TokyoTech's TRECVID2007 Notebook

Taichi Nakamura, Koichi Shinoda and Sadaoki Furui
Department of Computer Science, Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo 152-8552 Japan
nakamura@ks.cs.titech.ac.jp {shinoda,furui}@cs.titech.ac.jp

In this notebook we describe our TRECVID 2007 experiments. We TokyoTech team participated in high-level the feature extraction task.

1 Summary

For the high-level feature extraction task, we use visual features (visual words of keyframe images and motion features), and do not use any audio information. Maximum entropy models [1] are employed to model these visual features.

Because there was a material mistake in our submission, the inferred Average Precisions of our runs was almost zero. Therefore, in this notebook, we also show the results evaluated by ourselves with the truth judgments file (qrels file).

Figure 1 briefly shows our visual feature extraction process. From each keyframe, we first extract Affine regions, then describe each of the regions with a SIFT descriptor. We quantize the SIFT descriptor using a tree-cluster codebook. We call this quantized descriptor "visual word". In addition to the visual word itself, we combine the visual word and motion information of the corresponding Affine region to get a "motion feature". Finally, we use the number of occurrences of the visual words and the motion features to construct a feature vector for a maximum entropy model.

2 Tree cluster codebook

2.1 Region extraction

We use a sparse image representation [2] based on affine-invariant regions. Affine-invariant regions are detected by Harris-Affine and Hessian-Affine detectors. We use the implementation of Visual Geometry Group [4] with its default parameters. We extracted 20,747,632 Harris-Affine regions and 18,516,464 Hessian-Affine regions from the TRECVID 2005 development set.

2.2 Visual words

Each of the extracted regions is first described with a 128 dimensional SIFT descriptor (4x4-grid, 8-orientation). This descriptor is then quantized with tree-cluster codebooks constructed in advance from a training data set. We try two kinds of codebook sets, one is a codebook shared among all high-level features, and the other is a set of codebooks constructed for each high-level feature. Finally, for each keyframe, we count occurrences of each visual word, and use the count numbers to build a feature vector. We will explain the construction process of tree-cluster codebooks and the quantization process (shown in Figure 2) in more detail.

A tree-cluster codebook is constructed by recursively clustering the SIFT descriptors using the K-means clustering method (we used K=2). This is done as follows: we first divide all the descriptors into 2 clusters, then we divide the cluster members of each cluster into 2 clusters again, and so on. We stop dividing a cluster when the number of cluster members falls below a predetermined threshold. We set the threshold to "number of samples clustered" / "target number of leaf clusters".

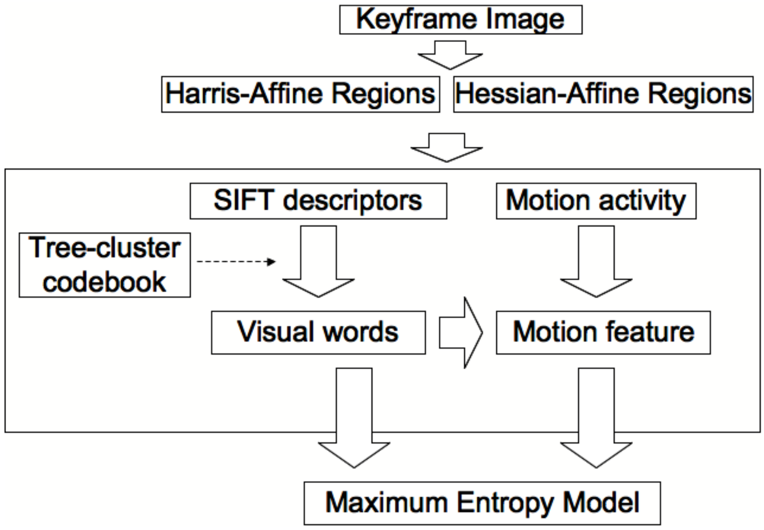

Figure 1: Overview of our visual feature extraction process.

In the quantization process, a SIFT descriptor is repeatedly quantized according to the tree-cluster codebook, and assigned to one of the leaf clusters. We also assign a descriptor to the parent node of the leaf cluster to make our visual words hopefully robust in regard to codebook sizes, as shown in Figure 3.

We note that, using tree-clustering, we can reduce the computational cost for quantizing each region to $O(\log N)$ from $O(N)$, which was the case for K-means clustering we used last year (N: Number of clusters).

3 Motion features

Motion information is important for some high-level features such as car, walking_running. We pay attention to regions where the motion is active, and use a visual word of these regions as a motion feature (shown in Figure 4).

