JOANNEUM RESEARCH and Vienna University of Technology at TRECVID 2011: Semantic Indexing and Instance Search

Werner Bailer*, Robert Sorschag†, Felix Lee*, Harald Stiegler*, Georg Schwendt*
*JOANNEUM RESEARCH, DIGITAL – Institute for Information and Communication Technologies
8010 Graz, Austria
Email: {firstName.lastName@joanneum.at}
†Vienna University of Technology, Interactive Media Systems Group
1040 Vienna, Austria
Email: sorschag@ims.tuwien.ac.at

ABSTRACT

We participated in two tasks: semantic indexing (SIN) and instance search (INS).

SIN runs

We submitted 4 light runs, using 4 MPEG-7 visual features and different kernels (on single frames and sequences). Runs use a different subsampled set of training data.

- L_A_JRS-VUT1_3: 500 training samples/concept, single frames
- L_A_JRS-VUT3_1: 500 training samples/concept, sequence of vectors
- L_A_JRS-VUT4_4: 1000 training samples/concept, single frames
- L_A_JRS-VUT5_2: 1000 training samples/concept, single frames, post-processing using semantic relations

Increasing the number of training samples increases the results only moderately, post-processing using semantic relations does not improve the results.

INS runs

We submitted 4 runs, using different (combinations) of features:

- F_X_NO_JRS_4: Baseline run: SIFT only
- F_X_NO_JRS_3: Fusion of all features: the best general configuration of each feature is used for all queries
- F_X_NO_JRS_2: Best single feature for each query using auto-selection
- F_X_NO_JRS_1: Fusion of best feature configurations for each query using auto-selection

The SIFT only run outperformes the fused runs. Fusion includes more relevant clips, but the SIFT only run has more top-ranked relevant clips.

I. SEMANTIC INDEXING

For the semantic indexing task we use a set of low-level features extracted from keyframes and train a classifier for each concept using SVMs. In the following, we briefly describe the features used, and the kernels we used in the experiments. We then discuss our results.

A. Features

1) MPEG-7: The following MPEG-7 [1] image features were extracted globally:

Color Layout describes the spatial distribution of colors. This feature is computed by clustering the image into 8x8 blocks and deriving the average value for each block. After computation of DCT and encoding, a set of low frequency DCT components is selected (6 for the Y, 3 for the Cb and Cr plane).

Dominant Color consists of a small number of representative colors, the fraction of the image represented by each color cluster and its variance. We use three dominant colors extracted by mean shift color clustering [2].

Color Structure captures both, color content and information about the spatial arrangement of the colors. Specifically, we compute a 32-bin histogram that counts the number of times a color is present in an 8x8 windowed neighborhood, as this window progresses over the image rows and columns.

EdgeHistogram represents the spatial distribution of five types of edges, namely four directional edges and one non-directional edge. We use a global histogram generated directly from the local edge histograms of 4×4 sub-images.

2) Bag of features (BoF): About 300 densely sampled image regions from 3 different scales are selected per keyframe. A 128 dimensional SIFT descriptor (4×4 subregions, 8 directions for orientation histograms) is extracted for each of these regions without computation of a dominant orientation. These SIFT descriptors are robust to certain kind of scale changes but the robustness to orientation changes is poor. We high cap and normalize the features as described in [3]. We extract MPEG-7 ColorLayout features from these regions. ColorLayout features [4] present the spatial distribution of colors in a very compact form. These features cluster an image or image region into sub-regions of 8×8 pixels and compute

the average pixel value for each of them to select the first low frequency coefficients of a discrete cosine transform.

Higher-level features are then generated by the popular bag-of-feature approach (BoF) where the SIFT and ColorLayout features are mapped to codewords. These codewords are generated in an off-line step using the k-means algorithm on about 100,000 features from randomly selected Flickr images. We use codebooks with 100 codewords which leads to two 100 dimensional BoF features for each keyframe. Each entry in one of these BoF features states the number of times a specific codeword was detected in a keyframe. The mapping between SIFT and ColorLayout features from a keyframe and their codewords is identified by nearest neighbor search with Euclidean distance. Beside global BoFs of the entire keyframes, we generated further versions where the keyframes are split into 2×2, 1×3, 3×1, and 3×3 regions in horizontal and vertical direction. A own 100 dimensional BoF feature is then generated for each partition and they are concatenated to 300, 400, and 900 dimensional features.

