T2R2 東京科学大学 リサーチリポジトリ
Science Tokyo Research Repository

論文 / 著書情報
Article / Book Information

| | |
|---|---|
| Title | A Fast and Accurate Video Semantic-Indexing System Using Fast MAP Adaptation and GMM Supervectors |
| Author | Nakamasa Inoue, Koichi Shinoda |
| Journal/Book name | IEEE Transactions on Multimedia, vol. 14, Issue: 4 Part 2, pp. 1196-1205 |
| Issue date | 2012, 8 |
| DOI | http://dx.doi.org/10.1109/TMM.2012.2191395 |
| URL | http://www.ieee.org/index.html |
| Copyright | (c)2012 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other users, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works for resale or redistribution to servers or lists, or reuse of any copyrighted components of this work in other works. |
| Note | このファイルは著者（最終）版です。<br>This file is author (final) version. |

Powered by T2R2 (Science Tokyo Research Repository)

# A Fast and Accurate Video Semantic Indexing System Using Fast MAP Adaptation and GMM Supervectors

Nakamasa Inoue, *Student Member, IEEE,* and Koichi Shinoda, *Senior Member, IEEE*

***Abstract*—We propose a fast maximum a posteriori (MAP) adaptation method for video semantic indexing that uses Gaussian mixture model (GMM) supervectors. In this method, a tree-structured GMM is utilzed to decrease the computational cost, where only the output probabilities of mixture components close to an input sample are precisely calculated. Experimental evaluation on the TRECVID 2010 dataset demonstrates the effectiveness of the proposed method. The calculation time of the MAP adaptation step is reduced by 76.2% compared to that of a conventional method. The total calculation time is reduced by 56.6% while keeping the same level of the accuracy.**

***Index Terms*—Video Semantic Indexing, GMM Supervectors, MAP Adaptation.**

## I. Introduction

RECENTLY, a large amount of video data has been made available. An effective video retrieval and searching system is demanded since it is often difficult to find relevant video manually. Although current keyword-based text search systems are sometimes useful for this purpose, they require metadata that describe the video contents. To automatically generate metadata, semantic indexing, i.e., assigning semantic concepts such as *airplane*, *bus*, *outside*, *nighttime*, and *singing* to video segments, is necessary. It has been a challenging task due to the semantic gap between the low-level features and high-level semantic concepts.

Most previous studies used statistical modeling to construct a model that describes the relationship between video and semantic concepts. In particular, statistical methods for generic object recognition in image processing have been effective for semantic indexing, since semantic indexing can be viewed as an extension of generic object recognition to video processing. The bag-of-visual-words (BoW) method [13], [14] that uses low-level features such as scale-invariant feature transform (SIFT) [15] is the most widely used method for generic object recognition. In this method, hard clustering, vector quantization (VQ), of features is used, but its quantization errors often degrade the indexing performance. Several soft clustering methods [20], [23] are introduced to solve this problem and have been proved to be effective. In particular, Gaussian mixture model (GMMs) are often preferred since it is a straightforward extension of VQ to a probabilistic framework.

In video semantic indexing, it is well-known that not only image features but also video-specific features such as audio features and movement features are important to improve the performance of video semantic indexing [2], [8], [9], [10]. In [2], we have shown the effectiveness of the combination of the audio hidden Markov model and the multi-frame feature extraction with GMM supervectors. Our method achieves higher accuracy than the best method of TRECVID 2010 semantic indexing [3], [4], [8]. However, video semantic indexing is more computationally expensive than image classification. In the testing phase of our method, 28.8%, 67.7%, and 3.5% of calculation time are spent for low-level feature extraction, GMM parameter estimation, and classification using SVM, respectively. Since the amount of video data to be processed is very large and rapidly increasing, it is strongly demanded to decrease these computational costs.

The speeded up robust features (SURF) [18] reduces the cost for low-level feature extraction by using integral images. The GPU implementation of SIFT [11], [12], and the BoW algorithm [7] have also been proposed. While these implementations are faster than a CPU implementation, the focus was not on the fast algorithm but on the fast implementation. In our method of GMM-supervector based semantic indexing, 67.7% of calculation time is spent for estimation of GMM parameter. Therefore, we aim at reducing its computational cost.

In this paper, we propose a fast maximum a posteriori (MAP) adaptation method using a tree-structured GMM to reduce the cost of the estimation of GMM parameters. Its basic idea is to calculate probabilities only for important Gaussian components and to skip the calculation for others. We consider a Gaussian component as important if it is located near the observed low-level feature. The tree structure is utilized to search the important Gaussian components. We expect that the performance of semantic indexing is not to be decreased by using the proposed method since probabilities for important Gaussian components are calculated precisely. We evaluated our system on the TRECVID 2010 video benchmark.

The rest of this paper is organized as follows. Previous studies are summarized in Section II. Our semantic indexing system based on the proposing fast MAP adaptation and GMM supervectors is described in Section III. Experimental results and conclusions are given in Section IV and Section V, respectively.

Copyright (c) 2010 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Manuscript received XXX;

N. Inoue and K. Shinoda are with the Department of Computer Science, Tokyo Institute of Technology, Tokyo, 152-8552 Japan (e-mail: inoue@ks.cs.titech.ac.jp; shinoda@cs.titech.ac.jp)

## II. Previous Studies

A basic approach for semantic indexing is the bag-of-visual-words approach [13], [14], which classifies videos by creating histograms of quantized low-level features (visual words). The bag-of-visual-words algorithm consists of three steps: 1) low-level feature extraction, 2) feature coding and 3) classification.

1) Low-level feature extraction
   SIFT [15] is the most widely used method for low-level feature extraction since it is robust against changes in scale, rotation, and illumination. However, the original SIFT only uses gray scale intensities. To capture color information, color descriptors such as hue histogram [16] and color SIFT [17] have been proposed and used in image classification.
   These low-level feature extraction methods usually consist of two phases: interest point detection and feature description. Therefore, approaches to fast feature extraction have focused on either of the two phases. The fast Hessian detector used in SURF [18] improves the speed of interest point detection by using integral images. With dense sampling [19], the phase of interest point detection can be skipped since grid-points are used as interest points. GPU implementations of SIFT [11], [12] and color SIFT [7] have been also proposed for feature description.
   To improve the accuracy of semantic indexing, the fusion of several types of low-level features has been used in recent studies. The combination of SIFT and color SIFT [8] have performed the best in semantic indexing at TRECVID [3], [4]. In [2], the multi-modal approach that uses SIFT and mel-frequency cepstral coefficients (MFCCs) improved the accuracy of semantic indexing. The MFCCs have first been proposed for speech recognition to describe the short-time spectral shape of audio frames.
2) Feature coding
   Vector quantization (VQ) techniques such as $k$-means clustering are typically used in this step. Each low-level feature is assigned to one of clusters with VQ in order to create histogram of low-level features. The soft-assignment of low-level features [20] has been proposed to reduce quantization error in VQ. Gaussian mixture models (GMMs) used in [23] can also be viewed as a soft-assignment clustering method. The GMMs usually perform better than the other clustering methods since it captures variances of low-level features for each cluster. Beyond the histogram-based method, the GMM supervector is first proposed as a speaker verification method [21], and then supervector coding [5] has been used for image classification. GMM supervector is made by concatenated the parameters of mixture components in a GMM and is used as instead of a histogram of low-level features. In these methods, an image (or an audio segment) is modeled by a GMM, i.e., a GMM parameter is estimated for each image.
   The Fisher vector [22], which describes the derivative of the gradient of the likelihood function, has been applied to image classification in [24]. The Fisher vector is equivalent to the GMM supervector if a GMM is used to compute likelihood in the generation of the Fisher vector as in [24].
   Although the GMM-based method outperforms the histogram-based bag-of-visual-words, it is computationally more expensive to estimate their parameters.
3) Classification
   The original bag-of-visual-words method [13], [14] used support vector machines (SVMs) for image classification. Multiple kernel learning (MKL) [25] enables to learn a linear combination of base kernels (e.g. kernels for different features). The weight coefficient for each base kernel is optimized during its training phase. Multiple kernel Fisher discriminant analysis (MK-FDA) [26] is an extension of the FDA to multiple kernels. These multiple kernel methods are computationally more expensive than the SVM. A linear combination of SVM scores still often performs well in terms of accuracy.

