Pattern Recognition Letters 49 (2014) 177–184

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Leaf margins as sequences: A structural approach to leaf identification [☆,☆☆]

Guillaume Cerutti [a,b,*], Laure Tougne [a,b], Didier Coquin [c], Antoine Vacavant [d]

[a] *Université de Lyon, CNRS, France*
[b] *Université Lyon 2, LIRIS, UMR5205, F-69676, France*
[c] *LISTIC, Domaine Universitaire, F-74944 Annecy le Vieux, France*
[d] *Clermont Université, Université d'Auvergne, ISIT, F-63001 Clermont-Ferrand, France*

A R T I C L E   I N F O

Article history:
Received 3 December 2013
Available online 1 August 2014

Keywords:
Leaf recognition
Multivariate sequences
Edit distance
Classification

A B S T R A C T

In the context of an automated leaf identification process, the use of thorough leaf margin descriptors is essential given the importance of this criterion in the determination of the species. The spatial properties of teeth along the leaf contour are something to keep track of, which is made possible through the use of structured representations. This paper introduces a sequence representation of leaf margins where teeth are viewed as symbols of a multivariate real valued alphabet. It presents the methods developed to make use of this description for classification and implementation in a mobile tree identifying application. The results of various classification methods are compared and discussed, both in terms of species recognition and of consistency with botanical concepts.

1. Introduction

The determination of plant species from field observation requires a necessary amount of botanical expertise, which puts it beyond the reach of most nature enthusiasts. The conception of a mobile identification application is a way to transmit botanical knowledge, through an explanatory leaf identification process, focusing on the discriminant shape criteria listed (with the associated obscure vocabulary) in flora. Such a process relies on a description of shapes that makes a semantic interpretation possible.

The use of curvature-based contour description is a very efficient tool for shape recognition. In the special case of tree leaves, the shape of the leaf margin is a common and highly discriminant criterion for the identification of species. Those margins show structures easily representable by curvature, but with various spatial properties of distribution or frequency, a dimension that most contour features would overlook. The method we propose is dedicated to keeping track of this spatial information by the introduction of a sequence-like structured representation, light enough to build species models that are embeddable in a smartphone application.

After a review of existing approaches for shape representation in Section 2, we present in Section 3 the methods we use to build the proposed representation and to manipulate it. Section 4 expounds the classification system based on our contour description, and the results and comparisons in terms of species identification and shape recognition are detailed in Section 5.

2. Related work

2.1. Contour description

A very rich representation of a contour is the Curvature-Scale Space (CSS) [20,19], obtained by piling up measures of curvature on a contour over increasing smoothing scales. It has been used for shape retrieval and even for leaf classification [18,3]. In this context, only the zero-crossings of the curvature (the CSS contours) are considered for description, leaving aside the actual values of the curvature. Locating teeth on the leaf margin is very close to detecting dominant points on a close curve, a well-studied problem [24] for which the use of the curvature-scale space proves to be of great interest [21].

Curvature constitutes anyway a very informative source of information on contour shapes, but it is often used for building aggregative features (HoCS [14], number of teeth [10,1], and average properties [10]) that completely overlook the spatial repartition of structures on the contour, like many other geometrical contour descriptors used for leaves (Angle Code Histogram

☆ This paper has been recommended for acceptance by A. Koleshnikov.

☆☆ This work has been supported by the French National Agency for Research with the reference ANR-10-CORD-005 (ReVeS Project).

* Corresponding author at: LIRIS, Université Lyon 2, 69676 Bron, France. Tel.: +33 4 78 77 23 51.

E-mail address: guillaume.cerutti@liris.cnrs.fr (G. Cerutti).

http://dx.doi.org/10.1016/j.patrec.2014.07.016
0167-8655/© 2014 Elsevier B.V. All rights reserved.

[27,28], Directional Fragment Histogram [9]). On the other hand, approaches keeping track of the spatiality do not provide a full description of the properties of the structures [18] and rely on a computationally expensive matching process [2,3].

Another contrasting approach consists in retaining all the contour information through the use of simple generic descriptions such as the Centroid-Contour Distance curve (CCD) [26], where shapes are straightforwardly converted into sequences that can be aligned for comparison [12]. Such direct representations of a contour, though being perfectly faithful to the shape they describe, do not select in their definition the *a priori* relevant information concerning the considered objects.

2.2. Sequence representations

Among structured representations, sequences, initially strings of characters, have the simplest structure and are hence the easiest to process. They can be compared through the computation of edit distances, based on the Levenshtein distance [15], efficiently determined along with optimal edition paths through dynamic programming [25], distances that can be normalized [17,16] for more genericity.

Real-valued sequences are more common in time series, and tools for comparison and alignment borrow concepts from string edit distances. The most usual is the Dynamic Time Warping distance [22], that can be modified to satisfy triangle inequality [6], or used outside of a temporal context, for shape indexing for instance [12].

Concerning sets of sequences, many approaches try to find groups and patterns in string data, through clustering [23] or Self-Organizing Maps [13]. There also exist methods to estimate the distribution of sequences in a set, finding the set median (the element of the set minimizing the sum of edit distances to all the other strings in the set), estimating the generalized median [7], or even the deviation of the set [11].

3. Sequences for leaf margins

The objects we study are segmented leaves, in other words binary images describing a contour whose properties we want to characterize. In addition to this input, we also have a polygonal model at hand, representing the global shape of the leaf, that we use for prior segmentation [5]. In particular, it provides a way of knowing in advance the main symmetry axis of the leaf, which is a very useful source of information for interpreting the contour.

