press, 2007.

[5] M. Gönen and E. Alpaydin, "Multiple kernel learning algorithms," *J. Mach. Learn. Res.*, vol. 12, pp. 2211–2268, Jul. 2011.

[6] C. Cortes, M. Kloft, and M. Mohri, "Learning kernels using local rademacher complexity," in *Proc. Neural Inf. Process. Syst.*, 2013, pp. 2760–2768.

[7] F. Aiolli and M. Donini, "EasyMKL: A scalable multiple kernel learning algorithm," *Neurocomputing*, vol. 169, pp. 215–224, Dec. 2015.

[8] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. Int. Joint Conf. Artif. Intell.*, 1995, pp. 1137–1145.

[9] D. Anguita, A. Ghio, L. Oneto, and S. Ridella, "In-sample and out-of-sample model selection and error estimation for support vector machines," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 9, pp. 1390–1406, Sep. 2012.

[10] M. Talagrand, "The Glivenko-Cantelli problem," *Ann. Probab.*, vol. 15, no. 3, pp. 837–870, 1987.

[11] V. Koltchinskii, "Rademacher penalties and structural risk minimization," *IEEE Trans. Inf. Theory*, vol. 47, no. 5, pp. 1902–1914, Jul. 2001.

[12] P. L. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: Risk bounds and structural results," *J. Mach. Learn. Res.*, vol. 3, pp. 463–482, Mar. 2003.

[13] L. Oneto, A. Ghio, S. Ridella, and D. Anguita, "Global rademacher complexity bounds: From slow to fast convergence rates," *Neural Process. Lett.*, vol. 43, no. 2, pp. 567–602, 2016.

[14] P. L. Bartlett, O. Bousquet, and S. Mendelson, "Local rademacher complexities," *Ann. Statist.*, vol. 33, no. 4, pp. 1497–1537, 2005.

[15] V. Koltchinskii, "Local rademacher complexities and oracle inequalities in risk minimization," *Ann. Statist.*, vol. 34, no. 6, pp. 2593–2656, 2006.

[16] L. Oneto, A. Ghio, S. Ridella, and D. Anguita, "Local rademacher complexity: Sharper risk bounds with and without unlabeled samples," *Neural Netw.*, vol. 65, pp. 115–125, May 2015.

[17] M. Kloft and G. Blanchard, "The local rademacher complexity of ℓ_p-norm multiple kernel learning," in *Proc. Neural Inf. Process. Syst.*, 2011, pp. 2438–2446.

[18] A. Lei, Y. Binder, Ü. Dogan, and M. Kloft, "Theory and algorithms for the localized setting of learning kernels," *J. Mach. Learn. Res.*, vol. 44, pp. 173–195, 2015.

[19] D. Anguita, A. Ghio, L. Oneto, and S. Ridella, "A deep connection between the Vapnik-chervonenkis entropy and the rademacher complexity," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2202–2211, Dec. 2014.

[20] S. V. N. Vishwanathan, N. N. Schraudolph, I. R. Kondor, and K. M. Borgwardt, "Graph kernels," *J. Mach. Learn. Res.*, vol. 11, pp. 1201–1242, Mar. 2010.

[21] G. Da San Martino, N. Navarin, and A. Sperduti, "Exploiting the odd framework to define a novel effective graph kernel," in *Proc. Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn.*, 2015, p. 219.

[22] G. Da San Martino, N. Navarin, and A. Sperduti, "A memory efficient graph kernel," in *Proc. Int. Joint Conf. Neural Netw.*, Jun. 2012, pp. 1–7.

[23] F. Costa and K. De Grave, "Fast neighborhood subgraph pairwise distance kernel," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 255–262.

[24] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt, "Weisfeiler-lehman graph kernels," *J. Mach. Learn. Res.*, vol. 12, pp. 2539–2561, Sep. 2011.

[25] G. Da San Martino, N. Navarin, and A. Sperduti, "A tree-based kernel for graphs," in *Proc. SIAM Int. Conf. Data Mining*, 2012, p. 12.

