An Integrated Approach to Text and Image Retrieval

The Lowlands Team at Trecvid 2005

Thijs Westerveld[1] Jan C. van Gemert[2] Roberto Cornacchia[1]
Djoerd Hiemstra[3] Arjen P. de Vries[1]

[1]CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands
{thijs,roberto,arjen}@cwi.nl

[2]University of Amsterdam
Kruislaan 403
1098 SJ Amsterdam
the Netherlands
jvgemert@science.uva.nl

[3]University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands
d.hiemstra@utwente.nl

Abstract

Our main focus for this year was on setting up a flexible retrieval environment rather than on evaluating novel video retrieval approaches. In this structured abstract the submitted runs are briefly described.

High-level feature extraction

We experimented with feature detectors based on visual information only, and compared Weibull-based and GMM-based detectors.

- LL-HF-WB-VisOnly Region-based Weibull models, visual only
- LL-HF-WBNWC-VisOnly Extended region-based Weibull models, visual only
- LL-HF-GMMQGM-VisOnly GMM-based models, query generation variant
- LL-HF-GMMDGM-VisOnly GMM-based models, document generation variant

We found large differences across topics. Some models are good for one topic other for the next. Future research has to show whether a combined approach is useful.

Search

In the search task we focused on a seamless integration of our visual and textual retrieval system, to allow for easy multimodal querying. We use the Nexi language for querying (see Section 3.1) and Ram for specifying visual retrieval models (see Section 3.3).

- M-A-1-LL-ram-text-1 manual text only run
- F-A-1-LL-ram-text-2 fully automatic text only run
- M-A-2-LL-ram-text-im-3 manual text + image run
- M-A-2-LL-ram-text-feat-4 manual text + high-level feature run
- M-A-2-LL-ram-text-im-feat-5 manual text + image + high-level feature run
- F-A-2-LL-ram-text-im-6 fully automatic text + image run
- F-A-1-LL-tijahpsql-text-7 fully automatic text only run

We experimented with a generic retrieval approach that used collection specific information only for training the high-level feature detectors. Runs making use of textual information perform around the median, adding visual information does not influence the results.

1 Introduction

The main focus of our participation in this year's TRECVID benchmarks has been on engineering rather than on retrieval or video modeling. We aim at developing an information retrieval framework that supports many diverse retrieval applications by means of one simple yet powerful query language that hides the implementation details from the application developer, while still giving control over the ranking process. Our TRECVID experiments are run using an XML retrieval system based on MONETDB, an open source database system developed at CWI [2]. Experiments for other benchmarks (HARD and Enterprise tracks at TREC and various tasks at INEX) are carried out using the same system.

The paper is organized as follows. We start with a description of the retrieval models used in the system in Section 2. Section 3 discusses the separate parts of the retrieval systems as they have been used so far, as well as the integration of these parts as a step towards the development of a generic search engine framework. Experimental results for the search and high-level feature tasks are discussed in Section 4. The paper ends with discussion on the envisaged search engine framework.

2 Retrieval models

We use probabilistic models to describe our data. For modeling text, we use statistical language models, visual data is modeled using either Weibull distributions or Gaussian mixture models The following subsections introduce the various models.

2.1 Language models

We model textual information, i.e., speech transcripts and machine translation output following the language modeling (LM) approach to information retrieval [14, 8]. For each textual segment a language model is estimated based on the distributions of terms in the segment. The models we use are hierarchical language models; they take the hierarchical structure of video into account. The likelihood of a shot given a query $Q = \{q_1, q_2, \ldots, q_n\}$ is estimated using an interpolation of the likelihoods of the query terms given the language models for the shot itself, its containing scene, the containing video and the collection:

$$\text{score}(\text{shot}_i|Q) = \prod_{q \in Q} [\alpha P(q_j|\text{Shot}_i) + \beta P(q_j|\text{Scene}_i) + \gamma P(q_j|\text{Video}_i) + \delta P(q_j|\text{Collection})] \quad (1)$$

The parameters α, β, γ and δ are mixing weights and sum to 1.

2.2 Weibull models for Visual Features

Modeling visual data heavily relies on qualitative features. Good features describe the relevant information in an image while reducing the amount of data representing the image. To achieve this goal, we use Wiccest features as introduced in [5]. Wiccest features combine color invariance with natural image statistics. Color invariance aims to remove accidental lighting conditions, while natural image statistics efficiently represent image data.

Color invariance aims at keeping the measurements constant under varying intensity, viewpoint and shading. In [6] several color invariants are described. We use the $\mathcal{W}$ invariant that normalizes the spectral information with the energy. This normalization makes the measurements independent of illumination changes under uniform lighting conditions.

