JOANNEUM RESEARCH and Vienna University of Technology at TRECVID 2010

Werner Bailer, Roland Mörzinger, Stefan Hölzl, Felix Lee, Harald Stiegler
JOANNEUM RESEARCH
DIGITAL – Institute of Information and Communication Technology
8010 Graz, Austria
Email: {firstName.lastName@joanneum.at}

Robert Sorschag
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, 2 with RBF kernel, 2 with a kernel combining appropriate kernels for the different features. Two runs were trained on the 2010 training set, two on the 2007 training set (for the 3 concepts shared between 2007 and 2010).

- L_A_JRS-VUT1_2: RBF kernel trained on 2010 set.
- L_A_JRS-VUT2_1: Combined kernel trained on 2010 set.
- L_B_JRS-VUT3_4: RBF kernel trained on 2007 set.
- L_B_JRS-VUT4_3: Combined kernel trained on 2007 set.

The combined kernel outperforms the RBF kernel on the 2010 data. For the RBF kernel, training on 2007 data yields worse results, for the combined kernel no clear trend can be seen.

INS runs

All runs use the same features and differ by the method for fusing and ranking results from these features.

- F_X_NO_JRS_max_max_4: For each shot, maximum similarity of features of all query samples.
- F_X_NO_JRS_topK_4: Top-k results for each feature ($k = 1000/nFeatures$).
- F_X_NO_JRS_w_bestR_2: Weighted linear combination of feature similarities, weights based on best ranked other query sample.
- F_X_NO_JRS_w_t100_3: Weighted linear combination of feature similarities, weights based on number of other query samples among top 100.

Features worked best for object queries, weighted fusion was better. For persons and objects a single feature outperformed the best fused result, for other types fused results were better than any single feature.

I. Semantic Indexing

For the semantic indexing task we use a set of low-level features extracted from key frames 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 4x4 sub-images.

2) Gabor Energy: This descriptor is computed by filtering the image with a bank of orientation and scale sensitive filters and calculating the mean and standard deviation of the filtered outputs in the frequency space. We applied a fast recursive Gabor filtering for 4 scales and 6 orientations.

3) Bag of features (BoF): Local Difference of Gaussian (DoG) points are extracted from each keyframe. To limit the number of DoG points per image a filter is used to retain only the 1000 DoG points with the highest contrast computed with respect to the neighboring pixels in scale space. Next, 128 dimensional SIFT descriptors (4x4 subregions, 8 directions for orientation histograms) are extracted for the filtered DoG points using the interest points scale information and the

dominant orientation (computed from the gradients of a small circular path around the interest point). We high cap and normalize the features as described in [3].

To generate more generalized, higher-level features we map each SIFT feature to codewords. These codewords were generated in an off-line step using k-means algorithm on about 100.000 features from randomly selected Flickr images. Here we use a codebook with 1000 codewords. Thus we get a 1000 dimensional BoF feature for each keyframe where each entry states the number of times a specific codeword was detected in this keyframe. To find the mapping between each SIFT feature from a keyframe and its codeword, we select the nearest neighbor in the codebook with Euclidean distance.

4) Number of faces: For each shot the *number of faces* is detected in the reference keyframe by using the face detection method implemented in OpenCV[1].

B. Combined kernel

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 [4]. SVM based classifiers are also commonly used for concept classification based on visual features. If we look at the TRECVID [5] 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) [6]. 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 [7] 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

$$\kappa_{combined}(x, x') = \prod_{i \in \{mpeg7, gabor, bof, nfaces\}} \kappa_i(x, x'), \quad (1)$$

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

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

The feature weights w_i are defined as

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

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_{gabor} and κ_{nfaces} are RBF kernels on the respective parts of the feature vector, and κ_{bof} is a histogram intersection kernel

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

with n being the size of the BoF vocabulary.

C. Results

We have submitted 4 runs, using two different kernels (radial basis function and the combined kernel described above) and training on the 2010 and 2007 training sets.

