Oxford TRECVID 2006 – Notebook paper

James Philbin, Anna Bosch, Ondřej Chum, Jan-Mark Geusebroek,
Josef Sivic and Andrew Zisserman
Department of Engineering Science
University of Oxford
United Kingdom

Abstract

The Oxford team participated in the high-level feature extraction and interactive search tasks. A vision only approach was used for both tasks, with no use of the text or audio information.

For the high-level feature extraction task, we used two different approaches, one using sparse and one using dense visual features to learn classifiers for all 39 required concepts, using the training data supplied by MediaMill [29] for the 2005 data. In addition, we also used a face specific classifier, with features computed for specific facial parts, to facilitate answering people-dependent queries such as "government leader". We submitted 3 different runs for this task. `OXVGG_A` was the result of using the dense visual features only. `OXVGG_OJ` was the result of using the sparse visual features for all the concepts, except for "government leader", "face" and "person", where we prepended the results from the face classifier. `OXVGG_AOJ` was a run where we applied rank fusion to merge the outputs from the sparse and dense methods with weightings tuned to the training data, and also prepended the face results for "face", "person" and "government leader". In general, the sparse features tended to perform best on the more object based concepts, such as "US flag", while the dense features performed slightly better on more scene based concepts, such as "military". Overall, the fused run did the best with a Mean Average (inferred) Precision (MAP) of 0.093, the sparse run came second with a MAP of 0.080, followed by the dense run with a MAP of 0.053.

For the interactive search task, we coupled the results generated during the high-level task with methods to facilitate efficient and productive interactive search. Our system allowed for several "expansion" methods based on the sparse and dense features, as well as a novel on the fly face classification system, which coupled a Google Images search with rapid Support Vector Machine (SVM) training and testing to return results containing a particular person within a few minutes. We submitted just one run, `OXVGG_TVI`, which performed well, winning two categories and coming above the median in 18 out of 24 queries.

1 High-level Feature Extraction

Our approach here is to train an SVM for the concept in question, then score all key frames in the test set by the magnitude of their discriminant (the distance from the discriminating hyper-plane), and subsequently rank the test shots by the score of their keyframes. We have developed three methods for this task, each differing in their features and/or kernel. Two of the methods are applicable to general visual categories (such as airplane, mountain and road) and the third is specific to faces. The first two methods differ in that one uses sparse (based on region detectors) monochrome features, and the other uses dense (on a regular pixel grid) colour features. We now describe the three methods in some detail.

1.1 Bag of visual word representation

The first approach uses a *bag of (visual) words* [27] representation for the frames, where positional relationships between features are ignored. This representation has proved successful for classifying images according to whether they contain visual categories (such as cars, horses, etc) by training an SVM [7]. Here we use the kernel formulation proposed by [31].

Features and bag of words representation: We used two types of affine region detectors, Hessian Laplace (HL) [21], and Maximally Stable Extremal Regions (MSER) [20]. Each region is then represented by a SIFT [19] descriptor using intensity only. This combination of detection and description generates features which are approximately invariant to an affine transformation of the image, see figure 1. These features are computed in all (representative and non-representative) keyframes. The 'visual vocabulary' is then constructed by vector quantizing the SIFT descriptors of a random subset of features from the training data using K-means. The K-means cluster centres define the visual words. Each feature type (HL and MSER) has its own vocabulary. We used two vocabulary sizes of 3,000 and 10,000 words for HL, and one vocabulary size of 3,000 words for MSER, ref-

Figure 1: An example of Hessian-Affine regions used in the bag of words method (section 1.1). Left: original image; right: sparse detected regions overlaid as ellipses.

ered to as HL 3K, HL 10K and MSER 3K, respectively. The SIFT features in each keyframe are then assigned to the nearest cluster centre, to give the visual word representation, and the number of occurrences of each visual word is recorded in a histogram. This histogram of visual words is the bag of visual words model for that frame.

SVM classification: To predict whether a keyframe from the test set belongs to a concept, an SVM classifier is trained for each concept. Specifically, a kernel SVM with χ^2 kernel

$$K(p,q) = e^{-\alpha\chi^2(p,q)}$$

where

$$\chi^2(p,q) = \sum_{i=1}^{N} \frac{(p_i - q_i)^2}{p_i + q_i}$$

is used. The parameter α in the kernel function is set to be an estimate of the average χ^2 distance between training images. We used the SVM-light [15] package.