When modeling scenes, edges are highly informative. Edges reveal where one region ends and another begins. Thus, an edge has at least twice the information content of a uniformly colored patch, since an edge contains information about all regions it divides. Besides serving as region boundaries, an ensemble of edges describes texture information. Texture characterizes the material an object is made of. Moreover,a compilation of cluttered objects can be described as texture information. Therefore, a scene can be modeled with textured regions.

Texture is described by the distribution of edges at a certain region in an image. Hence, a histogram of Gaussian derivative filters represents the edge statistics. Since there are more non-edge pixels then edge pixels, the distribution of edge responses for natural images always has a peak around zero, i.e.: many pixels have no edge responses. Additionally, the shape of the tails of the distribution is often in-between a power-law and a Gaussian distribution. This specific distribution can be well modeled with an integrated Weibull distribution [7]. This distribution is given by

$$\frac{\gamma}{2\gamma^{\frac{1}{\gamma}}\beta\Gamma(\frac{1}{\gamma})}\exp\left\{-\frac{1}{\gamma}\left|\frac{r-\mu}{\beta}\right|^{\gamma}\right\} \quad , \tag{2}$$

where r is the edge response to the Gaussian derivative filter and $\Gamma(\cdot)$ is the complete Gamma function, $\Gamma(x) = \int_0^\infty t^{x-1}e^{-1}dt$. The parameter β denotes the width of the distribution, the parameter γ represents the 'peakedness' of the distribution, and the parameter μ denotes the origin of the distribution.

To assess the similarity between Wiccest features, a goodness-of-fit test is utilized. The measure is based on the integrated squared error between the two cumulative distributions, which is obtained by a Cramér-von Mises measure. For two Weibull distributions with parameters F_β, F_γ and G_β, G_γ a first order Taylor approximation of the Cramér-von Mises statistic yields the log difference between the parameters. Therefore, a measure of similarity between two Weibull distributions F and G is given by the ratio of the parameters,

$$W^2(F,G) = \sqrt{\frac{\min(F_\beta,G_\beta)}{\max(F_\beta,G_\beta)}\frac{\min(F_\gamma,G_\gamma)}{\max(F_\gamma,G_\gamma)}} \quad . \tag{3}$$

The μ parameter represents the mode of the distribution. The position of the mode is influenced by uneven illumination and colored illumination. Hence, to achieve color constancy the values for μ may be ignored.

In summary, Wiccest features provide a color invariant texture descriptor. Moreover, the features rely heavily on natural image statistics to compactly represent the visual information.

2.3 Gaussian mixture models for visual features

We also experimented with Gaussian mixture models (GMMs) for modeling visual information. Here, keyframe images (w_i) are modeled as mixtures of Gaussians with a fixed number of components C:

$$P(\boldsymbol{x}|\omega_i) = \sum_{c=1}^{N_C} P(C_{i,c})\ \mathcal{G}(\boldsymbol{x}, \boldsymbol{\mu}_{i,c}, \boldsymbol{\Sigma}_{i,c}), \tag{4}$$

where N_C is the number of components in the mixture model, $C_{i,c}$ is component c of class model ω_i and $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the Gaussian density with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$:

$$\mathcal{G}(\boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^n|\boldsymbol{\Sigma}|}}e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}, \tag{5}$$

where n is the dimensionality of the feature space and $(\boldsymbol{x}-\boldsymbol{\mu})^T$ is the matrix transpose of $(\boldsymbol{x}-\boldsymbol{\mu})$. These Gaussian mixture models of the keyframes are used to represent the shots. The score of a shot given an video or image example, is determined by the likelihood that the corresponding model generates the feature vectors ($\mathcal{X} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}\}$) representing the example, just like in the LM case for text, we interpolate with a background model based on collection statistics:

$$score(\text{shot}_i) = \prod_{\boldsymbol{x}\in\mathcal{X}} \left[\lambda \cdot P(\boldsymbol{x}|\omega_i) + (1-\lambda)\cdot P(\boldsymbol{x}|\omega_i)\right] \tag{6}$$

The feature space of the vectors $\boldsymbol{x}$ is based on the DCT coefficients obtained from 8x8 pixel blocks. For more details of the features and the GMMs, see [22, 21, 20].

3 Retrieval Systems

3.1 xml text retrieval system

Our XML text retrieval system, TIJAH is based on the scored region algebra approach [11]. Each element in an XML tree naturally represents a region in the document. The scored region algebra provides functionality for scoring these regions (for example based on the language modeling principle), and for combining scored regions in a principled manner. To give an idea of the kind of combinations that this algebra is capable of it is useful to first briefly introduce the query language we use, NEXI.

