Imperial College and Johns Hopkins University at TRECVID

Arnab Ghoshal[+], Sanjeev Khudanpur[+], João Magalhães[*], Simon Overell[*], Stefan Rüger[*] and Alexei Yavlinsky[*]

{ag, khudanpur}@jhu.edu,

{j.magalhaes, simon.overell, s.rueger, alexei.yavlinsky}@imperial.ac.uk

[+] CLSP/ Electrical and Computer Engineering
Johns Hopkins University
Baltimore, MD 21218, USA

[*] Department of Computing
Imperial College London, South Kensington Campus
London SW7 2AZ, UK

Abstract

We describe our experiments for the high-level feature extraction and search tasks. For the search task, we tested the system we have used in previous years, which encapsulates content based image search, image browsing, automated image annotation and named entity extraction. For the feature task we apply the nonparametric density estimation model and the HMM-based concept specific image model.

1 Introduction

The retrieval system we use for the search task is functionally similar to that used for TRECVID 2004 (Heesch et al., 2004). The search process is based on query by example using global and tiled image features, image browsing with the NN^k network, automated image annotation and ASR indexing with named-entity extraction.

In the high-level feature detection task we apply the nonparametric density estimation model and the HMM-based concept specific image model. Section 2 and 3 describe, respectively, the experiments and results for the search task, and the high-level feature detection task.

2 Search Task

2.1 Graphical user interface

We have consolidated the paradigms of text-based search, content-based search, NN^k image browsing and temporal browsing into a unified interface, uBase. By providing tight integration of techniques and a rich set of user interactions, we aimed to equip the user with substantial navigational power.

Figure 1 shows the interface layout. The uBase front end is split into two panels. The main browsing panel takes up the top 80% of the screen, it has four tabs that the user can move between: *Image & Feature Search*, *Content Viewer*, *Search Basket* and *NN^k image browsing*.

The *Image & Feature Search* tab is where the user formulates their query. They can specify a free text string, named entities, visual concepts obtained using automated image annotaiton, whether the key-frame should contain a face and the image feature weights. The named entities and image annotations are selected from drop down lists; whether the key-frame should contain a face is selected with a radio button and the image feature weights with sliders. The relative importance of the free text string, named entitie and image annotations can also be specified with sliders.

- The majority of the *Image & Feature Search* tab is taken up by the browsing area, this is where the search results are displayed as image thumbnails. Clicking on a thumbnail will add the key-frame to the search basket; all the key-frames currently in the search basket are displayed with a yellow border around them. Each image also has super-imposed three mini-buttons, *add to query-by-example*, *go to temporal tab* and *go to the NN^k tab*. The search results are paginated, to browse through the pages of results three browsing buttons are provided: next page; previous page and back to beginning.
- The *Content Viewer tab* is split in half, on the left hand side metadata is displayed, on the right a full size image is displayed. The key-frame displayed in the Content Viewer is the last key-frame that was clicked on in either of the other three main browsing panel tabs.
- The *Search Basket* is where all the currently selected key-frames are displayed. The Search Basket contains tools to reorder the key-frames and remove them swiftly.
- The *NN^k browinsg panel* displays the thirty key-frames nearest to the selected key-frame in feature space (dependant on content-based features).

As with the Image & Feature Search, clicking on an image will select it and the same three mini-buttons are super-imposed.

Below the main browsing panel is the secondary browsing panel. This is also a tabbed panel however images in this panel are displayed in a horizontal line. The tabs making up the secondary browsing panel are *Query by Example*, *Query History*, *Temporal* and *NN^k Browsing History*.

- *Query by Example* tab displays all the images used to query by example.
- *Query History* tab displays an image representing each query made. The image is taken from the query by example panel and has the parameters of the query super-imposed. Clicking on the query will repeat it and take the user to the Image & Feature search tab where they can view the results.
- *Temporal* tab displays the key-frames appearing either side of the selected key-frame chronologically.
- *NN^k Browsing History* tab is similar to the query history. An image is displayed representing every NN^k neighbour search made and clicking one will return the user to the NN^k browsing tab.

When the user has finished selecting relevant images, they can fill in the tail-end of their search results by selecting the query from the query list they believe to have they greatest number of relevant key-frames as yet un-selected. By selecting the query and hitting the submit button, all their selected results and selected "best query" will be sent to the server.

The uBase front end is written in Dynamic XHTML and JavaScript, it is viewed with a standard web-browser. Communication is performed with the backend using GET & POST requests via AJAX. A strict model, view, controller architecture was adopted to minimise coupling.

