IBM Research TRECVID-2009 Video Retrieval System

Apostol Natsev[*], Shenghua Bao[†], Jane Chang[‡], Matthew Hill[*], Michele Merler[§], John R. Smith[*], Dong Wang[†], Lexing Xie[*], Rong Yan[*], Yi Zhang[¶]

Abstract

In this paper, we describe the IBM Research system for indexing, analysis, and copy detection of video as applied to the TRECVID-2009 video retrieval benchmark.

A. High-Level Concept Detection:

This year, our focus was on global and local feature combination, automatic training data construction from web domain, and large-scale detection using Hadoop.

1. *A_ibm.Global_6: Baseline runs using 98 types of global features and 3 SVM learning methods;*
2. *A_ibm.Combine2_5: Fusion of the 2 best models from 5 candidate models on global / local features;*
3. *A_ibm.CombineMore_4: Fusion of all 5 candidate models on global / local features;*
4. *A_ibm.Single+08_3: Single best model from the 5 candidate models, plus the old models from 2008;*
5. *C_ibm.Combine2+FlkBox_2: Combine A_ibm.Combine2_5 with automatic extracted training data from Flickr;*
6. *A_ibm.BOR_1: Best overall run, assembled from best models for each concept using heldout performance.*

Overall, almost all the individual components can improve the mean average precision after fused with the baseline results. To summarize, we have the following observations from our evaluation results: 1) The global and local features are complementary to each other, and their fusion results outperform either individual types of features; 2) The more features are combined, the better the performance, even with simple combination rules; 3) The development data collected automatically from the web domain are shown to be useful on a number of the concepts, although its average performance is not comparable with manually selected training data, partially because of the large domain gap between web images and documentary video;

B. Content-Based Copy Detection:

The focus of our copy detection system this year was in fusing 4 types of complementary fingerprints: a temporal activity-based fingerprint, keyframe-based color correlogram and SIFTogram fingerprints, and an audio-based fingerprint. We also considered two approaches (mean- and median-equalization) for score normalization and fusion across systems that produce vastly different score distributions and ranges. A summary of our runs is listed below:

1. *ibm.v.balanced.meanBAL: Video-only submission produced by fusing the temporal activity-based and keyframe color correlogram-based fingerprints after mean equalization and score normalization.*
2. *ibm.v.balanced.medianBAL: As above, but using the median scores as weighting factors.*
3. *ibm.v.nofa.meanNOFA: Similar to the first run, but with internal weights for our temporal method tuned more conservatively and a higher score threshold applied to our color feature based method.*
4. *ibm.v.nofa.medianNOFA: Similar to the meanNOFA run, but using the median scores for weighting.*
5. *ibm.m.balanced.meanFuse: For A+V runs, we used the same 2 video only methods, plus another video*

[*]IBM T. J. Watson Research Center, Hawthorne, NY, USA
[†]IBM China Research Lab, Beijing, China
[‡]IBM Software Group, Cambridge, MA
[§]Dept. of Computer Science, Columbia University
[¶]Machine Learning Dept., Carnegie Mellon Univ.

method (SIFTogram) and a temporal audio-based method. In this run, we used the mean scores of each constituent for weighting.

6. *ibm.m.balanced.medianFuse: As in the above run, but using median score for weighting.*

7. *ibm.m.nofa.meanFuse: As with the video-only runs, we adjusted internal parameters of the temporal methods and the thresholds for the other methods.*

8. *ibm.m.nofa.medianFuse: As in the m.nofa.meanFuse run, but using the median scores for weighting.*

Overall, the SIFTogram approach performed best, followed by the correlogram approach and the temporal activity-based fingerprint approach, while audio did not help. With respect to score normalization and fusion, we found median equalization to be more effective than mean equalization.

1 Introduction

This year the IBM team has participated in the TREC Video Retrieval Track, and submitted results for the High-Level Feature Detection and Content-Based Copy Detection tasks. This paper describes the IBM Research system and examines the approaches and results for both tasks.

The IBM team continues its investigation on high-level feature detection along three main directions: global / local feature combination, large-scale learning with Hadoop and automatic training data construction from web domain. First, we introduce multiple types of local SIFT-based features in addition to the original 98 types of global features, in view of the success of local features in previous evaluations. To efficiently learn from such a large pool of features, we generated the baseline results using robust subspace bagging using Hadoop. Multiple learning strategies have been tested. In addition, we provided a Type-C run to verify if training data automatically downloaded and filtered from Flickr can contribute to detecting concepts in the news domain. Finally, multiple combination strategies were utilized to augment the detection performance for individual concepts. The official evaluation results show that our best run achieved 56% improvement over the baseline run in terms of mean average precision.

For the task of copy detection, our focus was on the design and fusion of multiple complementary types of fingerprinting approaches. Specifically, we fuse results from the following 4 types of fingerprinting approaches:

1. A *color correlogram*-based method designed for matching very short copied segments under mild to moderate transformations that are largely color-preserving (e.g., compression, transcoding, noise, quality reduction, etc.).

2. A *SIFTogram*-based method, which is a global frame-level histogram of quantized visual codewords based on SIFT local features [8], designed for matching frames under wider variety of transformations, and especially ones that substantially perturb colors, such as gamma correction. Our main focus for this approach was on scalability and we show that we can still obtain significant performance boost without the need to do matching and spatial registration at the local interest point level, which would make this approach orders of magnitude more expensive.

3. A *temporal visual activity*-based method, designed for matching longer sequences of copied material under extreme compression, transcoding, and noise transformations, where weak evidence of frame-level matches can be accumulated over time to produce a strong segment-level match without false alarms.

4. A *temporal audio activity*-based method, which is similar to the above but is based on audio energy fingerprints.

We used the color correlogram-based and the visual activity-based methods to produce our submitted video only runs. In the audio + video runs, we also added the SIFTogram method (which wasn't ready for our video-only submission) and the audio-based method. Surprisingly, our preliminary audio-based approach ended up hurting our performance, suggesting that our audio-based fingerprinting was too simplistic for the task. In contrast, audio turned out to be an important factor in the A-V task for other participants, as the best results among all participants were obtained from purely audio-based systems.

