Kernel Sharing With Joint Boosting For Multi-Class Concept Detection

Wei Jiang
Columbia University
New York, NY

Shih-Fu Chang
Columbia University
New York, NY

Alexander C. Loui
Eastman Kodak Company
Rochester, NY

Abstract

Object/scene detection by discriminative kernel-based classification has gained great interest due to its promising performance and flexibility. In this paper, unlike traditional approaches that independently build binary classifiers to detect individual concepts, we proposed a new framework for multi-class concept detection based on kernel sharing and joint learning. By sharing "good" kernels among concepts, accuracy of individual weak detectors can be greatly improved; by joint learning of common detectors among classes, the required kernels and the computational complexity for detecting each individual concept can be reduced. We demonstrated our approach by developing an extended JointBoost framework, which was used to choose the optimal kernel and subset of sharing classes in an iterative boosting process. In addition, we constructed multi-resolution visual vocabularies by hierarchical clustering and computed kernels based on spatial matching. We tested our method in detecting 12 concepts (objects, scenes, etc) over 80+ hours of broadcast news videos from the challenging TRECVID 2005 corpus. Significant performance gains were achieved - 10% in mean average precision (MAP) and up to 34% average precision (AP) for some concepts like maps, building, and boat-ship. Extensive analysis of the results also revealed interesting and important underlying relations among concepts.

1. Introduction

This paper investigates the problem of automatic detection and recognition of the semantic categories, terms *concepts*, in images, such as scene (*e.g.* waterscape-waterfront), location (*e.g.* court) and objects (*e.g.* car). Among various approaches for concept detection, kernel-based discriminative methods, *e.g.* classifiers based on the Support Vector Machine (*SVM*), have been demonstrated effective for detecting generic concepts in previous works [1, 5, 6, 7, 9, 11]. A kernel computes the similarity between images and can be used to find the optimal decision boundary for classification. A large variety of kernels can be constructed based on different local part descriptors extracted from regions or patches, *e.g.* color descriptor [20], texture descriptor [13], SIFT descriptor [5], or the *bag-of-features* (*BOF*) representation [3, 8, 19, 21]. In BOF, images are represented by a *visual vocabulary* constructed by clustering the original local descriptors into a set of *visual tokens*. Based on these local representations, recently, the *pyramid matching* algorithms [5, 6, 9] have also been proposed to fuse information from multiple resolutions in the spatial domain or feature space. In these approaches, vocabularies constructed from the training data of each concept are used to compute kernels for different concepts, and detectors for each individual concept are independently learned in an one-vs-all manner.

In this paper, instead of building independent binary one-vs-all detectors, we explore this multi-class concept detection problem based on joint learning and vocabulary/kernel sharing. Different kernels designed for individual concepts are shared by all the concept detectors, and common detectors for multiple concepts are jointly learned. The motivation is intuitive: concepts are generally related to each other, *e.g.* the "boat-ship" concept is related to the "waterscape-waterfront" concept; by sharing vocabularies/kernels from interrelated concepts, each individual concept can be enhanced by incorporating the descriptive power from others. For example, as shown in Fig.1, the "boat-ship" concept is very difficult to recognize, due to the diverse object appearances and the resulting lacking of strong visual tokens. Kernels computed using such low-quality vocabularies from "boat-ship" itself usually do not have adequate discriminative power and thus yield unsatisfactory recognition performance. On the other hand, images from "waterscape-waterfront" have good consistency in visual appearances, and high quality vocabularies can be learned. Since "boat-ship" is strongly related to "waterscape-waterfront", kernels calculated using the good vocabularies from "waterscape-waterfront" provide useful complementary information to help detect "boat-ship". Moreover, the sharing of common classifiers among different concepts can be used to construct multi-class concept detectors. As will be shown in Section 3 and 4, required kernels for learning each individual concept as well as the detection complexity will be largely reduced.

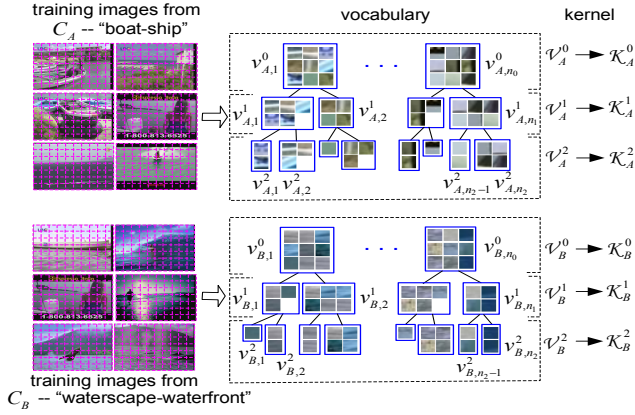We demonstrate our approach by developing a new

1

Figure 1. *Examples of vocabulary/kernel construction. For each concept,* e.g. C_A, *SIFT features are extracted from uniform grids over each training image from this concept. Hierarchical clustering is applied to construct vocabularies with different resolutions: $\mathcal{V}_A^0, \ldots, \mathcal{V}_A^L$ (L is vocabulary resolution and $L = 2$ in this example), where $\mathcal{V}_A^l$ consists of n_l visual tokens $\{v_{A,1}^l, \ldots, v_{A,n_l}^l\}$. Then vocabulary-spatial pyramid match kernels $\mathcal{K}_A^0, \ldots, \mathcal{K}_A^L$ are computed (see Sec.2.2 for details), one kernel for each vocabulary.*

multi-class concept detection framework. To jointly learn the kernel-based detectors and to share kernels from different concepts, we propose a joint boosting algorithm to automatically select the optimal kernel and the subset of sharing concepts in an iterative boosting process. We modify the traditional JointBoost algorithm [17] to incorporate the general Real AdaBoost method [4], so that kernel-based classifiers can be applied for joint learning. Our method provides a flexible framework that any type of kernel can be conveniently incorporated for multi-class concept detection.

