TRECVid 2010 Experiments at Dublin City University

Colum Foley[1], Jinlin Guo[1], David Scott[1], Paul Ferguson[2], Peter Wilkins[1],
Kealan Mc Cusker[2], Emma Sesmero Diaz[2], Cathal Gurrin[1], Alan F. Smeaton[1,2]
Xavier Giro-i-Nieto[3], Ferran Marques[3], Kevin McGuinness[2], Noel E. O'Connor[2]
[1]School of Computing and [2]CLARITY: Centre for Sensor Web Technologies
Dublin City University, Glasnevin, Dublin 9, Ireland
[3]Technical University of Catalonia (UPC) C. Jordi Girona, 31. 08034 Barcelona
colum.foley@computing.dcu.ie

Abstract

This year the DCU-CLARITY-iAD team participated in the both the instance search and interactive known-item search tasks of TRECVid 2010. For our instance search submission, we used classifiers to search for candidate objects in each keyframe. This was achieved by a coarse-to-fine search based on a hierarchical representation of the regions in the keyframes. Our results proved inconclusive, but we believe the method warrants further investigation. The 2010 interactive search task at TRECVid represents a number of firsts for the community and for our team it represents the first time that many of our team has participated in TRECVid. Our approach this year was to develop a simple and intuitive system which we felt could be used by video information seeking specialists and complete novices alike. To this end we have developed our 2010 TRECVid KIS system on an Apple iPad, the iPad is a new tablet computer developed by Apple, it represents a lean-back, relaxed and easy to use computer, likewise our search engine was designed to be easy to use by all users. Our underlying search engine allows for the three commonly used video search methods: text search, concept search and image search. For our experiments we compared the performance of our system when used by standard users from our research group versus novices with no technical expertise. Our results show that the two groups gave identical performance in terms of mean elapsed time.

1 Introduction

This year the team at Dublin City University from CLARITY: Centre for Sensor Web Technologies, and iAD: Information Access Disruptions participated in the instance search and interactive known-item search task (DCU-CLARITY-iAD).

The CDVP has participated in TRECVid for a number of years [12, 7, 3]. However, for most of the core team working on the interactive known-item search task, this year represented their first participation in TRECVid and we developed most of the components of the system from the ground-up. As such our approach this year was to develop simple, robust technologies for each of the major components of the search engine with a view to developing these further in the coming years.

Through our experiments we were interested in developing a system that could be used by users with no experience in using a sophisticated content-based video retrieval system. These users typify the majority of users that search through video collections such as YouTube daily, and represent the biggest audience for any new search technologies. In our research group we are also interested in exploring multi-modal access to video archives, and previously we have built video search engines that could be accessed through mobile phones and tabletop devices.

To this end we have developed a simple-to-use search engine that runs on the Apple iPad. The iPad is a new tablet computer developed by Apple and aimed at users of all ages and technical prowess. Our system attempts to support the main components of a state-of-the-art video search engine in a simple iPad App which users can interact with using their fingers.

Through our TRECVid experiments we were interested in exploring how novice users compare to expert users when using our TRECVid system: how they interact with the system, the types of search strategies they employ and how they perform in the search task.

For our interactive search experiments we submitted two official runs: one run that used users from our own team research group in CLARITY and iAD Dublin, (I_A_YES_DCU-CLARITY-iAD_run1_1) and another run that used business management students from the BI School of Management in Oslo, Norway (I_A_YES_DCU-CLARITY-iAD_novice1_1), these students represent our novice users, none had used an iPad before and have no experience using a content-based video search engine such as those used in TRECVid.

The results from our official experiments place our runs at 6th and 7th overall, which puts us mid-table. Our results show that the performance of novices versus experts is identical in terms of mean elapsed time.

Our instance search submission ('UpcSvmBor") was based upon decomposing each keyframe into a set of hierarchical regions represented as a binary partition tree, then matching these regions against the query topics using visual codebooks. There were, unfortunately, errors in the implementation that affected our results.

In this paper we describe in detail our work in TRECVid 2010, in Section 2 we describe our experiments in the instance search task and following that in Section 3 we describe our experiment in interactive known-item search.

2 Instance Search

The submitted run was the result of applying an object detection algorithm on the keyframes suggested in the test set and ranking the shots according to the highest obtained among their keyframes. This report describes the techniques involved in the process. Please note, however, the authors believe that the software implementation contained bugs at submission time, and as such the obtained results may not be an accurate reflection of the image analysis techniques, but as a consequence of errors in the source code.

The image representation considered in this experiment is based on a hierarchy of regions represented by a Binary Partition Tree (BPT). Every keyframe involved in the retrieval process was preprocessed with a segmentation algorithm that, in the first stage, creates an initial partition of the images and, in the second stage, iteratively merges the two most similar neighbouring regions. As a result, a hierarchical structure is obtained, where the root of the tree represents the complete keyframe and the leaves of the tree represent the regions in the initial partition. In our experiments we manually set the number of segments in the initial partition to $n = 200$, producing a final BPT containing 2 n - 1 = 399 nodes.

Once a BPT for each keyframe had been generated, the next step was to extract a set of visual descriptors for every region. The features we used for these visual descriptors were the region area, the aspect ratio of their oriented bounding box, the mean and variance of their dominant colours, and the texture edge histogram. The latter of the two features followed the implementation guidelines defined in the MPEG-7 standard.