[26] T. Gärtner, P. Flach, and S. Wrobel, "On graph kernels: Hardness results and efficient alternatives," in *Proc. Comput. Learn. Theory*, 2003, pp. 24–27.

[27] V. Bolón-Canedo, M. Donini, and F. Aiolli, "Feature and kernel learning," in *Proc. 23th Eur. Symp. Artif. Neural Netw. (ESANN)*, Bruges, Belgium, Apr. 2015, p. 173.

[28] A. Maurer and M. Pontil, "Structured sparsity and generalization," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 671–690, 2012.

[29] P. Germain, A. Lacasse, F. Laviolette, M. Marchand, and J.-F. Roy, "Risk bounds for the majority vote: From a PAC-Bayesian analysis to a learning algorithm," *J. Mach. Learn. Res.*, vol. 16, no. 4, pp. 787–860, 2015.

[30] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[31] S. Mendelson, "Improving the sample complexity using global data," *IEEE Trans. Inf. Theory*, vol. 48, no. 7, pp. 1977–1991, Jul. 2002.

[32] T. Liang, A. Rakhlin, and K. Sridharan. (2015). "Learning with square loss: Localization through offset rademacher complexity." [Online]. Available: https://arxiv.org/abs/1502.06134

[33] R. Lataıa and K. Oleszkiewicz, "On the best constant in the Khintchine-Kahane inequality," *Studia Math.*, vol. 109, no. 1, pp. 101–104, 1994.

[34] S. Mendelson, "On the performance of kernel classes," *The J. Mach. Learn. Res.*, vol. 4, pp. 759–771, Oct. 2003.

[35] J. Demmel, I. Dumitriu, and A. Holtz, "Fast linear algebra is stable," *Numer. Math.*, vol. 108, no. 1, pp. 59–91, 2007.

[36] B. Beauzamy, *Introduction to Banach Spaces Their Geometry*, vol. 68. Amsterdam, The Netherlands: North Holland, 2011.

[37] H. Kashima, K. Tsuda, and A. Inokuchi, "Marginalized kernels between labeled graphs," in *Proc. Int. Conf. Mach. Learn.*, 2003, pp. 321–328.

[38] P. Mahé, N. Ueda, T. Akutsu, J. L. Perret, and J. Vert, "Extensions of marginalized graph kernels," in *Proc. Int. Conf. Mach. Learn.*, 2004, p. 70.

[39] S. V. N. Vishwanathan, K. M. Borgwardt, and N. N. Schraudolph, "Fast computation of graph kernels," in *Proc. Neural Inf. Process. Syst.*, 2006, pp. 1449–1456.

[40] K. M. Borgwardt and H. P. Kriegel, "Shortest-path kernels on graphs," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2005, p. 8.

[41] N. Shervashidze *et al.*, "Efficient graphlet kernels for large graph comparison," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2009, pp. 488–495.

[42] N. Shervashidze and K. M. Borgwardt, "Fast subtree kernels on graphs," in *Proc. Neural Inf. Process. Syst.*, 2009, pp. 1660–1668.

[43] A. K. Debnath, R. L. Lopez de Compadre, G. Debnath, A. J. Shusterman, and C. Hansch, "Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity," *J. Med. Chem.*, vol. 34, no. 2, pp. 786–797, 1991.

[44] C. Helma, T. Cramer, S. Kramer, and L. De Raedt, "Data mining and machine learning techniques for the identification of mutagenicity inducing substructures and structure activity relationships of noncongeneric compounds," *J. Chem. Inf. Comput. Sci.*, vol. 44, no. 4, pp. 1402–1411, 2004.

[45] O. S. Weislow, R. Kiser, D. L. Fine, J. Bader, R. H. Shoemaker, and M. R. Boyd, "New soluble-formazan assay for HIV-1 cytopathic effects: Application to high-flux screening of synthetic and natural products for AIDS-antiviral activity," *J. Nat. Cancer Inst.*, vol. 81, no. 8, pp. 577–586, 1989.

