Mach Learn (2016) 105:155–198
DOI 10.1007/s10994-016-5560-1

An empirical study on image bag generators for multi-instance learning

Xiu-Shen Wei[1] · **Zhi-Hua Zhou**[1]

Received: 11 July 2014 / Accepted: 24 February 2016 / Published online: 22 March 2016

Abstract Multi-instance learning (MIL) has been widely used on diverse applications involving complicated data objects such as images, where people use a bag generator to represent an original data object as a *bag* of instances, and then employ MIL algorithms. Many powerful MIL algorithms have been developed during the past decades, but the bag generators have rarely been studied although they affect the performance seriously. Considering that MIL has been found particularly useful in image tasks, in this paper, we empirically study the utility of nine state-of-the-art image bag generators in the literature, i.e., *Row*, *SB*, *SBN*, *k-meansSeg*, *Blobworld*, *WavSeg*, *JSEG-bag*, *LBP* and *SIFT*. From the 6923 (9 bag generators, 7 learning algorithms, 4 patch sizes and 43 data sets) configurations of experiments we make two significant new observations: (1) Bag generators with a dense sampling strategy perform better than those with other strategies; (2) The standard MIL assumption of learning algorithms is not suitable for image classification tasks.

Keywords Multi-instance learning · Bag generator · Empirical study · Image bag generators

1 Introduction

In investigating the problem of drug activity prediction, Dietterich et al. (1997) proposed the notion of multi-instance learning (MIL). Contrasting to traditional single-instance learning, the multi-instance representation enables the learning process to exploit some inherent structure information in input patterns. Specifically, MIL receives a set of bags that are labeled

Editor: Stephen D. Scott.

✉ Zhi-Hua Zhou
zhouzh@nju.edu.cn; zhouzh@lamda.nju.edu.cn

Xiu-Shen Wei
weixs@lamda.nju.edu.cn

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

positive or negative, rather than receiving a set of instances which have positive or negative labels. In addition, instances in MIL bags have no label information. The goal of MIL is to learn a classifier from the training bags such that it can correctly predict the labels of unseen bags.

With the rapid development of MIL, it has been widely used on diverse applications, especially the tasks involving complicated data objects, such as image categorization (Chen et al. 2006; Song et al. 2013), image annotation (Tang et al. 2010; Zhu and Tan 2011), image retrieval (Zhang et al. 2002; Zhou et al. 2003; Xu and Shih 2012), medical image diagnosis (Fung et al. 2007), face detection (Zhang and Viola 2008), text categorization (Andrews et al. 2003) and so on. Particularly, when we tackle the image tasks, an image can be naturally partitioned into several semantic regions; each region is represented as a feature vector (an instance). Consequently, MIL solutions have been recognized as state-of-the-art image categorization/annotation methods, particularly for region-based image categorization/annotation (Chen and Wang 2004; Yang et al. 2006; Vijayanarasimhan and Grauman 2008).

In practice, users first use a bag generator to represent an original data object as a bag of instances, and then apply MIL algorithms. It is noteworthy that the bag generators are different from the feature extraction process; that is, a bag generator decides how an image will be represented by a set of patches, whereas a feature extraction process decides how each patch is characterized by a feature vector. As there are many different ways for representing one data object into multiple instances, bag generators are crucial to MIL learning performances. However, the evaluation of different bag generators has rarely been studied, although many effective MIL learning algorithms have been developed during the past decades (representative examples include EM-DD (Zhang and Goldman 2000), Citation-*k*NN (Wang and Zucker 2000), RIPPER-MI (Chevaleyre and Zucker 2001), miSVM (Andrews et al. 2003), MIBoosting (Xu and Frank 2004), miGraph (Zhou et al. 2009), MILES (Chen et al. 2006), MIForests (Leistner et al. 2010), etc.).

In this paper, we focus on image categorization tasks and empirically investigate the properties of nine popular image bag generators, i.e., *Row*, *SB*, *SBN* (Maron and Ratan 2001), *Blobworld* (Carson et al. 2002), *k-meansSeg* (Zhang et al. 2002), *WavSeg* (Zhang et al. 2004), *JSEG-bag* (Liu et al. 2008), *LBP* (Ojala et al. 2002) and *SIFT* (Lowe 2004). Note that here we are studying what kind of bag generators are suitable to MIL algorithms, rather than studying general image classification approaches or novel image feature representations. Readers interested in those topics can refer to Felzenszwalb et al. (2010), Chatfield et al. (2011) and Nguyen et al. (2009).

Given an image for bag generators, they first separate the image into a number of regions, and then represent a region as an instance. Thus, by setting different patch sizes, a bag generator can obtain multiple bags with different number of instances for the same image. To examine the impact of bag generators on the classification performances with different learning algorithms, we employ seven state-of-the-art MIL methods as the test bed, including Citation-*k*NN (Wang and Zucker 2000), miSVM (Andrews et al. 2003), MIBoosting (Xu and Frank 2004), miGraph (Zhou et al. 2009), MILES (Chen et al. 2006), miFV (Wei et al. 2014) and MIForests (Leistner et al. 2010). Forty-three data sets with diverse target concepts are created from the *COREL* and *MSRA* image data sources. In all, by combining nine bag generators (five of them with different patch sizes, i.e., Row, SB, SBN, *k*-meansSeg and JSEG-bag), seven learning algorithms and forty-three data sets, we set up an extensive empirical study with 6923 configurations. From the experimental results, we have some important observations. Specifically, some bag generators (i.e., SB, SBN and LBP) with the dense sampling strategy will outperform other generators in most cases, which is consistent

with the conclusion in computer vision (Li and Perona 2005; Nowak et al. 2006); miGraph, MIBoosting and miFV stress the relationship between instances in MIL bags, which do not adopt the standard MIL assumption (i.e., the bags are labeled positive in the way that if a bag contains at least one positive instance, otherwise it is labeled as negative), thus these learning algorithms could achieve better classification accuracy rates (cf. Table 5). Note that these two important observations have not been made before. Moreover, we analyze the utilities of these bag generators for different kinds of image classification tasks, i.e., scene classification and object classification. In addition, we also have some interesting findings about learning algorithms, and recommend several combinations of learning algorithm and bag generator for practical applications. In short, these observations, on one hand, give practical suggestions for bag generator selections with diverse needs, and on the other hand, they are insightful on designing better bag generators or MIL algorithms for image related tasks.

The rest of this paper is organized as follows. In Sect. 2, we briefly introduce multi-instance learning and the related works. In Sect. 3, we introduce some image bag generators, especially the ones which will be empirically studied later in this paper. We then give information about our empirical configurations in Sect. 4, including the learning algorithms, data sets and evaluation criteria used in our study. In Sect. 5, we present our empirical results. Finally, we summarize our main observations in Sect. 6 and conclude the paper.

2 Background

In the middle of the 1990s, Dietterich et al. (1997) investigated the problem of drug activity prediction. The goal was to use a model to predict whether a new molecule can be qualified to make some drug, through analyzing a set of known molecules. The difficulty of this problem was that, each molecule may have a wide range of possible types of low-energy shapes, but biochemistry experts at that time only knew which molecules were qualified to make drug, instead of knowing which special shapes played a decisive role.

In order to solve this problem, Dietterich et al. regarded each molecule as a *bag*, and regarded each kind of low-energy shapes of one molecule as an *instance* in its corresponding bag, thereby formulating *multi-instance learning*.

Formally, let $\mathcal{X}$ denote the instance space and $\mathcal{Y}$ the set of class labels. The task of multi-instance learning is to learn a function $f : 2^{\mathcal{X}} \to \{-1, +1\}$ from a given data set $\{(\mathbf{X}_1, y_1), (\mathbf{X}_2, y_2), \ldots, (\mathbf{X}_m, y_m)\}$, where $\mathbf{X}_i \subseteq \mathcal{X}$ is a set of instances $\{\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \ldots, \mathbf{x}_{n_i}^{(i)}\}$, $\mathbf{x}_j^{(i)} \in \mathcal{X} (j \in \{1, \ldots, n_i\})$, and $y_i \in \{-1, +1\}$ is the known label of X_i. In contrast, the task of traditional supervised learning is to learn a function $f : \mathcal{X} \to \mathcal{Y}$ from a given data set $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$, where $\mathbf{x}_i \in \mathcal{X}$ is an instance and $y_i \in \mathcal{Y}$ is the known label of $\mathbf{x}_i$.

Far beyond drug activity prediction, the multi-instance problem emerges naturally in a variety of challenging learning problems in image related tasks, including natural scene classification (Maron and Ratan 2001), image categorization/classification (Chen et al. 2006; Song et al. 2013), image annotation (Tang et al. 2010; Zhu and Tan 2011) and image retrieval (Zhang et al. 2002; Zhou et al. 2003; Xu and Shih 2012). In addition, MIL techniques have already been used on diverse applications, for example face detection (Zhang and Viola 2008; Viola et al. 2006), text categorization (Andrews et al. 2003; Settles et al. 2008), web mining (Zhou et al. 2005) and so on.

In the applications of multi-instance learning, Maron and Ratan (2001) were the first to apply the MIL framework to image tasks. In their work, several bag generators (i.e., Row, SB, SBN) for transforming images into image bags were presented and tested, and more importantly, they demonstrated that bag generators play a key role in developing a practical CBIR system based on multi-instance learning and significantly affect the performance of the retrieval.

After that, several image bag generators have been proposed during the past decade. However, very little work has been done on the evaluation of image bag generators. Even worse, it is usually the case that researchers used a new bag generator without comparing to existing bag generators (e.g., Carson et al. (2002) and Zhang et al. (2004)).

Zhou et al. (2003) compared the performances of several different bag generators when they proposed the ImaBag image bag generator. In their experiments, they compared ImaBag with Maron and Ratan's SBN (Maron and Ratan 2001) and Yang and Lozano-Pérez's bag generator (Yang and Lozano-Pérez 2000), yet merely by employing the Diverse Density (Maron and Lozano-Pérez 1998) algorithm. Later they showed that the performances of ImaBag were worse than that of SBN, but were much better than that of Yang and Lozano-Pérez's method. Additionally, Zhang et al. (2002) studied the performances of EM-DD (Zhang and Goldman 2000) across different image processing techniques based on SBN (Maron and Ratan 2001) and their k-meansSeg bag generator. However, they only reported k-meansSeg outperformed SBN in some cases. Compared to the previous work, we perform a very large number of experiments to study the utilities of nine state-of-the-art image bag generators and present an exhaustive evaluation of these image bag generators.

3 Image bag generators

Image bag generators extract information from an original image and then construct a set of instances which is regarded as an MIL bag. Depending on whether bag generators can distinguish the semantic components from images, image bag generators can be divided into two categories, i.e., *non-segmentation bag generators* and *segmentation bag generators*. Non-segmentation bag generators adopt a fixed strategy which is independent of image structures to extract instances from images. While segmentation bag generators try to segment an image into multiple semantic components, and construct MIL bags by using one instance to represent one corresponding semantic component.

In this paper, seven staple image bag generators are studied, including the simple method Row (Maron and Ratan 2001), the original image non-segmentation based methods SB and SBN (Maron and Ratan 2001), and the transformation based methods Blobworld (Carson et al. 2002), k-meansSeg (Zhang et al. 2002), WavSeg (Zhang et al. 2004) and JSEG-bag (Liu et al. 2008). Among which, *Row*, *SB* and *SBN* are non-segmentation bag generators, and *Blobworld*, *k-meansSeg*, *WavSeg* and *JSEG-bag* are segmentation bag generators. In addition, some local descriptors[1] in computer vision have been frequently applied to generate bags for MIL in recent years. Therefore, we employ two famous local descriptors, i.e., local binary patterns (LBP) (Ojala et al. 2002) and scale invariant feature transform (SIFT) (Lowe 2004), as bag generators to extract sets of features from images. Detailed descriptions of these

[1] Local descriptors are used in computer vision to represent interest regions' different characteristics of appearance or shape. They are distinctive, robust to occlusion, and do not require segmentation, which has proven to be very successful in applications, e.g., image classification, image retrieval, object recognition and texture recognition.

methods will be shown in the next subsections, followed by a brief introduction to some other bag generators and finally an abbreviated conclusion of these image bag generators.

3.1 Row, SB and SBN

Maron and Ratan (2001) proposed five bag generators with the same preprocessing steps. The most popular three among them are Row, SB and SBN which all work in the RGB color space.

Row. For an 8×8 filtered image, as Fig. 1a demonstrates, the bag is constructed as following: for each row, one instance is constructed by the mean color of this row and the mean color difference in the rows above and below it.

The *SB* bag generator is short for *Single Blob with no neighbors*. As shown in Fig. 1b, each instance is a 2×2 sized blob of the original image. Note that there is no overlapping between pairwise blobs.

The *SBN* bag generator is short for *Single Blob with Neighbors*, which takes the relationship between neighboring blobs into account. Each instance is constructed as the mean color value of a 2×2 blob and the color difference with its four neighboring blobs. See Fig. 1c.

Note that the key difference between SBN and SB lies in that SBN has overlapping blobs. Figure 2 is an example of the bag generating process of SBN for an 8×8 image. Here each

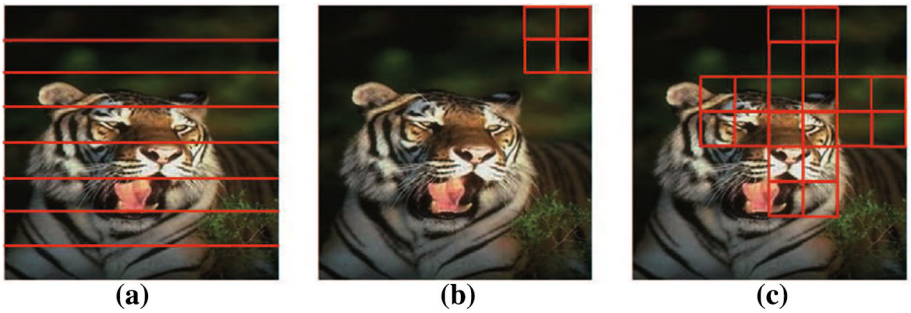

Fig. 1 The examples of the instances abstracting process of Row, SB and SBN bag generators. The original image is collected from the *Tiger* data set which is a sub-category in the *COREL* image data source (Color figure online)

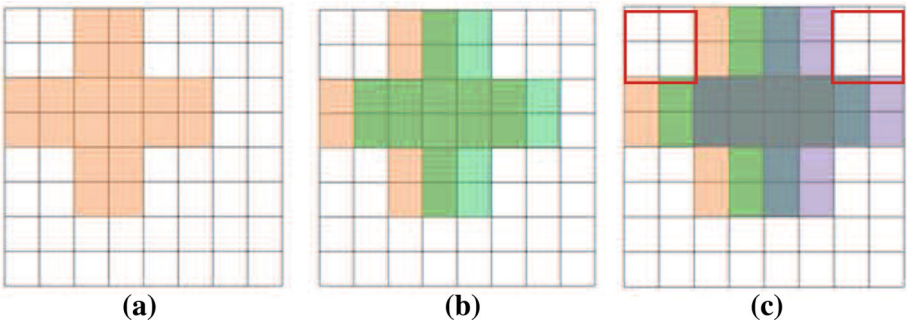

Fig. 2 The instances abstracting process of the SBN bag generator. The sliding window is *cross-shaped* shown in (**a**). From **a**–**c**, on each step the sliding window moves one pixel and abstracts one instance in current positions. The figures are best viewed in *color* (Color figure online)

Fig. 3 The stages of Blobworld processing: from pixels to region descriptions

blob is a 2×2 image patch and the sliding window is cross-shaped as presented in Fig. 2a. During the bag generation process, when the sliding window moves one pixel from left to right, SBN abstracts one instance from the image in the current position at that time. Thus, as shown from Fig. 2a–c, the abstracted instances overlap with each other.

Additionally, there also exists another difference between SB and SBN. SB traverses the whole image with no blind zones, while SBN produces blind zones on the corners when sliding window moves to the edge of the image, which could contribute nothing for image representation. In Fig. 2c, the northeast and northwest blobs highlighted with red rectangles are two blind zones.

3.2 Blobworld

The process of Blobworld (Carson et al. 2002) is as follows. First, Blobworld extracts each image pixel's color feature which has a three-dimensional color descriptor in the L*a*b* color space.[2] Second, it extracts texture features from the grayscale images, which are to get the anisotropy, the contrast and the polarity. So far the color/texture descriptor for a given pixel consists of six values: Three for color and three for texture. In the third step, we append the (x, y) position of the pixel to the previous feature vector. After obtaining the pixel features of 8-dimension, Blobworld groups pixels into regions by modeling the distribution of pixel features with a mixture of Gaussians. In order to divide these pixels into groups, it uses the Expectation-Maximization (EM) algorithm to estimate the maximum likelihood parameters of this mixture of K Gaussian components. Finally, Blobworld describes the color distribution and texture of each region for the MIL algorithms, i.e., the representation of each region in an image is one instance in a bag. The stages of Blobworld processing are depicted in Fig. 3.

3.3 *k*-meansSeg

Zhang et al. (2002) proposed the k-meansSeg bag generator method when they studied content-based image retrieval. In k-meansSeg, images are executed in the YCbCr color space[3] without any preprocessing. It defines a 4×4 image patch as a blob, represented by a six-dimensional vector. The first three dimensions are the mean values of three color components of these 16 (4×4) pixels, and the latter three dimensions are composed by three sub-bands, i.e., *HL*, *LH* and *HH*, which are obtained by the *Daubechies-4* wavelet transformation on the luminance (Y) component. Thus, the blobs of an original image can be expressed as:

$$blobs = \{\langle Y_i, Cb_i, Cr_i, HL(Y)_i, LH(Y)_i, HH(Y)_i\rangle \mid i = 1, 2, \ldots, n\}$$

where n is the number of 4×4 blobs.

[2] A L*a*b* color space is a color-opponent space with dimension L* for lightness and a* and b* for the color-opponent dimensions. One of the most important attributes of the L*a*b*-model is device independence. This means that the colors are defined independent of their nature of creation or the device they are displayed on.

