Two New Graphs Kernels in Chemoinformatics

Benoit Gaüzère, Luc Brun, Didier Villemin

Two New Graphs Kernels in Chemoinformatics

Benoit Gaüzère Luc Brun Didier Villemin

2012

Abstract

Chemoinformatics is a well established research field concerned with the discovery of molecule's properties through informational techniques. Computer science's research fields mainly concerned by chemoinformatics are machine learning and graph theory. From this point of view, graph kernels provide a nice framework combining machine learning and graph theory techniques. Such kernels prove their efficiency on several chemoinformatics problems and this paper presents two new graph kernels applied to regression and classification problems. The first kernel is based on the notion of edit distance while the second is based on subtrees enumeration. The design of this last kernel is based on a variable selection step in order to obtain kernels defined on parsimonious sets of patterns. Performances of both kernels are investigated through experiments.

1 Introduction

Chemoinformatics aims to predict or analyse molecule's properties through informational techniques. One of the major principle in this research field is the *similarity principle*, which states that two structurally similar molecules should have similar activities and properties. The structure of a molecule is naturally encoded by a labeled graph $G = (V, E, \mu, \nu)$, where the unlabeled graph (V, E) encodes the structure of the molecule while μ maps each vertex to an atom's label and ν characterizes a type of bond between two atoms (single, double, triple or aromatic).

A first family of methods introduced within the Quantitative Structure-Activity Relationship (QSAR) field is based on the correlation between molecule's descriptors and molecule's properties (e.g. molecule's boiling point). Vectors of descriptors may be defined from structural information (Cherqaoui and Villemin, 1994), physical properties or biological activities and may be used within any statistical machine learning algorithm to predict molecule's properties. Such a scheme allows to benefit from the large set of tools available within the statistical machine learning framework. However, the definition of a vector from a molecule, i.e. a graph, induces a loss of information. Moreover, for each application, the definition of a vectorial description of each molecule remains heuristic.

A slightly different approach is based on graph embedding. Within this framework, a vectorial description of a graph is automatically build from its encoding. A vectorial embedding of a graph may, for example, be defined by associating to each graph its distance to a set of graph prototypes (Emms et al., 2007). Another graph embedding approach is based on the spectral analysis of graphs (Caelli and Kosinov, 2004; Luo et al., 2006). In this last case, the embedding is deduced from the analysis of the eigenvectors and eigenvalues of the adjacency matrix. This rich framework is closely related to several graph embedding methods such as the ones based on random and quantic walks (Berchenko and Teicher, 2009; Emms et al., 2007) or the ones based on heat kernel equations (Elghawalby and Hancock, 2008, 2009).

A third family of methods, based on graph theory, may be decomposed in two subfamilies. The first subfamily (Poezevara et al., 2009), related to the data mining field, aims to discover subgraphs with a large difference of frequencies in a set of positive and negative examples. The second subfamily (Brun et al., 2010), more related to the machine learning field, builds a structural description of each class of molecule so that the classification is conducted by mean of a structural matching between each prototype and a graph representation of an input molecule. Both subfamilies are however mainly restricted to the classification field.

Graph kernels can be understood as symmetric graph similarity measures. Using a semi definite positive kernel, the value $k(G, G')$ where G and G' encode two input graphs corresponds to a scalar product between two vectors $\psi(G)$ and $\psi(G')$ in an Hilbert space. Graph kernels provide thus a natural connection between structural pattern recognition and graph theory on one hand and statistical pattern recognition on the other hand. A large family of kernels is based on the definition of a bag of patterns for each graph

and deduces graph similarity from the similarity between bags. Many of them are defined on linear patterns. For example, Kashima et al. (2004) base graph kernels definition on the comparison of sets of walks extracted from each graph. Ramon and Gärtner (2003) and Mahé and Vert (2008) define kernels using an infinite set of tree patterns instead of walks. These methods improve the limited expressiveness of linear features such as walks. A common drawback of kernels based on walks and tree patterns is that the global similarity between two graphs is based on an implicit enumeration of their common patterns. Such an implicit enumeration, on infinite bags, doers not allow to analyse the relevance of each pattern according to a given dataset or a given property. Using such kernels the weight of each pattern is thus set using generic functions with few parameters allowing to adapt the kernel to a given dataset. Instead of decomposing graphs into an infinite set of substructures (i.e. walks or trees), Shervashidze et al. (2009) define a kernel based on the distribution of a predefined set of unlabeled subgraphs, called *graphlets*. The weighting of each graphlet according to a specific dataset is also not proposed by the authors.

Another approach to the definition of graph kernels is proposed by Neuhaus and Bunke (2007) and Riesen et al. (2007). This approach aims to define semi definite positive kernels from the notion of edit distance. The main challenge of this approach is that edit distance does not always fulfill all requirements of a metric and hence does not always readily lead to a semi definite positive kernel. Using non definite positive kernel, the embedding space associated to the kernel is no more an Hilbert space but a Krein space (Ong et al., 2004) on which many usual mathematical results should be interpreted differently. Problems defined using non definite kernels usually provide a more direct formulation of the problem to minimise but should use with caution most of usual mathematical operations which must be interpreted within a Krein space. Conversely, methods based on definite positive kernel may have to regularize the kernel in order to force its definite positiveness but may apply confidently a large family of mathematical tools defined within Hilbert spaces.

Approaches based on bag of patterns and graph edit distance have complementary advantages and drawbacks. Indeed, the first approach based on the notion of bag of substructures does not readily allows to capture global information about graphs. This last point may be an advantage when a given physical property should be explained by the presence of several substructures instead of the global shape of the molecule. Conversely, properties

explained by global molecule's similarities may be better captured by an approach based on graph edit distance.

This paper presents two new kernels: A first kernel, presented in Section 2, is explicitly based on the problem stated by classification or regression kernel methods. This kernel, based on both graph edit distance and graph Laplacian kernel, is regularised in order to force the definite positiveness of the kernel. These points distinguish our kernel from the ones presented, for example, by Neuhaus and Bunke (2007). A method to update efficiently this kernel is also proposed.

Our second kernel, presented in Section 3, is related to Shervashidze et al. (2009) and Mahé and Vert (2008) as it is based on an enumeration of subtrees within an acyclic labeled graph. The first contribution of this kernel is an extension of Shervashidze et al. (2009) to larger and labeled subgraphs. The second contribution is, unlike Mahé and Vert (2008) method, an explicit enumeration of subtrees within a graph. This last point has important consequences. Indeed, while the set of potential acyclic subgraphs is very large and even infinite for infinite set of labels, our explicit enumeration of subtrees provides a finite set for each data set. This last point allows, using an appropriate variable selection method, to select the substructures which are pertinent for a given property. Such a selection step both provides useful information to interpret chemical mechanisms connected with a given property and allows to improve prediction results.

Using our approach, each molecule is described by a vector encoding the frequencies of its subtrees. From this point of view, our method may be connected with graph embedding methods. However, our vectors are embedded in a space of very large dimension. Indeed, considering a set of N differents labels, the number of different labeled trees up to size 6 is equal to $\Gamma_N^1 + \Gamma_N^2 + \Gamma_N^3 + 2\Gamma_N^4 + 3\Gamma_N^5 + 6\Gamma_N^6$, where Γ_N^k denotes the number of possible subsets with repetitions of size k among N. Since the atoms can be labeled by each of the 118 chemical elements, our feature space has a dimension higher than $\Gamma_{118}^6 \simeq 4.25 \times 10^9$. Nevertheless, for each dataset, we enumerate a finite set of different treelets which defines the subspace on which we encode our set of graphs. Therefore, unlike most of graph embedding methods, the final dimension of the space describing our set of molecules is determined from the data set and not a priori. The efficiency of these two approaches is finally demonstrated in Section 4 through experiments.

2 Kernel from Edit Distance

2.1 Edit Distance

An edit path between two graphs G and G' is defined as a sequence of operations transforming G into G'. Such a sequence may include vertex or edge addition, removal and relabeling. Given a cost function $c(.)$, associated to each operation, the cost of an edit path is defined as the sum of its elementary operation's costs. The minimal cost among all edit paths transforming G into G' is defined as the *edit distance* between both graphs. A high edit distance indicates a low similarity between two graphs while a small one indicates a strong similarity.

