Bag of Features Model Using the New Approaches: A Comprehensive Study

CHOUGRAD Hiba, ZOUAKI Hamid
LIMA, Laboratoire d'Informatique et Mathématique et leurs Applications, CHOUAIB DOUKKALI University
El Jadida, Morocco

ALHEYANE Omar
LMF, Laboratoire de Mathématiques Fondamentales, CHOUAIB DOUKKALI University
El Jadida, Morocco

***Abstract*—The major challenge in content based image retrieval is the semantic gap. Images are described mainly on the basis of their numerical information, while users are more interested in their semantic content and it is really difficult to find a correspondence between these two sides. The bag of features (BoF) model is an efficient image representation technique for image classification. However, it has some limitations for instance the information loss during the encoding process, an important step of BoF. This is because the encoding is usually done by hard assignment i.e. in vector quantization each feature is encoded by being assigned to a single visual word. Another notorious disadvantage of BoF is that it ignores the spatial relationships among the patches, which are very important in image representation. To address those limitations and enhance the results, novel approaches were proposed at each level of the BoF pipeline. In instance the combination of local and global descriptors for a better description, a soft-assignment encoding manner with a spatial pyramid partitioning for a more informative image representation and a maximum pooling to get the final descriptors. Our work aims to give a detailed version of the BoF, including all the levels of the pipeline, as a support leading to a better comprehension of the approach. We also compare and evaluate the state-of-the-art approaches and find out how these changes at each level of the pipeline could affect the performance and the overall classification results.**

Keywords*—Bag of features; Image classification; Local and global descriptors; Locality-constrained Linear Coding; Spatial pyramid; Pooling

I. INTRODUCTION

The Content-Based Image Retrieval (CBIR) systems are supposed to provide us with a simple accessible way for researching and retrieving images from a large scale image collection based on semantic and visual content of the images. The main problem in content-based image retrieval (CBIR) and visual recognition arises from the "semantic gap" which is the difference between the information that one can extract from the visual data and the interpretation that this same data has for a user in a given context. In order to fill this gap many challenges need to be overcome, for instance, there is the large scale nature of the problem where the datasets are large, the image descriptor dimension is large and the annotated training data is restricted. [1, 2, 3, 4]. Also, we need to deal with other factors, such as the viewpoint, scale, rotation and illumination changes [5, 6].

In addition to this, occlusion [7] and clutter [8] make the visual recognition task even harder since an object can be occluded and only part of it is visible. Finally, there is the elusive notion of similarity which includes: the intra-class variation, which is the diversity that exists between instances of the same object and the inter-class similarity, or the similitude between instances of different object categories.

Our work is based on the bag of features approach. The main idea is that a set of local image patches is sampled using some method (e.g. keypoints detector) then the local descriptors are extracted from the patches using (e.g. SIFT descriptor [5]). The resulting descriptors are then quantified to form a codebook (e.g. using K-means) then the image descriptors are encoded by being projected to the linear subspace of the closest visual words (e.g. using LLC). Now, we need to go from these obtained code vectors to a visual representation, for this we can use the spatial pyramid scheme in order to capture the spatial information of these descriptors. The local code vectors are aggregated using a pooling method to obtain the final image descriptor. The resulting global descriptor vector is used as a representation of the image for learning and classification purposes (e.g. it is used to learn an image classification rule based on an SVM classifier).

The main goal of this paper is to first give a helpful basis for the new researchers interested in image classification by providing them with a detailed version of the BoF method that includes all the levels of the pipeline. Normally, researchers focus on a specific level of it considering the other levels as known and understood. Then, at each level of the pipeline, we compare the standard methods used in bag of features versus the state-of-the-art methods to see when and how we can combine them to get the best performance accuracy. Finally, we'll clarify the added value of each one of these novel approaches, if it is for capturing more information or for reducing the computational cost.

In section 2, we briefly review the related work. We present the bag of feature pipeline in section 3 with all its levels. We define the purpose of each level and give the most relevant used approaches to achieve it. In section 4 the comparative system is described along with the experiments and results. Section 5 concludes the paper.

II. RELATED WORK

The bag of features (BoF) [9, 1, 10] approach has established itself as the state-of-the-art for image classification. It consists of five steps, the feature extraction phase, the codebook creation and encoding of the features phase, the

feature pooling and finally the learning and classification phase. Researchers tried to improve the standard bag of features [1] by optimizing each and all the steps in the pipeline.

The feature extraction phase is done in two steps, first we sample the patches using either interest points [1, 5, 9, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20] or densely with a regular grid [21, 22, 23, 24]. Until recently in [25], when they combined the two ways and suggested to find interest points within a dense grid. The second step would be to extract the features and get the descriptors, here again we can describe the patches using a global spatial layout like GIST [26], a local descriptor like SIFT [1, 16, 5, 9], a filter based [21, 22] or a raw patch based [14, 20, 15, 17, 19] representations.

To quantize local descriptors into visual words, we must first generate the visual vocabulary. For this purpose, algorithms like mixture of Gaussian [27], mean-shift [28] agglomerative clustering [14, 17] and the most popular of all K-means [1, 20, 19, 22] are used. For the encoding part the simplest way assigns a local descriptor to the closest visual word, giving one and only one nonzero coefficient. Also known as "hard"-assignment, this does not consider codeword ambiguity which causes a significant information loss [29, 30, 31]. In [32, 31], a "soft"-assignment coding is proposed to deal with these problems by assigning a local descriptor to all visual words in a weighted manner. Recently, sparse [33, 34] and local [34, 35, 36, 37] coding proved to be very efficient. They optimize a linear combination of few visual words to approximate a local descriptor and code it with the optimized coefficients.

