MICROSOFT RESEARCH ASIA TRECVID 2006 HIGH-LEVEL FEATURE EXTRACTION AND RUSHES EXPLOITATION

Xian-Sheng Hua, Tao Mei, Wei Lai, Meng Wang, Jinhui Tang, Guo-Jun Qi, Lvsong Li, Zhiwei Gu

Microsoft Research Asia
49 Zhichun Road, Beijing, 100080, P. R. China

{xshua, tmei, weilai}@microsoft.com

ABSTRACT

In this paper, we describe the MSRA experiments for TRECVID 2006, including details of the approaches and performance analyses for high-level feature extraction task and rushes exploitation task. For high-level feature extraction, we mainly investigated the benefit of unlabeled data by semi-supervised learning methods, including adaptive semi-supervised learning with kernel density estimation, manifold ranking, and transductive graph. Moreover, we performed fusion in two different levels: modality level and model level. We were ranked in the top 10 list in terms of mean average precision performance among all participants. For rushes exploitation, we detected the duplicate content based on ordinal video signature. We also performed video structuring (i.e. decomposing rushes into shots and sub-shots) and camera motion classification (i.e. classifying each sub-shot into static, pan, tilt, zoom, rotation, or object motion in terms of camera motion). Furthermore, we validated the approaches to concept modeling and detected 39 concepts on rushes data without re-training the visual models obtained in high-level feature extraction task.

Index Terms— video annotation, semi-supervised learning, kernel density estimation, manifold ranking, transductive graph, video signature

1. INTRODUCTION

In high-level feature extraction, we focused on semi-supervised learning methods by leveraging both labeled and unlabeled data. Support Vector Machine (SVM) [1] was adopted as the baseline, which is belonging to supervised and discriminative learning method. Then we adopted three semi-supervised learning methods, including adaptive semi-supervised learning with kernel density estimation (AdaSSLKDE) [2] [3], Manifold Ranking [4], and transductive graph (T-Graph) [5]. Among these three methods, AdaSSLKDE can be regarded as generative model while the other two as discriminative models. For each learning method, we trained different models on three low-level visual modalities. Then these models were fused across features and methods, i.e. average fusion for models across features for each single method and linear weighted fusion for models across methods. Finally, we submitted the following 6 runs to TRECVID:

- A_MSRA_TRECVID_1: linear weighted fusion of SVM, Manifold Ranking, AdaSSLKDE, and T-Graph.
- A_MSRA_TRECVID_2: linear weighted fusion of Manifold Ranking and AdaSSLKDE. We fused the outputs from discriminative and generative models together.
- A_MSRA_TRECVID_3: linear weighted fusion of SVM and AdaSSLKDE. This run achieved the best performance.
- A_MSRA_TRECVID_4: a single manifold ranking model fused across three low-level visual modalities.
- A_MSRA_TRECVID_5: a single AdaSSLKDE model fused across three low-level visual modalities.
- A_MSRA_TRECVID_6: a single SVM model fused across three low-level visual modalities. This baseline achieved 8.62% MAP.

The corresponding performances are listed in Table 1, where we found that A_MSRA_TRECVID_3 achieved the best MAP among the submitted 6 runs. Additionally, we tried to incorporate text-based features, i.e. SVM with TF-IDF features. Unfortunately, most text-based results for the 39 concepts were unacceptable, which indicates that transcripts embedded in video do not have high correlation to this set of concepts.

In addition, we participated in rushes exploitation task, including shot/sub-shot detection, interview/non-interview shot classification, camera motion classification, as well as four concepts detection (indoor, urban, person, and water body). In terms of concepts detection in rushes videos, we adopted the SVM based classifiers without re-training the visual models obtained in high-level feature extraction task.

Table 1. The description and performances of MSRA 6 runs for high-level feature extraction

RUN ID	Description	MAP
A_MSRA_TRECVID_1	SVM + Manifold Ranking + AdaSSLKDE + T-Graph	0.1026
A_MSRA_TRECVID_2	Manifold Ranking + AdaSSLKDE	0.0704
A_MSRA_TRECVID_3	SVM + AdaSSLKDE	0.1031
A_MSRA_TRECVID_4	Manifold Ranking	0.0732
A_MSRA_TRECVID_5	AdaSSLKDE	0.0632
A_MSRA_TRECVID_6	SVM	0.0862

2. HIGH-LEVEL FEATURE EXTRACTION

The pipeline for high-level feature extraction is shown in Figure 1. We firstly extracted seven kinds of low-level visual features and grouped them into three different modalities. Then for each learning method, we trained three models based on the three modalities and fused these models by linear combination. Finally, the different learning methods were linearly fused, in which the weight parameters were obtained by cross-validation experiments.

2.1. Low-level Feature Extraction

We extracted seven kinds of low-level visual features for each sub-shot key-frame, including color, texture and shape. As a result, there are 663-dimensional features extracted to represent a single key-frame:

- Color Histogram (64D) – global color represented as a 64-dimensional histogram in Lab color space.
- Color Autocorrelogram (144D) – based on 36 bin color histogram and 4 different distance k, i.e., $k = 1, 3, 5, 7$.
- Block-wise Color Moment (225D) – based on 5 by 5 division of images in Lab color space [2].
- Co-occurrence Texture (20D)
- Wavelet Texture (128D)
- Edge Distribution Layout (75D)
- Face (7D) – consisting of the face number, face area ratio, the position of the largest face.