The region-based descriptors were not directly used to generate the feature vectors used for retrieval, but to create a visual codebook specific for every query topic. The visual words represented in the codebooks were designed to match the different subparts over which every object class can be decomposed in a BPT. This intermediate level was motivated by the fact that in many cases the perceptual criteria used to merge the nodes in the BPT does not match the semantic criteria that would drive to the representation of every object by a single BPT node.

The visual codebooks were built after mapping the query masks provided by NIST in a set of BPT nodes. This mapping provides an estimation of the amount of sub-parts over which every object will be decomposed. The final amount was chosen as the maximum amount of sub-parts found among the five query instances. This figure was taken as the size of the codebook, and its words were defined by running the k-means algorithm among all annotated sub-parts and using specific distances for every region-based descriptor considered. These distances were averaged after a normalisation stage that tries to map every descriptor distance to a similar perceptual response.

Once a visual codebook had been built for every query topic, two feature vectors were used to represent each node in the BPT. The first of these feature vectors consisted of the similarity scores $(1 - \text{distance})$ between the BPT node and every visual word in the codebook. The second feature vector represented the maximum similarity score of all nodes in the subtree. In this way, every BPT node was represented by a feature vector representing the associated region, and a second feature vector describing the regions in the subtree below.

Our search algorithm was based on training a collection of four region-based SVM classifiers for every query topic. Two of the classifiers referred to the complete object and the two others to the parts that composed the objects. In every case, the first of the classifiers evaluated whether the subtree defined by a region contained an instance of the modelled object or part, while the second classifier directly assesses if the considered region represents itself an instance of the modelled object or part.

The trained classifiers were evaluated on every BPT in the test set. The object detection started by assessing the object inclusion classifier on the BPT root, so that all negative detections discarded any further analysis on the test keyframe. Those BPTs whose root was classified as container of an object were top-down explored by applying the second type of classifiers: the part inclusion classifier. Whenever this classifier discarded a subtree, no further exploration was performed through it. Every BPT node considered as a potential candidate to include an object part was also analysed with the part detection classifier. When the BPT was completely processed, all detected parts were combined to define new regions not contained in the input BPT, this way, avoiding object-split problems generated during BPT creation. The part combinations were evaluated with the last type of classifier: the object detection classifier. The combination with the best similarity score was assigned to the keyframe for retrieval.

The results that we obtained, and further development that was completed after submission, suggest that the implemented software contained bugs that preclude drawing conclusions about the virtues and drawbacks of the described technique.

The Image Processing Group of the Technical University of Catalonia has developed most tools used in the experiments. These engines were implemented in Java, C and C++ and, whenever possible, adopted MPEG-7/XML file format for data exchange. The SVM classifier implementation used libsvm by Chih-Chung Chang and Chih-Jen Lin from the National Taiwan University [1].

3 Interactive Known-Item Search

This year's interactive search task at TREVid represents a significant departure from the interactive search of previous years: the task this year is known-item search, where a user is attempting to locate a single item from the collection; the unit of retrieval is the whole video, rather than a video shot; the video data this year is general internet video, rather than broadcast content; the time per topic is down to 5 minutes. With these changes, the interactive search task more closely models a typical internet user's interactions with an online video search engine like YouTube. With this in mind, our approach in DCU this year was to attempt to build a content-based video retrieval system which could be used by everyday internet users such as the millions using YouTube daily.

Figure 1 provides an overview of our video search system. The user interface for the system was built as an application running on the iPad. The back-end search engine runs as a web service on a separate laptop and the front and back ends communicate through the HTTP Post protocol.

The search engine supports the three commonly used search methods for video retrieval:

- Text search
- Concept search
- Similarity search

We will now provide a detailed description of each of the main components of our search system.

3.1 Search Middleware

We developed a .NET web service as the backbone of our middleware layer, this communicates with our iPad user interface through the usage of HTTP POST calls. Each call invokes one of the two types of search functions within the web service:

- **Primary**. Achieved by using a combination of text and concept search methods. Users can input text queries directly, search based solely on concepts or use a combination approach. The return type of this type of searching is an XML document pertaining to the top 100 ranked videos.
- **Secondary**. An image similarity method used to invoke searching. After completing a primary search users can invoke the image similarity method to query-by-example, providing a keyframe as input. Results are also in the form of an XML document though this relates to the top 50 similar images.

