Learning Query-Class Dependent Weights in Automatic Video Retrieval

Rong Yan
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, 15213
yanrong@cs.cmu.edu

Jun Yang
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, 15213
juny@cs.cmu.edu

Alexander G. Hauptmann
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, 15213
alex+@cs.cmu.edu

ABSTRACT

Combining retrieval results from multiple modalities plays a crucial role for video retrieval systems, especially for automatic video retrieval systems without any user feedback and query expansion. However, most of current systems only utilize query independent combination or rely on explicit user weighting. In this work, we propose using query-class dependent weights within a hierarchial mixture-of-expert framework to combine multiple retrieval results. We first classify each user query into one of the four predefined categories and then aggregate the retrieval results with query-class associated weights, which can be learned from the development data efficiently and generalized to the unseen queries easily. Our experimental results demonstrate that the performance with query-class dependent weights can considerably surpass that with the query independent weights.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval models

General Terms

Algorithms, Performance, Theory

Keywords

Video Retrieval, Query Class, Modality Fusion, Learning

1. INTRODUCTION

The task of video retrieval is to search a large amount of video for clips relevant to an information need expressed in form of multimodal queries. The queries may consist merely of text or also contain images, audio or video clips that must be matched against the video clips in the collection. Specifically this paper focuses on the queries that attempt to find semantic contents such as specific people, objects and events in a broadcast news video collection. As indicated in Figure 1, our automatic video retrieval system needs to go through

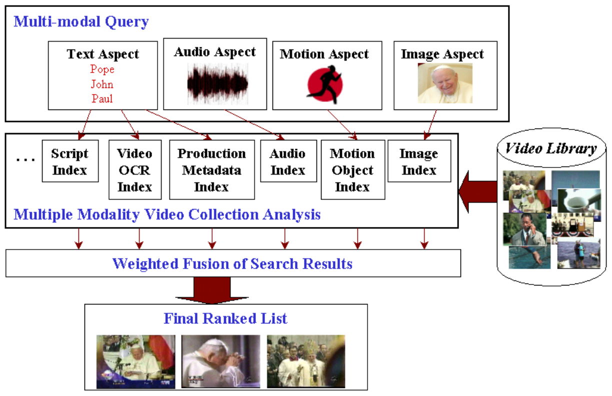


Figure 1: Overview of our video retrieval system

the following steps to find relevant clips for content-based queries without any user feedback and manual query expansion. First, various sets of index features are extracted from the video library through analysis of multimedia sources. For each video clip, different search modules are used to generate a vector of retrieval scores, indicating the similarity of this clip to a specific aspect of the query. Finally, these individual retrieval results are fused via an aggregation algorithm to produce an overall ranked list of video clips.

Inspired by the conventional probabilistic retrieval model, we frame the retrieval task using a mixture-of-expert architecture, where each expert is responsible for computing the similarity scores on some modality and the outputs of multiple retrieval experts are combined with their associated weights. Tailoring this architecture specifically for video retrieval, we adopt a re-ranking model where text features provide the primary evidence for locating relevant video content, while other features offer complementary clues to further refine the results.

However, given the large number of candidate retrieval experts available, the problem of selecting most effective experts and learning the optimal combination weights naturally follows. Most of current systems only utilize query independent combination strategies (namely uniform/fixed weights) or rely on explicit user weighting, which are either inflexible or unrealistic. Previous experiments [20] showed that with respect to the best query independent weights, the retrieval performance of standard TREC Video Retrieval Evaluation Task(TRECVID)[18] queries benefited from using query dependent weights for combination, even though they are trained from the imprecise pseudo relevance feedbacks. It also implied that once the training data are of higher accuracy, the performance can be further improved.

However, given the virtually infinite number of queries, it is impractical to learn weights on a per query basis. A trade-off needs to be found between the difficulty of providing training data and the ability of capturing the idiosyncrasy of each query. To this end, we can consider associating weights with a few pre-defined classes which consist of queries with similar characteristics. Hopefully we can achieve better retrieval performance by fusing multiple retrieval results with query-class specific weights. In this case, it is also legitimate to collect truth data for each query class because the number of classes is very limited, while the learned weights can be reused for other unseen queries as long as they belong to some of the predefined classes.

Therefore, we propose an automatic video retrieval approach which uses query-class dependent weights to combine multi-modality retrieval results. The user query is first classified into one of the four predefined query classes, i.e. finding named persons, named objects, general objects and scenes. Once the query is categorized, the retrieval scores from multiple modalities can be combined into a final probabilistic output with query-class associated weights, which are learned from the training data off-line. Our experiments confirm the effectiveness of the proposed approach using a video test collection of over 65 hours from the TREC video retrieval track [18].

