Measuring the Expressivity of Graph Kernels through the Rademacher Complexity

Luca Oneto[1], Nicolò Navarin[2], Michele Donini[2],
Alessandro Sperduti[2], Fabio Aiolli[2], and Davide Anguita[1]
1 - DIBRIS - University of Genova - Via Opera Pia 13, I-16145 Genova - Italy
2 - Department of Mathematics - University of Padua
Via Trieste, 63, I-35121 Padova - Italy

Abstract. Graph kernels are widely adopted in real-world applications that involve learning on graph data. Different graph kernels have been proposed in literature, but no theoretical comparison among them is present. In this paper we provide a formal definition for the expressiveness of a graph kernel by means of the Rademacher Complexity, and analyze the differences among some state-of-the-art graph kernels. Results on real world datasets confirm some known properties of graph kernels, showing that the Rademacher Complexity is indeed a suitable measure for this analysis.

1 Introduction

Among the different machine learning techniques applicable to structured domains, kernel methods are a well established solution because they can be defined directly on structured data. This relieves the user from the definition of a vectorial representation of the data, a time consuming and task-specific operation. When it comes to deal with graphs, several instances of graph kernels have been presented in literature. A recent advance in the field are fast kernels (near-linear time) that allow for an explicit, sparse feature space representation that can be successfully applied to big graph datasets [1, 2]. Each kernel considers as features different small substructures of the original graph. Empirical comparisons among different kernels can be found in literature [3, 4] but, with few exceptions [5], 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 kernel parameters) can be a critical phase for achieving satisfying predictive performance on a specific task. In particular, different kernels induce different hypothesis spaces. The learning process (defined by the learning algorithm), based on empirical observations (examples), aims to select the hypothesis in these spaces, with the smallest generalization error, namely the error on previously unseen observations [6]. The asymptotic analysis of the learning process, via a bound on the generalisation error, has been thoroughly investigated in the past [7, 6]. However, as the number of samples is limited in practice, finite sample analysis by means of complexity measure of the hypothesis space was proposed, and represented a fundamental advance in the field [6, 8, 9, 10, 11].

In the context of graph kernels, the expressiveness of a kernel is defined as its ability to distinguish between non-isomorphic examples. In [12] it is shown

that *complete* graph kernels (kernels that map each non-isomorphic 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 non-zero features generated by different kernels are independent to each other, then it is easy to see that the more non-zero features a kernel generates, the more it is able to discriminate between examples, and so the more it is expressive. However, this is not the case with structural features, where there are strong dependency relationships among them, i.e. a feature can be non-zero only if some specific features are too. In this case, there is no easy way to assess how expressive a kernel is.

In this paper, we propose a theoretically grounded and efficiently computable notion of expressiveness which is based on the Rademacher Complexity (RC), a powerful notion of complexity of an hypothesis space. We use it to measure the expressiveness of different graph kernels, and we try to understand the main differences among them, as a first step in the process of defining a rationale behind the kernel selection that goes beyond the empirical error estimation on a subset of the available data.

2 Graph Kernels

Let us start this section with some definitions and notations. 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 $\phi : \mathcal{X} \to \mathcal{V}$, where $\mathcal{V}$ is an Hilbert space, such that $k(x, x') = \langle \phi(x), \phi(x') \rangle$. Note that the input space $\mathcal{X}$ can be any space. In particular, in this paper we will focus on the space of graphs. A graph is a tuple $G = (V_G, E_G, L_G)$, where $V_G = \{1, \ldots, n\}$ is the set of vertices, $E_G = \{(i, j) | i, j \in V_G\}$ 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 path is a sequence of vertices $v_1 \cdot \cdot v_n$ where $(v_i, v_{i+1}) \in E_G$. A cycle is a path where $v_1 = v_n$. A tree is a directed acyclic graph where one vertex has no incoming edge.

