Fast Neighborhood Subgraph Pairwise Distance Kernel

Fabrizio Costa, Kurt De Grave {FABRIZIO.COSTA,KURT.DEGRAVE}@CS.KULEUVEN.BE
Dept. of Computer Science, K.U.Leuven, Celestijnenlaan 200A, 3001 Heverlee, Belgium

Abstract

We introduce a novel graph kernel called the Neighborhood Subgraph Pairwise Distance Kernel. The kernel decomposes a graph into all pairs of neighborhood subgraphs of small radius at increasing distances. We show that using a fast graph invariant we obtain significant speed-ups in the Gram matrix computation. Finally, we test the novel kernel on a wide range of chemoinformatics tasks, from antiviral to anticarcinogenic to toxicological activity prediction, and observe competitive performance when compared against several recent graph kernel methods.

1. Introduction

Since the introduction of convolution kernels in (Haussler, 1999), the decomposition approach has been the the guiding principle in kernel design for structured objects. According to such approach, a similarity function between discrete data structures can be obtained by decomposing each object into parts and by devising a valid local kernel between the subparts. For over ten years machine learning researchers have exploited the remarkable property that it is possible to efficiently compute this type of kernels even when objects admit an exponential number of decompositions. This is possible if an efficient method to enumerate the parts can be produced and if the sum over a potentially exponential number of local kernel can be performed in polynomial time (e.g. through dynamic programming). However, as the dimension of the feature space associated with the kernel becomes exponentially larger, there is an increasing probability that a significant fraction of the feature space dimensions will be poorly correlated with the target function. As a consequence, even when using large margin classifiers, one can fail to obtain models with good generalization performance (Ben-David et al., 2002). Possible remedies include down-weighting the contribution of larger fragments and/or bounding a priori their size. Alternatively one can try to identify a strong bias, relevant to the task at hand, and consider only a selected subset of structures to limit the dimension of the feature space without degrading the prediction performance. Here we limit the extracted substructures by design, following the physico-chemical intuition that the full molecule's behavior is contained in its electron density field. The continous field can be discretized using the notion of *functional group*, empirically discovered by chemists since a long time. In essence, a functional group is a specific molecular subgraph which can be viewed as characteristic local electron density distribution that remains fairly constant, independent of the environment. In the same spirit, we employ pairs of neighborhood subgraphs of increasing sizes (i.e. subgraphs induced by all "nearby" vertices, see Section 2). Since each vertex in a molecular graph gives rise to a constant, small number of such subgraphs, the neighborhood graphs can be efficiently enumerated in linear time. In Section 2.4 we show how to perform quick equality matches between large neighborhood subgraphs, which allows us to obtain very fast Gram matrix computation runtimes. We empirically verify in Section 4 that the proposed approach yields predictive models with competitive performance on a wide range of bio- and chemoinformatics tasks.

Appearing in *Proceedings of the 27^{th} International Conference on Machine Learning*, Haifa, Israel, 2010.

2. Method

2.1. Graph Definitions and Notation

We closely follow the notation in (Gross & Yellen, 2003). A graph $G = (V, E)$ consists of two sets V and E. The notation $V(G)$ and $E(G)$ is used when G is not the only graph considered. The elements of V are called *vertices* and the elements of E are called *edges*. The *distance* between two vertices, denoted $\mathcal{D}(u, v)$, is the length of the shortest path between them. The *neighborhood* of radius r of a vertex v is the set of vertices at a distance less than or equal to r from v and is denoted by $N_r(v)$. In a graph G, the *induced-subgraph*

on a set of vertices $W = \{w_1, \ldots, w_k\}$ is a graph that has W as its vertex set and it contains every edge of G whose endpoints are in W. The *neighborhood subgraph* of radius r of vertex v is the subgraph induced by the neighborhood of radius r of v and is denoted by $\mathcal{N}_r^v$. A *labeled graph* is a graph whose vertices and/or edges are labeled, possibly with repetitions, using symbols from a finite alphabet. We denote the function that maps the vertex/edge to the label symbol as $\mathcal{L}$. Two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic*, which we denote by $G_1 \simeq G_2$, if there is a bijection $\phi : V_1 \to V_2$, such that for any two vertices $u, v \in V_1$, there is an edge uv if and only if there is an edge $\phi(u)\phi(v)$ in G_2. An isomorphism is a structure-preserving bijection. Two labeled graphs are isomorphic if there is an isomorphism that preserves also the label information, i.e. $\mathcal{L}(\phi(v)) = \mathcal{L}(v)$. An *isomorphism invariant* or *graph invariant* is a graph property that is identical for two isomorphic graphs (e.g. the number of vertices and/or edges). A *certificate for isomorphism* is an isomorphism invariant that is identical for two graphs if and only if they are isomorphic.