According to Neuhaus and Bunke (2007), the computational cost of the exact edit distance grows exponentially with the size of graphs. Such a property limits the computation of exact edit distance to small graphs. To overcome this problem, Fankhauser et al. (2011) defined a method to compute an approximate edit distance in $O(nd^2)$ where n and d are respectively equal to the number of nodes and to the maximal degree of both graphs.

2.2 Graph Laplacian Kernel

Unfortunately, edit distance doesn't always define a metric and trivial kernels based on edit distance may not be semi definite positive (Section 1). Neuhaus and Bunke (2007) proposed several methods to overcome this important drawback. However, the proposed kernels are not explicitly based on the minimization problem addressed by kernel methods. This minimization problem may be stated as follows: Given a kernel k and a dataset of graphs $D = \{G_1, \ldots, G_n\}$, the Gram matrix K associated to D is an $n \times n$ matrix defined by $K_{i,j} = k(G_i, G_j)$. Within the kernel framework, a classification or regression problem based on K may be stated as the minimization of the following formula on the set of real vectors of dimension n:

$$f^* = \arg min_{f \in \mathbb{R}^n} CLoss(f, y, K) + f^t K^{-1} f \tag{1}$$

where $CLoss(.,.,.)$ denotes a given loss function encoding the distance between vector f and the vector of known values y.

Each coordinate f_i of such a vector corresponds to a value attached to graph G_i. A vector f may thus be considered as a function mapping each graph of the database $\{G_1, \ldots, G_n\}$ to a real value. As denoted by Steinke

and Schölkopf (2008), the term $f^t K^{-1} f$ in Equation 1 may be considered as a regularization term which counter balances the fit to data term encoded by function $CLoss(.,.,.)$. Therefore, the inverse of K (or its pseudo inverse if K is not invertible) may be considered as a regularization operator on the set of vectors of dimension n and hence on the set of real functions defined on $\{G_1, \ldots, G_n\}$. Conversely, the inverse (or pseudo inverse) of any semi definite positive regularization operator may be considered as a kernel. We thus follow a kernel construction scheme recently introduced (Brun et al., 2010) which first builds a semi definite positive regularization operator on the set of functions mapping each graph $\{G_1, \ldots, G_n\}$ to a real value. The inverse, or pseudo inverse of this operator defines a kernel on the set $\{G_1, \ldots, G_n\}$.

In order to construct this regularization operator, let us define a $n \times n$ adjacency matrix W by:

$$W_{ij} = e^{-\frac{d(G_i, G_j)}{\sigma}} \tag{2}$$

where $d(.,.)$ denotes the edit distance and σ is a tuning variable. The Laplacian of W is defined as $l = \Delta - W$ where Δ is a diagonal matrix defined by:

$$\Delta_{i,i} = \sum_{j=1}^{n} W_{i,j} \tag{3}$$

Classical results from spectral graph theory (Chung, 1997) establish that l is a symmetric semi definite positive matrix whose minimal eigenvalue is equal to 0. Such a matrix is thus not invertible. To overcome this problem, Smola and Kondor (2003) define the regularized Laplacian $\tilde{l} = I + \lambda l$ of W where λ is a regularization coefficient. The minimal eigenvalue of $\tilde{l}$ is equal to 1 and this matrix is thus definite positive. Moreover, given any vector f, we have:

$$f^t \tilde{l} f = \|f\|^2 + \lambda \sum_{i,j=1}^{n} W_{ij} (f_i - f_j)^2 \tag{4}$$

Intuitively, minimising Equation 4 leads to build a vector f with a small norm which maps graphs with a small edit distance (and thus a strong weight) to close values. Such a constraint corresponds to the regularization term required by Equation 1 in order to smoothly interpolate the test values y over the set of graphs $\{G_1, \ldots, G_n\}$. Our unnormalized kernel, is thus defined as $K_{un} = \tilde{l}^{-1}$.

Note that a regularized normalized Laplacian kernel may alternatively be considered by introducing the matrix:

$$\tilde{L} = \Delta^{-\frac{1}{2}} \tilde{l} \Delta^{-\frac{1}{2}} \tag{5}$$

We have in this case, for any vector f:

$$f^t \tilde{L} f = \sum_{i=1}^{n} \frac{f_i^2}{\Delta_{ii}} + \lambda \sum_{j=1}^{n} \frac{W_{ij}}{\sqrt{\Delta_{ii}\Delta_{jj}}} (f_i - f_j)^2$$

The matrix $\tilde{L}$ is definite positive and its associated kernel is defined as $K_{norm} = \tilde{L}^{-1}$. Note that, our regularized normalized Laplacian kernel is not defined as the inverse of the regularized normalized Laplacian:

$$I + \lambda \Delta^{-\frac{1}{2}} l \Delta^{-\frac{1}{2}} \tag{6}$$

This new formulation is however consistent with the regularization constraint which should be added to Equation 1 and provides significant advantages in the context of incoming data (Section 2.3).

Alternative regularization schemes (Smola and Kondor, 2003; Fouss et al., 2006) may be applied to the Laplacian matrix. One of this regularization scheme of particular interest is the Laplacian exponential diffusion matrix $\tilde{d} = exp(\lambda l)$ associated to the kernel $K = exp(-\lambda l)$. This last kernel may be considered as a generalisation of the regularized Laplacian with further constraints on the derivatives of function f. However, this kernel does not allow to update efficiently kernel values in the context of incoming data (Section 2.3).

2.3 Incoming Data

Let us first consider a kernel defined from the unnormalized Laplacian. Given our learning set $D = \{G_1, \ldots, G_n\}$, the test of a new graph G within a regression or classification scheme requires to update the unnormalized Laplacian l with this new graph and to compute the updated kernel defined as the inverse of the regularized and unnormalized Laplacian $K = (I + \lambda l)^{-1}$. This direct method has a complexity equal to $\mathcal{O}((n+1)^3)$, where n is the size of our data set. Such a method is thus computationally costly, especially for large datasets. In this section, we propose a method to reduce the complexity of this operation.

Given the regularized and unnormalized Laplacian $\tilde{l}_n = (I_n + \lambda(\Delta_n - W_n))$ defined on the dataset D, its updated version $\tilde{l}_{n+1}$ defined on $D \cup \{G\}$ may be expressed as follows:

$$\tilde{l}_{n+1} = \begin{pmatrix} \tilde{l}_n - \delta_n & B \\ B^t & 1 - \sum_i B_i \end{pmatrix}$$

where $B = (-\lambda exp(\frac{-d(G,G_i)}{\sigma}))_{i=\{1,\dots,n\}}$ is deduced from the weights between the new input graph G and each graph $(G_i)_{i=\{1,\dots,n\}}$ of our dataset and δ_n is a diagonal matrix with $(\delta_n)_{i,i} = B_i$.

The minimal eigenvalue of $\tilde{l}_{n+1}$ is equal to 1 (Section 2). This matrix is thus invertible, and its inverse may be expressed using a block inversion scheme:

$$K_{un} = (\tilde{l}_{n+1})^{-1} = \begin{pmatrix} \Gamma & \Theta \\ \Lambda & \Phi \end{pmatrix} \text{ with } \begin{cases} \Gamma &= E^{-1} + \Phi E^{-1} B B^t E^{-1} \\ \Theta &= -E^{-1} B \Phi \\ \Lambda &= -\Phi B^t E^{-1} \\ \Phi &= (1 - \sum_i B_i - B^t E^{-1} B)^{-1} \end{cases} \tag{7}$$

where $E = \tilde{l}_n - \delta_n$. Note that Φ corresponds to a scalar.

