On the Necessity of Graph Kernel Baselines

Till Schulz[1] and Pascal Welke[1]

Dept. of Computer Science, University of Bonn, Germany

Abstract. Naturally, graph structured data is not easy to learn from. As opposed to itemsets which can be represented by a table of fixed length, there is no obvious representation language for graphs which allows for an easy similarity measure in order to perform e.g. classification tasks on sets of graphs. There have been introduced numerous graph kernels which tackle the problem of defining a suitable similarity between graphs by incorporating structural information. In this article, however, we revert to the very simplistic approach which is to regard a graph as a (multi-) itemset made up of node and edge labels. We consider our method as a baseline and compare it to several established graph kernels on a wide range of benchmark datasets. Our observations suggest that for the overwhelming number of available datasets, actually utilizing the graphs' structure in graph kernels does not significantly improve the classification accuracy.

1 Introduction

Graph kernels have been a well established and significant research topic in learning from graph structured data for well over a decade. Starting with the random walk kernel [6], a wide variety of specialized kernels have been proposed. Some of the most significant approaches include kernels operating on subtrees such as the Weisfeiler Lehman Subtree kernel [15], as well as approaches measuring similarity based on common shortest paths such as [2]. Furthermore, (probabilistic) frequent subgraphs can be used to define graph kernels [5, 18].

Despite the richness of methods for the computation of similarities between graphs, the public availability of benchmark datasets and thus the possibility to evaluate these methods is rather sparse. As a consequence, an overwhelming amount of graph kernel publications evaluate their approaches on only very few benchmark graph datasets. To a great extend these benchmark datasets are limited to annotated molecular graphs of almost tree-like structure. While the graph community has somewhat unofficially agreed upon utilizing this fixed set of graph data, their suitability as benchmark tests is rarely questioned. Only recently larger corpora of benchmark datasets have become available [9] on which researchers started to evaluate graph kernels [4, 10].

In this paper, we (re-)introduce a simple kernel for labeled graphs which completely disregards all structure but merely considers a graph as a multi set of node and edge labels. This obviously challenges our intuition that the graph structure should be essential to obtain expressive similarity measures. It turns

out, however, that each tested graph kernel performs significantly better than our simple baseline method on only a small fraction of graph classification tasks in a large corpus hosted at TU Dortmund [9]. Often, they even perform significantly worse. This raises concerns on the suitability of currently available benchmark graph datasets for the evaluation of graph kernels.

2 The No-Graph Baseline Kernel

Similarity based learning requires a similarity measure between pairs of objects under consideration. The central task of graph kernels is to define a semantically relevant similarity measure that behaves like a scalar product in some Hilbert space in order to apply kernelized learning methods such as SVMs to graph structured data. All graph kernel papers that we are aware of (e.g., [2, 6, 7, 8, 10, 13, 15, 18, 19]) either explicitly or implicitly assume that the topological structure of the graphs should be taken into account when designing a kernel for graph-structured data in order to succeed at a given learning task.

While graph kernels try to consider properties of the graphs' topologies, we pursue the exact opposite approach. Our *No-Graph Kernel* (*NoG*) is specifically designed to disregard all topological structure and perceives a graph simply as a multiset of node and edge labels. We explicitly construct feature vectors for graphs by counting the number of times a label appears in the graph and apply the radial basis function (rbf) kernel to these feature vectors. Unlabeled graphs are considered as graphs with a single node label and a single edge label which yields a two-dimensional feature vector containing the number of nodes and edges only.

More formally, we consider undirected graphs of the form $G = (V, E, \ell_V, \ell_E)$ where V is a set of vertices, $E \subseteq \{\{u, v\} : u, v \in V, u \neq v\}$ is a set of edges and $\ell_V : V \to \Sigma_V$, $\ell_E : E \to \Sigma_E$ are labeling functions mapping vertices (resp. edges) to elements of non-empty, finite alphabets. We assume $\Sigma_V \cap \Sigma_E = \emptyset$. We then define the feature vector of G as

$$\varphi(G) := (\psi_l(G))_{l \in \Sigma_V \cup \Sigma_E}$$

with

$$\psi_l(G) = \begin{cases} |\{v \in V,\ \ell_V(v) = l\}| & \text{if } l \in \Sigma_V \\ |\{e \in E,\ \ell_E(e) = l\}| & \text{if } l \in \Sigma_E \end{cases}$$

