Two New Graph Kernels and Applications to Chemoinformatics

Benoit Gaüzère, Luc Brun, Didier Villemin

Two New Graph Kernels and Applications to Chemoinformatics

Benoit Gaüzère†, Luc Brun†, and Didier Villemin‡

†GREYC UMR CNRS 6072, ‡LCMT UMR CNRS 6507,
Caen, France
{benoit.gauzere,didier.villemin}@ensicaen.fr,
luc.brun@greyc.ensicaen.fr

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 the chemoinformatics field are machine learning and graph theory. From this point of view, graph kernels provide a nice framework combining machine learning techniques with graph theory. Such kernels prove their efficiency on several chemoinformatics problems. This paper presents two new graph kernels applied to regression and classification problems within the chemoinformatics field. The first kernel is based on the notion of edit distance while the second is based on sub trees enumeration. Several experiments show the complementary of both approaches.

Keywords: edit-distance, graph kernel, chemoinformatics

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 such as the number of atoms and molecule's properties (e.g. molecule's boiling point). Vectors of descriptors may be defined from structural information [2], 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, ie. a graph, induces a loss of information. Moreover, for each application, the definition of a vectorial description of each molecule remains heuristic.

A second family of methods, based on graph theory may be decomposed in two sub families. The first sub family [8], related to the data mining field, aims to discover sub graphs with a large difference of frequencies in a set of positive and negative examples. The second sub family [1], 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 sub families are however mainly restricted to the classification field.

Graph kernels can be understood as symmetric graph similarity measures. Using a semidefinite positive kernel, the value $k(G, G')$ where G, G' encode two input graphs corresponds to a scalar product between two vectors $\psi(G)$ and $\psi(G')$ in an Hilbert space (this space is only a Krein space if the kernel is non definite). 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. Kashima [5] defines graph kernels based on the comparison of sets of walks extracted from each graph. Ramon and Gärtner [9] and Mahé [6] define kernels using an infinite set of tree patterns instead of walks. These methods improve the limited expressiveness of linear features such as walks hence providing a priori a more meaningful similarity measure. Instead of decomposing graphs into an infinite set of substructures (ie walks or trees), Shervashidze and Borgwardt[12] compute the kernel from the distribution of a predefined set of subgraphs, called *graphlets*. An other approach to the definition of graph kernels is proposed by Neuhaus and Bunke [7]. This approach aims to define definite positive kernels from the notion of edit distance. The main challenge of this approach is that the edit distance does not fulfill all requirements of a metric and hence does not readily lead to a definite positive kernel.

This paper presents two new kernels: A first kernel, presented in Section 2, combines graph edit distance and graph Laplacian kernel notions in order to obtain a definite positive graph kernel. A method to update efficiently this kernel is also proposed. Our second kernel, presented in Section 3, uses a different approach based on an explicit enumeration of subtrees within an acyclic unlabeled graph. The efficiency and complementarity of these two kernels is finally demonstrated in Section 4 through experiments.

2 Kernel from 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 Bunke and Neuhaus[7], the computational cost of the exact edit distance grows exponentially with the size of the graphs. Such a property limits the computation of exact edit distance to small graphs. To overcome this problem, Bunke and Riesen[11] defined a method to compute a sub optimal edit distance. This method computes an approximate edit distance in $O(nv^2)$ where n and v are respectively equal to the number of nodes and to the maximal degree of both graph.

Unfortunately, edit distance doesn't define a metric and trivial kernels based on edit distance are not definite positive. Neuhaus and Bunke [7] proposed several method to overcome this important drawback, however the proposed kernels are not explicitly based on the minimization problem addressed by kernel methods. Such a 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.