In addition, we propose a new kernel construction algorithm, *vocabulary-spatial pyramid matching* (*VSPM*), which constructs multi-resolution visual vocabularies by hierarchical clustering and computes kernels based on spatial matching. This is essentially an extension of the previous spatial-pyramid matching [9] and vocabulary-guided pyramid matching [6] methods. Although impressive performance has been obtained by these two previous approaches, there are still some issues: as both indicated in [9] and our experiments, it is usually difficult for spatial pyramid matching to automatically choose an appropriate vocabulary resolution for each specific concept; vocabulary-guided pyramid matching does not consider the spatial layout of local features, which are proven to be effective for generic concept detection [9, 12]. In order to combine the power of multi-resolution spatial matching from the former and multi-layer vocabulary fusion from the latter, we explore such combined method to generate multiple vocabularies/kernels for each concept, which are further aggregated and shared by all concepts. Richer kernels are provided and the above problems are alleviated .

We evaluate our method in detecting 12 concepts (ob-

jects, scenes, etc) over 80+ hours (60000+ key frames) of broadcast news videos from the TRECVID 2005 corpus [18]. This is one of the largest and most challenging benchmark data sets existing for concept detection. Significant performance gains are achieved - 10% in mean average precision (MAP) and up to 34% average precision (AP) for some concepts like "maps", "building", and "boat-ship". Extensive analysis of the results also reveals interesting and important underlying relations among object classes. In addition, we avoid the linear increase of complexity at run time as the number of concepts grows, due to the sharing of detectors among concepts.

The remaining part of this paper is as follows. Section 2 introduces the vocabulary and kernel construction method. Section 3 describes our joint boosting algorithm. Section 4 gives experimental results. Section 5 is our conclusion.

2. Construction of Vocabulary/Kernel Pool

We start by defining notations. Assume there are M semantic concepts, $C_1, \ldots, C_M$. Let $\mathcal{D} = \{I_p, \mathbf{y}_{I_p}\}$ denote the data set, where I_p is an image and $\mathbf{y}_{I_p} = [y_{I_p}^1, \ldots, y_{I_p}^M]$ is ground truth concept labels of image I_p, with $y_{I_p}^j = 1$ or -1 representing the presence or absence of concept C_j in image I_p. Let $\mathcal{F}$ denote a certain feature space for local descriptors (*e.g.* 128 dimensional SIFT feature space used in [9]). In the following subsection, we will introduce the procedure for constructing multi-resolution vocabularies, followed by our vocabulary-spatial pyramid matching algorithm.

2.1. Vocabulary construction

The bag-of-features representation provides an effective framework for partitioning the original feature space $\mathcal{F}$ to establish a vocabulary of prototypical image features [3, 8, 19, 21]. These prototypes are usually called visual tokens. Recently, in [6] Grauman *et al.* construct multi-resolution vocabularies by hierarchical clustering. Images are represented with multi-level histograms defined over tokens of vocabularies, and kernel matching between images is computed based on histogram intersection. As discussed in [6], vocabularies established from hierarchical clustering take advantage of the underlying structure of feature space.

In this paper, we adopt the similar hierarchical vocabulary construction procedure and apply the pyramid matching process in the spatial domain. We incorporate the spatial layout of local descriptors which have been shown to be effective for generic concept detection [9, 12].

Specifically as shown in Fig.1, for each concept C_j, the local descriptors from the positive training samples are aggregated together, and hierarchical clustering is applied to generate $L+1$ coarse to fine vocabularies which partition the feature space $\mathcal{F}$ into a pyramid of non-uniformly shaped regions. Let $\mathcal{V}_j^l = \{v_{j,1}^l, \ldots, v_{j,n_l}^l\}$ denote the vocabulary at level l, $0 \leq l \leq L$, where each $v_{j,g}^l$ represents the gth visual

token in vocabulary $\mathcal{V}_j^l$. Based on these multi-level vocabularies, in the next subsection, a vocabulary-spatial pyramid matching algorithm will be developed to construct multi-resolution kernels for each individual concept.

2.2. Kernel construction

For each concept C_j, with each vocabulary $\mathcal{V}_j^l$, we construct a vocabulary-spatial pyramid match (VSPM) kernel based on spatial matching. We adopt the spatial pyramid matching algorithm proposed by Lazebnik *et al.* in [9], which computes correspondence between images by repeatedly subdividing images into a spatial pyramid and computing histogram intersections at multiple spatial resolutions. This method achieves robust performance for generic scene detection by incorporating spatial layout information of local descriptors [9]. In this paper, we extend the original spatial pyramid matching to incorporate multi-resolution vocabularies. As both evidenced by experiments in [9] and our experiments in Section 4 (Table 1 and Fig.5), different concepts favor different resolutions of vocabularies, and it is generally difficult to determine one optimal resolution for detecting all concepts in spatial pyramid matching. In our proposed method, we construct a pool of vocabularies over multiple resolutions and develop automatic mechanism for selecting the optimal ones.

