A Similarity-Based Approach for Shape Classification using Aslan Skeletons

Aykut Erdem [a,*], Sibel Tari [a],

[a] *Department of Computer Engineering, Middle East Technical University, TR-06531, Ankara, Turkey*

Abstract

Shape skeletons are commonly used in generic shape recognition as they capture part hierarchy, providing a structural representation of shapes. However, their potential for shape classification has not been investigated much. In this study, we present a similarity based approach for classifying 2D shapes based on their Aslan skeletons [1, 2]. The coarse structure of this skeleton representation allows us to represent each shape category in the form of a reduced set of prototypical trees, offering an alternative solution to the problem of selecting the best representative examples. The ensemble of these category prototypes is then used to form a similarity based representation space in which the similarities between a given shape and the prototypes are computed using a tree edit distance algorithm, and Support Vector Machine (SVM) classifiers are used to predict the category membership of the shape based on computed similarities.

Key words: shape skeletons, shape classification, similarity-based pattern recognition

* Corresponding author. Phone: +90 312 266 4577
Email addresses: `aykut@ceng.metu.edu.tr` (Aykut Erdem),

1 Introduction

Classification or *categorization* is among the most primary cognitive processes, described as the ability to group a very large or possibly infinite number of similar objects into a relatively small number of classes or categories; and to identify a novel object as a member of a particular class. Having a classification mechanism is vital from information retrieval perspective because organizing knowledge in a structured way offers efficient and economical use of limited resources when reasoning about the novel object. From the early days, object classification remains one of the biggest challenges for the computer vision community. Visual recognition and classification of objects require learning mechanisms which combine visual information with prior knowledge and experience. Compared to other visual cues, *e.g.* color, texture and spatial information, shape information is generally considered as a strong cue since a given object can be classified by its shape. In computer vision, shape classification, in itself, is a difficult problem as the shapes of objects present in nature exhibit a great variability due to not only geometric similarity transformations (i.e. translation, rotation, and scaling) but also visual transformations such as occlusion, deformation and articulation.

Based on how category knowledge is represented, classification studies in the literature can be grouped as *classical*, *prototype* and *exemplar* approaches [53]:

In *the classical approach*, it is believed that every category is constructed from a set of *essential features*, each of which is necessary and sufficient in defining that particular category (*e.g.* a bird category can be defined by the properties

stari@metu.edu.tr (Sibel Tari).

`can fly`, `has feathers`, `has wings` and `can sing`). According to this line of thinking, the boundaries of categories are all well-defined that a novel object is a member of a category if and only if it satisfies all the characteristics of that category. Hence, there is no notion of membership rating. Here, the major challenge is the process of feature selection which can be resolved only if one has domain knowledge about the problem at hand. However, in most of the real-world problems, gathering this information might be impossible.

In *the prototype approach*, the concept of prototype plays the central role in classification where each category is a fuzzy set that is constructed as a grouping of objects similar to the prototype which can be considered as either a summary representation formed by abstracting over previously encountered examples or just a typical example of the category [42]. For example, when instances of the bird category are represented with n-dimensional feature vectors, one can naively form the prototype by averaging over the collection of feature vectors, or the bird category can be represented only with a robin or a sparrow. When a novel object is encountered, classification is performed by comparing it with the prototypes of categories. The instance is then assigned to the most similar category, if the corresponding similarity is found to be greater than a threshold value.

In *the exemplar approach*, each category is represented not by a single prototype, but by a collection of exemplars referring to memory traces of some previously encountered examples of the category. Accordingly, classification depends on the similarities to the stored exemplars. For example, in classifying a novel object, for each category, one sums up all the similarities between the novel object and the exemplars of the category. Subsequently, the object is assigned to the category having the greatest cumulative similarity value.

The main difference between prototype and exemplar approaches is in how they define classification, *i.e.* whether classification relies on an abstraction over or a function of previously encountered objects [43]. In fact, these two approaches can be considered as the two extremes in a continuum. When the most typical exemplar of a category defines the category, the exemplar-based model becomes equivalent to a prototype-based one. Similarly, the exemplars defining a category might not refer to actual copies of the encountered examples but also to some kind of abstraction.

The *prototype* and *exemplar* approaches offer an important paradigm shift by requiring nothing but an ability to compute similarity, hence the issue of feature selection is replaced with the issue of defining similarity [13, 14, 16, 18–21, 23–25, 27, 29, 30, 33–40]. In literature, there are two major models of similarity. *Geometric models of similarity* treat objects as points in a multidimensional metric space where the similarity between two objects is inversely related to the distance between their representations in this space [49]. *Feature-based models of similarity* is defined as a function of common and distinguishing features [59].

In this paper, we address the task of classifying 2D shapes. Here, a critical issue is the representation, *i.e.* how to express a 2D shape. Existing representation schemes can be mainly classified into two categories: *boundary-based* and *skeleton-based*. In boundary-based representations, shapes are either expressed by a set of boundary points or by a set of boundary curves (or some features extracted from them) [10, 11, 22, 31, 32, 55]. Skeleton-based approaches, on the other hand, aims at capturing a structural representation of shape by modeling the shape in terms of a set of axial curves [2, 5, 12, 23, 28, 51, 57, 61, 63]. As skeleton-based representations can describe the spatial relationships among shape parts, they implicitly introduce insensitivity to articulation and deformation.

The full potential of shape skeletons has not been fully explored in the context of shape classification, despite their popularity in shape matching literature [1, 4, 9, 23, 35, 46, 52, 63]. Needless to say, any shape matching method can be utilized in an exemplar-based classification framework, *e.g.* by performing nearest neighbor search by matching the query shape against all the database shapes [11, 31, 45, 62]. However, in general, such approaches require a large number of training examples if the intra-class variation is large which makes classifying a query shape very time-consuming. Furthermore, selecting the best representative examples of a category is an open problem in itself. For example, in an exemplar-based approach [45] which uses shape skeletons, the authors avoided this complex problem by picking the exemplars of each category manually.

Rarity of skeleton-based shape classification studies in the literature can be linked to the practice that shape skeletons are represented by graphs or trees, and the classification tools which handle such graphical structures are not as widespread as those which use feature vectors since there is no clear definition of key statistical concepts such as mean or median of a set of graphical structures [13].

Obtaining an effective vectorial representation of a graphical structure is not that straightforward. One proposed approach is to encode the topological structure of graphs into low dimensional vector spaces, which have found its application in *skeleton-based indexing* [18, 50]. However, this is not a good solution for classification since the central idea is only to eliminate unrelated comparisons in a retrieval task.

In [8], Bai *et al.* present a feature-based approach for skeleton-based shape classification. The authors utilize the skeletonization method presented in [5]. After extracting the shape skeleton, they represent the shape with the set of shortest skeleton paths between any two endpoints of skeleton branches. The classification framework is derived from a Bayesian formulation adopted from [56]. The method is efficient as shape-to-shape matchings are eliminated. However, the level of abstraction is low since the approach does not involve learning class models from the training set. Recently, in [6], the authors have extended their framework in a way that the skeleton features are considered in conjunction with the contour features.