2.2. Kernel Definition and Notation

We follow the notation in (Haussler, 1999). Given a set X and a function $K : X \times X \to \mathbb{R}$, we say that K is a *kernel on* $X \times X$ if K is symmetric, i.e. if for any x and $y \in X$ $K(x, y) = K(y, x)$, and if K is *positive-semidefinite*, i.e. if for any $N \geq 1$ and any $x_1, \ldots, x_N \in X$, the matrix K defined by $K_{ij} = K(x_i, x_j)$ is positive-semidefinite, that is $\sum_{ij} c_i c_j K_{ij} \geq 0$ for all $c_1, \ldots, c_N \in \mathbb{R}$ or equivalently if all its eigenvalues are nonnegative. It is easy to see that if each $x \in X$ can be represented as $\phi(x) = \{\phi_n(x)\}_{n \geq 1}$ such that K is the ordinary l_2 dot product $K(x, y) = \langle \phi(x), \phi(y) \rangle = \sum_n \phi_n(x)\phi_n(y)$ then K is a kernel. The converse is also true under reasonable assumptions (which are almost always verified) on X and K, that is, a given kernel K can be represented as $K(x, y) = \langle \phi(x), \phi(y) \rangle$ for some choice of ϕ. In particular it holds for any kernel K over $X \times X$ where X is a countable set. The vector space induced by ϕ is called the *feature space*. Note that it follows from the definition of positive-semidefinite that the *zero-extension* of a kernel is a valid kernel, that is, if $S \subseteq X$ and K is a kernel on $S \times S$ then K may be extended to be a kernel on $X \times X$ by defining $K(x, y) = 0$ if x or y is not in S. It is easy to show that kernels are closed under summation, i.e. a sum of kernels is a valid kernel.

Let now $x \in X$ be a *composite structure* such that we can define $x_1, \ldots, x_D$ as its parts[1]. Each part is such that $x_d \in X_d$ for $d = 1, \ldots, D$ with $D \geq 1$ where each X_d is a countable set. Let R be the relation defined on the set $X_1 \times \ldots \times X_D \times X$, such that $R(x_1, \ldots, x_D, x)$ is true iff $x_1, \ldots, x_D$ are the parts of x. We denote with $R^{-1}(x)$ the inverse relation that yields the parts of x, that is $R^{-1}(x) = \{x_1, \ldots, x_D : R(x_1, \ldots, x_D, x)\}$. In (Haussler, 1999) it is demonstrated that, if there exist a kernel K_d over $X_d \times X_d$ for each $d = 1, \ldots, D$, and if two instances $x, y \in X$ can be decomposed in $x_1, \ldots, x_d$ and $y_1, \ldots, y_d$, then the following generalized convolution:

$$K(x, y) = \sum_{\substack{x_1, \ldots, x_d \in R^{-1}(x) \\ y_1, \ldots, y_d \in R^{-1}(y)}} \prod_{d=1}^{D} K_d(x_d, y_d)$$

is a valid kernel called a *convolution* or *decomposition* kernel[2]. In words: a decomposition kernel is a sum (over all possible ways to decompose a structured instance) of the product of valid kernels over the parts of the instance.

2.3. The Neighborhood Subgraph Pairwise Distance Kernel

Given the notation introduced in the previous sections, in the following we define the Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) as an instance of a decomposition kernel.

We define the relation $R_{r,d}(A^v, B^u, G)$ between two rooted graphs A^v, B^u and a graph G to be true iff both A^v and B^u are in $\{\mathcal{N}_r^v : v \in V(G)\}$, where we require that A^v (B^u) be isomorphic to some $\mathcal{N}_r$ to verify the set inclusion, and that $\mathcal{D}(u, v) = d$. In words: the relation $R_{r,d}$ selects all pairs of neighborhood graphs of radius r whose roots are at distance d in a given graph G.

We define $\kappa_{r,d}$ over $\mathcal{G} \times \mathcal{G}$ as the decomposition kernel on the relation $R_{r,d}$, that is:

$$\kappa_{r,d}(G, G') = \sum_{\substack{A_v, B_u \in R_{r,d}^{-1}(G) \\ A'_{v'}, B'_{u'} \in R_{r,d}^{-1}(G')}} \delta(A_v, A'_{v'})\delta(B_u, B'_{u'})$$

where the *exact matching kernel* $\delta(x, y)$ is 1 if $x \simeq y$ (i.e. if the graph x is isomorphic to y) and 0 otherwise. In words: $\kappa_{r,d}$ counts the number of identical pairs of

[1] Note that the set of parts needs not be a partition for the composite structure, i.e. the parts may "overlap".

[2] To be precise, the valid kernel is the zero-extension of K to $X \times X$ since $R^{-1}(x)$ is not guaranteed to yield a non-empty set for all $x \in X$.

neighboring graphs of radius r at distance d between two graphs (see Figure 1).

