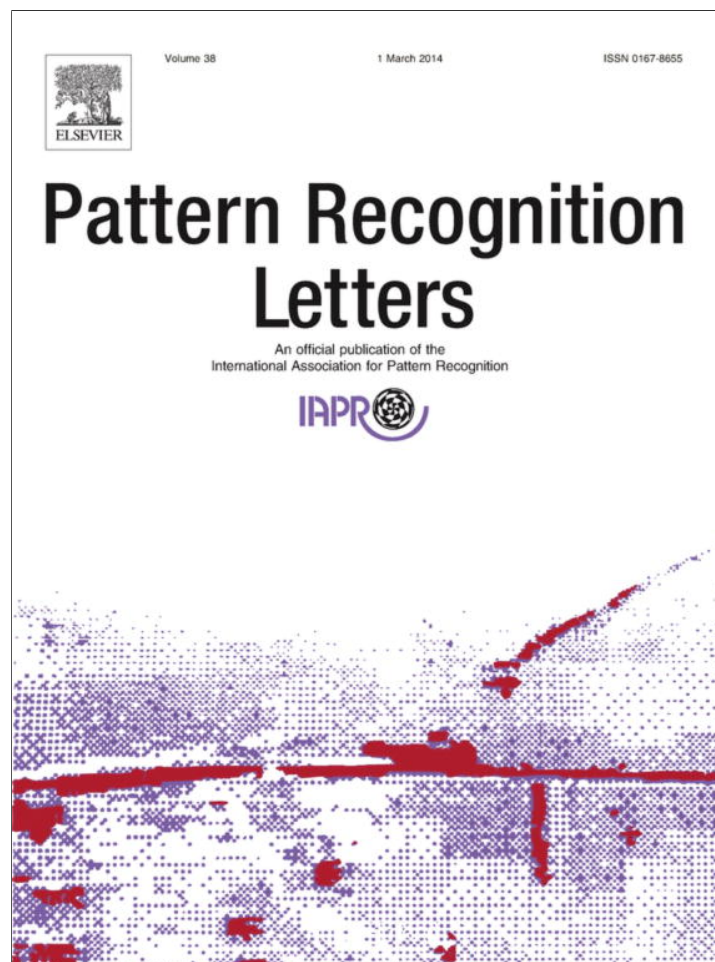


Provided for non-commercial research and education use.  
Not for reproduction, distribution or commercial use.



This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

<http://www.elsevier.com/authorsrights>



ELSEVIER

Contents lists available at ScienceDirect

## Pattern Recognition Letters

journal homepage: [www.elsevier.com/locate/patrec](http://www.elsevier.com/locate/patrec)

## Fast and efficient visual codebook construction for multi-label annotation using predictive clustering trees

Ivica Dimitrovski<sup>a,\*</sup>, Dragi Kocev<sup>b</sup>, Suzana Loskovska<sup>a</sup>, Sašo Džeroski<sup>b</sup><sup>a</sup> Faculty of Computer Science and Engineering, Ss Cyril and Methodius University, Rugjer Boshkovikj 16, MK-1000 Skopje, Macedonia<sup>b</sup> Department of Knowledge Technologies, Jožef Stefan Institute, Jamova cesta 39, SI-1000 Ljubljana, Slovenia

## ARTICLE INFO

## Article history:

Received 10 January 2013

Available online 6 November 2013

Communicated by Eckart Michaelsen

## Keywords:

Automatic image annotation

Visual codebook construction

Predictive clustering trees

Multi-label classification

## ABSTRACT

The bag-of-visual-words approach to represent images is very popular in the image annotation community. A crucial part of this approach is the construction of visual codebook. The visual codebook is typically constructed by using a clustering algorithm (most often  $k$ -means) to cluster hundreds of thousands of local descriptors/key-points into several thousands of visual words. Given the large numbers of examples and clusters, the clustering algorithm is a bottleneck in the construction of bag-of-visual-words representations of images. To alleviate this bottleneck, we propose to construct the visual codebook by using predictive clustering trees (PCTs) for multi-label classification (MLC). Such a PCT is able to assign multiple labels to a given image, i.e., to completely annotate a given image. Given that PCTs (and decision trees in general) are unstable predictive models, we propose to use a random forest of PCTs for MLC to produce the overall visual codebook. Our hypothesis is that the PCTs for MLC can exploit the connections between the labels and thus produce a visual codebook with better discriminative power. We evaluate our approach on three relevant image databases. We compare the efficiency and the discriminative power of the proposed approach to the literature standard –  $k$ -means clustering. The results reveal that our approach is much more efficient in terms of computational time and produces a visual codebook with better discriminative power as compared to  $k$ -means clustering. The scalability of the proposed approach allows us to construct visual codebooks using more than usually local descriptors thus further increasing its discriminative power.

© 2013 Elsevier B.V. All rights reserved.

## 1. Introduction

Many of the popular methods for image annotation are using a bag-of-visual-words to represent the visual content of an image (Nowak and Dunker, 2010; Everingham et al., 2012; van de Sande et al., 2010). The basic idea of this approach is first to sample a set of key-points (i.e., local regions) from the image using some method (e.g., densely, randomly or using a key-point detector). A local visual descriptor, such as the scale-invariant feature transform (SIFT) descriptor and normalized pixel values, is then calculated for each key-point. The resulting distribution of local descriptors is then quantified against a pre-specified visual codebook. The visual codebook converts the distribution of local descriptors into a histogram. The main issues that need to be considered when applying this approach include sampling of the key-points, selection of the type of local descriptor, building the visual

codebook (set of visual words) and assignment of the local descriptors to visual words from the visual codebook.

## 1.1. Motivation

The discriminative power of the visual codebook determines the performance of the annotation. However, the construction of the visual codebook and the assignment of the local descriptors to the visual words from the visual codebook is very often a bottleneck in image annotation (Philbin et al., 2007). This is mainly because  $k$ -means clustering, which is computationally expensive, is the most widely used method for visual codebook construction. The computational cost of the  $k$ -means algorithm is even more pronounced on complex datasets with a large number of images (van de Sande et al., 2010). Moreover,  $k$ -means clustering is a un-supervised learning algorithm and thus does not exploit the information contained in the label annotations.