Recently in [17], Daliri and Torre propose a feature-based approach, which at first glance seems similar to our approach in the sense that SVM classifiers together with tree-edit distance based kernel are used (except that they use boundary-fragments to obtain a symbolic representation of shape). However, closer inspection reveals that the way they form the representation space is totally different as they use a fixed dictionary of strings that do not represent actual classes.

Also, recently in [15], Chen *et al.* present an interesting matching-based classification method. Unlike the traditional approach, *i.e.* learning classifiers after obtaining matching results, they propose a unified framework which utilizes classification errors in learning class-specific parameters for matching.

Our classification algorithm can be regarded as a matching-based approach but not an exemplar-based method. We first construct category prototypes from the examples and learn class-specific distance measures. We then utilize the similarity information between the query shape and the category prototypes formed using the training examples. A pictorial view of our two-layered classification scheme is given in Figure 1. The input to the system is *Aslan skeleton* [1,2] of the given query shape. This particular skeleton representation, which will be briefly reviewed in Section 2, brings two important advantages. First, the extracted skeletons are quite stable as the method only retains numerically accurate features. Second, it allows us to form rooted depth-1 skeletal trees, and by virtue of which we devise a mechanism to compute similarity between a shape and a prototype representing a collection of shapes. The first layer of our classification system utilizes this mechanism to obtain a similarity based representation of the query shape which is simply a vector of similarities between the query shape and the set of M prototypes. We remark that each prototype is an abstraction over a disjoint set of shapes belonging to the same shape category. Hence, their number might exceed the number of existing categories, N. The second layer consists of N SVM classifiers whose outputs are used to compute a membership score for each shape category. In this way, the similarities which are initially calculated using common skeletal features are in a sense modified by taking into account the overall pairwise relationships among all the shape categories in the database. Here, it is also critical that the initially selected feature set needs to be as stable as possible (rather than being highly discriminative) so that the instabilities will not propagate through the calculations.

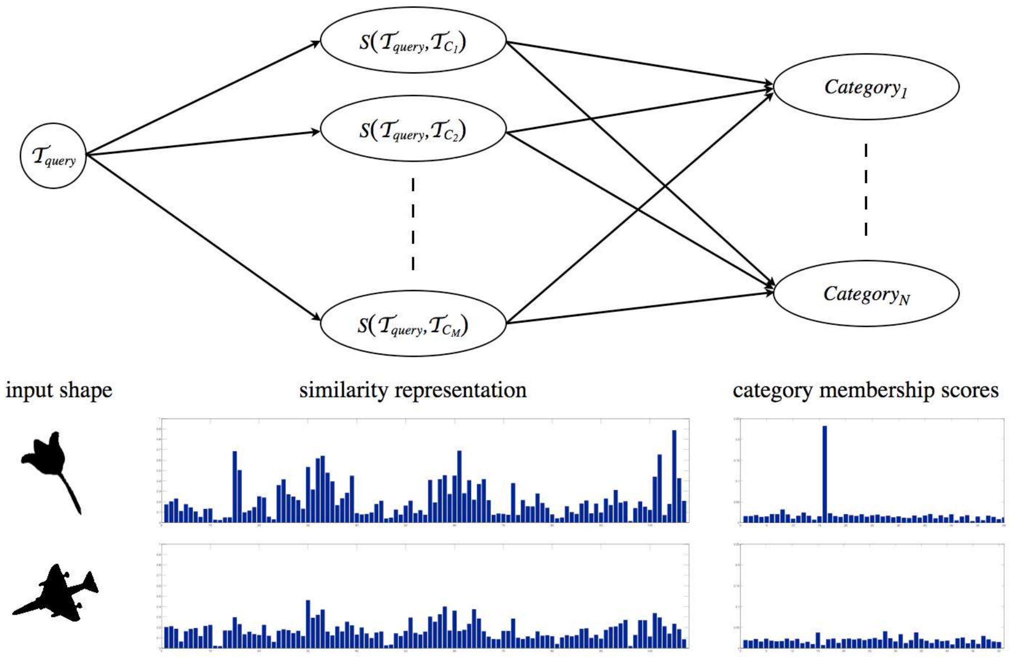
$\mathcal{T}_{query}$
$S(\mathcal{T}_{query},\mathcal{T}_{C_1})$
$S(\mathcal{T}_{query},\mathcal{T}_{C_2})$
$S(\mathcal{T}_{query},\mathcal{T}_{C_M})$
$Category_1$
$Category_N$
input shape
similarity representation
category membership scores

Fig. 1. Two-layered structure of the proposed classification scheme.

2 Aslan Skeleton

Shape skeletons have been successfully used for shape recognition [1,4,9,23,35, 46,52,63]. The strength of these representation schemes is their insensitivity to visual transformations such as articulation and deformation as a result of their ability to capture hierarchy of parts. Unfortunately, the skeleton descriptions of two visually similar shapes, when expressed as a graph, may be topologically different which poses an instability to which there are three possible solutions. The first is to post-process the skeletons by pruning problematic branches [3, 5, 28, 47, 61]. The second is to handle the problem during the matching step by utilizing error-tolerant matching algorithms [9,23,46,52,63]. The third [2] is an unorthodox approach of extracting only the numerically accurate skeleton branches by introducing excessive amount of regularization into the skeletonization process. Even though the skeletons extracted using [2] are too coarse and might not allow a detailed shape analysis (*e.g.* for analyzing brain structures where every detail in the shape boundary may be of prime importance), the representation is well suited for generic shape recognition [1,9]. Recently, new types of pruning-based methods have appeared in literature and these approaches compute coarse skeletons as well. In [5], skeleton pruning is performed in parallel with discrete curve evolution and this method have been successfully used for shape recognition [4] and classification [8] and object detection [7]. Another appealing idea is presented in [61] where pruning process takes into account global context of a set of skeletons. In addition to these new pruning-based methods, the skeletonization method in [28] employs a physics-based deformable model where the coarseness of extracted skeletons depends on a single parameter. Lastly, the approach in [35] is also of our interest since it offers a simplified graphical representation of a shape skeleton by analyzing ligature regions.

The process of extracting Aslan skeleton of a shape starts with computing a special distance function, the level curves of which are increasingly smoothed versions of the initial shape boundary. The surface has just a single extremum point which captures the center of a blob-like representation of the shape (Figure 2(a)). Once the distance function is computed for a given shape, skeleton branches are extracted and subsequently classified as either *positive* or *negative* by using the method in [57] (Figure 2(b)). In the resulting skeleton, every positive branch originates from a positive curvature maxima of the shape boundary, thus represents a protrusion of the shape. On the other hand, every negative branch emanates from either a negative curvature minima or a positive curvature minima, thus each negative branch generally represents an indentation. Moreover, shapes with n-fold symmetry at their centers always have n positive and n negative branches that survive and reach the corre-