1.1 Related Work

Designing the combination approaches for multiple modalities is of great importance to develop video retrieval systems. Westerveld et al[19] demonstrates how a combination of different models/modalities can affect the performance of video retrieval. They adopt a generative model inspired by language modeling approach and a probabilistic approach for image retrieval to rank the video shots. Final results are obtained by sorting the joint probabilities of both modalities. The video retrieval system proposed by Amir et al[1] performs a query-independent linear combination on both text/image retrieval systems, where the per-modality weights are chosen to maximize mean average precision score on development data. Gaughan et al[8] ranks the video shots based on the summation of feature scores and automatic speech retrieval scores, where the influence of speech retrieval is at 4 times that of any other features. Benitez et al[2] proposed content-based meta-search image search engine, called Metaseek. It assigns the new query images to one of the predefined clusters and selects one of the target image search engines based on their previous success of handling the similar queries. However, most of these systems either simply apply a query-independent combination approach, or rely on user feedback to decide the weights of multiple experts. It is desired to design a better fusion approach which takes query characteristics into account without explicit user feedback.

Query classification has been widely investigated in the community of information retrieval and query answering. Li et al[13] and VideoQA[21] adopt a hierarchial classification approach to categorize free-form factual queries. Five types of machine learning approaches are experimented by Dell et al [23] for automatic question classification task. Kang et al[12] classify the user queries into three categories, that is, the topic relevance task, the homepage finding task and the service finding task using various statistics from query words. Different linear weights of text information and hyperlink information will be assigned based on the query categories to improve the web document retrieval. The similar idea can be naturally extended to the context of video retrieval.

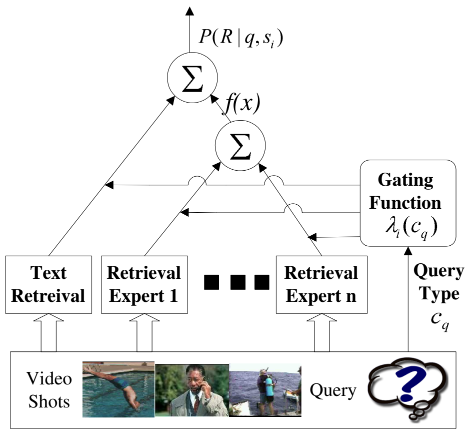


Figure 2: The two-level hierarchial mixture-of-expert architecture for video retrieval

2. A HIERARCHIAL MIXTURE-OF-EXPERT ARCHITECTURE

In the following discussion, we consider the task of shot-based video retrieval which is to find relevant video shots satisfying users' information need and a shot is defined as a sequence of video frames from a single camera operation. Conventional probabilistic retrieval models [17] rank documents by the probability that each document would be judged relevant to a given query. Following this model, the shot-based video retrieval task can be thought of as a supervised learning problem which aims at estimating the probability $P(R|q, s_i)$ where R is the relevance of query q to shots s_i. Typically, the video features can be constructed from multiple modalities. In broadcast news video archives, the candidate features we used include low-level ones such as color, texture, and edge features of video frames, motion energy, audio features, as well as high-level ones such as video transcript, faces, anchor, commercials, etc.

Multiple specific retrieval experts are usually developed to handle various features. Therefore, our retrieval system can be naturally formulated into a mixture-of-expert architecture [11]. The mixture of expert handles the retrieval task by first dividing the input space, i.e. query and document features, into a set of regions and accordingly fitting specific decisions to the data in these regions. The regions have "soft" boundaries which allow data fall simultaneously in multiple regions, and the weights of combining experts are adjusted by learning algorithm. This framework can be viewed as a single layer tree, or a single-layer mixture of expert(SME). Each expert at the leaves of the tree produces the probabilities of relevance $P_k(R|q, s_i)$ from one modality. The gating function at the nonterminals of the tree produces scalar outputs as the linear weights for each expert and blends the probabilistic outputs of experts using a generalized linear function

$$P(R|q, s_i) = f\left(\sum_k \lambda_k P_k(R|q, s_i)\right), \tag{1}$$

where λ_k is the linear weights. $f(\cdot)$ is generally chosen to be the identity function or the logistic function.

We need to design the gating function based on the query q and the video shot s_i. In this work, the gating function is a function of the query q, more specifically, it is determined by the category or class of the query. Under this assumption, Eq(1) can be rewritten as,

$$P(R|q,s_i) = f\left(\sum_k \lambda_k(c_q)P_k(R|q,s_i)\right), \tag{2}$$

where c_q are the query class of q as defined in the Section 3. It indicates that the same type of queries will share an identical set of weights λ_k. Some feature selection algorithm can be applied to pick out the most representative experts for each type of queries. For example, the face recognition expert can be selected for finding-person-type queries but the outdoors detector is not consistently helpful for them. As a future work, the gating function can be also related to the video source of s_i, which allows the weight variation when the video source changes.