Specifically, given a concept C_j and a vocabulary $\mathcal{V}_j^l$, the spatial pyramid matching method [9] hierarchically quantizes the spatial coordinates of images at $S+1$ different levels, with 4^s discrete blocks at each level s, $0 \leq s \leq S$. Then for each image I_p, a n_l-dimensional histogram $H_{j,k}^{s,l}(I_p) = [h_{j,k,1}^{s,l}(I_p), \ldots, h_{j,k,n_l}^{s,l}(I_p)]$ is calculated from block k at spatial level s, using a certain vocabulary $\mathcal{V}_j^l$, where $h_{j,k,g}^{s,l}(I_p)$ is the element bin over visual token $v_{j,g}^l$. The number of matches at spatial level s between two images I_p and I_q is computed by histogram intersection:

$$\mathcal{I}_j^{s,l}(I_p, I_q) = \sum_{k=1}^{4^s} \sum_{g=1}^{n_l} \min\left\{ h_{j,k,g}^{s,l}(I_p), h_{j,k,g}^{s,l}(I_q) \right\}$$

Since the matches found at level s include all the matches at level $s+1$, when we go down from fine (level $s+1$) to coarse (level s), the new matches found is given by $\mathcal{I}^s - \mathcal{I}^{s+1}$, $0 \leq s \leq S-1$. Assume that each level s is associated with a weight $\frac{1}{2^{S-s}}$, which favors matches found in finer cells because they involve increasingly similar features. The final VSPM kernel defined by vocabulary $\mathcal{V}_j^l$ is given by:

$$\mathcal{K}_j^l(I_p, I_q) = \mathcal{I}_j^{S,l}(I_p, I_q) + \sum_{s=0}^{S-1} \frac{\mathcal{I}_j^{s,l}(I_p, I_q) - \mathcal{I}_j^{s+1,l}(I_p, I_q)}{2^{S-s}}$$
$$= \frac{1}{2^S} \mathcal{I}_j^{0,l}(I_p, I_q) + \sum_{s=1}^{S} \frac{1}{2^{S-s+1}} \mathcal{I}_j^{s,l}(I_p, I_q) \quad (1)$$

For each concept C_j, we have $L+1$ multi-resolution VSPM kernels $\mathcal{K}_j^0, \ldots, \mathcal{K}_j^L$ defined by $L+1$ vocabularies. Instead of training one-vs-all detectors independently, we propose to share various kernels from different concepts

across multiple detectors. Specifically, all VSPM kernels from different concepts are aggregated together to generate a kernel pool $\mathbf{K}$. In the next section, a joint boosting algorithm is developed which provides a systematic data-driven framework to select optimal kernels from $\mathbf{K}$ to be shared and the optimal sets of concepts that share them.

Note that it is also possible to use these VSPM kernels in the one-vs-all manner similar to previous concept detection approaches [6, 9] (as shown in Fig.2 (a)). One method is the weighted combination of the $L+1$ kernels $\mathcal{K}_j^0, \ldots, \mathcal{K}_j^L$ from each concept C_j into one ensemble kernel:

$$\mathcal{K}_j(I_p, I_q) = \sum_{l=0}^{L} \omega_l \mathcal{K}_j^l(I_p, I_q) \quad (2)$$

Weight ω_l can be heuristically defined as in [6]. Then this ensemble kernel can be used to train one-vs-all classifiers to detect this concept independently from other concepts. Another way is to directly apply traditional boosting algorithms (*e.g.* Real AdaBoost [4]) to train one-vs-all detectors for each individual concept C_j by selecting appropriate kernels from $\mathcal{K}_0^l, \ldots, \mathcal{K}_j^l$ through boosting iterations. In our experiments in Section 4, we will use these two approaches as baselines to compare with our joint boosting algorithm.

3. Joint Learning with Joint Boosting

The idea of joint boosting is built upon the sharing of features and classifiers throughout multiple categories. Most of the existing concept detection approaches [3, 5, 6, 9, 13] recognize each individual concept C_j independently from other concepts. As shown in Fig.2 (a), these one-vs-all detectors train a binary classifier to detect each concept C_j, only based on the VSPM kernels $\mathcal{K}_j^0, \ldots, \mathcal{K}_j^L$ derived from this concept. While in joint boosting (shown in Fig.2 (b)), multi-class concepts are detected simultaneously by sharing a kernel pool consisting of all VSPM kernels from various concepts and sharing the classifiers during boosting iterations. The motivation is intuitive. For example, as shown in Fig.1, for the "boat-ship" concept, due to the diverse visual appearances of objects, the extracted vocabularies cannot provide adequate discriminative power. In such case, the high-quality VSPM kernels from related concepts (such as "waterscape-waterfront") may contain useful additional information to help detection. By sharing good kernels among detectors, individual concepts can be enhanced by incorporating descriptive power from others. Moreover, by sharing common detectors among concepts, required kernels and training samples for detecting individual concepts will be reduced [17]. In the next subsections, we will introduce the boosting framework followed by our joint boosting method.

3.1. Learning with boosting

The multi-class concept detection problem can be formulated as minimizing a multi-class objective function:

$$J = \sum_{j=1}^{M} \sum_{I \in \mathcal{D}} w_I^j e^{-y_I^j Q_I^j} \quad (3)$$
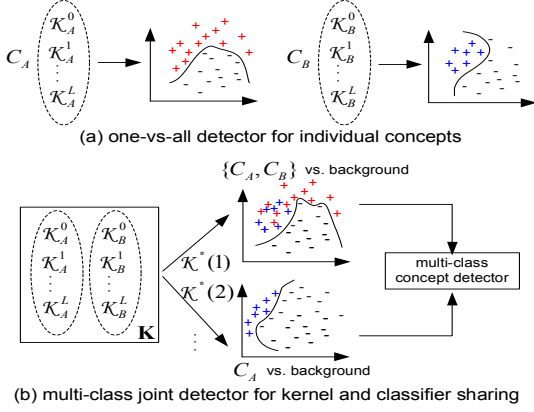
3