[46] N. Wale, I. Watson, and G. Karypis, "Comparison of descriptor spaces for chemical compound retrieval and classification," *Knowl. Inf. Syst.*, vol. 14, no. 3, pp. 347–375, 2008.

[47] G. Casella and R. L. Berger, *Statistical Inference*. Pacific Grove, CA, USA: Duxbury, 2002.

[48] D. Anguita, L. Ghelardoni, A. Ghio, L. Oneto, and S. Ridella, "The 'K' in K-fold cross validation," in *Proc. Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn.*, 2012, pp. 441–446.

Luca Oneto (M'17) was born in Rapallo, Italy, in 1986. He received the B.Sc. and M.Sc. degrees in electronic engineering from the University of Genoa, Genoa, Italy, in 2008 and 2010, respectively, and the Ph.D. degree from the School of Sciences and Technologies for Knowledge and Information Retrieval, University of Genoa, in 2014, with the thesis "Learning Based on Empirical Data."

He is currently an Assistant Professor with the University of Genoa. His current research interests include statistical learning theory, machine learning, and data mining.

Nicolò Navarin (M'17) received the B.Sc. and M.Sc. degrees in computer science from the University of Padova, Padua, Italy, in 2008 and 2010, respectively, and the Ph.D. degree in computer science from the University of Bologna, Bologna, Italy, in 2014.

In 2013, he was a Visiting Researcher with the Albert Ludwigs University of Freiburg, Freiburg im Breisgau, Germany. He currently holds a post-doctoral position in computer science with the Department of Mathematics, University of Padova. His current research interests include machine learning, kernel methods, learning for structured data, and bioinformatics.

Michele Donini received the bachelor's and master's degrees in mathematics from the University of Padova, Padua, Italy, in 2010 and 2012, respectively, and the Ph.D. degree from the School of Mathematics (computer science curriculum), University of Padova, in 2016.

He currently holds a post-doctoral position with the Istituto Italiano di Tecnologia and an Honorary Research Associate with the University College London, London, U.K. His current research interests include kernel methods, multiple kernel learning, and machine learning in general.

Sandro Ridella (M'93) received the Laurea degree in electronic engineering from the University of Genoa, Genoa, Italy, in 1966.

In the last five years, his scientific activity has been mainly involved in the field of neural networks. He is currently a Full Professor with the Dipartimento di Ingegneria Navale, Elettrica, Elettronica e delle Telecomunicazioni Department, University of Genoa, where he teaches circuits and algorithms for signal processing.

Alessandro Sperduti (SM'17) received the Ph.D. degree in computer science from the University of Pisa, Pisa, Italy.

Since 2002, he has been a Professor in computer science with the University of Padova, Padua, Italy. His current research interests include machine learning, neural networks, learning in structured domains, and data and process mining.

Dr. Sperduti has been the Chair of the Data Mining and Neural Networks Technical Committees of the IEEE Computational Intelligence Society and an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. He is currently an Action Editor for the journals *AI Communications*, *Neural Networks*, and *Theoretical Computer Science (Section C)*. He is on the Editorial Board of the *IEEE Intelligent Systems Magazine*.

Fabio Aiolli received the master's and Ph.D. degrees in computer science from the University of Pisa, Pisa, Italy.

He held a post-doctoral position with the University of Pisa, a Paid Visiting Scholar with the University of Illinois at Urbana–Champaign, Champaign, IL, USA, and a post-doctoral position with the University of Padova, Padua, Italy. He is currently an Assistant Professor with the University of Padova. His current research interests include machine learning and information retrieval.

Davide Anguita (SM'17) received the Laurea degree in electronic engineering and the Ph.D. degree in computer science and electronic engineering from the University of Genoa, Genoa, Italy, in 1989 and 1993, respectively.

He was a Research Associate with the International Computer Science Institute, Berkeley, CA, USA, where he was involved in special-purpose processors for neurocomputing. Then, he returned to the University of Genoa, where he is currently an Associate Professor of computer engineering with the Department of Informatics, Bioengineering, Robotics, and Systems Engineering. His current research interests include the theory and application of kernel methods and artificial neural networks.