NEXI (Narrow Extended XPath for INEX) is an XML query language developed for the INEX XML retrieval benchmark [18]. The language is a modification of XPath. The only axis steps it allows are descendant steps, but the filtering conditions are extended with *about* clauses. These clauses provide information retrieval functionality and allow the user to ask for sections about XML

`//Section[about(.,XML)]`

or speech segments about George W. Bush

`//SpeechSegment[about(.,George W. Bush)]`.

The latter already exemplifies one case where score combination is needed: scores for the separate query terms *George*, *W.* and *Bush* have to be combined in a meaningful manner. Another kind of combining scores comes to play when NEXI queries become more complicated and ask for contextual information. For example, speech segments about George W. Bush in videos about transportation

`//Video[about(.,transportation)]//SpeechSegment[about(.,George W. Bush)]`.

Given a NEXI query, the retrieval system returns a ranked list of XML elements. Performing the required text only run using this system is relatively simple. Assuming the ASR transcripts are pre-processed and segmented in such a manner that each shot is represented in an MPEG-7 document as a VideoSegment element with its own piece of text[1], then a simple NEXI query like

`//VideoSegment[about(.,airplane taking off)]`

will rank the shots in the collection.

Our XML retrieval system is a flexible framework offering a variety of retrieval models. However, for the experiments in the present paper, we restrict ourselves to the hierarchical language modeling approach to information retrieval as described in Section 2.1. This means each NEXI about clause (`about(`x`,Q)`) is translated to $P(\mathrm{x}|Q)$ (cf. Eq. 1).

3.2 Region queries for LMs

A simple query like `//VideoSegment[about(.,airplane taking off)]` will be translated to one or more scored region algebra expressions, for which the system then chooses a query execution plan, similar to rewriting SQL queries into relational algebra expressions in relational database systems. The translation to scored region algebra of the query above might for instance be the following:

`((<VideoSegment> CONTAINING 'airplane') CONTAINING 'taking') CONTAINING 'off'`

This will only find shots that contain all of the terms 'airplane', 'taking' and 'off' but no other shots. It is very unlikely that these shots exist, because shots are rather small, and the only way a shot could contain the exact query words is when a voice over says: "(I see an) airplane taking off" at exactly the same time when the video shows the airplane taking off (*and* the speech recognition system recognized the words correctly). So, the query plan above will not make a very good video retrieval system.

For this reason, the translation from conceptual query language (NEXI) to logical query plan (scored region algebra) is not fully determined in TIJAH. The translation of NEXI's *about* function has to be set by the application developer. This enables flexible ranking in TIJAH. For example, when developing a web search engine, the algebra expression will contain a static ranking component such as the Google PageRank of a page. When developing a video search engine that uses speech transcripts to retrieve video shots, the region algebra expression should contain the components from Equation 1. That is, the score of the shot is combined with the score of the scene that contains the shot, and combined with the score of the video that contains the shot, and combined with a so-called background score for the

[1] Section 4.2.1 discusses the pre-processing steps

term. Language modeling queries such as the one defined by Equation 1 can be processed as scored region queries in a straight-forward manner [10] as shown in Figure 1 where $\alpha = 0.4$, $\beta = 0.4$, $\gamma = 0.02$ and $\delta = 0.18$.

```
R1 := <VideoSegment>;
R2 := (0.4 SCALE R1) OR (0.133 SCALE (R1 ADJ R1 ADJ R1)) OR (0.2 SCALE <Video>) OR
      (0.18 SCALE <root>) ;
R3 := R1 CONTAINED_BY (R2 CONTAINING 'airplane');
R4 := R3 CONTAINED_BY (R2 CONTAINING 'taking');
R5 := R4 CONTAINED_BY (R2 CONTAINING 'off');
```

Figure 1: Region query for LM retrieval model

We do not know where the scene boundaries are, so we glue adjacent shots together to form a virtual scene. This is expressed by `(R1 ADJ R1 ADJ R1)`. Because of this, scenes will overlap and shots are contained in multiple scenes; in the example in three scenes, so we down-weighted β to $0.4/3 = 0.133$. For this reason, the results of the scored region algebra expression deviate somewhat from the results that are defined by Equation 1.

An alternative approach would be to introduce another implementation for the `CONTAINING` operator in our algebra. Different implementations of the `CONTAINING` operator are easily added to TIJAH, for instance to introduce alternative retrieval models and term weighting algorithms [12]. The hierarchical language model of Equation 1 might be put completely inside a `V_CONTAINING` operator (for 'video containing'), such that the query is translated simply into:

```
((<VideoSegment> V_CONTAINING 'airplane') V_CONTAINING 'taking') V_CONTAINING 'off'
```

However, this forces the application developer to directly operate on the physical storage structures of the system. We believe the former approach, where the application developer only operates on logical expressions, is more elegant and is more likely to be done by people who are no expert on the internal workings of the system.

