CRIM Notebook Paper - TRECVID 2011

Surveillance Event Detection

S. Foucher, *Member, IEEE,* M. Lalonde, *Member, IEEE*, L. Gagnon, *Member, IEEE*

Centre de recherche informatique de Montréal (CRIM)
405, Avenue Ogilvy, bureau 101
Montreal (Quebec) H3N1M3
{Samuel.Foucher, Marc.Lalonde, Langis.Gagnon}@crim.ca

Abstract

Approach we have tested in each of your submitted runs. For the "Object Put" event, we followed a dual foreground segmentation approach where the output difference between a short term and a long term model is used for triggering potential alerts. For Pointing, Embrace, CellToEar and PersonRuns, we applied the learning of compound spatio-temporal features based on a data mining method.

Relative contribution of each component of our approach. Our system is based on an action recognition approach which is mining spatio-temporal corners in order to detect configurations, called Compound Features, typical of an action of interest. The final detection is based on blobs around local frame-to-frame changes that are containing enough relevant compound features.

What we learned about runs/approaches and the research question(s) that motivated them. Overall, performances have improved from last year especially for PersonRuns. In addition, the training for PersonRuns was based on a standard action recognition dataset (KTH) independent of the TrecVid dataset which indicates that our implementation is behaving as expected. For Pointing, Embrace and CellToEar, results are not satisfying yet and the main reason is probably due to the fact that the training dataset derived from the development videos presents a large variability, is too noisy and too small in size in order to produce good rules. Also, given the complexity of the scenes composed of multiple action occurrences, occlusions and complex actions, the direct application of an action recognition method is a challenge. Going forward, performances could be improved if combined with other approaches such as a person tracker and also if the quality of the training set could be improved.

Introduction

This is the second year of participation for CRIM to the SED task and this time we provided results on five events by adding the Embrace and CellToEar ones. For this year, we focused primarily on improving the action recognition algorithm that used to be at the heart of our Pointing method in 2010 but was adapted this year for PersonRuns, Pointing, Embrace and CellToEar. The size of the training sets was significantly increased and was supplemented by the standard action recognition dataset KTH [4].

All the computations were performed on the "Mammouth" supercomputer located at the Center for Scientific Computing at the Université de Sherbrooke.

I – Object Put Event

The "Object Put" algorithm was unchanged from last year and is based on a very simple dual background model approach described in [1]. This year we were aiming at optimizing the detection threshold and the learning rate values for the short and long term models. The performance on Eval08 after parameter tuning is shown in Figure 1. However, this new set of parameters didn't lead to better performances compared to last year (see Section III).

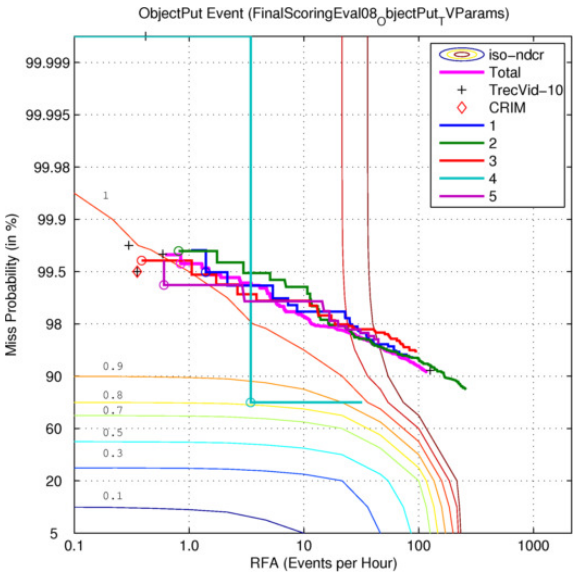

Figure 1: DET curve on Eval08 for ObjectPut.

II – Action Recognition

We pursued last year's implementation of an action recognition approach based on the hierarchical learning of Compound Features (CF) proposed recently by Gilbert *et al*. [2][3]. This method was then applied to PersonRuns, CellToEar, Pointing and Embrace. The following steps are involved:

1. Build an overcomplete set of Harris corners at various spatial scale and in the temporal domain.
2. Group corners within a 3x3x3 neighbourhood to form CF
3. CF are then encoded using information about cell position, scale and corner type to form transactions (or itemsets).
4. A data mining algorithm (APriori algorithm [5]) is applied in order to extract frequent itemsets from all the recorded level 1 transactions observed on the training set.
5. Transaction rules and associated confidence levels are derived from the frequent itemsets.
6. Rules that have fired at level 1 are used to form CF within a 6x6x6 neighbourhood at level 2.
7. The same data mining algorithm (step 4) is applied on level 2 transactions.
8. A third level of CF is formed but with a 2x2x2 spatio-temporal neighbourhood with cells extending to the limit of the image.

The last level (step 8) implicitly assumes that only one instance of a particular action is taking place at the same time which is of course rarely the case in the TrecVid data. Therefore, we slightly modified the last stage so that the spatial extent of the neighbourhood takes into account the expected size of a person given a position in the image and is derived from our camera geometric model (see [1] for details).

Training

Compared to last year, we greatly increased the number of training samples. In addition to samples taken from the evaluation set we added videos from the KTH dataset [4].

For Pointing, Embrace and CellToEar, we manually annotated some events in the TrecVid development set with the following guideline:

1. No action mixing (e.g. no walking and pointing).
2. No occlusions
3. The bounding box should encompass the total spatial extent of the event.
4. One single occurrence of the event during the event timeframe.

The resulting number of training samples is shown in Table 1. The negative examples were taken from the same TrecVid videos but outside the event bounding box. We also added training samples for which the video frame has been flipped horizontally; otherwise learned rules may be biased by the dominant people motion (for instance, most people move from left to right in Camera 1). So the total number of positive events for Pointing is 620 with a majority of events taken from Camera 1. For PersonRuns, because it was too difficult to build a ground truth on the TrecVid videos, we chose instead to take the KTH videos for the Running/Jogging actions as positive samples (25 persons running 4 times in 4 videos) and the Walking action as negative.

	TRECVID					KTH			
Events	**CAM1**	**CAM2**	**CAM3**	**CAM4**	**CAM5**	**Running**	**Jogging**	**Walking**	**Total**
Running						400	400	400	1200
Pointing	234	61	8		7				310 (1426 secs)
Embrace	12	33	125						170 (1114 secs)
CellToEar	10			37	41				88 (265 secs)

Table 1: Size of the training set for each event.

The resulting number of transactions at level 1 went from 1 million last year to about 30 millions for this year. The data mining algorithm (Apriori [5]) was run so that we are looking for rules with a minimum support of 5% and a minimum confidence level equal to the fraction of positive transactions if greater than 20%. The minimum support threshold is necessary in order to make sure that the derived rules are statistically significant. We also limited the rule size to 5 as recommended by [2]. For PersonRuns, we get 3886 rules at level 1, 3599 at level 2 and 147848 rules at level 3.

Detection

Another important issue is how to take a reliable decision in the presence of an event in a scene where many other actions are taking place (e.g. people walking). The original method derives a probability map from the firing rules as shown in Figure 2, in order to improve the map, we applied a Gaussian and temporal filtering; however this map is still difficult to threshold.