For each region of a keyframe, we calculate a motion activity value by taking the average of differences of the region's pixel values between the neighboring two frames . We then use a predetermined threshold for a motion activity value to determine whether a motion of the region is active or not. Finally, we combine the region's visual word and the activity information by leaving only the visual words where the motion is active. We call this remaining visual words "motion feature", and we again use the number of occurrences as a feature vector.

For the TRECVID 2007 run, we used a threshold of 20.0/255.0, which gave the best performance of 5-fold cross validation using the training dataset.

4 Feature extraction system

For each keyframe image, we concatenate a feature vector of visual words and a feature vector of motion features to get a feature vector for a classifier. With the feature vectors, we use a maximum entropy model (MEM) [1] to classify the presence/absence of each high-level feature. A MEM estimates the posterior distribution of label (presence or absence) given the features of a keyframe image. We use the implementation of MALLET [3].

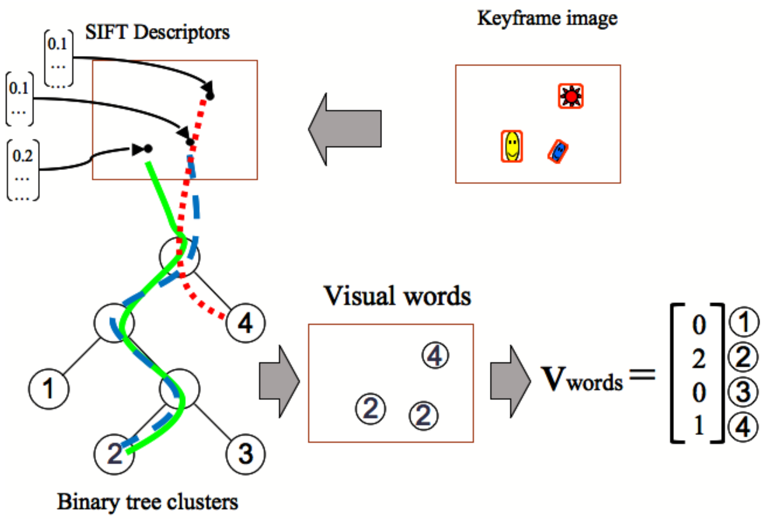

Figure 2: Quantization of SIFT descriptors using binary-tree-cluster.

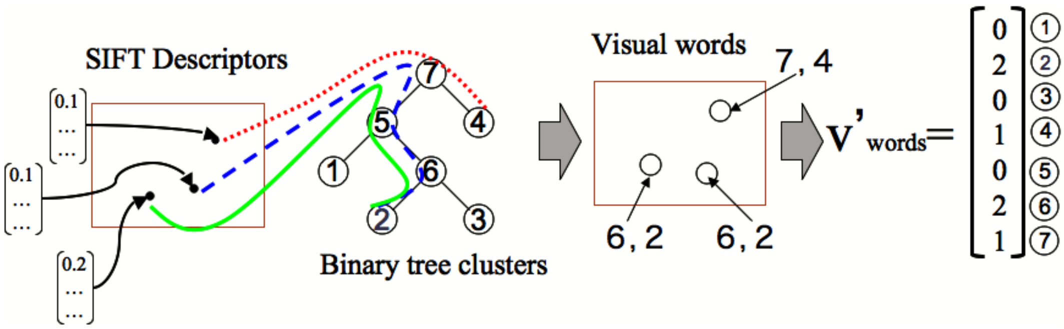

Figure 3: Example of visual words which use parent nodes.

5 Experiments

5.1 Experimental conditions

We used the TRECVID 2005 training data set to train MEMs. The A_Tok_1 run was trained on the complete training set using visual words and motion features. On the other hand, for A_Tok_2 run, we used only visual words. We constructed a codebook for each high-level feature by clustering SIFT descriptors of keyframe images where the high-level feature was present. When there existed more than 4,000 relevant keyframes, randomly sampled 4,000 keyframes were used. We also constructed a codebook which is shared among all the high-level features. To construct this codebook, we randomly sampled 10,000 keyframes and clustered their descriptors.

5.2 Results

Because there was a material mistake in our submission, the infAPs of our TRECVID 2007 runs were almost zero. Therefore, in this notebook, we also show the results evaluated by ourselves with the truth judgments file. Figure 5 shows the classification performance of our run A_tok_1 and re-evaluated run, together with the median and max performance of all the participants. Most of our performance is below the median.

