Shape classification using a flexible graph kernel

François-Xavier Dupé[*] and Luc Brun

GREYC UMR CNRS 6072,
ENSICAEN-Université de Caen Basse-Normandie,
14050 Caen France,
{francois-xavier.dupe,luc.brun}@greyc.ensicaen.fr

Abstract. The medial axis being an homotopic transformation, the skeleton of a 2D shape corresponds to a planar graph having one face for each hole of the shape and one node for each junction or extremity of the branches. This graph is non simple since it can be composed of loops and multiple-edges. Within the shape comparison framework, such a graph is usually transformed into a simpler structure such as a tree or a simple graph hereby loosing major information about the shape. In this paper, we propose a graph kernel combining a kernel between bags of trails and a kernel between faces. The trails are defined within the original complex graph and the kernel between trails is enforced by an edition process. The kernel between bags of faces allows to put an emphasis on the holes of the shapes and hence on their genre. The resulting graph kernel is positive semi-definite on the graph domain.

Key words: Shape, Skeleton, Support Vector Machine, Graph Kernel

1 Introduction

The medial axis being an homotopic transformation, the skeleton of a 2D shape is a 2D structure with as many holes as the shape. A natural way to encode such a structure by a graph consists in creating an edge for each branch of the skeleton and a node for each junction of branches or branch's extremity. The resulting graph is a non simple planar graph which may be enriched using information from the radius of the osculating circle along branches [1–5]. The shape comparison is thus transformed into a graph comparison problem. However, graph comparison methods robust against structural noise such as the maximal common sub-graph method or the related graph edit distance problem [6] have an exponential complexity on general graphs. Many authors use thus a simpler encoding of the skeleton leading to a comparison function with a reduced complexity.

Siddiqi [1] and Sebastian [7] transform the graph into a tree and apply a tree comparison scheme. Another method, introduced by Pelillo [8], transforms

[*] This work is performed in close collaboration with the laboratory Cyceron and is supported by the CNRS and the Région Basse-Normandie.

graphs into trees and then models the tree matching problem as a maximal clique problem within a specific association graph. A last method proposed by Bai and Latecki [4] matches end points (vertices with a degree one) and then compares paths between the end-points. Contrary to the previous approaches, this last method can deal with closed structures and thus takes the holes of the shape into account.

Although these methods have been developed for indexation and classification tasks, they can not be readily used within the kernel machine framework. This limitation is related to the lack of mathematical tools inside the graph domain. Neuhaus and Bunke [9] proposed an elegant framework for the construction of graph kernels based on edit distances. Another solution consists in using graph kernels such as random walk or marginalized graph kernel [10] which are positive semi-definite on the graph domain. Though, these kernels are easier to use, they lack the flexibility and the noise robustness provided by the kernels based on graph edit distances.

This paper follows a first contribution [11] where we defined the notion of path rewriting within the graph kernel framework. However this method is defined on trees and thus does not encode properly the holes of the shapes. First, we recall some definitions and then extend our graph kernel framework to trails (Section 2). Second, we propose to extend the rewriting process, initially defined on trees, to graphs (Section 3). Then, we propose to combine our graph kernel with a closed paths kernel which compares graphs' faces (Section 4). Finally, an experiment with a multi-class classifier is proposed to highlight the relevance of holes inside holed shapes (Section 5).

2 Bag of trails kernel

Let us consider a *graph* $G = (V, E)$ where V denotes the set of vertices and $E \subset V \times V$ the set of edges. We define a *simple-graph* as a graph with no multiple edges between two vertices and no loop (an edge linking a vertex with itself). We define a *trail* as an alternating sequence of vertices and edges with distinct edges and a *path* as a trail with distinct vertices. A *closed path* is a path whose first vertex is equal to the last one. A *bag of trails* T associated to G is defined as a set of trails of G whose cardinality is denoted by $|T|$. We finally denote by K_{trail} a generic trail kernel.

2.1 Mean kernels

By considering bags as sets, Suard [3] has proposed several kernels for bags of paths which are extensible to trails. Amongst these kernels, the mean kernel is proposed as a convolution kernel [12] between trails: let T_1 and T_2 denote two bags of trails, the *mean kernel* between these two bags is defined as:

$$K_{mean}(T_1, T_2) = \frac{1}{|T_1|}\frac{1}{|T_2|}\sum_{t \in T_1}\sum_{t' \in T_2} K_{trail}(t, t'). \quad (1)$$

This kernel is positive definite on the bag of trails domain if and only if K_{trail} is positive definite on the trail domain.