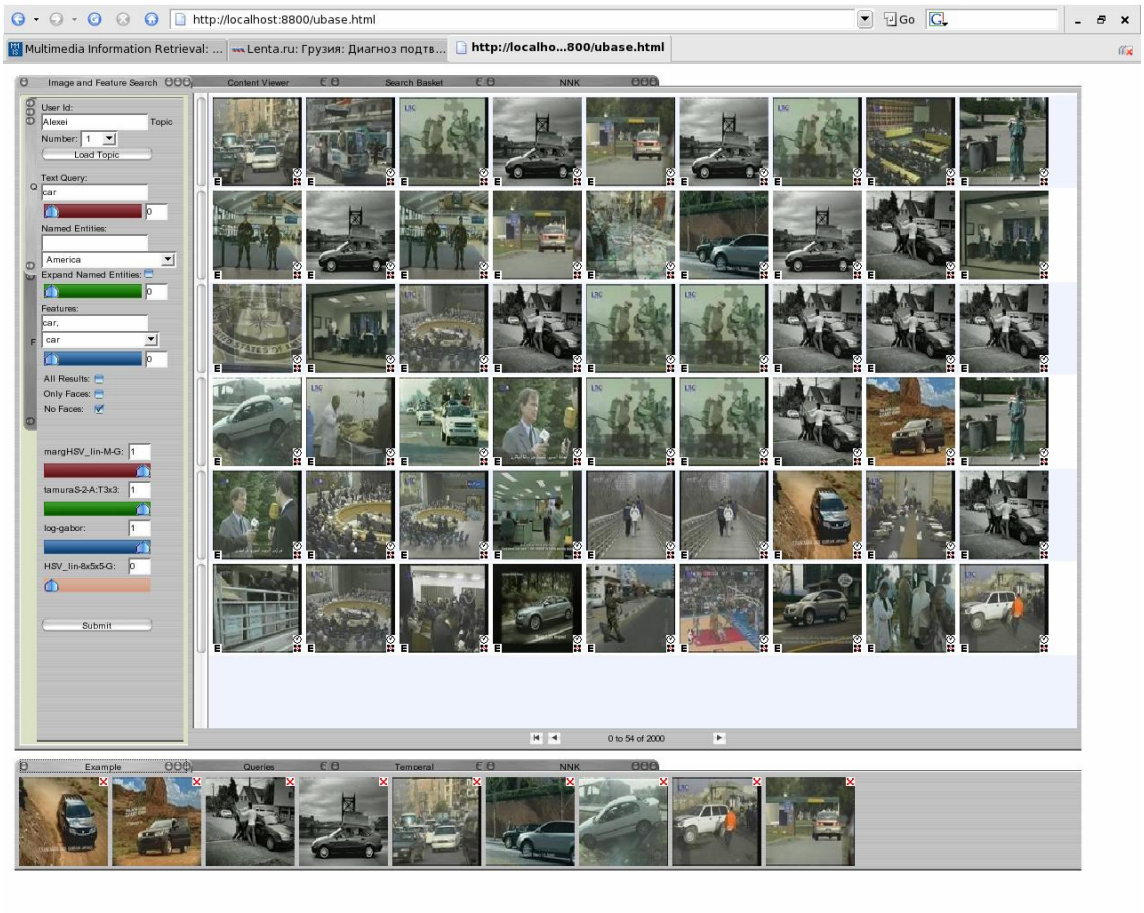


Figure 1 – uBase front end

2.2 Low-level features and content based image search

For the search task we used 4 low-level texture and colour features as well as the text from the LIMSI transcripts. Like last year, we cut of the bottom of each image as this often contained a news line. To capture local image information, we partition the images into tiles and obtain features from each tile. The final feature vector consists of a concatenation of the feature vectors for individual tiles. For features we used last year's TRECVID, see (Heesch et al., 2004).

Tamura features - We compute the three Tamura features coarseness, contrast and directionality as proposed in (Tamura, Mori, & Yamawaki, 1978) for each of 9x9 = 81 tiles. For each pixel we compute the values of each of the three features and compute for each tile a 3D histogram. See (Howarth & Rüger, 2004) for additional details and evaluation results.

Gabor filter - One of the most popular signal processing based approaches for texture feature extraction is the use of Gabor filters. These enable filtering in the frequency and spatial domain. A range of filters at different scales and orientations allows multichannel filtering of an image to extract frequency and orientation information. This can then be used to decompose the image into texture features. Our implementation of the Gabor filter is based on (Manjunath & Ma, 1996). To

each image we apply a bank of 6 orientation and 4 scale sensitive filters that map each image point to a point in the frequency domain.