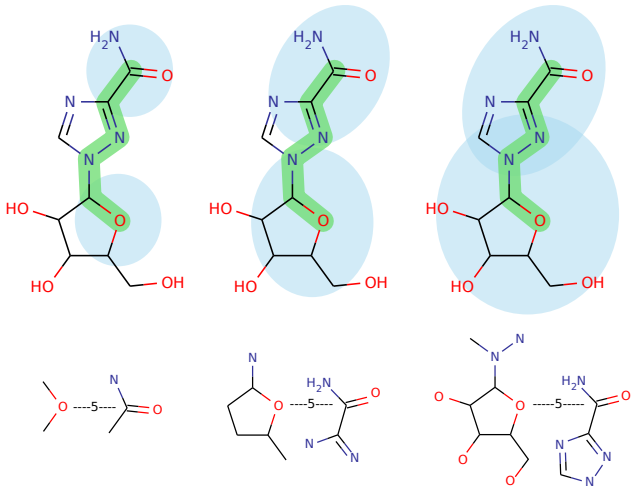

Figure 1. Illustration of pairs of neighborhood graphs for radius $r = 1, 2, 3$ and distance $d = 5$. Note that neighborhood graphs can overlap.

The Neighborhood Subgraph Pairwise Distance Kernel is finally defined as:

$$K(G, G') = \sum_r \sum_d \kappa_{r,d}(G, G').$$

For efficiency reasons however, in this work we consider the zero-extension of K obtained by imposing an upper bound on the radius and the distance parameter: $K_{r^*,d^*}(G, G') = \sum_{r=0}^{r^*} \sum_{d=0}^{d^*} \kappa_{r,d}(G, G')$, that is, we are limiting NSPDK to the sum of the $\kappa_{r,d}$ kernels for all increasing values of the radius (distance) parameter up to a maximum given value r^* (d^*). Furthermore we consider a normalized version of $\kappa_{r,d}$, that is: $\hat{\kappa}_{r,d}(G, G') = \frac{\kappa_{r,d}(G,G')}{\sqrt{\kappa_{r,d}(G,G)\kappa_{r,d}(G',G')}}$, to ensure that relations of all orders are equally weighted regardless of the size of the induced part sets[3].

Finally, it is easy to show that the Neighborhood Subgraph Pairwise Distance Kernel is a valid kernel as: 1) it is built as a decomposition kernel over the countable space of all pairs of neighborhood subgraphs of graphs of finite size; 2) the kernel over parts (the exact matching kernel) is a valid kernel; 3) the zero-extension to bounded values for the radius and distance parameters preserves the kernel property; and 4) so does the normalization step.

[3] As the number of neighborhood graphs grows exponentially with the radius, large (infrequent) subgraphs tend to dominate the kernel value with negative effects on the generalization performance of predictive systems.

2.4. Graph Invariant

The NSPDK includes an exact matching kernel over two graphs which is equivalent to solving the graph isomorphism problem (ISO). Since the existence of (deterministic) polynomial algorithms for ISO is still an open problem, we have to resort to one of two strategies: 1) limit the class of graphs under consideration and solve ISO exactly; or 2) give an approximate (fast) solution of ISO on general graphs. Here we opt for the latter solution since we are mainly concerned with application domains where the number of graphs to be processed are in the range of tens to hundreds of thousands and application specific pre-processing (see Section 3.2) might alter the class of the input graphs (making them non-outerplanar for example).

In this work we implement the exact matching kernel $\delta(G_h, G'_{h'})$ in two steps: 1) we compute a fast graph invariant encoding for G_h and $G'_{h'}$ via a label function $\mathcal{L}^g : \mathcal{G}_h \to \Sigma^*$, where $\mathcal{G}_h$ is the set of rooted graphs and Σ^* is the set of strings over a finite alphabet Σ; 2) we make use of a hash function $H : \Sigma^* \to \mathbb{N}$ to confront $H(\mathcal{L}^g(G_h))$ and $H(\mathcal{L}^g(G'_{h'}))$. In words: we produce an efficient string encoding of graphs from which we obtain a unique identifier via a hashing function from strings to natural numbers. In this way the isomorphism test between two graphs is reduced to a fast numerical identity test. Note that we cannot hope to exhibit an efficient certificate for isomorphism in this way, but only an efficient graph invariant at most, i.e. there will be cases where two non-isomorphic graphs are assigned the same identifier.

The graph encoding $\mathcal{L}^g(G_h)$ that we propose is best described by introducing new label functions for vertices and edges, denoted $\mathcal{L}^n$ and $\mathcal{L}^e$ respectively. $\mathcal{L}^n(v)$ assigns to vertex v the concatenation of the lexicographically sorted listed of distance-label pairs $\langle \mathcal{D}(v, u), \mathcal{L}(u) \rangle$ for all $u \in G_h$. Since G_h is a rooted graph we can exploit the knowledge about the identity of the root vertex h and include, for each vertex v, the additional information of the distance from the root node $\mathcal{D}(v, h)$. $\mathcal{L}^e(uv)$ assigns to edge uv the label $\langle \mathcal{L}^n(u), \mathcal{L}^n(v), \mathcal{L}(uv) \rangle$. $\mathcal{L}^g(G_h)$ assigns to the rooted graph G_h the concatenation of the lexicographically sorted list of $\mathcal{L}^e(uv)$ for all $uv \in E(G_h)$. In words: we relabel each vertex with a string that encodes the vertex distance from all other labeled vertices (plus the distance from the root vertex); the graph encoding is obtained as the sorted edge list, where each edge is annotated with the endpoints' new labels.