As denoted by Steinke [14], the term $f^t K^{-1} f$ in equation 1 may be considered as a regularization term which counter balance the fit to data term encoded by the 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. Such vectors may be considered as functions mapping each graph of the database to a real value. 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 [1] 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 defined by $W_{ij} = e^{-\frac{d(G_i, G_j)}{\sigma}}$, 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: $\Delta_{i,i} = \sum_{j=1}^{n} W_{i,j}$. Classical results from spectral graph theory [3] 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 [13] defines the regularized Laplacian $\tilde{l}$ of W as $\tilde{l} = I + \lambda l$ where λ is a regularization coefficient. The minimal eigen value of $\tilde{l}$ is equal to 1 and the matrix $\tilde{l}$ is thus definite positive. Moreover, given any vector f, we have :

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

Intuitively, minimising equation 2, 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 un normalized 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 $\tilde{L} = \Delta^{-\frac{1}{2}} \tilde{l} \Delta^{-\frac{1}{2}}$. 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 $I + \lambda \Delta^{-\frac{1}{2}} l \Delta^{-\frac{1}{2}}$. This new formulation is consistent with the regularization constraint which should be added to equation 1 and provides significant advantages in the context of incoming data (Section 2.1).

2.1 Incoming Data

Let us first consider a kernel defined from the un normalized 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 un normalized Laplacian l with this new graph and to compute the updated kernel defined as the inverse of the regularized and un normalized 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 un normalized 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 $B = (-\lambda exp(\frac{-d(G,G_i)}{\sigma}))_{i=\{1,\ldots,n\}}$ is deduced from the weights between the new input graph G and each graph $(G_i)_{i=\{1,\ldots,n\}}$ of our dataset and δ_n is a diagonal matrix with $(\delta_n)_{i,i} = B_i$.

The minimal eigen value 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:

$$K_{un} = (\tilde{l}_{n+1})^{-1} = \begin{pmatrix} \Gamma & \Theta \\ \Lambda & \Phi \end{pmatrix} \text{ with } \begin{cases} \Gamma = E^{-1} + \Phi E^{-1} B B^t E^{-1} \\ \Theta = -E^{-1} B \Phi \\ \Lambda = -\Phi B^t E^{-1} \\ \Phi = (1 - \sum_i B_i - B^t E^{-1} B)^{-1} \end{cases} \quad (3)$$

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

The computation of our new kernel, using equation 3, 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 the order K of $(I - \tilde{l}_n^{-1}\delta_n)^{-1}$:

$$(\tilde{l}_n - \delta_n)^{-1} = \tilde{l}_n^{-1}(I - \tilde{l}_n^{-1}\delta_n)^{-1} \approx \sum_{k=0}^{K} l_n^{-k-1}\delta_n^k \tag{4}$$

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 K greater than:

$$\frac{log(2\epsilon)}{\log(\max_{i=1,n} exp(\frac{-d(G,G_i)}{\sigma}))}. \tag{5}$$

Equation 4 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 may be achieved in KN^2.

If we now consider the regularized normalized Laplacian (Section 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:

$$K_{norm} = \Delta^{\frac{1}{2}}K_{un}\Delta^{\frac{1}{2}} \tag{6}$$

The update of the regularized and normalized Laplacian kernel may thus be deduced from the one of the regularized un normalized 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 2.1). 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. [12] describe a method to enumerate for any input unlabelled graph, all its connected subgraphs composed of up to 5 nodes. This efficient method provides up to 2048 patterns composed of connected subgraphs (called graphlets) of size lower or equal to 5. We propose here to adapt this method to the enumeration of sub-trees of acyclic unlabeled graphs up to size 6. The resulting patterns are called treelets (Fig. 1).

