Neurocomputing 358 (2019) 150–165

Contents lists available at ScienceDirect

Neurocomputing

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

Combining clustering and active learning for the detection and learning of new image classes

Luiz F.S. Coletta[a,*], Moacir Ponti[b], Eduardo R. Hruschka[b], Ayan Acharya[c], Joydeep Ghosh[c]

[a] School of Sciences and Engineering, São Paulo State University, Tupã, SP, Brazil
[b] University of São Paulo, SP, Brazil
[c] Department of Electrical & Computer Engineering, University of Texas, Austin, USA

ARTICLE INFO

Article history:
Received 21 May 2018
Revised 24 March 2019
Accepted 27 April 2019
Available online 10 May 2019

Communicated by Shiliang Sun

Keywords:
Image classification
Active learning
Clustering
Open set
Deep learning

ABSTRACT

Discriminative classification models often assume all classes are available at the training phase. As such models do not have a strategy to learn new concepts from available unlabeled instances, they usually work poorly when unknown classes emerge from future data to be classified. To address the appearance of new classes, some authors have developed approaches to transfer knowledge from known to unknown classes. Our study provides a more flexible approach to learn new (visual) classes that emerge over time. The key idea is materialized by an iterative classifier that combines Support Vector Machines with clustering via an optimization algorithm. An entropy and density-based selection strategy explores label uncertainty and high-density regions from unlabeled data to be classified. Selected instances from new classes are submitted to get labels and then used to improve the model. The proposed image classifier is consistently better than approaches that select instances randomly or from clusters. We also show that features obtained via Deep Learning methods improve results when compared with shallow features, but only using our selection strategy. Our approach requires fewer iterations to learn new classes, thereby significantly saving labeling costs.

1. Introduction

Digital visual content is responsible for a significant share of the current network traffic [1]. A huge number of images is constantly uploaded into the various platforms for image collection storage and visualization. Over the last decades, studies on image representations, retrieval and classification devoted efforts to develop increasingly accurate methods to better describe visual content, allowing indexing data based on image content, and also classifying images [2–5], which is essentially to assign labels to some image, given a predefined semantics.

As a rule of thumb, classification methods based on discriminative models (e.g., Logistic Regression, Support Vector Machines, and Deep Neural Networks) typically require training sets with a large number of labeled instances, which must be representative enough to correctly classify unseen (unlabeled) data [6]. However, although gathering masses of image data is easier nowadays, the process of labeling instances to compose a training set is an expensive and error-prone task [7,8]. Also, available image collections are often cluttered and unclean. Hence, there are cases where only a limited *source* of annotated data is in hand [9,10]. After training with this ill-posed *source*, many techniques do not fare well on new *target* data that represents a somewhat different input distribution [11,12]. For example, Deep Learning (DL) has been in the recipe for success of state-of-the-art visual recognition methods [13]. Having thousands of labeled instances (images) for each known class of the problem, researchers have reached impressive low error rates by using DL schemes [14–16]. However even such classification methods suffer from limitations, in particular when addressing problems with limited annotated training set [17].

Due to idiosyncrasies of the sampling process [18,19] and the lack of knowledge about the entire set of possible classes [20], in many practical domains one does not have instances of all classes during the training phase — i.e., instances of one or more classes could be missing while building a classification model. In particular, although some methods allow incrementally learning new classes [21], those often do not include a strategy to detect such new classes via unlabeled instances. This shortcoming has motivated our research, which specifically addresses the problem of learning new unknown classes in future data.

* Corresponding author.
E-mail addresses: luiz.coletta@unesp.br (L.F.S. Coletta), moacir@icmc.usp.br (M. Ponti), hruschka@usp.br (E.R. Hruschka), aacharya@utexas.edu (A. Acharya), ghosh@ece.utexas.edu (J. Ghosh).

https://doi.org/10.1016/j.neucom.2019.04.070
0925-2312/

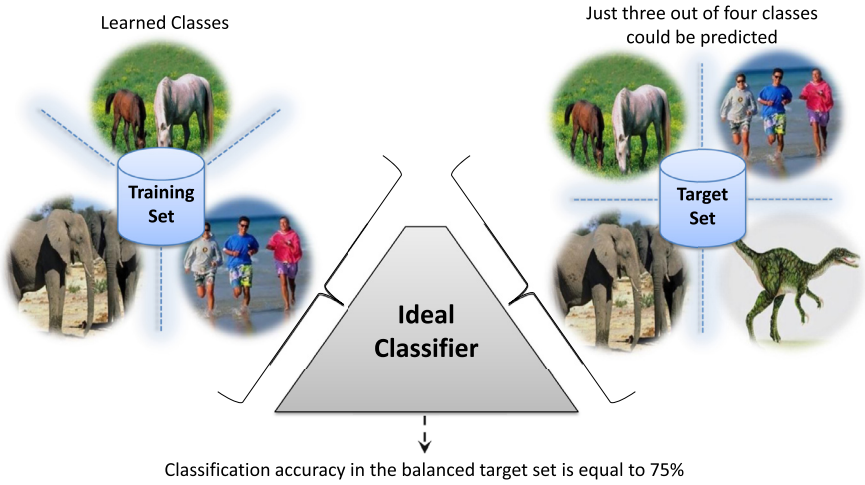

Fig. 1. A hypothetical ideal classifier trained to recognize horses, people, and elephants, makes mistakes in the presence of a new class of images representing dinosaurs.

This paper presents an investigation on classifiers that can deal with the emergence of instances from a new class, which should be quickly detected and learned. We are particularly interested in application scenarios for which there are small training sets, with specific visual classes missing. Image classification systems trained on these limited training sets will perform poorly when, among the images to be classified subsequently, some belong to unknown classes. For example, Fig. 1 considers that an *ideal classifier* has been induced from a training set with instances of three classes: "*Horses*", "*People*", and "*Elephants*". By inferring the class label of images in a balanced target set, which includes unlabeled instances from a new class "*Dinosaurs*", the classification accuracy cannot achieve more than 75% (because in reality this is a four-class problem). As the classifier has not learned the new class, it will predict dinosaurs as being horses, people, or elephants. In other words, classifiers commonly neglect the existence of new *concepts* by assuming a unique finite set of classes [20,22–24]. They are highly dependent on prior knowledge and typically do not support *concept changes* or the appearance of a new class.

Our main contributions are as follows. First, we propose a flexible iterative classifier, which allows the combination of practically any supervised algorithms with unsupervised ones aiming to improve classification results. Specifically, we have combined the simple and efficient *k*-Means algorithm [25] with an SVM [26] trained on both handcrafted and deep features. These settings were chosen because they show to be a competitive combination of alternatives for image classification [18,21,27,28]. Hence, our iterative classifier can be seen as an *ensemble*, which allows the coupling of different (un)supervised algorithms gathering their capabilities to achieve better results. Such a mechanism has been applicable in many real-world problems by means of the C^{3}E (Consensus between Classification and Clustering Ensembles) algorithm [29–31]. Second, we extended our iterative classifier to detect new classes taking advantage of unsupervised information generated by clustering algorithms. In more detail, our approach explores *label uncertainty* and *high-density regions* to find unlabeled instances that belong to new classes. Using active learning [32], some selected instances are labeled (by a domain expert) and then incorporated into the model to improve classification. By doing so, our classifier can learn new concepts/classes in an iterative fashion, being more suitable to real-world problems where incomplete knowledge occurs [20] and the adaptation of the model needs to recognize new classes over time [8,33]. Our third contribution is based on the image representation via Deep Learning (DL) methods. As our strategy to select instances from new classes in combination with deep features achieved very good results, we anticipate that it makes room for applications of DL in scenarios with few and/or missing classes instances. As already known, approaches based on DL [3,34] require a large amount of labeled data to achieve very good predictions and are based on a unique and finite set of known classes. Our iterative classifier relaxes this assumption by recognizing new classes, even with few labeled data, from the combination of different methods, including those based on DL.

The remainder of the paper is organized as follows. Section 2 addresses related studies and theoretical foundations on which our approach is grounded. Section 3 introduces the proposed iterative classifier, based on the C^{3}E algorithm, for the detection and learning of new classes. Section 4 outlines our experimental setup and the used datasets. Section 5 addresses experimental results and a discussion to show that the resulting image classifier can successfully handle new classes. Section 6 provides the conclusions and suggests directions for further research.

2. Related work

Image classification systems should ideally recognize a large number of visual classes. To reach this goal, some studies have concentrated efforts to smartly sample representative training sets [18,19]. In this context, active learning methods can select the most informative instances from unlabeled data to help building better classifiers [32]. To do this, they can take into account margin sampling, in which candidate instances to be in the training set are those that lie within the margin of Support Vector Machines (SVMs) [35]. Other approaches are based on a disagreement coefficient measured between different views (from independent and redundant sets of features) or different classifiers [36], so that instances that maximize such a criterion are preferable for selection. For this, some authors have also used multiple views and learners simultaneously [37]. Another strategy considers the selection of the most uncertain instances from an estimation of their posterior probability distribution of classes [38]. Similarly, Huang et al. [39] suggested to combine informative and representative measures for unlabeled instances. As their work is restricted to binary classification, some authors have combined uncertainty and diversity focusing on multi-class problems [40].

In order to situate our contribution within the literature, we here anticipate that our paper reports a method that makes use of concepts from active learning, classification, and clustering. It focuses on detecting new concepts and allows to learn them with

a few labeled instances. In comparison, other active learning studies often neglect scenarios in which one or more classes are not included in the first model, e.g. [38,40]. In addition, our approach allows combining the outputs from standard clustering and classification methods and requires minimal adaptations to discover new classes. Therefore a direct empirical comparison between our method and the frameworks in [38,40] would be difficult and unfair. On the other hand, studies that report comparisons of sampling techniques often point to random selection as a standard baseline, with competitive results when compared to many other strategies [39]. A more competitive approach that complements the comparison with a uniform random strategy is to obtain the instances by sampling from clusters. For example, selecting k instances as those nearest to the cluster centroids of a k-Means output [39,41].

The above studies emphasize that sampling limitations, as well as underlying supervised classification assumptions, can hinder suitable learning. From this perspective, classifiers capable of self-adaptation are essential, particularly when new unknown classes of images appear.

To address the emergence of new classes, Jun and Ghosh [10] proposed a semi-supervised spatially adaptive mixture model that enables the detection of unknown land-cover classes from hyperspectral images. Focusing on unsupervised visual category learning, Lee and Grauman [42] introduced a *context-aware* discovery algorithm that captures interactions among objects within images so that co-occurrences can identify new categories. Bart and Ullman [43] used feature (*image patch*) adaptation to produce a cross-generalization able to learn a new class from a single instance. They assumed that a feature is effective for a new class if a similar feature has proved useful for a previously learned class. Likewise, Lampert et al. [8,33] utilized attribute-based classification to transfer knowledge between classes and recognize new image classes that have no training instance. To do so, *high-level attributes* are learned in an intermediate step of a cascade classifier. Inspired by these works, *zero/one-shot learning* approaches have provided interesting results [44,45].

Scheirer et al. [20] employed the term *"Open Set"* to outline a more realistic classification scenario, in which incomplete knowledge of the world is present at training time and unknown classes can emerge over time. In contrast, they also denoted the restricted scenario, referred to by traditional classifiers as a *"Closed Set"*, in which a unique finite set of known classes is considered. The authors investigated open set problems by using a *1-vs-set machine*, which is an extension of 1-class and binary SVMs in which an additional plane is used to optimize the empirical and open space risk. By specializing the two existing planes, the classifier limits the positive region and avoids extending decision boundaries beyond the negative and unknown regions. In [11], the authors extended the idea to non-linear classifiers in a multi-class setting.

From a broader perspective, researchers from different areas have investigated methods to automatically learn concepts dissimilar from those already known [46,47]. In data stream applications, clustering has been used to identify changes in the underlying data distribution (*concept-drift*) and the emergence of new classes (*concept-evolution*) [48,49]. As the aforementioned studies, we have used the information provided by clustering algorithms to explore available unlabeled data. From this, and contrary to Riva et al. [21], new unknown concepts/classes are incrementally learned through a strategy that selects unlabeled instances representing novelty. Even when we are dealing with multi-class problems, our method differs from that introduced by Scheirer et al. [11], which only recognizes unknown classes to keep them apart from the class of interest. In more detail, our iterative classifier can incorporate and learn new classes over time. Unlike the mechanisms used by Lampert et al. [8,33], no coupling between known and unknown classes is required. In other words, our approach does not require transfer learning, but it is an alternative to fulfill a portion of the requirements addressed in Section 7.1 of [33] — particularly the following question raised by the authors: *"How can we build object recognition systems that adapt and incorporate new categories encountered?"*

In the context of image recognition tasks, Deep Learning (DL) methods are relevant, requiring huge amounts of labeled data [3,15]. But there are application scenarios where only a limited source of annotated data is available, which makes it difficult to train DL architectures [2,17,50]. Besides, incomplete knowledge (with missing classes) during the training phase requires self-adaptable models, which is little explored in DL frameworks. Such a gap is of our interest as well, so our method can be a step toward such a flexibility. Instead of using DL for classification we employed our iterative classifier on features extracted via a pre-trained Convolutional Neural Network (CNN), which is a common practice for finding feature embeddings [51]. As addressed next, our iterative classifier explore *label uncertainty* and *high-density regions* to discover new classes of images, which can be represented by either handcrafted global features, or deep features.

3. Detecting and learning new classes

As clustering algorithms naturally capture similarities between *objects*, similarity matrices constructed from cluster ensembles can be a source of supplementary information about the data explored [31,52] and can help, for example, in the identification of *high-density regions*. We have used this kind of information, along with estimations of the *label uncertainty*, to explore new classes of images. Our approach is materialized by an iterative classifier that combines classification and clustering to detect and learn new classes. The starting point was the C^{3}E algorithm [30,31,53], which is briefly revisited in Section 3.1. The extension of this algorithm to address new classes is introduced in Section 3.2.