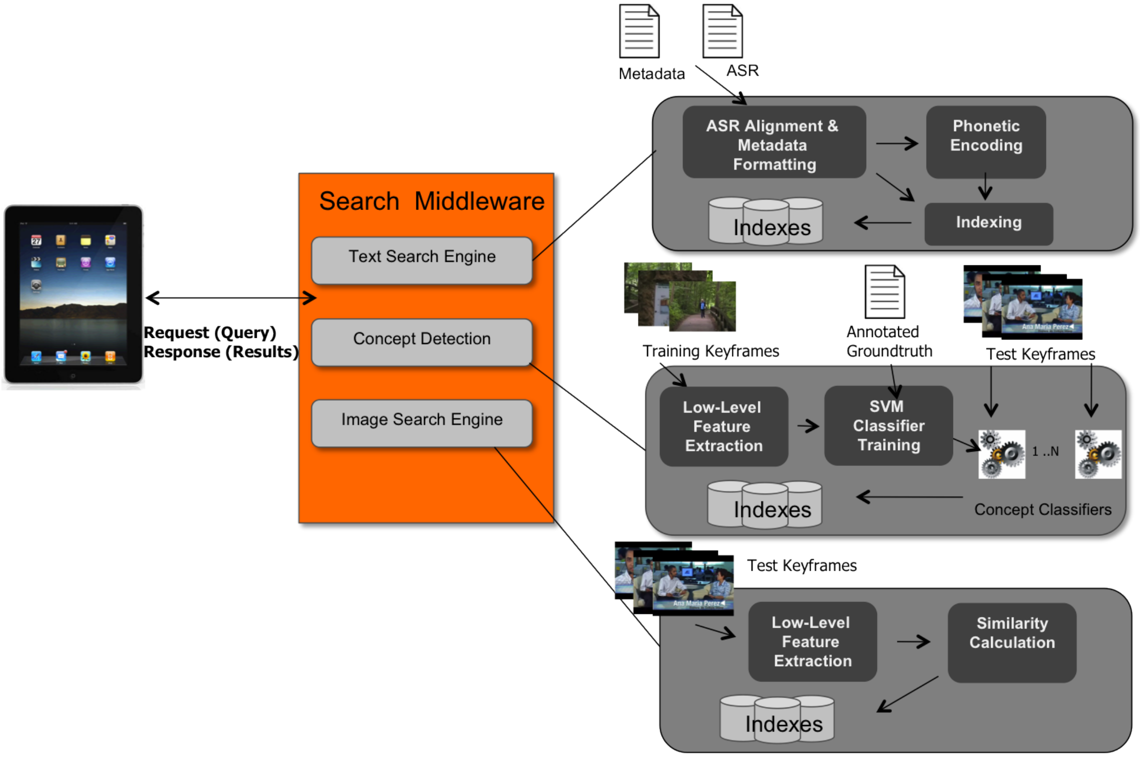


Figure 1: System Overview

The message passing layer is illustrated in Figure 2, the top portion of the search middleware shows the methods used during the primary and secondary searches, the web service is also responsible for the additional features:

- **Return Shot Timing**. This method is used to inform the interface of start and end time of a particular shot when its keyframe image is selected on the iPad device.
- **Query Oracle:** This method is used to check if a selected video is the relevant known-item for a search topic by querying the TRECVid oracle database.

The final responsibility of the web service is to log each interaction with the system based on which methods are invoked by the user, these logs form an invaluable source of information which will be used to analyse how users use the search engine and to aid in the implementation of future systems.

3.2 Text Search Engine

The text search engine we have chosen to use is the Terrier project developed by the University of Glasgow [11], we were able to invoke the interactive search component from our web service and receive ranked lists based on initial search criteria on three created indexes:

- **Source Data**. Contains metadata information pertaining to the entire video attained from the crawling of the internet archive and provided to us by NIST. The information stored in this index are generally considered author comments and give a good overview of each video in the collection.
- **Automatic Speech Recognition**. This index was created by utilising the spoken word in the video and was provided by LIMSI and Vecsys Research [5]. This information was indexed on the shot level by aligning the spoken word to its relevant shot bounds.
- **Phonetic Encoding**. Phonetic Encoding is concerned with representing the pronunciation of a word with a code made up of letters and numbers [6, 2]. Similar words will have the same code and can therefore be matched by the search engine. Having performed an analysis of several techniques we found that the NYSIIS system [10] was the best choice for our needs.

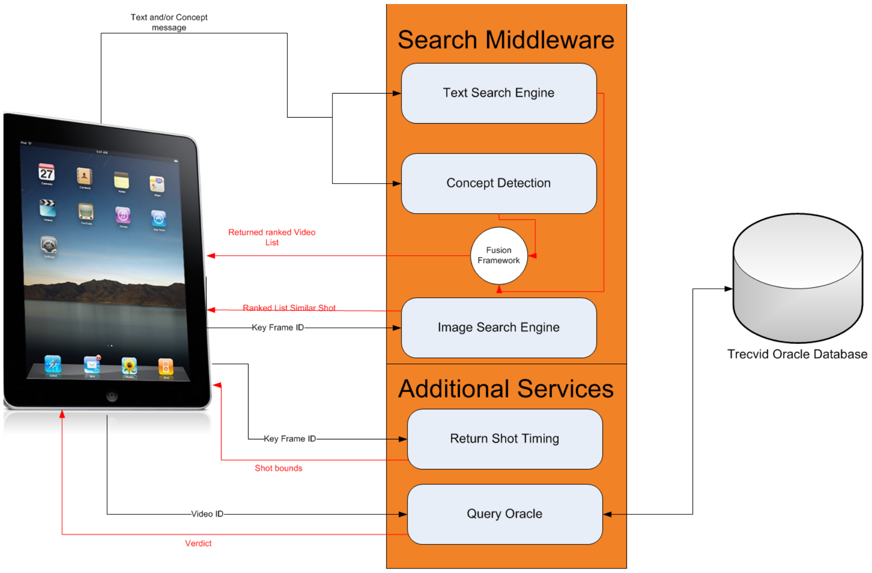


Figure 2: Overview of the message passing layer of .net web service

The output of this process is a set of similar sounding words to the words in the metadata and ASR which is then indexed by the search engine.

Separate indexes were employed to allow us to train different weighting schemes and make a decision as to which would give the greatest recall and precision, we achieved this by utilising the 122 training topics. We used $TF \times IDF$ ranking scheme on the indexed documents, this was chosen due to higher performance on the training set when compared to that of BM25 and PL2.

