Image Classification Using Class-Specific Codebooks and Distinctive Local Features with CSO based ANN Classifier

V.S.Renejit[1],D.Ferlin Deva Shahila [2], A.Joel Livin[3]

[1] M.E Student, Applied Electronics, LITES Thovalai ,India, [2] D.Ferlin Deva Shahila , Assistant Professor, ECE , LITES Thovalai, [3] A.Joel Livin, HOD, Assistant Professor, ECE Dept, LITES Thovalai, India,
[1]renejitvs@gmail.com[2] ferlin_franklin@yahoo.com [3] joellivin.ece@lites.edu.in

Abstract

The number of digital images has been increasing exponentially in the last few years. People have problems managing their image collections and finding a specific image. An automatic image categorization system could help them to manage images and find specific images. In this paper, we present a novel method for Image Classification Using Class-Specific Codebooks and Distinctive Local Features with hybrid classifier. Initially, we produce more useful codebooks by utilizing the Self Organizing Map (SOM) clustering and class-specific codebooks in the bag of Words (BOW) model. This contributes to the enactment of the classification performance. Then, the scalability of the image classification system is upgraded by employing a supervised clustering method in codebook generation composed with utilize of distinctive image features during the classification step. Finally, our feature selection method increases the classification performances also here we design a new optimal hybrid classifier called Cuckoo Search Optimization (CSO) based Artificial Neural Network (ANN) which also increase the performance. The proposed method is implemented on Matlab platform and tested on standard database The PASCAL Object Recognition Database Collection and performance are evaluated.

***Keywords**: Bag-of-words, class-specific codebooks, distinctive local features, image classification, self-organizing maps, Cuckoo Search Optimization, Artificial Neural Network.*

I. INTRODUCTION

Image classification has been a challenging research field for decades. Selection and utilization of low-level visual features is a critical step in developing effective classification systems, since these features are used to represent different visual properties of images [1]. The low-level visual features can be defined by either global attributes in images or around the salient image patches locally [2]. The global features, such as color distribution or edge histogram, characterize the entire content in an image, however, their utilization is limited since they are sensitive to imaging conditions such as view changes or occlusions [3]. The local features, on the other hand, are invariant to such conditions and provide more powerful image representations. The main advantage of local features over global features is the stability of the extracted descriptors under different orientation or scale changes, which provides reliable matching between the feature descriptors extracted from the same objects or scenes [4]. Therefore, the research field has been shifted from learning concepts by using global color or texture based features to local features generated around the interest points [5].

Contrary to the global image features, a large number of local descriptors are generated during the extraction of local features, which makes it computationally infeasible to utilize all image features in training and classification tasks [6]. To make it clearer, depending on the content of an image, from several hundreds to thousands of descriptors can be populated from a single image. Moreover, most of the real-world problems require analysis of large image collections, which includes high computational costs during the classifier learning process [7].

In order to address this problem, the bag-of-words (BOW) model, which is a well-known technique originated from the text retrieval domain, has been adapted to the image analysis [8]. The

BOW model, first creates a visual codebook through clustering the local features extracted from some training data, and then represents each image as an unordered collection of the visual-words, which are the cluster centres in the codebook [9-10].

II. RELATED WORKS

Yu Zhang *et al.* [11] have presented a novel method for encoding local binary descriptors for Visual Object Categorization (VOC). Nowadays, local binary descriptors, e.g. LBP and BRIEF, have become very popular in image matching tasks because of their fast computation and matching using binary bitstrings. However, the bottleneck of applying them in the domain of VOC lies in the high dimensional histograms produced by encoding these binary bitstrings into decimal codes. To solve this problem, they proposed to encode local binary bitstrings directly by the Bag-of-Features (BoF) model with Hamming distance. The advantages of this approach are two-fold: (1) It solves the high dimensionality issue of the traditional binary bitstring encoding methods, making local binary descriptors more feasible for the task of VOC, especially when more bits are considered; (2) It is computationally efficient because the Hamming distance, which is very suitable for comparing bitstrings, is based on bitwise XOR operations that can be fast computed on modern CPUs. The proposed method was validated by applying on LBP feature for the purpose of VOC. The experimental results on the PASCAL VOC 2007 benchmark show that our approach effectively improves the recognition accuracy compared to the traditional LBP feature.