3.1. Review of C^{3}E

Acharya et al. [31] introduced a framework that combines classifiers and clustering algorithms to improve the generalization capability of classification. Its core is a general optimization algorithm, named C^{3}E (Consensus between Classification and Clustering Ensembles), which explores a large class of loss functions [30]. Coletta et al. [53] investigated a simpler version of C^{3}E by employing a Squared Loss function (C^{3}E-SL). This algorithm has provided attractive empirical results, as it is flexible and computationally efficient in practice.

C^{3}E-SL assumes that an ensemble of classifiers (consisting of one or more classifiers) has been previously induced from a training set. This ensemble estimates initial class probability distributions for every instance $\mathbf{x}_i$ of a target/test set $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$. Such distributions are stored as c-dimensional vectors, $\{\boldsymbol{\pi}_i\}_{i=1}^n$, where c is the number of classes. Cluster ensembles can provide soft constraints for classifying the instances of $\mathcal{X}$. The rationale is that similar instances, found by clustering algorithms, are more likely to share the same class label. The probability distributions in $\{\boldsymbol{\pi}_i\}_{i=1}^n$ can be refined by using a similarity matrix $\mathbf{S} = \{s_{ij}\}_{i,j=1}^n$. $\mathbf{S}$ captures similarities between the instances of $\mathcal{X}$ so that each entry corresponds to the relative co-occurrence of two instances in the same cluster [52] (considering all of the data partitions built on $\mathcal{X}$).

In summary, C^{3}E-SL receives as input a set of vectors, $\{\boldsymbol{\pi}_i\}_{i=1}^n$, and a similarity matrix, $\mathbf{S}$, and outputs a consolidated classification for every instance in $\mathcal{X}$ which is represented by a set of vectors $\{\mathbf{y}_i\}_{i=1}^n - \mathbf{y}_i = p(C \mid \mathbf{x}_i)$, i.e., $\mathbf{y}_i$ is the estimated posterior class probability assignment for every instance in $\mathcal{X}$. To do so, C^{3}E-SL solves an optimization problem whose objective is to minimize J

in (1) with respect to the set of probability vectors $\{\mathbf{y}_i\}_{i=1}^{n}$:

$$J = \frac{1}{2}\sum_{i\in\mathcal{X}} \|\mathbf{y}_i - \boldsymbol{\pi}_i\|^2 + \alpha\frac{1}{2}\sum_{(i,j)\in\mathcal{X}} s_{ij}\|\mathbf{y}_i - \mathbf{y}_j\|^2 \,. \tag{1}$$

Algorithm 1 describes an update procedure that minimizes J in (1). In more detail, keeping $\{\mathbf{y}_j\}_{j=1}^{n} \setminus \{\mathbf{y}_i\}$ fixed, we can minimize J for every $\mathbf{y}_i$ by iteratively computing Eq. (2). The number of iterations, I, and the parameter that controls the relative importance of classifiers and cluster ensembles, α, can be automatically estimated from data following the parameter optimization procedure proposed in [53].

Algorithm 1: C^{3}E with Squared Loss function – C^{3}E-SL [53].

Input: $\{\boldsymbol{\pi}_i\}, \mathbf{S}, \alpha, I$.
Output: $\{\mathbf{y}_i\}$.
Initialize $\{\mathbf{y}_i\}$ such that $y_{i\ell} = \frac{1}{c}\ \forall i \in \{1,2,...,n\}$, $\forall \ell \in \{1,2,...,c\}$;
Repeat
  Update $\mathbf{y}_i\ \forall i \in \{1,2,...,n\}$ via Equation (2):

$$\mathbf{y}_i = \frac{\boldsymbol{\pi}_i + 2\alpha\sum_{j\neq i} s_{ij}\mathbf{y}_j}{1 + 2\alpha\sum_{j\neq i} s_{ij}}, \tag{2}$$

until *the number of iterations reaches* I;

The combination of classification and clustering has been shown to be useful for designing learning methods that are aware of the differences between training and target distributions [30]. We believe that this aspect can help in the exploration of future data, particularly for discovering new concepts/classes not observed during training.

3.2. Proposed iterative classifier

Our proposed classifier is an extension of the C^{3}E-SL algorithm that iteratively classifies images. Therefore, it can self-adapt over time to improve the classification model. In each iteration, a specialized search can detect new classes by exploring unlabeled instances that have highly uncertain labels and are located in dense regions, which are identified from clusters. Essentially, such instances might denote some novelty triggered by concept drift (i.e., from changes in the distribution of classes) or even be representative of new classes not observed during training. Highly uncertain labels from a particular classification can be captured based on entropy measures [40,54]. Given the c-dimensional probability vectors yielded by C^{3}E-SL, $\{\mathbf{y}_i\}_{i=1}^{n}$, the *classification entropy* for an instance i can be computed by Eq. (3):

$$e_i = \frac{-\sum_{j=1}^{c} y_{ij}\log_2 y_{ij}}{\log_2 c}. \tag{3}$$

Now, we aim at combining the classification entropy of images with the information about their surrounding densities, which can be computed from similarity matrices (as those processed by C^{3}E-SL). Let $\mathbf{H}^i$ be the set containing the h-nearest neighbors of a given instance i. The density around i can be computed as

$$d_i = \frac{1}{h}\sum_{j\in\mathbf{H}^i} s_{ij}\ , \tag{4}$$

where the values of s_{ij} correspond to the entries in a similarity matrix $\mathbf{S}$.

We have assumed that instances that are candidates for new classes are those from high-density regions and with high classification entropy (with respect to the known classes). Therefore, a candidate to represent a new class can be obtained by choosing the instance i for which $(e_i \times d_i)$ is maximized. We capture this notion with Entropy and Density-based Selection (EDS), which is summarized by Algorithm 2, whose output is a set of instances that are more likely to belong to new classes. To do so, it receives as input the entropy vector $\mathbf{e} = \{e_1, e_2, ..., e_n\}$, the density vector $\mathbf{d} = \{d_1, d_2, ..., d_n\}$, and a scalar P_2, which represents the number of selected candidates. Note that, in Step 5, we want to select P_2 instances that are as dissimilar as possible to each other so that these can significantly contribute to the improvement of the model. The output, $\boldsymbol{\Omega}$, stores the indexes of the P_2 instances from a target set ($\mathcal{X}$) that are more likely to belong to new classes.

Algorithm 2: Entropy and Density-based Selection (EDS).

Input: $\mathbf{e}, \mathbf{d}, P_2, \mathbf{S}_{n\times n}$.
Output: $\boldsymbol{\Omega} = \{\omega_j\}_{j=1}^{P_2}$.
For $j \leftarrow 1$ **to** P_2 **do**
  If $j = 1$ **then**
    $\omega_j = \underset{1\leqslant i\leqslant n}{arg\ max}\ [e_i \cdot d_i]$;
  else
    $\omega_j = \underset{1\leqslant i\leqslant n}{arg\ max}\ [e_i \cdot d_i \cdot (1 - \frac{1}{j-1}\sum_{r\in\boldsymbol{\Omega}} s_{ir})]$;
  end
  $e_{\omega_j} = -\infty$ *(to select distinct instances)*;
end

Based on active learning, the instances selected by Algorithm 2 (EDS) can be labeled by a domain expert so that new classes can be discovered. Accordingly, these instances can be incorporated into the training set, and then the model can be retrained. Specifically, C^{3}E-SL can be run again to infer the class of the remaining unlabeled instances in $\mathcal{X}$ – assuming that the supervised component was rebuilt from the updated training set and the similarity matrix $\mathbf{S}$ was reduced by the elimination of the corresponding entries of the P_2 selected instances. Algorithm 3 summarizes the above steps. Such a process can be repeated several times for the detection and learning of new classes. For the sake of simplicity, our iterative classifier, which combines Algorithms 3 (IC) and 2 (EDS), is named IC-EDS.

Algorithm 3: Iterative Classifier (IC) for learning new classes.

Input: $\{\boldsymbol{\pi}_i\}_{i=1}^{n}, \mathbf{S}_{n\times n}$.
Output: $\{\mathbf{y}_i\}_{i=1}^{q}$.
Run C^{3}E-SL from $\{\boldsymbol{\pi}_i\}_{i=1}^{n}$ and $\mathbf{S}_{n\times n}$ to provide $\{\mathbf{y}_i\}_{i=1}^{n}$ – *the classifier ensemble was built from* P_1 *labeled instances*;
$P_{tot} = 0$;
Repeat
  Obtain labels for P_2 instances provided by Algorithm 2 – $\mathbf{e}$ *and* $\mathbf{d}$ *are computed via Equations (3) and (4), respectively*;
  $P_{tot} = P_{tot} + P_2$;
  $q = n - P_{tot}$;
  Build a classifier ensemble from $(P_{tot} + P_1)$ labeled instances to obtain $\{\boldsymbol{\pi}'_i\}_{i=1}^{q}$;
  Update matrix $\mathbf{S}'_{q\times q}$ (*removing the rows and columns related to the* P_2 *labeled instances*);
  Run C^{3}E-SL from $\{\boldsymbol{\pi}'_i\}_{i=1}^{q}$ and $\mathbf{S}'_{q\times q}$ to provide $\{\mathbf{y}_i\}_{i=1}^{q}$;
until *a certain number of iterations is reached*;

4. Experimental setup

Experiments were based on the holdout method [55], in which datasets are randomly split into training and target/test sets. Instances from the target/test sets were not used to induce the classifiers at all, i.e., they constitute independent data not used to optimize any parameter of the resulting classifiers. More precisely, a certain number of labeled images (20% of the dataset) was used to train and validate the classifiers, and the remainder (the target set) was used to test them. To simulate the emergence of new classes, instances from a particular class were left out from the initial training sets, whereas the target sets always contained instances from all classes. To increase confidence in the results, algorithms were run five times per left out class, taking into account distinct initial training sets obtained from stratified random sampling (without replacement).

4.1. Setting up the algorithms

The iterative classifier introduced in Section 3.2 (IC-EDS) employs the C^{3}E-SL algorithm, which requires the information of two user-defined parameters (α and I). To automatically optimize these parameters from data, we adopted practical guidelines as in [53], such that an additional step for parameter optimization was performed (on each training set). Firstly, an SVM classifier was built on half of the available labeled instances. A dynamic differential evolution algorithm [53] then estimates the optimal pair of values, α^* and I^*, by minimizing the C^{3}E-SL misclassification rates in a validation set. This set contains the other half of the labeled instances where, as required by C^{3}E-SL, a cluster ensemble was induced. The best values estimated for α and I were fixed and used in IC-EDS.

In our study, C^{3}E-SL refines SVM classification with the help of information provided by clusters. Parameters C and γ of the non-linear SVM used were estimated by grid-search on the training set, as in Hsu et al. [56]. To obtain the components of the cluster ensemble, four sets of clusters (data partitions) were generated. Each set was produced by a particular subset of features[1]. We adopted the *ordered multiple runs* procedure [57], in which k-Means based on L2–norm (Euclidean distance) [25] is run 20 times for every number of clusters $k = \{k_{\min}, k_{min+1}, k_{\min+2}, ..., k_{\max}\}$, $k_{\min} = 2c$ and $k_{\max} = \sqrt{n}$. Therefore, each set has data partitions with different numbers of clusters, which were generated from different initialization. To build a similarity matrix, the best partition of each set, according to the silhouette criterion [57], is used.

Finally, from a (not comprehensive) empirical analysis we have used $h = 5$ to compute the densities around data points — Eq. (4). Besides, we have fixed five iterations for the Algorithm 3, each one selecting $P_2 = 5$ instances for labeling. Thus, 25 instances were used to improve the model and discover new classes.

4.2. Datasets and features

Experiments were carried out on the following datasets:

- Caltech-6, which is a subset of Caltech-101[2], containing 100 randomly selected images from the following classes: airplane, bonsai, chandelier, hawksbill, motorbike, and watch as compiled by [58].
- Coil-20[3] dataset, which consists of 1440 gray-scale images of 128×128 in size with a black background for 20 different objects. Each object was recorded under 72 different viewing angles [59];
- Corel-10[4] dataset that contains 1000 color photographs divided into 10 balanced categories — africans, beach, buildings, buses, dinosaurs, elephants, flowers, horses, mountains, and food [60];
- Supermarket Produce dataset comprises 1400 fruit/vegetable images divided into 14 balanced categories — agata potato, asterix potato, cashew, diamond peach, fuji apple, granny-smith apple, honeydew melon, kiwi, nectarine, orange, plum, williams pear, taiti lime, and watermelon. Instances of each class vary in lighting, position of the elements within the image, and the presence of cropping, occlusions, and shadows [61].

It is well known that Caltech-6 contains classes that are difficult to classify with global features, but better discriminated with Deep Learning (DL) features. Also, a well-behaved dataset (Coil-20) was used, followed by a dataset with natural images and more class overlapping (Corel-10). Finally, a dataset that has some confusion between specific classes (Supermarket Produce) was used. Fig. 2 shows the classes of each dataset. These datasets were described by global color and/or texture visual features, and also by DL features as follows:

Handcrafted global features: global descriptors are fast to compute and produce good overall performance for image retrieval and classification as shown by Penatti et al. [62]. Such features are obtained via extraction methods that operate on single channel images therefore requiring preprocessing RGB input images. Following guidelines for image processing for feature extraction [58], images must be quantized to 1 channel 8-bit/pixel or less [63]. Then, the following color and texture features — those with the best performance as found in previous studies [58,62] — were computed:

- **Border Interior Classification (BIC)**: a color extraction method that can encode structural information, i.e., the spatial distribution of colors throughout the image. This method generates a representation of the image color distribution by computing two histograms: one for border pixels and another for interior pixels. A pixel is classified as *border* if at least one of its neighbors has a different color, and it is classified as *interior* otherwise [64]. The final vector size has dimensions of $C \times 2$, where C is the number of color levels in the images;
- **Haralick-6**: based on gray-level co-occurrence matrices, this is one of the most used texture descriptors. It first calculates a co-occurrence matrix with size $C \times C$ using a fixed relationship of $\Delta(x, y) = (1, 0)$ between pairs of pixels. Then, it extracts 6 Haralick features from this matrix: Entropy, Homogeneity, Contrast, Correlation, Maximum Probability, and Uniformity [65].

