Fast Image Classification for Monument Recognition

GIUSEPPE AMATO, FABRIZIO FALCHI, and CLAUDIO GENNARO, ISTI-CNR

Content-based image classification is a wide research field that addresses the landmark recognition problem. Among the many classification techniques proposed, the *k*-nearest neighbor (*kNN*) is one of the most simple and widely used methods. In this article, we use *kNN* classification and landmark recognition techniques to address the problem of monument recognition in images. We propose two novel approaches that exploit *kNN* classification technique in conjunction with local visual descriptors.

The first approach is based on a relaxed definition of the local feature based image to image similarity and allows standard *kNN* classification to be efficiently executed with the support of access methods for similarity search.

The second approach uses *kNN* classification to classify local features rather than images. An image is classified evaluating the consensus among the classification of its local features. In this case, access methods for similarity search can be used to make the classification approach efficient.

The proposed strategies were extensively tested and compared against other state-of-the-art alternatives in a monument and cultural heritage landmark recognition setting. The results proved the superiority of our approaches.

An additional relevant contribution of this work is the exhaustive comparison of various types of local features and image matching solutions for recognition of monuments and cultural heritage related landmarks.

Categories and Subject Descriptors: H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models*

General Terms: Landmark Recognition

Additional Key Words and Phrases: kNN classification, object recognition, image classification, local features, tourism

ACM Reference Format:

Giuseppe Amato, Fabrizio Falchi, and Claudio Gennaro. 2015. Fast image classification for monument recognition. ACM J. Comput. Cult. Herit. 8, 4, Article 18 (August 2015), 25 pages.
DOI: http://dx.doi.org/10.1145/2724727

1. INTRODUCTION

Perhaps the easiest way to obtain information about something is to use a picture of the object of interest as a query. Consider, for instance, a cultural tourist who is in front of a monument and wants to have information about it. A very easy and intuitive action can be that of pointing to the monument with a smartphone and obtaining pertinent and contextual information.

This work was supported by the VISITO Tuscany project, funded by Regione Toscana; the Europeana network of Ancient Greek and Latin Epigraphy (EAGLE, grant agreement 325122), co-funded by the European Commission; and Mobility and Tourism in Urban Scenarios (MOTUS), co-funded by the Italian government.

Authors' addresses: G. Amato, F. Falchi, and C. Gennaro, ISTI-CNR, via G. Moruzzi 1, 56124 Pisa, Italy; emails: {giuseppe.amato, fabrizio.falchi, claudio.gennaro}@isti.cnr.it.

DOI: http://dx.doi.org/10.1145/2724727

The aim of this article is to discuss, propose, and compare techniques of image recognition that can be used to support the scenario just described. The proposed techniques have been thoughtfully tested and compared in a cultural heritage domain.

A commonly used approach to identify an object contained in a query image is to use the k-nearest neighbor (*kNN*) classification algorithm [Cover and Hart 1967]. At the most abstract level, a *kNN* classifier executes the following steps. Given a query image, the *kNN* algorithm scans a training set to retrieve the best-matching images. The most represented class (if any) among the retrieved images determines the class of the object contained in the query image.

A promising technique, increasingly applied with success in recent years in image-matching tasks is to compare images in terms of their local features. *Local features* (or *descriptors*) are visual descriptors of selected interest points, or keypoints, occurring in images [Lowe 2004; Bay et al. 2006; Rublee et al. 2011]. The comparison of two images, in terms of their local features, involves two steps: the detection of pairs of matching keypoints in the two images, and a geometric consistency check of the position of these matching keypoints. Determining the pairs of matching keypoints in two images involves finding pairs of local features whose mutual similarity is much higher than their similarity with other local features [Lowe 2004]. Checking the geometric consistency of the identified matching pairs implies finding a reasonable geometric transformation that maps the position of most of the matching keypoints of the first image to the position of the corresponding keypoints of the second image [Fischler and Bolles 1981]. Using local feature matching and geometric consistency check strategies, it is possible to rank images of a training set according to the degree with which they match the query image and then execute the *kNN* classification algorithm.

Other descriptors, such as Maximally Stable Extremal Region (MSER) [Matas et al. 2004] and Local Binary Pattern (LBP) [Ojala et al. 2002] can be used for image-matching tasks. However, according to the reported results in Mikolajczyk and Schmid [2005] and Mikolajczyk et al. [2005], local features similar to the Scale-Invariant Feature Transform (SIFT) descriptor generally perform best on object recognition problems. Moreover, methods such as SIFT, Speeded-Up Robust Features (SURF), and Oriented FAST and Rotated BRIEF (ORB) provide both an interest point detector and a feature descriptor implementation.

The idea of applying the *kNN* classification in combination with the geometric consistency technique is very effective for tasks where only a few objects need to be recognized and the training sets are small. The drawback of this approach is that it is not scalable when the number of training images used to describe the objects is very large. The execution of the *kNN* classification algorithm requires that the query image be sequentially compared with all images of the training set. To compare the query image with a single image of the training set, all local features of the query image must be compared with all local features of the training set image. Considering that each image is typically described by thousands of local features, this means that a single image comparison requires something like $1{,}000 \times 1{,}000$ local feature comparisons. This has to be repeated for all images of the training set, every time a new query image is processed.

For example, in the experiments that will be described in this article, the size of the training set is some orders of magnitude larger than the number of objects (monuments in our case) to be recognized. In fact, a query image related to a monument, such as a church or a tower, might be taken from an arbitrary position from anywhere around the monument, capturing just portions of the monument and of the landmark in which it is situated. Consequently, for a single monument, we could need hundreds of training images, depicting it from various points of views and perspectives, to obtain a high recognition quality. The recognition of small objects poses fewer problems given that, in many cases, such objects are entirely contained in the query image. In these cases, typically, just the orientation of the objects changes.

To reduce the cost of finding the best matches for local image features, some years ago the bag of visual words (BoW) [Sivic and Zisserman 2003] paradigm was introduced. With this technique, sometimes called *bag of features*, groups of very similar local features, taken from the entire training set, are clustered together and represented by their centroid (a representative feature for the entire cluster denoted visual word). The set of centroids is called the *visual word vocabulary*. An image is then represented by quantizing each feature to its nearest visual word. To decide whether two local features belonging to two different images match, it is sufficient to check whether they belong to the same cluster or, in other words, are represented by the same visual word.

The *kNN* classification technique can be successfully applied directly to the BoW representation. However, this approach still presents some scalability and effectiveness problems. Even with the use of inverted files to maintain relationships among features and images, Zhang et al. [2009] state the following: "A fundamental difference between an image query (e.g. 1,500 visual terms) and a text query (e.g. 3 terms) is largely ignored in existing index design. This difference makes the inverted list inappropriate to index images." In addition, the use of the BoW approach makes it difficult to efficiently perform a geometric consistency check, and the approximation introduced by the quantization of the local features reduces the effectiveness.

The approaches presented in this work lie in between these two extremes (direct use of local features on one side and BoW on the other). We still exploit the effectiveness of local features and geometric consistency, but we rely on the use of access methods for local image features [Zezula et al. 2006; Samet 2005] to scale to a large number of classes and training images. These strategies have been tested and compared against other state-of-the-art approaches in the context of landmark recognition for cultural heritage.

2. CONTRIBUTION OF THIS WORK

In this article, we compare several strategies to recognize the content of digital pictures against two novel proposed approaches. We particularly focus on discussing and evaluating how the various options and techniques perform in the applicative scenario of monument and cultural heritage related landmark recognition. The two new proposed approaches are based on image *kNN* classification techniques.

The first approach exploits the *kNN* classification to classify images and relies on a relaxed definition of the local feature based image to image similarity definition, which allows efficient index for similarity search to be used. Surprisingly, we show that in addition to increasing efficiency and scalability, this approach also increases effectiveness.

The second approach that we propose—the local feature based image classifier—uses *kNN* classification to classify individual local features of an image rather than the entire image. It consists of a two-step classification process: (1) *kNN* classification of individual local features and (2) classification of whole images evaluating the consensus among the classes and the confidences assigned to each local feature in step (1). In addition, this approach makes possible the usage of efficient indexes for similarity search to offer high efficiency and scalability without penalizing effectiveness. Tests were executed using various types of local features and applying geometric consistency check techniques.

An additional significant contribution of this work is the comparison between various types of local feature and image matching solutions in a monument and cultural heritage related landmark recognition scenario. As far as we know, no such complete and extensive comparisons have been performed previously in such a consistent and specific scenario.

A preliminary version of the approaches presented in this article was presented in Amato et al. [2011]. The novel contribution here, with respect to previous work, can be summarized as follows. We extensively investigated and experimented with different approaches of *kNN* classifications for landmark recognition, and in particular, we introduced the novel concept of local feature based image classification. We compared the proposed approaches using ORB and BRISK features in addition to SIFT and SURF features. We also compared our results against the BoW approach. Finally, we introduced the use of a geometric constraint check in combination with the local feature based image classifier. More experiments and analysis were also carried out.

The article is organized as follows. Section 3 presents other related work. Section 4 provides the background for the remainder of the article. Section 5 introduces the pairwise distance criterion used in classification algorithms. Section 6 contains the details of our proposed approaches, and Section 7 validates our proposed techniques. A concluding summary is given in Section 8.

3. RELATED WORK

In this work, we address the problem of landmark recognition and visual categorization with special focus on *kNN* classification and local image features. In Chen et al. [2009], a survey of the literature on mobile landmark recognition for information retrieval is given. The classification methods reported include SVM, Adaboost, the Bayesian model, HMM, and GMM. However, the survey does not report the *kNN* classification technique, which is the main focus of this article.

In Zheng et al. [2009], Google presented its approach to building a Web-scale landmark recognition engine. Most of the work reported was used to implement the Google Goggles service [Google 2010]. The image recognition is based on a *kNN* classifier using local feature matching. According to the authors, the recognition performance on more than 5,000 landmarks reaches an accuracy of 80.8%.

Popescu and Moëllic [2009] used a georeferenced collection of 5,000 landmarks worldwide to automatically annotate landmark images. They organized the landmarks spatially and classified the images using spatial distance together with *kNN* classification. The images to label are indexed using only the BoW approach.

A mobile landmark recognition system called *Snap2Tell* was developed in Chevallet et al. [2007]. However, the authors use a simple matching technique based on color histograms and a 1NN classifier, combined with localization information. For the task of image-based geolocation, a similar approach has been exploited in Hays and Efros [2008].

In Labbé [2014], a tutorial on how a system for object recognition can also be used for place recognition is given. The system uses local features to execute the recognition task.

In Fagni et al. [2010], various MPEG-7 global descriptors have been used to build *kNN* classifier committees. However, local features were not taken into consideration.

Boiman et al. [2008] propose an approach to 1NN image classification that uses a kd-tree structure for efficiency and is similar in spirit to one of the approaches presented in this article. This work also introduced a novel, nonparametric approach for image classification, the naive Bayes nearest neighbor (NBNN) classifier, which was further generalized by Timofte et al. [2013] by replacing the nearest-neighbor part with more elaborate and robust (sparse) representations (*kNN*, Iterative Nearest Neighbors (INN), Local Linear Embedding (LLE), etc.). Bosch et al. [2008] also use a *kNN* classifier in combination with probabilistic Latent Semantic Analysis for scene classification purposes. However, no access methods were used to handle efficiency issues in the case of large dimension problems.

kNN classifiers are also suitable for real-time learning applications such as three-dimensional object tracking. In Hinterstoisser et al. [2011], the authors exploit a simple nearest-neighbors classification using a set of "mean patches" that encode the average of the keypoints appearing over a limited set of poses. However, learning approaches do not scale very well with respect to the size of the keypoints database [Lourenço 2011].