1) Differences between kernels: Our hypothesis is that the kernel, that combines the results of appropriate kernel functions for the different features performs better than the RBF kernel that treats all the input features stacked together in one feature vector. The results are shown in Figure 1. In fact, the combined kernel outperforms the RBF kernel for 6 out of 10 concepts in terms of infAP, and yields also better mean and median infAP over the 10 concepts. This confirms the results reported in [10] on the TRECVID 2007 HLFE data set, where also slightly more than 50% of the concepts yield better results with the MPEG-7 kernel than with the RBF kernel. The strong improvement for demonstration and protest seems to be an outlier. The concept telephone seems not to be represented by any of our features, as the results are very poor with both of the kernels.

2) Generalization over data sets: In order to investigate how well the kernels generalize we trained both on the three concepts that the 2007 and 2010 data sets have in common (bus, boat or ship, demonstration or protest). As one would expect, the results for the RBF kernel drop between 20% and 50% when trained on the 2007 data instead on the 2010 data. The results for the combined kernel are different: for one feature (demonstration or protest) the result is 94% worse than when trained on the 2010 data, however, for the others the increase is 167% (boat or ship) and 33% (bus) respectively. One cannot draw a general conclusion from this result, as there are specific issues with these features: For boat or ship the combined kernel scored worse than the RBF kernel on the 2010 data against the overall trend, while with training on the 2007 data the results for both kernels are the same, this relativates the very high increase. The definition

[1] http://sourceforge.net/projects/opencvlibrary

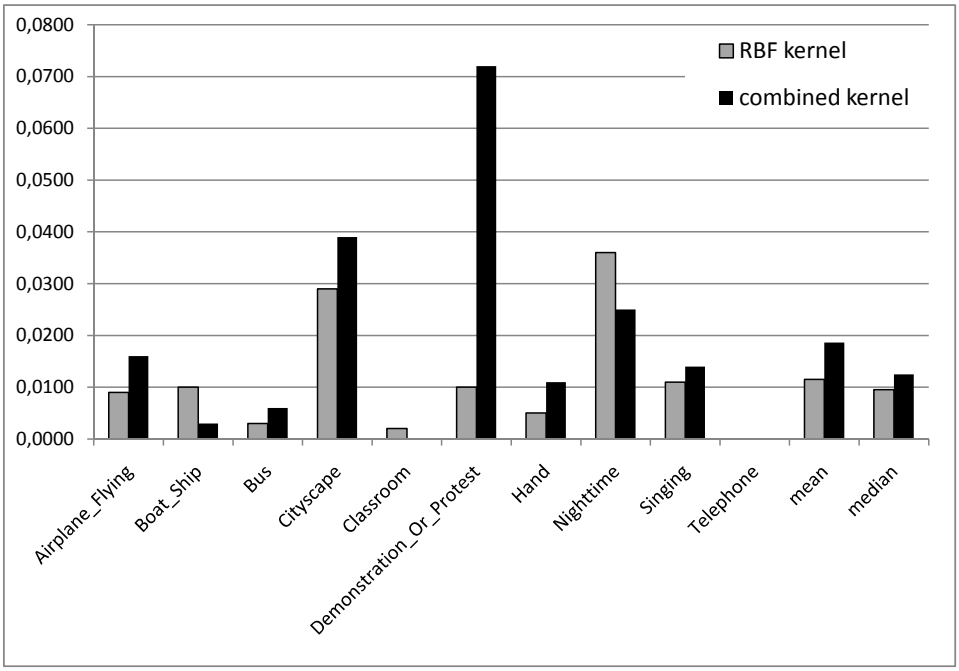

Fig. 1. Comparison of infAP of the two kernels on the 10 concepts of the light run.

of demonstration or protest has slightly changed from people marching in 2007, which might be one reason for the decrease in performance, the other being that the score for the combined kernel trained on the 2010 data is relatively high. However, the weights for the different MPEG-7 features determined as described above changed only moderately for this concept, while there are significant difference of the weights for the concept bus determined from the two training sets. For boat, one feature double it's weight, while the changes of the other feature are less significant.

D. Conclusion

The results show that it makes sense to use appropriate kernel functions for the features involved instead of the "off the shelf" RBF kernel. Although there is not an improvement for all concepts, the results are better for more then half of the concepts and also the median infAP improves. Concerning the generalization properties of the kernel across data sets it is unfortunately not possible to draw clear conclusions.

