Li and Lin *EURASIP Journal on Image and Video Processing* (2017) 2017:69
DOI 10.1186/s13640-017-0217-y

EURASIP Journal on Image and Video Processing

RESEARCH Open Access

A fast source-oriented image clustering method for digital forensics

Chang-Tsun Li[1,2] and Xufeng Lin[1*]

Abstract

We present in this paper an algorithm that is capable of clustering images taken by an unknown number of unknown digital cameras into groups, such that each contains only images taken by the same source camera. It first extracts a sensor pattern noise (SPN) from each image, which serves as the *fingerprint* of the camera that has taken the image. The image clustering is performed based on the pairwise correlations between camera fingerprints extracted from images. During this process, each SPN is treated as a random variable and a Markov random field (MRF) approach is employed to iteratively assign a class label to each SPN (i.e., random variable). The clustering process requires no a priori knowledge about the dataset from the user. A concise yet effective cost function is formulated to allow different "neighbors" different voting power in determining the class label of the image in question depending on their similarities. Comparative experiments were carried out on the Dresden image database to demonstrate the advantages of the proposed clustering algorithm.

Keywords: Image clustering, Markov random fields, Digital forensics, Sensor pattern noise, Multimedia forensics

1 Introduction

Nowadays, digital imaging devices, especially mobile phones with built-in cameras, have become an essential part of modern life. They enable us to record every detail of our life anytime and anywhere. Meanwhile, the rise of social media, such as Facebook, Twitter, and Instagram, has fostered and stimulated our interest in sharing photos and videos of life moments over social networks using mobile imaging devices. On the one hand, social media affords us a new way to express friendship, intimacy, and community. But on the other hand, the difficulty of verifying the profiles or identities of users on social networks also gives rise to the cyber crime.

A typical circumstance is that a number of images are collected under proper legal procedures from social networks for forensic analysis, but the devices which have been used to take these images are not available. If those images can be clustered into a number of groups, each including the images acquired by the same camera, the forensic investigators will be able to link the images to particular devices and in a better position to associate different social media accounts belonging to a person of interest. We refer to this task as *source-oriented image clustering*. This can be particularly useful in a variety of forensic cases, e.g., identifying fake user profiles, finding stolen camera devices, or defending against Internet defamation. Fortunately, with the advances in multimedia forensics, we are able to extract "device fingerprints" from images and videos and trace back to their source device. By resorting to device fingerprints extracted from images, source-oriented image clustering can be divided into two main sequential operations: the *extraction of device fingerprint* from images followed by an *image clustering* operation based on the device fingerprints. The main challenges in this scenario are:

- The investigator does not have the cameras that have taken the photos to generate quality reference device fingerprint.
- No prior knowledge about the number and types of the cameras are available.
- Given the sheer number of photos, analyzing each image in its full size is computationally prohibitive.

*Correspondence: xlin@csu.edu.au
[1]School of Computing and Mathematics, Charles Sturt University, Boorooma Street, Wagga Wagga, Australia
Full list of author information is available at the end of the article

1.1 The challenges of source-oriented image clustering and related works

There are many factors that affect the performance of the clustering system. One is the accuracy of the fingerprints extracted from images. Various forms of device fingerprints such as sensor pattern noise (SPN) [1–12], camera response function [13], re-sampling artifacts [14], color filter array (CFA) interpolation artifacts [15, 16], JPEG compression [17], and lens aberration [12, 18] have been proposed in recent years. Other device and image attributes such as binary similarity measures, image quality measures, and higher order wavelet statistics have also been adopted for identifying source imaging devices [19–22]. While many methods [13–16] make specific assumptions in their applications, SPN-based methods [1–12] do not require such assumptions to be satisfied and thus have drawn much more attention. Another merit of SPN is that it is unique to each device, which means it is capable of differentiating individual devices of the same model [1, 3, 5, 11]. These merits make SPN a good candidate for various digital forensic applications.

Another factor is the system's effectiveness in clustering images based on device fingerprints. The main objective in clustering applications is to group samples into clusters of similar features (e.g., the SPNs). Among a wide variety of methods, *k*-means [23, 24] and fuzzy *c*-means [25–27] have been intensively employed in various applications. However, classical *k*-means and fuzzy *c*-means clustering methods rely on the user to provide the number of clusters and initial centroids. Moreover, they are sensitive to outliers, and the computational complexities are very high for high-dimensional data, which make them unsuitable for clustering high-dimensional camera fingerprints.

The difficulty of specifying an appropriate cluster number also exists in graph clustering-based methods, such as [28–30]. In [31], the camera fingerprints clustering is formulated as a weighted graph clustering problem, where SPNs are considered as the vertices in a graph, while the weight of each edge is represented by the correlation between the SPN pair connected by the edge. A *k*-class spectral clustering algorithm [32] is employed to group the vertices into a number of partitions. To determine the optimal cluster number, the same spectral clustering algorithm is repeated for different value of *k* until the smallest size of the resultant clusters equals 1, i.e., one singleton cluster is generated. However, it is easy to form singleton clusters when some SPNs are severely contaminated by other interferences. So the feasibility of such manner of determining the optimal cluster number is still an issue.

To work without knowing the number of clusters, the agglomerative hierarchical clustering algorithms [33, 34] were adopted to cluster SPNs. Starting with the pairwise correlation matrix, the algorithms initially consider each SPN as a cluster and iteratively merge the two most similar clusters according to the average linkage criterion. At each iteration, an overall silhouette coefficient, which measures the cohesion inside clusters and the separation among clusters, is calculated to measure the quality of partition. This process stops when all SPNs have been merged into one cluster and the partition corresponding to the best clustering quality is deemed as the final partition. These two algorithms are relatively slow, because their time complexity is $\mathcal{O}(N^2 \log N)$, where N is the number of SPNs. Another limitation, which also exists in other more advanced hierarchical clustering-based algorithms such as CURE [35], ROCK [36], and CHAMELEON [37], is that once an object is assigned to a cluster, it will not be considered again in the ensuing process [38]. In the context of SPN clustering, the misclassification at the earlier stage is likely to induce error propagation in the succeeding merge and produce large clusters containing SPNs of different cameras.

Since the intrinsic quality of SPNs depends on many complex factors [11, 12, 39, 40], the average correlation between SPNs of one camera may be significantly different from that of other cameras. Therefore, SPN-based image clustering is a typical problem of finding clusters of different densities. The classical density-based algorithms, such as DBSCAN [41] and DENCLUE [42], are not applicable in this scenario, because their density-based definition of core points cannot identify the core points of varying density clusters. To overcome this problem, a shared nearest neighbor (SNN)-based clustering algorithm has been proposed in [43] to find clusters of different sizes and densities. However, choosing appropriate parameters for the algorithm is not easy if the data to be clustered is not well understood.

Considering the fact that the estimation of SPN improves if more images from the same camera are involved in the calculation, Bloy [44] presented an ad-hoc algorithm for clustering images based on SPN. The algorithm starts with selecting two images at random with their SPN correlation greater than an adaptive threshold that gradually increases as the number SPNs N in the cluster. The average SPN of this cluster is used as cluster centroid to attract more images whose SPN correlation with the centroid is greater than the adaptive threshold. This procedure repeats until the current cluster has grown to a pre-specified size (i.e., 50) or the entire dataset has been exhausted. If the cluster grows to the pre-specified size before the entire dataset is exhausted, a second pass through the dataset is conducted to include the images with similar SPN into the cluster without updating the centroid and the threshold. Once a cluster is formed, the algorithm repeats to form new clusters until no further clustering is possible. The algorithm allows the threshold to increase, but the adaptive

threshold is calculated from a quadratic curve, whose parameters are obtained by fitting the correlation values of four Canon cameras. However, the threshold's quadratic dependence on the number of SPNs is questionable. A clustering algorithm requires no a priori knowledge about the nature of the SPNs and the threshold is certainly more desirable.

To overcome the infeasibility of the manner of determining the optimal cluster number in [31], Amerini et al. [45] proposed a blind SPN clustering algorithm based on normalized cut criterion [46]. Similar to [31], SPNs are considered as the vertices in a graph and the weight of each edge measures the similarity between the two vertices connected by the edge. With the pairwise similarities between SPNs, the graph is bipartitioned recursively by finding the splitting point that minimizes the corresponding normalized cut. This recursive bipartition terminates when the mean value of intra-cluster weights is less than a pre-defined threshold T_h for all clusters. T_h is experimentally set to the value giving the best average performance on five datasets in terms of ROC curves. This normalized cut-based algorithm is fast and was reported to have better performance than [31] and [33] on datasets composed of hundreds of images taken by a few cameras.

More recently, Marra et al. introduced a two-step clustering algorithm in [47]. In the first step, the pairwise correlation matrix of SPNs is adjusted by subtracting a constant $\alpha = \mu_0 + 3\sigma_0$, where μ_0 and σ_0 are the mean and standard deviation, respectively, of the inter-camera correlations obtained from a training set. Then, the adjusted correlation matrix is fed into the correlation clustering algorithm [48] to generate a large number of over-partitioned clusters. While in the second step, an ad hoc refinement procedure is performed to progressively merge the clusters generated in the first step. The refinement step separates the clusters into two sets, a set of "large" clusters and a set of "small" clusters. If the majority of the SPNs in a small cluster are similar to (i.e., by comparing to a pre-defined threshold β) the centroid of a large cluster, the small cluster will be merged into the large cluster and the centroid will be updated accordingly. This process continues until no further merge can be performed. This algorithm was reported to outperform almost uniformly the state-of-the-art algorithms [47], but it requires all the SPNs to be retained in the RAM for efficiently updating the centroids of clusters, which makes it unsuitable for relatively large datasets.

