24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Improving Supervised Superpixel-Based Codebook Representations by Local Convolutional Features

César Castelo-Fernández[1] and **Alexandre X. Falcão**[2]

Abstract.

Deep learning techniques are the current state-of-the-art in image classification techniques, but they have the drawback of requiring a large number of labelled training images. In this context, Visual Dictionaries (Codebooks or Bag of Visual Words - BoVW) are a very interesting alternative to Deep Learning models whenever we have a low number of labelled training images. However, the methods usually extract interest points from each image independently, which then leads to unrelated interest points among the classes and they build one visual dictionary in an unsupervised fashion. In this work, we present: 1) the use of class-specific superpixel segmentation to define interest points that are common for each class, 2) the use of Convolutional Neural Networks (CNNs) with random weights to cope with the absence of labeled data and extract more representative local features, and 3) the construction of specialized visual dictionaries for each class. We conduct experiments to show that our method can outperform a CNN trained from a small set of labelled images and can be equivalent to a CNN with pre-trained features. Also, we show that our method is better than other traditional BoVW approaches.

1 INTRODUCTION

Convolutional Neural Networks (CNNs) have shown that are able to obtain state-of-the-art results in most image classification problems [14, 13, 32, 28], but it is well-known that they have the drawback of requiring very large amounts of annotated data to be trained properly (either from scratch or using transfer learning processes). In some domains, like biological or medical sciences, it is difficult to obtain large amounts of annotated data from qualified specialists, given that manual data annotation is an error-prone and time-consuming task [31].

In this sense, the use of techniques that can work well with small amounts of annotated data becomes very attractive, as is the case of techniques based on Visual Dictionaries (Bag of Visual Words - BoVW, or Codebooks [29]). These techniques were the most accurate for image classification before the development of modern deep learning techniques [22, 26]. We revisit BoVW and propose in this paper a new BoVW approach that extracts deep features and organize them as local features using BoVW, avoiding the necessity of optimizing millions of parameters as CNNs tipically do (by back-propagation).

BoVW detects a set of interest points from the images and after representing them by local feature vectors (using some image characterization technique), a visual dictionary is built to represent groups of similar local vectors (i.e., using a clustering process), obtaining the visual words. They can then represent the images by their histograms of visual words. After characterization, a supervised classifier can be trained using the final feature vectors. This process traditionally does not take into account the classes of the images and, at the end, a single dictionary is used to represent the entire dataset. We believe, however, that the use of a small set of pre-annotated images can considerably improve the results.

The method presented in this paper extends the work developed in [2]. As such, it exploits the category information that is available by detecting the interest points for a given class using a superpixel coordinate space that is created after performing a stacked superpixel segmentation with all the images that belong to that class. This technique represents the features from all the images of a given class as a graph (after image alignment) and obtains the superpixels as optimum-path trees inside that graph. The superpixels found for a given class are a more suitable reference space for interest point detection for that class since the superpixels are essentially regions with similar features in the images (like color and texture) which also adhere to the objects' boundaries. Finally, once the interest points are defined for all the classes, one visual dictionary is defined for every class. The clustering technique that we use automatically obtains the number of groups from the data itself.

The main contribution of this work regards to the use of deep convolutional features to improve the quality of the local feature vectors that are extracted from the detected interest points. This then leads to the construction of more representative class-specific visual dictionaries which create codebook representations that are able to compete with deep neural networks on a scenario where there is a very limited set of labelled training images. We show that the proposed BoVW approach can be trained with hundreds of images and still obtain results that are both comparable to deep networks that were pre-trained with millions of samples from a different domain (the ImageNet dataset), and much better than deep networks trained from scratch with the same (small) labelled dataset.

The remaining of this paper is organized as follows. Section 2 presents the main concepts related to codebook learning, Section 3 reviews the related works and Section 4 presents the proposed method. Finally, Section 5 presents and discusses the experiments and Section 6 presents the conclusions.

2 VISUAL CODEBOOK REPRESENTATION

A *codebook* or *visual dictionary* $\mathcal{D}$ is a set of *visual words* that are used to represent the set of n images $\mathcal{I} = \{\mathcal{I}_1, \cdots, \mathcal{I}_n\}$ as a set of

[1] Laboratory of Image Data Science, Institute of Computing, University of Campinas, Brazil, email: cesar.fernandez@ic.unicamp.br
[2] Laboratory of Image Data Science, Institute of Computing, University of Campinas, Brazil, email: afalcao@ic.unicamp.br

feature vectors. This set of feature vectors are extracted following the BoVW pipeline.

During the *interest points detection* phase we need to detect the set of m interest points $\Psi_i = \{p_{i,1}, \cdots, p_{i,m}\}$ that better represent a given image $\mathcal{I}_i$.

Then, during the *local features extraction* phase we compute for each image $\mathcal{I}_i$ a set of m local feature vectors $\mathcal{L}_i = \{l_{i,1}, \cdots, l_{i,m}\}$ that represent the set Ψ_i of interest points that were previously detected.

Most of the methods in the literature [29, 9, 27, 11, 26, 22] use the Scale-Invariant Feature Transform (SIFT) [16] or the Speeded-Up Robust Features (SURF) [1] to detect and characterize the interest points, but, simpler approaches like random or grid sampling can also be used.

The *visual dictionary construction* phase consists in the definition of the g visual words $\mathcal{D} = \{\mathcal{W}_1, \cdots, \mathcal{W}_g\}$ that are extracted from the set of local feature vectors from all the images $\mathcal{L} = \{\mathcal{L}_1, \cdots, \mathcal{L}_n\}$.

The most common method used to construct the visual dictionary is k-means or some variant of it [29, 9, 5, 27, 11, 26], but this method requires to know the number of clusters *a-priori*. Thus, it is interesting to study other clustering techniques that can find the natural number of clusters from the data.

During the *global feature vectors encoding* phase the set of n feature vectors $\mathcal{F} = \{\mathcal{F}_1, \cdots, \mathcal{F}_n\}$ is built. A given vector $\mathcal{F}_i$ represents the image $\mathcal{I}_i$ and it is composed by a set of features $\{f_{i,1}, \cdots, f_{i,g}\}$ that are defined using the set of visual words $\mathcal{D}$ by assigning the local feature vectors $\mathcal{L}_i$ to their closest visual words.

Traditionally, the most used approach was hard assignment [29], i.e., consider only the closest visual word to every sample. However, the more successful works [26, 22] use soft assignment, i.e., consider the k closest visual words to every sample. The latter seems to be the most promising approach and is the one that we are using for this work.