II. Instance Search

Our system for the instance search task is structured as follows. We use five different subsystems, each performing the search task for a certain type of feature. For bag of features, we use four different parameterizations, thus leading to eight different subsystems in total. Each subsystem is queried independently with each query 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. We use different methods to fuse these results in order to obtain the final ranked result list.

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 [11] in all keyframes. Every detected face region is further processed to generate a 10240 dimensional feature using Gabor wavelets. The Gabor features of all trainings keyframes (from the tv9.sv.test set) are generated and stored in an off-line step. This approach leads to a total number of 22856 face features in our database.

Whenever an instance search is performed, face detection and Gabor feature generation is done in all source-sample images of the query (without cropped objects). For each detected face, an approximated k-nearest neighbor search [12] is then performed to find the best matches in the database using Euclidean distance. The strategy for approximated k-nn search was selected automatically and uses linear feature algorithm in this case. As result of this subsystem a list with up to 2000 shots is generated, according to the keyframes where the matching faces were detected in. For queries where no face was detected, an empty result lists is generated there. We further utilize the queries meta information to use the Gabor Face Recognition subsystem only for 'actor' and 'character' queries.

Our implementation is done in C++ and uses the OpenCV library [13] for face detection, FFTW library [14] for Gabor feature generation, and the FLANN library [3] for approximated k-nn search.

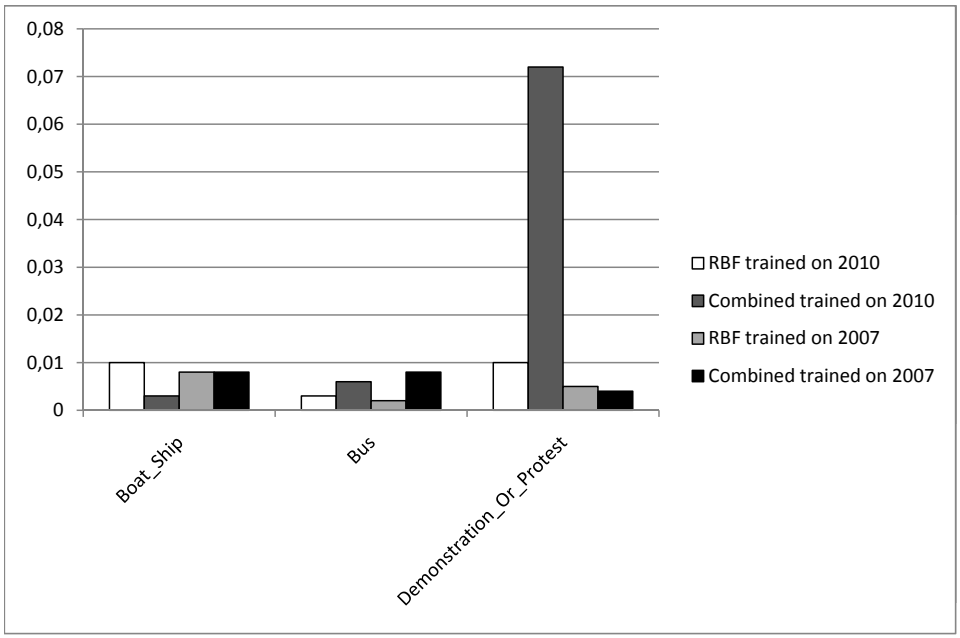

Fig. 2. Comparison of infAP of the three shared concepts for the different kernels trained on 2007 and 2010 data sets.

2) BoF Matching: BoF Features are generated in the same way as described in the SIN section I-A3 using filtered DoG interest points, SIFT features and codebooks that are generated with k-means in an offline step. In contrast to the SIN task, we generate two codebooks with a different number of codewords here. The first codebook has a size of 100 and the second a size of 1000. For both codebooks one BoF feature is generated for each keyframe in the test set, which leads to a database that consists of 153133 features for each codebook size. Furthermore, we have used two different approaches to generate BoF features for queries. On the one hand, we have used the whole source-sample images as input. On the other hand, we have used the cropped objects where no background is shown. The combination of different codebook sizes and different query images leads to 4 different configurations.