Although considered a supervised learning problem which simply fuses results from multiple experts, video retrieval has its distinct properties. First, based on evidence from the best-performing video retrieval systems in the TRECVID 2001–2003, text features are demonstrated to be the most reliable features for selecting semantically relevant shots in video retrieval. However, many complementary sources from various video modalities can be used to rerank the text retrieval outputs. These sources include audio features, motion vectors, visual features (e.g. color and edge histograms), and pre-defined high-level semantic features (e.g. a face detector and an anchor detector). Therefore, we adopt a reranking model for combining the output of experts, i.e., the overall ranking is provided by

$$P(R|q,s_i) = \lambda_1^u(c_q)P_1^u(R|q,s_i) + \lambda_2^u(c_q)P_2^u(R|q,s_i) \tag{3}$$

where the subscript u means the upper level mixture, $\lambda_1^u(c_q) + \lambda_2^u(c_q) = 1$, $P_1^u(\cdot)$ is the retrieval scores generated by text retrieval and $P_2^u(\cdot)$ is a generalized linear function of other expert outputs $P_k^l(\cdot)$ on the lower level,

$$P_2^u(R|q,s_i) = f\left(\sum_{k=1}^{m} \lambda_k^l(c_q)P_k^l(R|q,s_i)\right) \tag{4}$$

The architecture of our design is shown in Figure 2, which can be viewed as a two-level hierarchial mixture of experts(HME) [11]. λ_1^u is typically close to 1 because text retrieval usually provides most reliable results. This property allows text retrieval to dominate the retrieval results while the other weaker experts re-rank and adjust the output.

3. QUERY CLASSIFICATION

3.1 Defining query classes

The query classes are defined with two guidelines. First, using the class specific combination should outperform using the query-independent combination. This requires queries with similar characteristics to be grouped into the same class, and vice versa. Second, queries can be automatically classified into classes with reasonable accuracy, which is essential to apply this idea in real-word systems. In the following discussion, we mainly consider TRECVID queries as a standard set of queries with rich text descriptions, although our method is generally useful for queries with a small set of keywords. Some examples used by TRECVID can be seen in Figure 3, which are in the form of short imperative sentences like "Find shots of Yasser Arafat". After inspecting these queries, we define the following four query classes according to intent of the queries:

1. Find shots of Yasser Arafat.
2. Find shots of a rocket or missile taking off.
3. Find shots of the Tomb of the Unknown Soldier at Arlington National Cemetery.
4. Find shots of the front of the White House in the daytime with the fountain running.

Figure 3: Text Query Examples from TRECVID

Named person (P-query) queries for finding a named person, possibly with certain actions, e.g., "Find shots of Yasser Arafat" and "Find shots of Ronald Reagan speaking".

Named object (E-query) queries for a specific object with a unique name, which distinguishes this object from other objects of the same type. For example, "Find shots of the Statue of Liberty" and "Find shots of Mercedes logo" are such queries.

General object (O-query) queries for a certain type of objects, such as "Find shots of snow-covered mountain" and "Find shots of one or more cats". They refer to a general category of objects instead of a specific one among them, though they may be qualified by adjectives or other words.

Scene (S-query) queries depicting a scene with multiple types of objects in certain spatial relationships, e.g., "Find shots of roads with lots of vehicles" and "Find shots of people spending leisure on the beach". They differ from O-queries mainly by the number of the types of object involved.

Here the definition of query classes is different from the definitions of question categories in the TREC questions answering track, such as searching for location, numeric number and description[13], because our purpose is to improve the fusion of multiple retrieval results instead of extracting exact answers from text archives. Intuitively, each query class by our definition favors a specific set of features. For example, face presence, size, position information and face recognition are critical to P-query but of little value to other query classes. For both P-query and E-query, transcript is particularly important since such queries are more likely to have perfect match in transcript, and so is video OCR since proper names may appear on screen as overlaid text. On the other hand, visual features like color, texture, and shape can be helpful to the O-query and S-query. Overall, such classification captures query characteristics regarding feature effectiveness, and is therefore promising for better performance.

We design the query classes above mainly for news video collections, which may not be applicable for arbitrary video retrieval systems. As an example, movie archives may not have as many P-queries as news video archives, but they may instead have more S-queries. These systems might have different or even more query classes. But the idea of query-class specific retrieval is generally applicable. Moreover, due to the lack of effective features for depicting various motions, this classification does not address the action of the target being searched. As the result, queries like "Bill Clinton" and "Bill Clinton walking out of the oval office" are in the same class and treated using the same set of features, though the latter favors motion feature.

3.2 Query Classification by Text Processing

As can be found in Figure 3, the queries from TRECVID are generally in a regular form and primarily factual queries, but the size of existing query pool is relatively small so far. Therefore query classifiers based on a data-driven statistical inference approach might not be a good choice. Alternatively, we use the text processing techniques, which consists of three phases as named entity extraction, tagging and chunking, and syntactic parsing.