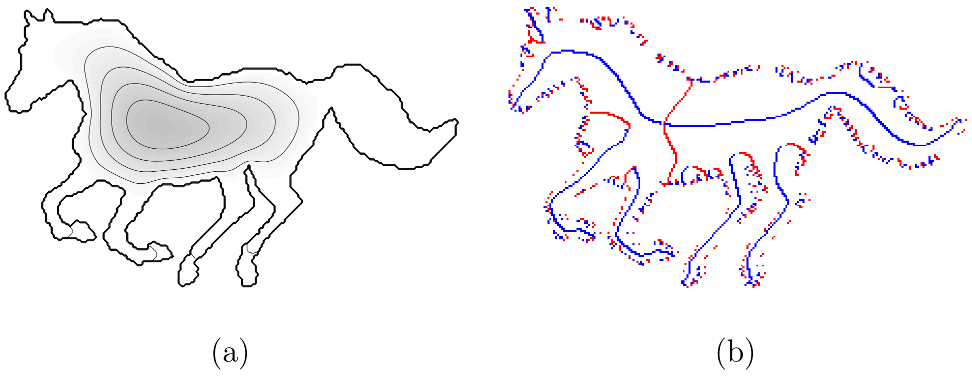
(a) (b)

Fig. 2. (a) Level curves of the distance function used for computing the skeleton. (b) Extracted skeleton branches using the method in [57]. Notice that each positive branch merges with a negative branch at some disconnection point, except the major ones. Positive and negative skeleton branches are respectively shown in blue and red.

sponding shape center, and these are referred to as *major branches* since they represent the most prominent features of the shape. The remaining pairs of positive and negative branches all meet at some *disconnection points*, terminating before reaching the shape center.

In the final representation, unimportant very short branches which are close to the shape boundary are pruned and the termination concept is artificially extended to the major positive branches for stability reasons discussed in [1,2]. The relative organization of the branches is then expressed in terms of the location of their disconnection points on a coordinate frame which is formed using the major negative branches. Note that the coordinate frame can be constructed in a number of ways depending on the choice of major branch, thus all these descriptions are stored. The skeletal attributes used to represent each skeleton branch are simply its type (*negative* or *positive*), the location of its disconnection point in polar coordinates (r, θ), and its length l measured in the formed coordinate frame. The representation appears to be coarse, but it has been shown that the spatial arrangement of disconnection points is quite robust under articulation of parts and non-rigid deformations [1,2]. One

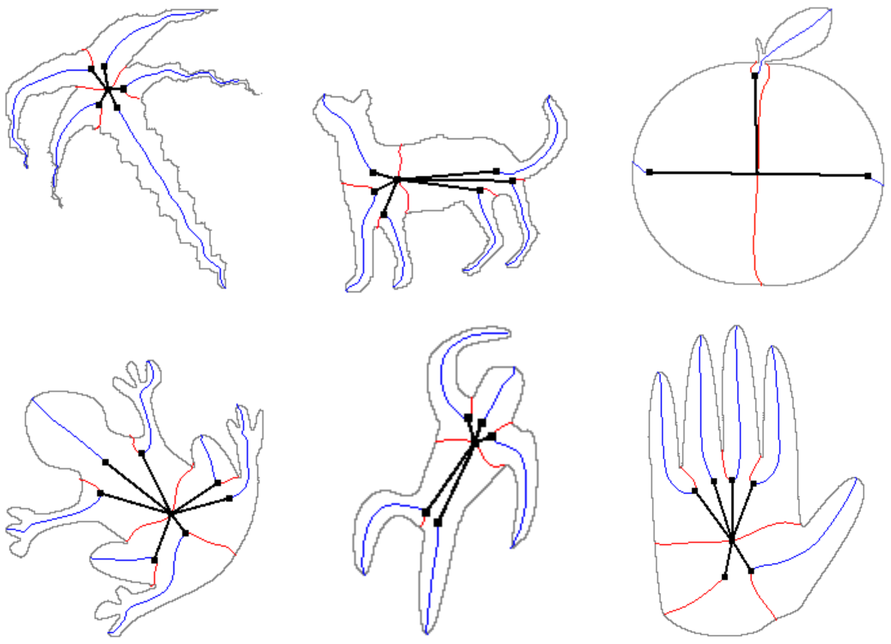

Fig. 3. Expressing Aslan skeletons with depth-1 trees. Note that each disconnection point (except the pruned major branches) gives rise to two different tree nodes, representing the positive and negative skeleton branches meeting at that disconnection point. For illustration purposes, only one node is drawn here (adapted from [9]).

limitation is that *shapes with holes* [4, 26, 28] can not be expressed easily in this representation because the construction of the global coordinate frame becomes complicated even though skeleton branches are extracted. Moreover, in the presence of large occlusion and severe missing parts, the shape center might move to a remote location, directly affecting the construction of the coordinate frame and accordingly the skeletal attributes.

In [9], Aslan skeleton was successfully utilized for shape recognition where the extracted skeletons are represented with *attributed ordered depth-1 trees* (Figure 3) and recognition is performed by matching the sketon tree of the query shape against those of the database shapes. The matching framework is based on a tree-edit distance algorithm [48]. The novelty lies in the fact that the semantic roles of the shapes to be matched are differentiated during the matching process. Matching involves comparing a query shape with a database shape whose class information is available. The set of all the shapes related to that class influences the edit costs, making the matching context-dependent. To provide this extra information, an auxiliary tree union structure, referred to as *category tree*, is introduced. Structurally, it is similar to an ordinary shape tree but its nodes store some statistics about the skeletal attributes stored in the corresponding nodes of the given shape trees.

More formally, let $\mathcal{T}_1 = \{u = (u^r, u^\theta, u^\ell, u^{type}) \mid u \in N_1\}$ and $\mathcal{T}_2 = \{v = (v^r, v^\theta, v^\ell, v^{type}) \mid v \in N_2\}$ stand for the query and the database shape trees. The matching algorithm solves the following minimization problem over the total cost of a sequence of edit operations:

$$dist\,(\mathcal{T}_1, \mathcal{T}_2) = \min_{\mathcal{S}} \left[\sum_{u \in \Lambda} \texttt{remove}\,(u) + \sum_{v \in \Delta} \texttt{insert}\,(v, \mathcal{B}) + \sum_{(u,v) \in \Omega} \texttt{change}\,(u, v, \mathcal{B}) \right] \quad (1)$$

where `remove`, `insert` and `change` are the three edit operations used to transform one tree into another. Here, Λ denotes the set of nodes removed from $\mathcal{T}_1$, Δ denotes the set of nodes inserted to $\mathcal{T}_1$ from $\mathcal{T}_2$ and Ω denotes the set of matched nodes, and $\mathcal{B}$ is the leaf node of the category tree associated with the node v of $\mathcal{T}_2$ that provides the context information in comparison.

The definitions of the three edit operations are as follows (for more details, refer to [9]):.

$$\texttt{remove}(u) = \left(\frac{u^\ell}{\ell_{max}(\mathcal{T}_1)} \right) (1 - u^r) \quad (2)$$

$$\texttt{insert}(v, \mathcal{B}) = \left(\frac{v^\ell}{\ell_{max}(\mathcal{T}_2)} \right) (1 - v^r) \times freq(\mathcal{B}) \quad (3)$$

$$\texttt{change}(u, v, \mathcal{B}) = \frac{f(u^r \mid v^r, \mathcal{B}) + f(u^\theta \mid v^\theta, \mathcal{B}) + 3f(u^\ell \mid v^\ell, \mathcal{B})}{5} \times freq(\mathcal{B}) \quad (4)$$

Since leaf nodes in a shape tree are ordered, the time complexity for matching two shape trees is $\mathcal{O}(mn)$, where m and n denote the number of leaves in the trees. A critical point is, of course, how the cost functions for the edit operations are defined. The cost functions not only affect the success of the matching algorithm but also its overall time complexity [45]. The advantages of the shape matching framework in [9] are two folds: First, the computational costs of edit operations are quite low and second, the number of leaf nodes of a shape tree is generally small as a consequence of the structure of Aslan skeleton.