Array of all frequencies of both the edge and vertex labels (assumed to be disjoint)

Let G, G' be two graphs and let k be the rbf kernel (on real valued vectors). Then our baseline kernel NoG is defined as

$$k_{\mathrm{NoG}}(G, G') = k(\varphi(G), \varphi(G')) \ .$$

3 Related Work

Our work confirms recent results in [4, 10, 12] that were obtained with different kernels on *sub-corpora* of the benchmark datasets considered in this paper. [4]

introduces the LDP kernel for unlabeled graphs that considers for each vertex the degree information of itself and its 1-neighborhood as a five-dimensional vector. The local degree profile (LDP) kernel is then defined as the concatenation of five histograms over all vertices in the graph. The LDP kernel is subject to five parameters which control the generation of histograms and which need to be tuned in the learning process. [10] considers three different baseline kernels which count the number of labeled subgraphs of sizes one, two, and three. This corresponds to graphlet kernels [16] using parameters $k = 1, 2, 3$, respectively. [12] considers vertex labels (i.e., single vertex graphlets) and alternatively the number of vertices only, on six benchmark datasets.

Our work differs from the above studies in a few ways: In contrast to [4], our NoG kernel uses edge and vertex label information but no topological information and does not have any parameters that need to be tuned for the feature extraction step. Compared to [12], our NoG kernel is more complex by considering edge labels as well. However, we do not consider graphlets consisting of two vertices (i.e., labeled vertex – edge – vertex triples) as done in [10] but merely utilize label information. Hence, we do not make use of any structural information of graphs whatsoever. Regarding the empirical evaluation, we consider more datasets than the above studies and employ a statistical analysis to measure whether differences in average accuracies are actually significant.

4 Experimental Evaluation

In this section, we evaluate our method by comparing it to several known graph kernels on a large subset of benchmark classification datasets from [9].

Datasets An overwhelming amount of datasets in [9] belong to the class of molecular graphs such as MUTAG, PTC, NCI and AIDS. Each graph represents a molecule made up of atoms (vertices) and covalent bonds (edges). Graphs are often annotated against whether or not they have a specific (bio-) chemical property. Another class of benchmark graph datasets deals with the prediction task of enzyme memberships such as ENZYMES and PROTEIN. A protein is modeled as a graph encapsulating information like structure, sequence and various chemical properties [3]. The datasets IMDB and REDDIT belong to a class of graphs which were extracted from online networks. IMDB consists of collaboration networks between actors/actresses each annotated against movie genres, whereas graphs in REDDIT represent user interactions in discussion forums with graphs being annotated by the type of forum [19]. [9] includes several attributed graphs which were generated from images. E.g., COIL-RAG consists of region adjacency graphs based on small segmented images of objects [14]. Graphs in the Letter datasets represent capital letters of the roman alphabet with edges being lines and nodes being their endpoints (with 2-dimensional coordinate attribute vectors) [14].

Kernels

Experimental Setup We compare a variety of well-established graph kernels to our baseline kernel (NoG). We used the implementations of the Weisfeiler

Lehman (wl) kernel [15], the graphlet sampling (gs) kernel [16], the shortest path (sp) [6] kernel, and the random walk (rw) [3] kernel as provided by the GraKel library [17]. Furthermore, we considered the (boosted) probabilistic frequent subtree kernels (psf, bpsf) [18] and the cyclic pattern kernel (cp) [7] using the authors' implementations and a frequent subgraph kernel (fsg) based on the FSG implementation [11].