Named entity extraction is mainly used to identify queries for named persons and objects, which contain proper names like people, location, or organization. Our named entity extraction method is similar to that used by MITRE [14] and BBN [3]. First, every word in a corpus of broadcast news transcript is manually labeled with a named-entity tag like *-person-*, *-location-*, *-organization-*, *-time-*, and *-none-*, resulting in a sequence alternating between words and tags, such as "*-person-* Barry, *-person-* Serafin, *-organization-* ABC, *-organization-* news, *-none-* in, *-location-* Washington". A trigram model is then trained from this tag-word sequence using statistical language modeling toolkit [5]. To detect named entities from unlabeled text, we build a Hidden Markov Model (HMM) that models each type of named-entity tag as a state and words as the observations of each state. The transition and emission probabilities of the HMM are provided by the trigram model. Viterbi algorithm is used to find the most likely sequence of states (or named-entity tags) that generates the observed sequence of words in the given text. After extracting the named entities from queries, we classify those containing people names as P queries and those containing organization and/or location names as E queries. The queries with both people and organization/location names are also put into the P class.

After the P and E-queries are recognized in the previous stage, some preliminary text analysis including POS-tagging and NP-chunking is conducted on the remaining queries to distinguish O-queries and S-queries. This starts with applying Brill's transformation-based part-of-speech (POS) tagger [4] on each query to obtain the POS of each word in the query. The tagged query is then fed into a text chunker, namely baseNP chunker [15], which divides the query into segments such as noun phrases (NP) and verb phrases (VP). Since one major distinction between queries for general objects and for scenes is the number of (different) objects, we label the queries with one longest-matched NP as O-queries, and those with multiple longest-matched NPs as S-queries. It is intuitive to count the number of objects by the number of NPs in the query, as objects are normally referred by noun phrases. We only count the longest-matched NPs because NPs can be nested. For example, the NP "a pink flower" recursively contains another NP "flower", both of which refer to the same object. Thus, the number of the longest NPs better approximates the number of objects a user intends to specify.

Although the method described above works reasonably well, it has an obvious drawback: a single longest NP may refer to multiple objects, such as in the S-query "a person diving into water" and "balloon in the sky", which our method classify as O-queries by mistake. Syntactic parsing is applied to correct the misclassified S-queries. Specifically, we send all the queries classified as O-queries, namely those with only longest NP, to Link Grammar Parser[9], which produces the syntactic structure of the queries. In this way, the internal structure of the longest NP can be seen. For example, "a person diving into water" is parsed as NP (NP (a person) VP (diving PP (into (NP water)))), where PP denotes prepositional phrase. The queries with a single NP which actually contains multiple NPs inside are re-assigned to S-queries, while the rest remain as O-queries. The only exception is queries with NP PP (prep NP) structure but having "of" as its preposition, such as "a cup of coffee", which actually refers to only one object since the NP after "of" is normally used to modify the NP before it. Hence, queries with such structure are classified as O-queries. This query classification method based on superficial text analysis is able to classify most TRECVID queries correctly as shown in our experiments. Note that this work does not necessarily depend on the rich description of the topic. Our query classification technique is also applicable when users only provide several simple keywords by using the named entities from transcripts.

4. LEARNING QUERY-CLASS DEPENDENT WEIGHTS

Learning query-class dependent weights λ_i for each query class c_q can be treated as a maximum likelihood estimation problem. Given a query class c_q, for any query q belonging to c_q, a set of training data $\{\mathbf{x^i}, y^i\}$ is collected where x^i is a set of expert outputs $(P_1^u(R|q,s_i), P_1^l(R|q,s_i), ..., P_m^l(R|q,s_i))$ and $y_i \in \{-1,+1\}$ indicates whether the shot s_i is relevant to q or not. We compute the log-likelihood by taking the logarithm of the product of $P(R|q,s_i)$,

$$l(\lambda^u; X) \;=\; \sum_i \log \sum_{t=1,2} \lambda_t^u P_t^u(R|q,s_i) \qquad (5)$$

where P_1^u is retrieval scores of the text retrieval, P_2^u is $f\left(\sum_{k=1}^m \lambda_k^l P_k^l(R|q,s_i)\right)$. We present a learning algorithm to estimate the parameter based on the Expectation Maximization (EM) algorithm[7]. To apply EM algorithm for estimating λ_t^u, we must define appropriate hidden variable so as to simplify the likelihood function. Here we define the hidden variables z_{it} for each shot s_i, where one and only one of the z_{it} is one while others are zero. $z_{it}=1$ means the prediction of the shot s_i is generated from the t^{th} retrieval expert. By taking the expectation of the complete-data likelihood, we obtain the E step by defining the auxiliary function $Q(\lambda, \lambda_j)$ for j^{th} iteration,

$$Q(\lambda^u; \lambda_j^u) = \sum_{t=1,2} \sum_i h_{it} \left(\log \lambda_t^u + \log P_t^u(R|q,s_i)\right). \qquad (6)$$

where h_{it} is the expectation of z_{it} give the observable variables $E(z_{it}|X)$. The M step is setting $\lambda_{j+1} = \arg\max_\lambda Q(\lambda; \lambda_j)$. We summarize the EM algorithm in Figure 4.