3.1. Leaf contour understanding

The idea beneath the definition of a margin shape descriptor is that it should make use of prior knowledge on what is discriminant in the context of species determination, a knowledge that derives from botany. Using this knowledge implies considering the object as a leaf, and producing a high-level interpretation of its shape enables the consideration of more specific descriptions, which would not be possible without any grasp of what kind of shape is being processed.

Botany tells us that the dentition of the leaves is something that defines the species. In other terms, the size, sharpness, orientation, frequency and regularity of teeth along the margin, as a human eye would qualify them, constitute a discriminant feature providing clues for identification, independently of the other shapes of the leaf.

Consequently, to simulate this characterization, we explicitly detect every tooth and pit on the leaf contour, leaving apart the base and apex areas that cannot be seen as actual teeth. The extraction of these dominant points is based on an analysis of the CSS transform of the contour described in [4]. The result is a visually interpretable representation of the contour where the basal and apical points are precisely defined, and where the teeth are located described by their properties in terms of size (scale) and sharpness (curvature).

One major advantage of landmarking the leaf contour in such a way is to implicitly provide some robustness to rotation. Once the base and apex are located, they can be used as reference points and any following processing will be done in this framework, skipping the otherwise compulsory aligning phase. Fig. 1 shows the stability of the contour interpretation to small rotations, guaranteed by the use of rotation independent information such as curvature in the positioning of well-defined leaf landmarks. Contour interpretation would also benefit any other non-aggregative descriptor (CCD, shape context) where providing a rotationwise stable origin point would potentially simplify the matching procedure.

The extraction and characterization of tooth points along the contour provides a condensed look on the discriminant information of a leaf margin. It allows the building of a simple and light representation that will make comparisons easier than with the complete contour signal, while focusing on elements on interest. It makes it possible to look at a higher level and build contour models of the species at a small computation and memory cost.

In our case, given a contour parametrized by its normalized arclength u, what we obtain is a set $D = \{u_i\}_{i=1...|D|}$ of dominant points, described through a CSS analysis by curvature κ and scale s. We also use the axis formed by the located base and apex points A and B to compute a vertical position π as the relative position of a point's projection on the segment $[BA]$ (0 corresponding to B and 1 to A). The resulting object is the set $P = \{P_i = (u_i^\star(u_i, u_B, u_A); \kappa_i; s_i; \pi_i), u_i \in D\}$.

To describe the margin, we separate the points in P into two sets P_L and P_R, one for each side of the leaf. Then, knowing the positions u_B and u_A of the base and apex points, it is possible to compute a relative arclength $u^\star$ for the two sides so that the two sets of points are on comparable scales. This results in the building of two parallel series of points as shown in Fig. 2 (a), where teeth and pits, with their characterization, spread from the base to the apex, with the same reference measure $u^\star$.

These series can be seen as sequences, illustrated by Fig. 2 (b), where the symbols are not letters of a discrete alphabet as in strings, but rather vectors $(\kappa; s; \pi)$ describing teeth in a 3-dimensional space. Defined this way, margin sequences become objects that represent accurately the profile of the contour, including strong spatiality information.

3.2. Manipulating margin sequences

In order to process such objects in a recognition framework, we want to be able to perform basic operations such as comparisons and averaging, which are necessary steps to consider margin

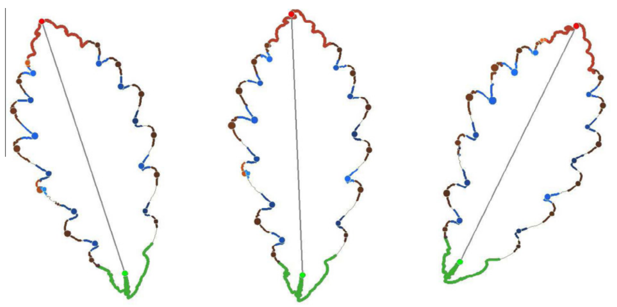
Fig. 1. Leaf contours obtained on rotated images, with located base area (🟩) and apex area (🟥), and detected teeth (▬) and pits (▬)).

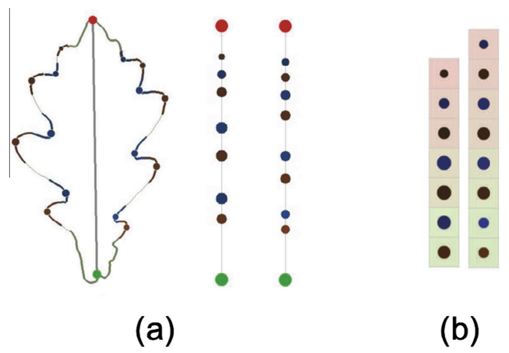

Fig. 2. Series of points obtained on both sides of a leaf margin (a) and their representations as sequences (b) with size, curvature (positive curvature 🟧 and negative curvature 🟦) and vertical position (base 🟩 apex).

strings at the same level as other vectorial descriptors. Among structural representations, sequences are the most simply defined and the easiest to manipulate, using common tools derived from string processing.