## III. Proposed Method

In this section, we describe our fast and accurate system for video semantic indexing, which uses fast MAP adaptation and GMM supervectors.

### A. Overview

The overview of our semantic indexing system is shown in Fig. 1. We assume videos are automatically segmented into shots in preprocessing. A shot consists of continuous frames without switching between cameras.

Our system consists of three parts. First, four types of low-level features (three types of SIFT features and MFCCs) are extracted from a video shot. The SIFT features are extracted by using three different interest point detectors: Harris-Affine [27], Hessian-Affine [27], and Dense [19]. The MFCCs, which describe the short-time spectral shape of audio frames, are extracted to capture audio information. The details of low-level feature extraction are described in Sec IV-B. Second, a GMM supervector is created for each type of features. A GMM models the distribution of low-level features extracted from a video shot. Its parameter is estimated by using MAP adaptation, in which a tree-structured GMM is utilized to reduce the computational cost for parameter estimation. Here, basic idea is to calculate probabilities only for *important* Gaussian components. We consider a Gaussian component as important if it is located near the observed low-level feature. The tree structure enables fast search of important Gaussian components. Third, the outputs of SVMs for the four feature types are fused to compute a final score.

### B. Gaussian Mixture Model

Let $X = \{x_i\}_{i=1}^{n}$ be a set of (one type of) low-level features extracted from a video shot. A probability distribution function (pdf) of a Gaussian mixture model (GMM) is given by

$$p(x|\theta) = \sum_{k=1}^{K} w_k \mathcal{N}(x|\mu_k, \Sigma_k), \tag{1}$$

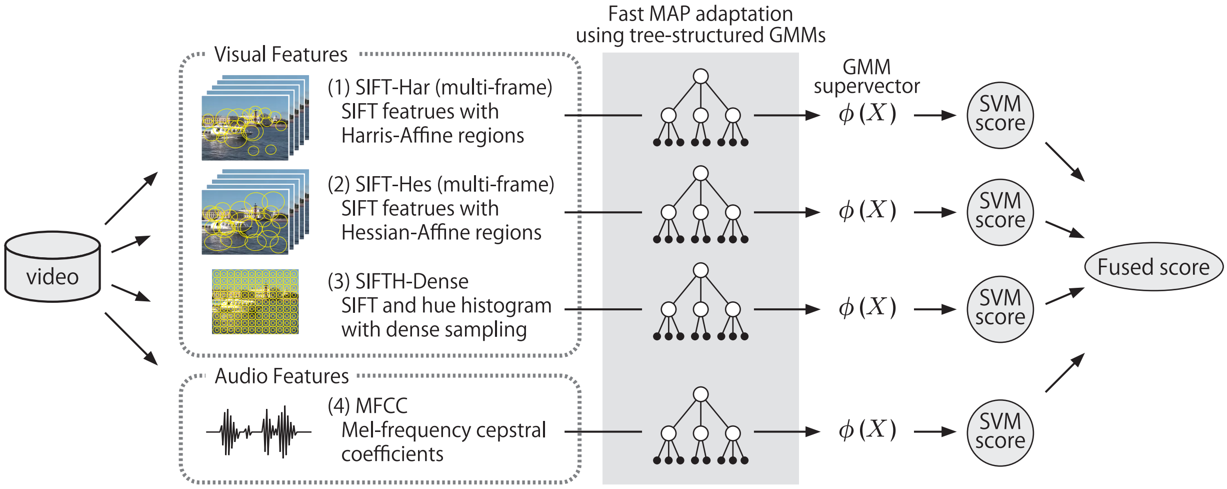
Fast MAP adaptation using tree-structured GMMs
Visual Features
(1) SIFT-Har (multi-frame) SIFT featrues with Harris-Affine regions
(2) SIFT-Hes (multi-frame) SIFT featrues with Hessian-Affine regions
(3) SIFTH-Dense SIFT and hue histogram with dense sampling
Audio Features
(4) MFCC Mel-frequency cepstral coefficients
video
GMM supervector
$\phi(X)$
SVM score
Fused score

Fig. 1. Overview of our semantic indexing system. Our system consists of three parts: 1) low-level feature extraction, 2) GMM supervector extraction by using fast MAP adaptation, and 3) SVM classification. First, four types of visual and audio features are extracted. Second, GMM parameters are estimated by using MAP adaptation. Tree-structured GMMs are used to improve the speed of MAP adaptation. Third, the outputs of SVMs for the four feature types are fused to compute a final score.

where $x$ is a low-level feature vector, $\theta = \{w_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ is a set of parameters, $K$ is the number of mixture components, $w_k$ is a mixture coefficient. $\mathcal{N}(\cdot|\mu_k, \Sigma_k)$ is a Gaussian pdf with a mean vector $\mu_k$ and a covariance matrix $\Sigma_k$.

The GMM parameters are often estimated by using an expectation maximization (EM) algorithm with the maximum likelihood criterion. However, a set of extracted feature vectors may not be enough to estimate the parameters precisely. In such cases, the alternative way is to use maximum a posteriori (MAP) adaptation. MAP adaptation, a parameter estimation using the MAP criterion, is robust against over-fitting caused by limited data since it uses a prior distribution. A GMM for prior distribution, namely a universal background model (UBM), is first estimated by applying the EM algorithm to all the training data. The UBM presents the feature distribution for the whole database.

In the proposed method, only mean vectors are adapted for each shot. The MAP solution gives the following equations:

$$\hat{\mu}_k = \frac{\tau \hat{\mu}_k^{(\mathrm{U})} + \sum_{i=1}^{n} c_{ik} x_i}{\tau + \sum_{i=1}^{n} c_{ik}}, \tag{2}$$

$$c_{ik} = \frac{w_k^{(\mathrm{U})} g_k^{(\mathrm{U})}(x_i)}{\sum_{k=1}^{K} w_k^{(\mathrm{U})} g_k^{(\mathrm{U})}(x_i)}, \tag{3}$$

$$g_k^{(\mathrm{U})}(x) = \mathcal{N}(x|\mu_k^{(\mathrm{U})}, \Sigma_k^{(\mathrm{U})}), \tag{4}$$

where $n$ is the number of feature vectors, and $\hat{\mu}_k^{(\mathrm{U})}$ is a mean vector of UBM, $g_k^{(\mathrm{U})}$ is a Gaussian component, $c_{ik}$ is a *responsibility* of a Gaussian component $g_k$ for a feature vector $x_i$, which is the posterior probability of $x_i$ being at $k$-th Gaussian component, and $\tau$ is a predefined hyper-parameter.

### C. Tree-structured GMMs

A tree structure of Gaussian components that makes calculation of Eq. (3) efficient is constructed from the UBM. Fig. 2 shows an example of a tree-structured GMM. Each leaf node corresponds to a Gaussian component of the UBM, and each other node has a single Gaussian obtained by combining corresponding Gaussian pdfs of descendant nodes. This tree structure is constructed by top-down clustering of Gaussian components. For a given set of Gaussian components $G = \{g_1, g_2, \cdots, g_K\}$, we define a combined single Gaussian $\mathcal{G}(G)$ by

$$\mathcal{G}(G) \overset{\mathrm{def}}{=} \mathcal{N}(\cdot|\bar{\mu}, \bar{\Sigma}), \tag{5}$$

$$\bar{\mu} = \frac{1}{K}\sum_{k=1}^{K} \mu_k, \tag{6}$$

$$\bar{\Sigma} = \frac{1}{K}\sum_{k=1}^{K} (\Sigma_k + \mu_k \mu_k^{\mathsf{T}}) - \bar{\mu}\bar{\mu}^{\mathsf{T}}. \tag{7}$$