Johns and Yang [2011] address the problem of recognizing a place depicted in an image by clustering similar database images to represent distinct scenes, and tracking local features that are consistently detected to form a set of real-world landmarks. In this work, features are first quantized and images are described as a BoW, allowing a more efficient means of computing image similarities. The closest *k* database images to the query image are then passed on to the second stage. Here, geometric verification prunes out false-positive feature matches from the first stage.

The idea of applying the BoW technique to transform images described by local features in vectors to exploit *kNN* classification is also used in Mejdoub and Ben Amar [2011]. In this study, the authors propose a new categorization tree based on the *kNN* algorithm. The proposed categorization tree combines both unsupervised and supervised classification of local feature vectors. The advantage of this tree is that it achieves a trade-off between accuracy and speedup of categorization. The proposed technique, however, involves several complex steps: a hierarchical lattice vector quantization algorithm, and a supervised step based on both feature vector labeling and a supervised feature selection method. In this respect, similar approaches in which high-dimensional descriptors based on local features, such as Vector of Locally Aggregated Descriptors (VLAD) [Jegou et al. 2010] and Locality Constraint Linear Coding (LLC) [Wang et al. 2010], are employed have become a topic of considerable interest in the development of classification systems (e.g., see Su et al. [2013], Amato et al. [2013], and Perronnin and Dance [2007]).

In Haase and Denzler [2011], state-of-the-art CBIR methods were tested to recognize landmarks in a large-scale scenario. The image dataset consists of 900 landmarks from 449 cities and 228 countries. BoW and visual phrase approaches were tested in combination with SVM and *kNN* classifiers. The best results were obtained by using a *kNN* classifier in combination with the BoW description.

Some approaches exploit a metric learning phase to improve the performance of metric-based *kNN* classification algorithms. Although these methods are reported to be effective, most of the existing applications are still limited to vector space models in which there is no connection to local features. For a recent survey on metric learning, see Bellet et al. [2013]. Within this topic, there is increased interest in local distance functions for nearest-neighbor classification on local image patches [Mahamud and Hebert 2003] or geometric blur features [Frome et al. 2007; Malisiewicz and Efros 2008; Zhang et al. 2006, 2011]. Note that such approaches, however, often map local features to multiresolution histograms and compute a weighted histogram intersection; approximate correspondence can be captured by a pyramid vector representation [Grauman and Darrell 2007].

Weighted voting is another common approach for improving *kNN* classifiers. Weights are usually either based on the position of an element in the *kNN* list or its distance to the observed data point [Zuo et al. 2008]. However, the hubness weighting scheme that was first proposed for high-dimensional data in Radovanović [2010] is slightly more flexible; each point in the training set has a unique associated weight, with which it votes whenever it appears in some *kNN* list, regardless of its position in the list. This idea was recently generalized into fuzzy *kNN* for local features [Tomašev et al. 2011]. This technique still relies on vector representation and therefore is only suitable for high-dimensional data such as codebooks of most representative SIFT features (BoW).

Finally, comparatively few papers have proposed the use of boosting techniques for *kNN* classification. Boosting methods adaptively change the distribution of the training set based on the performance of the previous classifiers [Garca-Pedrajas and Ortiz-Boyer 2009]. Unfortunately, to the best of our knowledge, all boosting techniques for *kNN* classification rely on a pairwise distance between objects to be classified. A good survey of *kNN* classification boosting can be found in Piro et al. [2013].

4. OBJECT RECOGNITION

In this section, we provide preliminaries and give an overview of the local features that we have used.

4.1 Notation and Preliminaries

Throughout this article, we represent each image I by a set of n local features l (i.e., $I = \{l_1, \ldots, l_n\}$). With a slight abuse of notation, we use the general notation $d()$ to denote the distance functions used for comparing images or local features.

Let S be a database of objects x and a d distance function for the objects; the k-th nearest neighbor of object q can then be recursively defined as

$$NN_k(q, S) = \begin{cases} x \in S \mid \forall y \in S \;\; d(q, y) \geq d(q, x) & \text{if } k = 1; \\ \\ NN_{k-1}(q, S \setminus \{NN_{k-1}(q, S)\}) & \text{if } k > 1. \end{cases} \quad (1)$$

The set of the first kNNs is defined as

$$kNN(q, S) = \{NN_{\hat{k}}(q, S) \mid \hat{k} = 1..k\}. \quad (2)$$

4.2 Local Features

In the past decade, the introduction of local features to describe image visual content, along with local feature matching and geometric consistency check approaches, has significantly advanced the performance of image content and object recognition techniques. In the following, we introduce these two strategies, which are at the basis of the classification techniques that we use to perform the recognition of monuments in images.

Local feature descriptors describe selected individual points or areas in an image. The extraction is executed in two steps. First, a set of keypoints in the image is detected. Second, the area around the selected keypoints is analyzed to extract a visual description. Keypoint selection strategies are appropriately designed to guarantee invariance to scale changes, and the same points are selected under different views of the same object. Local feature descriptors contain information that allow local feature matching, such as deciding that two local features from two different images represent the same point. Standard information on the position in the image, the orientation, and size of the region are typically associated with the visual information that depends on the particular local features. Various local features have been proposed. In this work, we tested SIFT, SURF, ORB, and BRISK.

4.2.1 *SIFT.* SIFT [Lowe 2004] is a representation of low-level image content that is based on a transformation of the image data into scale-invariant coordinates relative to local features. Local features are low-level descriptions of keypoints in an image. Keypoints are interest points in an image that are invariant to scale and orientation. Keypoints are selected by choosing the most stable points from a set of candidate locations. Each keypoint in an image is associated with one or more orientations, based on local image gradients. Image matching is performed by comparing descriptions of the keypoints in the images.

This extraction scheme has been used by many other local features, including the following ones. In particular, SIFT selects keypoints using a difference of Gaussians approach that can be seen as an approximation to the Laplacian that results in detecting blobs. The description of each keypoint and its neighbors (i.e., the blob) is based on an histogram of orientation gradients normalized with respect to the dominant orientations to be rotation invariant. We used publicly available software developed by Lowe [2005] to both detect keypoints and extract the SIFT features.

4.2.2 *SURF.* The basic idea of SURF [Bay et al. 2006] is quite similar to SIFT. SURF detects some keypoints in an image and describes them using orientation information. However, the SURF definition uses a new method for both the detection of keypoints and their description that is much faster while still guaranteeing a performance comparable to or even better than SIFT. Specifically, keypoint detection relies on a technique based on an approximation of the Hessian matrix. The descriptor of a keypoint is built considering the distortion of Haar-wavelet responses around the keypoint itself. We used the publicly available noncommercial software developed by the authors [Bay and Van Gool 2006] to both detect the keypoints and to extract the SURF features.

4.2.3 *ORB.* ORB [Rublee et al. 2011] is a very fast and effective local feature descriptor that selects keypoints using the FAST detector and builds features with an improved version of the BRIEF descriptors that offer rotational invariance. It is very fast in both the feature extraction phases and matching phases, which can be used for real-time applications even with low-power devices and without GPU acceleration. The descriptor has a binary format, and the simple Hamming distance is used for comparing local features.

4.2.4 *BRISK.* Similarly to ORB, BRISK [Leutenegger et al. 2011] is also a binary local feature descriptor. It uses a FAST-based keypoint detector and generates a bit-string descriptor from intensity comparisons retrieved by dedicated sampling of keypoint neighborhood. BRISK also uses the Hamming distance to compare local features. A comparison of ORB and BRISK together with BRIEF has been presented in Heinly et al. [2012].

4.3 Local Features Matching

Local features l automatically extracted from an image I are used to identify, in two distinct images I_i and I_j, couples of matching descriptors (l_i, l_j), where $l_i \in I_i$ and $l_j \in I_j$. Identifying matches requires comparing local descriptors using a distance function d, identifying a candidate match $l_j \in I_j$ for any $l_i \in I_i$, and filtering out matches with high probability to be incorrect.

For SIFT and SURF, the Euclidean distance is used, whereas the Hamming distance is the obvious choice for binary features such as ORB and BRISK.

The candidate match for l_i is typically the nearest local descriptor in I_j (i.e. $NN_1(l_i, I_j)$).

Filtering incorrect matches is the most difficult task. Lowe [2004] showed that the distance $d(l_i, NN_1 (l_i, I_j))$ is not a good measure of the quality of matches. Instead, he proposed to consider the ratio between the distance from l_i of the first and the second nearest neighbors in I_j—that is,

$$\sigma(l_i, I_j) = \frac{d(l_i, NN_1(l_i, I_j))}{d(l_i, NN_2(l_i, I_j))}. \tag{3}$$

Any matching pair of descriptors $\langle l_i, NN_1(l_i, I_j)\rangle, l_i \in I_i$ for which $\sigma(l_i, I_j) > c$, where c is a predefined threshold, is discarded. Thus, the set of candidate feature matches between image I_i and I_j is

$$M_\sigma(I_i, I_j) = \{\langle l_i, NN_1(l_i, I_j)\rangle \mid \sigma(l_i, I_j) < c, l_i \in I_i\}. \tag{4}$$

In Lowe [2004], it was reported that $c = 0.8$ allows us to eliminate 90% of the false matches while discarding less than 5% of the correct matches when using SIFT. In Amato and Falchi [2010], an experimental evaluation of classification effectiveness varying c for both SIFT and SURF confirmed the results obtained by Lowe. In the following, we will use $c = 0.8$ for both SIFT and SURF; we used $c = 0.9$ for the ORB and BRISK binary local features because it gave better performance.

We call the set of matches M_σ, defined earlier, the *plain distance ratio matches*. In the following, we will also define additional strategies to find the set matches, some of which are obtained starting from M_σ itself.

5. PAIRWISE IMAGE DISTANCE

Central to the concept of the *kNN* classifier is the definition of a pairwise image distance d between two images, which is based on how many features match and the closeness of the matches.

We define a distance function based on the plain distance ratio matches (Section 5.1) on the BoW quantization approach (Section 5.2). Finally, we extend the distance functions to also handle geometric consistency checks (Sections 5.3 and 5.4).

5.1 Local Feature Matching

The pairwise matching between two images is based on how many feature descriptors match. Given a set $M_\sigma(I_i, I_j)$ of candidate local feature matches (see Section 4.3) between two images I_i, I_j, we define the distance as

$$d_\sigma(I_i, I_j) = 1 - \frac{|M_\sigma(I_i, I_j)|}{|I_i|}. \tag{5}$$

Note that the proposed distance measure is not actually a distance measure, as it is not symmetric: $d_\sigma(I_i, I_j) \neq d_\sigma(I_j, I_i)$. Moreover, since $0 \leq d_\sigma \leq 1$, sometimes it is more convenient to use the concept of *similarity* $s_\sigma = 1 - d_\sigma$.

5.2 BoW Matching

The traditional BoW model used for text has been applied to images by treating image features as words. As for text documents, a BoW description is a sparse vector of number of occurrences of visual words taken from a predefined vocabulary. The assumption is that two features match if they have been assigned to the very same visual words. Thus, the BoW approach can also be used for efficient features matching (see Philbin et al. [2007] and Philbin [2010]).

The first step to describe images using visual words is to select some local features creating the *visual vocabulary*. The visual vocabulary typically is built grouping local descriptors of the dataset using a clustering algorithm such as k-means The second step consists of describing each image using the words of the vocabulary occurring in it.