Finally, the set of feature vectors $\mathcal{F}$ is used to train a mathematical model, i.e., a classifier that is able to separate the feature space into regions that correspond to the classes present in the problem. Since BoVW produces a large feature space, the most appropriate approach is to use a linear classifier like Support Vector Machines (SVM) [4] and it is the one used for this work.

3 RELATED WORKS

Fei-Fei and Perona [7] and Nowak *et al.* [20] showed that using simple approaches like random sampling can obtain results similar to SIFT or SURF in a fraction of the time. Moreover, similar to what we propose here, Tian and Wang [33] and Haas *et al.* [11] used superpixels to detect more robust interest points. However, those works compute the superpixels using the color features of each image, which is a different approach to ours in the sense that we are defining interest points that are common to all the images in a given class.

De Souza *et al.* [6] and Silva *et al.* [27] used simple descriptors like the Border/Interior Pixel Classification (BIC) [30] to obtain good results in less time than SURF approaches. BIC creates separated histograms for the homogeneous regions and the edges in the image, combining color and texture representations.

Recent works have shown that it is possible to mix BoVW and deep networks to improve the characterization of the interest points. Minaee *et al.* [18] used an adversarial autoencoder network to characterize the patches extracted from each image, outperforming some traditional methods. Gong *et al.* [9] proposed the use of features extracted with a CNN together with the features of a BoVW model and showed that their proposal slightly outperforms other deep learning architectures. Goh *et al.* [8] proposed a hybrid framework that combines a BoVW model with a hierarchical deep network using the BoVW as part of the fine-tuning process of the network. In our work we use deep features in a different way since we do not perform the back-propagation process of the neural neural network, we extract the deep features and encode them with a BoVW model.

Most of the methods in the literature use k-means or some variant of it to build the visual dictionary. Nister and Stewenius [19] proposed to use hierarchical k-means clustering (HKM) to quantize the feature vectors and Philbin *et al.* [23] proposed to use approximate k-means clustering (AKM), obtaining better results than HKM. Mikulik *et al.* [17] proposed an hybrid method that uses approximate hierarchical k-means clustering (AHKM) which outperformed both.

Nevertheless, k-means approaches need to know the number of clusters *a-priori*. De Souza *et al.* [6] have successfully used clustering by Optimum-Path Forest (OPF) [25] to build visual dictionaries, which does not require to know the number of clusters. However, their methodology builds a single dictionary to represent all the images, in contrast to ours which builds class-specific dictionaries.

As a matter of fact, most approaches in the literature, the traditional [29] and the most recent ones [18, 9, 5, 27, 6, 33, 11] define a single visual dictionary for the entire dataset. On the other hand, Zeng *et al.* [35] proposed a BoVW representation for visual tracking using one model for the object and another model for the background, which are both learned with a discriminative score using Bayesian inference. Also, Liu and Caselles [15] proposed the use of the label information as part of the feature vector and the cost function of the k-means algorithm to define specific visual words per class. One drawback of this approach, however, is the necessity of the number of clusters *a-priori*.

The most popular approach for feature vector encoding is soft assignment. Philbin *et al.* [24] showed that using multiple assignment improves the results obtained with a hard assignment approach proposed previously by themselves [23]. Jegou *et al.* [12] proposed a query-oriented multiple assignment approach that dynamically changes the number of chosen clusters. Mikulik *et al.* [17] also proposed a multiple assignment approach but using probabilistic relationships to choose the clusters. Cortes *et al.* [5] developed a new encoding strategy for BoVW to improve human action recognition in videos; they discriminate the non-informative visual words by analyzing the covering region of each word. Olaode and Naghdy [21] included spatial information of the visual words by detecting reoccurring patterns, thus generating what they called visual sentences to improve the performance of BoVW.

Silva *et al.* [27] proposed a framework to model BoVW as graphs, called Bag of Graphs, which adds spatial relations between visual words using the graph structure; the authors reported a good performance in several domains compared to traditional approaches.

4 PROPOSED METHOD

Fig. 1 summarizes the proposed method, which uses superpixel segmentation for interest point detection, random-weight convolutional features for local features representation and graph-based clustering for supervised visual dictionary construction.

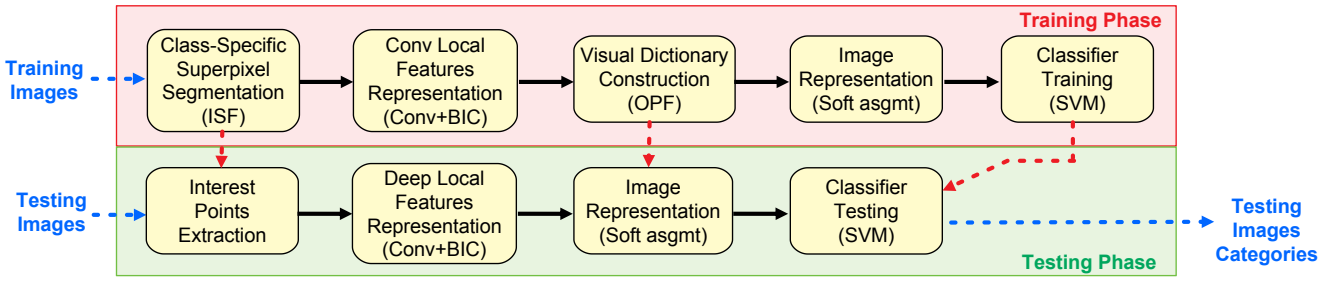

Figure 1. Proposed method for supervised BoVW construction based on random-weight convolutional features and superpixels.

4.1 Class-Specific Superpixel-Based Interest Point Detection

We propose an interest point detection method based on the segmentation of the image in superpixels defined for each class. For this, we use all the images of a given class to form a stacked image in which the superpixels are defined, creating a coordinate space that is common to all the images in that class. The points detected in this space will be related to the given class and also will be related to the object borders.

Our method is based on the Iterative Spanning Forest framework (ISF) [34], a graph-based method that defines the superpixels as trees inside the graph that represents the image.

4.1.1 *Initial seeds estimation*

We will use the segmentation masks from the parasite images produced by the methodology in [31] to define the initial seeds for ISF (per class).