We finally resort to a Merkle-Damgård construction based hashing function for variable-length data to map the graph encoding string to a 32-bit integer.

2.5. Algorithmic Complexity

The time complexity of the NSPDK depends on two key procedures: 1) the extraction of all pairs of neighborhood graphs $\mathcal{N}_r^v$ at distance $d = 0, \ldots, d^*$, and 2) the computation of the graph invariant for those subgraphs. The first procedure can be efficiently implemented by factoring it into a) the extraction of $\mathcal{N}_r^v$ for all $v \in V(G)$ and b) the computation of distances between pairs of vertices whose pairwise distance is less than d^*. For this latter step we can repeat a breadth-first (BF) visit up to distance d^* for each vertex in $O(|V(G)||E(G)|)$. Note that, on graphs with bounded (low) degree, the complexity is more realistically modeled as a linear function of $|V(G)|$ since a small d^* implies, in practice, that each bounded BF visit can be performed in constant time. The complexity of point a) is linear in the number of edges in the neighborhood (constant in practice for small r). Finally, the complexity of point 2) (the computation of the graph invariant for neighborhood graphs) can be analyzed in terms of i) the computation of the string encoding $\mathcal{L}^g(G_h)$ and ii) the computation of the hash function $H(\mathcal{L}^g(G_h))$. Part i) is dominated by the computation of all pairwise distances in $O(|V(G_h)||E(G_h)|)$ and the sorting of the relabeled edges, which has complexity $O(|V(G_h)||E(G_h)| \log |E(G_h)|)$ since edges are relabeled with strings containing the distance information of the endpoints from all other vertices. The hash function complexity (part ii)) is linear in the size of the string. We conclude that the overall complexity $O(|V(G)||V(G_h)||E(G_h)| \log |E(G_h)|)$ is dominated by the repeated computation of the graph invariant for each vertex of the graph. Since this is a constant time procedure for small values of d^* and r^*, we conclude that the NSPDK complexity is in practice linear in the size of the graph.

Note finally that, to reduce space complexity, we do not manage the hash collisions, as this would force the algorithm to keep in memory all the encoding key - hashed value pairs.

2.6. Related Work

The NSPDK combines in a kernel fashion ideas present in two popular chemoinformatics fingerprint methods: the *circular substructure* and the *atom pair representation*.

A circular substructure representation encodes the immediate neighborhood of an atom. It does so by assigning an initial code to an atom based on the atom type and other information such as the number of bonds, the electric charge, donor/acceptor tendency, etc. The code for an atom and all its neighborhood is then hashed to produce second order encodings. The process is then iterated a given number of times k. The order k corresponds to the radius in bonds up to which features are generated and typically $k = 1$ or 2. Popular descriptors of this type are the Extended Connectivity Fingerprints (ECFP) and the Functional Connectivity Fingerprints developed at SciTegic/Accelrys which have been shown to be effective in similarity search operations (Hert et al., 2004).

The atom pair representation is an adaptation of the pharmacophore points technique to the 2D rather than 3D structural representation. Here all pairs of atoms are encoded together with the length of the shortest path between them. Each atom is typically described by its type and the number of non-hydrogen atoms to which it is bonded or in terms of its binding properties such as being a cation, anion, neutral, hydrogen bond donor/acceptor, hydrophobic, etc . Popular descriptors of this type are the CATS (Chemically Advance Template Search) (Schneider et al., 1999) and the Similog keys (Schuffenhauer et al., 2003) where the number of occurrences of a particular pair (rather than its presence or absence as it is more usual in conventional fingerprint representations) is used. Typically only pairs for distances up to 10 bonds are used (Hert et al., 2004).

3. Empirical Evaluation

We primarily want to answer two questions about the proposed kernel:
Q1 How does the generalization performance of NSPDK compare to other recent graph kernels?
Q2 How does the experimental runtime of NSPDK compare to other fast graph kernels?

3.1. Data sets

NCI-60: The Developmental Therapeutics Program (DTP) at the U.S. National Cancer Institute (NCI) has checked a large number of compounds for evidence of the ability to inhibit the growth of human tumor cell lines[4]. The roughly balanced subset used by (Swamidass et al., 2005) has become a popular benchmark for QSAR algorithm research, often referred to as either NCI-60 or just NCI. The data set contains growth inhibition measurements on 60 cell lines. Each cell line has inhibition data on about 3500 compounds. There are 3910 compounds in the set in total.

HIV: The DTP also runs an AIDS antiviral screening, which has checked a large number of compounds for evidence of protection against HIV-1. The October

[4] http://dtp.nci.nih.gov/docs/cancer/cancer data.html

99 release of the database [5] contains the structures of 42687 molecules. Each of the compounds was tested twice, and 422 were confirmed to be active (CA), 1081 are moderately active (CM), and 41184 inactive (CI).

