Mining Compositional Features for Boosting

Junsong Yuan
EECS Dept., Northwestern Univ.
Evanston, IL, USA
j-yuan@northwestern.edu

Jiebo Luo
Kodak Research Labs
Rochester, NY, USA
jiebo.luo@kodak.com

Ying Wu
EECS Dept., Northwestern Univ.
Evanston, IL, USA
yingwu@eecs.northwestern.edu

Abstract

*The selection of weak classifiers is critical to the success of boosting techniques. Poor weak classifiers do not perform better than random guess, thus cannot help decrease the training error during the boosting process. Therefore, when constructing the weak classifier pool, we prefer the quality rather than the quantity of the weak classifiers. In this paper, we present a data mining-driven approach to discovering compositional features from a given and possibly small feature pool. Compared with individual features (*e.g. *weak decision stumps) which are of limited discriminative ability, the mined compositional features have guaranteed power in terms of the descriptive and discriminative abilities, as well as bounded training error. To cope with the combinatorial cost of discovering compositional features, we apply data mining methods (frequent itemset mining) to efficiently find qualified compositional features of any possible order. These weak classifiers are further combined through a multi-class AdaBoost method for final multi-class classification. Experiments on a challenging* 10*-class event recognition problem show that boosting compositional features can lead to faster decrease of training error and significantly higher accuracy compared to conventional boosting decision stumps.*

1. Introduction

Weak classifier (a.k.a. weak learner, base classifier) plays an important role in boosting. Good weak classifiers, although not strong individually, can be linearly combined to construct an accurate final classifier through boosting.

In vision applications, decision stump (a decision tree of only two nodes) is one of the most favored types of weak classifiers, where each stump is usually associated with an individual visual feature (*e.g.* a Haar feature). Despite of previous successful application through boosting decision stumps [11][19], we noticed in some complex classification problems, such a decision stump can be extremely weak due to its limited discriminative ability in separating two or multiple classes. Boosting these poor decision stumps thus often leads to very long training phase because little

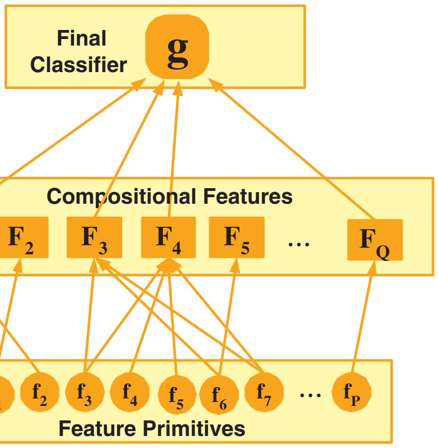

Final Classifier
g
Compositional Features
F_1 F_2 F_3 F_4 F_5 … F_Q
f_1 f_2 f_3 f_4 f_5 f_6 f_7 … f_P
Feature Primitives

Figure 1. Illustration of mining compositional features for boosting. Each $\mathbf{f}$ denotes a decision stump which we call it as a *feature primitive*. Our task is to discover compositional features $\mathcal{F}$ from feature primitive pool and boosting them to a strong classifier $\mathbf{g}$.

training error decreases at each boosting step. In such a case, better weak classifiers are desired, in order to decrease training error faster and obtain better generalization ability. Theoretically, it is an open question in boosting literature whether decision stumps or other types of weak classifiers (*e.g.* deeper decision trees) should be applied for boosting, because this depends on the structure of the unknown decision boundary function [4] [5] [15].

Given a pool of decision stumps $\mathbf{\Omega} = \{\mathbf{f}_i\}$, where each $\mathbf{f}_i$ is called a *feature primitive*, our goal is to discover strong enough compositional features $\mathcal{F} = \{\mathbf{f}_i\}$ for boosting (Fig. 1). Each compositional feature $\mathcal{F}$ contains one or more feature primitives, and its decision is determined by the responses of feature primitives. To balance between generalization and discrimination abilities, we require $\mathcal{F}$ to be both (1) descriptive features (*i.e.* high frequency in positive training data), and (2) discriminative features (*i.e.* high accuracy in prediction). Although feature primitives can be rather weak individually, we show that an appropriate composition of them can have guaranteed discriminative power with a guaranteed bound of training error. Compared with boosting decision stumps, boosting these higher-order rules

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.

can lead to faster convergence in training as well as better generalization if the decision boundary function is *not* in an additive form of original feature primitives [5].

To reduce the combinatorial cost in searching for compositional features [20], we apply data mining methods such as frequent itemset mining (FIM) for pattern discovery. Due to their computational efficiency, data mining methods are becoming popular in many vision applications, including visual object detection [14], classification [2] [3] [13] and image pattern discovery [21]. After the compositional features are discovered, we boost them by applying a multi-class AdaBoost method: stagewise additive modeling with exponential loss (SAMME) [22]. SAMME directly handles the K-class problem by building a single K-class classifier, instead of K binary ones. The solution of SAMME is consistent with the Bayes classification rule, thus it is optimal in minimizing the misclassification error. Experimental results of both simulated data and a challenging 10-class visual event recognition problem validate the advantages of boosting compositional features.

2. Induced Transactions for Data Mining

We consider a K-class classification problem. Suppose we are given a training dataset containing N samples of K classes: $\mathcal{D}_N = \{\mathbf{x}_t, c_t\}_{t=1}^N$, where $\mathbf{x}_t \in \mathbb{R}^P$ denotes the feature vector and $c_t \in \{1, 2, ..., K\}$ is the label of $\mathbf{x}_t$. The task is to find a classifier $\mathbf{g}(\cdot) : \mathbf{x} \rightarrow c$ from the training data, such that given a new query sample $\mathbf{x}$, we can assign it a class label $c \in \{1, 2, ..., K\}$. Instead of using the raw features $\mathbf{x}$ directly to estimate c, we consider a collection of induced binary features $\{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_P\}$ where each $\mathbf{f}_i : \mathbf{x} \rightarrow \{0, 1\}$ is a *feature primitive*. For example, $\mathbf{f}_i$ can be a *decision stump*:

$$\mathbf{f}_i(\mathbf{x}) = \begin{cases} f_i^+ & if\ \mathbf{x}(i) \geq \theta_i \\ f_i^- & if\ \mathbf{x}(i) < \theta_i \end{cases}, \quad (1)$$

or a decision stump when only positive response is considered:

$$\mathbf{f}_i(\mathbf{x}) = \begin{cases} f_i & if\ \mathbf{x}(i) \geq \theta_i \\ \emptyset & if\ \mathbf{x}(i) < \theta_i \end{cases}. \quad (2)$$

Here $\mathbf{x}(i)$ is value of $\mathbf{x}$ in the i_{th} dimension, and $\theta_i \in \mathbb{R}$ is the quantization threshold for $\mathbf{f}_i$. We call f_i the *feature item* associated with the feature primitive $\mathbf{f}_i$.