B. Kernels

Kernel methods, most notably Support Vector Machines (SVMs), have been widely applied to classification problems, also due to the availability of toolkits such as LibSVM [5]. SVM based classifiers are also commonly used for concept classification based on visual features. If we look at the TRECVID [6] 2009 High-Level Feature Extraction (HLFE) Task, all but 3 of the 42 submitters report the use of an SVM variant in some part of their approach (e.g. for classification based on low-level features or for fusion) [7]. Most of the groups use some low-level features which require other distances than the Euclidean distance between feature vectors, e.g. some of the MPEG-7 visual descriptors [4] or variants of histograms. But only about half of these groups mention the use of specific kernels for these features, while most seem to use the commonly applied radial basis function (RBF) kernel.

Despite the wide use of MPEG-7 visual features in the research community there is remarkably little work on defining kernels that appropriately model the proposed distance functions. A kernel combining different MPEG-7 features and considering the appropriate distance functions has been proposed in [8], [9] and has shown to perform better on a small still image data set.

We thus define a kernel that combines appropriate kernels for the different features. The kernel is defined as

$$\kappa_{combined}(x, x') = \kappa_{mpeg7}(x, x')\kappa_{bof}(x, x'), \qquad (1)$$

where κ_{mpeg7} is the kernel for MPEG-7 features described in [10]:

$$\kappa_{mpeg7}(x, x') = \prod_{i \in \{cld, dcd, csd, ehd\}} \exp(-\bar{w}_i \kappa_i(x, x')). \qquad (2)$$

The feature weights w_i are defined as

$$w_i(T) = \frac{\mathrm{var}(\{d_i(x_i^-, y_i^-) | \forall x^-, y^- \in T^-\})}{\mathrm{var}(\{d_i(x_i^+, y_i^+) | \forall x^+, y^+ \in T^+\})}, \qquad (3)$$

where x^+ (x^-) denotes a positive (negative) sample in the training set T and $d_i(\cdot)$ is the distance function for feature i. The weights are thus defined as the ratio of the variances of the feature distances among the negative and positive samples. The weights for the individual features are then normalized to obtain $\bar{w}_i = \frac{w_i}{\sum_{j \in \{cld, dcd, csd, ehd\}} w_j}$. In contrast to [8] we calculate the weights in advance and not iteratively during training.

κ_{bof} is a histogram intersection kernel

$$\kappa_{bof}(x, x') = \sum_{j=1}^{n} \min(x_i, x_i'), \qquad (4)$$

with n being the size of the BoF vocabulary.

As an alternative, we use a kernel that supports a sequence of such feature vectors, i.e. uses the features of multiple key frames per shot together. The individual feature vectors $x_1, \ldots, x_n$ of a shot are concatenated to a sequence feature vector X.

A kernel based on the longest common subsequence (LCSS) algorithm has been proposed in [10]. This kernel already allows plugging in any kernel for measuring the distance between the feature vectors of the samples of the two sequences, and includes the similarities in the result of the kernel. The kernel uses a recursive definition of LCSS and a threshold θ to decide if two feature vectors are considered as matching. The kernel function to determine the length of the single longest common subsequence is given as $\kappa_{\mathrm{LCSS}} = \mathrm{LCSS}(X, X')$. Similarity weighting can be achieved by performing backtracking of the longest sequence, summing the values of $\kappa_{combined}(\cdot)$ of the matches and normalizing.

In [10] the authors propose to consider all subsequences ending in the last element of either of the two sequences:

$$\kappa_{\mathrm{ALCSS}} = \begin{array}{l} \sum_{i=m}^{1} \mathrm{LCSS}((x_1, \ldots, x_i), X') + \\ \sum_{j=n-1}^{1} \mathrm{LCSS}(X, (x_1', \ldots, x_j')). \end{array} \qquad (5)$$

This requires backtracking of all sequences ending in the last element of either X or X', i.e., $O(n^2)$ backtracking steps. The underlying distance function of this kernel can also include constraints for the minimum lengths of subsequences considered matching and the maximum gap between two matching elements in a sequence. However, [10] does not use these constraints. The result of the kernel function is normalized to account for sequences of different lengths.

C. Using Semantic Relations

We have made use of the semantic relations between the concepts as a post-processing step. Only 37 *implies* relations apply to the concepts the light task. The following rules are applied to the set of predictions for the concepts for each shot (the probabilities from the prediction step are assumed to

be normalized s.t. the decision boundary for concept A is at $p(A) = 0.5$).

First, possible transitive relations are resolved, and added to the set of relations to be processed. Then, each relation between concepts A and B is processed depending on the predictions for these concepts for a shot:

a) Consistent prediction for concepts A and B: $p(B) = \max(\min(p(B) + 0.5(p(A) - 0.5), 1.0), 0.0)$

b) Predictions of A and B are contradicting the relation: We implemented three options: (i) optimizing recall, i.e. enforcing the relation, (ii) optimizing precision, i.e. removing the prediction that corresponds to the condition in the relation, and (iii) deciding based on the relative margin of the two predictions to the decision boundary.