SVM

Using a 10-fold cross-validation, the predictive performance was measured in terms of accuracy obtained by support vector machines (SVM). In each fold, we used a grid search to identify the optimal parameters for each kernel (such as SVM and further individual kernel parameters) on the test data using a 3-fold cross validation. The SVM parameter C was selected from the set $\{2^i : i \in \{-11, -9, -7, -5, -3, -1, 0, 1, 3, 5, 7, 9, 11\}\}$. For the Weisfeiler Lehman kernel, the grid search was performed over values k (the number of iterations) ranging from 1 to 8. The shortest path kernel was set up such that it regarded graph labels if available. In case of the random walk kernel, the grid search considered weight parameters $\lambda \in \{10^i : i \in \{-2, -3, -4\}\}$. For the graphlet sampling kernel, we set $\epsilon = 0.1$, $\delta = 0.1$ and $k \in \{3, 4, 5\}$ where k is the graphlet size (as suggested by [16]). The (boosted) probabilistic frequent subtree kernel, as well as the frequent subgraph kernel required an explicit pattern mining process in advance. We enumerated patterns up to size 10 that passed the frequency threshold of $\theta = 10\%$. For psf and bpsf we used a sampling parameter $k = 10$ (as suggested by [18]). Using the feature vectors defined by such patterns, we employed a linear kernel for the classification process. As for the cyclic pattern kernel, we utilized the simple variant operating with a linear kernel. Finally, for our baseline (NoG) we tuned the radial basis function parameter $\gamma \in \{2^i : i \in \{-1, -3, -5, -7, -9, -11\}\}$.

Statistical significance

To identify significant differences in the accuracies achieved by the individual kernels compared to the NoG kernel, we used the paired variant of Student's t-test on each dataset, resulting in 261 statistical tests. To make up for the large number of tests, we used the Benjamini and Hochbergs method [1] with a significance level of 0.05.

Observations Figure 1 on page 5 shows the prediction performances of established graph kernels compared to our baseline (NoG) kernel. We provide accuracy values for NoG; for the other kernels, we only provide accuracies if they significantly differ according to our statistical test. The coloring of the cells illustrates the degree of difference if significant. Otherwise the cell is left white. Blue cells indicate a favorable result of the comparing method, whereas red points out that the baseline performed significantly better.

In several cases we were not able to obtain results due to memory errors or time constraints. These cases are depicted in gray. In particular, we tried to compute the sets of frequent subgraphs for the fsg kernel using the FSG software [11]. However, the tool often did not terminate within 12 hours on our machine (Intel i7-4770, 16GB RAM) or alternatively produced a pattern set and corresponding feature vectors that were too large to fit into main memory in the downstream learning task. cp also failed in many cases, however, much