The positive and negative training examples are obtained from the MediaMill 2005 data. All shots provided as a ground truth were used as positive training examples. A set of $5,000$ negative examples were randomly sampled from the whole collection of keyframes. Identical frame detection based on the χ^2 distance was used to exclude frames identical to any of the positive examples from the set of negative examples. In the "Chart" concept, positive examples from "Maps" concept were included as negative examples.

The choice of which vocabulary to use for a particular concept, and the values of the SVM parameters (slack variables and error weight for misclassified positive/negative examples) was determined using a validation data set from the training data. For most concepts, using HL 10K alone proved sufficient. The exceptions were: "bus", "court", "maps", "office", "snow" and "sports" where HL 3K was superior; and "building", "car" and "crowd" where HL 3K together with MSER 3K was used.

Every keyframe from the test data is scored using the SVM and the shots are ranked by their highest scoring keyframes. Some results of the method are shown in Figure 2.

1.2 Spatial visual word representation

This classification method differs in three ways from the pure bag of visual word method of the previous section. First, the spatial position of the feature in the image is used in the matching – pairs of frames are scored more highly if the "visual words" occur at similar positions. The aim is to match the scene layout in the frame, i.e. not isolated objects which can change position. Second, dense (on a regular pixel grid) descriptors are used, rather than only sparse features which are only computed where affine region detections occur. Third, the SIFT descriptor is computed on colour channels so that colour edges contribute, rather than only using intensity gradients. We now give more details.

Dense features: SIFT descriptors [19] are computed at points on a regular grid with spacing M pixels, here $M = 10$. At each grid point SIFT descriptors are computed over circular support patches with radii r = 4, 8, 12 and 16 pixels. Consequently each point is represented by four SIFT descriptors. Multiple descriptors are computed to allow for scale variation between images. The patches with radii 4 do not overlap and the patches with radii 8, 12 and 16 overlap. The SIFT descriptors are computed for each HSV component. This gives a 128×3 dimension SIFT descriptor for each point. This captures the colour gradients (or edges) of the image. Note that the descriptors are rotation invariant. More details of the features and their performance in scene classification is given in [5]. The dense features are vector quantized into "visual words" using K-means clustering. Each keyframe contains 1,380 features. The K-means clustering was performed over 4,000,000 feature vectors selected at random. A vocabulary of 1,500 words is used here.

Spatial pyramid kernels: We use the method proposed by Lazebnik *et al.* [16] which is based on a spatial pyramid matching kernel [12]. Pyramid matching works by placing a sequence of increasingly coarser grids over the feature space and taking a weighted sum of the number of matches that occur at each level of resolution (L). At any fixed resolution, two points are said to match if they fall into the same bin of the grid; matches found at each resolution are weighted by α_l. In this case the pyramid matching is applied to the two-dimensional image space, and matches occur if descriptors assigned to the same visual word occur in the same histogram bin. The spatial layout is illustrated in figure 3.

The resulting spatial pyramid is an extension of the bag-of-words image representation, but it reduces to a standard bag-of-words when $L = 0$. We used this method with $L = 1$ which means that the inputs for the discriminative classifiers have a dimension of 1500×5 for a vocabulary of 1500 words. The kernel is defined as follow:

Figure 2: Example of bag of visual words classification (section 1.1): top ten results for basketball, maps, crowd, airplane, and military concepts.

$$K^L(X,Y) = \sum_{v=1}^{V} k^L(X_v, Y_v)$$

where V is the vocabulary size and $k^L(X,Y) = \sum_{l=0}^{L} \alpha_l I^l$, and I is the histogram intersection between the two feature vectors X and Y at each pyramid level. Normally, matches found at finer resolutions are weighted more highly than matches found at coarser resolutions. However due to the images used in TRECVID (where the spatial distribution is very variable between images from the same concept) we used here $\alpha_0 = 0.75$ and $\alpha_1 = 0.25$.

For each class, we used all the positive training shots and 10,000 randomly selected shots as negative examples from the 2005 training data supplied by MediaMill. Classification is carried out using a one-versus-all SVM (LIBSVM package [6]). Every representative keyframe from the test data is scored by the probability estimates provided by LIBSVM, and this in turn determines the shot ranking.