PTC: The 2000-2001 Predictive Toxicology Challenge (PTC) (Toivonen et al., 2003) was devised to stimulate the development of machine learning techniques for predictive toxicology models. The data originates from the US National Toxicology Program (NTP). The training and test sets have a different class distribution and a different prevailing mode of action , therefore we only use the (corrected) training set, which contains 417 molecules. The aim is to predict the carcinogenicity of the compounds in different rodents, in particular male mice (MM), female mice (FM), male rats (MR), and female rats (FR).

Bursi: (Kazius et al., 2005) have constructed a dataset of 4337 molecular structures with corresponding Ames data[6]. Ames is a short-term in vitro assay designed to detect genetic damage caused by chemicals and has become the standard test to determine mutagenicity. The distribution is 2401 mutagens and 1936 nonmutagens.

D&D: This is dataset of 1178 protein structures constructed by (Dobson & Doig, 2003) and transformed by (Shervashidze & Borgwardt, 2009) in a graph binary classification problem where the task is to distinguish enzymes from non-enzymes. Each protein has been converted into a graph, considering the amino acids as nodes[7] and considering two nodes linked if their 3D distance in the folded protein is less than 6 ångströms. Note that, while small molecules induce graphs with $\approx$ 30 nodes, protein graphs result in much larger graphs ($\approx$ 300 nodes), with some instances exhibiting several thousands of vertices.

3.2. Chemical Graph Augmentation

It can be argued that the compact representation of molecular graphs may be too concise to achieve optimal generalization performance in graph based machine learning algorithms. A possible remedy is to incorporate chemist domain knowledge, e.g. information on important functional groups and ring structures. A simple method to do so is the following: insert additional vertices representing each functional group and ring structure identified in the molecule; connect these new vertices to the vertices representing their constituent atoms; finally add a link for each pair of new vertices when they share constituent atoms, or when their constituent atoms share a bond. Note that all kernel methods, capable of operating on simple connected labeled graphs, can in principle benefit from this type of domain knowledge encoding without incurring any algorithmic modification overhead.

3.3. Benchmarking Graph Kernels

In the following we introduce several other graph kernels that will be used for benchmarking the proposed approach. In particular we restrict our attention to kernels that do not decompose the graph in walks or paths as it has been shown in (Menchetti et al., 2005) and (Shervashidze & Borgwardt, 2009) that they tend to exhibit lower accuracy and have higher runtimes. Here we consider the Graph Fragment Kernel (GFK) introduced in (Wale et al., 2008), the Weighted Decomposition Kernel (WDK) by (Menchetti et al., 2005), the Pairwise Maximum Common Subgraphs Kernel (PMCSK) introduced in (Schietgat et al., 2009), the Neighborhood Subgraph Kernel (NSK) similar in spirit to the fast kernel presented in (Shervashidze & Borgwardt, 2009), and the Pairwise Distance Kernel (PDK), similar to the Equal Length Shortest-Path Kernel by (Borgwardt & Kriegel, 2005).

GFK: The GFK feature space is obtained considering all connected subgraphs up to a given maximum number of edges. GFK differs from NSPDK as it considers an unbiased (i.e. all possible) type of subgraph rather than neighborhood subgraphs. Note that GFK induces larger explicit representations of molecular graphs even when limiting the subgraph size to relatively small values: on average more than 516 ± 381 features per molecule are generated when allowing subgraphs with less than 8 edges on the NCI-60 dataset. NSPDK, in a comparable setting, generates 28 ± 9 different neighborhood subgraphs per molecule, yielding 251 ± 143 features when considering subgraph pairs up to maximum distance 5.

WDK: In the WDK the neighborhood of a given radius is first associated to each vertex in a graph. The WDK is then computed as the product of an exact matching kernel over the vertex label with a kernel over the neighborhood edge multiset. Here the edge label information is augmented with the endpoints labels. Among the differences between WDK and NSPDK there are: the single vs. pairwise subgraph approach, and the "soft" similarity match vs. the "hard" isomorphism match of neighborhood subgraphs.

PMCSK: The PMCSK feature space is obtained considering the maximum common subgraph (MCS) be-

[5] http://dtp.nci.nih.gov/docs/aids/aids data.html

[6] http://www.cheminformatics.org/datasets/bursi

[7] The node label alphabet has size ≈ 90 rather than 20 as the various types of ambiguities are explicitly encoded as additional labels.

tween all pairs of instances in the training set. The authors show that, although the computation of the maximum common subgraph in the general case is an NP-hard problem, one can employ a polynomial-time algorithm if only outerplanar graphs are considered in combination with a special case of subgraph isomorphism called block-and-bridge-preserving (BBP) subgraph isomorphism (Schietgat et al., 2008). In addition to the pairwise vs. single subgraph approach, PMCSK differs from NSPDK in the specific type of subgraphs considered (MCSs vs. neighborhood graphs).

NSK: In the NSK the feature space is obtained considering the neighborhood subgraphs of increasing radii up to a maximum radius r^*. The NSK features are similar in spirit to those obtained by the circular substructure approach (see Sec. 2.6).

PDK: The PDK computes the similarity between two graphs by comparing all pairs of vertices annotated with their pairwise distance. We consider the zero-extension of PDK up to a maximum distance d^*. The PDK features are similar in spirit to those obtained by the atom pair representation approach (see Sec. 2.6).