In order to obtain different visual modalities, we spited these feature into three modalities by considering that feature dimensions of these modalities are close, as shown in Figure 1:

- Modality 1 (225D): block-wise color moment.
- Modality 2 (208D): color autocorrelogram and color histogram.
- Modality 3 (230D): co-occurrence texture, wavelet texture, edge distribution layout, and face.

2.2. High-level Feature Extraction

In high-level feature extraction, we focus on semi-supervised learning methods which leverage the information not only from labeled data, but also from a large amount of unlabeled data. The semi-supervised methods consist of AdaSSLKDE, Manifold Ranking, and T-Graph. SVM is adopted as a baseline.

2.2.1. Support Vector Machine (SVM)

SVM [1] was adopted as the baseline for high-level feature extraction, since it had achieved quite satisfactory performance in concept detection over the past few years. RBF is adopted as the kernel function. The classification of each concept was regarded as two-class problem. We ranked the shots according to the outputs from SVM [6]. The cross-validation is performed to select the best choice of the parameter C and γ, in which the principle for selecting parameters is average precision (AP) instead of average classification precision. Another issue in SVM training is the data imbalance problem. Since the amount of negative samples is much more than that of positive samples, the negative data are down-sampled and $1/3$ was reserved for training.

SVM was trained based on each visual modality. Thus we have three SVMs in total. These SVM models were fused by linear weights. Then, another cross-validation is performed to find the best weight parameters.

2.2.2. Manifold Ranking

The original manifold ranking algorithm is described in [4], for which maybe a name "graph-based SSL with normalized Laplacian" is more appropriate. Here our method is similar to the method adopted in [7], which has been modified in several places to be more efficient.

Let $L = \{1, 2, \ldots, \ell\}$ be the index set of labeled samples, and $U = \{\ell + 1, \ell + 2, \ldots, n\}$ be the index set of unlabeled samples. We define a vector $\mathbf{y}^+ = \{y_1^+, y_2^+, \ldots, y_n^+\}$, where $y_i^+ = 1$ if x_i is a labeled positive sample, and $y_i^+ = 0$ otherwise. Conversely, we define $\mathbf{y}^- = \{y_1^-, y_2^-, \ldots, y_n^-\}$, where $y_i^- = -1$ if x_i is a labeled negative sample, and $y_i^- = 0$ otherwise. Then we implement the manifold-ranking process as Algorithm 1.

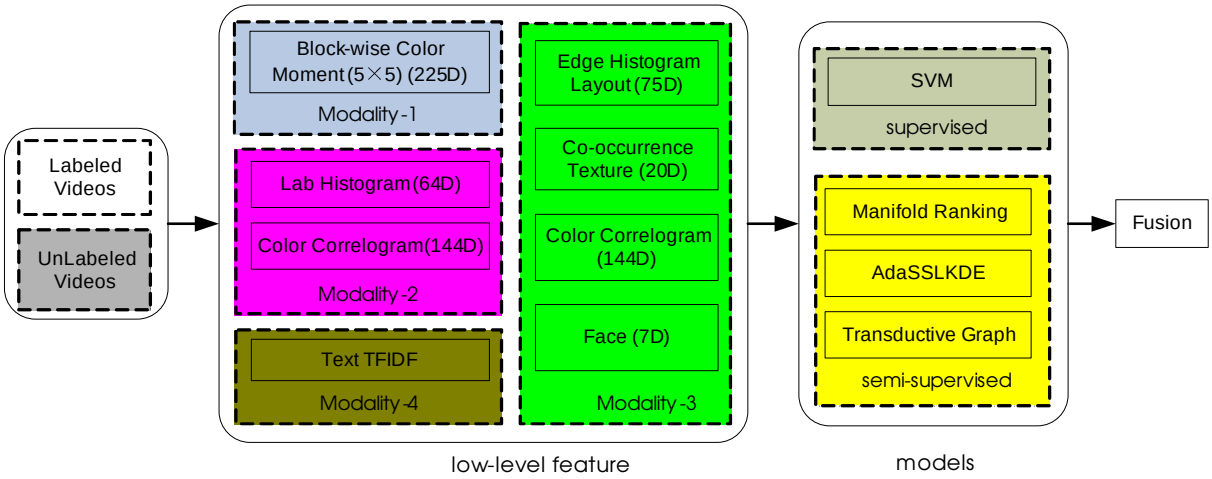

Fig. 1. The MSRA TRECVID 2006 Feature Extraction pipeline

Algorithm 1 Manifold Ranking

1: Define a sparse graph above all samples: x_i and x_j are connected if x_i belongs to the K-nearest neighborhood of x_j, and vice versa.
2: Define affine matrix $\mathbf{W}$ by letting $W_{ij} = D(x_i, x_j)$ if x_i and x_j are connected and $i \neq j$, and otherwise $W_{ij} = 0$.
3: Construct a matrix $\mathbf{S} = \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}}$ in which $\mathbf{D}$ is a diagonal matrix with its (i, i)-element equals to the sum of the i-th row of $\mathbf{W}$.
4: Initialize $[\mathbf{f}^+, \mathbf{f}^-]$. Then iterate $[\mathbf{f}^+, \mathbf{f}^-] = \alpha\mathbf{S} \times [\mathbf{f}^+, \mathbf{f}^-] + (1 - \alpha)[\mathbf{y}^+, \mathbf{y}^-]$ for T times, where α is a parameter in $(0,\ 1)$.