The computation of our new kernel, using equation 7, relies on the computation of the inverse of the matrix $E = \tilde{l}_n + \delta_n$ which may be efficiently approximated using a development to order N of $(I - \tilde{l}_n^{-1}\delta_n)^{-1}$:

$$(\tilde{l}_n - \delta_n)^{-1} = \tilde{l}_n^{-1}(I - \tilde{l}_n^{-1}\delta_n)^{-1} \approx \sum_{k=0}^{N} l_n^{-k-1}\delta_n^k \tag{8}$$

Such a sum converges since $\|\tilde{l}_n^{-1}\delta_n\|_2 < 1$, for $\lambda < 1$. Indeed:

$$\|\tilde{l}_n^{-1}\delta_n\|_2 \leq \|\tilde{l}_n^{-1}\|_2 \|\delta_n\|_2 \leq \|\delta_n\|_2 \leq \lambda \max_{i=1,n} exp(\frac{-d(G,G_i)}{\sigma})$$

The last term of this equation is strictly lower than one for any λ lower than one. Moreover, basic matrix calculus show that the approximation error is lower than ϵ for any N greater than:

$$\frac{log(2\epsilon)}{\log(\max_{i=1,n} exp(\frac{-d(G,G_i)}{\sigma}))}. \tag{9}$$

Equation 8 allows to approximate the inverse of $(\tilde{l}_n - \delta_n)$ by a sum of pre computed matrices l_n^{-k-1} multiplied by diagonal matrices. Using such pre calculus, the inverse of $(\tilde{l}_n - \delta_n)$ and hence the computation of our new kernel is achieved in Nn^2.

If we now consider the regularized normalized Laplacian (Section 2.2) $\tilde{L} = \Delta^{-\frac{1}{2}}\tilde{l}\Delta^{-\frac{1}{2}}$, its inverse is defined as: $\tilde{L}^{-1} = \Delta^{\frac{1}{2}}\tilde{l}^{-1}\Delta^{\frac{1}{2}}$ and we have:

$$K_{norm} = \Delta^{\frac{1}{2}} K_{un} \Delta^{\frac{1}{2}} \tag{10}$$

The update of the regularized and normalized Laplacian kernel may thus be deduced from the one of the regularized unnormalized Laplacian kernel.

3 Treelet Kernel

Kernels based on edit distance rely on a direct comparison of each pair of graph. An alternative strategy consists to represent each graph by a bag of patterns and to deduce the similarity between two graphs from the similarity of their bags. This strategy may provide semi definite kernels hereby avoiding the necessity to regularize the whole gram matrix for each incoming data (Section 2.3). As mentioned in Section 1, most of kernels of this family are based on linear patterns (bags of paths, trails or walks). Shervashidze et al. (2009) describe a method to enumerate, for any unlabeled graph, all its subgraphs (called graphlets) composed of up to 5 nodes. We propose here to adapt this method to the enumeration of subtrees of labeled and unlabeled acyclic graphs up to size 6. The resulting patterns are called treelets (Figure 1). This bound on the treelet's size corresponds to a compromise between the expressiveness of our kernel and the time required to enumerate all types of treelets. Indeed, as shown by Otter (1948) the number of different unlabeled treelets grows exponentially with the number of atoms. Moreover, according to Isaacs (1987), the influence of one atom on a chemical property does not usually exceed 3 bonds. Hence, considering larger set of treelets would forbid the design of specific procedures to retrieve each type of treelet without a clear gain on the prediction of chemical properties.

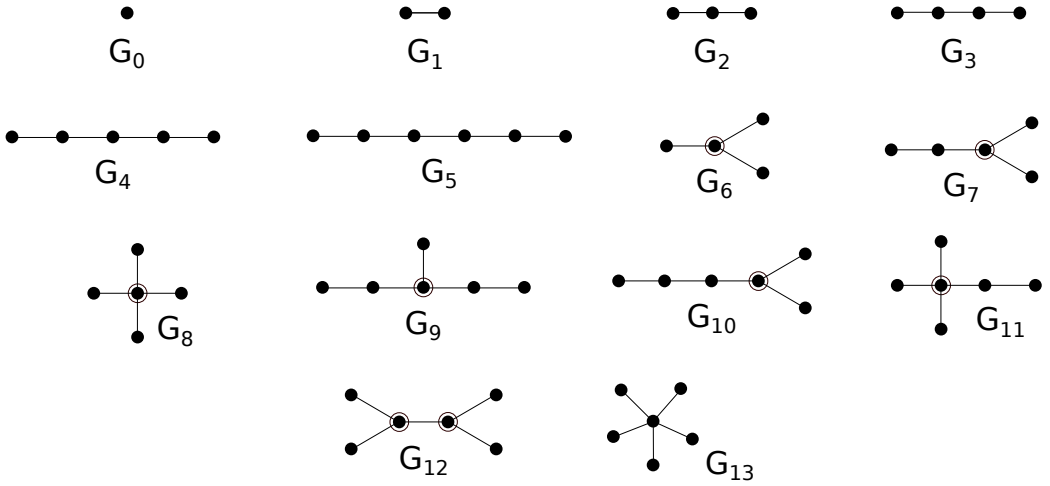


Figure 1: Acyclic and unlabeled graphlets whose maximal size is equal to 6. Centers of 3-star and 4-star are surrounded

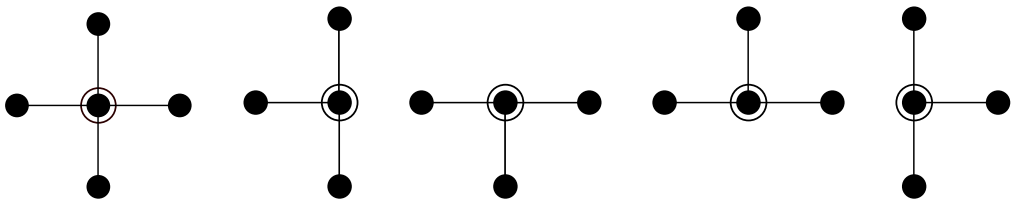
3.1 Computing Embedded Distribution

3.1.1 Structural Analysis

Each treelet corresponds to a subtree of some graph of our database. Such a treelet may be denoted as $t = (V_t, E_t, \mu_t, \nu_t)$ where μ_t and ν_t denote vertex and edge labeling functions. When dealing with unlabeled graphs, treelet enumeration consists to classify treelets according to their graph structures (V_t, E_t). The cardinal of V_t being bounded, only a finite set of tree structures called tree patterns may correspond to a treelet.

Following Shervashidze et al. (2009), the identification of tree patterns is initiated by an enumeration of all paths whose length is lower than or equal to 6. A recursive depth first search with a maximum depth equals to 6 is thus performed from each node of the graph. Note that, using such an enumeration, each path is retrieved from its two extremities and is thus counted twice. In order to prevent this problem, each path composed of at least two nodes is counted $\frac{1}{2}$ times. This first step provides the distribution of patterns G_0, G_1, G_2, G_3, G_4 and G_5 (Figure 1).

Our method to compute the distribution of remaining patterns is based on the detection of nodes of degree 3, 4 and 5. These nodes, respectively denoted R_{3-star}, R_{4-star} and R_{5-star}, are the center of *3-star*, *4-star* and *5-star* patterns. Note that a 4-star pattern (G_8) contains four 3-star patterns



(a) G_7 (b) Four decompositions of G_7 into G_4

Figure 2: Tree pattern G_7 contains 4 G_4.

Pattern	Source pattern	Condition
G_7	3-star	$\lvert\{v; v \in N(R_{3-star}); d(v) \geq 2\}\rvert \geq 1$
G_9	3-star	$\lvert\{v; v \in N(R_{3-star}); d(v) \geq 2\}\rvert \geq 2$
G_{10}	3-star	$\exists v_0 \in N(R_{3-star}); d(v_0) \geq 2$ and $\lvert\{v; v \in N(v_0) - \{R_{3-star}\}; d(v) \geq 2\}\rvert \geq 1$
G_{11}	4-star	$\lvert\{v; v \in N(R_{4-star}); d(v) \geq 2\}\rvert \geq 1$
G_{12}	3-star	$\lvert\{v; v \in N(R_{3-star}); d(v) \geq 3\}\rvert \geq 1$

Table 1: Conditions characterizing patterns derived from 3-star and 4-star. $N(v)$ and $d(v)$ denote respectively the set of neighbours and the degree of vertex v.