3.3 Semantic Concept Detection

In TRECVid 2010, we adopt two methods for semantic concept detection in videos utilising the SVM classification framework. One is based on a combination of three MPEG-7 descriptors and SURF. The other is based on the very popular Bag of Visual Word (BoW) model.

3.3.1 Method Based on MPEG-7 Descriptors and SURF

The approach flowchart is indicated in Figure 3. We build concept detectors by combing MPEG-7 colour and texture descriptors and SURF.
In most of existing literature on concept detection SVM has proven to be a solid choice, and indeed, it has become the default choice in most concept detection schemes. In this work the classification framework is implemented using LIBSVM (Version 2.91) [1]. The RBF kernel is chosen for its good classification results compared to polynomial and linear kernels.

Visual Feature Extraction: In this year, we only consider visual features, three MPEG-7 [9] colour and texture descriptors and SURF are extracted.

- **Colour Layout:** a compact descriptor which captures the spatial layout of the representative colours on a grid superimposed on an image.
- **Scalable Colour:** derived from a color histogram defined in the HSV color space which uses a Haar transform coefficient encoding, allowing scalable representation.
- **Edge Histogram:** represents the spatial distribution of edges in an image, edges being categorised into either vertical, horizontal, 45° diagonal, 135° diagonal and non-directional.
- **SURF:** Since an object can appear in each part of a keyframe, the feature about SURF we adopt is a histogram by grouping the interest points into regions. Given a keyframe and a set of keypoints, a 3×3 grid is defined. A 9-bin histogram which is a count of the keypoints that occur in each square is created (See Figure 4).

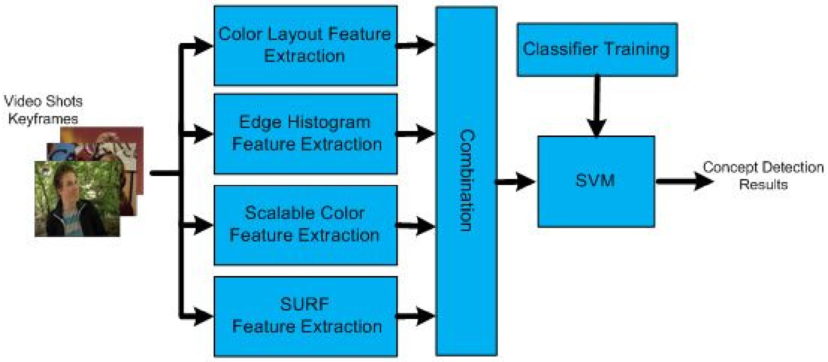

Video Shots Keyframes
Color Layout Feature Extraction
Edge Histogram Feature Extraction
Scalable Color Feature Extraction
SURF Feature Extraction
Combination
Classifier Training
SVM
Concept Detection Results

Figure 3: Flowchart for Concept detection based on MPEG-7 descriptors and SURF

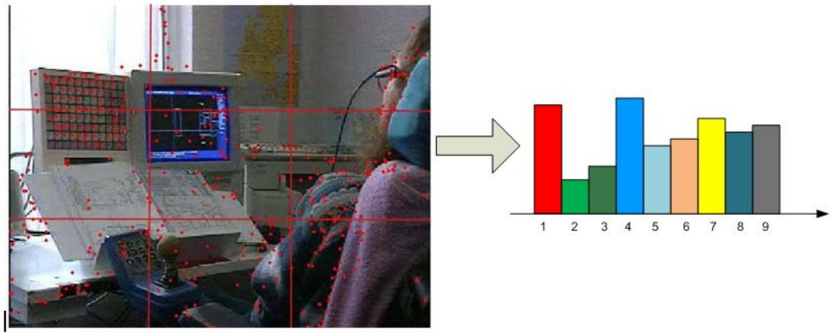


Figure 4: SURF feature Extraction

3.3.2 Method Based on BoW Model

For the past few years, systems based on the BoW model produced the best results on several large scale content based image and video retrieval benchmarks. In this year, we also try to employ the baseline of BoW model for concept detection. The flowchart for this method is shown in Figure 5.

SIFT Feature Extraction: the SIFT feature proposed by Lowe [8] has proved to be very successful in applications such as object recognition and image retrieval. To compute SIFT features we use the version described by Lowe [8].
Construction of Visual Vocabulary: In the construction of the visual vocabulary we employ the Hierarchal K-means algorithm to construct the visual vocabulary based on its advantages of simple and fast implementation. Five million SIFT descriptors were extracted from keyframes from the training data and these were clustered hierarchically using K-means to generate a vocabulary tree with 1296 leaf nodes (i.e 1296 visual words).
Visual Vocabulary Transformation: Soft assignment is utilised in the step of visual vocabulary transformation. For each keypoint in an image, instead of mapping it only to its nearest visual word, in soft assignment we select the top-*100* nearest visual words. Suppose we have a visual vocabulary of K visual words, we use a K-dimensional vector $[\omega_1, \omega_2, ...\omega_K]$ with each component representing the weight ω_t of a visual word t in an image such that

$$\omega_t = \sum_{i=1}^{100} \sum_{j=1}^{M_i} \frac{1}{2^{i-1}} sim\,(j,t)$$

where M_i represents the number of keypoints whose i_{th} nearest neighbour is the visual word t. The measure $sim(j,t)$ represents the Euclidean similarity between keypoint j and the visual word t. In this equation, the contribution of a keypoint is its similarity to word k weighted by $\frac{1}{2^{i-1}}$, representing that the visual word is its i_{th} nearest neighbour.
In the final system we developed 33 concepts based on types of concepts used in the training topics. They are: Animal, beach, beard, Black and White video, boat/ship, building,bus, car charts, cityscape, computers, computer screen, crowd, daytime outdoor,face, flower, ground vehicle, indoor, indoor sports, landscape,map,meeting, military, nighttime, office, outdoor, person, road, sky,