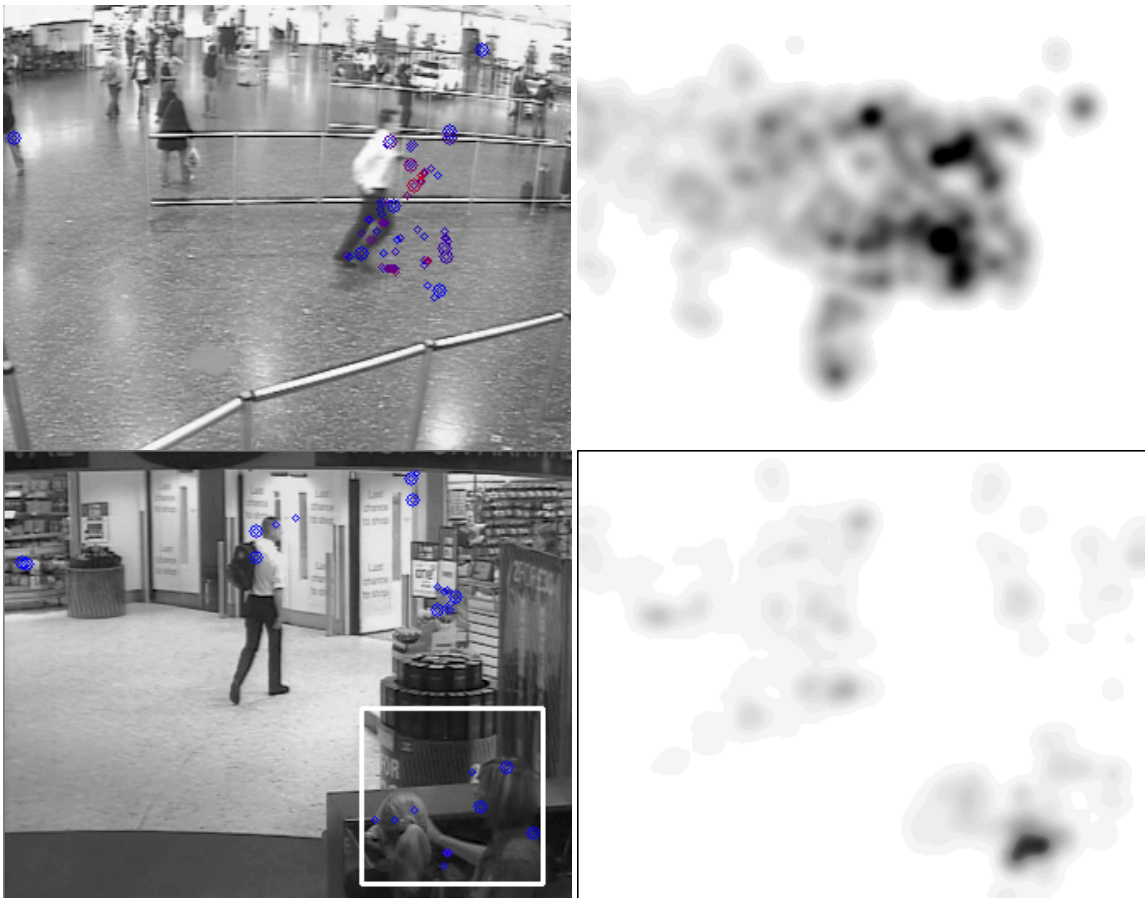

Figure 2: Rules firing for PersonRuns (top left) and corresponding probability map (top right, dark means high probability value). Rules firing for Pointing (bottom left) and corresponding probability map (top right, dark means high probability value).

Therefore, we adopted a simpler approach where we segment the frame-to-frame difference image leading to areas of motions (see Figure 3). We then compute the density of rules firing within each region of changes. The region with the higher rule density value is chosen and triggers an event if the average confidence value within this region is above a given threshold.

Figure 3: Action detection for PersonRuns based on a blob from a change detection.

III - Results

Results on the Development Set (Eval08)

We evaluated the performance on the Eval08 dataset (25 videos). We can see that the performance for PersonRuns (Figure 4 - left) was significantly increased when compared to last year's result on Eval09 with a global Min NDCR at 0.958 (PMiss=0.9221, RFA= 7.13). For Pointing (Figure 4 - right), the performance also increased notably with a shift of the DET curve to the left by one order of magnitude. For Embrace (Figure 5 - left), results are still one order of magnitude larger than the other teams in terms of RFA except maybe for Camera 5. Performance for CellToEar (Figure 5 - right) is similar to Embrace and close to performance levels reached by other teams last year.

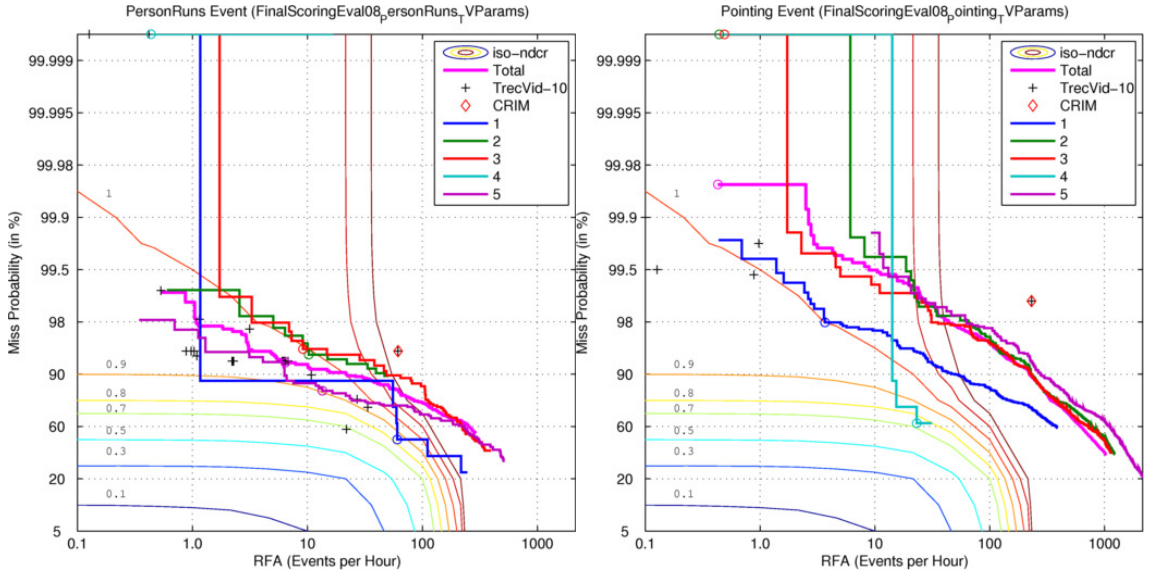

Figure 4: DET curves for PersonRuns and Pointing on the Eval08 dataset. Last year Min NDCR position for CRIM is shown as a red diamond marker.

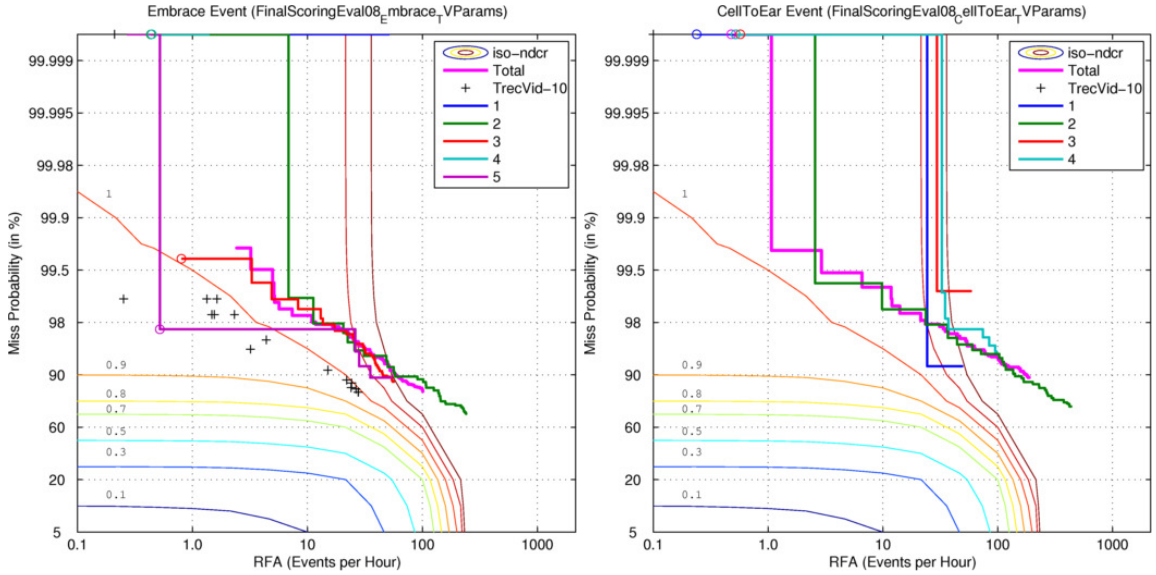

Figure 5: DET curves for Embrace (left) and CellToEar (right) on the Eval08 dataset. Last year Min NDCR position for CRIM is shown as a red diamond marker. Black crosses indicate Min NDCR positions on Eval09 by the other teams during the TrecVid-2010 competition.