3 The Proposed Method

3.1 Constructing Multiple Prototypes

In structural pattern recognition, the notion of *prototypes* for classification has been studied in a number of studies, *e.g.* [39, 41, 54]. Common to all these is a selection mechanism that proceeds based on a heuristic so that in the end the whole set of graphs or trees is represented with a reduced and more representative examples. In this regard, the structural equivalence of category trees and shape trees provides us an alternative and simple way to construct prototypes of shape categories. However, note that if a shape category contains outliers or has two or more subcategories, forming just a single category tree might be misleading. To overcome this drawback, we incorporate an additional clustering phase into the formation procedure of category trees described in [9] so that we form multiple category trees for each shape category. In devising the clustering step, we did not want to limit the number of category trees per each shape category, and hence, we employ a recursive clustering approach instead of using a method like k-medoids [11].

The steps of the revised formation procedure are as follows. Given a set of shape trees $\mathfrak{T}$, a temporary category tree $\mathcal{T}_{\mathcal{C}}$ is formed using the procedure proposed in [9]. Next, similar to the approach in [13], the most representative member of the set, which is denoted by $\mathcal{T}_m$, is identified as the shape tree that is most closest to $\mathcal{T}_{\mathcal{C}}$ (Equation 5). Consequently, $\mathfrak{T}$ is partitioned into two groups according to the measure $S(\mathcal{T})$ given in Equation 6, which simply returns a similarity value between 0 and 1. The shape trees having $S(\mathcal{T}) > 0.5$ (a threshold determined empirically) are removed from original set $\mathfrak{T}$ and used to form a category tree which represents a subcategory structure present in the set $\mathfrak{T}$. This procedure is repeated recursively until $\mathfrak{T}$ contains no shape trees. These steps are summarized in Algorithm 1.

$$\mathcal{T}_m = \arg\min_{\mathcal{T}\in\mathfrak{T}} dist(\mathcal{T}, \mathcal{T}_{\mathcal{C}})) \tag{5}$$

$$S(\mathcal{T}) = exp\left(-sim(\mathcal{T}, \mathcal{T}_m) \times \sum_{\substack{\mathcal{T}_i\in\mathfrak{T}\\ \mathcal{T}_i\neq\mathcal{T}}} \left(sim(\mathcal{T}, \mathcal{T}_i) - sim(\mathcal{T}_m, \mathcal{T}_i)\right)^2\right) \tag{6}$$

where $sim(\mathcal{T}_1, \mathcal{T}_2) = exp\left(-dist(\mathcal{T}_1, \mathcal{T}_2)\right)$.

Figure 4 shows the clustering results for two different sets of shapes which are obtained during the formation of the category trees. In each case, the procedure obtains two clusters, hence two separate category trees are formed as prototypes.

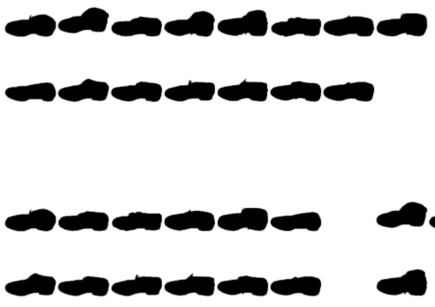
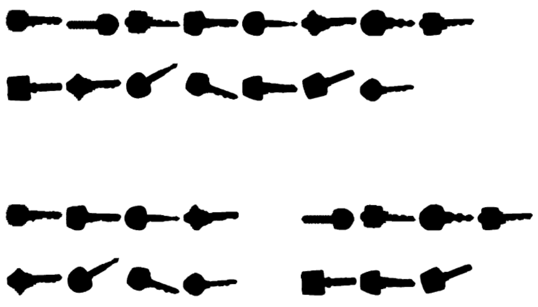
(a) (b)

Fig. 4. The clustering results for the given set of (a) `shoe` and (b) `key` shapes. In each figure, while the top row shows the given set of shapes, the bottom row shows the clustered shapes that are used in the formation of multiple category trees for the corresponding shape category.

Algorithm 1 Multiple Prototype Construction

```
Require: A set of shape trees $\mathfrak{T} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n\}, |\mathfrak{T}| > 0$
 1: repeat
 2:   $n \Leftarrow |\mathfrak{T}|$
 3:   Construct a temporary category tree $\mathcal{T}_C$ for the set of shape trees $\mathfrak{T}$
 4:   $\mathcal{T}_m \Leftarrow \arg\min_{\mathcal{T} \in \mathfrak{T}} dist(\mathcal{T}, \mathcal{T}_C)$
 5:   {Partitition $\mathfrak{T}$ into two subsets based on the distances to $\mathcal{T}_m$}
 6:   $\mathfrak{T}* \Leftarrow \emptyset$
 7:   for $i = 0$ to $n$ do
 8:     {Iterate over all the shape trees in $\mathfrak{T}$}
 9:     $S(\mathcal{T}_i) \Leftarrow exp\left(-sim(\mathcal{T}_i, \mathcal{T}_m) \times \sum_{\substack{\mathcal{T}_j \in \mathfrak{T} \\ \mathcal{T}_j \neq \mathcal{T}_i}} \left(sim(\mathcal{T}_i, \mathcal{T}_j) - sim(\mathcal{T}_m, \mathcal{T}_j)\right)^2\right)$
10:     {where $sim(\mathcal{T}_1, \mathcal{T}_2) = exp(-dist(\mathcal{T}_1, \mathcal{T}_2)$}
11:     if $S(\mathcal{T}_i) > 0.5$ then
12:       $\mathfrak{T}^* \Leftarrow \mathfrak{T}^* \bigcup \{\mathcal{T}_i\}$
13:     else
14:       $\mathfrak{T} \Leftarrow \mathfrak{T} - \{\mathcal{T}_i\}$
15:     end if
16:   end for
17:   Construct the category tree for the set of shape trees $\mathfrak{T}^*$, $\mathfrak{T}^* \subseteq \mathfrak{T}$
18: until $\mathfrak{T} = \emptyset$
```

3.2 Comparing A Shape With A Category Prototype

In our prior shape matching work [9], the role of category trees is to serve as an auxiliary structure to provide context information in matching. In this section, we utilize the structural equivalence between category trees and shape trees and present a mechanism to compare a shape with a collection of shapes where we treat the corresponding category tree as the prototype representing the set. Let $\mathcal{T}_{query}$ be the input shape tree to be compared with the category tree $\mathcal{T}_{\mathcal{C}}$. To compare these two structures, we first form a *mean shape tree* from $\mathcal{T}_{\mathcal{C}}$ by replacing the attributes at each node with the average values of the skeletal attributes which are associated with that node. Then, to compute the distance between the query shape and the mean shape tree, we adopt the tree-edit distance based matching method in [9] by replacing the label change cost function with a new one which is based on the generalization function proposed by Tenenbaum and Griffiths [58], while keeping the other two cost functions intact.