[3] The YCbCr color space is a family of color spaces used as a part of the color image pipeline in video and digital photography systems. It is not an absolute color space; rather, it is a way of encoding RGB information. "Y" is the luminance component and "Cb" and "Cr" are the blue-difference and red-difference chroma components, which are used in diverse applications, such as the JPEG and MPEG format.

After that, the k-means segmentation algorithm is employed on these six-dimensional vectors to segment the image into K segments, and thus one segment is corresponding to one instance. The unknown parameter K is set as 2 at the beginning of this method, then increases by cycling, and terminates until its stop conditions (Zhang et al. 2002). Finally, the ith instance in bags is obtained by averaging all the six-dimensional vectors representing all blobs in the ith segment:

$$bag = \Big\{\langle mean(Y_{ij}), mean(Cb_{ij}), mean(Cr_{ij}), \\ mean(HL(Y)_{ij}), mean(LH(Y)_{ij}), mean(HH(Y)_{ij})\rangle \mid i = 1, 2, \ldots, K\Big\}$$

where K is the number of segments and j's are all blobs of the ith segment.

3.4 WavSeg

Zhang et al. (2004) proposed the WavSeg bag generator to automatically construct multiple instances (regions) in MIL bags (images). WavSeg mainly involves the wavelet analysis and the Simultaneous Partition and Class Parameter Estimation (SPCPE) algorithm (Chen et al. 2000). In the first step, the images are preprocessed by the *Daubechies-1* wavelet transform. After the wavelet transformation, the high-frequency components will disappear in larger scale subbands and therefore, possible regions will be clearly evident. Then by grouping the salient points from each channel, an initial coarse partition can be obtained and passed as the input to the SPCPE segmentation algorithm. In Zhang et al. (2004), they showed that the wavelet transform could lead to better segmentation results, and additionally, it can produce other useful features such as texture features. In the following, WavSeg extracts both the local color and local texture features for each image region. When extracting the color features, they quantize the color space by using color categorization based on HSV value ranges (totally 13 representative colors for these ranges) of the HSV color space.[4] For the regions' texture features, the Daubechies-1 transform could generate three corresponding images in three frequency bands (i.e., HL, LH and HH) of the original image. For the wavelet coefficients in each of the above three bands, the mean and variance values are collected respectively. Therefore, the total six texture features are generated for each image region. The form of the bag generated by WavSeg is as follows:

$$bag = \{\langle hist_1, hist_2, ..., hist_{13}, \\ mean(HL_1), var(HL_1), mean(LH_1), var(LH_1), \\ mean(HH_1), var(HH_1), mean(HL_2), var(HL_2), \\ mean(LH_2), var(LH_2), mean(HH_2), var(HH_2)\rangle\}$$

3.5 JSEG-bag

Liu et al. (2008) proposed two bag generators. One of them is named JSEG-bag, which is based on the JSEG image segmentation algorithm (Deng and Manjunath 2001). The other one is named Attention-bag and based on the salient point-based technique. However, in their experiments, the results of Attention-bag are worse than those of SBN, so we only consider JSEG-bag.

[4] Hue-Saturation-Value (HSV) are one of the most common cylindrical-coordinate representations of points in a RGB color model. The HSV color space and its variants are proven to be particularly amenable to color image analysis.

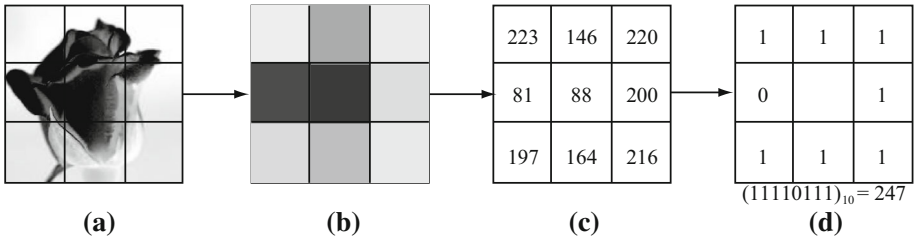

Fig. 4 Local binary patterns (LBP). Supposing **a** is a 3 × 3 pixel window, **c** is its corresponding *gray values*, and **d** is the 8-bits LBP

Because JSEG-bag is based on the JSEG algorithm, we first introduce that algorithm. Deng and Manjunath (2001) presented the JSEG image segmentation algorithm for unsupervised segmentation of color-texture regions in images and videos. The method consists of two independent steps: color quantization and spatial segmentation. In the first step, colors in an image are quantized into several representative classes that can be used to differentiate regions in the image. This quantization is performed in the color space alone without considering the spatial distributions. Afterwards, image pixel colors are replaced by their corresponding color class labels, thus forming a class-map of the image. The second step is the spatial segmentation on the class-map of the image.

In JSEG-bag, it firstly segments an image with the JSEG algorithm (Deng and Manjunath 2001). Then it selects the top k regions from the segmented image in order of decreasing regions' areas. Note that in our experiments, we vary the different values of k as 2, 6, and 10. In the third step of JSEG-bag, it computes the R, G and B color mean values of each region. Eventually, the image is converted into a corresponding image bag consisting of k 3-dimensional feature vectors (instances). The segmented result is shown in Fig. 6g.

3.6 Local binary patterns

Local binary pattern (LBP) (Ojala et al. 2002) is a local descriptor that captures the appearance of an image in a small neighborhood around a pixel. An LBP is a string of bits, with one bit for each of the pixels in the neighborhood. Each bit is turned on or off depending on whether the intensity of the corresponding pixel is greater than the intensity of the central pixel. Usually, these binary strings are pooled in local histograms, rather than directly using the binary strings.

The LBP in our experiments is from the open source library VLFeat.[5] In VLFeat, it implements only the case of 3 × 3 pixel neighborhoods which is found to be optimal in several applications. In particular, as shown in Fig. 4, the LBP centered on pixel (x, y) is a string of eight bits. Each bit is equal to one if the corresponding pixel is brighter than the central one. Pixels are scanned starting from the one to the right in anti-clockwise order. For a 3 × 3 neighborhood, an LBP is a string of eight bits and so there are 256 possible LBPs. In practice, the 256 patterns are further quantized into a 58 quantized patterns according to the uniform patterns (Heikkilä and Pietikäinen 2006). The quantized LBP patterns are further grouped into local histograms. In our experiments, we divide an image with 40 × 40 pixel windows. Then the quantized LBPs in each window are aggregated into a histogram by using bilinear interpolation along the two spatial dimensions. Thus, the bag generated by

[5] The *VLFeat* toolbox is available at http://www.vlfeat.org/.

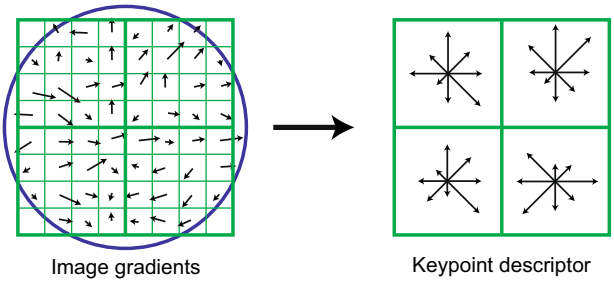

Fig. 5 A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the *left*. These are weighted by a Gaussian window, indicated by the *overlaid circle*. These samples are then accumulated into orientation *histograms* summarizing the contents over 4×4 subregions, as shown on the *right*, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a 2×2 descriptor array computed from an 8×8 set of samples, whereas the experiments in this paper use 4×4 descriptors computed from a 16×16 sample array (Color figure online)

LBP from a 240×360 image has totally 54 ($(240/40) \times (360/40) = 6 \times 9$) instances with 58 dimensions.

3.7 Scale invariant feature transform

A scale invariant feature transform (SIFT) feature (Lowe 2004) is a 3D spatial histogram of the image gradients in characterizing the appearance of an image keypoint. The first thing of computing SIFT descriptors is to extract SIFT keypoints, for whose details please refer to Lowe (2004). After collecting N SIFT keypoints, as shown in Fig. 5, for each SIFT keypoint, we compute the gradient magnitude and orientation at each image sample point in an image patch. These samples are weighed by the gradient norm and accumulated in a 3D histogram h, which forms the SIFT descriptor of the image patch. An additional Gaussian weighting function is applied to give less importance to gradients farther away from the keypoint center. Orientations are quantized into 8 bins and the spatial coordinates into four each. Therefore, the resulting SIFT descriptor is of dimension 128 ($8 \text{ bins} \times 4 \times 4 = 128$ bins). Note that, Fig. 5 just shows a 2×2 descriptor array computed from an 8×8 set of samples. In consequence, the bag generated by SIFT contains N instances of 128 dimensions.

3.8 Other image bag generators

Yang and Lozano-Pérez (2000) developed a bag generator called PRegions which is based on possible regions of images. This bag generator sets a list of possible *regions of interest* (ROI) in advance. After that, an image is divided into such overlapping regions. Each region is filtered and converted into a feature vector. In this way, the image is represented by a set of feature vectors. However, PRegions is very time-consuming and its performance is mediocre (Zhou et al. 2003).

In Zhou et al. (2003), a bag generator named ImaBag was presented. In the first step of this method, image pixels are clustered based on their colored and spatial features, where the clustering process is accomplished by a SOM neural network. Then, the clustered blocks are transformed into a specific number of regions by eliminating isolated pixels and merging scattered blocks. Finally, the resulting regions are converted into three-dimensional numerical instances of the image bag formed by their mean R, G, B values. Note that performance of

the two bag generators is much worse than that of SBN, as reported in Zhou et al. (2003), and this is the reason why we evaluate the other seven bag generators without these two.

3.9 Recap of bag generators

We have described specific techniques and computations performed by the nine state-of-the-art bag generators. In this section, we will provide a brief comparison and some conclusions about them.

As mentioned earlier, Row, SB and SBN are three non-segmentation bag generators only extracting color features. They segment the original images into multiple regions by using fixed strategies, which might divide objects into several parts. And it might be disadvantageous for SBN: its overlapping strategy could make one object in an original image (bags) be presented in multiple regions (instances) many times, which appears to be problematic based on the results shown in Sect. 5.2.1.

Blobworld, k-meansSeg, WavSeg and JSEG-bag are segmentation bag generators. They are similar in that they first segment original images into multiple regions (instances), and then extract features for presenting each local region. The different points among them are their different segmentation approaches. Blobworld and k-meansSeg extract the pixel-level or blob-level features firstly. After that, they cluster these pixels or blobs into several regions (instances), i.e., Gaussian Mixture Model of Blobworld and k-means of k-meansSeg. Finally, for each region, they compute the average value of pixels' or blobs' features in the same one region as the regions' features. WavSeg and JSEG-bag employ the SPCPE and JSEG segmentation algorithms, respectively, to segment original images. The final step of these two is extracting features from the multiple regions. In short, k-meansSeg and WavSeg contain both color and texture information of each region, and apart from this, Blobworld also contains the spatial information. However, JSEG-bag merely has color information. The segmentation results of different segmentation bag generators are shown in Fig. 6.

LBP and SIFT are two famous local descriptors employed as bag generators in this paper. They both compute the histogram-based features of local regions (instances) in images (bags). An important thing is that they both process the grayscale images, therefore their local features (i.e., the bits strings of LBP and the gradient distributions of SIFT) only contain texture information, without any color information.

In addition, from the view of sampling strategies, it is obvious to find that SB, SBN and LBP sample dense patches/regions to construct instances in bags. However, the SIFT descriptor (instance) is just based on the keypoints detected by SIFT detectors, rather than sampling dense local regions from original images. Moreover, the other bag generators only treat image segments as instances.

4 Empirical configurations

In this section, we first introduce seven state-of-the-art multi-instance learning algorithms used in our experiments. Then we describe the data sets and the evaluation criteria used in our empirical study.

4.1 Learning algorithms

Since the investigation of the drug activity prediction problem, many MIL algorithms have been proposed. According to a recent MIL review (Amores 2013), MIL algorithms are

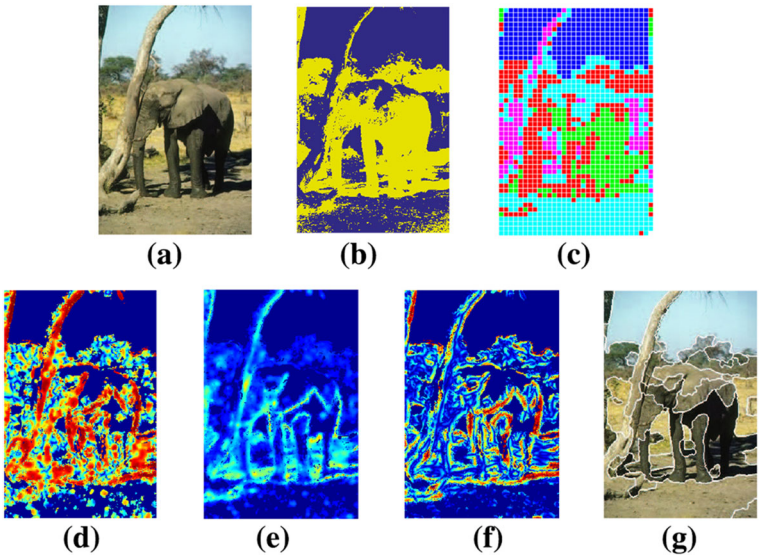

Fig. 6 Corresponding segmentation results of these four *segmentation bag generators*, i.e., WavSeg, *k*-meansSeg, Blobworld and JSEG-bag. Segmented regions are shown in their representative *colors*. The figures are best viewed in *color*. **a** Original image. **b** WavSeg. **c** kmeansSeg. **d** Anisotropy of Blobworld. **e** Contrast of Blobworld. **f** Polarity of Blobworld. **g** JSEG-bag (Color figure online)

grouped into three paradigms: (a) the Instance-Space (IS) paradigm; (b) the Bag-Space (BS) paradigm; (c) the Embedded-Space (ES) paradigm, based on how they manage the information from the multi-instance data. In short, for the IS paradigm, the discriminative information is considered to lie at the instance-level, while in the BS paradigm, the discriminative information is at the bag level. The MIL algorithms in ES explicitly or implicitly map each MIL bag to a single feature vector which summarizes the relevant information about the whole bag.

Therefore, we select the corresponding presentative learning algorithms from each paradigm, respectively, which are: *Citation-kNN* (Wang and Zucker 2000) and *miGraph* (Zhou et al. 2009) for the BS paradigm; *MIBoosting* (Xu and Frank 2004), *miSVM* (Andrews et al. 2003) and *MIForests* (Leistner et al. 2010) for the IS paradigm; *MILES* (Chen et al. 2006) and *miFV* (Wei et al. 2014) for the ES paradigm.

In addition, the assumptions in these MIL algorithms can be divided into two groups, i.e., the standard MIL assumption and the relaxed MIL assumptions. The standard MIL assumption is that *a bag is positive if and only if one or more of its instances are positive, otherwise it is labeled negative*. In this paper, Citation-*k*NN, miSVM, MILES and MIForests obey the standard assumption. The relaxed assumptions stress the relationship between instances in a bag in determining the bag's label, rather than one key instance can determine the bag's label assumed in the standard assumptions. For example, miGraph treated instances in each MIL bag as *non-i.i.d. samples* (Zhou et al. 2009); MIBoosting assumed that *all instances* contributed equally and independently to a bag's class label (Xu and Frank 2004); and miFV grouped instances in bags and encoded them into a new feature representation with the bag-level discriminative information, which implicitly assumed the instances in bag are *non-i.i.d.* (Wei et al. 2014). In the following, we try to give the key points about these MIL algorithms in this section.

Citation-kNN was proposed by Wang and Zucker (2000). The minimum Hausdorff distance was used as the bag-level distance metric, which is why we consider it to be a BS paradigm learning algorithm. In addition, when predicting the label of a new bag, Citation-*k*NN considers not only the bags that are the nearest neighbors of the new bag, but also the bags that count the new bag as their neighbor.

miGraph. Most previous MIL algorithms treat instances in the bags as independently and identically distributed. Considering that the instances in a bag are rarely independent in real tasks, Zhou et al. (2009) proposed miGraph to solve MIL problems by treating instances as non-i.i.d. samples. Their basic idea is to regard a bag as an entity to be processed as a whole (which demonstrates that it is a BS paradigm algorithm), and instances as inter-correlated components of the entity. miGraph implicitly constructs graphs by deriving affinity matrices and defines an efficient graph kernel considering the clique information.

MIBoosting. In contrast to the standard MIL assumption that there exists one or several "key" instances triggering the bag labels, MIBoosting (Xu and Frank 2004) assumes that all instances contribute equally and independently to a bag's label. Naturally, the process of predicting the label of a bag is conducted in two stages. In the first stage, MIBoosting determines each individual instance's class probabilities in a bag. And then, it combines these estimates to assign a class label to the bag in the second stage, which shows that it is an IS paradigm algorithm.

miSVM (Andrews et al. 2003) was designed for the instance-level classification problem. miSVM explicitly treats instance-level labels as unobserved integer variables, subjected to constraints defined by their bag-level labels. Intuitively, miSVM tries to look for an MI-separating linear discriminant such that at least one instance from every positive bag locates in the positive half-space, while all instances from negative bags locate in the negative half-space. Obviously, miSVM belongs to the IS paradigm.

MILES (Chen et al. 2006) converts MIL problems to standard supervised learning by embedding bags into an instance-based feature space (implicitly mapping) and selecting the most important features. They define a similarity measure between a bag and an instance. The coordinates of a given bag in the feature space represent the bag's similarities to various instances in the training set. At last, the 1-norm SVM is used to construct classifiers and select important features simultaneously.

miFV (Wei et al. 2014) is one kind of ES method with a vocabulary, and it is an efficient and scalable MIL algorithm. In miFV, the instances in MIL bags are first clustered into several "groups", and then mapped by its mapping function (explicitly mapping) into a new feature vector representation (i.e., Fisher Vector (Sánchez et al. 2013)) with the bag-level label, which implicitly assumes the instances are non-i.i.d. samples. Note that miFV encodes instances in bags into a bag-level feature vector, rather than embedding bags into an instance-based feature space which is done in MILES.