(a) one-vs-all detector for individual concepts

(b) multi-class joint detector for kernel and classifier sharing

Figure 2. *Illustration of learning procedures. (a) one-vs-all detectors recognize each individual concept, e.g. C_A, only based on kernels $\mathcal{K}_A^0, \ldots, \mathcal{K}_A^L$ derived from this concept, independently from other concepts. (b) in joint boosting, a kernel pool $\mathbf{K}$ (including kernels from all concepts) is shared by different detectors. First, using kernel $\mathcal{K}^*(1)$ (an optimal kernel chosen from $\mathbf{K}$), a binary classifier is used to separate C_A, C_B from the background, i.e. C_A and C_B share this classifier and kernel. Then using $\mathcal{K}^*(2)$, a binary classifier further picks out C_A. By sharing kernels individual concepts are enhanced by incorporating the descriptive power from others; by sharing classifiers fewer kernels and training samples are needed to detect each concept (see Sec.3.2 for details).*

where y_I^j is the ground-truth label of concept C_j for image I, and w_I^j is the sample weight for image I in concept C_j. Q_I^j is a discriminative function (the classification result) we want to learn. In one-vs-all independent detection methods, the multi-class concept detection problem is converted into a set of binary classification problems. That is, J is approximately optimized by minimizing $J_j = \sum_{I \in \mathcal{D}} w_I^j e^{-y_I^j Q_I^j}$ for each concept C_j independently. In contrast, in the joint learning scenario, we directly optimize J jointly by training a multi-class classifier. Since concepts are usually related to each other, joint learning is intrinsically reasonable and can better optimize J than independent training approaches.

The well-known boosting framework effectively combines many weak classifiers to form a powerful ensemble classifier. In the training stage, training samples are reweighted according to the training error from previous weak learners, and classifiers trained later are forced to focus on harder samples with higher weights. The boosting procedure is formulated by assuming an additive model for Q_I^j:

$$Q_I^j(T) = \sum_t q_I^j(t) \qquad (4)$$

where each $q_I^j(t)$ is the hypothesis generated by the tth weak learner in iteration t. Several variants of boosting algorithm have been developed, such as AdaBoost [15], Real AdaBoost [4], GentleBoost [4], LogitBoost [4], and boosting feature selection [16].

The original boosting algorithm is designed for binary classification, and several multi-class learning algorithms, *e.g.* AdaBoost.MO, AdaBoost.MH and AdaBoost.MR [15],

have been proposed as extensions of original boosting. In [17] Torralba *et al.* propose a JointBoost algorithm based on GentleBoost. This method tries to find the common feature axes and the corresponding weak learners that can be shared across different classes. In [14] the JointBoost algorithm is extended into a Joint-AdaBoost approach which enables incremental additions of new classes.

3.2. Joint boosting and kernel sharing

Motivated by the work of [14, 17], in this paper we develop our joint boosting algorithm based on Real AdaBoost due to its flexibility for incorporating the powerful kernel-based classifiers for concept detection and its compatibility for our proposed strategy to share kernels in the iterative boosting steps. Specifically, during each iteration t, our joint boosting method selects the optimal subset of concepts $\mathcal{S}^*(t)$ and the corresponding optimal kernel $\mathcal{K}^*(t)$. Positive training samples from any concept in $\mathcal{S}^*(t)$ are treated as positive, and a binary classifier is learned based on $\mathcal{K}^*(t)$ to separate these positive data from the background negative data (samples not belonging to any concept). The optimal $\mathcal{S}^*(t)$ and $\mathcal{K}^*(t)$ are chosen by searching from every possible subset $\mathcal{S}(t)$ of concepts using every possible kernel $\mathcal{K}(t)$ in kernel pool $\mathbf{K}$ so that the corresponding classifier has the smallest error on the weighted training set for all concepts.

In the Real AdaBoost formulation [4], during each iteration t, we want to minimize the objective function J in Eqn.(3) with respect to $q_I^j(t)$, $\mathcal{S}(t)$ and $\mathcal{K}(t)$ as follows:

$$J(q_I^j(t), \mathcal{S}(t), \mathcal{K}(t)) = \sum_{j=1}^M \sum_{I \in \mathcal{D}} w_I^j(t) e^{-y_I^j \left(Q_I^j(t-1) + q_I^j(t) \right)} \quad (5)$$

where $q_I^j(t)$ has the following form:

$$q_I^j(t) = \begin{cases} \frac{1}{2} \log \frac{p^t(y_I^j = 1 | I)}{p^t(y_I^j = -1 | I)} , & C_j \in \mathcal{S}(t) \\ k_c^j(t) & , others \end{cases} \quad (6)$$

$p^t(y_I^j = 1 | I)$ is the posterior probability of positive detection of concept C_j generated by the binary weak classifier that separates the positive training samples from the subset $\mathcal{S}(t)$ and the background samples. All concepts $C_j \in \mathcal{S}(t)$ share this binary classifier and have the same $p^t(y_I^j = 1 | I)$ value. $k_c^j(t)$ is a constant for concept C_j which is not selected to share the weak learner in iteration t. A natural choice of $k_c^j(t)$ is to use the prior, namely $p^t(y_I^j = 1 | I) = \frac{\sum_{I \in \mathcal{D}} w_I^j(t) \delta(y_I^j = 1)}{\sum_{I \in \mathcal{D}} w_I^j(t)}$, where $\delta(\cdot)$ is the indicator function. Then $k_c^j(t)$ is given by:

$$k_c^j(t) = \frac{1}{2} \log \frac{\sum_{I \in \mathcal{D}} w_I^j(t) \delta(y_I^j = 1)}{\sum_{I \in \mathcal{D}} w_I^j(t) \delta(y_I^j = -1)} \quad (7)$$

$k_c^j(t)$ depends on the empirical counts of the positive and negative training samples of C_j only, rather than the kernel-based classifier shared with other concepts. $k_c^j(t)$ prevents sharing of kernels due to asymmetry between the number of positive and negative samples for concept C_j.

Any type of classifiers can be used to generate the poste-

rior probability $p^t(y_I^j = 1|I)$ for concepts in $\mathcal{S}(t)$. In the paper, we use the SVM classifier for concept detection. To conduct sample re-weighting for SVM, we use the random sampling method. That is, during each iteration t, for each concept C_j, we form a new training set $\tilde{\mathcal{D}}^j(t)$ generated by random sampling training samples from the original training set $\mathcal{D}$ according to sample weights $w_I^j(t)$.