Once we get our code vectors we need to aggregate them in order to get the final descriptor, sum-pooling (average-pooling) which simply sums the coefficients, it has been commonly used to obtain the image-level. Recent work indicates that max-pooling which chooses the largest coefficient for a visual word can lead to a better classification performance [35, 38, 34, 39]. In order to compare and quantify the influence of some of these methods; in the different levels of the pipeline; and to see how these changes affect the classification performance, we need an image classifier that could predict image labels, once the image descriptors are extracted. For the BoF classification, usually the discriminative methods result in a higher classification accuracy, this is why for this paper we chose to use an SVM classifier [40] which has been the most popular classifier in the past decade.

III. Bag of Features

Inspired from the original text representation model, bag of words [41] used for document classification, BoF was introduced first by [9] in video retrieval then with [1] for image categorization. An image is represented as an unordered collection of visual words. BoF gives an extremely compact description of images as they are represented as histograms of local descriptors. The main idea is to obtain visual words (features) by quantizing the local descriptors of images in the dataset based on a visual vocabulary. The vocabulary is constructed by clustering a large set of local descriptors using algorithms like K-means [42, 43]. The algorithm takes as an input the training data description and gives as an output a set of clusters. Each cluster would be represented by one visual word. The image is now represented as a bag of visual words and a histogram can be built with a dimension equal to the visual vocabulary size, each bin will contain the visual word's frequency with respect to the image.

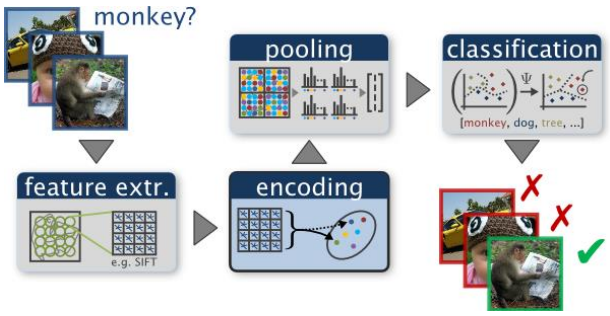

Fig. 1. The BoF pipeline by [4]

A. Feature sampling and extraction

The goal is to obtain a representative set of image patches covering the most relevant information in a given image. After detecting the most interesting regions in each image, the feature extraction pipeline starts in order to compute the vectors that will describe these regions. A local descriptor is used in image categorization and object recognition tasks and also to match similar object instances. Many methods for feature description can be employed; in our work we adopted the commonly most used feature descriptors that achieved state-of-the-art results on several benchmarks over the years.

SIFT descriptor [5] is a sparse feature representation that consists of both feature extraction and detection. To detect scale-invariant characteristic points, the SIFT approach uses cascaded filters, where the difference of Gaussians (DoG), is calculated on rescaled images progressively. Then interest points are described by gradient orientation histograms to get a128-dimensional vector as the SIFT representation for a pixel. Given an image, SIFT finds all the keypoints in the image with respect to the gradient feature of each pixel. Every keypoint contains the information of its location, local scale and orientation. Then, based on each keypoint, SIFT computes a local image descriptor. Combining all the local descriptors, we get the complete features from the image.

A fast alternative is **dense SIFT descriptor** (DSIFT) which is provided by the VLFeat [44] open source library. DSIFT is an extension of the SIFT algorithm. It makes some new assumptions: (a) the location of each keypoint is not from the gradient feature of the pixel, but from a predesigned location; (b) the scale of each keypoint is all the same which is also predesigned; (c) the orientation of each keypoint is always zero. With this assumptions, DSIFT can acquire more features in less time than SIFT does. Figure 2 below shows the features extracted by both SIFT and DSIFT descriptors from the same images. **SURF** [6] based on the same principles and steps of SIFT. It has a Hessian-based detector and a distribution-based descriptor generator. It uses a blob detector based on the Hessian to find points of interest. In contrast to the Hessian-Laplacian detector by [18], SURF also uses the determinant of the Hessian for selecting the scale, as it is done by [45].

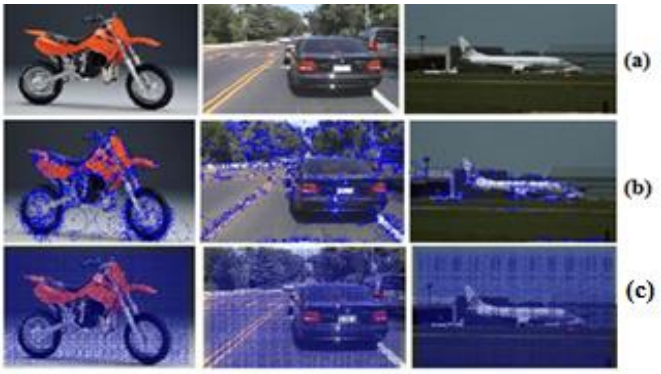

Fig. 2. (a) original images (b) Visualization of SIFT features (c) Visualization of dense SIFT features

An orientation is first assigned to the keypoint. Then a square region is constructed around the keypoint and rotated according to the orientation. The region is split up regularly into smaller 4x4 square sub-regions. For each sub-region, we compute a few simple features at 5x5 regularly spaced sample points. The horizontal and vertical Haar-wavelet responses *dx* and *dy* are calculated and summed up over each sub-region and form a first set of entries to the feature vector. The absolute values of the responses |*dx*| and |*dy*| are also calculated, together with the sum of vector to form a four-dimensional descriptor. And for all 4x4 sub-regions, it results in a vector of length 64.

GIST [26] captures the spatial characteristics of the scene categories. The main idea is to split an image using a regular grid and compute average response magnitudes of a number of Gabor filters in each spatial cell. It uses Gabor filter to extract a holistic description of a lot properties in the image. These properties are highly related to the underlying scene where the image was taken. It will help determine what type of object the image is showing. The resulting descriptor encodes the existence of edge-like local structures at various orientations and scales.

B. Creation of the codebook and feature quantization

The encoding phase transforms the local descriptor obtained to a new form, using the visual vocabulary (codebook or dictionary).