Toru Tamaki *et al.* [12] have reported on the performance of a recognition system for classifying NBI images of colorectal tumors into three types (A, B, and C3) based on the NBI magnification findings. To deal with the problem of computer-aided classification of NBI images, they explored a local feature-based recognition method, bag-of-visual-words (BoW), and provide extensive experiments on a variety of technical aspects. The proposed prototype system, used in the experiments, consists of a bag-of-visual-words representation of local features followed by Support Vector Machine (SVM) classifiers. A number of local features are extracted by using sampling schemes such as Difference-of-Gaussians and grid sampling. In addition, in this paper we propose a new combination of local features and sampling schemes. Extensive experiments with varying the parameters for each component are carried out, for the performance of the system is usually affected by those parameters, e.g. the sampling strategy for the local features, the representation of the local feature histograms, the kernel types of the SVM classifiers, the number of classes to be considered, etc. The recognition results are compared in terms of recognition rates, precision/recall, and F-measure for different numbers of visual words. The proposed system achieves a recognition rate of 96% for 10-fold cross validation on a real dataset of 908 NBI images collected during actual colonoscopy, and 93% for a separate test dataset.

Hong Pan *et al.* [13] have proposed a novel model to mine and derive class-specific codebook for categorical object detection and classification. In particular, the codebook is built from a pool of heterogeneous local descriptors using an effective feature selection scheme. The resulting class-specific codebook strengthens the class discriminability by learning the most discriminative part codewords constructed from their preferable local descriptors. The advantage of our class-specific codebook comes from two aspects. 1). As we collect a variety of heterogeneous descriptors during the learning of local codebook, each target object class can always be represented by its most preferable descriptors. Moreover, even each part code word can also find its suitable descriptors. 2). The feature selection process further picks out the most discriminative object parts that separate the target object class from background and other classes. Experimental results on several widely used datasets show that benefits from our class-specific object codebook which fuses complementary visual cues remarkably improve the detection and classification performance for both rigid and non-rigid articulated objects.

Hui-Lan Luo *et al.* [14] have proposed three methods of constructing visual codebook ensembles. The first technique introduced diverse individual visual codebooks by randomly choosing interesting points. The second technique was based on a random sub training image data set with random interesting points. The third method directly utilized different patch information for constructing an ensemble with high diversity. The codebook ensembles were learned to capture and convey image properties from different aspects. Based on these codebook ensembles, different types of image presentations could be obtained. A classification ensemble could be learned based on the different expression data sets from the same training image set. The use of a classification ensemble to categorize new images can lead to improved performance. The detailed experimental analyses on several data sets revealed that the present ensemble approaches were resistant to variations in view, lighting, occlusion, and intra class variations. In addition, they resulted in state-of-the-art performance in categorization.

ByoungChul Ko *et al.* [15] have developed a novel method for detecting view-independent objects in a cluttered background with partial occlusion using shared features. These shared features are selected as common features among classes while the detectors used for each class are trained jointly rather than independently using shared features, which reduces the number of classifiers. They developed an exhaustive greedy selection method for selecting shared features and training their classifiers using only the shared features. The exhaustive greedy selection method randomly selects an exhaustive set of rectangular local features in a normalized object window and selects n significant shared local features from 12 different viewpoints and their effective shared classifiers using random forests. An integral histogram based on oriented-center symmetric local binary pattern (OCS-LBP) descriptor is used to represent a shared feature and to reduce the feature dimensions effectively. The final score is summed bilinearly using the probabilities of neighboring views to determine the location and viewpoint of the object because each view overlaps with neighboring views. The proposed algorithm was successfully applied to the PASCAL VOC 2012 dataset and its detection performance was better than other methods.

III. PROPOSED METHODOLOGY FOR IMAGE CLASSIFICATION

In this proposed methodology, we propose novel approaches to the BOW model aiming to generate more informative codebooks: First, we employ a different clustering method based on the self-organizing maps (SOM) as an alternative to the classical k-Means clustering method. Secondly, instead of building a global codebook for all classes, we employ a supervised clustering approach in the codebook construction through using the class labels available in the training set. In this representation, we train individual codebooks for image classes, which we call as class-specific codebooks in the paper. Thirdly, we introduce a unique feature selection method to determine the distinctive image features based on the class-specific codebooks. To the best of knowledge, the utilization of class-specific codebooks together with the use of distinctive image features have not been investigated before in BOW-based image classification systems, although SOMs are previously utilized for different applications, such as unsupervised image categorization or codebook generation. The process flow of the proposed method is as follows.

- Images from Pascal Collections
- SIFT Based Feature Extraction
- Codebook Generation
- PCA based Feature Selection
- Hybrid CSO based ANN

We build a class-specific codebook for each image category by utilizing the class labels available in training data. During the clustering process, we perform two main steps namely; Feature Extraction and Codebook Generation. In the first step, the SIFT features extracted from the training data are grouped according to the class labels, and in the second step, a new codebook is formed for each image class individually by using the features of the corresponding class only. In this representation, each SOM structure represents a codebook for a specific image class, and the SOM units are utilized as cluster centers, i.e., visual-words or codewords, for that class. In third step, the most discriminative features are selected by using Principal Component Analysis (PCA). Finally, in the last step the visual images are classified by using Cuckoo Search Optimization (CSO) based Artificial Neural Network (ANN). The architecture of the proposed method is shown in below figure 3.1.