Results on the Test Set (Eval09)

Results provided by NIST (see Table 2 and Figure 6 below) on Eval09 are consistent with what we observed on Eval08, the best Min NDCR was obtained for PersonRuns. However, the level of false alarms is higher for Pointing compared to Embrace and CellToEar despite the fact that it is based on a larger training set.

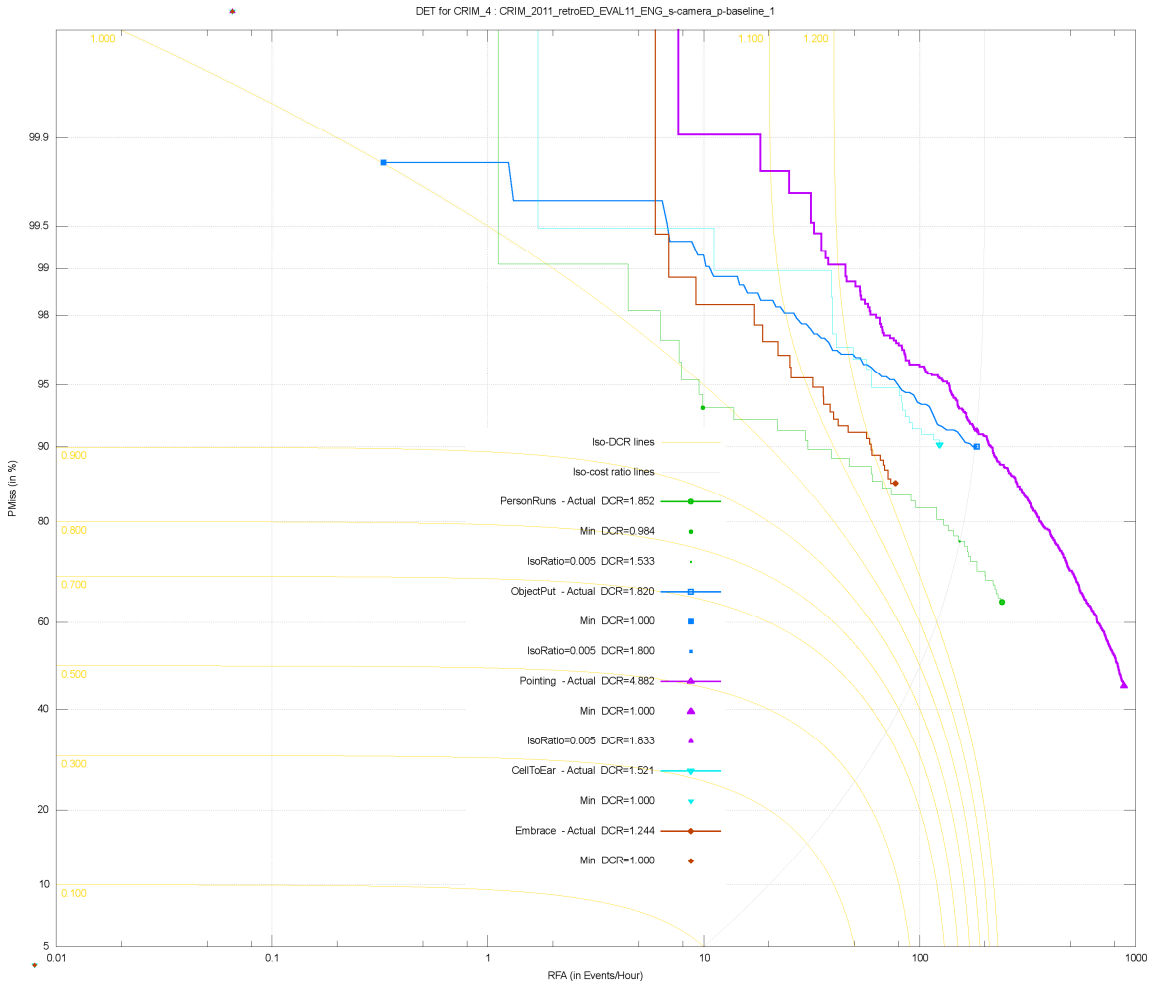

Figure 6: DET curves for PersonRuns and Pointing on the Eval09 dataset.

	Inputs			Actual Decision DCR Analysis								Minimum DCR Analysis			
	#Targ	#NTarg	#Sys	#CorDet	#Cor!Det	#FA	#Miss	RFA	PMiss	DCR	Dec. Tresh	RFA	PMiss	DCR	Dec. Thresh
CellToEar	194	1888	1907	19	0	1888	175	123.8258	0.9021	1.5212	0.9615	0.0656	1.0000	1.0003	3.8983
Embrace	175	1180	1205	25	0	1180	150	77.3911	0.8571	1.2441	1.2588	0.0656	1.0000	1.0003	5.2649
ObjectPut	621	2805	2867	62	0	2805	559	183.9679	0.9002	1.8200	0.6988	0.3279	0.9984	1.0000	1.0000
PersonRuns	107	3682	3720	38	0	3682	69	241.4866	0.6449	1.8523	2.5003	9.9034	0.9346	0.9841	26.4594
Pointing	1063	13507	14089	582	0	13507	481	885.8663	0.4525	4.8818	2.4222	0.0656	1.0000	1.0003	66.0428

Table 2: Actual Miss rate and False Alarm rates on Eval09 for each event.

Conclusions

For our second year in this SED evaluation campaign, the objective was to improve results from last year especially regarding the level of false alarms. We also provided results for two more events (CellToEar and Embrace). Results from the action recognition method are mixed due to the difficulty to form a clean training set from the TrecVid videos. Results on PersonRuns are promising but the detection step based on blobs is probably not optimal in all situations (e.g. someone running among a walking crowd).

The original action recognition method was not designed for a scene with multiple occurrences of the same action (e.g. several people walking). In particular, the last step of the method is problematic as it looks for rules in the entire image. We modified this step for a local approach with a neighborhood function of the camera geometric model. Results on the various cameras show that the performance is affected by the level of clutter and the object resolution in the scene. Cameras 1 and 4 with lesser depth of view have generally better results.

Increasing the training database will likely require the use of a 64-bit version of the APriori algorithm as we have reached the memory limit under windows 32-bit (~ 2 Gb in memory). However, the main difficulty for the training is to build a "clean" database of action units. The few samples taken from the evaluation corpus usually exhibit large variations in pose, background clutter and are usually composed of a mixture of actions (e.g. person walking and pointing). An idea could be to supplement the TrecVid samples with an in-house database in order to reinforce relevant action patterns.

The detection process needs to be refined also. We have adopted a strategy different from Gilbert *et al.* but that is probably not optimal in all situations. The current version runs on smaller frame size at about 4-5 fps, this loss of resolution does not help the detection process especially for actions in the background.

Acknowledgments

This work has been supported by MDEIE of the "Gouvernement du Québec". The authors wish to thank the RQCHP (Compute Canada) for giving access and support to their high performance computing resources.

References

[1] S. Foucher, M. Lalonde, L. Gagnon, "CRIM Notebook Paper - TRECVID 2010 Surveillance Event Detection", In NIST TREC Video Retrieval Evaluation Workshop 2010 (TRECVID 2010). Gaithesburg, MD, November 15-16, 2010.