There are cases where two relations have the same target, so that potentially concurrent updates could happen. We implemented different strategies to combine these updates as their minimum, maximum, mean or median. However, no differences in terms of the results have been observed.

D. Results

We varied the following parameters in our runs:

- training set: TV10 (IACC.A only) or TV11 (IACC.A and IACC.1)
- subsample of training set: random sample of the training set with at most k samples per concept
- features: MPEG-7 features only or MPEG-7 features and global BoF
- kernel: single feature vector or sequence of feature vectors per shot
- semantic relations: post-process results using semantic relations

Table I shows an overview of the official and additional SIN runs and their parameters, as well as the mean infAP over all concepts of the light task.

Note that the result for the sequence kernel are incorrect due to an error in the test data file. This will be corrected in the final version of the paper.

The use of the global BoF features in addition did not improve the results (in contrast to TV10 experiments). This will be further investigated, as well as the use of block-based BoF features.

E. Conclusion

Training the classifier on the same number of samples of the IACC.A data set only yields slightly worse results than sampling from both IACC.A and IACC.1.

There is an improvement of the results with increasing number of samples from the training set. However, the improvement is only moderate.

The use of semantic relations did not improve the results. Some concepts were unaffected, for others the results got worse. It seems that the probabilities generated by the SVM are not appropriate for using them to compare/adjust reliabilities of the different concepts in a post-processing step.

II. Instance Search

Our system for the instance search task is structured as follows. We use four different subsystems, each performing the search task for a certain type of feature with different configurations. A couple of different dissimilarity measures [11] (Canberra distance, Correlation distance, Cosine distance, Jeffrey divergence, Squared distance, PsiSquare statistics, and the Minkowski family distances: Manhattan distance, Euclidian distance, and Fractional distance) are used in the matching step for each feature. We have used the entire query images for feature extraction as well as the cropped objects without background. This leads to a total number of 18 different configurations per feature type. Each subsystem is queried independently with each sample and returns a ranked result list with a similarity value for each result. We thus have for each query *number of samples* $\times$ *number of subsystems* results. Different fusion methods were used to combine the results of one module (that stem from different images of the same query) and to combine the results of different modules in order to obtain the final ranked result list.

All features have been extracted from a set of clustered key frames per clip. This clustering was performed as follows. First, densely sampled key frames have been extracted using every tenth frame. Color bars, black and white frames have been detected and omitted by testing the variances of the pixel columns. Then we extracted global SIFT features from each of these key frames and matched them against each other using Euclidian distance. Key frames that exceed an initial threshold of 0.5 against all key frames of the existing clusters are used to generate a new cluster. After all key frames are processed the number of clusters is evaluated and if only one cluster exists or if the number of clusters is higher than six, new clusters are generated with an adaptive threshold (multiplied by 0.95 or 1.05). A total number of 69,591 key frames is generated from all clips.

In the following, we describe the subsystems for the different features and the fusion methods. We then discuss the results.

A. Subsystems

1) Gabor Face Recognition: We perform face detection using the well-known Viola-Jones AdaBoost approach [12] on three scaled versions of each keyframe with the longer image side sizes of 160, 320, and 640 pixels. Polygon intersection ensures that each face region is not used from multiple scales. After this, every detected face region is processed to generate a 10,240 dimensional feature using Gabor wavelets.

The Gabor features of all trainings key frames are generated and stored in an off-line step. During instance search, a linear k-nearest neighbor search [13] is performed for each detected face in a query image to identify the best matches in the database using the dissimilarity measures mentioned above. For query images where no face was detected, an empty result lists is generated. Our implementation is done in C++ using the CORI recognition infrastructure [14], the OpenCV library [15]

id	training set	no. samples	features	kernel	relations	infAP
1	TV11	500	MPEG-7	single	no	0.0102
2	TV10	500	MPEG-7	single	no	0.0079
3	TV11	500	MPEG-7	sequence	no	0.0040
4	TV11	1000	MPEG-7	single	no	0.0105
5	TV11	1000	MPEG-7	single	yes, based on run 4 (relative)	0.0043
5a	TV11	1000	MPEG-7	single	yes, based on run 4 (precision)	0.0040
5b	TV11	1000	MPEG-7	single	yes, based on run 4 (recall)	0.0040
5e	TV11	1000	MPEG-7	single	yes, based on run 6 (relative)	0.0047
6	TV11	5000	MPEG-7	single	no	0.0118
7	TV11	500	MPEG-7, global BoF	single	no	0.0046

TABLE I
OVERVIEW OF SIN RUNS (1,3,4,5 ARE THE OFFICIAL RUNS).

for face detection, and FFTW library [16] for Gabor feature generation.