For fusion across runs, we noted that the two temporal approaches had very different score distributions than the two frame-based matching methods (correlogram and

SIFTogram). This was due to boosting factors in our temporal sequence matching approach, which tend to generate extreme values for confident matches, similar to a power-law distribution. Simple range normalization was therefore not sufficient to equalize the score ranges across all runs before fusion, and we instead use mean- or median-equalization, followed by a modified linear fusion scheme, which takes into account not only the overall confidence scores but also the agreement of matched segments asserted by each method. Between the score normalization methods, we found median-equalization to be more effective than mean-equalization, likely due to the fact that the median is less sensitive to outliers and the extreme score values produced by the temporal approaches. There were only isolated instances in which the mean equalization method outperformed the median.

Overall, in looking at the results, we were also surprised at the difficulty in choosing the threshold for the actual NDCR metric. We also note that the "balanced" profile admits very few false alarms, and suggest that a more truly balanced profile be included in the future.

2 High-level Feature Detection

Our concept detection system includes multiple base and meta-level learning algorithms such as robust subspace bagging with SVMs, cross-domain learning with web data, and so on. It also consists of different fusion strategies for leveraging multi-modal relationships. We continue improving the general SVM learning algorithms to accommodate a larger set of global and local visual features, and re-implement the learning algorithms on a MapReduce-based distributed learning system called Hadoop. The details of these components are explained in the rest of this section.

2.1 Video Descriptors

All of our features are extracted from the representative keyframes of each video shot. These keyframes are provided by LIG[3] and AT&T [7]. Because learning on a rich set of low-level features has been shown to be effective in improving the concept detection performance, we have significantly increased the number of feature types to be 98, by means of generating 13 different visual descriptors on 8 granularities (i.e., global, center, cross, grid, horizontal parts, horizontal center, vertical parts and vertical center)[1]. The relative performance within a given feature modality (e.g., color histogram vs color correlogram) is typically consistent across all concepts/topics, but the relative importance of one feature modality vs. another varies from one concept to the other.

We apply cross validation on the development data to evaluate the generalizability of each individual feature. In the following, we have listed a sample set of descriptors that achieved top overall performance for the concept modeling task:

- Color Histogram (CH)—global color represented as a 166-dimensional histogram in HSV color space.
- Color Correlogram (CC) — global color and structure represented as a 166-dimensional single-banded auto-correlogram in HSV space using 8 radii depths.
- Color Moments (CM) — localized color extracted from a 5x5 grid and represented by the first 3 moments for each grid region in Lab color space as a normalized 225-dimensional vector.
- Wavelet Texture (WT)—localized texture extracted from a 3x3 grid and represented by the normalized 108-dimensional vector of the normalized variances in 12 Haar wavelet sub-bands for each grid region.
- Edge Histogram (EH)—global edge histograms with 8 edge direction bins and 8 edge magnitude bins, based on a Sobel filter (64-dimensional).

We also generated 4 types of local SIFT-based features using the feature extraction tool provided by University of Amsterdam [11], e.g., SIFT, C-SIFT, RG-SIFT and Opponent-SIFT. For each type of local features, we created a 4000-dimensional codebook by clustering 1 million local features using k-means, and converted each keyframe into bag-of-word representations [10]. In particular, we used the Harris-Laplace interest point detector and soft bin assignment with a sigma parameter of 90.

[1]The final number of features is slightly smaller than expected because some of the visual descriptors are only generated on a selected set of granularities

2.2 Baseline Methods

We used the annotations officially provided by the collaborative annotation forum organized by LIG [3]. In the learning process, the development data are randomly partitioned into three collections: 70% as the training set, 15% as the validation set, and 15% as the held-out set. Most of our following algorithms are learned on the training and validation data, while the fusion strategies are determined based on the held-out data.

For each type of features, we applied both the baseline SVM learning algorithm without any data sampling, as well as an efficient ensemble approach called "robust subspace bagging" (RB-SBag), which enjoys several advantages over SVMs such as being highly efficient in learning/prediction, robustly performing with theoretical guarantee, and easy to parallelize on a distributed learning system [12]. From the training data, the algorithm first learns N base models, each of which is constructed from a balanced set of bootstrapped samples from the positive data and the negative data with sample ratio r_d, unless the sample size is larger than data size. On the feature side, if the training data contains multiple feature descriptors, such as color correlogram, edge histogram, etc., each descriptor is iteratively selected. Then the algorithm can either use the entire descriptor space, or further sample a subset of features with a rate of r_f. The default parameters for r_d is 0.2 and r_f is 1. Each model is associated with its 2-fold cross validation performance, where average precision is chosen in this case.

To minimize the sensitivity of the parameters for each base model, we choose the SVM parameters based on a grid search strategy. In our experiments, we build the SVM models with different values on the RBF kernel parameters, the relative cost factors of positive vs. negative examples, the feature normalization schemes, and the weights between training error and margin. The optimal learning parameters are selected based on the performance measure on the same 2-fold cross validation on training data. For each low-level feature, we select one optimal configuration to generate the concept models.

To reduce the risk of overfitting, we control the strength and correlation of the selected base models by adding a forward model selection step. In more details, we reserve a portion of the labeled training data to serve as a validation set $\mathcal{V}_c$ for forward model selection. The algorithm

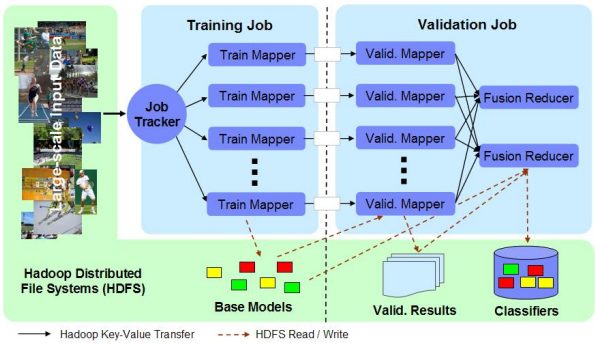

Figure 1: Illustration of the Map-Reduce implementation for Robust Subspace Bagging.

iteratively selects the most effective base model from the model pool, adds it to the composite classifier without replacement, and evaluates its average precision on $\mathcal{V}_c$. Finally, it outputs the ensemble classifier with the highest average precision, where the number of selected base models i is usually much smaller than N. This selection step is very fast, and typically prunes more than 70-80% base models in practice.