The detailed joint boosting algorithm is summarized in Fig.3. Originally, we need to search all possible $N(2^M - 1)$ combinations of concepts and kernels, where M is the number of concepts, and N is the number of kernels in $\mathbf{K}$. To reduce the computational complexity, during each iteration t, we adopt the forward selection procedure from [17]: in the first step, we try every concept using every possible kernel in the kernel pool, and then select the best pair of kernel $\mathcal{K}^*(t)$ and concept $C_1^*(t)$ which has the smallest cost based on Eqn.(5). Then in the next step, we set $\mathcal{S}_1(t) = \{C_1^*(t)\}$, and for $r = 2, \ldots, M$, a concept $C_r^*(t)$ is added to $\mathcal{S}_{r-1}(t)$ to form $\mathcal{S}_r(t) = \mathcal{S}_{r-1}(t) \cup C_r^*(t)$ so that the joint detector of subset $\mathcal{S}_r(t)$ has the smallest cost among all possible $C_r(t)$. Finally $\mathcal{S}_M(t)$ will include all concepts. Then from the M candidate subsets $\mathcal{S}_1(t), \ldots, \mathcal{S}_M(t)$, we select $\mathcal{S}^*(t)$ with the smallest cost. Note that since our weak learners generate hypotheses according to Eqn.(6) and samples are re-weighted in the same way as the original Real AdaBoost: $w_I^j(t) = w_I^j(t-1)e^{-y_I^j q_I^j(t)}$, our joint boosting algorithm inherits the convergence property of the Real AdaBoost algorithm [4]. In other words, the multi-class objective function is guaranteed to improve in every iterative step.

The major difference between our joint boosting algorithm and the original JointBoost [17] is that we use Real AdaBoost for sharing kernels, while the original JointBoost uses GentleBoost [4] for sharing feature axes. GentleBoost is a "gentle" version of Real AdaBoost, where the "gentle" Newton stepping is used to improve the objective function J in each step. In contrast, for Real AdaBoost exact optimization of the objective function is performed in each step. GentleBoost uses logistic regression as the weak learner, which has less flexibility (in terms of weak learner) than Real AdaBoost where any type of classifier can be adopted to generate the hypotheses. Therefore, to incorporate the powerful kernel-based discriminative classifiers for detecting generic concepts, we build our joint boosting algorithm based on the general Real AdaBoost method.

4. Experiments

In this section, we will introduce the details about feature extraction and the experimental setting, followed by the experimental results.

4.1. Feature extraction

Similar to [9], SIFT descriptors are extracted from each 16x16 block over a grid with spacing of 8 pixels. For each concept, the SIFT features from the positive training samp-

---

algorithm: Joint Boosting

Input: The data set $\mathcal{D} = \{I_p, \mathbf{y}_{I_p}\}$, and the kernel pool: $\mathbf{K} = \{\mathcal{K}_1^0, \ldots, \mathcal{K}_1^L, \ldots, \mathcal{K}_M^0, \ldots, \mathcal{K}_M^L\}$ (M concepts, and L multi-resolution kernels for each individual concept).

- Initialization: $\forall j$, set $w_I^j(0) = 1$, $Q_I^j(0) = 0$.
- For iteration $t = 1, \ldots, T$
    - get training set $\tilde{\mathcal{D}}^j(t)$ for each concept C_j by random sampling $\mathcal{D}$ according to weights $w_I^j(t)$.
    - select the optimal kernel $\mathcal{K}^*(t)$ and subset of concepts $\mathcal{S}^*(t)$ in a forward selection procedure:
        * find the optimal pair of kernel $\mathcal{K}^*(t)$ and concept $C_1^*(t)$ by searching from all possible pairs.
        * set $\mathcal{S}_1(t) = \{C_1^*(t)\}$
        * Repeat for $r = 2, \ldots, M$
          Get $\mathcal{S}_r(t) = \mathcal{S}_{r-1}(t) \cup C_r^*(t)$ by finding the optimal $C_r^*(t)$ from all possible $C_r(t) \notin \mathcal{S}_{r-1}(t)$.
        * Select the optimal $\mathcal{S}^*(t)$ from all candidate subsets $\mathcal{S}_1(t), \ldots, \mathcal{S}_M(t)$.
    - Train a SVM classifier based on $\mathcal{S}^*(t)$ and get $q_I^j(t)$.
    - For each concept C_j:
        * re-weight samples: $w_I^j(t) = w_I^j(t-1)e^{\{-y_I^j q_I^j(t)\}}$
        * Update decision: $Q_I^j(t) = Q_I^j(t-1) + q_I^j(t)$.

Figure 3. *The joint boosting algorithm based on Real AdaBoost.*

les are aggregated and k-means clustering is applied to get a visual vocabulary $\mathcal{V}^0$ at level $l = 0$, consisting of 100 visual tokens ($n_0 = 100$). Then hierarchical clustering is applied to generate higher level vocabularies as follows: for each token v_i^l, $0 \leq l \leq L-1$, we perform k-means clustering to divide this token to two sub visual tokens v_{2i}^{l+1} and v_{2i+1}^{l+1}. That is, the sizes of vocabularies have $n_l = 2n_{l-1}$. In our implementation we have three levels in total, *i.e.* $L = 2$. So the maximum size of vocabulary is $n_2 = 400$.