Among the different graph kernels available in literature, in this paper we focus on the ones that are considered state-of-the-art from both the predictive power and computational complexity points of view. Weisfeiler-Lehman (WL) graph kernels [4] are based on the recursive WL color refinement procedure. The principal member of this family, the Fast Subtree WL kernel, in an efficient way ($O(m)$), maps a graph in a RKHS where each feature represents a subtree-walk pattern (subtrees where vertices can appear multiple times). The value associated to 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 h (user-specified parameter) iterations. The number of non-zero features associated to a graph is at most nh. The Ordered Decomposition DAGs kernel framework (ODD) [5] considers as non-zero features in the RKHS the trees that appear as

subtrees of the input graphs. It exploits the shortest-path (up to length h) DAG decompositions starting from each node in the graph to generate DAG structures, and then extracts tree features from them. Each tree-feature is weighted as $f \cdot \lambda^{dim}$, where f is the frequency of the feature, dim its dimension (the number of vertices in the tree) and $\lambda > 0$ a weighting parameter. The time complexity of the main representative of this family, ODD_{ST_h}, is, under mild conditions, $O(n \log n)$, and the number of generated features for a graph is at most $n\rho^h$. The Neighborhood Subgraph Pairwise Distance kernel (NSPDK) [3] considers as features pairs of small-sized subgraphs (up to radius h) that appear in an input graph at a (shortest-path) distance of at most d. The number of features generated by this kernel is $\frac{hn\rho^d}{2}$, that lies between hn and $\frac{hn^2}{2}$. Note that WL is very close to a degenerate case of NSPDK, where $d = 0$.

3 Rademacher Complexity of RKHS

We consider the conventional binary classification problem [6]: based on a random observation of $X \in \mathcal{X}$, one has to estimate $y \in \mathcal{Y} = \{\pm 1\}$ by choosing a suitable hypothesis $h : \mathcal{X} \to \mathcal{Y}$, in a set of possible ones $\mathcal{H}$. A learning algorithm selects $h \in \mathcal{H}$ by exploiting a set of labeled samples $\mathcal{D}_n = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)\}$. Examples in $\mathcal{D}_n$ are sampled i.i.d. according to the distribution μ over $\mathcal{X} \times \mathcal{Y}$. The generalisation error $L(h) = \mathbb{E}_\mu \ell(h(\boldsymbol{x}), y)$ associated to an hypothesis $h \in \mathcal{H}$ is defined through a loss function $\ell(h(\boldsymbol{x}), y) : \mathcal{Y} \times \mathcal{Y} \to [0, 1]$. Since we are dealing with binary classification problems ℓ usually counts the number of misclassified samples $\ell_H(h(\boldsymbol{x}), y) = [h(\boldsymbol{x}) \neq y]$. As μ is unknown, $L(h)$ cannot be explicitly computed, thus we have to resort to its empirical estimator, namely the empirical error $\widehat{L}_n(h) = {}^1\!/_n \sum_{i=1}^n \ell\left(h\left(\boldsymbol{x}_i\right), y_i\right)$. Note that $\widehat{L}_n(h)$ is a biased estimator, since the data used for selecting the model and for computing the empirical error coincide. We estimate this bias by studying the discrepancy between the generalisation error and the empirical error. For this purpose the Global RC, defined as $\widehat{R}_n(\mathcal{H}) = \mathbb{E}_{\boldsymbol{\sigma}} \sup_{h \in \mathcal{H}} {}^2\!/_n \sum_{i=1}^n \sigma_i \ell\left(h\left(\boldsymbol{x}_i\right), y_i\right)$ where $\boldsymbol{\sigma} = [\sigma_1 | \cdots | \sigma_n]$ are n $\{\pm 1\}$–valued i.i.d. Rademacher random variables for which $\mathbb{P}(\sigma_i = +1) = \mathbb{P}(\sigma_i = -1) = {}^1\!/_2$, can be exploited [8, 9]:

Theorem 1 ([8, 9])**.** *Let us consider a space of functions $\mathcal{H}$. Then $\forall h \in \mathcal{H}$, the generalisation error can be upper-bounded with probability $(1 - 2e^{-x})$. In particular $L(h) \leq \widehat{L}_n(h) + \widehat{R}_n(\mathcal{H}) + 3\sqrt{{}^x\!/_{2n}}$.*