To find a pair of Gaussian components which are close to each other, a distance measure between them is needed. The sum of Kullback-Leibler divergence from $g_k$ to $g_{k'}$ and that of from $g_{k'}$ to $g_k$ is used for this measurement as follows:

$$d(g_k, g_{k'}) = \int g_k(x) \log \frac{g_k(x)}{g_{k'}(x)} dx + \int g_{k'}(x) \log \frac{g_{k'}(x)}{g_k(x)} dx \tag{8}$$

$$= \frac{1}{2}\Big( (\mu_k - \mu_{k'})^{\mathsf{T}} (\Sigma_k^{-1} + \Sigma_{k'}^{-1})(\mu_k - \mu_{k'}) + \mathrm{tr}(\Sigma_{k'}^{-1}\Sigma_k) + \mathrm{tr}(\Sigma_k^{-1}\Sigma_{k'}) - 2d \Big). \tag{9}$$

As for the following tree-construction algorithm, it is assumed that the maximum number of child nodes for each layer (with the exception of the leaf layer) is given. For example, if the maximum number of child nodes for the first layer (which only has a root node) is two and that for the second layer is three, the resulting tree will be designed as shown in Fig. 2. In this case, a tree with a depth of three (including the leaf layer) is obtained. This tree-structured GMM is denoted as

$$\mathcal{T}_{(2,3)} = (V, E, G_{\mathrm{TREE}}), \tag{10}$$

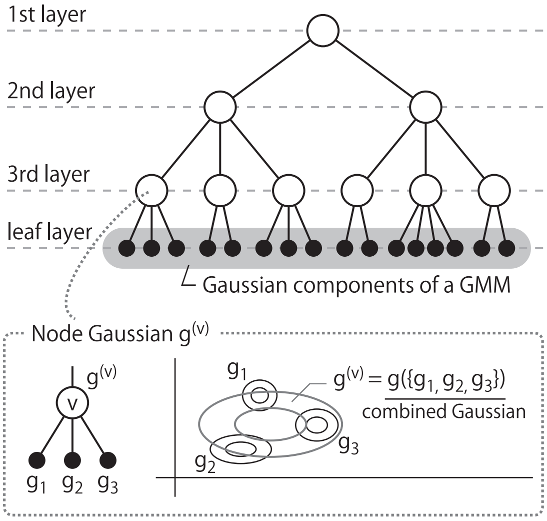

Fig. 2. An example of a tree-structured GMM $\mathcal{T}_{(2,3)}$.

where $V$ is a set of nodes, $E$ is a set of edges, and $G_{\text{TREE}} = \{g^{(v)} | v \in V\}$ is a set of Gaussian pdfs. In general, a node at the $t$-th layer of a tree $\mathcal{T}_{(P_1, P_2, \cdots, P_T)}$ has, at most, $P_t$ child nodes.

The node pdfs $g^{(v)}$ of a tree $\mathcal{T}_{(P_1, P_2, \cdots, P_T)}$ are created by the following algorithm. The basic idea is to apply hierarchical $k$-means clustering for Gaussian components. Note that $G_{\text{UBM}} = \{g_1^{(\text{U})}, g_2^{(\text{U})}, \cdots, g_K^{(\text{U})}\}$ is a set of mixture components of the UBM, $G^{(v)}$ is a subset of $G_{\text{UBM}}$ corresponding to node $v$, and $g^{(v)}$ is a Gaussian pdf for node $v$.

1) Prepare a queue and enqueue $(r, G^{(r)})$, where $r$ is the root node, $G^{(r)} = G_{\text{UBM}}$, and $g^{(r)} \leftarrow \mathcal{G}(G^{(r)})$.
2) (Initialization for $k$-means) Dequeue $(v, G^{(v)})$. Let $\{c_p\}_{p=1}^{P}$ be the child nodes of node $v$. Select the initial cluster centers $g^{(c_p)}$ from the given set of Gaussian components $G^{(v)}$, where the min-max selection method is used instead of random selection. The min-max selection method is explained in Appendix.
3) (Clustering by $k$-means) Assign each Gaussian component in $G^{(v)}$ to the nearest child node, i.e.

$$G^{(c_p)} \leftarrow \{g \in G^{(v)} \mid p = \operatorname*{argmin}_{p'} d(g, g^{(c_{p'})})\}. \tag{11}$$

Update $g^{(c_p)}$ by using Eq. (5) as follows:

$$g^{(c_p)} \leftarrow \mathcal{G}(G^{(c_p)}), \tag{12}$$

Repeat this step until the following sum of distance converges:

$$D = \sum_{p=1}^{P} \sum_{g \in G^{(c_p)}} d(g, g^{(c_p)}). \tag{13}$$

4) For $p = 1, 2, \cdots, P$, enqueue $(c_p, G^{(c_p)})$ if $c_p$ is not in the $(T+1)$-th layer and $|G^{(c_p)}| > 1$. Go to step 5 if the queue is empty; otherwise, return to step 2.
5) For each node $v$ in the $(T+1)$-th layer, create leaf nodes $\ell$ for each $g_k^{(\text{U})} \in G^{(v)} \subset G_{\text{UBM}}$ and set

$$g^{(\ell)} = g_k^{(\text{U})}. \tag{14}$$

### D. Fast MAP Adaptation

A fast MAP adaptation technique which estimates $c_{ik}$ in Eq. (3) efficiently by using a tree-structured GMM is explained in the following. For a constructed tree-structured GMM $\mathcal{T}_{(P_1, P_2, \cdots, P_T)}$, node weights are first defined as follows:

a) For each leaf node $\ell$, set

$$w^{(\ell)} = w_k^{(\text{U})}, \tag{15}$$

if $g^{(\ell)} = g_k^{(\text{U})} \in G_{\text{UBM}}$.

b) Calculate weights for $t = T+1, T, \cdots, 1$ as follows. For each node $v$ in the $t$-th layer,

$$w^{(v)} = \sum_{c \in C(v)} w^{(c)}, \tag{16}$$

where $C(v)$ is a set of child nodes of the node $v$.

The algorithm for estimating $c_{ik}$ for feature vector $x_i$ is described as follows. The key idea is to construct a GMM of active nodes $V_A$, which means vector $x_i$ will be assigned to descendants of nodes in $V_A$. $|V_A|$ is kept small by obtaining active nodes from the root node.

1) Set $V_A \leftarrow \{r\}$, where $r$ is the root node.
2) Expand active nodes by making child nodes of the active nodes active:

$$V_A \leftarrow \bigcup_{v \in V_A} C(v), \tag{17}$$

where $C(v)$ is a set of child nodes of the node $v$. Here, $C(\ell) = \{\ell\}$ is used for leaf nodes $\ell$ to keep the leaf nodes active.

3) Consider a GMM for active node $V_A$ given by

$$p(x|V_A) = \sum_{v \in V_A} \tilde{w}^{(v)} g^{(v)}(x), \tag{18}$$

where

$$\tilde{w}^{(v)} = \frac{w^{(v)}}{\sum_{v \in V_A} w^{(v)}}. \tag{19}$$

Calculate

$$c_i^{(v)} = \frac{\tilde{w}^{(v)} g^{(v)}(x_i)}{\sum_{v \in V_A} \tilde{w}^{(v)} g^{(v)}(x_i)} = \frac{w^{(v)} g^{(v)}(x_i)}{\sum_{v \in V_A} w^{(v)} g^{(v)}(x_i)}. \tag{20}$$

4) Keep a node $v$ active if $c_i^{(v)}$ is larger than the predetermined threshold $c_{\text{TH}}$, i.e.

$$V_A \leftarrow \{v \in V_A \mid c_i^{(v)} > c_{\text{TH}}\}. \tag{21}$$

5) If all nodes in $V_A$ are leaf nodes, output

$$\hat{c}_{ik} = \begin{cases} c_i^{(\ell)} & (\ell \in V_A,\ g^{(\ell)} = g_k^{(\text{U})}) \\ 0 & (\text{otherwise}) \end{cases}. \tag{22}$$

Otherwise, return to Step 2.

Since the observed $\hat{c}_{ik}$ for non-active nodes is 0, the sum in Eq. (2) can be calculated for non-zero $\hat{c}_{ik}$ only. Our fast MAP adaptation requires $O(n \log K)$ time which improves $O(nK)$ time for the basic MAP adaptation. Moreover, calculation speed and levels of approximation can be controlled by selecting the threshold $c_{\text{TH}}$ in $0 < c_{\text{TH}} \leq 1$. Note that the same $c_{ik}$ as given by Eq. (3) is obtained when $c_{\text{TH}}$ is set to 0 (because all leaf nodes will be active at the final step).