To finish the M step, we need to maximize the weighted log-likelihood on the lower-level mixture of experts,

$$l(\lambda^l; X) = \sum_i h_{i2} \log f\left(\sum_{k=1}^m \lambda_k^l P_k^l(R|q,s_i)\right) \qquad (7)$$

```
Input: $P_t^u(R|q,s_i)$, t=1,2, and $y_i \in \{-1,+1\}$
Output: $\sum_{t=1}^{2} \lambda_t P_t^u(R|q,s_i)$ which optimizes $l(\lambda;X)$.
Algorithm:
Initialize $\lambda_i^{(0)}$ such that $\forall i, 0 < \lambda_i^{(0)} < 1, \sum_i \lambda_i^{(0)} = 1$
For $j = 1, 2, ....$
  1. E-step: Compute expectation
     $$h_{it}^{(j)} = \frac{\lambda_t^{(j)} P_t(R|q,s_i)}{\sum_t \lambda_t^{(j)} P_t(R|q,s_i)}$$
  2. M-step: Update parameter $\lambda_t^{(j+1)} = \frac{1}{n}\sum_i h_{it}^{(j)}$
  3. M-step: Maximize the weighted log-likelihood in (7)
  4. Check convergence if $|l(\lambda^{(j+1)};X) - l(\lambda^{(j)};X) < \epsilon|$
```

Figure 4: An EM algorithm for learning linear combination weights

where h_{i2} is the expectation for the second hidden variable z_{i2} and the link function f is a generalized linear function. It is typically chosen as an identity function or a logistic function. We will discuss these two cases as follows.

4.1 Linear Combination

When the link function f is defined as an identity function, Eq(7) can be rewritten as,

$$l(\lambda^l;X) = \sum_i \log h_{i2} \sum_{k=1}^{m} \lambda_k^l P_k^l(R|q,s_i), \qquad (8)$$

where the sum of λ_k is equal to 1 and each λ_k is larger than 0. A similar EM algorithm is applied to estimate the λ_k^l, which results in a nested EM learning framework. However, since not every expert prediction is consistently useful, we proceed an additional feature selection algorithm before the EM algorithm to filter out the inconsistent experts and provide more generalized outputs. In more details, we apply a χ^2 test to select features with confidence interval 0.1 [22]. The χ^2 statistics is generally computed to measure the dependence of two random variables. In our work, we use it to measure the dependence of each expert prediction and the corresponding labels. If an expert output is suggested to be independent to its labels the expert will be eliminated. Note that the expert with continuous probabilistic output will be converted to discrete output with a threshold of 0.5.

4.2 Logistic Regression

When the link function f is defined as a logistic function, Eq(7) can be rewritten as,

$$l(\lambda;X) = -\sum_i h_{i2} \log\left(1 + \exp\left(\lambda_0^l + \sum_{k=1}^{m} \lambda_k^l P_k^l(R|q,s_i)\right)\right). \qquad (9)$$

In this case, there are no constraints on the choice of λ_k because the output of logistic function will map any real value into probabilistic output in [0,1]. To handle the logistic function, we apply a boosting type algorithm for learning the combination of experts due to its handy implementation. The boosting algorithm shown in Figure 5 are modified from the parallel update algorithm proposed by Collins et al[6]. For each round k, the algorithm first updates the distribution $q_{k,i}$ in a manner that increases the weights of examples which are misclassified. A new step, i.e. step 2, is added

```
Input: Matrix $\mathbf{M} \in [-1,1]^{m\times n}$ where $M_{ij} = y_i P_j^l(R|q,s_i)$. $N^+$: the number of positive examples, $N^-$: the number of negative examples
Output: $\sum_k \lambda_k P_k(R|q,s_i)$ which optimizes $l(\lambda;X)$.
Algorithm:
Let $\lambda_1 = 0$, $q_0 = (0.5, 0.5, ...)$
For $k = 1, 2, ....$
  1. Compute distribution $q_k$ given $\mathbf{M}$, $\delta_k$ and $q_{k-1}$
     $q_{k+1,i} = q_{k,i}\exp\left(-\sum_{j=1}^{n}\delta_{k,j}M_{i,j}\right)$
  2. For every positive example $\mathbf{x_i}$, balance the distribution $q_{k,i} = h_{i2}N^- q_{k,i}/N^+$
  3. For $j = 1, .., n$ :
     $W_{k,j}^+ = \sum_{i:sign(M_ij)=+1} q_{k,i}|M_{ij}|$
     $W_{k,j}^- = \sum_{i:sign(M_ij)=-1} q_{k,i}|M_{ij}|$
     $\delta_{k,j} = \frac{1}{2}\log\left(W_{k,j}^+/W_{k,j}^-\right)$
  4. Update parameter: $\mathbf{\lambda_{k+1}} = \mathbf{\lambda_k} + \delta_k$
```