Notice that in Step 2, we have adopted three different distance metrics for $D(x_i, x_j)$, including L_1 distance, L_2 distance, and Cauchy distance, i.e.

$$D_1(x_i, x_j) = \exp\left\{-\frac{\|x_i - x_j\|}{\sigma}\right\} \quad (1)$$

$$D_2(x_i, x_j) = \exp\left\{-\frac{\|x_i - x_j\|^2}{\sigma^2}\right\} \quad (2)$$

$$D_c(x_i, x_j) = \frac{1}{1 + \|\frac{x}{\sigma}\|^2} \quad (3)$$

In this way, we obtain $\mathbf{f}^+$ and $\mathbf{f}^-$, which are named as positive score and negative score, respectively. As generally positive samples are scarcer and more compact than negative samples, they should contribute more in concept learning [7]. Thus, here we fuse $\mathbf{f}^+$ and $\mathbf{f}^-$ as follows

$$\mathbf{f} = \left(\frac{1}{frequency} - 1\right) \times \mathbf{f}^+ + \mathbf{f}^- \quad (4)$$

where $frequency$ measure is estimated as the ratio of positive samples and all samples. It implies that if a concept is scarcer, the positive samples contribute more. We rank the samples according to the measure $\mathbf{f}$.

2.2.3. Adaptive Semi-supervised Learning with Kernel Density Estimation (AdaSSLKDE)

1. SSLKDE

Firstly we introduce semi-supervised learning with kernel density estimation (SSLKDE). The SSLKDE is developed on classical Kernel Density Estimation (KDE) approach. Instead of only adopting labeled samples in traditional KDE method, both labeled and unlabeled samples are leveraged to estimate class conditional probability densities in SSLKDE. The density $p_k(x)$ is estimated based on non-parametric extended kernel density estimation without any model assumption, i.e.,

$$p_k(x) = \frac{\sum_{j \in L} P_k(x_j)\kappa(x - x_j) + \lambda \sum_{j \in U} P_k(x_j)\kappa(x - x_j)}{\sum_{j \in L} P_k(x_j) + \lambda \sum_{j \in U} P_k(x_j)} \quad (5)$$

where the posterior probability $P_k(x_j)$ and density $p_k(x)$ can be estimated by a bi-directional relationship, κ is a kernel function, the parameter λ is used to modulate the effect from unlabeled data, and L and U denote labeled and unlabeled data, respectively. More details about its derivation can be found in [2].

An interesting phenomenon is that the formulation of SSLKDE is exactly the same to a graph-based method, which is proposed in [8]. More details about the connection between SSLKDE and graph-based SSL methods can also be found in [2]. Here we only give an iterative implementation method of SSLKDE in Algorithm 2 (here we consider a standard K-class classification problem).

Here we also try three different kernels: Exponential ker-

Algorithm 2 SSLKDE

1: Select a kernel function κ, which satisfies $\kappa(x) > 0$ and $\int \kappa(x)dx = 1$.
2: Define matrix $\mathbf{P}$ as

$$P_{ij} = \frac{\kappa(x_j - x_i)}{\sum\limits_{i \in L \cup U} \kappa(x_j - x_i)} \tag{6}$$

3: Initialization of posterior probability matrix $\mathbf{F} = \left[\mathbf{F}_L^T,\ \mathbf{F}_U^T\right]^T$. Here $\mathbf{F}$ is an $n \times K$ matrix, where F_{ij} is the posterior probability of class j giving x_i.
4: Estimate densities based on the posterior probability matrix $\mathbf{F}$, and then re-calculate posterior probabilities according to Bayes rule (details can be found in [2]). It can be expressed as $\mathbf{F} = \mathbf{PF}$ directly.
5: Adjust the posterior probabilities of the labeled data as $\mathbf{F}_L = (1-t) \times \mathbf{F}_L + t \times \mathbf{Y}$, where $\mathbf{F}_L$ is the above $n \times K$ part of matrix $\mathbf{F}$, $\mathbf{Y}$ is the labeling matrix which is defined as $Y_{ij} = \delta(y_i = j)$ (δ is indicator function, i.e., $\delta[ture] = 1$, $\delta[false] = 0$), and t is a weighting factor which satisfies $0 < t \leqslant 1$.
6: Repeat from step 2 until $\mathbf{F}$ converges

nel, Gaussian kernel, and Cauchy kernel, i.e.,

$$\kappa_e(x) = \frac{1}{(2\sigma)^d} \exp\Big\{ - \frac{\|x\|}{\sigma} \Big\} \tag{7}$$

$$\kappa_g(x) = \frac{1}{(2\pi)^{d/2}\sigma^d} \exp\Big\{ - \frac{\|x\|^2}{2\sigma^2} \Big\} \tag{8}$$

$$\kappa_c(x) = \frac{1}{(\pi\sigma)^d} \times \frac{1}{1 + \|\frac{x}{\sigma}\|^2} \tag{9}$$

Besides that, we also adopt a similar method as in manifold ranking to fuse positive scores and negative scores

$$f_i = \big(\frac{1}{frequency} - 1\big) \times F_{i1} + F_{i2} \tag{10}$$

Compare the above implementation scheme with that of manifold ranking, we can also find that these two methods are very similar. Although SSLKDE has the same formulation to a graph-based SSL method, it gives a novel probabilistic viewpoint of graph-based SSL methods. As KDE is an extensively studied topic, there are many existing works over this classical method, and these works can be easily adapted to semi-supervised versions.