MIForests (Leistner et al. 2010) brings the advantage of random forests to multi-instance learning. MIForests treats the (hidden) labels of instances as random variables defined over a space of probability distributions, which is obviously an IS paradigm algorithm. Thus, they formulate multi-instance learning as an optimization procedure where the labels of the instances become the optimization variables. After that, they disambiguate the instance labels by iteratively searching for distributions that minimize the overall learning objective.

Table 1 *COREL* images

ID	Category name	ID	Category name
0	African people and villages	10	Dogs
1	Beach	11	Lizards
2	Historical building	12	Fashion models
3	Buses	13	Sunset scenes
4	Dinosaurs	14	Cars
5	Elephants	15	Waterfalls
6	Flowers	16	Antique furniture
7	Horses	17	Battle ships
8	Mountains and glaciers	18	Skiing
9	Food	19	Deserts

Note that, in our experiments, we selected the corresponding optimal parameters of these learning algorithms via cross validations on their training data. The specific information of parameters can be found in Section II of the Appendix.

4.2 Data sets

The data sets used in our experiments are taken from *COREL* and *MSRA* image data sources which are very representative and frequently used in many image tasks of MIL researches.

COREL images consist of 2000 images from 20 CD-ROMs published by the COREL Corporation, and thus contain 20 categories where each category contains 100 images.[6] Images are in the JPEG format of 384×256 or 256×384 image resolution. The category names are listed in Table 1 along with the identifiers for these 20 categories. Figure 7 shows some sample images from *COREL* images.

MSRA images are the second version of *MSRA-MM* data set (Li et al. 2009). The image data set was collected from Microsoft Live Search and it contains about 1 million images manually classified into 8 categories, i.e., *Animal*, *Cartoon*, *Event*, *Object*, *Scene*, *PeopleRelated*, *NamedPerson*, and *Misc*. We here select 10 sub-categories from them and each sub-category has 500 images. Note that instead of the standard original image resolution in *COREL* images, these images from *MSRA* images are in different image resolutions. The sub-category names of *MSRA* images are listed in Table 2. Some sample images from *MSRA* images are shown in Fig. 8.

Elephant, *Tiger* and *Fox* are another three data sets from *COREL* images. Note that in this paper, *we only consider the binary classification problem*. For these 3 data sets, they are constructed as follows. We treat each of them as the positive examples, and randomly sample 100 images from other categories as the negative ones. After that, we randomly partition the positive (negative) images into two equal parts, one half used for training while the other is used for testing.

On the other hand, we construct 3 image collections, i.e., *1000-Image* (10 data sets, i.e., Category 0–9 from *COREL* images), *2000-Image* (20 data sets, i.e., Category 0-19 from *COREL* images), and *MSRA* (10 data sets, i.e., 10 sub-categories from *MSRA* images). For each image collection, one-against-one strategy is used to construct data sets, which means that examples from one category are regarded as positive while examples from one of the remaining categories are regarded as negative. If the positive category is already selected, it

[6] The image data sets are available at http://www.cs.olemiss.edu/~ychen/ddsvm.html.

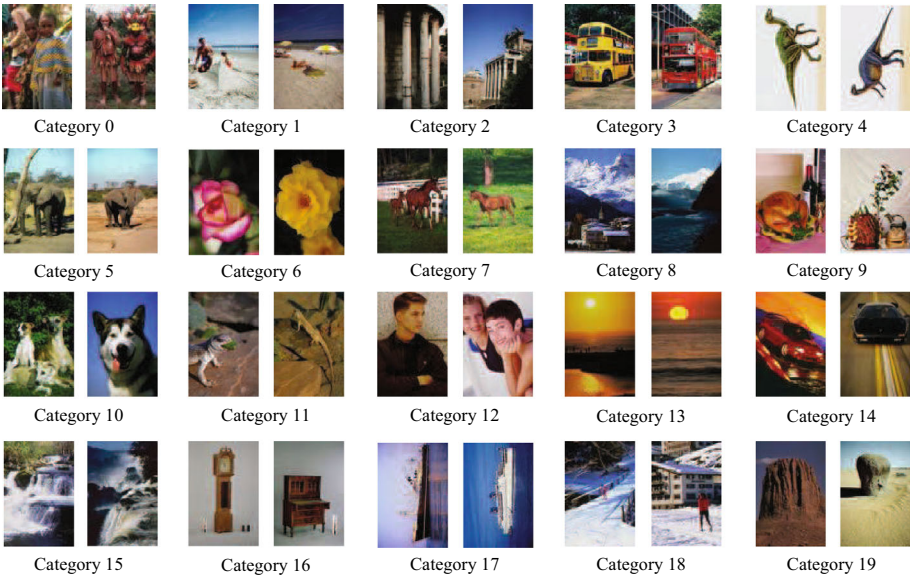

Fig. 7 Images randomly sampled from 20 categories of *COREL* images. Each category has two sample images. The figures are best viewed in *color* (Color figure online)

Table 2 10 sub-categories from *MSRA* images

ID	Sub-category name	ID	Sub-category name
0	Bald eagle	5	Audi
1	Bob esponja	6	Converse
2	Bugs bunny	7	Waterfall
3	Sonic	8	Basketball
4	Firework	9	American flag

Fig. 8 Images randomly sampled from 10 sub-categories of *MSRA* images. Each sub-category has two sample images. Note that images from *MSRA* images have different resolutions, but in order to make them look neat, we show them in the same size. The figures are best viewed in *color* (Color figure online)

will have 9 (19/9) possible choices for *1000-Image* (*2000-Image*/*MSRA*). For all the possible pairs of datasets, we randomly partition the positive (negative) images into two equal parts for training (test), which is the same as what we do on *Elephant*, *Fox* and *Tiger*. Moreover, on the training data of *Elephant*, *Tiger*, *Fox* and these three image collections, we run two times two-fold cross validations to obtain the corresponding optimal parameters for each learning

Table 3 Bag-size of each bag generator with its different patch sizes on the *2000-Image* data collection

	Row_4	Row_8	Row_16	Row_32	SB_4	SB_8
Bag-size	4 × 9	8 × 9	16 × 9	32 × 9	4 × 12	16 × 12
	SB_16	SB_32	SBN_4	SBN_8	SBN_16	SBN_32
Bag-size	64 × 12	256 × 12	4 × 15	9 × 15	121 × 15	729 × 15
	kmeansS._4	kmeansS._8	kmeansS._16	kmeansS._32	Blob.	WavS.
Bag-size	4.7 × 6	6.3 × 6	3.6 × 6	3.0 × 6	1.7 × 286	2 × 19
	J.-bag_2	J.-bag_6	J.-bag_10	LBP	SIFT	–
Bag-size	2 × 3	6 × 3	10 × 3	35 × 58	40 × 128	–

For example, the *Row* bag generator with the 4 × 4 patch size, we denote it by "*Row_4*". And for bag-size, we report it as a formula, i.e., "4 × 9", which means each bag generated by *Row_4* contains 4 instances with 9-dimention

algorithm. Finally, on each data set, we repeat the experiments three times with different training/test data splittings, and report the average classification accuracy rates.

In order to study the effect of patch size on learning, we vary the patch size of Row, SB, SBN and *k*-meansSeg among four different values, i.e., different patch sizes with 4 × 4, 8 × 8, 16 × 16, and 32 × 32. Note that the patch size in each bag generator has a different meaning. In Row, SB and SBN, the original images are resized into assigned patch sizes. But, the patch size in *k*-meansSeg is the size of the sliding window. For JSEG-bag, we vary the value of top *k* as 2, 6 and 10. WavSeg and Blobworld do not involve different patch sizes. In addition, considering the computational cost of learning algorithms in our experiments, we employ LBP and SIFT to extract 35 and 40 instances per image, respectively. However, some combinations (e.g., "miSVM with LBP", "MILES with SIFT", etc.) still can not return results in 7 days, cf. Table 16 in the Appendix. We present the corresponding bag-size of bag generators (with different patch sizes) as shown in Table 3.

4.3 Evaluation criteria

As mentioned earlier, because the number of positive bags is equal to the one of the negative bags in these data sets of our experiments, the impact of class imbalance can be ignored. So, we use *accuracy* as the evaluation criterion to evaluate the classification performances of bag generators with different MIL algorithms. In addition, in order to perform performance analysis among several combinations (bag generators+learning algorithms), the *Friedman test* is employed here which is widely-accepted as the favorable statistical test for comparisons of multiple algorithms over a number of data sets. The experimental results are shown in the next section.

4.3.1 Accuracy

Accuracy is used as a statistical measure of how well a binary classification test correctly identifies or excludes a condition. That is, in our experiments, accuracy is the number of true prediction test bags (both true positive bags and true negative bags) with respect to the total number of test bags:

$$accuracy = \frac{\sharp\, true\ positive\ bags + \sharp\, true\ negative\ bags}{\sharp\, test\ bags}$$

4.3.2 The Friedman test

Accuracy can clarify the difference between the performances of *one* bag generator applied with different learning algorithms. However, it is not sufficient when we try to use it to clarify the differences between *multiple* bag generators with different learning algorithms. In that case, we use the *Friedman test* for testing the significance of differences between *multiple* bag generators applied with *multiple* different learning algorithms.

The Friedman test (Demšar 2006) is a non-parametric equivalent of the repeated-measures ANOVA (Fisher 1959). It ranks the combinations of MIL algorithms and bag generators for each test data of corresponding image data sets separately, the best performing combination (bag generator+learning algorithm) getting the rank of 1, the second best rank 2 and so on. Given k comparing combinations (bag generators+learning algorithms) and N data sets, let r_i^j denote the rank of the jth combination on the ith data set (mean ranks are shared in case of ties). Let $R_j = \frac{1}{N}\sum_i r_i^j$ denote the average rank for the jth combination, under the null hypothesis (i.e., all combinations have "equal" performance), the following Friedman statistic F_F will be distributed according to the F-distribution with $k-1$ numerator degrees of freedom and $(k-1)(N-1)$ denominator degrees of freedom:

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}, \text{ where } \chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4}\right]$$

If the Friedman statistics F_F is larger than the corresponding critical values, the null hypothesis of "equal" performance among the combinations will be rejected. After that, for further analyzing the relative performance among the comparing combinations, the Nemenyi test (Nemenyi 1963) is used. The detailed results can be found in the next section.

5 Empirical results

In this section, we present and discuss the experimental results of the evaluations. The performances of image bag generators are demonstrated mainly in two aspects, i.e., *accuracy comparison* and *method observations*.

5.1 Accuracy comparison

In Table 4, we report the experimental results of all combinations (bag generator+learning algorithm) on the *Elephant*, *Fox*, *Tiger*, *1000-Image*, *2000-Image* and *MSRA* image data sets. In the following, we discuss the empirical results in two views, i.e., *the view of bag generator and the one of learning algorithm*. Finally, we recommend several outstanding combinations for practical applications.

5.1.1 From the bag generator view

As shown in Table 4, when combined with learning algorithms, SB, SBN and LBP most frequently achieve the best image classification performance on all the data sets. In order to have an overview of all the combinations' classification performance, we rank these combinations according to the decreasing order of classification accuracy rates, which is shown in Table 5. In this table, we can easily find that SB, SBN and LBP achieve satisfactory classification performance. As aforementioned in Sect. 3.9, SB, SBN and LBP extract features of

Table 4 The general image classification accuracy rates (mean±std. deviation) of the combinations (bag generators + learning algorithms)

	Row	SB	SBN	*k*-meansSeg	Blobworld	WavSeg	JSEG-bag	LBP	SIFT
Elephant									
Citation-*k*NN	.842 ± .081	.838 ± .079	.841 ± .067	.868 ± .080•	.500 ± .000	.818 ± .071	.789 ± .077	.500 ± .000	.685 ± .068
miGraph	**.865 ± .071**	.918 ± .051	.912 ± .057	.895 ± .042	.574 ± .062	.844 ± .060	.597 ± .049	**.930 ± .041**•	.527 ± .054
MIBoosting	.856 ± .069	**.920 ± .052**•	**.917 ± .055**	**.906 ± .060**	**.846 ± .082**	.802 ± .079	**.841 ± .096**	.837 ± .109	.741 ± .103
miSVM	.809 ± .091	.807 ± .056	.831 ± .093	.823 ± .040	.845 ± .079•	.842 ± .068	.743 ± .101	.759 ± .049	.584 ± .044
MILES	.836 ± .081	.902 ± .059•	.868 ± .062	.822 ± .081	.778 ± .097	.719 ± .086	.612 ± .063	.837 ± .083	.533 ± .061
miFV	.828 ± .083	.873 ± .062	.900 ± .054	.888 ± .051	.818 ± .089	**.866 ± .053**	.820 ± .079	.902 ± .047•	**.847 ± .075**
MIForests	.841 ± .073	.901 ± .069•	.874 ± .053	.829 ± .051	.780 ± .093	.725 ± .051	.615 ± .076	.846 ± .050	.532 ± .080
Fox									
Citation-*k*NN	.778 ± .085	.765 ± .071	.749 ± .089	.799 ± .094•	.500 ± .000	.750 ± .080	.723 ± .081	.500 ± 000	.689 ± .058
miGraph	.803 ± .075	.834 ± .099	**.846 ± .088**	**.833 ± .081**	.553 ± .035	.750 ± .085	.546 ± .062	**.894 ± .083**•	.528 ± .052
MIBoosting	**.808 ± .081**	**.882 ± .076**•	.845 ± .080	.829 ± .098	**.824 ± .077**	.758 ± .074	.748 ± .109	.810 ± .101	.714 ± .112
miSVM	.789 ± .078	.742 ± .080	.772 ± .079	.773 ± .089	.795 ± .078	.799 ± .085•	.679 ± .094	.648 ± .102	.558 ± .045
MILES	.789 ± .077	.819 ± .082	.797 ± .079	.735 ± .092	.686 ± .076	.639 ± .085	.577 ± .055	.830 ± .105•	.529 ± .051
miFV	.784 ± .089	.826 ± .093	.815 ± .075	.804 ± .092	.803 ± .083	**.816 ± .084**	**.770 ± .108**	.885 ± .087•	**.817 ± .088**
MIForests	.790 ± .089	.823 ± .098	.802 ± .073	.740 ± .090	.686 ± .082	.641 ± .081	.575 ± .099	.828 ± .087•	.527 ± .092
Tiger									
Citation-*k*NN	.826 ± .076	.810 ± .089	.812 ± .081	.840 ± .064	.500 ± .000	.844 ± .062•	.789 ± .081	.500 ± .000	.685 ± .061
miGraph	.852 ± .072	.896 ± .062	.891 ± .060	.860 ± .063	.549 ± .042	.802 ± .061	.603 ± .042	**.912 ± .053**•	.526 ± .049
MIBoosting	**.867 ± .083**	**.927 ± .041**•	**.905 ± .049**	**.865 ± .068**	**.888 ± .053**	.846 ± .053	**.835 ± .092**	.830 ± .104	.758 ± .112
miSVM	.820 ± .069	.794 ± .073	.812 ± .062	.815 ± .061	.833 ± .054	.844 ± .047•	.735 ± .074	.693 ± .098	.562 ± .034
MILES	.826 ± .073	.859 ± .066•	.800 ± .091	.779 ± .102	.744 ± .072	.664 ± .101	.600 ± .035	.833 ± .089	.534 ± .042
miFV	.828 ± .087	.854 ± .072	.867 ± .063	.827 ± .060	.825 ± .052	**.890 ± .051**•	.806 ± .076	.879 ± .070	**.842 ± .099**
MIForests	.830 ± .084	.861 ± .063•	.799 ± .052	.781 ± .063	.742 ± .050	.662 ± .055	.606 ± .081	.836 ± .067	.534 ± .091

Table 4 continued

	Row	SB	SBN	*k*-meansSeg	Blobworld	WavSeg	JSEG-bag	LBP	SIFT
1000-Image									
Citation-*k*NN	.829 ± .084	.829 ± .082	.833 ± .081	.865 ± .071•	.500 ± .000	.844 ± .064	.776 ± .089	.500 ± .000	.755 ± .093
miGraph	.874 ± .064	.927 ± .048	.917 ± .052	.900 ± .061	.584 ± .049	.843 ± .058	.613 ± .053	**.954 ± .049**•	.543 ± .044
MIBoosting	**.875 ± .062**	**.935 ± .049**•	**.928 ± .052**	**.913 ± .071**	**.891 ± .055**	.827 ± .064	.847 ± .082	.855 ± .100	.800 ± .095
miSVM	.837 ± .067	.808 ± .056	.829 ± .076	.853 ± .082	.810 ± .074	**.870 ± .060**•	.751 ± .073	.725 ± .091	.569 ± .056
MILES	.855 ± .073	.894 ± .058•	.876 ± .080	.837 ± .085	.771 ± .071	.693 ± .085	.630 ± .060	.868 ± .073	.534 ± .046
miFV	.848 ± .079	.888 ± .053	.897 ± .056	.892 ± .070	.868 ± .059	.732 ± .072	**.933 ± .048**•	.877 ± .070	**.848 ± .072**
MIForests	.859 ± .071	.897 ± .058•	.881 ± .062	.843 ± .068	.778 ± .057	.690 ± .074	.636 ± .048	.871 ± .072	.541 ± .066
2000-Image									
Citation-*k*NN	.832 ± .088	.828 ± .083	.821 ± .084	.867 ± .074•	.500 ± .000	.838 ± .075	.776 ± .088	.500 ± .000	.722 ± .099
miGraph	.865 ± .070	.910 ± .063	.902 ± .062	.886 ± .084	.558 ± .058	.830 ± .098	.602 ± .047	**.938 ± .048**•	.543 ± .051
MIBoosting	**.870 ± .071**	**.926 ± .060**•	**.920 ± .059**	**.906 ± .068**	**.891 ± .065**	.828 ± .070	**.837 ± .101**	.856 ± .097	.807 ± .093
miSVM	.832 ± .081	.803 ± .082	.817 ± .089	.840 ± .086	.819 ± .063	.867 ± .062•	.745 ± .089	.734 ± .086	.572 ± .048
MILES	.845 ± .089	.888 ± .068•	.864 ± .072	.811 ± .091	.748 ± .082	.722 ± .106	.619 ± .061	.872 ± .079	.538 ± .049
miFV	.833 ± .076	.875 ± .062	.897 ± .063	.887 ± .063	.808 ± .071	**.875 ± .061**	.822 ± .080	.924 ± .058•	**.874 ± .077**
MIForests	.847 ± .076	.890 ± .064•	.869 ± .063	.810 ± .062	.750 ± .073	.728 ± .061	.622 ± .077	.877 ± .057	.542 ± .078
MSRA									
Citation-*k*NN	.824 ± .065	.823 ± .054	.854 ± .057•	.797 ± .073	.500 ± .000	.776 ± .088	.779 ± .071	.509 ± .004	.665 ± .064
miGraph	.860 ± .051	**.936 ± .036**•	**.930 ± .030**	.858 ± .053	.682 ± .052	.841 ± .069	.656 ± .053	**.894 ± .056**	.660 ± .083
MIBoosting	**.887 ± .054**	.915 ± .045	.928 ± .035	**.863 ± .070**	**.929 ± .045**•	**.893 ± .053**	**.863 ± .072**	.868 ± .061	**.843 ± .065**
miSVM	.825 ± .066	.829 ± .053	.845 ± .054•	.772 ± .072	.816 ± .069	.834 ± .072	.744 ± .092	N/A	N/A
MILES	.743 ± .044	.758 ± .049•	.756 ± .038	.739 ± .049	.707 ± .048	.655 ± .074	.651 ± .062	N/A	N/A
miFV	.818 ± .060	.832 ± .063	.892 ± .046•	.830 ± .062	.868 ± .054	.847 ± .076	.797 ± .070	.866 ± .068	.820 ± .072
MIForests	.746 ± .075	.755 ± .064	.760 ± .062•	.737 ± .061	.710 ± .070	.659 ± .059	.655 ± .082	.515 ± .054	.506 ± .079

The highest average accuracy of one row is marked with "•", and the highest one of one column is marked in bold. *N/A* indicates that these combinations could not return a result in 7 days

dense regions (instances) from original images (bags). The observations from Tables 4 and 5 illustrate sampling dense regions to construct instances in bags provides better results than other segmentation-based bag generators. Meanwhile, it is not a pure coincidence, because in the computer vision community, dense sampling has already shown to improve results over sparse interest points for image classification (Li and Perona 2005; Nowak et al. 2006).