Figure 5: A boosting type algorithm with parallel-update optimization for logistic regression

to balance the distribution between positive and negative examples and incorporate the additional weights. The same feature selection algorithm mentioned in Section 4.1 is applied to filter out the noisy experts.

5. EXPERIMENTAL RESULTS

5.1 Description of retrieval experts

In this section we describe a subset of the retrieval experts that are available in our retrieval system, including text retrieval, image retrieval based on color and edge feature, anchor detector, commercial detector, outdoors detector, new subject monologue detector, face detetion/recognition. Here we only describe two major retrieval experts and discuss the probability generation methods because of space limit. More details of the other experts can be found in [10].

5.1.1 Text Retrieval

Generally the most reliable retrieval expert is the retrieval component for text features. The text features processed by our system span several dimensions: Production metadata such as titles and published descriptions of the video, Automatic speech transcripts(ASR), Closed captions (CC) and Video optical character recognition (VOCR) extracted from the overlaid text visible on the video frames.

VOCR is useful to capture names of people that are sometimes not explicitly referred to in the transcript, event names, and location names, as well as product names in commercials. Our experiments have shown that speech transcripts are important supplementary sources to closed captions, especially in commercial advertisements where closed captions are not available. Closed captions are synchronized to the spoken words with corresponding time alignment information from the speech transcripts. Words from a stopword list are removed, and the Porter stemming algorithm is used to remove morphological variants. All the retrieval is done using the OKAPI BM-25 formula [16]. Queries are automatically reformulated by extracting the noun phrases from the text description of the original TRECVID queries.

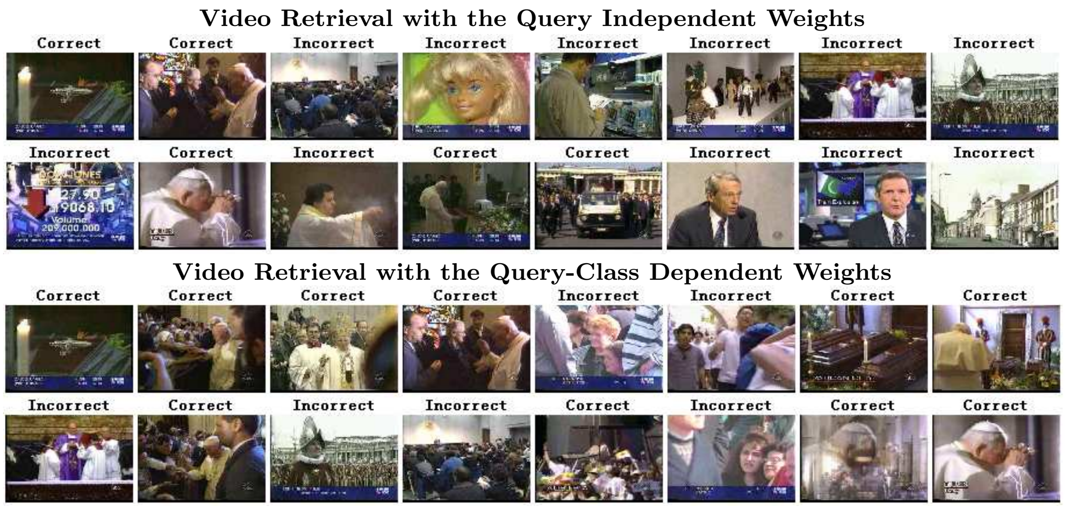


Figure 6: The keyframes of top 16 retrieved shots for the query "Finding shot of Pope John Paul II". Both set of weights are learned from the development data. The words "corret/incorrect" above the keyframes indicate whether the corresponding shots are relevant to the query or not

5.1.2 *Image Retrieval*

For image retrieval we generated two types of low-level features, i.e., color features and edge features. The color features are computed as the cumulative color histograms for each separate color channel from the HSV color space. Another low-level feature set is the canny edge direction histogram. A canny edge detector is applied to extract the edges from images. The edge histogram includes a total of 73 bins. First 72 bins represents the edge directions quantized at 5 degree interval and the last bin represents a count of the number of pixels that are not contributed to any edges. We compute a harmonic mean of the Euclidean distances from each image example to the shot's keyframe as the distance between multiple image examples and video shots.