3.3 Ram for GMMs

For our participation to last year's TRECVID, we implemented the GMM retrieval model for visual information (Section 2.3) on top of MONETDB, using its native MIL query language. The resulting MIL query processed faster than its MATLAB equivalent, which we used in the past. However, a manual writing of such complex relational queries is a difficult and error-prone task.

This year, we used RAM to realize the same retrieval model on top of MONETDB. RAM (Relational Array Mapping) [19] is a tool developed at CWI to facilitate the implementation of scientific applications on top of relational database systems. The basic data structure of RAM is the multi-dimensional array. The RAM front-end provides a comprehension based array query-language, to express array-specific queries concisely (comprehension syntax is explained in [3]). Array comprehensions allow users to specify a new array by declaring its dimensions and a function to compute the value for each of its cells. An optimized relational plan is internally devised for the array query at hand and this is finally translated to an actual query, ready to be fed to the preferred relational back-end.

The main advantage of this approach is that the application developer can concentrate on the retrieval model and express it in a declarative manner, while the physical implementation is realized by the system. This abstraction is similar to the one provided by SQL interfaces to database systems. However, the array paradigm can be seen as a further abstraction over the relational layer, which introduces a new step: the array query plan is translated to an optimized relational query plan, which is then translated to an optimized physical query plan by the relational engine, as usual.

The RAM array-query that implements the GMM retrieval model is showed in Figure 2. Notice that no details about the physical organization of the data or the physical plan are explicit in the query, which results in extremely compact and readable notation, close to the original mathematical formulas (cf. Eq 4, 5 and 6). The following persistent arrays, resulting from the indexing phase, are used: `Pr`=$P(C)$, `Mu`=$\boldsymbol{\mu}$, `S`=$\boldsymbol{\Sigma}$, `X`=$\boldsymbol{x}$. Variables and constants start with `$`.

```
Scores     = [ avg([log((0.9*psd(s,d)) + (0.1*ps(s))) | s<$NSamples]) | d<$NDocs ]
ps(s)      = avg([ psd(s,d) | d<$NDocs ])
psd(s,d)   = sum([ Pr(c,d) * norm(c,d) * act(c,s,d) | c<$NComps ])
act(c,s,d) = exp(-0.5 * sum([pow(X(n,s)-Mu(n,c,d), 2) / S(n,c,d) | n<$NFeats ]))
norm(c,d)  = 1 / sqrt(pow(2*$PI,$NFeats) * prod([S(n,c,d) | n<$NFeats ]))
```

Figure 2: RAM query for GMM retrieval model

This query was translated by RAM and executed by MONETDB/X100 [23], the new pipelined database engine currently being developed at CWI. The satisfying performance measured (comparable to our manual implementation of last year) was a welcome result, next to the achievement of our original main goal: reduce the user effort in formulating the desired retrieval model on top of a DBMS.

As a next step toward this goal, we realized an integration of the RAM-powered visual retrieval in the TIJAH probabilistic framework. The integration realized is only at its first stage: limited to the GMM retrieval model for visual information only, not flexible (it is TRECVID-specific), and not particularly elegant. However, it served as a useful case study for our work on the parameterized search system that we envisage for the next future: SPIEGLE.

3.4 Spiegle for multimedia

SPIEGLE is a system for generic information retrieval from a multitude of collections and data types. The idea is that the system is flexible and easily configurable to the needs of a specific collection. The loosely coupled integration of TIJAH and RAM as used for this year's TRECVID experiments is a first step towards the parameterizable search engine generator that SPIEGLE will become.

Two main adaptations of the TIJAH system are needed to handle the TRECVID collection:

- The use of the hierarchical language model instead of the commonly used two level mixture of foreground and background probabilities.
- Mapping TIJAH's XML element identifiers to the shot identifiers as used in the RAM system (and in TRECVID's evaluation).

For the hierarchical language model, we need to administer the links between shots, scenes and videos. Since no scene segmentation was available for the collection, we simply assumed each window of five consecutive shots forms a scene. This scene information is not explicitly listed in the XML documents, hence we need to keep tables that link shots to scenes and to videos. Based on these tables term frequencies in shots can be aggregated to obtain scene and video statistics and to compute hierarchical language model scores from them.

To be able to use the visual parts of the topics, i.e., the example images and example videos, we extended the NEXI language with a special about clause: `imabout()`. This clause takes as arguments a set of XML elements (like the original about clause, and an identifier of the query (e.g., *topic167_7*, to refer to the 7th example in topic167). This `imabout()` clause is translated to a call to the GMM functions implemented in RAM (Section 3.3. To facilitate linking from the XML elements to shot identifiers, we stored the identifiers explicitly as attributes of the `<VideoSegment>` elements in the XML document. Another prerequisite is that all visual examples are pre-processed (i.e., stored as feature vectors) and known to the RAM module.