Without loss of generality, we use the decision stump considering positive response only (Eq. 2) for illustration. Given a collection of P features, we have an *item vocabulary* $\mathbf{\Omega} = \{f_1, f_2, ..., f_P\}$ containing P items. As illustrated in Fig. 2, now a training sample $\mathbf{x} \in \mathbb{R}^P$ can be transferred into a *transaction*:

$$\mathcal{T}(\mathbf{x}) = \{\mathbf{f}_1(\mathbf{x}), \mathbf{f}_2(\mathbf{x}) ..., \mathbf{f}_P(\mathbf{x})\} \subseteq \mathbf{\Omega},$$

according to the responses of P feature primitives. The induced transaction dataset $\mathbf{T} = \{\mathcal{T}_t\}_{t=1}^N$ contains a collection of N training samples, where each $\mathcal{T}$ corresponds to a data sample $\mathbf{x}$. By transforming continuous features $\mathbf{x} \in \mathbb{R}^P$ into discrete transactions, we can perform traditional data mining algorithm, such as frequent itemset mining. In Sec. 3 and Sec. 4, we discuss how to take advantage of efficient data mining method to search for informative features for classification.

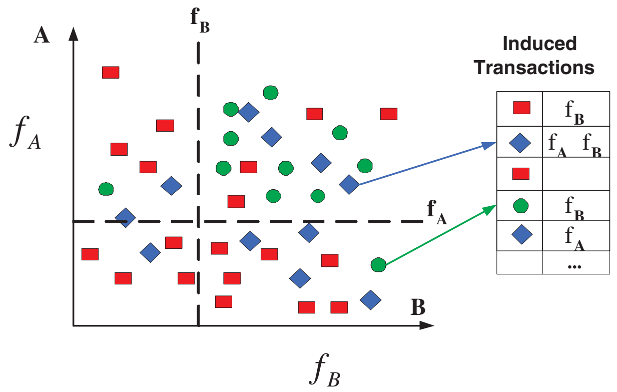


Figure 2. Illustration of the induced transaction. By partitioning the feature space into sub-regions through decision stumps $\mathbf{f}_A$ and $\mathbf{f}_B$, we can index the training samples in terms of the sub-regions they are located. Only positive responses are considered. For example, a transaction of $\mathcal{T}(\mathbf{x}) = \{f_A, f_B\}$ indicates that $\mathbf{f}_A(\mathbf{x}) > \theta_A$ and $\mathbf{f}_B(\mathbf{x}) > \theta_B$.

3. Mining Compositional Features

For each feature primitive $\mathbf{f}_i$, we can use it to predict the class label. A *primitive classification rule* is thus of the form:

$$\mathbf{f}_i(\mathbf{x}) = f_i \quad \Longrightarrow \quad \hat{c}(\mathbf{x}) = k,$$

where $k \in \{1, 2, ..., K\}$, and $\hat{c}(\mathbf{x})$ is the predicted label of $\mathbf{x}$. Since a classification rule based on an individual $\mathbf{f}$ is usually of low accuracy, it is of our interests to find compositional feature $\mathcal{F} = \{f_i\} \subseteq \mathbf{\Omega}$ which can be more accurate. Given a compositional feature $\mathcal{F}$, we define its classification rule as:

$$\mathcal{F}(\mathbf{x}) = \begin{cases} k & if\ \mathcal{F} \subseteq \mathcal{T}(\mathbf{x}) \\ 0 & otherwise \end{cases}, \quad (3)$$

where $k \in \{1, 2, ..., K\}$ is the predicted class; $\mathcal{F}(\mathbf{x}) = 0$ implies that $\mathcal{F}$ cannot make decision on $\mathbf{x}$.

Following the terms in data mining literature, we call $\mathcal{F}$ as a *feature item-set*. Given an itemset $\mathcal{F}$, the transaction $\mathcal{T}_t$ which includes $\mathcal{F}$ is called an *occurrence* of $\mathcal{F}$, *i.e.*, $\mathcal{T}_t$ is an occurrence of $\mathcal{F}$, if $\mathcal{F} \subseteq \mathcal{T}(\mathbf{x}_t)$. We denote by $\mathbf{T}(\mathcal{F})$ the set of all occurrences of $\mathcal{F}$ in $\mathbf{T}$, and the *frequency* of an itemset $\mathcal{F}$ is denoted by:

$$frq(\mathcal{F}) = |\mathbf{T}(\mathcal{F})| = |\{t : \mathcal{F} \subseteq \mathcal{T}(\mathbf{x}_t)\}|.$$

Considering it is important to evaluate the quality of a compositional classification rule, we first give an analysis of the perfect rule.

Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.

Definition 1 perfect classification rule
A compositional classification rule $\mathcal{F}^*$ is perfect if $\exists\ k \in \{1, 2, ...K\}$*, such that*

$$\begin{aligned} descriptive: \quad & P(\mathcal{F}^*(\mathbf{x}) = k | c(\mathbf{x}) = k) = 1 \quad (4) \\ discriminative: \quad & P(c(\mathbf{x}) = k | \mathcal{F}^*(\mathbf{x}) = k) = 1 \quad (5) \end{aligned}$$

In Definition 1, we specify two conditions for the perfect rule $\mathcal{F}^*$, where $P(\mathcal{F}^*(\mathbf{x}) = k|c(\mathbf{x}) = k) = 1$ is the descriptive ability, while $P(c(\mathbf{x}) = k|\mathcal{F}^*(\mathbf{x}) = k) = 1$ is the discriminative ability. Since its classification result is the same as the ground truth, $\mathcal{F}^*$ is the best possible rule for class k. However, exhaustive search for $\mathcal{F}^*$ is computationally demanding due to the combinatorial complexity. Because each $\mathbf{f}_i$ can generate two possible outcomes: f_i or $\emptyset$, the total number of all possible classification rules is $2^{|\mathbf{\Omega}|}$. Thus efficient search methods are required to make the feature selection process computationally feasible. Even worse, such a perfect rule $\mathcal{F}^*$ may not always exist in the case of noisy training data [12], where positive and negative samples are not perfectly separable. In such a case, we need to sacrifice the strict conditions of selecting optimal $\mathcal{F}^*$ for sub-optimal ones. In other words, instead of searching for perfect rule $\mathcal{F}^*$, we search for a collection of weaker rules $\mathbf{\Psi} = \{\mathcal{F}_i\}$. With its justification later, we define the sub-optimal compositional rule in Definition 2.