The feature consists of the mean and standard deviation of the total response of each filter across the image..

Marginal HSV colour moments – For this descriptor, we formed individual histograms for each of the three colour channels and computed the mean and second central moment of each marginal colour histogram.

HSV global colour histogram – This descriptor is a 3D histogram that captures the global colour distribution of each image. It is quantised such that there are 8 bins for hue and 5 each for value and saturation. The lowest value bin is not partitioned into hues since they are not easy for people to distinguish.

Keyframe indexing for content-based search and NN^k browsing was performed in the same manner as described in (Heesch et al., 2004).

2.3 Concept search

The keyframes were automatically annotated with respect to the 39 concepts that were part of this year's feature task evaluation. Automatic annotation was performed using the model in (Yavlinsky, Schofield, & Rüger, 2005).

2.4 Text processing

The ASR data was indexed using the Lucene text search engine, with named entities extracted using GATE.

2.4.1 Named entity extraction

News articles have a large number of references to named entities that quickly place the reader into the context of the news piece. Sometimes the same named entity is referred to in different ways (e.g. "British prime minister", "Mr. Blair", "Tony Blair"). Thus, the detection of references to all named entities is the problem that we addressed in this part of the system.

Named entity recognition systems rely on lexicons and textual patterns either manually crafted or learned from a training set of documents. We used the GATE named entity recognition system proposed in (Cunningham, 2004).

2.4.2 Modelling co-occurrences of named entities

The extracted named entities were mined for association rules using the AprioriTid algorithm described in (Agrawal & Srikant, 1994). The rules mined were of the form $A \rightarrow B$ where A and B are non-empty sets of named entities. Rules were saved with a confidence of greater than *50%* and a support of 20 documents or more.

When the server application first starts all the association rules are read into a hash-table. Every time the user queries named entities they can request for them to be expanded. If the user's queries contain all the entities listed in the A part of a rule, the B part of the corresponding rule will be added to their query.

2.4.3 Text indexing

The text corpus was indexed with Lucene 2.0[1], which were later used to search the articles corpus. The information retrieval model we used was the vector space model without terms frequencies (binary term weight). This decision is due to the small size of each document that could cause a large bias for some terms. Terms are extracted from news corpus in the following way:

1. Splits words at punctuation characters, removing punctuation, however, a dot that's not followed by whitespace is considered part of a term;

2. Splits words at hyphens and generates a term: unless there's a number in the term, in which case the whole term is interpreted as a product number and is not split;

3. Recognizes email addresses and internet hostnames as one term;

4. Removes every stop words;

5. Index a document by its extract terms (lowercase).

2.4.4 Text search

Once the news articles are indexed with Lucene, the query terms will be extracted in the same way that the documents' terms were, and a similarity measure is taken between the query's terms and all indexed documents. The similarity function is given by the following expression:

$$\text{score}(q,d) = \frac{\sum_{t \in q} tf(t \in d) \cdot idf^2(d \subset t, D) \cdot norm(d)}{\sqrt{\sum_{t \in q} tf(t \in d)^2}},$$

where $tf(t \in d)$ is the t term frequency for the given document d (in our case is 0 or 1), $idf(d \subset t, D)$ is the frequency of documents d containing the term t in the D collection, and $norm(d)$ is a normalization constant given by the total number of terms in document d.

2.5 Application backend

The application backend was developed within a JavaServerPages framework hosted on an Apache Tomcat server. The server accepted requests from the web-based front end described in section 2.1 and returned results in XML format together with the image thumbnails.

2.6 Experiments and results

In the search task we conducted the following experiments:

[1] Apache Lucene - http://lucene.apache.org/

Interactive runs: One interactive run were carried out. In this run, the users were invited to use content-based search.

Block design: With a total of four users, 24 topics and one system variant, we chose an experimental design that required each user to execute a total of 6 querie\s.

Our search results obtained the mean average precision 0.125, with each query result lying close to its respective topic median, as shown in Figure 2.

