ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

On the Definition of Complex Structured Feature Spaces

Nicolò Navarin[1], Dinh V. Tran[2] and Alessandro Sperduti[1] *

1- University of Padova - Department of Mathematics
Via Trieste 63, 35121 Padova, Italy

2- University of Freiburg - Department of Computer Science
Georges-Köhler-Allee 106, 79110 Freiburg, Germany

Abstract. In this paper, we propose a graph kernel whose feature space is defined by combining pairs of features of an existing *base* graph kernel. Furthermore, we propose a variation where the feature space is adaptive with respect to the learning task at hand, allowing to learn a representation suited to it. Experimental results on six real-world graph datasets from different domains show that the proposed kernels are able to get a consistent performance improvement over the considered base kernel, and over previously defined feature combination methods in literature.

1 Introduction

When dealing with machine learning on structured data, kernel methods are one of the most popular approaches. Unlike the majority of machine learning techniques, their application to any type of data is painless as long as a kernel function for such data is defined. Dealing with data represented as graphs is challenging, since even the basic operations can be computationally expensive (e.g. the graph isomorphism problem). Although several attempts have recently been pursued towards the definition of deep neural networks or deep probabilistic models for graph data (see [1, 2] and references therein), kernel methods for graphs remain among the best performing approaches to use. The approach that most graph kernels follow is to compare two graphs with respect to specific types of local substructures they share. For efficiency reasons, those kernels tend to use *local* compact features, e.g. small connected subgraphs. Recently, some methods have been proposed to increase the expressiveness of *local* graph kernels, by combining pairs of local features at the cost of an increased computational complexity [3, 4]. These approaches consider only *homogeneous* features (i.e. substructures that share some characteristics, e.g. the maximum shortest-path distance between any two vertices) as members of the pairs, to reduce the computational complexity of the kernel with respect to considering all the possible pairs, but the final kernel expressiveness may be hurt by this choice. In this paper, we define a kernel that, while allowing pairs of non-homogeneous features from a *base* kernel, enforces constraints on admissible pairs, so to keep under control the computational complexity of the kernel. In a nutshell, the idea is to

*This project was funded, in part, by DFG project, BA 2168/3-3, Germany.

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

associate to each node of a graph a subset of features from a base kernel. Then pairs of nodes in a graph are considered and those that satisfy a given set of topological constraints (on the shortest-path distance between vertices) are used to generate new features obtained by the Cartesian product of the subsets of features associated to the two nodes. The weight associated to each new pair of features is obtained by multiplying the weights of the single constituent features. In addition to that, we introduce a mechanism to define adaptive kernels, i.e. kernels defined on the basis of a specific set of training graphs. Experimental results on several datasets show the advantage of the proposed approach.

2 Background

We start providing some definitions and notation conventions. We consider a graph as a triplet $G = (V^G, E^G, \lambda^G(\cdot))$, in which V^G is the set of nodes (or vertices), $E^G \subseteq V^G \times V^G$ is the edge set, and $\lambda^G : V^G \to \mathcal{L}$ is the node labeling function that assigns a discrete label in $\mathcal{L}$ to each node in the graph. When clear from the context, we omit the reference to G for ease of notation. Given a vertex v, its d-neighborhood is defined as the set of nodes with *shortest path* distance exactly equal to d from v, i.e. $\mathcal{N}_v^d = \{u | sp(v,u) = d\}$ where $sp(u,v)$ is the function returning the length of the shortest path between two vertices u and v in a graph. A kernel on $\mathcal{X}$, the input space, is a symmetric positive semi-definite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ computing a score (*similarity*) between pairs of instances. Kernel functions compute the dot product between two objects in a *Reproducing Kernel Hilbert Space* (RKHS), i.e.: $k(x,y) = \langle \phi(x), \phi(y) \rangle$ where $\phi : \mathcal{X} \to \mathcal{H}$ is a function mapping instances from $\mathcal{X}$ to the RKHS (or feature space) $\mathcal{H}$. Different kernels define different feature spaces.