To resolve these issues, Moosmann et al. (2008) and Uijlings et al. (2009) propose to use supervised tree-based machine learning methods to construct the visual codebook. These methods are able to exploit the image annotations in the context of single-label classification, i.e., for images annotated with only a single label.

\* Corresponding author. Tel.: +389 2 3099 159.

E-mail addresses: [ivica.dimitrovski@finki.ukim.mk](mailto:ivica.dimitrovski@finki.ukim.mk) (I. Dimitrovski), [Dragi.Kocev@ijs.si](mailto:Dragi.Kocev@ijs.si) (D. Kocev), [suzana.loskovska@finki.ukim.mk](mailto:suzana.loskovska@finki.ukim.mk) (S. Loskovska), [Saso.Dzeroski@ijs.si](mailto:Saso.Dzeroski@ijs.si) (S. Džeroski).

This means that the proposed approaches can handle image databases with relatively simple images with single annotations. However, in reality, the images are annotated with multiple labels: An image depicting some street will probably also depict buildings, cars, trees and people.

### 1.2. Contribution of the research

In this paper, we present a method for fast and efficient construction of visual codebooks that is able to use images with multiple annotations/labels. We use predictive clustering trees (PCTs) for multi-label classification (MLC) (Blockeel et al., 1998) to decrease the time needed to construct the visual codebook and, at the same time, to improve its discriminative power. PCTs are capable of annotating an instance with multiple labels simultaneously and thus exploiting the interactions that may occur among the different labels.

Although most visual codebooks are built without using the labels, our approach uses the available annotations and the interactions among them to guide the visual codebook construction process. The PCTs are trained to perform multi-label classification: The visual codebook is then constructed by assigning a distinct visual word to each leaf of the tree. The PCTs can be considered as a data-driven and a semantic approach to visual codebook construction because they rely on the available image annotations (labels).

The main research questions that we are addressing in this manuscript are as follows. First, we investigate whether the proposed method for codebook construction is more efficient and scalable than the literature standard (i.e.,  $k$ -means). Next, we test the discriminative power of the obtained visual codebook and compare it to the discriminative power of the codebook obtained using  $k$ -means. Furthermore, we investigate the influence of the number of selected key-points used to obtain the visual words on the discriminative power of the codebook. Finally, we examine the influence of the number of PCTs for MLC used for codebook construction on the codebook's discriminative power.

### 1.3. Organization of the paper

The remainder of this paper is organized as follows. In Section 2, we present related work. Section 3 introduces predictive clustering trees and their extension for multi-label classification. In Section 4, we explain the experimental setup. The obtained results and a discussion thereof are given in Section 5. Section 6 concludes the paper.

## 2. Related work

Many studies have shown that the bag-of-visual-words approach exhibits an impressive performance for image annotation problems (Everingham et al., 2012; Nowak, 2010; Nowak and Huiskes, 2010). A crucial step in the bag-of-visual-words approach is the codebook construction. The visual codebook can be constructed using unsupervised or supervised machine learning methods.

The unsupervised methods are most widely used by the image annotation community. Typically, the visual codebook is constructed by applying  $k$ -means clustering to the key-points, i.e., the local descriptors (e.g., SIFT) extracted from the images (van de Sande et al., 2010; Lowe, 2004). The  $k$ -means algorithm has two serious limitations when applied to the image annotation problem (Jurie and Triggs, 2005). First, it works with small visual codebooks, i.e., with only thousands of visual words, while many datasets may have tens of thousands of visual words. Second, it constructs more clusters close to the most frequently occurring features. These limitations can be addressed with the hierarchical

$k$ -means (HKM) approach (Nister and Stewenius, 2006) and radius-based clustering (van Gemert et al., 2010).

The supervised machine learning methods for constructing visual codebooks use the label annotations of the images to guide the construction of the visual codebooks. They can use single-label or multi-label methods. With the first type of methods, a visual codebook is constructed for each label separately and then these are aggregated over all possible labels. The second type of methods constructs a single visual codebook valid for all labels. Our approach belongs to the second type of methods.

Uijlings et al. (2009) and Chatfield et al. (2011) have recently published detailed overviews of bag-of-words methods for creation of visual codebooks. These surveys compare several state-of-the-art methods for un-supervised and supervised creation of visual codebooks used in the context of single-label classification or multi-class classification. The comparison of tree-based and  $k$ -means methods for visual codebook construction reveals that the tree-based methods are more efficient than methods based on  $k$ -means. However, the improvement of the computational efficiency comes with the price of decreasing the discriminative power of the codebook (i.e., the classifier using the tree-based codebook produced worse annotations). The methods from the literature have, so far, considered the construction of visual codebooks in the context of single-label classification/annotation or multi-class classification/annotation. Another method, proposed by Wojcikiewicz et al. (2010), constructs a separate codebook for each possible label and then reconciles the several codebooks using agglomerative information bottleneck. The experimental evaluation over a single image database with small number of labels per image showed small increase of performance as compared to classical  $k$ -means clustering. All of these methods, in the process of codebook construction, do not consider the label dependencies, i.e., that if an image is annotated with the label *cloud*, it will probably be also annotated with the label *sky*.

In order to alleviate all of these issues, we propose here a method for constructing a visual codebook for multi-label classification/annotation problems that can explore the existing connections between the labels. This is due to the fact that we construct a single predictive model that is valid for the complete label space. Moreover, decision trees in combination with random forest ensemble methods have not yet been considered for building visual codebooks for a large number of visual concepts (labels), where an image can be annotated with multiple labels (multi-label image annotation problems). To this end, we propose to use random forest of PCTs for MLC, considering that there are many real-life challenging multi-label image annotation problems (Nowak and Dunker, 2010; Nowak, 2010; Nowak and Huiskes, 2010).