2.3 Distributed Learning with MapReduce and Hadoop

The need for distributed computing is apparent for modeling semantic concepts on massive multimedia data, which can range anywhere from tens of gigabytes, to terabytes or even perabytes. Inspired by the map and reduce functions commonly used in functional programming, Dean and Ghemawat [4] introduced a parallel computation paradigm called MapReduce. Its popular open-source implementation, Hadoop [1], has been successfully deployed to process hundreds of terabytes of data on at least 10,000 processors. Compared with other parallel programming frameworks, MapReduce provides the necessary simplicity by making the details of parallelization, fault-tolerance, data distribution and load balancing transparent to users. Also, this model is easily applicable to a wide range of data-intensive problems, such as machine learning, information extraction, indexing, graph construction and so on [4].

The programming model of MapReduce is as follows. Its basic data structures are a set of $\langle key, value \rangle$ pairs

with user-specific interpretation. Two individual functions are needed for any computation, called *Map* and *Reduce*. The *Map* function first reads a list of input keys and associated values, and produces a list of intermediate $\langle key, value \rangle$ pairs. After grouping and shuffling intermediate pairs with the same keys, the *Reduce* function is applied to perform merge operations on all intermediate pairs for each key, and to output pairs of $\langle key, value \rangle$[2]. This model provides sufficient high-level information for parallelization, where the Map function can be executed in parallel on non-overlapping data partitions, and the Reduce function can be executed in parallel on intermediate pairs with the same keys. Its abstraction can be summarized by the following pseudo-code,

$$\begin{aligned} map &: (k_1, v_1) \rightarrow list(k_2, v_2), \\ reduce &: (k_2, list(v_2)) \rightarrow list(k_3, v_3). \end{aligned}$$

Because of its ensemble structure, RB-SBag can be straightforwardly transformed into a two-stage MapReduce process. Figure 1 illustrates the main idea of the MapReduce implementation for RB-SBag based on Hadoop. The first MapReduce job only contains a *training map function*, designed to generate and store the pool of base models, without using any reduce functions. The abstraction for its input and output key-values can be written as,

$$map_t : ([i, t], L_{train}) \rightarrow ([i, t], L_h),$$

where i is concept index, t is the bag index and their joint vector $[i, t]$ forms the mapping keys. For values, L_{train} is the location of training data, and L_h is the location of the output base model h. After all the base models are produced, the next MapReduce job computes the prediction results on the validation set $\mathcal{V}$ using a *validation map function*, conducts forward model selection and combines multiple models into composite classifiers using a *fusion reduce function*. Its abstraction can be written similarly,

$$\begin{aligned} map_v &: ([i, t], L_h) \rightarrow list(i, [L_h, L^h_{pred}]), \\ reduce_f &: (i, list[L_h, L^h_{pred}]) \rightarrow (i, L_{Ci}), \end{aligned}$$

where L^h_{pred} refers to the location of prediction results on $\mathcal{V}$, and L_{Ci} refers to the final composite classifiers.

[2]Note that the input and output $\langle key, value \rangle$ pairs can have different formats

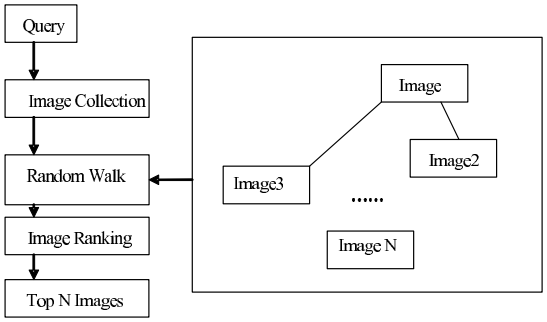

Figure 2: Random walk re-ranking of noisy Flickr data

2.4 Automatic Training Data Construction from Web Domain

In this section we consider the feasibility of leveraging open source multimedia data from various web resources. Online channels provide rich sources for multimedia training data. User-generated and user-tagged multimedia content can help us understand the visual semantics through crowd-sourcing, and improve the process of high-level feature detection in terms of cross-domain applicability and robustness. In particular, we consider images downloaded from Flickr as our main data source.

However, a large amount of noisy labels might be introduced if we simply use top-ranked examples from Flickr as the positive examples. Therefore, we applied a random walk model to automatically filter out the outlier images, while keeping the image diversity for the following learning step. Figure 2 shows the overview of our proposed model. The system takes a textual query q as input and downloads the top I images from Flickr. Following that, the association matrix among images M_{ij} is built based on visual similarity between images i and j, and a random walk is performed over it. Finally, we re-rank the set of images based on the random walk result and return the top N re-ranked images as the most relevant and diverse images for the given query.

Specifically, our basic model is based on the observation that if an image has more similar images in the initial Flickr list, it is more likely to be relevant. Therefore, we compute a random walk over the relevant images for query q, and measure the probability p^k_i that a random

Description	Run	Type	MAP
Global	A_ibm.Global_6	A	0.07925
Global + Local, Best 2 models	A_ibm.Combine2_5	A	0.11845
Global + Local, All best models	A_ibm.CombineMore_4	A	0.11995
Global + Local + TREC08, Single best model	A_ibm.Single+08_3	C	0.08515
A_ibm.Combine2_5 + Flickr	C_ibm.Combine2+FlkBox_2	C	0.0879
Best Overall Run	A_ibm.BOR_1	C	0.12355

Table 1: IBM TRECVID 2009 High level Feature Detection Task – Submitted Runs

surfer will end up at image i after k steps. Let $\vec{P}^k$ denote the probability vector over all images at step k. Initially, the probability to choose image i is equal for all images, while for subsequent steps, we perform a random walk using affinity matrix M:

$$\vec{P}^{k+1} = M \cdot \vec{P}^k, \quad p_i^0 = 1/I \tag{1}$$

In this case, the affinity matrix is calculated based on visual similarity between pairs of images. Given two images i and j and their feature vectors v_i and v_j, we firstly reduce the vector length by removing the vector positions where both vectors' value is 0. Assume the new vectors are v_i' and v_j', we calculate their cos-similarity as follows:

$$sim_{cos}(v_i', v_j') = \frac{v_i' * v_j'}{||v_i'|| * ||v_j'||}, \tag{2}$$

The final similarity between two images is the average cos-similarities over the *color_correlogram_global*, *wavelet_texture_global*, and *edge_histogram_global* features.