We also show the results of three experiments evaluated with the 5-fold cross validation on the TRECVID 2005 development set. In the first experiment, we examined the performance of the shared

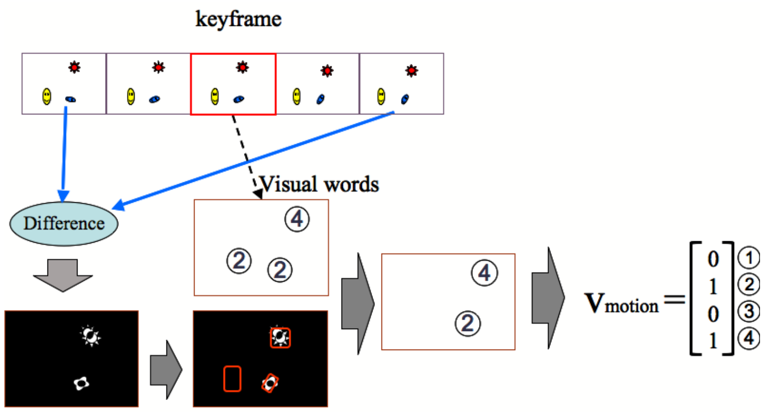

Figure 4: Motion feature.

codebook and the separate codebooks by changing their codebook sizes (Table 1). When a size of a codebook is small (500, 1000 leaves), separate codebooks perform better. This should be because the separate codebook is specialized for the corresponding high-level feature. However, with a larger codebook size (3000 leaves), shared codebooks outperform separate codebooks.

Next, we examined the effectiveness of using the parent node of each leaf node. Table 3 shows performance when we change the codebook size and nodes of tree-cluster codebook used for feature vectors. Leaf depth 1 means that only leaves are used for feature vectors, depth 2 means leaves and their parents are used, depth 3 means leaves, parents, and grandparents are used. We can see that there is no clear effect of leaf depth. Therefore, it is reasonable to choose the leaf depth for each high-level feature automatically using the training data.

Next, we examined the effectiveness of the motion features. For this experiment, we used the shared codebook of 3000 leaves. Figure 2 shows the performance of 39 high-level features using three types of feature vectors: visual word only (Word), motion feature only (Motion), and both (Word+Motion). The mean average precision of Word+Motion improved by 0.8% from Word. We can see that Word+Motion works well with most of the high-level features, but not so well for some high-level features like person or face. Therefore, like the choice of nodes, it is reasonable to choose the type of feature vectors for each high-level feature automatically using the training data. Figure 6 shows the performance of four selected high-level features using the three types of feature vectors. Word+Motion gave the best performance for all of the four high-level features.

References

[1] A. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, Vol. 22(1), pp. 39–71, 1996.

[2] S. Agarwal and D. Roth, "Learning a sparse representation for object detection," In *ECCV*, 113130, 2002.

[3] McCallum and Andrew Kachites, "MALLET: A Machine Learning for Language Toolkit", http://mallet.cs.umass.edu, 2002.

[4] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky. T. Kadir, and L. Van Gool, L. "A Comparison of Affine Region Detectors," In *IJCV*, 43-72, 2005.

Table 1: Performance of a shared codebook and separate codebooks with different codebook sizes