## 3. Visual codebook construction using predictive clustering trees

### 3.1. The task of multi-label classification (MLC)

The problem of single-label classification is concerned with learning from examples, where each example  $\mathbf{x} \in \mathcal{X}$  ( $\mathcal{X}$  denotes the domain of descriptive variables for the examples) is associated with a single label  $\lambda_i$  from a finite set of disjoint labels  $\mathcal{L} = \{\lambda_1, \lambda_2, \dots, \lambda_Q\}$ ,  $Q > 1$ . For  $Q > 2$ , the learning problem is referred to as *multi-class classification*. On the other hand, the task of learning a mapping from an example  $\mathbf{x} \in \mathcal{X}$  to a set of labels  $\mathcal{Y} \subseteq \mathcal{L}$  is referred to as a *multi-label classification*. In contrast to multi-class classification, alternative labels in multi-label classification are not assumed to be mutually exclusive: multiple labels may be associated with a single example, i.e., each example can be a member of more than one class. Labels in the set  $\mathcal{Y}$  are relevant




Image	Descriptive variables						Target variables												
	descriptor1	descriptor2	descriptor3	descriptor4	descriptor5	descriptor6	..	landscape	mountains	sky	clouds	plants	trees	flowers	water	lake	river	sea	..
	0.02	0.12	0.06	0.11	0.86	0.17	...	1	0	0	0	1	0	1	0	0	0	0	...
	0.10	0.02	0.34	0.98	0.61	0.01	...	1	1	1	0	1	0	0	0	0	0	0	...
	0.87	0.02	0.82	0.34	0.15	0.08	...	1	0	1	1	0	0	0	1	0	0	1	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Fig. 1. An excerpt from the dataset ImageCLEF@ICPR2010 (Nowak, 2010) for the task of MLC for image annotation. The table contains a set of images with their visual descriptors (descriptive variables) and annotations (labels). The set of possible image labels include: Plants, Flowers, Trees, Sky, Clouds, Water, Lake, River, Sea, Mountains, Day, Night, Sunny, Sunset sunrise, Animals, Food, Vehicle, etc. The complete list of labels is given in the Supplementary material.

Table 1 The top-down induction algorithm for PCTs.

procedure PCT (E) returns tree	procedure BestTest(E)
1: $(t^*, h^*, \mathcal{P}^*) = \text{BestTest}(E)$	1: $(t^*, h^*, \mathcal{P}^*) = (\text{none}, 0, \emptyset)$
2: <b>if</b> $t^* \neq \text{none}$ <b>then</b>	2: <b>for each</b> possible test $t$ <b>do</b>
3: <b>for each</b> $E_i \in \mathcal{P}^*$ <b>do</b>	3: $\mathcal{P} =$ partition induced by $t$ on $E$
4: $tree_i = \text{PCT}(E_i)$	4: $h = \text{Var}(E) - \sum_{E_i \in \mathcal{P}} \frac{ E_i }{ E } \text{Var}(E_i)$
5: <b>return</b> $\text{node}(t^*, \cup_i \{tree_i\})$	5: <b>if</b> $(h > h^*) \wedge \text{Acceptable}(t, \mathcal{P})$ <b>then</b>
6: <b>else</b>	6: $(t^*, h^*, \mathcal{P}^*) = (t, h, \mathcal{P})$
7: <b>return</b> $\text{leaf}(\text{Prototype}(E))$	7: <b>return</b> $(t^*, h^*, \mathcal{P}^*)$

and labels in the set  $\mathcal{L} \setminus \mathcal{Y}$  are irrelevant for a given example. An example dataset that is used for multi-label classification of images is given in Fig. 1.

In other words, multi-label classification/annotation is concerned with learning from examples, where each example is associated with multiple labels (Tsoumakas and Katakis, 2007). These multiple labels belong to a predefined set of labels. In the case of multi-label classification, the goal is to construct a predictive model that will provide a list of relevant labels for a given, previously unseen example.

3.2. Predictive clustering trees for multi-label classification

Predictive clustering trees (PCTs) (Blockeel et al., 1998) generalize decision trees (Breiman et al., 1984) and can be used for a variety of learning tasks, including different types of prediction and clustering. The PCT framework views a decision tree as a hierarchy of clusters: the top-node of a PCT corresponds to one cluster containing all data, which is recursively partitioned into smaller clusters while moving down the tree. The leaves represent the clusters at the lowest level of the hierarchy and each leaf is labeled with its cluster's prototype (prediction).

PCTs can be induced with a standard top-down induction of decision trees (TDIDT) algorithm (Breiman et al., 1984). The algorithm is presented in Table 1. It takes as input a set of examples (E) and outputs a tree. The heuristic (h) that is used for selecting the tests (t) is the reduction in variance caused by partitioning (P) the instances (see line 4 of BestTest procedure in Table 1). By maximizing the variance reduction the cluster homogeneity is maximized and it improves the predictive performance. If no acceptable test can be found (see line 6), that is, if the test does not significantly reduce

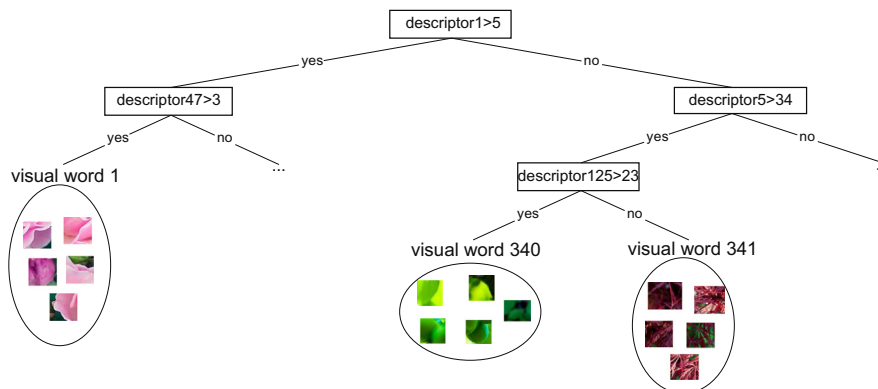


Fig. 2. An example of a predictive clustering tree for MLC constructed using SIFT descriptors of image key-points and the image labels. The PCT is constructed using the dataset presented in Fig. 1. The internal nodes contain tests on the descriptors, while the leaves contain clusters of local key-points and their annotations.