4.2. Data sets

Our experiments are carried out over a challenging dataset: 80+ hours broadcast news videos from the TRECVID 2005 corpus [18]. This is one of the largest and most challenging data sets existing for concept detection. It contains 137 international news videos, which are divided to sub-shots and 60000+ key frames are extracted from each sub-shot. The key frames are labeled with 39 concepts from LSCOM-Lite ontology [10], *e.g.* "car", "outdoor", etc. In our experiment 12 diverse concepts are manually selected, including objects (*e.g.* "flag-us" and "boat-ship"), locations (*e.g.* "urban"), and scenes (*e.g.* "waterscape-waterfront" and "vegetation"). One semantic concept, "government-leader", is also included as a challenging case. Although recognition of such highly specific semantic concept seems very difficult, it is made more feasible by the consistent visual scenes (like press conference rooms) often seen in images including government leaders. The selected 12 concepts have moderate performances (not too high or too low) [1, 18], which makes them suitable for comparing various concept detection algorithms. Example images for the 12
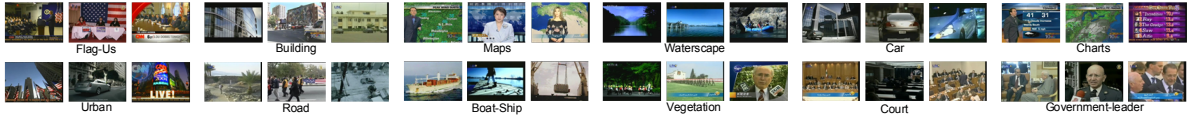
Figure 4. *Example images for the TRECVID data set.*

concepts are shown in Fig.4. In TRECVID data, images can belong to different concepts, *i.e.*, concepts are not exclusive. The training set consists of at most 2000 positive images for each concept and 1000 "background" images (negative to all 12 concepts). All images are randomly sampled from 90 training videos. 16 videos from the remaining set are randomly selected as the test data. Finally there are 11952 images for training and 6507 images for evaluation.

The AP (average precision) and MAP (mean average precision) are used as performance measurements. AP is an official TRECVID performance metric, which is related to multi-point average precision value of a precision-recall curve [18]. To calculate AP for concept C_j, we first rank the test data (from all concepts) according to the classification posteriors of concept C_j. Then from top to bottom, the precision after each positive sample is calculated. These precisions are averaged over the total number of positive samples for C_j. AP favors highly ranked positive samples and combines precision and recall values in a balanced way. MAP is calculated by averaging APs across all concepts.

4.3. Comparison between different kernels

First, we evaluate our VSPM kernel using a baseline approach (called VSPM baseline in the following experiments), where for each concept C_j, an ensemble kernel is generated based on Eqn.(2) and an one-vs-all SVM classifier is trained using the ensemble kernel to detect this concept (as discussed in Section 2.2). We compare this baseline with a state-of-the-art approach [9]: SVM classifiers based on the spatial pyramid match kernel using a single vocabulary. Table 1 shows the MAP comparison, and Fig.5 shows the per-concept AP performance. From the results, the VSPM baseline slightly outperforms single-vocabulary kernels in terms of MAP, and performs best for 5 concepts in terms of AP, while kernels with single-vocabulary wins over the remaining 7 concepts. Also, no single vocabulary setting outperforms others over all concepts. The single-vocabulary kernel at different levels, $l = 0, 1, 2$, achieves the best performance for 1, 2, and 4 concepts respectively. These results show that different concepts favor different resolutions of kernels, and the VSPM baseline which averages multi-resolution kernels with heuristic predefined weights does not result in unambiguous performance gains.

4.4. Vocabulary sharing with joint boosting

In this experiment, we compare our joint boosting with two baseline approaches: the VSPM baseline presented in the above subsection, and the regular boosting approach where Real AdaBoost is directly applied to detect individual concepts independently (as discussed in Section 2.2). Fig.6 shows the AP performance over each of the 12 concepts. The regular boosting uses 120 weak classifiers (10 iterations for each concept) and joint boosting uses 100 weak classifiers (100 iterations for joint boosting). From the figure we can see that regular boosting actually hurts the performance over most of the concepts. This phenomenon indicates that severe overfitting occurs by using regular boosting due to the small positive training set for many concepts, e.g. 254 positive samples out of 41847 training samples for "boat-ship". As demonstrated in [2], with such few and noisy positive training samples, regular boosting is usually very sensitive and overfits greatly. Compared with the baselines, our joint boosting method consistently achieves superior performance for most of the concepts with the limited training data. This phenomenon is consistent with previous literatures. As shown in [17, 14], when the number of training samples are reduced, joint boosting can still get good performance, since through sharing weak learners and kernels among different concepts the required training kernels and samples are reduced for detecting each individual concept [17]. On average, the overall MAP improvement of joint boosting is 24% and 10%, respectively, compared with regular boosting and VSPM baseline. The performance improvements over many concepts are significant, *e.g.* compared with VSPM baseline, joint boosting achieves the following AP gain: "building" 30%, "maps' 34%, "urban" 10%, "boat-ship" 17%, "vegetation" 19%, and "government-leader" 11%.

In addition, Fig.7 shows the evolution of the MAP performance during boosting iterations. When more weak classifiers are added in, joint boosting can achieve better per-

Table 1. *MAP: VSPM baseline vs. single-vocabulary kernels*

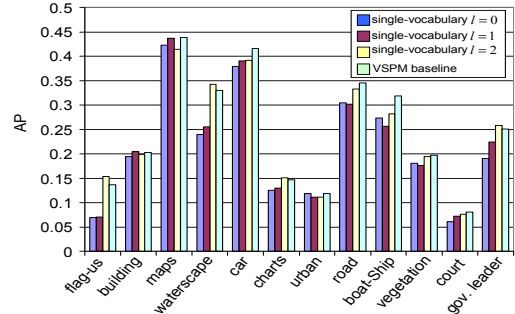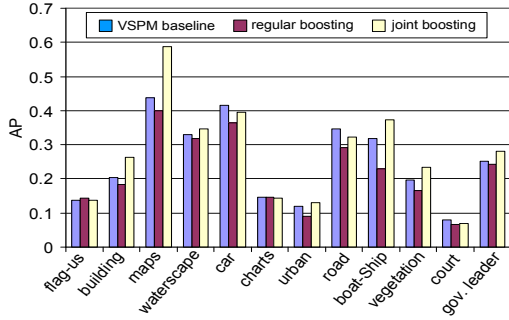| single-vocabulary kernel			VSPM baseline
$l=0$	$l=1$	$l=2$	
0.213	0.219	0.242	0.248