4 Experimental results

4.1 High-level feature extraction

For the high-level feature task, we experimented with a visual information only approach. This section first describes how the Weibull similarity as described in Section 2.2 and the Gaussian mixture models (Section 2.3) are used for high-level feature extraction and then discusses our results.

4.1.1 Contextures: Regional Texture Descriptors and their Context

Building towards semantic access to video collections, we aim to express complex scenes in low-level semantics like vegetation, water, fire, sky etc. These proto-concepts provide a first step to automatic access to video content. Given a fixed vocabulary of proto-concepts, we assign a confidence measure of the occurrence of the proto-concepts in a shot. Different combinations of an occurrence histogram of proto-concepts provide a sufficient characterization of a complex scene. We introduce the notion of contextures, where a configuration of global texture and local texture information and its context are used to describe visual scene information.

In order to recognize concepts based on low-level visual analysis, we annotated 15 different proto-concepts: building (321), car (192), charts (52), crowd (270), desert (82), fire (67), US-flag (98), maps (44), mountain (41), road (143), sky (291), smoke (64), snow (24), vegetation (242), water (108), where the number in brackets indicates the number of annotation samples of that concept. Fig. 3 shows an example of some regional annotations. We again used the TRECVID 2005 common annotation effort as a basis for selecting relevant shots containing the proto-concepts. In those shots, we annotated rectangular regions where the proto-concept is visible for at least 20 frames.

Sky Building Road

Figure 3: Three examples of annotated regions in video.

The visual detectors aim to decompose an image in proto-concepts like vegetation, water, fire, sky etc. To achieve this goal, an image is divided up in several overlapping rectangular regions. The regions are uniformly sampled across the image, with a step size of half a region. The region size has to be large enough to assess statistical relevance, and small enough to capture local textures in an image. We utilize a multi-scale approach, using small and large regions. An example of region sampling is displayed in figure 4.

A visual scene is characterized by both global as well as local texture information. For example, a picture with an aircraft in mid air might be described as "sky, with a hole in it". To model this type of information, we use a proto-concept occurrence histogram where each bin is a proto-concept. The values in the histogram are the similarity responses of each proto-concept annotation, to the regions in the image.

We use the proto-concept occurrence histogram to characterize both global and local texture information. Global information is described by computing an occurrence histogram accumulated over all regions in the image. Local information is taken into account by constructing another occurrence histogram for only the response of the best region. For each proto-concept, or bin, b the accumulated occurrence histogram and the best occurrence histogram are constructed by,

$$\begin{aligned} H_{accumulated}(b) &= \sum_{r \in R(im)} \sum_{a \in A(b)} W^2(a,r) \quad , \\ H_{best}(b) &= \arg\max_{r \in R(im)} \sum_{a \in A(b)} W^2(a,r) \quad , \end{aligned}$$

where $R(im)$ denotes the set of regions in image im, A(b) represents the set of stored annotations for proto-concept b, and W^2 is the Cramér-von Mises statistic as introduced in equation 3.

We denote a proto-concept occurrence histogram as a contexture for that image. We have chosen this name, as our method incorporates texture features in a context. The texture features are given by

Figure 4: An example of dividing an image up in overlapping regions. In this particular example, the region size is a $\frac{1}{2}$ of the image size for both the x-dimension and y-dimension. The regions are uniformly sampled across the image with a step size of half a region. Sampling in this manner identifies nine overlapping regions.

the use of Wiccest features, using color invariance and natural image statistics. Furthermore, context is taken into account by the combination of both local and global region combinations.

Contextures can be computed for different parameter settings. Specifically, we calculate the contextures at scales $\sigma = 1$ and $\sigma = 3$ of the Gaussian filter. Furthermore, we use two different region sizes, with ratios of $\frac{1}{2}$ and $\frac{1}{6}$ of the x-dimension and y-dimensions of the image. Moreover, contextures are based on one image, and not based on a shot. To generalize our approach to shot level, we extract 1 frame per second out of the video, and then aggregate the frames that belong to the same shot. We use two ways to aggregate frames: 1) average the contexture responses for all extracted frames in a shot and 2) keep the maximum response of all frames in a shot. This aggregation strategy accounts for information about the whole shots, and information about accidental frames, which might occur with high camera motion.

4.1.2 GMMs for high-level feature extraction

We compared the contexture based approach to a GMM based approach. This approach basically treats feature extraction as a form of query by example searching. The only difference with the traditional search task is that instead of one or a few examples, we now have a large set of annotated data as an example of a given feature. For the GMM based approach we used the regional annotations described in the previous section. Starting from this set of feature examples, we followed two strategies:

QueryGeneration Ranking of the shots is based on the likelihood that the shot's model generates the set of feature examples. In this approach, a GMM (ω_i) is estimated for each shot in the collection; models are ranked using Equation 6 where the set of feature vectors $\mathcal{X}$ is the union of the feature vectors obtained from all annotated regions.