Figure 4 shows 9 matched shots for the government leader category, retrieved from the entire TRECVID 2006 test collection. These scenes have a similar layout. For example: a person centred in the middle of the image, or a two person meeting with a background wall containing some pictures. In all these cases if the image is divided into four bins (as shown in Figure 3), the similarity of the information in each of them is evident. This shows, somewhat surprisingly, that spatial information is an important feature for such concepts.

1.3 Face representation

For the concepts "Face", "Person" and "Government leader" we have developed a representation based on detected frontal faces. For the concepts "Face" and "Person" we use the output (detection strength) of a face detector [22] applied

Figure 3: Spatial information used for the scene representation (section 1.2). Each dimension of the image is divided by two obtaining 4 bins.

to all representative and non-representative test collection keyframes to rank the shots. Each shot is scored by its best scoring keyframe, and each keyframe is scored by the face detection with the strongest response.

For the "Government leader" concept we trained a (face based) person X detector for four prominent government leaders: "George Bush", "Condoleezza Rice", "Ariel Sharon" and "Hu Jintao". Each test shot is scored by the best scoring person X detector. The person X detectors are SVM classifiers on facial feature descriptors learnt from images downloaded from Google Image search. Details are given in section 2.1. Using this approach, for the "government leader" category over the TRECVID 2006 test data, 82 out of the top 100 ranked shots are correctly labelled as depicting one of the specified government leaders.

1.4 Merging lists

To fuse the ranked lists, generated using the different methods, we used a weighted Borda count method [2], which assigns votes to each candidate depending on its rank in the list. These votes are then accumulated over each list to fuse and the final rank is generated by sorting the candidates in

Figure 4: Examples of scene matching (section 1.2) for the concept "government leaders". Spatial layout is important for such cases as people are often framed in a common manner/pose for single individuals, pairs, small groups etc.

non-increasing order over a weighted sum of the votes, where the weights quantify some measure of relative confidence between the multiple sources. The weightings are determined from a validation set, taken from the training data, for each concept.

2 Interactive Search

For interactive search we can access the ranked concept lists (as described in the previous section), together with any images provided with the query, as a starting point for a search. We have developed a set of expansion-search algorithms for: particular objects, texture layout, and colour layout; that can be applied to any keyframe or query image. These can be used to efficiently harvest new shots, and in turn these new shots can be used to harvest others. In the following sections we describe the expansion methods and the user interface that enables these to be accessed efficiently. We start by describing our run-time person X search, which provides a further method for generating an initial ranked list.

2.1 Search for faces of specific people

This section describes our approach to search for specific people in the video (person X detection) using their imaged face appearance [14, 26, 30]. This is implemented by training an SVM classifier on descriptors extracted from (close to) frontal face detections. The positive training data is obtained using Internet image search. For example, we search for "Condoleezza Rice" using Google Image search and obtain hundreds of images containing her face. Examples of returned images are shown in figure 5. The resulting SVM classifier is applied to all frontal faces detected in all keyframes (representative and non-representative) of the TRECVID 2006 test data. Utilizing parallel processing power of several machines, the entire search process from placing the query to the returned ranked lists of TRECVID 2006 test shots takes only few minutes. This allows us to search for any person (searchable on Google Images) in the TRECVID 2006 test video footage on the fly, thereby greatly expanding the limited set of pre-defined concepts.

Figure 5: Example images with frontal face detections overlaid returned by Google image search queried for "Condoleezza Rice". Note that downloaded images may contain faces of other people. In our interactive search application the user is presented with all detected faces and manually labels 20-200 positive face exemplars containing the person of interest. For this example query, we downloaded 783 images containing 299 frontal face detections of which 190 were images of Condoleezza Rice.

In the following, we describe the face processing pipeline, give details of the classifier training procedure, and show some example retrieval results.