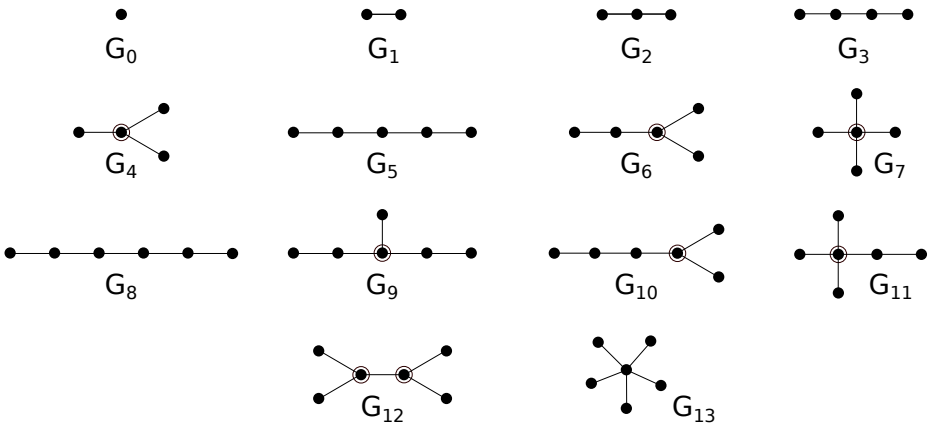
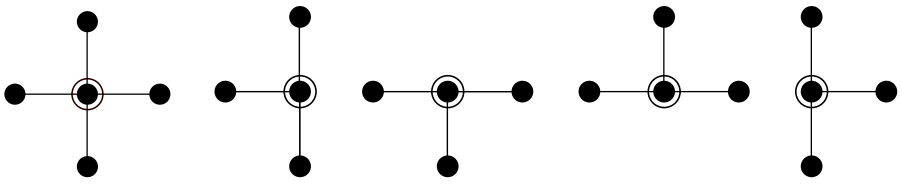


Fig. 1. Acyclic and unlabeled graphlets of maximum size equals to 6. Centers of 3-star and 4-star are surrounded

3.1 Computing Embedded Distribution

Following [12], our enumeration of all treelets starts by an enumeration of all paths with a length lower or equal to 6. A recursive depth first search with a max depth equals to 6 from each node of the graph is thus performed. 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. With this first step, the distribution of treelets G_0, G_1, G_2, G_3, G_5 and G_8 is computed (Fig. 1).

To compute the distribution of the remaining treelets, our method is based on the detection of nodes of degree 3 and 4. These nodes are respectively called R_{3-star} and R_{4-star} and are the center of the *3-star* and *4-star* treelets. Note that a 4-star treelet (G_7) contains four 3-star treelets (Fig. 2). This first degree analysis allows to compute the distribution of treelets G_4 and G_7. Treelets G_6, G_9, G_{10} and G_{12} are enumerated from the neighbourhood of 3-star treelets. For example, treelet G_6 requires a 3-star with at least one neighbour of R_{3-star} with a degree greater or equal to 2. Treelet G_{11} is the only sub tree derived from a 4-star. Properties characterizing treelets with a 3 or 4 star are summarized



(a) G_7 (b) G_4 decompositions of G_7

Fig. 2. G_7 contains 4 G_4.

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

Treelet	Source treelet	Condition
G_6	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	$\lvert\{v; v \in N(R_{3-star}); d(v) \geq 3\}\rvert \geq 1$

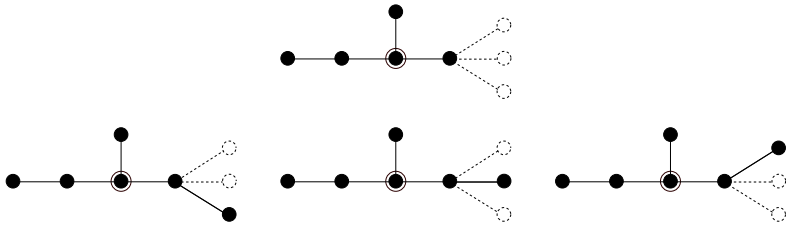


Fig. 3. Three permutations of G_9 sharing the same core.

in Table 1. Note that treelet 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 $\frac{1}{2}$.

Note that conditions summarized in Table 1 define necessary conditions for the existence of a given treelet centered around a 3 or 4 star. However, such conditions does not guarantee the uniqueness of such a treelet. Fig. 3 shows such an example: the rightest node of G_9 has a degree equals to 4 within the input graph whereas a degree greater or equal to 2 is required to define treelet G_9. Three different treelet G_9 may thus be built from the same five nodes. This configuration thus induces to count G_9 three times from the graph represented in Fig. 3(a) One may easily check that no isomorphism exists between treelets depicted in Fig. 1. Moreover, has shown by Read [10], the number of different alkanes composed of up to 6 carbons is equal to 13. Within our context, an alkane may be considered as an unlabeled acyclic graph whose vertex degree is bounded by 4. Therefore, treelet G_{13} which is the only treelet with a vertex of degree greater than 4 does not corresponds to a feasible alkane. The remaining 13 treelets in Fig. 1 represents, up to isomorphisms, all the unlabeled acyclic graphs whose size is lower than 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 6.When all treelets from a graph G have been enumerated, a vector representing treelet distribution is computed. Each component of this vector, denoted the *spectrum* of G, is equal to the frequency of a given treelet in G:

$$f_i(G) = \lvert(G_i \subset G)\rvert \tag{7}$$

Table 2. Comparison of addition methods.

Method	Classification Accuracy
KMean [15]	80% (55/68)
KWMean [4]	88% (60/68)
Trivial Similarity Kernel from Edit Distance [7]	90% (61/68)
Normalized Standard Graph Laplacian Kernel (Eq. 6)	90% (61/68)
Normalized Fast Graph Laplacian Kernel (Eq. 6)	90% (61/68)
Random Walk Kernel [16]	82% (56/68)

3.2 Definition of Treelet Kernel

A first idea to define a kernel from treelets consists to perform the inner product of vectors encoding the spectrum of graphs. Unfortunately, the inner product doesn't highlight spectrum similarities. For example, two graphs with nearly equal spectrum but with a low number of occurrences for each treelet are considered as less similar than two graphs having a same high number of treelet G_0 (ie same size) but a distribution of others treelets highly dissimilar. We thus use RBF kernels in order to better highlight differences between two spectra:

$$k_{Treelet}(G, G') = \sum_{k=0}^{N} e^{-\frac{(f_k(G)-f_k(G'))^2}{\sigma}} \tag{8}$$

where σ is a tuning variable used to weight the differences between treelet distribution and N is the number of enumerated treelets. Our kernel may thus be considered as a basic RBF kernel between two vectors and is hence definite positive.

4 Experiments

Our first experiment evaluates the graph Laplacian kernel on a classification problem. This problem is defined on the monoamine oxidase dataset(MAO)[1] which is composed of 68 molecules divided into two classes: 38 molecules inhibits the monoamine oxidase (antidepressant drugs) and 30 does not. These molecules are composed of different types of atoms with simple bonds 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 graph Laplacian kernel using approximate graph edit distance [11] with node substitution and edge deletion costs set to 1 and edge substitution cost set to the sum of incident node substitution costs. Graph Laplacian kernel methods obtain a classification accuracy of 90% which

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

corresponds to the highest score. Note that the other method obtaining 90% of classification accuracy is also based on the edit distance. This last kernel may however be non definite positive.

We may additionally notice that the use of our fast inversion method (Section 2.1) does not modify graph Laplacian kernel's classification accuracy (Table 2, lines 4 and 5). The number of iterations required by this fast inversion method is determined by equation 5. Our experiments performed on the MAO database show that a value of ϵ equal 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 2.1 is $0.273ms$ on the MAO database while this time is equal to $0.498ms$ using a direct inverse matrix computation. The ratio between both execution times is nevertheless about 1.8 hence showing a significant gain. Our treelet kernel is not tested against this database since this kernel is devoted to unlabeled graphs.

Our second experiment is based on a database of alkanes [2]. 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 described in [2] is composed of 150 molecules, associated to their respective boiling points. Using the same protocol than [2], 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.

Table 3 shows results obtained by different methods. Poor results obtained by graph Laplacian kernel can be explained by the lack of information when dealing with unlabeled graphs. Indeed, using such graphs, the heuristic used to approximate graph edit distance [11] maps the set of vertices of both graphs using uniquely the degree of vertices. Such a method thus consider several mappings as equivalent if several vertices with a same degree exist in both graphs. In this case, the sub optimal graph edit distance induces a poor graph discrimination. This lack of local information within unlabeled graphs also explains the poor results obtained by Kmean and random walk kernels. Indeed, these kernels are based on linear structures which are only discriminated by their lengths within unlabeled graphs. On the other hand, treelet kernel (with $\sigma = 0,25$) outperforms previous results of [2] based on neural networks combined with chemical descriptors.