Figure 5. *Per-concept AP: VSPM baseline vs. single-vocabulary kernels. l indicates the vocabulary resolution level.*

Figure 6. *Per-concept AP of joint boosting and two baseline approaches: VSPM baseline and regular boosting.*

formance, confirming the convergence property of the joint boost algorithm. Note in the current experiment, we stop at 100 weak learners, and the performance curve is still rising. This is in contrast to the regular boosting algorithm, where the performance keeps decreasing throughout the iterations.
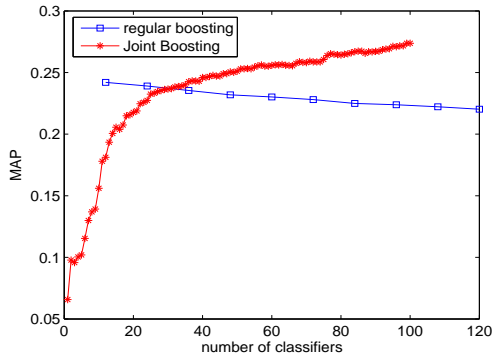

Figure 7. *Performance evolution of joint boosting and regular boosting with different numbers of weak classifiers.*

4.5. Exploring concept relations

In this subsection, we explore the relationships between the 12 concepts in TRECVID data through analysis of the experimental results from joint boosting. There are 36 kernels in the kernel pool, 3 kernels for each concept. Only 19 kernels are selected to be used by the joint boosting algorithm. Fig.8 shows the frequency of the usage for each kernel. From the figure, kernels of some concepts, *e.g.* "maps" and "boat-ship", are seldom used. These concepts are either low-supported rare concepts, or the supporting images have large diversity in visual appearances (*e.g.* boat-ship as shown in Fig.1). Thus the quality of visual vocabularies from these concepts are relatively poor. This result is consistent with above experiments that these concepts can benefit a lot using kernels from other concepts, and joint boosting has significant performance improvement over "maps" and "boat-ship" compared with independent detectors.

Fig.9 gives the details about the usage of kernels by each concept. The figure clearly shows the sharing of vocabularies across different concepts. For example, "boat-ship" uses the kernels from "waterscape-waterfront" frequently in addition to its own kernels, and "maps" uses the kernels from
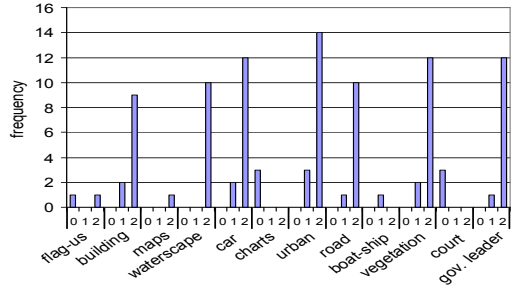

Figure 8. *Usage of individual kernels by 100 weak classifiers. The horizontal axis shows kernels from different concepts, where numbers 0,1,2 represent multiple resolutions, i.e. l. Kernels from "maps" and "boat-ship" are seldom used. Interestingly, these two concepts are enhanced significantly in joint boosting by leveraging kernels from other related concepts.*

many other concepts such as "charts", "flag-us", etc. This is because of the high correlation between "boat-ship" and "waterscape-waterfront", and between "maps" and "charts" etc. Since images from "waterscape-waterfront" or "charts" etc have high consistency, the high-quality visual tokens extracted for 'waterscape-waterfront' or "charts" etc provide important semantic cues for detecting "boat-ship" or "maps" respectively. In addition, Fig.10 shows the AP evolution of "boat-ship" and "maps" during 100 iterations, where the kernels corresponding to big performance changes are also listed. Note the same kernel may be used multiple times throughout the iterative process. Fig.10 further confirms that the performance of "boat-ship" and "maps" are helped significantly by sharing kernels from "waterscape-waterfront" and "charts" etc respectively.

4.6. Computational complexity

The test stage of joint boosting is very fast, while its training stage needs more computational time than regular boosting or VSPM baseline without boosting. Since training can be done offline, joint boosting works efficiently in practice. During the test stage, joint boosting only needs T SVM classification, one for each iteration, to classify M concepts simultaneously. While regular boosting needs $M \cdot T$ SVM classification. In the training stage, for each iteration, joint boosting needs to train $M \cdot N + \sum_{i=1}^{M-1} i = \frac{(M-1)M}{2} + MN$ classifiers for finding the optimal kernel and subset of concepts, where N is the total size of kernel pool $\mathbf{K}$. This is larger than the number needed for regular boosting – $M \cdot L$ classifiers during each iteration.