[2] A Gilbert, J. Illingworth, R.Bowden, "Action Recognition using Mined Hierarchical Compound Features", Accepted for IEEE Trans Pattern Analysis and Machine Learning. 2010.

[3] A. Gilbert , J. Illingworth, R. Bowden, "Fast Realistic Multi-Action Recognition using Mined Dense Spatio-temporal Features", In Proc. Int. Conference Computer Vision (ICCV09), Kyoto, Japan.

[4] C. Schuldt, I. Laptev, B. Caputo, "Recognizing Human Actions: a Local SVM Approach," In Proc. of International Conference on Pattern Recognition (ICPR'04), vol. III, pp. 32–36, 2004.

[5] Christian Borgelt, "Recursion Pruning for the Apriori Algorithm", 2nd Workshop of Frequent Item Set Mining Implementations (FIMI 2004, Brighton, UK).

CRIM AT TRECVID-2011: CONTENT-BASED COPY DETECTION USING NEAREST-NEIGHBOR MAPPING

Vishwa Gupta, Parisa Darvish Zadeh Varcheie[1]*, Langis Gagnon, Gilles Boulianne*

Centre de recherche informatique de Montréal (CRIM)

{Vishwa.Gupta, Parisa.Darvish, langis.gagnon, gilles.boulianne}@crim.ca

ABSTRACT

We report results on content-based audio and video copy detection for TRECVID 2011 CBCD evaluation using nearest-neighbor mapping. The nearest-neighbor mapping was used successfully in audio copy detection for TRECVID 2009 with excellent results (min NDCR of 0.06 averaged over all seven transforms for actual no FA case). For this reason, we decided to implement nearest-neighbor mapping for video copy detection also. For video copy detection using nearest-neighbor mapping, the idea is to first map each video frame of the test to the closest query video frame. We then move the query over the test to find the test segment with the highest number of matching frames. This nearest-neighbor mapping lead to good matching scores even when the query video was distorted and contained occlusions. We test these algorithms on TRECVID 2009 and 2010 content-based copy detection evaluation data. For both these tasks, the nearest-neighbor video copy detection gives minimal normalized detection cost rate (min NDCR) comparable to that achieved with audio copy detection for the same task.

We augment audio copy detection by using three different feature parameters: MFCC, equalized MFCC, and Gaussianized MFCC. Pooling the results from the three feature parameters gives the lowest miss rate, and when combined with video copy detection, we get significantly improved audio+video copy detection results. For TRECVID-2011 CBCD evaluation, we get the lowest min NDCR for 25 out of 56 transforms for the actual no FA case. All our runs (V48A66T58B, V48A66T65B, V48A66T160, V48A66T60) were the same except for the thresholds.

Index Terms— audio copy detection, video copy detection, content-based copy detection.

1. INTRODUCTION

There are many applications of audio and video copy detection: for copyright control, for monitoring advertisement campaigns of businesses, for monitoring ads of competitors for business intelligence, and for law enforcement investigations. Content-based copy detection offers an alternative to watermarking. In watermarking, only the content that has been watermarked can be detected, while content-based copy detection can detect any copy for which there is a copy in the search database.

The content-based copy detection got a big boost with the TRECVID-2008 CBCD (content-based copy detection) evaluation, and has continued with TRECVID 2009, 2010, and 2011 evaluations. Many research labs have participated in these CBCD evaluations with many different algorithms for copy detection. The copy detection performance has improved significantly in the last 3 years. In TRECVID 2008, the emphasis was on copy detection with only a small penalty for false alarms. In TRECVID 2009, the emphasis shifted to no FA (false alarms) case, with a penalty of 1000 for each false alarm. The no FA case was divided into two: optimal and actual. In the optimal case, a separate threshold for rejection per transform (see Sec. 5) is computed to minimize NDCR (normalized detection cost rate). In the actual case, one threshold for all the transforms is specified *a priori* by the participants. This threshold is then used to estimate the min NDCR for all the transforms. So the actual case is much more difficult than the optimal case.

TRECVID 2009 is the last time when NIST evaluated audio, video, and audio+video copy detection separately. For this evaluation, CRIM achieved the lowest min NDCR for audio only copy detection [1] for all categories (optimal balanced, actual balanced, optimal no FA, actual no FA) and for all the seven transformations of the audio queries. Most of our results were around min NDCR of 0.06. This was primarily due to the nearest-neighbor mapping [2][3] that was used to map test frames to the nearest query frames. For TRECVID 2009 *Video only* copy detection, ATT labs [1][4] got the lowest min NDCR for *optimal no FA* runs. Their min NDCR varied between 0.22 to 0.68. The best *actual no FA* results were shared by three different labs (for different transforms) [1], and the min NDCR varied between 0.22 and 0.69. Even though CRIM got only average results in *video only* copy detection evaluations for TRECVID 2009, we still got the lowest min NDCR for *audio+video* evaluations for *actual balanced* and *actual no FA* runs for all the 49 transforms [1]. This was primarily due to the *audio only* copy detection results.

Because nearest-neighbor mapping gave such good results for audio copy detection, we decided to implement it for video copy detection also. The first step in video copy detection using nearest-neighbor mapping is to choose frame-based local video features that are suitable for nearest-neighbor search. In 2010 TRECVID CBCD evaluations, Peking University [5] [6] got the lowest min NDCR. They used many different detectors over local and global visual features, and then fused the results. It is difficult to pick one feature that stands out. However, NTT [7] also got very good results with only one feature set: temporally normalized local visual features. We used these features as our starting point. We computed seven discrete temporally normalized features per frame and used these features to search for video copies. Using these features, we show that the nearest-neighbor search outperforms the search algorithm similar to that used by NTT. We also show that 16 unquantized features worked much better than the seven quantized features when we use nearest-neighbor search. The min NDCR for video copy detection that we achieved for TRECVID 2009 data varies between 0 and 0.097. This is better than the average min NDCR of 0.06 that we achieved for audio copy detection in TRECVID 2009. This is better than the best TRECVID 2009 *video only* copy detection results (with

[1]Parisa Darvish is now with Genetec Inc, Montreal, pdarvish@genetec.com

min NDCR varying between 0.22 and 0.68) [1].

We have enhanced audio copy detection by using three different feature parameters for search: MFCC, equalized MFCC and Gaussianized MFCC. The Gaussianized MFCC's gave the best results. Pooling the three features gave the lowest miss rate. We report results on both TRECVID-2010 queries and TRECVID-2011 queries with these new audio and video copy detection algorithms.

2. VIDEO COPY DETECTION SYSTEM OVERVIEW

The overall system shown in Fig. 1 first computes the video fingerprints of the video query. We tried two different video fingerprints. One fingerprinting method is based on the video features used by NTT for TRECVID 2010 [7]: we compute 16 averaged pixel values per frame per color (RGB). These values then go through local temporal normalization in a window of 10 frames, and the top 7 values (based on maximum deviation from the mean) are then selected for quantization. For each test frame, we then find the closest query frame. The query frame number becomes the fingerprint for the test frame. In the second feature set, we used 16 normalized values per frame without quantization, and we computed the fingerprint for each test frame using these 16 unquantized features. These fingerprints result in even lower min NDCR.

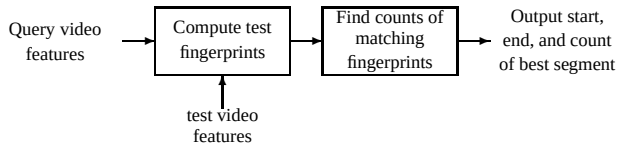

Fig. 1. *Video copy detection algorithm using fingerprints.*

We use these fingerprints to find test segments that may be copies of the queries. We match fingerprints by moving the query over the test and counting the total fingerprint matches for each alignment of the query with the test. One such alignment is shown in Fig. 2. In this alignment, the matching test segment is identified by the matching start frame (frame 3), the last matching frame (frame 7), and the number of fingerprint matches (3 matches). The total count of matches over all the aligned frames is a measure of confidence in the match. The best matching segment is the segment with the highest count. We tried both counts and counts/sec as a confidence measure. It turns out that counts work much better than counts/sec. This is similar to our experience with audio copy detection [2].