2.1.1 Face processing pipeline

Below we overview our face processing pipeline (for details see [8, 26]). The aim here is to find people in the images and extract descriptors of their facial appearance. The face processing pipeline is applied to both the training data (im-

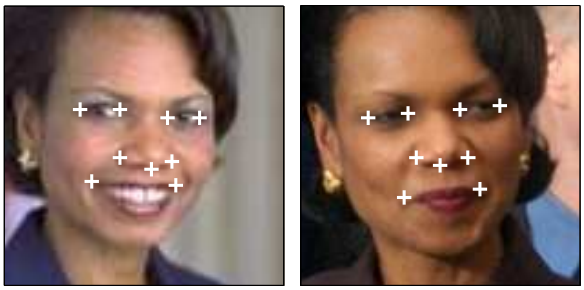

Figure 6: Localized facial features in the top-left and bottom-right image of figure 5.

ages returned from Google Image search) and the test data (keyframes from the TRECVID 2006 collection).

At the first stage of processing we run a frontal face detector [22] on each image. To achieve a low false positive rate, a conservative threshold on detection confidence is used. The use of a frontal face detector restricts the video content we can search to near-frontal faces, but typically gives a much greater reliability of detection than is currently obtainable using multi-view face detection [17].

Facial feature localization: The output of the face detector gives an approximate location and scale of the face. In the next stage, the facial features are located in the detected face region. Nine facial features are located: the left and right corners of each eye, the two nostrils and the tip of the nose, and the left and right corners of the mouth. Additional features corresponding to the centres of the eyes, a point between the eyes, and the centre of the mouth, are defined relative to the located features.

To locate the facial features, a model combining a generative model of the feature positions with a discriminative model of the feature appearance is applied. The probability distribution over the joint position of the features is modelled using a mixture of Gaussian trees, a Gaussian mixture model in which the covariance of each component is restricted to form a tree structure with each variable dependent on a single "parent" variable. This model is an extension of the single tree proposed in [10]. Fuller details of the facial feature detection are given in [8, 9].

Figure 6 shows examples of the face detection and feature localization. The facial features can be located with high reliability in the faces detected by the face detector despite variation in pose, lighting, and facial expression.

Representing face appearance: A representation of the face appearance is extracted by computing descriptors of the local appearance of the face around each of the located facial features. Extracting descriptors based on the feature locations [26] gives robustness to pose variation, lighting, and partial occlusion compared to a global face descriptor [13, 25]. Errors may be introduced by incorrect localization of the features, which become more difficult to localize in extremely non-frontal poses, but using a frontal face detector restricts this possibility.

Before extracting descriptors, the face region proposed by the face detector is further geometrically normalized to reduce the scale uncertainty in the detector output and the effect of pose variation, e.g. in-plane rotation. An affine transformation is estimated which transforms the located facial feature points to a canonical set of feature positions, and the face is mapped by this transformation into a canonical frame [1, 4]. A facial feature descriptor is then formed by taking the vector of pixels in a circular region about each facial feature point and normalizing to obtain local photometric invariance. A circle around a facial feature in the canonical frame corresponds to an ellipse in the original image. The descriptor for the face is formed by concatenating the descriptors for each facial feature.

2.1.2 The Support Vector Machine classifier

The retrieval of a particular person is facilitated by first training a two-class SVM classifier on the person's facial appearance. The built classifier is then applied to facial descriptors extracted from the TRECVID test data. Details of both the training and testing stage are given now.

Training data: The positive training data is obtained by querying an Internet image search engine by the name of the target person, e.g. "Condoleezza Rice" or "Dick Cheney". Currently we use Google Image search but multiple image search engines such as MSN or Yahoo could be used. Frontal faces and their descriptors are extracted from all downloaded images as described above. Typically we download 600-800 images, which results in 150-300 extracted frontal faces. Examples are shown in figure 5. Downloaded images may contain faces of other people, which appear together with the target person and here we rely on the user to label the relevant face detections using a simple labelling interface. Labelling 50-100 faces takes about 1-2 minutes. In the future, the labelling process can be further simplified by a pre-clustering procedure [4].

The negative training data consists of (i) 2,463 faces of about 20 people (mainly politicians) downloaded from Google Images and (ii) 2,008 faces of anchors extracted from keyframes of the "Anchor person" concept from the TRECVID 2005 training collection. In total, the negative training data contains 4,471 faces and remains fixed throughout the entire run of the interactive search.

Training the classifier: Given the positive and negative training data, an SVM classifier with a Radial Basis Func-

tion kernel is trained using a publicly available implementation of the SVM learning algorithm [15]. Parameters of the classifier such as the width of the kernel and the trade-off between the training error and margin were optimized using cross-validation off-line and kept fixed throughout the whole interactive search run.