(Figure 2). This first enumeration of nodes of degrees 3, 4 and 5 provides the distribution of patterns G_6, G_8 and G_{13}. Patterns G_7, G_9, G_{10} and G_{12} are enumerated from the neighbourhood of 3-star patterns. For example, the tree pattern G_7 matches the neighborhood of a 3-star node if at least one of the neighbours of this 3-star has a degree greater than or equal to 2. Pattern G_{11} is the only subtree derived from a 4-star. Properties characterizing patterns with a 3 or 4-star are summarized in Table 1. Note that pattern G_{12} is symmetric since it contains two centers of 3-star. Such a treelet will thus be counted twice (once from each of its 3-star) and must be counted for $\frac{1}{2}$.

One may easily check that no isomorphism exists between patterns depicted in Figure 1. Moreover, as shown by Cayley (1875), the number of different acyclic and unlabeled graphs with up to 6 vertices and a maximum degree of 4 is equal to 13. Therefore, tree patterns G_0 to G_{12} (Figure 1) represent, up to isomorphisms, all unlabeled acyclic graphs whose size is lower

than or equal to 6 and whose vertex degree is bounded by 4. Adding G_{13} to this set provides all unlabeled acyclic graphs with a size lower than or equal to 6.

3.1.2 Labeling Information

Method described in Section 3.1.1 allows us to associate each treelet to a tree pattern. Using databases of unlabeled graphs, this method can be used to enumerate the number of treelets corresponding to each tree pattern. However, using databases of labeled graphs, treelets with different vertices and edge labels may be associated to a same tree pattern. We propose in this section to identify each treelet by a code composed of two parts: A structural part corresponding to the index of its tree pattern (Section 3.1.1) and a canonical key defined as a sequence of vertices and edge labels. Such a sequence is specific to each tree pattern and designed so as two treelets with a same code must be isomorphic.

Such a key is trivial for linear patterns, i.e. paths. Each path may indeed be associated to two sequences composed of alternated vertices and edge's labels encoding the two possible traversals of this path. By convention, the canonical key of such linear tree patterns is defined as the sequence with the lowest lexicographic order.

Morgan numbering Let us consider a non linear treelet $t = (V_t, E_t, \mu_t, \nu_t)$, where μ_t and ν_t denote respectively vertex and edge labeling functions. Our canonical key for non linear patterns is based on the extended connectivity concept introduced by Morgan (1965). This concept is based on an additional vertex labeling function λ from V to $\mathbb{N}$ called extended labeling. This function is defined by an iterative process which initialises each extended label $\lambda(v)$ to the degree of v. This initial labeling is then extended by assigning to each vertex the sum of extended labels of its neighbors. This summation process is iterated while it increases the number of different labels. The resulting set of extended labels is the same for two isomorphic graphs and is unique for each tree pattern. Figure 3 shows the extended labels computed on non linear patterns.

Since two adjacent vertices v and v' of a treelet may be compared according to $\lambda(v)$ and $\lambda(v')$, our extended label defines a partial order relationship between adjacent vertices of a treelet. Following Morgan (1965), we encode this partial order relationships by a rooted tree. Treelets G_6 to G_{11} have

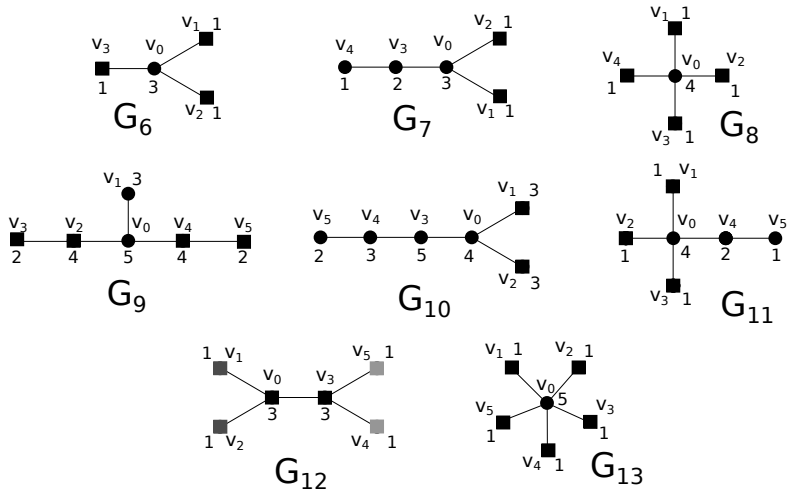


Figure 3: Patterns with Morgan's extended labeling. Possible permutations are represented by square nodes. Different permutations within a same treelet are encoded by different gray levels.

a unique local maximum of the extended labeling function. We thus root these trees on this unique vertex. Treelet G_{12} has two local maxima located on vertices v_0 and v_3 (cf. Figure 3). This treelet is thus associated to two rooted trees respectively rooted on v_0 and v_3.

Key Construction Scheme Our construction scheme of the canonical key of a treelet is based on a traversal of its rooted tree. The construction of our key requires to sort the children of each internal vertex of the tree in order to define a unique traversal and hence a unique key. This sorting step is achieved through the following recursive construction of our canonical key: The key of each leaf v, $key(v)$, is defined as the null label. For each internal node v of the tree, let us consider its set of children $\{v_1, \ldots, v_n\}$. This set is first sorted according to $\lambda(v_i)$ and then according to the chain defined as the concatenation of $\mu_t(v_i)$, $\nu_t(v, v_i)$ and $key(v_i)$. Assuming such an order on $\{v_1, \ldots, v_n\}$, the key of vertex v is defined as:

$$key(v) = \left(\bigodot_{i=1}^{n} \mu_t(v_i).\nu_t(v, v_i) \right) . \bigodot_{i=1}^{n} key(v_i) \tag{11}$$

where $\bigodot$ denotes the concatenation operator.

Using such a construction scheme, the label of each vertex is encoded in the key of its father. In order to integrate the label of the root node, we define the key of a tree rooted on vertex r as $\mu_t(r).key(r)$.

Treelets G_6 to G_{11} are encoded by a single rooted tree, and their canonical code is defined as the index of their tree pattern concatenated with the key of their rooted tree. On the other hand, treelet G_{12} is associated to two trees rooted on its two local maxima. Its canonical code is defined as the index of its tree pattern (12) concatenated with the lowest key of its two associated rooted trees according to the lexicographic order.

Note that, unlike Morgan's representation, our key does not encode the structure of the graph which is explicitly encoded by the index of the tree pattern.

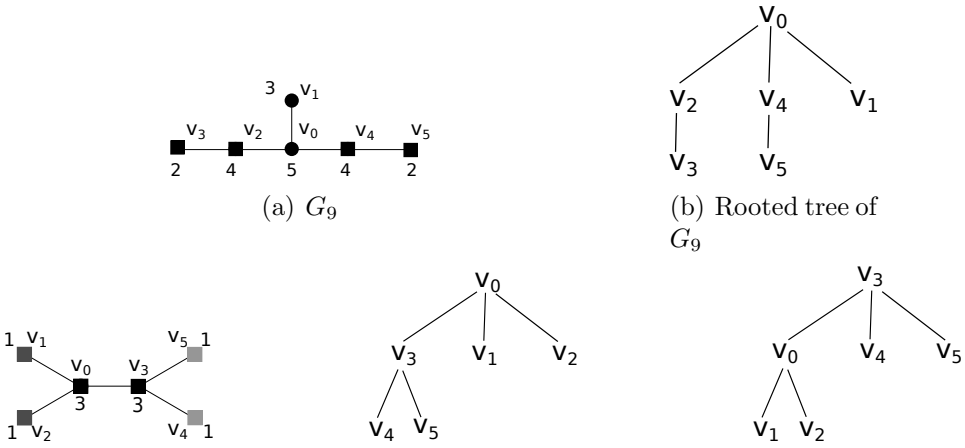
3.1.3 Equal keys and graph isomorphism

Our canonical key is based on extended labels deduced from the structure of the treelet and from vertex and edge labeling functions. Hence, two isomorphic treelets are associated to a same canonical key. Conversely, since one may easily build a linear pattern from its canonical key, two linear patterns (G_0 to G_5) with a same canonical key must be isomorphic.