1) Creating the codebook:

K-means clustering k-means clustering is a vector quantization method and probably the most common way of constructing the visual vocabulary. It aims to partition N descriptors into k clusters in which each descriptor belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells [46]. To get an example of a visual word from the pipeline we save the paths to images for which we used the descriptors to create the visual words, so that we can visualize our visual vocabulary like shown in figure 3.

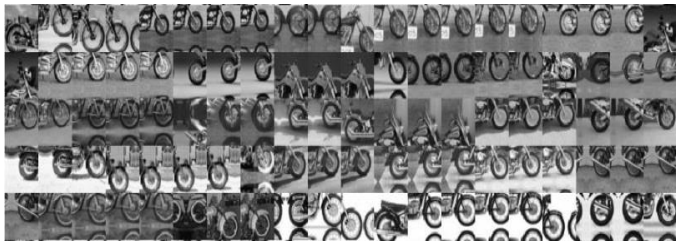

Fig. 3. The image patches corresponding to a resulting visual word from our pipeline, after the codebook creation

2) Encoding: The standard BoF method for encoding transforms the local descriptor into a more adapted form using the codebook [1, 42, 9], but it has some shortcomings for instance the information loss caused by vector quantization where a descriptor can be similar to many visual words or be different from all of them. Recently, the hard quantization of features was replaced by alternative methods that would keep more information about the original image features (soft-assignment). A patch is assigned to multiple visual words in a weighted manner according to its proximity to vocabulary centers in the local descriptor space [47, 30, 31].This has been done in two ways: (1) by expressing features as combinations of visual words (e.g., soft quantization [32], local linear encoding [35]). (2) by recording the difference between the features and the visual words (e.g., Fisher encoding [48], super-vector encoding [49]).

In the encoding step we chose to use three different approaches, first a hard assignment encoding as the standard BoF vector quantization method. Then, in order to better apprehend the new soft-assignment technique, we used a distance-based soft quantization method and a reconstruction-based soft assignment method renowned for its good performance.

Nearest Neighbors requires two parameters the number of neighbors (k) and the distance measure (e.g. Euclidean). The algorithm repeats iteratively the calculation for each descriptor. The code vector (representation) created will be of the size of the dictionary and will contain the frequency count of the assignment of each descriptor to a visual word. To do the assignment we use an L2 distance and pick the nearest neighbor visual word to each descriptor.

The 1-NN algorithm is as follow: Given the data $D = \{x_i, y_i\}$, a distance function d and input x we find: $j = \arg\min_i d(x, x_i)$ and return y_i. The problem is that the strict memorization aspect of 1-NN leads to information loss. One way to deal with this is local averaging; instead of just one neighbor, we find K nearest neighbors and use them in a weighted manner. All neighbors are used, but with different weights. Closer neighbors receive higher weights. The weighting function (kernel function) is the Gaussian: $K(x, x_i) = \exp\left\{\frac{-d^2(x,x_i)}{\sigma^2}\right\}$

The K-Nearest Neighbors with Gaussian kernel similarity metric is as follow: Given the data $D = \{x_i, y_i\}$, the Kernel function K and input x. If $y \in \pm 1$ the return weighted majority is $\text{sign}(\sum_{i=1}^{n} K(x, x_i) y_i)$

Locality-constrained Linear (LLC) Coding hard-assignment selects the first closest visual word, it is fast but it gives a large quantization error. Recently [35] proposed a reconstruction based approach where features are "reconstructed" by a linear combination of visual words.

Let x be a set of D-dimensional local descriptors extracted from an image, i.e. $x = [x_1, x_2, \dots, x_N] \in R^{DxN}$ where D is the descriptor's dimension (e.g. SIFT=128) and N is the number of descriptors (DxN matrix).

Given a codebook with M entries, $B = [b_1, b_2, \dots, b_M] \in R^{DxM}$ .The coding scheme converts each descriptor into an M-dimensional code to generate the final image representation. For a descriptor x and a codebook B, coefficients are chosen to solve:

$$\arg\min_{\gamma} \sum_{i=1}^{N} \|x_i - B\gamma_i\|^2 + \lambda \left\| d_{i \odot} \gamma_i \right\|^2 \text{ s.t. } 1^T \gamma_i = 1, \forall i$$

Which means x_i could approximately be reconstructed by $B\gamma_i$ .With $d_i = \exp\left(\frac{\text{dist}(x_i, B)}{\sigma}\right)$ is the locality adaptor which gives lower weights to the basis vectors that are different from the input descriptor x_i and vice versa, this leads to the locality constraint.

$\text{dist}(x_i, B) = [\text{dist}(x_i, b_1), \dots, \text{dist}(x_i, b_M)]^T$, where $\text{dist}(x_i, b_j)$ is the Euclidian distance between x_i and b_j. σ is a parameter controlling the weight decay speed for the locality adaptor. $\odot$ denotes the element-wise multiplication and $1^T \gamma_i = 1$ insures sparseness.

The distance regularization of LLC effectively performs feature selection: it selects local bases for each descriptor and form a local coordinate system, and in practice only those bases close to x_i in feature space have non-zero coefficients.

This suggests, a fast approximation of LLC can be developed by removing the regularization completely and instead using the K (K < D <M) nearest neighbors of x_i as a set of local bases B_i and solve a smaller linear system to get the code vectors: $\arg\min_{\tilde{\gamma}} \sum_{i=1}^{N} \|x_i - B_i \tilde{\gamma}_i\|^2$ s.t $1^T \tilde{\gamma}_i = 1, \forall i$ which reduces the computation complexity from $O(M^2)$ to $O(M + K^2)$ where $K \ll M$.