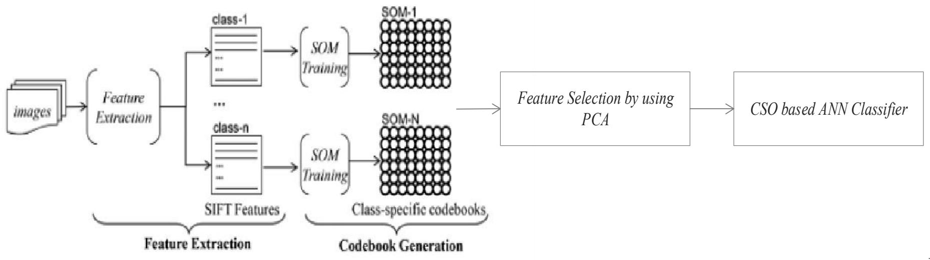

Figure 3.1: Architecture of Proposed Method

3.1 Input Database

The images used in the proposed method are collected from the database "The PASCAL Object Recognition Database Collection" from that database we have collected ten type of image classes like leopard, motorbikes, airplanes, barrel, ceiling fans, cell phones, elephant, laptops, strawberry and tick. Totally 1000 images are collected out that 700 images are used for training purpose and 300 images are used for testing purpose.

3.2 Feature Extraction

The scale invariant feature transform (SIFT) algorithm, developed by Lowe, is an algorithm for image features generation which are invariant to image translation, scaling, rotation and partially invariant to illumination changes and affine projection. Calculation of SIFT image features is performed through the four consecutive steps which are briefly described in the following:

- Scale-space local extrema detection

The features locations are determined as the local extrema of Difference of Gaussians (DOG pyramid). To build the DOG pyramid the input image is convolved iteratively with a Gaussian kernel of σ = 1.6. The last convolved image is down-sampled in each image direction by factor of 2, and the convolving process is repeated. This procedure is repeated as long as the down sampling is possible. Each collection of images of the same size is called an octave. All octaves build together the so-called Gaussian pyramid, which is represented by a 3D function L (x, y, σ). The DOG pyramid D (x, y, σ) is computed from the difference of each two nearby images in Gaussian pyramid. The local extrema (maxima or minima) of DOG function are detected by comparing each pixel with its 26 neighbours in the scale-space (8 neighbours in the same scale, 9 corresponding neighbours in the scale above and 9 in the scale below). The search for extrema excludes the first and the last image in each octave because they do not have a scale above and a scale below respectively. To increase the number of extracted features the input image is doubled before it is treated by SIFT algorithm, which however increases the computational time significantly. In the method presented, the image doubling is avoided but the search for extrema is performed over the whole octave including the first and the last scale. In this case the pixel comparing is carried out only with available neighbours.

- Keypoint localization

The detected local extrema are good candidates for keypoints. However, they need to be exactly localized by fitting a 3D quadratic function to the scale-space local sample point. The quadratic function is computed using a second order Taylor expansion having the origin at the sample point. Then, local extrema with low contrast and such that correspond to edges are discarded because they are sensitive to noise.

- Orientation Assignment

Once the SIFT-feature location is determined, a main orientation is assigned to each feature based on local image gradients. For each pixel of the region around the feature location the gradient magnitude and orientation are computed respectively as:

$$m(x,y)=\sqrt{((L(x+1,y,\sigma)-L(x-1,y,\sigma))^2+(L(x,y+1,\sigma)-L(x,y-1,\sigma))^2} \rightarrow (1)$$

$$\theta(x,y)=\arctan((L(x,y+1,\sigma)-L(x,y-1,\sigma))/(L(x+1,y,\sigma)-L(x-1,y,\sigma)) \rightarrow (2)$$

The gradient magnitudes are weighted by a Gaussian window whose size depends on the feature octave. The weighted gradient magnitudes are used to establish an orientation histogram, which has 36 bins covering the 360 degree range of orientations. The highest orientation histogram peak and peaks with amplitudes greater than 80% of the highest peak are used to create a key point with this orientation. Therefore, there will be multiple keypoints created at the same location but with different orientations.

- Keypoint Descriptor

The region around a keypoint is divided into 4x4 boxes. The gradient magnitudes and orientations within each box are computed and weighted by appropriate Gaussian window, and the coordinate of each pixel and its gradient orientation are rotated relative to the keypoints orientation. Then, for each box an 8 bins orientation histogram is established. From the 16 obtained orientation histograms, a 128 dimensional vector (SIFT-descriptor) is built. This descriptor is orientation invariant, because it is calculated relative to the main orientation. Finally, to achieve the invariance against change in illumination, the descriptor is normalized to unit length.

3.3 Code Generation

The Self-Organizing Map (SOM) was first introduced by Teuvo Kohonen. It can organize multidimensional data in such a way that similar samples are closer to each other and less similar samples are further away from each other. Hence, it can be used to generate a codebook by setting the size of the SOM to be the size of the codebook.

