# Unifying discriminative visual codebook generation with classifier training for object category recognition

Liu Yang, Rong Jin, Rahul Sukthankar, Frédéric Jurie

# Unifying Discriminative Visual Codebook Generation with Classifier Training for Object Category Recognition

Liu Yang[1] Rong Jin[1] Rahul Sukthankar[2,3] Frederic Jurie[4]
yangliu1@cse.msu.edu rongjin@cse.msu.edu rahuls@cs.cmu.edu frederic.jurie@inrialpes.fr
[1]Dept. CSE, Michigan State Univ. [2]Intel Research Pittsburgh [3]Robotics Institute, Carnegie Mellon [4]LEAR Group - CNRS - INRIA
Supplemental material available at: http://www.cs.cmu.edu/~rahuls/pub/

## Abstract

*The idea of representing images using a bag of visual words is currently popular in object category recognition. Since this representation is typically constructed using unsupervised clustering, the resulting visual words may not capture the desired information. Recent work has explored the construction of discriminative visual codebooks that explicitly consider object category information. However, since the codebook generation process is still disconnected from that of classifier training, the set of resulting visual words, while individually discriminative, may not be those best suited for the classifier. This paper proposes a novel optimization framework that unifies codebook generation with classifier training. In our approach, each image feature is encoded by a sequence of "**visual bits**" optimized for each category. An image, which can contain objects from multiple categories, is represented using aggregates of visual bits for each category. Classifiers associated with different categories determine how well a given image corresponds to each category. Based on the performance of these classifiers on the training data, we augment the visual words by generating additional bits. The classifiers are then updated to incorporate the new representation. These two phases are repeated until the desired performance is achieved. Experiments compare our approach to standard clustering-based methods and with state-of-the-art discriminative visual codebook generation. The significant improvements over previous techniques clearly demonstrate the value of unifying representation and classification into a single optimization framework.*

## 1. Introduction

A popular technique for representing image content for object category recognition is the bag of visual words model. The idea is motivated by the success of similar techniques in text information retrieval [15] and text categorization [8], where documents are represented as a vector of word counts. The key idea behind applying this representation for object category recognition is to quantize the continuous high-dimensional space of image features (e.g., SIFT descriptors [10]) to a manageable vocabulary of "visual words". This is typically achieved by grouping the low-level features collected from a large image corpus into a specified number of clusters using an unsupervised algorithm such as k-means [6]. By treating the center of each cluster as a word in a codebook, one can map each feature extracted from a novel image onto its closest visual word, and represent the image by a histogram over the vocabulary of visual words. Several studies [1,4,14,17,18] have shown promising performance for this approach in object category recognition.

One major problem with the current approach is that the unsupervised construction of the visual codebook is unable to take object categories into account; consequently, the codebook may not be particularly informative to the category recognition task. Several approaches have recently been proposed to construct *discriminative* visual vocabularies that explicitly incorporate category-specific information [3, 13, 18]. Farquhar *et al*. [3] construct class-specific visual vocabularies using the Maximum A Posterior (MAP) approach. Winn *et al*. [18] reduce a large vocabulary constructed by a clustering algorithm using pair-wise word merging. Moosmann *et al*. [13] build discriminative visual word vocabularies using randomized clustering forests. Larlus and Jurie [9] propose a generative model that integrates visual vocabulary construction with classifier training. Perronnin *et al*. [14] characterize images using a set of category-specific histograms, where each histogram describes whether the content can best be modeled by the universal vocabulary or by its corresponding category vocabulary. However, despite these efforts, there are two significant shortcomings with most of the existing approaches (Figure 1a) for bag-of-words based object category recognition:

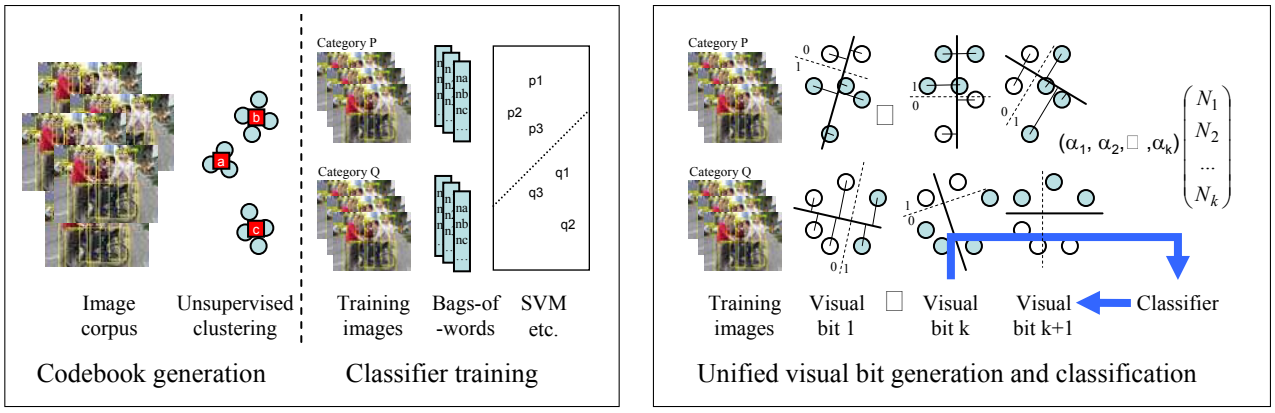


Figure 1. (a) standard approach: visual codebook generation and classifier training are two disconnected stages. (b) proposed approach: the two phases are interleaved into a single optimization framework, where the representation and the classifiers are iteratively refined.

- Existing approaches typically separate the process of visual codebook generation from the process of classifier training. This is suboptimal since the set of visual words, while individually discriminative, could overlap with each other in terms of the information provided to the target object categories. As a result, the combination of identified visual words may not be the most effective for classifier training.
- Another shortcoming with almost all of the existing approaches is that mapping the image features to distinct visual words forces the relationship between any two image features to be all-or-none, i.e., two image features are either assigned to the same visual word or to different visual words. This may not be appropriate when two different image features share partial similarity. This is supported by Perronnin *et al*. [14], where a soft assignment is employed to map image features to visual words.

We propose a novel framework for object category recognition that addresses both of these issues by unifying the process of building a visual representation with that of training classifiers (Figure 1b). Rather than quantizing image features using a single universal codebook, we construct category-specific visual words for each feature. Each visual word is composed of a sequence of "visual bits" that capture different aspects of the image feature. A visual bit is generated by thresholding the projection[1] of the image feature along a (category-specific) direction; additional bits are incrementally generated to refine the visual word during the progress of the algorithm. This scheme solves the problem of expressing partial similarity between features since their corresponding visual words can agree on some visual bits and differ on others. An image (which can contain objects from multiple categories) is represented by a set of vectors, one for each object category. Each element in this vector corresponds to a category-specific visual bit for the visual words present in the image and encodes a count of their values; this vector is analogous to a category-specific histogram of visual words, except that the counts are aggregated at the bit rather than word level. We train a discriminative classifier for each object category that processes the image vector and indicates the degree to which a given image matches a particular object category. Unlike standard image representations, where the codebook is generated in a single step before classification, our approach interleaves stages of representation and classifier refinement. Based on the performance of the classifier, we incrementally refine the visual words by adding a new visual bit. Then we retrain the classifier to incorporate the updated representation. Repeating this process is guaranteed to improve recognition accuracy (on the training set). Although visual bits are conceptually similar to the notion of *chopping* in Fleuret and Blanchard [5], the latter was limited to pairwise image comparisons while ours are employed for the more difficult task of category recognition.

Our work is similar in spirit to Larlus and Jurie [9]'s efforts to unify codebook generation and classifier training into a single generative model. The key difference is that we focuse on developing a *discriminative* classifier for object category recognition, and is motivated by the observation that discriminative classifiers can outperform generative classifiers for text categorization, particularly when the vocabulary size is large [8].

## 2. A Unified Optimization Framework for Object Category Recognition

This section presents our proposed framework for object category recognition. We first describe the unified framework, and then present an iterative algorithm based

[1]We describe the approach using linear projections. In practice, we observe slight benefits by employing kernel-based nonlinear projections.

on bound optimization to effectively minimize our objective function.

### 2.1. The Unified Framework

Let $\mathcal{T} = (X_1, X_2, \ldots, X_N)$ denote the set of training images where $N$ is the total number of training examples. Each image $X_i$ is represented by a bag of low-level image features $(\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \ldots, \mathbf{x}_{i,n_i})$ where $n_i$ is the number of low-level features extracted from image $X_i$. Every image feature $\mathbf{x}_{i,j} \in \mathbb{R}^d$ is a vector of $d$ dimensions. Let $\mathbf{y}_i \subseteq \{1, 2, \ldots, m\}$ denote the subset of categories that are assigned to image $X_i$ where $m$ is the total number of object categories. As described in the introduction, our goal is to identify the visual bits that are maximally informative to the target object categories, and a classification model that linearly combines the visual bits to effectively determine the object categories present in any given image.