A standard result of linear algebra (e.g. [5]) states that if M is a symmetric matrix, and v is a vector not orthogonal to the principal eigenvector of the matrix, $\lambda_1(M)$, then the unit vector in the direction of $M^k v$ converges to $\lambda_1(M)$ as k increases to infinity. Here the affinity metric M is symmetric, and $\vec{P}^0$ is not orthogonal to $\lambda_1(M)$, so the sequence $\vec{P}^k$ converges to a limit $\vec{P}^*$, which signals the termination of our model.

Intuitively, the basic model above is able to improve the relevance of the image list with the given query, however, it tends to generate near-duplicated images in the final returned list, which is useless in many applications including image classification. To penalize the near-duplication and improve the diversity, we further propose an absorbing random walk framework as shown bellow.

- **Loop** until top N images are found:
  1. Find top image i_k by random walk over M;
  2. Set $M(i_k, *) = 0$ and repeat.

2.5 Fusion Methods

We applied ensemble fusion methods to combine all concept detection results generated by different modeling techniques or different features. In particular, we used a heuristic weighted linear fusion approach to merge the models. When more than one models are combined, the best model (according to the AP on the validate set) is assigned with a combination weight of 1, and the weight of a sub-optimal model is determined from the ratio of its AP to the AP of the best model. If the ratio is larger than 0.95, the sub-optimal model is assigned with weight 1. Similarly, we assigned weight 0.75 when $0.75 \leq ratio \leq 0.95$; 0.5 when $0.5 \leq ratio < 0.75$; and dropped models with $ratio < 0.5$.

To generate submission runs, we first apply the above ensembe fusion within the individual approaches and then fuse detection results as described below and in Table 1.

1. Global: Learning with global visual features using baseline SVM methods with RBF kernel, RB-SBag with RBF and χ^2 kernels;
2. Local: Learning with local features using RB-SBag with RBF and χ^2 kernels;
3. Flickr: Learning from auto-cleaned Flickr examples;
4. TREC08: The prediction results from the best IBM TRECVID'2008 runs on 10 overlapping concepts;
5. BOR: Best overall run by compiling the best models based on heldout performance for each concept.

2.6 Submitted Systems and Results

We have generated multiple runs of detection results based on the approaches presented before. A number of runs are submitted to NIST for official evaluation with their submission name shown. The mean inferred average precision is used as the measure of the overall performance of the systems. Table 1 lists the performance of the submitted runs. As can be observed, the baseline global run offers a reasonable starting performance for the following combination, but local features clearly provide considerable and complementary benefits to the prediction results using global features. Selecting and fusing the two best models from 5 models based on global and local features can improve the detection performance of Global baseline by 50%. Fusing more models can produce slightly higher performance, evidenced by the higher MAP of $A\_ibm.CombineMore\_4$. It is somewhat surprising that the automatically selected web data does not give any improvement over the original training data. This clearly shows the large domain gap between web images and documentary videos. Finally, the best overall run brings consistent improvement in MAP over runs of all flavors and raise the MAP to 0.123, or equivalently, 56% performance improvement over Global baseline.

3 Copy Detection

For the copy detection task, our focus was on the design and fusion of multiple complementary types of fingerprinting approaches. The overall processing flow is illustrated in Figure 3. We considered both frame-based and temporal sequence-based fingerprinting methods in visual and audio domains. We also considered two approaches for score normalization and fusion across systems that produce vastly different score distributions and ranges, and we used a novel fusion scheme that incorporates cross-detector agreement into the confidence-based fusion process. Finally, we leveraged approximate nearest-neighbor query techniques, and a sophisticated matching process, for improving scalability without increasing false alarm rates.

3.1 Fingerprinting Methods

We used two distinct methods for fingerprinting videos—one based on extracting temporal activity-based fingerprints over sequences of visual or audio frames; the other based on fingerprinting of individual frames. The frame-based approach is designed to detect very short segments of copied material, where strong matches at the frame level can be sufficient to infer segment duplication without false alarms. The temporal sequence-based approach on the other hand is designed to catch copied segments, where individual frames may not match strongly enough but weak consistent matches over a longer temporal sequence can accumulate enough evidence to declare a segment match without false alarms. The two approaches were designed to be complementary, although we note that due to the design of the TRECVID CBCD task itself, the frame-based approach performs much better since the copied segments in the synthetically generated TRECVID CBCD queries are generally very short (i.e., between 3 sec and 1 min).