At the end of the process, each image is described as a set of visual words. More formally, the BoW framework consists of a group of cluster centers, referred to as visual words $W = \{w_1, w_2, \ldots, w_k\}$ [Turcot and Lowe 2009]. Let b_W be a function that assigns a visual word to each local descriptor l_i of an image I_i as follows:

$$b_W(l_i) = \arg_w NN_1(l_i, W). \tag{6}$$

Let $B_W(I_i)$ be the set of visual words corresponding to the local features of the image I_i, for instance,

$$B_W(I_i) = \{b_W(l_i) \ \ \forall l_i \in I_i\}, \tag{7}$$

that we are able to convert images into a vector of visual word occurrences, as for standard full-text retrieval term frequency (TF) approach:

$$tf_j(I_i) = |\{j\} \cap B_W(I_i)|, \tag{8}$$

where $tf_m(I_i)$ is the m-th element of the vector of visual words and corresponds to the number of occurrences of w_m in the set $B_W(I_i)$. To compare image word occurrence, cosine similarity can be used:

$$d_w(I_i, I_j) = 1 - \frac{\sum_{m=1}^{n} tf_m(I_i) tf_m(I_j)}{\sqrt{\sum_{m=1}^{n} tf_m(I_i)^2}\sqrt{\sum_{m=1}^{n} tf_m(I_j)^2}} \tag{9}$$

More advanced weighting schemes based on information retrieval technology such as TF-IDF can be used (e.g., see Tirilly et al. [2010]). Using these similarity functions, traditional inverted files can be used to search nearest-neighbor images.

5.3 Geometric Consistency Constraints

To further improve the effectiveness of the pairwise image matching described earlier, geometric consistency constraints can be exploited. The problem is to determine a transformation that maps the positions of the keypoints in the first image to the positions of the corresponding keypoints of the second image. Only matches consistent with this transformation are retained. As discussed previously, the coordinates of the keypoints, together with the size and orientation of the region, are associated with each local descriptor.

The algorithms used to estimate such a transformation are typically the Random Sample Consensus (RANSAC) [Fischler and Bolles 1981] and Least Median of Squares. However, fitting methods such as RANSAC or Least Median of Squares perform poorly when the percentage of correct matches falls much below 50%. Fortunately, much better performance can be obtained by clustering features in the scale and orientation space using the Hough transform as suggested in Lowe [2004].

Estimating a transformation using RANSAC involves (1) randomly selecting the requested number of matches for the given transformation estimation, (2) evaluating the transformation itself, and (3) selecting the matches that are consistent with it.

A geometric transformation maps a point $\vec{p} = (p_x, p_y)$ to a second point $\vec{p'} = (p'_x, p'_y)$. In the following, we report the most common types of transformations that can be searched.

Each of the following transformations can be used as a filter for a set of candidate matches M. In fact, the subset of matches that are consistent with the evaluated transformation is presumed to be a more reliable set of candidate matches with respect to the original M.

5.3.1 *Hough Transform* ($\mathcal{F}_{HOU}$). Hough transform is used to cluster matches into groups that agree upon a particular model pose (intuitively, the same point of view description of an object). Hough transform identifies clusters of features with a consistent interpretation by using each feature to vote for all object poses that are consistent with the feature [Lowe 2004]. When clusters of features are found that vote for the same pose of an object, the probability of the interpretation being correct is much higher than for any single feature. In our experiments, we create a Hough transform entry predicting the model orientation and scale from the match hypothesis. A pseudorandom hash function is used to insert votes into a one-dimensional hash table in which collisions are easily detected. The Hough transform is typically used to increase the percentage of inliers before estimating a transformation (typically using RANSAC). However, the greater cluster can be considered to be the subset of most relevant matches.

Therefore, we define $\mathcal{F}_{HOU}(M)$ as the subset of candidate matches M that belongs to the greater cluster obtained with the Hough transform. For our experiments, we used the same parameters proposed in Lowe [2004]—that is, bin size of 30 degrees for orientation, a factor of 2 for scale, and 0.25 times the maximum model dimension for location.

Considering the clusters of matches created by the Hough transform, it is possible to estimate a transformation that can map the points of one image onto another.

5.3.2 *RST* ($\mathcal{F}_{RST}$). Rotation, scale, and translation (RST) transformation can be formalized as follows:

$$\begin{bmatrix} p'_x \\ p'_y \end{bmatrix} = \begin{bmatrix} s * cos(\theta) & -sin(\theta) \\ sin(\theta) & s * cos(\theta) \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}, \tag{10}$$

where θ is the angle of the counter clock rotation, s is the scaling, and $\vec{t}$ is the translation. Estimating this transformation requires two pairs of matching points ($\vec{p}$ and $\vec{p'}$).

5.3.3 *Affine* ($\mathcal{F}_{AFF}$). Affine transformation is a linear transformation (rotation, scaling, reflection, and shear) followed by a translation:

$$\begin{bmatrix} p'_x \\ p'_y \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}. \tag{11}$$

Note that an RST transformation is a special case of a general affine transformation. Affine transformation allows one shear mapping and/or reflection in addition to translation, rotation, and scaling [Prince 2012]. A shear transformation leaves all points on one axis fixed, whereas the other points are shifted parallel to the axis by a distance proportional to their perpendicular distance from that axis. Estimating an affine transformation requires three pairs of matching points.

5.3.4 *Homography* ($\mathcal{F}_{HMG}$). Homography is an invertible projective transformation from the real projective plane to the projective plane that maps lines to straight lines. Any two images in the same planar surface in space are related by a homography:

$$w \begin{bmatrix} p'_x \\ p'_y \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}, \tag{12}$$

where w is a scale parameter. Please note that an affine transformation is a special type of general homography whose last row is fixed to $h_{31} = 0, h_{32} = 0, h_{33} = 1$. Estimating this transformation requires four pairs of matching points.

5.3.5 *Isotropic Scaling.* Typically, the coordinates of the points of the local features extraction algorithms are reported in pixels of the image. However, a normalization can improve the effectiveness of the transformation estimation. In this work, we use an isotropic scaling [Hartley 1995] that scales and translates the pixel coordinates so as to bring the centroid of the set to the origin and the average distance from the centroid to $\sqrt{2}$.

5.4 Enhancing Pairwise Image Matching with Geometric Consistency Constraint

Geometric consistency checks can be used when comparing two images still using image distance defined in Equation (5), and by replacing M_σ, with the matches remaining after geometric filtering described earlier.

In the following, we give five options for defining the set of candidate matches M, using M_σ defined in Section 4.3, and the filtering criteria defined in Section 5.3. Used in conjunction with Equation (5), these options result in five similarity functions. Specifically, the five different instantiations for M are as follows:

—Plain distance ratio matches M_σ
—Hough matches $M_{HOU} = \mathcal{F}_{HOU}(M_\sigma)$
—RST matches $M_{RST} = \mathcal{F}_{RST}(M_\sigma)$
—Affine matches $M_{AFF} = \mathcal{F}_{AFF}(M_\sigma)$
—Homography matches $M_{HMG} = \mathcal{F}_{HMG}(M_\sigma)$.

The performance of these similarity functions will be compared later in Section 7.

The BoW approach can also be exploited to define a set of candidate matches that can be used as a basis for the geometric consistency checks. In this scenario, we do not use the cosine distance to calculate similarities between vectors; we directly match the BoW components:

$$\dot{M}(I_i, I_j) = \{\langle l_i, l_j \rangle \mid b_w(l_i) = b_w(l_j)\}. \tag{13}$$

In this case, $\dot{M}$ is used in place of M_σ, employed in the previous section. Similarly as before, we define four sets of candidate matches that are obtained by filtering the set $\dot{M}$ obtained through the BoW approach.

Thus, in our experiments, in addition to cosine TF and cosine TF-IDF, we test the following BoW-based approach:

—Hough matches $\dot{M}_{HOU} = \mathcal{F}_{HOU}(\dot{M})$
—RST matches $\dot{M}_{RST} = \mathcal{F}_{RST}(\dot{M})$
—Affine matches $\dot{M}_{AFF} = \mathcal{F}_{AFF}(\dot{M})$
—Homography matches $\dot{M}_{HMG} = \mathcal{F}_{HMG}(\dot{M})$.

6. *kNN* IMAGE CLASSIFICATION

Document classification has two flavors: as single label and multilabel. In the single-label classification, documents may belong to only one class, whereas in the multilabel classification, documents may belong to more than one class [Korde and Mahender 2012]. In this article, we only consider single-label document classification.

Let S be a database of objects $x$and d the distance function for the objects; let us have a predefined set of *classes* (also known as labels or categories) $C = \{c_1, \ldots, c_m\}$. Single-label document classification [Dudani 1975] is the task of automatically approximating or estimating, by means of a function $\hat{\Phi} : S \to C$, called the *classifier*, an unknown *target function* $\Phi : S \to C$, that defines how documents should be classified.

6.1 Single-Label *kNN* Classification

The single-label distance-weighted *kNN* classifier is one of the most simple and widely used methods for semisupervised learning. It first executes a *kNN* search between the objects of the *training set* T. The training set is the subset $T \subset S$ of data used to fit the classifier, and for which we know the target function Φ. The result of this operation is the set $kNN(x, T)$ of labeled documents belonging to the training set, ordered with respect to the increasing values of the distance function d. The label assigned to the object x by the classifier is the class $c_j \in C$ that maximizes the sum of a similarity between x and the documents labeled c_j in the ranked list $kNN(x, T)$. The similarity between two objects can be calculated as $s = 1-d$ since, without loss of generality, we assume that $0 \leq d \leq 1$ always holds true. The classification task starts by computing the score $z^k(x, c_j)$ for each label $c_j \in C$:

$$z_s^k(x, c_j) = \sum_{y \in kNN(x,T)\,:\,\Phi(y)=c_j} s(x, y)\ . \tag{14}$$

The class that obtains the maximum score is then chosen:

$$\hat{\Phi}_s^k(x, T) = \arg\max_{c_j \in C} z_s^k(x, c_j), \tag{15}$$

where $\hat{\Phi}_s^k(x, T)$ is the *classification function*. All *kNN* classifier algorithms that we present in the next sections share the same basic principle and hence the same classification function. They differ in the way that the confidence of the classification is computed and can be used to decide whether the predicted label has a high probability to be correct. A special case is the local feature based classification, which uses *kNN* classification for the individual local fractures to estimate the confidence of the whole image classification.

6.2 Similarity-Based *kNN* Image Classification

This section discusses how to classify images using a *kNN* classifier relying on pairwise image distances as seen in Section 5. The techniques defined in this section are the baseline approach to image classification using local feature.

This approach, along with approach based on BoW, will be compared against our proposed methods discussed in Sections 6.3 and 6.4.

Given a set of images $\mathcal{I}$ and a predefined set of *classes* $C = \{c_1, \ldots, c_m\}$, the *kNN* classification can be obtained with the function: $\hat{\Phi}^k_s(I_i, \mathcal{I})$ defined in Equation (15), in which in place of the similarity $s = 1 - d$, we can exploit one of the pairwise image distances defined in Section 5. A typical way of evaluating the confidence is 1 minus the ratio between the *score* obtained by the second-best label and the best label—that is,

$$\nu_{doc}\left(\hat{\Phi}^k_s, I_i\right) = 1 - \frac{\arg \max\limits_{c_j \in C \setminus \{\hat{\Phi}^k_s(I_i)\}} z^k(I_i, c_j)}{\arg\max\limits_{c_j \in C} z^k(I_i, c_j)} .$$

This classification confidence can be used to decide whether the predicted label has a high probability of being correct. A value of ν close to one denotes a high confidence.