Definition 2 *: (λ_1, λ_2)-***compositional rule**
A compositional feature $\mathcal{F} \subset \mathbf{\Omega}$ is called (λ_1, λ_2)-compositional rule if $\exists\ k \in \{1, 2, ..., K\}$*, such that:*

$$\begin{aligned} sup.: \quad P(\mathcal{F}(\mathbf{x}) = k) \quad & \geq \quad \lambda_1 \\ conf.: \quad P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k) \quad & \geq \quad \lambda_2 \times P(c(\mathbf{x}) = k) \end{aligned}$$

The first condition requires that $\frac{frq(\mathcal{F})}{N} \geq \lambda_1$, which is the support requirement in mining frequent patterns [7]. A rule of low support covers few training samples. Such a classification rule has limited ability to generalize, even if it can predict accurately on few number of training samples. The second condition requires that the rule is accurate enough for prediction, such that most covered samples are correctly classified. This condition corresponds to the confidence of a rule in data mining literature [7]. Different from traditional data mining methods which usually set a fixed confidence threshold, we consider the class prior to handle imbalanced training data. A weak rule $\mathcal{F}$ that satisfies both conditions are viewed as useful rules for future use.

To further justify our criteria of (λ_1, λ_2)-compositional rule, we develop Definition 2 into two weak conditions:

$$\begin{aligned} P(\mathcal{F}(\mathbf{x}) = k|c(\mathbf{x}) = k) \quad & \geq \quad \lambda_2 \times P(\mathcal{F}(\mathbf{x}) = k), \quad (6) \\ P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k) \quad & \geq \quad \lambda_2 \times P(c(\mathbf{x}) = k), \quad (7) \end{aligned}$$

where Eq. 6 is obtained because

$$\begin{aligned} P(\mathcal{F}(\mathbf{x}) = k|c(\mathbf{x}) = k) \quad & = \quad \frac{P(\mathcal{F}(\mathbf{x}) = k, c(\mathbf{x}) = k)}{P(c(\mathbf{x}) = k)} \\ & \geq \quad \frac{P(\mathcal{F}(\mathbf{x}) = k)\lambda_2 P(c(\mathbf{x}) = k)}{P(c(\mathbf{x}) = k)} \\ & = \quad \lambda_2 P(\mathcal{F}(\mathbf{x}) = k). \end{aligned}$$

Comparing the conditions for perfect rule (Definition 1) with Eq. 6 and Eq. 7, we can see that weak rules in Definition 2 only need to satisfy weak descriptive and discriminative conditions, thus they are sub-optimal features compared with perfect feature $\mathcal{F}^*$.

We further notice that the two requirements in Eq. 6 and Eq. 7 are actually an equivalent one:

$$\frac{P(\mathcal{F}(\mathbf{x}) = k, c(\mathbf{x}) = k)}{P(c(\mathbf{x}) = k)P(\mathcal{F}(\mathbf{x}) = k)} \geq \lambda_2,$$

given $P(\mathcal{F}(\mathbf{x}) = k) \geq \lambda_1$. When $\frac{P(\mathcal{F}(\mathbf{x})=k,c(\mathbf{x})=k)}{P(c(\mathbf{x})=k)P(\mathcal{F}(\mathbf{x})=k)} = 1$, it indicates independent events $c(\mathbf{x}) = k$ and $\mathcal{F}(\mathbf{x}) = k$. In order to make sure $\mathcal{F}$ is informative for prediction (*e.g.* performing better than random guess), we require $\lambda_2 > 1$. The other parameter $0 < \lambda_1 \leq 1$ controls the support of a rule, which influences the generalization ability of the rule.

Moreover, according to Eq. 6, we need $\lambda_2 P(\mathcal{F}(\mathbf{x}) = k) \leq 1$. Since $P(\mathcal{F}(\mathbf{x}) = k) \geq \lambda_1$, we have

$$\lambda_1 \leq P(\mathcal{F}(\mathbf{x}) = k) \leq \frac{1}{\lambda_2}, \quad (8)$$

which indicates that qualified rules are those of mild-frequency. This actually explains why we need to discard the most common and uncommon words in the "bag-of-word" approach [16]. Here, it says that we should discard not only common and uncommon words, but also common and uncommon word combinations. As we can see in Eq. 6 and Eq. 7, such "word-combinations" of mild frequency are informative features for classification.

Based on Eq. 8, we further have $\lambda_1\lambda_2 \leq 1$. Let $r_k = P(c(\mathbf{x} = k))$, we also have $\lambda_2 \leq \frac{1}{r_k}$ since we need $\lambda_2 P(c(\mathbf{x}) = k) \leq 1$ in Eq. 7. Combining all results, we obtain the conditions for feasible parameters λ_1 and λ_2:

Proposition 1 feasible parameters of data mining
The following requirements must be satisfied to avoid mining non-informative or an empty set of (λ_1, λ_2)*-compositional rules according to Definition 2.*

$$0 < \lambda_1 \leq \frac{1}{\lambda_2} < 1 < \lambda_2 \leq \min\{\frac{1}{\lambda_1}, \frac{1}{r_k}\}. \quad (9)$$

Eq. 9 thus gives the guidance in selecting λ_1 and λ_2 for effective data mining, which avoids mining in vain for compositional features.

Based on Proposition 1, we present the major theoretical result in this paper in Theorem 1, where we show that λ_1 and λ_2 can bound the training error of the (λ_1, λ_2)-compositional rule $\mathcal{F}$.

Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.