The NSK and the PDK are special cases of NSPDK: NSK is obtained considering the NSPDK with a maximum distance $d^* = 0$ while PDK is obtained considering the NSPDK with a maximum radius $r^* = 0$. The optimal value for the remaining free parameter (r^* for NSK and d^* for PDK) is experimentally determined via cross-validation.

3.4. NSPDK Empirical Properties

We measured the size of the neighborhood graphs used in the computation of NSPDK with radius ranging between 1 and 4 for the NCI-60 dataset, obtaining 3, 6, 10 and 13 vertices respectively (and approximately the same values for the edge count). We observe that NSPDK can consider significantly larger subgraphs if compared to the GFK (7 edges) with comparable runtimes (see Section 4).

We tested whether the graph invariant proposed in 2.4 is in fact an isomorphism certificate on chemical graphs. On subgraphs extracted from the NCI-60 dataset the answer is affirmative: we have never found non-isomorphic graphs in the set of graphs that received the same identifier. The exact isomorphism test has been computed via the VFLib Graph Matching Library[8].

We computed the number of hash collisions when encoding pairs of neighborhood subgraphs at distances ranging from 0 to 10, on the NCI-60 dataset: for neighborhood subgraphs of radius 2 we do not have collisions, for radius 3 we have 2 collisions out of 551198 unique pairs and for radius 4 we have 15 collisions out of 667505 unique pairs. We conclude that the error introduced by hashing collisions is in practice negligible.

3.5. Experimental setup

We tested the predictive performance of SVM-Light (Joachims, 1999) by stratified 10-fold cross-validation, keeping folds identical between kernels.

We evaluated the generalization performance of the kernels and the augmentation method by the area under the receiver operating characteristic (AUROC) (the plot of the fraction of true positives versus the fraction of false positives).

A number of parameters were optimized by internal cross-validation on the Bursi data. We allowed each kernel to be optionally composed with a polynomial kernel of degree 3, 5, or 7, or with an RBF kernel with gamma equal to 0.1. The trade-off between training error and margin ("C") was selected from {1,10,100}. The maximum radius r^* was selected from {0,1,2,3,4,5}, and maximum distance d^* from {3,4,5,6,8,10,12,14,20}. Augmented and unaugmented graphs were optimized separately. The most frequently selected parameter values for each kernel as found in Bursi were then used for the other datasets, except for D&D where they were optimized separately due to the different nature of the data. For D&D, only radii up to three were considered.

All kernels were normalized before composition. The cost factor (-j) was set to the prevalence ratio of negative to positive examples in the training set. All other SVM-Light parameters were left at their default value.

We used a radius of 4 for the WDK, motivated by the results on the HIV dataset by (Menchetti et al., 2005).

For GFK, we used the AFGen program of (Wale et al., 2008) to obtain the feature vector. The maximum subgraph size was set to the default value of 7 edges. We did not implement the length-differentiated min-max kernel but rather used the same extensive kernel parameter optimization as for all other kernels. We could not run AFGen on augmented graphs because these exceed its hard-coded maximum node degree.

The same code, with the appropriate parameter settings, has been used for the NSPDK, NSK and PDK.

PMCSK was not optimized as described above, but used a Tanimoto kernel, no cost factor, and internally cross-validated C values as in (Schietgat et al., 2009).

[8] http://amalfi.dis.unina.it/graph/db/vflib-2.0/doc/vflib.html

The third degree polynomial was chosen for all kernels except for unaugmented PDK and WDK which used fifth degree, and augmented NSPDK which used a linear kernel. "C" was always 1 except for augmented NSPDK where it was 10. The optimal choice for r^* was 3, except for augmented NSK where 2 was sufficient. There was some variation in the choice of d^*: 12 for unaugmented PDK, 3 for augmented PDK, 5 for unaugmented NSPDK and 4 for augmented NSPDK.