When the number of classes, and consequently the number of training images, are large, the use of the simple distance based on local feature matching presented in Section 5.1 becomes prohibitive in terms of efficiency, as it implies the sequential scan of the entire training set. In this case, it is useful to introduce some approximations that allow the problem to be managed more efficiently. A widely used solution is that of using the BoW approach presented in Section 5.2.

As already stated, this is just a baseline method to image classification. In the next section, we will propose a new solution that selects only the most promising local feature pairs in images to be matched. This approach can use indexes for efficient similarity search to speed up the classification process. Surprisingly, this method is more efficient and more effective as well, even if just a subset of the local features in images are matched. We call this approach *dataset matching*, as the set of promising pairs are selected by submitting similarity queries on the entire database of local features.

6.3 Dataset Matching

The distance measure defined in Section 5.1 is a direct application of the techniques developed by the computer vision community and requires the direct comparison of each pair of images. In fact, the distances are neither metric nor symmetric, and the complexity of the distance evaluation prevents the use of any sort of indexing. Therefore, given a query, searching for the k nearest images to a given query image requires a complete sequential scan of the archive.

On the other hand, the BoW approach makes the search process much faster than when executing a sequential scan of the training set. However, the quantization introduced by the visual vocabulary reduces the effectiveness of the method.

In this respect, we propose an efficient pairwise image matching that relies on access methods for metric space [Zezula et al. 2006] and at the same time increases the effectiveness, even with respect to the approach discussed in Section 5.1.

Let I_i be the image that we want to classify and $S = \{I_1, \ldots, I_M\}$ the entire training set of images of size M. We propose to retrieve for every local feature l_i of I_i the $\overline{k}$ closest local features from the union Ω of all local features in all images of the training set $\Omega = \cup_i I_i$. We denote the $\overline{k}$ closest local features to l_i as $\overline{k}NN(l_i, \Omega)$, and we call it the set of *candidate matches*.

Since the distance d functions for comparing local features are metric distances (SIFT and SURF use Euclidian distance; ORB and BRISK use Hamming distance), metric [Zezula et al. 2006] or spatial [Samet 2005] access methods can be used to efficiently execute $\overline{k}NN(l_i, \Omega)$.

We define the matches between an image I_i and any $I_j \in S$ as.

$$\bar{M}(I_i, I_j) = \{\langle l_i, l_j \rangle | l_i \in I_i \wedge l_j = NN_1(l_i, \overline{k}NN(l_i, \Omega) \cap I_j)\}. \tag{16}$$

In short, we select the matching local features in two images,only considering the candidate matching local features obtained executing the nearest-neighbor similarity search query $\overline{k}NN(l_i, \Omega)$. Note that $\overline{k}NN(l_i, \Omega)$ need to be executed just once for every local feature l_i of I_i independently of the size M of the training set.

In this scenario, $\bar{M}$ is used in place of the candidate set of matches M_σ, employed in Section 5.1 for evaluating the distance between two images:

$$d_S(I_i, I_j) = 1 - \frac{|\bar{M}(I_i, I_j)|}{|I_i|}. \tag{17}$$

The matching function $\bar{M}$, defined in this section, can also be enhanced by the use of the four geometric filtering criteria, defined in Section 5.4. Thus, in Section 7, we also test the following:

—Plain nearest neighbor matches $\bar{M}$

—Hough matches $\bar{M}_{HOU} = \mathcal{F}_{HOU}(\bar{M})$

—RST matches $\bar{M}_{RST} = \mathcal{F}_{RST}(\bar{M})$

—Affine matches $\bar{M}_{AFF} = \mathcal{F}_{AFF}(\bar{M})$

—Homography matches $\bar{M}_{HMG} = \mathcal{F}_{HMG}(\bar{M})$.

6.4 Local Feature Based *kNN* Classifier

In the previous section, we considered the classification of an image I_i as a process of retrieving the most similar images in the *training set* $\mathcal{T}_l$ and then applying a *kNN* classification technique to predict the class of I_i.

In this section, we propose a different approach that classifies an image I_i in two steps:

(1) each local feature $l_i \in I_i$ is first individually classified considering the local features of all images in the training set $\mathcal{T}_l$; and

(2) the whole image is classified considering the class assigned to each local feature and the confidence of the classification evaluated in step 1.

Note that by classifying the local features individually, such as before assigning a label to an image, we might lose the implicit mutual information between the interest points of an image. However, surprisingly, we will see that this method gives a better performance than the other approaches.

In the next sections, we will define four distinct algorithms for local feature classification (i.e., step 1). All proposed algorithms require searching for similar local features for each of the local features belonging to the image.

6.4.1 *Step 1: Local Feature Classification.* All of the following *kNN* local feature classifiers are applications of the single-label distance-weighted *kNN* discussed in Section 6.1. They make use of a similarity function that can be obtained from the distance measure d between local features by applying the well-known transformation $s = 1 - d/d_{MAX}$.

1NN LF Classifier ($\hat{\Phi}^f$). The simplest way to classify a local feature is to consider the label of its closest neighbor in $\mathcal{T}_l$. The *1NN local features classifier* $\hat{\Phi}^1_s(l_x)$ assigns the label of the closest neighbor in $\mathcal{T}_l$ to a local feature l_x. The confidence of the classification assigned is the similarity between l_i and its nearest neighbor. Formally,

$$\nu\left(\hat{\Phi}_s^1, l_x\right) = s(l_x, NN_1(l_x, \mathcal{T}_l)).$$

Please note that this classifier does not require any parameter to be set. Moreover, the similarity search over the local features training set is a simple 1NN search.

Weighted kNN *LF Classifier* ($\hat{\Phi}^k$). The weighted *kNN* LF classifier is the natural application of the *kNN* classification function $\hat{\Phi}_s^k(l_x, \mathcal{T}_l)$ on local features. The confidence is similarly based on the ratio between second best and best class as follows:

$$\nu\left(\hat{\Phi}_s^k, l_x\right) = 1 - \frac{\arg \max\limits_{c_j \in C \setminus \{\hat{\Phi}_s^k(l_x, \mathcal{T}_l)\}} z_s^k(l_x, c_j)}{\arg\max\limits_{c_i \in C} z_s^k(l_x, c_i)}.$$

Note that for $k = 1$, we degenerate to the 1NN LF classifier case, whereas the measure of confidence is different. In fact, 1NN always assigns 1 as confidence when $k = 1$, whereas the *kNN* LF classifier considers the first nearest-neighbor similarity as a measure of confidence. This classifier requires parameter k to be chosen.

LF matching classifier ($\hat{\Phi}^m$). The *LF matching classifier* decides the candidate label similarly to the 1NN LF classifier (i.e., $\hat{\Phi}_s^1(l_x, \mathcal{T}_l)$), whereas the confidence value of the selected label is evaluated using the idea of the distance ratio discussed in Section 4.3:

$$\nu\left(\hat{\Phi}_s^1, l_x\right) = \begin{cases} 1 \text{ if } \dot{\sigma}(l_x, \mathcal{T}_l) < c \\ 0 \text{ otherwise} \end{cases}.$$

The distance ratio $\dot{\sigma}$ is computed considering the nearest local feature to l_x and the closest local feature that has a label different from the nearest local feature. Following the idea of Lowe explained in Section 4.3, we define the similarity ratio $\dot{\sigma}$ as

$$\dot{\sigma}(l_x, \mathcal{T}_l) = \frac{d(l_x, NN_1(l_x, \mathcal{T}_l))}{d(l_x, NN_2^*(l_x, \mathcal{T}_l))},$$

where $NN_2^*(l_x, \mathcal{T}_l)$ is the closest neighbor known to be labeled differently than the first.

Note that searching for $NN_2^*(l_x, \mathcal{T}_l)$ cannot be directly translated into a standard *kNN* search. However, the *kNN* implementation in metric spaces is generally performed starting with an infinite range and reducing this during the evaluation, considering at any time the current NN_k. The same approach can be used for searching $NN_2^*(l_x, \mathcal{T}_l)$. In fact, although k is not known in advance, the current NN_2^* during the similarity search can be used to reduce the range of the query. Thus, the similarity search needed for the evaluation of $\dot{\sigma}(l_x, T_r)$ can be implemented by slightly modifying the standard algorithms developed for metric spaces (e.g., see Zezula et al. [2006]).

Parameter c used in the definition of the confidence is equivalent to that used in Lowe [2004] and Bay et al. [2006]. We will see in Section 7.3 that $c = 0.8$ proposed in Lowe [2004] is able to guarantee good performance. It is worth noting that c is the only parameter to be set for this classifier considering that the similarity search performed over the local features in $\mathcal{T}_l$ does not require a parameter k to be set.

Weighted LF distance ratio classifier ($\hat{\Phi}^w$). The *weighted LF distance ratio classifier* is an extension of the *LF matching classifier* defined in the previous section. However, the confidence here is not binary, but is a fuzzy measure derived from the distance ratio. Given that the greater the confidence, the better the matching, we define the assigned label and the respective confidence as

$$\nu\left(\hat{\Phi}_s^1, l_x\right) = (1 - \dot{\sigma}(l_x, \mathcal{T}_l))^2.$$

The intuition is that it could be preferable not to filter nonmatching features on the basis of the distance ratio, but to adopt $1 - \dot{\sigma}(l_x, \mathcal{T}_l)$ as a measure of confidence for the classification of the whole image. The value is then squared to emphasize the relative importance of greater distance ratios.

Please note that for this classifier, we do not have to specify either a distance ratio threshold c or k. Thus, this classifier has no parameters.

Weighted LF distance ratio with geometric constraints $\hat{\Phi}_g^w$. It is also possible to combine the classification approach of the weighted LF distance ratio classifier with the geometric consistency filtering power presented in Section 5.3.

First, we perform a nearest-neighbor search for each local feature on the image I_i to be classified. At the end of this process, for each local feature $l_i \in I_i$, we apply geometric consistency filtering to obtain sets of candidate matches for $\langle I_i, I_j \rangle$ image pairs. Finally, we merge the local features pairs $\langle l_i, l_j \rangle$ in all filtered matches $\bar{M}$—that is,

$$\mathcal{M}_g = \bigcup_j \bar{M}_g(I_i, I_j),$$

where g stands for HOU (Hough), RST (rotation, scale, and translation), AFF (affine), and HMG (homography) as explained in Section 5.3. Please note that I_j are at most all the images having at least one feature in $\bar{k}NN(l_i, \mathcal{T}_l)\ \forall l_i \in \mathcal{T}_l$. Furthermore, given that each geometric consistency filter requires a minimum number of points to be applied, the cardinality of I_j typically is much smaller.

The classification process is then performed as for the *weighted LF distance ratio classifier* only considering the filtered set of local features $\mathcal{M}_g$ obtained with one of the specific filters (*HOU*, *RST*, *AFF*, or *HMG*) (i.e., $\hat{\Phi}_s^1(l_x, \mathcal{M})$) and the following confidence:

$$\nu\left(\hat{\Phi}_s^1, l_x\right) = (1 - \dot{\sigma}(l_x, \mathcal{M}))^2.$$

6.4.2 *Step 2: Image Classification.* In the following, we assume that the label of each local feature l_x, belonging to images in the training set $\mathcal{T}_l$, is the label assigned to the image to which it belongs (i.e., I_x):

$$\forall l_x \in I_x,\ \forall I_x \in T, \Phi(l_x) = \Phi(I_x). \tag{18}$$

In other words, we assume that the local features generated over interest points of images in the training set can be labeled as the image to which they belong. Note that the local features classifier can manage the noise introduced by this label propagation from the whole image to the local features. In fact, we will see that when very similar training local features are assigned to different classes, a local feature close to them is classified with a low confidence. The experimental evaluation reported later in Section 7.3 confirms the validity of this assumption.