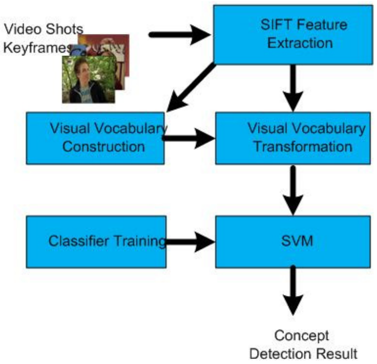


Figure 5: Flowchart for Concept detection based on BoW model

snow, stadium, tree, and vegetarian

3.4 Content-based Image Search

Content-based image search allows users to select a shot on the interface and to find shots visually-similar from the collection. For each keyframe in the collection we extracted three low level MPEG-7 features namely: Colour Layout, Edge Histogram and Scalable colour as described in Section 3.3.1.

For each feature we calculated the similarity between each pair of images in the collection. In order to reduce the space requirement for storing the resulting indexes we only stored the top 1000 similar keyframes for each keyframe. Having calculated the set of similar keyframes for each keyframe in the collection we then combine the scores for each feature into an overall similarity score for a pair of keyframes. For data fusion we we first normalise using MinMax normalisation, formally formally given by equation 1 before using CombSUM [4] fox to combine the normalised resultlists. The final similarity measures are then stored in the database for retrieval at query-time.

$$Norm_{score(x)} = \frac{Score_x - Score_{min}}{Score_{max} - Score_{min}} \tag{1}$$

3.5 Fusion Framework

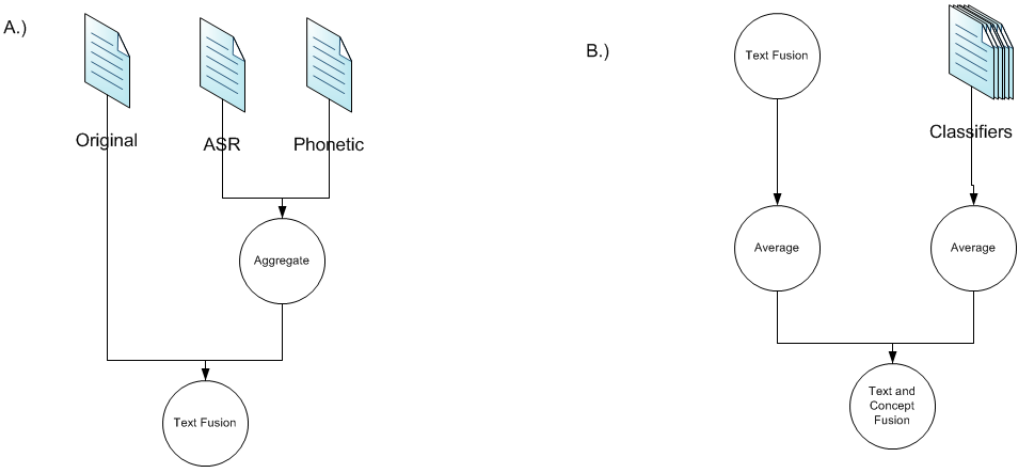


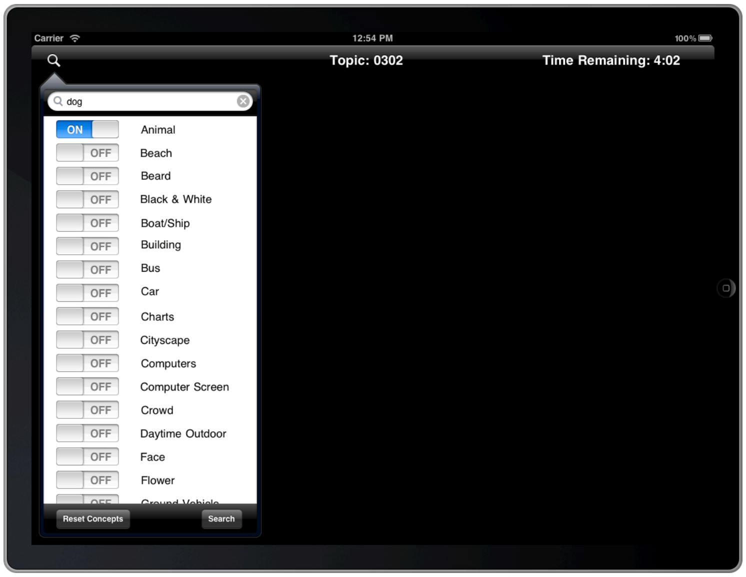
Figure 6: Overview of fusion (a) Text Fusion (b) Text and Concept Fusion

Our system has multiple data sources which require alignment, weighting and re-ranking, this is achieved by the utilisation of the following methods:

- **Text Only Fusion**. As seen previously, outputs from our text search engine are three ranked lists, one from each of our created indexes. We employ data fusion techniques to combine the data. First the ASR and Phonetic indexes are aggregated to video level, this aids the merger with the metadata which is already at video level, the unit of retrieval. Next we use MinMax normalisation before applying the weighting scheme on the resultant lists and evoking CombSUM [4] to combine the weighted normalised resultlists. This is illustrated in Figure 6, image A.
- **Text and Concept Fusion**. Our concepts differ from their usage in previous TRECVid systems and as such do not act as filters but instead are employed in a boosting technique when used in tandem with our text search. As illustrated in Figure 6 image B the text fusion described previously is used an input to our text and concept fusion. The resultlist is averaged based on the number of indexes, in this case three, and sent to merge with the normalised concept lists.