Within our key construction scheme, the label of a vertex with an unique extended label is located at a fixed position within the canonical key of its treelet. The label of such a vertex may thus be retrieved without any ambiguity from a canonical key. However, a set of children $\{v_1, \ldots, v_n\}$ of a same parent node v with a same extended label will be sorted according to the sequence of edge and vertex labels $\mu_t(v_i)\nu_t(v, v_i)key(v_i)$. This sorting step provides a unique label for two isomorphic treelets but does not allow us to distinguish between any permutations of vertices $\{v_1, \ldots, v_n\}$. We have thus to check, for each treelet, that all permutations of vertices allowed by our code correspond to isomorphic treelets.

Vertices with a same extended label within tree patterns G_6, G_7, G_8, G_{10}, G_{11} and G_{12} are represented by black squares (■) in Figure 3. For each tree pattern, using our key construction scheme, these vertices of degree one are the child of the unique vertex they are connected to. Therefore, our key does not distinguish any permutation among these vertices. Note that, since these vertices have a degree one and are connected to a same vertex, any permutation swapping two of these vertices leads to an isomorphic treelet.

The rooted tree associated to treelet G_9 does not allow us to distinguish



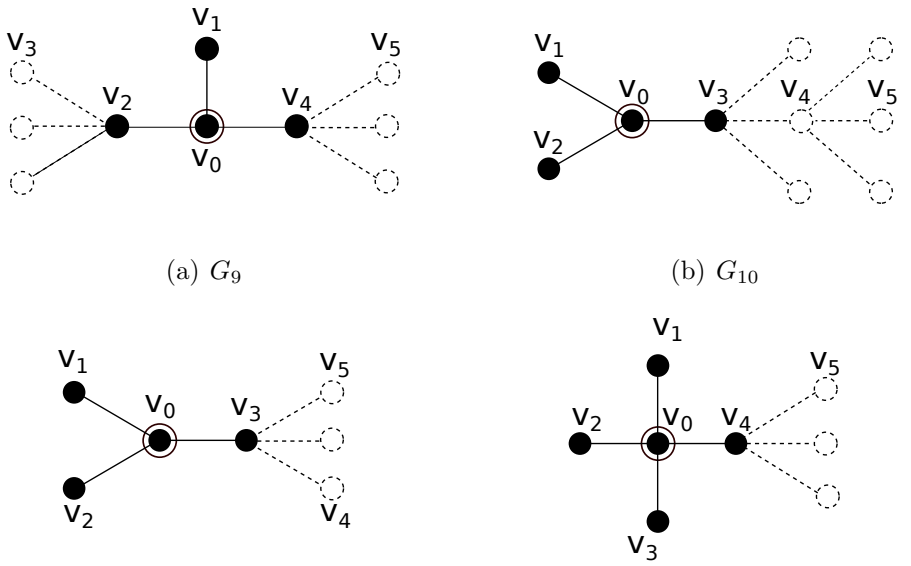
(a) G_9 (b) Rooted tree of G_9 (c) G_{12} (d) First rooted tree of G_{12} (e) Second rooted tree of G_{12}

Figure 4: Rooted trees associated to treelets based on the extended labeling function λ. Values of λ is indicated next to each vertex in (a) and (c).

between branches v_2v_3 and v_4v_5 (Figure 5(b)) since both v_2 and v_4 have an extended label equal to 4. However, the simultaneous permutation of v_2 with v_4 and v_3 with v_5 provides an isomorphic graph. In the same way, the canonical key of G_{12} (Figure 5(c)) does not allow us to distinguish permutations between v_0 and v_3 nor permutations between v_1, v_2 on one hand and v_4, v_5 on the other hand. However, as illustrated in Figure 5(c), all these permutations provide isomorphic treelets. Treelets G_0 to G_{12} are thus uniquely identified by their codes up to isomorphisms.

3.1.4 Complexity

Our treelet enumeration being restricted to tree patterns, its complexity for the enumeration of treelets up to size 5 is bounded by the one of Shervashidze et al. (2009) : $\mathcal{O}(nd^4)$, where n is the number of nodes and d is the maximum degree of the graph. Using a depth first search, unlabeled linear patterns can be enumerated in $\mathcal{O}(nd^{k-1})$, the enumeration of paths of length 6 may thus be achieved in $\mathcal{O}(nd^5)$. The detection of p-stars requires to enumerate all subsets of p neighbours of each node of the graph. This enumeration is achieved in $\mathcal{O}(nd^p)$ operations. The enumeration of G_6, G_8 and G_{13} is thus respectively performed in $\mathcal{O}(nd^3)$, $\mathcal{O}(nd^4)$ and $\mathcal{O}(nd^5)$ operations. The enumeration of



(a) G_9 (b) G_{10}

(c) G_{12} (d) G_{11}

Figure 5: Neighbourhood analysis required to enumerate treelets of size 6 based on 3 and 4-stars.

treelets deduced from 3-stars is performed by the following operations:

- G_9: For each pair of extremities of a 3-star (e.g. v_2 and v_4 in Figure 5(a)), we must enumerate all pairs of vertices (e.g. v_3 and v_5 in Figure 5(a)), where each vertex belongs to the neighbourhood of one of the selected extremity.

- G_{10}: For each extremity, we determine all paths of length 2 (e.g. v_4v_5 in Figure 5(b)) starting on the selected extremity.

- G_{12}: For each extremity (e.g. v_3 in G_{12}, Figure 5(c)), we enumerate all pairs of neighbours (e.g. v_4 and v_5 in Figure 5(c)) of the selected extremity.

All these operations require at most $\mathcal{O}(d^2)$ operations. Since the set of 3-stars is enumerated in $\mathcal{O}(nd^3)$, the complexity of the enumeration of G_9, G_{10} and G_{12} is performed in $\mathcal{O}(nd^5)$ operations. Moreover, the enumeration of treelets corresponding to G_{11} is performed from a 4-star by selecting one neighbour of each extremity (Figure 5(d)). Since this last operation is performed in $\mathcal{O}(d)$

and the enumeration of all 4-stars is performed in $\mathcal{O}(nd^4)$, the enumeration of all treelets corresponding to G_{11} is performed in $\mathcal{O}(nd^5)$. Therefore, the enumeration of all unlabeled treelets is performed in $\mathcal{O}(nd^5)$. Since each unlabeled treelet is finite, the complexity required to compute its key is performed in $\mathcal{O}(1)$. Therefore, the overall complexity required to enumerate labeled treelets from a graph remains equals to $\mathcal{O}(nd^5)$. Note that considering bounded degree graphs, the complexity required to enumerate treelets is linear with the number of graph's nodes.

3.2 Treelet Kernel Definition

Based on our treelet enumeration, we define a function f which associates to each graph G a vector $f(G)$ encoding the distribution of its treelets. Each component of this vector, denoted the *spectrum* of G, is equal to the frequency of a given treelet t in G:

$$f(G) = (f_t(G))_{t \in \mathcal{T}(G)} \text{ with } f_t(G) = |(t \subseteq G)| \tag{12}$$

where $\mathcal{T}(G)$ denotes the set of treelets of G.

Given the function f, the kernel between two graphs is defined as a sum of kernels between the different number of treelets common to both graphs:

$$k_{Treelet}(G, G')q = \sum_{t \in \mathcal{T}(G) \cap \mathcal{T}(G')} K\left(f_t(G), f_t(G')\right) \tag{13}$$

where $\mathcal{T}(G) \cap \mathcal{T}(G')$ denotes the set of common treelets between G and G'. The kernel K between $f_t(G)$ and $f_t(G')$ corresponds to a definite positive kernel between real numbers. In our experiments (Section 4), the RBF, scalar and polynomial kernels have been tested in order to select the best one on each dataset.

3.3 Treelet Weighting

Among the set of treelets found within a dataset, some of them may be irrelevant to explain a given property. Such irrelevant treelets may induce useless calculus and decrease performances of regression or classification algorithms. A first trivial approach to this problem considers all possible permutations over the set of treelets. This approach implies to test 2^p different sets of

Algorithm 1 Forward Selection.

$P = Treelets$
$S = \varnothing$
$nb\_treelets = |P|$
for $i = 0 \rightarrow nb\_treelets$ **do**
  $t = \arg min_t RSS(S \cup t), t \in P$
  $P = P - t$
  $S_{i+1} = S_i \cup t$
end for
return $\arg min_{S_i} RSS(S_i), i \in [0, nb\_treelets]$

Algorithm 2 Backward Elimination.