C. Aggregation of the encoded local descriptors to obtain the final image descriptor

Now that we have our encoded descriptors, an image is still represented by too many code vectors (encoded descriptors), so we'll need to pool these vectors in order to describe the image with one final descriptor. In the standard BoF pipeline we perform hard assignment for coding and then directly start aggregating the encoded local descriptors with sum pooling. In our work, we extend the old version of BoF by first adding the spatial pyramid scheme to capture the spatial information, we then proceed with the aggregation using either sum pooling or max pooling; the latter is a newly used approach that appears to be giving promising results.

1) The spatial pyramid matching (SPM) [50]: aims to incorporate the global spatial layout into the image representation. Inspired from the original work of [16], the spatial pyramid creates a pyramid of regular grids with increasingly finer cells. Each spatial cell will give us a histogram of visual words and the concatenation of all the histograms in a weighted manner gives us the final descriptor.

In the original work they construct a three-level pyramid. In a level $i \in \{0,\dots,n\}$ of the pyramid, the image is divided into 2^i spatial cells. Each cell has a histogram of visual words, and the image is represented with $\sum_{i=0}^{n} 2^i \times M$ vector, where M is the length of the codebook. This is a form of spatial pooling when we add the spatial position by assigning the histogram of each level with a weight, then we aggregate all the weighted histograms to get the final descriptor.

2) Pooling: Given the coding coefficient γ of each local descriptor in an image, a pooling operation is often used to obtain an image level representation p where $p \in R^M$ with M the total number of visual words.

Sum or average pooling [50] With sum-pooling, the i^{th} component of p is $p_j = \sum_{i=1}^{l} \gamma_{ij}$, where l is the total number of local features in an image. Dividing p_j by l gives us sum or average pooling. The histogram of number of occurrences of visual words in an image is essentially obtained by applying sum pooling to hard-assignment coding results.

Max-pooling [34] The i^{th} component of p is defined as $p_j = \max_i \gamma_{ij}$, where i=l,...,l . For each codeword we select the feature with the maximum coefficient γ. Max-pooling often gives better classification than average pooling. When used with hard-assignment coding scheme, max-pooling gives a binary histogram, indicating the presence or absence of each visual word in an image.

D. The classification phase

In image categorization, the goal is to automatically annotate images with predefined categories. Once the image descriptors are extracted, image labels are predicted using a set of classifiers.

SVM classification: In our work we rely on Support Vector Machine (SVM) classifiers to carry out the classification task. SVM is a discriminative classifier formally defined by a separating hyper plane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyper plane which categorizes new examples. Given a labeled set $L = \{x_i, y_i\}_{i=1}^{l}$, where $x \in R^d$ and $y \in \{-1, +1\}$ .A linear maximal margin classifier $f(x) = w^T x + b$ can be found by solving:

$\min_{w,b,\xi} \sum_{i=1}^{l} \xi_i + \lambda \|w\|^2$ s.t. $y_i(w^T x + b) \geq 1, \ \xi_i \geq 0, \forall i$, $i = 1, \dots, l$. For a hyper plane $w \in R^d$, and its offset $b \in R$.

The extent of the SVM classifier lies in its easy extension to the nonlinear case [51]. Highly nonlinear nature of data can be taken into account by using the kernel trick such that the hyper plane is found in a feature space induced by an adapted kernel function $K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$. SVM handles nonlinearly separable problems using kernel functions by

projecting the data points into a higher-dimensional feature space $x \to \Phi(x)$. The Multi-class problem can be discriminated by performing, several "one-versus-all" classifications.

It was empirically found that, to achieve good performance, traditional SPM has to use classifiers with nonlinear Mercer kernels, e.g., Chi-square kernel. However, the nonlinear classifier has to afford additional computational complexity, bearing $O(n^3)$ in training and $O(n)$ for testing in SVM, where n is the number of support vectors. This implies a poor scalability of the SPM approach for real applications [34, 35].

To improve the scalability, researchers opted for a nonlinear feature representations that work better with linear classifiers, e.g. the works of [52, 53, 34, 35] where the final representation generated by using these encodings, achieved state-of-the-art performances on several benchmarks with a linear SVM classifier. In our experiments, we use a one-versus-all linear SVM classifier, implemented by LibSVM toolbox [55] and the optimized values of the parameters of SVM models are given by cross validation.

IV. Experiments and Results

In this section, we first give the detailed implementation of our experiment then we evaluate the different methods introduced in Section 3 in each level of the BoF pipeline.

A. Implementation

1) Fetching the data: We follow the experimental setup of [16, 34], where the training set contains 30 images and the testing set contains 50 images per class. We create our ground truth matrix that will contain the labels of the testing images.

2) Training phase:

a) Detection and extraction of features: the local descriptors used are DSIFT or SURF that will be combined sometimes with the global descriptor GIST for experimental purposes.

- **DSIFT:** For the patch sampling process we set the spacing of the dense grid to 6 pixels, and the sampling window is set to 16*16 pixels.
- **SURF:** We detect SURF keypoints using a threshold set to 55 as it gives the best results for our datasets, and we select the strongest features. We then extract the SURF descriptors.
- **GIST:** We convolve the image with 32 Gabor filters at 4 scales, 8 orientations, producing 32 feature maps of the same size of the input image. Then, we divide each feature map into 16 regions (by a 4x4 grid), and average the feature values within each region. Finally, we concatenate the 16 averaged values of all 32 feature maps, resulting in a 512-dimension GIST descriptor.

b) Codebook computation: we use K-means algorithm on the descriptors extracted in the previous step in order to create our vocabulary; we chose a maximum number of iterations for K-means equal to 50 to generate 1024 visual words as suggested in the work of [55].

c) Coding: In this phase the descriptors are represented by a code vector using the codebook generated in the previous step. Here, we used 3 encoding algorithms, the standard hard assignment NN, KNN with a Gaussian kernel measure and LLC known for its good performance.