4. Results and Discussion

In Table 1, we present an overview of the AUROC performance for an SVM model trained with different graph kernels over unaugmented and augmented molecular graphs (denoted G and G_a respectively). In the same table we indicate in boldface the methods with highest *accuracy*, or not significantly worse according to the binomial sign test at $p < 0.05$. We observe that the proposed NSPDK is never significantly worse than the most accurate method. We note moreover that NSPDK performance compares favorably with the best results reported in (Wale et al., 2008) (the state-of-the-art on HIV and NCI-60 to the authors knowledge). As a side note, we report that the relative error reduction, when using the augmented molecular graphs vs. the unaugmented ones, varies from 5% for PDK, to 3% for WDK, to 1% for PMCSK, NSK and NSPDK; this result is in agreement with the intuition that graph methods sensible to larger fragments automatically capture most of the functional group related information. The error rates of augmented NSK and NSPDK on Bursi are about 14.5%. Unaugmented NSPDK and augmented PMCSK achieve 15%. Since (Kazius et al., 2005) mention that the average interlaboratory reproducibility error of Ames tests is 15%, one cannot hope, on this dataset, to do much better. Results for the D&D dataset have been computed only for NSK and NSPDK due to infeasible runtimes for WDK and PMCSK, and unmet hard-wired constraints on the vertex labels and degree for GFK. NSK achieved AUROC 81.4% with radius 0 (and around 80% with higher radii), while NSPDK achieved 85.9% with radius 1 and maximum distance 3 (85.5% with higher radii). The runtimes for the Gram matrix computation where $7.9 \cdot 10^2$ and $8.2 \cdot 10^2$ seconds respectively. We observe that NSPDK achieves a 30.2% error reduction over the best results reported by (Shervashidze & Borgwardt, 2009) with comparable runtimes. In Table 2 we report the runtime required for the Gram matrix computation for the different kernels, normalized but not composed. Augmented molecular graphs have a significantly larger number of vertices and edges, hence are slower to process. The time required for augmentation is shown separately. Obviously, the augmentation step is of linear time complexity in the number of molecules, while the Gram matrix computation is quadratic. The programs were executed on a single core of an Intel Core2 Quad Q9550 CPU (2.8GHz), except for the HIV dataset which was run on an Intel Xeon E5420 CPU (2.5GHz) due to 64-bit support of the operating system.

Table 2. Net CPU time of graph kernels in seconds

	NCI-60	HIV	PTC	Bursi
# of mol.	3910	42687	417	4337
Aug. time	$3.5 \cdot 10^2$	$3.4 \cdot 10^3$	$3.4 \cdot 10^1$	$1.2 \cdot 10^2$
GFK(G)	$3.5 \cdot 10^1$	$1.4 \cdot 10^4$	$3.1 \cdot 10^0$	$7.3 \cdot 10^1$
WDK(G)	$1.8 \cdot 10^3$	$1.6 \cdot 10^5$	$8.0 \cdot 10^0$	$1.1 \cdot 10^3$
WDK(G_a)	$2.3 \cdot 10^3$	$2.3 \cdot 10^5$	$1.4 \cdot 10^1$	$1.5 \cdot 10^3$
PMCSK(G)	$2.8 \cdot 10^5$	$3.3 \cdot 10^{4*}$	$6.2 \cdot 10^2$	$3.5 \cdot 10^5$
PMCSK(G'_a)	$2.8 \cdot 10^5$	$3.3 \cdot 10^{4*}$	$6.3 \cdot 10^2$	$3.5 \cdot 10^5$
PDK(G)	$4.2 \cdot 10^1$	$3.9 \cdot 10^3$	$1.0 \cdot 10^0$	$3.6 \cdot 10^1$
PDK(G_a)	$7.7 \cdot 10^1$	$4.2 \cdot 10^3$	$2.0 \cdot 10^0$	$5.7 \cdot 10^1$
NSK(G)	$6.2 \cdot 10^1$	$3.1 \cdot 10^3$	$2.8 \cdot 10^0$	$5.1 \cdot 10^1$
NSK(G_a)	$3.5 \cdot 10^2$	$6.0 \cdot 10^3$	$1.4 \cdot 10^1$	$2.0 \cdot 10^2$
NSPDK(G)	$1.2 \cdot 10^2$	$1.0 \cdot 10^4$	$3.4 \cdot 10^0$	$1.1 \cdot 10^2$
NSPDK(G_a)	$4.6 \cdot 10^2$	$1.9 \cdot 10^4$	$1.6 \cdot 10^1$	$2.9 \cdot 10^2$

* MCSs derived only from the 1503 CA-CM molecules.

We observe that: 1) the runtime for the WDK neighborhood soft matching is one order of magnitude higher than the graph invariant identity test for NSPDK; 2) the runtime for extracting and matching maximum common subgraphs for all pairs of molecules in PMCSK is three orders of magnitude higher than the graph invariant extraction and identity test for NSPDK; 3) runtimes for random walk kernels and tree kernels are, as reported in (Shervashidze & Borgwardt, 2009), five to six orders of magnitude higher (estimated on two NCI datasets) than for NSPDK.

5. Conclusions

In this paper we presented a novel fast graph kernel based on exact matching between pairs of small subgraphs. Empirical results confirm the intuition that using relatively large fragments in a pairwise fashion improves generalization performance on a wide range of bio- and chemoinformatics tasks. Moreover, the use of fast graph invariant procedures allows a speed-up of several orders of magnitude for Gram matrix computations when compared with kernels based on soft matching or more complex subgraph definition.

The source code of the kernel can be obtained from `http://dtai.cs.kuleuven.be/ml/systems`.