The final global feature vector for each dataset was computed as follows:

- *Coil-20:* the grayscale images were quantized in 8 color levels. The regions corresponding to the background were ignored, resulting in 15 BIC features. The Haralick-6 was also computed for this dataset, and concatenated with the BIC descriptor, resulting in 21 dimensions;
- *Corel-10* and *Supermarket Produce*: the RGB images were quantized in 64 color levels; only the BIC features were extracted, resulting in 128 dimensions.

The choice of features as well as the quantization levels for each dataset were defined following previous studies [58,62]. While Coil-20 is a grayscale image dataset, therefore requiring texture to complement the color description (in terms of intensity levels), the other two datasets are well described by color features only.

Deep features using VGGNet-16: those features were obtained by using the activation maps of a VGGNet-16 model pre-trained on the ImageNet dataset. By performing a forward pass using the

[1] The subset of features was formed by 20% of the original features, which were randomly selected (without replacement).

[2] http://www.vision.caltech.edu/Image_Datasets/Caltech101/.

[3] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php.

[4] http://wang.ist.psu.edu/docs/related/.

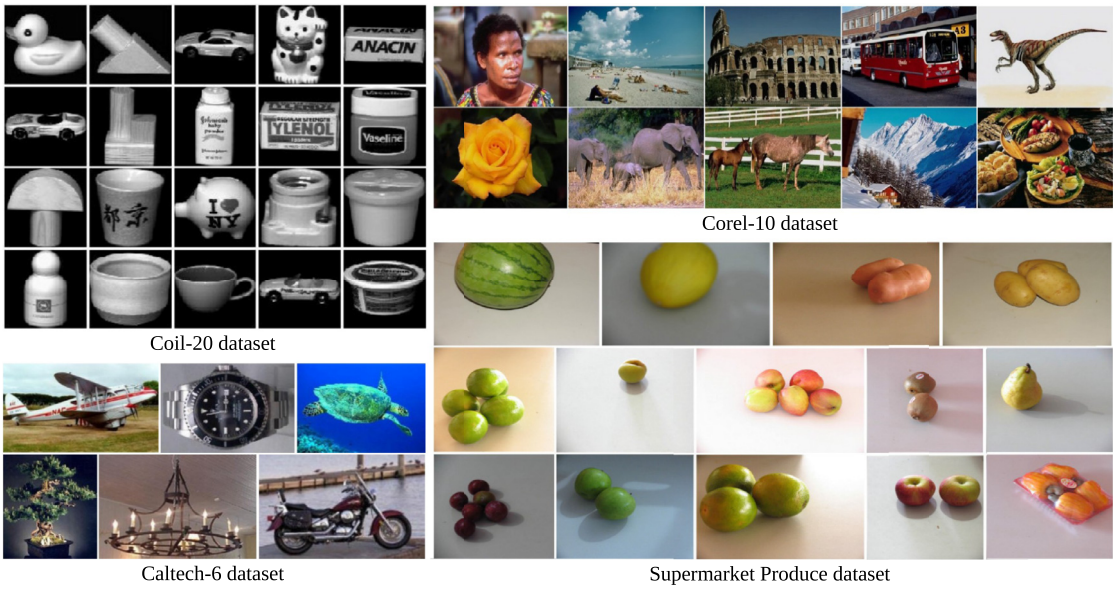

Fig. 2. Instances of each class of the datasets used in the experiments.

whole image resized to 224×224, we extracted the output of the first fully connected (dense) layer of the VGGNet-16 architectures (FC1) forming a vector with 4,096 values. This process has shown to produce good general purpose features for image classification [51]. A Principal Component Analysis (PCA) was then employed to obtain the first 128 principal components, resulting in a dimensionality that is comparable to those used in the global features.

5. Empirical evaluation

This section provides a thorough experimental analysis of the IC-EDS algorithm introduced in Section 3.2. For comparison purposes, IC-EDS was also assessed against two baseline algorithms often employed in active learning studies [39,40]. Such comparison is made by replacing our proposed search (Algorithm 2) by the baseline approaches. The first, named IC-RS, performs a uniform sampling of instances to be labeled as in Algorithm 3 (IC) but with a Random Selection (RS) in Step 4. The second baseline is a clustering-based method referred to as IC-KM that uses k-Means (in Step 4 of Algorithm 3) to select the k instances as those nearest to the centroids of the k clusters. All algorithms were implemented in Matlab and to obtain their inputs the Weka API[5] was used.

In theory, IC-EDS should get better results than IC-RS because it performs a smarter search for new concepts/classes. By uniformly sampling the examples, IC-RS may also find new classes. Therefore, we first show that our method performs better than this random search. The clustering method, IC-KM, should be able find new classes as isolated clusters. In order to avoid cluttered display of the results, we only show the global performance of the IC-KM method in comparison with the other selection methods. The comparison between the three methods, which is essential for validating the selection of candidate instances, is reported after illustrating the behaviour of our IC-EDS on two simpler datasets. All these experiments were run in a computer with 1.9 GHz Intel Quad-Core i7-8650U and 16 GB of RAM running Linux.

5.1. Illustrative example

Two small datasets are used in this section: Coil-3 (a subset of 3 classes from Coil-20), and Caltech-6 (a subset of 6 classes from Caltech-101). Experiments in this section can be seen as a *pedagogical example* of IC-EDS behavior operating in the presence of a new unknown class. Specifically, Coil-3 comprises images labeled as "*Duck*", "*Cat*", and "*Piggy*" represented by global features. As shown in Fig. 3, initial training sets have only labeled instances of classes "*Duck*" and "*Cat*" (14 instances for each class). In this sense, IC-EDS can initially recognize only ducks and cats, but it should detect and learn the omitted class "*Piggy*" while classifying images from the target sets. We used balanced target sets with 57 unlabeled images for each class ("*Duck*", "*Cat*", and "*Piggy*").

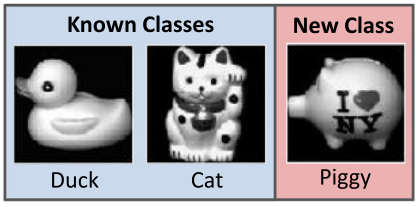

Fig. 3. Instances of classes in the illustrative global feature data. Only ducks and cats were used to build the classifier, thus IC-EDS should be able to discover the new unknown class "*Piggy*" present in the target sets.

Fig. 4 shows the proportions per class of images selected from target sets for each IC-EDS iteration (as defined in Section 4, IC-EDS was run five times; each time, the algorithm was tested on a specific target set). In the first iteration of the algorithm, 56% of the selected images belong to the left out class "*Piggy*". In the next iteration, this proportion reached 84%. To illustrate the role of EDS selection, an example of "*Piggy*" image in this iteration showed density 0.95 and label uncertainty of 0.96. Fig. 5 depicts a similarity matrix in which "*Duck*", "*Cat*", and "*Piggy*" were (to some extent) found by cluster ensembles, which helped in the classification refinement (by C^{3}E-SL), as well as in the detection of new classes (Algorithm 2). After gathering sufficient examples from the new class, in the third iteration the proportions of selected images per class became more similar, and "*Piggy*" error rates decreased to zero, indicating that IC-EDS got to recognize the "*Piggy*" class.

Similarly, when we performed five iterations of IC-EDS on Caltech-6 formed by deep features and with the class 6 ("*Watch*") missing from the training sets, its error rates strictly decreased to

[5] https://www.cs.waikato.ac.nz/ml/weka/.

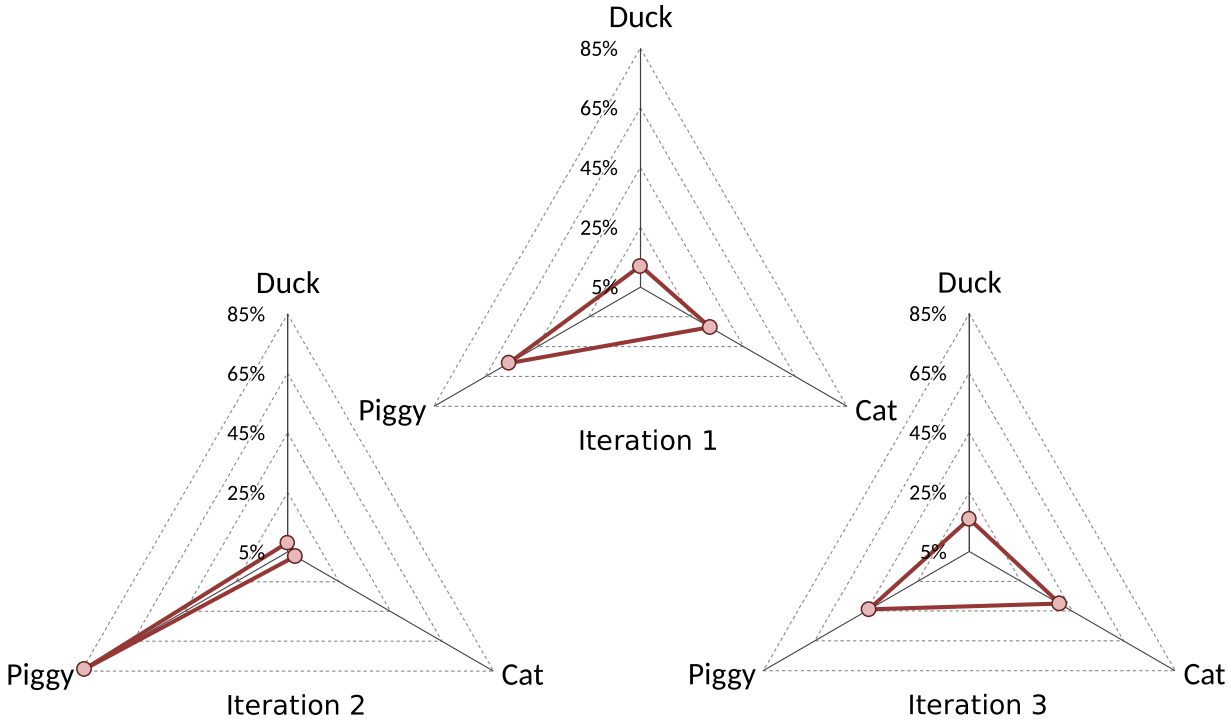

Fig. 4. Proportions of selected instances in each iteration.

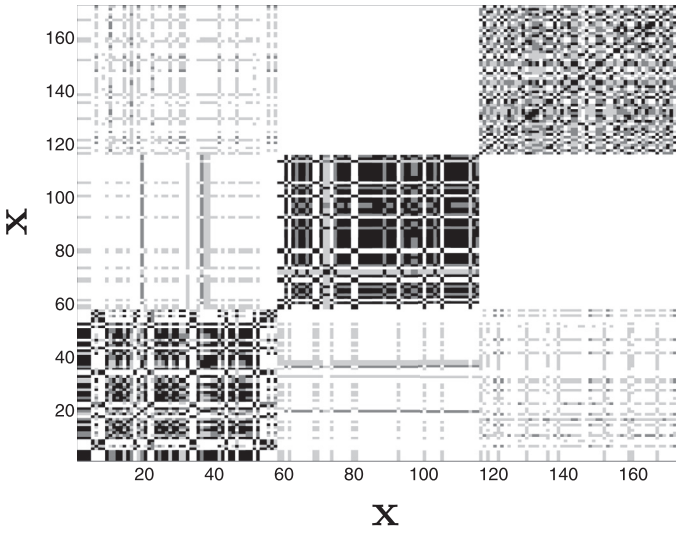

Fig. 5. Similarity matrix generated by cluster ensembles captured the existence of three clusters (dark regions) and helped in the refinement of SVM classification and detection of a new class ("*Piggy*"). The clusters from bottom-left to top-right have "*Duck*", "*Cat*", and "*Piggy*" instances, respectively.

0.03. Fig. 6 shows the proportions (per class) of selected images in each IC-EDS iteration. Thus, the behavior of the algorithm on Caltech-6 was similar to when Coil-3 was used. Note the tendency of selecting images from the omitted class 6 were similar for both datasets. This suggests that, initially, when a certain class is unknown, IC-EDS detects it because the unknown concept is dissimilar from those already known. After learning such a class, there was not much novelty to capture, so the proportions become more alike.

We designed IC-EDS to find instances from high density regions and that have high classification entropy. Although our desideratum is to have a joint effect of both density and entropy, it may be instructive to study them separately. To do so, we continue to explore the results on Caltech-6, but now focusing on some particular folds (training sets) sampled according to the experimental setup described in Section 4. We tested two strategies: (i) selecting instances as originally done by Algorithm 2 (EDS), in which entropy and density measures are combined and (ii) selecting instances with EDS operating only with the entropy measure (*i.e.*, density information was not considered at all). Fig. 7 shows the number of instances from class 6 that were selected in the first iteration of IC-EDS. Note that the combination of entropy and density yields to the detection of more instances from the omitted class — as compared to using entropy only. As expected, high label uncertainty, by itself, may not be enough to identify instances from new classes. In this sense, note that high density regions are less prone to favor the selection of outliers.