3.1.1 Temporal Sequence-Based Fingerprints

This method is based on extracting feature vectors from short overlapping subsequences of variable length. We scan the video as a time series and detect interesting "events" along the time series, which form starting and ending points for each feature vector. The local minima and maxima of a frame-global energy feature are used as proxies for identifying these events. We select a minimum and maximum sequence duration, which are globally fixed constraints, and generate sets of overlapping event boundary-aligned segments spanning one or more events. We partition each of the generated segments into a fixed number of equi-sized regions, compute an overall energy measure for each region, and produce a fixed length vector which represents the time sequence. Each vector is normalized with respect to its mean and range, and indexed into a bracket with other vectors of the same temporal span for nearest neighbor lookup later. This process results in many overlapping fingerprint sequences of varying lengths, from the minimum to the maximum sequence duration. The temporal method is designed to be robust with respect to color transforms, blur, noise, compression and geometric transforms such as flipping and

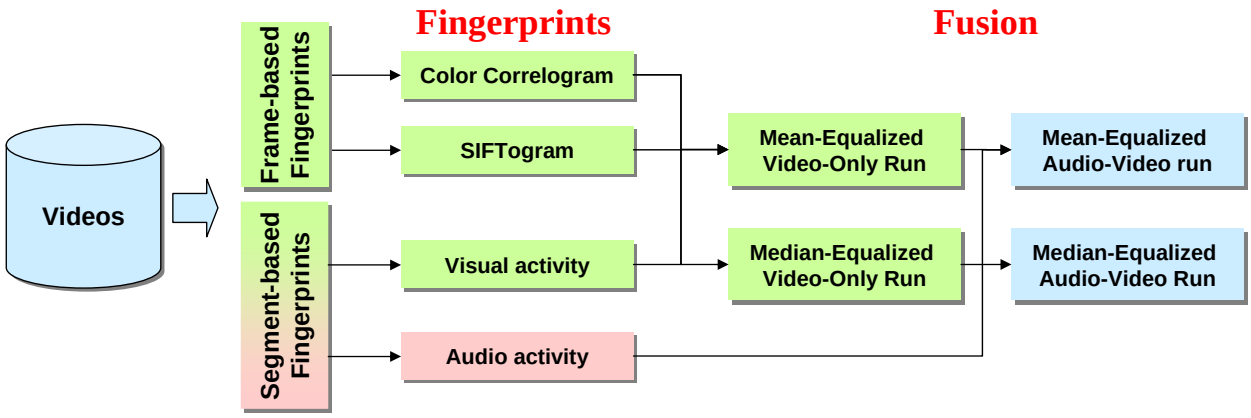

Figure 3: Overview of fingerprinting approaches and generated runs for TRECVID 2009 CBCD task.

rotation. It is however dependent on having a sufficiently long matched segment. We applied this technique for extracting temporal activity-based fingerprints both from the visual and audio domains using energy-based elemental features from each domain.

3.1.2 Frame-Based Color Fingerprints

With the frame-based methods, we sample video frames at a regular interval of 1 frame per second, and extract visual descriptors from each frame.[3] The first descriptor we considered was the *color correlogram* [6], which captures the local spatial correlation of pairs of colors, and is essentially a second-order statistic on the color distribution. The color correlogram is rotation-invariant and was designed to tolerate moderate changes in appearance and shape due to viewingpoint changes, camera zoom, noise, compression, and to some extent, shifts, crops, resizing and aspect ratio changes. We use a "cross"-layout formulation of the correlogram which extracts the descriptor from two horizontal and vertical central image stripes, emphasizing the center portion of the image and disregarding the corners. The cross formulation doubles the dimensionality of the descriptor but improves robustness due to text/logo overlay, borders, small crops and shifts, etc. It is also invariant to horizontal or vertical flips, while still capturing some spatial layout information. We extract an auto correlogram in a 166-dimensional quantized HSV color space, resulting in a 332-dimensional overal descriptor length for the cross color auto- correlogram feature vector. De-bordering and filtering blank frames as a preprocessing step avoids degenerate and / or useless feature vectors, and we use a contrast-limited histogram equalization to normalize contrast and gamma. The reference vectors are then indexed for nearest neighbor lookup later. The correlogram fingerprint performs well against mild to moderate geometric transforms, resampling, noise, and linear intensity changes but does not handle non-linear gamma correction changes or hue/saturation transforms, which lead to dramatic changes in color.

3.1.3 Frame-Based SIFTogram Fingerprints

SIFT features have been found to be very powerful descriptors for image retrieval'[8]. However, computationally, it is very difficult to scale local point matching against a large reference set of video frames and corresponding image patches. For example, a typical image will generate on the order of 500 interest points and corresponding SIFT features, which would result in roughly 700M image patch fingerprints for our reference set of 1.4M frames extracted from the 400-hour TRECVID CBCD reference set. Instead, we use the "bag-of-words" approach [10] to leverage the retrieval power and color-, rotation-, shift-, and scale-invariance of lo-

[3]We did not bother reducing the number of frames by applying a keyframe detector since we wanted to oversample frames for better robustness, and did not want to depend on the accuracy of a keyframe detector.

cal features while balancing computation time. Following this method, we extract SIFT local features from the reference videos and generate a codebook of representatives by clustering 1M sample SIFT features into 1000 clusters. The centroids of these clusters become *visual codewords*, which are used to quantize any SIFT feature into a discrete visual word. For each sampled frame in the reference and query video sets, we then compute a histogram of the codewords, making a global feature from the set of local ones. The number of codewords is the dimensionality of the feature vector, in our case, 1000. We made use of the "Color Descriptor" software from the University of Amsterdam [11]. In particular, we used the Harris-Laplace interest point detector, the plain SIFT descriptor, and soft bin assignment with a sigma parameter of 90. Once extracted, reference set features are placed into a nearest neighbor index for retrieval later. Query videos are processed using the same pre-generated codebook. This "SIFTogram" feature is robust with respect to changes in colors, gamma, rotation, scale, shift, borders and a certain amount of crop or overlaid text/graphics.