Let $\lambda = \{\lambda_1, \cdots, \lambda_n\}$ be the labels and $\Phi = \{\phi_1, \cdots, \phi_n\}$ be the segmentation masks for the images $\mathcal{I} = \{\mathcal{I}_1, \cdots, \mathcal{I}_n\}$. The set of seeds $\mathcal{S}^{(i)}$ for class i are obtained by applying grid sampling inside the segmentation mask $\varphi^{(i)}$ of class i, which is defined as $\varphi^{(i)} = \bigcup_{\forall j, \lambda_j = i}[\phi_j]$.

Then, class-specific stacked images $\{\mathcal{I}^{(1)}, \cdots, \mathcal{I}^{(c)}\}$ are created for the c classes, such that the feature vector $\vec{\mathcal{I}}^{(i)}(p)$ for each pixel $p \in \mathcal{I}^{(i)}$ is defined as the union of the feature vectors from all the images in the class i, that is, $\vec{\mathcal{I}}^{(i)}(p) = \{\vec{\mathcal{I}}_j(p) \mid \forall j, \lambda_j = i\}$. This operation requires the images to be in the same reference space (simple or deformable image registration might be necessary).

Note that this procedure will also work in a scenario where we have multi-class images. In this case, the image $\mathcal{I}_i$ will have one segmentation mask for each class, i.e., $\phi_i = \{\phi_i^{(1)}, \cdots, \phi_i^{(c)}\}$.

4.1.2 *Superpixels definition with ISF*

Next, we use ISF to define the set of superpixels $\{\Omega^{(1)}, \cdots, \Omega^{(c)}\}$ for the c classes using the images $\{\mathcal{I}^{(1)}, \cdots, \mathcal{I}^{(c)}\}$ and the seeds $\{\mathcal{S}^{(1)}, \cdots, \mathcal{S}^{(c)}\}$.

In the ISF framework, a given stacked image $\mathcal{I}^{(i)}$ is represented as a graph $\mathcal{G}_1^{(i)} = (\mathcal{I}^{(i)}, \mathcal{A}_1)$, such that, a pixel $p \in \mathcal{I}^{(i)}$ is represent by its feature vector $\vec{\mathcal{I}}^{(i)}(p)$ in the node $v \in \mathcal{G}_1^{(i)}$. The arcs in the graph are defined by the adjacency relation $\mathcal{A}_1$ (e.g. 4-neighbors) and they are weighted by a cost function $f_1(\cdot)$ that takes into account distances in the feature space and in the image domain.

A path $\pi_{s \rightsquigarrow t} = \langle s = t_1, t_2, \cdots, t_n = t \rangle$ is a sequence of adjacent pixels from s to t, being that π_t can be the result of extend π_s by an arc (s, t). The function $f_1(\cdot)$ can be used to compute an *optimum-path* π_t^* to any $t \in \mathcal{I}^{(i)}$, provided that π_t^* is optimum according to the function $f_1(\cdot)$, that is, $f_1(\pi_t) \leq f_1(\tau_t)$ for any other path $\tau_t \in \Pi_{\mathcal{G}_1^{(i)}}$ (the set of paths in $\mathcal{G}_1^{(i)}$). According to this, we can obtain an optimal partition of the feature space as:

$$\Pi^*_{\mathcal{G}_1^{(i)}}(t) = \min_{\forall \pi_t \in \Pi_{\mathcal{G}_1^{(i)}}} \{f_1(\pi_t)\} \quad (1)$$

where $\Pi^*_{\mathcal{G}_1^{(i)}}$ is a *predecessor map* with no cycles representing a *optimum-path forest* that contains the optimum path π_t^* for every $t \in \mathcal{G}_1^{(i)}$. The set of optimum trees $\{\omega_1, \cdots, \omega_{s^{(i)}}\} \in \Pi^*_{\mathcal{G}_1^{(i)}}$ constitutes the set of superpixels $\Omega^{(i)}$ for class i.

The optimization process from Eq. 1 takes a set of initial seeds as input and then recomputes them according to the superpixels, repeating the entire process with the recomputed seed by a given number of iterations.

According to [34], the cost function $f_1(\cdot)$ can be defined as:

$$f_1(\pi_{s_j \rightsquigarrow s} \cdot \langle s, t \rangle) = f_1(\pi_s) + (||\vec{v}(t) - \vec{\mu}(s_j)||\alpha)^\beta + ||s, t|| \quad (2)$$

where $\alpha \geq 0$ controls shape regularity, $\beta \geq 1$ controls boundary adherence, $\vec{v}(t)$ is the feature vector of pixel t and $\vec{\mu}(s_j)$ is the mean feature vector of the superpixel of the previous iteration whose seed is s_j.

4.1.3 *Interest points detection*

Next, for each set $\Omega^{(i)}$ we compute the superpixels' boundaries and perform a grid sampling inside those boundaries, i.e., given a stride value, a set of uniformly-spaced points $\psi^{(i)} = \{p_1, \cdots, p_{m^{(i)}}\}$ that follows the superpixels' boundaries is chosen (interest points for class i). Finally, the set Ψ_i of m interest points for a given image $\mathcal{I}_i$ can be defined as $\Psi_i = \{\psi^{(1)}, \cdots, \psi^{(c)}\}$, where $m = |\psi^{(1)}| + \cdots + |\psi^{(c)}|$.

4.2 Random-Weight Convolutional Features as Local Feature Vectors

We propose to use a Convolutional Neural Network (CNN) with the aim of improving the quality of the local feature vectors that are extracted to represent the interest points. We are basically applying the set of operations of the network to the patches that are defined around

each point. The result is a set of filtered images for each patch and then the BIC algorithm [30] is used to encode the information on the resulting images and form the local feature vectors.

On a traditional CNN, a back-propagation process is normally performed during a given number of epochs to refine some set of pre-trained weights that were obtained with another dataset (transfer learning). This process, however, is time-consuming and it is not straightforward to define the parameters and the methods that will be used, such as the number of epochs, learning rate, optimization algorithm, loss function, etc. Different choices at this point can lead to very different results and this has become in one of the biggest problem in deep learning nowadays.

We aim at using the CNN as a feature extractor and, as such, it is not necessary to have the classification layers (i.e., fully-connected and softmax) that are needed to perform back-propagation. Also, it is important to mention that by using random weights we concentrate in network architecture learning instead of parameter learning [3]. We use random kernel weights that are standardized to have zero-mean and one-norm, which turn them into texture extractors.

