FXPAL Experiments for TRECVID 2004

John Adcock, Andreas Girgensohn, Matthew Cooper, Ting Liu[1], Lynn Wilcox, and Eleanor Rieffel
{adcock, andreasg, cooper, tliu, wilcox, rieffel}@fxpal.com

FX Palo Alto Laboratory Inc.
3400 Hillview Ave, Bldg 4
Palo Alto, CA 94304

1 Shot Boundary Detection

1.1 Summary of submitted runs

For shot boundary detection, our approach combines pairwise similarity analysis and supervised classification. Using primitive low-level image features, we build secondary features based on inter-frame dissimilarity. The secondary features are motivated by prior work on media segmentation in which a kernel function is correlated along the main diagonal of a similarity matrix to construct a frame-indexed novelty measure. In contrast to many previous approaches, the kernel functions combine *all* pairwise dissimilarity information in a neighborhood of L frames around the current frame. These secondary features are used as input to an efficient k-Nearest-Neighbor (kNN) classifier. The classifier labels each frame as a shot boundary or non-boundary, and the classifier outputs are minimally processed to determine the final segmentation.

Our performance was very good, and further validates the use of supervised classification in this application context. Additionally, the approach allows us to systematically assess performance variations due to alternate feature parameterizations, similarity measures, and secondary features. Our runs include two basic systems in which $L=5$ and $L=10$. To reduce dimensionality in the latter case, we use random projection. The random projection degrades performance in cut boundary detection, but enhances it in gradual boundary detection over the $L=5$ case. This suggests generally that dimension reduction for gradual boundary detection will not lead to performance degradation.

1.2 Motivation

Traditional shot boundary detection systems are comprised of three components: low-level frame-indexed feature extraction, inter-frame feature similarity comparison, and segmentation by extrema detection. Although TRECVID represents an enormous step towards associating performance variations with specific design choices among these system components, a systematic analysis of performance trade-offs with design parameters remains an elusive goal. A key obstacle is the numerous *ad hoc* thresholding schemes commonly used to detect cut and gradual shot boundaries from frame-indexed novelty or dissimilarity scores.

1. Ting Liu is with the Auton Lab in the School of Computer Science at Carnegie Melon University

Qi *et al.* [1] used supervised classification in lieu of thresholding for the final step in shot boundary detection. In their work, frame-indexed features are input to a kNN classifier to label frames as boundaries or non-boundaries. The behavior of the kNN classifier has been extensively studied, and it provides a principled alternative to thresholding. Additionally, the use of kNN for shot boundary detection provides a means to more generally study performance variations due to other system design parameters such as the choice of low-level features or the design of the frame-indexed novelty score.

For TRECVID 2004, we build on this approach and propose input features to the kNN classifier that provide improved performance over those employed in [1]. Our input features are motivated by our work on video segmentation using similarity analysis from TRECVID 2003. By combining similarity analysis and supervised classification, our system performs among the top teams.

1.3 System description

Our system first extracts histograms in the YUV colorspace. Global frame histograms and block histograms are extracted; the block histograms use a 4 x 4 uniform spatial image grid. Denote the global histogram and block histogram data for N frames by the matrices $\mathbf{V^{(G)}}$ and $\mathbf{V^{(B)}}$, respectively:

$$V^{(G)} = \left[V_1^{(G)} \ \cdots \ V_N^{(G)} \right] \quad , \quad V^{(B)} = \left[V_1^{(B)} \ \cdots \ V_N^{(B)} \right]$$

Given the low-level histogram features, we then compute secondary features based on inter-frame similarity analysis. For this, the extracted frame features are compared using a variant of the Chi-square similarity measure [2]:

$$D(V_i, V_j) = \sum_b \frac{(V_i(b) - V_j(b))^2}{(V_i(b) + V_j(b))^2} \qquad (1)$$

All possible pairwise comparisons between frames are readily visualized as a similarity or affinity matrix. Define the matrices $\mathbf{S^{(G)}}$ and $\mathbf{S^{(B)}}$ with the *(i,j)* element equal to the similarity between frames i and j using (1), computed from $\mathbf{V^{(G)}}$ and $\mathbf{V^{(B)}}$ respectively. Time, or the frame index, runs along both axes as well as the diagonal. The matrices have minimum dissimilarity (zero) along the leading diagonal where each frame is compared to itself. Because D is symmetric, $\mathbf{S^{(G)}}$ and $\mathbf{S^{(B)}}$ are also symmetric.