$S = Treelets$
$nb\_treelets = |S|$
for $i = 0 \rightarrow nb\_treelets$ **do**
  $t = \arg min_t RSS(S - t), t \in S$
  $S_{i+1} = S_i - t$
end for
return $\arg min_{S_i} RSS(S_i), i \in [0, nb\_treelets]$

treelets, where p is the number of different treelets of the dataset. Such a brute force approach is intractable even for small values of p.

In order to define a set of relevant treelet in a less greedy way, we propose two iterative approaches defined for regressions problems. The first one, called forward selection, is an iterative method which starts with an empty set and adds one treelet at each step. The chosen treelet is the one which produces the best regression result (Alg. 1) on the training set using a leave one out procedure. To evaluate the quality of the result, we use the Residual Sum of Squares (RSS) defined as the sum of squared prediction error made for each molecule. A similar approach consists to start from the whole set of treelets and to remove one treelet at each iteration. This second approach is called backward elimination (Alg. 2). Both methods imply to test $\frac{p(p+1)}{2}$ set of treelets. The main difference between these two approaches concerns groups of treelets which are strongly related to a given property when considered simultaneously. For such groups, the removal of any treelet by backward elimination induces a strong increase of the prediction error. These groups are thus preserved by backward elimination algorithm. On the other hand, such groups may not be created by forward selection since the addition of a single treelet may not induce a significant decrease of the prediction error.

4 Experiments

4.1 Classification Problem

Our first experiment evaluates graph kernels defined in Section 2 and 3 on a classification problem. This problem is defined on the monoamine oxidase (MAO) dataset[1] which is composed of 68 molecules divided into two classes: 38 molecules inhibit the monoamine oxidase (antidepressant drugs) and 30 do not. These molecules are composed of different chemical elements and are thus encoded as labeled graphs. Classification accuracy is measured for each method using a leave one out procedure with a two-class SVM. This classification scheme is made for each of the 68 molecules of the dataset.

Table 2 shows results obtained by different methods. The first two lines correspond to methods based on linear patterns. Suard et al. (2002) (Line 1) computes the similarity between two graphs from the average similarity be-

[1]Except AIDS dataset, all databases in this section are available on the TC15 Web page: http://www.greyc.ensicaen.fr/iapr-tc15/links.html#chemistry

	Method	Classification Accuracy
(1)	Suard et al. (2002)	80% (55/68)
(2)	Vishwanathan et al. (2010)	82% (56/68)
(3)	Neuhaus and Bunke (2007)	90% (61/68)
(4)	Riesen et al. (2007)	91% (62/68)
(5)	Normalized Standard Graph Laplacian Kernel	90% (61/68)
(6)	Normalized Fast Graph Laplacian Kernel	90% (61/68)
(7)	Mahé and Vert (2008)	96% (65/68)
(8)	Treelet Kernel	94% (64/68)

Table 2: Classification accuracy on the monoamine oxidase (MAO) dataset.

tween each pair of paths extracted from both graphs. Vishwanathan et al. (2010) (Line 2) counts the number of identical random walks of two graphs. Line 3 (Neuhaus and Bunke, 2007) corresponds to a Gaussian kernel on the suboptimal edit distance and Line 4 (Riesen et al., 2007) corresponds to an embedding of the graphs into a vector encoding the edit distance between the graphs and a set of selected prototypes. Lines 5 and 6 correspond to graph Laplacian kernel (Eq. 10) using a suboptimal graph edit distance (Fankhauser et al., 2011) with node substitution and edge deletion costs set to 1 and edge substitution cost set to the sum of incident node substitution costs. Then, the last two lines correspond to methods based on non linear patterns. Mahé and Vert (2008) (Line 7) deduce the similarity between two graphs from the number of common tree-patterns and treelet kernel (Line 8) is defined in Section 3. In this experiment, this last kernel is defined using a RBF kernel between treelet's frequencies (Equation 13).

Graph Laplacian kernel methods obtain a classification accuracy of 90%. We can note that others prediction models based on edit distance obtains similar results: (Riesen et al., 2007) obtains 91% and (Neuhaus and Bunke, 2007)'s obtains 90% of classification accuracy. This last kernel may however be non semi definite positive. We may additionally notice that the use of our fast inversion method (Table 2, Line 6) defined in Section 2.3 does not

	Method	Classification Accuracy
(1)	Riesen and Bunke (2008)	97.3%
(2)	Suard et al. (2002)	98.5%
(3)	Vishwanathan et al. (2010)	98.5%
(4)	Neuhaus and Bunke (2007)	99.7%
(5)	Riesen et al. (2007)	98.2%
(6)	Graph Laplacian Kernel	99.3%
(7)	Treelet Kernel	99.1%

Table 3: Results on AIDS dataset.

modify graph Laplacian kernel's classification accuracy (Table 2, Line 5). The number of iterations required by this fast inversion method is determined by Equation 9. Our experiments performed on the MAO database show that a value of ϵ equals to 10^{-4} induces a maximum of 9 iterations hence allowing to update the gram matrix in $\mathcal{O}(9n^2)$ instead of $\mathcal{O}(n^3)$ using a standard matrix inversion method. The low value of n on this dataset ($n = 68$) does not induce an important gain on execution time since the average time to update a Gram matrix using method described in Section 2.3 is $0.273ms$ on the MAO database while this time is equal to $0.498ms$ using a direct matrix inversion. The ratio between both execution times is nevertheless about 1.8 hence showing a significant gain.

Thanks to the fact that they take into consideration more structural information than methods based on linear patterns, Mahé and Vert's kernel (Line 7) and our treelet kernel (Line 8) obtain the best results. Note that our method enumerates 153 differents treelets. Mahé and Vert kernel's best results are obtained when considering large and complex trees, i.e. when setting maximum depth and penalization parameters to high values.

Our second classification experiment has been performed on a graph database provided by Riesen and Bunke (2008). This database defined from the AIDS Antiviral Screen Database of Active Compounds is composed of 2000 chemical compounds some of them being disconnected. These chemical compounds have been screened as active or inactive against HIV and they are split into three different sets:

- A train set composed of 250 compounds used to train SVM.

- A validation set composed of 250 compounds used to find parameters giving the best accuracy result.
- A test set composed of remaining 1500 compounds used to test the classification model.

Table 3 shows results obtained by different methods on this dataset. Note that results obtained by Mahé and Vert (2008) are not displayed for this dataset since the source code provided by the authors is restricted to molecules with a degree bounded by 4. The method presented by Riesen and Bunke (Table 3, Line 1) uses the graph edit distance and k-Nearest Neighbour classifier. Others methods based on edit distance, Neuhaus and Bunke (2007) (Table 3, Line 4), Riesen et al. (2007) (Table 3, Line 5) and graph Laplacian kernel (Table 3, Line 6) improve accuracy of the prediction model and obtain better results than simple graph edit distance. Gaussian kernel on edit distance (Table 3, Line 4) combined with SVM obtains the best results on this dataset. Note that the regularization added to Gaussian edit distance kernel affects prediction accuracy while keeping good results. However, fast Gram matrix update hasn't be tested in this dataset since best results are obtained using a parameter $\lambda > 1$ (Section 2.3). Then, Treelet Kernel (Table 3, Line 7), based on non linear patterns and a scalar product kernel between frequencies, outperforms methods based on linear patterns (Table 3, Lines 2 and 3) but without reaching the accuracy of methods based on edit distance. Note however, that 4875 different treelets are enumerated from the train set. A treelet selection procedure for classification applied on this dataset should improve prediction accuracy.

4.2 Regression Problem

Our first regression experiment is based on a database of alkanes (Cherqaoui and Villemin, 1994). An alkane is an acyclic molecule solely composed of carbons and hydrogens. A common encoding consists to implicitly encode hydrogen atoms using the valency of carbon atoms. Such an encoding scheme allows to represent alkanes as acyclic unlabeled graphs. The alkane dataset is composed of 150 molecules associated to their respective boiling points. Using the same protocol than Cherqaoui and Villemin (1994), we evaluate the boiling point of each alkane using several test sets composed of 10% of the database, the remaining 90% being used as training set.