**Visual bits** A key challenge in defining the objective function arises from the fact that each image is comprised of a multitude of low-level features, and it is unclear which subsets of these features are related to which of the (multiple) categories by which each image is labeled. We address this challenge by introducing the *visual bit* function $g_k(\mathbf{x}, y) : \mathbb{R}^d \times \{1, 2, \ldots, m\} \rightarrow \{0, 1\}$ that determines the relevance of the low-level feature $\mathbf{x}$ to category $y$ in the $k$th iteration. In particular, visual bit $g_k(\mathbf{x}, y)$ outputs 1 when the low-level feature $\mathbf{x}$ is deemed to be relevant to category $y$ in the $k$th iteration, and 0 otherwise. A linear classifier is used to represent the visual bit function, i.e.,

$$g_k(\mathbf{x}, y) = I(\mathbf{x}^\top \mathbf{w}_k^y - b_k^y) = \begin{cases} 1 & \mathbf{x}^\top \mathbf{w}_k^y > b_k^y \\ 0 & \mathbf{x}^\top \mathbf{w}_k^y \leq b_k^y \end{cases}, \quad (1)$$

where $\mathbf{w}_k^y \in \mathbb{R}^d$ is a weight vector and $b_k^y$ is the threshold. Alternately, visual bits can be generated using a nonlinear function, such as the RBF kernel. The *relevance function* of the low-level feature $\mathbf{x}$ to category $y$ across $T$ iterations is then expressed by

$$f(\mathbf{x}, y) \quad = \quad \sum_{k=1}^{T} \alpha_k g_k(\mathbf{x}, y), \quad (2)$$

where $\alpha_k \geq 0$ is a combination weight.

**Loss function** Using the relevance function defined above, we follow the exponential model and define the probability of classifying a low-level feature $\mathbf{x}_{i,j}$ of image $X_i$ into one of the assigned categories $y \in \mathbf{y}_i$ as follows:

$$e(\mathbf{x}_{i,j}, y) = \frac{\exp(f(\mathbf{x}_{i,j}, y))}{\sum_{y'=1}^{m} \exp(f(\mathbf{x}_{i,j}, y'))}, \quad (3)$$

Given an image $X_i = (\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,n_i})$ and one of its assigned categories $y \in \mathbf{y}_i$, the loss function for not classifying $X_i$ into category $y$, denoted by $l(X_i, y)$, is then defined as follows:

$$l(X_i, y) = \frac{n_i}{\sum_{j=1}^{n_i} e(\mathbf{x}_{i,j}, y)}. \quad (4)$$

Evidently, the larger the probability $e(\mathbf{x}_{i,j}, y)$, the smaller the loss function $l(X_i, y)$.

**Remark I** Note that we did not define the loss function as

$$l(X_i, y) = 1 - \frac{1}{n_i} \sum_{j=1}^{n_i} e(\mathbf{x}_{i,j}, y). \quad (5)$$

The key difference between Eqns. (4) and (5) is how they penalize images with low classification probabilities $e(\mathbf{x}_{ij}, y)$ for all features. Eqn. (5) would assess such images a relatively low penalty since $l(X_i, y)$ is related to the average of the individual feature losses. By contrast, Eqn. (4) is much harsher on such images because $l(X_i, y)$ grows rapidly when the sum of $e(\mathbf{x}_{i,j}, y)$ is small. If we use the quantity $\min_i \sum_j e(\mathbf{x}_{i,j}, y_i)/n_i$ as a measure of the classification margin, we find that Eqn. (4) is likely to generate a larger classification margin than Eqn. (5), and is therefore a better choice of loss function.

**Remark II** We also would like to point out the close relationship between the loss function in Eqn. (4) and the exponential loss function (i.e., $\sum_{i=1}^{N} \exp(-H(X_i, y_i))$) employed by AdaBoost [16]. To illustrate this connection, we consider a binary image classification problem with $y \in \{-1, +1\}$, and assume that each image $X_i$ only consists of a single low-level feature $\mathbf{x}_i$. We then have $l(X_i, y)$ simplified as

$$l(X_i, y) - 1 = \exp\left( - \sum_{k=1}^{T} \alpha_k \left[ g_k(\mathbf{x}_i, y_i) - g_k(\mathbf{x}_i, -y_i) \right] \right).$$

The above loss function is almost identical to the exponential loss function used by AdaBoost if we define $H(X_i, y_i) = g_k(\mathbf{x}_i, y) - g_k(\mathbf{x}_i, -y)$. Minimizing an exponential loss function not only reduces the training error but also maximizes the classification margins; this has been the key to the success of AdaBoost [16].

**Objective function** Since each image $X_i$ can be simultaneously assigned to multiple object categories, we define the loss function for image $X_i$ as $l(X_i) = \sum_{y \in \mathbf{y}_i} l(X_i, y)$. Finally, the overall objective function is defined as the sum of the individual loss functions for all the training images,

i.e.,

$$\mathcal{L} = \sum_{i=1}^{N} l(X_i) = \sum_{i=1}^{N} \sum_{y \in \mathbf{y}_i} l(X_i, y). \quad (6)$$

Given the objective function defined above, the goal of the proposed framework is to find the visual bits $\{g_k(\mathbf{x}, y)\}_{k=1}^{T}, y = 1, \ldots, m$ and combination weights $\{\alpha_k\}_{k=1}^{T}$ that minimize $\mathcal{L}$.