More recently a local version of the RC, the Local RC, has been developed [11, 10] in order to discard those functions that will never be selected during the learning phase. Unfortunately its computation results in an NP-Hard problem [8, 10, 11]. Instead, if $\mathcal{H}$ is a RKHS, computing the RC can be done efficiently. Let us consider that $h \in \mathcal{H}$ can be expressed as $h(\boldsymbol{x}) = \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x})$, where $\boldsymbol{\phi} : \mathcal{X} \to \mathbb{R}^{\mathcal{D}}$ and $\|\boldsymbol{w}\|^2 \leq W^2$; besides, let us exploit a ρ-admissible loss function: $|\ell(h_1(\boldsymbol{x}), y) - \ell(h_2(\boldsymbol{x}), y)| \leq \rho |h_1(\boldsymbol{x}) - h_2(\boldsymbol{x})|$. Consequently, let us show how to compute the RC of this space. Let us consider the following matrix $\Phi = [\boldsymbol{\phi}(\boldsymbol{x}_1) | \cdots | \boldsymbol{\phi}(\boldsymbol{x}_n)]^T$ and vector $\boldsymbol{y} = [y_1 | \cdots | y_n]^T$. Consequently we can state that:

$$\widehat{R}_n(\mathcal{H}) = \mathbb{E}_{\boldsymbol{\sigma}} \sup_{h \in \mathcal{H}} \frac{2}{n} \sum_{i=1}^n \sigma_i \ell\left(h\left(\boldsymbol{x}_i\right), y_i\right) \leq \mathbb{E}_{\boldsymbol{\sigma}} \sup_{h \in \mathcal{H}} \frac{2\rho}{n} \sum_{i=1}^n \sigma_i h\left(\boldsymbol{x}_i\right)$$

$= \mathbb{E}_{\boldsymbol{\sigma}} \sup_{\|\boldsymbol{w}\| \leq W} \frac{2\rho}{n} \sum_{i=1}^{n} \sigma_i \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i) = \frac{2\rho W}{n} \mathbb{E}_{\boldsymbol{\sigma}} \left| \sum_{i=1}^{n} \sigma_i \boldsymbol{w} \cdot \boldsymbol{\phi}(\boldsymbol{x}_i) \right|$

$= \frac{2\rho W}{n} \mathbb{E}_{\boldsymbol{\sigma}} \sqrt{\boldsymbol{\sigma}^T \Phi \Phi^T \boldsymbol{\sigma}} \leq \frac{2\rho W}{n} \sqrt{\mathbb{E}_{\boldsymbol{\sigma}} \boldsymbol{\sigma}^T \Phi \Phi^T \boldsymbol{\sigma}} = \frac{2\rho W}{n} \sqrt{\sum_{i=1}^{n} \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x}_i)}$. (1)

Note that the quantity of Eq. (1) can be easily computed from the data. If, for example, $\boldsymbol{w}_C^*$ is the solution of a Support Vector Machine [13] with $b = 0$ [14] for a particular value of its hyperparameter C over $\mathcal{D}_n$, we have basically minimised the convex upper-bound of ℓ_H in the space $\mathcal{H}$ defined by $W = \|\boldsymbol{w}_C^*\|$ and $\sigma = 1$.

Moreover, let us consider the domain Γ of vectors $\boldsymbol{\gamma} \in \mathbb{R}_+^n$ such that $\Gamma = \{\boldsymbol{\gamma} \in \mathbb{R}_+^n : \sum_{i:y_i=+1} \gamma_i = 1, \sum_{i:y_i=-1} \gamma_i = 1\}$. Given two generic points in the convex hulls of positive and negative examples in feature space, specified by a vector $\boldsymbol{\gamma} \in \Gamma$, then their squared distance can be computed by: $\| \sum_{i:y_i=+1} \gamma_i \boldsymbol{\phi}(\boldsymbol{x}_i) - \sum_{i:y_i=-1} \gamma_i \boldsymbol{\phi}(\boldsymbol{x}_i) \|^2 = \boldsymbol{\gamma}^T Y \Phi \Phi^T Y \boldsymbol{\gamma}$ where $Y = \mathrm{diag}(\boldsymbol{y})$. It follows that the minimum distance between the convex hulls of positive and negative examples can be lower bounded as follows [15]:

$$\min_{\boldsymbol{\gamma} \in \Gamma} \boldsymbol{\gamma}^T Y \Phi \Phi^T Y \boldsymbol{\gamma} \geq \frac{2}{n} \min\{\lambda_1, \cdots, \lambda_n\} = {}^2/_n \lambda_{\min}, \tag{2}$$

where $\lambda_1, \cdots, \lambda_n$ are the eigenvalues of the matrix $\Phi \Phi^T$. Similarly, we can give an upper bound to the maximum distance of training examples which depends on the maximal eigenvalue, that is [15]:

$$\max_{\boldsymbol{\gamma} \in \Gamma} \boldsymbol{\gamma}^T Y \Phi \Phi^T Y \boldsymbol{\gamma} \leq \max\{\lambda_1, \cdots, \lambda_n\} = \lambda_{\max}. \tag{3}$$