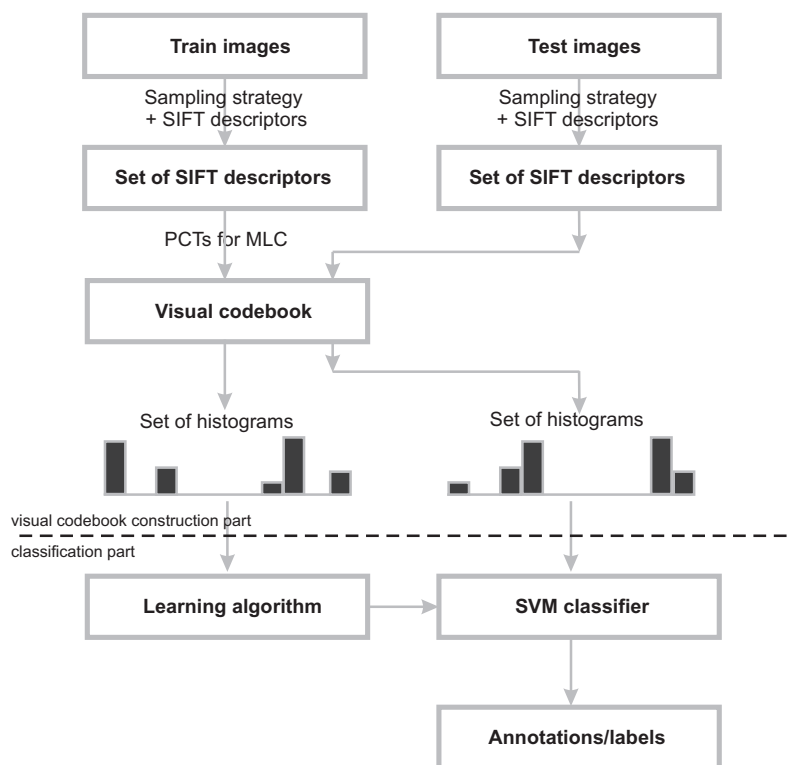


Fig. 3. Architecture of the proposed system for fast and efficient visual codebook construction and multi-label annotation.

the variance, then the algorithm creates a leaf and computes the prototype of the instances belonging to that leaf.

The main difference between the algorithm for learning PCTs and other algorithms for learning decision trees is that the former considers the variance function and the prototype function (that computes a label for each leaf) as parameters that can be instantiated for a given learning task.

In this work, we focus on the task of multi-label classification, which can be considered as a special case of multi-target prediction (Kocev et al., 2013). Therefore, the variance function is computed as the sum of the *Gini indices* (Breiman et al., 1984) of the labels, i.e.,  $Var(E) = \sum_{i=1}^Q Gini(E, \lambda_i)$ ,  $Gini(E, \lambda_i) = 1 - freq_{\lambda_i}^2$ , where  $Q$  is the number of labels,  $\lambda_i$  is a given label and  $freq_{\lambda_i}$  is the frequency of the label  $\lambda_i$ . The prototype function returns a vector of probabilities for the set of labels that indicate whether an example is labeled with a given label.

Fig. 2 presents an example PCT for MLC. The tree is a predictive model which predicts the presence of different visual concepts in an image. The internal nodes contain tests on the descriptors (in this example, SIFT descriptors), while the leaves assign the annotations for a given image. For a detailed description of PCTs for multi-target prediction the reader is referred to Blockeel et al. (1998) and Kocev et al. (2013). The PCT framework is implemented in the `CLUS` system, which is available at <http://clus.sourceforge.net>.

### 3.3. Visual codebook construction

The architecture of the system for fast and efficient visual codebook construction and multi-label annotation is presented in Fig. 3. The system consists of a visual codebook construction part and a classification part. The visual codebook construction part implements the construction of the visual codebook and the image descriptions. The classification part implements the multi-label annotation algorithm based on SVMs.

The visual codebook construction part starts with the generation of the local descriptors for the training images. The SIFT descriptors describe the local shape of a region (i.e., a key-point) using edge orientation histograms (Lowe, 2004). The key-points can be obtained using several approaches: dense sampling, random sampling or using a key-point detector. However, Zhang et al. (2007) have shown that the dense sampling strategy, which samples an image grid in a uniform fashion using a fixed pixel interval between key-points, yields a visual codebook with the best discriminative power. Considering this, we use dense sampling of the key-points with an interval distance of 6 pixels and sample at multiple scales ( $\sigma = 1.2$  and  $\sigma = 2.0$ ). We compute a 128-dimensional SIFT descriptor for each image key-point, because SIFT descriptors are the most widely used local descriptors by the image annotation community. Note that the SIFT descriptors from a given image share the same annotation labels.

Next, the obtained local descriptors for the key-points from the training images are used to create the visual codebook (van de Sande et al., 2010; Lowe, 2004). From all of the key-points of the training images, a smaller subset is randomly selected. Typically, *k*-means clustering is applied to the local descriptors and the resulting clusters represent the visual words. In our system, we use PCTs for MLC to construct the visual codebook. We create the training dataset as follows. Each row represent a image key-point described with the local descriptors (as descriptive variables) and the labels of the image from which the key-point was taken. In this way, the image labels guide the process of the construction of the visual words. Since the variance function of the PCTs includes the informations about all of the labels, the codebook obtained in this way exploits the label interaction for creating a better codebook. To control the size of the visual codebook, we apply pre-pruning of the trees by requiring a given minimum number of instances in each tree leaf. Each leaf of the tree is a separate visual word and all leaves constitute the visual codebook.