Suppose $\mathcal{B}$ and u denote nodes in $\mathcal{T}_{\mathcal{C}}$ and $\mathcal{T}_{query}$, respectively, and $\mathcal{X} = \left\{x^{(i)} \mid i = 1, \ldots, freq(\mathcal{B})\right\}$ be the set of corresponding nodes associated with the node $\mathcal{B}$. The generalization function in (7) approximates the conditional probability $p(u \in \mathcal{B} \mid \mathcal{X})$, which calculates the probability that the skeleton branch described with the node u might be an instance of the skeleton branch $\mathcal{B}$ appearing in the shape category formed by the examples $\mathcal{X}$.

$$g_{\mathcal{B}}(u) = \frac{exp\left(-\left(\tilde{d}_r/\sigma_r + \tilde{d}_\theta/\sigma_\theta + \tilde{d}_\ell/\sigma_\ell\right)\right)}{\left[\left(1 + \frac{\tilde{d}_r}{(r_{max}-r_{min})}\right)\left(1 + \frac{\tilde{d}_\theta}{(\theta_{max}-\theta_{min})}\right)\left(1 + \frac{\tilde{d}_\ell}{(\ell_{max}-\ell_{min})}\right)\right]^{freq(\mathcal{B})-1}} \tag{7}$$

where the value of $\tilde{d}_i$ $(i \in \{r, \theta, \ell\})$ equals to 0 if the value of the corresponding node attribute i falls inside the range of the attribute space spanned by the examples in the set $\mathcal{X}$. If this is not the case, the value is determined as the Euclidean distance from the value stored in u to the nearest one in the set $\mathcal{X}$ along the corresponding attribute space. Here, σ_r, σ_θ and σ_l are the scaling parameters which are taken as $\sigma_r = 1$, $\sigma_\theta = 2\pi$ and $\sigma_l = 2$ in the experiments.

When the values of the skeletal attributes r, θ and l specified in the node u moves away from the range of corresponding attribute spaces spanned by the examples in the set, probability decreases based on an exponential decay function. The adaptive behavior of the generalization function is demonstrated in Figure 5 for a sample skeleton branch appearing in a shape category. As the number of examples forming the category increases, the degree of generalization is adjusted to better describe the characteristics of the corresponding family of skeleton branch.

Based on the function $g_{\mathcal{B}}(u)$, we redefine the cost function of `change` operation for comparing a shape tree with a category tree as follows:

$$\texttt{change}^*(u, \mathcal{B}) = 1 - g_{\mathcal{B}}(u) \tag{8}$$

3.3 A Similarity-Based Approach for Shape Classification

The reference set used for defining a similarity space is comprised of the set of the category trees which are constructed using the procedure proposed in Section 3.1. Denoting this reference set by $\mathcal{R} = \left\{\mathcal{T}_{\mathcal{C}_1},\ \mathcal{T}_{\mathcal{C}_2},\ \ldots\ \mathcal{T}_{\mathcal{C}_M}\right\}$, where

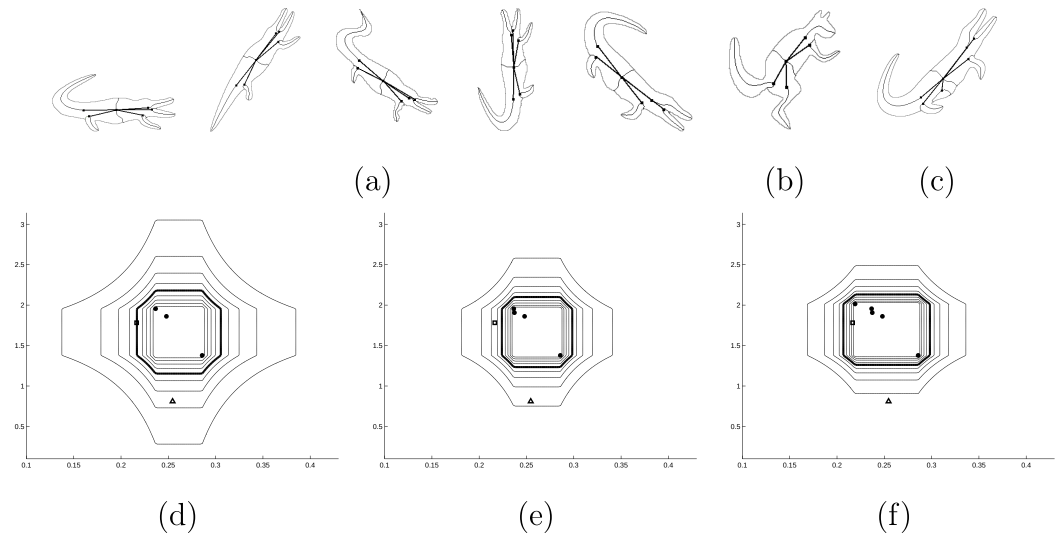

Fig. 5. Adaptive behavior of the generalization function for a sample skeleton branch appearing in the `crocodile` category. (a) Shapes forming the category. (b)-(c) A squirrel and a newly observed crocodile shape as query shapes. (d)-(f) The plots of the generalization function for the crocodile's back leg as the number of shapes forming the category increases from 3 to 5. For illustrative purpose, here, we only consider the skeletal attributes r and θ, neglecting the length attribute ℓ. Contours show the value of generalization function in increments of 0.1 where thick ones correspond to $p(u \in \mathcal{B} \mid \mathcal{X}) = 0.5$. The location of disconnection points (r, θ) are denoted by circles for the database shapes, triangle for the squirrel shape and square for the newly observed crocodile shape. As the number of examples increases, the degree of generalization become more accurate and the function better encodes within-class similarity and between-class variability.

M is the total number of category trees formed for N number of categories ($M \geq N$), a given query shape can be embedded as a point in the similarity space by taking negative exponential of the vector of distances between the shape tree of the query shape and the existing category trees:

$$\mathbf{S}(\mathcal{T}_{query}, \mathcal{R}) = exp\Big(-\Big(dist(\mathcal{T}_{query}, \mathcal{T}_{\mathcal{C}_1}),\ dist(\mathcal{T}_{query}, \mathcal{T}_{\mathcal{C}_2}),\ \ldots,\ dist(\mathcal{T}_{query}, \mathcal{T}_{\mathcal{C}_m})\Big)\Big) \tag{9}$$

In this similarity space, for each shape category, we train a separate SVM classifier with Gaussian kernel [44, 60] based on one-vs-all approach, where in the training phase, the similarity representation of members of that shape category is labeled as positive examples with `(1)` whereas the members of all other categories are labeled as negative examples with `(-1)`.

In classifying a novel shape, the vector of computed similarities is fed to all of the learned SVM classifiers, each outputting a scalar value. Then, a membership score for each category is obtained by normalizing these outputs according to Luce's choice rule [33, 34], as follows:

$$p(\mathcal{T}_{query} \in \mathcal{C}_i) = \frac{exp(\mathcal{Y}_i)}{\sum_{j=1}^{N} exp(\mathcal{Y}_j)} \tag{10}$$

where $\mathcal{T}_{query}$ and $\mathcal{Y}_i$ respectively denote the shape tree of the input shape and the output of the SVM classifier trained specifically for the i^{th} shape category $\mathcal{C}_i$.