As already stated, given $l_x \in I_x$, the classifier $\hat{\Phi}$ of step 1 returns both a class $\hat{\Phi}(l_x) = c_i \in C$ to l_x and a numerical value $\nu(\hat{\Phi}, l_x)$ that represents the confidence that $\hat{\Phi}$ has in this decision.

The whole image is classified, given the label $\hat{\Phi}(l_x)$ and the confidence $\nu(\hat{\Phi}, l_x)$ assigned to its local features $l_x \in I_x$ during the first phase, using a confidence-rated majority vote approach. We first compute a score $z(l_x, c_i)$ for each label $c_i \in C$. The score is the sum of the confidences obtained for the local features predicted as c_i. Formally,

$$z(I_x, c_i) = \sum_{l_x \in I_x, \hat{\Phi}(l_x) = c_i} \nu(\hat{\Phi}, l_x).$$

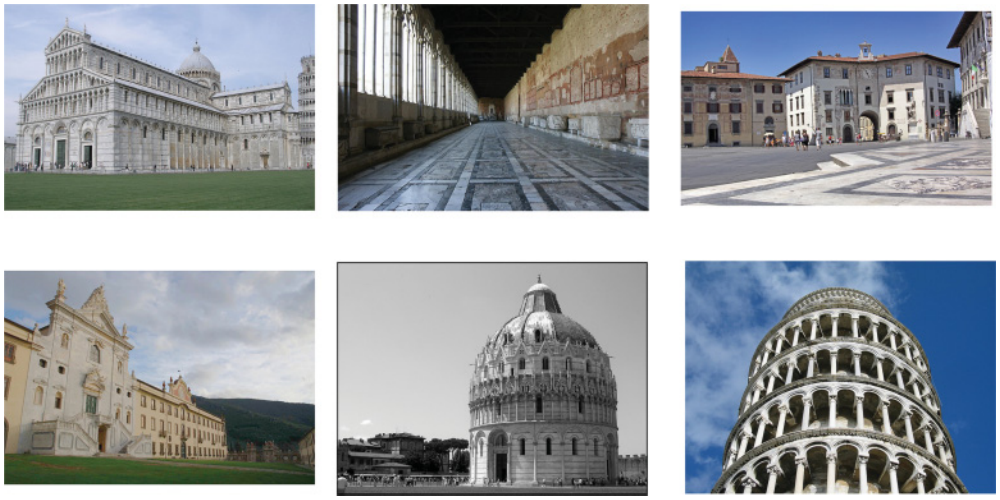


Fig. 1. Example images taken from the Pisa dataset (images available by Flikr under a Creative Commons License agreement).

The label that obtains the maximum score is then chosen:

$$\hat{\Phi}(I_x) = \arg\max_{c_j \in C} z(I_x, c_j).$$

As a measure of confidence for the classification of the whole image, we use the ratio between the predicted and the second-best class:

$$\nu_{img}(\hat{\Phi}, I_x) = 1 - \frac{\arg\max_{c_j \in C - \hat{\Phi}(l_x)} z(I_x, c_j)}{\arg\max_{c_i \in C} z(I_x, c_i)}.$$

This whole image classification confidence can be used to decide whether the predicted label has a high probability of being correct.

7. EXPERIMENTAL EVALUATION

The aim of this performance analysis is to evaluate the classification effectiveness of the different strategies of *kNN* classification combined with various types of local features with and without geometric consistency checks.

7.1 Dataset, Ground Truth, and Experiment Settings

The dataset that we used for our tests is publicly available and composed of 1,227 photos of 12 monuments or cultural heritage related landmarks located in Pisa.

It was created during the VISITO Tuscanyproject[1] and was also used in Amato et al. [2010] and Amato and Falchi [2010, 2011]. The photos have been crawled from Flickr, the well-known online photo service. The IDs of the photos used for these experiments together with the assigned label and extracted features can be downloaded from VISITO [2011]. In the following, we list the classes that we used and the number of photos belonging to each class. In Figure 1, we report an example for each class in the same order as they are reported in the following list:

[1] http://www.visitotuscany.it/.

(1) *Battistero* (104 photos)—the baptistery of St. John
(2) *Camposanto Monumentale (exterior)* (46 photos)
(3) *Camposanto Monumentale (portico)* (138 photos)
(4) *Camposanto Monumentale (field)* (113 photos)
(5) *Certosa* (53 photos)—the charterhouse
(6) *Chiesa della Spina* (112 photos)—Gothic church
(7) *Guelph tower* (71 photos)
(8) *Duomo* (130 photos)—the cathedral of St. Mary
(9) *Palazzo dell'Orologio* (92 photos)—building
(10) *Basilica of San Piero* (48 photos)—church of St. Peter
(11) *Palazzo della Carovana* (101 photos)—building
(12) *Leaning Tower* (119 photos)—leaning campanile

To build and evaluate a classifier for these classes, we divided the dataset into a *training set* ($\mathcal{T}_l$) consisting of 226 photos (approximately 20% of the dataset) and a *test set* consisting of 921 photos (approximately 80% of the dataset). The image resolution used for feature extraction is the standard resolution used by Flickr (maximum 500 pixels for either the height or width).

The total number of local features extracted by the SIFT and SURF detectors were about 1,000,000 and 500,000, respectively. The number of local features per image varies between 113 and 2,816 for SIFT and 50 and 904 for SURF. ORB was tested setting the feature extractor to identify both 500 and 1,000 local features. The number of local features detected for BRISK was less than 500.

Various classifiers were created using the local features taken into considerations and the definitions given in Section 6.

7.2 Performance Measures

To evaluate the effectiveness of the classifiers on the *test set*, we use the microaveraged *accuracy* and micro- and macroaveraged *precision*, *recall*, and F_1.

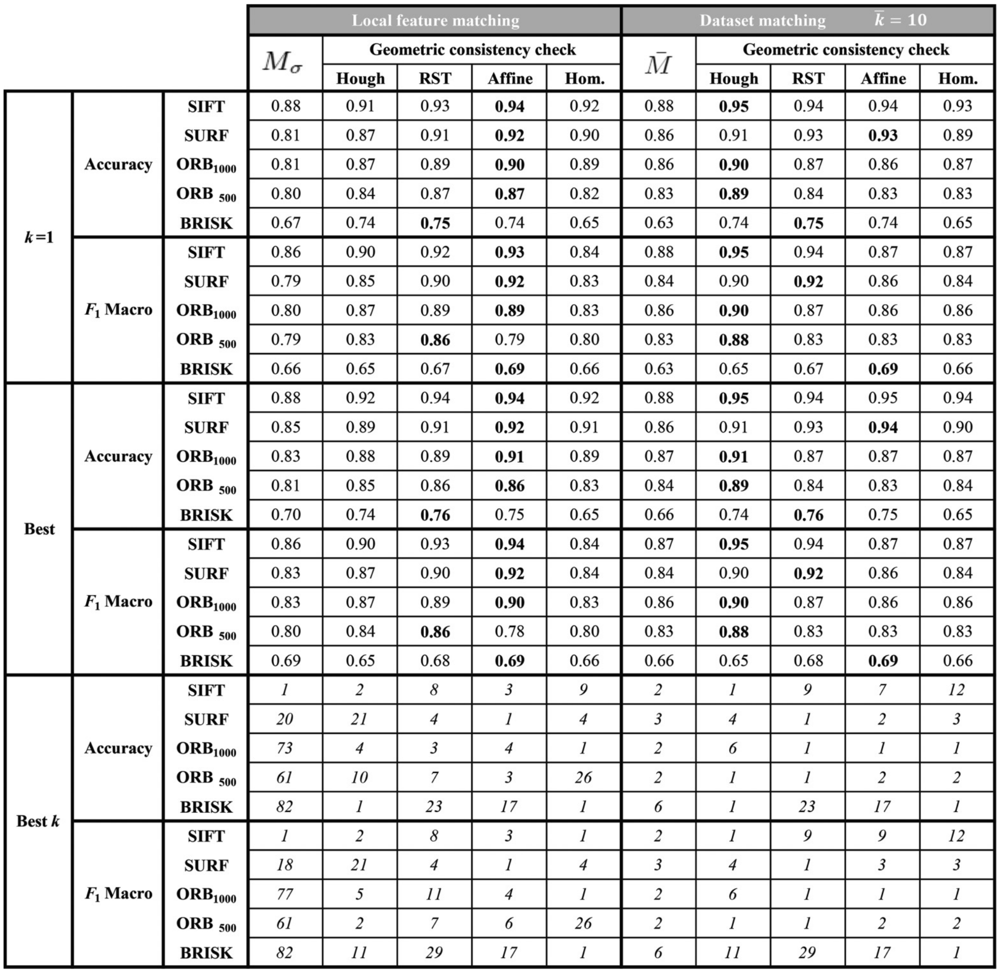
In macroaveraging, the performance metrics are calculated for each class and then the average of all is evaluated. In microaveraging, the average is calculated across all individual classification decisions made by a system [Chau and Chen 2008].

Precision is defined as the ratio between the number of correctly predicted and the overall number of predicted documents for a specific class. *Recall* is the ratio between the number of correctly predicted and the overall number of documents for a specific class. F_1 is the harmonic mean of *precision* and *recall*.

Note that for the *single-label* classification task, microaveraged *accuracy* is defined as the number of documents correctly classified divided by the total number of documents of the same label in the *test set* and is equivalent to the microaveraged *precision*, *recall*, and F_1 scores. Therefore, in the tables discussed in the following, we only report the values of *accuracy* and F_1 Macro.

7.3 Similarity-Based Image *kNN* Classification Results

In Figure 2, we report the results obtained by both local feature matching (see Section 5.1) and dataset matching (see Section 6.3). Given that the *kNN* classifier requires the parameter *k*, we report the results obtained for $k = 1$ (row labeled $k = 1$), the best results obtained varying $k \in [1, 100]$ (row labeled Best), and the value *k* at which the best result was obtained (row labeled Best *k*). Figure 3 reports results obtained by using the BoW approach (see Section 5.2). The details are discussed in the following sections.