The first tool we use to process margin sequences is the edit distance, defined as a normalized weighted Levensthein distance adapted to the case of a vectorial alphabet, close to the edit distance with real penalty [6]. Considering two sequences $P_1 = (P_{1,i})_{i=1\dots|P1|}$ and $P_2 = (P_{2,j})_{j=1\dots|P2|}$ we define the three elementary operation costs necessary to the computation of the optimal edition path, the insertion and deletion costs depending on the size of the element (we denote as λ the null element and empty string), and the substitution on a weighted distance between the two vectors:

$$c(P_{1,i} \to \lambda) = (s_{1,i})^{\frac{1}{2}}, \quad c(\lambda \to P_{2,j}) = (s_{2,j})^{\frac{1}{2}},$$
$$c(P_{1,i} \to P_{2,j}) = \left(\sum_{x \in \kappa,s,\pi} \omega_x (x_{1,i} - x_{2,j})^2\right)^{\frac{1}{2}} \tag{1}$$

The edit distance $d^\star(P_1, P_2)$ is then simply the cost of the optimal edition path $O^\star(P_1, P_2)$, obtained through dynamic programming. To ensure that we obtain comparable distances, we adapted an existing method to normalize this distance [16] in order to keep it between 0 and 1 and achieve a greater genericity with regard to the length of the compared sequences. The final distance term we use is based on the maximal distance between two sequences (as the cost of the path obtained by deleting one sequence and inserting the other) and can then be written as:

$$d_{edit}(P_1, P_2) = \frac{2 \times d^\star(P_1, P_2)}{d^\star(P_1, P_2) + d_{max}(P_1, P_2)}, \quad d_{max}(P_1, P_2)$$
$$= \sum_{i=1}^{|P1|} c(P_{1,i} \to \lambda) + \sum_{j=1}^{|P2|} c(\lambda \to P_{2,j}) \tag{2}$$

To make leaf margin sequences convenient objects, we also need to be able to average some of them. One first step towards averaging is to compute a median of two margin sequences P_1 and P_2, for instance to end up with one sequence per leaf, a way to produce one convenient description as shown in Fig. 4. To do so, we simply compute the optimal edition path $O^f(P_1, P_2)$ and consider only substitution operations. The average sequence $\mathrm{med}(P_1, P_2)$ will then be the sequence of the middle points of the substituted vectors:

$$\mathrm{med}(P_1, P_2) = \left(\frac{P_{1,i} + P_{2,j}}{2}\right)_{(P_{1,i} \to P_{2,j}) \in O^\star(P_1, P_2)} \tag{3}$$

For a larger set of sequences $S = \{P_s\}_{s=1\dots M}$ a way to derive a representative object is to estimate its generalized median $\mathrm{med}(S)$. It is defined as the sequence of the space Ω, formed by any vectors, that minimizes the sum of edit distances to all the elements of the set:

$$\mathrm{med}(S) = \underset{P \in \Omega}{\mathrm{argmin}} \left(\sum_{P_s \in S} d_{edit}(P, P_s)\right) \tag{4}$$

The generalized median estimation algorithm we use is derived from the heuristic introduced in [7] for strings, and considers as a starting point the set median $\mathrm{med}_S(S)$, the element of the set minimizing the same sum of distances:

$$P_{\mathrm{med}} \leftarrow \mathrm{med}_S(S) = \underset{P_1 \in S}{\mathrm{argmin}} \left(\sum_{P_2 \in S} d_{edit}(P_1, P_2)\right) \tag{5}$$

The idea is then to iteratively update the current sequence to get closer to the generalized median. This is done by computing the edition paths of the current estimated median to every sequence in the set, and selecting at each position of P_{med} the most frequent operation. The modifications that occur most often indicate the direction to follow to reduce the sum of edit distances, as the subsequent suppression of the operations from the edition paths will reduce their costs.

Contrary to strings where possible operations, as the alphabet, form a discrete set, we can not expect to encounter the exact same operation more than once with real valued vectors. However, to decide which operation is the most represented, we need a way to group operations. Consequently, for each position of the current estimated median, we chose to count operations along five types:

- Deletion of the symbol $P_{\mathrm{med},i}$:
$$D_{S,i} = \left\{(P_{\mathrm{med},i} \to \lambda) \in O_s^\star, s = 1 \dots M\right\}$$
- Substitution of $P_{\mathrm{med},i}$ by a symbol of same curvature:
$$S_{S,i}^+ = \left\{(P_{\mathrm{med},i} \to P_{s,j}) \in O_s^\star, s = 1 \dots M | \kappa_{\mathrm{med},i} \times \kappa_{s,j} > 0\right\}$$
- Substitution of $P_{\mathrm{med},i}$ by a symbol of opposite curvature:
$$S_{S,i}^- = \left\{(P_{\mathrm{med},i} \to P_{s,j}) \in O_s^\star, s = 1 \dots M | \kappa_{\mathrm{med},i} \times \kappa_{s,j} < 0\right\}$$
- Insertion of a symbol of same curvature after $P_{\mathrm{med},i}$:
$$I_{S,i}^+ = \left\{(\lambda \to P_{s,j}) \in O_s^\star, s = 1 \dots M | (P_{\mathrm{med},i} \to P_{s,j-1}) \in O_s^\star \wedge \kappa_{\mathrm{med},i} \times \kappa_{s,j} > 0\right\}$$
- Insertion of a symbol of opposite curvature after $P_{\mathrm{med},i}$:
$$I_{S,i}^- = \left\{(\lambda \to P_{s,j}) \in O_s^\star, s = 1 \dots M | (P_{\mathrm{med},i} \to P_{s,j-1}) \in O_s^\star \wedge \kappa_{\mathrm{med},i} \times \kappa_{s,j} < 0\right\}$$

Among the three first sets for the position i, only the one with most votes is selected, as it will determine the fate of the vector $P_{\mathrm{med},i}$. Insertions, on the other hand, may occur in addition to the transformation of $P_{\mathrm{med},i}$, and they should be taken into account whenever they occur in more that half of the cases (*i.e.* if $|I_{S,i}^+| > \frac{M}{2}$ for instance). The retained operations at each position (deletion/substitution and potential insertions) define an average edition path O_S that is built by lines up the selected operations, but with their average symbol as second operand. For instance if substitution is kept at position i, the average edition path will contain the operation:

$$O_{S,i} \leftarrow \left(P_{\mathrm{med},i} \to \frac{1}{|S_{S,i}^+|} \sum_{(P_{\mathrm{med},i} \to P_{s,j}) \in S_{S,i}^+} P_{s,j}\right)$$