We build input features for the kNN classifier based on pairwise inter-frame dissimilarity. Many video segmentation systems are based on frame-indexed novelty scores which can be formulated as correlations of specific kernel functions along the main diagonal of a similarity matrix [3]. The kernel functions are L x L matched filters to detect structures in the similarity matrix corresponding to shot boundaries. Various criteria describing shot boundaries motivate different kernels. We construct input

features for each frame *n* using local neighborhoods of the similarity matrix centered at **S***(n,n)*. From experimentation, the best shot segmentation results corresponded to the checkerboard kernel of [4]. For this case, we build secondary frame-indexed feature vectors X_n:

$$X_n = \begin{matrix} [S(n-L, n-L) & S(n-L, n-L+1) & \ldots S(n-L, n+L) \\ S(n-L+1, n-L) & S(n-L+1, n-L+1) & \ldots S(n-L+1, n+L) \\ \ldots & \ldots & \ldots \\ S(n+L, n-L) & S(n+L, n-L+1) & \ldots S(n+L, n+L)\,]^T \end{matrix} \quad (2)$$

These features provide a complete local characterization of the inter-frame similarity near frame *n*. Specifically, X_n includes all pairwise similarity comparisons between frames within an *L* frame neighborhood of the current frame, *n*. These comparisons are computed separately using the global and block histogram features and concatenated as input to the kNN classifier.

In practice, there is no need to compute the complete similarity matrix. Instead, we only compute the portion of the matrix around the main diagonal with width *L*. For TRECVID, we considered the cases in which *L=5* and *L=10*. Thus computing the similarity data is *O(N)*. As well, for symmetric dissimilarity measures, such as (1), the dimensionality of X_n can be cut in half. For our experiments, the secondary feature dimensionality is 90 (*L=5*) or 380 (*L=10*).

We use an efficient exact implementation of the kNN classifier documented in [5] provided by the AutonLab at Carnegie Melon University. On this task it offers speedups over naive kNN on the order of a factor of 20. We apply the classifier hierarchically as in [1]. In the first step, cut transitions are detected. The frames that are labelled as non-cuts are re-classified as gradual transitions or non-transitions. Each classification is performed independently. The classification outputs are processed minimally. To detect cuts, we label the frame with the most nearest neighbors in the cut class in a 20 frame neighborhood as a cut, to avoid labelling spurious frames that are close in time as cuts. We look for frames labelled as graduals in contiguous units of at least 20 frames and no more than 200 frames to locate gradual transitions. To vary the precision and recall, we vary a threshold from 1 to *k*. When the number of nearest neighbors of a frame in the boundary class equals or exceeds the threshold, we label the frame as a boundary. There is no temporal filtering of the classifier outputs.

To reduce memory requirements and further accelerate the kNN classifier, we have also experimented with random projection [6] for dimension reduction. We project the 380-dimensional secondary features (for *L=10*) down to 100 dimensions, so that the classification complexity is the approximately the same for all submitted runs.

TABLE 1. The table lists parameters describing the various submitted runs

System	k	Threshold	L	Random Projection
FS05_04	11	4	5	none
FS05_05	11	5	5	none
FS05_06	11	6	5	none

TABLE 1. The table lists parameters describing the various submitted runs

System	k	Threshold	L	Random Projection
FS05_07	11	7	5	none
FS05_08	11	8	5	none
FS10_04	11	4	10	to 100 dims
FS10_05	11	5	10	to 100 dims
FS10_06	11	6	10	to 100 dims
FS10_07	11	7	10	to 100 dims
FS10_08	11	8	10	to 100 dims

1.4 Submitted runs

We submitted ten runs for the shot boundary detection task. For each run, we used the eight videos from the 2003 shot boundary detection task from either ABC or CNN for training data. We then randomly removed 90% of the non-transition frames from the training data to reduce its size. For all runs, $k = 11$. The differences between the runs are documented in the Table 1.