			Local feature matching					Dataset matching $\bar{k} = 10$				
			M_σ	Geometric consistency check				$\bar{M}$	Geometric consistency check			
				Hough	RST	Affine	Hom.		Hough	RST	Affine	Hom.
k=1	Accuracy	SIFT	0.88	0.91	0.93	**0.94**	0.92	0.88	**0.95**	0.94	0.94	0.93
		SURF	0.81	0.87	0.91	**0.92**	0.90	0.86	0.91	0.93	**0.93**	0.89
		ORB_{1000}	0.81	0.87	0.89	**0.90**	0.89	0.86	**0.90**	0.87	0.86	0.87
		ORB_{500}	0.80	0.84	0.87	**0.87**	0.82	0.83	**0.89**	0.84	0.83	0.83
		BRISK	0.67	0.74	**0.75**	0.74	0.65	0.63	0.74	**0.75**	0.74	0.65
	F_1 Macro	SIFT	0.86	0.90	0.92	**0.93**	0.84	0.88	**0.95**	0.94	0.87	0.87
		SURF	0.79	0.85	0.90	**0.92**	0.83	0.84	0.90	**0.92**	0.86	0.84
		ORB_{1000}	0.80	0.87	0.89	**0.89**	0.83	0.86	**0.90**	0.87	0.86	0.86
		ORB_{500}	0.79	0.83	**0.86**	0.79	0.80	0.83	**0.88**	0.83	0.83	0.83
		BRISK	0.66	0.65	0.67	**0.69**	0.66	0.63	0.65	0.67	**0.69**	0.66
Best	Accuracy	SIFT	0.88	0.92	0.94	**0.94**	0.92	0.88	**0.95**	0.94	0.95	0.94
		SURF	0.85	0.89	0.91	**0.92**	0.91	0.86	0.91	0.93	**0.94**	0.90
		ORB_{1000}	0.83	0.88	0.89	**0.91**	0.89	0.87	**0.91**	0.87	0.87	0.87
		ORB_{500}	0.81	0.85	0.86	**0.86**	0.83	0.84	**0.89**	0.84	0.83	0.84
		BRISK	0.70	0.74	**0.76**	0.75	0.65	0.66	0.74	**0.76**	0.75	0.65
	F_1 Macro	SIFT	0.86	0.90	0.93	**0.94**	0.84	0.87	**0.95**	0.94	0.87	0.87
		SURF	0.83	0.87	0.90	**0.92**	0.84	0.84	0.90	**0.92**	0.86	0.84
		ORB_{1000}	0.83	0.87	0.89	**0.90**	0.83	0.86	**0.90**	0.87	0.86	0.86
		ORB_{500}	0.80	0.84	**0.86**	0.78	0.80	0.83	**0.88**	0.83	0.83	0.83
		BRISK	0.69	0.65	0.68	**0.69**	0.66	0.66	0.65	0.68	**0.69**	0.66
Best k	Accuracy	SIFT	*1*	*2*	*8*	*3*	*9*	*2*	*1*	*9*	*7*	*12*
		SURF	*20*	*21*	*4*	*1*	*4*	*3*	*4*	*1*	*2*	*3*
		ORB_{1000}	*73*	*4*	*3*	*4*	*1*	*2*	*6*	*1*	*1*	*1*
		ORB_{500}	*61*	*10*	*7*	*3*	*26*	*2*	*1*	*1*	*2*	*2*
		BRISK	*82*	*1*	*23*	*17*	*1*	*6*	*1*	*23*	*17*	*1*
	F_1 Macro	SIFT	*1*	*2*	*8*	*3*	*1*	*2*	*1*	*9*	*9*	*12*
		SURF	*18*	*21*	*4*	*1*	*4*	*3*	*4*	*1*	*3*	*3*
		ORB_{1000}	*77*	*5*	*11*	*4*	*1*	*2*	*6*	*1*	*1*	*1*
		ORB_{500}	*61*	*2*	*7*	*6*	*26*	*2*	*1*	*1*	*2*	*2*
		BRISK	*82*	*11*	*29*	*17*	*1*	*6*	*11*	*29*	*17*	*1*

Fig. 2. Similarity-based image *kNN* classification results using the local feature and dataset matches for $\bar{k} = 10$.

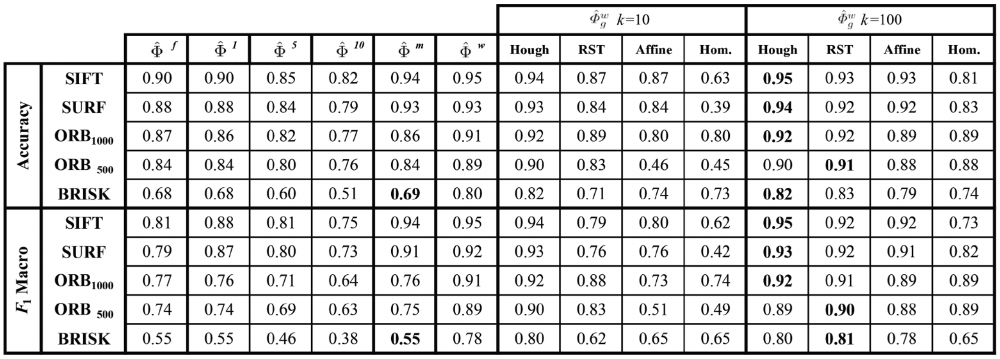
7.3.1 *Local Feature Matching.* Comparing the results obtained by the various similarity functions for the local feature matching comparison approach, we can see that geometric consistency checks are able to significantly improve the quality of the classification process. The best performance was generally obtained using the distance function that makes use of the affine geometric constraint. Only ORB with 500 local features and BRISK sometimes obtain a better F_1 Macro, when using RTS geometric constraint checks. Overall, SIFT provides the highest effectiveness, achieving an *accuracy* and F_1 Macro of 0.94, with $k = 3$. However, ORB and BRISK have a more compact size and are easier to manage, given that they are binary features. Thus, even if their effectiveness is a little lower, their usage can be justified in terms of efficiency.

7.3.2 *Dataset Matching.* In Figure 2, we also report the results obtained by the dataset matching approach using $\bar{k} = 10$—that is, performing a 10 nearest neighbors search for each local feature in the query over the local features in the training set. In our experiments, we also tested $\bar{k} = 30, 50,$ and 100, obtaining comparable but worse results. We note that a peculiar feature of this approach

			Bag of Words					
			cosine	cosine TF-IDF	Geometric consistency check			
					Hough	RST	Affine	Hom.
$k=1$	Accuracy	SIFT	0.86	0.87	**0.88**	0.88	0.81	0.88
		SURF	0.85	0.84	0.85	0.85	0.82	**0.86**
		ORB$_{1000}$	0.78	0.79	0.78	0.78	0.75	**0.81**
		ORB $_{500}$	0.75	0.76	0.77	0.77	0.77	**0.82**
		BRISK	0.59	**0.63**	0.52	0.59	0.52	0.59
	F_1 Macro	SIFT	**0.88**	0.87	0.87	0.87	0.80	0.80
		SURF	**0.84**	0.83	0.75	0.75	0.72	0.79
		ORB$_{1000}$	0.77	0.78	0.77	0.77	0.75	**0.80**
		ORB $_{500}$	0.73	0.74	**0.76**	0.76	0.76	0.74
		BRISK	0.58	0.62	0.51	0.48	0.44	0.51
Best	Accuracy	SIFT	0.88	**0.90**	0.88	0.88	0.83	0.89
		SURF	0.86	0.86	**0.88**	0.88	0.83	0.86
		ORB$_{1000}$	0.79	0.81	0.80	0.82	0.81	**0.84**
		ORB $_{500}$	0.78	0.78	0.79	0.82	0.81	**0.84**
		BRISK	0.61	**0.65**	0.53	0.63	0.59	0.61
	F_1 Macro	SIFT	0.87	**0.87**	0.87	0.87	0.82	0.81
		SURF	0.84	**0.85**	0.78	0.78	0.76	0.79
		ORB $_{500}$	0.78	0.80	0.79	0.80	0.80	**0.82**
		ORB$_{1000}$	0.73	0.77	0.78	0.76	**0.79**	0.76
		BRISK	0.59	**0.64**	0.52	0.52	0.49	0.53
Best k	Accuracy	SIFT	*7*	*8*	*2*	*4*	*4*	*6*
		SURF	*3*	*9*	*15*	*7*	*13*	*2*
		ORB$_{1000}$	*25*	*22*	*86*	*83*	*48*	*70*
		ORB $_{500}$	*28*	*33*	*38*	*90*	*74*	*79*
		BRISK	*33*	*17*	*29*	*64*	*76*	*50*
	F_1 Macro	SIFT	*3*	*3*	*1*	*2*	*4*	*6*
		SURF	*7*	*9*	*15*	*7*	*13*	*2*
		ORB $_{500}$	*25*	*22*	*86*	*84*	*48*	*94*
		ORB$_{1000}$	*28*	*33*	*38*	*91*	*69*	*79*
		BRISK	*17*	*17*	*26*	*24*	*31*	*44*

Fig. 3. Classification results using the BoW approach with a vocabulary of 100k features.

is that it relies on spatial or metric access methods for similarity searching to significantly improve efficiency of classification with very large training sets. Notably, this approach often performs better than the local feature matching approach. The intuition to justify this behavior is that the *kNN* search performed between all local features in the training set is able to reduce the number of false matches. The best results are obtained using Hough and RTS geometric constraint checks, with an *accuracy* and F_1 Macro of $0,95$ and $0,94$, respectively, using SIFT. Hough obtains the best performance using $k = 1$,



								$\hat{\Phi}^w_g$ k=10				$\hat{\Phi}^w_g$ k=100			
		$\hat{\Phi}^f$	$\hat{\Phi}^1$	$\hat{\Phi}^5$	$\hat{\Phi}^{10}$	$\hat{\Phi}^m$	$\hat{\Phi}^w$	**Hough**	**RST**	**Affine**	**Hom.**	**Hough**	**RST**	**Affine**	**Hom.**
Accuracy	**SIFT**	0.90	0.90	0.85	0.82	0.94	0.95	0.94	0.87	0.87	0.63	**0.95**	0.93	0.93	0.81
	SURF	0.88	0.88	0.84	0.79	0.93	0.93	0.93	0.84	0.84	0.39	**0.94**	0.92	0.92	0.83
	ORB$_{1000}$	0.87	0.86	0.82	0.77	0.86	0.91	0.92	0.89	0.80	0.80	**0.92**	0.92	0.89	0.89
	ORB $_{500}$	0.84	0.84	0.80	0.76	0.84	0.89	0.90	0.83	0.46	0.45	0.90	**0.91**	0.88	0.88
	BRISK	0.68	0.68	0.60	0.51	**0.69**	0.80	0.82	0.71	0.74	0.73	**0.82**	0.83	0.79	0.74
F_1 Macro	**SIFT**	0.81	0.88	0.81	0.75	0.94	0.95	0.94	0.79	0.80	0.62	**0.95**	0.92	0.92	0.73
	SURF	0.79	0.87	0.80	0.73	0.91	0.92	0.93	0.76	0.76	0.42	**0.93**	0.92	0.91	0.82
	ORB$_{1000}$	0.77	0.76	0.71	0.64	0.76	0.91	0.92	0.88	0.73	0.74	**0.92**	0.91	0.89	0.89
	ORB $_{500}$	0.74	0.74	0.69	0.63	0.75	0.89	0.90	0.83	0.51	0.49	0.89	**0.90**	0.88	0.89
	BRISK	0.55	0.55	0.46	0.38	**0.55**	0.78	0.80	0.62	0.65	0.65	0.80	**0.81**	0.78	0.65

Fig. 4. Local feature based classifier results.

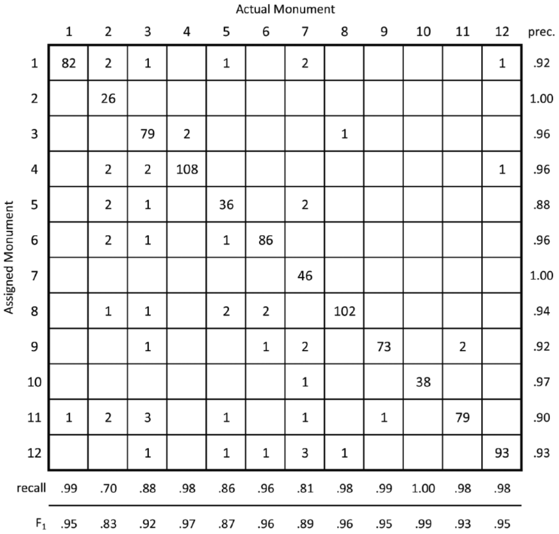
and RTS needs $k = 9$. As executing Hough transformation costs much less than RST, Hough is the best choice in this case. As previously, SIFT is the local feature that offers the best results in this case.