We presented our preliminary study in [49], where each SPN is treated as a random variable and Markov random field (MRF) is used to iteratively update the class labels. Based on the pairwise correlation matrix, a reference similarity is determined using the k-means ($k = 2$) clustering algorithm and a *membership committee*, which consists of the most similar SPNs of each SPN, is established. The similarity values and the class labels assigned to the members of membership committee are used to estimate the likelihood probability of assigning each class label to the corresponding SPN. Then, the class label with the highest probability is assigned to the SPN. This process terminates when there are no more class label changes in two consecutive iterations. This algorithm performs well on small datasets, but its performance deteriorates as the size of dataset grows. Moreover, it is very slow because the likelihood probability involves all the class labels in the membership committee and has to be calculated for every SPN in every iteration. The time complexity is nearly $\mathcal{O}(N^3)$ in the first iteration, which makes it computationally prohibitive for large datasets. Therefore, a faster and more reliable algorithm that can handle large datasets is desirable for source-oriented image clustering.

1.2 Our contributions

In view of the aforementioned challenges in the context of device fingerprint-based image clustering, we conduct an in-depth study based on the work in [49] and propose a fast clustering framework for images of unknown sources. It makes several major contributions:

- First, we propose a fast and reliable algorithm for clustering camera fingerprints. Aiming at overcoming the limitations of the work in [49], the proposed algorithm makes the following improvements: (1) redefining the similarity in terms of the shared nearest neighbors; (2) speeding up the calculation of the reference similarity; (3) refining the determination of the membership committee; (4) reducing the complexity of calculations in each iteration; and (5) accelerating the speed of convergence. Not only the presentation of the clustering methodology is more comprehensive and detailed in this work, but also the proposed algorithm is much more efficient and reliable than that in [49].
- Secondly, we discuss in detail the related SPN clustering algorithms, namely the spectral, the hierarchical, the shared nearest neighbor, the normalized cut, and our previous MRF-based clustering methods [49]. These algorithms are evaluated and compared on real-world databases to provide insight into the pros and cons of each algorithm and offer a valuable reference for practical applications.
- Finally, we evaluate the proposed algorithm on a large and challenging image database which contains 7400 images taken by 74 cameras, covering 27 camera models and 14 brands, while the database used in [49]

includes only six cameras. Furthermore, the quality of clustering is characterized by F1-measure and Adjusted Rand Index, which are more suitable for evaluating clustering results than only the true positive rate or accuracy used in [31, 33, 34, 49].

1.3 Outline of this paper

The remainder of this work is organized as follows. The formulation and discussion of the proposed algorithm are given in Section 2. In Section 3, respectively. The parameter selection of the proposed algorithm as well as the comparison with other related works is presented. Finally, Section 5 concludes this work.

2 Methods

To facilitate the clustering, the SPN of a small block at the center of each of the given N images are extracted. An $N \times N$ correlation matrix is established, with one element, (i, j), representing the correlation between the SPNs of image i and j. Then, an alternative similarity matrix in terms of shared nearest neighbors is constructed from the correlation matrix. By making the pairwise similarities available in the matrix, the system does not have to repeat the similarity calculation when the similarity of the same pair of images is required again in the iterative clustering process. Although the number of image classes (cameras) can be much greater than 2, for each image, there are only two types of similarity: intra-class and inter-class. Based on the similarity matrix, each SPN is treated as a random variable to be assigned a class label, and a reference similarity r is estimated to serve as a rough boundary between the intra- and inter-class similarities in order to encode a cost function using a Markov random field (MRF). Separating the similarities into intra- and inter-class similarities enables us to find clusters of different densities, because the average intra-class similarity indicates the "density" of the cluster that each SPN belongs to. In the following subsections, we will provide the details of the proposed algorithm.

2.1 SPN extraction

Given an image I, the following equation is used to extract the SPN, n, from a block of the size specified by the user from the center:

$$n = I - \mathcal{F}(I), \tag{1}$$

where $\mathcal{F}$ is the denoising algorithm proposed in [50]. Each SPN is further preprocessed by the Wiener filtering (WF) in the DFT domain [2] to suppress the non-unique artifacts. Note that the reason we do not use our recent preprocessing scheme in [11] is that, the peaks in the DFT spectrum of a single SPN are not as distinct as those in the spectrum of a clean reference SPN.

2.2 Establishment of similarity matrix

During the process of clustering, similarities between SPNs are to be used to determine the class membership of each image (or SPN). As will be seen in Section 2.3.4, the process of class label update, which involves the calculation of similarities between SPNs, has to be iterated until the stop criterion is met. However, repeating the similarity calculation of the same SPN pairs is time-consuming. Therefore, the purpose of establishing an $N \times N$ similarity matrix is to calculate the similarity only once for each SPN pair. When a similarity value is needed at any stage, the value is retrieved from the similarity matrix. The similarity between any two SPNs n_i and n_j is initially measured by the normalized cross correlation (NCC)

$$\rho_{ij} = \frac{(n_i - \bar{n}_i) \cdot (n_j - \bar{n}_j)}{\|n_i - \bar{n}_i\| \cdot \|n_j - \bar{n}_j\|}, \quad i, j \in [1, N], \tag{2}$$

where $\|\cdot\|$ is the L_2 norm and the mean value is denoted with a bar. In this way, we establish an $N \times N$ correlation matrix ρ, with element ρ_{ij} indicating the closeness between SPNs n_i and n_j. Because of the symmetrical nature of the correlation matrix and that the elements (self-correlations) along the diagonal axis is always 1, only $N \times (N-1)/2$ correlations need to be calculated.

However, due to the varying qualities of SPNs of different cameras, the average correlation between SPNs of one camera may be different from that of another camera. As exemplified in Fig. 1a, the average correlation the class highlighted by the green rectangle is higher than that of the class highlighted by the blue rectangle. This problem makes the clustering of SPNs more challenging. An alternative definition of similarity in terms of shared nearest neighbors, as proposed in [36, 43, 51], is a promising way to overcome this problem. Specifically, the similarity W_{ij} between two SPNs n_i and n_j is redefined as

$$W_{ij} = |\mathbb{N}(n_i) \cap \mathbb{N}(n_j)|, \tag{3}$$

where $\mathbb{N}(n_i)$ and $\mathbb{N}(n_j)$ are, respectively, the κ-nearest neighbors of n_i and n_j constructed from the correlation matrix ρ. So W_{ij} measures the number of κ-nearest neighbors shared by n_i and n_j. The constructed similarity matrix in terms of shared nearest neighbors (SNN) is shown in Fig. 1b, where the divergences of similarities in different classes have been significantly reduced. Also note that, even when the SNN similarity is applied, the intra-class connectivity remains weak for the images taken by Casio EX-Z150, as highlighted in the red rectangle. The underlying reason is the irregular geometric distortions related to the different focal length settings when capturing different images, as reported in [52].

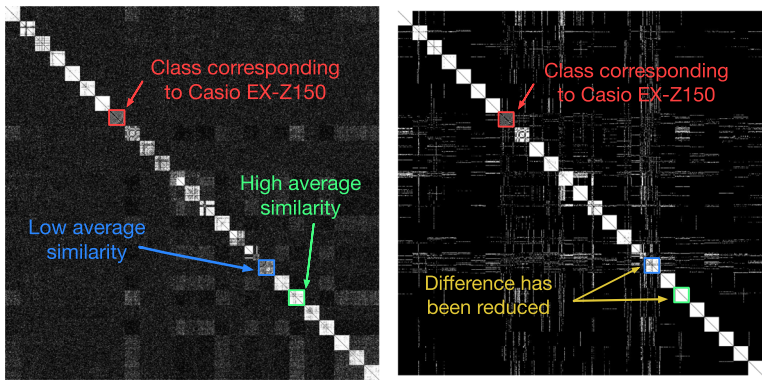

Fig. 1 Pairwise similarities of 1000 images taken by 25 cameras (each responsible for 40 images). **a** Correlation matrix ρ. **b** Similarity matrix W in terms of shared κ-nearest neighbor ($\kappa = 15$)

Procedure 1 $f = \text{fastClustering}(W, m)$

1: $f \leftarrow \text{randPerm}(N)$; ▷ Assign unique random class labels
2: $r_i \leftarrow \text{fastSplit}(W_{i:}, ini, low, high)$; ▷ Find the reference similarity of n_i
3: $C_i \leftarrow \text{establishMC}(W_{i:}, m)$; ▷ Establish the MC of n_i
4: $p_i \leftarrow 0$; $q_i \leftarrow 2$; $stable \leftarrow 2$; $iteration \leftarrow 0$;
5: **while** $stable > 0$ **or** $++iteration \leq 50$ **do**
6: **for** $i = 1$ to N **do**
7: **if** p_i **then continue**; ▷ Stop updating for n_i if $p_i = 1$
8: $L_i \leftarrow \{f_j | j \in C_i\}$; ▷ Class labels in the MC of n_i
9: **for each** $l \in L_i$ **do**
10: $U_i(l, W_{C_i}, L_i) \leftarrow \sum_{j \in C_i} s(l, f_j)(W_{ij} - r_i)$;
11: **end for**
12: $\hat{f}_i \leftarrow \arg\min_{l \in L_i} U(l, W_{C_i}, L_i)$;
13: **if** $f_i \neq \hat{f}_i$ **then**
14: $f_i \leftarrow \hat{f}_i$; $p_i \leftarrow 0$; ▷ Update the class label of n_i
15: **else**
16: **if** $--q_i < 1$ **then**
17: $p_i \leftarrow 1$; ▷ Stop updating for n_i
18: **end if**
19: **end if**
20: **if** $\sum_i p_i == 0$ **then**
21: $stable--$;
22: **end if**
23: **end for**
24: **end while**
25: **return** f;

2.3 Clustering

Taking the similarity matrix W as input, the task of this step is to identify image groups such that each group corresponds to one camera. Our previous experience of using Markov random field (MRF) approach to image segmentation [53, 54] and many others' successful applications of MRF [9, 55–57] suggest that the local characteristics of MRFs (also known as *Markovianity*) allow global optimization problems to be solved iteratively by taking local information into account. Suppose there are K classes of images in the subset, with the value of K unknown, and denote $D = \{d_k | k = 1, 2, \ldots, K\}$ as the set of class labels and $f_i \in D$ as the class label of SPN n_i. By considering the label f_i of each SPN n_i as a random variable, the objective of clustering is to assign an optimal class label d_k to each random variable n_i in an iterative manner until the stop criterion is met. The pseudo code of clustering is shown in Procedure 1, and the details will be explained as follows.