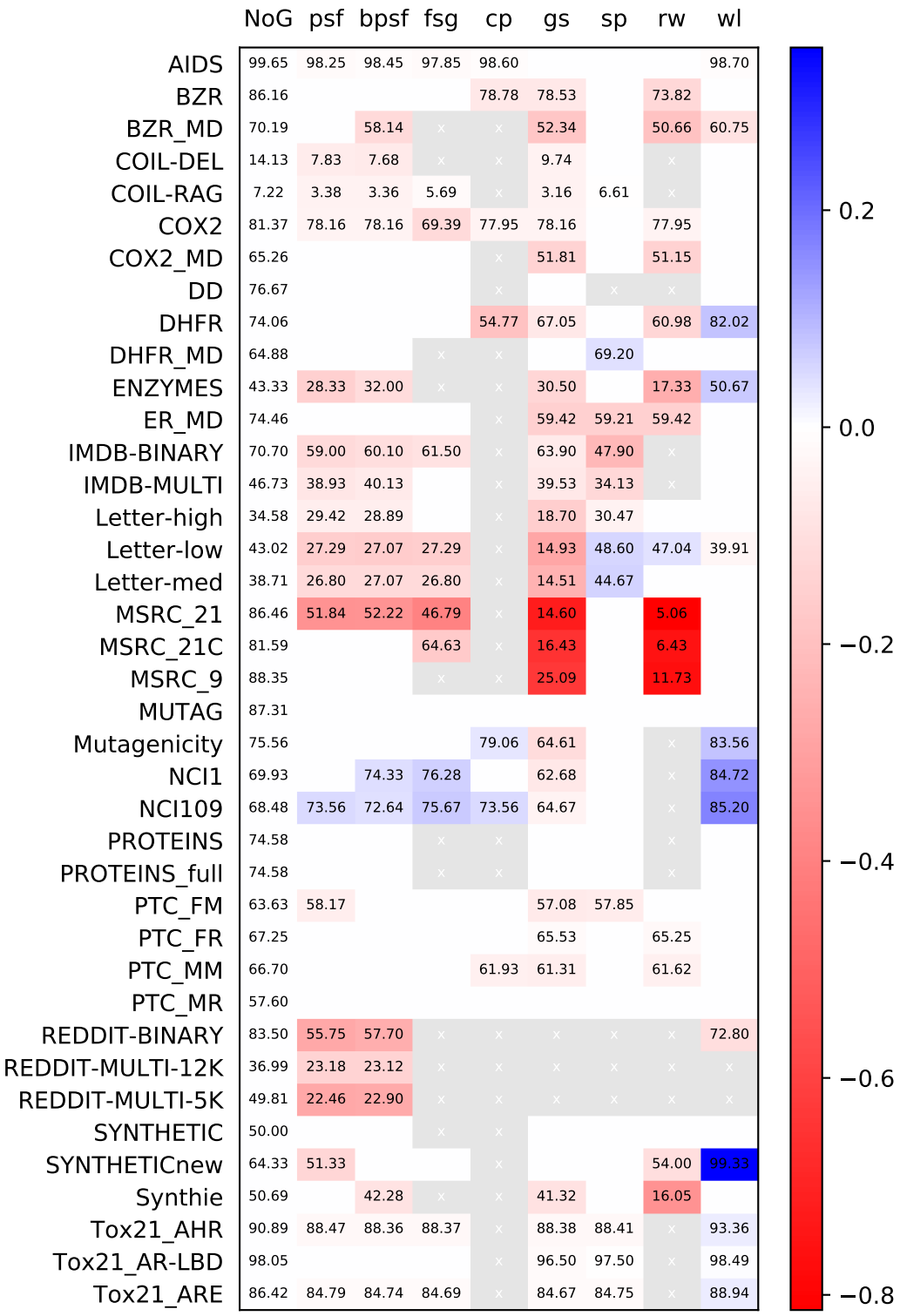

Fig. 1. Prediction measures in accuracy (in %) achieved by graph kernels compared to the NoG kernel. Red color indicates a significantly worse result, blue a significantly better result. White cells indicate no significant difference, while gray cells indicate failure to compute the kernel on a dataset.

sooner than FSG due to an out of memory error. This is due to the fact that our implementation of cp does not have a parameter that stops it from possibly enumerating exponentially many cycles. Some graph kernels from [17] were also very slow to compute. In particular, rw could not terminate in reasonable time on many datasets.

Our results show that for an overwhelming amount of datasets the baseline method does not perform significantly worse than the comparing graph kernels. Rather the opposite is the case. With the exception of the Weisfeiler Lehman kernel, the comparing methods fall behind the baseline on more datasets than they exceed it. Furthermore, even wl outperforms the baseline in only very few cases like NCI1, NCI109 and SYNTHETICnew.

5 Conclusion

In this article we introduced a very simple kernel that considers graphs as multi sets of their vertex and edge labels while completely disregarding their topology. We evaluated a variety of state-of-the-art graph kernels by comparing them to our method on a wide range of benchmark datasets. The results show that there are only very few combinations of kernels and datasets for which the predictive power of a graph kernel for the task at hand significantly exceeds our baseline method. This suggests that a majority of established graph benchmark datasets are not well suited as an indicator for the quality of graph kernels, making the term 'benchmark' quite misleading. We found no general indication that the sophisticated graph similarity measures, which were evaluated in this article, clearly improve classification results on these datasets over our very simplistic approach.

Our findings for one suggest the need for more challenging graph datasets that highlight the power of graph kernels. In particular, we require datasets for which the classification accuracy depends on the type and amount of structural information that is implicitly being considered in a graph kernel. On a different note, in order to properly evaluate graph kernels, a set of baseline methods is imperative. This also requires a comparison method between the different approaches using statistical measures. As a consequence, we hope for a more systematic evaluation of graph kernels in the community.