Let us explore a bit more the results depicted in Fig. 7 by using Fig. 8, which depicts radar plots where each axis represents the true class label of a selected instance in the first iteration of IC-EDS. Recall from Fig. 6 that, in the first iteration, 84% of the selected instances are from the (omitted) class 6 ("*Watch*"). As one can infer from the depicted number of instances of class 6, there is a one-to-one correspondence between the folds in Figs. 8 and 7. The polygon area for each radar plot represents the proportion of the h nearest neighbors sharing the same class label of the selected instance. These neighbors are the ones used to compute (from the similarity matrix) the density for each candidate instance, and thus presumably represent them. Fig. 8 shows that for folds 1, 3, and 4, all the neighbors share the same class label of the selected instances (class 6). Folds 2 and 5 show a different scenario, where not all of the neighbors belong to the same class. Note for fold 5 that two candidates are from class 2 (already known by the classification model). Indeed, these are representative of their local region, because all their neighbors belong to class 2 as well. So, our approach allows to find regions where instances are likely to share the same class label.

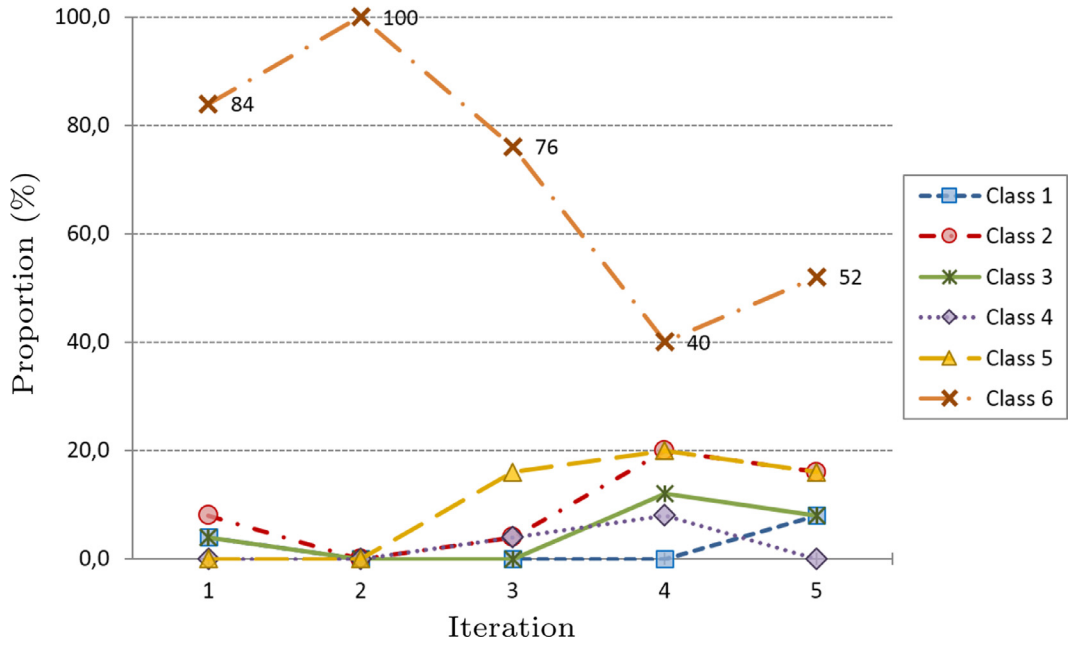

Fig. 6. Proportions of selected instances when class 6 ("Watch") of the Caltech-6 dataset was left out to be recognized.

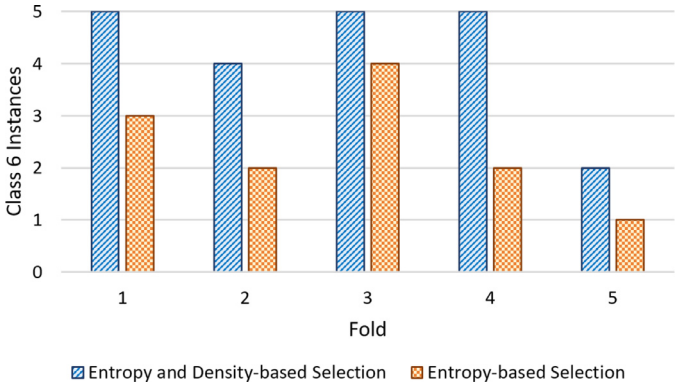

Fig. 7. Number of instances from class 6 ("*Watch*") selected in the first iteration of IC-EDS.

5.2. Results on Coil-20

We now compare IC-EDS with its baseline, IC-RS, which randomly selects instances to be labeled. Fig. 9 shows the proportions (per class) of selected images after five iterations of both algorithms when instances from class 13 ("*Piggy*") were left out of the initial training sets. Unlike the illustrative experiment in Section 5.1, we are now dealing with a more difficult problem because the algorithms must detect a new class within a set of twenty classes, of which nineteen were seen by the classifiers. Notice that we are also comparing the performance of the algorithms for the dataset formed by Global Features (GF) and deep features via VGGNet-16 (VGG16). As expected, IC-RS selected an uniform proportion of instances from different class (ranging from 1.6% to 8.8%), whereas IC-EDS selected instances from the unknown class more often. More precisely, instances from classes 2, 3, 13, and 16 were more frequently selected. In particular, 8.4% of the chosen instances belonged to the omitted class "*Piggy*" when IC-EDS and Global Features (IC-EDS-GF) were used, but this percentage was of 20.8% by using Deep Features (IC-EDS-VGG16), decreasing the class error rate by over 20% independently of the used features — see Table 1 further on.

Let us examine scenarios when omitting two other Coil-20 classes:

- Class 2 – "*Wooden Part 1*" (see Fig. 10): after 5 iterations, 14.4% of the instances selected by IC-EDS-GF belonged to class 2, whereas for IC-EDS-VGG16 this percentage was 30.4%. After updating the model with 25 new instances, the error rates were around 0.85 for IC-EDS-GF and IC-RS versions, while dropping to near 0.30 with the use of IC-EDS-VGG16. This shows that the VGG16 representation was better for both selection of instances and to add discriminative information to the model.
- Class 4 – "*Cat*" (see Fig. 11): IC-EDS selected around 38.4% of "Cat" instances (with both GF or VGG-16 features). Notice

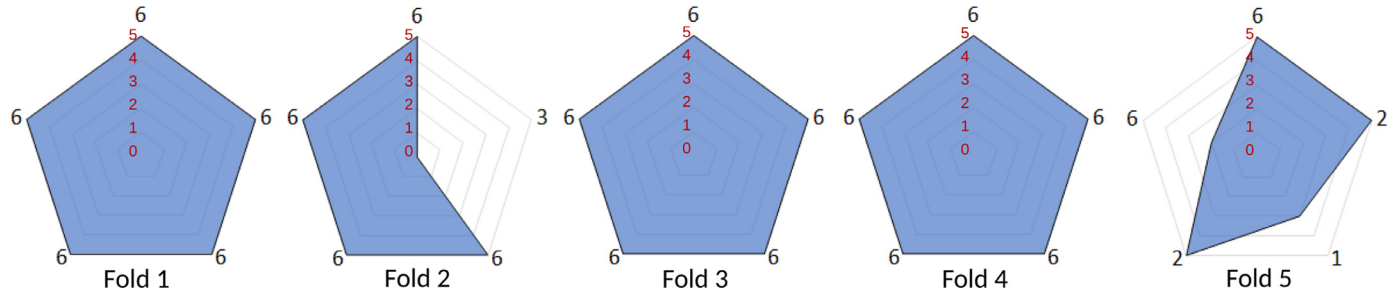

Fig. 8. Proportion of nearest neighbors sharing the same class label of a selected instance (axes show the class label of selected instances — five instances per fold).

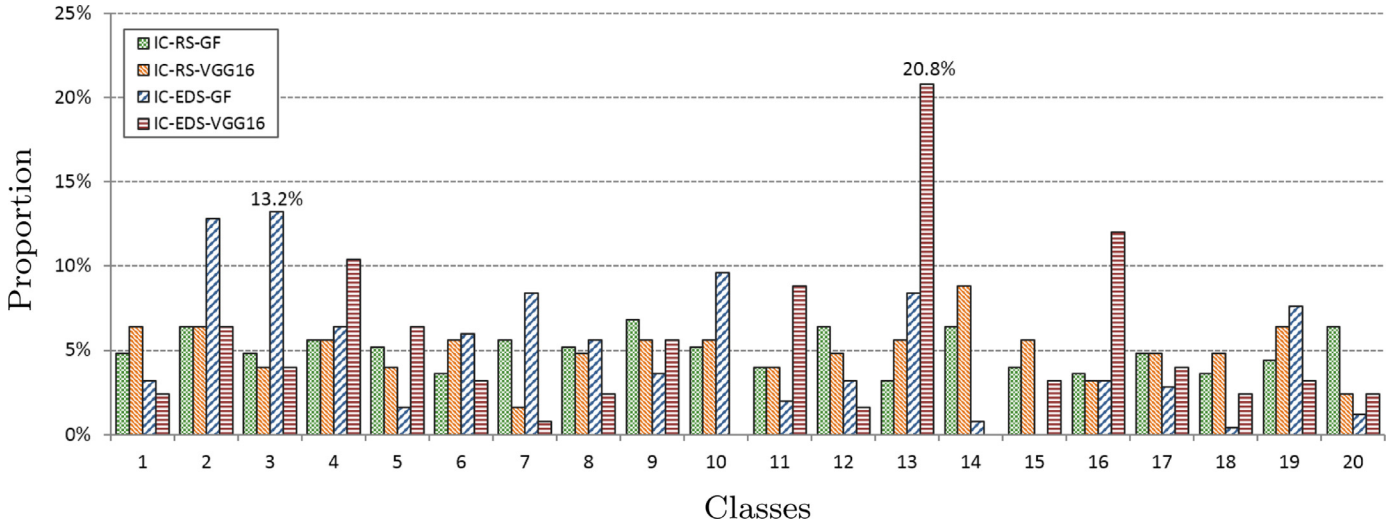

Fig. 9. Proportions of selected instances per class after 5 iterations when class 13 ("*Piggy*") was initially omitted in order to have it be discovered later.

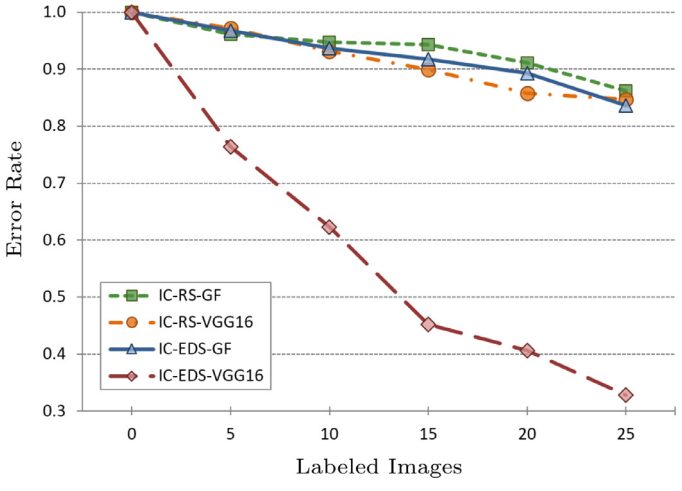

Fig. 10. Class 2 ("*Wooden Part 1*") error rates.

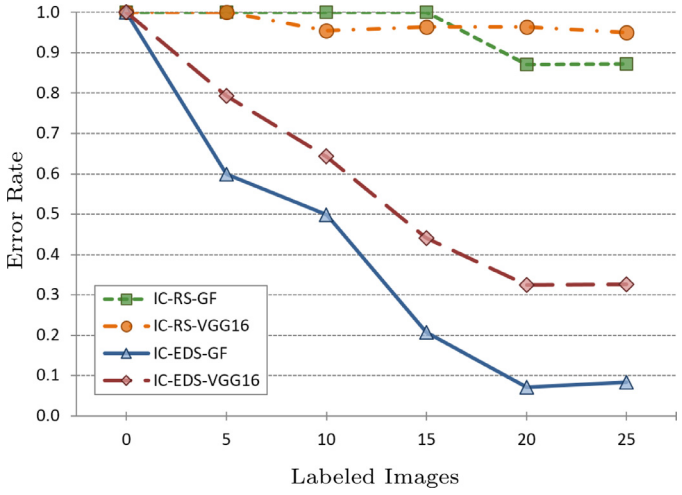

Fig. 11. Class 4 ("*Cat*") error rates.

Table 1
Selected images (%) and error rates for each left out class of Coil-20. The algorithms were run for 5 iterations (25 selected and manually labeled instances). Standard deviations are within parentheses, and the best results are highlighted in bold.