2) BoF Matching: BoF Features are generated in the same way as described in the SIN section I-A2 using densely sampled regions from three different scales, SIFT and ColorLayout features, and codebooks with 100 clusters that are generated using the k-means algorithm. In an offline process, global BoFs are then generated for both feature types from every clustered key frame. BoF matching is then performed with k-NN search of the BoFs from each query image against the BoFs from the clustered key frames. In addition to the above mentioned dissimilarity measures, we compute the histogram intersection between a query BoF Q and a clustered key frame BoF T:

$$d = \sum_{i=0}^{i<cbSize} \max\left(Q\left(i\right), T\left(i\right)\right) - T\left(i\right) \tag{6}$$

where i is the current codebook index. This histogram intersection was used because the (cropped) query images contain only the query object while additional background objects can be shown in the clustered key frames.

3) Mean Shift segments: A Mean Shift segmentation was performed to generate about 10 segments per clustered key frame and query image. We use the Mean Shift implementation proposed in [17], which works on a quantized color space. The resulting region of the segmentation are merged as follows. Starting from the smallest region, we merge a region with its largest neighbor, until either the number of regions is below 10 or the size of the smallest region's width, height or area is above $0.02\times$ the image width, height or area. One SIFT feature and one ColorLayout feature have been extracted from each segment and k-NN search was used again for matching.

4) SIFT: SIFT [3] is used as another subsystem in the instance search task. For each query versus database image match, SIFT keypoints and its corresponding descriptors are extracted as proposed in [3]. The descriptors are computed using 4×4 cells and 8 bins for histogram of oriented gradients. Since some queries provide only few keypoints reside in the object mask, we extent the query keypoint set by such keypoints extracted from the neighboring regions around. In particular, if the object mask covers less than 10 keypoints, we fill up the set with keypoints next to the mask boundary. A special treatment to find objects of the type 'person' is that we use the entire SIFT keypoints including those extracted from the background. The reason is that we do not expect that specific persons can be recognized using SIFT. But it is possible to find the target person in the same location or scene. For the search in the database image pool, each query descriptor is compared with all descriptors of each database image. Then, the top 5 matches are kept for further assessment. The score of a SIFT descriptor match is composed of two criteria: the actual descriptor match from query to database image and the consistency of their neighboring keypoints. For that, the coordinates of (at most 10) neighboring keypoints are projected from query image to database image according to the orientation and the scales of both center descriptors. Then the keypoints in database image next to these coordinates are taken into account. That is, the histogram distance between the center keypoints and their neighboring keypoint matches are used in the following score function

$$sc(dist) = -\sum_i \log\left(dist_i + 1\right)/maxDist, \tag{7}$$

where $maxDist = 10^6$ and matches with distances greater than $maxDist$ are discarded. Note that the score function prefers single good match over several moderate matches. Finally, the maximum of the scores is used as rank in the result list.

B. Fusion methods

As mentioned above, each INS module generates one output list for each sample image. Thus, multiple output lists exist for each query and module that have to combined first. Secondly, we had to to merge the output lists of different modules for combined runs, as described in the next section. For the sake of simplicity, we used following fusion strategy for both tasks.

$$rating_i = best_i * \#lists * \frac{\#entries_i}{\#possibleEntries} - best_i \tag{8}$$

Thereby, we compute a ranking value for each object instance i contained in the results lists based on the best position of one of its samples. Depending on the recognition approach, multiple entries of the same instance in one result list are possible or not. After this, we order all instances again

according this ranking value. The intuition behind this fusion is to order the instances similar to a zip fastener starting from their best entries. The order of entries with the same best value (originating from different result lists) is thereby determined from the number of entries.

C. Runs

Following four different runs have been performed: (1) a baseline run with SIFT descriptors only, (2) fusion of all modules, the best general configuration of each module is used for all queries, (3) best single feature for each query, (4) fusion of best module configurations for each query. The best features of run three and four are thereby calculated with the auto-selection approach of [18]. In this process, we compared the results of different sample images of the same query against each other using all sub-modules, extraction types, and dissimilarity measure. This auto-selection identifies the position of the other sample images of the same query for each configuration and selects the one(s) where the sample images are ranked best. We suppose that recognition approaches that perform well to match the sample images of the same query against each other, are well suited to retrieve further instances of the same object.