1.4.1 Time complexity

We also compiled timing information for each run. Video decoding and feature extraction was performed once using C code which has not been in any way optimized. All timing experiments were performed on a machine with an AMD Athlon 64 3500+ processor. This required 24882.35 seconds for the twelve videos in the test set. The computation of the secondary features required 159.18 seconds for the *L=5* case and 454.892 seconds for the *L=10* case. The first five systems result from one classification run; likewise the second five are also the result of a single run. The total processing time for the *L=5* run after feature extraction is 20183 seconds. For the *L=10* run which includes the random projection, the time was 21825 seconds. Combining the feature extraction and processing times, the *L=5* systems required 45232.269 seconds, and the *L=10* runs required approximately 47170 seconds. Thus overall, the systems require about twice real time, although our code could undoubtedly be further optimized

1.4.2 Detection performance

Overall, the systems all performed well, as documented in Table 2. Interestingly, the additional similarity features included in the *L=10* case seem to aid slightly in gradual boundary detection performance, even after random projection. At the same time, the random projection of these features seemed to degrade performance in cut boundary detection. This is expected since the adjacent frame comparisons (e.g. $\mathbf{S}(n,n-1)$, $\mathbf{S}(n-1,n-2)$, etc.) that are crucial for cut detection are mixed with other pairwise similarities in the random projection. However, this mixing doesn't degrade gradual bound-

ary detection performance. Other linear dimension reduction techniques can be expected to impact performance similarly.

TABLE 2. Performance of submitted runs for shot boundary detection.

System	Mean Recall	Mean Prec.	Mean F-Score	Cut Recall	Cut Prec.	Cut F-Score	Grad Recall	Grad Prec.	Grad F-Score
FS05_04	0.879	0.864	0.871	0.932	0.926	0.929	0.767	0.738	0.752
FS05_05	0.871	0.876	0.873	0.924	0.931	0.927	0.759	0.747	0.760
FS05_06	0.857	0.895	0.875	0.908	0.938	0.923	0.750	0.800	0.774
FS05_07	0.832	0.917	0.872	0.890	0.947	0.918	0.709	0.847	0.771
FS05_08	0.798	0.936	0.861	0.870	0.955	0.910	0.644	0.847	0.745
FS10_04	0.886	0.855	0.870	0.926	0.926	0.926	0.801	0.721	0.759
FS10_05	0.880	0.868	0.874	0.914	0.934	0.924	0.808	0.743	0.774
FS10_06	0.871	0.881	0.876	0.905	0.943	0.924	0.799	0.760	0.779
FS10_07	0.855	0.898	0.870	0.888	0.949	0.917	0.784	0.795	0.789
FS10_08	0.827	0.917	0.817	0.869	0.953	0.909	0.738	0.839	0.785
TRECVID MEAN	0.725	0.727	0.709	0.831	0.763	0.776	0.502	0.578	0.565

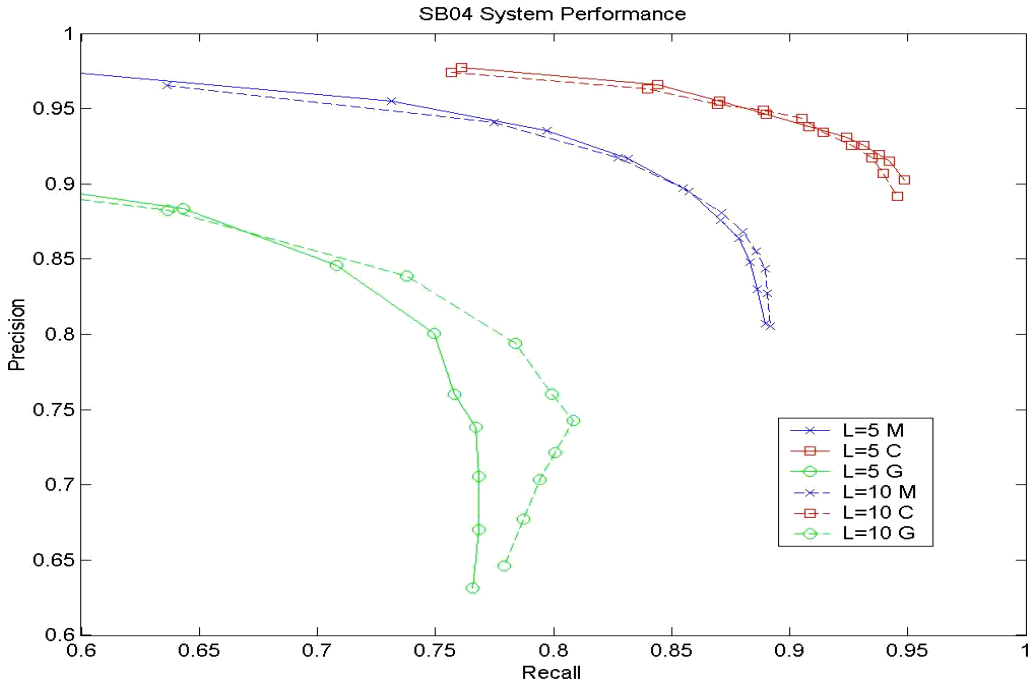


Figure 1: Performance of the shot boundary detection system. Solid curves indicate performance for L=5, and dashed curves show performance for L=10. The curves show performance for overall boundary detection (x's), cut boundary detection (squares), and gradual boundary detection (circles).