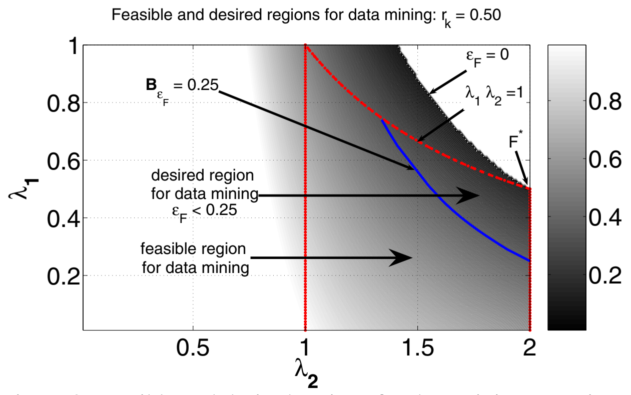


Figure 3. Feasible and desired regions for data mining. We simulate the upper bound $\mathbf{B}_{\epsilon_{\mathcal{F}}} = \frac{1}{\lambda_2} - \lambda_1\lambda_2 r_k$, with a specific choice of $r_k = 0.5$. The shading region corresponds to $0 \leq \mathbf{B}_{\epsilon_{\mathcal{F}}} \leq 1$. As shown in the color bar, the darker the shade, the smaller the upper bound of ϵ and thus are more favorable for data mining. The shading region above the red curve ($\lambda_1\lambda_2 > 1$) is infeasible for data mining since no rule exists there according to Proposition 1. The region below the red curve ($\lambda_1\lambda_2 \leq 1$) and between two red lines ($1 \leq \lambda_2 \leq \frac{1}{r_k}$) is the feasible choice of data mining parameters λ_1 (support) and λ_2 (confidence). The region below the red curve while above the blue curve ($\mathbf{B}_{\epsilon_{\mathcal{F}}} = 0.25$) is the desired region where we can get rules with $\epsilon_{\mathcal{F}} \leq 0.25$. At the optimal point $\lambda_1 = \frac{1}{\lambda_2} = r_k$, where the red curve (tight bound of training error) meets the boundary $\epsilon_{\mathcal{F}} = 0$, we get minimum training error and thus the perfect classification rule.

Theorem 1 training error bound of (λ_1, λ_2)-comp rule *For a (λ_1, λ_2)-compositional rule $\mathcal{F}$ predicting for the k_{th} class, its training error of class k is upper bounded by:*

$$\epsilon_{\mathcal{F}} \leq \frac{1}{\lambda_2} - \lambda_1\lambda_2 r_k = \mathbf{B}_{\epsilon_{\mathcal{F}}}, \quad (10)$$

where $r_k = P(c(\mathbf{x}) = k)$ denotes the prior of class k and $\epsilon_{\mathcal{F}} = P(\mathcal{F}(\mathbf{x}) \neq c(\mathbf{x})|\mathcal{D}_N)$ is the empirical error on training data $\mathcal{D}_N$. Specifically, the upper bound is tight, i.e. *equality holds in Eq. 10, if (1) the two equalities hold in Definition 2 and (2) $\lambda_1\lambda_2 = 1$. Moreover, when $\lambda_1 = \frac{1}{\lambda_2} = r_k$, we have $\epsilon_{\mathcal{F}} = 0$.*

The prove of Theorem 1 is in the Appendix. As illustrated in Fig. 3, Theorem 1 states that given a frequent pattern $\mathcal{F}$, we can upper bound its training error on class k by its support (λ_1) and confidence of the association rule regarding to class k (λ_2). When $\lambda_1\lambda_2 = 1$, the bound is tight.

In mining (λ_1, λ_2)-compositional rules in Definition 2, we perform a two-step method in Alg. 1. First, we perform through frequent itemset mining (FIM) algorithms to find compositional features with high support. After that we filter non-discriminative rules that do not satisfy the confidence condition. As each (λ_1, λ_2)-compositional feature can also be a discriminative classification rule according to Eq. 6 and Eq. 7, Alg. 1 finally ends up with a collection of sub-optimal classification rules whose training error is upper bounded according to Theorem 1.

Although it is impossible to search for perfect compositional features directly due to the combinatorial complexity, it is computationally efficient to perform such a two-step method in Alg. 1, by taking advantage of the FIM algorithms. Specifically, FIM algorithms takes advantage of the monotonic property of frequent itemsets (Apriori algorithm) or applying a prefix-tree structure store compressed information of frequent itemsets (FP-growth algorithm), in order to find them efficiently [7]. In this paper we apply the FP-growth algorithm to implement closed-FIM [6] for discovering frequent patterns.

Algorithm 1: Mining Compositional Rules

```
input  : Training dataset $\mathcal{D} = \{\mathcal{T}_i, c_i\}_{i=1}^N$, where $\mathcal{T}_i \subseteq W_{\mathbf{\Omega}}$,
         $c_i \in \{1, 2, ..., K\}$
         parameters: $\lambda_1, \lambda_2$ satisfying Eq. 9
output : a pool of weak rules: $\mathbf{\Psi} = \{\mathcal{F}_i\}$
1 Init: $\mathbf{\Psi} = \emptyset$;
2 FIM: Perform closed FIM on $\mathbf{T}$ based on the support
  parameter $\lambda_1$, and obtain a set of compositional features
  $\mathbf{\Psi} = \{\mathcal{F} : frq(\mathcal{F}) > \lfloor \lambda_1 \times N \rfloor\}$
3 foreach $\mathcal{F} \in \mathbf{\Psi}$ do
4   if $P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k) < \lambda_2 r_k, \forall k$ then
5     $\mathbf{\Psi} \longleftarrow \mathbf{\Psi} \backslash \mathcal{F}$
6 Return $\mathbf{\Psi}$
```

4. Multi-class AdaBoost

After discovering $\mathbf{\Psi} = \{\mathcal{F}_i\}$, we need to boost them for a final classifier. Each mined rule $\mathcal{F}_i \in \mathbf{\Psi}$ is a weak classifier for a certain class $k \in \{1, 2, ..., K\}$. We follow the stagewise additive modeling with exponential loss (SAMME) formulation for multi-class AdaBoost in [22]. Given the training data $\{\mathbf{x}_i, c_i\}$, our task is to find a regression function $\mathbf{g} : \mathbf{x} \rightarrow \mathbb{R}^K$, *i.e.*, $\mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), ..., g_K(\mathbf{x})]^T$, to minimize the following objective function:

$$\min_{\mathbf{g}(\mathbf{x})} \quad \sum_{i=1}^{N} L(\mathbf{y}_i, \mathbf{g}(\mathbf{x}_i)) \quad (11)$$

$$s.t. \quad g_1 + ... + g_K = 0, \quad (12)$$

where $L(\mathbf{y}, \mathbf{g}) = \exp\left[-\frac{1}{K}(y_1 g_1 + ... + y_K g_K)\right] = \exp(-\frac{1}{K}\mathbf{y}^T\mathbf{g})$ is the multi-class exponential loss function. $\mathbf{y} = (y_1, ..., y_K)^T$ is the K-dimensional vector associated with the output c, where

$$y_k = \begin{cases} 1 & if\ c = k \\ -\frac{1}{K-1} & if\ c \neq k \end{cases}. \quad (13)$$