For each image $\mathcal{I}_i$ we extract the patches $\Sigma_i = \{\sigma_{i,1}, \cdots, \sigma_{i,m}\}$ that correspond to the interest points Ψ_i and perform a feed-forward pass with every $\sigma_{i,j}$, which produces a set of n_k filtered images $\hat{\sigma}_{i,j} = \{\hat{\sigma}_{i,j,1}, \cdots, \hat{\sigma}_{i,j,n_k}\}$, where n_k depends on the architecture of the CNN.

Regardless of the chosen architecture, we need to set the stride for the convolution and max-pooling operations to 1, since the BIC algorithm that is used in the next stage requires the images to be in their original size. Another option could be to resize the images but according to our tests this produces worse results. This is probably because the resizing process produces noise while a stride value of 1 preserves the information in the resulting filtered image.

4.2.1 Convolutional features encoding using BIC

After obtaining the filtered patch images $\hat{\sigma}_{i,j}$, the most common encoding strategy for a CNN is using flattening over $\hat{\sigma}_{i,j}$. However, this would lead to a very large local feature vector.

The Border/Interior Pixel Classification descriptor (BIC) [30] is a very efficient descriptor that encodes texture in an image by analyzing the neighborhood of each pixel to determine if it belongs to a border or interior (homogeneous) region.

For a given filtered patch image $\hat{\sigma}_{i,j,r}$, we compute the quantized image $\hat{\sigma}'_{i,j,r}$ (i.e., a simplified image with less colors) and for every pixel $p \in \hat{\sigma}'_{i,j,r}$ we say that:

- p is a *border* pixel $\Leftrightarrow \exists q \in \mathcal{A}_2(p)\ ,\ \hat{\sigma}'_{i,j,r}(p) \neq \hat{\sigma}'_{i,j,r}(q)$
- p is an *interior* pixel $\Leftrightarrow \forall q \in \mathcal{A}_2(p)\ ,\ \hat{\sigma}'_{i,j,r}(p) = \hat{\sigma}'_{i,j,r}(q)$

where $\mathcal{A}_2(p)$ is defined as the 4-neighbor adjacency for pixel p.

Next, we compute the color histograms $h^1_{i,j,r}$ and $h^2_{i,j,r}$ of the border and interior pixels that were defined and the feature vector of $\hat{\sigma}_{i,j,r}$ is defined as $h_{i,j,r} = h^1_{i,j,r} + h^2_{i,j,r}$. Finally, the feature vector $\mathcal{H}_{i,j}$ for the image patch $\sigma_{i,j}$ is defined as $\mathcal{H}_{i,j} = \{h_{i,j,1} + \cdots + h_{i,j,n_k}\}$ and the set of local feature vectors for image $\mathcal{I}_i$ is defined as $\mathcal{L}_i = \{\mathcal{H}_{i,1}, \cdots, \mathcal{H}_{i,m}\}$.

4.3 Class-Specific Visual Dictionary Learning

We use the Optimum-Path Forest classifier (OPF) [25] to build the visual dictionaries. OPF does not need to know the number of clusters *a-priori*, since it follows the distribution of the groups in the feature space. Moreover, different from most of the approaches in the literature, we build specific visual dictionaries for each class, which gives better results.

Note that using specific dictionaries for each class produces more portable visual dictionaries across different applications. Whenever a new application contains some of the classes used to create the visual dictionaries, the extracted knowledge can be harnessed.

4.3.1 Initial graph definition

Given the set of local feature vectors $\mathcal{L}$, we define the set $\mathcal{L}' = \{\mathcal{L}^{(1)}, \cdots, \mathcal{L}^{(c)}\}$, such that $\mathcal{L}^{(i)}$ represents the local feature vectors from class i.

Next, we define a graph $\mathcal{G}_2^{(i)} = (\mathcal{L}^{(i)}, \mathcal{A}_3)$ using the feature vectors from class i, which is a k-nn graph, i.e., $q \in \mathcal{A}_3(p) \Leftrightarrow q$ is one of the k nearest neighbors of p. A given sample $s \in \mathcal{G}_2^{(i)}$ is represented by its feature vector $\vec{v}(s) \in \mathcal{L}^{(i)}$ and it is weighted by the Probability Density Function (PDF) of s. The PDF measures how densely are the samples distributed in the feature space and according to [25], it can be computed using the k-nn graph as:

$$\rho(s) = \frac{1}{\sqrt{2\pi\sigma^2}|\mathcal{A}_3(s)|} \sum_{t \in \mathcal{A}_3(s)} \exp\left(\frac{-d^2(s,t)}{2\sigma^2}\right) \tag{3}$$

where $\sigma = \max_{\forall (s,t) \in \mathcal{A}_3} \left\{\frac{d(s,t)}{3}\right\}$ and $d(s,t)$ is the Euclidean distance between $\vec{v}(s)$ and $\vec{v}(t)$.

4.3.2 Clusters computation using OPF

For a given graph $\mathcal{G}_2^{(i)}$, the clusters are defined as the set of optimum trees $\{\mathcal{T}_1, \cdots, \mathcal{T}_{g^{(i)}}\}$ that belong to the *optimum-path forest* $\Pi^*_{\mathcal{G}_2^{(i)}}$ resultant from the partition of that graph according to a cost function $f_2(\cdot)$.

The optimum-path forest $\Pi^*_{\mathcal{G}_2^{(i)}}$ can be defined as:

$$\Pi^*_{\mathcal{G}_2^{(i)}}(t) = \max_{\forall \pi_t \in \Pi_{\mathcal{G}_2^{(i)}}} \{f_2(\pi_t)\} \tag{4}$$

where $\Pi^*_{\mathcal{G}_2^{(i)}}$ is represented as a *predecessor map* with no cycles.

According to Rocha *et al.* [25], the function f_2 is defined as:

$$f_2(\langle t \rangle) = \begin{cases} \rho(t) \text{ if } t \in \mathcal{R}_i \\ h(t) \text{ otherwise} \end{cases}$$
$$f_2(\pi_s \cdot \langle s,t \rangle) = \min\{f_2(\pi_s), \rho(t)\} \tag{5}$$

where $\mathcal{R}^{(i)}$ is the set of roots in $\mathcal{G}_2^{(i)}$ and $h(t)$ is a *handicap* value.

The set of roots $\mathcal{R}^{(i)}$ is defined as the maxima of the PDF distribution of $\mathcal{G}_2^{(i)}$, which dictates the number of clusters in the OPF graph. The handicap value $h(t)$ is used to avoid local maxima in the PDF (i.e., too many clusters).