This path O_S formed by the sequence of these average operations is then applied to the current median estimation to obtain

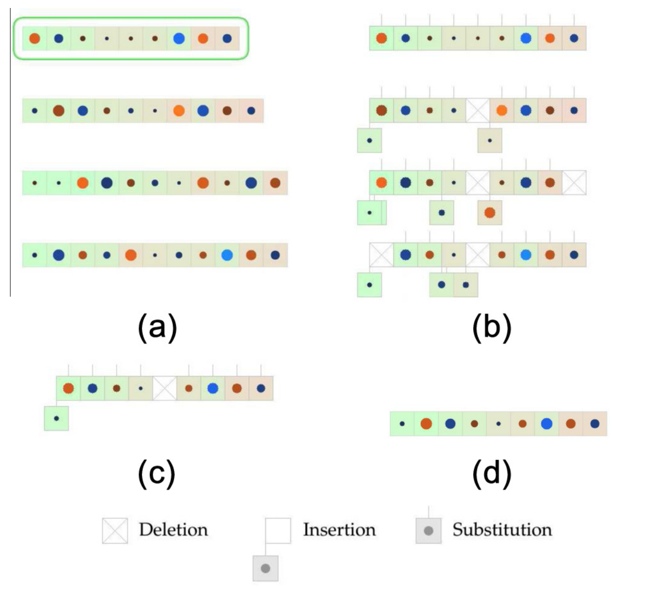

Fig. 3. One step of the generalized median estimation algorithm: set of sequences (a) and computation of the set median (marked in green), edition paths of the current estimated median to the sequences of the set (b), average edition path obtained from these paths (c), and re-estimated median (d). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

a new P_{med}. If changes have been made, this whole process is repeated, until the estimated edition path to apply to P_{med} is simply $\{(P_{\text{med},i} \rightarrow P_{\text{med},i})\}_{i=1\ldots N_{\text{med}}}$ and no further transformation can be found. The final sequence $P^{\star}_{\text{med}}$ corresponds then to the estimated generalized median of the sequence set $\text{med}(S)$.

The Fig. 3 illustrates a first iteration of this process. Among the 4 sequences of the set, the set median is selected (a) as a first estimation of the median. The operation paths transforming this sequence into all the sequences of the set are then computed (b). By counting the operations at each position, we build an average path (c). For instance in the first position, there are 3 substitutions and 1 deletion, and it is the substitution (by the average symbol of the 3) that will be selected. This path is finally applied to the current estimated median to produce its new estimation (d) on which the process will be repeated.

Intuitively, the successive selection of the most common transformations used to reach the sequences of the set ensures that the overall edit distance is reduced (the operations at the next step will necessarily be less costly for most sequences of S) and the estimation process converges towards a good estimation of the generalized median of the set S. This notion of approximate median sequence proves very handy, since it condenses a set of sequences into one unique representative, which makes comparisons easier to carry out. We dispose now of tools to compare and establish similarities between margin sequences, and to build models after a certain number of them, a way to perform generalization from individual data.

4. Margin sequence classification

4.1. Sequences for description

The main application of the string processing tools will be to use the leaf margin sequences as convenient descriptors for the identification of tree species. The first step is to sum up the information extracted on a leaf margin into one single descriptor, which we perform by computing the estimated median of the two sequences P_L and P_R representing the left and right margin:

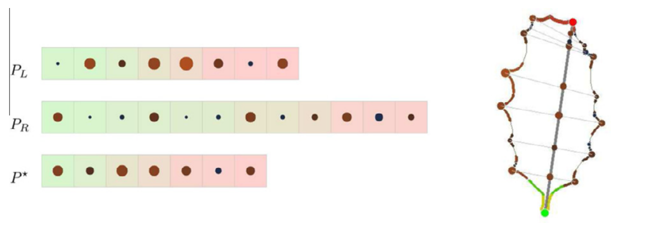

Fig. 4. Averaging two margin sequences and matching points to create a single leaf margin descriptor.

$$P^{\star} = \text{med}(P_L, P_R)$$

What is actually performed by this averaging process based on substitution operations is a 1-to-1 matching of the teeth and pits detected on each side, so that only the structures that can be associated with a similar structure on the other side are kept in the description. This alignment process, as well as the averaging of symbols in the median sequence, is a way to smooth the imperfections in the segmentation. An example of such matching is given in Fig. 4.

This unique margin descriptor constitutes first a visually explicit synthesis of the teeth and pits encountered on the margin of the leaf, independently of the other shape criteria of the leaf. It is already interpretable at a high level, while being a processing-ready object, through the tools introduced earlier. It allows also an almost direct conversion into a curve, if we translate the points into convex and concave parts, and might then be used as an interesting visualization tool for an automatic reconstruction of the analyzed shape.

4.2. Species identification

The primary use of margin sequences is to perform species recognition. Considering leaf identification as a supervised classification problem, the goal is to use a labeled training database $\mathcal{S}$ to classify new examples into classes formed by species $\{E_e\}_{e=1\ldots N}$. Considering margin sequences, the base will be constituted of sequences associated with a class index ϵ : $\mathcal{S} = \{(P_n; E_{\epsilon(n)})\}$. For each species identified by the index e, we can isolate a subset of sequences $S_e = \{P_n | \epsilon(n) = e\}$.