Algorithm 3.1 shows the SOM clustering algorithm. Data is a matrix of input samples, c defines the number of elements in the SOM, l defines how much the learning rate α is decreased after each iteration, and D is the size of the neighborhood. The SOM is a map and each unit of the map defines a cluster centre point. It is initialized randomly in the beginning. *Bmu* is the best matching unit for the $sample_i$. The *findBestMatchingUnit* function finds the closest unit from the SOM and returns the index of the unit, which is stored as *bmu*.

Algorithm 3.1 *SOM = build_SOM(data, c, l, D)*

```
SOM = datarandI
while $(SOM - oldSOM)^2 > threshold$ do
    for I = 1 to numOfSamples do
        bmu = findBestMatchingUnit($sample_i$)
        for j = 1 to D do
            $SOM_{bmu+j} = SOM_{bmu+j} + \alpha * (data_i - SOM_{bmu+j})$
        end for
    end for
    $\alpha = \alpha * l$
end while
```

The SOM uses a neural network that processes information similarly to the brain. It uses competitive learning and it is unsupervised. Hence, it is called self-organizing. Numerous applications have shown that the SOM is capable of managing and organizing large amounts of data. Most SOM applications are visualizations of high dimensional data, but it can be used in many other cases as well.

The structure of a SOM network is presented in Figure 3.2. The SOM can have 1-N dimensions, but usually the SOM is a 2-dimensional grid and each cell is connected to its eight neighbours. The grid is organized in a way that neighbours are similar to each other and dissimilar to items that are further away. The cell in the middle is the winner, i.e., the best matching unit (BMU), and the neighboring cells are updated based on the distance from the middle.

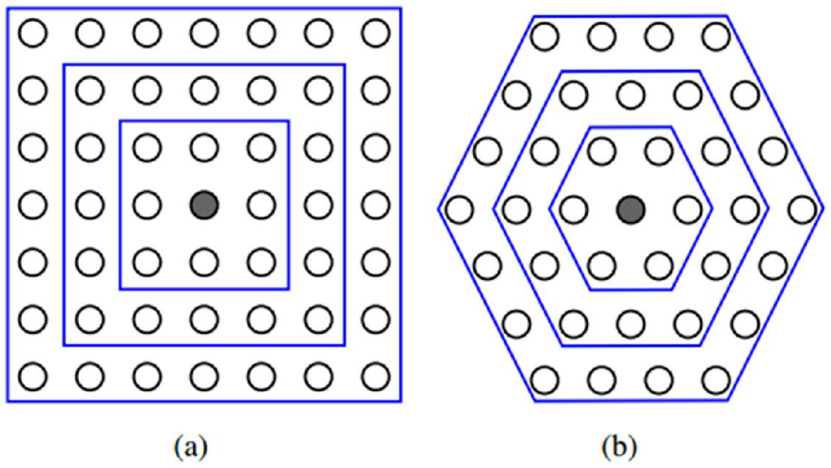

Figure 3.2: Topological neighbourhoods in SOM: (a) 8 neighbours; (b) 6 neighbours

There are three options to initialize a SOM. It can be initialized randomly, using initial samples, or using linear initialization. In random initialization, the weights of the SOM are set randomly. When using initial samples to initialize the SOM, randomly chosen samples are used as the weights in the network. In linear initialization, the SOM is initialized linearly, so the network will be ordered after initialization. Units of the network are chosen randomly, but order is also preserved. Then map units closer are more similar than units that are further away.

After the initialization, the SOM is taught iteratively by feeding it with a large number of training samples. In each round, the learning-rate factor is decreased and teaching turns from global rough teaching to local fine tuning. The SOM computes distances between the cells in the SOM and the sample. The cell with the smallest distance to the input sample is chosen as the BMU. The weights of the BMU and its neighbours are updated.

3.4 Feature Selection

It is possible to use the Principal Component Analysis (PCA) to decrease the number of dimensions of descriptors. It can decrease the dimensions of data while preserving most of the information. The descriptors are histograms of the gradients. Some of the bins in the histograms are more important than the others because some of the bins have more variation than others. They are therefore more important. Every time dimensions are decreased, some information is lost, but dimension decrease is usually unavoidable if memory usage needs to be reduced.

Let us have a vector x with p dimensions. Then

$$\mu_x = E(x) \quad \rightarrow(3)$$

is the mean value vector of x. The covariance matrix of the vector x is

$$C_x = E\left((x-\mu_x)(x-\mu_x)^T\right) \quad \rightarrow(4)$$

where c_{ij} is a component of the C_x and defines the covariances between x_i and x_j. The covariance matrix C_x is symmetric and thus we can solve eigenvalues λ_i and eigenvectors e_i from

$$C_x e_i = \lambda_i e_i,\ i = 1,\ldots.,n \quad \rightarrow(5)$$

By solving the eigenvalues and ordering them in descending order, it is possible to choose the direction for each dimension that has the largest variance and hence contains the most information. The PCA finds dimensions with maximum variances and uses them to describe the data.