The parameter k in the graph $\mathcal{G}_2^{(i)}$ controls the granularity of the clusters and it can be optimized in a given interval $[1 \cdots k_{max}]$ using the graph-cut measure.

4.3.3 Visual dictionary definition

The set of visual words for class i can be defined as $\mathcal{W}^{(i)} = \{r_1, \cdots, r_{g^{(i)}}\}$, where r_i is the root of the optimum tree $\mathcal{T}_i \in \mathcal{G}_2^{(i)}$. Finally, the visual dictionary $\mathcal{D}$ for the entire image set $\mathcal{I}$ is defined as $\mathcal{D} = \{\mathcal{W}^{(1)} + \cdots + \mathcal{W}^{(c)}\}$.

5 EXPERIMENTS

In this section we present the experimental setup, the datasets and measures that were used and also several tests that evaluate different aspects of the proposed method.

5.1 Experimental setup

All the experiments related to BoVW were performed using an Intel® Core i7-7700 processor with 64 GB of RAM, running Ubuntu 16.04, and all the experiments regarding traditional deep learning models were implemented with PyTorch and executed in a Nvidia® Titan Xp with 12 GB of RAM.

5.2 Datasets

The context of evaluation of the proposed methodology is the classification of intestinal parasite images, which is a domain where image annotation requires knowledge from trained specialists. As such, we deal with large unsupervised datasets.

We used optical microscopy images from the 15 most common species of human intestinal parasites in Brazil, organized in two datasets by the methodology proposed in [31]: *Protozoan cysts* and *Helminth eggs*. Each dataset contains several classes that are very similar among them (Fig. 2a and Fig. 2b) and also very similar to fecal impurities (Fig. 2c). Table 1 shows the details of the datasets, where the underlined values represent the impurities in each dataset.

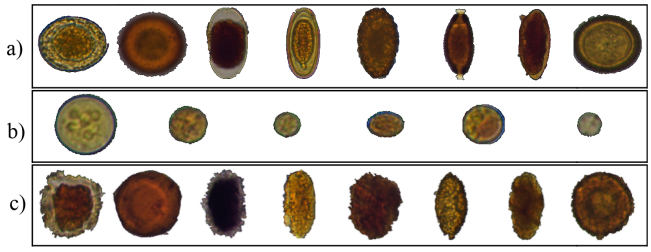

Figure 2. Examples of a) *Helminth Eggs*, b) *Protozoan Cysts* and c) Fecal impurities.

All the tests use the soft assignment methodology to build the final feature vector for the images and a Linear SVM classifier (training and testing phases).

The performance metrics used are: a) Accuracy (percentage of true positives); and b) Cohen's Kappa (level of agreement between the classifier and the dataset ground-truth, considering the results obtained for each class) [10].

Table 1. Datasets of the 15 species of intestinal parasites.

Dataset	Samp	Class	Samples per class
P.Cysts	17696	6+1	869, 659, 1783, 1931, 3297 309, 8848
H.Eggs	5751	8+1	501, 83, 286, 103, 835, 435 254, 379, 2875

In the following sections we will use a reduced version of the *P.Cysts* dataset without impurities to discuss several aspects of the method and then, we will show the final results for all the datasets. The dataset was splitted using stratified sampling into: 1) Z_{FL} for feature learning (10%); 2) Z_{CL} for classifier learning (45%) and 3) Z_E for classifier evaluation (45%).

5.3 Class-specific interest points detection and dictionary estimation

We will first address the impact of the use of category information to both interest point detection and dictionary estimation, as well as the benefits of using the class-specific superpixel segmentation. Since we aim at showing the impact of the aforementioned aspects independently of the use of deep convolutional features, we use the vanilla BIC descriptor in this section.

Table 2 shows the results. As we can see, the best results are obtained using class-specific superpixels (ISF-class) and class-specific dictionaries (OPF-class). Moreover, we can also see that ISF-class is better than Grid sampling regardless of the chosen strategy to build the dictionary (Unsup-OPF or OPF-class). At the same time, the use of OPF-class is better than Unsup-OPF regardless of the chosen strategy to extract the interest points (Grid or ISF-class). On the other hand, the number of visual words is larger for the chosen methodology, but this is justified by the higher accuracy rates.

Table 2. Evaluation of dictionary estimation and interest points detection methods.

Measure	Unsup-OPF Grid	OPF-class Grid	OPF-class ISF-class
kappa	80.53	81.68	84.09
# words	576	804	1266

5.4 CNN architectures for local feature vectors extraction

We will now address the impact of the CNNs to build the local feature vectors. We first test simple random-weight CNN architectures, such as networks with 1, 2 and 3 layers and then compare those results with well-known architectures like AlexNet [14] (Table 3 shows the architectures used). All the tests use ISF-class to detect the interest points and OPF-class to build the visual dictionaries. Since we want to focus on studying the impact of the network architecture and not the encoding strategy, we are using the raw values of the filtered images produced by the CNNs (flattening).

Table 3. Architectures of the random-weight networks tested.

Network	Layers	Kernels		ReLU	Pool
		Num	Size		Size
1 layer	1	8	9	yes	4
2 layers	1,2	8,16	9,9	yes	4,4
3 layers	1,2,3	8,16,32	9,9,9	yes	4,4,4
AlexNet	1,2,5	64,192,256	11,5,3	yes	3,3,3
	3,4	384,256	3,3	yes	–

Table 4 shows the results for the different network architectures and with different stride values. The patch size around each interest point is set to 9 and, for the sake of speed, the stride for the superpixel-based interest point detector is set to 25.

The first aspect that is worth mentioning is that the AlexNet and many other popular CNN architectures use stride values in the middle of the convolution and max-pooling operations. This is done for speed-up reasons since the images are being down-sampled as the

Table 4. Evaluation of network architectures and stride values (using raw convolutional features).

Metric	1 layer		2 layers	3 layers	AlexNet
	strd=4	strd=1	strd=1	strd=1	strd=1
kappa	68.91	72.78	73.15	74.21	77.96
accuracy	76.88	79.59	79.84	80.59	83.29
#words	332	439	406	437	536
#feats p/word	648	648	1296	2592	20736

network goes deeper, but it is not clear if this causes losing information. Hence, we tested this aspect and the use of a stride value of 1 gives almost 4% more in accuracy (columns 2 and 3 in Table 4).

Next, we fixed the stride to 1 and tested the networks with 2 and 3 layers and the AlexNet. As we can see in the columns 3-6 of Table 4 the more layers we add to the network, the higher the accuracy (the kappa measure went from 72.78 to 77.96) and also the higher the number of words.