2.3.1 *Assign unique initial class labels*

Because the number of classes K is unknown, before the first iteration of the labeling process starts, each SPN n_i is treated as a *singleton* cluster and assigned a unique random class label, as shown in step 1 of Procedure 1. That is to say that $K = N$ and $f_i = d_i$, $i \in 1, 2, \ldots, N$. The class label of each SPN in question will be updated iteratively in Step 14 of Procedure 1 based on (1) the similarities between the SPN in question and the SPNs in its membership committee and (2) the current class labels of the SPNs in the membership committee. So eventually when the algorithm converges, or the stop criterion is met, images taken by the same camera will be given the same class label. By doing so, the algorithm starts with a set of N singleton clusters without requiring the user to specify the number of clusters.

2.3.2 Calculate reference similarity

Although the actual number of classes, K, is unknown, we can expect that normally the similarities between SPNs of the same class (called *intra-class similarity*) are greater than the similarities between SPNs of different classes (called *inter-class similarity*). So for each SPN n_i, its inter-class and intra-class similarities are expected to be separable. In [49], a simple k-means clustering method ($k = 2$) is used to cluster the $N - 1$ similarity values into two groups (one as *intra-class* and the other *inter-class*). Then, the average of the centroids of the two clusters is taken as a *reference similarity* r to separate the two distributions. However, the general-purpose k-means is slow and quickly becomes inefficient for large datasets. Be aware that we are dealing with the binary separation of one-dimensional data, and we have the prior knowledge of the approximate range where the cutoff point should lie in, so we propose a fast method, which shares the same essence as k-means, to iteratively search for the appropriate cutoff point, as shown in step 2 of Procedure 1, where $W_{i:}$ is the similarities between n_i and the other $(N - 1)$ SPNs.

The details of the algorithm are given in Procedure 2. We assume that the reference similarity r to be determined lies in between $[\,low, high]$, so the sum and size of the similarities between $(0, low)$ and $(high, N)$ can be pre-calculated before the iterative update. The purposes of limiting the search range to $[\,low, high]$ are twofold. First, it narrows down the search range and therefore speeds up the search process. Second, it forces the optimal r to fall within an appropriate range so as to alleviate the problem of "local minimal". The search process is further sped up by specifying a value *ini* as the initial r. In step 6 of Procedure 2, $\mathcal{I}$ represents the "binary" (0 or 1) class labels of the similarities in $[\,low, high]$. The midpoint of the means of the two classes is used to update r, as provided in step 9 of Procedure 2. The update terminates when label assignments no longer change. Incorporating *ini*, *low* and *high* to facilitate the determination of r makes the search process faster and more flexible. In some cases, such prior information is already known to the user. In our experiments, low, ini and high were set to 0, 1 and 5, respectively.

The output r of Procedure 2 serves the purpose of dividing the intra- and inter-class similarities and can be used to encode the cost function to be defined in Eq. (6). Although the similarities are both scene- and device-dependent, we could expect that most intra-class similarities are greater than r while most inter-class similarities are less than r. It is also intuitive that, for most cases, a similarity value farther away from r on the left-hand side indicates a higher probability that the two corresponding images are taken by different devices. On the other hand, we have higher confidence in believing that a similarity value farther away from r on the right-hand side indicates that the two corresponding images are taken by the same device. The closer to r, the less confidence we have on the similarity value in telling us the situation. This suggests that, if we treat classification as an optimization problem, the distance between a similarity W_{ij} and r can be used to encode an objective function for guiding the search for the optimal class label of each image. We will explain how we make use of this useful information in Section 2.3.4.

Procedure 2 $r = \text{fastSplit}(v, ini, low, high)$

1: $\mathcal{L} = \{v_i | v_i < low\}$;
2: $\mathcal{H} = \{v_i | v_i > high\}$;
3: $\mathcal{M} = \{v_i | v_i \geq low \&\& v_i \leq high\}$;
4: $sz\_L \leftarrow \text{size}(\mathcal{L})$; $sum\_L \leftarrow \text{sum}(\mathcal{L})$;
5: $sz\_H \leftarrow \text{size}(\mathcal{H})$; $sum\_H \leftarrow \text{sum}(\mathcal{H})$;
6: $\mathcal{I} \leftarrow \{\mathcal{M} > ini\}$;
7: **do**
8: $\mathcal{J} \leftarrow \mathcal{I}$;
9: $r \leftarrow 0.5 \times \left(\frac{sum\_L + \text{sum}(\mathcal{M}_{\tilde{\mathcal{I}}})}{sz\_L + \text{sum}(\tilde{\mathcal{I}})} + \frac{sum\_H + \text{sum}(\mathcal{M}_{\mathcal{I}})}{sz\_H + \text{sum}(\mathcal{I})} \right)$;
10: $\mathcal{I} \leftarrow \{\mathcal{M} > r\}$;
11: **while** $\mathcal{I} \neq \mathcal{J}$
12: **return** r;

2.3.3 Establish membership committee

When determining the class label for each SPN n_i, instead of involving the entire dataset in the decision-making process, the theory of Markov random fields allows us to involve only a small local "neighborhood" of that SPN. As displayed in Step 3 of Procedure 1, we establish a "neighborhood" C_i (i.e., *membership committee* (MC) in [49] and this work) with m *key* members that are most similar to n_i. The membership committee can be efficiently established by partially selecting the m SPNs with the largest similarities in each row of W (e.g., using the partial sorting algorithm proposed in [58]). Note that the reason we use the term "membership committee", instead of "neighborhood", is because information such as similarities and current class labels of the SPNs within the membership committee determines the class label (i.e., membership) of the SPN in question. The m key members contribute "positive" votes (i.e., class labels) which tell the system what the most likely labels are, while the similarity value encoded in the cost function and the associated probability tell the system whether a committee member is a likely one. In so doing, we could ensure that in the main feature of Markov random fields, the local characteristics [53] are exploited in the clustering process.

2.3.4 Update class labels using MRF

During the clustering process, each SPN is iteratively visited and re-labeled until the stop criterion is met. In terms of Markov random fields, when an SPN n_i is being visited, the probability $p(\cdot)$ of assigning each class label l currently assigned to the members of C_i is calculated using

$$p(f_i = l|W_{C_i}, L_i) = \frac{1}{Z_i} e^{-U_i(l, W_{C_i}, L_i)}, \tag{4}$$

where f_i is the class label of SPN n_i, W_{C_i} is the similarities between SPN n_i and the corresponding members of C_i, i.e., $W_{C_i} = \{W_{ij} | j \in C_i\}$, and L_i is the set of class labels currently assigned to the members of C_i, i.e., $L_i = \{f_j | j \in C_i\}$, $l \in L_i$. Z_i is the partition function [25]

$$Z_i = \sum_{l \in L_i} e^{-U_i(l, W_{C_i}, L_i)}, \tag{5}$$

where $U_i(l, W_{C_i}, L_i)$ is the cost of assigning label l to n_i given W_{C_i} and L_i. It is defined as

$$U_i(l, W_{C_i}, L_i) = \sum_{j \in C_i} s(l, f_j)(W_{ij} - r_i), \tag{6}$$

where W_{ij} is the similarity (see Eq. (2)) between n_i and n_j ($j \in C_i$), r_i is the *reference similarity* described in Section 2.3.2, and $s(l, f_j)$ is a sign function defined as

$$s(l, f_j) = \begin{cases} +1, & l \neq f_j \\ -1, & l = f_j. \end{cases} \tag{7}$$

It is clear to see that the probability of each label l to be assigned to f_i is based on the observed data and the current local class configuration L_i. From the cost function $U(\cdot)$ in Eq. (6), we can see that the closer the similarity W_{ij} is to r_i, the less significant n_j is in determining the class label for SPN n_i. From the sign function in Eq. (7) and its role in Eq. (6), we can see that the formulation of Eq. (6) encourages appropriate label assignment with a reward (i.e., a *negative* cost $U(\cdot)$ to increase the probability of that label). By the same token, it penalizes inappropriate decisions to reduce the probability of assigning an inappropriate label by imposing a *positive* cost $U(\cdot)$. The following explains these two cases, each with two different scenarios.

- **Rewarding appropriate label assignments**
  Scenario 1: If $W_{ij} < r_i$ (i.e., SPNs n_i and n_j belong to *different* classes) and the label l under investigation is *different* from f_j (i.e., $l \neq f_j$), then $s(l, f_j) = +1$ will be used in Eq. (6). As a result, a *negative* value of $s(l, f_j)(W_{ij} - r_i)$ is contributed to $U(\cdot)$, which will in turn *increase* the probability of $p(f_i = l|W_{C_i}, L_i)$ in Eq. (4).
  Scenario 2: If $W_{ij} > r_i$ (i.e., SPNs n_i and n_j belong to the *same* class) and the label l under investigation is also the *same* as f_j (i.e., $l = f_j$), then $s(l, f_j) = -1$ will be used in Eq. (6). A *negative* value of $s(l, f_j)(W_{ij} - r_i)$ is contributed to $U(\cdot)$, which will in turn *increase* the probability of $p(f_i = l|W_{C_i}, L_i)$ in Eq. (4).
- **Penalizing inappropriate label assignments**
  Scenario 3: If $W_{ij} < r_i$ (i.e., SPNs n_i and n_j belong to *different* classes) but the label l under investigation is the *same* as f_j (i.e., $l = f_j$), then $s(l, f_j) = -1$ will be used in Eq. (6). As a result, a *positive* value of $s(l, f_j)(W_{ij} - r_i)$ is contributed to $U(\cdot)$, which will in turn *reduce* the probability of $p(f_i = l|W_{C_i}, L_i)$ in Eq. (4).
  Scenario 4: If $W_{ij} > r_i$ (i.e., SPNs n_i and n_j belong to the *same* class) but the label l under investigation is *different* from f_j (i.e., $l \neq f_j$), then $s(l, f_j) = +1$ will be used in Eq. (6). A *positive* value of $s(l, f_j)(W_{ij} - r_i)$ is contributed to $U(\cdot)$, which will in turn *reduce* the probability of $p(f_i = l|W_{C_i}, L_i)$ in Eq. (4).