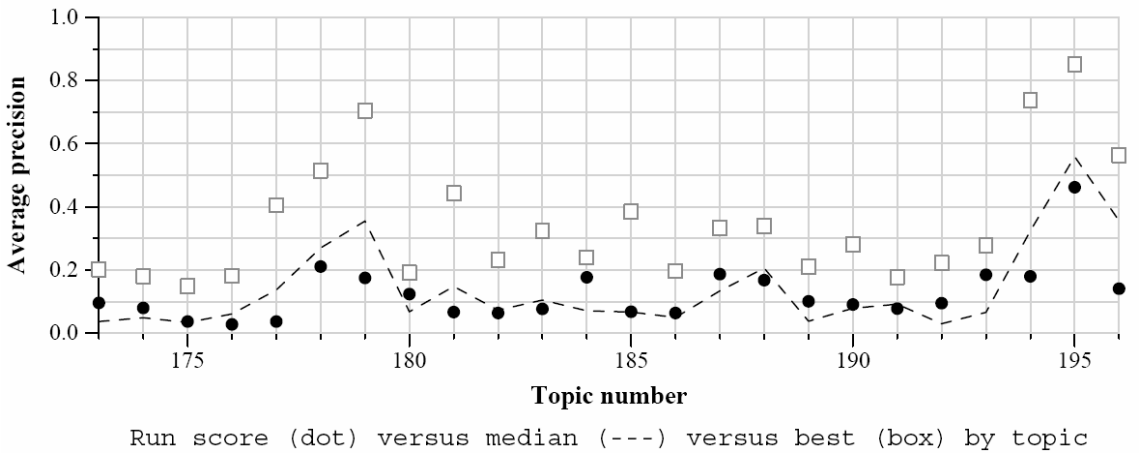
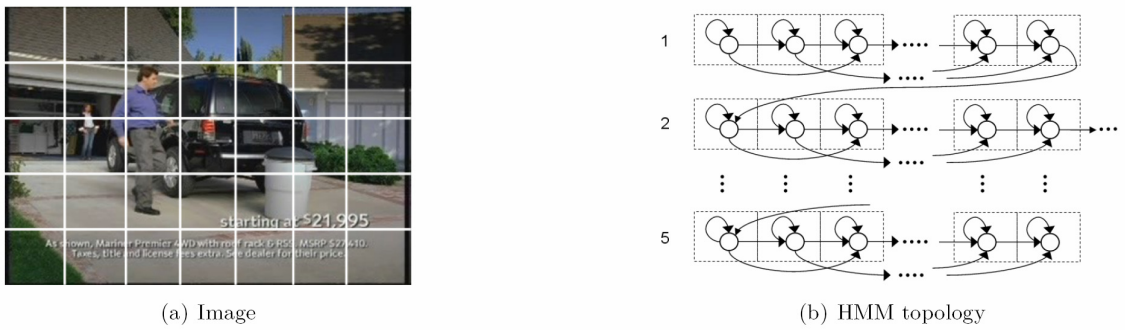


Figure 2. Search task results for the interactive run

3 High-level Feature Detection Task

3.1 Concept Specific Image Models



(a) Image (b) HMM topology

Figure 3. HMM Topology for the Concept-Specific Models.

Image retrieval for a text query c is essentially the task of deciding, for each image I, whether or not it contains the concept c. This motivates the following concept-specific model, in which one pair of HMMs is estimated for each concept: an HMM $\mathcal{H}_c^+$ from all images that contain the concept c, and an $\mathcal{H}_c^-$ from all images that do not contain the concept. Both $\mathcal{H}_c^+$ and $\mathcal{H}_c^-$ have simple left-to-right topologies, with as many states as there are rectangular image regions, as shown in Figure 3.

The parameters $f_{\mathcal{H}_c^+}(\cdot|s)$ and $p_{\mathcal{H}_c^+}(s'|s)$ of each $\mathcal{H}_c^+$ are chosen to maximize the likelihood of images containing c:

$$\mathcal{H}_c^+ = \arg\max_{\mathcal{H}} \prod_{I_l : c \in C_l} f(I_l|\mathcal{H}),$$

where $\mathcal{L} \equiv \{(I_1, C_1), \ldots, (I_L, C_L)\}$ denotes the training set. Similarly the parameters of each $\mathcal{H}_c^-$ are estimated to maximize the likelihood of images not containing c.