### E. GMM Supervector SVM

The combination of GMM supervectors and support vector machines (SVMs) was first proposed as a speaker recognition method [21] and has been applied to image and video recognition [2], [6]. GMM supervectors are created for each shot and are given by

$$\phi(X) = \begin{pmatrix} \tilde{\mu}_1 \\ \tilde{\mu}_2 \\ \vdots \\ \tilde{\mu}_K \end{pmatrix}, \ \tilde{\mu}_k = \sqrt{w_k^{(\mathrm{U})}} \left( \Sigma_k^{(\mathrm{U})} \right)^{-\frac{1}{2}} \hat{\mu}_k, \qquad (23)$$

where $\hat{\mu}_k$ is an adapted mean vector obtained from Eq. (2), and $\theta^{(\mathrm{U})}$ is the GMM parameter for the UBM. The dimension of GMM supervectors is $Kd$, where $K$ is the number of Gaussian components and $d$ is the dimension of the low-level feature vector. The weighted sum of Mahalanobis distances between corresponding Gaussian pairs is obtained by calculating the squared Euclidean distance between two GMM supervectors. RBF kernels are used for SVM classification:

$$k(X, X') = e^{-\gamma \|\phi(X) - \phi(X')\|^2}, \qquad (24)$$

where $\gamma$ is a kernel parameter. An SVM is trained for each semantic concept and for each type of features. Note that the proposed method can be used for creating the Fisher vectors [22], [24].

### F. Score Fusion

SVMs for the four types of features described in Section IV-B are combined by calculating the weighted sum of detection scores as

$$f(X) = \sum_{\substack{\mathrm{F} \in \{\text{SIFT-Har, SIFT-Hes,} \\ \text{SIFTH-D, MFCC}\}}} \beta_{\mathrm{F}} f_{\mathrm{F}}(X), \qquad (25)$$

where $\beta_{\mathrm{F}}$ is a combination weight and $f_{\mathrm{F}}(X)$ is a detection score for the feature F. The combination weights are optimized for each semantic concept by cross validation.

## IV. Experiments

### A. Database and Task

Our experiments were conducted on the TRECVID 2010 dataset [3], [4]. The dataset consists of 400 hours of Internet archive videos with creative commons licenses. The shot boundaries and key-frame images are automatically detected and provided with the video data. Half of the videos were used for training, and the others were used for testing. The number of shots was 119,685 for training and 146,788 for testing.

We evaluated our system on the TRECVID 2010 Semantic Indexing benchmark. The task is to detect the 30 semantic concepts (including objects, events and scenes) listed in Table I. They are considered useful for video searching.

TABLE I
The 30 target semantic concepts in the TRECVID 2010 dataset

| | |
|---|---|
| Airplane Flying | Female Human Face Closeup |
| Animal | Flowers |
| Asian People | Ground Vehicles |
| Bicycling | Hand |
| Boat ship | Mountain |
| Bus | Nighttime |
| Car Racing | Old People |
| Cheering | Running |
| Cityscape | Singing |
| Classroom | Sitting down |
| Dancing | Swimming |
| Dark-skinned People | Telephones |
| Demonstration Or Protest | Throwing |
| Doorway | Vehicle |
| Explosion Fire | Walking |

The annotated labels for the 30 concepts are also provided with the video data (including testing videos for evaluation). The labels for training data are created using a collaborative annotation system [28]; however, some of the training shots are still unlabeled. It was assumed that the unlabeled samples are negative since the annotation system is based on an active learning method that requires shots appearing to be positive samples to be annotated preferentially. On the other hand, labels for testing videos are attached on the basis of the submission pool of TRECVID 2010, which allows precise estimation of average precision.

The evaluation measures are Mean average precision (Mean AP) and the calculation time of the testing phase. Mean AP is defined as the mean of APs over all 30 target concepts. APs are calculated as

$$\mathrm{AP} = \frac{1}{R} \sum_{r=1}^{N} \mathrm{Pr}(r) \mathrm{Rel}(r), \qquad (26)$$

where $R$ is the number of positive samples, $N$ is the number of testing samples, $\mathrm{Pr}(r)$ is the precision at the rank of $r$ and $\mathrm{Rel}(r)$ takes a value of one if the $r$-th shot is positive; otherwise, it takes zero. The AP is estimated by using a method called inferred average precision (Inf AP), proposed in [31].

### B. Experimental Conditions

The following four types of visual and audio features are extracted from video data.

1) SIFT features with Harris-Affine detector (SIFT-Har)
Scale-invariant feature transform (SIFT) [15] is a low-level feature extraction method that is widely used for object categorization. The extracted features are invariant to image scaling and changing illumination. The Harris-Affine local region detector [27], which is an extension of the Harris corner detector, provides affine-invariant local regions. The proposed method uses 32-dimension SIFT features, whose dimensions are reduced from 128 to 32 by applying principal component analysis (PCA). The SIFT features are extracted from every other frame.
2) SIFT features with Hessian-Affine detector (SIFT-Hes)
SIFT features are extracted with a Hessian-Affine detector [27], which is complementary to the Harris-Affine detector. The combination of several different detectors can improve the robustness against noise. Features are extracted from every other frame, and PCA is applied to reduce their dimensions from 128 to 32.

TABLE II
THE AVERAGE NUMBERS OF EXTRACTED FEATURES.

| Feature | # of features per shot |
|---|---|
| SIFT-Har | 19,536 |
| SIFT-Hes | 18,986 |
| SIFTH-Dense | 30,000 |
| MFCC | 5,160 |

3) SIFT and hue histogram with dense sampling (SIFTH-Dense)
To capture color information, SIFT features and hue histograms [16] are combined. As a result, 164 dimensional low-level features (which consist of 128-dimension SIFT features and 36-dimension hue histograms) are obtained. PCA is also used to reduce the dimensions to 32. This feature is extracted only from key frames by using dense sampling, which provides a much larger number of low-level features than sparse sampling such as the Harris-Affine and Hessian-Affine detectors.

4) MFCC audio features (MFCC)
Mel-frequency cepstral coefficients (MFCCs), which describe the short-time spectral shape of audio frames, are extracted to capture audio information. Semantic concepts related to people speaking, talking, and singing can be detected by using MFCCs since MFCCs are effective for speech recognition and audio classification. The 38-dimension audio features consist of 12-dimension MFCCs, 12-dimension $\Delta$ MFCCs, 12-dimension $\Delta\Delta$ MFCCs, 1-dimension $\Delta$ log-power and 1-dimension $\Delta\Delta$ log-power are extracted. Here, “$\Delta$” means the derivative of the feature.

SiftGPU [12] and Mikolajczyk's implementation [27] were used for SIFT feature extraction. MFCC features were extracted by using a speech recognition toolkit HTK [29]. The average numbers of features per shot are summarized in Table II.

The number of mixtures (vocabulary size) $K$ for GMMs was 512 for visual features and 256 for audio features. For computational efficiency, it was assumed that covariance matrices are diagonal. Hyper parameter $\tau$ in the MAP adaptation was set to 20.0, which is the standard value of the toolkit HTK. Parameter $\gamma$ in the RBF kernel was $\gamma = \bar{d}^{-1}$, where $\bar{d}$ is pre-calculated average distance between two GMM supervectors in training data. SVMs were trained for each semantic concept by using the libSVM implementation [30]. Combination weights for the fusion in Eq. (25) were optimized by using two-fold cross validation on training data.

For tree-structured GMMs, the optimal tree structure $\mathcal{T}_{\text{opt}}$ was selected as

$$\mathcal{T}_{\text{opt}} = \operatorname*{argmin}_{\mathcal{T} \in \mathcal{S}} \mathrm{CT}(\mathcal{T}), \tag{27}$$

where $\mathrm{CT}(\mathcal{T})$ is calculation time when the tree $\mathcal{T}$ is used and

$$\mathcal{S} = \{\mathcal{T}_{(P_1,P_2,\cdots,P_T)} \mid 1 \leq T \leq 5, 1 \leq P_t \leq 5\}. \tag{28}$$

The trees $\mathcal{T}_{(4,4,5)}$, $\mathcal{T}_{(4,5,5)}$, $\mathcal{T}_{(3,4,4,5)}$ and $\mathcal{T}_{(3,3)}$ were selected for SIFT-Har, SIFT-Hes, SIFTH-Dense and MFCC, respectively. Parameter $\alpha$ in Eq. (32) was fixed to 0.1.