From these four scenarios, we can also see that the farther away W_{ij} is from r_i, the greater the reward (penalty) will be when an appropriate (inappropriate) decision is made. Like other MRF approaches to optimization problems, deterministic and stochastic relaxation [54] can be used to pick a new label l for f_i based on $p(f_i = l|W_{C_i}, L_i)$. Because of the low convergence rate of stochastic relaxation, we pick label l in a deterministic sense according to

$$\hat{f}_i = \arg\max_{l \in L_i} p(f_i = l|W_{C_i}, L_i). \tag{8}$$

Since Z_i is the same for all class labels in L_i, maximizing $p(f_i = l|W_{C_i}, L_i)$ is equivalent to minimizing $U_i(l, W_{C_i}, L_i)$, as implemented in step 12 of Procedure 1. The m most similar SPNs in the membership committee play a decisive role in determining the class label, so once the label of an SPN has been determined by Eq. (8), it actually triggers a convergence process among the SPNs in its membership committee. As a consequence, the class labels of most SPNs quickly become stable, and continually updating those labels helps little in improving the performance. Therefore, we stop updating the class label of an SPN if no label changes in two consecutive iterations, as shown in step 7 of Procedure 1. This configuration significantly reduces the number of

SPNs updated in each iteration and has little effect on the performance. Finally, the *stop criterion* we employ is that there are no changes of class labels to any SPNs in two consecutive iterations, or the iteration number reaches 50.

3 Discussion

The reasons why the classifier can work without the user specifying the reference similarity r and the number of classes K can be summarized as follows.

- The fact that the similarity values between each SPN and the rest of the dataset can be grouped into *intra-class* and *inter-class* as described in Section 2.3.2 facilitates *adaptive* determination of the reference similarity r automatically. This adaptability also allows the algorithm to get rid of the tricky threshold specification (e.g., the similarity threshold used in [44] and the binarization threshold in [59]).
- The clustering process starts with a class label space as big as the entire dataset (i.e., the worse case with each SPN n_i as a *singleton* cluster) and the *most* similar SPNs are always kept in n_i's membership committee C_i, so the clusters can merge and converge to a certain number of final clusters quickly. The term $W_{ij} - r_i$ in Eq. (6) also provides adaptability and helps the clustering to converge because it gives more say to the SPNs with the similarity value farther away from the reference similarity r in determining the class label for the SPN in question.

4 Results

4.1 Experimental setup

We conducted the experiments on the Dresden image database [60]. 7400 images acquired in JPEG format by 74 cameras (each responsible for 100 images), covering 27 camera models and 14 manufacturers, were involved in the experiments. We only considered the green channel of each image and tested our proposed algorithm on image blocks of three different sizes, namely $s = 1024\times1024$, $s = 512 \times 512$, and $s = 512 \times 256$ pixels. All the experiments were performed on a laptop with an Intel(R) Core(TM) i7-6600U CPU @2.6 GHz and a RAM of 16 GB.

4.2 Evaluation measures

We used the ground-truth class labels to evaluate the clustering results. To avoid confusion, we will refer to the images from the same camera as a *class* and refer to those clustered into the same group by the clustering algorithm as a *cluster*. Suppose $\Omega = \{\omega_1, \omega_2, \ldots, \omega_j, \ldots, \omega_J\}$ are the set of ground-truth classes, and N images are partitioned to a set of clusters, $C = \{c_1, c_2, \ldots, c_i, \ldots, c_I\}$, by clustering algorithm. We used different measures to evaluate the quality of clustering. The first measure is F1-measure:

$$\mathcal{F} = 2 \cdot \frac{\mathcal{P} \cdot \mathcal{R}}{\mathcal{P} + \mathcal{R}}, \tag{9}$$

where the average precision rate $\mathcal{P}$ and the average recall rate $\mathcal{R}$ are defined as

$$\begin{cases} \mathcal{P} = \sum_i |c_i \cap \omega_{j_i}| / \sum_i |c_i| \\ \mathcal{R} = \sum_i |c_i \cap \omega_{j_i}| / \sum_i |\omega_{j_i}|. \end{cases} \tag{10}$$

Here, $|c_i|$ is the size of cluster c_i, $|\omega_{j_i}|$ is the size of the most frequent class, ω_{j_i}, in cluster c_i.

Another popular measure of clustering quality is Rand Index [61], which measures the agreement between C and Ω. Among the $\binom{N}{2}$ distinct pairs, there are four different types of pairs:

i) True positive pair: images in the pair fall in the same class in Ω and in the same cluster in C.
ii) True negative pair: images in the pair fall in different classes in Ω and in different clusters in C.
iii) False positive pair: images in the pair fall in different classes in Ω but in the same cluster in C.
iv) False negative pair: images in the pair fall in the same class in Ω but in different clusters in C.

The Rand Index RI is defined as:

$$RI = \frac{TP + TN}{TP + FP + TN + FN}, \tag{11}$$

where TP, TN, FP, and FN are the numbers of true positive, true negative, false positive, and false negative pairs, respectively. RI ranges from 0 to 1, but its expectation $\overline{RI}$ does not equal to 0. To get rid of this bias, we adopted the Adjusted Rand Index [62]:

$$\mathcal{A} = \frac{RI - \overline{RI}}{1 - \overline{RI}}. \tag{12}$$

The last measure we used is the ratio of the number of discovered clusters to the number of ground-true classes:

$$\mathcal{N} = \frac{n_d}{n_g}, \tag{13}$$

where n_d is the number of discovered clusters and n_g is the number of ground-truth classes. We will refer to $\mathcal{N}$ as the *cluster-to-class ratio* in the rest of this paper.

Note that for $\mathcal{F} \in [0, 1]$ and Adjusted Rand Index $\mathcal{A} \in [-1, 1]$, a higher value indicates better clustering performance. For $\mathcal{N}$, a value close to 1 does not necessarily indicate a good performance, but a value much larger than 1 does indicate that the clustering algorithm produces a large number of small or even singleton clusters.

4.3 Parameter settings

Two parameters need to be set for our proposed algorithm, the size of the nearest neighbors κ and the size of the membership committee m. κ determines the highest

SNN similarity between SPNs, because two SPNs can only share at most κ nearest neighbors according to Eq. (3). If κ is too small, even two dissimilar SPNs are likely to have a similarity of κ and the difference between the "similar" and "dissimilar" pairs will be obscured. On the other hand, if κ is too large, the SNN similarity is insensitive to local variations, and the algorithm tends to produce large clusters containing the images from different actual "classes". Similarly, if m is too small, there will be not enough information for determining the class label of SPNs. As a consequence, the assigned labels remain random and unreliable, which hinders the convergence of the algorithm and result in many singleton clusters. While if m is too large, more and more dissimilar SPNs will be involved in the calculation and therefore mislead the algorithm to make wrong decisions.

To see how κ and m affect the clustering quality, we applied the proposed clustering algorithm on two small subsets randomly sampled from the Dresden database. The first subset (i.e., $\mathcal{D}_4$ in Section 4.4.1) consists of 1000 images taken by 50 cameras, with the number of images acquired by different cameras ranging from 10 to 60, and the second subset (i.e., $\mathcal{D}_5$ in section 4.4.1) consists of 1024 images, with 24 additional singleton images (i.e., they are from 24 different cameras) added to the first subset. We varied κ from 1 to 50 and m from 5 to 50.

Results on the first and the second subset are shown in the first and second row of Fig. 2, respectively. As can be seen, if κ is too small (e.g., < 5), the algorithm produces many small clusters and results in a very low recall rate (see Fig. 2b, f). As κ increases, the clusters belonging to different classes are likely to be merged together, which gives rise to a lower precision rate (see Fig. 2a, e). For the size of the membership committee, a small m leads to a low recall rate (see Fig. 2a, e). As m goes up to a point where "enough" similar SPNs can help to make trustworthy decisions, it strikes a good balance between the precision rate and the recall rate, and therefore achieves a favorable F1-measure and Adjusted Rand Index (see Fig. 2c, d, g, and h). But if we keep increasing m, there is a chance to decrease the precision rate (see Fig. 2a, e) due to the misleading information provided by the membership committee. The results on the first and the second subset share very similar patterns and trends, but the areas corresponding to high clustering quality (i.e., the areas highlighted in dark red in Fig. 2d, h) shrink towards the left-bottom corner. It indicates that a relatively smaller κ or m is preferable when singleton images are present in the database. In our following experiments, both κ and m are set to 15.