For BoF matching of each sample image from a query against the database we use a k-nn search. Thereby the distance between a query feature Q and a database test feature T was computed with the histogram intersection:

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

where i is the current codebook index. This histogram intersection was chosen because the query images (at least in the cropped BoF version) contain only the query object while the keyframes in the database that show the query object usually contain additional background objects.

3) 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 have been extracted. Only interest points inside the query object ROI are considered. For each query image descriptor its nearest neighbor (Euclidean distance) in the analyzed image is taken in order to vote in a 2D-histogram for the query's position (scale and orientation have been omitted). The 2D-histogram's width and height are ten times smaller than the original image's dimensions. If a bin in the 2D histogram has more than five votes, a hit has been assumed. The higher the histogram's peak, which means many consistent descriptor matches w.r.t. position, the higher the ranking score of the analyzed image.

In contrast to Lowe's proposal of using a ratio of the best and second best descriptor match, this approach takes the best match without regarding the second best. Practice has shown that similar descriptors originating from different locations in the image often block each other which results in a missed object hit. For object detection an additional assumption is made, which holds most of the time: There is a maximum of one object instance in an image. This assumption eases object detection because only one peak in the histogram has to be found. Due to the SIFT descriptors' distinctiveness, matching descriptors originating from the same object locations (both query and database image) focus their votes to the same histogram bin, if an object exists. In contrast voting entries in the histogram are widespread if the object does not exist in the image, preventing to form a peak in the histogram.

4) Histogram of oriented gradients (HOG): HOG [15] is an established feature developed initially for fast detection and tracking of pedestrians. Meanwhile, it turns out to be useful for detection of other objects as well. HOG describes intensity gradients by a gradient magnitudes histogram bins with a fixed number of bins according to their orientation. In order to utilize the spatial information, a rectangular detection window is split into sub-regions called "cells". We divided

the orientations from 0 to 180 degrees into 9 HOG-bins. The rectangular regions are divided into 2×2 cells or 1×4 in case of landscape format query sample as the zebra crossings. To handle the scale problem, the cell size of a search window starts from 8×8, 16×16 and 32×32 for small size ($\leq$ 1000 pixels), middle size ($\leq$ 10000 pixels) and large size ($>$ 10000 pixels) query objects, respectively. Then we enlarge the search window size by factor 1.5 and 2.0. E.g., for small query sample with less than 1000 pixels, 8 further runs with cell sizes 8×12, 8×11, 12×8, 12×12, 12×16, 16×8, 16×12 and 16×16 are done. Due to the large amount of keyframes, we shift the search window for 1/4 cell size, but at least 4 and at most 16 pixels. The features with 36 HOG-bins in total are normalized that the $L2$-norm equals to 1. The Euclidean distance is used to compare two histograms.

5) Region covariance descriptor: Region covariance [16] is a representation for rectangular image regions where a covariance matrix is used to describe the distribution, or more precisely the correlation of features. This descriptor is robust against noise and illumination changes. The covariance matrix provides a natural way to combine different feature types such as pixel color, pixel intensity and image gradients, etc. Especially, the spatial arrangement of the feature points can be taken into account by associating the pixel coordinates. Since the covariance matrices do not lie in a Euclidean space, an eigenvalue-based distance function is used to measure the similarity between two $d \times d$-covariance matrices C_1 and C_2

$$d(C_1, C_2) = \sqrt{\sum_{i=1}^{d} [\ln \lambda_i(C_1, C_2)]^2}$$

where λ_i is the i-th eigenvalue of the generalized eigenvalue problem $C_1 x = \lambda C_2 x$.

For the instance search, the x-, y-pixel coordinates, the RBG-color values and the first order derivatives of the pixel intensity are used. The first order derivatives are calculated using a Sobel-filter with the same window size as for HOG method. The cell structures are skipped since the spatial information is already captured in the region covariance descriptor.

B. Fusion methods

We compare four different methods for fusing the results per subsystem and query sample. Two simple methods are only based on the similarity values in the result lists, and two use feature weighting based on the similarity values of other samples from the same query.