	Shared codebook				Separate codebooks			
	3000	2000	1000	500	3000	2000	1000	500
sports	**0.242**	0.186	0.136	0.166	**0.222**	0.219	0.212	0.134
entertainment	0.224	0.280	**0.282**	0.261	0.211	0.266	**0.290**	0.274
weather	**0.452**	0.425	0.381	0.254	0.400	**0.431**	0.430	0.364
court	0.065	**0.076**	0.051	0.025	0.022	0.045	**0.061**	0.041
office	**0.058**	0.041	0.019	0.026	**0.056**	0.029	0.017	0.013
meeting	**0.072**	0.056	0.060	0.069	0.059	0.050	0.074	**0.072**
studio	**0.560**	0.517	0.551	0.559	0.582	0.548	0.569	**0.598**
outdoor	0.402	**0.423**	0.414	0.418	0.412	**0.427**	**0.427**	0.424
building	0.094	0.092	0.108	**0.109**	0.067	0.075	**0.111**	0.110
desert	**0.037**	0.031	0.029	0.011	**0.027**	0.018	0.011	0.008
vegetation	**0.049**	0.034	0.046	0.046	0.039	0.028	**0.040**	0.038
mountain	**0.099**	0.095	0.068	0.037	0.076	**0.086**	0.053	0.000
road	0.085	0.071	0.093	**0.094**	0.080	0.062	0.082	**0.086**
sky	0.182	0.197	0.263	**0.276**	0.165	0.172	0.268	**0.286**
snow	0.259	**0.260**	0.224	0.149	0.242	**0.265**	0.238	0.163
urban	0.082	0.056	0.088	**0.096**	0.063	0.047	0.085	**0.089**
waterscape_waterfront	**0.223**	0.170	0.091	0.082	**0.215**	0.177	0.099	0.000
crowd	0.248	0.306	0.348	**0.368**	0.225	0.272	0.355	**0.364**
face	0.800	**0.813**	0.817	0.788	0.811	0.836	**0.840**	0.826
person	0.865	**0.870**	0.869	0.853	0.873	**0.884**	0.876	0.869
government_leader	0.086	0.067	**0.103**	0.085	0.103	0.081	**0.116**	0.103
corporate_leader	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
police_security	**0.044**	0.038	0.005	0.007	**0.046**	0.023	0.010	0.012
military	0.073	0.060	0.073	**0.084**	0.051	0.046	0.069	**0.080**
prisoner	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
animal	**0.355**	0.309	0.167	0.064	**0.320**	0.312	0.181	0.072
computer_tv_screen	**0.172**	0.150	0.144	0.164	0.164	0.156	0.158	**0.183**
flag_us	**0.133**	0.118	0.089	0.037	0.059	0.088	**0.101**	0.091
airplane	**0.106**	0.086	0.066	0.038	**0.092**	**0.092**	0.089	0.059
car	0.152	0.111	0.150	**0.155**	0.134	0.115	**0.180**	0.160
bus	0.103	**0.125**	0.081	0.045	**0.101**	**0.101**	0.099	0.070
truck	0.117	**0.128**	0.102	0.052	**0.086**	0.083	0.070	0.035
boat_ship	**0.141**	0.099	0.086	0.030	0.149	**0.158**	0.102	0.043
walking_running	0.059	0.047	0.065	**0.069**	0.041	0.039	**0.050**	0.000
people_marching	**0.176**	0.175	0.135	0.098	**0.149**	0.133	0.109	0.050
explosion_fire	**0.043**	0.037	0.036	0.015	0.027	**0.031**	0.020	0.009
natural_disaster	0.150	0.150	**0.203**	0.134	0.175	0.175	**0.189**	0.120
maps	**0.281**	0.259	0.196	0.144	0.197	**0.212**	0.205	0.153
charts	**0.480**	0.467	0.472	0.453	**0.483**	0.475	0.476	0.375
Mean Average Precision	**0.210**	0.201	0.192	0.172	0.195	0.196	**0.199**	0.187

Table 2: Performance of the 39 high-level features using three types of feature vectors: visual word only (Word), motion feature only (Motion), and both (Word+Motion).

	Word	Motion	Word+Motion
sports	0.242	0.086	**0.266**
entertainment	0.224	0.108	**0.246**
weather	**0.452**	0.043	0.451
court	0.065	0.004	**0.068**
office	0.058	0.016	**0.070**
meeting	0.072	0.006	**0.102**
studio	0.560	0.125	**0.607**
outdoor	**0.402**	0.274	0.369
building	0.094	0.028	**0.099**
desert	0.037	0.003	**0.038**
vegetation	0.049	0.014	**0.064**
mountain	0.099	0.012	**0.107**
road	0.085	0.043	**0.112**
sky	0.182	0.059	**0.221**
snow	0.259	0.124	**0.276**
urban	0.082	0.016	**0.101**
waterscape_waterfront	0.223	0.061	**0.245**
crowd	0.248	0.139	**0.295**
face	**0.800**	0.445	0.747
person	**0.865**	0.645	0.809
government_leader	0.086	0.023	**0.104**
corporate_leader	0.000	0.000	0.000
police_security	**0.044**	0.010	0.037
military	0.073	0.024	**0.079**
prisoner	0.000	0.000	0.000
animal	0.355	0.102	**0.356**
computer_tv_screen	0.172	0.017	**0.203**
flag_us	**0.133**	0.004	0.122
airplane	0.106	0.043	**0.129**
car	0.152	0.094	**0.207**
bus	0.103	0.000	**0.105**
truck	0.117	0.012	**0.132**
boat_ship	**0.141**	0.049	0.130
walking_running	0.059	0.043	**0.070**
people_marching	**0.176**	0.052	0.172
explosion_fire	0.043	0.009	**0.060**
natural_disaster	0.150	0.053	**0.156**
maps	**0.281**	0.001	0.248
charts	0.480	0.141	**0.488**
MeanAveragePrecision	0.199	0.075	**0.207**