Threshold $c_{\text{TH}}$ for the fast MAP adaptation was set to 0.001. Here, a low threshold was set so as to keep detection performance high. Experiments using different thresholds were also conducted (see Subsection IV-C4).

In the experiments, calculation time was measured by using a single core of Intel Xeon 2.93 GHz CPU. Calculation time without feature extraction time is reported since some features were pre-extracted by using GPUs. The average feature extraction time per shot was 0.38 sec by using a GPU NVIDIA Tesla M2050.

### C. Results

*1) Mean Inf APs:* Table III summarizes obtained Inf APs and Mean Inf AP for each types of low-level features and two fusion methods: visual fusion and multi-modal fusion. The visual fusion is a combination of three types of visual features (SIFT-Har, SIFT-Hes, and SIFTH-Dense). The multi-modal fusion combines the MFCC in addition to the visual features. As a result, we can see that the Mean Inf APs using tree-structured GMMs are comparable to those using no trees. Some example video shots for training and testing sets are shown in Fig. 3.

*2) Calculation time:* Table IV lists calculation times for MAP adaptation using different features and different trees. The results for binary trees ($\mathcal{T}_{\text{binary}}$) are also listed in the table. The calculation speed when the optimal tree is used on average 4.2 times faster than when trees were not used; that is, calculation time was reduced by 76.2%.

Fig. 4 shows calculation time for each step in the testing phase of the proposed semantic indexing system. The testing cost was reduced, on average, by 56.6% by using tree-structured GMMs. The second and third highest costs were for the PCA projection and the SVM prediction (including calculation of kernels). The SVM prediction can be speed up by using linear kernels instead of RBF kernels. To avoid decrease in detection accuracy, a possible compromise is to use linear kernels for roughly ranking shots and re-evaluate high-ranked shots by using RBF kernels.

*3) Analysis of estimation error:* Estimation errors of $c_{ik}$ were evaluated from the mean absolute error (MAE), given as follows:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} |\hat{c}_{ik} - c_{ik}|, \tag{29}$$

where $\hat{c}_{ik}$ and $c_{ik}$ are given by Eq. (22) and Eq. (3), respectively. The MAE for SIFTH-Dense was 0.32 on average (note that $0 \leq \mathrm{MAE} \leq 2$). Although we have estimation errors of $c_{ik}$ in the fast MAP adaptation algorithm, they can be cancelled when the distance in Eq. (24) is calculated since the same errors occur in training and testing phases.

*4) Effect of using different thresholds:* Table V lists the results obtained using different thresholds $c_{\text{TH}}$ for the fast MAP adaptation. The number of leaf nodes that are active (at least once in Eq. (17) ) and MAE are also listed in the table.

As $c_{\text{TH}}$ gets higher, the calculation time shortens, but Mean Inf AP was decreased when $c_{\text{TH}} = 0.1$ and 0.5. Moreover, the number of active leaf nodes decreases, and MAE increases.

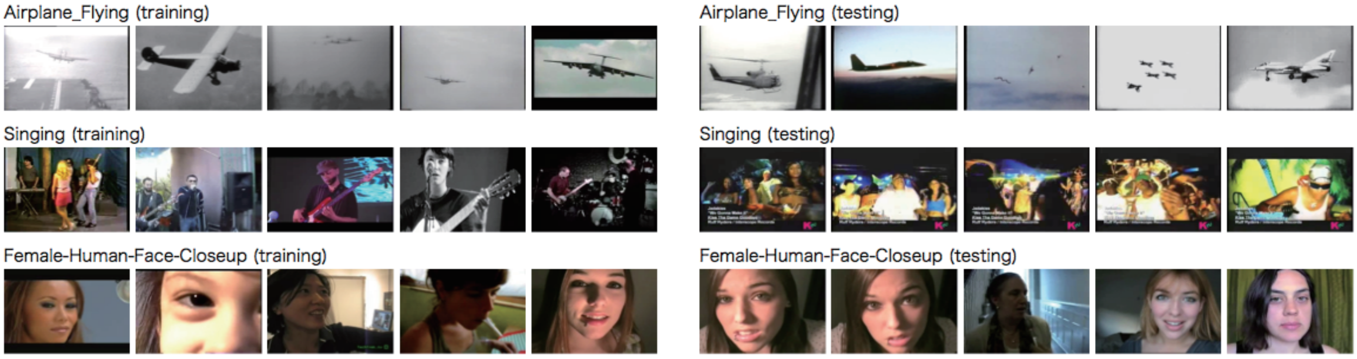
Airplane_Flying (training)
Airplane_Flying (testing)
Singing (training)
Singing (testing)
Female-Human-Face-Closeup (training)
Female-Human-Face-Closeup (testing)

Fig. 3. Example video shots for training and testing sets. The top 5 results obtained by using our system (multi-modal fusion) are shown in the right side of the figure.

TABLE III
RESULTING INFERRED AVERAGE PRECISIONS (INF APS) FOR EACH SEMANTIC CONCEPT AND FOR EACH METHOD. MEAN INF APS ON THE TESTING SET AND MEAN APS ON A TWO-FOLD CROSS-VALIDATION SPLIT OF THE TRAINING DATA ARE ALSO SHOWN.