Class left out	Selection (%)				Error rate			
	IC-RS-GF	IC-RS-VGG16	IC-EDS-GF	IC-EDS-VGG16	IC-RS-GF	IC-RS-VGG16	IC-EDS-GF	IC-EDS-VGG16
1 ("*Duck*")	4.0 (4.90)	5.6 (6.07)	13.6 (11.17)	**33.6** (5.37)	0.95 (0.06)	0.85 (0.16)	**0.41** (0.29)	0.44 (0.21)
2 ("*Wooden Part 1*")	6.4 (6.07)	4.8 (4.38)	14.4 (8.76)	**30.4** (21.28)	0.86 (0.08)	0.85 (0.18)	0.84 (0.13)	**0.33** (0.21)
3 ("*Car 1*")	4.8 (3.35)	3.2 (1.79)	8.0 (5.66)	**29.6** (8.29)	0.77 (0.14)	0.90 (0.17)	0.97 (0.05)	**0.54** (0.38)
4 ("*Cat*")	2.4 (3.58)	2.4 (3.58)	**38.4** (21.09)	**38.4** (27.80)	0.87 (0.24)	0.95 (0.09)	**0.08** (0.12)	0.33 (0.28)
5 ("*Anacin*")	3.2 (4.38)	8.0 (4.00)	4.0 (2.83)	**33.6** (7.27)	0.93 (0.09)	0.82 (0.19)	0.88 (0.10)	**0.37** (0.27)
6 ("*Car 2*")	4.8 (3.35)	4.0 (1.00)	2.4 (3.58)	**28.8** (12.46)	0.62 (0.23)	0.96 (0.04)	0.97 (0.05)	**0.45** (0.37)
7 ("*Wooden Part 2*")	7.2 (3.35)	1.6 (2.19)	16.0 (16.73)	**28.8** (16.35)	0.85 (0.12)	0.95 (0.07)	**0.39** (0.22)	0.52 (0.13)
8 ("*Talc*")	4.0 (2.83)	1.6 (2.19)	24.8 (25.52)	**50.4** (21.84)	0.85 (0.25)	0.95 (0.06)	0.61 (0.23)	**0.03** (0.08)
9 ("*Tylenol*")	2.4 (2.19)	6.4 (7.27)	19.2 (8.67)	**26.4** (15.13)	0.87 (0.14)	0.89 (0.07)	**0.54** (0.34)	**0.54** (0.33)
10 ("*Vaseline*")	5.6 (2.19)	4.8 (6.57)	31.2 (18.63)	**38.4** (27.51)	0.87 (0.12)	0.85 (0.10)	**0.23** (0.17)	0.32 (0.43)
11 ("*Wooden P. 3*")	4.8 (7.16)	4.0 (4.00)	10.4 (8.29)	**35.2** (25.20)	0.84 (0.22)	0.82 (0.19)	0.62 (0.30)	**0.15** (0.15)
12 ("*Cup 1*")	5.6 (4.56)	2.4 (3.58)	22.4 (20.12)	**31.2** (24.07)	0.45 (0.40)	0.77 (0.37)	0.09 (0.13)	**0.00** (0.00)
13 ("*Piggy*")	3.2 (2.19)	5.6 (2.19)	8.4 (9.21)	**20.8** (4.38)	0.78 (0.21)	0.92 (0.06)	0.79 (0.18)	**0.78** (0.16)
14 ("*Pot 1*")	4.8 (3.35)	4.0 (4.00)	18.4 (15.39)	**36.8** (25.67)	0.87 (0.20)	0.90 (0.06)	0.27 (0.20)	**0.22** (0.34)
15 ("*Half Cov. Pot*")	3.2 (3.35)	7.2 (3.35)	11.2 (14.25)	**38.4** (15.39)	0.55 (0.47)	0.49 (0.28)	**0.00** (0.00)	**0.00** (0.00)
16 ("*Pot 2*")	4.8 (3.35)	4.8 (3.35)	11.2 (20.67)	**33.6** (20.51)	0.21 (0.44)	0.68 (0.20)	**0.01** (0.01)	0.03 (0.07)
17 ("*Uncovered Pot*")	2.4 (3.58)	4.0 (6.93)	12.0 (12.33)	**34.4** (13.45)	0.73 (0.43)	0.63 (0.41)	0.65 (0.32)	**0.02** (0.05)
18 ("*Cup 2*")	4.0 (2.83)	3.2 (7.16)	12.0 (13.27)	**37.6** (34.94)	0.25 (0.43)	0.80 (0.23)	0.30 (0.45)	**0.00** (0.00)
19 ("*Car 3*")	3.2 (1.79)	2.4 (2.19)	11.2 (13.08)	**35.2** (8.67)	0.91 (0.14)	0.95 (0.05)	0.74 (0.25)	**0.40** (0.41)
20 ("*Cream Cheese*")	8.0 (2.83)	4.0 (2.83)	1.6 (3.58)	**47.2** (34.34)	0.40 (0.55)	0.90 (0.10)	0.80 (0.45)	**0.17** (0.14)

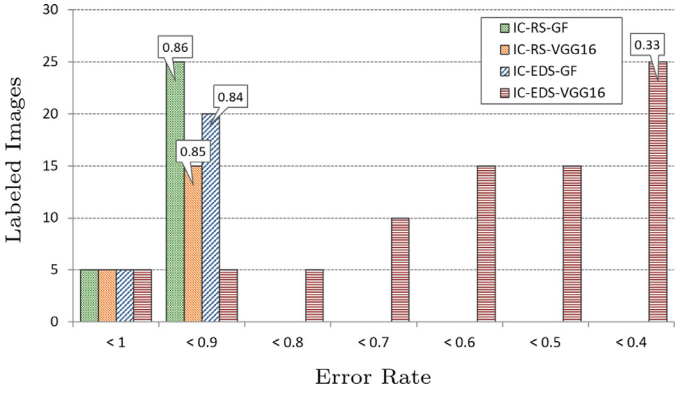

Fig. 12. Number of labeled instances to reach a certain error rate level for the omitted class "*Wooden Part 1*".

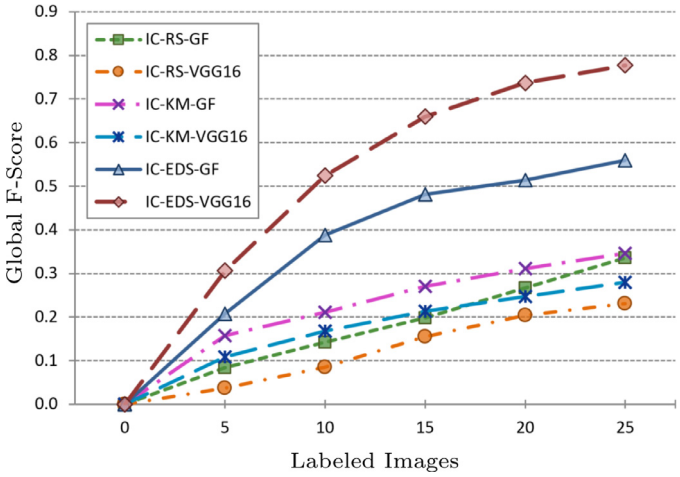

Fig. 14. Global F-Scores for Coil-20.

that IC-EDS-GF reduced the class error to less than 0.1, and to around 0.3 when using deep features (IC-EDS-VGG16), just by labeling 20 instances (i.e., after 4 iterations). In contrast, by using IC-RS counterparts, only 2.4% (which approximates 1/20) "*Cat*" images were selected and their error rates only decreased to around 0.9.

Those results suggest that, if an appropriate set of features is available, our iterative classifier is able to significantly improve results recognizing in few iterations new classes that may appear.

It is known that parsimonious and reliable labeling is a desirable property of image classifiers. Therefore, now we analyze the impact on the omitted class error rate when labeling and including new images in the classification model. For both Figs. 12 and 13, we anticipate that if a particular algorithm could not get a particular error rate, then its respective bar (for the labeled images) does not appear on the graph. For instance, in Fig. 12 only IC-EDS-VGG16 was able too get error rates less than 0.8. This analysis considers the two already studied classes:

- "*Wooden Part 1*": Fig. 12 shows that with random algorithms (IC-RS) as well as with Global Features (GF), the error rate decreased from 1 to around 0.85 by labeling 25 (IC-RS-GF), 15 (IC-RS-VGG16) and 20 (IC-EDS-GF) instances, while 0.33 error was achieved after 25 instances included with IC-EDS-VGG16. So, in this case, the latter can be considered more parsimonious than the others.
- "*Cat*": a similar behavior is shown in Fig. 13 but in this case IC-EDS was superior for both feature spaces: labeling 20 instances was sufficient to achieve error rate below 0.4 with IC-EDS, whereas, for the same number of instances the IC-RS algorithms best results were error around 0.9.

In general, IC-EDS algorithms required fewer labeled images to achieve the same (or lower) error rates obtained by IC-RS counterparts, achieving better accuracy results in general.

To summarize the results, Fig. 14 shows the global F-Score computed by averaging twenty F-Scores, each one corresponding to a left out class from dataset Coil-20. Notice that this figure depicts the global performance of the two baseline algorithms, IC-RS and IC-KM. As expected, IC-EDS algorithms indeed yielded better results than both IC-RS and IC-KM. Table 1 shows the percentage of selected images and error rates for each omitted class when the algorithms reached 5 iterations (i.e., when 25 instances had been manually labeled). The numerical results for the IC-KM were omitted in this table for the sake of compactness and because they are similar to those achieved by IC-RS algorithm. IC-EDS algorithms

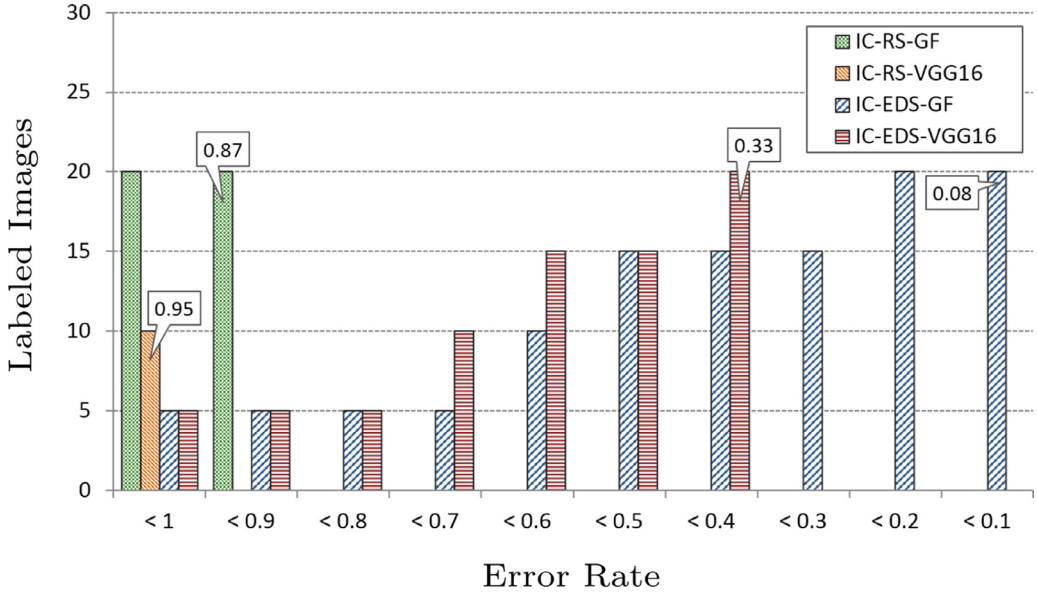

Fig. 13. Number of labeled instances to reach a certain error rate level for the omitted class "*Cat*".

Table 2
Selected images (%) and error rates for each left out class of Corel-10. The algorithms were run for 5 iterations (25 selected and manually labeled instances). Standard deviations are within parentheses, and the best results are highlighted in bold.

Class left out	Selection (%)				Error rate			
	IC-RS-GF	IC-RS-VGG16	IC-EDS-GF	IC-EDS-VGG16	IC-RS-GF	IC-RS-VGG16	IC-EDS-GF	IC-EDS-VGG16
1 (*"Africans"*)	11.2 (4.38)	12.8 (5.93)	36.0 (11.66)	**44.8** (22.70)	0.81 (0.18)	0.81 (0.00)	0.64 (0.11)	**0.55** (0.29)
2 (*"Beach"*)	9.6 (6.07)	8.0 (4.90)	17.6 (7.80)	**48.8** (24.56)	0.49 (0.31)	0.97 (0.04)	0.41 (0.23)	**0.37** (0.12)
3 (*"Buildings"*)	16.0 (6.32)	9.6 (4.56)	28.8 (14.53)	**47.2** (31.04)	0.73 (0.21)	0.67 (0.29)	0.51 (0.14)	**0.03** (0.06)
4 (*"Buses"*)	12.8 (6.57)	11.2 (5.22)	14.4 (12.52)	**33.6** (8.76)	0.88 (0.19)	0.71 (0.28)	0.97 (0.02)	**0.50** (0.28)
5 (*"Dinosaurs"*)	13.6 (4.56)	11.2 (5.22)	17.6 (6.69)	**33.6** (8.29)	0.21 (0.44)	0.82 (0.14)	**0.04** (0.05)	0.49 (0.33)
6 (*"Elephants"*)	8.0 (8.00)	9.6 (3.58)	8.0 (8.00)	**50.4** (21.09)	0.86 (0.11)	0.88 (0.13)	0.97 (0.03)	**0.12** (0.21)
7 (*"Flowers"*)	8.8 (6.57)	10.4 (2.19)	**28.0** (4.00)	24.0 (14.14)	0.50 (0.10)	0.92 (0.13)	0.75 (0.23)	**0.65** (0.18)
8 (*"Horses"*)	8.8 (3.35)	9.6 (4.56)	3.2 (1.79)	**50.4** (19.31)	0.92 (0.09)	0.87 (0.16)	0.99 (0.01)	**0.11** (0.08)
9 (*"Mountains"*)	10.4 (5.37)	12.0 (9.38)	4.8 (7.16)	**36.0** (13.56)	0.93 (0.07)	0.92 (0.05)	0.98 (0.03)	**0.55** (0.15)
10 (*"Food"*)	7.2 (1.79)	12.0 (4.90)	9.6 (4.56)	**50.4** (23.08)	0.91 (0.12)	0.75 (0.18)	0.86 (0.13)	**0.25** (0.13)

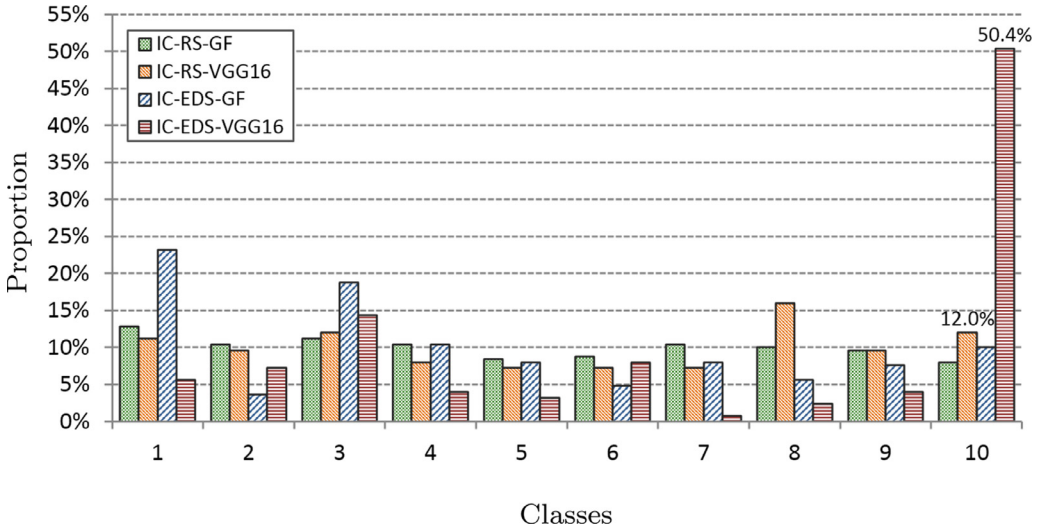

Fig. 15. Proportions of selected instances per class after 5 iterations when class 10 ("*Food*") was initially omitted in order to have it be discovered later.

worked very well because for the 35% of the omitted classes, they reduced their error rates to lower than 0.1 (three cases reached to 0.0). More specifically, IC-EDS-VGG16 selected more instances from those left out than the rest of the algorithms and provided lower error rates for 65% of these cases (IC-EDS-GF got better results for 25% of the cases). Therefore, using deep features, a good average F-Score was obtained by using our iterative classifier based on EDS algorithm. Note that IC-RS and IC-KM reached an F-Score around 0.3, regardless of the set of features employed.