2.1 Graph Kernels

In this section, we describe several graph kernels proposed in literature, in terms of the graph substructures that they considers as features.
The Marginalized Graph Kernel (MGK) [5] considers as features all the possible random walks in a graph, with a complexity of $O(|V|^3)$. The Shortest Path (SP) Kernel associates a feature to each pair of node labels at a certain (unbounded) shortest-path distance [6]. The complexity of the kernel is $O(|V|^4)$. The graphlet kernel [7] considers all subgraphs up to a fixed number of nodes k, with complexity $O(|V|^k)$. An efficient kernel based on tree patterns is the Weisfeiler-Lehman (WL) Subtree Kernel that counts the number of identical subtree-walk patterns obtained by breadth-first visits [3]. The complexity of the kernel is $O(|E|h)$, where h is the a-priori selected number of WL iterations, that corresponds to the maximum depth of the considered patterns. A family of graph kernels based on visits has been proposed in [8]. One of the most efficient and effective instances of the framework (ODD_{ST}) considers as features subtrees of the input graph. The framework has been extended to consider graphs with continuous attributes [9], that we leave as a future work. More recent frameworks for graph

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

kernels addressing efficiency or efficacy have been proposed in [10, 11]. All the kernels presented in this section, consider *local* substructures as features.

3 Related Works

One possible way to define more expressive feature spaces for graph kernels, is to combine multiple *local* features, so to obtain more *global* ones. [12] proposes to make the features of the random walk kernels more discriminative (thus to increase the complexity of the features) substituting the original labels of input graphs with the *Morgan Index*. [3] follows a similar idea, proposing to adopt the Weisfeiler-Lehman (WL) kernel as a relabeling procedure, and the shortest path kernel on the relabeled graphs. The features of the resulting WL-SP kernel are pairs of WL labels (from the same WL iteration), that will match only if the shortest paths that connect the associated nodes in the two graphs have the same length. The WL-SP kernel shows a very high computational complexity (mostly due to the SP kernel), with more than a year CPU time required for the kernel matrix computation on some common benchmark datasets. NSPDK kernel presented in [4], has been the first step towards the definition of efficient graph kernels with complex structural features. The idea is to construct a feature space where each feature is composed by a pair of subgraphs of the original graph, at a maximum shortest-path distance d. However, for computational reasons, the pairs of base features are constrained to be homogeneous, i.e. to share the same *radius* (see original paper for more details).

4 Feature Space Composition Kernels

In a nutshell, we aim to design kernels whose features are pairs of a base kernel's features. Let us start defining the base kernel that we consider.
We consider the Weisfeiler-Lehman (WL) subtree kernel. Let us define j as $0 \leq j \leq h$, and $\phi^{\mathrm{WL}_j}(v)$ as the vectorial representation of the set of features (actually for WL it is just one feature) corresponding to tree patterns rooted in v generated at iteration j. Let us define the WL graph *node* kernel as $k_j^{\mathrm{WL}}(u,v) = \langle \phi^{\mathrm{WL}_j}(v), \phi^{\mathrm{WL}_j}(u) \rangle$. We will adopt these definitions throughout the paper. It is possible to show that both WL-SP and NSPDK can be defined as combinations of the k_j^{WL} node kernel. Both kernels consider just pairs of features generated by the same value of j. In our proposal, we aim to alleviate this requirement.

Let us define as base kernel $k^{\mathrm{WL}}(u,v) = \sum_{j=0}^{h} k_j^{\mathrm{WL}}(u,v)$ and its corresponding feature space $\phi^{\mathrm{WL}}(v) = \sum_{j=0}^{h} \phi_j^{\mathrm{WL}}(v)$ for $v \in V^G$, i.e. we consider all the features generated by WL at different iterations up to h for the node v. We omit the dependency from h for ease of notation. The 2-dimensional feature space of our proposed Feature Space Composition kernel can be defined as:

$$\phi^{\mathrm{FSC}}(G) \quad = \quad \sum_{v \in V^G} (\phi^{\mathrm{WL}}(v) \sum_{i=0}^{D} \sum_{u \in \mathcal{N}_v^i} \phi^{\mathrm{WL}}(u)^{\intercal}). \tag{1}$$

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

In words, the kernel computes the similarity between two graphs by counting the common pairwise features associated to nodes at the same distance (up to a maximum distance D) in the two input graphs. We denote this kernel as $\mathrm{FSC}_{\mathrm{WL}}(G_1, G_2) = \langle \phi^{\mathrm{FSC}}(G_1), \phi^{\mathrm{FSC}}(G_2) \rangle$.
It is possible to show that $\mathrm{FSC}_{\mathrm{WL}}$ is strictly more expressive than NSPDK, fixed the same value for the hyper-parameters, i.e. $D = d$. Consequently, it is also more expressive than WL-SP given D is big enough (i.e. the maximum graph *diameter* in the dataset).