The single PCTs are very fast to construct, but they are unstable, i.e., can change substantially for small changes in the data. To

overcome this limitation and to further improve the discriminative power of the visual codebook, we construct a small random forest of PCTs for MLC (Moosmann et al., 2008). Each PCTs is obtained both by bootstrap sampling the training set and by randomly changing the feature set during learning. More precisely, at each node in the decision tree, a random subset of the input attributes is taken, and the best feature is selected from this subset (instead of the set of all attributes). The number of attributes that are retained is given by a function  $f(x)$  of the total number of input attributes  $D$ ,  $f(x) = \lfloor \log_2 D \rfloor + 1$  (note that one can consider also other functions, such as  $f(x) = D$ ,  $f(x) = \sqrt{D}$ ,  $f(x) = \lfloor 0.1 \times D \rfloor$ ) (Breiman, 2001). The final visual codebook is then obtained by concatenating the individual visual codebooks from each PCT in the forest.

We then obtain the image descriptions for the training images. For each image, we sort all of the local SIFT descriptors (for all of the key-points) through the PCTs for MLC and the leaf that it falls into gives the corresponding visual word. The votes for each visual word are then accumulated (over all local descriptors) for each image into a fixed-size histogram (i.e., each visual word is a descriptive variable). The images are thus described with both the histograms of the visual words and the labels.

The produced image descriptions are then used in the image classification part of the system to learn a predictive model (i.e., classifier). In our system, we construct support vector machines (SVMs) with a  $\chi^2$  kernel, which is the image annotation community standard (van de Sande et al., 2010). The predictive model will be used to produce annotations for un-seen (i.e., test) images. For each un-seen image, the key-points are first sampled in a similar way as for the codebook construction part by using dense sampling. Next, local SIFT descriptors for the sampled key-points are calculated. These descriptors are then sorted through the PCTs for MLC (from the codebook construction part) to obtain the image histogram, i.e., image description. Finally, using the image description and the SVM classifier the system produces annotations for the given image.

### 3.4. Computational complexity aspects

We assume that the training set contains  $N$  instances and  $D$  descriptive attributes and  $Q$  is the number of possible image labels. Furthermore,  $I$  is the number of iterations for  $k$ -means and  $w$  is the number of clusters. Kocev et al. (2013) derive in detail the computational complexity of PCTs. Here, we briefly summarize and discuss the complexity of PCTs for MLC.

The overall complexity for constructing a PCT for MLC is  $\mathcal{O}(DN \log^2 N) + \mathcal{O}(QDN \log N) + \mathcal{O}(N \log N)$ . The upper bounds of this complexity is given by the first two terms. If we assume that  $Q > \log N$  (the images are annotated with many labels) then the complexity of the PCTs for MLC is  $\mathcal{O}(QDN \log N)$ . On the other hand, if the number of images is much larger than the number of labels, i.e.,  $\log N > Q$  then the complexity is  $\mathcal{O}(DN \log^2 N)$ . Recall that we use a randomized version of the tree construction algorithm: bootstrap sampling of the training set and by randomly changing the feature set during learning. This means that the training set has less examples  $N'$  (where typically  $N' = 0.632 \times N$  (Breiman, 1996)) and that at each node less variables are used  $D' = f(D)$ . This speeds-up significantly the PCT construction algorithm.

The computational complexity of  $k$ -means clustering for construction of  $w$  clusters is  $\mathcal{O}(wIND)$  (Manning et al., 2008). Let us compare the computational complexity of both methods. In the case where the number of labels is large ( $Q > \log N$ ) or small ( $Q < \log N$ ), the ratio of the computational complexities depends from the ratio  $\frac{Q \log N}{wI}$  or  $\frac{\log^2 N}{wI}$ , respectively. If this ratio is bigger than 1, then PCTs have worse complexity than the  $k$ -means, and, otherwise, if it is smaller than 1. However, note that in real world scenarios  $Q$  and  $\log N$  are much smaller than  $w$ : typically the

images are annotated with at most hundreds labels and the databases contain at most millions of images ( $\log_2 10^6 < 20$ ), while there are typically thousands of visual words in a codebook and depending on the problem at hand the number of iterations can be several tens or even hundreds. If we also consider the speed-up from the randomization of the tree-construction, we can conclude that the PCTs for MLC are much more computationally efficient than the  $k$ -means clustering.

## 4. Experimental design

### 4.1. Experimental questions

The goal of this study is to answer the following questions:

1. Is the proposed method more efficient than the literature standard in terms of:
  - (a) codebook construction?
  - (b) projecting an image into a histogram?
2. Does the use of image labels during the process of visual codebook construction with PCTs for MLC improve the discriminative power of the visual codebook?
3. Is the proposed method scalable for a large number of key-points considered for codebook construction?
4. How does the number of PCTs for MLC influence the discriminative power of the codebook?

For answering questions 1a, 1b and 2, we compare our approach of using PCTs for visual codebook construction to the community standard of using  $k$ -means for the same task. Uijlings et al. (2009) have shown that  $k$ -means produces a visual codebook with best discriminative power (i.e., the classifiers using this codebook produce the most correct annotations). Furthermore, Chatfield et al. (2011) also select  $k$ -means clustering with a combination of histogram encoding as a baseline in their experiments. The other methods suggested in these studies are not applicable in our context because we are only focusing on the partitioning of the local descriptors space and not on the encoding phase (we adopt histogram encoding). More specifically, we compare the execution times of the two methods for visual codebook construction: We measure the time needed to obtain the clusters in  $k$ -means and the time needed to train the PCTs for MLC. Furthermore, we measure the time needed to obtain the final histograms which are used for classification. Finally, we compare the predictive performance of the visual codebook constructed using PCTs for MLC with the performance of the one constructed using the  $k$ -means algorithm.

Next, we address question 3 by constructing different visual codebooks by varying the number of key-points considered for codebook construction. We construct these codebooks just for the proposed method. Performing experiments of such scale using  $k$ -means will require much more computational time.