D. Results

The best mean precision (0.221) and mean average precision (0.170) was achieved by the first run with SIFT descriptors only. These results are above the median for almost all objects but below the top results. However, the results between the four runs contains some surprises. The runs where only one visual feature was used (run one and three) returned 611 and 613 relevant shots from a total amount of 1830. However, while the first run achieved the highest mean precision, the opposite is true for run three while run. The fused runs (run two and four) returned more relevant shots (847 and 783) but their mean precision and mean average precision values are significantly worse that the baseline runs. According to the 'precision at n shots' statistics, these runs outperform the baseline run between 200 and 500 shots while a much lower precision is achieved in the first 30 shots. The run times are similarly high in all queries and runs where SIFT features are involved (between 100 and 300 minutes) while analysis times of a few minutes or even seconds are measured for all other queries.

It must be noted that while the results of the fused runs are rather balanced for the different types of queries (at least for object and location), the SIFT only run scores significantly better for locations (0.3709) than for the other categories.

E. Conclusion

As in our TV10 experiments the SIFT only run outperformed the different fusion methods. The fused runs return more relevant results in total, however, the SIFT only run has more relevant results on the lower ranks. This property makes it probably more suitable for practical retrieval problems.

ACKNOWLEDGMENTS

The authors would like to thank Christian Schober, Georg Thallinger, Werner Haas and Horst Eidenberger for their feedback and support.

The research leading to these results has received funding from the European Union's Seventh Framework Programme under the grant agreements no. FP7-215475, "2020 3D Media – Spatial Sound and Vision" (http://www.20203dmedia.eu/) and no. FP7-248138, "FascinatE – Format-Agnostic SCript-based INterAcTive Experience" (http://www.fascinate-project.eu/), as well as from the Austrian FIT-IT project "IV-ART – Intelligent Video Annotation and Retrieval Techniques".

REFERENCES

[1] "Information technology-multimedia content description interface: Part 3: Visual," ISO/IEC 15938-3, 2001.

[2] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[4] B. Manjunath, J.-R. Ohm, V. Vasudevan, and A. Yamada, "Color and texture descriptors," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 6, pp. 703 –715, Jun. 2001.

[5] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[6] A. F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and TRECVid," in *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*. New York, NY, USA: ACM Press, 2006, pp. 321–330.

[7] W. Kraaij and G. Awad, "TRECVID-2009 high-level feature task: Overview," http://www-nlpir.nist.gov/projects/tvpubs/tv9.slides/tv9.hlf.slides.pdf, 2009.

[8] D. Djordjevic and E. Izquierdo, "Relevance feedback for image retrieval in structured multi-feature spaces," in *MobiMedia '06: Proceedings of the 2nd international conference on Mobile multimedia communications*. New York, NY, USA: ACM, 2006, pp. 1–5.

[9] ——, "Kernels in structured multi-feature spaces for image retrieval," *Electronics Letters*, vol. 42, no. 15, pp. 856 – 857, 20 2006.

[10] W. Bailer, "A feature sequence kernel for video concept classification," in *Proceedings of 17th Multimedia Modeling Conference*, Taipei, TW, Jan. 2011.

[11] H. Liu, D. Song, S. Rger, R. Hu, and V. Uren, "Comparing dissimilarity measures for content-based image retrieval," in *Aisa Information Retrieval Symp*, 2008, pp. 44–50.

[12] P. A. Viola and M. J. Jones, "Rapid object detection using a boosted cascade of simple features," in *CVPR (1)*, 2001, pp. 511–518.

[13] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *VISSAPP (1)*, 2009, pp. 331–340.

[14] R. Sorschag, "Cori: A configurable object recognition infrastructure," in *Int. Conf. on Signal and Image Processing Applications*, 2011.

[15] "OpenCV library," http://opencv.willowgarage.com/.

[16] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005, special issue on "Program Generation, Optimization, and Platform Adaptation".

[17] W. Bailer, P. Schallauer, H. B. Haraldsson, and H. Rehatschek, "Optimized mean shift algorithm for color segmentation in image sequences," in *Image and Video Communications and Processing*, ser. Proc. of SPIE-IS&T Electronic Imaging, A. Said and J. G. Apostolopoulos, Eds., vol. 5685. San Jose, CA, USA: SPIE, Jan. 2005, pp. 522–529. [Online]. Available: http://www.joanneum.at/uploads/tx_publicationlibrary/img2457.pdf

[18] R. Sorschag, "How to select and customize object recognition approaches for an application?" in *Int. Conf. on MultiMedia Modeling*, 2012.