2. AdaSSLKDE

Here we illustrate an improved method named AdaSSLKDE (Adaptive Semi-Supervised Learning by Manifold Kernel Density Estimation), which is developed based on an existing *adaptive KDE* method [4]. The main idea of these methods is to vary the kernel bandwidths according to the sparseness degree of the data such that broader kernel is applied in the region of low density.

In [8], Vincent et al. proposed a novel adaptive KDE method named *Manifold KDE*, which defines a multivariate Gaussian kernel by exploiting local structure, including both sparseness degree and local principle directions. More detailed motivations of this work can be found in [4]. It considers a Gaussian kernel over sample x_i according to its local covariance as follows

$$\kappa_i(x) = \frac{1}{\sqrt{2\pi|C_i|}} \exp\Big\{ - \frac{1}{2} x^T C_i x \Big\} \tag{11}$$

where covariance matrix C_i is defined as

$$C_i = \frac{1}{N} \sum_{j \in N(x_i)} (x_j - x_i)^T (x_j - x_i) \tag{12}$$

We further simplify this method by only considering diagonal covariance matrix, which yields manifold Gaussian kernel as follows

$$\begin{cases} \kappa_g(x_i, x) = \frac{1}{(2\pi)^{\frac{d}{2}} \prod\limits_{r=1}^{d} \sigma_{ir}} \exp\Big\{ - \frac{1}{2} \sum\limits_{r=1}^{d} \frac{x^2}{\sigma_{ir}^2} \Big\} \\ \sigma_{ir}^2 = \frac{\sigma_0^2}{N} \sum\limits_{j \in N(x_i)} (x_{jr} - x_{ir})^2 \end{cases}$$

Here σ_0 is named as global bandwidth factor, which controls the global scale of bandwidth factors σ_{id}.

Similarly, we obtain manifold Exponential kernel as follows:

$$\begin{cases} \kappa_e(x_i, x) = \frac{1}{2^d \prod\limits_{r=1}^{d} \sigma_{ir}} \exp\Big\{ - \sum\limits_{r=1}^{d} \frac{\|x\|}{\sigma_{ir}} \Big\} \\ \sigma_{ir} = \frac{\sigma_0}{N} \sum\limits_{j \in N(x_i)} (\|x_{jr} - x_{ir}\|) \end{cases}$$

Then we only need to replace P_{ij} defined in step 2 in Algorithm 2 by

$$P_{ij} = \frac{\kappa(x_j, x_i)}{\sum\limits_{i \in L \cup U} \kappa(x_j, x_i)} \tag{13}$$

The other implementation steps of AdaSSLKDE are the same to SSLKDE.

2.2.4. Transductive Graph (T-Graph)

In addition to the above three methods, we also adopted a Transductive Graph (T-Graph) framework for high-level feature extraction, which directly focuses on predicting the available samples in a current unlabeled pool, instead of trying to build a good classifier workable for any unavailable data [5]. In this framework, a number of hierarchical clustering results are constructed both from labeled and unlabeled data. We aim to make the clusters as pure as possible, i.e., samples in

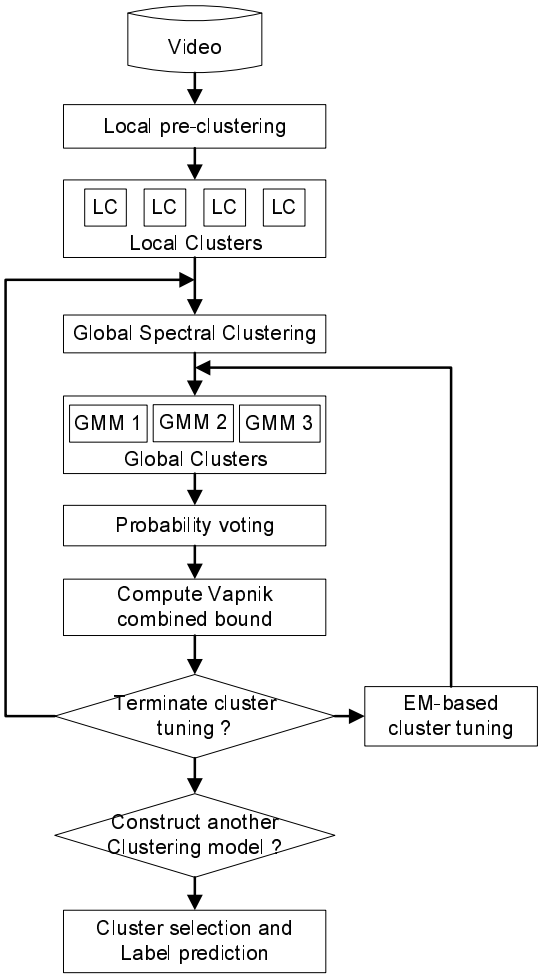

Fig. 2. T-Graph framework

the same cluster have the same label. To further purify these hierarchical clustering results, an EM based cluster tuning algorithm is iteratively employed. Based on these clustering results, several hypotheses are generated by probability voting among labeled samples in the obtained clusters. From these hypotheses, one of them is chosen according to the Vapnik combined bound, and it is then applied to predict the labels of unlabeled samples.