7.3.3 *Bag of Words.* In Figure 3, we report the results obtained by the BoW approach described in Section 5.2, using a vocabulary of 100k features selected using the k-means algorithm. As established in the literature, typically the more the words, the better the results. In our experiments, we are dealing with a dataset of about one million features. Thus, 100k of visual words is the highest value for which it makes sense to perform a clustering algorithm. The results in this case are worse than those obtained with our proposed approaches discussed in Sections 7.3.1 and 7.3.2. In fact, both *accuracy* and F_1 Macro never exceed 0.9. Moreover, the geometric consistency checks do not significantly improve performance; this is particularly true for F_1. The intuition is that the candidate matches found using the BoW approach are much too noisy. Standard cosine and TF-IDF similarity measures are more suitable for this scenario. It is worth noting that the k-means algorithm for selecting the 100k words was executed over the whole dataset, although it would have been more correct to only consider the training images. In fact, the test images should not be used during any training phase. However, we preferred to compare our approach in this scenario, even if the BoW performance is actually overestimated.

7.4 Local Feature Based Image Classifier Results

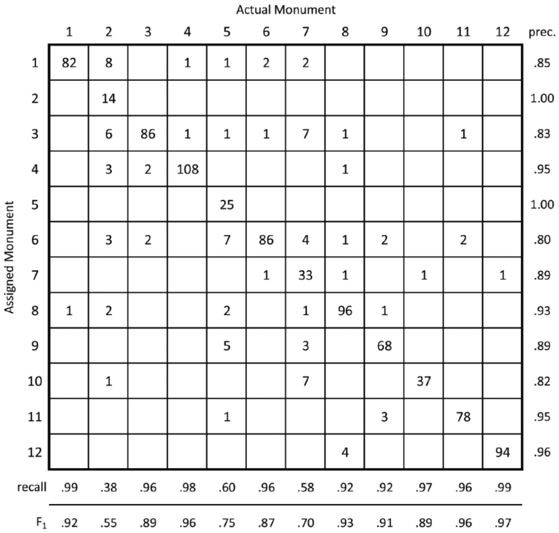
Figure 4 reports the results obtained with the local feature based classifier (see Section 6.4). Similarly to the approach based on image to local features matching, this approach also allows us to significantly improve efficiency, relying on metric or spatial access methods for similarity searching. In fact, local features can be classified using distance functions that can be easily indexed using these access methods.

Experiments show that very good results are obtained even without geometric constraint checks. For instance, using SIFT, we obtained values of accuracy and F_1 Macro of 0.95 simply with the weighted local feature classifier. In just a few cases, Hough transformation slightly improves the performance. However, the improvement obtained does not justify the extra efficiency cost involved. For instance, for all binary local features, improvements range from 0.01 to 0.02 in both accuracy and F_1 Macro using Hough with respect to the simple weighted local feature classifier.



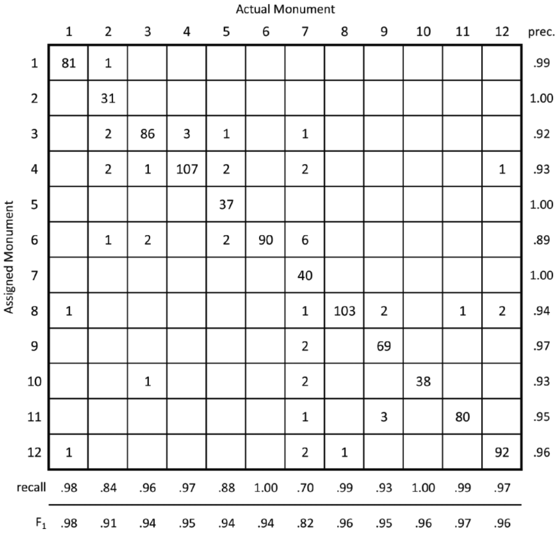
		Actual Monument												
		1	2	3	4	5	6	7	8	9	10	11	12	prec.
Assigned Monument	1	82	2	1		1		2					1	.92
	2		26											1.00
	3			79	2				1					.96
	4		2	2	108								1	.96
	5		2	1		36		2						.88
	6		2	1		1	86							.96
	7							46						1.00
	8		1	1		2	2		102					.94
	9			1			1	2		73		2		.92
	10							1			38			.97
	11	1	2	3		1		1		1		79		.90
	12			1		1	1	3	1				93	.93
	recall	.99	.70	.88	.98	.86	.96	.81	.98	.99	1.00	.98	.98	
	F_1	.95	.83	.92	.97	.87	.96	.89	.96	.95	.99	.93	.95	

Fig. 5. Confusion matrix obtained by the *local features matching* with *affine* geometric consistency check and $k = 3$. Overall $acc = 0.94$ and $F_1 Macro = 0.93$.



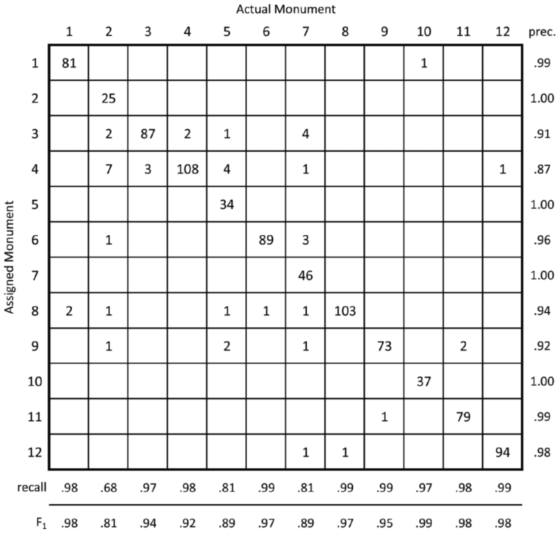
		Actual Monument												
		1	2	3	4	5	6	7	8	9	10	11	12	prec.
Assigned Monument	1	82	8		1	1	2	2						.85
	2		14											1.00
	3		6	86	1	1	1	7	1			1		.83
	4		3	2	108				1					.95
	5					25								1.00
	6		3	2		7	86	4	1	2		2		.80
	7						1	33	1		1		1	.89
	8	1	2			2		1	96	1				.93
	9					5		3		68				.89
	10		1					7			37			.82
	11					1				3		78		.95
	12								4				94	.96
	recall	.99	.38	.96	.98	.60	.96	.58	.92	.92	.97	.96	.99	
	F_1	.92	.55	.89	.96	.75	.87	.70	.93	.91	.89	.96	.97	

Fig. 6. Confusion matrix obtained by the BoW approach using cosine and TF-IDF and $k = 8$. Overall $acc = 0.90$ and $F_1 Macro = 0.87$.



		Actual Monument												
		1	2	3	4	5	6	7	8	9	10	11	12	prec.
Assigned Monument	1	81	1											.99
	2		31											1.00
	3		2	86	3	1		1						.92
	4		2	1	107	2		2					1	.93
	5					37								1.00
	6		1	2		2	90	6						.89
	7							40						1.00
	8	1						1	103	2		1	2	.94
	9							2		69				.97
	10			1				2			38			.93
	11							1		3		80		.95
	12	1						2	1				92	.96
	recall	.98	.84	.96	.97	.88	1.00	.70	.99	.93	1.00	.99	.97	
	F_1	.98	.91	.94	.95	.94	.94	.82	.96	.95	.96	.97	.96	

Fig. 7. Confusion matrix obtained by the *dataset matching* approach with *Hough* geometric consistency check and $k = 1$. Overall $acc = 0.95$ and $F_1 Macro = 0.95$.



		Actual Monument												
		1	2	3	4	5	6	7	8	9	10	11	12	prec.
Assigned Monument	1	81									1			.99
	2		25											1.00
	3		2	87	2	1		4						.91
	4		7	3	108	4		1					1	.87
	5					34								1.00
	6		1				89	3						.96
	7							46						1.00
	8	2	1			1	1	1	103					.94
	9		1			2		1		73		2		.92
	10										37			1.00
	11									1		79		.99
	12							1	1				94	.98
	recall	.98	.68	.97	.98	.81	.99	.81	.99	.99	.97	.98	.99	
	F_1	.98	.81	.94	.92	.89	.97	.89	.97	.95	.99	.98	.98	

Fig. 8. Confusion matrix obtained by the *weighted LF distance ratio classifier* $\hat{\Phi}^w$. Overall $acc = 0.95$ and $F_1 Macro = 0.95$.

Overall, the performance of this approach is comparable to the best results obtained by the approach based on image to local feature matching. However, its efficiency is higher, as it does not require geometric constraint checks to be performed.

To compare per-class results, in Figure 5, 6, 7, and 8 we report the confusion matrices for the most relevant classifiers tested according to the results reported in the previous sections. All results were obtained using SIFT to be comparable. We report actual classes by column and the assigned ones by row. Each monument is indicated by the number used in Section 7.1 for describing the dataset. The last rows show specific monument recall and $F_1 Macro$, whereas in the last column we report the precision. The overall more difficult to recognize monuments were *Camposanto Monumentale* (2), *Certosa* (5), and *Guelph tower* (7). Comparing the matrices, we see major variations on the relative and absolute performance obtained by the various approaches on (2) and (7). For instance, *dataset matching* (Figure 7) and *weighted LF distance ratio classifier* $\hat{\Phi}^w$ (Figure 8) have overall similar performance, but they obtained significantly different results in these two classes.

8. CONCLUSIONS

In this article, we have developed several strategies for efficient landmark recognition, which combine two different approaches to *kNN* classification applying different methods to match local descriptors.

The results of the experiments, conducted in a cultural heritage scenario, revealed that the proposed approaches gave better performance than other state-of-the-art approaches.

Among the techniques that we proposed, the local feature based classifier gave the best performance. With this classifier, we can improve efficiency by using metric or spatial access methods. In addition, the effectiveness provided is generally equal to or better than the other methods. The great advantage of this method is that it offers high performance even without geometric consistency checks, thus further raising efficiency. Comparisons were executed using various types of local features. The best performance was always obtained using SIFT. Although binary features (ORB and BRISK) were generally slightly worse, they can further boost efficiency, given their compactness and convenience for mobile applications.

A system built with the proposed image recognition approach is mainly intended to be used by visitors (tourists) of cities with cultural heritage related landmarks (i.e., using a smartphone) to recognize and get information on monuments that they see. Clearly, these techniques can also be used to build systems to be used by researchers to retrieve information on artworks that are mainly described by their visual appearance. In this respect, we are using these techniques to provide access to databases of ancient inscriptions and epigraphy in the European Union funded EAGLE project [EAGLE 2014]. The traditional way of retrieving information from an epigraphic database is, for instance, that of submitting text queries related to place where the item has been found, or where it currently stored. Using our techniques, it is possible to retrieve information by simply using a picture of the epigraph as a query.

REFERENCES

Giuseppe Amato and Fabrizio Falchi. 2010. kNN based image classification relying on local feature similarity. In *Proceedings of the 3rd International Conference on Similarity Search and Applications (SISAP'10)*. ACM, New York, NY, 101–108.

Giuseppe Amato and Fabrizio Falchi. 2011. Local feature based image similarity functions for kNN classfication. In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence (ICAART'11)*. 157–166.