| Semantic concept | SIFT-Har | | SIFT-Hes | | SIFTH-Dense | | MFCC | | Visual fusion | | Multi-modal fusion | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | No tree | $\mathcal{T}_{\text{opt}}$ | No tree | $\mathcal{T}_{\text{opt}}$ | No tree | $\mathcal{T}_{\text{opt}}$ | No tree | $\mathcal{T}_{\text{opt}}$ | No tree | $\mathcal{T}_{\text{opt}}$ | No tree | $\mathcal{T}_{\text{opt}}$ |
| Airplane_Flying | 0.064 | 0.064 | 0.080 | 0.078 | 0.032 | 0.030 | 0.001 | 0.001 | 0.105 | 0.105 | 0.105 | **0.117** |
| Animal | 0.039 | 0.041 | 0.034 | 0.035 | 0.026 | 0.020 | 0.002 | 0.002 | 0.068 | 0.073 | **0.076** | **0.076** |
| Asian_People | 0.024 | 0.029 | 0.014 | 0.015 | 0.001 | 0.002 | **0.041** | **0.041** | 0.012 | 0.009 | 0.012 | 0.009 |
| Bicycling | 0.039 | 0.041 | 0.040 | 0.033 | 0.029 | 0.026 | 0.000 | 0.000 | 0.045 | **0.056** | 0.045 | **0.056** |
| Boat_Ship | 0.046 | 0.044 | 0.040 | 0.041 | 0.050 | 0.049 | 0.000 | 0.000 | **0.085** | 0.084 | **0.085** | 0.084 |
| Bus | 0.012 | 0.012 | 0.011 | 0.013 | 0.007 | 0.009 | 0.000 | 0.000 | 0.018 | 0.016 | **0.021** | 0.016 |
| Car_Racing | 0.021 | 0.019 | 0.014 | 0.013 | **0.060** | 0.054 | 0.000 | 0.000 | 0.040 | 0.040 | 0.056 | 0.043 |
| Cheering | **0.053** | 0.051 | 0.044 | 0.045 | 0.033 | 0.037 | 0.008 | 0.008 | 0.052 | 0.051 | 0.052 | 0.051 |
| Cityscape | 0.090 | 0.098 | 0.109 | 0.110 | 0.125 | 0.108 | 0.009 | 0.009 | 0.180 | 0.177 | **0.185** | 0.179 |
| Classroom | 0.004 | 0.005 | 0.020 | **0.022** | 0.010 | 0.010 | 0.000 | 0.000 | 0.015 | 0.011 | 0.017 | 0.021 |
| Dancing | 0.034 | 0.036 | 0.030 | 0.028 | 0.034 | 0.033 | 0.001 | 0.001 | **0.068** | 0.067 | **0.068** | 0.067 |
| Dark-skinned_People | 0.089 | 0.088 | 0.073 | 0.071 | 0.118 | 0.133 | 0.138 | 0.139 | 0.151 | 0.159 | **0.208** | 0.203 |
| Demonstration_Or_Protest | 0.095 | 0.095 | 0.065 | 0.069 | 0.130 | 0.121 | 0.001 | 0.001 | **0.137** | 0.132 | **0.137** | 0.132 |
| Doorway | 0.084 | 0.082 | 0.068 | 0.067 | 0.073 | 0.068 | 0.000 | 0.001 | 0.098 | 0.097 | **0.104** | 0.098 |
| Explosion_Fire | 0.025 | 0.025 | 0.026 | 0.025 | 0.045 | 0.043 | 0.011 | 0.011 | **0.050** | 0.047 | **0.050** | 0.047 |
| Female-Human-Face-Closeup | 0.139 | 0.124 | 0.096 | 0.105 | 0.125 | 0.121 | 0.021 | 0.021 | 0.169 | 0.175 | 0.173 | **0.178** |
| Flowers | 0.030 | 0.028 | 0.019 | 0.017 | 0.029 | 0.028 | 0.001 | 0.001 | 0.043 | **0.044** | 0.043 | **0.044** |
| Ground_Vehicles | 0.159 | 0.165 | 0.148 | 0.150 | 0.153 | 0.151 | 0.021 | 0.020 | **0.211** | 0.210 | 0.208 | 0.206 |
| Hand | 0.078 | 0.073 | 0.062 | 0.073 | 0.047 | 0.055 | 0.000 | 0.000 | **0.092** | 0.089 | 0.090 | 0.090 |
| Mountain | 0.059 | 0.055 | 0.053 | 0.054 | 0.192 | **0.194** | 0.003 | 0.003 | 0.180 | 0.169 | 0.182 | 0.164 |
| Nighttime | 0.072 | 0.073 | 0.055 | 0.054 | 0.120 | 0.113 | 0.002 | 0.002 | 0.127 | **0.133** | 0.120 | 0.132 |
| Old_People | 0.043 | 0.045 | 0.040 | 0.041 | 0.022 | 0.023 | 0.013 | 0.011 | 0.059 | 0.058 | 0.061 | **0.063** |
| Running | 0.039 | 0.041 | 0.047 | 0.045 | 0.020 | 0.018 | 0.000 | 0.000 | 0.073 | **0.077** | 0.073 | **0.077** |
| Singing | 0.112 | 0.105 | 0.069 | 0.074 | 0.069 | 0.068 | 0.086 | 0.090 | 0.154 | 0.158 | 0.182 | **0.188** |
| Sitting_Down | 0.002 | 0.001 | 0.001 | 0.001 | 0.001 | 0.001 | 0.000 | 0.000 | 0.002 | 0.003 | 0.003 | **0.004** |
| Swimming | 0.121 | 0.131 | 0.162 | 0.164 | **0.343** | 0.339 | 0.199 | 0.199 | 0.173 | 0.186 | 0.278 | 0.276 |
| Telephones | 0.008 | 0.010 | 0.006 | 0.006 | 0.008 | 0.009 | 0.001 | 0.000 | 0.010 | 0.012 | 0.010 | **0.018** |
| Throwing | 0.059 | 0.059 | 0.063 | 0.063 | 0.019 | 0.016 | 0.019 | 0.020 | 0.062 | 0.065 | 0.062 | **0.066** |
| Vehicle | 0.158 | 0.150 | 0.163 | 0.172 | 0.150 | 0.146 | 0.015 | 0.014 | **0.205** | 0.200 | **0.205** | 0.200 |
| Walking | 0.093 | 0.103 | 0.135 | 0.138 | 0.061 | 0.060 | 0.002 | 0.002 | 0.134 | 0.142 | 0.135 | **0.143** |
| Mean InfAP | 0.063 | 0.063 | 0.060 | 0.061 | 0.071 | 0.070 | 0.020 | 0.020 | 0.094 | 0.095 | **0.102** | **0.102** |
| Mean AP on validation set 1 | 0.078 | 0.078 | 0.081 | 0.082 | 0.105 | 0.107 | 0.028 | 0.028 | 0.147 | 0.148 | 0.153 | **0.154** |
| Mean AP on validation set 2 | 0.084 | 0.085 | 0.092 | 0.091 | 0.111 | 0.111 | 0.028 | 0.027 | 0.158 | 0.158 | **0.162** | 0.161 |

It can thus be concluded that calculation time should be reduced not by setting a high threshold $c_{\text{TH}}$ but by selecting a better-structured tree to keep detection performance high. In particular, $c_{\text{TH}}$ should be equal to or smaller than 0.01.

*5) Effect of using different tree structures:* Fig. 5 shows calculation time obtained using different tree structures. The tree of $\mathcal{T}_{(3,4,4,5)}$ was the best in terms of calculation time. We can see that the tree should not be too deep to improve the speed of MAP adaptation. Fig. 6 shows MAE obtained using different tree structures. MAE can be reduced by changing the tree structure. However, we conclude that any tree structures will not be the cause for decreasing final performance since there was no decrease in Mean Inf AP even in the case of MAE $= 0.53$ in Table V.

*6) Comparison With Other Methods:* Fig. 7 compares Mean Inf APs obtained in the above-described experiment with those values obtained by the other methods used at TRECVID 2010. Our fusion methods got better results than the best result reported at TRECVID 2010 (0.900).

Fig. 8 shows the results of significance test obtained by

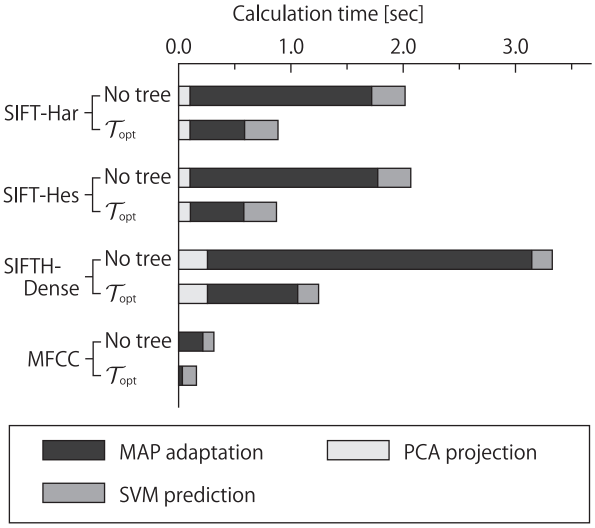

Fig. 4. Calculation time for each step (The lower bars for each feature show the time in the case that the optimized tree was used)

TABLE IV
CALCULATION TIME (SEC) FOR MAP ADAPTATION. CALCULATION TIME WAS MEASURED BY USING A SINGLE CORE OF INTEL XEON 2.93 GHZ CPU.

| Feature | No tree | $\mathcal{T}_{\text{opt}}$ | $\mathcal{T}_{\text{binary}}$ |
|---|---|---|---|
| SIFT-Har | 1.62 | 0.49 | 0.98 |
| SIFT-Hes | 1.67 | 0.48 | 1.00 |
| SIFTH-Dense | 2.89 | 0.81 | 1.89 |
| MFCC | 0.22 | 0.03 | 0.08 |

applying partial randomization test ($p < 0.05$). The muti-modal fusion was significantly better than the visual fusion. Our method performed better than the other methods in TRECVID 2010 for semantic concepts related to human and human actions such as "Singing" and "Dancing" since we used audio features. However, there was no significant difference between the muti-modal fusion and the top result in TRECVID 2010. This result shows that the performance can be improved by combining a larger number of visual features since the top ranked methods in TRECVID 2010 used more than 10 types of visual features.

Although our final goal is to develop a generic methods for automatically assigning semantic concepts to videos, overall performances are still low compared with that of human annotation. One future challenge is detection of many kinds of semantic concepts; however, we have to consider which concepts are really useful for applications of video search.