## 2.2. Optimization Algorithm

To identify the optimal visual bits and combination weights that minimize the objective function $\mathcal{L}$, we propose an iterative algorithm based on bound optimization and coordinate descent. This algorithm iteratively generates visual bits $g_k(\mathbf{x}, y), y = 1, \ldots, m$ based on the performance of the classifier from the current iteration, and then computes the optimal combination weight for the new visual bits. In the following discussion, we follow the convention where the quantities for the current iteration are marked by the symbol $'$ to distinguish them from the quantities obtained in previous iterations.

We denote by $f(\mathbf{x}, y)$ the relevance function obtained thus far, and by $\{g(\mathbf{x}, y)\}_{y=1}^{m}$ and $\alpha$ the new visual bits and their combination weight obtained in the current iteration. Then the new relevance function and loss function, denoted by $f'(\mathbf{x}, y)$ and $l'(\mathbf{x}_{i,j}, y)$, are computed as follows

$$\begin{aligned} f'(\mathbf{x}, y) &= f(\mathbf{x}, y) + \alpha g(\mathbf{x}, y) \\ l'(\mathbf{x}_{i,j}, y) &= \frac{\exp(f(\mathbf{x}_{i,j}, y)) \exp(\alpha g(\mathbf{x}_{i,j}, y))}{\sum_{y'=1}^{m} \exp(f(\mathbf{x}_{i,j}, y')) \exp(\alpha g(\mathbf{x}_{i,j}, y'))} \\ &= \frac{e(\mathbf{x}_{i,j}, y) \exp(\alpha g(\mathbf{x}_{i,j}, y))}{\sum_{y'=1}^{m} e(\mathbf{x}_{i,j}, y') \exp(\alpha g(\mathbf{x}_{i,j}, y'))}. \end{aligned}$$

Finally, the new objective function, denoted by $\mathcal{L}'$, is expressed as follows:

$$\mathcal{L}' = \sum_{i=1}^{N} \sum_{y \in \mathbf{y}_i} l'(X_i, y) = \sum_{i=1}^{N} \sum_{y \in \mathbf{y}_i} \frac{n_i}{\sum_{j=1}^{n_i} l'(\mathbf{x}_{i,j}, y)}$$

Our goal is to search for the visual bits $\{g(\mathbf{x}, y)\}_{y=1}^{m}$ and combination weight $\alpha$ that minimizes the objective function $\mathcal{L}'$.

Due to the complexity of $l'(\mathbf{x}, y)$, directly optimizing $\mathcal{L}'$ is computationally difficult. In particular, the combination weight $\alpha$ is coupled with visual bits $g(\mathbf{x}, y)$ through $l'(\mathbf{x}, y)$, making the optimization problem challenging. The following two lemmas provide the basis for decoupling $\alpha$ from $g(\mathbf{x}, y)$ in $\mathcal{L}'$.

**Lemma 1** *The following inequality holds for any image $X_i$, visual bit $g(\mathbf{x}, y)$, and combination weight $\alpha \geq 0$*

$$\frac{l'(X_i, y)}{l(X_i, y)} \leq \sum_{j=1}^{n_i} \sum_{y'=1}^{m} \frac{q_{i,j}(y) e(\mathbf{x}_{i,j}, y')}{\exp(\alpha[g(\mathbf{x}_{i,j}, y) - g(\mathbf{x}_{i,j}, y')])},$$

*where*

$$q_{i,j}(y) = \frac{e(\mathbf{x}_{i,j}, y)}{\sum_{j'=1}^{n_i} e(\mathbf{x}_{i,j'}, y)}. \quad (7)$$

**Lemma 2** *The following inequality holds for any visual bit $g(\mathbf{x}, y)$ and combination weight $\alpha \geq 0$*

$$\exp(\alpha[g(\mathbf{x}, y) - g(\mathbf{x}, y')]) \leq \frac{e^{-3\alpha} + e^{3\alpha} + 1}{3} - \frac{1 - e^{-3\alpha}}{3}(g(\mathbf{x}, y') - g(\mathbf{x}_{i,j}, y)).$$

Both lemmas can be verified by using the convexity of reciprocal function and exponential function. The proofs of Lemmas 1 and 2 are provided in Appendices A and B in the supplemental material.

Using Lemmas 1 and 2, we obtain the following theorem that separates $\alpha$ from $g(\mathbf{x}, y)$. As a result, we can search for the optimal visual bits independently from the combination weight.