5.1.3 *Probability Generation from Expert Outputs*

As mentioned before, each retrieval expert as the leaf node has to generate the probabilistic output of relevance $P(R|q, s_i)$. Since various experts generally produce incompatible retrieval output, different approaches are employed to normalize the retrieval scores. For a $\{0,1\}$-value output experts, we smooth the probability $P(R|q, s_i)$ when the expert's output is zero or one to be ϵ or $1 - \epsilon$ where ϵ is set to 10^{-4}. For a probabilistic output expert, we simply use the output as the probabilistic estimation. For other real-value output experts, the probabilistic estimation can be derived from the rank distribution where the posterior probability for shot s_i can be derived from $P(R|q, s_i) = 1 - Rank(s_i)/N$, where N is the number of shots in the collection. This approximation allows a simpler form of probability estimation. Other learning algorithms such as logistic regression could be chosen to convert scores into probabilities, but the necessity to learn the parameters on the fly limits their usage in retrieval.

In addition, it is also necessary to decide if the normalized expert output is representing the probability of relevance $P(R|q, s_i)$ or irrelevance $P(NR|q, s_i)$. The output of text and image retrieval experts are always representing $P(R|q, s_i)$. But it is not always true for semantic feature detectors such as the face detector. For example, for the P-query the output of face detector is treated as the probability

Query Class		P	E	O	S	Overall
Truth Labels		14	12	30	28	84
QC1	Correct	14	10	28	21	73(87%)
	Miss	0	2	2	7	11(13%)
	False Alarm	1	1	8	1	11(13%)
QC2	Correct	14	10	28	26	78(93%)
	Miss	0	2	2	2	6(7%)
	False Alarm	1	1	3	1	6(7%)

Table 1: The Results of Query Classification

of relevance, but for the O-query it is treated as the probability of irrelevance. Basically we will check if the mean of expert outputs for the relevant training data is smaller than that of the irrelevant training data. If so, it means the output represents $P(NR|q, s_i)$ which will be mapped into $P(R|q, s_i) = 1 - P(NR|q, s_i)$, otherwise $P(R|q, s_i)$ is set to the output directly.

5.2 Query Classification Results

Our automatic query classification method described in Section 3.2 is evaluated using the TRECVID queries used in past 3 years, totally 84 queries. Each testing query is first manually labeled with the query class it belongs to. All the manual labels are checked by two human subjects without any disagreement found. Then the manual labels are compared with the class labels assigned by our query classification method. The comparison results are summarized in Table 1, where QC-1 and QC-2 refer to the method without and with syntactic parsing respectively.

Our method works very well and quite reliably in classifying the TRECVID queries, achieving 87% without using syntactic parsing and 93% using it. P-queries and E-queries are almost perfectly classified, which implies the effectiveness of our named entity extraction algorithm. For QC-1, the large number of false alarms on O-query and misses on S-query are due to its mistaking some S-queries with only one longest-matched NP as O-queries. We can see that QC-2 by syntactic parsing corrects 5 out of 8 such misclassifications. It fails to correct the other 3 due to the parsing errors caused by complicated structures of the NP. For example, when parsing "a hot air balloon in the sky" our parser erro-

Models	Weights	Learning	MAP	Prec10	Prec30	Prec100	Rec10	Rec30	Rec100
TEXT	QI	N/A	0.15	0.24	0.19	0.13	0.09	0.17	0.30
HME	QI	Logistic	0.14	0.29	0.23	0.14	0.08	0.16	0.31
HME	QI	Oracle	0.18	0.34	0.25	0.15	0.12	0.20	0.35
SME	QCD	Logistic	0.16	0.33	0.25	0.14	0.12	0.20	0.31
HME	QCD	Linear	0.19	0.29	0.25	0.16	0.12	0.22	0.38
HME	QCD	Logistic	0.20	0.34	0.26	0.16	0.13	0.22	0.38
SME	QD	Logistic	0.17	0.34	0.25	0.14	0.12	0.19	0.29
HME	QD	Logistic	0.21	0.39	0.27	0.16	0.15	0.24	0.38

Table 2: Comparison between various retrieval approaches in terms of multiple retrieval criteria. HME: hierarchial mixture of experts, SME: single-layer mixture of experts, TEXT: text retrieval, QI: query independent, QCD: query-class dependent, QD: query dependent. See text for more details.

neously put "air" as a noun and "balloon" as a verb, resulting in a classification error. Besides parsing errors, another source of errors is the implicit priority of query classes. For example, the only false alarm on P-query, which also accounts for a miss on E-query, is because an E-query "Find shots of Price Tower designed by Frank Lloyd Wright" which has been classified as P-query since higher priority is given to the P-query over other named entities. We realize that the query pool is still not large enough to provide convincing result for other types of queries. To evaluate the proposed query classification scheme, we might need to collect more queries in the future work.