However, due to the limited number of runs to be submitted, we did not submit this single model to TRECVID. Instead, we fuse T-Graph with the other models. In our initial validation experiments, we found that the performance of T-Graph is also less than the other three single models.

As shown Figure 2, in the "outer cycle", the above global clustering process (i.e., the "outer cycle") is repeated and different sets of global clusters (*clustering models*) are obtained. These clustering models are generated from different clustering parameters, such as the number of global cluster components and so on. From these different hierarchical clustering results we obtain several hypotheses by *probability voting*. Among these hypotheses, the one that minimizes the *Vapnik combined bound* is selected to predict the labels of shots in the unlabeled pool. This T-Graph framework can also be viewed as a model selection process and the *Vapnik combined bound* is applied as a model selection criterion.

2.2.5. Fusion Across Methods

As we have mentioned before, in each single model method, we fuse the three models trained on three low-level visual modalities by linear combination. The linear weights are obtained by cross-validation.

In the stage of fusion across methods, we also applied linear fusion methods to combine all concept detection hypotheses output from different models. As we have four single models, we can get $2^4 = 16$ combinations in total. Actually, we submitted three combinations:

- Linear weighted fusion of SVM, manifold ranking, AdaSSLKDE, and T-Graph.
- Linear weighted fusion of manifold ranking and AdaSSLKDE.
- Linear weighted fusion of SVM and AdaSSLKDE.

The selection of submitting the three fusion methods is based on the consideration that, using discriminative and generative models simultaneously could boost the performance of each other.

2.3. Experimental Results

Figure 3 shows the distribution of positive examples of all concepts in TRECVID 2006 development data. This kind of information could intuitively explain the difficulty for detecting some concepts with small appearing frequency. For those concepts with considerable appearing frequency such as People, Face, and Outdoor, the performances are expected to be better than others.

Table 1 lists the MAP of 6 submitted runs. Figure 4 shows the AP performance of MSRA 6 runs for high-level feature extraction. We can see that SVM still achieves the best performance (MAP $= 0.0862$) among the four single models (i.e. SVM, Manifold ranking, AdaSSLKDE, and T-Graph). We can also observe from Table 2 that among 20 concepts used for validation by NIST, SVM achieves only one best performance (i.e., concept 35). However, most of the performances (APs) of SVM are more stable, and in average better than the other single model methods. This may explain the reason why the MAP of SVM is the best among four single models.

From Table 1, we can see that the MAP of Manifold ranking and AdaSSLKDE are 0.0732 and 0.0632, respectively, which are a little lower than that of SVM. However, from Table 2, we can observe that Manifold ranking achieve the best performance in four concepts (i.e., concept 1, 22, 23, and 26),

Table 2. The AP performance of MSRA 6 runs submitted to TRECVID 2006, the bold ones indicate the best among 6 runs for each concept

RUN ID	1	3	5	6	10	12	17	22	23	24	26	27	28	29	30	32	35	36	38	39
A_MSRA_TRECVID_1	0.217	**0.305**	**0.003**	0.205	0.013	**0.104**	**0.036**	0.034	0.002	0.087	0.004	**0.283**	0.109	**0.004**	**0.158**	0.018	0.035	0.048	**0.266**	**0.122**
A_MSRA_TRECVID_2	0.296	0.233	0.002	0.094	0.016	0.046	0.028	0.034	0.002	0.058	0.005	0.119	0.078	0.002	0.044	0.018	0.014	0.025	0.177	0.116
A_MSRA_TRECVID_3	0.288	0.291	0.001	**0.205**	0.010	0.102	0.035	0.031	0.002	**0.098**	0.004	0.271	**0.109**	0.003	0.160	**0.019**	0.027	**0.049**	0.259	0.099
A_MSRA_TRECVID_4	**0.297**	0.303	0.002	0.094	0.016	0.038	0.022	**0.052**	**0.005**	0.046	**0.006**	0.121	0.079	0.002	0.030	0.018	0.008	0.022	0.194	0.108
A_MSRA_TRECVID_5	0.267	0.199	0.001	0.065	**0.019**	0.043	0.027	0.034	0.002	0.045	0.003	0.118	0.078	0.004	0.043	0.017	0.014	0.025	0.143	0.116
A_MSRA_TRECVID_6	0.203	0.275	0.001	0.194	0.005	0.078	0.029	0.000	0.001	0.078	0.003	0.275	0.029	0.001	0.145	0.019	**0.037**	0.046	0.249	0.056