Let us now slightly modify the kernel in eq.(1), considering the relevance of each feature with respect to the target function, making the derived kernel adaptive with respect to the specific learning task. More in detail, let us consider a binary classification task $\mathcal{T}$ and w^{WL} be the weight vector associated to a classifier obtained by training an SVM using the vanilla WL kernel on the training examples associated to $\mathcal{T}$. Since an explicit representation of the feature space is used, there will be a one to one correspondence between a feature $f \in \phi^{\mathrm{WL}}$ and the weight w_f in w^{WL}. This correspondence can then be exploited to select a pair of features if and only if they support the same class, e.g.

$$\phi^{\mathrm{FSC}^{\mathrm{SVM}}} = A \circ \sum_{v \in V^G} (\phi^{\mathrm{WL}}(v) \sum_{i=0}^{D} \sum_{u \in \mathcal{N}_v^i} \phi^{\mathrm{WL}}(u)^{\intercal}), \tag{2}$$

where $\circ$ indicates the Hadamard (element-wise) product, and A is defined as

$$A = \frac{1}{2}(sign(w^{\mathrm{WL}}) sign(w^{\mathrm{WL}})^{\intercal} + \mathbb{1}),$$

where $sign(x)$ returns the sign of x and is applied component-wise, $\mathbb{1}$ is the matrix of all ones of suitable size. The kernel derived in this way can be made computationally more efficient since it deals with less features. We denote this kernel as $\mathrm{FSC}_{\mathrm{WL}}^{\mathrm{SVM}}$.
The computational complexity of the kernel in eq. (1) is $O(|V|^3)$ (details omitted for lack of space). Note that this complexity is lower w.r.t WL-SP, that has $O(h|V|^4)$ complexity due to the SP kernel computation. Finally, for the SVM variant in eq. (2), it requires to train an SVM on the training data using the base kernel, that is faster by definition.

5 Empirical Evaluation

We consider for our comparisons six real-world graph datasets representing chemo and bioinformatics data: CAS, CPDB, AIDS, NCI1, NCI109 and GDD; see [8] for a detailed description. As classifier, we adopt an SVM. We consider as *base* kernels: the WL kernel [3], $\mathrm{ODD}_{\mathrm{ST}}$ [8], the p-random walk kernel, that is a kernel that compares random walks up to length p in two graphs (special case of [5]), and the *graphlet* kernel [7] (restricted at k=3 according to [3] for efficiency reasons). As for feature combination methods, we consider two kernels in literature, i.e. NSPDK [4] kernel and the Weisfeiler-Lehman Shortest-Paths (WL-SP)

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

Table 1: Average accuracy ± standard deviation on six datasets of: five *baseline* kernels, two kernels based on *feature composition* (NSPDK and WL-SP), and the two *proposed kernels* $\mathrm{FSC}_{\mathrm{WL}}$ and $\mathrm{FSC}_{\mathrm{WL}}^{\mathrm{SVM}}$. *: kernels whose performance difference with respect to the top-ranked kernel is statistically significant. **: reported from [3].

Kernel	CAS	CPDB	AIDS	NCI1	GDD	NCI109
p-random walk	70.16* ±0.20	64.14* ±1.35	73.55* ±0.49	- ±−	- ±−	- ±−
3-Graphlet	71.10* ±0.48	67.36* ±0.96	73.98* ±0.65	69.68* ±0.52	74.92* ±1.40	68.07* ±0.31
$\mathrm{ODD}_{\mathrm{ST}}$	83.34 ±0.31	76.44* ±0.62	81.51* ±0.74	82.10* ±0.42	75.27* ±0.68	81.91* ±0.42
WL	83.32* ±0.37	76.36* ±1.48	82.02* ±0.4	84.41* ±0.49	75.46* ±0.98	85.02* ±0.44
NSPDK	83.60 ±0.34	76.99* ±1.15	82.71* ±0.66	83.45* ±0.43	74.09* ±0.91	84.17* ±0.33
WL-SP**	-	-	-	84.55	79.43	83.53
$\mathrm{FSC}_{\mathrm{WL}}$	83.55 ±0.20	79.25 ±0.75	83.34* ±0.25	85.08* ±0.25	79.81* ±0.40	84.90* ±0.42
$\mathrm{FSC}_{\mathrm{WL}}^{\mathrm{SVM}}$	**83.63** ±0.36	**79.41** ±0.67	**83.55** ±0.28	**86.18** ±0.23	**80.77** ±0.42	**86.17** ±0.23

[3]. Finally, we consider our two proposed kernels $\mathrm{FSC}_{\mathrm{WL}}$ and $\mathrm{FSC}_{\mathrm{WL}}^{\mathrm{SVM}}$. We report the average accuracy results, together with the standard deviations, of 10 different repetitions of a *nested* 10-fold cross-validation (CV) procedure. The hyper-parameters of the various methods were tuned in the *inner* CVs in the following ranges: the iteration parameter, h, of WL in $\{1, 2, \ldots, 6\}$, the distances D of FSC and d of NSPDK in $\{1, 2, \ldots, 6\}$, the SVM C in $\{10^{-4}, \ldots, 10^{4}\}$. A 10×10 CV t-test with confidence level 95% (and 10 degrees of freedom) has been executed between each pair of kernels on all datasets [13] (with the exception of WL-SP kernel, which results are reported from [3]).