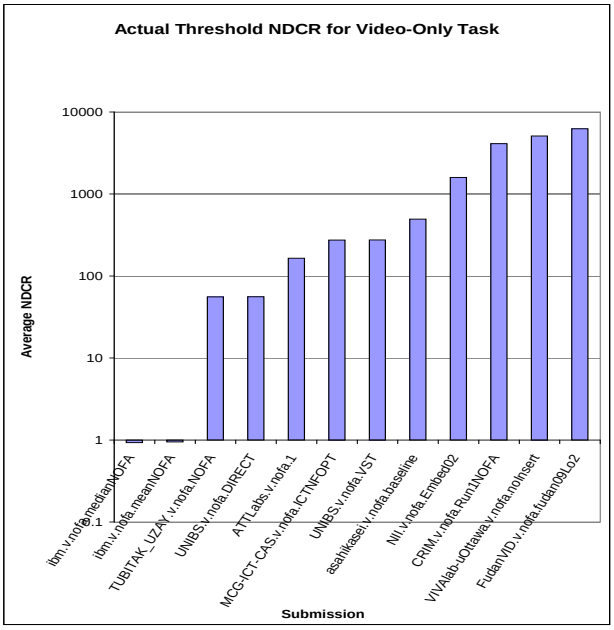

Figure 4: NOFA submissions Actual NDCR for 2009

3.2 Indexing

To enable us to do experiments quickly, and substantially reduce our query processing time, we used an approximate nearest neighbor index for querying the reference set features. Specifically, for the temporal sequence-based descriptors, we used the ANN library[4] [2] for approximate nearest-neighbor searching, and for the frame-based descriptors, we used the Fast Library for Approximate Nearest Neighbor (FLANN) package[5] [9]. Both are open-source, and of the two, FLANN is more recent and includes tools for automating indexing algorithm and parameter selection based on the data set being indexed, which ultimately lead to better performance.[6] We found that with the correlogram based visual method, FLANN sped up our query times by a factor of 50 without having a significant effect on accuracy.

[4]http://www.cs.umd.edu/ mount/ANN/

[5]http://www.cs.ubc.ca/ mariusm/index.php/FLANN/FLANN

[6]We used two libraries for approximate nearest-neighbor search due to pre-existing legacy code, which was based on the older ANN library.

3.3 Fusion

Each of our submissions was the result of fusing together two or more of our four methods. For the video-only task submissions, we fused the results from the temporal sequence method and the cross color correlogram (or "visual") method. For the audio+video task, we fused those two along with the SIFTogram and temporal audio sequence methods. We used a relatively simple fusion process, dividing the score from each component's best match by either the mean or median score from the 2008 data for that component. We found that the median worked better, due to the fact that outliers on the high end of scores (very positive matches) would raise the mean, causing the scores from that component method to be deemphasized in the fused result, if that inflated mean was used as a normalization factor. We also added a bonus to a fused match based on the F1 score of the combined result. We note that better performance could likely be achieved, particularly in a truly balanced-cost profile, if we considered fusion of secondary results as well as the highest-scoring results from each method.

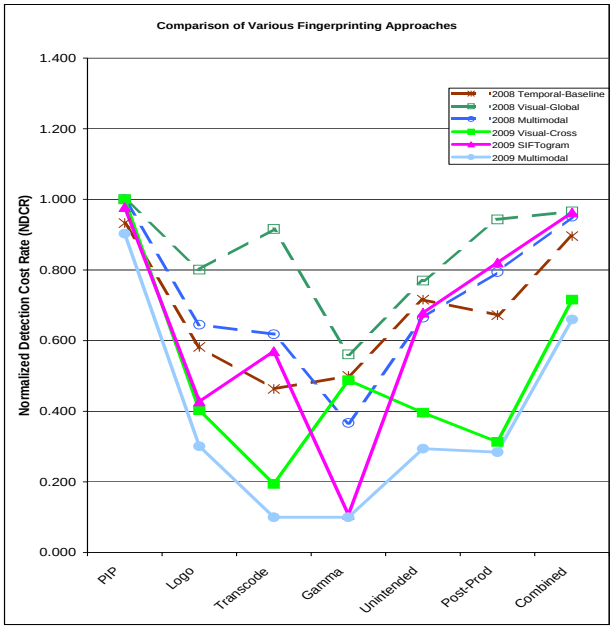

Figure 5: Component run results on TV-2008 data

3.4 CBCD Results

For our video-only submissions, we used just the temporal and color correlogram based methods. For the later submissions in the audio+video task, we also added the SIFTogram and audio-temporal methods. We focused the tuning of our system performance on the transformations that we felt were most likely to be encountered on the web, in live copy detection systems. This comprised transforms three through six: insertions of a pattern, reencoding, change of gamma, and decrease in quality. These transformations can occur as users copy and share videos without even intending to transform the video. We anticipated that our system, with its exclusive use of frame-global features, would be unlikely to do well on the picture-in-picture transform or on some of the heavy post-production type of transforms, and we simply focused on eliminating false alarms for those cases.

3.4.1 A Note on Application Profiles

Our discussion of results focuses on the "no false alarms" profile. We have found that zero false alarms is a highly desired feature in systems for commercial deployment. Additionally, for the CBCD task, the "balanced" profile is actually very similar to the NOFA profile. The Normalized Detection Cost Ratio is computed as

$$NDCR = P_{miss} + \beta * R_{FA}$$

where

$$\beta = CFA/(C_{miss} * R_{target}) = 2$$

for the balanced profile. The false alarm rate R_{FA}, is defined as

$$R_{FA} = FP/T_{queries}$$

where $T_{queries} \approx 7.3$ hours for the 201 queries in the 2009 dataset. Therefore,

$$NDCR \approx P_{miss} + 0.28 * FP$$

This means that each false alarm increases NDCR by 0.28. Since we can obtain a trivial NDCR of 1.0 by submitting an empty result set, the balanced profile is essentially a "3-false-alarm profile".

Furthermore, in our analysis of the reported results, we focus on the optimal NDCR metric, rather than considering the actual submitted thresholds. This is due to the fact that nearly universally, the thresholds submitted by participants resulted in NDCR scores greater than 1.0 (i.e., at least 1 false alarm), which would be worse than a trivial empty submission. Figure 4 shows the actual NDCR scores for submitted thresholds for the NOFA profile of the video-only task. Although the IBM runs did achieve an actual NDCR slightly less than 1, we think these results underscore the fact that the difficulty of threshold selection is an interesting problem by itself. One possible explanation for this poor generalization of threshold by nearly all participants could be due to the fact that the transformation parameters used in the TRECVID 2009 CBCD task were slightly changed from those in 2008. Therefore, using the 2008 training dataset to optimize parameters and select thresholds did not generalize to the 2009 data. It may be that a deployed copy detection system would have to determine its operating thresholds in-situ, as queries are processed, and feedback is manually entered into the system.