3.6 iPad User Interface

From a user interface (UI) perspective our goal was to develop a system that was easy and intuitive to use for both novice and expert users, while still allowing the user to utilise the underlying search technologies. This trade-off between the power of functionality and simplicity of use is a well know design issue. By using an iPad device with a touch screen input and by developing a new interface specifically designed for that device we aimed to strike a balance between functionality and ease of use.



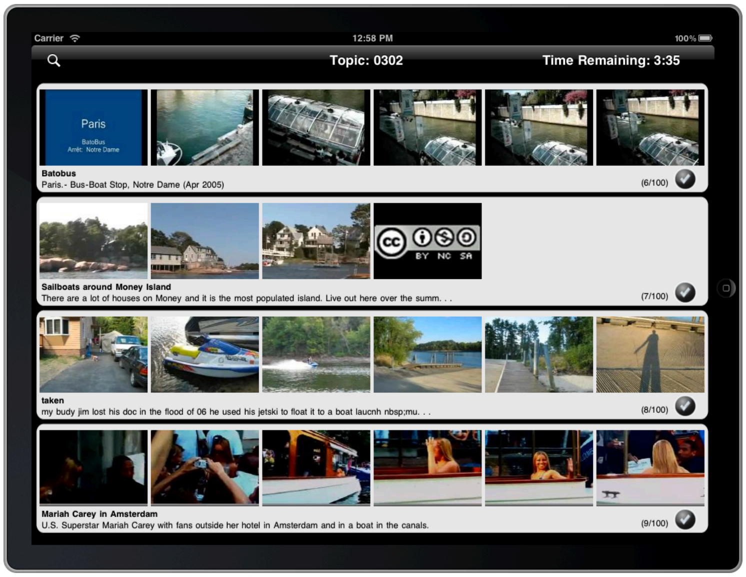
dog

ON	Animal
OFF	Beach
OFF	Beard
OFF	Black & White
OFF	Boat/Ship
OFF	Building
OFF	Bus
OFF	Car
OFF	Charts
OFF	Cityscape
OFF	Computers
OFF	Computer Screen
OFF	Crowd
OFF	Daytime Outdoor
OFF	Face
OFF	Flower
OFF	Ground Vehicle

Figure 7: Search Panel

Upon starting the application the user is required to enter a unique user id, which allows the system to control the tasks assigned to the user and the system can then track the progress of the user. Once the user has chosen to start a new topic they are presented with a search panel (as shown in Figure 7). Here they can input a text query as well as select from a list of 33 predefined semantic concepts. The video results are returned in ranked order to the user: for each video the title and description as well as a set of keyframes for each shot is shown (the user can scroll to the right to see more for each video) as shown in Figure 8. The top ranked shot for each video appears first in the list, with a maximum of 10 keyframes being displayed (selected temporally throughout the video).

By tapping on a keyframe the video will playback from that point and the user can also use the video seek bar to quickly scan through the video (shown in Figure 9). Once the user finds what they believe to be the correct video they can use the "check" button which queries the Oracle



Batobus
Paris.- Bus-Boat Stop, Notre Dame (Apr 2005)

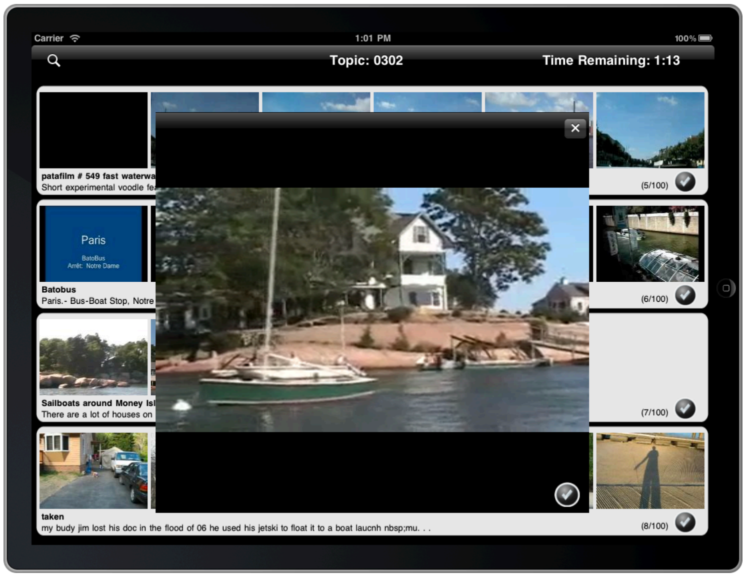
Sailboats around Money Island
There are a lot of houses on Money and it is the most populated island. Live out here over the summ. . .

taken
my budy jim lost his doc in the flood of 06 he used his jetski to float it to a boat laucnh nbsp;mu. . .

Mariah Carey in Amsterdam
U.S. Superstar Mariah Carey with fans outside her hotel in Amsterdam and in a boat in the canals.