Nevertheless, one problem of using a more complex architecture is that the dimension of the visual words grows up significantly (it went from 2592 to 20736 features). At this point, the use of the BIC algorithm to characterize the filtered images becomes very useful, since it will quantize the feature vectors and produce a representation of the same dimension regardless of the chosen architecture.

Table 5 shows the results of using the BIC algorithm over the convolutional features with 4, 8 and 16 bins compared with the prior scheme. Note that for these tests we are using a stride value of 7 in the interest point detection phase to improve the accuracy results.

Table 5. Comparison of flattened convolutional features and BIC convolutional features. All the tests use the AlexNet network.

Metric	Flat	BIC		
		4 bins	8 bins	16 bins
kappa	86.98	85.30	86.83	86.80
accuracy	90.14	88.88	90.01	90.01
#words	4713	3210	4268	8301
#feats per word	20736	2048	4096	8192

As the table shows, the performance is equivalent in terms of kappa, but there is an important difference in terms of the number of features of each visual word (almost 5x less features for BIC-4). The impact of the number of features per word relies on the time needed to obtain the visual words (clustering algorithm). We will then use the BIC-8 variant to have a balance between kappa score and number of features per word.

5.5 Comparison with traditional CNN methods

We are now interested in evaluating how well our method performs in comparison to a traditional CNN.

We used 3 baselines based on the AlexNet architecture for comparison:

1. AN_{FT}: pre-trained weights, fine tuning with Z_{FL} + Z_{CL} and evaluation with Z_E.
2. AN_{FF}: pre-trained weights, frozen feat extraction layers, training with Z_{FL} + Z_{CL} and evaluation with Z_E.
3. AN_{RW}: random weights, training with Z_{FL} + Z_{CL} and evaluation with Z_E.

Note that we use Z_{FL} + Z_{CL} to train the CNN models since they are used during the learning phase of our method. However, our model uses only Z_{FL} to learn the features (visual words) and Z_{CL} to train the SVM.

All the CNN baselines use Cross Entropy Loss, SGD optimizer, 100 epochs and a learning rate of 0.001 with a scheduler that multiplies its value by 0.1 every 7 epochs. The size of the batch can have an important impact on the performance, so we conducted tests with different batch sizes.

Table 6 shows the results for our method and the 3 baselines aforementioned using batch sizes of 4, 16 and 64. As we can see, AN_{FT} obtained the best results, followed by our method. However, note that our method obtained approximately 7% more accuracy than AN_{FF} and approximately 50% more accuracy than AN_{RW}. Also, it is important to note that the advantage in accuracy obtained by AN_{FT} depends on the batch size.

Table 6. Comparison of our methodology with traditional CNNs.

Metric	B.S.	Method			
		Ours	AN_{FT}	AN_{FF}	AN_{RW}
kappa	4	86.83	94.52	78.64	0.00
	16		90.74	74.32	0.00
	64		80.35	66.88	6.95
acc	4	90.01	95.85	83.96	37.32
	16		93.03	80.82	37.32
	64		85.32	75.56	40.50

We would like to highlight that our method performed very well considering that the network that used transfer learning was trained using millions of images from the ImageNet dataset. Also note that the accuracy is very low for the network initialized with random weights, because it is not using transfer learning. Moreover, the kappa value is 0 because the accuracy was 0% for some of the classes.

5.6 Comparison with other BoVW methods

In this section we compare the performance of our method to other BoVW approaches:

1. BoVW_{OPF}: grid samp, BIC desc, OPF and soft asgmt [6].
2. BoVW_{KM}: grid samp, BIC desc, k-Means and soft asgmt [29].

Table 7 presents the results of out method compared to the 2 baselines. As we can see, our method obtained the highest accuracy. The advantage over BoVW_{KM} demonstrates that OPF creates better dictionaries and the advantage over BoVW_{OPF} demonstrates that the use of class-specific superpixel segmentation detects better interest points and that the use of convolutional features improves the local features.

Table 7. Comparison of our method with other BoVW methods.

Metric	Method		
	Ours	BoVW_{OPF}	BoVW_{KM}
kappa	86.83	81.05	44.03
accuracy	90.01	85.65	59.79
#words	4268	587	1000
#feats per word	4096	1024	1024

5.7 Final results

On this section we present the results for all the methods using all the datasets. In this case, the datasets were splitted using stratified sampling into: 1) Z_{FL} for feature learning (1%); 2) Z_{CL} for classifier learning (49.5%) and 3) Z_E for classifier evaluation (49.5%).