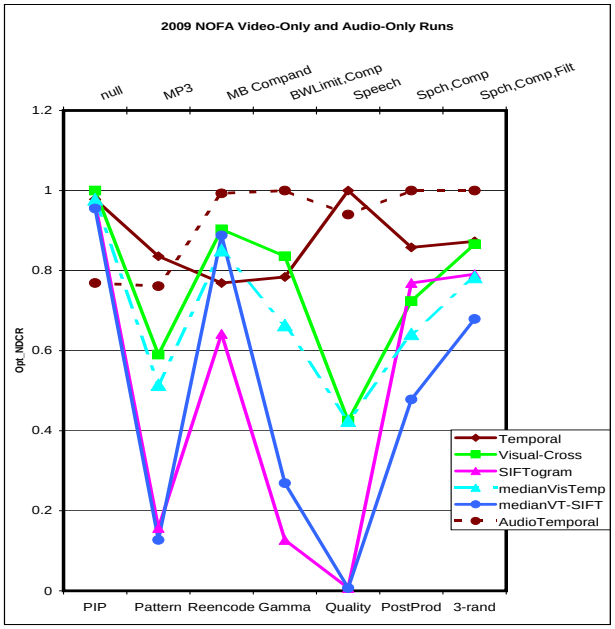

Figure 6: Component run results on TV-2009 data

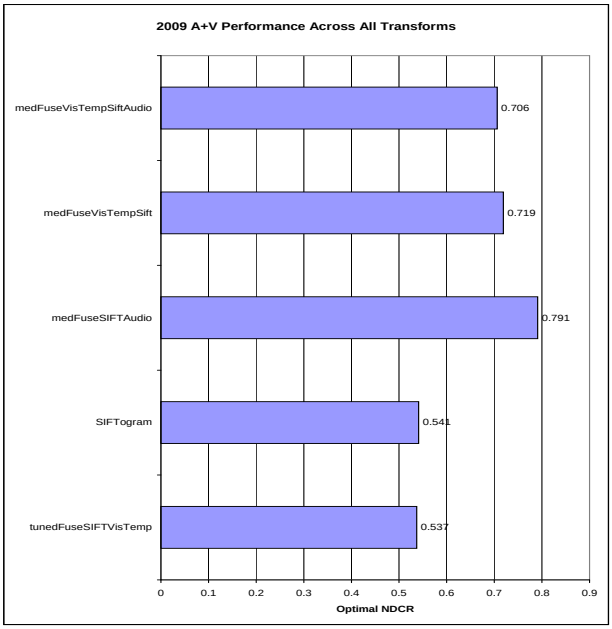

Figure 7: A+V Results for various IBM Runs on 2009 Data

3.4.2 Discussion of Results

We tested each of the methods we developed or improved for TRECVID 2009 on the TRECVID CBCD 2008 data and obtained the results shown in figure 5. Figure 5 also shows our results from our 2008 submissions for comparison. Finally, figure 5 also shows the fused runs which combine the results of the temporal, color and SIFTogram methods. These fused runs are labeled in the figure 5 as 2008 Multimodal and 2009 Multimodal. The "2008 Multimodal" run was a fusion of the 2008 temporal method and the 2008 visual method. The run labeled "2009 Multimodal" fused together the 2009 temporal, 2009 visual and SIFTogram methods. By tuning our fusion parameters, we were able to generate a fused run ("2009 Multimodal") which outperformed each single component run on the 2008 data. Figure 6 shows the corresponding results for the 2009 methods on the 2009 data. It turned out that for the 2009 data set, the fusion method was not always superior. In fact, when a single optimal NDCR threshold is selected for all transforms, instead of per-transform, the minimal NDCR for the SIFTogram alone is significantly better than the same measure for the fused runs which we submitted, as can be seen in figure 7. In that figure, "medFuseVisTempSiftAudio" refers to our submitted audio+visual run, which was produced by fusing together component scores, using the median score value as a normalizing factor. The line labeled "SIFTogram" shows the results for just that component. After receiving the ground truth for the 2009 results, we still found it difficult to improve on the SIFTogram result solely by fusing with different normalizing parameters. The run "tunedFuseSIFTVisTemp" represents a fusion of the video temporal, color and SIFTogram methods that slightly outperforms the SIFTogram-only method, but the difference is negligible.

In comparing our submitted runs with those of other participants, we are pleased to report leading performance for our ibm.m.nofa.medFuse run on transform 6 in the audio+video task, which we had targeted. In figure 9 we show the optimal NDCR for the T6-related AV transforms alongside other submissions with an NDCR less than 1. Transform 6 consisted of 3 transformations chosen randomly from the following: blur, change of gamma (T4), frame dropping, contrast, compression (T3), ratio, and addition of white noise. As can be seen from figure 6,

our performance on this transform is due largely to the SIFTogram component. We did not have the SIFTogram component of our system ready for the video-only run submission, but we later computed the results of this method on the 2009 video-only queries, and they are shown in 8. Figure 8 shows the optimal NDCR for a single optimal threshold chosen for all the query videos transformed with T3, T4, T5 and T6. Our SIFTogram run would have outperformed our component fusion runs on this measure, as well as the runs submitted by other participants, on these 4 transforms. We also note that our system achieved a high degree of performance considering the CPU time used, as can be seen in figure 10. This graph shows the CPU time consumed per query video plotted against the optimal NDCR achieved over all the video-only queries for all of the participants. The IBM runs are marked in blue, while other runs are red. There was a wide range of reported CPU times, so we have used a log scale on the horizontal axis. Points closest to the origin are preferred, and by this metric our submitted ibm.m.nofa.medFuse run is surpassed by only one other institution. We also include 3 runs in figure 10 which were not submitted. They are the color frame based method and the SIFTogram method alone, along with a fusion of the two. The SIFTogram performs best, but at a higher computational cost. The reported times are for a quad-core, parallel implementation.