Among the possible choices to associate an example with a label, the simplest is a nearest neighbour (NN) classification that requires matching an example with all the sequences in $\mathcal{S}$. The decision is then made by assigning the class of the closest example (1NN), or the most represented among the k best matches (kNN). These approaches can be generalized to produce a ranking of species from the order of the matches. This is a basic yet effective option, but it requires a great deal of computation, growing with the size of the database, especially given the quadratic complexity of the edit distance.

To avoid that much computation on objects that are less easily processed than vectorial data, a solution to limit the number of necessary comparisons would be to keep only one model per species. The most obvious approach is then to consider as the best candidate to represent the species E_e the estimated median sequence computed on the set S_e. That is the solution we chose, since its lightness in computation time is definitely appealing, as well as its ability to synthesize a large amount of information into one model, that is still interpretable at a high level:

$$P_e = \text{med}(S_e) = \text{med}(\{P_n | \epsilon(n) = e\})$$

The species models we obtain through this computation are sequences of symbols which are much more regular and synoptic than the sequences obtained on one given leaf, as show the sequences provided on the Fig. 5. The estimation of the median sequence provides then a very interesting generalization, at least from a visual point of view, as what appears is a visibly faithful summary of the properties of the leaf margin for a species, a canonical margin that gives a good idea of the shape generally presented by a species.

Given a new example, the ranking of classes is then simply performed by comparing the example P with the models accounting for each and every species, i.e. by computing for all the species E_e the edit distances $d_{edit}(P, P_e)$. The species will then simply be ordered by ascending order of the distances. In the case when the distances are equal, we assume the species to be *ex-quo*; this is justified by the fact that the margin alone is not enough to identify a species, and the margin sequences are destined to be combined with vectorial descriptors accounting for the global shape of the leaf, and its basal and apical shapes [5].

One may argue that a simple canonical model is not enough to cover the diversity of shapes taken by a species, and there is actually a risk that some individuals of a species will lie quite far from this ideal representative. Capturing variability is difficult for structural approaches, and even if there are ways to characterize the distribution of individuals in a set of sequences [11], taking it into account to perform better comparisons is a complex issue. Here lies probably the most significant limit of structural methods.

To overcome this difficulty, one solution is to go back more or less implicitly in a vectorial space, where classes can be more easily modeled, and where usual classification algorithms can be applied. This is what is made possible by sequence kernels. A kernel is a semi-definite positive function that associates a pair of elements to a scalar value that can be expressed as an inner product. By its definition, the distance $d_{edit}(\cdot, \cdot)$ constitutes a kernel on the domain of margin sequences, and its value for every pair of sequences $d_{edit}(P_n, P_{n'})$ can be known. Based on these values it is possible to use classification algorithms using only inner-product-like terms, such as SVM, without the need of making explicit the vectorial formulation from which ensues the inner product. Sequence kernels are a way of grasping the distribution of structured elements in a set through their relationships to one another, that might be very useful in the context of species identification.

5. Results & discussion

To validate our structured approach and measure its relevance, we performed large scale tests over a subset of the Pl@ntLeaves 2012 database [8] comprising the Scan and Pseudo-scan images of the 88 simple-leaved Angiosperm species out of the 126 of the database. It consists of 5340 plain background images (in order to limit the influence of the segmentation method on the results)

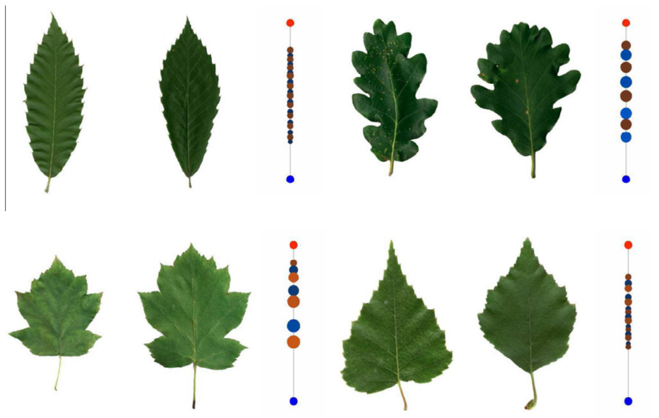
Fig. 5. Species estimated median margin sequences for two tree species, computed on the Pl@ntLeaves 2012 database [8].

forming 88 unevenly distributed classes, some species being much more represented than others. The tests included comparing our leaf description to existing shape descriptors to establish its competitiveness.

5.1. Compared methods

We wanted to locate the performance of our method describing leaf margins by sequences among state-of-the-art descriptors. However, it is complicated to find descriptors measuring the same information as we do, i.e. only the margin information decorrelated from the global shape of the leaf and from other local shapes. Consequently we tried to include contour-based descriptors that are comparable to ours but with a generally less limited scope, and compared them with both our margin description and adding the full leaf description [5]. The implemented methods were the following:

- Basic curvature measures **[CURV]**: simply average and standard deviation values of the curvature computed over a contour at 4 different scales; this is assumed to be the rawest measure of the contour information.
- Curvature-scale parameters **[CSSP]** [4]: our previous method, parameters describing the teeth and pits on the margin, computed from the points detected on the contour using the CSS, and characterizing the size, sharpness and distribution of teeth on the margin.
- Histograms of curvature over scale **[HoCS]** [14]: state-of-the-art contour description, built from the CSS by creating histograms of curvature values (21 bins) over different scales (25 values); an aggregative yet quite complete description of what is present on a contour.
- Inner-Distance Shape Contexts **[IDSC]** [2]: heavy shape description computed by sampling the contour (20 sample points) and building at all sample points log-radial histograms of the other points viewed by paths remaining inside the shape; an extensive and expensive shape representation.
- Contour bags of visual words **[CBoW]**: computed using SURF keypoints and descriptors extracted on the binary segmented images; visual vocabulary of size 1000 learned by clustering, each image then being represented by a fixed-size histogram of its words from the vocabulary.