In order to compare two kernels defined by $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ through the RC of Eq. (1), we will normalize it in this way: without loss of generality [10] we set $W = 1$, then we divide by the difference between the maximum distance of training examples and the minimum distance between the convex hulls of positive and negative examples, finally we remove the constant factors ρ and n. Consequently the Normalised RC of a kernel defined by $\boldsymbol{\phi}$ can be introduced as:

$$\widehat{N}(\boldsymbol{\phi}) = \sum_{i=1}^{n} \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x}_i) / (\lambda_{\max} - {}^2/_n \lambda_{\min}). \tag{4}$$

This quantity is able to measure how much expressive is a kernel defined by $\boldsymbol{\phi}_1$ with respect to another one defined by $\boldsymbol{\phi}_2$, since we have scaled the eigenvalues with respect to the geometry of the space and removed the constant factors. Moreover $\widehat{N}(\boldsymbol{\phi})$ is theoretically grounded by its connection with the generalisation error of a linear separator in the space defined by $\boldsymbol{\phi}$.

Dataset/kernel	AIDS		CAS		CPDB		NCI1	
	max	min	max	min	max	min	max	min
NSPDK	7.9	1.5	8.3	1.6	9.6	1.8	6.8	1.3
WL	1.7	1.1	2.0	1.2	2.7	1.4	1.5	1.1
ODD_{ST}	58.9	1	5.6	1	4.6	1.1	30.1	1

Table 1: Maximum and minimum $\widehat{N}$ achieved by different kernels.

4 Experimental results

In this section, we analyze the Normalised RC defined in Eq. (4) of three state-of-the-art graph kernels, presented in Section 2, on the AIDS, CAS, CPDB and NCI1 datasets (see [5] for a detailed description). The datasets represent

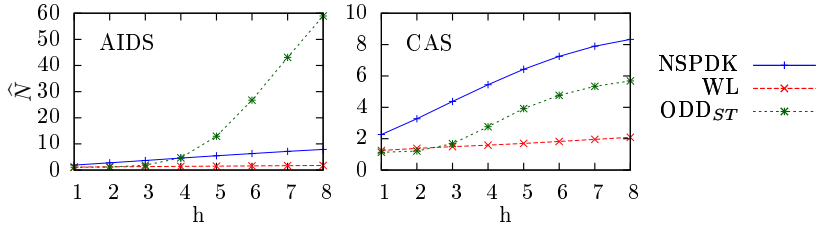

Fig. 1: $\widehat{N}$ as a function of the h parameter for kernels and datasets.

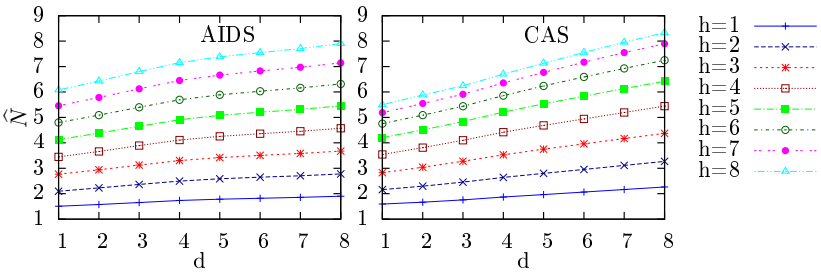

Fig. 2: $\widehat{N}$ of the $NSPDK$ kernel for different values of h and d, and datasets.

chemical compounds as graphs, where each node corresponds to an atom, and edges encode the bonds between them.

Table 1 reports the maximum and minimum value of Eq. (4) for each kernel/dataset combination. The WL kernel is the one with the smaller associated feature space. Its maximum $\widehat{N}$ values are the smallest ones, indicating that it is the least expressive among the considered kernels. In the same Table 1, it is possible to see that for two out of four datasets (CAS and CPDB), the NSPDK kernel achieves the highest $\widehat{N}$ values, while in the other two its expressiveness remains consistent. On the other hand, ODD$_{ST}$ is able to achieve very high $\widehat{N}$ values on AIDS and NCI1 datasets.