The symmetric constrain $g_1 + ... + g_K = 0$ is to guarantee the unique solution of $\mathbf{g}$, otherwise adding a constant to all g_k will give the same loss since $\sum_{i=1}^K y_i = 0$. It can be seen that the optimal solution of Eq. 12 is:

$$\mathbf{g}^*(\mathbf{x}) = \arg\min_{\mathbf{g}(\mathbf{x})} E_{\mathbf{y}|\mathbf{x}} \exp[-\frac{1}{K}(y_1 g_1 + ... + y_K g_K)]$$
$$s.t. \quad g_1 + ... + g_K = 0. \quad (14)$$

Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.

It is notable that the solution of Eq. 14 is consistent with the Bayes classification rule in minimizing the misclassification error [22]:

$$\arg\max_k g_k^*(\mathbf{x}) = \arg\max_k Prob(c = k|\mathbf{x}). \qquad (15)$$

Compared with AdaBoost.MH which needs to perform K one-against-all classifications, SAMME performs K-class classification directly. It only needs weak classifiers better than random guess (*e.g.* correct probability larger than $1/K$), rather than better than $1/2$ as two-class AdaBoost requires.

We modify original SAMME [22] in considering compositional features for boosting. By boosting compositional features, we actually consider a functional ANOVA decomposition [5] of $\mathbf{g}(\mathbf{x})$ by applying weak rules $\mathcal{F}$ of any possible orders:

$$\begin{aligned}\mathbf{g}(\mathbf{x}) &= \sum_{m=1}^{M} \alpha^m \mathcal{F}^m(\mathbf{x}) \\ &= \sum \mathcal{F}_i(\mathbf{x}(i)) + \sum \mathcal{F}_{ij}(\mathbf{x}(i), \mathbf{x}(j)) + ...,\end{aligned}$$

where $\alpha^m \in \mathbb{R}$ are weight coefficients.

Our compositional boosting method is listed in Alg. 2. From Alg. 2, we can see one major difference between SAMME and two-class AdaBoost is in step 8, where an extra $\log(K-1)$ is added to guarantee $\alpha^m > 0$ when $err^m < \frac{K-1}{K}$. In the case of $K = 2$, it is equivalent to the original two-class AdaBoost because $\log(K-1) = 0$.

Algorithm 2: SAMME on Compositional Features

```
input  : A pool of compositional rules $\mathbf{\Psi} = \{\mathcal{F}_j\}$
         Training dataset $\mathcal{D} = \{\mathbf{x}_i, c_i\}_{i=1}^N$
         Iteration number, $M$
output : a strong classifiers: $\mathbf{g}(\cdot) : \mathbf{x} \rightarrow \{1, 2, ..., K\}$
1  Init: set the training sample weights $w_i = 1/N$,
   $i = 1, 2, ..., N$.
2  for $m = 1, 2, ..., M$ do
3      Select a classifier from the pool $\mathbf{\Psi}$:
4          $\mathcal{F}^m(\mathbf{x}) = \arg\min_{\mathcal{F}_i \in \mathbf{\Psi}} \sum_{i=1}^N w_i \mathbb{I}(c_i \neq \mathcal{F}^m(\mathbf{x}_i))$,
5          $\mathbf{\Psi} = \mathbf{\Psi} \backslash \mathcal{F}$.
6      Compute weighted training error:
7          $err^m = \frac{\sum_{i=1}^N w_i \mathbb{I}(c_i \neq \mathcal{F}^m(\mathbf{x}_i))}{\sum_{i=1}^N w_i}$.
8      Compute: $\alpha^m = \log \frac{1-err^m}{err^m} + \log(K-1)$.
9      Update sample weight:
10         $w_i \leftarrow w_i \cdot \exp[\alpha^m \mathbb{I}(c_i \neq \mathcal{F}^m(\mathbf{x}_i))]$.
11     Re-normalize $w_i$.
12 Return $\mathbf{g}(\mathbf{x}) = \arg\max_k \sum_{m=1}^M \alpha^m \cdot \mathbb{I}(\mathcal{F}^m(\mathbf{x}) = k)$
```

Moreover, It is important to notice that each $\mathcal{F}$ is only a binary classifier for a specific class k. We transfer it into a K-class classifier by randomly guessing the rest $K-1$ class label when $\mathcal{F}(\mathbf{x}) = 0$. Specifically we estimate the weighted training error at step 4 as:

$$\begin{aligned}&\sum_{i=1}^{N} w_i \mathbb{I}(c_i \neq \mathcal{F}^m(\mathbf{x})) \\ =\ & \sum_{c_i \neq k} w_i \mathbb{I}(\mathcal{F}^m(\mathbf{x}) = k) + \frac{K-1}{K} \sum_{i=1}^{N} w_i \mathbb{I}(\mathcal{F}^m(\mathbf{x}) = 0),\end{aligned}$$

where $\mathbb{I}(\cdot)$ denotes the binary indicator function. Therefore $\mathcal{F}$ is a K-class classifier performing better than random guess if its error bound $\epsilon_{\mathcal{F}} < \frac{1}{2}$.