The major drawback of this kernel is the information averaging when bags are composed of many trails. Such a loss of information may be avoided using a weighted mean kernel [13]. The design of this kernel assumes that most of the relevant information of a bag is located near its mean trail. Let T_1 and T_2 denote two bags of trails, then the *weighed mean kernel* is defined as:

$$K_{weighted}(T_1, T_2) = \frac{1}{|T_1|}\frac{1}{|T_2|}\sum_{t\in T_1}\sum_{t'\in T_2} < K_{trail}(t,m), K_{trail}(t',m') >^d \\ \frac{\omega(t)}{W}\frac{\omega(h')}{W'}K_{trail}(t,t'). \quad (2)$$

where $d \in \mathbb{R}^+$, m and m' denote the mean trails of T_1 and T_2, $\omega(t)$ (resp. $\omega(t')$) denotes the sum of the edge's weights of t (resp. t') and W (resp. W') the whole weight of the graph containing t (resp. t'). The trail kernel between a trail t and the mean trail m is defined as: $K_{trail}(t,m) = \frac{1}{|T|}\sum_{t_i\in T} K_{trail}(t,t_i)$. The weighted mean kernel is a convolution kernel based on a scalar product (the distances to the mean trail) and the trail kernel K_{trail}. So it is positive definite if and only if K_{trail} is positive definite.

2.2 A first trail kernel

For its marginalized kernel, Kashima proposed a walk kernel based on a tensor product [14]. As trails are particular walks, the walk kernel remains available. Let t and t' denote two trails, the *trail kernel* denoted $K_{classic}$ is defined as 0 if $|t| \neq |t'|$ and as follows otherwise:

$$K_{classic}(t,t') = K_v(\varphi(v_1), \varphi(v'_1))\prod_{i=2}^{|t|} K_e(\psi(e_{v_{i-1}v_i}), \psi(e_{v'_{i-1}v'_i}))K_v(\varphi(v_i), \varphi(v'_i)), \quad (3)$$

where $\varphi(v)$ and $\psi(e)$ denote respectively the vectors of features associated to the vertex v and the edge e. The terms K_v and K_e denote two kernels for respectively vertex's and edge's features. $K_{classic}$ is a tensor product kernel and so is positive definite if and only if K_e and K_v are two positive definite kernels. For the sake of flexibility and simplicity, we use Gaussian RBF kernels based on the distance between the attributes.

3 Edition kernel on trails

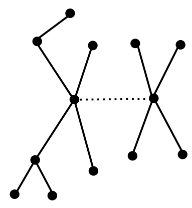
The main issue with skeleton based graphs is that two different graphs may encode similar shapes. Two different kind of structural noise may appear inside a skeleton: ligatures produced by noise on the boundary and elongations produced by a general deformation of the shape. Usually, this structural noise is tackled using edition operations on graphs. However, within a bag of trails framework we must consider edition operations on trails. The effect of the structural noise on a trail is twice: addition of edges and addition of vertices.

We suppose that the edges of our graph are associated to a weight which encodes their relevance. Torsello [15] has proposed such a relevancy measure: for each edge this measure approximates the length of the boundary associated to the skeleton's branch encoded by this edge. Using this weight, we compute the relevance of each vertex and edge inside a trail: the relevance of an edge corresponds to its weight and the relevance of a vertex corresponds to the weight of the sub-graph (e.g. the sum of the weight of all the sub-graph's edges) connected to the trail by this vertex. When graphs are trees [11], the sub-graphs correspond to sub-trees and the computation of the relevancy measure of vertices is unambiguous. Fig. 1a shows for example a path within a tree, where the sub-trees related to the two vertices of the path are clearly defined and so their weight.