**Theorem 1** *The following inequality holds for any visual bit $g(\mathbf{x}, y)$ and combination weight $\alpha \geq 0$*

$$\mathcal{L}' \leq \frac{e^{-3\alpha} + e^{3\alpha} + 1}{3} \mathcal{L} - \frac{1 - e^{-3\alpha}}{3} \sum_{y=1}^{m} \sum_{i=1}^{N} \sum_{j=1}^{n_i} g(\mathbf{x}_{i,j}, y) T_{i,j}^{y},$$

*where*

$$T_{i,j}^{y} = \sum_{y' \in \mathbf{y}_i} l(X_i, y') q_{i,j}(y') [\delta(y, y') - e(\mathbf{x}_{i,j}, y))]. \quad (8)$$

The above theorem can easily verified by combining the results in Lemmas 1 and 2. The proof of Theorem 1 is given in Appendix C in the supplemental material.

Using the result in Theorem 1, to minimize the objective function $\mathcal{L}'$, we can minimize its upper bound. This is equivalent to maximizing the sum $\sum_{y=1}^{m} \sum_{i=1}^{N} \sum_{j=1}^{n_i} g(\mathbf{x}_{i,j}, y) T_{i,j}^{y}$. Using the definition of visual bits in Eqn. (1), we arrive at the following optimization problem for every category $y$:

$$\min_{\mathbf{w}_y, b_y} \sum_{i=1}^{N} \sum_{j=1}^{n_i} [1 - I(z_{i,j}^{y}(\mathbf{x}_{i,j}^{\top} \mathbf{w}_y - b_y))] |T_{i,j}^{y}|, \quad (9)$$

where $z_{i,j}^{y} = \text{sign}(T_{i,j}^{y}) \in \{-1, +1\}$. Note that the above optimization problem is independent from the combination weight $\alpha$. The problem in Eqn. (9) is difficult to optimize due to the indicator function $I(\cdot)$. We simplify the calculation by replacing $1 - I(h)$ with its upper bound $\max(0, 1 - h)$; the resulting optimization problem becomes

$$\min_{\mathbf{w}_y, b_y} \sum_{i=1}^{N} \sum_{j=1}^{n_i} |T_{i,j}^{y}| \max(0, 1 - z_{i,j}^{y}(\mathbf{x}_{i,j}^{\top} \mathbf{w}_y - b)). \quad (10)$$

The following theorem shows an equivalent form for the above optimization problem that allows for more efficient computation.

**Theorem 2** *The optimization problem in Eqn. (10) is equivalent to the following problem:*

$$\begin{aligned} \min_{\mathbf{w}_y, b_y} \quad & \sum_{i=1}^{N}\sum_{j=1}^{n_i} |T(y)_{i,j}|\varepsilon_{i,j}^y \\ \text{s. t.} \quad & z_{i,j}^y \left(\mathbf{w}_y^\top \mathbf{x}_{i,j} - b_y\right) \geq 1 - \varepsilon_{i,j}^y, \varepsilon_{i,j}^y \geq 0. \end{aligned} \tag{11}$$

The above theorem can be proved by introducing a slack variable $\varepsilon_{i,j}^y$ to upper bound every $\max(0, 1 - z_{i,j}^y(\mathbf{x}_{i,j}^\top \mathbf{w}_y - b))$. To stabilize the solution, we introduce a regularizer $\lambda|\mathbf{w}|_2^2/$ into the objective function where $\lambda$ is the regularization parameter, i.e.,

$$\begin{aligned} \min_{\mathbf{w}_y, b_y} \quad & \frac{\lambda}{2}|\mathbf{w}_y|_2^2 + \sum_{i=1}^{N}\sum_{j=1}^{n_i} |T(y)_{i,j}|\varepsilon_{i,j}^y \\ \text{s. t.} \quad & z_{i,j}^y \left(\mathbf{w}_y^\top \mathbf{x}_{i,j} - b_y\right) \geq 1 - \varepsilon_{i,j}^y, \varepsilon_{i,j}^y \geq 0 \end{aligned} \tag{12}$$

Note that the above problem is similar to the optimization problem solved by the support vector machine (SVM) except that each example is weighted by the factor $|T_{i,j}^y|$. This observation enables us to exploit existing off-the-shelf SVM packages using the following sampling strategy. We simply sample $K$ low-level features according to their weights $|T(y)_{i,j}|$ and then train an SVM classifier using the sampled data. This sampling approach also allows us to handle a large number of training images with modest computational cost.