Finally, we answer question 4 by varying the number of PCTs for MLC used to obtain the visual codebook. In order to preserve the same size of the visual codebook (4000 visual words), we use different values for the pre-pruning of the trees. More specifically, as the number of trees grows, the size of the trees reduces in order to keep the number of leaves stable.

### 4.2. Image databases

We evaluate the proposed method for visual codebook construction on three challenging image databases: PASCALVOC2007 (Everingham et al., 2012), ImageCLEF@ICPR2010 (Nowak, 2010) and ImageCLEF2010 (Nowak and Huiskes, 2010). These benchmark databases have approximately equal number of training images,

**Table 2**

Properties of the image databases for evaluation of the image annotation algorithms.

Image database	#Train images	#Test images	#Labels	#Labels per image
PASCAL VOC 2007	5011	4952	20	1.46
ImageCLEF@ICPR 2010	5000	3000	53	8.68
ImageCLEF 2010	5000	3000	93	12.06

but different number of testing images, number of possible labels and labels per image. These databases are briefly summarized in Table 2.

#### 4.3. Visual features and visual codebook construction

We extract key-points from the images using dense sampling at a spatial pyramid configuration of  $1 \times 1$ . Note that the proposed method can be easily applied to other spatial pyramid configurations and thus further increase the predictive power (Lazebnik et al., 2006). We then calculate 128 dimensional SIFT descriptors for each key-point.

For the  $k$ -means algorithm, we are clustering the selected 300000 SIFT descriptors into 4000 clusters, thus obtaining a visual codebook with 4000 words (Nowak and Dunker, 2010; Everingham et al., 2012; van de Sande et al., 2010; Uijlings et al., 2009). In order to directly compare this to our method, we use the same 300,000 descriptors to construct the codebook using PCTs for MLC. Moreover, Moosmann et al. (2008) show that a small ensemble with 4 trees gives good performance, hence, we also construct a forest with 4 PCTs. To limit the number of words, in this case, we pruned the trees requiring a minimum of around 200 key-points per leaf. With this we ensure that the visual codebook has approximately 4000 words. Note that we first perform the experiments with our method and then use the same number of words for the  $k$ -means algorithm. For example, the random forest of PCTs for MLC constructs a visual codebook with 4070 words for the ImageCLEF 2010 database, thus the number of clusters for  $k$ -means was set to 4070. To further explore the power of the proposed method, we perform two additional experiments by varying the number of key-points used to construct the codebook and the number of constructed PCTs for MLC.

#### 4.4. Classifier setup

Classification (in this case predicting image labels) is an independent step in automatic image annotation. Since our goal is to evaluate the visual codebook construction, we use SVMs with  $\chi^2$  kernel as classifiers for all of the visual codebooks. We used the libSVM implementation of SVMs (Chang and Lin, 2001) with probabilistic output (Lin et al., 2007) to train the classifier.

To solve the multi-label classification problems, we employ the *one-vs-all* approach. Namely, we build a binary classifier for each visual concept: the examples associated with that visual concept are labeled positive and the remaining examples are labeled negative. This results in an imbalanced ratio of positive versus negative training examples. We resolve this issue by adjusting the weights of the positive and negative class (van de Sande et al., 2010). We also optimize the cost parameter  $C$  of the SVMs using an automated parameter search procedure (van de Sande et al., 2010). For the parameter optimization, we separate 20% of the training set and use it as validation set. After finding the optimal  $C$  value, the SVM is trained on the whole set of training images.

#### 4.5. Evaluation measure

The most widely used performance evaluation measure in image annotation is the *mean average precision* (MAP) (Everingham

et al., 2012; Nowak and Huiskes, 2010). Considering this, we adopt MAP to evaluate the performance of the annotation algorithms and thus the discriminative power of the visual codebooks. The MAP is calculated as follows. First, for each image label the precision-recall curve is computed by monotonically decreasing the value of the precision and setting the precision for a given recall value  $r$  to the maximum precision obtained for any recall value  $r' \geq r$ . Next, the average precision is computed as the area under this curve by numerical integration. There is no need for an approximation since the curve is piecewise constant. Finally, the MAP value is obtained by taking the mean value of the average precisions of all labels. A more detailed description and the source code for computing the MAP value can be found in Everingham et al. (2012).

## 5. Results and discussion

### 5.1. Efficiency of the codebook construction

We measure the efficiency of both method in terms of time needed to construct the codebook and time needed to project the images into a histogram. Table 3 presents the time needed to construct the visual codebook using  $k$ -means and random forests of PCTs for MLC. The results show that random forests of PCTs are creating the visual codebook consistently faster than the  $k$ -means algorithm: The average speed-up ratio is  $\sim 24.4$  times.

Next, we consider the time needed to project an image into a histogram. The  $k$ -means algorithm on average requires 0.850 s to produce a histogram for an image, while the random forest of PCTs needs only 0.021 s. In the former case, the projection of an image is actually a nearest neighbor search in the space of image descriptions and is much slower than a pass through the trees in the random forest, which is applied in the latter case. The average speed-up ratio is  $\sim 40.5$  times. In conclusion, these results confirm our first hypothesis that random forests of PCTs for MLC construct the visual codebook very efficiently and thus can be successfully scaled to larger problems.

The efficiency of the SVM classifiers does not directly depend of the codebook itself. Namely, the computational complexity of the SVMs mainly depends on the number of training examples (i.e., training images) (Chang and Lin, 2001). Since both methods use the same training images and the size of the image descriptors is the same (4000 visual words), the time needed to construct the SVM in approximately the same.