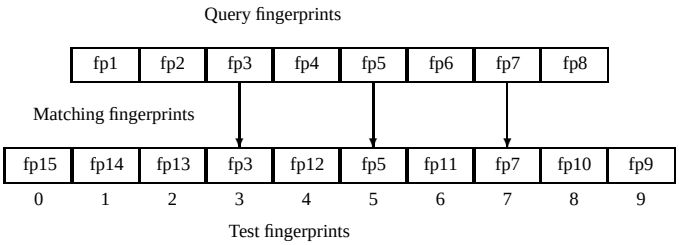

Fig. 2. *One example of matching query to a test.*

3. FEATURE PARAMETERS

We experimented with two different feature parameters. The first feature parameter is similar to the temporally normalized and quantized features used by NTT [7]. These features are computed as follows: Let $v_c(p,t)$ represent RGB value of a pixel in a video frame at time t, where p = pixel coordinate, $c \in \{R,G,B\}$. We divide the frame into 16 sub-squares and compute raw RGB value $x_c(i,t)$ in each square as

$$x_c(i,t) = \frac{1}{|I_i|}\sum_{p\in I_i} v_c(p,t),$$

where I_i $(i = 1,2,...,16)$ is a whole set of pixels in the i^{th} sub image. The temporally normalized features $y_c(i,t)$ are computed from $x_c(i,t)$ using a 10-frame window as follows:

$$y_c(i,t) = \frac{1}{\sigma_c(i,t)}(x_c(i,t) - \mu_c(i,t)), \quad where$$

$$\mu_c(i,t) = \frac{1}{M}\sum_{j=-[M/2]}^{M-[M/2]-1} x_c(i,t+j), \quad and$$

$$\sigma_c(i,t) = \left(\frac{1}{M}\sum_{j=-[M/2]}^{M-[M/2]-1}(x_c(i,t+j) - \mu_c(i,t))^2\right)^{1/2}$$

are average and standard deviation computed over a time window of M frames.

For each color, we choose 7 features for each frame that have the largest deviation from the temporal mean, that is, we choose seven values of i that have the maximum values for $z_c(i,t)$ where

$$z_c(i,t) = |(x_c(i,t) - \mu_c(i,t))|.$$

Each of these 7 chosen $x_c(i,t)$ values is then quantized between 0 and 5. Each of these values is stored as a (value, position) pair. In other words, there are 21 (value, position) pairs per video frame.

We tried two different algorithms to search for a given query in the test set using these features. One algorithm was similar to that used by NTT [7] where we move the query over the test and count all the matching (value, position) pairs. The test with the highest matching count is considered the best match, and the start and end of matches in the test correspond to the matching test segment. The total matching count of the (value,position) pairs is used as a confidence value. We call this search as *value-position matching*. The results for TRECVID 2009 for transforms 3, 4, and 5 (see Sec. 5) are shown in second row of Table 1. We use only transforms 3, 4, and 5 because they do not contain any flip, shift or picture-in-picture transforms. So we can use the extracted features directly for search.

The second search process we used with these features was the nearest-neighbor search (explained in the next section). For this search, we map each test frame to the closest query frame. To compute the closest query frame, we augment the seven (value, position) pairs with pairs (-1, position) for all the missing positions, and then compute the absolute sum S between a test frame and a query frame as

$$S = \sum_{i=0}^{15} |(y_c^{'}(i,t) - q_c^{'}(i,k))|.$$

where $y_c^{'}(i,t)$ is the quantized value of $y_c(i,t)$ in position i for the test frame t, and $q_c^{'}(i,k)$ is the quantized value in position i for the query frame k. We label the test frame as the query frame number k that gives the lowest sum S. The nearest-neighbor k is efficiently computed on a GPU [8]. We then search for the test frame segment that gives the highest nearest-neighbor count (computed as shown in Fig. 2). Table 1 compares the min NDCR for the nearest-neighbor search (row 3) versus the value-position matching search (row 2). The nearest-neighbor search outperforms the value-position matching search. The reason is that when the query matches a test segment, the nearest-neighbors will be ordered leading to a high matching count. When the query does not match a test segment, the nearest neighbors will be random, leading to very small counts. This is similar to our experience with audio copy detection [2].

The second feature set we used was the unquantized features $y_c(i,t)$ for all 16 positions. For search, we used the nearest-neighbor search with these unquantized values. The third row in Table 1 shows the min NDCR for this feature. These features gave the best results. So these features were used in the rest of the experiments.

Table 1. *Minimal NDCR for optimal no FA for different feature parameters and search algorithms.*

Transform	3	4	5
value-position matching	.052	.269	.067
nearest-neighbor search: discrete features	.007	.082	0.0
nearest-neighbor search: unquantized features	0.0	.037	0.0

The query goes through many transforms which affect the position of the feature parameters. For flip transform, we flipped the 16 feature vectors of each frame of the query. This leads to two feature sets per query: flipped and unflipped features. Each feature set is searched independently. Similarly, there were 5 picture-in-picture (PiP) positions (upper left, upper right, lower left, lower right, and center), and for each PiP position, there were three different sizes (0.5, 0.4, 03). This lead to 15 additional different feature sets for each of the flipped and non-flipped positions. So all together, we generate 32 different feature sets per query that are searched independently. We then retain the test segment that gives the highest matching count (using the nearest-neighbor search). Because of flip and picture-in-picture transforms, the search is 32 times slower. We did not do anything for the shift transforms. It turned out that queries with shift transforms were detected, but with significantly lower counts (or confidence).

4. SEARCHING VIDEO QUERY IN TEST USING NEAREST-NEIGHBOR MAPPING

The search for the test segment that matches the query is as follows. Each test frame is labeled as a frame number that corresponds to the query frame closest to the test. For example, in Fig. 3, the number inside each test frame corresponds to the query frame closest to that test frame. Frame 0 of the test matches frame 4 of the query, frame 1 of the test matches frame 1 of the query, ... We keep a count $c(i)$ for each frame i of test as a possible starting point for the query. In other words, count $c(i)$ corresponds to the total number of frames that match when the query is overlaid on top of the test starting with frame i (same as shown in Fig. 2). This is computed incrementally as follows. Assume that for each test frame i, $m(i)$ is the query frame closest to the test frame i. Then for each test frame i, we increment the count $c(i - m(i))$. We also update the starting test frame, and the last test frame corresponding to frame $(i - m(i))$. The count $c(j)$ then corresponds to the number of matching frames between the test and the query if the query was overlaid starting at frame j. The frame j with the highest count $c(j)$ and the corresponding start and end matching frames is the best matching segment. The final matching counts for the search example are shown in the bottom row of Fig. 3. In the given example, frame 3 of the test has the highest matching count. These counts are accumulated separately for each color (RGB) and then summed to get the final count.

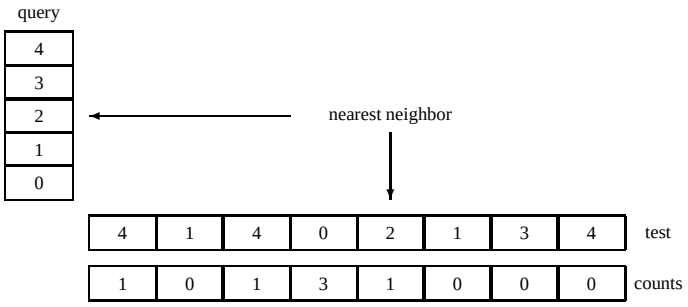

Fig. 3. *One example search using NN fingerprints.*

5. DATASET FOR COPY DETECTION

The data for copy detection for TRECVID 2009 comes from NIST sponsored TRECVID 2008 and 2009 CBCD evaluations [9] [10]. The queries are from the TRECVID 2009 evaluations. In TRECVID 2009, there were 201 original video queries transformed 7 different ways (Transforms 2, 3, 4, 5, 6, 8, 10 in Table 2). Each original query is supposed to occur one or zero times in the test video. The test set for TRECVID 2009 consists of a total of 385 hours of video.