- **NN:** To represent our training and testing images as a histogram of visual words, for each image we will densely sample the descriptors and simply count how many fall into each cluster according to our visual word vocabulary. This is done by finding the nearest neighbor k-means centroid for every feature.
- **KNN:** K=5 Neighbors are used, the kernel Gaussian function $K(x, x_i) = \exp\{-d^2(x, x_i)/\sigma^2\}$ is then calculated for each visual word using the Euclidian distance d .We used a Gaussian kernel and chose σ=500 since it gave the best results according to our experiments.
- **LLC:** Using the Approximated Locality-constraint Linear Coding requires setting the number of neighbor visual words M considered for each encoded descriptor. This is set to K=5 as suggested in the original work [35]. As for The regularization constant λ in the computation of the projections is set to 10^{-4}.

d) Aggregation to get the final descriptor: Once we get all our code vectors, we need to aggregate them in order to get the final image descriptor.

- **Spatial pyramid:** The spatial pyramid is set to 3 levels so the image is divided increasingly into 2^i spatial cells with i= {0,1,2}.
- **Pooling:** we compared the two known methods average and max pooling. For sum-pooling, we'll average all the code vectors to create the final descriptor and for max-pooling we select the feature with the maximum coefficient.

e) Learning: once the code vectors are ready, we get our training matrix with the training labels that will be used by the SVM classifier in order to create the SVM model and generate the primal variable w of linear SVM.

3) Testing phase: in this phase we follow the same steps (from a to d) explained earlier in the training phase but using the test images instead of the training images.

- **Classification:** we classify the resulting code vectors using SVM, which contains the code vectors of the training set, their labels, the SVM model and the code vectors of the testing set. The classifier will predict the label of each image in the testing set and put it in a matrix.

4) Performance and accuracy: Multi-class classification is done with the trained SVM that learned to separate each class from the rest; a test image is assigned the label of the classifier with the highest response.

We Use the leave-one-out cross-validation framework to tune the parameters. We perform a 5-fold cross-validation where we train on 4 folds and we test on the remaining fold,

one at-a-time, and then the average accuracy is reported. Based on the accuracy we select the best parameters values.

To compute the classification accuracy ((number of well classified images/ number of images) *100) we compare our ground truth labels with the predicted ones and build the confusion matrix for our system figure 4.

B. Experiments and results

The results obtained were run on 30 training and 50 testing examples and a 1024 sized codeword dictionary. We tested the setup on the standard workstation (Xeon X5650 2.67 GHz, 6 core (12 CPU), 32 GB ram) using Matlab. We added the computing time column (in seconds) to evaluate the computational cost of the methods when running the offline process part of the image classification.

The results section is composed of three main sets of experiments, we start with the first phase of the BoF which is the feature extraction, so we compare the results obtained using different descriptors with a standard version of the pipeline which uses NN for coding, no spatial pyramid and simple sum pooling.

We then compare the different encoding methods one can use in the encoding phase of the BoF pipeline; we combined these methods with the different descriptors from the last setup.

At last we compare the pooling phase of the BoF pipeline, here we added the spatial pyramid scheme to the last setup and combined it with the different pooling methods.

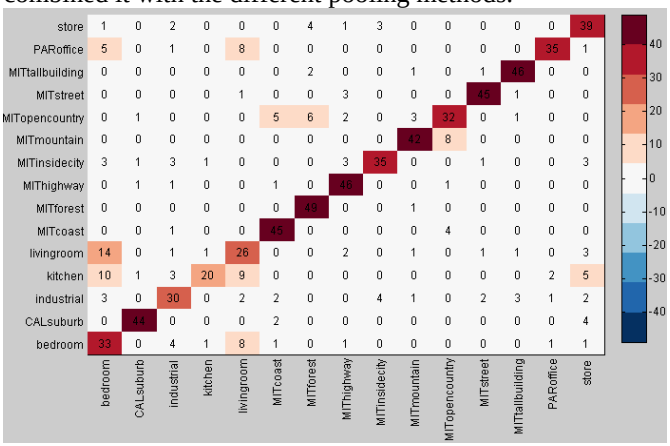

Fig. 4. Confusion matrix for 15 scene categories dataset resulting from our BOF pipeline, using SIFT+GIST as descriptors, LLC for encoding, SPM and max pooling. Along the diagonal we find the number of well-classified test images out of 50. The entry located in the i^{th} row, j^{th}comlumn represents the number of images of class i being misclassified to class j

1) 15 scene categories:

Our first dataset is composed of fifteen scene categories which were gradually built. The initial 8 classes were collected by Oliva and Torralba [26], and then 5 categories were added by Fei-Fei and Perona [56]; finally, 2 additional categories were introduced by Lazebnik et al. [50]. Each category has 200 to 400 images all of them in jpeg format, and the average image size is 300 × 250 pixels. This dataset contains a wide range of outdoor and indoor scene environments. A lot of works were tested on this dataset; most of them focus on dictionary learning, quantization and classification methods and use spatial pyramid matching [50]. It is widely recognized that whole spatial layout information is effective on this dataset. The experimental results are shown in the tables forthcoming.

2) Indoor dataset:

Our second dataset of experiments is the Indoor dataset [57]. It contains 67 Indoor categories, and a total of 15620 images in different resolutions. The number of images varies across categories, but there are at least 100 images per category. All images are in jpeg format. This database is known for being one of the most challenging open problems in high level vision. Most scene recognition models that give good results for outdoor scenes perform poorly in the indoor domain. The main difficulty is that while some indoor scenes can be well characterized by global spatial properties, others are better described by the objects they contain. Thus, to address the indoor scenes recognition problem we need a system that can exploit local and global discriminative information.

3) Descriptor comparison:

We know that in BoF model, keypoints are extracted from the grey level images and the local descriptors don't contain color information. Furthermore, BoF model captures only the local information, losing the overall distribution of visual information. The performance could be enhanced if we can combine global features and local descriptors together.