3.4 The Complexity of The Approach

The training step of our approach involves building the similarity space (*i.e.* constructing the category prototypes) and learning multiple SVM classifiers in the corresponding similarity space. The time cost for constructing the prototypes can be given as $\mathcal{O}(Nm^2kt)$ if we assume there are N shape categories, each containing m instances and the recursive clustering step iterates k times. Specifically, at each iteration, we need to perform m^2 matchings to compute the pairwise distances with t being the cost of a single matching (which linearly depends on the number of leaf nodes of the trees). In our experiments, k is always found to be smaller than 3. Assuming a total of M ($M \geq N$) prototypes are constructed, the complexity of learning multiple SVM classifiers (using a naive implementation) is $\mathcal{O}(Mn^3)$, where n is the number of training examples. Hence, the overall cost of training stage is $\mathcal{O}(Nm^2kt + Mn^3)$. The testing stage is also made up of two steps. Given a query shape, first, it is matched with the category prototypes, and then the resulting similarity representation is fed to the SVM classifiers. Hence, the time cost of testing is much lower than that of the training stage.

4 Experimental Results

In this section, we quantitatively analyze the effectiveness of our approach on the shape data set used in [9], which contains 50 shape categories, each of which has 20 samples – a total of 1000 shapes. For the experiments, we randomly generated 100 partitions of the data set into training and test sets, where for each category, we sampled 15 instances for training and the rest for testing. In the evaluation of the methods, we examined two measures. The first is the average classification rate computed over the 100 test sets whereas the second one is the number of comparisons performed in a single query. Ideally, we expect to have a high classification accuracy while having a low computational cost, determined in our experiments by the number of comparisons (ignoring the cost of comparing the query shape with a prototype and the cost of training stage).

First, we explored the significance of both the category splitting step developed for forming a more descriptive similarity representation space and training SVM classifiers in the corresponding similarity space. For each of the 100 training sets generated randomly, the total number of prototypes created by the proposed category splitting procedure changes between 103 and 119, which is about twice of the actual number of categories. As shown in Table 1, the advantage of this step is clearly visible even if a simple 1-nearest neighbor (1-NN) classifier which does not involve learning is used for classification that the average classification rate increases about from 77.13% to 83.09%. However, notice that the gain comes at the expense of doubling the number of comparisons performed in a single query as the number of prototypes is nearly twice as much. Moreover, as proposed in Section 3.3, if we build a second layer and train multiple SVM classifiers in the similarity space defined by the prototypes [1], the average classification performance can be further boosted up to 91.18%. When the category splitting is ignored, the classification rate by SVM training only reaches to 87.80%. In Table 1, we also provide the standard deviations of the classification rates. Note that the proposed approach (with both category splitting and SVM training) has the lowest standard deviation. See the Supplementary Material for a sample partition of the database and the corresponding classification results.

Table 1
Quantitative analysis of the proposed approach. The effects of category splitting and SVM training.

	Without learning (1-NN)		**With SVM training**	
	w/ cat. splitting	w/o cat. splitting	w/ cat. splitting	w/o cat. splitting
Avg. classification rate	83.09 (2.99)	77.13 (2.59)	**91.18 (1.66)**	87.80 (1.95)
# of comparisons performed in a single query	~ 100	50	~ 100	50

Following the first set of experiments, we then compared our approach with some other methods, including another skeleton-based approach [8] and the ones based on a popular boundary-based shape representation, namely Inner-

[1] We used SVM^{light} [27] package and perform 5-fold cross validation in order to select the best values for the parameters γ and C, which correspond to the radius of RBF kernel and the weighting factor for misclassification penalty, respectively.

Distance Shape Context (IDSC) [32] [2]. As shown in Table 2, compared to our approach, the method of Bai *et al.* [8] performs slightly better, having an average classification rate of 91.66%. Besides, unlike the other approaches considered here (including our proposed method), it is a feature-based approach. It does not require matching the query shape with the prototype or the database shapes, but instead the features extracted from the query shape are compared with the features collected for each category from its instances, where these distances are combined to compute a class-conditional probability. Although this view eliminates the time-consuming job of matching operation, the approach does not involve learning any class-specific features and hence, it may not scale well for very large data sets.

Next, we focused on the widely popular IDSC representation [32] and built several shape classification schemes based on this boundary-based representation. Based on the matching framework proposed in [32], if we use 1-NN classifier, the average classification rate is found to be 80.51%, which requires comparing the query shape to each of the 750 shapes in the training set. In order to reduce the number of matchings, we performed k-medoid algorithm [11] with $k = 3$ to choose 3 exemplars for each category. In this case, the classification rate is equal to 71.26%, which is quite low compared to other results. Lastly, we utilized a complex classifier, so-called SVM-KNN [62], where the idea is to first locate the nearest-neighbors of the query and then train a local multi-class SVM based on the distances. SVM-KNN classifier achieves better performance rates when the number of neighbors is large, *e.g.* for K=10 and

[2] In forming and matching the IDSC descriptors, we used the implementation of the authors publicly available at `http://www.ist.temple.edu/~hbling`. The descriptions are generated based on 5 inner-distance bins and 12 inner-angle bins after uniformly sampling 100 landmark points across the shape boundary.

Table 2
Quantitative analysis of the proposed approach. Comparison with some previous approaches.

	Based on shape skeletons		**Based on Inner-Distance Shape Context (IDSC)**				
	Our Approach	The method in [8]	1-NN (3-Medoids)	1-NN	SVM-KNN (K=10)	SVM-KNN (K=100)	SVM-KNN (K=750)
Avg. classification rate	91.18 (1.66)	91.66 (1.73)	71.26 (2.23)	80.51 (1.92)	83.06 (1.80)	91.14 (1.38)	93.33 (1.46)
# of comparisons performed in a single query	~ 100	-	150	750	750	750	750

K=100, the classification rates are 83.06% and 91.14%, respectively. When all the shapes in the database is used, which reduces SVM-KNN to a single multi-class SVM, the performance rate is as high as 93.33%. However, note that in all these cases, the number of comparisons performed in a single query is always equal to 750, which is the number of shapes in the training set.

In summary, we obtained a high classification accuracy on the shape data set used while requiring a relatively less of number of comparisons in a single query. Moreover, as demonstrated in Figure 1, an advantage of our approach is that if the category of the query shape is not known, all the membership scores will be quite low. In such a situation, the network structure of the classification system can also be modified by adding additional nodes to both the first and the second layers to recognize that newly observed shape category in the future. This provides our rationale for not using a single multi-class SVM in our framework.

From an information retrieval perspective, one of the functions of classification is to eliminate unrelated comparisons in a retrieval task. As a supplementary experiment, we evaluate the indexing performance of the proposed classification framework on our former category-influenced matching [9]. We first performed classification to identify the top five most similar categories for the given shape. Then, we eliminated the categories which are below a certain threshold (0.2) after normalizing the corresponding membership scores.