2 Interactive Search

2.1 Summary of submitted runs

We submitted 9 runs for the interactive search task; 3 different systems for each of 3 pairs of subjects. Each of the 6 total subjects answered his set of 12 topics once, and the system used 3 different methods to augment the search-identified list of relevant shots. All systems include a first step bracketing the manually-identified shots with the shots immediately preceding and following. System 1 (*WEIGHTED*) ranks each user-entered query by its recall against the set of manually-identified relevant shots and forms a single combined weighted query against the body of unlabeled shots. System 2 (*LSA1*) uses the text of the manually-identified shots to form a single text-based LSA query of the unlabeled shots. System 3 (*LSA2*) does the same but treats the text for each manually-identified shot as an independent query and these queries are then combined. The complete set of submitted runs in priority order:

1. I_A_1_AL_1_1: user group 1 with *WEIGHTED* query post-processing
2. I_A_1_AL_2_2: user group 1 with *LSA1* query post-processing
3. I_A_1_AL_3_3: user group 1 with *LSA2* query post-processing
4. I_A_1_AL_1_4: user group 2 with *WEIGHTED* query post-processing
5. I_A_1_AL_2_5: user group 2 with *LSA1* query post-processing
6. I_A_1_AL_3_6: user group 2 with *LSA2* query post-processing
7. I_A_1_AL_1_7: user group 3 with *WEIGHTED* query post-processing
8. I_A_1_AL_2_8: user group 3 with *LSA1* query post-processing
9. I_A_1_AL_3_9: user group 3 with *LSA2* query post-processing.

Our performance was strong overall finishing in the top 4 groups, with the 9 runs coming in 3-5 and 8-13 out of all submissions (see Figure 6). The MAP performance between user groups greatly outweighed the performance of the different systems within each group. User group 2 performed best overall followed by groups 3 and 1. Among system types, the *WEIGHTED* system performed best for groups 1 and 3 and 2nd best for group 2. *LSA1* performed 2nd best for groups 1 and 3 and best for group 2. *LSA2* was the worst performing system in all cases.

2.2 Overview

The interactive search interface was designed for efficient browsing and rich visualization of search results. Query results are displayed as a list of story thumbnails, sized in proportion to their query relevance. The story-level graphical summaries (thumbnails) use query relevance to build a query-related montage of the underlying shot thumbnails. Visual cues are widely used throughout the application to represent query-relevance and navigation history as well as shots included and excluded from the results list.

Keyboard shortcuts are used throughout to reduce the amount of mousing required to sift through results lists. We use a 2 level video segmentation with an automatically generated story-level segmen-

tation supplementing the reference shot segmentation. Text-based LSA analysis of the transcripts is used to build the story-level segmentation. User text searches are performed with literal or LSA-based text search and optionally combined with a still-image query-by-example capability. The LSI of story segments is also leveraged in the UI to allow the user 2 different ways to search for “similar” stories or shots.

2.3 Data Pre-processing

We perform a completely automatic pre-processing step to identify topic or story units to augment the reference shot boundaries. These story segments provide the basic unit of retrieval during queries.

To accomplish this segmentation we use the reference shot boundaries and the ASR transcripts. We build a latent semantic space (LSS) treating the stopped and stemmed [9] text tokens for each shot in the testing corpus as a separate document, adding words from adjacent shots to maintain a minimal number of tokens in each document. We then project the text for each shot into this shot-based LSS and compute a similarity matrix for each video using cosine similarity on the reduced-order vectors (one vector per shot). A checkerboard kernel is passed over the similarity matrix and points of highest novelty are chosen as story boundaries, as in [4]. A post-processing step assures the sanity of the boundary sizes and finds new boundaries in overly large segments.

In preparation for interactive operation full-text ASR transcripts are built for both shot-level and story-level segmentations using Lucene [7] and color correlograms are computed for each shot thumbnail image.