4.4 Comparisons and analyses

To illustrate the advantages of our proposed algorithm, we compared it with other five clustering methods: (1) the multi-class spectral clustering (SC) method [31], (2) the hierarchical clustering (HC) method [34], (3) the shared nearest neighbor clustering (SNNC) method [43], (4) the normalized cut-based clustering (NCUT) method [45], and (5) the Markov random field based clustering (MRF) method [49]. We did not include Boly's algorithm [44] and

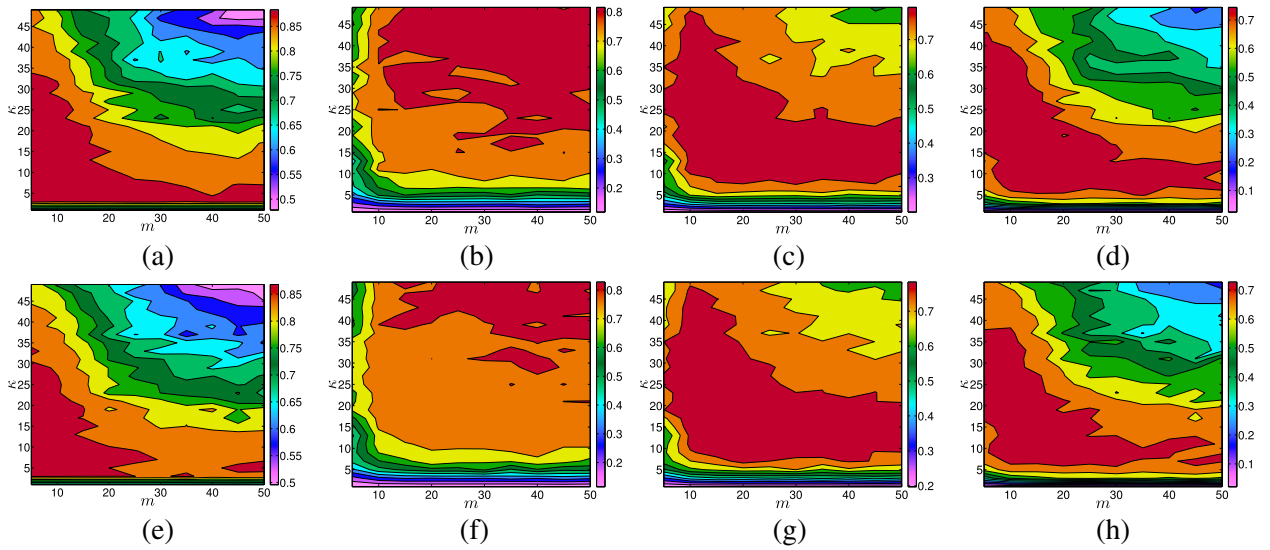

Fig. 2 How κ and m affect the clustering quality. **a** Precision rates on $\mathcal{D}_4$, **b** recall rates on $\mathcal{D}_4$, **c** F1-measures on $\mathcal{D}_4$, **d** adjusted Rand Indexes on $\mathcal{D}_4$, **e** precision rates on $\mathcal{D}_5$, **f** recall rates on $\mathcal{D}_5$, **g** F1-measures on $\mathcal{D}_5$, and **h** adjusted Rand Indexes on $\mathcal{D}_5$

Marra's algorithm [47], because both algorithms retain the fingerprints in the RAM for updating the centroids of clusters, which makes them unsuitable for relatively large datasets. Moreover, for Marra's algorithm [47], the selection of β, the average level of correlation for same-camera residuals, can be tricky since β varies across different cameras.

For fair comparison, we used a fully connected graph for *SC* rather than the sparse k-nearest neighbor graph in [31]. Also, note that we did not use the hierarchical clustering proposed in [33] for comparison, because we found in experiments that the algorithm in [34] performs slightly faster and better than that in [33]. For SNNC, there are three parameters: the size of the nearest neighbors κ, the similarity threshold *Eps* for calculating the SNN density, and the density threshold *MinPts* for finding the core points. We set κ to the same value as our proposed algorithm, i.e., $\kappa = 15$, and set *Eps* and *MinPts* to 2 and 10, respectively. For NCUT, we set the aggregation threshold T_h to 0.0037 rather than the 0.037 in [45], which results in many singleton clusters. For MRF, to avoid going into infinite iterations when the algorithm does not converge, we set the maximum number of iterations to 50. We will conduct two experiments, one on datasets of fixed size with varying class distributions and different levels of clustering difficulty to test the *adaptability* of algorithms and the other on datasets of varying sizes to test the *scalability* of algorithms.

4.4.1 Clustering on datasets of fixed size

In this experiment, we first set up four datasets of fixed size and based on the Dresden database. As we know that, images acquired by cameras of the same model may undergo the same or similar image processing pipeline. As a result, the non-unique artifacts left in the images make them more difficult to be distinguished from each other. We therefore categorize the clustering difficulties into *easy* and *hard* levels. For the easy level, the images in different classes are taken by cameras of different models, while on the hard level, images in some of the different classes are taken by devices of the same model. Additionally, it is common in practical situation that the numbers of images vary widely across devices. So we categorize the distributions of images into *symmetric* and *asymmetric*. Based on these considerations, we set up the following four different datasets:

- $\mathcal{D}_1$: easy symmetric dataset, which consists of 1000 images taken by 25 cameras of different models (each accounting for 40 images). It nearly covers all the popular camera manufacturers, such as Canon, Nikon, Olympus, Pentax, Samsung, and Sony.
- $\mathcal{D}_2$: easy asymmetric dataset. 20, 30, 40, 50, and 60 images are alternatively selected from the images taken by each of the 25 cameras in $\mathcal{D}_1$ to make up a total of 1000 images.
- $\mathcal{D}_3$: hard symmetric dataset, which consists of 1000 images taken by 50 cameras (each accounting for 20 images). The 50 cameras only cover 12 models, so some of them are of the same model.
- $\mathcal{D}_4$: hard asymmetric dataset. 10, 15, 20, 25, and 30 images are alternatively selected from the images taken by each of the 50 cameras in $\mathcal{D}_3$ to make up a total of 1000 images.

Our proposed clustering algorithm essentially exploits the affinities between neighboring images. Thus, it would be interesting to see how the proposed algorithm deals with singleton classes, i.e., classes composed by only one single image as no other images from the same camera are present in the database. Recall that the images in our entire database are from 74 cameras and the images in dataset $\mathcal{D}_4$ are from 50 of them. We therefore randomly select one image from those taken by each of the remaining 24 cameras and add them to $\mathcal{D}_4$ to form an extra database, $\mathcal{D}_5$, consisting of 1024 images. This setting allows us to investigate the influence of singleton classes by comparing the performance on $\mathcal{D}_4$ and $\mathcal{D}_5$.

We tested the six algorithms on $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$, $\mathcal{D}_4$, and $\mathcal{D}_5$. For each dataset, three pairwise correlation matrices are calculated using SPNs of three different sizes, namely 1024×1024, 512×512, and 512×256 pixels. The results on $\mathcal{D}_1 - \mathcal{D}_5$ are listed in Tables 1, 2, 3, 4, and 5, respectively. The best F1-measures, adjusted Rand Indexes, and the cluster-to-class ratios are highlighted in bold.

As can be seen, SC performs poorly on challenging datasets $\mathcal{D}_3$, $\mathcal{D}_4$, and $\mathcal{D}_5$ even using SPNs of 1024×1024 pixels, with $\mathcal{F} = 0.18$, $\mathcal{A} = 0.06$ in Table 3 and even worse in Tables 4 and 5. However, SC performs surprisingly better on smaller block sizes, 512×512 pixels. The rather contradictory results are due to the stop criterion of SC, because the algorithm terminates when the size of the smallest cluster equals 1. But the smallest class size of $\mathcal{D}_3$ and $\mathcal{D}_4$ is no larger than 20, so it is easy to form singleton clusters and result in premature termination of the algorithm, while the more ambiguous information in the SPNs extracted from smaller image blocks impedes the separation of images taken by different cameras and therefore forces the algorithm to try out more possible partitions. For example, the optimal number of partitions determined by SC is 5 when using SPNs of 1024×1024 pixels on $\mathcal{D}_3$, but when using SPNs of 512×512 pixels, the number of partitions increases to 9, which is closer to the ground truth class number 50 and therefore ends up with a better performance. Because of the larger

Table 1 Comparison of clustering algorithms on $\mathcal{D}_1$

	1024 × 1024					512 × 512					512 × 256				
Algorithms	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$
SC [31]	0.36	0.99	0.53	0.27	0.36	0.23	0.94	0.36	0.08	0.24	0.25	0.88	0.39	0.10	0.28
HC [34]	0.71	0.57	0.63	0.54	1.24	0.94	0.18	0.30	0.72	5.20	0.87	0.10	0.19	0.49	8.40
SNNC [43]	0.90	0.48	0.63	0.77	1.88	0.63	0.23	0.34	0.42	2.68	0.59	0.19	0.29	0.35	3.08
NCUT [45]	0.86	0.16	0.27	0.65	5.44	0.73	0.24	0.36	0.51	3.08	0.68	0.24	0.36	0.48	2.80
MRF [49]	0.96	0.43	0.60	0.87	2.24	0.82	0.41	0.55	0.62	2.00	0.78	0.65	0.71	0.60	1.20
Proposed	0.99	0.80	0.88	0.93	1.24	0.93	0.83	0.88	0.84	1.12	0.86	0.72	0.78	0.72	1.20

Table 2 Comparison of clustering algorithms on $\mathcal{D}_2$

	1024 × 1024					512 × 512					512 × 256				
Algorithms	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$
SC [31]	0.34	0.98	0.51	0.17	0.28	0.35	0.97	0.51	0.19	0.32	0.41	0.78	0.54	0.21	0.44
HC [34]	0.82	0.58	0.68	0.63	1.48	0.50	0.77	0.61	0.28	0.68	0.85	0.14	0.24	0.59	5.60
SNNC [43]	0.88	0.48	0.62	0.76	1.80	0.60	0.19	0.29	0.37	2.80	0.54	0.16	0.24	0.27	3.16
NCUT [45]	0.86	0.20	0.33	0.71	4.28	0.75	0.30	0.43	0.61	2.56	0.72	0.38	0.50	0.58	1.76
MRF [49]	0.97	0.49	0.65	0.91	1.88	0.79	0.47	0.59	0.58	1.56	0.76	0.50	0.60	0.53	1.44
Proposed	0.97	0.78	0.86	0.91	1.20	0.92	0.87	0.89	0.87	1.04	0.88	0.83	0.85	0.78	1.04

Table 3 Comparison of clustering algorithms on $\mathcal{D}_3$

	1024 × 1024					512 × 512					512 × 256				
Algorithms	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$
SC [31]	0.10	1.00	0.18	0.06	0.10	0.17	0.94	0.29	0.09	0.18	0.41	0.72	0.52	0.20	0.56
HC [34]	0.71	0.50	0.58	0.44	1.44	0.81	0.21	0.33	0.52	3.92	0.76	0.13	0.22	0.31	5.94
SNNC [43]	0.45	0.26	0.32	0.23	1.74	0.35	0.15	0.21	0.11	2.34	0.30	0.12	0.17	0.07	2.50
NCUT [45]	0.92	0.13	0.22	0.52	7.30	0.72	0.14	0.24	0.37	4.96	0.53	0.15	0.23	0.25	3.58
MRF [49]	0.51	0.47	0.49	0.21	1.08	0.44	0.45	0.44	0.15	0.96	0.32	0.47	0.38	0.11	0.68
Proposed	0.87	0.75	0.80	0.74	1.16	0.68	0.64	0.66	0.48	1.06	0.52	0.47	0.49	0.16	1.10