For the 2010 and 2011 TRECVID copy detection evaluations, the test set consists of roughly 12000 videos from internet archives for a total of 400 hours of video. There are 201 original video queries (different from 2009) transformed 8 different ways (transforms 1 2 3 4 5 6 8 10).

Table 2. *Query video transforms used in TRECVID CBCD evaluations.*

Transform	Description
T1	Cam Cording
T2	Picture in picture (PiP) Type 1: original video in front of background video
T3	Insertions of pattern
T4	Strong re-encoding
T5	Change of gamma
T6, T7	Decrease in quality: blur, gamma, frame dropping, contrast, compression, ratio, white noise
T8, T9	Post production transforms: crop, shift, contrast, caption, flip, insertion of pattern, PiP type 2
T10	Combination of everything

The audio queries for TRECVID 2009, 2010, and 2011 are transformed 7 different ways as shown in Table 3. The last three transforms where the audio is mixed with irrelevant speech are particularly difficult to detect.

Table 3. *Query audio transformations used in TRECVID 2009/2010/2011.*

Transform	Description
T1	nothing
T2	mp3 compression
T3	mp3 compression and multiband companding
T4	bandwidth limit and single-band companding
T5	mix with speech
T6	mix with speech, then multiband compress
T7	bandpass filter, mix with speech, compress

6. VIDEO COPY DETECTION RESULTS

6.1. TRECVID 2009 results

The copy detection using the 16 floating-point temporally normalized values per frame was run on 1407 queries and 385 hours of test video from TRECVID 2009 CBCD evaluations. The min NDCR for the optimized no false-alarm case (Rtarget = 0.5/hr, CMiss = 1, CFA = 1000)[9] are shown in Table 4. Note that when we search 32 sets of features (Table 4) instead of one (row 4, Table 1), the min NDCR for transform 4 goes up from 0.037 to 0.052. Since we are searching 32 times, the mean processing time per query varies from 591 secs to 1052 secs. The reason for CPU time variation is that we terminate the computing whenever we get a match with a count greater than 50.

Table 4. *Min NDCR for 2009 video queries for no FA for features with 16 unquantized values/frame using nearest-neighbor search.*

Transform	2	3	4	5	6	8	10
min NDCR	.022	0	.052	0	0	.037	.097
mean F1	.818	.816	.808	.801	.812	.833	.798
avg proc time	1052	589	645	593	591	632	765

These results (Table 4) are probably the best published results on video copy detection for TRECVID 2009 [1]. In general, the min NDCR for video copy detection is significantly worse than for audio copy detection, but these results are better than the min NDCR we achieved on audio copy detection for the same task [2]. Our audio copy detection results from this paper are shown in Table 5. When we compare Table 4 with Table 5, we can see that min NDCR for video copy detection is significantly better than that for audio copy detection except for transform 10.

Table 5. *Min NDCR for optimal no FA for* **audio** *copy detection using NN-based fingerprints.*

Transform	1	2	3	4	5	6	7
min NDCR	.052	.06	.067	.06	.06	.075	.082
mean F1	.921	.936	.924	.89	.92	.90	.90
avg proc time	20.4	20.3	20.3	20.5	20.9	21.2	21

6.2. TRECVID 2010 results

The TRECVID 2010 CBCD evaluations test set consists of completely new videos collected from the web. This new set of videos is characterized by a high degree of diversity in creator, content, style, production qualities, orginal collection device/encoding, language, etc - as is common in much of *web video*. In 2009, there were 838 test video files for a total of 385 hours of video. In 2010, there are over 12000 files for a total of 400 hours of video. In other words, these videos are in general less than 4.1 minutes in duration. Many of these videos are slide shows with varying durations of each slide.

In compiling the copy detection results, we noticed that there were many duplicate test files for many queries. To compile the results correctly, we removed these duplicate files. The final results using the unquantized 16 values per frame features with nearest-neighbor search are shown in Table 6. Note that the mean processing time per query is around 2100 secs. The reason is that we cannot terminate the processing when we find a reference with count greater than 50, because there can be multiple test videos matching the query. In such a case, only one of these video matches will have a matching audio (the correct choice). Also, the evaluation rules require us to find all the duplicates.

Table 6. *Min NDCR for 2010 video queries for no FA for features with 16 unquantized values/frame using nearest-neighbor search.*

Transform	1	2	3	4	5	6	8	10
min NDCR	.62	.455	.045	.18	.03	.142	.187	.27
mean F1	.82	.80	.83	.82	.83	.81	.83	.81

As we can see from Table 6, the min NDCR is significantly worse for 2010 data than for 2009 data. The reason is simple. In 2009 videos, there are no slide shows, while 2010 data has several slide shows. The feature parameters we have used are based on temporal variability. When there is no temporal variability, then the features are either zero or one. This leads to many more false matches. For 2009 data, the largest count for false alarms is 36, while the largest count for false alarms for 2010 data is 51. This affects significantly the picture-in-picture (PiP) transforms. Inherently, PiP transforms show significantly fewer matches than for videos without PiP. With the false alarm threshold going up, all the transforms with PiP (transforms 2, 8 and 10) are adversely affected. Transforms 4 and 6 have lower resolution, and they are similarly adversely affected. Transform 1 is camcording, and the video frames have a lot of jitter, leading to fewer matches and therefore they are also adversely affected by the higher threshold for false alarms.

The optimal no FA results shown in Table 6 use separate threshold for each transform. In reality, we do not know *a priori* which transform is being used. So, in actual case, we can only use one threshold across all transforms. Table 7 gives results when we use one threshold across all transforms. For 2009 queries, this threshold was 36, while for 2010 queries, this threshold was 51. We notice that for 2009 queries, except for transform 10, the min NDCR is the same as it was for one optimal threshold per transform. For the 2010 queries, min NDCR has gone up for all transforms except for transform 5. This increase is primarily due to the slide shows, which result in higher threshold for the false alarms. We probably need to use static feature parameters for the slide shows in order to reduce these false alarms.

7. AUDIO COPY DETECTION

For audio copy detection also, we used the nearest-neighbor fingerprints to search for copies of the query in the test set. We followed audio copy detection algorithms similar to those used in [2]. In

Table 7. *Min NDCR for 2009 and 2010 video queries for no FA case when we use one threshold across all transforms.*

Transform	1	2	3	4	5	6	8	10
2009		.022	0	.052	0	0	.037	.12
2010	.71	.455	.045	.186	.03	.164	.238	.29

[2], we first did a fast search using energy difference fingerprints to narrow the search to a few choices. These choices were then rescored using the nearest-neighbor fingerprints derived from cepstral features and its first differences. For the TRECVID 2009 task, this scheme worked very well. However, for TRECVID 2010 task, the fast search using the energy difference features missed too many correct choices. So for TRECVID 2010, we switched to a complete search with the nearest-neighbor fingerprints. This gave much better min NDCR than searching with energy difference fingerprints, and then rescoring with the nearest-neighbor fingerprints.

Even in this new scenario, the results for TRECVID 2010 were significantly worse than what we had achieved in TRECVID 2009 [2]. We soon found out that the problem was the silent segments in the audio files. Many audio queries and audio test files have silent segments that are exactly the same. So we used a voice activity detector to locate silent segments. These silent segments were skipped when computing the matching counts. Otherwise, the entire algorithm was the same as in Sec. 4. Even with this change in the test and query files, there remained some short audio segments that were entirely zeros. These segments seemed to be some kind of filler segments for video frame timing synchronization with audio. We wrote a small perl script to identify all audio segments over 100 ms long that were entirely zeros. These segments were similarly skipped during nearest-neighbor search. These two changes made the audio only min NDCR for TRECVID 2010 much more reasonable.