2.4 Search Engine

Queries are specified by a combination of text and images. The searcher can opt to perform a text-only or image-only search by leaving the image or text query area empty.

The searcher can choose between a literal keyword text search and a latent semantic analysis (LSA) based text search. We use only the provided ASR transcript to provide text for story and shot segments. The literal text search is based on a Lucene [7] back end and ranks each story based on the tf-idf values of the specified keywords. In this mode the story relevance, used for results sorting and thumbnail scaling and color coding as described in following sections, is determined by the Lucene retrieval score. When the LSA based search is used [8], the query terms are projected into a latent semantic space (LSS) of dimension 100 and scored in the reduced dimension space against the text for each story and each shot using a cosine similarity function. In this mode, the cosine similarity value determines the query relevance score. In our application the LSS was built treating the text from each story segment (determined as described in Section 2.3) as a single document. The TRECVID 2004 test data yielded an original document space of 3061 stories and 12098 terms.

When determining text-query relevance for shots, each shot gets the average of the retrieval score based on the actual shot text and the retrieval score for its parent story. The shots inherit relevance from their stories.

An image similarity matching capability is provided based on color correlograms [10]. Any shot thumbnail in the interface can be dragged into the query bar and used as part of the image query. For

Figure 2: The interactive search interface. (A) Story keyframe summaries in the search results (B) Search text and image entry (C) TRECVID topic display (D) Media player and keyframe zoom (E) Story timeline (F) Shot keyframes (G) Relevant shot list

each thumbnail the color correlogram is computed and compared to the correlogram for every shot thumbnail in the corpus. To generate an image-similarity relevance score at the story level, the maximum score from the component shots is propagated to the story.

The document scores from the text search are combined with document scores from the image similarity to form a final overall score by which the query results are sorted. A query returns a ranked list of stories.

2.5 Interface Elements.

The interactive search system is pictured in Figure 2. The TRECVID test question and supporting images are shown in section C. Text and image search elements are entered by the searcher in section B. Search results are presented as a list of story visualizations in section A. A selected story is shown in the context of the video from which it comes in section E and expanded into shot thumbnails in section F. When a story or shot icon is moused-over an enlarged image is shown in section D. When a video clip is played it is also shown in section D. User selected shot thumbnails are displayed in section G. A tooltip can be seen in section F displaying keywords for the currently selected shot (drawn from the words with the highest tf-idf values) with query-related keywords shown in bold. All stories and shots display a similar tooltip when dwelled over.

Figure 3: Story keyframe montage example. The keyframe montage on the left is constructed from the shot keyfames comprising the story on the right weighted by their relevance to the query “Sam Donaldson”.

2.5.1 Thumbnails

Shots are visualized with thumbnails made from the primary keyframe drawn from the reference shot segmentation. Story thumbnails are built in a query-dependent way. The 4 shot thumbnails that score highest against the current query are combined in a grid. The size allotted to each portion in this 4-image montage is determined by the shot’s score relative to the query. Figure 3 shows an example of this where the query was “Sam Donaldson” and the shots most relevant to the query are allocated more room in the story thumbnail, in this case the 2 shots of the 9 total shots in the story that depict Sam Donaldson.

2.5.2 Overlays

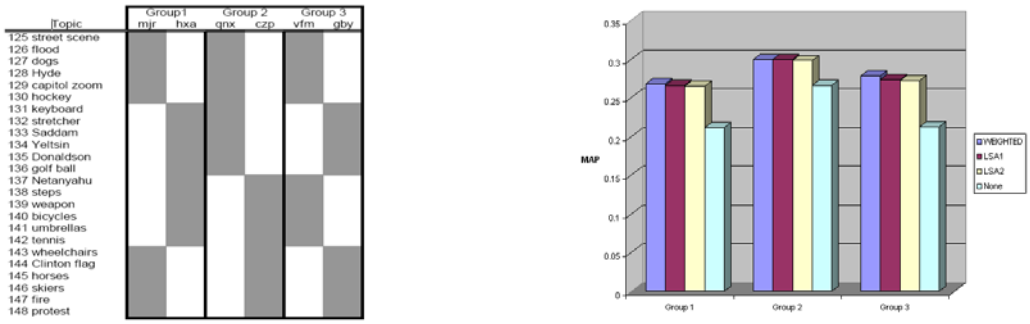
Semi-transparent overlays are used to provide 3 cues. A gray overlay on a story icon indicates that it has been previously visited (see Figure 2 A and E). A red overlay on a shot icon indicates that it has been explicitly excluded from the relevant shot set (see Figure 2 F). A green overlay on a shot icon indicates that it has been included in the results set (see Figure 2 F).