Bibliography

[1] Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal statistical society: series B (Methodological) **57**(1), 289–300 (1995). https://doi.org/10.1111/j.2517-6161.1995.tb02031.x

[2] Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: IEEE International Conference on Data Mining (ICDM) Proceedings. pp. 74–81. IEEE Computer Society (2005). https://doi.org/10.1109/icdm.2005.132

[3] Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. Bioinformatics **21**(1), 47–56 (2005). https://doi.org/10.1093/bioinformatics/bti1007

[4] Cai, C., Wang, Y.: A simple yet effective baseline for non-attribute graph classification. ICLR'19 workshop on Representation Learning on Graphs and Manifolds (2018), http://arxiv.org/abs/1811.03508

[5] Deshpande, M., Kuramochi, M., Wale, N., Karypis, G.: Frequent substructure-based approaches for classifying chemical compounds. Transactions on Knowledge and Data Engineering **17**(8), 1036–1050 (2005). https://doi.org/10.1109/tkde.2005.127

[6] Gärtner, T., Flach, P., Wrobel, S.: On graph kernels: Hardness results and efficient alternatives. In: Schölkopf, B., Warmuth, M.K. (eds.) Learning Theory and Kernel Machines. pp. 129–143. Springer (2003)

[7] Horváth, T., Gärtner, T., Wrobel, S.: Cyclic pattern kernels for predictive graph mining. In: Kim, W., Kohavi, R., Gehrke, J., DuMouchel, W. (eds.) ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), Proceedings. pp. 158–167 (2004). https://doi.org/10.1145/1014052.1014072

[8] Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: International Conference on Machine Learning (ICML) Proceedings. pp. 321–328 (2003), http://dl.acm.org/citation.cfm?id=3041838.3041879

[9] Kersting, K., Kriege, N.M., Morris, C., Mutzel, P., Neumann, M.: Benchmark data sets for graph kernels (2016), http://graphkernels.cs.tu-dortmund.de

[10] Kriege, N.M., Johansson, F.D., Morris, C.: A survey on graph kernels. CoRR (2019), http://arxiv.org/abs/1903.11835

[11] Kuramochi, M., Karypis, G.: An efficient algorithm for discovering frequent subgraphs. Transactions on Knowledge and Data Engineering **16**(9), 1038–1051 (2004). https://doi.org/10.1109/TKDE.2004.33, http://glaros.dtc.umn.edu/gkhome/pafi/overview

[12] Orlova, Y., Alamgir, M., von Luxburg, U.: Graph kernel benchmark datasets are trivial, https://sites.google.com/site/feast2015/program/downloads/orlova2015graph.pdf, extended Abstract at FEAST 2015: ICML Workshop on Features and Structures

[13] Ramon, J., Grtner, T.: Expressivity versus efficiency of graph kernels. In: Proceedings of the First International Workshop on Mining Graphs, Trees and Sequences. pp. 65–74 (2003)

[14] Riesen, K., Bunke, H.: IAM graph database repository for graph based pattern recognition and machine learning. In: da Vitoria Lobo, N., et al. (eds.) Structural, Syntactic, and Statistical Pattern Recognition. pp. 287–297. Springer (2008)

[15] Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman graph kernels. Journal of Machine Learning

Research **12**, 2539–2561 (2011), http://www.jmlr.org/papers/volume12/shervashidze11a/shervashidze11a.pdf

[16] Shervashidze, N., Vishwanathan, S.V.N., Petri, T., Mehlhorn, K., Borgwardt, K.M.: Efficient graphlet kernels for large graph comparison. In: Dyk, D.A.V., Welling, M. (eds.) International Conference on Artificial Intelligence and Statistics (AISTATS) Proceedings. pp. 488–495 (2009), http://www.jmlr.org/proceedings/papers/v5/shervashidze09a.html

[17] Siglidis, G., Nikolentzos, G., Limnios, S., Giatsidis, C., Skianis, K., Vazirgiannis, M.: Grakel: A graph kernel library in python. http://arxiv.org/abs/1806.02193 (2018), https://github.com/ysig/GraKeL

[18] Welke, P., Horváth, T., Wrobel, S.: Probabilistic frequent subtrees for efficient graph classification and retrieval. Machine Learning **107**(11), 1847–1873 (2018). https://doi.org/10.1007/s10994-017-5688-7

[19] Yanardag, P., Vishwanathan, S.: Deep graph kernels. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) Proceedings. pp. 1365–1374 (2015). https://doi.org/10.1145/2783258.2783417