To these approaches we add the two methods we propose, based on the structured representation designed to explicitly capture the specificities of leaf margins:

- Leaf margin sequences **[LMS]**: the proposed descriptor here considered alone to represent the leaf; obviously this is not sufficient as margin is not fully representative of the species, but it may be interesting to see where it is located among the other methods.
- Full leaf description **[Folia]**: the proposed descriptor for leaf margins, to which we add vectors describing the global shape, basal shape and apical shape of the leaf [5] to take all these decorrelated aspects in the process of identification; this is the method used in the iPhone application Folia.[1]

5.2. Classification results

Most of the selected methods were designed for matching individuals rather that training a classifier, which is why we decided to measure the performances of both of those rather incompatible

[1] https://itunes.apple.com/app/folia/id547650203.

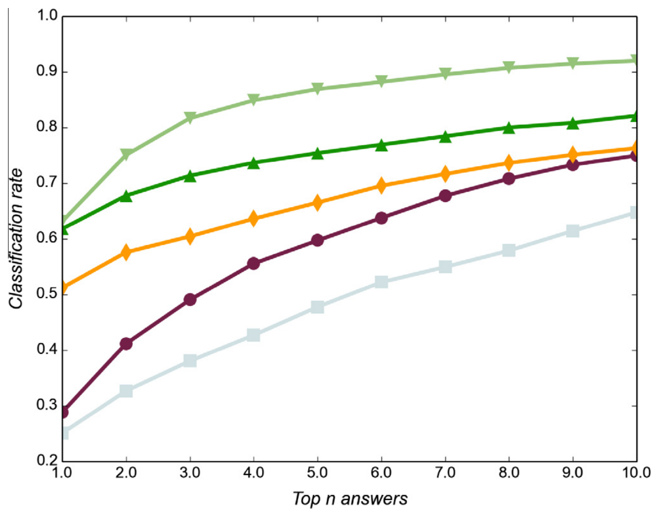

Fig. 6. Classification results for the contour descriptors: ● **[CURV]** (Gaussian), ♦ **[CSSP]** (Gaussian), ▲ **[LMS]** (Median), ■ **[LMS]** (K-SVM) and ▼ **[LMS]** (Matching).

approaches. To keep things simple and comparable, we adopted the same classification approaches in all the cases. For matching, we performed a nearest neighbour classification, whereby the species are ranked as the appear in the ranking of individuals, with the specificity that when two species share the same distance or similarity measure, they are assigned the same rank.

For classification, we first used a naive classification algorithm. We considered all data as vectorial and built Gaussian models to represent the classes, excepted for margin sequences where classes are represented by only estimated median sequences. In this simple model case, we separate histogram-like data for which it is actually just an average, and distances are computed as euclidean distances that gave the best results among other possibilities (chi-square, intersection). When the data was initially under the form of vectors, we used a so-called ellipsoid surface distance [5], designed to take variability into account. For descriptors needing alignment before comparison (**[IDSC]** notably), the best alignment was selected when building the class models.

We also used the edit distances between margin sequences as a kernel to train a multi-class SVM (**[LMS] (K-SVM)**) and provide a way to take better account of the variability of the classes. The classifier actually consists of a set of 1vs1 SVMs making a decision for any pair of classes, and the final classification is based on a number of such binary votes received by each class.

To measure performance, we split the dataset into a training set containing 2/3 of images, and a validation set containing the remaining 1/3. Only the first set is used for matching and for the training of the classifiers, and the recognition performance is measured on the second one. Rather than one raw good classification rate, we compute cumulative rank histograms, corresponding to the rates of examples for which the correct species is among the *n* first answers, *n* ranging from 1 to 10.

The first comparison provided in Fig. 6 concerns only the margin descriptors introduced by us, and demonstrates beyond possible doubt that the spatial representation **[LMS]** carries much more information than the summarized description of teeth properties **[CSSP]** and obviously than the very basic curvature measures **[CURV]**. It appears that a significant proportion of information is lost in the averaging process and by the fact that it overlooks variability, as it is shown by the performance of the nearest neighbour matching. Still, there is a factor of 50 between the computation time of both approaches, which is enough, along with the overall good performance to justify choosing the lighter path.

These results are somewhat biased by an aspect of the method we use. With **[LMS]**, as well as with **[CSSP]**, absolutely all of the leaf margins for which no teeth were detected are sent on the same point of the descriptor space (an empty sequence in the case of margin sequences). This is acceptable as long as the distance information is kept. Since the leaves actually present the same margin shape, no information is lost by saying that the distance to all the species with an entire margin is the same, as any confusion will be resolved by the other descriptors. This is however a major problem for classifiers that give an answer in terms of votes only, and lose track of the indeterminacy a measure of distance to classes provides. The fact that a decision on the class is forced in the definition of the 1vs1 SVMs explains why the kernel SVM classification performs so poorly, as entire margins will lead to classification errors instead of class indifferenciation.

To refine this analysis, we perform the same classification on a restriction of the training set to species that are not supposed to present such flat, untoothed margins. They represent 60 of the 88 initial species, and 2850 images to process. The removed species being allegedly the easiest regarding the margin, the performance can be expected to drop even if the number of classes is lower. The results of this experiment are given in Fig. 7.