5.3. Results on Corel-10

Corel-10 is a more complex dataset in terms of visual content, including background and clutter. As in the previous section, IC-EDS using deep features (VGG16) showed higher proportions of selected images for the classes left out than IC-EDS with Global Features (GF) — except when class "*Flowers*" was omitted. Table 2 summarizes the results of IC-EDS and IC-RS, and also includes the error rates for each class when they were taken out from the training sets. All algorithms labeled a total of 25 images (after 5 iterations). By observing the table, one interesting aspect is the difficulty to get low errors for the IC-EDS-GF, even selecting high proportions of the omitted class. This is probably due to the lack of quality of the features, not due to classification model. It is worth noticing the result for IC-EDS-GF when class "*Dinosaurs*" was left out, because this class is the only one with constant background, thus facilitating its separation from the remaining ones by using Global Features. In contrast, with IC-EDS-VGG16 very low error rates were achieved for the classes "*Buildings*", "*Elephants*", and "*Horses*". Moreover, IC-EDS-VGG16 obtained lower errors than those achieved by IC-EDS-GF for the 90% of the classes taken out.

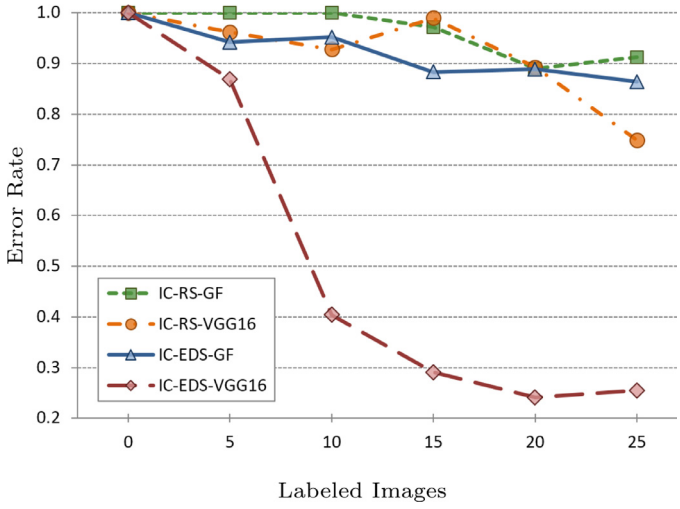

Fig. 16. Class 10 ("*Food*") error rates.

The behavior of our iterative classifier for class 10 is illustrated by Fig. 15, which depicts the proportions of selected images when "*Food*" images are not used in the initial training. According to this, and taking into account the error rates in Fig. 16, IC-EDS-GF cannot discriminate images of foods much better than random

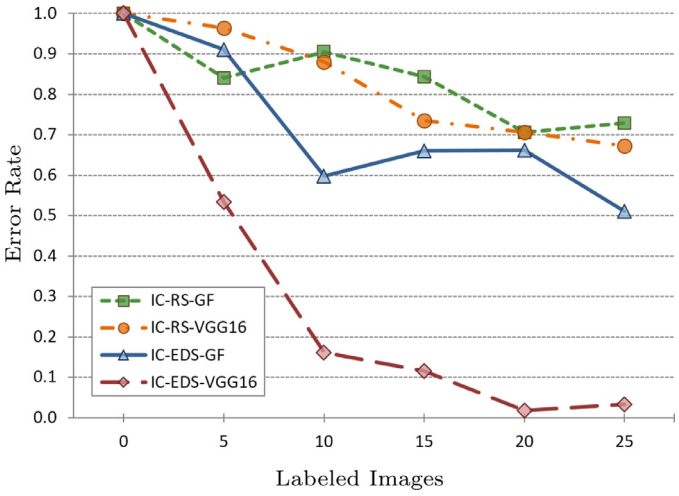

Fig. 17. Class 3 (“*Buildings*”) error rates.

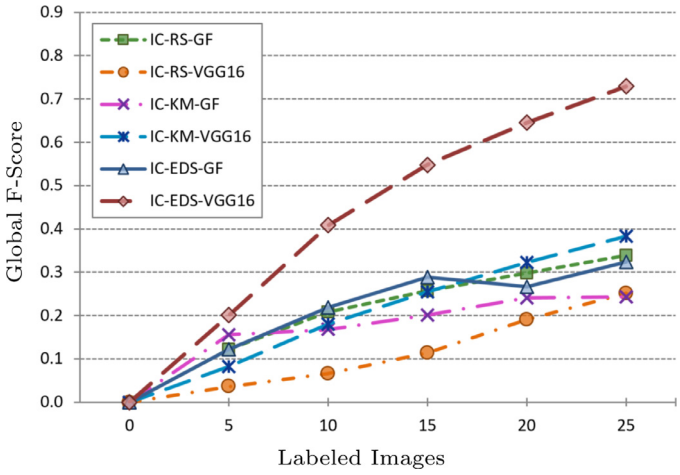

Fig. 19. Global F-Scores for Corel-10.

baseline algorithms. To shed light on this discussion, we note that: (i) the feature space impacts, the learning of a new, unknown class. As shown in Fig. 15, images from classes 1 (“*Africans*”), 3 (“*Buildings*”), and 4 (“*Buses*”) were more frequent than those from the omitted class 10 when IC-EDS-GF was run. Therefore, when the new class overlaps those already learned, the selection strategies may encounter difficulties to be better than random selection of instances. However, once the feature space is adequate, the IC-EDS identifies the new class and significantly decreases its error rate (see Fig. 16); (ii) the performances for either Random Selection (RS) or Entropy and Density-based Selection (EDS) tend to be more similar as the number of iterations increases. As shown in Section 5.1, IC-EDS is better at finding images that represent a new class in early iterations, in particular for a proper set of features. Later, classes of selected images become more uniform (as in IC-RS). According to Fig. 17, which depicts the error rate curves when class 3 (“*Buildings*”) was left out, IC-EDS-GF yielded a sharper error decrease on the first iterations, but after that the random selection methods tend to catch up with it.

Considering the labeling costs, Fig. 18 shows the number of labeled images necessary to achieve specific error rates. From this perspective, IC-EDS algorithms are more effective than their baselines, IC-RS, for the omitted class “*Buildings*”. Notice that to obtain an error rate lower than 0.6, 25 images had to be labeled by using IC-EDS-GF, whereas error lower than 0.1 was achieved by IC-EDS-VGG16 with only 20 labeled images.

Fig. 19 shows the global F-Scores yielded by averaging individual F-Scores computed for each left out class for the IC-EDS, IC-RS, and IC-KM algorithms. Although the results suggest that Corel-10 is a challenging dataset for the exploration of new classes, good results were obtained, particularly with few labeled images for IC-EDS-VGG16. Once again it suggests that, the better the feature space, more IC-EDS is able to improve results with few selected images.

5.4. Results on supermarket produce

In this dataset, IC-EDS algorithms also yielded the best results with similar overall behavior in terms of feature space and error rate per labeled images. Fig. 20 illustrates the proportions of selected images when class 9 (“*Nectarine*”) was omitted. While IC-RS algorithms got proportions between 2% and 10%, IC-EDS-GF and

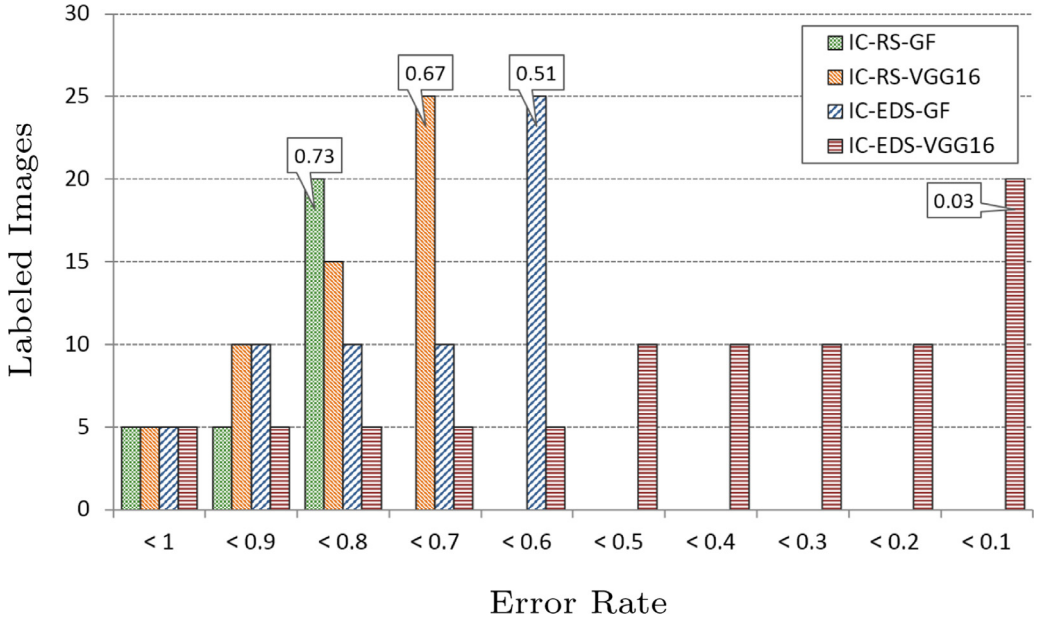

Fig. 18. Number of labeled instances to reach a certain error rate level for the omitted class “*Buildings*”.

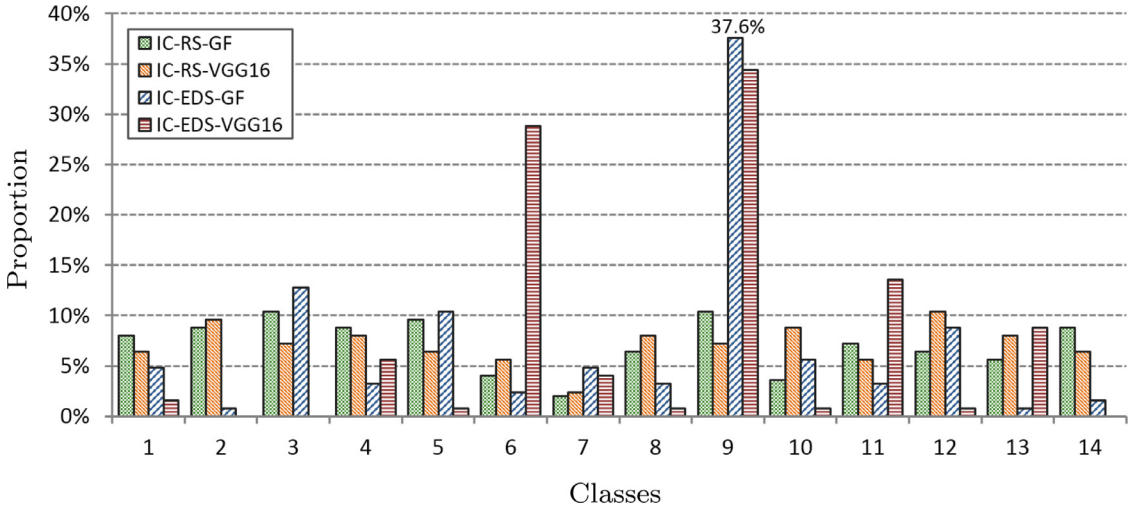

Fig. 20. Proportions of selected instances per class after 5 iterations when class 9 ("*Nectarine*") was initially omitted in order to have it be discovered later.

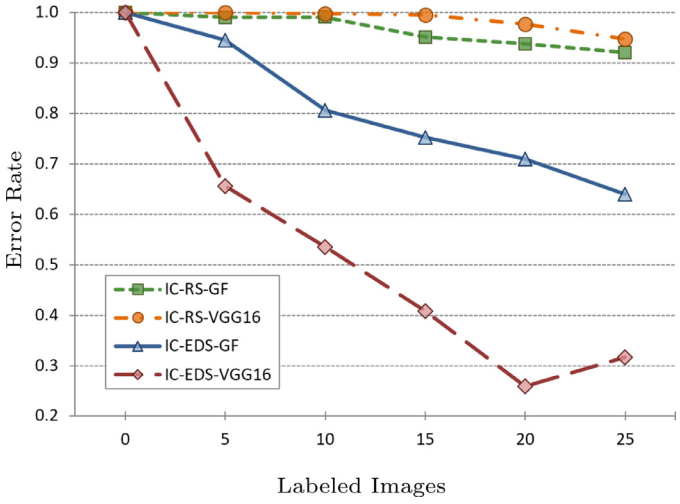

Fig. 21. Class 9 ("*Nectarine*") error rates.