	Method	Average Error ($^\circ C$)	RMSE ($^\circ C$)
(1)	Cherqaoui and Villemin (1994)	3.11	3.70
(2)	Neuhaus and Bunke (2007)	5.42	10.01
(3)	Riesen et al. (2007)	5.27	7.10
(4)	Suard et al. (2002)	4.66	6.21
(5)	Vishwanathan et al. (2010)	10.61	16.28
(6)	Graph Laplacian Kernel	10.79	16.45
(7)	Mahé and Vert (2008)	2.41	3.48
(8)	Treelet Kernel	1.41	1.92

Table 4: Boiling point prediction on alkane dataset. RMSE stands for Root Mean Squared Error.

The first line of Table 4 shows results obtained by a method based on neural networks. The other lines correspond to methods based on graph kernels and boiling point prediction was performed using a kernel ridge regression. Poor results obtained by methods based on edit distance, including graph Laplacian kernel, can be explained by the lack of local information when dealing with unlabeled graphs. Indeed, using such graphs, the heuristic used to approximate graph edit distance (Fankhauser et al., 2011) maps the set of vertices of both graphs using uniquely the degree of vertices. Such a method thus considers several mappings as equivalent if several vertices with a same degree exist in both graphs. In this case, the suboptimal graph edit distance thus induces a poor graph discrimination. This lack of local information within unlabeled graphs also explains the poor results obtained by Suard et al. (2002) and Vishwanathan et al. (2010). Indeed, using unlabeled graphs, these kernels are based on linear structures which are only discriminated by their lengths. On the other hand, kernels based on non linear patterns, treelet kernel and Mahé and Vert (2008), outperform previous results of Cherqaoui and Villemin (1994) based on neural networks combined with chemical descriptors. Note that we didn't select relevant patterns in this experiment since our set of unlabeled treelet is composed of only 13 treelets. In this experiment, the kernel between treelet's frequencies, on which is based our treelet kernel, is set to the RBF kernel.

	Method	Average Error ($^\circ C$)	RMSE ($^\circ C$)
(1)	Cherqaoui et al. (1994a)	3.01	5.102
(2)	Neuhaus and Bunke (2007)	6.84	9.04
(3)	Riesen et al. (2007)	7.39	9.94
(4)	GLK (Diffusion Process)	7.27	9.84
(5)	Suard et al. (2002)	7.28	12.37
(6)	Vishwanathan et al. (2010)	14.64	18.95
(7)	Mahé and Vert (2008)	5.53	9.50
(8)	Treelet Kernel (TK)	4.90	7.80
(9)	TK with Forward Selection	3.23	4.36
(10)	TK with Backward Elimination	2.63	3.70

Table 5: Boiling point prediction on acyclic molecule dataset. TK stands for treelet kernel; GLK for Graph Laplacian Kernel.

	Method	Average Error($^\circ C$)	RMSE($^\circ C$)
(1)	Neuhaus and Bunke (2007)	7.78	10.27
(2)	Riesen et al. (2007)	7.66	10.19
(3)	GLK (Diffusion Process)	8.58	11.99
(4)	Suard et al. (2002)	7.22	12.24
(5)	Vishwanathan et al. (2010)	14.57	18.72
(6)	Mahé and Vert (2008)	6.42	11.02
(7)	Treelet Kernel (TK)	5.27	8.10
(8)	TK with Forward Selection	4.89	7.05
(9)	TK with Backward Elimination	4.87	6.75

Table 6: Boiling point prediction on acyclic molecule dataset using 90% of the dataset as train set and remaining 10% as test set. TK stands for treelet kernel; GLK for Graph Laplacian Kernel.

	Method	Gram Matrix (s)	Prediction (s)
(1)	Neuhaus and Bunke (2007)	1.35	0.05
(2)	Riesen et al. (2007)	2.74	0.01
(3)	GLK (Diffusion Process)	2.86	2.72
(4)	Suard et al. (2002)	7.83	0.18
(5)	Vishwanathan et al. (2010)	19.10	0.57
(6)	Mahé and Vert (2008)	4.98	0.03
(7)	Treelet Kernel (TK)	0.07	0.01
(8)	TK with Forward Selection	0.07	0.01
(9)	TK with Backward Elimination	0.07	0.01

Table 7: Empirical execution times measured on the acyclic molecule regression problem.

The second regression problem is based on a dataset composed of 185 acyclic molecules (Cherqaoui et al., 1994a). This problem also consists in predicting boiling point of molecules. In contrast with the previous dataset, these molecules contain several hetero atoms and are thus represented as acyclic labeled graphs. In order to compare our results, we first use a leave one out method to predict boiling points, as described in Cherqaoui et al. (1994a). Our variable selection step requires $\mathcal{O}(p^2)$ regressions, where p denotes the number of different treelets enumerated on a dataset. This step requires about 4 hours on this dataset using a standard desktop computer. Note that this step is performed during the training set and does not influence the time required to predict the property of a molecule. However, due to this important computational time, we performed in this experiment a single variable selection step on the whole dataset of 185 molecules, instead of considering all subsets of 184 molecules as required by the leave one out protocol.

The first line of Table 5 shows results obtained using a neural network based on the embedding frequencies of 20 substructures chosen by a chemical expert (Cherqaoui et al., 1994a). Then, the next three lines show results obtained by methods based on the notion of edit distance. The Gram matrix associated to the Gaussian kernel defined on suboptimal edit distance (Line 2) obtains intermediate results. Line 3 shows results obtained by graph embed-

ding method using graph edit distance and prototype selection as defined in Riesen et al. (2007). The graph Laplacian kernel (Line 4) with diffusion process regularization ($K = e^{-\lambda l}$, Section 2.2) obtains similar results than alternative methods based on edit distance. We should however note that the regularization of the Laplacian induces a slight decrease of classification results obtained by Neuhaus and Bunke (2007) on this dataset. The performances of the regularized Laplacian ($K = (I + \lambda l)^{-1}$, Section 2.2) on this dataset are below the one of the graph Laplacian kernel with diffusion process regularization and are thus not displayed in Table 5. Methods based on linear patterns (Suard et al., 2002; Vishwanathan et al., 2010) (Lines 5 and 6) suffer from the lack of expressiveness of linear patterns and obtain poor results when predicting the boiling point of such molecules.

Conversely, methods based on non linear patterns, Mahé and Vert (2008) (Line 7) and treelet kernel using a RBF kernel between treelet's frequencies (Line 8), outperform methods based on linear patterns but without reaching the efficiency of Cherqaoui et al. (1994a). This lack of efficiency can be explained by the number of different treelets induced by this dataset. Indeed, treelet kernel method enumerates 142 different treelets in this dataset and some of them are not related with the property to predict. Therefore, information brought by these treelets can be assimilated to noise which decreases prediction results. These prediction results can be improved by applying one of the two treelet selection methods described in Section 3.3. As shown in Table 5, prediction errors made using these selection methods (Lines 9 and 10) outperform results obtained by alternative regression methods on this dataset. Note that forward selection method reduces the set of features to 26 treelets and backward elimination reduces it to 56 treelets. Backward elimination obtains the best result by taking into account information brought by combinations of treelets (Section 3.3). An important difference between treelet kernel approach and Mahé and Vert (2008) is that treelet kernel method computes explicitly the distribution of each treelet within a molecule. This explicit enumeration allows us to weight each treelet separately while Mahé and Vert (2008) only weight features according to their depth or branching cardinality. This difference explains the better results obtained by our treelet kernel combined with a backward selection.

Table 6 shows results obtained on the same dataset using 90% of the dataset as train set and remaining 10% as test set. The method of Cherqaoui et al. (1994a) is not displayed in this table since only results based on a leave one out procedure are available. On this experiment, our treelet kernel is

based on a RBF kernel between treelet's frequencies and our variable selection step is performed only using the train set. With this restricted training set, kernels based on non linear patterns still outperform kernel based on linear ones and our treelet kernel combined with a backward selection step obtain the lowest prediction error.