Table 3. Boiling point prediction on alkane dataset.

Method	Average error (C)	Standard deviation (C)	Correlation
Neural Network [2]	3.11453	3.69993	0.9827
KMean [15]	4.65536	6.20788	0.9918
Random Walk Kernel [16]	10.6084	16.2799	0.9057
Graph Laplacian Kernel	10.7948	16.4484	0.9412
Treelet Kernel	1.40663	1.91695	0.9992

5 Conclusion

In this paper we proposed a graph Laplacian kernel based on a sub optimal graph edit distance combined with an efficient update of the kernel in order to predict relevant properties of incoming data. Our experiments show the efficiency of this kernel on databases composed of complex molecules with several hetero atoms. However, this kernel performs poorly on unlabeled graphs. We thus propose a new kernel based on treelet enumeration for unlabeled acyclic graphs. This kernel outperforms results obtained by state of the art methods on this dataset but remains restricted to unlabeled graphs. Our future work will be devoted to overcome this last limitation by extending treelet kernel to labeled graphs.

References

1. L. Brun, D. Conte, P. Foggia, M. Vento, and D. Villemin. Symbolic learning vs. graph kernels: An experimental comparison in a chemical application. *Proc. of the 1st Int. Workshop on Querying Graph Structured Data*, 2010.
2. D. Cherqaoui and D. Villemin. Use of a neural network to determine the boiling point of alkanes. *J. Chem. Soc. Faraday Trans.*, 90:97–102, 1994.
3. F.R.K. Chung. *Spectral graph theory*. AMSP, 1997.
4. F.-X. Dupé and L. Brun. Tree covering within a graph kernel framework for shape classification. In *ICIAP*, pages 278–287, 2009.
5. H. Kashima, K. Tsuda, and A. Inokuchi. *Kernels for graphs*, chapter 7, pages 155–170. MIT Press, 2004.
6. P. Mahé and J.-P. Vert. Graph kernels based on tree patterns for molecules. *Machine Learning*, 75(1):3–35, October 2008.
7. M. Neuhaus and H. Bunke. *Bridging the gap between graph edit distance and kernel machines*. World Scientific Pub Co Inc, 2007.
8. G. Poezevara, B. Cuissart, and B. Crémilleux. Discovering emerging graph patterns from chemicals. In *Proc. of the 18th ISMIS 2009*, pages 45–55, Prague, 2009. LNCS.
9. J. Ramon and T. Gärtner. Expressivity versus efficiency of graph kernels. In *1st Int. Workshop on Mining Graphs, Trees and Sequences*, pages 65–74, 2003.
10. R. Read. Some recent results in chemical enumeration. In Y. Alavi, D. Lick, and A. White, editors, *Graph Theory and Applications*, volume 303 of *Lecture Notes in Mathematics*, pages 243–259. Springer Berlin / Heidelberg, 1972.
11. K. Riesen and H. Bunke. Approximate graph edit distance computation by means of bipartite graph matching. *Image and Vision Computing*, 27(7):950–959, 2009.
12. N. Shervashidze, S. V.N. Vishwanathan, T. H. Petri, K. Mehlhorn, and K. M. Borgwardt. Efficient graphlet kernels for large graph comparison. In *Proceedings of AIStats*, pages 488–495, 2009.
13. A.J. Smola and R. Kondor. Kernels and regularization on graphs. In *Learning theory and Kernel machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop*, page 144. Springer Verlag, 2003.
14. F. Steinke and B. Schölkopf. Kernels, regularization and differential equations. *Pattern Recogn.*, 41:3271–3286, November 2008.
15. F. Suard, A. Rakotomamonjy, and A. Bensrhair. Kernel on bag of paths for measuring similarity of shapes. In *European Symposium on Artificial Neural Networks*, 2002.
16. S.V.N. Vishwanathan, K. M. Borgwardt, I. R. Kondor, and N. N. Schraudolph. Graph Kernels. *Neural Networks*, 2008.