Figure 8: Search Results

webservice and provides feedback to the user to let them know if they have found the correct video. At any point during the search the user can tap on the search icon, which displays the search panel and allows them to refine their search. In addition to this, by double tapping on any keyframe the user can invoke a content-based image similarity search that returns video keyframes that appear visually similar to the one they have selected, as shown in Figure 10. After the allocated 5 minutes have elapsed or after the user successfully finds the relevant video the system returns to a topic start page.



patafilm # 549 fast waterwa
Short experimental voodle fe

Batobus
Paris.- Bus-Boat Stop, Notre

Sailboats around Money Isl
There are a lot of houses on

taken
my budy jim lost his doc in the flood of 06 he used his jetski to float it to a boat laucnh nbsp;mu. . .

Figure 9: Video Playback

3.7 Experiments

Through our experiments in TRECVid this year we wanted to compare the performance of novice users against our in-house expert users. In particular we wanted to see if our attempts to develop a

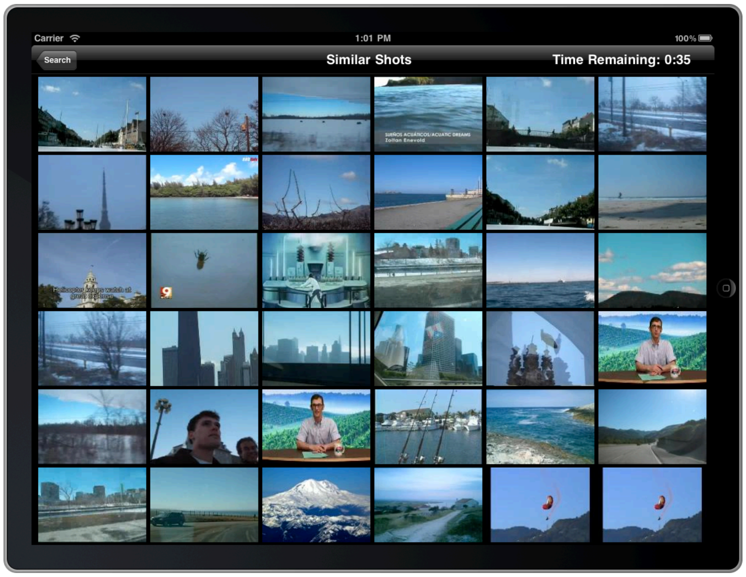


Figure 10: Similarity Search

content-based video search system which could be used by novices and experts alike were successful. Also, we wanted to compare the performance of our iPad search system against others from the community.

We recruited 6 users from our research group to complete the task in-house. All of these users had experience working with content-based video search systems and many had participated as users in previous TRECVid experiments completed in DCU, as such this group represents our *expert* users. We also recruited 12 users to participate through our iAD partners in the BI School of Management in Oslo, Norway. None of these users had experience using a sophisticated content-based video search system and none had hands-on experience with using an iPad before. These users represent our *novice* users.

Each participant completed 12 search topics and one training topic during the experiments. We used the latin-squares experimental design in order to assign users to topics and the ordering of presenting topics to each user was randomised in order to reduce the effects of learning bias.

The interactive known-item search task at TRECVid 2010 had 6 teams submit a total of 14 runs. Each run belonged to a certain category depending on the training type and whether the metadata XML was used or not. For both of our runs we used the metadata XML (condition: "YES") and used only the IACC training data (training type: "A"). Figure 11 presents the results for all submissions to the interactive known-item search, our two runs are highlighted. Both runs represent results from multiple users where we have picked the best time for each topic in order to populate our submission. Overall our runs came 6th and 7th, however when we compare ourselves against groups with the same condition and training type both runs move one place up to 5th and 6th.

In the expert run there were a total of 9 topics (out of a total of 22) for which none of our participants found the correct video, interestingly the novice users only missed 8. The fact that users could not find the correct video for these topics is not surprising, having observed the user experiments it was clear that users found the topics to be either very easy or very difficult.

Figure 12 presents the results for user satisfaction as compared across the groups. Again our results here are about mid-table. Perhaps more interestingly for us, as part of our post-experiment questionnaire we asked our users to score the system in terms of ease-of-use on a scale of 1-7, for this our novice users gave the system a median score of 6, with experts giving a median score of 6.5.

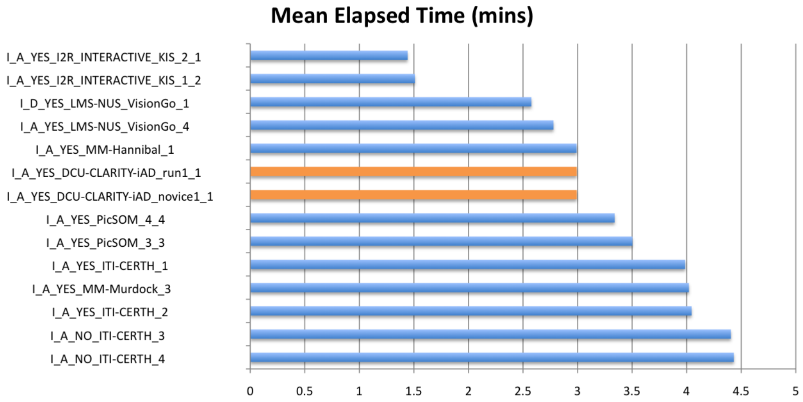


Figure 11: Results from the 2010 TRECVid known-item search, the results of our two runs are highlighted