Both DSIFT/SURF and GIST feature representations describe different features in their target images. So we combine global GIST features with local DSIFT or SURF features in order to see if these complementary features could be used together to increase the image classification accuracy. Table1 shows the accuracy results for the different image descriptors. We used the simple vector quantization scheme with NN for coding, no spatial pyramid and simple sum pooling in order to compare the descriptors.

TABLE I. ACCURACY ACHIEVED WITH THE DIFFERENT FEATURE DESCRIPTORS

	Image descriptor	Accuracy	Time
15 Scene Categories	DSIFT	63.34 %	429.98s
	SURF	48.54 %	291.53s
	DSIFT+GIST	**66.94 %**	831.07s
	SURF+GIST	65.07 %	672.64s
Indoor	DSIFT	**32.18 %**	14810.57s
	SURF	26.39 %	7455.70s
	DSIFT+GIST	30.51 %	15495.33s
	SURF+GIST	29.89 %	10725.74s

TABLE II. ACCURACY ACHIEVED WITH THE DIFFERENT ENCODING METHODS

	Image descriptor	Encoding	Accuracy	Time		Image descriptor	Encoding	Accuracy	Time
15 scene categories	DSIFT	NN	63.34 %	429.98s	Indoor	DSIFT	NN	32.18 %	14810.57s
	DSIFT	KNN	64.67 %	475.21s		DSIFT	KNN	**32.27%**	15884.10s
	DSIFT	LLC	64.67 %	557.40s		DSIFT	LLC	32.03 %	18903.22s
	DSIFT+GIST	NN	**66.94 %**	831.07s		DSIFT+GIST	NN	30.51 %	15495.33s
	DSIFT+GIST	KNN	65.60 %	870.64s		DSIFT+GIST	KNN	30.36 %	17293.81s
	DSIFT+GIST	LLC	66.14 %	959.64s		DSIFT+GIST	LLC	30.30 %	20701.02s
	SURF	NN	48.54 %	291.53s		SURF	NN	26.39 %	7455.70s
	SURF	KNN	49.34 %	319.36 s		SURF	KNN	22.12 %	7688.52s
	SURF	LLC	50.40 %	411.95s		SURF	LLC	26.18 %	9699.77s
	SURF+GIST	NN	65.07%	672.64s		SURF+GIST	NN	29.89 %	10725.74s
	SURF+GIST	KNN	64.27%	699.48s		SURF+GIST	KNN	29.98 %	11553.35s
	SURF+GIST	LLC	64.67 %	791.55s		SURF+GIST	LLC	29.89 %	12429.21s

TABLE III. ACCURACY ACHIEVED USING SPATIAL PYRAMID AND DIFFERENT POOLING METHODS

	Image descriptor	Encoding	SPM	Pooling	Accuracy	Time		Image descriptor	Encoding	SPM	Pooling	Accuracy	Time
15 scene categories	DSIFT	NN	Yes	Sum	64.94 %	467.10s	Indoor	DSIFT	NN	Yes	Sum	32.45 %	17923.76s
	DSIFT	NN	Yes	Max	67.47 %	469.76s		DSIFT	NN	Yes	Max	37.20 %	17784.88s
	DSIFT	KNN	Yes	Sum	67.87%	611.88s		DSIFT	KNN	Yes	Sum	30.60 %	22740.40s
	DSIFT	KNN	Yes	Max	69.87 %	618.42s		DSIFT	KNN	Yes	Max	36.45 %	22297.42s
	DSIFT	LLC	Yes	Sum	65.07 %	694.75s		DSIFT	LLC	Yes	Sum	33.32 %	26032.58s
	DSIFT	LLC	Yes	Max	70.67 %	661.22s		DSIFT	LLC	Yes	Max	37.35 %	25602.24s
	DSIFT+GIST	NN	Yes	Sum	64.80 %	855.31s		DSIFT+GIST	NN	Yes	Sum	29.71 %	18118.85s
	DSIFT+GIST	NN	Yes	Max	73.74 %	878.74s		DSIFT+GIST	NN	Yes	Max	38.99 %	18930.11s
	DSIFT+GIST	KNN	Yes	Sum	64.67 %	944.66s		DSIFT+GIST	KNN	Yes	Sum	29.47 %	22971.09s
	DSIFT+GIST	KNN	Yes	Max	74.14 %	1004.65s		DSIFT+GIST	KNN	Yes	Max	38.42 %	27083.96s
	DSIFT+GIST	LLC	Yes	Sum	64.67 %	1031.66s		DSIFT+GIST	LLC	Yes	Sum	29.47 %	25801.05s
	DSIFT+GIST	LLC	Yes	Max	**75.60 %**	1078.20s		DSIFT+GIST	LLC	Yes	Max	**39.23 %**	26664.71s
	SURF	NN	Yes	Sum	46.27 %	383.22s		SURF	NN	Yes	Sum	23.44 %	12306.90s
	SURF	NN	Yes	Max	56.40 %	382.35s		SURF	NN	Yes	Max	31.41 %	12498.50s
	SURF	KNN	Yes	Sum	44.80 %	645.51s		SURF	KNN	Yes	Sum	15.05 %	20238.74s
	SURF	KNN	Yes	Max	60.94 %	632.27s		SURF	KNN	Yes	Max	29.98 %	20160.84s
	SURF	LLC	Yes	Sum	43.34 %	735.12s		SURF	LLC	Yes	Sum	17.50 %	22281.54s
	SURF	LLC	Yes	Max	58.14 %	690.94s		SURF	LLC	Yes	Max	31.41 %	20763.74s
	SURF+GIST	NN	Yes	Sum	64.14 %	728.63s		SURF+GIST	NN	Yes	Sum	29.50 %	14263.72s
	SURF+GIST	NN	Yes	Max	69.34 %	799.80s		SURF+GIST	NN	Yes	Max	36.81 %	15183.18s
	SURF+GIST	KNN	Yes	Sum	64.54 %	894.66s		SURF+GIST	KNN	Yes	Sum	29.50 %	21631.45s
	SURF+GIST	KNN	Yes	Max	70.40 %	1015.97s		SURF+GIST	KNN	Yes	Max	36.93%	22830.13s
	SURF+GIST	LLC	Yes	Sum	64.40 %	981.35s		SURF+GIST	LLC	Yes	Sum	29.50 %	24736.23s
	SURF+GIST	LLC	Yes	Max	69.74 %	1067.70s		SURF+GIST	LLC	Yes	Max	36.99 %	24167.81s

As we can see from table1 DSIFT descriptor gives better results than SURF descriptor, although it is much more time consuming than the latter.