Table 4 Comparison of clustering algorithms on $\mathcal{D}_4$

	1024 × 1024					512 × 512					512 × 256				
Algorithms	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$
SC [31]	0.03	0.64	0.06	0.00	0.04	0.18	0.94	0.30	0.05	0.14	0.37	0.83	0.52	0.17	0.36
HC [34]	0.51	0.65	0.57	0.23	0.76	0.83	0.30	0.44	0.65	2.66	0.79	0.16	0.26	0.44	4.66
SNNC [43]	0.48	0.29	0.36	0.26	1.52	0.44	0.19	0.27	0.20	2.08	0.33	0.13	0.19	0.09	2.26
NCUT [45]	0.88	0.13	0.23	0.55	6.32	0.71	0.20	0.31	0.48	3.34	0.59	0.23	0.33	0.40	2.28
MRF [49]	0.63	0.56	0.59	0.31	1.04	0.56	0.54	0.55	0.26	0.94	0.51	0.55	0.53	0.22	0.86
Proposed	0.90	0.77	0.83	0.81	1.10	0.74	0.76	0.75	0.58	0.86	0.64	0.65	0.64	0.39	0.86

Table 5 Comparison of clustering algorithms on $\mathcal{D}_5$

	1024 × 1024					512 × 512					512 × 256				
Algorithms	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{N}$
SC [31]	0.03	0.64	0.06	0.00	0.03	0.17	0.94	0.29	0.04	0.09	0.36	0.80	0.50	0.15	0.24
HC [34]	0.46	0.65	0.54	0.22	0.50	0.81	0.31	0.45	0.65	1.72	0.78	0.15	0.26	0.44	3.35
SNNC [43]	0.47	0.27	0.34	0.24	1.08	0.44	0.20	0.28	0.21	1.38	0.35	0.14	0.20	0.10	1.55
NCUT [45]	0.89	0.13	0.23	0.58	4.68	0.73	0.18	0.29	0.51	2.77	0.58	0.19	0.29	0.36	2.01
MRF [49]	0.60	0.54	0.57	0.29	0.70	0.55	0.55	0.55	0.24	0.62	0.43	0.57	0.49	0.16	0.49
Proposed	0.88	0.79	0.83	0.79	0.74	0.74	0.76	0.75	0.58	0.61	0.63	0.65	0.64	0.40	0.58

class size and easier separation of different classes, SC is able to produce much better results on $\mathcal{D}_1$ and $\mathcal{D}_2$, with $\mathcal{F} = 0.53, \mathcal{A} = 0.27$ for $\mathcal{D}_1$ and $\mathcal{F} = 0.51, \mathcal{A} = 0.17$ for $\mathcal{D}_2$.

The performance of HC is generally good in terms of F1-measure, but it is not as good as reported in [33] and [34], where the datasets used were less challenging in terms of both the number of cameras and the number of images captured by each camera. An interesting observation is that when using SPNs of 512 × 512 and 512 × 256 pixels, HC achieves the highest precision rates in most cases. But as can be seen in Tables 1, 3, 4, and 5, the high $\mathcal{P}$ comes at the expense of low $\mathcal{R}$, which means HC tends to over-partition the datasets. This is also reflected in the values of $\mathcal{N}$ that are much larger than 1 in the corresponding rows.

SNNC performs well on the easy datasets $\mathcal{D}_1$ and $\mathcal{D}_2$, but its performance on $\mathcal{D}_3 - \mathcal{D}_5$ is yet far from satisfactory. We found that SNNC is very sensitive to parameters. For example, if we increase *Eps* from 2 to 4, the performance drops dramatically for $\mathcal{D}_1$ (with $\mathcal{F} = 0.40, \mathcal{A} = 0.34$) and $\mathcal{D}_2$ (with $\mathcal{F} = 0.33, \mathcal{A} = 0.30$). Another drawback of SNN is that it does not cluster all data points, because it discards the non-core data points that are not within a radius of *Eps* of a core point (i.e., the noise points in [43]). When the parameters are not set appropriately, a large fraction of data points may be identified as noise points. Taking dataset $\mathcal{D}_3$ for example, about 25% of SPNs are identified as noise and discarded when *Eps* is set to 4. However, it is difficult to determine the "right" parameters that are applicable to different datasets.

Similar to HC, NCUT tends to over-partition the datasets, which results in high precision rates, low recall rates, and cluster-to-class ratios much higher than 1. It is worth noting that there are some inconsistencies between $\mathcal{F}$ and $\mathcal{A}$ measures. Taking the measures on dataset $\mathcal{D}_4$ using SPNs of 1024 × 1024 pixels (i.e., Table 3) for example, the $\mathcal{F} = 0.22$ of NCUT is the second worst among the six methods, but its $\mathcal{A} = 0.52$ turns out to be the second best performance. The main reason is that the measure $\mathcal{F}$ tends to be in favor of clusters of large granularity. To see this, let us consider clustering a dataset of 1000 images taken by 10 cameras, each responsible for 100 images. If all the 1000 images are grouped into one cluster, $\mathcal{A}$ gives a measure of 0 while $\mathcal{F}$ gives a measure of 0.18. If for each camera, its 80 images form a cluster and the remaining 20 images form 20 singleton clusters, then $\mathcal{A}$ gives a measure of 0.76, but $\mathcal{F}$ only gives a measure of 0.09. So $\mathcal{F}$ is more prone to heavily penalize singleton clusters.

By resorting to the MRF approach and the shared κ-nearest neighbor technique, our proposed algorithm is able to find high-quality clusters. Using SPNs of 1024 × 1024 pixels, it outperforms other five algorithms in terms of both F1-measure and adjusted Rand Index. It achieves $\mathcal{F} >= 0.86, \mathcal{A} >= 0.91$ on easy datasets ($\mathcal{D}_1$ and $\mathcal{D}_2$) and $\mathcal{F} >= 0.80, \mathcal{A} >= 0.74$ on hard datasets ($\mathcal{D}_3 - \mathcal{D}_5$). Even using the SPNs of 512 × 256 pixels, $\mathcal{F}$ and $\mathcal{A}$ can be as high as 0.72 on the two easy datasets. Compared with the MRF method in [49], the proposed algorithm shows a significantly better performance. On challenging datasets ($\mathcal{D}_3 - \mathcal{D}_5$), the improvement can be as high as 80% (e.g., $\mathcal{D}_3$) and 170% (e.g., $\mathcal{D}_5$) in terms of $\mathcal{F}$ and $\mathcal{A}$, respectively. But the 24 singleton classes added to $\mathcal{D}_5$ do decrease the precision rate as some of the singleton classes may be wrongly attributed to the clusters close to them. One attractive feature of our proposed algorithm is that it is able to find clusters with high precision rate (i.e., high purity). The high precision rate is important and preferable in the context of forensic investigation, because the false attribution error (i.e., $1 - \mathcal{P}$) can cause more serious problems, such as accusing an innocent person.

4.4.2 Clustering on datasets of varying sizes

In the second experiment, we aim to compare the time complexities and the clustering qualities of the six algorithms on datasets of varying sizes. To generate the datasets, we incrementally added 1000 images captured by 10 cameras (100 images per camera) to an empty dataset until all the 74 cameras in the Dresden database had been

covered. The six algorithms were run and evaluated on each of these datasets.

The log-scale running time (in seconds) is shown in Fig. 3. Our proposed algorithm requires the calculation of the SNN similarity before clustering, so the time used to calculate the SNN similarity is highlighted in green in the stacked bar. Since the running time obtained using SPNs of different lengths exhibit the same trend, we only show the running time for SPNs of 1024×1024 pixels. As can be observed in Fig. 3, MRF is the slowest one, followed by HC and SC. Although SC repeats spectral clustering process several times to search for the optimal number of clusters, the time complexity of each clustering process is only $\mathcal{O}\left(N^{\frac{3}{2}}K + NK^2\right)$ (K is the number of partitions), which is lower than the time complexity $\mathcal{O}(N^2 \log N)$ of HC when $N \gg m$. Our proposed algorithm is slightly slower than NCUT and SNNC but is much faster than the other three algorithms. By reducing the complexity of calculation in each iteration and accelerating the convergence, the speed of the algorithm has been significantly improved when compared to our preliminary study in [49]. Most of the running time of our proposed algorithm is spent on constructing the SNN similarity matrix. When the SNN similarity matrix is available, the actual clustering process only takes less than 40 s (the orange bar beneath the green bar in Fig. 3) even for the dataset containing 7400 images.

The clustering qualities of different algorithms are illustrated in Fig. 4. As can be seen, our proposed algorithm performs apparently better than the other three algorithms in terms of both the F1-measure and adjusted Rand Index. In particular, when using the SPNs of 1024×1024 pixels, our proposed algorithm delivers a 23% higher $\mathcal{F}$ and a 13% higher $\mathcal{A}$ on average than the second best algorithm (see Fig. 4g, j). Its precision rate consistently stays at a very high level ($>$ 96%). Even using the SPNs of 512×256 pixels, the precision rate can reach about 80%. The high precision rate makes the proposed algorithm attractive in forensic applications. Another important observation is that while the performances of the other five algorithms, especially SC and SNNC, decline considerably in terms of $\mathcal{F}$ or $\mathcal{A}$ when the size of the dataset increases, the performance of our proposed algorithm is quite stable in terms of $\mathcal{F}$, $\mathcal{A}$, and $\mathcal{N}$. This high stability is preferable in practice when applying the algorithm to new databases.