The first column of Table 7 shows computation times required to compute the Gram matrices of the different kernels studied in this section on our dataset of labeled acyclic molecules (see Tables 5 and 6 to evaluate the performances of these kernels). The second column of Table 7 describes the mean time required to predict the boiling point of each molecule on the same data set using a leave one out procedure. Thanks to the low bounded degree of molecular graphs, treelet enumeration can be performed efficiently (Table 7, Line 7). In addition, treelet enumeration is performed once for each graph and only a sum of RBF kernels between treelet's frequencies is performed for each pair of graphs. Conversely, using an implicit enumeration, we have to perform two enumerations for each pair of graph. Note that the mean prediction time required by Graph Laplacian Kernel is similar to the one required to compute the whole Gram matrix since we have to inverse the regularized Laplacian matrix to compute the updated kernel.

5 Conclusion

We have proposed two new graph kernels using different approaches. The first one is based on the notion of edit distance and is adapted to cases where the similarity between molecules must consider the global shape of the molecule. Thanks to different regularization methods, we define a valid graph kernel from a suboptimal edit distance and we propose a solution to improve the computational complexity of this kernel in case of incoming data for one type of regularisation. The results obtained by this kernel are equal or slightly below the one obtained by alternative methods based on the edit distance which do not perform a regularization step. This results confirms the fact that a regularization step slightly transforms the problem to be minimized by kernel methods. However, such a regularization step is the price to pay in order to work in an Hilbert space and have a clear interpretation of all the mathematical operations performed by kernel methods. Note, nevertheless that some kernel methods have been specifically designed to work on Krein space with a clear interpretation on their limited application domain.

Our second kernel is based on the decomposition of a graph into a set of distinct substructures called treelets. Such a kernel is more specifically designed to capture the similarity between molecules sharing many sub structures. In contrast with many existing graph kernel methods only based on linear patterns, our method is based on linear and non linear patterns which provide additional local information. Moreover, our explicit enumeration of treelets allows both to define an efficient kernel since we avoid an explicit comparison of all pairs of treelets of two graphs and it additionally allows us to integrate within our kernel's definition a treelet selection step which improves the accuracy of our regression results. Our treelet kernel indeed outperforms all other kernels on all our regression experiments. This treelet selection step is however, not yet defined for classification methods and our kernel obtains on these kind of experiments good results which are however always outperformed by an alternative kernel. Designing a specific treelet selection step for classification tasks is one of the priority of our future work. Other perspectives include the extension of our treelet kernel in order to encode ring information while keeping an efficient enumeration.

References

Berchenko, Y., Teicher, M., 2009. Graph embedding through random walk for shortest paths problems. In: Proceedings of the 5th international conference on Stochastic algorithms: foundations and applications. SAGA'09. Springer-Verlag, Berlin, Heidelberg, pp. 127–140.

Brun, L., Conte, D., Foggia, P., Vento, M., Villemin, D., 2010. Symbolic learning vs. graph kernels: An experimental comparison in a chemical application. In: Proceedings of the 14th Conference on Advances in Databases and Information Systems (ADBIS 2010). pp. 31–40.

Caelli, T., Kosinov, S., 2004. An eigenspace projection clustering method for inexact graph matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 26, 515–519.

Cayley, A., 1875. On the analytic forms called trees, with applications to the theory of chemical combinations. Reports British Assoc. Adv. Sci. 9, 427–460.

Cherqaoui, D., Villemin, D., 1994. Use of a neural network to determine the boiling point of alkanes. J. Chem. Soc. Faraday Trans. 90, 97–102.

Cherqaoui, D., Villemin, D., Mesbah, A., Cense, J. M., Kvasnicka, V., 1994a. Use of a Neural Network to Determine the Normal Boiling Points of Acyclic Ethers, Peroxides, Acetals and their Sulfur Analogues. J. Chem. Soc. Faraday Trans. 90, 2015–2019.

Chung, F., 1997. Spectral graph theory. American Mathematical Society.

Elghawalby, H., Hancock, E. R., 2008. Graph characteristic from the gauss-bonnet theorem. In: Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition. SSPR & SPR '08. Springer-Verlag, Berlin, Heidelberg, pp. 207–216.

Elghawalby, H., Hancock, E. R., 2009. Geometric characterizations of graphs using heat kernel embeddings. In: Hancock, E. R., Martin, R. R., Sabin, M. A. (Eds.), IMA Conference on the Mathematics of Surfaces. Vol. 5654 of Lecture Notes in Computer Science. Springer, pp. 124–142.

Emms, D., Wilson, R. C., Hancock, E., 2007. Graph embedding using quantum commute times. In: Escolano, F., Vento, M. (Eds.), 6th IAPR-TC15 International Workshop GbRPR 2007. IAPR TC15, Springer-Verlag, pp. 371–382.

Fankhauser, S., Riesen, K., Bunke, H., 2011. Speeding up graph edit distance computation through fast bipartite matching. In: Jiang, X., Ferrer, M., Torsello, A. (Eds.), Graph-Based Representations in Pattern Recognition. Vol. 6658 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 102–111.

Fouss, F., Yen, L., Pirotte, A., Saerens, M., 2006. An experimental investigation of graph kernels on a collaborative recommendation task. In: Sixth International Conference on Data Mining. pp. 863–868.

Isaacs, N., 1987. Physical Organic Chemistry. Longman Sc. Tech.

Kashima, H., Tsuda, K., Inokuchi, A., 2004. Kernels for graphs. MIT Press, Ch. 7, pp. 155–170.

Luo, B., Wilson, R. C., Hancock, E. R., 2006. A spectral approach to learning structural variations in graphs. Pattern Recognition 39 (6), 1188–1198.

Mahé, P., Vert, J.-P., Oct. 2008. Graph kernels based on tree patterns for molecules. Machine Learning 75 (1), 3–35.

Morgan, H. L., 1965. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. J. of Chem. Doc. 5 (2), 107–113.

Neuhaus, M., Bunke, H., 2007. Bridging the gap between graph edit distance and kernel machines. World Scientific Pub Co Inc.

Ong, C. S., Mary, X., Canu, S., Smola, A. J., 2004. Learning with non-positive kernels. In: Proceedings of the twenty-first international conference on Machine learning. ICML '04. ACM, New York, NY, USA, pp. 639–646.

Otter, R., July 1948. The number of trees. The Annals of Mathematics 49 (3), 583–599.

Poezevara, G., Cuissart, B., Crémilleux, B., 2009. Discovering emerging graph patterns from chemicals. In: Proceedings of the 18th International Symposium on Methodologies for Intelligent Systems (ISMIS 2009). LNCS, Prague, pp. 45–55.

Ramon, J., Gärtner, T., 2003. Expressivity versus efficiency of graph kernels. In: 1st Int. Workshop on Mining Graphs, Trees and Sequences. pp. 65–74.

Riesen, K., Bunke, H., 2008. Iam graph database repository for graph based pattern recognition and machine learning. In: Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition. SSPR & SPR '08. Springer-Verlag, Berlin, Heidelberg, pp. 287–297.

Riesen, K., Neuhaus, M., Bunke, H., 2007. Graph embedding in vector spaces by means of prototype selection. In: Escolano, F., Vento, M. (Eds.), 6th IAPR-TC15 International Workshop GbRPR 2007. IAPR TC15, Springer-Verlag, pp. 383–393.

Shervashidze, N., Vishwanathan, S. V., Petri, T. H., Mehlhorn, K., Borgwardt, K. M., 2009. Efficient graphlet kernels for large graph comparison. In: 12th International Conference on Artificial Intelligence and Statistics (AISTATS). pp. 488–495.

Smola, A., Kondor, R., 2003. Kernels and regularization on graphs. In: Schlkopf, B., Warmuth, M. (Eds.), Learning Theory and Kernel Machines. Vol. 2777 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 144–158.

Steinke, F., Schölkopf, B., November 2008. Kernels, regularization and differential equations. Pattern Recognition 41, 3271–3286.

Suard, F., Rakotomamonjy, A., Bensrhair, A., 2002. Kernel on bag of paths for measuring similarity of shapes. In: European Symposium on Artificial Neural Networks. pp. 355–360.

Vishwanathan, S., Borgwardt, K. M., Kondor, I. R., Schraudolph, N. N., 2010. Graph Kernels. Journal of Machine Learning Research 11, 1201–1242.