For an unlabeled image collection $\{I\}$, we calculate the likelihoods $f(I|\mathcal{H}_c^+)$ and $f(I|\mathcal{H}_c^-)$ for each concept c, and then rank-order the images according to the likelihood ratio

$$\texttt{score}(I, c) = \frac{f(I|\mathcal{H}_c^+)}{f(I|\mathcal{H}_c^-)} = \frac{f(x_1^T|\mathcal{H}_c^+, s_0)}{f(x_1^T|\mathcal{H}_c^-, s_0)}.$$

In practice, we have found that the Viterbi approximation

$$\texttt{score}(I, c) \approx \frac{\max_{s_1^T} \prod_{t=1}^T f_{\mathcal{H}_c^+}(x_t|s_t)\, p_{\mathcal{H}_c^+}(s_t|s_{t-1})}{\max_{s_1^T} \prod_{t=1}^T f_{\mathcal{H}_c^-}(x_t|s_t)\, p_{\mathcal{H}_c^-}(s_t|s_{t-1})}$$

results in nearly identical retrieval performance.

3.1.1 Adapting to individual video sources

We reported in (Ghoshal & Khudanpur, 2006) that supervised adaptation of our models to individual video sources gives significant improvements over using source-independent (SI) models. A Bayesian approach to source adaptation is to consider the source-dependent (SD) mean vectors $\mu_{m,s}^{(k)}$ for a source k to themselves be random vectors and, given some source-specific training data, to compute the values of the SD means with the maximum *a posteriori* probability (MAP) (Lee, Lin, & Jung, 1991) . In particular, if we assume *a priori* that $\mu_{m,s}^{(k)}$ is Gaussian with mean equal to its SI value $\hat{\mu}_{m,s}$, and variance τ^{-2}, then the MAP estimate of $\mu_{m,s}^{(k)}$ given some source-specific data is

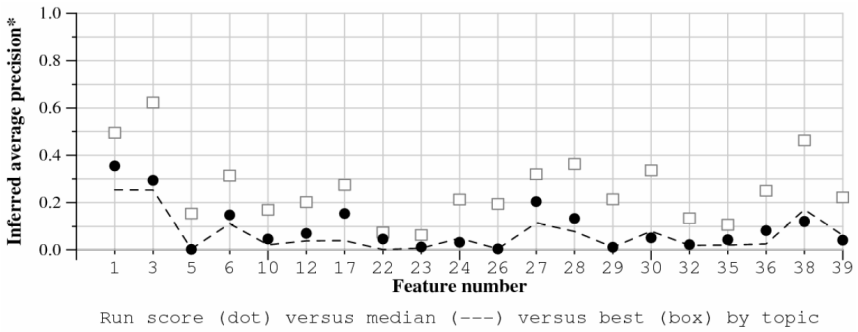
$$\hat{\mu}_{m,s}^{(k)} = \frac{\sum_{t=1}^T \gamma_{m,s}(t) x_t^{(k)} + \tau \hat{\mu}_{m,s}}{\sum_{t=1}^T \gamma_{m,s}(t) + \tau},$$

where $\gamma_{m,s}(t)$ is the posterior probability under the SI HMM that $x_t^{(k)}$, the t-th image feature of video-source k, was "emitted" by mixture component m of state s. Setting $\tau = 0$ results in maximum likelihood estimation of the SD means from only the source-specific data, which may result in over-fitting. τ is chosen empirically to temper this effect (Gauvain & Lee, 1994).

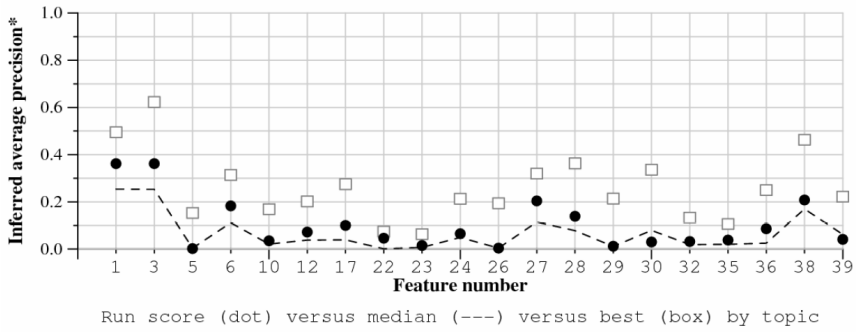
3.1.2 Experiments and results

Since this year's search collection includes video sources that were not present in the training data, we expect degraded performance on data from the unseen sources. One can think of source adaptation being one way of tackling this problem. However, initial experiments with *unsupervised*

source adaptation have met with limited success. We submitted a source independent (SI) system as our baseline, and compare it with a system where the models were adapted to the sources seen in the training data. It is unclear whether some kind of score normalization is required to combine ranked lists from SD (for sources seen in the training data) and SI (for the new sources) systems. We took the simple approach of merging the ranked lists and sorting by their raw scores. As seen from the results in Figure 4 the source adapted system performs better than the SI system. We are currently investigating methods to make the systems more robust to data from unseen sources.