Giuseppe Amato, Fabrizio Falchi, and Claudio Gennaro. 2011. Geometric consistency checks for kNN based image classification relying on local features. In *Proceedings of the 4th International Conference on Similarity Search and Applications (SISAP'11)*. ACM, New York, NY, 81–88. DOI:http://dx.doi.org/10.1145/1995412.1995428

Giuseppe Amato, Fabrizio Falchi, and Paolo Bolettieri. 2010. Recognizing landmarks using automated classification techniques: An evaluation of various visual features. In *Proceedings of the 2nd International Conference on Advances in Multimedia (MMEDIA'10)*. IEEE, Los Alamitos, CA, 78–83.

Giuseppe Amato, Paolo Bolettieri, Fabrizio Falchi, and Claudio Gennaro. 2013. In *Similarity Search and Applications*. Lecture Notes in Computer Science, Vol. 8199. Springer, 245–256.

Herbert Bay and Luc Van Gool. 2006. SURF: Speeded Up Robust Features. Retrieved June 14, 2015, from http://www.vision.ee.ethz.ch/~surf/.

Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. 2006. SURF: Speeded Up Robust Features. In *Computer Vision—ECCV 2006*. Lecture Notes in Computer Science, Vol. 3951. Springer, 404–417.

Aurélien Bellet, Amaury Habrard, and Marc Sebban. 2013. A survey on metric learning for feature vectors and structured data. arXiv:1306.6709.

Oren Boiman, Eli Shechtman, and Michal Irani. 2008. In defense of nearest-neighbor based image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. 1–8.

Anna Bosch, Andrew Zisserman, and Xavier Munoz. 2008. Scene classification using a hybrid generative/discriminative approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 4, 712–727.

Michael Chau and Hsinchun Chen. 2008. A machine learning approach to Web page filtering using content and structure analysis. *Decision Support Systems* 44, 2, 482–494.

Tao Chen, Kui Wu, Kim-Hui Yap, Zhen Li, and Flora S. Tsai. 2009. A survey on mobile landmark recognition for information retrieval. In *Proceedings of the 10th International Conference on Mobile Data Management Systems, Services, and Middleware (MDM'09)*. 625–630.

Jean-Pierre Chevallet, Joo-Hwee Lim, and Mun-Kew Leong. 2007. Object identification and retrieval from efficient image matching. Snap2Tell with the STOIC dataset. *Information Processing and Management* 43, 2, 515–530.

Thomas Cover and Phil Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13, 1, 21–27.

Sahibsingh Dudani. 1975. The distance-weighted k-nearest-neighbour rule. *IEEE Transactions on Systems, Man and Cybernetics* SMC-6, 4, 325–327.

EAGLE. 2014. Europeana Eagle Project. Retrieved June 14, 2015, from http://www.eagle-network.eu/.

Tiziano Fagni, Fabrizio Falchi, and Fabrizio Sebastiani. 2010. Image classification via adaptive ensembles of descriptor-specific classifiers. *Pattern Recognition and Image Analysis* 20, 1, 21–28.

Martin A. Fischler and Robert C. Bolles. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24, 6, 381–395.

Andrea Frome, Yoram Singer, and Jitendra Malik. 2007. Image retrieval and classification using local distance functions. In *Proceedings of the 2006 Conference on Advances in Neural Information Processing Systems*, Vol. 19. 417.

Nicols Garca-Pedrajas and Domingo Ortiz-Boyer. 2009. Boosting k-nearest neighbor classifier by means of input space projection. *Expert Systems with Applications* 36, 7, 10570–10582.

Google. 2010. Search for Pictures with Google Goggles. Retrieved June 14, 2015, from http://www.google.com/mobile/goggles/.

Kristen Grauman and Trevor Darrell. 2007. The pyramid match kernel: Efficient learning with sets of features. *Journal of Machine Learning Research* 8, 725–760.

Daniel Haase and Joachim Denzler. 2011. Comparative evaluation of human and active appearance model based tracking performance of anatomical landmarks in locomotion analysis. In *Proceedings of the 8th Open German-Russian Workshop Pattern Recognition and Image Understanding (OGRW'11)*. 96–99.

Richard I. Hartley. 1995. In defense of the eight-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision (ICCV'95)*. IEEE, Los Alamitos, CA, 1064.

James Hays and Alexei A. Efros. 2008. IM2GPS: Estimating geographic information from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. 1–8. DOI:http://dx.doi.org/10.1109/CVPR.2008.4587784

Jared Heinly, Enrique Dunn, and Jan-Michael Frahm. 2012. Comparative evaluation of binary features. In *Computer Vision—ECCV 2012*. Lecture Notes in Computer Science, Vol. 7573. Springer, 759–773. DOI:http://dx.doi.org/10.1007/978-3-642-33709-3_54

Stefan Hinterstoisser, Vincent Lepetit, Selim Benhimane, Pascal Fua, and Nassir Navab. 2011. Learning real-time perspective patch rectification. *International Journal of Computer Vision* 91, 1, 107–130.

Herve Jegou, Matthijs Douze, Cordelia Schmid, and Patrick Perez. 2010. Aggregating local descriptors into a compact image representation. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*. 3304–3311.

Edward Johns and Guang-Zhong Yang. 2011. From images to scenes: Compressing an image cluster into a single scene model for place recognition. In *Proceedings of the 2011 IEEE International Conference on Computer Vision (ICCV'11)*. 874–881.

Vandana Korde and C. Namrata Mahender. 2012. Text classification and classifiers: A survey. *International Journal of Artificial Intelligence and Applications* 3, 2, 85–99.

Mathieu Labbé. 2014. Find-Object. Retrieved June 15, 2015, from http://introlab.github.io/find-object/.

Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. 2011. BRISK: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 IEEE International Conference on Computer Vision (ICCV'11)*. IEEE, Los Alamitos, CA, 2548–2555.

Miguel Lourenço. 2011. Local Invariant Features. Retrieved June 15, 2015, from http://arthronav.isr.uc.pt/∼mlourenco/files/tutorial.pdf.

David G. Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60, 2, 91–110.

David Lowe. 2005. Demo Software: SIFT Keypoint Detector. Retrieved June 14, 2015, from http://www.cs.ubc.ca/∼lowe/keypoints/.

Shyjan Mahamud and Martial Hebert. 2003. The optimal distance measure for object detection. In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1. IEEE, Los Alamitos, CA, I-248–I-255.

Tomasz Malisiewicz and Alexei A. Efros. 2008. Recognition by association via learning per-exemplar distances. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. 1–8.

Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. 2004. Robust wide-baseline stereo from Maximally Stable Extremal Regions. *Image and Vision Computing* 22, 10, 761–767.

Mahmoud Mejdoub and Chokri Ben Amar. 2011. Classification improvement of local feature vectors over the KNN algorithm. *Multimedia Tools and Applications* 64, 1, 197–218. http://dx.doi.org/10.1007/s11042-011-0900-4

Krystian Mikolajczyk and Cordelia Schmid. 2005. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 10, 1615–1630.

Krystian Mikolajczyk, Bastian Leibe, and Bernt Schiele. 2005. Local features for object class recognition. In *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV'05)*, Vol. 2. IEEE, Los Alamitos, CA, 1792–1799.

Timo Ojala, Matti Pietikainen, and Topi Maenpaa. 2002. Multiresolution gray-scale and rotation invariant texture classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24, 7, 971–987.

Florent Perronnin and Christopher Dance. 2007. Fisher kernels on visual vocabularies for image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*. 1–8.

James Philbin, Ondjiriej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. 2007. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*. 1–8.

James Philbin. 2010. *Scalable Object Retrieval in Very Large Image Collections*. Ph.D. Dissertation. University of Oxford.

Paolo Piro, Richard Nock, Wafa Bel Haj Ali, Frank Nielsen, and Michel Barlaud. 2013. Boosting k-nearest neighbors classification. In *Advanced Topics in Computer Vision*. Advances in Computer Vision and Pattern Recognition, Vol. 2013. Springer, 341–375.

Adrian Popescu and Pierre-Alain Moëllic. 2009. MonuAnno: Automatic annotation of georeferenced landmarks images. In *Proceedings of the ACM International Conference on Image and Video Retrieval (CIVR'09)*. ACM, New York, NY, Article No. 11.

Simon J. D. Prince. 2012. *Computer Vision: Models, Learning, and Inference.* Cambridge University Press, New York, NY.

Milos Radovanović. 2010. Nearest neighbors in high-dimensional data: The emergence and influence of hubs. *Journal of Machine Learning Research* 11, 2487–2531.

Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *Proceedings of the 2011 IEEE International Conference on Computer Vision (ICCV'11)*. IEEE, Los Alamitos, CA, 2564–2571.

Hanan Samet. 2005. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, San Francisco, CA.

Josef Sivic and Andrew Zisserman. 2003. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV'03)*, Vol. 2. IEEE, Los Alamitos, CA, 1470.

Yu-Chuan Su, Tzu-Hsuan Chiu, Guan-Long Wu, Chun-Yen Yeh, Felix Wu, and Winston Hsu. 2013. Flickr-tag prediction using multi-modal fusion and meta Information. In *Proceedings of the 21st ACM International Conference on Multimedia (MM'13)*. ACM, New York, NY, 353–356.

Radu Timofte, Tinne Tuytelaars, and Luc Gool. 2013. Naive Bayes image classification: Beyond nearest neighbors. In *Computer Vision ACCV 2012*. Lecture Notes in Computer Science, Vol. 7724. Springer, 689–703. http://dx.doi.org/10.1007/978-3-642-37331-2_52

Pierre Tirilly, Vincent Claveau, and Patrick Gros. 2010. Distances and weighting schemes for bag of visual words image retrieval. In *Proceedings of the International Conference on Multimedia Information Retrieval (MIR'10)*. ACM, New York, NY, 323–332.

Nenad Tomašev, Milos Radovanović, Dunja Mladenic, and Mirjana Ivanovic. 2011. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. In *Proceedings of the 7th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM'11)*. 16–30.

Panu Turcot and David G. Lowe. 2009. Better matching with fewer features: The selection of useful features in large database recognition problems. In *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops'09)*. IEEE, Los Alamitos, CA, 2109–2116.

VISITO. 2011. VISITO Tuscany Object Recognition Dataset. Retrieved June 14, 2015, from http://www.fabriziofalchi.it/pisaDataset/.

Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. 2010. Locality-constrained Linear Coding for image classification. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*. 3360–3367.

Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal, and Michal Batko. 2006. *Similarity Search: The Metric Space Approach*. Advances in Database Systems, Vol. 32. Springer-Verlag.

Hao Zhang, Alexander C. Berg, Michael Maire, and Jitendra Malik. 2006. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Vol. 2. 2126–2136.

Xiao Zhang, Zhiwei Li, Lei Zhang, Wei-Ying Ma, and Heung-Yeung Shum. 2009. Efficient indexing for large scale visual search. In *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision*. 1103–1110.

Ziming Zhang, Jiawei Huang, and Ze-Nian Li. 2011. Learning sparse features on-line for image classification. In *Image Analysis and Recognition*. Lecture Notes in Computer Science, Vol. 6753. Springer, 122–131.

Yantao Zheng, Ming Zhao, Yang Song, Hartwig Adam, Ulrich Buddemeier, Alessandro Bissacco, Fernando Brucher, Tat-Seng Chua, and Hartmut Neven. 2009. Tour the world: Building a Web-scale landmark recognition engine. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09)*. 1085–1092.

Wangmeng Zuo, David Zhang, and Kuanquan Wang. 2008. On kernel difference-weighted k-nearest neighbor classification. *Pattern Analysis and Applications* 11, 3–4, 247–257.

Received April 2014; revised November 2014; accepted January 2015