In summary, our system makes use of frame-global features in a focused effort to trade off some limitations to achieve higher speed. The system still manages to do very well on transforms related to quality degradation which are common online.

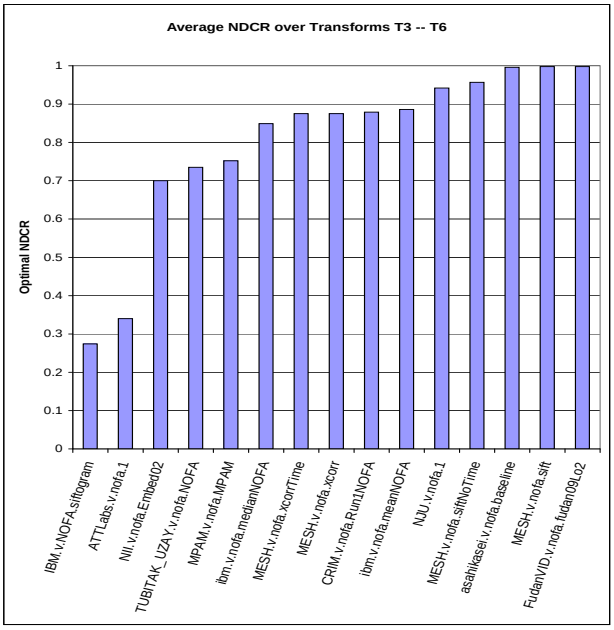

Figure 8: Video-Only Results for T3-T6 queries in aggregate. We show an "unofficial" run of our SIFTogram method compared with the fused runs we submitted and other participants.

4 Conclusions

IBM Research team participated in the TREC Video Retrieval Track Concept Detection and Copy Detection tasks. In this paper, we have presented results and experiments for both tasks. For Concept Detection, we found global and local features to be complementary to each other, and their fusion results outperform either individual types of features. We also note that the more features are combined, the better the performance, even with simple combination rules. Finally, development data collected automatically from the web domain are shown to be useful on a number of the concepts. On Content Based Copy Detection, the SIFTogram method with approximate nearest neighbor indexing proved to be particularly efficient and highly accurate on the targeted transforms.

5 Acknowledgments

This material is based upon work supported by the US Government. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the US Government.

References

[1] Hadoop. http://hadoop.apache.org/.

[2] S. Arya and D. Mount. ANN: library for approximate nearest neighbor searching. *Trees*, 3:4.

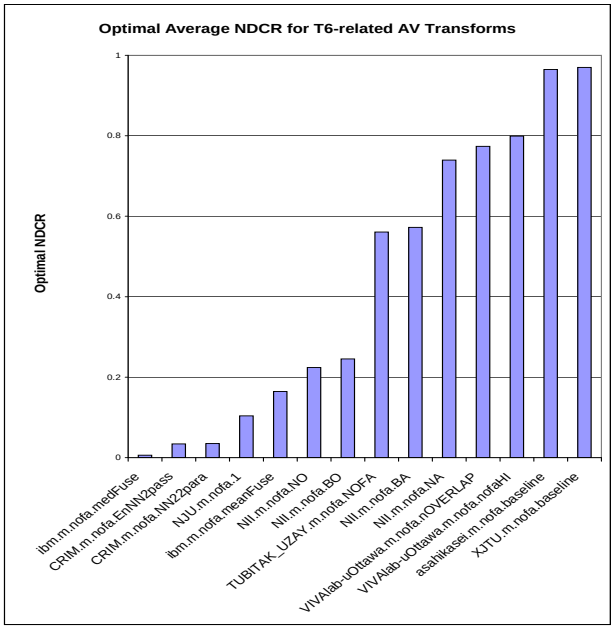

Figure 9: Audio+Video Results for T6. IBM's best run is on the left.

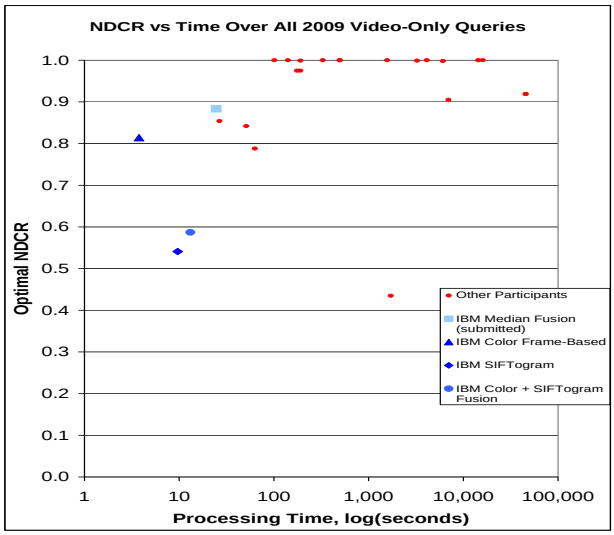

Figure 10: Log time vs NDCR for various participants - IBM in blue

[3] S. Ayache and G. Quenot. Evaluation of active learning strategies for video indexing. In *Proceedings of Fifth International Workshop on Content-Based Multimedia Indexing (CBMI'07)*, 2007.

[4] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, 2008.

[5] G. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1989.

[6] J. Huang, S. Kumar, M. Mitra, W. Zhu, and R. Zabih. Spatial color indexing and applications. *International Journal of Computer Vision*, 35(3), December 1999.

[7] Z. Liu, D. Gibbon, E. Zavesky, B. Shahraray, and P. Haffner. A fast, comprehensive shot boundary determination system. In *Proc. of IEEE International Conference on Multimedia and Expo*, 2007.

[8] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[9] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications (VISAPP'09)*, 2009.

[10] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. In *ICCV*, 2005.

[11] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, in press, 2010.

[12] R. Yan, M. Fleury, M. Merler, A. Natsev, and J. R. Smith. Large-scale multimedia semantic concept modeling using robust subspace bagging and mapreduce. In *ACM Multimedia Workshop on Large-Scale Multimedia Retrieval and Mining (LS-MMRM)*, Oct. 2009.