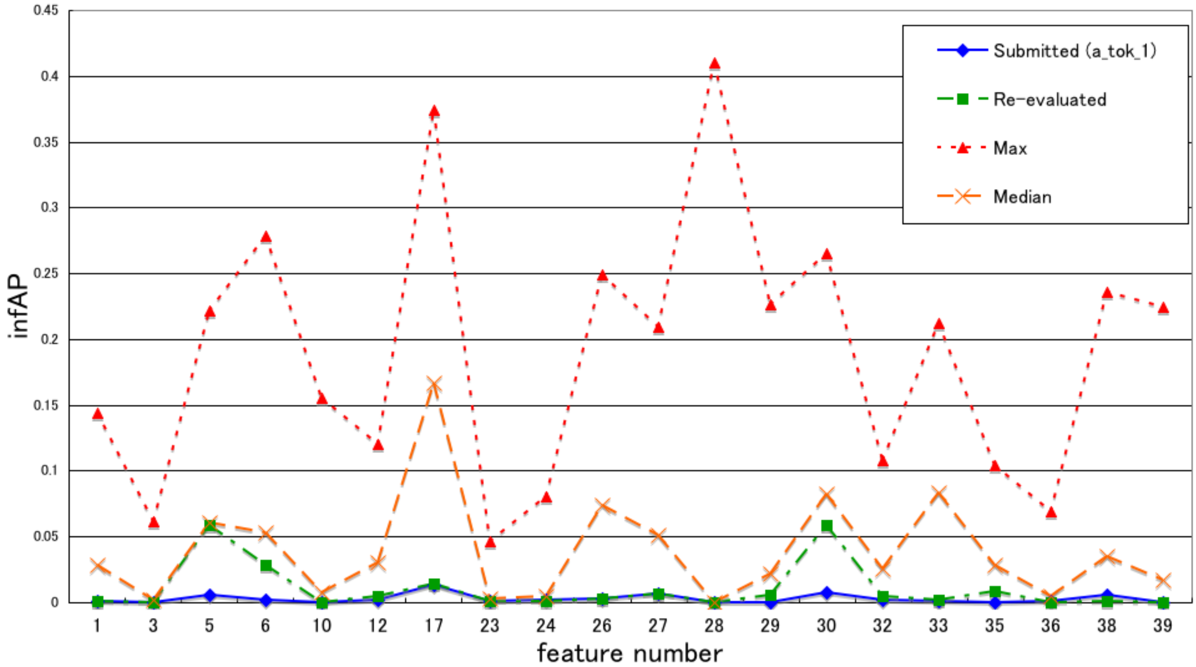

Figure 5: Performance of our run A_tok_1 and re-evaluated run, together with the median and max performance of all the participants.

leaf depth / codebook size	animal			car			walking_running			building		
	1	2	3	1	2	3	1	2	3	1	2	3
3000	**0.355**	0.323	0.311	**0.156**	0.147	0.143	**0.054**	0.052	0.050	**0.087**	0.071	0.080
2000	**0.309**	0.272	0.236	0.109	0.109	**0.119**	**0.047**	0.041	0.044	0.092	0.092	**0.105**
1000	**0.167**	0.150	0.132	0.150	**0.158**	0.157	0.065	0.067	**0.070**	0.107	**0.108**	0.106
500	0.064	0.064	**0.067**	0.150	**0.154**	0.141	**0.069**	0.063	0.062	**0.109**	**0.109**	0.104

Table 3: Performance when we change the codebook size, and the nodes of tree-cluster codebook used for feature vectors. Leaf depth 1 means that only leaves are used for feature vectors, depth 2 means leaves and parents of leaves are used, and depth 3 means grandparents of leaves are also used.

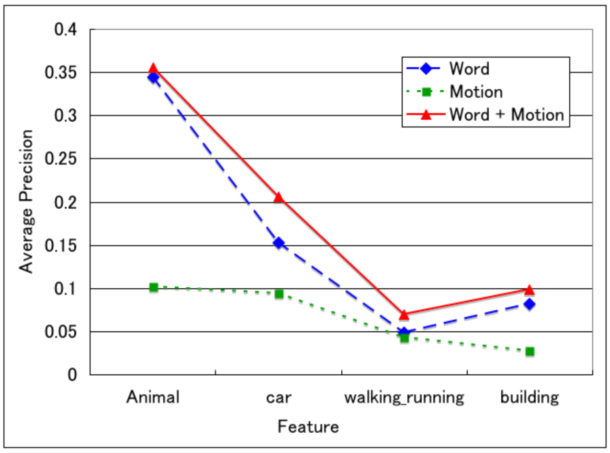

Figure 6: Performance of four selected high-level features using the three types of feature vectors. Word+Motion gave the best performance for all of the four high-level features.