Classification: The test data consists of 23,508 frontal face detections extracted (off-line) from all the keyframes (both representative and non-representative) in the TRECVID 2006 test collection. The face detections are ranked by the magnitude of the discriminant and each shot is ranked by its top ranked face detection. Figure 7 shows examples of retrieval results.

On the fly search: To achieve on the fly search, several stages of the search pipeline have been parallelized to run on multiple (up to 20) machines. The multi-threaded crawler can download about 400 images from Google Image search in about 30 seconds on a single machine. The face processing takes 0.1-5 seconds on a single image depending on the image resolution and the number of extracted faces. A set of 600 downloaded images can be processed in about a minute on up to 20 machines. The negative training data and the test data are processed off-line. The SVM training is performed on a single machine and takes between 20-60 seconds. Finally, the classification of the test data (23,508 face detections) is again performed in parallel and takes less than a minute. The entire end-to-end process typically takes less than 5 minutes, including the user labelling of positive face detections.

Note that the on the fly face search provides a good way to start the interactive visual search for particular people and is complementary to the text-based search of automatic speech recognition transcripts. For example, in the case of the interactive query "Find shots of Condolleezza Rice", we first obtained about 13 correct shots using the face search (see figure 7a), which we then expanded using neighbouring shots and various expansions techniques described in section 2.2 and section 2.3 to a total of 58 shots containing "Condoleezza Rice".

(a) (b) (c)

Figure 7: The top 9 ranked shots retrieved from the entire TRECVID 2006 test collection for query on (a) "Condoleezza Rice", (b) "Hu Jintao", (c) "Saddam Hussein". Each shot is shown by the best ranked face detection within the shot. The TRECVID 2006 test collection is represented by about 23,000 close to frontal faces, detected in both the representative and non-representative keyframes. The top 100 ranked shots contain 13, 25 and 90 correctly retrieved shots for (a), (b) and (c) respectively. Building a face based person X detector on the fly provides a starting point for the interactive search. See text (section 2.1).

2.2 Search for specific objects – Video Google

Some of the queries in a TRECVID interactive session can be partially answered by using real-time search for specific objects over the corpus. The object used for search can be directly or indirectly related to the actual query. For example, to find George Bush, one could try and search for the presidential seal, often seen on the lectern at presidential press conferences (see figure 8). Another example might be to search for the NBA logo to find basketball matches. Here

we describe an implementation of the "Video Google" approach [27] for the TRECVID corpus. The aim is to retrieve shots containing a specific object despite changes in scale, viewpoint and illumination. The visual query is specified at runtime by outlining the object in an example image.

The approach works by representing a query object by appearance regions, partially invariant to viewpoint and illumination changes, and using these to match similar objects from the rest of the corpus [18]. The SIFT descriptors for these appearance regions are vector quantized into visual words, and a bag of visual words representation used for each key frame. With this representation, standard efficient text retrieval methods [3] can be employed to enable object retrieval in a Google-like manner.

Features: Here the Hessian-Affine feature detector [23] is used since it gave the best performance on validation data.

The bag of words representation for each key frame is then obtained in a similar manner to that of section 1.1. For the TRECVID test data, $52,343,280$ descriptors were generated, with 362 per keyframe on average. To compute the clusters, we use a distributed K-means algorithm which runs over every descriptor in the dataset. Empirically, setting $K = 150,000$ gives good results for the TRECVID data.

We then apply standard text retrieval methods to search over the dataset. A sparse term-document matrix, T, is generated whose element $T_{i,j}$ is equal to the frequency of visual word i in keyframe j. We apply a simple TF-IDF weighting scheme [24] to this matrix to boost the effect of rarely occurring (and thus discriminative) words from commonly occurring ones. Each column of this matrix is also normalized to speed-up the calculation of the L_2 distance during retrieval.

Object search: At retrieval time, the user specifies a rectangular region of an image to search on. All the visual words falling inside the rectangle are accumulated and a sparse frequency vector is generated. This is reweighted by the IDF weights calculated from the entire dataset and normalized to have unit L_2 norm. All the documents from the corpus are then ranked in non-increasing order by the normalized scalar product to the query vector, which is equivalent to ranking in non-decreasing order using the L_2 distance.