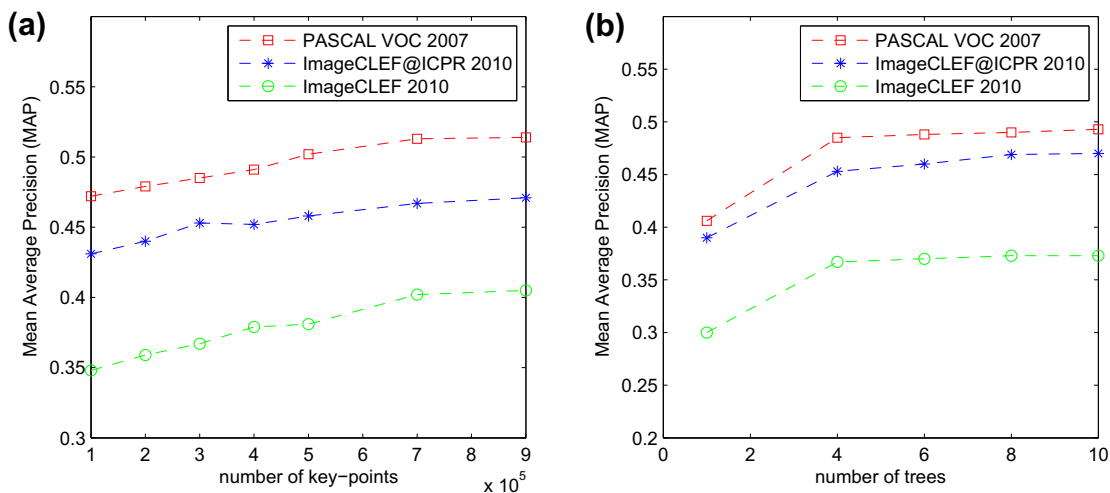
### 5.2. Discriminative power of the codebook

We discuss the predictive performance of the SVM classifiers obtained using the two visual codebooks. The inspection of the re-

**Table 3**

Comparison of the efficiency and the performance of the visual codebooks. The efficiency is measured as time needed to construct the codebook and the performance as MAP.

Image database	Efficiency [s]		Performance [MAP]	
	$k$ -Means	PCTs for MLC	$k$ -Means	PCTs for MLC
PASCAL VOC 2007	12334.820	456.114	0.477	0.485
ImageCLEF@ICPR 2010	11977.230	466.829	0.425	0.453
ImageCLEF 2010	11209.750	544.740	0.329	0.367



**Fig. 4.** Performance of the classifiers (mean average precision) from the codebooks constructed with: (a) different number of key-points and 4000 visual words and (b) different number of PCTs for MLC and 4000 visual words.

sults for mean average precision given in Table 3 reveals two major conclusions. First, the visual codebook constructed with random forests of PCTs for MLC outperforms the one constructed with  $k$ -means clustering on all three databases. Second, the improvement is larger for the databases with a larger average number of labels per image. The latter means that our approach makes use of the interactions and the connections that exist between the labels, thus resulting in a visual codebook with better discriminative power.

The average precisions for each label in the PASCALVOC2007, the ImageCLEF@ICPR2010 and the ImageCLEF2010 database are given in the Supplementary material. To begin with, our approach increases the average precision for 70% of the labels from the PASCALVOC2007 database by 0.017 and decreases the average precision of the remaining 30% of the labels by 0.015. Next, for the ImageCLEF@ICPR2010 database, our approach increases the average precision for 90.6% of the labels by 0.033 and decreases the average precision for the remaining labels by 0.017. Finally, for the ImageCLEF2010 database, our approach increases the average precision for 91.4% of the labels by 0.044 and decreases the average precision for the remaining labels by 0.017. In summary, the more labels per image a database has, the proposed approach increases the average precision for a larger portion of the labels by a larger value.

### 5.3. Scalability of the codebook construction

In this section, we discuss the scalability of the proposed approach and its practical implications. The results show that our approach is much more efficient and has better discriminative power than  $k$ -means clustering. This means that we may now construct visual codebooks selecting a larger number of image key-points. Namely, the number of images in the databases and the number of possible labels are constantly growing, thus selecting around 300,000 key-points and 4000 words may not be sufficient to capture the diversity that exists in the large databases. Construction of larger visual codebook with  $k$ -means is computationally expensive or even impossible for extreme values for the number of key-points. On the other hand, our approach can easily produce larger codebook using larger portions of the key-points that will have better discriminative power (Philbin et al., 2007).

To test this hypothesis, we constructed different codebooks by varying the number of key-points that are considered for learning

the PCTs for MLC. Note that the size of the codebook remains the same as for the previous experiments, i.e., 4000 words. We present the results in Fig. 4(a). The results show that including more key-points in the process of codebook construction increases the discriminative power of the codebook (i.e., the predictive power of the subsequently constructed classifiers) for the three image databases. The biggest performance increase is for the ImageCLEF2010 database: From 0.348 for 100,000 key-points to 0.405 for 900,000 key-points, i.e., a relative increase of  $\sim 16.4\%$ . The second largest performance increase is for the ImageCLEF@ICPR2010 database: From 0.431 for 100,000 to 0.471 for 900,000 key-points, i.e.,  $\sim 9.3\%$ . Finally, the performance increase for the PASCALVOC2007 database is from 0.472 for 100,000 to 0.514 for 900,000 key-points, i.e.,  $\sim 8.9\%$ . All in all, the inclusion of more key-points in the process of visual codebook construction increases the discriminative power of the codebook. Note that largest increases are obtained for the datasets with the largest numbers of labels: It seems that the performance increase is related to the increase of the number of labels per image.

### 5.4. Complexity of the model for codebook construction

We investigate the influence of the number of PCTs for MLC on the discriminative power of the codebook. Moosmann et al. (2008) show that 4 trees already give a satisfactory predictive performance. We present results in Fig. 4(b) using different number of trees to create the codebook. Note that the codebook size was kept at 4000, as in all other experiments. This was achieved by limiting the tree growth. The results show that the largest performance increase occurs when the number of trees increases from 1 to 4. Using more than 4 trees does not change significantly the discriminative power of the codebook.