DocumentGeneration The process is reversed; Shots are ranked by the likelihood of being generated by a high-level feature model. In this strategy, the roles in Equation 6 are reversed: ω_i is a model estimated from the annotated regions, and $\mathcal{X}$ are the shots of a shot.

4.1.3 Results

We aim to investigate the building blocks of the contexture features. Contextures may be used to describe a visual scene by decomposing a scene in proto-concepts. Because the proto-concepts are an important part of the contextures, we investigate the results of proto-concept detection only, where the whole scene contextures were submitted by another group [17].

Since we use the proto-concepts only, this corresponds to the responses of single bins of the occurrence histogram. Consequently, we could only submit results for the concepts we annotated, see section **??**. For the 3 TRECVID topics we did not annotate, we chose a prototype that was somewhat related, i.e.: 38:People walking = .crowd., 45:prisoner = .fire., sports=.vegetation..

Table 1: AP per topic for submitted runs on high-level feature task.

run ID	38	39	40	41	42	43	44	45	46	47
LL-HF-GMMDGM-VisOnly	0.112	0.003	0.036	0.004	0.060	0.082	0.028	0.001	0.074	0.019
LL-HF-GMMQGM-VisOnly	0.068	0.002	0.097	0.003	0.012	0.005	0.011	0.000	0.001	0.022
LL-HF-WBNWC-VisOnly	0.118	0.001	0.094	0.025	0.084	0.070	0.042	0.000	0.011	0.067
LL-HF-WB-VisOnly	0.115	0.010	0.123	0.046	0.083	0.063	0.015	0.000	0.021	0.020

We submitted two versions of the visual-only features. The difference is caused by ongoing development on the visual analysis. Specifically, we improved the Weibull fit to be more robust and we added the proto-concept *car*. To compare the results between the new (LL-HF-WBNWC-VisOnly) and the old (LL-HF-VisOnly)version, we submitted both versions.

Comparing the results of the two versions of the visual features, it shows that the newer features are slightly better. For the seven topics where we had annotated proto-concepts for, the new features outperform the old features for four topics. For four topics (42:Building, 43:Waterscape, 44:Mountain,47:Car) the new features are better, where for three topics (39:Explosion, 40:Map, 41:US-flag) the old features perform better.

In the GMM based variants we do not see as many differences across topics, document generation outperforms query generation for all topics (although for some topics the difference is small). Comparing the GMM based approach to the contextures results, again we see differences between topics. For many topics the difference between the approaches is small, but for some topics the contexture approach is clearly better (40:Map, 44:Mountain, 47:Car), for others the GMM based document generation approach gives better results (43: Waterscape, 46:Sports).

Average precision scores for both contexture based and GMM based variants are listed in Table 1. Future research will have to show whether a combination of the various models can improve results. We also plan to investigate the combination of the proto-concepts with textual information.

4.2 Search

For the search section we experimented with the integrated Tijah and Ram systems. Simple Nexi queries allow for easy experimentation with combinations of modalities. This section describes the pre-processing of the textual data and the experimental runs we did.

4.2.1 Pre-processing

An intrinsic problem in video analysis is the multi modality of the data stream and the lack of alignment between the different modalities. The original speech recognition output is typically segmented using silence or speaker turns. Clearly these do not necessarily match shot boundaries. In previous editions of Trecvid, the speech transcripts were cut up to match the shots. This may however not be the best basis for text retrieval, since the coherence of the textual parts is lost. Ideally, searching in the ASR output should be based on the speech segments, but then it is hard to combine with visual information or to relate the results to the predefined master shot reference [13]. How to properly model retrieval from mis-aligned resources is planned for future work.

For the present experiments however, we constructed xml documents in which the text was aligned to the shots. Exact cutting of the speech transcripts at the shot boundaries was not possible, because timings in the transcripts were only available at the segment level. We used an XQuery extension that allows selecting of xml elements based on the position of the piece of data it refers to, rather than the position of the element in the xml document [1]. This allows us to construct a document of shots, with for each shot all overlapping speech segments. Note that this means that a speech segment on a shot boundary gets assigned to both shots. The constructed documents are indexed using the Tijah system and used in our retrieval experiments. Figure 5 shows a fragment of one of the documents in the collection.

```
<VideoSegment id="shot100_10" in_msec="33500" out_msec="37136">
  <text record_id="7">
  <timespan out_msec="35930" in_msec="32260" />car emissions
  cents)</text>
  <text record_id="8">
  <timespan out_msec="36650" in_msec="35930" />, was</text>
 </VideoSegment>
 <VideoSegment id="shot100_11" in_msec="37137" out_msec="40240">
  <text record_id="9">
  <timespan out_msec="42840" in_msec="37250" />, "the most
  memorable assignments for us to</text>
 </VideoSegment>
 <VideoSegment id="shot100_12" in_msec="40240" out_msec="42909">
  <text record_id="9">
  <timespan out_msec="42840" in_msec="37250" />, "the most
  memorable assignments for us to</text>
 </VideoSegment>
 <VideoSegment id="shot100_13" in_msec="42909" out_msec="47113">
  <text record_id="10">
  <timespan out_msec="46820" in_msec="42950" />twenty seven
  MSNBC</text>
 </VideoSegment>
```