The first consequence is that K-SVM is now as competitive as the other methods, when the classes all sent to a single point are suppressed. It equals the species median sequences, which tends to prove that the fact that variability is taken into account does not compensate the flattening caused by the projection on the kernel. This fact is underlined by the performance of **[CSSP]**, which deals with the variability of classes and ranks at the same level as the rigid median sequence, but much lower than the raw sequence data, that still conveys a richer information since the matching performance of **[CSSP]** remains lower than with **[LMS]**.

Concerning other methods that were originally designed to constitute stand-alone descriptions of the leaf, it is more pertinent to compare them not only with the margin descriptor but also with the full set of shape descriptors we introduced **[Folia]**. We used the same approach with all methods, building a single model per species, and classifying a new example according to the distances to each of the models. For greater genericity, we returned to the full dataset with entire species, since a decision will be made this time through the use of other descriptors. The performances we obtained are given in the Fig. 8.

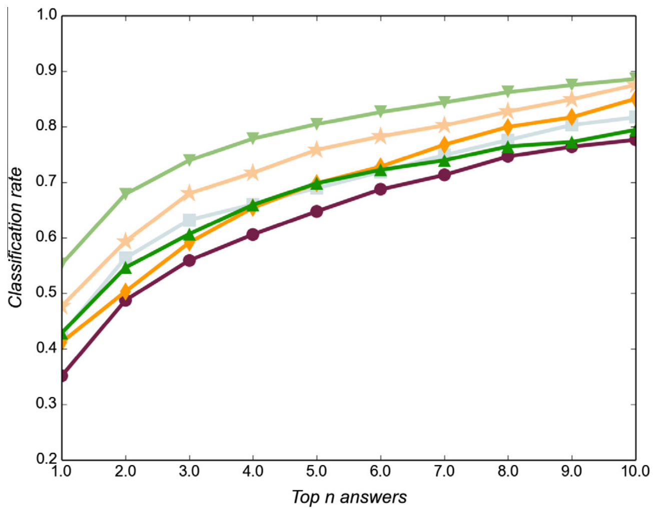

Fig. 7. Classification results for the contour descriptors on non-entire species: ● **[CURV]** (Gaussian), ♦ **[CSSP]** (Gaussian), ★ **[CSSP]** (Matching), ▲ **[LMS]** (Median), ■ **[LMS]** (K-SVM) and ▼ **[LMS]** (Matching).

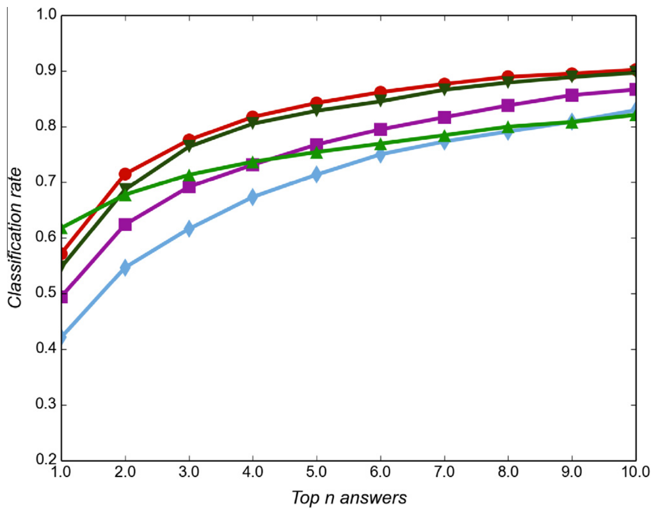

Fig. 8. Classification results for complete the leaf descriptors: ▲ **[LMS]** (Median), ▼ **[Folia]** (Median + Gaussian), ♦ **[CBoW]** (Average), ■ **[HoCS]** (Average) and ● **[IDSC]** (Aligned Average).

Table 1
Comparison of classification time for one example and space requirements of the different methods

Method	Classif. Time	Memory Space
[HoCS] (Average)	0.27 ms	450 Kb
[IDSC] (Average)	15.53 ms	1400 Kb
[CSSP] (Gaussian)	0.36 ms	90 Kb
[LMS] (Median)	8.71 ms	18 Kb
[Folia] (Median + G)	9.38 ms	170 Kb
[HoCS] (Matching)	111.42 ms	35000 Kb
[IDSC] (Matching)	653.61 ms	115000 Kb
[LMS] (Matching)	438.22 ms	1700 Kb

The results show that our compact description, consisting of light vectors (of size $9+5+5$) plus a sequence like structure, slightly outperforms larger histogram-like features such as bags-of-words (size 1000) and HoCS (size 21×25) and competes with the extensive description that is shape contexts (size 20×64 and alignment required). It justifies the use of an explicit high-level description of leaves since it performs comparably to state-of-the-art methods while providing at the same time an interpretable representation of what is captured from the image.

To complete this analysis, we measured the performance in terms of execution time and memory requirements of the different approaches, as displayed in Table 1. The tests were performed on a 2.7 GHz Intel Core i7 processor, with 4Go RAM.

Given our goal of developing a fully on-device mobile application, and the factor 10 that generally can be measured between the execution times on computer and smartphone, any kind of matching process would be irrelevant. Our method offers then the best compromise between performance and lightness of execution, giving the correct the species among the first five answers in more than 80% of cases, with a classification time under 10 ms.

6. Conclusions

The approach presented in this paper introduces a very explicit structured representation as a way to describe and synthesize the information present on a leaf margin, while decorrelating it from the other shapes of a leaf. The simple and efficient tools proposed to work with this description allow to consider it as a fundamental part of a species recognition process.

Combined with a compact vectorial description of a leaf's global and local shapes, it proves to convey as much information for classification as state-of-the-art shape descriptors, generally consisting of a heavier definition. The classification performance, along with the high-level visual representation it provides, have led us to include this method in our tree species identification application for iPhone 1 with satisfying results.