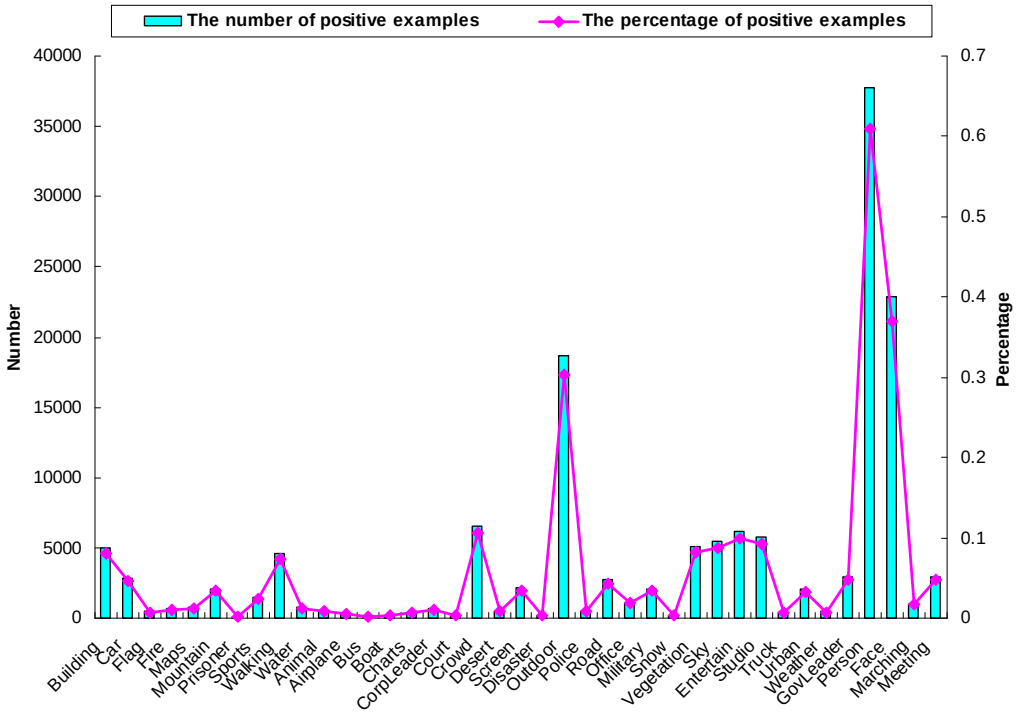

Fig. 3. The distribution of positive examples of all concepts in TRECVID 2006 data

and AdaSSLKDE achieves the best in one concept (i.e., concept 10). But since the performances of Manifold ranking and AdaSSLKDE are not stable, the MAP from these two methods is less than that of SVM. Therefore, we can draw the conclusion that semi-supervised methods such as Manifold ranking and AdaSSLKDE outperform supervised methods such as SVM in some specific concepts, but semi-supervised methods are not as stable as supervised methods.

As shown in Table 1, A_MSRA_TRECVID_3 (i.e., SVM + AdaSSLKDE) achieves the best performance (MAP= 0.1031) among the 6 submitted runs. Meanwhile, A_MSRA_TRECVID_1 (linear fusion of all single model methods) has the very close performance (MAP = 0.1026) to A_MSRA_TRECVID_3. This can be deduced from Table 2 that, there are 9 best performances coming from A_MSRA_TRECVID_1 run and 5 best from A_MSRA_TRECVID_3, respectively. The reason that A_MSRA_TRECVID_3 achieves the best performance lies in adopting the most effective discriminative and generative models together.

3. RUSHES EXPLOITATION

Rushes are unedited videos, with long shots taken by still camera, highly repetitive shots and natural sound. We participated in rushes exploitation task and finished the tasks required by TRECVID, including redundancy (i.e., duplicate) detection and six concepts detection (i.e., interview, fixed camera, indoor, urban, person, and water body). Additionally, we performed video structuring and camera motion classification on rushes.

Figure 6 shows the flowchart for rushes exploitation task. At first, rushes are decomposed into candidate shots by visual content similarity [9]. Since there are quite small changes between the redundant shots (e.g., the filmmakers may cut an unsatisfactory interview shot and take another similar shot from the beginning of this interview), typical shot detection methods based on color histogram similarity can not detect this kind of changes. Therefore, it is critical to classify these candidate shots into interview or non-interview shots based on visual content stability. For interview shots, a refinement post-processing based on detecting pixel changes is performed