We then combined the GIST feature vectors with the encoding of DSIFT/SURF descriptors by appending the feature vectors together. The best accuracy rate was achieved by combining DSIFT and GIST for the fifteen dataset with 66.94% and 32.18% using DSIFT solely for the Indoor dataset. DSIFT is slower than SURF but gives better results. When we combined the local descriptors with GIST the time of calculations increased but so did the accuracy. GIST and DSIFT are slow but captures more information.

4) Encoding comparison:

To compare the encoding phase of the pipeline we kept the same codebook with 1024 entries and used the different descriptors with our encoding methods (NN, KNN, and LLC). Table 2 shows the accuracy results for using different coding schemes without spatial pyramid and with sum pooling.

We see that the results are more less the same for the different encoding methods, where there is a minor difference between the accuracies when using the same image descriptor. However, there is an important difference in numbers when the encoding methods are combined with different descriptors. We can see that the best accuracy was achieved for the fifteen dataset by DSIFT+GIST descriptor combined with NN for encoding, and with DSIFT descriptor combined with KNN for the Indoor dataset. As for the computational cost, it is distinct from the results that the most expensive encoding method is LLC followed by KNN then lastly the standard NN for vector quantization.

5) Spatial pyramid with pooling comparison:

For the last phase we compare the different pooling methods.Table3 shows the accuracy results for combining the different descriptors and coding methods with spatial pyramid then aggregating with either max or sum pooling. We notice that the accuracy increased immediately when we added the spatial pyramid scheme and max pooling but so did the computational time. These two recent approaches enabled our setup to achieve the best performance with 75.60% for the fifteen dataset and 39.23% for the Indoor dataset, both using DSIFT+GIST as a descriptor and LLC for encoding.

For the performance we can surely assert that max pooling do improve the overall classification over the widely used sum pooling method, but we can't say much about the difference in computational time between the two approaches.

C. Discussion

In order to evaluate the new approaches in the bag of features model, we compared them to the previous standard methods with respect to the same codebook and many other constraints as mentioned in the implementation details. The paper gives the results using two popular datasets and a linear SVM classification.

The local descriptor DSIFT is the better choice for a good feature extraction and when combined with the global descriptor GIST it captures more information, resulting in an even better performance.

The different encoding methods NN, KNN, LLC gave more less the same results when combined with average pooling and no spatial pyramid, but when combined with spatial pyramid and max pooling LLC gave the best results. The important fact is that spatial pyramid matching and max pooling did significantly improve the performance of all the encoding methods. The improvement of the encoding methods accuracies is about 5 to 10%. This shows that spatial information is important for image classification.

Including both local and global descriptors for the first step of the BoF model, enhanced the performance. Local context and global quantized information were combined to make conventional features more discriminative. Using those resulting robust features we built our vocabulary with K-means algorithm which represents the online process part of the BoF model.

For the encoding step, we used 3 different approaches the hard assignment NN algorithm, a soft assignment distance-based KNN algorithm and a soft assignment reconstruction method LLC. Vector quantization with NN is fast but gives poor results; LLC is much more efficient since the assignment is optimized as to minimize the reconstruction error unlike the purely distance-based assignment with KNN.

The computational efficiency of spatial pyramid matching combined with max pooling for the last step of the BoF model yielded the highest classification rates on challenging data for instance the Indoor dataset.

The importance of this work lies in the details about each level of the pipeline that includes the methods used and their efficiency. This results in an apprehension of each step, which leads us to a better understanding of the BoF model as a whole.

Our setup enabled us to achieve the best performance using DSIFT and GIST as descriptors since they capture more information, then LLC for encoding which was combined with spatial pyramid matching and max pooling. The classification was then measured on the basis of the BoF model by using a linear SVM classifier. These changes in each level of the pipeline resulted in an enhancement of the results over the basic bag of features model. The new approaches maybe a bit costly but the gained efficiency is worth the added computing time.

V. Conclusion

This paper presents an overall insight on the bag of features model. We proposed an image classification pipeline which we used to compare and evaluate the standard bag features methods with the new ones and it appears that these changes in each step of the pipeline did indeed affect the performance.

In fact, we came to the conclusion that the right combination of the methods used in each level of the pipeline as we did in our setup leads to better results. The first amelioration took place in the first step of the pipeline where we combined some descriptors used for features extraction that captures local and global context information. Actually this improvement is due to the global GIST descriptor which adds a spatial envelope to the local features and generates a more complete description of the image.

In the next step we compared different encoding methods, first NN representing the hard assignment vector quantization, then KNN for a distance-based soft assignment and lastly LLC as a reconstruction-based soft assignment. The resulting code vectors were finally used in the last step of the pipeline where another improvement took place, when we added a spatial pyramid layer for when we want to aggregate our code vectors combined with sum pooling as a basic aggregation method or max pooling as a new approach. The SPM max-pooling combination clearly boosted the performance results.

At last, we can say that each step of the pipeline holds part of the final results which means that selecting the methods to use is the most crucial part of the BoF model. In fact, the good performance we achieved is mainly due to the recent approaches we introduced in each step of the pipeline which increased the classification accuracy over the standard bag of features baseline.

References

[1] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In ECCV, 2004.

[2] P. Quelhas, F. Monay, J.-M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. Van-Gool. Modeling scenes with local descriptors and latent aspects. In ICCV, pages 883–890,2005.