7.1. Results on 2010 queries

While compiling the audio only copy detection results, we found many duplicate audio test files. Some duplicates were both audio+video duplicates, while some were audio only duplicates (videos did not match). For audio only processing, we removed both these duplicates in order to compare relative performance of different feature sets. The audio only duplicates were not removed for audio+video combined search.

There is another issue with the audio test files. Because these files are collected from internet archives, many audio files have a lot of music. For some test files, even though the entire file may not be a duplicate, parts of the audio are the same. This became evident when we listened to query and test segments with high matching counts. This happens for queries that are imposters (they are not supposed to have a matching audio or video segment). These short matching audio segments are the reason why the false alarm threshold has to be high. (This was not the case in TRECVID 2009). Because of this high threshold, the min NDCR for both the optimal and actual no FA case is significantly higher than that achieved in [2].

The min NDCR for the cepstral and delta cepstral parameters is shown in the second row of Table 8. The min NDCR is roughly three times higher than the average min NDCR of about 0.06 for TRECVID 2009 audio copy detection. To lower the min NDCR, we tried equalized cep (by subtracting the average value of cepstral features in each file to produce zero mean features) (row 3 in Table 8), and Gaussianized cepstra [11] (non-linear transformation of the cepstrum so that its probability distribution becomes Gaussian in a moving 3-sec window) (row 4 in Table 8). From Table 8, we can see that the Gaussianized cepstral features give the lowest min NDCR. When we combine the results from the three features (by choosing the matching segment with the highest counts), the results (row 5 in Table 8) are not as good as those for Gaussianized cep. For combining the results, when the test segments overlap for different features, we keep the segment with the highest score. The mean processing time for the combined case is shown in the last row. This processing time is much higher than in [2] because we are searching with the nearest-neighbor fingerprints (instead of energy-difference fingerprints). The mean F1 has gone down because we now sum all the overlapping matching segments, instead of choosing the largest matching segment. This leads to lower min NDCR, but it also lowers the mean F1.

Table 8. *Minimal NDCR for different features for 2010 audio queries for actual no FA case with one threshold for all transforms.*

Transform	1	2	3	4	5	6	7
cep	.156	.164	.201	.208	.231	.238	.305
equalized cep	.179	.186	.186	.186	.208	.216	.223
Gaussian. cep	.126	.149	.186	.156	.201	.216	.231
combined	.141	.156	.201	.194	.216	.223	.276
mean F1	.69	.67	.68	.70	.69	.69	.72
mean proc time	566	563	569	566	573	575	575

When we generate combined audio+video results, the critical value for audio copy detection is the number of missed queries. These missed queries are shown in Table 9. As we can see from this table, the combined features give the lowest number of missed queries. Compared to Gaussianized features, the combined features reduce the missed queries by 11%. These combined features were then combined with video copy detection for audio+video submission.

Table 9. *Total missed 2010 audio queries for different features.*

Transform	1	2	3	4	5	6	7
cep	14	16	22	25	19	20	29
equalized cep	24	25	25	24	26	26	29
Gaussianized cep	15	17	20	18	23	26	29
combined features	14	16	18	17	18	20	28

8. AUDIO+VIDEO COPY DETECTION

We combined the audio+video submission by first generating separately audio and video submissions with multiple choices, and then combining the two. In combining the audio and video results, each was given equal weight. During combination, if a query matched both the audio and video parts, then this match was given priority over single audio or video match. The reason is that there were a few test videos where only the audio or video parts match. So they are duplicates for audio only or video only. These will match the respective audio or video queries very well, but only the correct one will match both the audio and video queries. Also, for each query, only the best choice was kept. This is also important, since just audio or video matches can lead to high scores. Keeping these bad choices will significantly lower the min NDCR. Therefore, we eliminated these single match entries whenever both audio and video matched.

In the case where both the audio and video segments overlapped, the resulting segment was the *or* of the two segments. This was necessary because using only the audio segment or the intersection (of the audio and video segments) resulted in a few false alarms. The *or* reduced the F1 score compared to using just the audio segment. However, using the *or* of the segments was considered a safer alternative than generating false alarms with high scores. When we combine the audio+video, we get the optimal min NDCR for the no FA case as shown in Table 10. Note that, for optimal case, only seven transforms have min NDCR over 0.1. In the table, transform T1 is combination of video transform 1, audio transform 1, T2 is combination of video transform 1, audio transform 2, ..., T8 is combination of video transform 2, audio transform 1, and so on. Note that the mean processing time per query is between 2600 to 2700 secs, and the mean F1 is around 0.7.

Table 10. *Minimal NDCR for audio+video combined results for 2010 queries for optimal no FA case. Note T=transform.*

T	min NDCR	T	min NDCR	T	min NDCR	T	min NDCR
T1	.052	T15	.022	T29	.022	T50	.052
T2	.06	T16	.022	T30	.022	T51	.06
T3	.104	T17	.022	T31	.022	T52	.075
T4	.09	T18	.022	T32	.022	T53	.067
T5	.179	T19	.022	T33	.022	T54	.082
T6	.142	T20	.022	T34	.022	T55	.09
T7	.164	T21	.022	T35	.022	T56	.119
T8	.045	T22	.022	T36	.022	T64	.052
T9	.045	T23	.022	T37	.03	T65	.052
T10	.097	T24	.03	T38	.03	T66	.067
T11	.067	T25	.022	T39	.03	T67	.067
T12	.127	T26	.075	T40	.09	T68	.067
T13	.097	T27	.037	T41	.045	T69	.075
T14	.119	T28	.045	T42	.06	T70	.082

When we compute results for the actual case where we only have one threshold for all the transforms, two queries cause this threshold to be high (thresholds of 160 and 136). For the threshold of 160, the min NDCR is shown in Table 11. For the actual no FA case shown in Table 11, the min NDCR for many transforms has doubled compared to the optimal no FA case (Table 10).

If we ignore the two queries that cause the high thresholds of 160 (video only match) and 136 (audio only match of notes in wrong place due to superimposed speech), then the next highest threshold for false alarms is 58. For a threshold of 58, Table 12 gives the min NDCR for each transform. Transforms with false alarms above this threshold are marked with "xxx". There are 11 transforms which have false alarms above this threshold, resulting in bad min NDCR. For the rest of the transforms, the min NDCR is close to the optimal min NDCR for the no FA case (Table 10). For this reason, we submitted two no FA submissions with thresholds of 160 and 60.

8.1. Results on 2011 queries

Based on our experience with TRECVID 2010 queries, we submitted two audio+video submissions for no FA case. The submissions were the same except for the thresholds of 60 and 160. The results for individual transforms for actual no FA case are shown in table 13. The min NDCR entries in bold are for lowest min NDCR among all sites. For the threshold of 60, we get lowest min NDCR for 22 transforms. For the threshold of 160, we get the lowest min NDCR for another three transforms. Overall, we get lowest min NDCR for 25 transforms. Note that, for 12 different transforms, we miss only one query. The min NDCR of 107 for some transforms implies that there was one query with score over the threshold of 60 that should have been rejected. A min NDCR of 214 implies two queries with a score above the threshold. Similarly, a min NDCR of 320 implies three queries above the score of 60 that should have been rejected. There are only two such cases. The reason for the high false scores is that there is either part of audio or video that are duplicates of the test, but it is a false alarm because both audio and video are not duplicates. We cannot really eliminate all such entries, since eliminating all entries for which we do not find both audio and video significantly increases the min NDCR. This is because it is difficult to detect many audio and video entries that have gone through transforms difficult to detect. The mean processing time per audio+video query is around 2770 secs. The mean F1 is around 0.7.