In addition, because the image data sets contain various types of image classes, we partition every data collection (i.e., *1000-Image*, *2000-Image* and *MSRA*) into two main parts: object-style classification and scene-style classification. In *1000-Image*, the categories 0, 1 and 8 are treated as scene-style classification, while the remaining categories are the object-style. In *2000-Image*, the categories 0, 1, 8, 13, 15, 18 and 19 are treated as scene-style. Finally, in *MSRA*, the sub-categories 4 and 7 are treated as scene-style. Besides the general image classification results (in Table 4), we also report the accuracy rates of object classification performances and scene classification performances in Tables 6 and 7, respectively.

As shown in Table 6, it has an almost identical trend of the average image classification results (in Table 4). Here we focus on the scene-style classification. Compared with the object-style classification results in Table 6, from Table 7, we can find Row's performance becomes prominent and LBP gets worse in most cases. In order to directly compare these results, we report them in Table 8 and do the pairwise t test. From that table, we can see the performances of Row on scene-style classification are significantly better than the ones of object-style. And the performances of SB, SBN and LBP are comparable. Moreover, the accuracy rates of LBP on scene-style classification are lower than the ones of object-style in most cases. In addition, similar to Table 5, we report the scene-style classification results in ranks shown in Table 9. In this table, we only rank the top eight combinations, which also shows: SB and SBN still outperform others; Row becomes prominent; while LBP is not satisfactory. That is straightforward. Because for scene classification, color features have strong discriminative information, while some other features (e.g., texture features) might be not strongly useful. As aforementioned, Row extracts color features, while LBP extracts the texture patterns from gray scale images.

5.1.2 From the learning algorithm view

Recall the classification results and ranks presented in Tables 4 and 5. From the view of learning algorithms, as shown in Table 4, miGraph and MIBoosting achieve the greatest number of wins in performance. Table 5 makes it clear which algorithm performs better. In addition, miFV also has satisfactory accuracy rates. Similar observations are demonstrated in Table 9. These observations can be explained by the fact that the miGraph, MIBoosting and miFV algorithms do not adopt the standard MIL assumption. And instead, miGraph and miFV explicitly or implicitly assume that the instances in the bag are non-i.i.d. samples; MIBoosting takes advantage of aggregating properties of bags. Note that it is unclear whether real problems really follow the standard MIL assumption. In particular, in image-related tasks, the position-relation among the patches/pixels are crucial; for instance, given the same set of patches/pixels, putting them into different positions will result in different image semantics, leading to different labels. For example, in the image of a "beach" shown in Fig. 7, the "sand" and "sea" must co-occur. However, if only one of these things occurs in the image then it will be "non-beach", e.g., the images of "deserts" only contain "sand". Thus, it is not strange that the performances of miGraph, MIBoosting and miFV on image classification are better than algorithms that assume the instances as i.i.d. samples, especially on the bag-level prediction tasks.

Table 5 Ranks of average accracy rates of each combination on *Elephant*, *Fox*, *Tiger* and these three image data collections

Combination	Eleph.	Fox	Tiger	1000-Img.	2000-Img.	MSRA	Ave. Rank
miGra.+LBP	1•	1•	2⋆	1•	1•	7	2.2•
MIBoost.+SB	2⋆	3	1•	2⋆	2⋆	5	2.5⋆
miGra.+SB	3	6	4	5	5	1•	4.0
MIBoost.+SBN	4	5	3	4	4	4	4.0
miGra.+SBN	5	4	5	6	7	2⋆	4.8
miFV+LBP	8	2⋆	8	16	3	12	8.2
MIBoost.+*k*meansS.	6	9	11	7	6	13	8.7
miFV+SBN	10	17	10	10	8	8	10.5
MIBoost.+Blob.	21	12	7	13	9	3	10.8
miGra.+*k*meansS.	11	7	13	8	13	16	11.3
miFV+SB	14	11	15	14	15	23	15.3
MIBoost.+Row	19	19	9	18	19	9	15.5
MIForests+SB	9	13	12	9	10	40	15.5
MILES+SB	7	14	14	11	11	38	15.8
miFV+*k*meansS.	12	20	29	12	12	24	18.2
miGra.+Row	18	21	16	19	23	15	18.7
miFV+WavS.	17	16	6	50	16	18	20.5
miFV+SIFT	20	15	19	30	17	29	21.7
miSVM+WavS.	24	25	18	21	22	22	22.0
MIBoost.+LBP	29	18	27	27	25	11	22.8
C.-*k*NN+*k*meansS.	16	24	21	24	21	32	23.0
MIForests+SBN	13	23	41	15	20	37	24.8
MIForests+LBP	32	10	22	20	14	57	25.8
MILES+SBN	15	26	40	17	24	39	26.8
MILES+LBP	30	8	25	23	18	59	27.2
MIBoost.+J.-bag	27	41	23	31	30	14	27.7
miFV+Blob.	40	22	32	22	43	10	28.2
MIForests+Row	25	28	26	25	26	41	28.5
MIBoost.+WavS.	44	37	17	41	36	6	30.2
miFV+Row	36	31	28	29	31	30	30.8
miSVM+Blob.	22	27	24	42	39	31	30.8
miFV+J.-bag	39	35	38	3	37	33	30.8
MILES+Row	31	30	31	26	27	43	31.3
miGra.+WavS.	23	39	39	34	34	21	31.7
C.-kNN+Row	33	32	30	38	32	27	32.0
C.-*k*NN+SBN	26	40	35	37	38	17	32.2
C.-*k*NN+WavS.	41	38	20	32	29	35	32.5
miSVM+*k*meansS.	37	33	34	28	28	36	32.7
miSVM+Row	42	29	33	35	33	26	33.0
C.-*k*NN+SB	28	36	37	39	35	28	33.8
miSVM+SBN	34	34	36	40	40	19	33.8
miSVM+SB	43	42	42	43	45	25	40.0

Table 5 continued

Combination	Eleph.	Fox	Tiger	1000-Img.	2000-Img.	MSRA	Ave. Rank
MIForests+*k*meansS.	35	43	43	33	42	45	40.2
MILES+*k*meansS.	38	44	45	36	41	44	41.3
MIBoost.+SIFT	50	46	46	44	44	20	41.7
C.-*k*NN+J.-bag	45	45	44	46	46	34	43.3
MIForests+Blob.	46	49	48	45	47	46	46.8
MILES+Blob.	47	48	47	47	48	47	47.3
miSVM+J.-bag	49	50	49	49	49	42	48.0
C.-*k*NN+SIFT	53	47	51	48	53	49	50.2
MIForests+WavS.	51	52	53	53	51	51	51.8
miSVM+LBP	48	51	50	51	50	62	52.0
MILES+WavS.	52	53	52	52	52	53	52.3
MIForests+J.-bag	54	55	54	54	54	54	54.2
MILES+J.-bag	55	54	56	55	55	55	55.0
miGra.+J.-bag	56	58	55	56	56	52	55.5
miGra.+Blob.	58	57	58	57	58	48	56.0
miSVM+SIFT	57	56	57	58	57	61	57.7
miGra.+SIFT	61	60	61	59	59	50	58.3
MIForests+SIFT	60	61	59	60	60	56	59.3
MILES+SIFT	59	59	60	61	61	60	60.0
C.-*k*NN+LBP	62	62	63	62	63	58	61.7
C.-*k*NN+Blob.	63	63	62	63	62	63	62.7

The first rank one in one column is followed by "•", and the second rank one is followed by "⋆". "Ave. Rank" is the average value of all the ranks on these data sets

5.1.3 Recommended combinations

In this section, we recommend several combinations (bag generator+learning algorithm) which have outstanding performance in image classification tasks. In Table 5, we list all the combinations by their corresponding ranks. Focusing on the top eight ones, we do the *Friedman test* for them.

In our setting, we have $k = 8$ comparing combinations and $N = 6$ image data sets. According to Sect. 4.3.2, we first compute the Friedman statistics. The Friedman statistics in our setting is $F_F = 9.0554$, which is larger than the critical values (i.e., 2.29) at significance level $\alpha = 0.05$, therefore the null hypothesis is clearly rejected. It indicates that there are significant differences between the performance of these eight combinations. Consequently, we need to proceed with a post-hoc test to further analyze the relative performance among these combinations. As we are interested in comparing all combinations with each other, the *Nemenyi test* (Nemenyi 1963) is employed. The performance of two combinations is significantly different if the corresponding average ranks differ by at least the critical difference:

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$$

Table 6 The object-style classification accuracy rates (mean±std. deviation) of the combinations (bag generators + learning algorithms)

	Row	SB	SBN	*k*-meansSeg	Blobworld	WavSeg	JSEG-bag	LBP	SIFT
1000-Image									
Citation-*k*NN	.831 ± .083	.830 ± .078	.829 ± .049	.867 ± .067 •	.500 ± .000	.847 ± .061	.783 ± .083	.500 ± .000	.748 ± .076
miGraph	.871 ± .062	.931 ± .039	.917 ± .053	.903 ± .061	.579 ± .042	.846 ± .054	.615 ± .052	**.951 ± .041**•	.544 ± .043
MIBoosting	**.877 ± .059**	**.938 ± .035**•	**.933 ± .053**	**.910 ± .066**	**.893 ± .055**	.829 ± .065	.852 ± .088	.857 ± .093	.791 ± .094
miSVM	.839 ± .069	.812 ± .064	.833 ± .076	.856 ± .078	.810 ± .067	**.875 ± .060**•	.753 ± .072	.725 ± .082	.567 ± .041
MILES	.855 ± .059	.898 ± .065•	.871 ± .074	.840 ± .082	.770 ± .061	.691 ± .088	.635 ± .060	.866 ± .071	.540 ± .042
miFV	.844 ± .062	.892 ± .054	.905 ± .053	.898 ± .058	.871 ± .057	.738 ± .074	**.933 ± .046**•	.882 ± .068	**.848 ± .069**
MIForests	.855 ± .071	.901 ± .062•	.875 ± .061	.841 ± .085	.774 ± .058	.694 ± .092	.634 ± .056	.869 ± .041	.537 ± .049
2000-Image									
Citation-*k*NN	.833 ± .083	.830 ± .081	.822 ± .082	.864 ± .070•	.500 ± .000	.845 ± .072	.771 ± .063	.500 ± .000	.734 ± .083
miGraph	.870 ± .073	.914 ± .058	.904 ± .058	.886 ± .087	.560 ± .057	.831 ± .099	.599 ± .053	**.953 ± .041**•	.544 ± .044
MIBoosting	**.873 ± .072**	**.926 ± .054**•	**.916 ± .059**	**.908 ± .063**	**.888 ± .060**	.828 ± .070	**.837 ± .089**	.858 ± .091	.809 ± .100
miSVM	.831 ± .077	.801 ± .080	.814 ± .099	.844 ± .080	.826 ± .064	.866 ± .063•	.743 ± .081	.738 ± .093	.572 ± .041
MILES	.846 ± .083	.887 ± .066•	.867 ± .080	.811 ± .094	.749 ± .080	.729 ± .085	.623 ± .060	.879 ± .088	.542 ± .047
miFV	.835 ± .074	.884 ± .063	.898 ± .061	.890 ± .062	.811 ± .070	**.877 ± .059**	.825 ± .078	.928 ± .051•	**.879 ± .078**
MIForests	.850 ± .087	.886 ± .065•	.869 ± .080	.813 ± .096	.751 ± .080	.733 ± .093	.624 ± .060	.883 ± .080	.544 ± .046
MSRA									
Citation-*k*NN	.815 ± .060	.823 ± .059	.851 ± .052•	.796 ± .073	.500 ± .000	.776 ± .095	.778 ± .070	.503 ± .004	.661 ± .074
miGraph	.863 ± .051	**.937 ± .036**•	**.932 ± .030**	.854 ± .051	.684 ± .053	.842 ± .069	.659 ± .055	**.895 ± .053**	.658 ± .075
MIBoosting	**.887 ± .067**	.912 ± .042	.921 ± .040	**.860 ± .070**	**.924 ± .043**•	**.897 ± .052**	**.854 ± .072**	.864 ± .061	**.837 ± .069**
miSVM	.824 ± .063	.834 ± .051	.839 ± .054•	.765 ± .067	.813 ± .065	.826 ± .074	.739 ± .089	N/A	N/A
MILES	.740 ± .042	.758 ± .042•	.758 ± .036•	.741 ± .050	.709 ± .051	.656 ± .068	.651 ± .062	N/A	N/A
miFV	.812 ± .058	.828 ± .060	.887 ± .048•	.828 ± .063	.863 ± .055	.840 ± .062	.791 ± .068	.858 ± .065	.823 ± .071
MIForests	.739 ± .058	.758 ± .060	.761 ± .049	.743 ± .064	.709 ± .050	.657 ± .065	.654 ± .072	.828 ± .072•	.623 ± .069

The highest average accuracy of one row is marked with "•", and the highest one of one column is marked in bold. *N/A* indicates that these combinations could not return a result in 7 days

Table 7 The scene-style classification accuracy rates (mean±std. deviation) of the combinations (bag generators+learning algorithms)

	Row	SB	SBN	*k*-meansSeg	Blobworld	WavSeg	JSEG-bag	MIBoosting	SIFT
1000-Image									
Citation-*k*NN	.859±.055	.811±.109	.832±.082	.868±.091•	.500±.000	.833±.076	.733±.034	.500±.000	.735±.087
miGraph	**.884±.060**	.873±.074	.866±.089	**.877±.062**	.571±.052	.817±.093	.580±.029	**.895±.052**•	.510±.009
MIBoosting	.879±.050	**.886±.096**	**.887±.073**•	.866±.112	.866±.084	.817±.071	.796±.080	.813±.098	.777±.087
miSVM	.844±.067	.757±.073	.828±.073	.827±.054	.738±.085	**.857±.052**•	.733±.064	.766±.072	.580±.023
MILES	.875±.065•	.861±.072	.853±.077	.864±.080	.717±.090	.718±.0898	.599±.070	.871±.072	.503±.005
miFV	.830±.089	.870±.084	.868±.083	.867±.099	**.868±.072**	.732±.065	**.898±.066**•	.850±.089	**.876±.056**
MIForests	.879±.076•	.864±.075	.855±.063	.869±.051	.718±.087	.722±.053	.603±.066	.872±.076	.602±.023
2000-Image									
Citation-*k*NN	.859±.070•	.829±.082	.838±.066	.859±.073•	.500±.000	.777±.071	.771±.090	.500±.000	.647±.087
miGraph	.873±.062	.892±.060•	.884±.058	.846±.095	.534±.050	.803±.088	.601±.042	**.875±.053**	.536±.050
MIBoosting	**.877±.065**	**.902±.070**•	**.891±.061**	**.878±.083**	**.882±.072**	.814±.074	**.832±.098**	.834±.101	.756±.095
miSVM	.845±.070•	.803±.070	.833±.079	.813±.087	.818±.073	.835±.072	.742±.095	.739±.072	.583±.050
MILES	.855±.071	.869±.066	.875±.060•	.843±.083	.750±.090	.697±.083	.610±.066	.818±.077	.532±.052
miFV	.852±.063	.864±.062	.883±.070•	.869±.073	.778±.094	**.845±.073**	.816±.074	.861±.070	**.815±.072**
MIForests	.855±.070	.869±.075	.876±.080•	.844±.090	.752±.073	.699±.072	.610±.092	.819±.071	.556±.043
MSRA									
Citation-*k*NN	.907±.021•	.894±.020	.890±.013	.904±.022	.500±.000	.887±.021	.883±.008	.501±.002	.805±.003
miGraph	.926±.010	**.944±.030**	**.949±.019**•	.920±.021	.723±.008	.901±.005	.683±.031	.917±.015	.652±.011
MIBoosting	**.942±.032**	.943±.030	.949±.021	**.934±.023**	**.964±.015**•	.899±.074	**.927±.024**	**.918±.010**	.843±.013
miSVM	.929±.025•	.898±.016	.909±.035	.870±.031	.881±.014	**.902±.031**	.890±.018	N/A	N/A
MILES	.814±.019	.815±.011•	.789±.027	.804±.021	.729±.014	.755±.004	.646±.073	N/A	N/A
miFV	.924±.023	.917±.013	.941±.025•	.910±.013	.921±.004	.894±.015	.910±.040	.882±.021	**.889±.021**
MIForests	.813±.020	.820±.016•	.792±.035	.806±.030	.731±.011	.755±.029	.648±.018	.802±.022	.613±.025