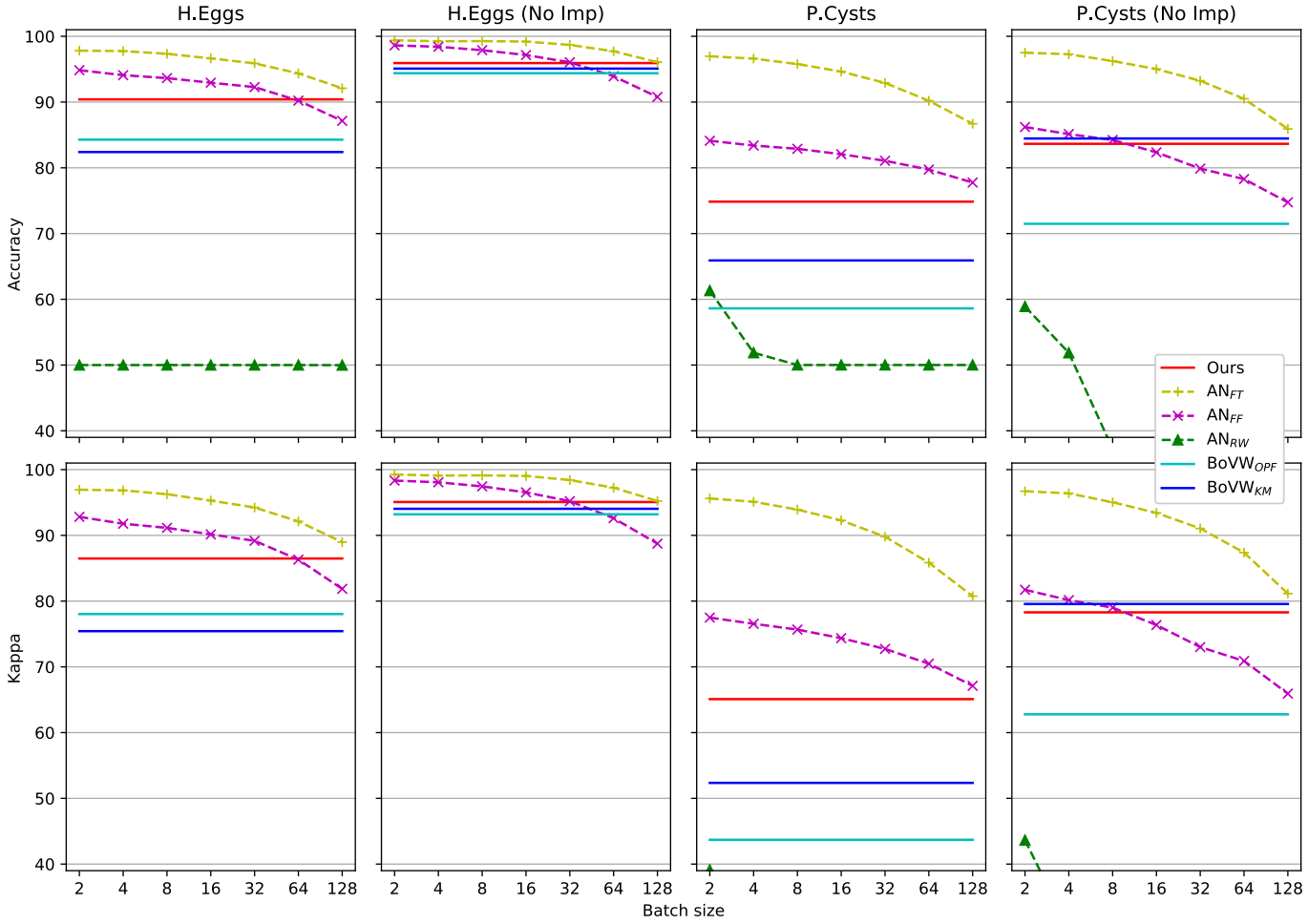

Figure 3. Accuracy and kappa results for all the methods in all the datasets (with and without impurities).

Note that we are using only 1% of each dataset to learn the features (visual dictionary) since we aim at demonstrating that it is possible to construct very good dictionaries from small labelled datasets.

Fig. 3 shows the accuracy and kappa results for our method compared to the other methods using all the datasets with and without impurities (resulting in 4 datasets). Note that the plots show results for the CNN models using batch sizes ranging from 2 to 128. Also, Table 8 presents a summary of the plots, considering only the best results for the CNN models.

As we can see, our method is better than the other BoVW approaches in 3 of 4 datasets, is much better than the AN_{RW} approach in all the cases and comparable in some cases to AN_{FF} (which used the pre-trained weights but did not refine them). Note that in some plots AN_{RW} is missing, that is because it obtained less than 40%.

We would like to highlight that in the case of the *H.Eggs* datasets, our method obtained a difference of more than 30% (with impurities) and more than 60% (without impurities) compared to AN_{RW}. This is an indication that a CNN needs to be trained with with a lot of labelled images, while our method is able to learn features from a training set with as low as 57 images (*H.Eggs* dataset). Finally, note the difference in the use of the accuracy and the kappa measures. In some cases, the accuracy is almost 20% higher than the kappa measure which is an evidence that the accuracy is not a good measure since it camouflages the classification errors in some classes.

Table 8. Summary of the final kappa results for all the methods in all the datasets (with and without impurities)

Method	Dataset			
	H.Eggs	***H.Eggs (N.I.)***	***P.Cysts***	***P.Cysts (N.I.)***
Ours	86.5 ± 0.3	95.1 ± 1.3	65.1 ± 8.1	78.3 ± 2.1
AN_1	96.9 ± 0.6	99.3 ± 0.2	95.6 ± 0.4	96.7 ± 0.2
AN_2	92.8 ± 0.3	98.3 ± 0.3	77.5 ± 0.6	81.7 ± 0.8
AN_3	0.0 ± 0.0	7.2 ± 10.1	39.1 ± 3.8	43.6 ± 3.4
BoVW_1	78.0 ± 1.2	93.2 ± 0.5	43.7 ± 2.3	62.8 ± 4.9
BoVW_2	75.4 ± 0.8	94.0 ± 0.4	52.4 ± 2.7	79.6 ± 0.5

AN_1: AN_{FT}, AN_2: AN_{FF}, AN_3: AN_{RW}, BoVW_1: BoVW_{OPF}, BoVW_2: BoVW_{KM}

6 CONCLUSIONS

We have presented a novel approach to learn class-specific visual dictionaries from a set of interest points extracted from a set of class-specific stacked images. We showed that the characterization of the interest points with random-weight convolutional features encoded

with the BIC descriptor builds better local feature vectors. We performed comparisons of our method with several baselines, showing that our method achieves better performance than traditional BoVW approaches. Also, we showed that it is possible to build effective visual dictionaries from very small sets of labelled images which are able to beat a CNN initialized with random weights and that can also be comparable with CNN models loaded with pre-trained weights.

ACKNOWLEDGEMENTS

The authors thank FAPESP (grants 2014/12236-1 and 2017/03940-5), CNPq (grant 303808/2018-7) and CAPES (finance code 001) for the financial support.

REFERENCES

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, 'Speeded-Up Robust Features (SURF)', *Computer Vision and Image Understanding*, **110**(3), 346 – 359, (2008). Similarity Matching in Computer Vision and Multimedia.

[2] C. Castelo-Fernández and A.X. Falcão, 'Learning Visual Dictionaries from Class-Specific Superpixel Segmentation', in *Computer Analysis of Images and Patterns (CAIP'19)*, pp. 171–182. Springer International Publishing, (2019).

[3] G. Chiachia, A. X. Falcão, N. Pinto, A. Rocha, and D. Cox, 'Learning Person-Specific Representations From Faces in the Wild', *IEEE Transactions on Information Forensics and Security*, **9**(12), 2089–2099, (Dec 2014).

[4] C. Cortes and V. Vapnik, 'Support-Vector Networks', *Machine Learning*, **20**(3), 273–297, (Sep 1995).

[5] X. Cortés, D. Conte, and H. Cardot, 'A New Bag of Visual Words Encoding Method for Human Action Recognition', in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 2480–2485, (Aug 2018).