Table 1. Generalization performance of kernels on unaugmented and augmented molecular graphs

	NCI-60 (avg.)	HIV CA vs. CM	HIV CACM vs. CI	HIV CA vs. CI	PTC (avg.)	Bursi
AUROC (%)						
GFK(G)	77.8 ± 2.3	82.0 ± 4.7	82.8 ± 1.9	93.9 ± 2.6	62.6 ± 10	89.6 ± 0.3
WDK(G)	71.1 ± 2.4	83.1 ± 4.3	82.9 ± 1.8	94.0 ± 3.4	62.1 ± 7.7	88.0 ± 0.4
WDK(G_a)	**80.0 ± 2.3**	84.2 ± 4.3	83.9 ± 1.7	95.0 ± 2.7	65.1 ± 8.7	90.8 ± 0.2
PMCSK(G)	79.6 ± 2.2	**82.6 ± 6.2**	81.8 ± 2.2	93.0 ± 3.7	64.5 ± 8.8	90.5 ± 1.3
PMCSK(G'_a)	**80.3 ± 2.2**	**82.8 ± 6.2**	**83.2 ± 2.1**	93.4 ± 3.4	65.6 ± 8.8	**91.5 ± 1.1**
PDK(G)	73.4 ± 2.6	81.6 ± 4.6	77.7 ± 1.9	92.6 ± 3.2	61.2 ± 9.7	82.7 ± 0.3
PDK(G_a)	77.8 ± 2.4	82.1 ± 4.2	83.4 ± 2.1	94.5 ± 2.4	64.6 ± 9.9	89.3 ± 0.3
NSK(G)	79.1 ± 2.2	**84.2 ± 4.9**	84.3 ± 2.0	95.3 ± 1.5	67.4 ± 9.4	91.6 ± 0.2
NSK(G_a)	79.4 ± 2.2	**84.4 ± 4.5**	84.1 ± 2.2	94.9 ± 2.1	67.1 ± 9.3	**91.8 ± 0.2**
NSPDK(G)	79.5 ± 2.2	**83.9 ± 5.6**	83.8 ± 2.1	**95.6 ± 1.3**	**69.3 ± 9.5**	**91.7 ± 0.3**
NSPDK(G_a)	**80.1 ± 2.2**	**84.1 ± 4.8**	**84.9 ± 2.1**	**95.1 ± 2.0**	**68.9 ± 9.8**	**92.0 ± 0.2**

Acknowledgments

The authors thank Leander Schietgat for providing assistance for his PMCSK implementation. The authors are funded by GOA/08/008 "Probabilistic Logic Learning".

References

Ben-David, S., Eiron, N., and Simon, H. U. Limitations of learning via embeddings in euclidean half spaces. *J. of Mach. Learning Research*, 3:441–461, 2002.

Borgwardt, K and Kriegel, H. Shortest-path kernels on graphs. In *Proc. ICDM*, Jan 2005.

Dobson, Paul D. and Doig, Andrew J. Distinguishing enzyme structures from non-enzymes without alignments. *J. Mol. Biol.*, 330(4):771 – 783, 2003.

Gross, J L and Yellen, J. *Handbook of Graph Theory (Discrete Mathematics and Its Applications)*. CRC, 1 edition, Dec 2003. ISBN 1584880902.

Haussler, D. Convolution kernels on discrete structures. Technical Report UCS-CRL-99-10, UC Santa Cruz, 1999.

Hert, J, Willett, P, Wilton, D, and Acklin, P. Comparison of topological descriptors for similarity-based virtual screening using multiple bioactive reference structures. *Org. Biomol. Chem.*, Jan 2004.

Joachims, T. Making large-scale SVM learning practical. In Scholkopf, B., Burges, C., and Smola, A. (eds.), *Advances in Kernel Methods - Support Vector Learning*. MIT-Press, 1999.

Kazius, J., McGuire, R., and Bursi, R. Derivation and validation of toxicophores for mutagenicity prediction. *J. Med. Chem.*, 48(1):312–320, 2005.

Menchetti, S., Costa, F., and Frasconi, P. Weighted decomposition kernels. In *Proc. ICML*, pp. 585–592, 2005.

Schietgat, L., Ramon, J., Bruynooghe, M., and Blockeel, H. An efficiently computable graph-based metric for the classification of small molecules. In *Proc. DS*, pp. 197–209, 2008.

Schietgat, L., Costa, F., Ramon, J., and De Raedt, L. Maximum common subgraph mining: A fast and effective approach towards feature generation. In *Proc. MLG*, pp. 1–3, July 2009.

Schneider, G, Neidhart, W, Giller, T, and Schmid, G. "scaffold-hopping" by topological pharmacophore search: A contribution to virtual screening. *Angew. Chem. Int. Ed.*, Jan 1999.

Schuffenhauer, A, Floersheim, P, Acklin, P, and Jacoby, E. Similarity metrics for ligands reflecting the similarity of the target proteins. *J. Chem. Inf. Comput. Sci*, 43(2):391–405, 2003.

Shervashidze, N and Borgwardt, K. Fast subtree kernels on graphs. *NIPS*, 2009.

Swamidass, S. J., Chen, J., Bruand, J., Phung, P., Ralaivola, L., and Baldi, P. Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. *Bioinformatics*, 21 (suppl1):i359–368, 2005.

Toivonen, H., Srinivasan, A., King, R. D., Kramer, S., and Helma, C. Statistical evaluation of the predictive toxicology challenge 2000-2001. *Bioinformatics*, 19(10):1183–1193, 2003.

Wale, N., Watson, I.A., and Karypis, G. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowl. inf. syst.*, 14:347–375, 2008.