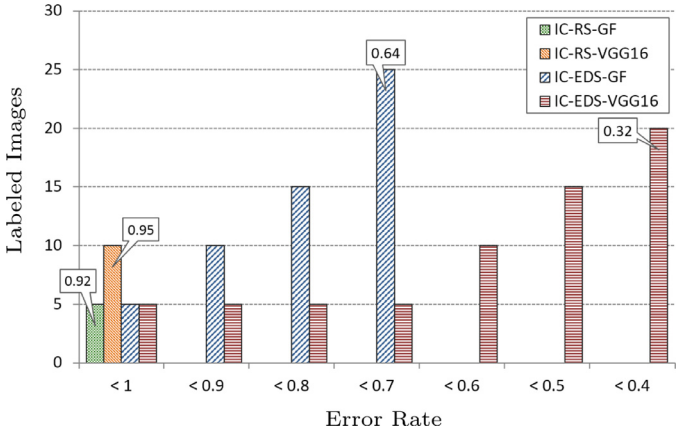

Fig. 22. Number of labeled instances to reach a certain error rate level for the omitted class "*Nectarine*".

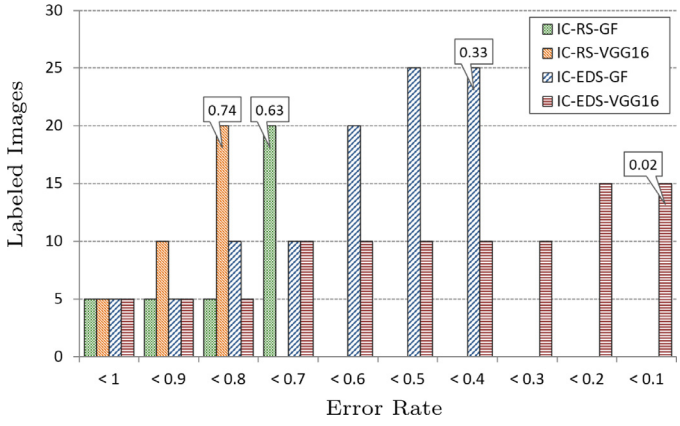

Fig. 23. Number of labeled instances to reach a certain error rate level for the omitted class "*Plum*".

IC-EDS-VGG16 found nectarines in 37.6% and 34.4% of the selected images, respectively. As a result, IC-EDS provided faster error reduction, as shown in Fig. 21.

To clarify how practical our iterative classifier is in terms of the labeling task, Figs. 22 and 23 illustrate the labeling efforts necessary for achieving a certain error rate when the "*Nectarine*" and "*Plum*" classes were omitted. For the "*Nectarine*" class, by labeling 25 images (i.e., after 5 iterations of the algorithm), IC-EDS-GF obtained an error rate below 0.7, but IC-EDS-VGG16 required only 20 labeled instances to produce an error two times lower. As for the "*Plum*" class, reaching an error rate lower than 0.7 required only 10 labeled instances for IC-EDS algorithms, whereas 20 instances selected with the random (IC-RS) to reach similar error.

The performance of the IC-EDS and its two baseline algorithms is summarized in Fig. 24, which shows the global F-Score computed from fourteen individual F-Scores, each one for an omitted class. For this dataset, our approach got the best results, except when IC-EDS-GF is compared with IC-KM-GF from the fourth iteration on. Nevertheless, we shall highlight the features learned from deep learning, which notably improved results for IC-EDS-VGG16.

The overall results for IC-EDS and IC-RS are summarized in Table 3, which shows the proportions of selected images and error rates for each algorithm when a particular class was left out from the initial training sets. Algorithms based on IC-EDS discovered the omitted classes more accurately than their counterparts, which randomly select instances to be labeled (IC-RS). In 93% of the cases, the highest proportions of left out class selected were obtained by IC-EDS-VGG16 — only when nectarines were taken out IC-EDS-GF selected more. Remarkable error rates were also reached by IC-EDS-VGG16 (for classes 6, 8, 11, and 13), and only for the class 14 ("*Watermelon*") IC-EDS-GF was better than the former.

Table 3
Selected images (%) and error rates for each left out class of Supermarket Produce. The algorithms were run for 5 iterations (25 selected and manually labeled instances). Standard deviations are within parentheses, and the best results are highlighted in bold.

Class left out	Selection (%)				Error rate			
	IC-RS-GF	IC-RS-VGG16	IC-EDS-GF	IC-EDS-VGG16	IC-RS-GF	IC-RS-VGG16	IC-EDS-GF	IC-EDS-VGG16
1 (*"Agata Potato"*)	10.4 (4.56)	10.4 (6.07)	17.6 (7.80)	**33.6** (20.12)	0.72 (0.14)	0.74 (0.25)	0.76 (0.15)	**0.42** (0.31)
2 (*"Asterix Potato"*)	8.0 (6.32)	3.2 (3.35)	20.0 (14.14)	**36.0** (19.60)	0.64 (0.39)	0.98 (0.01)	0.63 (0.02)	**0.17** (0.16)
3 (*"Cashew"*)	8.8(3.35)	6.4 (4.56)	6.4 (4.56)	**32.8** (9.12)	0.36 (0.16)	0.93 (0.10)	0.70 (0.33)	**0.33** (0.23)
4 (*"Dia. Peach"*)	4.0 (5.66)	8.8 (5.22)	13.6 (9.21)	**35.2** (26.44)	0.86 (0.24)	0.79 (0.13)	0.72 (0.30)	**0.13** (0.19)
5 (*"Fuji Apple"*)	5.6 (6.07)	7.2 (3.35)	21.6 (9.21)	**24.0** (6.32)	0.82 (0.13)	0.93 (0.04)	0.58 (0.19)	**0.53** (0.35)
6 (*"Gran. Apple"*)	4.0 (4.90)	8.0 (7.48)	20.0 (12.00)	**53.6** (35.28)	0.79 (0.29)	0.72 (0.30)	0.65 (0.30)	**0.00** (0.00)
7 (*"Hon. Melon"*)	1.6 (2.19)	10.4 (5.37)	12.0 (6.32)	**38.4** (20.12)	0.87 (0.15)	0.79 (0.13)	0.77 (0.26)	**0.29** (0.34)
8 (*"Kiwi"*)	6.4 (8.29)	6.4 (5.37)	9.6(4.56)	**39.2** (26.44)	0.84 (0.19)	0.93 (0.09)	0.90 (0.12)	**0.08** (0.06)
9 (*"Nectarine"*)	10.4 (6.07)	7.2 (4.38)	**37.6** (13.45)	34.4 (30.01)	0.92 (0.10)	0.95 (0.06)	0.64 (0.17)	**0.32** (0.25)
10 (*"Orange"*)	4.0 (4.00)	8.0 (6.93)	12.0 (4.00)	**38.4**(23.43)	0.75 (0.23)	0.85 (0.18)	0.68 (0.43)	**0.29** (0.26)
11 (*"Plum"*)	9.6 (6.07)	8.0 (4.00)	28.0 (7.48)	**43.2** (28.34)	0.63 (0.23)	0.74 (0.30)	0.33 (0.15)	**0.02** (0.03)
12 (*"Williams Pear"*)	4.8 (3.35)	3.2 (3.35)	28.0 (11.66)	**32.0** (25.61)	0.94 (0.09)	1.00 (0.00)	0.55 (0.21)	**0.46** (0.19)
13 (*"Taiti Lime"*)	3.2 (5.22)	2.4 (2.19)	5.6 (6.07)	**43.2** (18.20)	0.66 (0.46)	0.99 (0.01)	0.78 (0.34)	**0.08** (0.19)
14 (*"Watermelon"*)	6.4 (3.58)	13.6 (8.29)	18.4 (15.65)	**41.6** (22.20)	0.40 (0.47)	0.66 (0.36)	**0.05** (0.02)	0.18 (0.21)

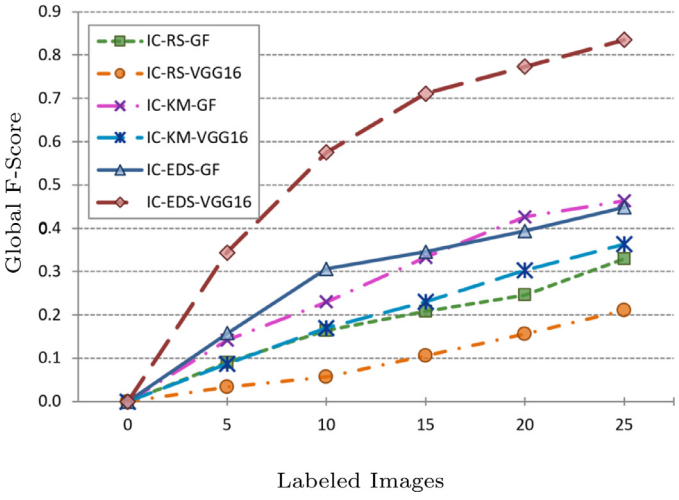

Fig. 24. Global F-Scores for Supermarket Produce.

5.5. Discussion

The results observed in the small datasets Coil-3 and Caltech-6 were consistent with respect to the full datasets: Coil-20, Corel-10, and Supermarket Produce. Our proposed algorithm (IC-EDS) effectively used unlabeled data to improve classification results, and also discovered new classes. In particular, when features describing images are good enough to discriminate the unknown class from the ones already existing in the current model, then IC-EDS algorithm is consistently better than its baseline algorithms (IC-RS and IC-KM). Experimental evidence suggests that global features often do not offer sufficient information to allow detecting new image classes, apart from a few specific cases. Remarkably better results are achieved when using deep features that were generated by using a forward pass on pre-trained Deep Neural Networks. This is important because extracting such features does not require to label images.

Let us highlight some particular results of F-Scores computed when deliberately removing each class from the training set, and selecting up to 25 images to be labeled from a set of unlabeled instances:

- In Coil-20, IC-EDS achieved an average F-Score of 0.78 with deep features (VGG16) and 0.56 with Global Features (GF), whereas IC-RS and IC-KM did not get any better than 0.35;
- In Corel-10, GF were not sufficiently informative, so IC-EDS-VGG16 got an F-Score equal to 0.73, against less than 0.38 for the other algorithms;
- In Supermarket Produce, GF are more informative when compared to the previous datasets. In fact, in the original study of this dataset, the authors successfully used GF by training with the same number of examples from all classes [61]. Still, in our experiments with missing classes, IC-EDS-VGG16 obtained an F-Score of 0.83, which is superior to IC-EDS-GF (0.45), IC-KM (0.46), and IC-RS (0.33).

5.6. Time complexity analysis

The asymptotic time complexity of Algorithm 1 (C^3E-SL) is $O(c \cdot n^2)$, where c is the number of classes and n is the number of instances in the target set [30,31]. Our strategy for the selection of instances (Algorithm 2 — EDS) is quadratic w.r.t. P_2 that is, the number of selected candidates. As the proposed algorithm for learning new classes (Algorithm 3 — IC-EDS) performs C^3E-SL and EDS iteratively, its computational complexity per iteration is $O(n^2 + P_2^2 \cdot n)$. However, the bottleneck of IC-EDS is the rebuilding of the classifier ensemble, whose cost relies on the chosen components. The removal of entries (of labeled instances) from the similarity matrix also demands quadratic time complexity. Notice that building of the similarity matrix, as input for the IC-EDS algorithm, is $O(n^2)$ as well.

6. Conclusions

We introduced a flexible image classifier that deals with the appearance of new classes over time. It combines supervised and unsupervised algorithms, making use of the supplementary information provided by clustering algorithms to help in the detection of new classes. In particular, our Iterative Classifier (IC) employs an optimization algorithm to combine classification and clustering algorithms. In each iteration, an Entropy and Density-based Selection (EDS) explores and selects unlabeled instances (from a target set) that have highly uncertain labels and are located in dense regions. These instances are likely to represent new concepts/classes and, as such, are then labeled and used to update the classification model for the next iteration.

Experimental results show that the IC-EDS can successfully discover new classes over time using unlabeled instances, even on target sets with many different classes. In addition, the algorithm allows parsimonious selection of instances, which decreases costs of labeling.

Our contribution is a significant step towards classifiers that can become aware of new concepts/classes that may appear over time while using both labeled and unlabeled examples. We also showed that the iterative selection of instances from new classes performed better when using IC-EDS on features obtained from a forward pass on a pre-trained Convolutional Neural Network.

The main limitations of the proposed approach are: (i) the extra time needed for optimization of required parameters; (ii) the sensitivity to used feature space, particularly when it enables overlapping of known with unknown classes, which hampers the learning of the algorithm; (iii) and the rebuilding of the classifier ensemble in each algorithm iteration, which increases the computational cost (however, notice that incremental classifiers can be used for time savings).

Aspects such as the impact of the type and number of classifiers used and the methods of inducing "good" data partitions will be explored in future studies. Therefore, a more comprehensive investigation of a variety of settings may further clarify the capabilities (and potential limitations) of the proposed classifier. More specifically, the number of neighbors in Eq. (4) is a parameter that deserves further studies. From a theoretical perspective, studies of the properties of Eqs. (3) and (4) for detecting new classes are interesting and could help in the design of improvements.

Conflict of Interest

None.

Acknowledgements

We would like to acknowledge the São Paulo Research Foundation (FAPESP), grant #2017/00357-7, for providing financial support.