Finally, the optimal $\alpha$ that minimizes the upper bound of $\mathcal{L}'$ can be computed as

$$\alpha = \frac{1}{2}\log\left(\frac{\sum_{i=1}^{N}\sum_{j=1}^{n_i} A_{i,j}(0)B_{i,j}(1)}{\sum_{i=1}^{N}\sum_{j=1}^{n_i} A_{i,j}(1)B_{i,j}(0)}\right), \tag{13}$$

where

$$\begin{aligned} A_{i,j}(z) &= \sum_{y=1}^{m} e(\mathbf{x}_{i,j}, y)\delta(g(\mathbf{x}_{i,j}, y), z) \\ B_{i,j}(z) &= \sum_{y \in \mathbf{y}_i} l(X_i, y)q_{i,j}(y)\delta(g(\mathbf{x}_{i,j}, y), z) \end{aligned}$$

The derivation for the optimal $\alpha$ is given in Appendix D of the supplemental material. Figure 2 describes the steps for the proposed boosting algorithm for object category recognition.

Finally, although the above procedure minimizes the upper bound of the objective function, we can show that the true objective is indeed reduced exponentially provided the optimal $\alpha$ obtained in each iteration is not very small. This result is summarized by the following theorem.

- Initialize $f(\mathbf{x}_{i,j}, y) = 0$ for every low-level feature $\mathbf{x}_{i,j}$ and every class $y$
- For $k = 1, 2, \ldots, T$
  - Compute $e(\mathbf{x}_{i,j}, y)$ and $q_{i,j}(y)$ for every low-level feature $\mathbf{x}_{i,j}$ and every class $y$ using Eqns. (3) and (7)
  - Compute $l(X_i, y)$ for each image $X_i$ and every class $y$ using Eqn. (4)
  - Compute $T_{i,j}^y$ using Eqn. (8) and pseudo class label $z_{i,j}^y = \text{sign}(T_{i,j}^y)$ for every low-level feature $\mathbf{x}_{i,j}$ and for every class $y$.
  - For each class $y = 1, \ldots, m$
    - Sample $K$ low-level features according to the weights $|T_{i,j}^y|$
    - Compute $\mathbf{w}_y$ and $b_y$ for each class $y$ using Eqn. (12)
    - Compute $g(\mathbf{x}_{i,j}, y)$ using Eqn. (1).
  - Compute $\alpha$ using Eqn. (13)
  - Exit loop if $\alpha \leq 0$
  - Update $f(\mathbf{x}_{i,j}, y) \leftarrow f(\mathbf{x}_{i,j}, y) + \alpha g(\mathbf{x}_{i,j}, y)$

Figure 2. Unified algorithm for object category recognition

**Theorem 3** *Let $\alpha_t$, $q_{i,j}^t(y)$, $e_t(X_i, y)$, $A_{i,j}^t(z)$ and $B_{i,j}^t(z)$ be the quantities computed in the $t$th iteration by running the algorithm in Figure 2. Then, the objective function after $T$ iterations, denoted by $\mathcal{L}_T$, is bounded as follows:*

$$\mathcal{L}_T \leq \mathcal{L}_0 \prod_{t=1}^{T}\left(1 - \frac{(\exp(\alpha_t) - 1)^2}{\exp(2\alpha_t) + \eta_t}\right), \tag{14}$$

*where*

$$\eta_t = \frac{\sum_{i=1}^{N}\sum_{j=1}^{n_i}\sum_{z=0}^{1} B_{i,j}(z)\left(\sum_{y'=1}^{m}\delta(g_t(\mathbf{x}_{i,j}, y'), z)\right)}{\sum_{i=1}^{N}\sum_{j=1}^{n_i} A_{i,j}(0)B_{i,j}(1)}$$

The proof is in Appendix E of the supplemental material. Figure 3 shows an example of how the objective function decreases steadily over 175 iterations.

After training, we obtain a set of visual bits $g_k(\mathbf{x}, y), k = 1, 2, \ldots, T$ for each class $y$ (one bit is generated for each category in each iteration). To determine if a test image $X = (\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_t)$ should be assigned to object category $y$, we first compute the binary representation of each low-level feature using the visual bits. We then compute a representation for the image $X$ that consists of a vector of counts $\mathbf{h} = (h_1, h_2, \ldots, h_T)$. Each element in this image vector corresponds to a visual bit and is simply a count of the number of features in this image whose corresponding bit

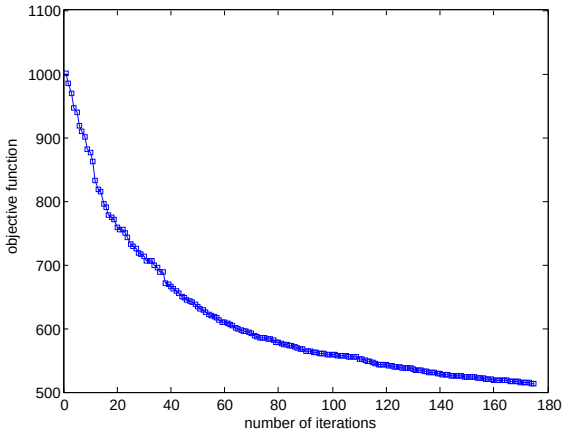


Figure 3. Objective function at different iterations

is set. Since the number of features in the image is known, this image vector can be interpreted as a histogram. A score is computed for $X$ by $\sum_{k=1}^{T} \alpha_k h_k$. Image $X$ is assigned to category $y$ if its score is larger than a threshold that is also learned from the data.