## V. CONCLUSION

A fast and accurate semantic indexing system using fast MAP adaptation and GMM supervectors was proposed. A tree-structured GMM was constructed to quickly calculate posterior probabilities for each mixture component of a GMM. The calculation time for MAP adaptation was reduced by 76.2% from the conventional method, while high detection performance was maintained. Our future work will focus on a GPU implementation of the fast MAP adaptation and feature extraction.

TABLE V
COMPARISON OF MEAN INF AP, CALCULATION TIME (SEC) FOR MAP ADAPTATION, NUMBER OF LEAF NODES $|V_A|$ AND MEAN ABSOLUTE ERROR (MAE) OF $c_{ik}$ BY USING DIFFERENT THRESHOLDS $c_{\text{TH}}$ FOR THE SIFTH-DENSE FEATURE.

| $c_{\text{TH}}$ | Mean Inf AP | Calc. time | $\|V_A\|$ | MAE |
|---|---|---|---|---|
| 0.001 | 0.695 | 0.81 | 17.0 | 0.32 |
| 0.01 | 0.699 | 0.68 | 11.2 | 0.53 |
| 0.1 | 0.660 | 0.59 | 7.3 | 0.80 |
| 0.5 | 0.641 | 0.53 | 5.4 | 0.98 |

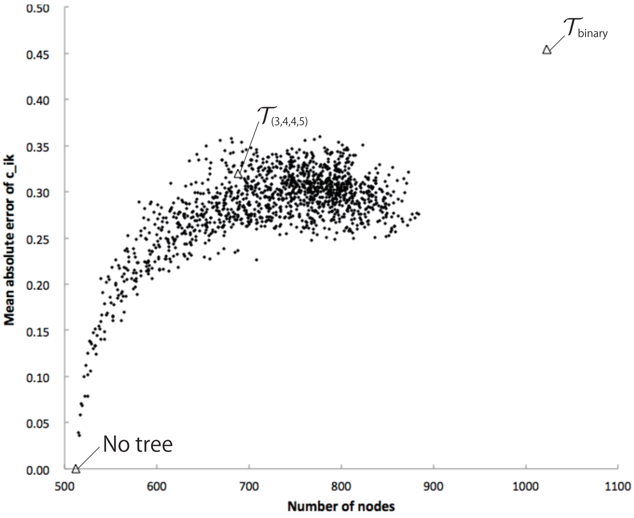

Fig. 5. Mean absolute error (MAE) of $c_{ik}$ obtained using different tree structures (the SIFTH-Dense feature and $c_{\text{TH}} = 0.001$ were used). 1,364 trees of depth at most 5 that have at most 5 children per node and the binary tree are tested. All MAE were less than 0.05.

## APPENDIX

For the initialization for $k$-means clustering (Step 2 in the tree-construction algorithm in Sec III-C),we use the min-max selection method. This method is known to provides better initial values than random selection. This method first selects from $G^{(v)}$ a node set whose nodes are distant from each other, and then sets a cluster center at an internal dividing point between node $v$ and each of the selected nodes.

2-1) Choose the mixture component $\tilde{g}^{(c_1)}$ that has the largest distance to $g^{(v)}$, i.e.,

$$\tilde{g}^{(c_1)} = \underset{g \in G^{(v)}}{\operatorname{argmax}} \quad d(g, g^{(v)}). \tag{30}$$

2-2) For $p = 2, \cdots, P$, choose $\tilde{g}^{(c_p)}$ from the rest of mixture components which belong to the node $v$ and not yet assigned to any child node, i.e.,

$$\tilde{g}^{(c_p)} = \underset{g \in G^{(v)}_{p-1}}{\operatorname{argmax}} \min_{1 \leq p' < p} d(g, \tilde{g}^{(c_{p'})}), \tag{31}$$

where $G^{(v)}_{p-1} = G^{(v)} \setminus \{\tilde{g}^{(c_1)}, \cdots, \tilde{g}^{(c_{p-1})}\}$. If $G^{(v)}_{p-1}$ is an empty set, the child node is deleted from the tree.

2-3) For $p = 1, 2, \cdots, P$, set the parameters of child Gaus-

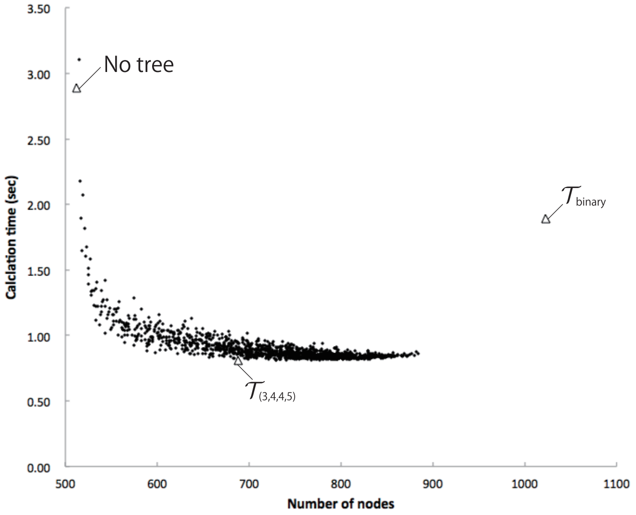

Fig. 6. Calculation time obtained using different tree structures (the SIFTH-Dense feature and $c_{\text{TH}} = 0.001$ were used). 1,364 trees of depth at most 5 that have at most 5 children per node and the binary tree are tested. $\mathcal{T}_{(3,4,4,5)}$ was the best tree and was selected as the optimized tree.

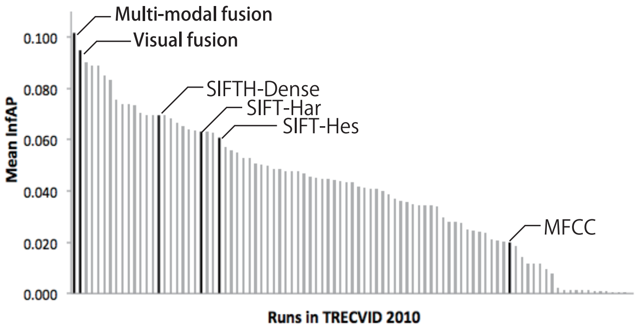

Fig. 7. Comparison of Mean Inf AP with runs of the TRECVID 2010.

sian pdfs $g^{(c_p)}$ as follows:

$$g^{(c_p)} \leftarrow \mathcal{N}(\cdot|\bar{\mu}, \bar{\Sigma}), \tag{32}$$

$$\bar{\mu} = \alpha\tilde{\mu}^{(c_p)} + (1-\alpha)\mu^{(v)}, \tag{33}$$

$$\begin{aligned}\bar{\Sigma} =& \alpha(\tilde{\Sigma}^{(c_p)} + \tilde{\mu}^{(c_p)}(\tilde{\mu}^{(c_p)})^{\mathsf{T}}) \\ &+ (1-\alpha)(\Sigma^{(v)} + \mu^{(v)}(\mu^{(v)})^{\mathsf{T}}) \\ &- \bar{\mu}\bar{\mu}^{\mathsf{T}},\end{aligned} \tag{34}$$

where $0 \leq \alpha \leq 1$ is a weight parameter to mix the selected pdf $\tilde{g}^{(c_p)} = \mathcal{N}(\cdot|\tilde{\mu}^{(c_p)}, \tilde{\Sigma}^{(c_p)})$ and their parent pdf $g^{(v)} = \mathcal{N}(\cdot|\mu^{(v)}, \Sigma^{(v)})$.

| | | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Multi-modal Fusion ($\mathcal{T}_{\text{opt}}$) | 1 | □ | ■ | ■ | □ | □ | □ | □ | ■ | ■ | ■ | ■ | ■ | ■ |
| Multi-modal Fusion (No tree) | 2 | | ■ | ■ | □ | □ | □ | □ | ■ | ■ | ■ | ■ | ■ | ■ |
| Visual Fusion ($\mathcal{T}_{\text{opt}}$) | 3 | | | □ | □ | □ | □ | □ | □ | ■ | ■ | ■ | ■ | ■ |
| Visual Fusion (No tree) | 4 | | | | □ | □ | □ | □ | □ | ■ | ■ | ■ | ■ | ■ |
| TRECVID_1 | 5 | | | | | □ | □ | □ | ■ | □ | ■ | ■ | ■ | ■ |
| TRECVID_2 | 6 | | | | | | □ | □ | □ | □ | □ | □ | □ | □ |
| TRECVID_3 | 7 | | | | | | | □ | □ | □ | □ | □ | □ | □ |
| TRECVID_4 | 8 | | | | | | | | □ | □ | □ | ■ | ■ | ■ |
| TRECVID_5 | 9 | | | | | | | | | □ | □ | □ | □ | □ |
| TRECVID_6 | 10 | | | | | | | | | | □ | □ | □ | □ |
| TRECVID_7 | 11 | | | | | | | | | | | □ | □ | □ |
| TRECVID_8 | 12 | | | | | | | | | | | | □ | □ |
| TRECVID_9 | 13 | | | | | | | | | | | | | □ |
| TRECVID_10 | 14 | | | | | | | | | | | | | |

Fig. 8. Results of partial randomization test ($p < 0.05$). Significant differences among top 10 runs in TRECVID 2010 and our fusion methods are shown. A black cell shows that there is significant difference between two methods.