For most of the queries all subsystems were used, with the following exceptions: For person and character queries, HoG and region covariance were not used, as preliminary experiments indicated that they were not performing well for these types of queries. The Gabor descriptor was applied only to query samples, in which a face has been detected. This was the case in all character queries, all but one object query, but also in the location query and in two of the object queries.

The first of the simple fusion method determines the score for each shot as the maximum similarity of all features of all samples for the query. The shots are then ranked according to these scores and the top 1000 shots are reported. The other method takes the top-k shots for each feature and sample, where k is 1000 over the number of features used for this query and then reranks the results based on the similarities from the single features.

In order to weight the features, we determined for each feature the similarity scores between a query sample and all the other samples of the same query. The ranks of the other samples in the list of similarities from matching against the database is determined. Then two different weights are calculated:

- Best rank: For each query sample and feature, the best ranked other sample from the same query is determined. The mean best rank $\bar{r}_i$ for feature f_i is calculated from N query samples as

$$\bar{r}_i = \frac{\sum_{k=1}^{N}(r_{i,k})}{N}, \tag{5}$$

  the weight is

$$w_{bestR}(f_i) = \frac{\max_{\forall f_j}(\bar{r}_j) - \bar{r}_i}{\sum_{\forall f_k} \max_{\forall f_j}(\bar{r}_j) - \bar{r}_k}. \tag{6}$$

- Number in top 100: For each query sample and feature, number of other samples from the same query ranked among the top 100 $n100_i$ is determined. The weight is then given as

$$w_{t100}(f_i) = \frac{\bar{n}100_i}{\sum_{k=1}^{N} \bar{n}100_k}, \tag{7}$$

  with $\bar{n}100_i$ being the mean number of samples among the top 100 for the different query samples.

For both types of weights, the fused score for each shot is calculated as a weighted sum of the scores for each feature of the maximum matching query sample. The result list is then reranked according to the fused score and the top 1000 shots are reported.

C. Results

The results for the individual queries and the mean MAPs of the query types are shown in Figure 3. Over all queries, the weighted fusion methods perform better than the simple ones with 0.0066 for weighting based on the number ranked among the top 100 and 0.0087 for weighting based on the top ranked sample compared to 0.0051 for max/max and 0.0059 for top-k. As most of our features do not specifically target person recognition, the results for object queries are better than the overall results with MAPs for all fusion methods above 0.01. The best fusion method is still one of the weighted methods, but the improvement is only minor.

We also analyzed the performance of each of the subsystems, the results are shown in Figure 4. For the mean over all queries, the Gabor face feature yields a slightly higher score than the best fused result, although it has not been applied to all the queries. SIFT scores only slightly worse than the best