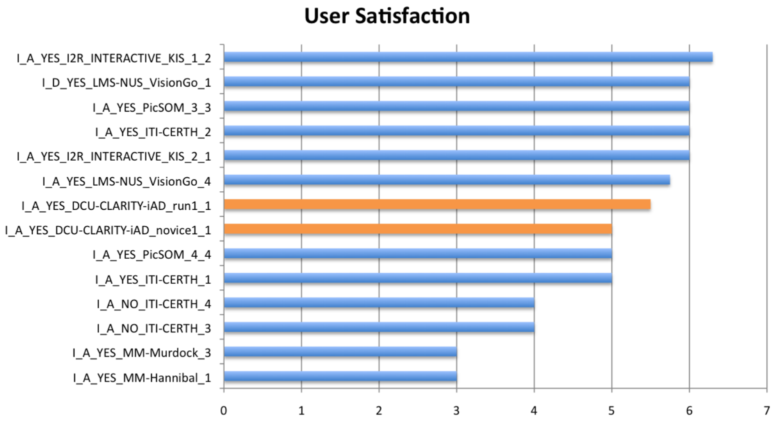


Figure 12: User-Satisfaction results from the 2010 TRECVid known-item search, the results of our two runs are highlighted

4 Conclusions

In this paper we presented our experiments in Instance Search and Interactive search at this year's TRECVid workshop.

Our instance search system was based on using support vector machines trained on each query topic to classify objects and parts of objects in each keyframe. To achieve this, keyframes were first decomposed into a hierarchical region representation using binary partition tree segmentation. Using this representation permitted extracting descriptors of the objects and regions in an image on a coarse-to-fine scale. Classifiers were then applied to these representations in a top-down approach, recursively checking if the query object was likely to be found somewhere below the current node in the tree. The tree nodes found in this recursive search were then combined and matched against the query topic using a second object matching classifier. Although there were problems with our implementation, we believe that the general approach has promise, and plan to continue the work in the future.

For interactive known-item search our approach this year was to develop a system which appeals to the vast majority of users which search for video daily on YouTube. To this end we attempted to develop a video search engine which supports content-based searching and is accessible to users of all abilities. The results from our official experiments place our runs at 6th and 7th overall and 5th and 6th when we compare against systems using the same conditions. Our results show that the performance of novices versus experts is identical in terms of mean elapsed time. Through our post-experiment analysis we are investigating why this is the case. One explanation would be that our attempts to build a search engine that could be used by novices and experts alike was successful. Another explanation could lie in the topics used in the search task. Through observations of the experiments we found that both sets of users found the majority of topics to be either very easy or very difficult. The lack of topics of medium difficulty may have constrained our ability to distinguish the differences in performance of different users. Nonetheless through our experimental logs and questionnaires we can still gain valuable insights into the techniques used by both sets of users and their experiences in using our system.

Acknowledgements

We would like to thank Espen Andersen and his students in BI, Oslo for participating in our experiments. The research was funded by by Science Foundation Ireland under grant 07/CE/I1147 and through iAD - information Access Disruptions, a centre for research-based innovation with CRI number: 174867, funded in part by the Norwegian Research Council.

References

[1] C. Chang, C.C. And Lin. Vector machines. *LIBSVM: a Library for Support:http://www.csie.ntu.edu.tw/˜cjlin/libsvm/*, 2001.

[2] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 19:1–16, 2007.

[3] C. Foley, C. Gurrin, G. Jones, H. Lee, S. M. Givney, N. O'Connor, S. Sav, A. F. Smeaton, and P. Wilkins. Trecvid 2005 experiments at dublin city university. In *TRECVid 2005 - Text REtrieval Conference TRECVID Workshop*, MD, USA, 2005. National Institute of Standards and Technology.

[4] E. A. Fox and J. A. Shaw. Combination of Multiple Searches. In *Text {REtrieval} Conference*, pages 243–249, 1994.

[5] J.-L. Gauvain, L. Lamel, and G. Adda. The limsi broadcast news transcription system. *Speech Commun.*, 37(1-2):89–108, 2002.

[6] A. M. Khan, K. S. Mckinley, R. Bentzur, D. Feinberg, D. Frampton, S. Z. Guyer, M. Hirzel, A. Hosking, M. Jump, H. Lee, J. Eliot, B. Moss, A. Phansalkar, D. Stefanovi?, T. Vandrunen, D. V. Dincklage, P. Christen, and P. Christen. A comparison of personal name matching: Techniques and practical issues. In *in Workshop on Mining Complex Data (MCD06), held at IEEE ICDM06, Hong Kong*, pages 290–294, 2006.

[7] M. Koskela, P. Wilkins, T. Adamek, A. F. Smeaton, and N. O'Connor. Trecvid 2006 experiments at dublin city university. In *TRECVid 2006 - Text REtrieval Conference TRECVid Workshop*, 2006.

[8] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int'l J. Computer Vision*, 60:91–110, 2004.

[9] MPEG. MPEG-7 Overview(Version 10). *http://mpeg.chiariglione.org/standards/mpeg-7/mpeg-7.htm.*

[10] NYSIIS. Comprehensive perl archive network. [online]. *http://search.cpan.org/?krburton/String-Nysiis-1.00/Nysiis.pm.*

[11] I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald, and C. Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006)*, 2006.

[12] P. Wilkins, T. Adamek, G. Jones, N. O'Connor, and A. F. Smeaton. Trecvid 2007 experiments at dublin city university. In *TRECVid 2007 - Text REtrieval Conference TRECVid Workshop*, 2007.