Horizontal colored bars are used along the top of stories and shots to indicate the degree of query-relevance, varying from black to bright green. The same color scheme is used in the timeline depicted in Figure 2 D.

2.6 Post Query Processing

When the searcher decides to end his task by pressing the “end question” button or when the 15 minute allotted time expires, the search system uses 3 different methods to perform an automated search process to fill out the remaining slots in the TRECVID allowed 1000 shot result list.

In all cases the shot list is first augmented by adding the shots immediately preceding and immediately following each shots in the relevant list. This bracketing action ignores any user judging - that is, even shots that were explicitly marked as not-relevant will be included in this bracketing step. In the remainder of the methods, shots that have been explicitly marked as non-relevant are not considered for inclusion in the results list.



(a) (b)

Figure 4: (a) The topics assigned to the 6 searchers and the searcher groupings used to form complete runs (b) Overall MAP performance by user group and post-processing system type employed. The “None” column is the MAP performance of the user selected shots without any automatic augmentation

2.6.1 Weighted Query History (*WEIGHTED*)

In this scheme, the searcher’s query history is re-used along with the list of relevant shots. Each query is re-issued (both text and image components) and its precision measured against the list of relevant shots. For each user-issued query, find the N shots with the highest score (N = 500 in our application). Give the highest ranked shot 500 points, the next 499,... Add the points of all the shots that are in the result set. The normalized sum of the points for each query is its weight for the weighted average. The relevancy values from all previously issued queries are then combined with these weights for all unjudged shots, and the top scoring shots are used to fill the remaining slots in the 1000 shot run result list.

2.6.2 Single LSA Query (*LSA1*)

In this mode the text from the shots that have been judged by the searcher to be relevant is combined to form a single LSA-based text query. This query is applied to the unjudged shots and the highest scoring ones retained for the result list.

2.6.3 Multiple LSA Query (*LSA2*)

In this mode, each shot in the set of shots judged relevant by the searcher is used as an individual LSA-based text query. The relevancy values for each shot are then weighted and combined based on the precision against the relevant list exactly as in the *WEIGHTED* method.

2.7 Tests and Results

We employed 6 searchers to each answer 12 topics in a standard latin square arrangement as depicted in Figure 4a. We then grouped searchers who had answered complementary sets of topics to create 3 groups of 2 searchers. For each of these searcher groups we submitted 3 different systems corresponding to the 3 post-processing techniques for filling out the shot lists (*WEIGHTED*, *LSA1*, *LSA2*). A complete list of submitted runs by ID can be found in Section 2.1 Figure 4(b) summarizes the MAP performance of the 3 searcher groups by system type, and Figure 5 by topic and searcher group.

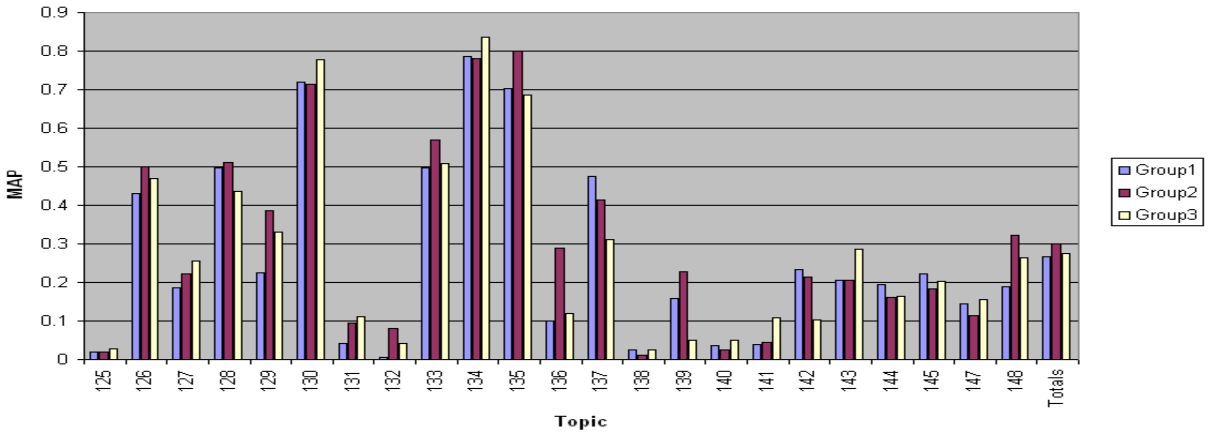


Figure 5: MAP Performance by topic and searcher group.