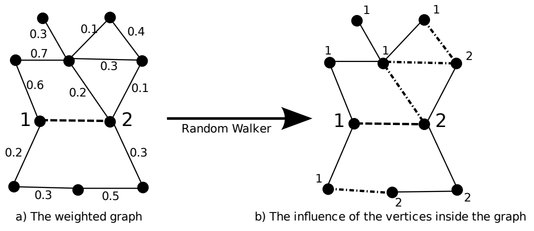
However, with holed shapes, graphs are not trees and the definition of the relevancy of vertices is not straightforward. Indeed, sub-graphs may connect several vertices of the considered trail. We propose to solve this difficulty by using the random walker diffusion algorithm [16] where normalized edge's weights are considered as transition probabilities. For each vertex v_i of the trail, this diffusion algorithm associates to each vertex v_l of the graph the probability $p_{l,i}$ that a random walker starting at v_l first reach v_i. Each vertex of the graph is then associated with the vertex of the trail with the maximal probability. The sub graph induced by this set of vertices is called the influence zone of the trail's vertex. However, the random walker is designed for simple-graphs. We thus transform our non-simple graph into a simple one by defining the transition probabilities between vertices as follows: loops are removed and multi-edges between two vertices are transformed into a single edge whose weight is the sum of edges' weights. Single edges between vertices are kept unchanged. Note that this transformation is only used for the random walker algorithm. Our trails and the sub-graphs encoding the influence zones are both defined within the initial non simple graph.

The weight of the influence zone of a vertex v is defined as the sum of 1) the weight of the edges within the influence zone and 2) a ratio of the weight of the edges shared with another influence zone (i.e. edges whose vertices belong to two influence zones). For example, the dash-dotted edges within Fig. 1b are shared by the two influence zones. Let v_1 and v_2 be the two incident vertices of an edge of weight w, v_1 (resp. v_2) is associated to its influence zone by a probability p_1 (resp. p_2) then we define as $\frac{p_1}{p_1+p_2}w$ (resp. $\frac{p_2}{p_1+p_2}w$) the part of the weight associated to the influence zone related to v_1 (resp. v_2). Fig. 1b shows an example of influence zone of trail vertices (the trail is defined by the dashed line): remark the importance of the influence zone of vertex 1 compared to the influence zone of vertex 2.

Given a relevancy measure of each vertex and edge of a trail we introduce two edition operations: vertex suppression followed by edge merging and edge contraction (or suppression for loops). The cost of an operation is defined as the relevancy measure of the removed edge or vertex. Finally, we defined an edition function κ which applies the cheapest edition. Then $\kappa^i(t)$ denotes the trail t after i editions. In addition, we denote by $\text{cost}_i(t)$ the cumulative cost of



(a) A path within a tree



(b) The random walker algorithm

Fig. 1. Influence zones: (a) Example of tree with a selected path (dotted edge) (b) The influence of the vertices of the dashed trail between the vertices 1 and 2 using the random walker.

operations leading to $\kappa^i(t)$. Finally, we construct the edition kernel as a weighted convolution kernel between the trails and their rewritings:

$$K_{edit}(t,t') = \frac{1}{D+1}\sum_{k=0}^{D}\sum_{l=0}^{D}\exp\left(-\frac{\text{cost}_k(t)+\text{cost}_l(t')}{2\sigma_{cost}^2}\right)K_{classic}(\kappa^k(t),\kappa^l(t')), \tag{4}$$

where D is the maximal number of editions and σ_{cost} the RBF parameter of the cost kernel which penalizes edition. This kernel is a convolution kernel [12] and is positive definite if and only if K_{trail} is positive definite.

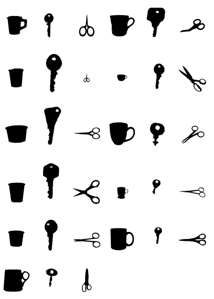
4 Closed paths kernel

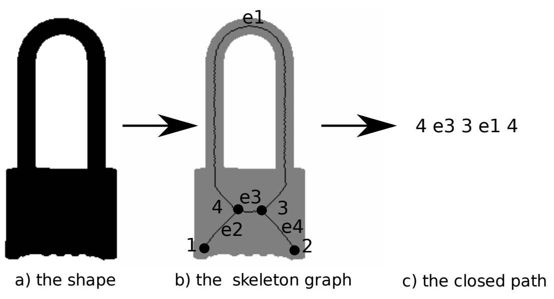
The faces of a skeletal graph encode the holes of a shape and represent as such important information about the shape. When using the previously defined kernels, faces are just encoded as trails. So when constructing a bag of trails, these particular trails may not appear in the bag or may be drowned with many other trails. Thus it is relevant to put an emphasis on faces when dealing with holed shapes.

Several kernels based on cycles have been proposed for graphs [17]. However these kernels are not designed for shape classification for two main reasons: they don't consider the orientation of faces and they are not restricted to cycles encoding faces.