## References

[1] N. Inoue, and K. Shinoda. A Fast MAP Adaptation Technique for GMM-supervector-based Video Semantic Indexing Systems. In Proc. of *ACM Multimedia* (short paper), Scottsdale, AZ, USA, 2011.

[2] N. Inoue, and et al. High-Level Feature Extraction using SIFT GMMs and Audio Models. In Proc. of *ICPR*, Istanbul, Turkey, 2010.

[3] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In Proc. of *ACM Multimedia MIR* workshop, Santa Barbara, CA, USA, 2006.

[4] A. F. Smeaton, P. Over, and W. Kraaij. High-Level Feature Detection from Video in TRECVid: a 5-Year Retrospective of Achievements. In A. Divakaran, editor, *Multimedia Content Analysis, Theory and Applications*, pages 151–174, Springer Verlag, Berlin, 2009.

[5] X. Zhou, K. Yu, T. Zhang, and T. Huang. Image classification using super-vector coding of local image descriptors. In Proc. of *ECCV*, Heraklion, Crete, Greece, 2010.

[6] X. Zhou, X. Zhuang, S. Yan, S.-F. Chang, M. H.-Johnson, and T. S. Huang. Sift-bag kernel for video event analysis. In Proc. of *ACM Multimedia*, Vancouver, British Columbia, Canada, 2008.

[7] Koen E. A. van de Sande, Theo Gevers, Cees G. M. Snoek. Empowering Visual Categorization With the GPU. In *IEEE Transactions on Multimedia*, vol. 13 (1), pages 60–70, 2011.

[8] C.G.M. Snoek, K.E.A. van de Sande, O. de Rooij, B. Huurnink, E. Gavves, D. Odijk, M. de Rijke, Th. Gevers, M. Worring, D.C. Koelma, and A.W.M. Smeulders The MediaMill TRECVID 2010 Semantic Video Search Engine. In Proc. of *TRECVID workshop*, Gaithersburg, MD, USA, 2010.

[9] S.-F. Chang, D. Ellis, W. Jiang, K. Lee, A. Yanagawa, A.C. Loui, and J. Luo, Large-scale Multimodal Semantic Concept Detection for Consumer Video. In Proc. of *ACM Multimedia MIR* workshop, Augsburg, Germany, 2007.

[10] W. Jiang, C. Cotton, S.-F. Chang, and D. Ellis, Short-Term Audio-Visual Atoms for Generic Video Concept Classification. In Proc. of *ACM Multimedia*, Beijing, China, 2009.

[11] S. N. Sinha, J. michael Frahm, M. Pollefeys, and Y. Genc. Gpu-based video feature tracking and matching. In Proc. of *EDGE* workshop, Chapel Hill, NC, USA, 2006.

[12] C. Wu. SiftGPU: A GPU implementation of sift. http://cs.unc.edu/˜ccwu/siftgpu, 2007.

[13] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In Proc. of *SLCV workshop, ECCV*, Prague, Czech Republic, 2004.

[14] J. Yang and A. G. Hauptmann. Evaluating bag-of-visual-words representations in scene classification. In Proc. of *ACM Multimedia MIR* workshop, Augsburg, Germany, 2007.

[15] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *In IJCV*, vol. 60 (2), pages 91–110, 2004.

[16] J. van de Weijer and C. Schmid. Coloring local feature extraction. In Proc. of *ECCV*, Graz, Austria, 2006.

[17] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32(9), pages 1582–1596, 2010.

[18] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. In Proc. of *ECCV*, Graz, Austria, 2006.

[19] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In Proc. of *ICCV*, Rio de Janeiro, Brazil, 2007.

[20] J. C. V. Gemert, J. mark Geusebroek, C. J. Veenman, and A. W. M. Smeulders. Kernel codebooks for scene categorization. In Proc. of *ECCV*, Marseille, France, 2008.

[21] W. M. Campbell, D. E. Sturim, and D. A. Reynolds. Support vector machines using gmm supervectors for speaker verification. In *IEEE Signal Processing Letters*, vol. 13, pages 308–311, 2006.

[22] T. Jaakkola and D. Haussler. Exploiting generative models in discriminative classifiers. In *In Advances in Neural Information Processing Systems 11*, pages 487–493. MIT Press, 1998.

[23] F. Perronnin, C. Dance., G. Csurka, and M. Bressan, Adapted Vocabularies for Generic Visual Categorization. In Proc. of *ECCV*, Graz, Austria, 2006.

[24] F. Perronnin and C. Dance. Fisher Kernels on Visual Vocabularies for Image Categorization. In Proc. of *CVPR*, Minneapolis, Minnesota, USA, 2007.

[25] F. R. Back and G. R. G. Lanckriet. Multiple Kernel Learning, Conic Duality, and the SMO Algorithm In Proc. of *ICML*, Banff, Alberta, Canada, 2004.

[26] F. Yan, J. Kittler, K. Mikolajczyk, and A. Tahir. Non-sparse multiple kernel learning for fisher discriminant analysis. In Proc. of *ICDM*, Miami, Florida, USA, 2009.

[27] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. In *IJCV*, vol. 60(1), pages 63–86, 2004.

[28] S. Ayache and G. Quenot. Video corpus annotation using active learning. In Proc. of *ECIR*, Glasgow, Scotland, UK, 2008.

[29] S. J. Young, G. Evermann, M. J. F. Gales, D. Kershaw, G. Moore, J. J. Odell, D. G. Ollason, D. Povey, V. Valtchev, and P. C. Woodland. The htk book, version 3.4, 2006.

[30] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm, 2001.

[31] E. Yilmaz, E. Kanoulas, and J. A. Aslam. A simple and efficient sampling method for estimating ap and ndcg. In Proc. of *ACM SIGIR*, Singapore, 2008.

**Nakamasa Inoue** received the B.E. degree in 2009, and the M.E. degree in 2011, both in computer science from the Tokyo institute of Technology, Japan. He is a JSPS research fellow (DC1). He is currently pursuing the Ph.D. degree at the Tokyo Institute of Technology. He is a student member of IEEE.

**Koichi Shinoda** received his B.S. in 1987 and his M.S. in 1989, both in physics, from the University of Tokyo. He received his D.Eng. in computer science from the Tokyo Institute of Technology in 2001. In 1989, he joined NEC Corporation, Japan, and was involved in research on automatic speech recognition. From 1997 to 1998, he was a visiting scholar with Bell Labs, Lucent Technologies, Murray Hill, NJ. From June 2001 to September 2001, he was a Principal Researcher with Multimedia Research Laboratories, NEC Corporation. From October 2001 to March 2002, he was an Associate Professor with the University of Tokyo. He is currently an Associate Professor with the Tokyo Institute of Technology. His research interests include speech recognition, video information retrieval, statistical pattern recognition, and human interfaces. Dr. Shinoda received the Awaya Prize from the Acoustic Society of Japan in 1997 and the Excellent Paper Award from the Institute of Electronics, Information, and Communication Engineers IEICE in 1998. He is an Associate Editor of Computer Speech and Language. He is a member of IEEE, ACM, ASJ, IEICE, IPSJ, and JSAI.