The main limitation of this structural approach is the difficulty of taking into account the variability of classes. This results in certain drawbacks regarding classification algorithms, which have difficulties in including both the spatial dimension and the heterogeneity of the species in their processing. A solution would be to use string processing tools to create a spatialized representation of variability that would allow a better understanding of this aspect, but it stills constitutes a challenge for structured representations.

However, using sequences for leaf margins proves to be a very effective way of keeping track of all the discriminant information on a leaf contour, and might open way to a semantic interpretation of the margin shape. Such a possibility could be very interesting in the context of an interactive educational application, as a powerful tool not only to identify tree species but more importantly to transmit a recognition skill necessary for a further exploration of botany.

References

[1] A. Arora, A. Gupta, N. Bagmar, S. Mishra, A. Bhattacharya, A plant identification system using shape and morphological features on segmented leaflets, in: CLEF, 2012.
[2] P. Belhumeur, D. Chen, S. Feiner, D. Jacobs, W. Kress, H. Ling, I. Lopez, R. Ramamoorthi, S. Sheorey, S. White, L. Zhang, Searching the world's herbaria: a system for visual identification of plant species, in: ECCV, 2008.
[3] C. Caballero, M.C. Aranda, Plant species identification using leaf image retrieval, in: ACM CVIR, 2010, pp. 327–334.
[4] G. Cerutti, L. Tougne, D. Coquin, A. Vacavant, Curvature-scale-based contour understanding for leaf margin shape recognition and species identification, in: VISAPP, 2013a.
[5] G. Cerutti, L. Tougne, J. Mille, A. Vacavant, D. Coquin, Understanding leaves in natural images a model-based approach for tree species identification, CVIU 117 (2013) 1482–1501.
[6] L. Chen, R. Ng, On the marriage of lp-norms and edit distance, in: Proceedings of the 30th International Conference on Very Large Data Bases, 2004, pp. 792–803.
[7] I. Fischer, A. Zell, String averages and self-organizing maps for strings, in: Proceedings of the neural computation, 2000, pp. 208–215.
[8] H. Goëau, P. Bonnet, A. Joly, I. Yahiaoui, D. Barthelemy, N. Boujemaa, J.F. Molino, The ImageCLEF 2012 Plant Identification Task, 2012.
[9] H. Goëau, A. Joly, I. Yahiaoui, P. Bonnet, E. Mouysset, Participation of INRIA & Pl@ntNet to ImageCLEF 2011 plant images classification task, in: CLEF, 2011.
[10] C. Im, H. Nishida, T.L. Kunii, A hierarchical method of recognizing plant species by leaf shapes, in: IAPR MVA, 1998, pp. 158–161.
[11] J.M. Jolion, The deviation of a set of strings, Pattern Anal. Appl. 6 (2003) 224–231.
[12] E. Keogh, L. Wei, X. Xi, M. Vlachos, S.H. Lee, P. Protopapas, Supporting exact indexing of arbitrarily rotated shapes and periodic time series under euclidean and warping distance measures, VLDB J. 18 (2009) 611–630.
[13] T. Kohonen, P. Somervuo, Self-organizing maps of symbol strings, Neurocomputing 21 (1998) 19–30.
[14] N. Kumar, P. Belhumeur, A. Biswas, D. Jacobs, W. Kress, I. Lopez, J.V.B. Soares, Leafsnap: a computer vision system for automatic plant species identification, in: ECCV, 2012, pp. 502–516.
[15] V.I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, Sov. Phys. Dokl. 10 (1966) 707–710.
[16] Y. Li, B. Liu, A normalized levenshtein distance metric, PAMI 29 (2007) 1091–1095.
[17] A. Marzal, E. Vidal, Computation of normalized edit distance and applications, PAMI 15 (1993) 926–932.
[18] F. Mokhtarian, S. Abbasi, Matching shapes with self-intersections: application to leaf classification, IP 13 (2004) 653–661.
[19] F. Mokhtarian, A. Mackworth, A theory of multiscale, curvature-based shape representation for planar curves, PAMI 14 (1992) 789–805.
[20] F. Mokhtarian, A.K. Mackworth, Scale-based description and recognition of planar curves and two-dimensional shapes, PAMI 8 (1986) 34–43.

[21] S.C. Pei, C.N. Lin, The detection of dominant points on digital curves by scale-space filtering, Pattern Recognit. 25 (1992) 1307–1314.

[22] H. Sakoe, S. Chiba, A dynamic programming approach to continuous speech recognition, in: Proceedings of the 7th International Congress on Acoustics, 1971, pp. 65–69.

[23] M. Sebban, A.M. Landraud, String clustering and statistical validation of clusters, in: Advances in AI, AI'98, vol. 1418, 1998, pp. 298–309.

[24] C.H. Teh, R.T. Chin, On the detection of dominant points on digital curves, PAMI 11 (1989) 859–872.

[25] R.A. Wagner, M.J. Fischer, The string-to-string correction problem, J. ACM 21 (1974) 168–173.

[26] Z. Wang, Z. Chi, D. Feng, Q. Wang, Leaf image retrieval with shape features, in: Advances in Visual Information Systems of Lecture Notes in Computer Science, vol. 1929, 2000, pp. 41–52.

[27] Z. Wang, Z. Chi, D. Feng, Q. Wang, Shape based leaf image retrieval, IEEE Proc. Vision Image Signal Process. 150 (2003) 34–43.

[28] B.A. Yanikoglu, E. Aptoula, C. Tirkaz, Sabanci-okan system at imageclef 2012: combining features and classifiers for plant identification, 2012.