An efficient comparison of faces within a shape recognition framework requires a kernel robust against structural noise. We propose to encode each hole by a unique closed path which describes the corresponding face. This path begins at the closest vertex to the gravity center of the shape and crosses the edges using a counter-clockwise orientation. For example, the hole of the padlock in Fig. 2b is described by the closed path "4 e3 3 e1 4". Finally, two closed paths encoding faces are simply compared using a trail kernel such as $K_{classic}$ (section 2.2) or K_{edit} (section 3).

However, while comparing two closed paths, we may have to face to alignment errors due to the selection of the initial vertex. In order to enforce the robustness of our kernel, shifted versions of the closed paths are also compared. For example,





a) Holed shapes databases b) Computation of the closed path of a padlock shape.

Fig. 2. Holed shapes and closed paths computation.

the face in Fig. 2b presents two vertices at an equal distance to the gravity center and the closed path ”3 e1 4 e3 3” is thus an acceptable path which corresponds to a shifted version of the previous path. We define the function $\mu_i(t)$ which performs a circular shift of i edges of the path t clockwise if i is positive and counter-clockwise i is negative.

The *shift kernel* is then defined as the weighted convolution between paths and their shifted versions using a trail kernel denoted K_{trail}:

$$K_{shift}(t,t') = \frac{1}{(2p+1)^2} \sum_{i=-p}^{p} \sum_{j=-p}^{p} \exp\left(-\frac{|i|+|j|}{2\sigma^2_{closed}}\right) K_{trail}(\mu_i(t), \mu_j(t')), \quad (5)$$

where p is the maximal number of shifts. This kernel is positive definite if and only if K_{trail} is positive definite. Finally, the closed paths kernel is defined as the mean kernel between all the closed paths surrounding the faces of two planar graphs G_1 and G_2:

$$K_{closed\ paths}(G_1, G_2) = \frac{1}{|C(G_1)|} \frac{1}{|C(G_2)|} \sum_{t \in C(G_1)} \sum_{t' \in C(G_2)} K_{shift}(t,t'), \quad (6)$$

where $C(G_1)$ (resp. $C(G_2)$) denotes the set of closed paths encoding the faces of G_1 (resp. G_2) and $|C(G_1)|$ (resp. $|C(G_2)|$) denotes the size of the set $C(G_1)$ (resp. $C(G_2)$). This kernel is positive definite if and only if K_{shift} is positive definite.

Finally, a kernel denoted $K_{combined}$ is built using the two proposed kernels:

$$K_{combined}(G_1, G_2) = (1-\gamma)K_{weighted}(T_1, T_2) + \gamma K_{closed\ paths}(G_1, G_2), \quad (7)$$

where T_1 (resp. T_2) is the bag of trails associated to G_1 (resp. G_2), $\gamma \in [0,1]$ is a tuning variable, $K_{weighted}(2)$ denotes our bag of trails kernel and $K_{closed\ paths}$ our closed paths kernel (6). This kernel is positive definite on the union of the bag of trails and bag of faces domains as it is defined as the addition of two positive definite kernels multiplied by positive coefficients [18].

	$K_{combined}$					$K_{weighted}$					Random Walk				
Classes	(1)	(2)	(3)	(4)	(5)	(1)	(2)	(3)	(4)	(5)	(1)	(2)	(3)	(4)	(5)
(1)	8	2	1			7	4				4	6			1
(2)		11					11				2	8	1		
(3)			11				2	9					10	1	
(4)				11					11					11	
(5)	1				10	1	3	2		5		2		1	8

Table 1. Confusion matrix on 5 classes of shapes: (1) Cups, (2) Keys, (3) Scissors, (4) Dudes and (5) Tools.

5 Experiments

We propose an experiment using a multi-class classifier [19]. The test database is built by adding shapes with holes (Fig. 2a) to the Kimia 99 shapes database [20]. Three kernels are used: the combination of the weighted mean kernel with the closed paths kernel denoted $K_{combined}(7)$, the weighted mean kernel alone denoted $K_{weighted}$ and the random walk kernel [10]. The trail kernel used within the weighted mean and the shift kernels (Section 4) is the edition kernel K_{edit} (Section 3).

For this experiment, the bags of trails (Section 2) were composed of 2 percent of the heaviest paths amongst all the trails with up to 9 edges. The maximal number of editions (Section 3) was set to 9 and the number of shifts (Section 4) to 5. For efficiency reason, the random walk [10] is performed on an augmented version of the maximal spanning tree: while considering an edge which is implied in the formation of a cycle or a loop, we change one of its incident vertices into a new vertex (of degree 1) with the same characteristics in order to break the cycle or loop. Using this trick, the graph may be encoded by an adjacency matrix and efficient random walk kernels [10] based on such an encoding may be used.