While the earlier discussion describes each visual bit as being generated by thresholding a linear projection of the data, one can easily employ a non-linear kernel, such as an RBF. Experiments indicate that the RBF kernel improves performance for some object categories at the expense of learning an additional parameter using cross-validation.

## 3. Evaluation

We evaluate our approach on the PASCAL VOC Challenge 2006 data set [2]. This challenging dataset contains 5304 images with 9507 annotated objects. Ten annotated object classes are provided: bicycle, bus, car, motorbike, cat, cow, dog, horse, sheep and person. The populations of training/validation and test sets are well balanced across the distributions of images and objects by class. As a multi-object classification task, for each of the ten object classes, the goal is to predict the presence/absence of at least one object of that class in a test image. The binary classification performance for each object class, is measured quantitatively by the area under the ROC curve (AUR)

Our evaluation focuses on the challenging problem of object category recognition given a limited number of labeled images. Our training set (common across all methods) consists of 100 randomly-selected images. Classification models are learned from this common small training set, by different algorithms. The AUR is computed based on the prediction for the 500 randomly selected PASCAL testing images. Each experiment is repeated ten times, and the AUR averaged over these trials is reported.

### 3.1. Local Detectors and Features

Our experiments follow the methodology described in [11]. For image representation, we employ the Harris-Laplace interest point detector [12], which identifies corner-like regions, in conjunction with the SIFT [10] descriptor to extract low-level features from each image. For consistency, we ensure that the same set of low-level image features are used by all of the methods in our experiments.

### 3.2. Implementation and Baseline Methods

We implement the proposed approach in Figure 2 by using an RBF kernel for visual bit classification. The maximum number of iterations is set to 300. We compare our approach against two baseline methods: (1) a standard method that constructs a visual dictionary using k-means followed by an SVM for classification [19], and (2) a state-of-the art method for discriminative codebook generation [9]. We summarize these briefly here.

Our implementation for the first baseline (denoted **KM-SVM**) builds a visual codebook using k-means with 1000 cluster centers. Each low-level feature is quantized to its nearest cluster; thus, an image is represented by a 1000-bin histogram over these clusters. The SVM classifier uses the $\chi^2$ kernel [7] computed over this bag-of-features. We find that the $\chi^2$ kernel is somewhat sensitive to the choice of $C$; our experiments use $C = 5$.

The second baseline (denoted **ERCF**) builds a discriminative visual codebook using the extremely-random classification forest algorithm [13]. ERCF starts by building randomized decision trees to predict object category labels from a low-level image feature. This tree is not used a classifier; rather, the leaf of the decision tree is employed as a spatial code. Since a random decision tree is likely to have a high variance ERCF employs a set of trees, and the labels of their leaf nodes are stacked to form the quantized representation for the feature. An image can then be represented by a histogram over these visual words.

### 3.3. Results

Table 1 summarizes the AUR results for all three methods. Our algorithm outperforms the baseline KM-SVM on every category and demonstrates significant improvements over ERCF on 8/10 classes. Specifically, in comparison to ERCF, we see dramatic improvements in the "bus" category, whose AUR jumps from 0.708 to 0.930, and strong improvements on "sheep" from 0.747 to 0.842, "car" from 0.731 to 0.875, and "dog" from 0.706 to 0.761. On "horse" and "person", the proposed algorithm is outperformed slightly by ERCF (but not by KM-SVM).