Following to that, the query shape is compared to only the shapes belonging to the categories in the final candidate list. The resulting distance values along with the associated normalized membership information are then used to compute a new matching score as:

$$d^*(\mathcal{T}_1, \mathcal{T}_2) = 1 - exp(-dist(\mathcal{T}_1, \mathcal{T}_2)) \times p^*(\mathcal{T}_1 \in \mathcal{C}) \tag{11}$$

where $dist(\mathcal{T}_1, \mathcal{T}_2)$ denotes the category-influenced matching score and $p^*(\mathcal{T}_1 \in \mathcal{C})$ is the membership score normalized with respect to the retrieved shape categories.

We evaluate the effect of the proposed indexing strategy by repeating the experiments in [9] with a prior classification step. Due to space limitations, the results for a sample partition are given in the Supplementary Material. In the experiments, each query shape is only compared with the examples of the retrieved categories when $p^*(\mathcal{T}_1 \in \mathcal{C}) > 0.2$ (a threshold we set empirically). Figure 6 shows the average precision-recall curves for the category-influenced matching that includes a prior classification step. The experimental results reveal that performing a prior classification step contributes in achieving better precision values at each recall level with a less computation effort.

5 Summary and Conclusion

In this paper, a novel shape classification framework is presented, where the skeletonization approach in [1, 2] is used as the underlying shape representation. The proposed classification scheme has a two-layered structure. In the first layer, the distances between the input shape and the category prototypes

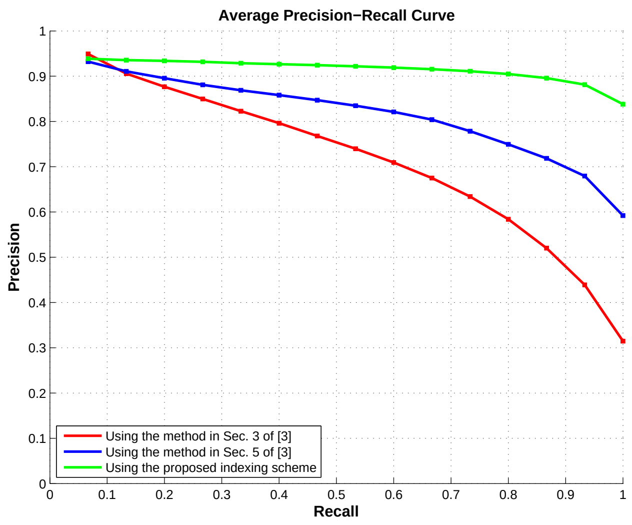

Fig. 6. Average precision-recall curves. At each recall level, compare the precision values of the proposed indexing scheme (shown in green) to those of the matching methods in Sec. 3 and Sec. 5 of [9] (respectively shown in red and in blue).

are computed to provide a similarity based representation of the query shape. Every category prototype has the same hierarchical structure with the query shape, thus making the comparison straightforward. The distance computations are performed using a tree edit distance algorithm which employs the generalization function proposed by Tenenbaum and Griffiths [58] in order to take into account the within-category variability. The second layer takes the collected similarity values produced by the first layer as input and produces the final membership score by taking the between-category variability into account. In our experiments, we obtain a classification rate 91.18% on a diverse shape database of 1000 shapes. As a supplementary experiment, we also tested the indexing performance of our classification framework in a shape retrieval task. As a final remark, we would like to emphasize that initially we start with a coarse and stable skeleton representation and let the discriminative features emerge as a result of our layered scheme which considers both the within-category similarities and the between-category variabilities.

Acknowledgements

This study was financially supported in part by the research grants TUBITAK 105E154 and 108E015. Aykut Erdem was supported in part by TUBITAK-BAYG PhD scholarship. The authors are grateful to Xiang Bai for providing his results for comparison.

References

[1] C. Aslan, A. Erdem, E. Erdem, S. Tari, Disconnected skeleton: Shape at its absolute scale., IEEE Trans. Pattern Anal. Mach. Intell. 30 (12) (2008) 2188–2203.

[2] C. Aslan, S. Tari, An axis-based representation for recognition, in: ICCV, vol. 2, 2005, pp. 1339–1346.

[3] J. August, K. Siddiqi, S. W. Zucker, Ligature instabilities in the perceptual organization of shape, Comput. Vis. Image Underst. 76 (3) (1999) 231–243.

[4] X. Bai, L. J. Latecki, Path similarity skeleton graph matching, IEEE Trans. Pattern Anal. Mach. Intell. 30 (7) (2008) 1282–1292.

[5] X. Bai, L. J. Latecki, W.-Y. Liu, Skeleton pruning by contour partitioning with discrete curve evolution, IEEE Trans. Pattern Anal. Mach. Intell. 29 (3) (2007) 449–462.

[6] X. Bai, W. Liu, Z. Tu, Integrating contour and skeleton for shape classification, in: IEEE Workshop on NORDIA (in conjunction with ICCV), 2009, pp. 360–367.

[7] X. Bai, X. Wang, W. Liu, L. J. Latecki, Z. Tu, Active skeleton for non-rigid object detection, in: ICCV, 2009, pp. 575–582.

[8] X. Bai, X. Yang, D. Yu, L. J. Latecki, Skeleton-based shape classification using path similarity, International Journal of Pattern Recognition and Artificial Intelligence 22 (4) (2008) 733–746.

[9] E. Baseski, A. Erdem, S. Tari, Dissimilarity between two skeletal trees in a context, Pattern Recognition 42 (3) (2009) 370–385.

[10] R. Basri, L. Costa, D. Geiger, D. Jacobs, Determining the similarity of deformable shapes, Vision Research 38 (15) (1998) 2365–2385.

[11] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, IEEE Trans. Pattern Anal. Mach. Intell. 24 (4) (2002) 509–522.

[12] G. Borgefors, G. Ramella, G. S. di Baja, Hierarchical decomposition of multiscale skeletons, IEEE Trans. Pattern Anal. Mach. Intell. 23 (11) (2001) 1296–1312.

[13] H. Bunke, S. Günter, X. Jiang, Towards bridging the gap between statistical and structural pattern recognition: Two new concepts in graph matching, in: ICAPR, 2001, pp. 1–11.

[14] E. Chávez, G. Navarro, R. Baeza-Yates, J. L. Marroquín, Searching in metric spaces, ACM Computing Surveys 33 (3) (2001) 273–321.

[15] L. Chen, J. J. McAuley, R. S. Feris, T. S. Caetano, M. Turk, Shape classification through structured learning of matching measures, in: CVPR, 2009, pp. 365–372.

[16] C. M. Cyr, B. B. Kimia, 3D object recognition using shape similiarity-based aspect graph, in: ICCV, vol. 1, 2001, pp. 254–261.

[17] M. R. Daliri, V. Torre, Classification of silhouettes using contour fragments, Comput. Vis. Image Underst. 113 (9) (2009) 1017–1025.

[18] M. F. Demirci, R. H. van Leuken, R. C. Veltkamp, Indexing through Laplacian spectra, Comput. Vis. Image Underst. 110 (3) (2008) 312–325.

[19] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, Wiley-Interscience Publication, 2000.