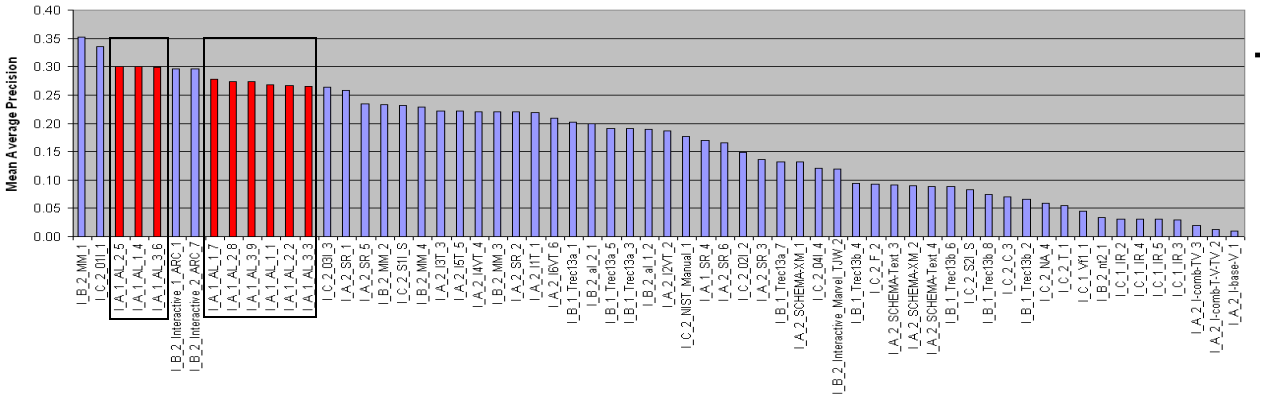


Figure 6: The FXPAL interactive search MAP scores with the entire set of submissions.

The MAP performance between user groups greatly outweighs the performance of the different systems within each group. User group 2 performed best overall followed by groups 3 and 1. Among system types, the *WEIGHTED* system performed best for groups 1 and 3 and 2nd best for group 2. *LSA1* performed 2nd best for groups 1 and 3 and best for group 2. *LSA2* was the worst performing system in all cases when viewed in the aggregate.

Figure 6 shows the MAP performance of the 9 FXPAL runs compared to all TRECVID submissions.

3 Acknowledgments

We thank Andrew Moore and the Auton Lab at Carnegie Melon University for making their k-nearest-neighbor software available for use in our shot boundary detection system.

4 References

1 Y. Qi, A. Hauptman, and T. Liu. Supervised Classification for Video Shot Segmentation. *Proc. IEEE Intl. Conf. on Multimedia & Expo, 2003*.

2 N.V. Patel and I.K. Sethi, “Video shot detection and characterization for video databases”, *Pattern Recognition*, Special Issue on Multimedia, vol. 30, pp. 583-592, April 1997.

3 M. Cooper. Video Segmentation Combining Similarity Analysis and Classification. *Proc. ACM Multimedia*, 2004.

4 M. Cooper and J. Foote. Scene Boundary Detection Via Video Self-Similarity Analysis. *Proc. IEEE Intl. Conf. on Image Processing*, 2001.

5 T. Liu, A. Moore, and A. Gray. Efficient Exact k-NN and Nonparametric Classification in High Dimensions. *Proc. Neural Information Processing Systems*, 2003.

6 D. Fradkin and D. Madigan. Experiments with Random Projection for Machine Learning. *Proc. ACM SIGKDD*, 2003.

7 Jakarta Lucene. http://jakarta.apache.org/lucene/docs/index.html.

8 Michael W. Berry, Susan T. Dumais, Gavin W. O'Brien, Using linear algebra for intelligent information retrieval, *SIAM Review*, v.37 n.4, p.573-595, Dec. 1995

9 M. F. Porter. An algorithm for suffix stripping. Program, 14(3):130--137, 1980.

10 J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih. Image indexing using color correlograms. In Proc. IEEE Comp. Soc. Conf. Comp. Vis. and Patt. Rec., pages 762--768, 1997