Figure 5: Example fragment of indexed document

4.2.2 Runs

The integration of the retrieval systems for the various modalities, and more importantly the adoption of the NEXI query language allowed for easy testing of combinations of different modalities. Adding an extra modality now boils down to adding an extra `about()` clause to the NEXI query. For example, the following queries are used for some of our manual runs for topic 167 (*Find shots of an airplane taking off*).

```
//VideoSegment[about(.,airplane taking off)]
```

```
//VideoSegment[about(.,airplane taking off) and imabout(.,topic0167_7)]
```

```
//VideoSegment[about(.,airplane taking off) and imabout(.,topic0167_7) and featabout(.,Sky)]
```

As stated before, this years' efforts were targeted more at combining systems than at comparing video retrieval approaches. Nevertheless, the results of the various approaches are shown in Table 2. Clearly the usefulness of the image models tested is limited. Runs that make no use of textual information have extremely low scores, and for runs that do use textual information, using additional information from the image models does not help. A combination of textual search and high-level feature search (M-A-2-LL-RAM-TEXT-FEAT-4) is –according to the Wilcoxon signed-rank test– significantly better than the corresponding text only baseline (M-A-1-LL-RAM-TEXT-1), but the difference is negligible.

In general, performance is relatively poor compared to the top ranked systems participating at TRECVID. Our runs are often at or just below the median performance. We would like to stress once more though, that this performance is achieved without any collection specific training or analysis.

5 Discussion

The shallow integration of the TIJAH and RAM systems used this year for combined textual and visual searching in the TRECVID collection, was a useful case study for our work on the parameterized search engine SPIEGLE. In past TRECVID and TREC evaluations, we have used MonetDB [4] as well as standard information retrieval software like the TNO VSM engine [9] and Lemur [15] to develop new applications of information retrieval technology. For these TREC participations, it was often necessary to re-implement parts of the existing system, such as reimplementing APIs, introducing new APIs, and sometimes introducing new indexing and storage structures. Of course, the development of research prototypes is a tedious and time-consuming job, but in our experience, deploying information retrieval

Table 2: MAP for submitted and supplemental runs on search task

run ID	MAP
M-A-1-LL-RAM-TEXT-1	0.0346
F-A-1-LL-RAM-TEXT-2	0.0323
M-A-2-LL-RAM-TEXT-IM-3	0.0331
M-A-2-LL-RAM-TEXT-FEAT-4	0.0347
M-A-2-LL-RAM-TEXT-IM-FEAT-5	0.0332
F-A-2-LL-RAM-TEXT-IM-6	0.0205
F-A-1-LL-TIJAHPSQL-TEXT-7	0.0291
F-A-2-LL-RAM-IM-S	0.0000
M-A-2-LL-RAM-IM-FEAT-S	0.0003
M-A-2-LL-RAM-IM-S	0.0003

software is a time-consuming job in *any* non-standard environment or application.

When deploying an information retrieval system, it is the application developer's job to translate the user query (usually just some keywords), to operations on inverted files and ranking operations on the results. This is easy when the application envisaged can be handled by some standard software components. However, standard solutions are often not good enough. General purpose retrieval components, such as general purpose web search engines, do not provide sufficient functionality in many scenarios. For some time now, companies like Google have started to develop special purpose search solutions like Froogle (searching products) and Google Scholar (searching scientific articles). Today, the development of such specialized applications is the job of information retrieval specialists, but in the near future however, any software developer should be able to develop applications like this in pretty much the same way as he/she would currently develop office automation applications using relational database management systems: design a database schema; come up with SQL queries; make a nice user interface; done! We call these search engines of the future *parameterized search engines*. As future work, we will explore the possibilities for developing the *parameterized search engine* SPIEGLE: a search engine providing a high-level query language that supports many diverse search applications, as well as providing flexible ranking of search results. This years' TREC [16] and TRECVID experiments are a first step towards that goal.

References

[1] Wouter Alink. XIRAF – an XML-IR approach to digital forensics. Master's thesis, University of Twente, October 2005.

[2] P. A. Boncz. *Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications*. Ph.d. thesis, Universiteit van Amsterdam, Amsterdam, The Netherlands, May 2002.