[3] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In CVPR, 2007.

[4] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In BMVC, 2011.

[5] Lowe, D.: Distinctive image features from scale-invariant keypoints. 91–110, In IJCV, 2004.

[6] Bay,H,. Tuytelaars, T., &Van Gool, L. "SURF: Speeded Up Robust Features", In 9th ECCV, 2006.

[7] Hsiao, E., Hebert, M.: Occlusion reasoning for object detection under arbitrary viewpoint. In CVPR, 2012.

[8] Hinterstoisser, S., Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., Lepetit, V.: Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In: ICCV, 2011.

[9] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In ICCV, volume 2, pages 1470–1477, Oct. 2003.

[10] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In ICCV, 2005.

[11] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. In IJCV, 60(1):63-86, 2004.

[12] C.Harris and M. Stephens. A combined corner and edge detector. In AVC, pages 147–151, 1988.

[13] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. IVC, 22(10):761–767, 2004.

[14] Agarwal, S., Awan, A., Roth, D.: Learning to detect objects in images via a sparse, part-based representation. In PAMI 26 1475–1490, 2004.

[15] Fergus, R., Fei-Fei, L., Perona, P., Zisserman, A.: Learning object categories from google's image search. In ICCV, 2005

[16] Grauman, K., Darrell, T.: Efficient image matching with distributions of local invariant features. In: CVPR., 2005

[17] Leibe, B., Schiele, B.: Interleaved object categorization and segmentation. In: BMVC, 2003.

[18] Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In: ECCV. I: 128, 2002.

[19] Weber, M., Welling, M., Perona, P.: Unsupervised learning of models for recognition. In: ECCV. I: 18–32, 2000.

[20] Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. CVPR. II: 264–271, 2003.

[21] Leung, T., Malik, J.: Representing and recognizing the visual appearance of materials using three-dimensional textons. In: IJCV 43 29–44, 2001.

[22] Winn, J., Criminisi, A., Minka, T.: Object categorization by learned universal visual dictionary. In: ICCV, 2005.

[23] Agarwal, A., Triggs, B.: Hyperfeatures – multilevel local coding for visual recognition. In: ECCV, 2006.

[24] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In ECCV, 2006.

[25] P. Pham, M. Moens, and T. Tuytelaars. Cross-media alignment of names and faces. In ITM, 12(1):pp.13–27, 2010.

[26] Oliva, A. Torralba: Modeling the shape of the scene: a holistic representation of the spatial envelope. In IJCV, 2001.

[27] J. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor. Improving" bag-ofkeypoints" image categorisation: Generative models and pdf-kernels.TR, University of Southampton, 2005.

[28] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In ICCV, volume 1, pages 604–610,2005.

[29] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In CVPR,2008.

[30] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman.Lost in quantization: Improving particular object retrieval in large scale image databases. In CVPR, 2008.

[31] J.C. van Gemert, C.J. Veenman, A.W.M.Smeulders, and J.-M. Geusebroek. Visual word ambiguity. PAMI, 32(7):1271 –1283, 2010.

[32] J. C. V. Gemert, J. mark Geusebroek, C. J. Veenman, and A. W. M. Smeulders. Kernel codebooks for scene categorization. In ECCV 2008, pages 696–709, 2008.

[33] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In NIPS, pages 801–808, 2006.

[34] J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In CVPR., 2009.

[35] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. CVPR 2010.

[36] J. Yang, K. Yu, and T. Huang. Supervised translation invariant sparse coding. CVPR, 2010.

[37] J. Yang, K. Yu, and T. Huang. Efficient highly over-complete sparse coding using a mixture model. In ECCV 2010, 2010.

[38] Y. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in vision algorithms. In ICML, 2010.

[39] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. CVPR 2010.

[40] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. Machine Learning, 20(3):273–297, September 1995.

[41] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. In JIPM, 25(5):513-523,1988.

[42] MacQueen, J. B. Some Methods for Classification and Analysis of Multivariate Observations, In 5th BSMSP, pp. 281-297. 1967.

[43] DUDA, R., HART, P., & STORK, D. Pattern classification. 2 edn. New York: John Wiley & Sons. 2001.

[44] Vedaldi and B. Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In ICM, pages 1469–1472, Available at : www.vlfeat.org/. 2010

[45] Lindeberg, T.: Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention.In IJCV 11 283–318 . 1993

[46] Voronoi, Georgy . "Nouvelles applications des paramètres continus à la théorie des formes quadratiques". JFDRAM 133(133): 97–178, 1908.

[47] Y.-G. Jiang, C.-W. Ngo, and J. Yang. Towards optimal bag-of-features for object categorization and semantic video retrieval. In ACM ICIVR 2007.

[48] F. Perronnin, J. Sánchez, T. Mensink, Improving the fisher kernel for large-scale image classification, In: ECCV pp. 143–156, 2010.

[49] X. Zhou, K. Yu, T. Zhang, T. S. Huang, Image classification using super-vector coding of local image descriptors, In: ECCV, pp. 141–154, 2010.

[50] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In CVPR, 2006.

[51] B. Scholkopf and A. J. Smola, Learning with Kernels, MIT Press, Cambridge, Mass, USA, 2002.

[52] X. Zhou, N. Cui, Z. Li, F. Liang, and T. Huang. Hierarchical gaussianization for image classification. Proc. of ICCV, 2009.

[53] K. Yu, T. Zhang, and Y. Gong. Nonlinear learning using local coordinate coding. Proc. of NIPS'09, 2009.

[54] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM TIST, 2:27:1--27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2011.

[55] Shabou and H. Le Borgne. Locality-constrained and spatially regularized coding for scene categorization, In CVPR, pp. 3618-3625, 2012.

[56] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," In CVPR, 2005.

[57] A.Quattoni and A.Torralba. Recognizing Indoor scenes. In CVPR 2009.