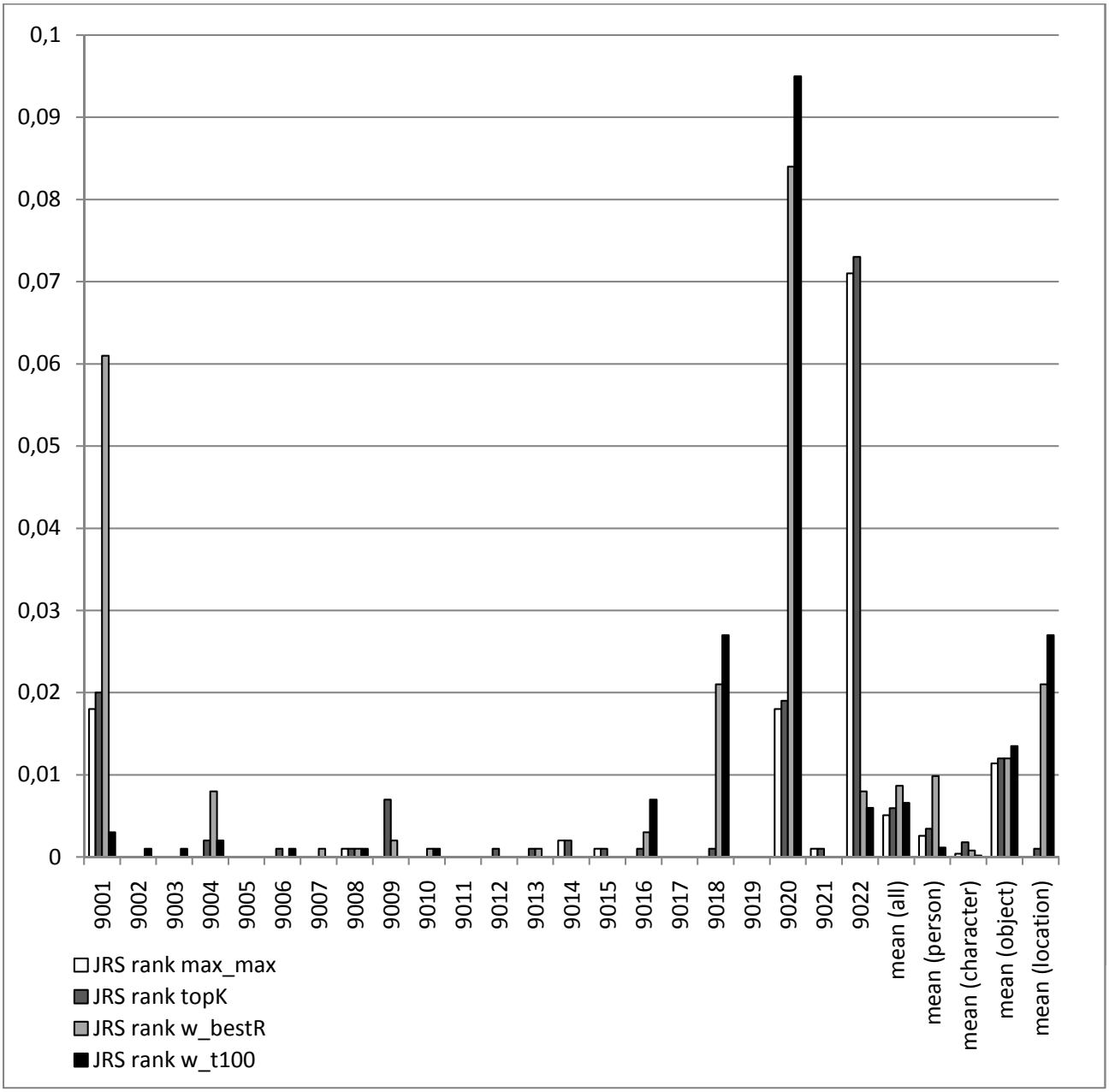

Fig. 3. MAP for the different fusion methods for each of the concepts and means (of query category and overall).

fused result. For person queries the Gabor descriptor scores clearly better than the best fused result, the other features score clearly worse. For object queries SIFT outperforms the best fused result by about 50%. For character queries and the location query the best fused result outperforms all single features.

It has to be noted that due a mistake in generating the feature databases, some shots were reported more than once in the results. This means that possible some more relevant shots could have been added at the end of the result list. However, as this applies only to the least relevant shots, the impact on the overall results is negligible.

The reported runtimes are calculated as follows. The runtimes are the sums of the response times of all subsystems, assuming that these queries are performed in a serialized way on a single core system. As some subsystems do some preprocessing or indexing while other subsystems do all required processing on the fly, we have included preprocessing times in order to be independent of these differences in implementations. The system can be very easily parallelized by running the queries for each sample and each subsystem on separate cores/machines. As fusion is only done on the result lists, the time needed for fusion is minimal compared to feature-based matching.

D. Conclusion

Overall the weighted fusion methods perform better than the simple ones. However, for homogeneous groups of queries (such as person or object queries) single features outperform even the best fused results. In this cases the influence of the fusion method is also only minimal. The results show that no fusion method is satisfactory, as the weight of the best feature is diminished, so that the best fused result is worse than the best single feature. This issue needs to be addressed in future research.

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] 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.

[5] 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.

[6] 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.

[7] 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.

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

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

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

[14] 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".

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, pp. 886–893.