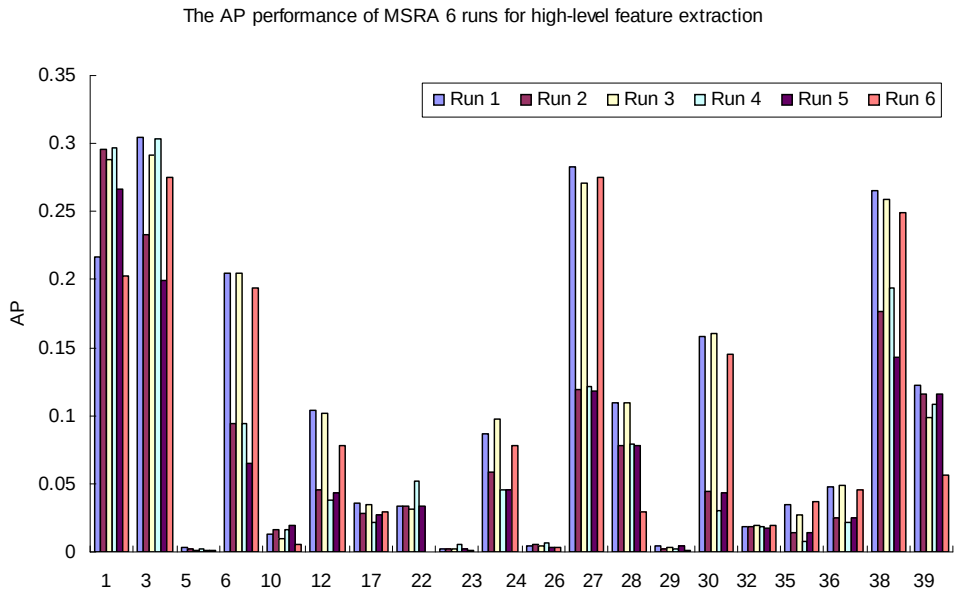

Fig. 4. The AP performance of MSRA 6 runs for high-level feature extraction

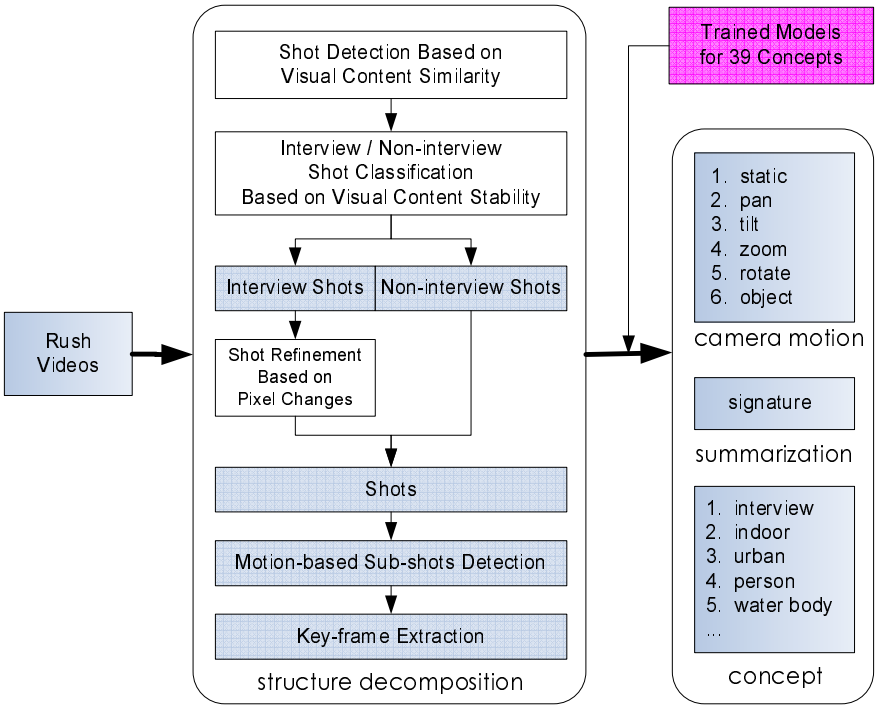

Fig. 5. The flowchart for TRECVID rushes exploitation task

to detect the small changes between consecutive duplicate shots. After shot detection, we further decompose shots into successive sub-shot by a motion-based method [10]. Each shot is decomposed into several sub-shots, in which each sub-shot corresponds to one unique camera motion. We also classify the camera motion into six categories based on the affine

Table 3. The description and performances of MSRA 6 runs for high-level feature extraction

Camera Motion	Recall	Precision	$F1$
Static	0.9513	0.9146	0.9326
Pan	0.7519	0.7147	0.7328
Tilt	0.7113	0.7263	0.7339
Zoom	0.0211	0.2727	0.0392
Rotation	0.1667	0.3333	0.2222
Object Motion	0.2500	0.2500	0.2500

motion parameters, including *static*, *pan*, *tilt*, *zoom*, *rotation*, or *object motion*. Then each sub-shot is represented by a key-frame. The concepts are further detected based on the SVM model obtained from high-level feature extraction task. In order to detect duplicate video shots, we adopted the ordinal video signature in [11] to represent a shot, and detected the mostly identical shots based on signature similarities. As a result, the last shots in the successive duplicate shots are reserved for summarization.

3.1. Structure Decomposition

We firstly detect candidate shots based on color histogram similarity [9]. In rushes, typically the visual content changes between successive interview shots are quite small. As a result, there are a lot of missed shot boundaries in interview shots. A straightforward approach to this issue is to classify the candidate shots into interview/non-interview shots, and detect the small changes in interview candidate shots. Generally, interview shots depict much more stable visual content than non-interview shots. The reason is that filmmakers' intention during capturing an interview shot is to capture person's expression, thus the visual elements composing the picture (i.e., person and background) do not change significantly. Figure 6 gives sample shots belonging to interview and non-interview shots. We adopted the variance of color entropy over all frames in a shot as the measurement of content *stability*, and used a single threshold on *stability* to decide whether this shot is an interview shot. In this way, all the candidate shots are classified into interview/non-interview shots.

(a) interview shot (b) non-interview shot

Fig. 6. Sample shots of interview/non-interview

For interview shots, we used the pixel based similarity instead of global color histogram based similarity to detect the shot boundary within interview shots, because pixel based similarity is more sensitive to small changes, thus the missed boundaries can be detected during this refinement step.

Table 4. The performance of MSRA duplicate detection

–	Recall	Precision	$F1$
Duplicate	0.7015	0.6222	0.6595

In rushes data, a shot may be taken by a considerable long time. Therefore, shot is not a suitable temporal unit in rushes. In stead, sub-shot, which is defined as sub-segment within a shot depicting a unique camera motion [10], is selected as the basic unit for rushes content analysis.

We further used affine motion model to estimate the camera motion parameters. Based on the affine parameters, the motion based method is adopted to detect sub-shot boundaries and classify a sub-shot into six categories in terms of camera motion, i.e., *static*, *pan*, *tilt*, *zoom*, *rotation*, or *object motion*. As a result, the 49 hours' rush videos provided by TRECVID are decomposed into 1845 shots and 3469 sub-shots. The performance of camera motion classification is listed in Table 3.