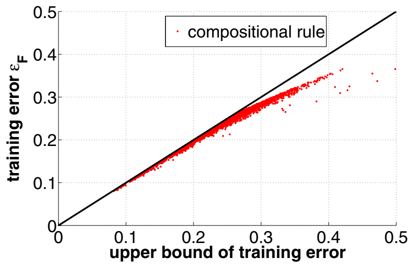
5. Experiments

5.1. UCI data sets

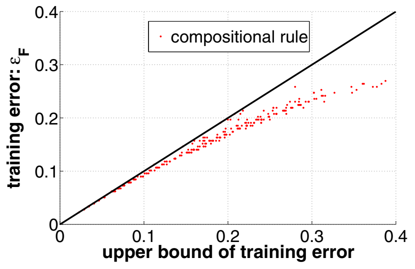
To validate Theorem 1, we select 3 data sets from the UCI Machine Learning Repository for evaluation: (1) breast cancer Wisconsin (diagnostic) data set which consists of both malignant and benign samples, (2) wine data set which consists of 3 classes of data samples, and (3) multiple features data set which consists of handwritten numerals ('0'–'9') extracted from a collection of Dutch utility maps. We apply different strategies to quantize the continuous features in 3 data sets. For the breast cancer and wine data sets, as there are only a few features (30 and 13 respectively), we select the mean value ($\theta_i = \mu_i = E[\mathbf{x}(i)]$) at each individual feature $\mathbf{x}_i$ for quantization and consider both positive and negative items in generating transactions (Eq. 1). For the handwritten numerals data set, each sample contains 649 features. To alleviate the computational cost of FIM, we apply $\theta_i = \mu_i + \sigma_i$ for quantization and only consider the positive items ($\mathbf{x}(i) \geq \theta_i$) in generating transactions (Eq. 2), where σ_i is the standard variance of the i_{th} feature $\mathbf{x}(i)$.

For each data set, we set $\lambda_1 = \min_k r_k$ and $\lambda_2 = \frac{1}{2\lambda_1}$, such that $\lambda_1\lambda_2 = 0.5 < 1$. The compositional feature discovery result is presented in Table 5.1. We discover in total 12597, 266 and 48452 qualified (λ_1, λ_2)-compositional rules for the breast cancer, wine and handwritten numeral data set, respectively. The best class-specific rule discovered for these three data sets has training error 0.081, 0.022, and 0.017, respectively. All the discovered rules has smaller training error than the theoretical upper bound $\mathbf{B}_{\epsilon_{\mathcal{F}}}$. To further compare the real training error $\epsilon_{\mathcal{F}}$ and the upper bound, we present all discovered rules in Fig. 4. Each point corresponds to a discovered compositional rule $\mathcal{F}$. The x-coordinate of a point gives its real classification error $\epsilon_{\mathcal{F}}$. The y-coordinate gives the upper bound training error calculated on its own support and confidence values according to Theorem 1). It is surprising to see that for most discovered rules in all data sets, the real classification error of a rule $\mathcal{F}$ is close to its own theoretical upper bound. We also notice the smaller the $\epsilon_{\mathcal{F}}$, the closer $\epsilon_{\mathcal{F}}$ to its own upper bound. These results show that the derived upper bound in Theorem 1 is quite tight for compositional rules.

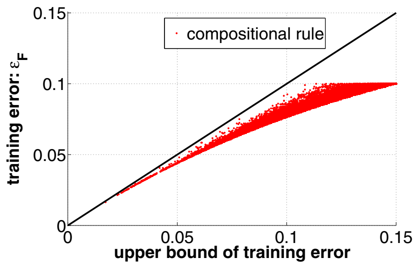
Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.



(a)



(b)



(c)

Figure 4. Comparison of real training error $\epsilon_{\mathcal{F}}$ and its theoretical upper bound: (a) breast cancer data set, (b) wine data set, and (c) handwritten numeral data set. The closer the point to the 45 degree line, the tighter the upper bound.

Table 1. Compositional feature discovery in 3 UCI data sets.

	cancer	wine	numeral
# class	2	3	10
# feature	30	13	649
# sample	569	178	2000
θ_i	μ_i	μ_i	$\mu_i + \sigma_i$
λ_1	0.37	0.27	0.10
λ_2	1.34	1.85	5.00
# frq. itemsets	12,729	342	156,734
# comp. feature	12,597	266	48,452
$\mathbf{B}_{\epsilon_{\mathcal{F}}}$	0.561	0.406	0.150
max $\epsilon_{\mathcal{F}}$	0.366	0.270	0.100
min $\epsilon_{\mathcal{F}}$	0.081	0.022	0.017
aver. $\epsilon_{\mathcal{F}}$	0.241	0.150	0.081

5.2. An event recognition problem

The real test is an event recognition problem. The goal is to recognize typical events from personal photo collections, where each event corresponds to a specific human activity taking place in a certain environment, and captured by a collection of images taken during the event: $\mathbf{E}_i = \{I_j\}_{j=1}^{|\mathbf{E}_i|}$, where I_j denotes an image. We chose 10 types of frequently occurring events with reasonably distinctive visual characteristics, inspired by the tag statistics revealed by Flickr.com: $\mathcal{C} = \{$*Christmas activity, backyard activity, ball game, beach, birthday, city walk, hiking, road trip, skiing, wedding*$\}$. They include both outdoor and indoor events. In general, event recognition is more challenging and complicated than scene recognition due to the higher semantics involved [9] - the visual content can vary dramatically from one instance to another (as shown later, boosting decision stumps did not perform well).

For each event $\mathbf{E}$, it can be uniquely labeled with one of the 10 event classes: $l(\mathbf{E}_i) \in \mathcal{C}$. The experimental dataset contains 88 individual events, where each event contains a variable number of 7 to 108 images. There are 3453 images in total in the dataset. Due to the limited number of events, we perform leave-one-out test to report all results. Without extra mentioning, the support threshold is set as $\lambda_1 = \frac{1}{11} \approx \min_k r_k$ and the confidence threshold is set as $\lambda_2 = 4$. Therefore for a specific class k, its training error bound is: $\epsilon_k \leq \frac{1}{\lambda_2} - \lambda_1 \lambda_2 r_k = \frac{1}{4} - \frac{4}{11} r_k$ (theorem 1). The quantization parameters θ_i determine the transactions and thus have large influences on the mining and classification results. To carefully select θ_i, for each feature dimension $\mathbf{x}(i)$, we estimate its mean $\mu_i = E[\mathbf{x}(i)]$ and variance $\sigma^2 = Var[\mathbf{x}(i)]$. Then we set $\theta_i = \mu_i + \tau \times \sigma_i$, where $\tau > 0$ is a global parameter decided though leave-out-out cross validation.

Visual vocabularies and feature primitives

Visual vocabularies have proved to be an effective way of building visual recognition systems, *e.g.*, for scene recognition [8]. An image is partitioned by a fixed grid and represented as an unordered set of image patches. Suitable descriptions are computed for such image patches and clustered into bins to form a "visual vocabulary". In this study, we adopted the same methodology and extended it to consider both color and texture features for characterizing each image grid.