A more careful examination indicates that our method also noticeably reduces the *standard deviation* in AUR to below 0.02 for all the categories. The most significant case

Table 1. AUR on PASCAL 2006 with 100 training examples.

| Class | KM-SVM | ERCF | Our Method |
|---|---|---|---|
| sheep | $0.551 \pm 0.046$ | $0.747 \pm 0.017$ | $0.842 \pm 0.008$ |
| bus | $0.618 \pm 0.030$ | $0.708 \pm 0.024$ | $0.930 \pm 0.005$ |
| cat | $0.697 \pm 0.011$ | $0.753 \pm 0.015$ | $0.759 \pm 0.016$ |
| bicycle | $0.750 \pm 0.026$ | $0.744 \pm 0.021$ | $0.782 \pm 0.021$ |
| car | $0.654 \pm 0.043$ | $0.731 \pm 0.019$ | $0.875 \pm 0.007$ |
| cow | $0.519 \pm 0.026$ | $0.751 \pm 0.026$ | $0.790 \pm 0.017$ |
| dog | $0.670 \pm 0.011$ | $0.706 \pm 0.026$ | $0.761 \pm 0.012$ |
| horse | $0.503 \pm 0.016$ | $0.712 \pm 0.025$ | $0.671 \pm 0.009$ |
| motor | $0.496 \pm 0.017$ | $0.733 \pm 0.019$ | $0.782 \pm 0.013$ |
| person | $0.551 \pm 0.035$ | $0.729 \pm 0.015$ | $0.722 \pm 0.007$ |

is "bus", whose standard deviation in AUR is reduced from $0.024$ for ERCF to $0.005$. The large variance in AUR of the baseline models indicates that, given a small number of training images, many feature clusters could only appear in a few training images. As a result, the association between the feature clusters and the class labels cannot reliably be established. Based on these results, we conclude that the proposed algorithm is effective both at improving the classification accuracy and at reducing its variance.

Figure 4 shows examples of interest points that receive high weights using our algorithm. The $100$ keypoints with the highest weights are marked by yellow faces and red edges while other keypoints in the image are marked by black faces and blue edges. It is reassuring to see that, for each category of interest, the top keypoints are primarily located on the target objects even though the raw keypoints are widely scattered over both foreground and backgrond regions of the image. Note that the masked images were not used during training, so this is a clear indication that our algorithm has correctly learned to associate the right features to the object category despite the weakly-labeled data.

## 4. Conclusion

This paper proposes a novel framework for object category recognition that unifies visual codebook generation with classifier training. Two key features distinguish this work from existing approaches for object category recognition. First, unlike the clustering approaches that associate each image feature with a single visual word, we encode each image feature by a vector of visual bits. Second, in contrast to the standard practice that separates the processes for visual codebook generation and classifier training, the proposed approach unifies these two processes in a single optimization framework under one objective function. An iterative algorithm is presented to efficiently identify the optimal visual bits and their associated weights. Experiments on the PASCAL 2006 dataset demonstrate that the proposed unified approach is a significant advance over state-of-the-art approaches for object category classification.

## References


[1] G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[2] M. Everingham, A. Zisserman, C. Williams, and L. Van Gool. The 2006 PASCAL visual object classes challenge.

[3] J. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor. Improving "bag-of-keypoints" image categorisation. Technical report, University of Southampton, 2005.

[4] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition*, 2005.

[5] F. Fleuret and G. Blanchard. Pattern recognition from one example by chopping. In *NIPS*, 2005.

[6] J. Hartigan and M. Wang. A k-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[7] E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh. On the significance of real-world conditions for material classification. In *European Conference on Computer Vision*, 2004.

[8] T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pages 137–142. Springer-Verlag, 1998.

[9] D. Larlus and F. Jurie. Latent mixture vocabularies for object categorization. In *British Machine Vision Conference*, 2006.

[10] D. Lowe. Distinctive image features form scale-invariant keypoints. In *International Journal of Computer Vision*, 2004.

[11] M. Marszalek and C. Schmid. Spatial weighting for bag-of-features. In *Computer Vision and Pattern Recognition*, 2006.

[12] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. In *International Journal of Computer Vision*, 2004.

[13] F. Moosmann, B. Triggs, and F. Jurie. Randomized clustering forests for building fast and discriminative visual vocabularies. In *NIPS*, 2007.

[14] F. Perronnin, C. Dance, G. Csurka, and M. Bressian. Adopted vocabularies for generic visual categorization. In *European Conference on Computer Vision*, 2006.

[15] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., 1986.

[16] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999.

[17] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision*, 2003.

[18] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *International Conference on Computer Vision*, 2005.

[19] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classifcation of texture and object categories: An in-depth study. Technical Report RR-5737, INRIA Rhône-Alpes, 2005.

(a) sheep (b) sheep (c) bus (d) cat

(e) bicycle (f) bicycle (g) car (h) car

(i) cow (j) cow (k) cow (l) dog

(m) dog (n) dog (o) horse (p) horse

(q) horse (r) motorbike (s) motorbike (t) person

Figure 4. Example images from the PASCAL VOC 2006 database. The presence of highly-weighted interest points (marked in yellow face/red edge) on objects that correspond to the category of interest demonstrate that our algorithm correctly learns the association between the right features and the category in spite of weakly-labeled data. Remaining interest points found by the detector are shown in black face/blue edge.