Similarly, for the balanced case, we gave two submissions dif-

Table 11. *Minimal NDCR for audio+video combined results for 2010 queries for actual no FA case (threshold 160).*

T	min NDCR	T	min NDCR	T	min NDCR	T	min NDCR
T1	.126	T15	.022	T29	.022	T50	.052
T2	.149	T16	.022	T30	.022	T51	.06
T3	.194	T17	.022	T31	.022	T52	.075
T4	.186	T18	.022	T32	.022	T53	.067
T5	.223	T19	.029	T33	.029	T54	.082
T6	.223	T20	.022	T34	.022	T55	.09
T7	.283	T21	.029	T35	.029	T56	.119
T8	.089	T22	.044	T36	.052	T64	.09
T9	.104	T23	.052	T37	.052	T65	.09
T10	.126	T24	.067	T38	.074	T66	.097
T11	.126	T25	.067	T39	.067	T67	.097
T12	.134	T26	.082	T40	.097	T68	.119
T13	.149	T27	.089	T41	.089	T69	.119
T14	.186	T28	.119	T42	.119	T70	.156

Table 12. *Minimal NDCR for audio+video combined results for 2010 queries for actual no FA case with a threshold of 58.*

T	min NDCR	T	min NDCR	T	min NDCR	T	min NDCR
T1	.059	T15	.022	T29	.022	T50	xxx
T2	.067	T16	.022	T30	.022	T51	xxx
T3	.104	T17	.022	T31	.022	T52	xxx
T4	.09	T18	.022	T32	.022	T53	xxx
T5	xxx	T19	.022	T33	.022	T54	xxx
T6	.142	T20	.022	T34	.022	T55	xxx
T7	.164	T21	.022	T35	.022	T56	xxx
T8	.052	T22	.022	T36	.030	T64	.052
T9	.060	T23	.022	T37	.03	T65	.060
T10	.097	T24	.037	T38	.037	T66	.074
T11	.067	T25	.022	T39	.03	T67	.067
T12	xxx	T26	xxx	T40	xxx	T68	.074
T13	.111	T27	.045	T41	.052	T69	.082
T14	.119	T28	.045	T42	.06	T70	.082

Table 13. *Minimal NDCR for audio+video combined results for 2011 audio and video queries for actual no FA case with a threshold of 60. The min NDCR in boldface are the best results across all sites.*

T	min NDCR	T	min NDCR	T	min NDCR	T	min NDCR
T1	214	T15	**.007**	T29	**.007**	T50	214
T2	214	T16	**.015**	T30	**.007**	T51	**.045**
T3	214	T17	107	T31	**.007**	T52	**.045**
T4	214	T18	**.007**	T32	**.007**	T53	107
T5	214	T19	**.007**	T33	**.007**	T54	**.045**
T6	107	T20	**.007**	T34	.007	T55	**.045**
T7	107	T21	**.007**	T35	.007	T56	**.052**
T8	320	T22	107	T36	107	T64	214
T9	214	T23	**.045**	T37	.060	T65	214
T10	214	T24	**.045**	T38	.052	T66	214
T11	214	T25	**.037**	T39	**.052**	T67	214
T12	320	T26	**.052**	T40	**.060**	T68	107
T13	214	T27	.052	T41	.060	T69	107
T14	214	T28	.060	T42	.06	T70	107

fering in thresholds only: the two thresholds were 58 and 65. The min NDCR for each transform for actual balanced case at the threshold of 65 are shown in table 14. The min NDCR values in bold are the lowest among all submissions. Note that 13 transforms have only one missed query (min NDCR of 0.007).

Table 14. *Minimal NDCR for audio+video combined results for 2011 audio and video queries for actual balanced case with a threshold of 65. The min NDCR in boldface are the best results across all sites.*

T	min NDCR	T	min NDCR	T	min NDCR	T	min NDCR
T1	.415	T15	**.007**	T29	**.007**	T50	.273
T2	.423	T16	**.015**	T30	**.007**	T51	**.045**
T3	.430	T17	**.007**	T31	**.007**	T52	**.045**
T4	.430	T18	**.007**	T32	**.007**	T53	**.045**
T5	.452	T19	**.007**	T33	**.007**	T54	**.045**
T6	.338	T20	**.007**	T34	**.007**	T55	**.045**
T7	.361	T21	**.007**	T35	**.007**	T56	**.052**
T8	.447	T22	.152	T36	.166	T64	.288
T9	.340	T23	.045	T37	.060	T65	.288
T10	.355	T24	.045	T38	.052	T66	.288
T11	.348	T25	.037	T39	.052	T67	.296
T12	.477	T26	.052	T40	**.060**	T68	.204
T13	.363	T27	.052	T41	**.060**	T69	.196
T14	.378	T28	.060	T42	**.06**	T70	.204

9. CONCLUSIONS

We show that nearest-neighbor mapping of test frames to query frames works as well for video copy detection as it did for audio copy detection. We tried two different feature sets with this nearest neighbor mapping. One feature set was similar to that used by NTT [7] where we used 7 discrete features per frame per color. The other feature set used 16 unquantized features per frame per color. We show that the unquantized features gave significantly lower min NDCR than the discrete features. For the 2009 TRECVID CBCD data, the min NDCR varies between 0 and 0.097 depending on the transform. These results are better than the min NDCR we achieved on audio copy detection for the same data. For the 2010 TRECVID CBCD data, the min NDCR varies between 0.03 and 0.7. The results on 2010 data are worse because of many slide shows where the temporal variability is zero for many consecutive frames. For the slide shows, we need to come up with new static features and modified search in order to detect them.

We enhanced the audio copy detection by using three different feature parameters for copy detection: MFCC, equalized MFCC, and Gaussianized MFCC. Combining the results from the three feature parameters significantly reduces the number of missed queries. Combining the results from audio and video copy detection leads to very low min NDCR for many transforms. For 2011 audio+video queries, we obtained min NDCR of 0.007 for 12 transforms for the actual no FA case (we missed only one query). For 25 different transforms for the actual no FA case, we got the lowest min NDCR among all the submissions.

10. REFERENCES

[1] W. Kraaij, G. Awad, P. Over, "Slides: TRECVID 2009 Content-based Copy Detection Task" 2009, [online] www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2009.

[2] V. Gupta, G. Boulianne, P. Cardinal, " CRIM's content-based audio copy detection system for TRECVID 2009", to be published in *Multimedia Tools and Applications Journal*, Springer Netherlands, DOI: 10.1007/s11042-010-0608-x.

[3] M. Héritier, V. Gupta, L. Gagnon, G. Boulianne, S. foucher, P. Cardinal, "CRIM's content-based copy detection system for TRECVID" 2009, [online] www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2009.

[4] Z. Liu, B. Shahraray, T. Liu, "AT&T Research at TRECVID 2009 Content-based Copy Detection" 2009, [online] www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2009.

[5] W. Kraaij, G. Awad, "Slides: TRECVID 2010 Content-based Copy Detection Task" 2010, [online] www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html.

[6] Y. Li, L. Mou, M. Jiang, C. Su, X. Fang, M. Qian, Y. Tian, Y. Wang, T. Huang, W. Gao, "PKU-IDM @ TRECVid 2010: Copy Detection with Visual-Audio Feature Fusion and Sequential Pyramid Matching", [online] www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html.

[7] R. Mukai, T. Kurozumi, K. Hiramatsu, T. Kawanishi, H. Nagano, K. Kashino, "NTT Communications Science Laboratories at TRECVID 2010 Content-Based Copy Detection", Proc. TRECVID 2010, Gaitersburg, MD, USA.

[8] P. Cardinal, V. Gupta, G. Boulianne, "Content-based Advertisement Detection", Interspeech 2010, pp. 2214-2217.

[9] "Guidelines for the TRECVID 2009 Evaluation" 2009, [online] www-nlpir.nist.gov/projects/tv2009/

[10] W. Kraaij, G. Awad, and P. Over, "TRECVID-2008 Content-based Copy Detection", [online]. www-nlpir.nist.gov/projects/tvpubs/tv8.slides/CBCD.slides.pdf.

[11] J. Pelecanos and S. Sridharan, "Feature warping for robust speaker verification", Proc. Odyssey Spkr Lang. Recog. Workshop, Crete, Greece, 2001, pp. 213–218.