The results of some example queries are shown in figure 8.

2.3 Expansion-search using the spatial layout of texture or colour

In order to enable a quick search for similar images, for each keyframe, we precomputed the 20 most similar keyframes. The similarity was measured using the χ^2 distance and was computed over two different spatial layout descriptors: texture and colour.

For the textural spatial layout, we used the same descriptors as in section 1.2 with a spatial pyramid with $L = 1$. The colour spatial layout is based on the opponent colour model [11]

$$\begin{aligned} I &= (R+G+B)/3 \\ O_1 &= (R+G-2B)/4+128 \\ O_2 &= (R-2G+B)/4+128 \end{aligned}$$

On level L_0, the I channel is quantized into 64 histogram bins and each of the O_1 and O_2 channels are quantized into 32 histogram bins. On each subsequent (finer) level, the number of histogram bins is divided by 4. On level L_1, there are 16, 8, and 8 histogram bins used for the channels respectively, and on level L_2 only 4, 2, and 2 bins. Since the number of spatial bins is multiplied by 4 on each finer level, this approach keeps the amount of data constant for each level. Level L_2 was weighted 2 times more than levels L_0 and L_1.

Example query expansion results are shown in figures 9 and 10 for the texture and colour expansion respectively.

2.4 The interface

In designing a successful user-interface for TRECVID it is important to specify which goals such an interface should meet. The system must make it easy for a user to combine the many different streams of data in an efficient and intuitive manner. In our case, the main data sources were:

1. Pre-generated results from the 39 pre-defined concepts using the sparse visual features (section 1.1).
2. Pre-generated results from the 39 pre-defined concepts using the dense visual features (section 1.2).
3. On the fly results from the facial classifier (section 2.1).
4. Interactive Video Google object recognition (section 2.2).
5. Interactive "colour expansion" (section 2.3).
6. Interactive "texture expansion" (section 2.3).

Inspired by the *CrossBrowser* approach [28], the main interface view contains two axes (see figure 11). The x-axis represents the temporal ordering of the shots in the corpus and enables the user to move backwards and forwards in time. The y-axis displays the rank ordering for the currently loaded list. There is often a high level of temporal coherency between subsequent shots and exploiting this is crucial to good interactive performance. Frequently, in searching forwards and backwards from one relevant shot, the user would find more shots relevant to the query.

The interface allows for rapid access to the data sources mentioned in the following ways. Any of the pre-generated

Figure 8: Examples of querying using "Video Google". The query image is shown on the left, with the query object outlined in yellow, and the returned ranked keyframes follow. Top row: search on the presidential seal. The 1st, 3rd, 4th, 7th, 8th and 9th results are shown; second row: search on the words "RICE IN IRAQ". The top 6 results are shown; Bottom row: search on part of the us flag. The 1st, 2nd, 3rd, 4th, 6th and 7th results are shown.

Figure 9: Example of queries using the "texture expansion". The query image is shown on the left-hand side, with the results following. From the top row, the queries are *military*, *mountain* and *water*.

Figure 10: Examples of queries using the "colour expansion". The query image is shown on the left-hand side, with the results following. From the top row, the queries are *chimney*, *demonstration* and *football*.

results can be loaded into one of ten "live" lists in the system. Lists can then be appended, trimmed or fused at will to give the user a list which can be labelled (as correct or not for the topic). In addition, a facial classifier can be trained at runtime to give the user an extra source from which he or she can start to label. Once a "good" list has been generated, the user can quickly find positive and negative examples by navigating. Once some good examples have been found, the user can then use one of the expansion methods to "grow" the positive examples. Using either the colour or the texture expansions worked well for scene-like queries.

A run using the interface for an example query "Find Condoleezza Rice" will now be described. As this is a "Person X" style query, we start by using the on the fly face classifier. The user enters "Condoleezza Rice" in the real-time labeller, which automatically downloads the top 600–800 images from Google Images – the user can label them as positive or negative as they are downloaded and processed, and when enough training images have been collected (up to 200), the user sends the examples to the SVM trainer for classification. Once the results have been returned to the user (see figure 7), he or she can start labelling them and navigating using the arrow keys to find other examples. Relevant keyframes, containing overlayed text of her name (see figure 8), can be searched on using Video Google, the top 20 (currently unlabelled) results being added below the user's current position. The expansion methods (colour, texture, and near-duplicate using χ^2 on the bag of words representation) can be used on any interviews or press conferences to find the same event shown on different channels (again, the top 20 unlabelled results are added below the users current position for all expansion methods). Once the user is finished, the list is sorted and the results saved.