To extract color features, an image grid is further partitioned into 2×2 equal size sub-grids. Then for each subgrid, we extract the mean R, G and B values to form a $4 \times 3 = 12$ feature vector which characterizes the color information of 4 sub-grids. To extract texture features, we apply a 2×2 array of histograms with 8 orientation bins in each sub-grid. Thus a $4 \times 8 = 32$-dimensional SIFT descriptor is applied to characterize the structure within each image grid, similar in spirit to [8] [1]. In our experiments, if an image is larger than 200k pixels, we first resize it to 200k. We then set image grid size of 16×16 with overlapping sampling interval 8×8. Typically, one image generates 117 such grids.

After extracting all the raw image features from image grids, we build separate color and texture vocabularies by clustering all the image grids in the training dataset through k-means clustering. In our experiments, we set both vocabularies of size 500. By accumulating all the grids in an event (a collection of images), we obtain two normalized histograms for an event, $\mathbf{h}^c$ and $\mathbf{h}^t$, corresponding to the word distribution of color and texture vocabularies,

Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.

respectively. Concatenating $\mathbf{h}^c$ and $\mathbf{h}^t$, we end up with an normalized word histogram: $\sum_{i=1}^{1000} \mathbf{h}_i(\mathbf{E}) = 1$. Each bin in the histogram indicates the occurrence frequency of the corresponding word. We only consider the positive responses of $\mathbf{f}_i$ when the i_{th} word appears frequently enough (*i.e.* $\mathbf{h}_i(\mathbf{E}) > \theta_i$) in the whole event. We have two types of visual vocabularies $\mathbf{\Omega}_c$ and $\mathbf{\Omega}_t$, where $\mathbf{f}_i \in \mathbf{\Omega}_c$ is the color primitive, whereas $\mathbf{f}_i \in \mathbf{\Omega}_t$ is the texture primitive. Denoting the complete vocabulary as $\mathbf{\Omega} = \mathbf{\Omega}_c \cup \mathbf{\Omega}_t$, we discover compositional rule $\mathcal{F} \subset \mathbf{\Omega}$ which can contain integrated information of both color and texture.

Event recognition results

The leave-one-out test result is showed in Table 2, when $\tau = 1.4$ ($\theta_i = \mu_i + 1.4\sigma_i$). The iteration number of boosting is 400. The main confusion comes from the indoor social events, such as *birthday*, *Christmas* and *wedding*. We also notice confusion between *hiking* and *backyard activities*, possibly due to their visual similarity.

Table 2. Boosting compositional features: class confusion matrix of leave-one-out test results. Each row indicates the classification results of the corresponding class. The overall accuracy is 80.7%.

	Ch	by	bg	be	bi	ci	hi	rt	sk	we
Chr	**8**	0	0	0	1	0	0	0	0	1
byd	0	**4**	0	0	0	1	3	0	0	0
bgm	1	1	**4**	0	0	0	0	1	0	0
bea	0	0	0	**8**	0	0	0	0	0	0
bir	2	0	1	0	**7**	0	0	0	0	0
cit	0	0	0	0	0	**9**	0	0	0	0
hik	0	0	0	0	0	0	**10**	0	0	0
rtp	0	0	0	0	0	1	0	**7**	1	0
ski	0	0	0	1	0	0	0	1	**5**	0
wed	1	0	0	0	0	0	0	0	0	**9**

The composition of the mined feature pool $\mathbf{\Psi}$ is listed in table 3. Among the 400 weak classifiers finally selected through boosting, 42% of them are compositional features of high-orders and the rest are decision stumps.

Table 3. Order distribution of the mined weak classifier pool $\mathbf{\Psi}$. The values are averaged by all the leave-one-out tests.

$\lvert\mathcal{F}\rvert$	1	2	3	4	≥ 5	total
mined #	271.6	120.1	62.9	31.2	40.2	525.9
used #	232.3	86.1	37.6	16.8	27.1	400.0

We compare the results of boosting compositional features with conventional decision stumps in Fig. 5. Due to the high redundancy among compositional features because of sharing primitive features, we do not allow a same compositional feature to be selected again during boosting (see step 5 in Alg. 2). In comparison, we allow re-used stumps to follow conventional boosting. After feature mining, we obtain a pool of 526 compositional features. On the other hand, the pool of decision stumps contains 1000 features,

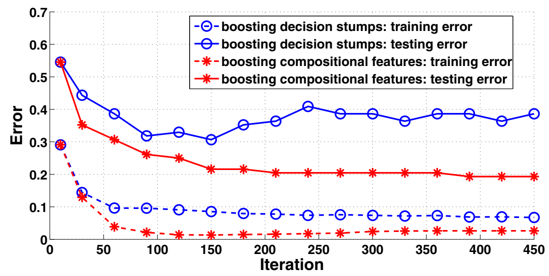


Figure 5. Comparison of error between boosting decision stumps and boosting compositional features. The error is calculated through averaging all leave-one-out tests.

which correspond to 2000 types of weak rules since we consider both positive and negative items of an individual feature. The quantization parameters are the same as those of compositional features. From Fig. 5, we notice that training error decreases faster in our method (iteration 150, error 1.3%) than in conventional decision stumps (iteration 400, error 6.7%). Better generalization ability is also obtained. Despite that more complex weak features are applied in our method, we do not observe over-fitting in the first 450 iterations, whereas boosting decision stumps surprisingly does. This validates the regularization of the discovered rules by their support. In summary, while this event recognition problem is indeed challenging for conventional decision stumps (70.0%), the proposed mining and boosting approach is able to achieve significantly improved accuracy (80.7%).

6. Relations to the Existing Methods

Shared features for multi-class recognition

In [17], decision stumps are shared by K classifiers during boosting. In our case, decision stumps are shared by compositional features, where each is a weak classifier for a specific class. Instead of building K binary classifiers by boosting decision stumps, we build a single K-class classifier by boosting compositional features.

Boosting decision trees:
Each compositional feature is essentially an internal node of a decision tree. However, instead of growing a whole tree through greedy search, we discover all important branches or tree nodes (of any possible depth) for boosting. We regularize the tree based on the quality of its leaf nodes as opposed to the tree depth.

Feature selection:
Instead of selecting a single collection of good features, we discover many feature collections (*i.e.* compositional features), where each can be a subset of good features. These feature collections are eventually fused through boosting. We do not employ greedy search in finding good features [10] [18] and we account for high-order dependencies among individual features.

Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.

7. Conclusions

We present a data mining-driven approach to discovering compositional features for multi-class boosting, which is successfully applied to a challenging 10-class event recognition problem. Compared with boosting decision stumps, we achieve faster error decreasing in training as well as better generalization in testing. We show how the data mining parameters (*i.e*. support λ_1 and confidence λ_2) influence the descriptive and discriminative abilities of the mined features $\mathcal{F}$, and obtain the upper bound of the training error. This gives us the theoretical guidance in selecting appropriate parameters for data mining. Finally, by applying the SAMME algorithm in boosting, our method naturally handles the multi-class problem by combining the mined binary compositional rules.

Acknowledgment

We thank Dr. Wei Hao for visual feature extraction, Shengyang Dai and Liangliang Cao for helpful discussions. This work was supported in part by National Science Foundation Grants IIS-0347877 and IIS-0308222.

Appendix

We prove theorem 1 here. Given a (λ_1, λ_2)-weak compositional rule $\mathcal{F} \in \mathbf{\Psi}$, we have $P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k) \geq \lambda_2 r_k$ and $P(\mathcal{F}(\mathbf{x}) = k) \geq \lambda_1$ according to Def. 2. Then we can upper bound the incorrect prediction of positive training samples:

$$\begin{aligned} & P(\mathcal{F}(\mathbf{x}) \neq k, c(\mathbf{x}) = k) \\ = \; & P(c(\mathbf{x}) = k) - P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k)P(\mathcal{F}(\mathbf{x}) = k) \\ \leq \; & r_k - \lambda_1\lambda_2 r_k. \end{aligned} \tag{16}$$

Furthermore, the incorrect prediction of negative samples can be upper bounded as:

$$\begin{aligned} & P(\mathcal{F}(\mathbf{x}) = k, c(\mathbf{x}) \neq k) \\ = \; & [1 - P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k)]\, P(\mathcal{F}(\mathbf{x}) = k) \\ \leq \; & (1 - \lambda_2 r_k)\frac{1}{\lambda_2} \qquad (17) \\ = \; & \frac{1}{\lambda_2} - r_k, \qquad (18) \end{aligned}$$

where we apply $P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k) \geq \lambda_2 r_k$ and $P(\mathcal{F}(\mathbf{x}) = k) \leq \frac{1}{\lambda_2}$ (Eq. 8) to derive Eq. 17. Finally, the error probability bound is

$$\begin{aligned} \epsilon_{\mathcal{F}} \; = \; & P(\mathcal{F}(\mathbf{x}) \neq k, c(\mathbf{x}) = k) + P(\mathcal{F}(\mathbf{x}) = k, c(\mathbf{x}) \neq k) \\ \leq \; & \frac{1}{\lambda_2} - \lambda_1\lambda_2 r_k. \end{aligned} \tag{19}$$

The above bound is tight, *i.e*., the equality of Eq. 19 holds, if both equalities hold in Eq. 16 and Eq. 17. It can be seen that if the equality holds for both conditions in Def. 2, *i.e*. $P(\mathcal{F}(\mathbf{x}) = k) = \lambda_1$ and $P(c(\mathbf{x}) = k|\mathcal{F}(\mathbf{x}) = k) = \lambda_2 P(c(\mathbf{x}) = k)$, the equality of Eq. 16 holds. Moreover, if $P(\mathcal{F}(\mathbf{x}) = k) = \frac{1}{\lambda_2}$, the equality of Eq. 17 holds. In such a case, we have $P(\mathcal{F}(\mathbf{x}) = k) = \lambda_1 = \frac{1}{\lambda_2}$, which requires $\lambda_1\lambda_2 = 1$.

References

[1] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Proc. IEEE Intl. Conf. on Computer Vision*, 2007.

[2] H. Cheng, X. Yan, J. Han, and C.-W. Hsu. Discriminative frequent pattern analysis for effective classification. In *Proc. of Intl. Conf. on Data Engineering*, 2007.

[3] P. Dollar, Z. Tu, H. Tao, and S. Belongie. Feature mining for image classification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

[4] J. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5), 2001.

[5] J. Friedman, T. Hastie, and R. Tibshirani. Addititive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:337–407, 2000.

[6] G. Grahne and J. Zhu. Fast algorithms for frequent itemset mining using fp-trees. *IEEE Transaction on Knowledge and Data Engineering*, 2005.

[7] J. Han, H. Cheng, D. Xin, and X. Yan. Frequent pattern mining: current status and future directions. In *Data Mining and Knowledge Discovery*, 2007.

[8] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2006.

[9] L.-J. Li and L. Fei-Fei. What, where and who? classifying events by scene and object recognition. In *Proc. IEEE Intl. Conf. on Computer Vision*, 2007.

[10] D. Lin and X. Tang. Conditional infomax learning: an integrated framework for feature extraction and fusion. In *Proc. European Conf. on Computer Vision*, 2006.

[11] C. Liu and H. Shum. Kullback-leibler boosting. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 587–594, 2003.

[12] T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

[13] S. Nowozin, K. Tsuda, T. Uno, T. Kudo, and G. Bakir. Weighted substructure mining for image analysis. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

[14] T. Quack, V. Ferrari, B. Leibe, and L. V. Gool. Efficient mining of frequent and distinctive feature configurations. In *Proc. IEEE Intl. Conf. on Computer Vision*, 2007.

[15] L. Reyzin and R. E. Schapire. How boosting the margin can also boost classifier complexity. In *Proc. Intl. Conf. on Machine Learning*, 2006.

[16] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2004.

[17] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(5):854–869, 2007.

[18] N. Vasconcelos and M. Vasconcelos. Scalable discriminant feature selection for image retrieval and recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2004.

[19] P. Viola and M. J. Jones. Robust real-time face detection. *Intl. Journal of Computer Vision*, 57(2):137–154, 2004.

[20] T.-F. Wu, G.-S. Xia, and S.-C. Zhu. Compositional boosting for computing hierarchical image structures. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

[21] J. Yuan, Y. Wu, and M. Yang. Discovery of collocation patterns: from visual words to visual phrases. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

[22] J. Zhu, S. Rosset, H. Zou, and T. Hastie. Multi-class adaboost. *Technique Report*, 2005.

Authorized licensed use limited to: Northwestern University. Downloaded on May 21, 2009 at 21:54 from IEEE Xplore. Restrictions apply.