5.1 Results and Discussion

Table 1 reports the results of our evaluation. The kernels that consider pairs of features (NSPDK and WL-SP) perform better in almost all cases compared to the base WL kernel, with the exceptions of NSPDK on NCI1 and GDD, and both NSPDK and WL-SP on NCI109. Considering the new proposals, $\mathrm{FSC}_{\mathrm{WL}}$ performs better than both NSPDK and WL-SP in almost all cases. This is consistent with our feature space analysis reported in Section 4. It performs also almost always better than the base WL kernel, with the only exception of the NCI109 dataset, where the performances of $\mathrm{FSC}_{\mathrm{WL}}$ and WL are comparable. Let us now consider the $\mathrm{FSC}_{\mathrm{WL}}^{\mathrm{SVM}}$. This instance actually learns a representation that is specific for the considered task. Recent trends in machine learning show that this characteristic tends to be beneficial for the predictive

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

performance. In fact, $\text{FSC}_{\text{WL}}^{\text{SVM}}$ is consistently the best performing method in all the considered datasets. The reported improvements are in almost all cases statistically significant compared to other kernels (the only exception being the CAS dataset, where its improvements over ODD_{ST} and NSPDK are marginal).

6 Conclusions

In this paper, we propose a graph kernel whose feature space is constructed by pairs of base kernel features satisfying predefined sets of constraints. Moreover, we define a second kernel that is adaptive with respect to the target task. Experiments on various datasets prove that the new kernels show better performance compared with other feature combination methods in literature. Future works include: *i)* definition of a general framework for constructing complex kernels starting from existing ones; *ii)* the extension of the feature combination idea to graphs with continuous attributes.

References

[1] Dinh V. Tran, Nicolò Navarin, and Alessandro Sperduti. On Filter Size in Graph Convolutional Networks. In *IEEE Symposium on Deep Learning, SSCI*, 2018.

[2] Bacciu Davide, Errica Federico, and Micheli Alessio. Contextual graph markov model: A deep and generative approach to graph processing. In *ICML*, 2018.

[3] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M. Borgwardt. Weisfeiler-Lehman Graph Kernels. *JMLR*, 12:2539–2561, 2011.

[4] Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In *ICML*, pages 255–262. Omnipress, 2010.

[5] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Marginalized kernels between labeled graphs. In *ICML*, pages 321–328. AAAI Press, 2003.

[6] K.M. Borgwardt and Hans-Peter Kriegel. Shortest-Path Kernels on Graphs. In *ICDM*, pages 74–81, Los Alamitos, CA, USA, 2005. IEEE.

[7] Nino Shervashidze, Kurt Mehlhorn, Tobias H Petri, S V N Vishwanathan, Karsten M Borgwardt, Tobias H Petri, Kurt Mehlhorn, and Karsten M Borgwardt. Efficient graphlet kernels for large graph comparison. In *AISTATS*, volume 5, pages 488–495, Clearwater Beach, Florida, USA, 2009. CSAIL.

[8] Giovanni Da San Martino, Nicolò Navarin, and Alessandro Sperduti. Ordered Decompositional DAG Kernels Enhancements. *Neurocomputing*, 192:92–103, 2016.

[9] Giovanni Da San Martino, Nicolò Navarin, and Alessandro Sperduti. Tree-based kernel for graphs with continuous attributes. *IEEE TNNLS*, 29(7):3270–3276, 2018.

[10] Marion Neumann, Roman Garnett, Christian Bauckhage, and Kristian Kersting. Propagation kernels: efficient graph kernels from propagated information. *Machine Learning*, jul 2015.

[11] Francesco Orsini, Paolo Frasconi, and Luc De Raedt. Graph invariant kernels. *IJCAI*, pages 3756–3762, 2015.

[12] Pierre Mahé, Nobuhisa Ueda, Tatsuya Akutsu, Jean-Luc Perret, and Jean-Philippe Vert. Extensions of marginalized graph kernels. In *ICML*, page 70, 2004.

[13] Nathalie Japkowicz and Mohak Shah. *Evaluating learning algorithms: A classification perspective*, volume 9780521196000. Cambridge University Press, 2011.