References

[1] O. Arandjelovic and A. Zisserman. Automatic face recognition for film character retrieval in feature-length films. In *Proc. CVPR*, pages 860–867, 2005.

[2] J. Aslam and M. Montague. Models for metasearch. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 276–284. ACM Press, 2001.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, ISBN: 020139829, 1999.

[4] T. Berg, A. Berg, J. Edwards, M. Maire, R. White, Y. W. Teh, E. Learned-Miller, and D. Forsyth. Names and faces in the news. In *Proc. CVPR*, pages 848–854, 2004.

[5] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via plsa. In *Proc. ECCV*, pages IV:517–530, 2006.

[6] C. Chang and C. Lin. *LIBSVM: a library for support vector machines*, 2001.

[7] G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[8] M. Everingham, J. Sivic, and A. Zisserman. Hello! my name is... buffy – automatic naming of characters in tv video. In *Proc. BMVC.*, 2006.

[9] M. Everingham and A. Zisserman. Regression and classification approaches to eye localization in face images. In *Proc. Int. Conf. Autom. Face and Gesture Recog.*, 2006.

[10] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1), 2005.

[11] J. Geusebroek, R. van den Boomgaard, A. Smeulders, and H. Geerts. Color invariance. *IEEE Trans. Pattern Anal. Machine Intell.*, 23(12):1338–1350, 2001.

[12] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proc. ICCV*, pages I:357–364, Oct 2005.

[13] B. Heisele, P. Ho, J. Wu, and T. Poggio. Face recognition: component–based versus global approaches. *CVIU*, 91:6–21, 2003.

[14] L. Jin, S. Satoh, F. Yamagishi, D. Le, and M. Sakauchii. Person X detector. In *TrecVid Workshop*, 2004.

[15] T. Joachims. Making large-scale svm learning practical. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.

[16] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. CVPR*, pages II:2169–2178, 2006.

[17] S. Z. Li and Z. Q. Zhang. Floatboost learning and statistical face detection. *IEEE PAMI*, 26(9), 2004.

[18] D. Lowe. Object recognition from local scale-invariant features. In *Proc. ICCV*, pages 1150–1157, Sep 1999.