3.4 Classification

Neural network is a three-layer standard classifier with n input nodes, l hidden nodes and k output nodes. It is examined that if the two hidden layers are used then one hidden layer is to associate every pair in one important unit and second is regarded as to be the real hidden layer after classifying the input data in the first hidden layer. For the proposed work, the input layers are the features, HU_a Hidden Units and one output unit, f. The structure of the artificial neural network is demonstrated in the following figure 3.3.

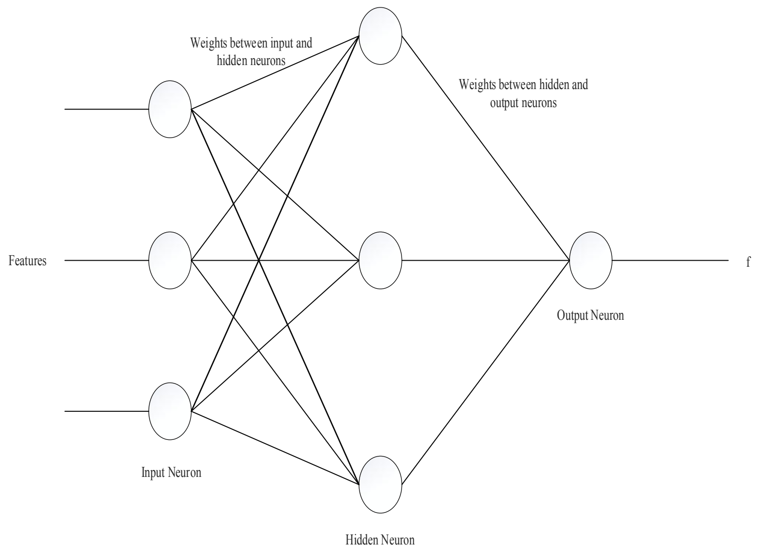
Weights between input and hidden neurons
Weights between hidden and output neurons
Features
f
Output Neuron
Input Neuron
Hidden Neuron

Figure 3.3: Structure of artificial neural network

The steps involved in artificial neural network is given below

1) Set weights for every neuron's except the neurons in the input layer.

2) Generate the neural network with the extracted features as the input units, H Hidden units and f as the output unit.

3) The calculation of the proposed Bias function for the input layer is,

$$X = \sum_{n'=0}^{NF-1} w_{1(n')} A_1(n') + w_{2(n')} A_2(n') + w_{3(n')} A_3(n') + ... + w_{7(n')} A_7(n') \rightarrow (6)$$

The activation function for the output layer is calculated as

$$Active(X) = \frac{1}{1+e^{-X}} \rightarrow (7)$$

4) Identify the learning error as given below

$$E = \frac{1}{NF} \sum_{n'=0}^{NF-1} Y_{n'} - Z_{n'} \rightarrow (8)$$

Where, E- learning rate of FFBNN.

$Y_{n'}$- Desired outputs.

$Z_{n'}$ - Actual outputs.

NF – Number of features

In Neural Network, Back Propagation Algorithm is utilized as the Learning Algorithm. Back Propagation Algorithm is a supervised learning technique and moreover it is an overview of delta rule. To produce the training set, it wants a dataset of the required output for various inputs. Generally, Back Propagation Algorithm is helpful for neural Networks. This learning algorithm needs that the activation function used by the neurons be differentiable. In the process the weights for the neurons of hidden layer and the output layer were assigned using particle swarm optimization algorithm. The Proposed Bias function and the activation function are calculated for the neural networks.

- Weight Optimization by CSO

In ANN, Cuckoo Search Optimization (CSO) is applied to optimize the initial weights of back propagation network, so that the optimized back propagation network has better predicted output. The elements include initializing CSO, determining fitness function, updating position operator, selecting operator, replacing operator, and eliminating operator in order to find the cuckoo individual with the best fitness. The detailed steps of the back propagation algorithm are as follows.

(1) Initializing CSO. Cuckoo individual is encoded in the real-coded form, and each individual is composed of real-number string that consists of the following four parts: connection weights between the hidden layer and output layer, connection weights between the hidden layer and the input layer, the bias in the output layer, and the hidden layer. Each cuckoo individual contains all the weights and bias in back propagation network. According to the weights in back propagation network, a certain back propagation network can be constructed.