[3] P. Buneman, L. Libkin, D. Suciu, V. Tannen, and L. Wong. Comprehension syntax. *SIGMOD Record*, 23(1):87–96, 1994.

[4] A. P. de Vries. The Mirror DBMS at TREC-9. In *Proceedings of the Nineth Text Retrieval Conference (TREC-9)*, pages 171–177, Gaithersburg, MD, USA, November 2000.

[5] J.M. Geusebroek. Distinctive and compact color featuress for object recognition. In *(submitted)*, 2005.

[6] J.M. Geusebroek, R. van den Boomgaard, A.W.M. Smeulders, and H. Geerts. Color invariance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(12):1338–1350, 2001.

[7] J.M. Geusebroek and A. W. M. Smeulders. A six-stimulus theory for stochastic texture. *International Journal of Computer Vision*, 62(1/2):7–16, 2005.

[8] Djoerd Hiemstra. A linguistically motivated probabilistic model of information retrieval. In Christos Nicolaou and Constantine Stephanidis, editors, *Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries (ECDL)*, volume 513 of *Lecture Notes in Computer Science*, pages 569–584. Springer-Verlag, 1998.

[9] Djoerd Hiemstra and Wessel Kraaij. Twenty-one at trec-7: ad-hoc and cross-language track. In *Proceedings of the seventh Text Retrieval Conference TREC-7*, pages 227–238, Gaithersburg, MD, USA, 1999.

[10] Djoerd Hiemstra and Vojkan Mihajlovic. A database approach to information retrieval: The remarkable relationship between language models and region models. Technical Report 05-35, Centre for Telematics and Information Technology, 2005.

[11] J. List, V.Mihajlovic, G.Ramírez, A.P. de Vries, D. Hiemstra, and H.E. Blok. Tijah: Embracing ir methods in xml database. *Information Retrieval*, 8(4):547 – 570, December 2005.

[12] Vojkan Mihajlovic, Henk Ernst Blok, Djoerd Hiemstra, and Peter M.G. Apers. Score region algebra: Building a transparent xml-ir database. In *Proceedings of the fourteenth International Conference on Information and Knowledge Management (CIKM)*, 2005.

[13] C. Petersohn. Fraunhofer HHI at trecvid 2004: Shot boundary detection system. In *TREC Video Retrieval Evaluation Online Proceedings, TRECVID*, 2004.

[14] J. M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, 1998.

[15] H. Rode and D. Hiemstra. Conceptual Language Models for Context-Aware Text Retrieval. In *Proceedings of the 13th Text REtrieval Conference Proceedings (TREC)*, 2005.

[16] Henning Rode, Georgina Ramírez, Thijs Westerveld, Djoerd Hiemstra, and Arjen P. de Vries. The lowlands' TREC experiments 2005. In *Proceedings of the fourteenth Text Retrieval Conference, TREC2005, Notebook papers*, 2005.

[17] C.G.M. Snoek, J.C. van Gemert, J.M. Geusebroek, B. Huurnink, D.C. Koelma, G.P. Nguyen, O. de Rooij, F.J. Seinstra, and A.W.M. Smeulders. The mediamill trecvid 2005 semantic video search engine. In *TREC Video Retrieval Evaluation, TRECVID, Notebook papers*, 2005.

[18] Andrew Trotman and Börkur Sigurbjörnsson. Narrowed extended xpath i (NEXI). In Norbert Fuhr, Mounia Lalmas, Saadia M alik, and Zoltan Szlavik, editors, *Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004, Dagstuhl Castle, Germany, December 6-8, 2004, Revised Selected Papers*, volume 3493. Springer-Verlag GmbH, may 2005. http://www.springeronline.com/3-540-26166-4.

[19] A. R. van Ballegooij, A. P. de Vries, and M. L. Kersten. RAM: Array processing over a relational DBMS. Technical Report INS-R0301, CWI, Amsterdam, The Netherlands, March 2003.

[20] Nuno Vasconcelos. *Bayesian Models for Visual Information Retrieval*. PhD thesis, Massachusetts Institut of Technology, 2000.

[21] T. Westerveld. *Using generative probabilistic models for multimedia retrieval*. Ph.d. thesis, University of Twente, Enschede, The Netherlands, November 2004.

[22] Thijs Westerveld and Arjen P. de Vries. Generative probabilistic models for multimedia retrieval: query generation versus document generation. *IEE Proceedings - Vision, Image and Signal Processing*, 152(6):852–858, 2005.

[23] M. Zukowski, P. A. Boncz, N. Nes, and S. Héman. MonetDB/X100 - A DBMS In The CPU Cache. *IEEE Data Engineering Bulletin*, 28(2):17–22, ? 2005.