The experiment consists in the classification of the whole database into 5 classes (2 classes from the Kimia databases and 3 classes of holed shapes). The training set was composed of 5 shapes of each class taken arbitrarily. The classifier algorithm [19] is based on kernel principal analysis and quadratic discriminant analysis and so considers both inter-classes and intra-classes properties. Tab. 1 shows the confusions matrices of the three kernels. The $K_{combined}$ kernel shows very good results with some confusion on the cups. The $K_{weighted}$ kernel shows good results, but is very confused on tools. This confusion comes from the few trails contained inside the bag of trails which are not sufficient for a proper class separation. The random walk kernel shows good results too with confusion on tools and on cups. The confusion on the cups is due to the maximal spanning tree which conducts to a loss in the description of the faces of the graph.

6 Conclusion

We have defined in this paper a positive semi-definite kernel for shape classification which holds several properties: it is robust to noise and takes holes into account. The experiment shows the importance of a correct description of the graph and the underlying shape within the shape classification framework. In the future, we plan to further improve the bag of trails kernel on two points: the selection of the trails and the combination of the trail kernel results. These two points are indeed crucial as they directly influence the efficiency and the properties of the kernel.

References

1. Siddiqi, K., Shokoufandeh, A., Dickinson, S.J., Zucker, S.W.: Shock graphs and shape matching. Int. J. Comput. Vision **35**(1) (1999) 13–32
2. Ruberto, C.D.: Recognition of shapes by attributed skeletal graphs. Pattern Recognition **37**(1) (2004) 21–31
3. Suard, F., Rakotomamonjy, A., Bensrhair, A.: Kernel on bag of paths for measuring similarity of shapes. In: European Symposium on Artificial Neural Networks, Bruges-Belgique (April 2007)
4. Bai, X., Latecki, J.: Path Similarity Skeleton Graph Matching. IEEE PAMI **30**(7) (2008)
5. Goh, W.B.: Strategies for shape matching using skeletons. Computer Vision and Image Understanding **110** (2008) 326–345
6. Bunke, H.: On a relation between graph edit distance and maximum common subgraph. Pattern Recognition Letters **18**(8) (1997) 689–694
7. Sebastian, T., Klein, P., Kimia, B.: Recognition of shapes by editing their shock graphs. IEEE Trans. on PAMI **26**(5) (2004) 550–571
8. Pelillo, M., Siddiqi, K., Zucker, S.: Matching hierarchical structures using association graphs. IEEE Trans. on PAMI **21**(11) (Nov 1999) 1105–1120
9. Neuhaus, M., Bunke, H.: Edit-distance based kernel for structural pattern classification. Pattern Recognition **39** (2006) 1852–1863
10. Vishwanathan, S., Borgwardt, K.M., Kondor, I.R., Schraudolph, N.N.: Graph kernels. Journal of Machine Learning Research **9** (2008) 1–37
11. Dupé, F.X., Brun, L.: Edition within a graph kernel framework for shape recognition. In: GBR 2009. (2009) accepted.
12. Haussler, D.: Convolution kernels on discrete structures. Technical report, Department of Computer Science, University of California at Santa Cruz (1999)
13. Dupé, F.X., Brun, L.: Tree covering within a graph kernel framework for shape classification. In: ICIAP 2009. (2009) submitted.
14. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernel between labeled graphs. In: In Proc. of the Twentieth International conference on machine Learning. (2003)
15. Torsello, A., Hancock, E.R.: A skeletal measure of 2d shape similarity. CVIU **95** (2004) 1–29
16. Grady, L.: Random Walks for Image Segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on **28**(11) (2006) 1768–1783
17. Horváth, T.: Cyclic pattern kernels revisited. In: PAKDD, Springer (2005) 791–801
18. Berg, C., Christensen, J.P.R., Ressel, P.: Harmonic Analysis on Semigroups. Springer-Verlag (1984)
19. Wang, J., Plataniotis, K., Lu, J., Venetsanopoulos, A.: Kernel quadratic discriminant for small sample size problem. Pattern Recognition **41**(5) (2008) 1528–1538
20. LEMS: shapes databases. http://www.lems.brown.edu/vision/software/