The highest average accuracy of one row is marked with "•", and the highest one of one column is marked in bold. *N/A* indicates that these combinations could not return a result in 7 days

Table 8 Results of Row, SB, SBN and LBP on object-style and scene-style classification

Datasets	Algorithms	Row		SB		SBN		LBP	
		Obj.	Scene	Obj.	Scene	Obj.	Scene	Obj.	Scene
1000-Img.	Citation-*k*NN	.831	.859	.830	.811	.829	.832	.500	.500
	miGraph	.871	.884	.931	.873	.917	.866	.951	.895
	MIBoosting	.877	.879	.938	.886	.933	.887	.857	.813
	miSVM	.839	.844	.812	.757	.833	.828	.725	.760
	MILES	.855	.875	.898	.861	.871	.853	.866	.871
	miFV	.844	.830	.892	.870	.905	.868	.882	.850
	MIForests	.855	.879	.901	.864	.875	.855	.869	.872
2000-Img.	Citation-*k*NN	.833	.859	.830	.829	.822	.838	.500	.500
	miGraph	.870	.873	.914	.892	.904	.884	.953	.875
	MIBoosting	.873	.877	.926	.902	.916	.891	.858	.834
	miSVM	.831	.845	.801	.803	.814	.833	.738	.739
	MILES	.846	.855	.887	.869	.867	.875	.879	.818
	miFV	.835	.852	.884	.864	.898	.883	.928	.861
	MIForests	.850	.855	.886	.869	.869	.876	.883	.819
MSRA	Citation-*k*NN	.815	.907	.823	.894	.851	.890	.503	.501
	miGraph	.863	.926	.937	.944	.932	.949	.895	.917
	MIBoosting	.887	.942	.912	.943	.921	.949	.864	.918
	miSVM	.824	.929	.834	.898	.839	.909	N/A	N/A
	MILES	.740	.814	.758	.815	.758	.789	N/A	N/A
	miFV	.812	.924	.828	.917	.887	.941	.858	.882
	MIForests	.739	.813	.758	.820	.761	.792	.828	.802
Pairwise *t* test		*h*	*p*	*h*	*p*	*h*	*p*	*h*	*p*
		1	0.005	0	0.998	0	0.783	0	0.727

On the top of the table, we report the accuracy rates of each combination of two style image classification. On the buttom, we present the pairwise t test between object- and scene-style classification. In pairwise t test, $h = 0$ indicates that the null hypothesis ("means are equal") cannot be rejected at the 5 % significance level. In addition, the p value indicates the validity of the null hypothesis. The larger the p value, the more valid the result is. *N/A* indicates that these combinations could not return a result in 7 days

Table 9 Ranks of scene-style classification results of each combination (bag generator + learning algorithm) on three collections, i.e., *1000-Image*, *2000-Image* and *MSRA*

Combination	1000-Image	2000-Image	MSRA	Average rank
MIBoost.+SB	4*	1$^{\bullet}$	4	3.0$^{\bullet}$
MIBoost.+SBN	3$^{\bullet}$	3	3	3.0*
miGra.+SB	11	2*	5	6.0
MIBoost.+Row	6	8	6	6.7
miGra.+SBN	20	4	2*	8.7
miGra.+Row	5	12	11	9.3
MIBoost.+Blob.	22	6	1$^{\bullet}$	9.7
miFV+SBN	18	5	7	10.0

The best performance in one column is followed by "$^{\bullet}$", and the second best one is followed by "*". "Average Rank" is the average value of all the ranks on these three data sets

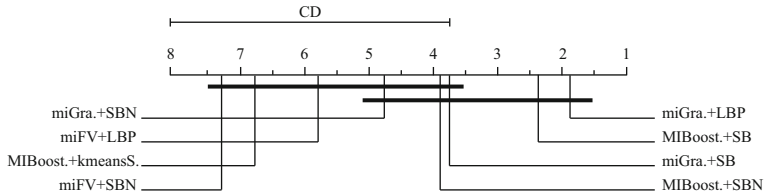

Fig. 9 Comparison of the top 8 combinations against each other with the Nemenyi test. Groups of combinations that are not significantly different (at $p = 0.05$) are connected

For Nemenyi test, we have $q_\alpha = 3.031$ at significance level $\alpha = 0.05$ and thus $CD = 4.2865$ (when $k = 8, N = 6$). To visually present the relative performance of these top eight combinations, Fig. 9 illustrates the CD diagrams (Demšar 2006). When comparing all the combinations against each other, we connect the groups of combinations that are not significantly different. We also show the critical difference above the figure. As shown in this figure, "*miGra.+LBP*" and "*MIBoost.+SB*" significantly outperform against the other combinations. However, the experimental data is not sufficient to reach any conclusion regarding "*miGra.+SB*", "*MIBoost.+SBN*", "*miGra.+SBN*" and "*miFV+LBP*", i.e., we cannot tell which group they belong to.

In consequence, we recommend "miGraph with LBP" and "MIBoosting with SB" as the best two combinations for image classification tasks.

5.2 Method observations

In this section, we will first report some findings about SB and SBN. And then, we present some interesting observations about the patch sizes of certain bag generators.

5.2.1 Similar performance phenomenon of SB and SBN

Regardless of whether we are considering Tables 4 or 5, we can easily find that the classification performances of SB and SBN are quite similar. Moreover, Fig. 10 presents the accuracy rates of SB and SBN on two image data collections, i.e., *2000-Image* and *MSRA*. Figure 10c and f demonstrate the difference-value (D-value) of SB and SBN on these two data sets, respectively. From the figures, we can easily find that SB and SBN perform quite similarly: The D-value of SB and SBN on *2000-Image* is not larger than 0.06; and the one on *MSRA* is smaller than 0.07. We also do the pairwise t test between SB and SBN, and the results are presented in Table 10: The performances of SB and SBN are not significantly different from each other. As illustrated in Sect. 5.2.1, SB abstracts instances without overlapping, while SBN is with overlapping by a cross shaped sliding window. However, why the performances of these two could be so similar? Here we focus on the overlapping in SBN.

Figure 11 shows the overlapping of SBN with the 16×16 and 64×64 patch size, respectively. The number of the color-bar stands for the number of overlapping. As shown in this figure, when the patch size increases, the overlapping regions become larger, and cover the whole image eventually. In consequence, the larger the overlapping of SBN is, the more redundancy exists among instances in the same bag. This explains why SB and SBN could have similar performances. Moreover, one instance's feature of SBN is the RGB values of each pixel in a 2×2 blob and the corresponding color difference with its 4 neighboring blobs. However, one SB instance is just the RGB values of each pixel in a 2×2 blob. The similar performance phenomenon also indicates that the difference with blob's neighbors in

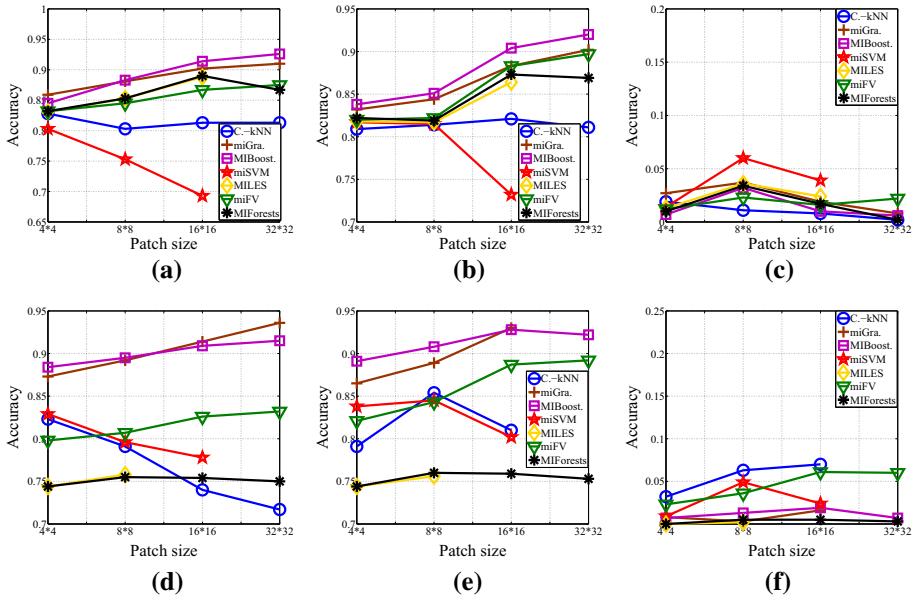

Fig. 10 Similar performances of SB and SBN on two image data collections. The figures in the *first row* are the results on *2000-Image*, and the ones in the *second row* are the results on *MSRA*. **a** and **d** are the classification accuracy figures of SB with different patch sizes; **b** and **e** are the ones of SBN. In addition, we present the difference-value (D-value) of SB and SBN on these two data sets, which are shown in **c** and **f**, respectively. The figures are best viewed in *color*. **a** SB on *2000-Img*. **b** SBN on *2000-Img*. **c** D-value on *2000-Img*. **d** SB on *MSRA*. **e** SBN on *MSRA*. **f** D-value on *MSRA* (Color figure online)

SBN might not be useful. Meanwhile, SBN usually produce much more instances than SB (cf. Table 3), which will cause a much larger computational cost. Therefore, it is better to choose SB to be bag generator, instead of SBN.

In addition, when the patch size is small, the blind zones in SBN account for a large proportion of the original image. However, when it increases, the blind zones will become negligible. Thus, if the key objects locate in the corner of images, using SBN as bag generator might not be a good choice; or it is more suitable to use SBN with a large patch size to abstract instances.

5.2.2 Observations about patch size

In our experiments, we vary different patch sizes for some bag generators. Here we report some findings about that. As shown in Fig. 12, the figures in each row represent the classification results of one bag generator combined with four different learning algorithms. For example, Fig. 12a shows the accuracy rates of the combination (i.e., "miGraph with SB") on the six image data sets. The horizontal axis is with different patch sizes (from left to right) in order of increasing instances' numbers. Moreover, Fig. 12b is the result of "MIBoosting with SB"; (c) is for "miFV with SB"; (d) is for "miSVM with SB".

If we treat miGraph, MIBoosting and miFV as an algorithm group, which do not obey the standard MIL assumption, we will find an interesting observation: when the number of instances increases, their classification accuracy rates will also increase. (Certainly, "miGraph with k-meansSeg" is a small exception.) This phenomenon supports our findings in Sect. 5.1.2 again. In other words, more instances are helpful to infer the relationship of instances in bags with the bag's label. In this way, more instances mean a more accurate relationship in the

Table 10 Pairwise t test between SB and SBN on two image data collections, i.e., *2000-Image* and *MSRA*

Datasets	Algorithms	SB_4	SBN_4	SB_8	SBN_8	SB_16	SBN_16	SB_32	SBN_32
2000-Img.	Citation-kNN	.828	.809	.803	.814	.813	.821	.813	.811
	miGraph	.859	.832	.881	.844	.902	.883	.910	.902
	MIBoosting	.845	.838	.883	.851	.914	.904	.926	.920
	miSVM	.803	.817	.753	.815	.693	.732	N/A	N/A
	MILES	.831	.818	.853	.817	.888	.864	N/A	N/A
	miFV	.832	.820	.845	.822	.867	.883	.875	.897
	MIForests	.832	.822	.853	.819	.890	.873	.867	.869
MSRA	Citation-kNN	.823	.791	.791	.854	.740	.810	.717	N/A
	miGraph	.873	.865	.892	.889	.914	.930	.936	N/A
	MIBoosting	.884	.891	.895	.908	.909	.928	.915	.922
	miSVM	.829	.838	.796	.845	.778	.802	N/A	N/A
	MILES	.744	.744	.758	.756	N/A	N/A	N/A	N/A
	miFV	.798	.821	.807	.843	.826	.887	.832	.892
	MIForests	.744	.744	.755	.760	.754	.759	.750	.753
Pairwise t test		h	p	h	p	h	p	h	p
		0	0.727	0	0.775	0	0.600	0	0.617

On the top of the table, we report the accuracy rates of each combination. On the buttom, we present the pairwise t test between SB and SBN of different patch sizes. In pairwise t test, $h = 0$ indicates that the null hypothesis ("means are equal") cannot be rejected at the 5 % significance level, which also shows SB and SBN are not siginificantly different from each other. In addition, the p value indicates the validity of the null hypothesis. *N/A* indicates that these combinations could not return a result in 7 days

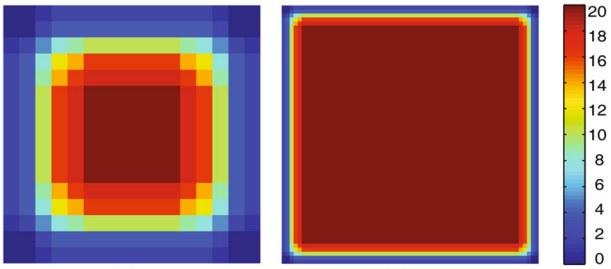

Fig. 11 Overlappings of the SBN bag generator with the **a** 16 × 16 and **b** 64 × 64 patch size. The *warm color* represents the number of overlappings is *large*, while the *cool color* represents the one is *small*. Note that, the maximum number of the overlapping is 20. The minimum one is 0, which indicates there are *blind zones* in the four corners of this image. The figures are best viewed in *color* (Color figure online)

bag, which is good for miGraph etc. to build a better MIL classifier. On the other hand, for miSVM, which obeys the standard MIL assumption, fewer instances yield a more accurate MIL classifier, especially for image related tasks. But for the other MIL algorithms obey the standard MIL assumption, i.e., Citation-kNN and MILES, their results with instances increasing show no apparent pattern. In addition, the figures in the first three columns of SB and SBN in Fig. 12 also demonstrate the effect of dense sampling. Because SB and SBN with larger patch size will extract more instances (image regions), which indicates more dense sampling from original images.

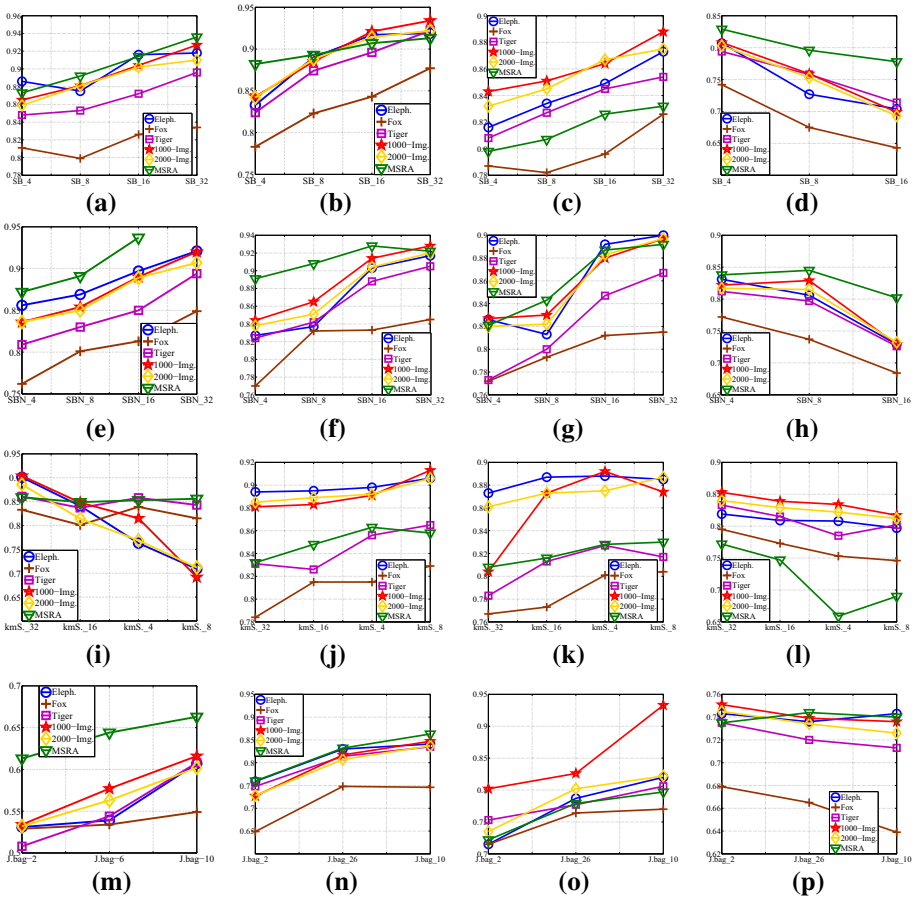

Fig. 12 Accuracy rates figures of combinations (i.e., miGraph, MIBoosting, miFV and miSVM with SB, SBN, kmeansSeg and JSEG-bag) on the six image data sets. Note that, the *vertival axis* is the averagy accuracy rates, and the *horizontal* one is different patch sizes in order of increasing instances' numbers. The *different marks* stand for different image data sets. There also lack of some combinations' results, e.g., "miSVM with SB in the 32×32 patch size", which caused by its large computational cost. In addition, "kmS." is short for the *k*-meansSeg bag generator; "J.-bag" is "JSEG-bag"; "miGra." is "miGraph"; and "MIBoost." is "MIBoosting". The figures are best viewed in color. **a** miGra.+SB. **b** MIBoost.+SB. **c** miFV+SB. **d** miSVM+SB. **e** miGra.+SBN. **f** MIBoost.+SBN. **g** miFV+SBN. **h** miSVM+SBN. **i** miGra.+kmS. **j** MIBoost.+kmS. **k** miFV+kmS. **l** miSVM+kmS. **m** miGra.+J.-bag. **n** MIBoost.+J.-bag. **o** miFV+J.-bag. **p** miSVM+J.-bag (Color figure online)

6 Summary of findings

In this paper, we have presented an empirical study on image bag generators for multi-instance learning. Our main findings are summarized as follows.