3.2. Duplicate Detection

Duplicates are the shots taken several times due to some mistakes. Indeed, the speech similarity between duplicates is more reliable than visual similarity. However, we focus on visual similarity in this year. A shot is regarded as duplicate when there are other segments (segments could be shots or sub-segments in the shots) in the same video containing the same visual content, even the duration of these shots are different.

We adopted the ordinal signature based method [11] to detect duplicates in rushes. Firstly, the video frames were sampled at a fixed ratio (e.g., 5 fps). Each sampled frame was divided into N_x by N_y regions, and the average gray intensity in each region was computed. Then the set of average intensities was sorted in ascending order and the rank was assigned to each region. If we choosed $N_x = N_y = 3$, then each sampled frame could be represented by a 9-dimensional $(9 = 3 \times 3)$ signature vector. The sequence matching process proposed in [11] is used to identify the duplicate shots based on the ordinal signature measurement. Within a rush video, we assigned the shots belonging to the same duplicates with identical label. Finally, the last shot (in terms of time) within a group of duplicates is selected for composing the summary. The performance of duplicate detection is listed in Table 4.

3.3. Concept Detection

For concept detection, we did not train any new model for rushes. Instead, we leveraged the SVM models obtained from high-level extraction task to do concept detection. Since we did not have ground truth of each concept, we used modified average precision (AP) to validate our models. The modified

Table 5. The AP performance of MSRA concept detection on rushes

Concept	$AP(\alpha = 200)$		$AP(\alpha = 500)$		$AP(\alpha = 1000)$	
	$AP_1(\alpha)$	$AP_2(\alpha)$	$AP_1(\alpha)$	$AP_2(\alpha)$	$AP_1(\alpha)$	$AP_2(\alpha)$
Indoor	0.0779	0.3539	0.0567	0.2727	0.0453	0.2386
Urban	0.0274	0.2192	0.0137	0.1676	0.0074	0.1519
Person	0.2403	0.4956	0.2756	0.5122	0.2913	0.5277
Water body	0.0794	0.4291	0.0465	0.2941	0.0254	0.2594

APs are defined as:

$$AP_1(\alpha) = \frac{1}{\alpha}\sum_{j=1}^{\alpha}\left(\frac{R_j}{j}I_j\right) \quad (14)$$

$$AP_2(\alpha) = \frac{1}{R_\alpha}\sum_{j=1}^{\alpha}\left(\frac{R_j}{j}I_j\right) \quad (15)$$

where α is the depth for labeling, R_j is the number of relevant sub-shots in the top j returned sub-shots, and R_α is the number of true sub-shots in the depth of α. Let $I_j = 1$ if the j-th sub-shot is relevant and 0 otherwise. We fixed the parameter α to 200, 500, and 1000, respectively. Five users were invited to label the returned sub-shots at these three different depths. The performances of the four concepts detection is listed in Table 5.

4. CONCLUSIONS

We participated in high-level feature extraction and rushes exploitation tasks in TRECVID 2006. In high-level feature extraction, we focused on leveraging the large amount of unlabeled data for concept detection by proposing several kinds of semi-supervised learning methods, such as AdaSSLKDE and Manifold ranking. We have observed from the results that semi-supervised learning methods outperformed supervised learning methods (SVM) in some concepts, but semi-supervised learning methods are not as stable as supervised learning methods in most concepts. As a result, the mean average precision of semi-supervised methods is lower than that of supervised methods. We also observed that combining discriminative and generative models together is able to achieve satisfactory performance.

In rushes exploitation, we used rule-based method to detect interview/non-interview shots, and adopted the ordinal signature to help detect redundant shot (i.e., duplicate). Additionally, we used motion-based methods to decompose shot into sub-shots, and further classify sub-shots into six categories in terms of camera motions.

5. REFERENCES

[1] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.

[2] M. Wang, X.-S. Hua, Y. Song, X. Yuan, S. Li, and H.-J. Zhang, "Automatic video annotation by semi-supervised learning with kernel density estimation," in *Proceedings of ACM International Conference on Multimedia*, Santa Barbara, CA, Oct 2006.

[3] M. Wang, X.-S. Hua, Y. Song, and L.-R. Dai, "Semi-supervised by kernel density estimation: Exploiting unlabeled data in automatic video annotation," *submitted to IEEE Trans. on Multimedia: Special issue on Semantic Image and Video Indexing in Broad Domains*.

[4] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Scholkopf, "Learning with local and global consistency," in *Proceedings of Advances in Neural Information Processing System*, 2004.

[5] R. El-Yaniv and L. Gerzon, "Effective transductive learning via PAC-Bayesian model selection," *Technical Report CS-2004-05, Technion-Israel Institute of Technology*, 2004.

[6] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," .

[7] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of International Conference on Machine Learning*, 2003.

[8] P. Vincent and Y. Bengio, "Manifold parzen windows," in *Proceedings of Advances in Neural Information Processing System*, 2003.

[9] H.-J. Zhang, A. Kankanhalli, and S. W. Smoliar, "Automatic partitioning of full-motion video," *Multimedia Systems*, vol. 1, no. 1, pp. 10–28, June 1993.

[10] T. Mei, X.-S. Hua, H.-Q. Zhou, and S. Li, "Modeling and mining of users' capture intention for home videos," *to Appear in IEEE Trans. on Multimedia*.

[11] X.-S. Hua, X. Chen, and H.-J. Zhang, "Robust video signature based on ordinal measure," in *Proceedings of IEEE International Conference on on Image Processing*, Singapore, Oct 2004, pp. 685–688.