Let us now try to understand how the kernel parameters influence the $\widehat{N}$ values of the resulting kernels. Figure 1 shows the highest $\widehat{N}$ values that it is possible to achieve fixing the h parameter of the considered kernels, on two datasets (for the other two datasets the situation is similar, thus the plots are omitted for lack of space). WL shows the slowest-growing complexity. NSPDK grows consistently in both the datasets. On AIDS, the complexity of ODD$_{ST}$ kernel grows exponentially in h. Figure 2 shows in more detail the behavior of NSPDK kernel as a function of the two parameters h and d. The growth in complexity is linear with respect to both the parameters. Figure 3 refers to ODD$_{ST}$ kernel. In this case, the kernel complexity grows non-linearly with the λ parameter. It is interesting to point out that different kernels may reach the same complexity with different h parameters. In that case, a user (or an algorithm for parameter or kernel selection) aims to pick the kernel/parameter

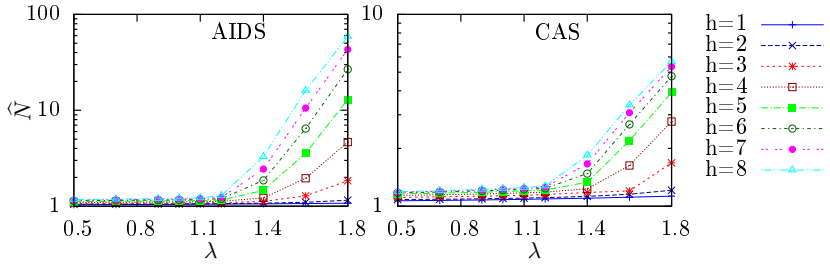

Fig. 3: $\widehat{N}$ of the ODD_{ST} kernel for different values of h and λ, and datasets.

combination that is more efficient to compute.

5 Conclusions and future works

We presented in this paper a principled way to measure the expressiveness of graph kernels. The measure is based on the (Global) Rademacher complexity. This measure can, in future, be applied to perform kernel/parameters selection. Moreover, more complex measures based on the *local* Rademacher complexity will be explored.

References

[1] G. Da San Martino, N. Navarin, and A. Sperduti. Exploiting the ODD framework to define a novel effective graph kernel. In *ESANN*, 2015.
[2] G. Da San Martino, N. Navarin, and A. Sperduti. A memory efficient graph kernel. In *International Joint Conference on Neural Networks*, 2012.
[3] F. Costa and K. De Grave. Fast neighborhood subgraph pairwise distance kernel. In *International Conference on Machine Learning*, 2010.
[4] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, and K. M. Borgwardt. Weisfeiler-Lehman Graph Kernels. *Journal of Machine Learning Research*, 2011.
[5] G. Da San Martino, N. Navarin, and A. Sperduti. A Tree-Based Kernel for Graphs. In *Proceedings of the Twelfth SIAM International Conference on Data Mining*, 2012.
[6] V. N. Vapnik. *Statistical learning theory*. Wiley–Interscience, 1998.
[7] M. Talagrand. The glivenko-cantelli problem. *Annals of probability*, 15(3):837–870, 1987.
[8] P. L. Bartlett and S. Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *The Journal of Machine Learning Research*, 3:463–482, 2003.
[9] L. Oneto, A. Ghio, S. Ridella, and D. Anguita. Global rademacher complexity bounds: From slow to fast convergence rates. *Neural Processing Letters*, pages (in–press), 2015.
[10] P. L. Bartlett, O. Bousquet, and S. Mendelson. Local rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537, 2005.
[11] L. Oneto, A. Ghio, S. Ridella, and D. Anguita. Local rademacher complexity: Sharper risk bounds with and without unlabeled samples. *Neural Networks*, 65:115–125, 2015.
[12] T. Gartner, P. Flach, S. Wrobel, and T. Gärtner. On Graph Kernels: Hardness Results and Efficient Alternatives. In *Computational Learning Theory*, 2003.
[13] L. Oneto, S. Ridella, and D. Anguita. Tikhonov, ivanov and morozov regularization for support vector machine learning. *Machine Learning*, In Press.
[14] T. Poggio, S. Mukherjee, R. Rifkin, A. Rakhlin, and A. Verri. b. In *Uncertainty in Geometric Computations*, 2002.
[15] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.