(a) Source independent models (MAP=0.093)



(b) Source adapted models (MAP=0.102)

Figure 4. TRECVID 2006 results for HMM-based models.

3.2 Nonparametric Density Estimation

We evaluated an approach to automated image annotation described in (Yavlinsky, Schofield, & Rüger, 2005) on this year's TRECVID feature detection task, where each high-level feature is treated as a keyword. We describe this approach briefly below. Denote a keyword as w and a keyframe image feature vector as x. The posterior probability of w given x can then be computed as

$$p(w \mid x) = \frac{f(x \mid w)p(w)}{f(x)}.$$

The density in the feature space conditional upon the assignment of a particular keyword, $f(x|w)$, is estimated using a nonparametric technique known as 'kernel smoothing'. Where x is a vector $(x_1,\ldots, x_d)$ of real-valued image features, we define the kernel estimate of $f(x|w)$ as

$$\hat{f}(x \mid w) = \frac{1}{|T_w|} \sum_{x^{(i)} \in T_w}^{n} k\left(x - x^{(i)}; h\right),$$

where $x^{(1)}...x^{(n)}$ is the sample of images with label w in the training set T_w, where k is a kernel function that we place over each point $x^{(i)}$. We used the following kernel in our experiments:

$$k(t,h) = \frac{1}{h^d} \prod_{l=1}^{d} e^{-\frac{|t_l|}{h_l}},$$

where $t = x - x^{(i)}$. We set each bandwidth parameter h_l by scaling the sample standard deviation of feature l by the same constant λ, computed previously using a cross-validation approach on a different dataset. The prior is defined as

$$p(w) = \frac{1}{|W|},$$

where $|W|$ is the size of the vocabulary. We use a uniform keyword prior because we do not assume that the relative frequency of keywords in the training set is indicative of the true probability with which those keywords would be observed. Finally, we make the approximation $f(x) = \sum_w f(x \mid w)p(w)$ for simplicity. For a given keyword the images are ranked according their posterior probabilities of that keyword.

3.2.1 Experiments and results

Owing to a software bug, the nonparametric model achieved performance far lower than last year (0.005 on 39 concepts compared to. 0.218 in 2005 on 10 concepts). We look forward to re-evaluating this approach using the ground truth relevance judgments provided by NIST to establish the fair performance of this approach.

4 Conclusion

Our search approach, developed in previous years, continues to perform reasonably well. The performance of Nonparametric Density Estimation feature detection method remains to be established on this year's test data, whilst the HMM-based concept-specific image models achieved competitive performance.

5 References

Agrawal, R & Srikant, S. (1994). *Fast algorithms for mining association rules.* Int'l Conference on Very Large Databases.

Cunningham, H. (2004). GATE, a General Architecture for Text Engineering. *Computers and the Humanities, 36*(2), 223-254.

Gauvain, J-L., Lee, C-H. (1994, April 1994). Maximum a Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains. *IEEE Trans. on Speech and Audio Processing, 2(2)*, 291–298

Ghoshal, A., Khudanpur ,S. (2006). Source Adaptation for Improved Content-Based Video Retrieval. *IEEE ICASSP*.

Heesch, D., Howarth, P., Magalhães, J., May, A., Pickering, M., Yavlinsky, A., et al. (2004, November 2004). *Video retrieval using search and browsing.* TREC Video Retrieval Evaluation Workshop, Gaithersburg, MD, USA.

Howarth, P., & Rüger, S. (2004, July 2004). *Evaluation of texture features for content-based image retrieval.* Int'l Conference on Image and Video Retrieval, Dublin, Ireland.

Lee, C-H., Lin, C-H., Juang, B-H. (1991, April 1991). A Study on Speaker Adaptation of the Parameters of Continuous Density Hidden Markov Models. *IEEE Trans. on Signal Processing, 39(4),* 806–814

Manjunath, B., & Ma, W. (1996). Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 18*(8), 837-842.

Tamura, H., Mori, S., & Yamawaki, T. (1978). Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man and Cybernetics, 8*(6), 460-472.

Yavlinsky, A., Schofield, E., & Rüger, S. (2005, July 2005). *Automated image annotation using global features and robust nonparametric density estimation.* Int'l Conference on Image and Video Retrieval, Singapore.