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

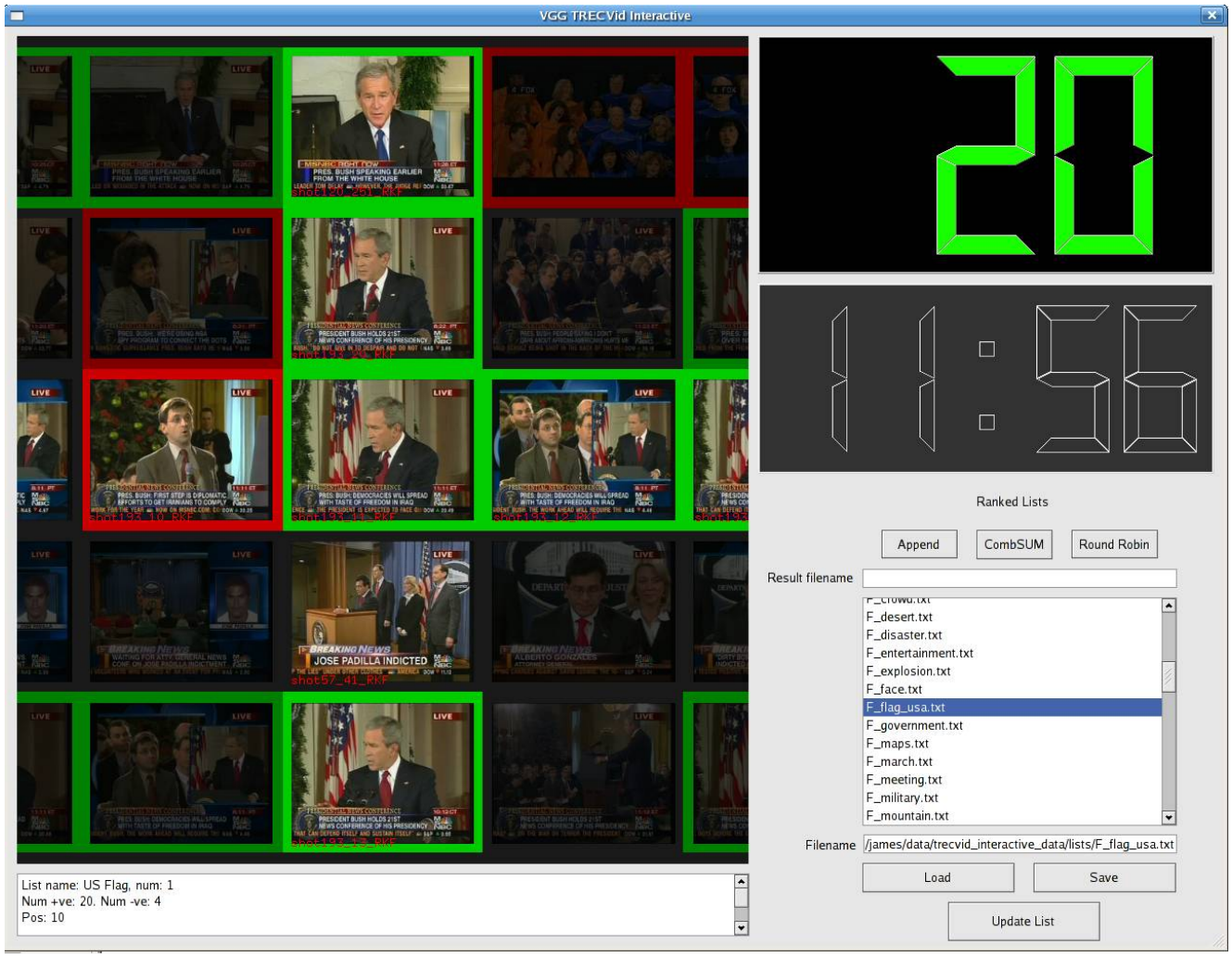

Figure 11: Finding George Bush from the US flag category. The currently found positive examples are labelled green, the negative examples are labelled red. The horizontal axis shows shots in temporal order, the vertical axis shows shots in current rank order. The user can navigate using the arrow keys. Ranked lists from the various different sources can be loaded or fused from the pane to the right. All of the expansion methods are invoked using a single key press, except for the Video Google method, for which the user specifies a search rectangle using the mouse. The green number shows the number of currently labelled positive examples. The clock shows the amount of time left for answering the current query.

[20] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proc. BMVC.*, pages 384–393, 2002.

[21] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proc. ECCV*. Springer-Verlag, 2002.

[22] K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *Proc. ECCV*. Springer-Verlag, May 2004.

[23] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *IJCV*, 65(1/2):43–72, 2005.

[24] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *IPM*, 24(5):513–523, 1988.

[25] G. Shakhnarovich and B. Moghaddam. *Handbook of face recognition*, chapter Face recognition in subspaces. Springer, 2004.

[26] J. Sivic, M. Everingham, and A. Zisserman. Person spotting: video shot retrieval for face sets. In *International Conference on Image and Video Retrieval (CIVR 2005), Singapore*, 2005.

[27] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, volume 2, pages 1470–1477, Oct 2003.

[28] C. Snoek, J. van Gemert, J. Geusebroek, B. Huurnink, D. Koelma, G. Nguyen, O. de Rooij, F. Seinstra, A. Smeulders, C. Veenman, and M. Worring. The mediamill TRECVID 2005 semantic video search engine. In *Proceedings of the 3rd TRECVID Workshop*, November 2005.

[29] C. Snoek, M. Worring, J. van Gemert, J. Geusebroek, and A. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the ACM International Conference on Multimedia*, October 2006.

[30] J. Yang, M. Chen, and A. Hauptmann. Finding person X: Correlating names with visual appearances. In *Proc. CIVR*, 2004.

[31] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classifcation of texture and object categories: An in-depth study. Technical Report RR-5737, INRIA Rhône-Alpes, Nov 2005.