References

[1] Cisco, Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2014-2019 White Paper (2014).
[2] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.
[3] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proceedings of the Advances in Neural Information Processing Systems 25, Curran Associates, Inc., 2012, pp. 1097–1105.
[4] J. Zhang, M. Marszałek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: a comprehensive study, Int. J. Comput. Vis. 73 (2) (2007) 213–238.
[5] O. Chapelle, P. Haffner, V.N. Vapnik, Support vector machines for histogram-based image classification, IEEE Trans. Neural Netw. 10 (5) (1999) 1055–1064.
[6] U. von Luxburg, B. Schölkopf, Statistical learning theory: Models, concepts, and results, in: D.M. Gabbay, S. Hartmann, J. Woods (Eds.), Inductive Logic, Handbook of the History of Logic, 10, North-Holland, 2011, pp. 651–706.
[7] M. Guillaumin, J. Verbeek, C. Schmid, Multimodal semi-supervised learning for image classification, in: Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2010, pp. 902–909.
[8] C. Lampert, H. Nickisch, S. Harmeling, Learning to detect unseen object classes by between-class attribute transfer, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009, 2009, pp. 951–958.
[9] J.G. Wang, Z.D. Cao, B.H. Yang, S.W. Ma, M.R. Fei, H. Wang, Y. Yao, T. Chen, X.F. Wang, A mothed of improving identification accuracy via deep learning algorithm under condition of deficient labeled data, in: Proceedings of the 2017 36th Chinese Control Conference (CCC), 2017, pp. 2281–2286.
[10] G. Jun, J. Ghosh, Semisupervised learning of hyperspectral data with unknown land-cover classes, IEEE Trans. Geosci. Remote Sens. 51 (1) (2013) 273–282.
[11] W. Scheirer, L. Jain, T. Boult, Probability models for open set recognition, IEEE Trans. Pattern Anal. Mach. Intell. 36 (11) (2014) 2317–2324.
[12] S. Saxena, S. Pandey, P. Khanna, A semi-supervised domain adaptation assembling approach for image classification, Pattern Anal. Appl. (2017).
[13] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, ImageNet large scale visual recognition challenge, Int. J. Comput. Vis. (IJCV) 115 (3) (2015) 211–252.
[14] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[15] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, ArXiv e-prints (2014).
[16] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, Deepface: Closing the gap to human-level performance in face verification, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 1701–1708.
[17] S. Zhou, Q. Chen, X. Wang, Active deep learning method for semi-supervised sentiment classification, Neurocomputing 120 (2013) 536–546.
[18] D. Tuia, M. Volpi, L. Copa, M. Kanevski, J. Munoz-Mari, A survey of active learning algorithms for supervised remote sensing image classification, IEEE J. Sel. Top. Signal Process. 5 (3) (2011) 606–617.
[19] G.M. Foody, A. Mathur, Toward intelligent training of supervised image classifications: directing training data acquisition for SVM classification, Remote Sens. Environ. 93 (1-2) (2004) 107–117.
[20] W. Scheirer, A. de Rezende Rocha, A. Sapkota, T. Boult, Toward open set recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (7) (2013) 1757–1772.
[21] M. Riva, M. Ponti, T. de Campos, One-class to multi-class model update using the class-incremental optimum-path forest classifier, in: ECAI, 2016, pp. 216–224.
[22] A. Torralba, A. Efros, Unbiased look at dataset bias, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011, 2011, pp. 1521–1528.
[23] D.J. Hand, Classifier technology and the illusion of progress, Stat. Sci. 21 (1) (2006) 1–14.
[24] R. Duin, E. Pekalska, Open issues in pattern recognition, in: Computer Recognition Systems, in: Advances in Soft Computing, 30, Springer Berlin Heidelberg, 2005, pp. 27–42.
[25] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognit. Lett. 31 (8) (2010) 651–666.
[26] V. Vapnik, The Nature of Statistical Learning Theory, Springer Science & Business Media, 2013.
[27] Y. Lin, F. Lv, S. Zhu, M. Yang, T. Cour, K. Yu, L. Cao, T. Huang, Large-scale image classification: fast feature extraction and SVM training, in: CVPR 2011, 2011, pp. 1689–1696.
[28] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, S. Bengio, Why does unsupervised pre-training help deep learning? J. Mach. Learn. Res. 11 (2010) 625–660.
[29] N.F.F. da Silva, L.F.S. Coletta, E.R. Hruschka, E.R.H. Jr., Using unsupervised information to improve semi-supervised tweet sentiment classification, Inf. Sci. 355-356 (2016) 348–365.
[30] A. Acharya, E.R. Hruschka, J. Ghosh, S. Acharyya, An optimization framework for combining ensembles of classifiers and clusterers with applications to nontransductive semisupervised learning and transfer learning, ACM Trans. Knowl. Discov. Data 9 (1) (2014) 1:1–1:35.
[31] A. Acharya, E.R. Hruschka, J. Ghosh, S. Acharyya, C3E: a framework for combining ensembles of classifiers and clusterers, in: Multiple Classifier Systems, 6713, 2011, pp. 269–278.
[32] B. Settles, Active learning, Synth. Lect. Artif. Intell. Mach. Learn. 6 (1) (2012) 1–114.
[33] C. Lampert, H. Nickisch, S. Harmeling, Attribute-based classification for zero-shot visual object categorization, IEEE Trans. Pattern Anal. Mach. Intell. 36 (3) (2014) 453–465.
[34] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[35] G. Schohn, D. Cohn, Less is more: active learning with support vector machines, in: ICML, Citeseer, 2000, pp. 839–846.
[36] Y. Zhou, S. Goldman, Democratic co-learning, in: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, 2004. ICTAI 2004, IEEE, 2004, pp. 594–602.
[37] Q. Zhang, S. Sun, Multiple-view multiple-learner active learning, Pattern Recognit. 43 (9) (2010) 3113–3119.
[38] J. Zhou, S. Sun, Gaussian process versus margin sampling active learning, Neurocomputing 167 (2015) 122–131.
[39] S.-J. Huang, R. Jin, Z.-H. Zhou, Active learning by querying informative and representative examples, in: Proceedings of the Advances in neural information processing systems, 2010, pp. 892–900.
[40] Y. Yang, Z. Ma, F. Nie, X. Chang, A.G. Hauptmann, Multi-class active learning by uncertainty sampling with diversity maximization, Int. J. Comput. Vis. 113 (2) (2015) 113–127.
[41] M. Ponti, Relevance image sampling from collection using importance selection on randomized optimum-path trees, in: Proceedings of the 2017 Brazilian Conference on Intelligent Systems (BRACIS), IEEE, 2017, pp. 198–203.
[42] Y.J. Lee, K. Grauman, Object-graphs for context-aware visual category discovery, IEEE Trans. Pattern Anal. Mach. Intell. 34 (2) (2012) 346–358.
[43] E. Bart, S. Ullman, Cross-generalization: learning novel classes from a single example by feature replacement, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005, 1, 2005, pp. 672–679vol. 1.
[44] R. Socher, M. Ganjoo, C.D. Manning, A. Ng, Zero-shot learning through cross-modal transfer, in: C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 26, Curran Associates, Inc., 2013, pp. 935–943.
[45] M. Rohrbach, M. Stark, B. Schiele, Evaluating knowledge transfer and zero-shot learning in a large-scale setting, in: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),, 2011, pp. 1641–1648.
[46] M. Markou, S. Singh, Novelty detection: a review–part 1: statistical approaches, Signal Process. 83 (12) (2003) 2481–2497.

[47] M. Markou, S. Singh, Novelty detection: a review–part 2: neural network based approaches, Signal Process. 83 (12) (2003) 2499–2521.
[48] M. Masud, Q. Chen, L. Khan, C. Aggarwal, J. Gao, J. Han, A. Srivastava, N. Oza, Classification and adaptive novel class detection of feature-evolving data streams, IEEE Trans. Knowl. Data Eng. 25 (7) (2013) 1484–1497.
[49] L.I. Kuncheva, Classifier ensembles for detecting concept change in streaming data: Overview and perspectives, in: Proceedings of the 2nd Workshop SUEMA, 2008, 2008, pp. 5–10.
[50] C. Luo, J. Wang, G. Feng, S. Xu, S. Wang, Do deep convolutional neural networks really need to be deep when applied for remote scene classification? J. Appl. Remote Sens. 11 (4) (2017) 042613.
[51] M.A. Ponti, L.S. Ribeiro, T.S. Nazare, T. Bui, J. Collomosse, Everything you wanted to know about deep learning for computer vision but were afraid to ask, in: SIBGRAPI–Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T 2017), 2017, pp. 17–41.
[52] A. Strehl, J. Ghosh, Cluster ensembles - a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (2002) 583–617.
[53] L.F.S. Coletta, E.R. Hruschka, A. Acharya, J. Ghosh, A differential evolution algorithm to optimise the combination of classifier and cluster ensembles, Int. J. Bio-Inspired Comput. 7 (2) (2015) 111–124.
[54] F.J. Van der Wel, L.C. Van der Gaag, B.G. Gorte, Visual exploration of uncertainty in remote-sensing classification, Comput. Geosci. 24 (4) (1998) 335–343.
[55] I.H. Witten, E. Frank, Data mining: practical machine learning tools and techniques, 2nd, 2005.
[56] C.-W. Hsu, C.-C. Chang, C.-J. Lin, et al., A practical guide to support vector classification, Taipei, Technical Report (2003) 1–16.
[57] R.J. Campello, E.R. Hruschka, V.S. Alves, On the efficiency of evolutionary fuzzy clustering, J. Heuristics 15 (2009) 43–75.
[58] M. Ponti, T.S. Nazaré, G.S. Thumé, Image quantization as a dimensionality reduction procedure in color and texture feature extraction, Neurocomputing 173 (2016) 385–396.
[59] S.A. Nene, S.K. Nayar, H. Murase, Columbia Object Image Library (COIL-20), Technical Report, 1996.
[60] J. Wang, J. Li, G. Wiederhold, Simplicity: semantics-sensitive integrated matching for picture libraries, IEEE Trans. Pattern Anal. Mach. Intell. 23 (9) (2001) 947–963.
[61] A. Rocha, D.C. Hauagge, J. Wainer, S. Goldenstein, Automatic fruit and vegetable classification from images, Comput. Electron. Agric. 70 (1) (2010) 96–104.
[62] O. Penatti, E. Valle, R. Torres, Comparative study of global color and texture descriptors for web image retrieval, J. Vis. Commun. Image Represent. 23 (2) (2012) 359–380.
[63] M. Ponti, C. Picon, Color description of low resolution images using fast bitwise quantization and border-interior classification, in: Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, Brisbane, Australia, 2015.
[64] R.O. Stehling, M.A. Nascimento, A.X. Falcao, A compact and efficient image retrieval approach based on border/interior pixel classification, in: Proceedings of the CIKM'02 - 11th ACM Int. Conf. Information Knowledge Management, 2002, pp. 102–109.
[65] R. Haralick, K. Shanmugan, I. Dinstein, Textural features for image classification, IEEE Trans. Syst. Man Cybern. SMC-3 (6) (1973) 610–621.

Luiz F. S. Coletta has been with the School of Sciences and Engineering (FCE) of the São Paulo State University (UNESP/Tupã) since 2016, where he is currently an Assistant Professor. He obtained his Ph.D. in Computer Science from the Institute of Mathematics and Computer Science (ICMC) at University of São Paulo (USP) at São Carlos, Brazil. Previously, he received his M.Sc. degree in Computer Science in 2011, and his B.Sc. degree in Information Systems in 2009, both from the same institution (ICMC-USP). His research interests include Data Mining, Machine Learning, Clustering, Active and Semi-supervised Learning, as well as Bioinspired Computing.

Moacir Antonelli Ponti is, since 2010, with the Institute of Mathematics and Computer Sciences, University of São Paulo, São Carlos, Brazil, where he is currently Associate Professor. In 2016 he was visiting researcher at the Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey. He obtained his Ph.D. from the Federal University of São Carlos. His research interests include Machine Learning, Computer Vision, Signal and Image Processing. During his career, he published over 40 scientific papers in conferences and journals.

Eduardo Hruschka is currently head of data science at Itaú Bank. From 2013 to 2016, he was chief data scientist of the startup "Big Data". Prior to that, he had an academic career. He received his Ph.D. degree in Computational Systems from COPPE/Federal University of Rio de Janeiro in 2001. Then he worked for several private universities until 2007, when he joined the University of São Paulo (USP), where he is currently a part-time associate professor of the Department of Computer Engineering and Digital Systems. From 2010 to 2012, he worked as a postdoctoral researcher at the University of Texas at Austin. In his academic career, he has published over than 100 scientific papers on data science, machine learning, and artificial intelligence.

Ayan Acharya received his Ph.D. from the Department of Electrical and Computer Engineering at University of Texas at Austin, where he was associated with Intelligent Data Exploration and Analysis Laboratory (IDEAL) and the Machine Learning Research Group. Previously, he has worked at companies like eBay Research Lab, Qualcomm Inc, and Yahoo! Research. He has served as a reviewer for many leading journals in machine learning and artificial intelligence, such as Machine Learning Journal, Elsevier Information Sciences, IEEE TKDE, and ACM TKDD. He has also served on the program committee of many of the top-tier machine learning conferences that include NIPS, AAAI, ICDM, and ICML.

Joydeep Ghosh is currently the Schlumberger Centennial Chair Professor of Electrical and Computer Engineering at the University of Texas, Austin. He also serves as the Chief Scientist of Cognitivescale, which was selected in 2018 by the World Economic Forum as one of the 100 emerging companies (across all technologies worldwide), that are likely to have the most positive impact on business and society in the near future. He joined the UT-Austin Faculty in 1988 (BTech '83) and the University of Southern California (PhD '88). He is the Founder-Director of Intelligent Data Exploration and Analysis Lab (IDEAL) and a fellow of the IEEE. He has taught graduate courses on data mining and web analytics every year to both UT students and to industry, for over a decade. He was voted as Best Professor in the Software Engineering Executive Education Programme at UT.