## 6. Conclusion

In this paper, we presented a method for fast and efficient construction of visual codebooks for image annotation. The construction of a codebook is an essential part of the bag-of-visual-words approach to image annotation. It should thus be efficient and deliver a codebook with high discriminative power. However, the construction of a visual codebook is a bottleneck in the bag-of-visual-words approach, because it typically uses  $k$ -means clustering over several hundreds of thousands of image key-points to obtain



several thousands visual words. Existing methods are able to solve these issues only partially: When using trees for constructing the visual codebook, they sacrifice some of the discriminative power of the codebook to construct it more efficiently. Moreover, existing tree-based methods are applicable only in the context of single-label and multi-class classification. In this paper, we proposed to use predictive clustering trees (PCTs) for multi-label classification (MLC). In this way, we efficiently construct visual codebooks and increase the discriminative power of the codebook.

We evaluated the proposed method on three challenging image databases: PASCALVOC2007, ImageCLEF@ICPR2010 and ImageCLEF2010. Each of the selected image databases contains a large number of images and a varying number of labels and labels per image. We compare the visual codebooks resulting from our method against the visual codebooks obtained with  $k$ -means clustering. In our experimental evaluation, we used SVMs with  $\chi^2$  kernel as classifiers – the most widely used classifiers in the image annotation community.

The visual codebooks are compared by their efficiency and the discriminative power they offer to the classifier. The experimental results measuring the efficiency show that both constructing a visual codebook and projecting an image into a histogram with a random forest of PCTs for MLC is much faster than with  $k$ -means clustering. The discriminative power of the codebooks was assessed through the predictive power of the SVM classifiers: We measured the mean average precision for the three databases. The results show that our method consistently outperforms  $k$ -means clustering. Moreover, the difference in predictive performance increases with the number of average labels per image. This means that our method exploits the connections between the labels in the database and constructs a visual codebook with better discriminative power than  $k$ -means clustering.

To further explore the scalability of the proposed method, we used it to construct visual codebooks with a much larger number of key-points than usually considered. The results revealed that using more key-points yielded codebooks with even better discriminative power. All in all, the proposed method is more efficient and offers a better visual codebook to the classifier as compared to the current community standard.

### Acknowledgements

We would like to acknowledge the support of the European Commission through the project MAESTRA - Learning from Massive, Incompletely annotated, and Structured Data (Grant number ICT-2013-612944).

### Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at <http://dx.doi.org/10.1016/j.patrec.2013.10.016>.

### References

- Blockeel, H., Raedt, L.D., Ramong, J., 1998. Top-down induction of clustering trees. In: International Conference on Machine Learning. Morgan Kaufmann, pp. 55–63.
- Breiman, L., 1996. Bagging predictors. *Machine Learning* 24 (2), 123–140.
- Breiman, L., 2001. Random forests. *Machine Learning* 45 (1), 5–32.
- Breiman, L., Friedman, J., Olshen, R., Stone, C.J., 1984. *Classification and Regression Trees*. Chapman, Hall/CRC.
- Chang, C.-C., Lin, C.-J., 2001. LIBSVM: a library for support vector machines. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.
- Chatfield, K., Lempitsky, V., Vedaldi, A., Zisserman, A., 2011. The devil is in the details: an evaluation of recent feature encoding methods. In: British Machine Vision Conference, pp. 76.1–76.12.
- Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A., 2012. The PASCAL visual object classes challenge (VOC2012) results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
- Jurie, F., Triggs, B., 2005. Creating efficient codebooks for visual recognition. In: International Conference on Computer Vision, pp. 604–610.
- Kocev, D., Vens, C., Struyf, J., Džeroski, S., 2013. Tree ensembles for predicting structured outputs. *Pattern Recognition* 46 (3), 817–833.
- Lazebnik, S., Schmid, C., Ponce, J., 2006. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: IEEE conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2169–2178.
- Lin, H.-T., Lin, C.-J., Weng, R.C., 2007. A note on Platt's probabilistic outputs for support vector machines. *Machine Learning* 68, 267–276.
- Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60 (2), 91–110.
- Manning, C.D., Raghavan, P., Schütze, H., 2008. *Introduction to Information Retrieval*. Cambridge University Press.
- Moosmann, F., Nowak, E., Jurie, F., 2008. Randomized clustering forests for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30 (9), 1632–1646.
- Nister, D., Stewenius, H., 2006. Scalable recognition with a vocabulary tree. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2161–2168.
- Nowak, S., 2010. ImageCLEF@ICPR contest: challenges, methodologies and results of the photo annotation task. In: International Conference on Pattern Recognition, pp. 489–492.
- Nowak, S., Dunker, P., 2010. Overview of the CLEF2009 large-scale visual concept detection and annotation task. In: Multilingual Information Access Evaluation II. Multimedia Experiments, 10th Workshop of the Cross-Language Evaluation Forum, pp. 94–109.
- Nowak, S., Huiskes, M.J., 2010. New strategies for image annotation: overview of the photo annotation task at ImageCLEF 2010. In: CLEF (Notebook Papers/LABs/Workshops).
- Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A., 2007. Object retrieval with large vocabularies and fast spatial matching. In: IEEE conference on Computer Vision and Pattern Recognition, pp. 1–8.
- Tsoumakas, G., Katakis, I., 2007. Multi-label classification: an overview. *International Journal of Data Warehousing and Mining* 3 (3), 1–13.
- Uijlings, J., Smeulders, A., Scha, R., 2009. Real-time bag of words, approximately. In: ACM International Conference on Image and Video Retrieval, pp. 1–8.
- van de Sande, K.E.A., Gevers, T., Snoek, C.G.M., 2010. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32 (9), 1582–1596.
- van Gemert, J.C., Snoek, C.G.M., Veenman, C.J., Smeulders, A.W.M., Geusebroek, J.M., 2010. Comparing compact codebooks for visual categorization. *Computer Vision and Image Understanding* 114 (4), 450–462.
- Wojcikiewicz, W., Binder, A., Kawanabe, M., 2010. Shrinking large visual vocabularies using multi-label agglomerative information bottleneck. In: 17th IEEE International Conference on Image Processing, pp. 3849–3852.
- Zhang, J., Marszałek, M., Lazebnik, S., Schmid, C., 2007. Local features and kernels for classification of texture and object categories: a comprehensive study. *International Journal of Computer Vision* 73 (2), 213–238.