- SB, SBN and LBP outperform other bag generators in most cases, which indicates that sampling dense regions to construct instances will provide better classification performance. It is also consistent with the conclusion in the computer vision community (Li and Perona 2005; Nowak et al. 2006). In the future, it is better to incorporate dense sampling into new image bag generators.

- The assumptions adopted by different MIL algorithms critically determine their performances on many tasks. The miGraph, MIBoosting and miFV algorithms that assume non-i.i.d. instances or take advantage of aggregating properties of bags work well on image classification tasks, whereas algorithms adopt the standard MIL assumption do not. Therefore, in the future, it is preferable to design new learning algorithms by considering the nature of the relationship between instances in MIL bags for image related tasks.
- The performances of SB and SBN are quite similar. However, SBN will lead to a larger number of instances and larger time cost than SB. In practice, it is a better choice to select SB as the bag generator, instead of SBN.
- For different image classification tasks, such as object classification and scene classification, different kinds of instances' features are the key point of classification accuracy. For example, if the task is scene classification, bag generators which contain color features will have satisfactory accuracy rates, while the ones containing texture features might be unsatisfactory.
- There are interesting observations about several combinations. For miGraph, MIBoosting or miFV, when they are combined with SB, SBN, *k*-meansSeg or JSEG-bag, along with the number of instances increasing, their classification accuracy also increases, while miSVM has the opposite behavior. These observations not only support the second finding, they also demonstrate the effect of dense sampling.
- There are several recommended combinations for practical applications. "miGraph with LBP" and "MIBoosting with SB" are two of the best combinations for image classification (cf. Fig. 9).

7 Conclusions

Multi-instance learning has achieved great success in applications with complicated objects such as image categorization. While most research interests focus on designing MIL algorithms, bag generators are rarely studied. In this paper, we provide an empirical study with thousands of configurations on state-of-the-art image bag generators. From these empirical results, we make some interesting observations that are helpful for both image classification and multi-instance learning. In the future, better image bag generators or MIL algorithms might be designed based on the experimental observations. We also believe similar studies could be made in other MIL applications, e.g., text bag generators for text categorization tasks.

Acknowledgments The authors want to thank the associate editor and anonymous reviewers for their helpful comments and suggestions. This research was supported by the National Science Foundation of China (61333014, 61321491), Tencent Fund, and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

Appendix: Detailed experimental results

Section I: The detailed averagy accuracy rates of each combination (bag generator + learning algorithm) on *Elephant*, *Fox*, *Tiger*, *1000-Image*, *2000-Image* and *MSRA*

See Tables 11, 12, 13, 14, 15 and 16.

Table 11 Detailed averagy accuracy rates of each combination on the *Elephant* data set

Elephant	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	.842 ± .081	.865 ± .071	.845 ± .068	.809 ± .091	.822 ± .099	.828 ± .083	.825 ± .068
Row_8	.810 ± .084	.860 ± .054	.844 ± .085	.786 ± .069	.819 ± .080	.816 ± .105	.824 ± .087
Row_16	.800 ± .090	.841 ± .061	.854 ± .076	.724 ± .094	.817 ± .096	.817 ± .075	.816 ± .077
Row_32	.789 ± .089	.863 ± .063	.856 ± .069	.763 ± .057	.841 ± .085	.770 ± .094	.841 ± .073
SB_4	.838 ± .079	.886 ± .056	.829 ± .086	.807 ± .056	.833 ± .079	.834 ± .098	.834 ± .086
SB_8	.804 ± .086	.875 ± .080	.890 ± .075	.727 ± .096	.859 ± .074	.816 ± .078	.860 ± .075
SB_16	.833 ± .063	.916 ± .060	.915 ± .056	.704 ± .054	.902 ± .059	.849 ± .068	.896 ± .057
SB_32	.815 ± .070	.918 ± .051	.920 ± .052	N/A	N/A	.873 ± .062	.901 ± .069
SBN_4	.791 ± .117	.850 ± .082	.827 ± .080	.831 ± .093	.823 ± .073	.826 ± .101	.828 ± .082
SBN_8	.808 ± .101	.866 ± .076	.837 ± .069	.806 ± .090	.813 ± .073	.803 ± .085	.813 ± .069
SBN_16	.841 ± .067	.896 ± .064	.903 ± .058	.729 ± .070	.868 ± .062	.892 ± .060	.873 ± .056
SBN_32	.826 ± .060	.912 ± .057	.917 ± .055	N/A	N/A	.900 ± .054	.874 ± .053
kmeansS._4	.868 ± .080	.756 ± .202	.906 ± .060	.808 ± .040	.820 ± .076	.888 ± .051	.825 ± .061
kmeansS._8	.863 ± .073	.705 ± .184	.898 ± .057	.797 ± .053	.819 ± .087	.885 ± .049	.823 ± .056
kmeansS._16	.860 ± .074	.835 ± .149	.895 ± .052	.809 ± .054	.822 ± .081	.887 ± .051	.829 ± .051
kmeansS._32	.849 ± .065	.895 ± .042	.894 ± .049	.819 ± .058	.813 ± .092	.873 ± .056	.816 ± .048
Blob.	.500 ± .000	.574 ± .062	.846 ± .082	.845 ± .079	.778 ± .097	.818 ± .089	.780 ± .093
WavS.	.818 ± .071	.844 ± .060	.802 ± .079	.842 ± .068	.719 ± .086	.866 ± .053	.725 ± .051
J.-bag_2	.774 ± .081	.526 ± .025	.759 ± .096	.743 ± .110	.556 ± .060	.715 ± .052	.557 ± .096
J.-bag_6	.788 ± .083	.534 ± .050	.830 ± .101	.736 ± .092	.588 ± .073	.787 ± .109	.587 ± .101
J.-bag_10	.789 ± .077	.597 ± .049	.841 ± .096	.743 ± .101	.612 ± .063	.820 ± .079	.615 ± .076
LBP	.500 ± .000	.930 ± .041	.837 ± .109	.759 ± .049	.837 ± .083	.902 ± .047	.846 ± .050
SIFT	.685 ± .068	.527 ± .054	.741 ± .103	.584 ± .044	.533 ± .061	.847 ± .075	.532 ± .080

N/A indicates these combinations can not return results in 7 days

Table 12 Detailed averagy accuracy rates of each combination on the *Fox* data set

Fox	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	.778 ± .085	.794 ± .089	.780 ± .106	.789 ± .078	.761 ± .090	.789 ± .089	.763 ± .087
Row_8	.767 ± .082	.786 ± .064	.803 ± .079	.762 ± .088	.774 ± .086	.775 ± .078	.775 ± .063
Row_16	.745 ± .088	.803 ± .075	.808 ± .081	.723 ± .115	.770 ± .079	.784 ± .085	.773 ± .078
Row_32	.733 ± .087	.788 ± .082	.800 ± .076	.729 ± .083	.789 ± .077	.764 ± .087	.790 ± .089
SB_4	.765 ± .071	.811 ± .080	.783 ± .096	.742 ± .080	.757 ± .105	.787 ± .084	.761 ± .079
SB_8	.701 ± .082	.799 ± .090	.824 ± .088	.675 ± .074	.769 ± .101	.782 ± .096	.772 ± .088
SB_16	.749 ± .077	.826 ± .087	.846 ± .074	.643 ± .067	.819 ± .082	.796 ± .095	.823 ± .098
SB_32	.771 ± .072	.834 ± .099	.882 ± .076	N/A	N/A	.826 ± .098	.822 ± .099
SBN_4	.730 ± .105	.761 ± .091	.770 ± .091	.772 ± .093	.756 ± .093	.772 ± .081	.756 ± .090
SBN_8	.749 ± .089	.793 ± .081	.832 ± .076	.737 ± .079	.778 ± .071	.793 ± .066	.778 ± .083
SBN_16	.742 ± .086	.808 ± .079	.833 ± .077	.684 ± .072	.797 ± .079	.812 ± .079	.791 ± .079
SBN_32	.733 ± .075	.846 ± .088	.845 ± .080	N/A	N/A	.815 ± .075	.802 ± .073
kmeansS._4	.799 ± .094	.833 ± .081	.815 ± .122	.753 ± .092	.727 ± .099	.804 ± .092	.730 ± .083
kmeansS._8	.775 ± .096	.810 ± .096	.829 ± .098	.745 ± .080	.723 ± .099	.801 ± .100	.722 ± .097
kmeansS._16	.776 ± .102	.799 ± .090	.815 ± .094	.766 ± .083	.735 ± .092	.773 ± .097	.740 ± .090
kmeansS._32	.797 ± .090	.832 ± .089	.784 ± .117	.773 ± .089	.724 ± .104	.767 ± .097	.727 ± .089
Blob.	.500 ± .000	.553 ± .035	.824 ± .077	.795 ± .078	.686 ± .076	.803 ± .083	.686 ± .082
WavS.	.750 ± .080	.750 ± .074	.758 ± .074	.799 ± .085	.639 ± .085	.816 ± .084	.641 ± .081
J.-bag_2	.709 ± .064	.523 ± .035	.649 ± .117	.679 ± .094	.552 ± .041	.715 ± .088	.555 ± .034
J.-bag_6	.723 ± .081	.529 ± .047	.748 ± .109	.665 ± .109	.568 ± .049	.764 ± .120	.572 ± .048
J.-bag_10	.723 ± .082	.546 ± .062	.746 ± .150	.639 ± .122	.577 ± .055	.770 ± .108	.575 ± .099
LBP	.500 ± .000	.894 ± .083	.810 ± .101	.648 ± .102	.830 ± .105	.885 ± .087	.828 ± .087
SIFT	.689 ± .058	.528 ± .052	.714 ± .112	.558 ± .045	.529 ± .051	.817 ± .088	.527 ± .092

N/A indicates these combinations can not return results in 7 days

Table 13 Detailed averagy accuracy rates of each combination on the *Tiger* data set

Tiger	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	.826 ± .076	.852 ± .072	.844 ± .094	.820 ± .069	.819 ± .086	.828 ± .087	.820 ± .067
Row_8	.813 ± .079	.827 ± .078	.846 ± .096	.797 ± .081	.814 ± .081	.807 ± .077	.817 ± .079
Row_16	.814 ± .095	.846 ± .071	.867 ± .083	.798 ± .081	.826 ± .073	.807 ± .071	.830 ± .084
Row_32	.803 ± .101	.834 ± .085	.863 ± .086	.781 ± .077	.816 ± .081	.812 ± .077	.819 ± .084
SB_4	.794 ± .097	.848 ± .072	.822 ± .084	.794 ± .073	.792 ± .096	.808 ± .100	.795 ± .073
SB_8	.765 ± .098	.853 ± .090	.877 ± .063	.758 ± .068	.826 ± .098	.827 ± .103	.828 ± .089
SB_16	.810 ± .089	.872 ± .063	.898 ± .061	.714 ± .034	.859 ± .066	.845 ± .071	.861 ± .063
SB_32	.805 ± .070	.896 ± .062	.927 ± .041	N/A	N/A	.854 ± .072	.847 ± .066
SBN_4	.761 ± .109	.806 ± .097	.824 ± .102	.797 ± .091	.775 ± .103	.773 ± .090	.775 ± .098
SBN_8	.775 ± .092	.828 ± .073	.842 ± .073	.812 ± .062	.739 ± .098	.800 ± .078	.742 ± .072
SBN_16	.799 ± .086	.844 ± .084	.888 ± .057	.726 ± .051	.800 ± .091	.847 ± .071	.799 ± .052
SBN_32	.812 ± .081	.891 ± .060	.905 ± .049	N/A	N/A	.867 ± .063	.826 ± .056
kmeansS._4	.840 ± .064	.851 ± .047	.856 ± .070	.785 ± .059	.779 ± .102	.827 ± .060	.781 ± .063
kmeansS._8	.807 ± .070	.839 ± .065	.865 ± .068	.802 ± .059	.750 ± .121	.817 ± .053	.749 ± .063
kmeansS._16	.813 ± .082	.836 ± .075	.826 ± .075	.815 ± .061	.756 ± .083	.813 ± .060	.760 ± .075
kmeansS._32	.805 ± .079	.860 ± .063	.831 ± .079	.805 ± .065	.735 ± .099	.783 ± .067	.739 ± .062
Blob.	.500 ± .000	.549 ± .042	.888 ± .053	.833 ± .054	.744 ± .072	.825 ± .052	.742 ± .050
WavS.	.844 ± .062	.802 ± .061	.846 ± .053	.844 ± .047	.664 ± .101	.890 ± .051	.662 ± .055
J.-bag_2	.758 ± .075	.505 ± .025	.748 ± .087	.735 ± .071	.573 ± .024	.753 ± .075	.578 ± .024
J.-bag_6	.789 ± .081	.543 ± .032	.814 ± .109	.720 ± .071	.600 ± .035	.777 ± .093	.598 ± .032
J.-bag_10	.781 ± .095	.603 ± .042	.835 ± .092	.713 ± .068	.596 ± .036	.806 ± .076	.606 ± .081
LBP	.500 ± .000	.912 ± .053	.830 ± .104	.693 ± .098	.833 ± .089	.879 ± .070	.836 ± .067
SIFT	.685 ± .061	.526 ± .049	.758 ± .112	.562 ± .034	.534 ± .042	.842 ± .099	.534 ± .091

N/A indicates these combinations can not return results in 7 days

Table 14 Detailed averagy accuracy rates of each combination on the *1000-Image* data collection

1000-Image	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	.833 ± .081	.849 ± .066	.852 ± .071	.837 ± .067	.823 ± .081	.836 ± .067	.823 ± .072
Row_8	.829 ± .840	.858 ± .063	.874 ± .055	.812 ± .055	.836 ± .076	.848 ± .079	.837 ± .053
Row_16	.824 ± .077	.868 ± .059	.874 ± .059	.801 ± .082	.855 ± .073	.834 ± .074	.859 ± .071
Row_32	.815 ± .081	.874 ± .064	.875 ± .062	.786 ± .058	.850 ± .066	.821 ± .074	.849 ± .059
SB_4	.829 ± .082	.865 ± .067	.845 ± .075	.808 ± .056	.829 ± .079	.843 ± .077	.832 ± .077
SB_8	.788 ± .077	.881 ± .066	.886 ± .073	.759 ± .096	.862 ± .078	.851 ± .065	.865 ± .071
SB_16	.794 ± .080	.904 ± .054	.927 ± .046	.699 ± .057	.894 ± .058	.864 ± .067	.897 ± .058
SB_32	.802 ± .065	.927 ± .048	.935 ± .049	N/A	N/A	.888 ± .053	.880 ± .047
SBN_4	.823 ± .085	.830 ± .068	.844 ± .069	.822 ± .086	.817 ± .084	.827 ± .075	.818 ± .069
SBN_8	.825 ± .089	.854 ± .079	.865 ± .069	.829 ± .076	.816 ± .089	.830 ± .068	.817 ± .068
SBN_16	.833 ± .081	.887 ± .074	.914 ± .053	.729 ± .057	.876 ± .080	.880 ± .071	.881 ± .062
SBN_32	.803 ± .069	.917 ± .052	.928 ± .052	N/A	N/A	.897 ± .056	.843 ± .051
kmeansS._4	.865 ± .071	.808 ± .171	.891 ± .070	.834 ± .060	.833 ± .077	.892 ± .070	.835 ± .069
kmeansS._8	.869 ± .071	.688 ± .186	.913 ± .071	.817 ± .079	.837 ± .085	.872 ± .073	.843 ± .068
kmeansS._16	.861 ± .080	.847 ± .136	.883 ± .073	.839 ± .069	.812 ± .090	.873 ± .069	.814 ± .073
kmeansS._32	.867 ± .069	.900 ± .061	.881 ± .091	.853 ± .082	.813 ± .094	.804 ± .085	.813 ± .093
Blob.	.500 ± .000	.584 ± .049	.891 ± .055	.810 ± .071	.771 ± .071	.868 ± .059	.778 ± .057
WavS.	.844 ± .064	.843 ± .058	.827 ± .064	.870 ± .060	.693 ± .085	.732 ± .072	.690 ± .074
J.-bag_2	.753 ± .070	.530 ± .032	.727 ± .111	.751 ± .073	.576 ± .043	.802 ± .079	.577 ± .110
J.-bag_6	.771 ± .071	.576 ± .054	.817 ± .094	.739 ± .099	.612 ± .054	.826 ± .078	.615 ± .094
J.-bag_10	.776 ± .089	.613 ± .053	.847 ± .082	.736 ± .124	.630 ± .060	.933 ± .048	.636 ± .048
LBP	.500 ± .000	.954 ± .049	.855 ± .100	.725 ± .091	.868 ± .073	.877 ± .070	.871 ± .072
SIFT	.755 ± .093	.543 ± .044	.800 ± .095	.569 ± .056	.534 ± .046	.848 ± .072	.541 ± .066

N/A indicates these combinations can not return results in 7 days

Table 15 Detailed averagy accuracy rates of each combination on the *2000-Image* data collection