[20] R. Duin, D. de Ridder, D. Tax, Featureless pattern classification, Kybernetika 34 (4) (1998) 399–404.

[21] S. Edelman, Representation and Recognition in Vision, MIT Press, 1999.

[22] Y. Gdalyahu, D. Weinshall, Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes, IEEE Trans. Pattern Anal. Mach. Intell. 21 (12) (1999) 1312–1328.

[23] D. Geiger, T. L. Liu, R. V. Kohn, Representation and self-similarity of shapes, IEEE Trans. Pattern Anal. Mach. Intell. 25 (1) (2003) 86–99.

[24] R. L. Goldstone, Similarity, interactive activation, and mapping, Journal of Experimental Psychology: Learning, Memory, and Cognition 20 (1) (1994) 3–28.

[25] U. Hahn, N. Chater, L. B. Richardson, Similarity as transformation, Cognition 87 (1) (2003) 1–32.

[26] M. Hilaga, Y. Shinagawa, T. Kohmura, T. L. Kunii, Topology matching for fully automatic similarity estimation of 3D shapes, in: SIGGRAPH, 2001, pp. 203–212.

[27] T. Joachims, Making large-scale SVM learning practical, in: B. Schölkopf, C. Burges, A. Smola (eds.), Advances in Kernel Methods - Support Vector Learning, MIT Press, 1999, pp. 169–184.

[28] S. Krinidis, V. Chatzis, A skeleton family generator via physics-based deformable models, IEEE Transactions on Image Processing 18 (1) (2009) 1–11.

[29] J. K. Kruschke, ALCOVE: An exemplar-based connectionist model of category learning, Psychological Review 99 (1) (1992) 22–44.

[30] B. Landau, L. B. Smith, S. S. Jones, The importance of shape in early lexical learning, Cognitive Development 3 (1988) 299–321.

[31] L. J. Latecki, R. Lakamper, Shape similarity measure based on correspondence of visual parts., IEEE Trans. Pattern Anal. Mach. Intell. 22 (10) (2000) 1185–1190.

[32] H. Ling, D. Jacobs, Shape classification using the inner-distance, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2) (2007) 286–299.

[33] R. D. Luce, Individual Choice Behavior: A Theoretical Analysis, Wiley, New York, 1959.

[34] R. D. Luce, The choice axiom after twenty years, Journal of Mathematical Psychology 15 (1977) 215–233.

[35] D. Macrini, K. Siddiqi, S. Dickinson, From skeletons to bone graphs: Medial abstraction for object recognition, in: CVPR, 2008, pp. 1–8.

[36] D. Marr, Vision: A Computational Investigation into the Human Representation and Processing of Visual Information, W. H. Freeman and Company, New York, 1982.

[37] D. Medin, M. M. Schaffer, Context theory of classification learning, Psychological Review 85 (1978) 207–238.

[38] R. M. Nosofsky, Attention, similarity, and the identification-categorization relationship, Journal of Experimental Psychology: General 115 (1) (1986) 39–57.

[39] R. Paredes, E. Vidal, Learning prototypes and distances: A prototype reduction technique based on nearest neighbor error minimization, Pattern Recognition 39 (2) (2006) 180–188.

[40] E. Pekalska, R. P. W. Duin, The Dissimilarity Representation for Pattern Recognition. Foundations and Applications, World Scientific, 2005.

[41] E. Pekalska, R. P. W. Duin, P. Paclík, Prototype selection for dissimilarity-based classifiers, Pattern Recognition 39 (2) (2006) 189–208.

[42] E. Rosch, Principles of categorization, in: E. Rosch, B. B. Lloyd (eds.), Cognition and Categorization, John Wiley & Sons Inc, 1978, pp. 27–48.

[43] B. H. Ross, V. S. Makin, Prototype versus exemplar models in cognition, in: R. J. Sternberg (ed.), The Nature of Cognition, chap. 7, The MIT Press, 1999, pp. 205–241.

[44] B. Schölkopf, A. J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, 2001.

[45] T. B. Sebastian, P. N. Klein, B. B. Kimia, Shock-based indexing into large shape databases, in: ECCV, 2002, pp. 731–746.

[46] T. B. Sebastian, P. N. Klein, B. B. Kimia, Recognition of shapes by editing their shock graphs, IEEE Trans. Pattern Anal. Mach. Intell. 26 (5) (2004) 550–571.

[47] D. Shaked, A. M. Bruckstein, Pruning medial axes., Comput. Vis. Image Underst. 69 (2) (1998) 156–169.

[48] D. Shasha, K. Zhang, Approximate tree pattern matching, in: Pattern Matching Algorithms, Oxford University Press, 1997, pp. 341–371.

[49] R. N. Shepard, Toward a universal law of generalization for psychological science, Science 237 (1987) 1317–1323.

[50] A. Shokoufandeh, D. Macrini, S. Dickinson, K. Siddiqi, S. W. Zucker, Indexing hierarchical structures using graph spectra, IEEE Trans. Pattern Anal. Mach. Intell. 27 (7) (2005) 1125–1140.

[51] K. Siddiqi, S. Bouix, A. Tannenbaum, S. W. Zucker, Hamilton-Jacobi skeletons, Int. J. Comput. Vision 48 (3) (2002) 215–231.

[52] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, S. W. Zucker, Shock graphs and shape matching, Int. J. Comput. Vision 35 (1) (1999) 13–32.

[53] E. E. Smith, D. Medin, Categories and concepts, Harvard University Press, Cambridge, 1981.

[54] B. Spillmann, M. Neuhaus, H. Bunke, E. Pekalska, R. P. W. Duin, Transforming strings to vector spaces using prototype selection, in: SSPR, 2006, pp. 287–296.

[55] A. Srivastava, S. Joshi, W. Mio, X. Liu, Statistical shape analysis: Clustering learning and testing, IEEE Trans. Pattern Anal. Mach. Intell. 27 (4) (2005) 590–602.

[56] K. B. Sun, B. J. Super, Classification of contour shapes using class segment sets, in: CVPR, vol. 2, 2005, pp. 727–733.

[57] S. Tari, J. Shah, H. Pien, Extraction of shape skeletons from grayscale images, Comput. Vis. Image Underst. 66 (2) (1997) 133–146.

[58] J. B. Tenenbaum, T. L. Griffiths, Generalization similarity and Bayesian inference, Behavioral and Brain Sciences 24 (2001) 629–640.

[59] A. Tversky, Features of similarity, Psychological Review 84 (1977) 327–352.

[60] V. N. Vapnik, The Nature of Statistical Learning Theory, Springer, 1995.

[61] A. D. Ward, G. Hamarneh, The groupwise medial axis transform for fuzzy skeletonization and pruning, IEEE Trans. Pattern Anal. Mach. Intell. 32 (6) (2010) 1084 – 1096.

[62] H. Zhang, A. C. Berg, M. Maire, J. Malik, SVM-KNN: Discriminative nearest neighbor classification for visual category recognition, in: CVPR, 2006, pp. 2126–2136.

[63] S. C. Zhu, A. L. Yuille, Forms: A flexible object recognition and modeling system, Int. J. Comput. Vision 20 (3) (1996) 187–212.