5 Conclusions

In this work, we have proposed a novel algorithm for clustering images taken by an unknown number and unknown types of digital cameras based on the sensor pattern noises extracted from images. The clustering algorithm infers the class memberships of images from a random initial membership configuration in the dataset. By giving different "neighbors" different voting power in the concise yet effective cost function depending on their similarity with the image in question, the algorithm is able to converge to the optimal cluster configuration accurately and efficiently. The experiments on the Dresden image database show that the proposed clustering scheme is fast and delivers very good performance. Despite the present advances, the most time-consuming step for image clustering based on SPNs is the calculation of the pairwise similarity matrix due to the high dimension of SPNs. We are currently working towards formulating a compact representation of SPNs in order to facilitate the large-scale source-oriented image clustering.

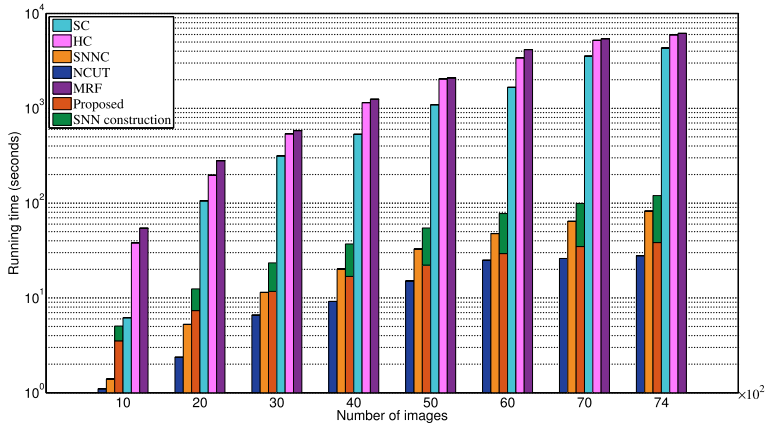

Fig. 3 Comparison of the running time (in seconds) of six clustering algorithms using SPNs of 1024×1024 pixels

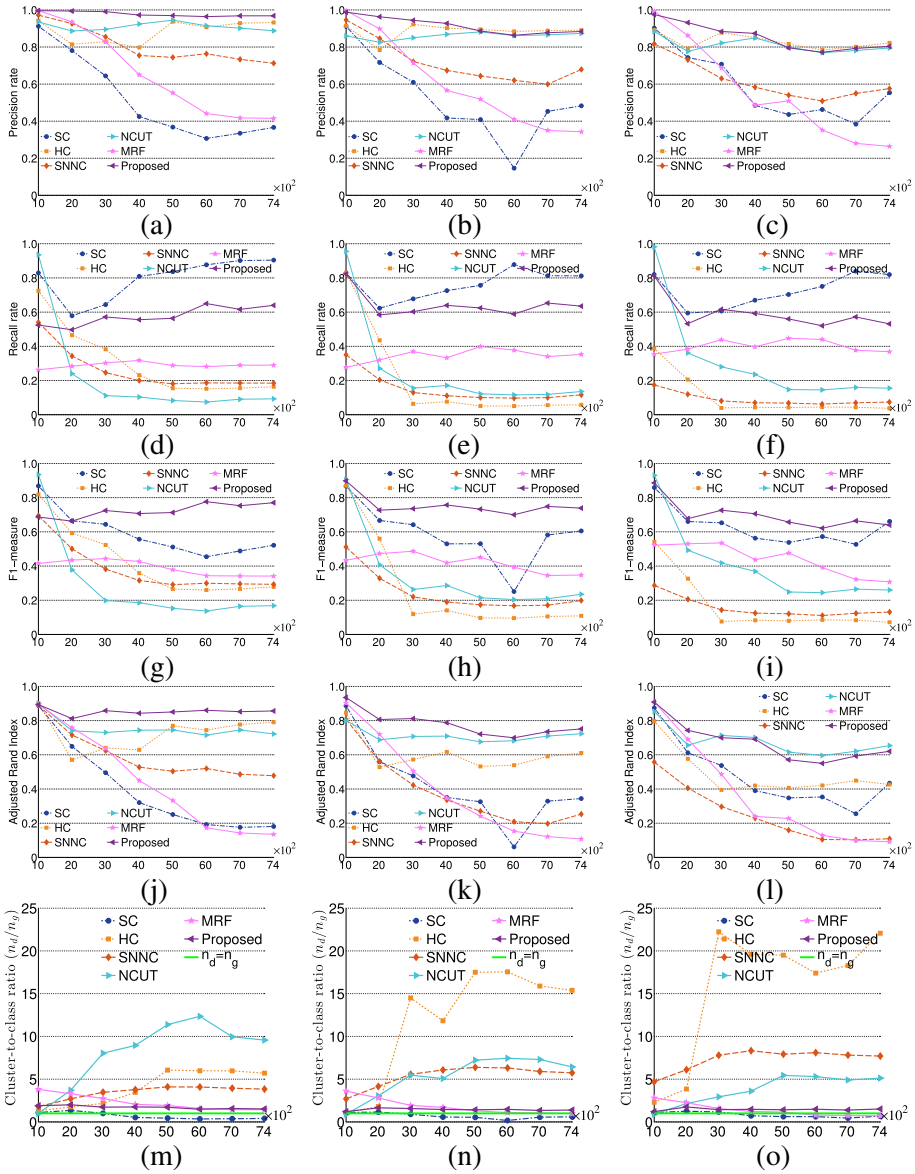

Fig. 4 Comparison of different clustering algorithms on datasets of varying sizes. **a** Precision rates, image block size $s = 1024 \times 1024$ pixels; **b** precision rates, $s = 512 \times 512$; **c** precision rates, $s = 512 \times 256$; **d** recall rates, $s = 1024 \times 1024$; **e** recall rates, $s = 512 \times 512$; **f** recall rates, $s = 512 \times 256$; **g** F1-measures, $s = 1024 \times 1024$; **h** F1-measures, $s = 512 \times 512$; **i** F1-measures, $s = 512 \times 256$; **j** adjusted Rand Indexes, $s = 1024 \times 1024$; **k** adjusted Rand Indexes, $s = 512 \times 512$; **l** adjusted Rand Indexes, $s = 512 \times 256$; **m** cluster-to-class ratios, $s = 1024 \times 1024$; **n** cluster-to-class ratios, $s = 512 \times 512$; and **o** cluster-to-class ratios, $s = 512 \times 256$

Acknowledgements
This work is partly supported by the EU project, Computer Vision Enabled Multimedia Forensics and People Identification (Project no. 690907; Acronym: IDENTITY), funded through the EU Horizon 2020 - Marie Skodowska-Curie Actions -Research and Innovation Staff Exchange action.

Funding
This study received funding from the EU project, Computer Vision Enabled Multimedia Forensics and People Identification (Project no. 690907; Acronym: IDENTITY), funded through the EU Horizon 2020 - Marie Skodowska-Curie Actions -Research and Innovation Staff Exchange action.

Availability of data and materials
The datasets analyzed during the current study are available in the Dresden Image Database, [http://forensics.inf.tu-dresden.de/ddimgdb/].

Authors' contributions
First, we propose a fast and reliable algorithm for clustering camera fingerprints. Aiming at overcoming the limitations of the work in [49], the proposed algorithm makes the following improvements: (1) redefining the similarity in terms of the shared nearest neighbors; (2) speeding up the calculation of the reference similarity; (3) refining the determination of the membership committee; (4) reducing the complexity of calculations in each

iteration; and (5) accelerating the speed of convergence. Not only the presentation of the clustering methodology is more comprehensive and detailed in this work, but also the proposed algorithm is much more efficient and reliable than that in [49]. Secondly, we discuss in detail the related SPNs clustering algorithms, namely the spectral, the hierarchical, the shared nearest neighbor, the normalized cut, and our previous MRF-based clustering methods [49]. These algorithms are evaluated and compared on real-world databases to provide insight into the pros and cons of each algorithm and offer a valuable reference for practical applications. Finally, we evaluate the proposed algorithm on a large and challenging image database which contains 7400 images taken by 74 cameras, covering 27 camera models and 14 brands, while the database used in [49] includes only six cameras. Furthermore, the quality of clustering is characterized by F1-measure and adjusted Rand Index, which are more suitable for evaluating clustering results than only the true positive rate or accuracy used in [31, 33, 34, 49]. Both authors read and approved the final manuscript.

Ethical approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Author details
[1]School of Computing and Mathematics, Charles Sturt University, Boorooma Street, Wagga Wagga, Australia. [2]Department of Computer Science, University of Warwick, Gibbet Hill Road, CV4 7AL Coventry, UK.