5. Conclusion

We proposed a novel multi-class concept detection framework based on kernel sharing and joint learning. By sharing "good" kernels among concepts, accuracy of individual detectors can be improved; by joint learning of common detectors across different classes, required kernels and computational complexity for detecting individual concepts can be reduced. We demonstrated our approach by
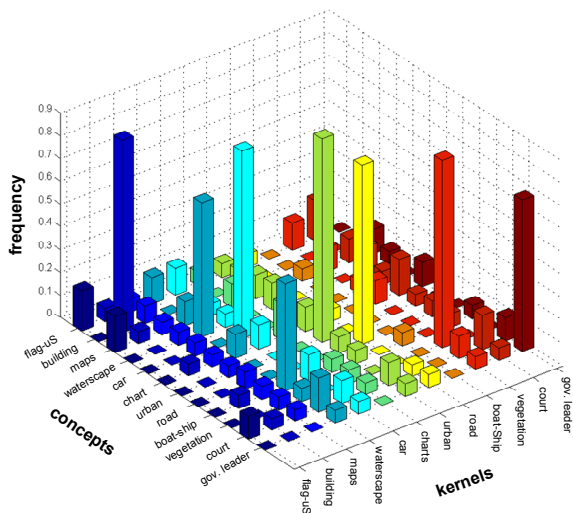
Figure 9. *Sharing of kernels among different concepts. Each bar gives the frequency of a concept using a kernel in 100 iterations.*
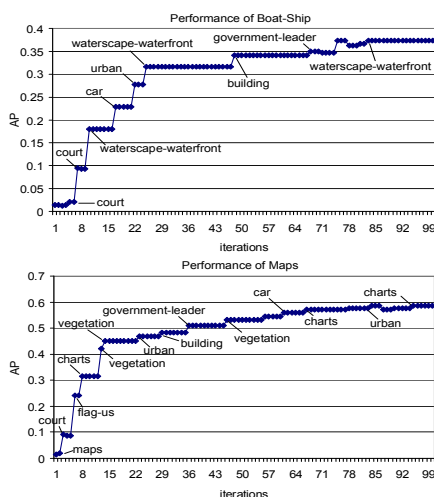

Figure 10. *Evolution of AP performance through iterations.*

developing an extended JointBoost framework, which automatically selected optimal kernels and subsets of sharing concepts in an iterative boosting process. Our joint boosting method can be applied for multi-class concept detection using any type of kernel. In this paper, we proposed a VSPM kernel for each individual concept based on the bag-of-features representation. Our method constructed multi-level kernels over multi-resolution visual vocabularies and multi-resolution spatial coordinates. We have tested our method in detecting 12 concepts (objects, scenes, etc) over 80+ hours of broadcast news videos from the TRECVID 2005 corpus, which is one of the largest and most challenging benchmark data sets for concept detection. Significant performance gains were achieved–10% in MAP and up to 34% AP for some concepts like "maps", "building", and "boat-ship". Extensive analysis of the results also revealed interesting relations among concepts. Our results also confirm the great potential of the proposed direction – leveraging strong visual tokens constructed from some concepts to help detection of challenging categories.

6. Acknowledgment

References

[1] A. Amir, *et al*. IBM research TRECVID-2004 video retrieval system. *Proc. NIST TRECVID Workshop*, 2004. 1, 5

[2] T.G. Dieterich. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, randomization. *Machine Learning*, 40(2), 1999. 6

[3] R. Fergus, *et al*. Object class recognition by unsupervised scale-invariant learning. *Proc.CVPR*, 264–271, 2003. 1, 2, 3

[4] J. Friedman and *et al*. Additive logistic regression: a statistical view of boosting, 1998. Dept. Statistics, Stanford University Technical Report. 2, 3, 4, 5

[5] K. Grauman and T. Darrel. The pyramid match kernel: discriminative classification with sets of image features. *Proc. ICCV*, 2:1458–1465, 2005. 1, 3

[6] K. Grauman and T. Darrel. Approximate correspondences in high dimensions. *Advances in NIPS*, 2006. 1, 2, 3

[7] A. Holub and P. Perona. A discriminative framework for modeling object class. *Proc. CVPR*, 1:664–671, 2005. 1

[8] W. Jiang and *et al*. Similarity-based online feature selection in content-based image retrieval. *IEEE Trans. on Image Processing*, (3):702–712, 2006. 1, 2

[9] S. Lazebnik and *et al*. Beyond bags of features: spatial pyramid matching for recognizing natual scene categories. *Proc. CVPR*, 2:2169–2178, 2006. 1, 2, 3, 5, 6

[10] LSCOM. DTO LSCOM Lexicon Definitions and Annotations. *http://www.ee.columbia.edu/dvmm/lscom/* 5

[11] E. Nowak and *et al*. Sampling strategies for bag-of-features image classification. *Proc. ECCV*, 2006. 1

[12] A. Oliva, *et al*. Modeling the shape of the scene:a holistic representation of the spatial envelope. *IJCV*, 43(2), 2001. 2

[13] A. Opelt and *et al*. Generic object recognition with boosting. *IEEE Trans. on PAMI*, 28(3):416–431, 2006. 1, 3

[14] A. Opelt and *et al*. Incremental learning of object detectors using a visual shape alphabet. *Proc.CVPR*, 1:3 10,2006. 4, 6

[15] R. Schapire and Y. Singer. Improved boosting algorithm using confidence-rated predictions. *Proc. Annual Conf. on Computational Learning Theory*, pages 80–91, 1998. 4

[16] K. Tieu and P. Viola. Boosting image retrieval. *Proc. CVPR*, pp.228–235, 2000. 4

[17] A. Torralba and *et al*. Sharing features: effective boosting procedure for multiclass object detecion. *Proc. CVPR*, 2:762–769, 2004. 2, 3, 4, 5, 6

[18] TRECVID. Trec video retrieval evaluations, 2005. in http://wwwnlpir. nist.gov/projects/trecvid/. 2, 5, 6

[19] C. Wallraven and *et al*. Recognition with local features: the kernel recipe. *Proc. ICCV*, 1:257–264, 2003. 1, 2

[20] D. Zhang, *et al*., A generative-discriminative hybrid method for multi-view object detection. *Proc. CVPR*, vol.2, 2006. 1

[21] J. Zhang and *et al*. Local features and kernels for classification of texture and object categories: an in-depth study, 2005. Technical Report RR-5737, INRIA Phône-Alpes. 1, 2