(2) Determining Fitness Function. The initial weights of back propagation network can be determined according to the best individual. After training the back propagation network, it is used to predict the output. The fitness value of cuckoo individual *F* is the sum of the absolute error between the desired output and the predicted output *E* as follows:

$$F = E = \frac{1}{NF} \sum_{n'=0}^{NF-1} Y_{n'} - Z_{n'} \qquad \rightarrow (9)$$

where *n* is the node number of the output layer in back propagation network.

(3) Updating Position Operator. A cuckoo (say i) is randomly chosen in the cuckoo population and its position is updated according to (10). The fitness (F_i) of the ith cuckoo at generation t and position $x_i(t)$ is evaluated by (11).

$$x_i^{t+1} = x_i^t + \beta s \otimes H(p_a - \varepsilon) \otimes (x_j^t - x_k^t) \qquad \rightarrow (10)$$

where x_j^t and x_k^t are two different randomly selected cuckoos, $H(u)$ is a Heaviside function, ε is a random number, and s is the step size. On the other hand, the exploration step is implemented by using Lévy flights as follows:

$$x_i^{t+1} = x_i^t + \beta s \otimes H(p_a - \varepsilon) \otimes (x_j^t - x_k^t) \qquad \rightarrow (10)$$

where $L(s, \lambda) = (\lambda\Gamma(\lambda)\sin(\pi\lambda/2)/\pi)(1/s^{1+\lambda})$, $(s, s0 > 0)$, $\beta > 0$ is the scaling factor, and its value can be determined by the problem of interest.

(4) Selecting Operator. Similarly, another cuckoo (say j, $i \neq j$) is randomly chosen in the cuckoo population and its position fitness (F_j) of the ith cuckoo at generation *t* and position $x_j(t)$ is evaluated by (9).

(5) Replacing Operator. If the fitness value of the cuckoo *i* is bigger than the cuckoo *j*, that is, $F_i > F_j$, x_j is replaced by the new solution.

(6) Eliminating Operator. In order to make the population in an optimum state all the time, $ceil(n*p_a)$ worst cuckoos are removed in each generation. At the same time, in order to make the population size unchanged, $ceil(n*p_a)$ cuckoos would randomly be generated. The cuckoos with the best fitness will be passed directly to the next generation. Here, *ceil(x)* rounds the elements of *x* to the nearest integers towards infinity.

Back Propagation network in CSO is similar to an ordinary BP network, and the detailed steps can be represented as follows.

(1) Determining Back Propagation Network Structure. The weights are randomly initialized, and then they are encoded according to the CSO algorithm. The encoded weights are input into the CSO in order to optimize the Back Propagation network, followed by the CSO algorithm.

(2) Construct CSO based Back Propagation Network. The optimal weights obtained from the CSO algorithm are used to construct optimal network. The training set is used to train the network and the training error is calculated. When the training error meets the requirements, training of the network stops.

(3) Predicted Output. The test set is input into the trained network to predict output. The flowchart of the proposed method is shown in figure 4.

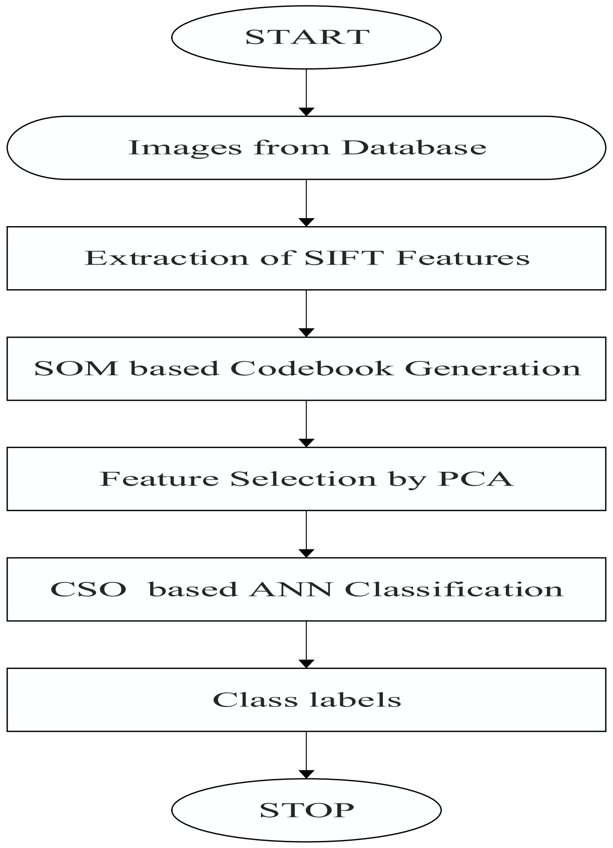
START
Images from Database
Extraction of SIFT Features
SOM based Codebook Generation
Feature Selection by PCA
CSO based ANN Classification
Class labels
STOP

Figure 3.4: Flowchart of Proposed Method

IV. RESULT & DISCUSSION

The proposed method is implemented on Matlab working platform with the following specifications.