[6] Luis A. de Souza, L.C.S. Afonso, A. Ebigbo, A. Probst, H. Messmann, R. Mendel, C. Hook, C. Palm, and J.P. Papa, 'Learning Visual Representations with Optimum-Path Forest and its Applications to Barrett's Esophagus and Adenocarcinoma Diagnosis', *Neural Computing and Applications*, (Jan 2019).

[7] L. Fei-Fei and P. Perona, 'A Bayesian Hierarchical Model for Learning Natural Scene Categories', in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pp. 524–531 vol. 2, (June 2005).

[8] H. Goh, N. Thome, M. Cord, and J. Lim, 'Learning Deep Hierarchical Visual Feature Coding', *IEEE Transactions on Neural Networks and Learning Systems*, **25**(12), 2212–2225, (Dec 2014).

[9] X. Gong, L. Yuanyuan, and Z. Xie, 'An Improved Bag-of-Visual-Word Based Classification Method for High-Resolution Remote Sensing Scene', in *2018 26th International Conference on Geoinformatics*, pp. 1–5, (June 2018).

[10] K.L. Gwet, *Handbook of Inter-Rater Reliability: The Definitive Guide to Measuring the Extent of Agreement Among Raters*, Advanced Analytics, LLC., Gaithersburg, MD, 4 edn., September 2014.

[11] S. Haas, R. Donner, A. Burner, M. Holzer, and G. Langs, 'Superpixel-Based Interest Points for Effective Bags of Visual Words Medical Image Retrieval', in *Medical Content-Based Retrieval for Clinical Decision Support*, pp. 58–68, Berlin, Heidelberg, (2012). Springer Berlin Heidelberg.

[12] H. Jégou, M. Douze, and C. Schmid, 'Improving Bag-of-Features for Large Scale Image Search', *International Journal of Computer Vision*, **87**(3), 316–336, (2010).

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, 'Caffe: Convolutional Architecture for Fast Feature Embedding', in *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, pp. 675–678, New York, NY, USA, (2014). ACM.

[14] A. Krizhevsky, I. Sutskever, and G.E. Hinton, 'ImageNet Classification with Deep Convolutional Neural Networks', in *Advances in Neural Information Processing Systems 25*, 1097–1105, Curran Associates, Inc., (2012).

[15] Y. Liu and V. Caselles, 'Supervised Visual Vocabulary with Category Information', in *Advanced Concepts for Intelligent Vision Systems*, pp. 13–21, Berlin, Heidelberg, (2011). Springer Berlin Heidelberg.

[16] D.G. Lowe, 'Distinctive Image Features from Scale-Invariant Keypoints', *Int. J. Comput. Vision*, **60**(2), 91–110, (November 2004).

[17] Mikulik, A. and Perdoch, M. and Chum, O. and Matas, J., 'Learning Vocabularies over a Fine Quantization', *International Journal of Computer Vision*, **103**(1), 163–175, (2013).

[18] S. Minaee, Y. Wang, A. Aygar, S. Chung, X. Wang, Y. W. Lui, E. Fieremans, S. Flanagan, and J. Rath, 'MTBI Identification from Diffusion MR Images using Bag of Adversarial Visual Features', *IEEE Transactions on Medical Imaging*, (2019).

[19] D. Nister and H. Stewenius, 'Scalable Recognition with a Vocabulary Tree', in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pp. 2161–2168, Washington, DC, USA, (2006). IEEE Computer Society.

[20] E. Nowak, F. Jurie, and B. Triggs, *Sampling Strategies for Bag-of-Features Image Classification*, 490–503, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[21] A. A. Olaode and G. Naghdy, 'Elimination of Spatial Incoherency in Bag-of-Visual Words Image Representation using Visual Sentence Modelling', in *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pp. 1–6, (Nov 2018).

[22] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier, 'Large-scale image retrieval with compressed Fisher vectors', in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3384–3391, (June 2010).

[23] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, 'Object Retrieval with Large Vocabularies and Fast Spatial Matching', in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, (June 2007).

[24] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, 'Lost in Quantization: Improving Particular Object Retrieval in Large Scale Image Databases', in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8, (June 2008).

[25] L.M. Rocha, F.A.M. Cappabianco, and A.X. Falcão, 'Data Clustering as an Optimum-path Forest Problem with Applications in Image Analysis', *Int. J. Imaging Syst. Technol.*, **19**(2), 50–68, (June 2009).

[26] J. Sánchez and F. Perronnin, 'High-Dimensional Signature Compression for Large-Scale Image Classification', in *CVPR 2011*, pp. 1665–1672, (June 2011).

[27] F.B. Silva, R. de O. Werneck, S. Goldenstein, S. Tabbone, and R. da S. Torres, 'Graph-Based Bag-of-Words for Classification', *Pattern Recognition*, **74**, 266 – 285, (2018).

[28] K. Simonyan and A. Zisserman, 'Very Deep Convolutional Networks for Large-Scale Image Recognition', *arXiv*, **abs/1409.1556**, (2014).

[29] J. Sivic and A. Zisserman, 'Video Google: A Text Retrieval Approach to Object Matching in Videos', in *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pp. 1470–1477, Washington, DC, USA, (2003). IEEE Computer Society.

[30] R.O. Stehling, M.A. Nascimento, and A.X. Falcão, 'A Compact and Efficient Image Retrieval Approach Based on Border/Interior Pixel Classification', in *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM '02, pp. 102–109, New York, NY, USA, (2002). ACM.

[31] C. T. N. Suzuki, J. F. Gomes, A. X. Falcão, J. P. Papa, and S. Hoshino-Shimizu, 'Automatic Segmentation and Classification of Human Intestinal Parasites from Microscopy Images', *IEEE Transactions on Biomedical Engineering*, **60**(3), 803–812, (March 2013).

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S.E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, 'Going Deeper with Convolutions', *arXiv*, **abs/1409.4842**, (2014).

[33] L. Tian and S. Wang, 'Improved Bag-of-Words Model for Person Re-Identification', *Tsinghua Science and Technology*, **23**(2), 145–156, (April 2018).

[34] J.E. Vargas-Muñoz, A.S. Chowdhury, E.B. Alexandre, F.L. Galvão, P.A.V. Miranda, and A.X. Falcão, 'An Iterative Spanning Forest Framework for Superpixel Segmentation', *IEEE Transactions on Image Processing*, **28**(7), 3477–3489, (July 2019).

[35] F. Zeng, Y. Ji, and M.D. Levine, 'Contextual Bag-of-Words for Robust Visual Tracking', *IEEE Transactions on Image Processing*, **27**(3), 1433–1447, (March 2018).