2000-Image	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	.832 ± .088	.852 ± .070	.849 ± .076	.832 ± .081	.828 ± .087	.832 ± .072	.827 ± .071
Row_8	.828 ± .079	.856 ± .073	.866 ± .075	.815 ± .067	.839 ± .082	.833 ± .076	.844 ± .075
Row_16	.820 ± .083	.862 ± .067	.870 ± .071	.798 ± .099	.844 ± .086	.828 ± .075	.847 ± .076
Row_32	.815 ± .084	.865 ± .070	.864 ± .069	.805 ± .071	.845 ± .089	.813 ± .075	.850 ± .078
SB_4	.828 ± .083	.859 ± .072	.845 ± .075	.803 ± .082	.831 ± .088	.832 ± .087	.832 ± .072
SB_8	.803 ± .082	.881 ± .067	.883 ± .064	.753 ± .105	.853 ± .077	.845 ± .073	.853 ± .067
SB_16	.813 ± .084	.902 ± .062	.914 ± .054	.693 ± .065	.888 ± .068	.867 ± .067	.890 ± .064
SB_32	.813 ± .077	.910 ± .063	.926 ± .060	N/A	N/A	.875 ± .062	.867 ± .063
SBN_4	.809 ± .094	.832 ± .079	.838 ± .073	.817 ± .089	.818 ± .089	.820 ± .079	.822 ± .077
SBN_8	.814 ± .081	.844 ± .077	.851 ± .063	.815 ± .088	.817 ± .089	.822 ± .074	.819 ± .078
SBN_16	.821 ± .084	.883 ± .068	.904 ± .063	.732 ± .058	.864 ± .072	.883 ± .061	.873 ± .067
SBN_32	.811 ± .070	.902 ± .062	.920 ± .059	N/A	N/A	.897 ± .063	.869 ± .063
kmeansS._4	.867 ± .074	.769 ± .180	.892 ± .070	.822 ± .067	.809 ± .095	.875 ± .063	.809 ± .179
kmeansS._8	.865 ± .071	.707 ± .184	.906 ± .068	.812 ± .073	.811 ± .091	.887 ± .063	.809 ± .185
kmeansS._16	.859 ± .075	.812 ± .154	.889 ± .079	.829 ± .088	.808 ± .094	.873 ± .066	.810 ± .062
kmeansS._32	.859 ± .071	.886 ± .084	.885 ± .083	.840 ± .086	.803 ± .095	.861 ± .068	.807 ± .089
Blob.	.500 ± .000	.558 ± .058	.891 ± .065	.819 ± .063	.748 ± .082	.808 ± .071	.750 ± .073
WavS.	.838 ± .075	.830 ± .098	.828 ± .070	.867 ± .062	.722 ± .106	.875 ± .061	.728 ± .061
J.-bag_2	.751 ± .080	.530 ± .033	.728 ± .111	.745 ± .089	.566 ± .048	.735 ± .069	.567 ± .034
J.-bag_6	.776 ± .088	.557 ± .052	.807 ± .103	.734 ± .111	.607 ± .056	.802 ± .081	.612 ± .052
J.-bag_10	.767 ± .093	.602 ± .047	.837 ± .101	.726 ± .125	.619 ± .061	.822 ± .080	.622 ± .077
LBP	.500 ± .000	.938 ± .048	.856 ± .097	.734 ± .086	.872 ± .079	.924 ± .058	.877 ± .057
SIFT	.722 ± .099	.543 ± .051	.807 ± .093	.572 ± .048	.538 ± .049	.874 ± .077	.542 ± .078

N/A indicates these combinations can not return results in 7 days

Table 16 Detailed averagy accuracy rates of each combination on the *MSRA* data collection

MSRA	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	.818 ± .056	.848 ± .050	.865 ± .058	.825 ± .066	.732 ± .043	.805 ± .064	.735 ± .058
Row_8	.824 ± .065	.863 ± .047	.887 ± .054	.805 ± .072	.739 ± .044	.818 ± .060	.741 ± .059
Row_16	.796 ± .065	.860 ± .051	.882 ± .052	.774 ± .087	.743 ± .044	.815 ± .060	.746 ± .075
Row_32	.785 ± .073	.860 ± .046	.881 ± .046	.745 ± .098	.740 ± .041	.803 ± .062	.744 ± .045
SB_4	.823 ± .054	.873 ± .047	.884 ± .054	.829 ± .053	.744 ± .044	.798 ± .066	.744 ± .059
SB_8	.791 ± .065	.892 ± .047	.895 ± .053	.796 ± .056	.758 ± .049	.807 ± .059	.755 ± .064
SB_16	.740 ± .064	.914 ± .043	.909 ± .042	.778 ± .071	N/A	.826 ± .059	.754 ± .043
SB_32	.717 ± .070	.936 ± .036	.915 ± .045	N/A	N/A	.832 ± .063	.750 ± .049
SBN_4	.791 ± .087	.865 ± .042	.891 ± .042	.838 ± .055	.744 ± .037	.821 ± .055	.744 ± .046
SBN_8	.854 ± .057	.889 ± .046	.908 ± .035	.845 ± .054	.756 ± .038	.843 ± .053	.760 ± .062
SBN_16	.810 ± .047	.930 ± .030	.928 ± .035	.802 ± .039	N/A	.887 ± .041	.759 ± .039
SBN_32	N/A	N/A	.922 ± .032	N/A	N/A	.892 ± .046	.753 ± .039
kmeansS._4	.730 ± .080	.846 ± .052	.863 ± .070	.659 ± .115	.739 ± .049	.830 ± .062	.730 ± .070
kmeansS._8	.741 ± .077	.852 ± .050	.858 ± .070	.690 ± .104	.738 ± .045	.828 ± .058	.731 ± .072
kmeansS._16	.737 ± .118	.845 ± .055	.848 ± .072	.747 ± .077	.732 ± .046	.816 ± .061	.737 ± .061
kmeansS._32	.797 ± .073	.858 ± .053	.832 ± .067	.772 ± .072	.721 ± .048	.808 ± .063	.723 ± .067
Blob.	.500 ± .000	.682 ± .052	.929 ± .045	.816 ± .069	.707 ± .048	.868 ± .054	.710 ± .070
WavS.	.776 ± .088	.841 ± .069	.893 ± .053	.834 ± .072	.655 ± .074	.847 ± .076	.659 ± .059
J.-bag_2	.751 ± .066	.611 ± .032	.760 ± .077	.735 ± .090	.582 ± .091	.722 ± .073	.584 ± .077
J.-bag_6	.799 ± .072	.640 ± .035	.832 ± .086	.744 ± .092	.633 ± .067	.779 ± .068	.635 ± .087
J.-bag_10	.799 ± .071	.656 ± .053	.863 ± .072	.740 ± .134	.651 ± .062	.797 ± .070	.655 ± .082
LBP	.509 ± .004	.892 ± .056	.868 ± .061	N/A	N/A	.866 ± .068	.515 ± .068
SIFT	.665 ± .064	.660 ± .083	.843 ± .065	N/A	N/A	.820 ± .072	.506 ± .790

N/A indicates these combinations can not return results in 7 days

Section II: The corresponding optimal parameters for each combination on the *2000-Image* and *MSRA* data collections

Here we introduce the parameters of each learning algorithm, followed by the corresponding optimal parameters for different combinations on these two data collections. The implementations of Citation-kNN and miSVM are by using the Multiple Instance Learning Toolbox.[7] The implementation of MIBoosting is by using the WEKA data mining software.[8] The other implementations of learning algorithms and bag generators are all from the authors. All the parameters were selected according to a two times two-fold cross validation on the training set. Because we repeat the experiments three times with different training/test data splitting, we also select the optimal parameters three times on the corresponding training sets. Tables 17, 18, and 19 show the optimal parameters of three splittings on *2000-Image*. Tables 20, 21, and 22 show the ones on *MSRA*.

- *Citation-kNN*: There are three parameters in this MIL algorithm, i.e., "R", "C" and "H". The "R" indicates the number of nearest references, and "C" is the number of nearest citers, and "H" is rank of the Hausdorff distance. We chose R from 1 to 5 with step size 2, and C from 1 to 3 with step size 2, and H from 1 to 2.
- *miGraph*: The main parameters of miGraph are "γ" and "thr". For the "c" which used in SVM, we fixed it as 100. In miGraph, "γ" is the parameter of the RBF kernel, and "thr" is the threshold used in computing the weight of each instance. We chose thr from 0.1 to 0.9 with step size 0.2. For γ, its value is chosen from the set of {1.25, 2.5, 5, 10, 20}.
- *MIBoosting*: In MIBoosting, we use the pruned "J48" as the base learner, and the other parameters of "J48" is the default parameters in Weka. For the maximum number of boost iterations N, we chose it from the set of {10, 50, 100}.
- *miSVM* has three main parameters, i.e., "C", "K" and "E" (or "G"). "C" is the parameter in SVM. "K" indicates the kernel function, i.e, the polynomial kernel or the RBF kernel. If we use the polynomial kernel, the third parameter in miSVM is "E" which is the degree of polynomial; if the RBF kernel is used, the third one is "G" which is the γ in the RBF kernel. We chose C as 1, 50 or 100. For E, its values is from 1 to 3 with step size 1. For G, we chose its value as 0.1, 0.5 or 1.
- *MILES*: Three parameters σ^2, λ and μ need to be specified for MILES. We fixed $\mu = 0.5$ as the authors do in Chen et al. (2006), which penalized equally on errors in the positive class and the negative class. For σ^2, we chose its value from 5 to 15 with step size 2. For λ, we chose from 0.1 to 0.6 with step size 0.1.
- *miFV*: There are two main parameters in this learning algorithm. One of them is the number of Gaussian components K in GMM, the other is the PCA energy (as noted by "PE"), which reflects how much information is left after using PCA. We chose K from 1 to 5 with step size 1, and PE from 0.8 to 1 with step size 0.1.
- *MIForests*: The main parameters of MIForests are the number of trees "N" and the depth of tree "d". For "N", its value is chosen from the set of {10, 50, 100}, and for "d", it is from {10, 20, 30}. In addition, we followed the cooling schedule described in Leistner et al. (2010).

[7] The Multiple Instance Learning Toolbox is available at http://prlab.tudelft.nl/david-tax/mil.html.

[8] The WEKA software is available at http://sourceforge.net/projects/weka/files/weka-3-6/3.6.12/weka-3-6-12.zip/download.

Table 17 The optimal parameters of the first splitting on the *2000-Image* data collection

2000-Image	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	$H=2, R=3, C=3$	$\gamma=20, thr=0.5$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=3, PE=1$	$N=50, d=20$
Row_8	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.5$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=3, PE=1$	$N=50, d=20$
Row_16	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.9$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=30$
Row_32	$H=2, R=5, C=3$	$\gamma=20, thr=0.1$	$N=100$	$C=100$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=3, PE=1$	$N=50, d=20$
SB_4	$H=2, R=5, C=3$	$\gamma=5, thr=0.1$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.3$	$K=1, PE=1$	$N=100, d=20$
SB_8	$H=1, R=5, C=3$	$\gamma=20, thr=0.1$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.2$	$K=2, PE=1$	$N=50, d=20$
SB_16	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $E=0.5$	$\sigma^2=7, \lambda=0.1$	$K=3, PE=1$	$N=50, d=20$
SB_32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	N/A	N/A	$K=5, PE=1$	$N=100, d=20$
SBN_4	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.3$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=1, PE=1$	$N=50, d=30$
SBN_8	$H=2, R=3, C=3$	$\gamma=5, thr=0.5$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=7, \lambda=0.1$	$K=2, PE=1$	$N=50, d=30$
SBN_16	$H=2, R=5, C=3$	$\gamma=5, thr=0.5$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=50, d=20$
SBN_32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	N/A	N/A	$K=3, PE=1$	$N=50, d=20$
kmeansS._4	$H=2, R=5, C=3$	$\gamma=5, thr=0.7$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=1, PE=1$	$N=100, d=20$
kmeansS._8	$H=2, R=3, C=5$	$\gamma=10, thr=0.7$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=30$
kmeansS._16	$H=2, R=3, C=5$	$\gamma=20, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=100, d=20$
kmeansS._32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=3, PE=1$	$N=100, d=30$
Blob.	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.7$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.4$	$K=3, PE=1$	$N=50, d=20$
WavS.	$H=1, R=5, C=3$	$\gamma=5, thr=0.5$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=1, PE=1$	$N=50, d=30$
J.-bag_2	$H=1, R=3, C=3$	$\gamma=1.25, thr=0.3$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.1$	$K=4, PE=1$	$N=50, d=20$
J.-bag_6	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.3$	$N=100$	$C=100$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.1$	$K=4, PE=1$	$N=50, d=20$
J.-bag_10	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.6$	$K=5, PE=1$	$N=50, d=20$
LBP	$H=2, R=5, C=3$	$\gamma=10, thr=0.9$	$N=50$	$C=100$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=30$
SIFT	$H=2, R=5, C=3$	$\gamma=5, thr=0.5$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.6$	$K=1, PE=0.8$	$N=50, d=20$

"N/A" indicates the combination is very time-consuming. In miSVM, "PolyK" stands for the polynomial kernel, while "RBFK" is short for the RBF kernel

Table 18 The optimal parameters of the second splitting on the *2000-Image* data collection

2000-Image	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	$H=1, R=3, C=3$	$\gamma=5, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=30$
Row_8	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=3, PE=1$	$N=50, d=20$
Row_16	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=100$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=3, PE=1$	$N=50, d=20$
Row_32	$H=2, R=5, C=3$	$\gamma=20, thr=0.1$	$N=100$	$C=100$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=3, PE=1$	$N=50, d=20$
SB_4	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.9$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=7, \lambda=0.1$	$K=1, PE=1$	$N=100, d=20$
SB_8	$H=2, R=5, C=3$	$\gamma=20, thr=0.1$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.2$	$K=2, PE=1$	$N=50, d=30$
SB_16	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'RBFK', $E=0.5$	$\sigma^2=7, \lambda=0.1$	$K=2, PE=1$	$N=50, d=20$
SB_32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	N/A	N/A	$K=4, PE=1$	$N=50, d=20$
SBN_4	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=30$
SBN_8	$H=2, R=3, C=3$	$\gamma=5, thr=0.5$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=7, \lambda=0.1$	$K=2, PE=1$	$N=50, d=30$
SBN_16	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.7$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=50, d=20$
SBN_32	$H=2, R=3, C=5$	$\gamma=5, thr=0.9$	$N=50$	N/A	N/A	$K=3, PE=1$	$N=50, d=20$
kmeansS._4	$H=1, R=5, C=3$	$\gamma=5, thr=0.7$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=7, \lambda=0.1$	$K=1, PE=1$	$N=50, d=30$
kmeansS._8	$H=2, R=3, C=5$	$\gamma=10, thr=0.5$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=100, d=20$
kmeansS._16	$H=2, R=3, C=3$	$\gamma=10, thr=0.7$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.2$	$K=4, PE=1$	$N=100, d=20$
kmeansS._32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=20$
Blob.	$H=2, R=5, C=3$	$\gamma=5, thr=0.7$	$N=50$	$C=100$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.3$	$K=2, PE=1$	$N=50, d=20$
WavS.	$H=1, R=5, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.2$	$K=1, PE=1$	$N=50, d=20$
J.-bag_2	$H=2, R=3, C=3$	$\gamma=1.25, thr=0.1$	$N=50$	$C=100$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.3$	$K=3, PE=1$	$N=50, d=30$
J.-bag_6	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.3$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.1$	$K=4, PE=1$	$N=50, d=30$
J.-bag_10	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.3$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.6$	$K=5, PE=1$	$N=50, d=20$
LBP	$H=2, R=5, C=3$	$\gamma=10, thr=0.7$	$N=50$	$C=100$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.3$	$K=2, PE=1$	$N=100, d=20$
SIFT	$H=2, R=3, C=3$	$\gamma=5, thr=0.7$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.5$	$K=2, PE=0.8$	$N=50, d=20$

"N/A" indicates the combination is very time-consuming. In miSVM, "PolyK" stands for the polynomial kernel, while "RBFK" is short for the RBF kernel

Table 19 The optimal parameters of the third splitting on the *2000-Image* data collection

2000-Image	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	$H=2, R=5, C=3$	$\gamma=20, thr=0.9$	$N=100$	$C=100$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.2$	$K=2, PE=1$	$N=50, d=20$
Row_8	$H=2, R=5, C=3$	$\gamma=5, thr=0.5$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.2$	$K=2, PE=1$	$N=50, d=30$
Row_16	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.2$	$K=4, PE=1$	$N=50, d=20$
Row_32	$H=2, R=3, C=3$	$\gamma=20, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=4, PE=1$	$N=50, d=10$
SB_4	$H=1, R=5, C=3$	$\gamma=5, thr=0.3$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=1, PE=1$	$N=50, d=20$
SB_8	$H=2, R=3, C=5$	$\gamma=20, thr=0.1$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.2$	$K=1, PE=1$	$N=100, d=10$
SB_16	$H=2, R=3, C=3$	$\gamma=20, thr=0.1$	$N=50$	$C=100$, $K=$ 'RBFK', $E=0.5$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=20$
SB_32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	N/A	N/A	$K=4, PE=1$	$N=50, d=20$
SBN_4	$H=1, R=5, C=3$	$\gamma=2.5, thr=0.1$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=7, \lambda=0.1$	$K=2, PE=1$	$N=50, d=30$
SBN_8	$H=1, R=3, C=3$	$\gamma=5, thr=0.5$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=7, \lambda=0.1$	$K=2, PE=1$	$N=50, d=20$
SBN_16	$H=2, R=5, C=3$	$\gamma=20, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=20$
SBN_32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	N/A	N/A	$K=5, PE=1$	$N=50, d=20$
kmeansS._4	$H=1, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=7, \lambda=0.1$	$K=1, PE=1$	$N=100, d=20$
kmeansS._8	$H=2, R=3, C=5$	$\gamma=5, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=100, d=20$
kmeansS._16	$H=2, R=3, C=5$	$\gamma=20, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=7, \lambda=0.1$	$K=3, PE=1$	$N=50, d=30$
kmeansS._32	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=30$
Blob.	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.9$	$N=100$	$C=100$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.5$	$K=2, PE=1$	$N=50, d=20$
WavS.	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=100$	$C=100$, $K=$ 'PolyK', $E=1$	$\sigma^2=7, \lambda=0.1$	$K=2, PE=1$	$N=50, d=10$
J.-bag_2	$H=1, R=3, C=3$	$\gamma=1.25, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.2$	$K=3, PE=1$	$N=100, d=20$
J.-bag_6	$H=1, R=5, C=3$	$\gamma=1.25, thr=0.3$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.1$	$K=4, PE=1$	$N=50, d=30$
J.-bag_10	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.3$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.5$	$K=4, PE=1$	$N=50, d=20$
LBP	$H=1, R=5, C=3$	$\gamma=10, thr=0.9$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.3$	$K=3, PE=1$	$N=100, d=20$
SIFT	$H=2, R=3, C=5$	$\gamma=5, thr=0.9$	$N=50$	$C=100$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.6$	$K=2, PE=0.8$	$N=50, d=30$