[16] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *Proceedings of European Conference on Computer Vision (ECCV)*, Graz, Austria, May 2006, pp. 589–600.

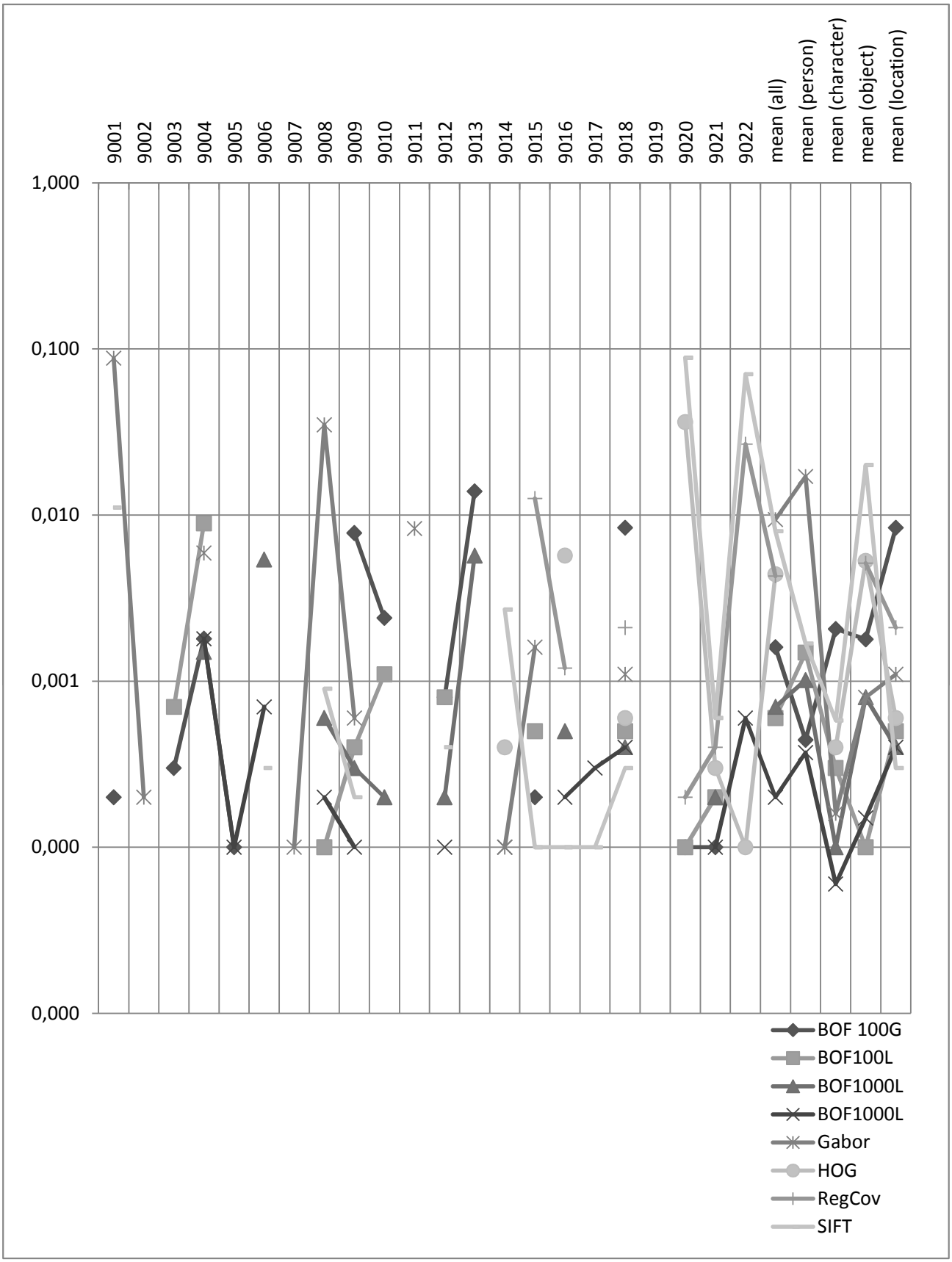

Fig. 4. MAP (log scale) for the different features for each of the concepts and means (of query category and overall).