Processor	: Intel i5 @ 3GHz
RAM	: 8GB
Operating system	: windows 8
Matlab version	: R2013a

In this paper we have collected the images from The PASCAL Object Recognition Database Collection" from that database we have collected ten type of image classes like leopard, motorbikes, airplanes, barrel, ceiling fans, cell phones, elephant, laptops, strawberry and tick. Some of the sample images present in our dataset are shown in figure 4.1.

Figure 4.1: Sample Images from Pascal Database

Before presenting the classification results, we give some statistical information about the size of the class-specific codebooks used in the experiments. As described in the previous section, for each image class, we build a class-specific codebook using the images that include at least one occurrence of the class by applying SOM method. The size of each codebook is determined by using (10).

$$S = 5 \times \sqrt{D} \qquad \rightarrow (10)$$

where D is the total number of descriptors used in SOM training. Table 4.1 shows some statistics about 5 classes used in the experiments along with the size of the codebooks calculated by using (10).

Table 4.1: Statistical Information of Object Classes

Classes	Total Objects	SIFT Descriptors (*x*1000)	Codebook Sizes
Leopard	1123	289	2550
Motorbikes	306	70	1344
Airplanes	1001	175	2150
Barrel	556	152	1858
Ceiling Fans	213	55	1123
Cellphones	101	35	987
Elephant	457	151	1765
Laptops	341	85	1480
Strawberry	112	40	1100
Tick	385	95	1520

Table 4.2 presents the average quantization errors of the codebooks generated. These are the average distances obtained from multiple runs of clustering methods for the classes shown in Table 4.1. As can be seen in Table 4.2, the image features are closer to the clusters generated by SOM. The reason behind this result is the application of the neighborhood function in SOM, which forces to update the cluster center along with its neighbours.

Table 4.2: Average Quantization errors in SOM Clusters

Classes	SOM
Leopard	10.54
Motorbikes	9.71
Airplanes	9.99
Barrel	9.85

Ceiling Fans	9.61
Cellphones	9.54
Elephant	9.83
Laptops	9.73
Strawberry	9.58
Tick	9.80

Table 4.3 presents the classification results of 10 classes obtained over class-specific codebooks generated by SOM. The use of class-specific codebooks increases the overall classification performance in all classes. Although the size of the codebooks are around 2 K in the class-specific approach, the overall classification performance are increased for all image categories.

Table 4.3: Classification Results

Classes	**SOM**
Leopard	76.5
Motorbikes	77.6
Airplanes	68.5
Barrel	76.4
Ceiling Fans	66.4
Cellphones	53.3
Elephant	71.4
Laptops	72.3
Strawberry	75.5
Tick	78.9

In this part of the experiments, we perform the distinctive feature selection method and investigate its effect in image classification. In addition to the SVM, Naive Bayesian and KNN classifiers used in the previous tests, we employ two different classification techniques, namely the ANN and proposed CSO-ANN throughout the experiments. We present the classification results of 10 object classes using SOM-generated class-specific codebooks in Table 4.4.

Table 4.4: Classification Results of Different Classifiers

Classes	**Classifiers**				
	SVM	**Bayesian**	**KNN**	**ANN**	**Proposed**
Leopard	65.5	41.4	21.3	66.4	70.5
Motorbikes	61.3	43.3	13.3	61.1	65.4
Airplanes	75.4	46.7	20.3	72.3	85.6
Barrel	56.7	41.2	18.5	58.9	63.5
Ceiling Fans	81.2	59.8	54.6	75.3	84.3
Cellphones	75.9	50.3	47.6	78.5	80.2

Elephant	76.8	40.5	41.3	77.4	77.4
Laptops	64.3	50.2	50.4	63.9	64.4
Strawberry	56.7	42.3	38.6	57.6	60.12
Tick	55.2	40.1	38.9	56.9	61.3

From the table we can see that the proposed method has obtained improved classification accuracy compared with other type of classifiers which we can clearly understand from the graph which is shown in figure 4.2.