Received: 21 April 2017 Accepted: 3 October 2017
Published online: 16 October 2017

References

1. J Lukas, J Fridrich, M Goljan, Digital camera identification from sensor pattern noise. IEEE Trans. Inf. Forensics Secur. **1**(2), 205–214 (2006)
2. N Khanna, GT-C Chiu, JP Allebach, EJ Delp, in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process*. Forensic techniques for classifying scanner, computer generated and digital camera images, (Las Vegas, 2008), pp. 1653–1656
3. C-T Li, Source camera identification using enhanced sensor pattern noise. IEEE Trans. Inf. Forensics Secur. **5**(2), 280–287 (2010)
4. C-T Li, Y Li, Color-decoupled photo response non-uniformity for digital image forensics. IEEE Trans. Circ. Syst. Video Technol. **22**(2), 260–271 (2012)
5. X Kang, Y Li, Z Qu, J Huang, Enhancing source camera identification performance with a camera reference phase sensor pattern noise. IEEE Trans. Inf. Forensics Secur. **7**(2), 393–402 (2012)
6. Y Tomioka, Y Ito, H Kitazawa, Robust digital camera identification based on pairwise magnitude relations of clustered sensor pattern noise. IEEE Trans. Inf. Forensics Secur. **8**(12), 1986–1995 (2013)
7. A Pande, S Chen, P Mohapatra, J Zambreno, Hardware architecture for video authentication using sensor pattern noise. IEEE Trans. Circ. Syst. Video Technol. **24**(1), 157–167 (2014)
8. TH Thai, R Cogranne, F Retraint, Camera model identification based on the heteroscedastic noise model. IEEE Trans. Image Process. **23**(1), 250–263 (2014)
9. G Chierchia, G Poggi, C Sansone, L Verdoliva, A Bayesian-MRF approach for PRNU-based image forgery detection. IEEE Trans. Inf. Forensics Secur. **9**(4), 554–567 (2014)
10. S Chen, A Pande, K Zeng, P Mohapatra, Live video forensics: Source identification in lossy wireless networks. IEEE Trans. Inf. Forensics Secur. **10**(1), 28–39 (2015)
11. X Lin, C-T Li, Preprocessing reference sensor pattern noise via spectrum equalization. IEEE Trans. Inf. Forensics Secur. **11**(1), 126–140 (2016)
12. X Lin, C-T Li, Enhancing sensor pattern noise via filtering distortion removal. IEEE Signal Process. Lett. **23**(3), 381–385 (2016)
13. Y-F Hsu, S-F Chang, in *Proc. IEEE Int. Conf. Multimedia and Expo*. Image splicing detection using camera response function consistency and automatic segmentation, (Beijing, 2007), pp. 28–31
14. AC Popescu, H Farid, Exposing digital forgeries by detecting traces of resampling. IEEE Trans. Signal Process. **53**(2), 758–767 (2005)
15. H Cao, AC Kot, Accurate detection of demosaicing regularity for digital image forensics. IEEE Trans. Inf. Forensics Secur. **4**(4), 899–910 (2009)
16. A Swaminathan, M Wu, KR Liu, Nonintrusive component forensics of visual sensors using output images. IEEE Trans. Inf. Forensics Secur. **2**(1), 91–106 (2007)
17. MJ Sorell, in *Proc. Int. Conf. Forensic Appl. and Tech. in Telecom., Inf. Multimedia Workshop*. Conditions for effective detection and identification of primary quantization of re-quantized jpeg images (ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), (Adelaide, 2008), p. 18
18. K San Choi, EY Lam, KK Wong, in *Electron. Imag*. Source camera identification using footprints from lens aberration (Int. Society for Optics and Photonics, San Jose, 2006), pp. 60690–60690
19. O Celiktutan, B Sankur, I Avcibas, Blind identification of source cell-phone model. IEEE Trans. Inf. Forensics Secur. **3**(3), 553–566 (2008)
20. G Xu, S Gao, YQ Shi, R Hu, W Su, in *Proc. the 8th Int. Workshop Digit. Watermarking*. Camera-model identification using Markovian transition probability matrix, (2009), pp. 294–307
21. P Sutthiwan, J Ye, YQ Shi, in *Proc. the 8th Int. Workshop Digit. Watermarking*. An enhanced statistical approach to identifying photorealistic images, (University of Surrey, Guildford, 2009), pp. 323–335
22. I Amerini, R Becarelli, B Bertini, R Caldelli, Acquisition source identification through a blind image classification. IET Image Process. **9**(4), 329–337 (2015)
23. R-Z Wang, Y-D Tsai, An image-hiding method with high hiding capacity based on best-block matching and K-means clustering. Pattern Recogn. **40**(2), 398–409 (2007)
24. W Zhong, G Altun, R Harrison, PC Tai, Y Pan, Improved K-means clustering algorithm for exploring local protein sequence motifs representing common structural property. IEEE Trans. NanoBiosci. **4**(3), 255–265 (2005)
25. J Cui, J Loewy, EJ Kendall, Automated search for arthritic patterns in infrared spectra of synovial fluid using adaptive wavelets and fuzzy c-means analysis. IEEE Trans. Biomed. Eng. **53**(5), 800–809 (2006)
26. D Dembélé, P Kastner, Fuzzy c-means method for clustering microarray data. Bioinformatics. **19**(8), 973–980 (2003)
27. NR Pal, K Pal, JM Keller, JC Bezdek, A possibilistic fuzzy c-means clustering algorithm. IEEE Trans. Fuzzy Syst. **13**(4), 517–530 (2005)
28. B Hendrickson, RW Leland, A multi-level algorithm for partitioning graphs. Supter Comput. **95**, 28 (1995)
29. G Karypis, V Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comput. **20**(1), 359–392 (1998)
30. U Von Luxburg, A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
31. B Liu, H-K Lee, Y Hu, C-H Choi, in *Proc. IEEE Int. Workshop Inf. Forensics Security*. On classification of source cameras: A graph based approach, (Seattle, 2010), pp. 1–5
32. SX Yu, J Shi, in *Proc. IEEE Int. Conf. Comput. Vision*. Multiclass spectral clustering, (Nice, 2003), pp. 313–319
33. R Caldelli, I Amerini, F Picchioni, M Innocenti, in *Proc. IEEE Int. Workshop Inf. Forensics Security*. Fast image clustering of unknown source images, (Seattle, 2010), pp. 1–5
34. LJG Villalba, ALS Orozco, JR Corripio, Smartphone image clustering. Expert Syst. with Appl. **42**(4), 1927–1940 (2015)
35. S Guha, R Rastogi, K Shim, in *ACM SIGMOD Record*. Cure: an efficient clustering algorithm for large databases, vol. 27 (ACM, Seattle, 1998), pp. 73–84
36. S Guha, R Rastogi, K Shim, in *Proc. Int. Conf. Data Eng*. Rock: A robust clustering algorithm for categorical attributes (IEEE, Sydney, 1999), pp. 512–521
37. G Karypis, E-H Han, V Kumar, Chameleon: Hierarchical clustering using dynamic modeling. Computer. **32**(8), 68–75 (1999)
38. R Xu, D Wunsch, Survey of clustering algorithms. IEEE Trans. Neural Netw. **16**(3), 645–678 (2005)
39. M Chen, J Fridrich, M Goljan, J Lukás, Determining image origin and integrity using sensor noise. IEEE Trans. Inf. Forensics Secur. **3**(1), 74–90 (2008)

40. C-T Li, R Satta, Empirical investigation into the correlation between vignetting effect and the quality of sensor pattern noise. IET Comput. Vision. **6**(6), 560–566 (2012)
41. M Ester, H-P Kriegel, J Sander, X Xu, et al, in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. A density-based algorithm for discovering clusters in large spatial databases with noise, vol. 96, (Portland, 1996), pp. 226–231
42. A Hinneburg, DA Keim, in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. An efficient approach to clustering in large multimedia databases with noise, vol. 98, (New York, 1998), pp. 58–65
43. L Ertöz, M Steinbach, V Kumar, in *Proc. Int. Conf. Data Mining*. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data (SIAM, San Francisco, 2003), pp. 47–58
44. GJ Bloy, Blind camera fingerprinting and image clustering. IEEE Trans. Pattern Anal. Mach. Intell. **30**(3), 532–534 (2007)
45. I Amerini, R Caldelli, P Crescenzi, AD Mastio, A Marino, Blind image clustering based on the normalized cuts criterion for camera identification. Signal Process. Image Commun. **29**(8), 831–843 (2014)
46. J Shi, J Malik, Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
47. F Marra, G Poggi, C Sansone, L Verdoliva, in *Proc. Int. Workshop on Inf. Forensics and Security (WIFS)*. Correlation clustering for prnu-based blind image source identification, (Abu Dhabi, 2016), pp. 1–6
48. N Bansal, A Blum, S Chawla, Correlation clustering. Mach. Learn. **56**(1-3), 89–113 (2004)
49. C-T Li, in *Proc. IEEE Int. Symp. Circuits Syst*. Unsupervised classification of digital images using enhanced sensor pattern noise, (Paris, 2010), pp. 3429–3432
50. K Dabov, A Foi, V Katkovnik, K Egiazarian, Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE Trans. Image Process. **16**(8), 2080–2095 (2007)
51. RA Jarvis, EA Patrick, Clustering using a similarity measure based on shared near neighbors. IEEE Trans. Comput. **100**(11), 1025–1034 (1973)
52. T Gloe, S Pfennig, M Kirchner, in *Proc. ACM Workshop Multimedia Security*. Unexpected artefacts in PRNU-based camera identification: a 'Dresden Image Database' case-study, (Coventry, 2012), pp. 109–114
53. R Wilson, C-T Li, A class of discrete multiresolution random fields and its application to image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **25**(1), 42–56 (2003)
54. C-T Li, Multiresolution image segmentation integrating gibbs sampler and region merging algorithm. Signal Process. **83**(1), 67–78 (2003)
55. M Vignes, F Forbes, Gene clustering via integrated Markov models combining individual and pairwise features. IEEE/ACM Trans. Comput. Biol. Bioinf. **6**(2), 260–270 (2009)
56. X Zhang, X Hu, X Hu, E Park, X Zhou, Utilizing different link types to enhance document clustering based on Markov Random Field model with relaxation labeling. IEEE Trans. Syst. Man Cybern. Part A: Syst. Humans. **42**(5), 1167–1182 (2012)
57. AL Varna, M Wu, in *Proc. IEEE Int. Workshop Inf. Forensics Security*. Modeling content fingerprints using Markov random fields, (London, 2009), pp. 111–115
58. C Martınez, in *Proc. ACM-SIAM Workshop on Algorithm Engineering and Experiments and 1st ACM-SIAM Workshop on Analytic Algorithmics and Combinatorics*. Partial quicksort, (New Orleans, 2004), pp. 224–228
59. X Lin, CT Li, Large-scale image clustering based on camera fingerprints. IEEE Trans. Inf. Forensics Security. **12**(4), 793–808 (2017)
60. T Gloe, R Böhme, The Dresden image database for benchmarking digital image forensics. J. Digit. Forensic Practice. **3**(2-4), 150–159 (2010)
61. WM Rand, Objective criteria for the evaluation of clustering methods. J. Am. Stat. Assoc. **66**(336), 846–850 (1971)
62. L Hubert, P Arabie, Comparing partitions. J. Classification. **2**(1), 193–218 (1985)