"N/A" indicates the combination is very time-consuming. In miSVM, "PolyK" stands for the polynomial kernel, while "RBFK" is short for the RBF kernel

Table 20 The optimal parameters of the first splitting on the *MSRA* data collection

MSRA	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=3, PE=1$	$N=50, d=30$
Row_8	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=50, d=20$
Row_16	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=50, d=20$
Row_32	$H=2, R=5, C=3$	$\gamma=10, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=50, d=20$
SB_4	$H=2, R=5, C=3$	$\gamma=10, thr=0.5$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=3, PE=1$	$N=50, d=30$
SB_8	$H=2, R=5, C=3$	$\gamma=10, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=5, PE=1$	$N=50, d=20$
SB_16	$H=1, R=5, C=3$	$\gamma=10, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	N/A	$K=5, PE=0.9$	$N=50, d=30$
SB_32	$H=2, R=5, C=3$	$\gamma=10, thr=0.1$	$N=50$	N/A	N/A	$K=4, PE=0.9$	$N=50, d=20$
SBN_4	$H=2, R=5, C=5$	$\gamma=1.25, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=9, \lambda=0.1$	$K=1, PE=1$	$N=50, d=20$
SBN_8	$H=2, R=5, C=5$	$\gamma=1.25, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=9, \lambda=0.1$	$K=1, PE=1$	$N=50, d=20$
SBN_16	$H=2, R=5, C=3$	$\gamma=20, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	N/A	$K=4, PE=1$	$N=50, d=20$
SBN_32	N/A	N/A	$N=50$	N/A	N/A	$K=5, PE=1$	$N=50, d=20$
kmeansS._4	$H=1, R=5, C=5$	$\gamma=1.25, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=100, d=20$
kmeansS._8	$H=1, R=5, C=5$	$\gamma=1.25, thr=0.3$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=100, d=10$
kmeansS._16	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=20$
kmeansS._32	$H=2, R=3, C=5$	$\gamma=5, thr=0.9$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=30$
Blob.	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.4$	$K=1, PE=1$	$N=50, d=20$
WavS.	$H=1, R=5, C=3$	$\gamma=10, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=30$
J.-bag_2	$H=2, R=5, C=5$	$\gamma=1.25, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.5$	$K=1, PE=1$	$N=50, d=30$
J.-bag_6	$H=2, R=5, C=5$	$\gamma=1.25, thr=0.7$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=15, \lambda=0.3$	$K=4, PE=1$	$N=100, d=20$
J.-bag_10	$H=2, R=5, C=5$	$\gamma=1.25, thr=0.5$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.5$	$K=3, PE=1$	$N=50, d=20$
LBP	$H=2, R=5, C=3$	$\gamma=10, thr=0.3$	$N=50$	N/A	N/A	$K=3, PE=1$	$N=50, d=30$
SIFT	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.1$	$N=50$	N/A	N/A	$K=5, PE=0.9$	$N=50, d=20$

"N/A" indicates the combination is very time-consuming. In miSVM, "PolyK" stands for the polynomial kernel, while "RBFK" is short for the RBF kernel

Table 21 The optimal parameters of the second splitting on the *MSRA* data collection

MSRA	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.3$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=20$
Row_8	$H=1, R=5, C=3$	$\gamma=2.5, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=4, PE=1$	$N=50, d=20$
Row_16	$H=2, R=3, C=5$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.2$	$K=5, PE=1$	$N=50, d=10$
Row_32	$H=2, R=3, C=3$	$\gamma=1.25, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.2$	$K=5, PE=1$	$N=50, d=20$
SB_4	$H=2, R=5, C=3$	$\gamma=10, thr=0.1$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.2$	$K=3, PE=1$	$N=100, d=10$
SB_8	$H=2, R=5, C=3$	$\gamma=10, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.4$	$K=4, PE=1$	$N=50, d=30$
SB_16	$H=2, R=5, C=3$	$\gamma=10, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	N/A	$K=5, PE=1$	$N=50, d=20$
SB_32	$H=2, R=3, C=3$	$\gamma=5, thr=0.1$	$N=50$	N/A	N/A	$K=5, PE=1$	$N=50, d=30$
SBN_4	$H=1, R=5, C=3$	$\gamma=5, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=9, \lambda=0.3$	$K=2, PE=1$	$N=50, d=20$
SBN_8	$H=2, R=5, C=3$	$\gamma=5, thr=0.9$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=9, \lambda=0.3$	$K=1, PE=1$	$N=50, d=30$
SBN_16	$H=2, R=5, C=3$	$\gamma=5, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	N/A	$K=4, PE=1$	$N=50, d=20$
SBN_32	N/A	N/A	$N=50$	N/A	N/A	$K=5, PE=1$	$N=50, d=20$
kmeansS._4	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.1$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.3$	$K=4, PE=1$	$N=50, d=20$
kmeansS._8	$H=1, R=5, C=5$	$\gamma=1.25, thr=0.3$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=15, \lambda=0.5$	$K=3, PE=1$	$N=50, d=20$
kmeansS._16	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.3$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=50, d=30$
kmeansS._32	$H=2, R=3, C=3$	$\gamma=5, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.3$	$K=4, PE=1$	$N=50, d=20$
Blob.	$H=1, R=5, C=3$	$\gamma=1.25, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=15, \lambda=0.3$	$K=2, PE=1$	$N=100, d=10$
WavS.	$H=2, R=5, C=3$	$\gamma=10, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.1$	$K=3, PE=1$	$N=50, d=20$
J.-bag_2	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.3$	$K=2, PE=1$	$N=50, d=20$
J.-bag_6	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.5$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.3$	$K=3, PE=1$	$N=50, d=20$
J.-bag_10	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.5$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.5$	$K=4, PE=1$	$N=50, d=30$
LBP	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.3$	$N=100$	N/A	N/A	$K=4, PE=1$	$N=50, d=20$
SIFT	$H=2, R=5, C=5$	$\gamma=1.25, thr=0.9$	$N=50$	N/A	N/A	$K=4, PE=0.9$	$N=50, d=20$

"N/A" indicates the combination is very time-consuming. In miSVM, "PolyK" stands for the polynomial kernel, while "RBFK" is short for the RBF kernel

Table 22 The optimal parameters of the third splitting on the *MSRA* data collection

MSRA	Citation-kNN	miGraph	MIBoosting	miSVM	MILES	miFV	MIForests
Row_4	$H=1, R=3, C=3$	$\gamma=2.5, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=2, PE=1$	$N=50, d=30$
Row_8	$H=2, R=3, C=3$	$\gamma=5, thr=0.5$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.2$	$K=5, PE=1$	$N=50, d=20$
Row_16	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.2$	$K=4, PE=1$	$N=50, d=20$
Row_32	$H=2, R=5, C=3$	$\gamma=10, thr=0.9$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=4, PE=1$	$N=50, d=30$
SB_4	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.5$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.2$	$K=4, PE=1$	$N=100, d=10$
SB_8	$H=1, R=5, C=3$	$\gamma=1.25, thr=0.1$	$N=100$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=7, \lambda=0.1$	$K=3, PE=1$	$N=50, d=30$
SB_16	$H=2, R=5, C=3$	$\gamma=1.25, thr=0.5$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	N/A	$K=5, PE=0.1$	$N=50, d=20$
SB_32	$H=2, R=3, C=3$	$\gamma=1.25, thr=0.5$	$N=100$	N/A	N/A	$K=5, PE=0.9$	$N=50, d=20$
SBN_4	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.1$	$N=50$	$C=50$, $K=$ 'PolyK', $E=2$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=20$
SBN_8	$H=2, R=5, C=5$	$\gamma=2.5, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=2, PE=1$	$N=50, d=30$
SBN_16	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	N/A	$K=3, PE=1$	$N=50, d=20$
SBN_32	N/A	N/A	$N=100$	N/A	N/A	$K=5, PE=1$	$N=50, d=30$
kmeansS._4	$H=2, R=5, C=3$	$\gamma=10, thr=0.1$	$N=50$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=15, \lambda=0.5$	$K=4, PE=1$	$N=50, d=20$
kmeansS._8	$H=2, R=5, C=5$	$\gamma=10, thr=0.1$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.2$	$K=5, PE=1$	$N=50, d=30$
kmeansS._16	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.3$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.1$	$K=5, PE=1$	$N=50, d=20$
kmeansS._32	$H=1, R=3, C=5$	$\gamma=10, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.2$	$K=5, PE=1$	$N=100, d=20$
Blob.	$H=2, R=3, C=3$	$\gamma=2.5, thr=0.1$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.1$	$K=2, PE=1$	$N=50, d=20$
WavS.	$H=1, R=3, C=3$	$\gamma=2.5, thr=0.9$	$N=50$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=15, \lambda=0.4$	$K=3, PE=1$	$N=50, d=20$
J.-bag_2	$H=2, R=5, C=3$	$\gamma=2.5, thr=0.5$	$N=100$	$C=50$, $K=$ 'RBFK', $G=0.5$	$\sigma^2=5, \lambda=0.5$	$K=2, PE=1$	$N=50, d=30$
J.-bag_6	$H=2, R=3, C=5$	$\gamma=10, thr=0.1$	$N=100$	$C=50$, $K=$ 'PolyK', $E=1$	$\sigma^2=5, \lambda=0.3$	$K=3, PE=1$	$N=50, d=20$
J.-bag_10	$H=1, R=5, C=5$	$\gamma=1.25, thr=0.5$	$N=50$	$C=50$, $K=$ 'RBFK', $G=1$	$\sigma^2=5, \lambda=0.5$	$K=4, PE=1$	$N=50, d=20$
LBP	$H=1, R=5, C=3$	$\gamma=1.25, thr=0.1$	$N=50$	N/A	N/A	$K=4, PE=1$	$N=50, d=20$
SIFT	$H=2, R=3, C=5$	$\gamma=2.5, thr=0.3$	$N=100$	N/A	N/A	$K=4, PE=0.9$	$N=50, d=30$

"N/A" indicates the combination is very time-consuming. In miSVM, "PolyK" stands for the polynomial kernel, while "RBFK" is short for the RBF kernel

References

Amores, J. (2013). Multiple instance classification: Review, taxonomy and comparative study. *Artificial Intelligence*, *201*, 81–105.

Andrews, S., Tsochantaridis, I., & Hofmann, T. (2003). Support vector machines for multiple-instance learning. In *Advances in neural information processing systems* (Vol. 15, pp. 561–568). Cambridge, MA: MIT Press.

Carson, C., Belongie, S., Greenspan, H., & Malik, J. (2002). Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(8), 1026–1038.

Chatfield, K., Lempitsky, V., Vedaldi, A., & Zisserman, A. (2011). The devil is in the details: An evaluation of recent feature encoding methods. In *Proceedings of 22nd British machine vision conference*. Dundee, Scotland, pp. 1–12.

Chen, S., Sista, S., Shyu, M., & Kashyap, R. L. (2000). An indexing and searching structure for multimedia database systems. In *Proceedings of IS&T/SPIE conference storage and retrieval for media databases*. San Jose, CA, pp. 262–270.

Chen, Y., Bi, J., & Wang, J. Z. (2006). MILES: Multiple-instance learning via embedded instance selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(12), 1931–1947.

Chen, Y. X., & Wang, J. Z. (2004). Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research*, *5*, 913–939.

Chevaleyre, Y., & Zucker, J. D. (2001). A framework for learning rules from multiple instance data. In *Proceedings of 12th European conference on machine learning*. Freiburg, Germany, pp. 49–60.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, *7*, 1–30.

Deng, Y. N., & Manjunath, B. S. (2001). Unsupervised segmentation of color-texture regions in images and video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(8), 800–810.

Dietterich, T. G., Lathrop, R. H., & Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, *89*(1–2), 31–71.

Felzenszwalb, P., Girshick, R., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(9), 1627–1645.

Fisher, R. (1959). *Statistical methods and scientific inference* (2nd ed.). New York: Hafner Publishing Company.

Fung, G., Dundar, M., Krishnapuram, B., & Rao, R. B. (2007). Multiple instance learning for computer aided diagnosis. In *Advances in neural information processing systems* (Vol. 19, pp. 425–432). Cambridge, MA: MIT Press.

Heikkilä, M., & Pietikäinen, M. (2006). A texture-based method for modeling the background and detecting moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(4), 657–662.

Leistner, C., Saffari, A., & Bischof, H. (2010). MIForests: Multiple-instance learning with randomized trees. In *Proceedings of 11th European onference on computer vision*. Crete, Greece, pp. 29–42.

Li, F. F., & Perona, P. (2005). A Bayesian hierarchical model for learning natural scene categories. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*. San Diego, CA, pp. 524–531.

Li, H., Wang, M., & Hua, X. S. (2009). MSRA-MM 2.0: A large-scale web multimedia dataset. In *Proceedings of 9th international conference on data mining workshops*. Miami, Florida, pp. 164–169.

Liu, W., Xu, W. D., Li, L. H., & Li, G. L. (2008). Two new bag generators with multi-instance learning for image retrieval. In *Proceedings of 3rd IEEE conference on industrial electronics and applications*. Singapore, pp. 255 – 259.

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

Maron, O., & Lozano-Pérez, T. (1998). A framework for multiple-instance learning. In *Advances in neural information processing systems* (Vol. 10, pp. 570–576). Cambridge, MA: MIT Press.

Maron, O., & Ratan, A. L. (2001). Multiple-instance learning for natural scene classification. In *Proceedings of 18th international conference on machine learning*. Williamstown, MA, pp. 425–432.

Nemenyi, P. B. (1963) *Distribution-free multiple comparisons*. PhD thesis.

Nguyen, M. H., Torresani, L., Torre, F., & Rother, C. (2009). Weakly supervised discriminative localization and classification: A joint learning process. In *Proceedings of 12th international conference on computer vision*. Kyoto, Japan, pp. 1925–1932.

Nowak, E., Jurie, F., & Triggs, B. (2006). Sampling strategies for bag-of-features image classification. In *Proceedings of 9th European conference on computer vision*. Graz, Austria, pp. 490–503.

Ojala, T., Pietikäinen, M., & Mäenpää, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(7), 971–987.

Sánchez, J., Perronnin, F., Mensink, T., & Verbeek, J. (2013). Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision*, *105*(3), 222–245.

Settles, B., Craven, M., & Ray, S. (2008). Multiple instance active learning. In *Advances in neural information processing systems* (Vol. 20, pp. 1289–1296). Cambridge, MA: MIT Press.

Song, X. F., Jiao, L. C., Yang, S. Y., Zhang, X. R., & Shang, F. H. (2013). Sparse coding and classifier ensemble based multi-instance learning for image categorization. *Signal Processing*, *93*, 1–11.

Tang, J. H., Li, H. J., Qi, G. J., & Chua, T. S. (2010). Image annotation by graph-based inference with integrated multiple/single instance representations. *IEEE Transactions on Multimedia*, *12*, 131–141.

Vijayanarasimhan, S., & Grauman, K. (2008). Keywords to visual categories: Multiple-instance learning for weakly supervised object categorization. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*. Anchorage, Alaska, pp. 1–8.

Viola, P., Platt, J., & Zhang, C. (2006). Multiple instance boosting for object detection. In *Advances in neural information processing systems* (Vol. 18, pp. 1419–1426). Cambridge, MA: MIT Press.

Wang, J., & Zucker, J. D. (2000). Solving multiple-instance problem: A lazy learning approach. In *Proceedings 17th international conference on machine learning*. San Francisco, CA, pp. 1119–1125.

Wei, X. S., Wu, J., & Zhou, Z. H. (2014). Scalable multi-instance learning. In *Proceedings of 14th international conference on data mining*. Shenzhen, China, pp. 1037–1042.

Xu, X., & Frank, E. (2004). Logistic regression and boosting for labeled bags of instances. In *Proceedings of 8th Pacific-Asia conference on knowledge discovery and data mining*. Sydney, Australia, pp. 272–281.

Xu, Y. Y., & Shih, C. H. (2012). Content based Image retrieval using multiple instance decision based neural networks. In *Proceedings of IEEE international conference on computational intelligence and cybernetics*. Bali, Indonesia, pp. 175–179.

Yang, C., & Lozano-Pérez, T. (2000). Image database retrieval with multiple-instance learning techniques. In *Proceedings of 16th international conference on data engineering*. San Diego, CA, pp. 233–243.

Yang, C. B., Dong, M., & Hua, J. (2006). Region-based image annotation using asymmetrical support vector machine-based multiple-instance learning. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*. New York, NY, pp. 2057–2063.

Zhang, C., & Viola, P. (2008). Multi-instance learning pruning for learning efficient cascade detectors. In *Advances in neural information processing systems* (Vol. 20, pp. 1681–1688). Cambridge, MA: MIT Press.

Zhang, C. C., Chen, S., & Shyu, M. (2004). Multiple object retrieval for image databases using multiple instance learning and relevance feedback. In *Proceedings of IEEE international conference on multimedia and expo*. Sydney, Australia, pp. 775–778.

Zhang, Q., & Goldman, S. A. (2000). EM-DD: An improved multiple-instance learning technique. In *Proceedings of 16th IEEE international conference on data engineering*. San Diego, California, USA, pp. 233–243.

Zhang, Q., Goldman, S. A., Yu, W., & Fritts, J. E. (2002). Content-based image retrieval using multiple-instance learning. In *Proceedings of 19th international conference on machine learning*. Sydney, Australia, pp. 682–689.

Zhou, Z. H., Zhang, M. L., & Chen, K. J. (2003). A novel bag generator for image database retrieval with multi-instance learning. In *Proceedings of 15th IEEE international conference on tools with artificial intelligence*. Sacramento, CA, pp. 565–569.

Zhou, Z. H., Jiang, K., & Li, M. (2005). Multi-instance learning based web mining. *Applied Intelligence*, *22*(2), 135–147.

Zhou, Z. H., Sun, Y. Y., & Li, Y. F. (2009). Multi-instance learning by treating instances as non-I.I.D. samples. In *Proceedings of 26th international conference on machine learning*. Montreal, Canada, pp. 1249–1256.

Zhu, S. L., & Tan, X. Q. (2011). A novel automatic image annotation method based on multi-instance learning. *Procedia Engineering*, *15*, 3439–3444.