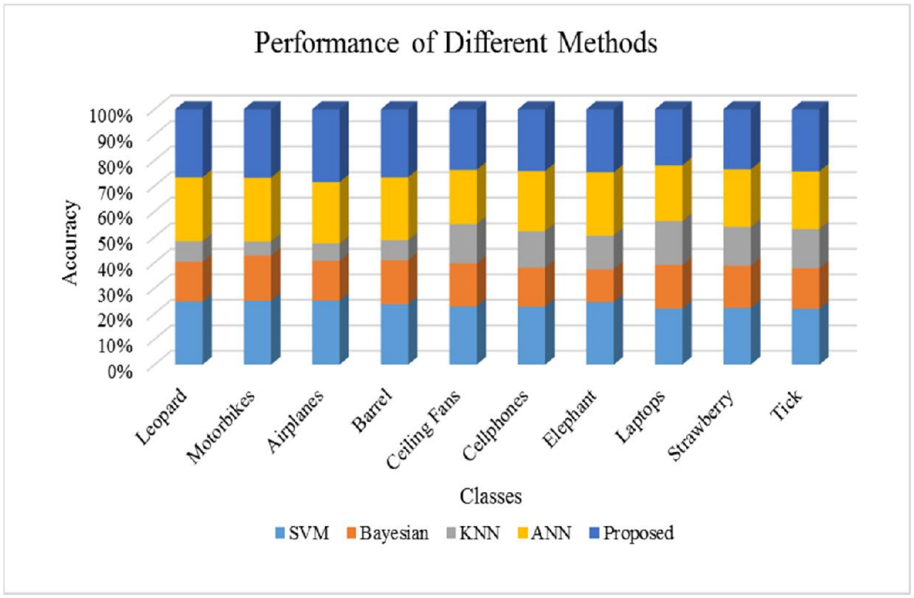

Figure 4.1: Sample Images from Pascal Database

V. CONCLUSION

We present a novel BOW approach for image classification issue manipulating the distinctive local features along with the SOM neural networks. We inspect numerous aspects of the classification system in this paper, such as the consequence of distinctive features in different classification techniques, and the size of codebooks along with the utilization of SOM. The codebooks generated by SOM, and the former gives superior classification results according to our experiments. Also in this proposed method we have designed a new classifier called Hybrid CSO based ANN classifier and compared with other classifiers and result obtained by the proposed method indicate that the proposed method is the best method for object classification in a large collections of image.

REFERENCES

[1] Vailaya A, Figueiredo M. A, Jain, A. K, and Zhang H. J, "Image classification for content-based indexing", IEEE Transactions on Image Processing, Vol. 10, No. 1, pp. 117-130, 2001.

[2] Serrano N, Savakis A. E, and Luo J, "Improved scene classification using efficient low-level features and semantic cues", Pattern Recognition, Vol. 37, No. 9, pp. 1773-1784, 2004.

[3] Nowak E, Jurie F, and Triggs B, "Sampling strategies for bag-of-features image classification", In Proceedings of Springer Berlin Heidelberg on Computer Vision–ECCV, pp. 490-503, 2006.

[4] Lowe D. G, "Distinctive image features from scale-invariant keypoints. International journal of computer vision", Vol. 60, No. 2, pp. 91-110, 2004.

[5] Permuter H, Francos J, and Jermyn I, "A study of Gaussian mixture models of color and texture features for image classification and segmentation", Pattern Recognition, Vol. 39, No. 4, pp. 695-706, 2006.

[6] Gao S, Tsang I. W. H, Chia L. T, and Zhao P, "Local features are not lonely–Laplacian sparse coding for image classification", In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3555-3561, 2010.

[7] Cai H, Yan F, and Mikolajczyk K, "Learning weights for codebook in image classification and retrieval", In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2320-2327, 2010.

[8] Zhou L, Zhou Z, and Hu D, "Scene classification using a multi-resolution bag-of-features model", Pattern Recognition, Vol. 46, No. 1, pp. 424-433, 2013.

[9] Yang L, Jin R, Sukthankar R, and Jurie F, "Unifying discriminative visual codebook generation with classifier training for object category recognition", In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8, 2008.

[10] Wu J, and Rehg J. M, "Beyond the euclidean distance: Creating effective visual codebooks using the histogram intersection kernel", In Proceedings of IEEE International Conference on Computer Vision, pp. 630-637, 2009.

[11] Zhang Y, Zhu C, Bres S, and Chen L, "Encoding local binary descriptors by bag-of-features with hamming distance for visual object categorization", In Advances in Information Retrieval, Springer Berlin Heidelberg, pp. 630-641, 2013.

[12] Toru Tamaki, Junki Yoshimuta, Misato Kawakami, Bisser Raytchev, Kazufumi Kaneda, Shigeto Yoshida, Yoshito Takemura, Keiichi Onji, Rie Miyaki, and Shinji Tanaka, "Computer-aided colorectal tumor classification in NBI endoscopy using local features", Medical Image Analysis, Vol. 17, pp. 78–100, 2013.

[13] Pan H, Zhu Y, Qin A. K, and Xia L, "Mining heterogeneous class-specific codebook for categorical object detection and classification.", In Proceedings of IEEE International Conference on Image Processing (ICIP), pp. 3132-3136, 2013

[14] Hui-Lan Luo, Hui Wei, and Loi Lei Lai, "Creating Efficient Visual Codebook Ensembles for Object Categorization", IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, Vol. 41, No. 2, pp. 238-253, 2011.

[15] ByoungChul Ko, Ji-Hun Jung, and Jae-Yeal Nam, "View-independent object detection using shared local features", Journal of Visual Languages and Computing, Vol. 28, pp. 56–70, 2015.