5.3 Video Retrieval Results

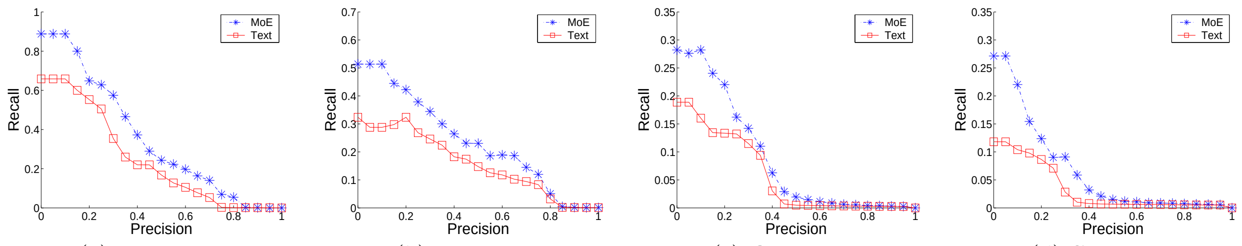
Our video retrieval experiments followed the guidelines for the manual search task in the TRECVID 2003, which require an automatic system to search without human feedback for video shots. The shot boundaries are provided by TRECVID. The system needs to search 25 query topics in a 65-hour news video collection of ABC World News Tonight, CNN Headline News and C-SPAN programming. The definitive information about this collection can be found at the TRECVID web site [18]. The retrieval units were video shots defined by a common shot boundary reference. The evaluation results are reported in terms of the mean average precision(MAP) and precision/recall at top N retrieved shots. The ground truth is provided officially by TRECVID. We use the Informedia client [10] to collect the relevant video shots for 20 out of all 25 queries from the development collection, which is another 65-hour news video collection from a different period. There are no relevant shots available for the other 5 queries. Although we cannot guarantee all the relevant shots can be found, the collection generally provides a high coverage for the relevance data based on our previous experience. We used the manual labeled query class for training while the predicted query class for testing.

Figure 6 illustrates the advantage of using the query-class dependent weights against query-independent weights. This improvement is achieved by successful combining of different expert outputs, such as the face detector and the image retrieval expert, using the weights learned for the P-query. There are 10 relevant shots retrieved out of top 16 shots with query-class dependent weights, compared with 5 relevant shots retrieved with query-independent weights.

Table 2 lists a more detailed comparison for various retrieval approaches in terms of mean average precision at top 400 shots and precision/recall at 10, 30 and 100 shots. We present three baseline combination strategies as compared to the proposed approaches, i.e. the text retrieval(**Text+QI**), learning the query-independent weights using logistic regression(**HME+QI+Logistic**) and an oracle of the best query-independent weights assuming the ground truth on the testing collection is available (**HME+QI+Oracle**). For the proposed approach using query-class dependent weights, the performance using both linear combination and logistic regression are reported(**HME + QCD+Linear**, **HME +QCD + Logistic**), where the development data of queries in the same query class are aggregated into a larger training pool for weight learning.

Surprisingly, learning a set of query-independent weights from the development data produces even worse performance than the text retrieval alone, which again indicates the difficulty of multi-modality combination. In contrast, the proposed approaches with both learning algorithms are considerably superior to the baseline strategies on both mean average precision and precision/recall at several levels. Overall they have around 5% improvement over text retrieval and even 2% improvement over the global oracle in terms of mean average precision. On average the logistic regression algorithm achieves a slightly better performance than the linear combination algorithm.

To further investigate whether it is sufficient for learning query-class dependent weights compared to learning query dependent weights, we also list the retrieval results learned on a per query basis(**HME+QD+Logistic**), which, although impractical, can be considered a upper bound for the proposed approaches. For the queries without any relevant shots in the development set, it naturally backs off to use the class-specific weights. It can be observed that the performance of learning on a per query basis is not significantly better than that of learning on query class basis. This observation shows that using the query-class specific weights might be a reasonable choice which can be learned from a tractable number of training data and generalized to the unseen queries without suffering from remarkable performance hurt. Finally, we show how the system can benefit from the hierarchial mixture of expert model(HME) compared to the single-layer mixture of expert model(SME) with the same learning setting(**SME + QCD+Logistic**, **SME +QD+Logistic**). Although the SME model can also gain improvement using query dependent weights, the increase is not as large as that of the HME model, which can be partially explained by the overfitting problem of learning the text retrieval weights. Figure 7 depicts the interpolated precision-recall curve of the proposed approaches against text retrieval for every query class. It can be found that the hierarchial mixture-of-expert model consistently outperforms text retrieval by a noticeable margin.



(a) **P-query** (b) **E-query** (c) **O-query** (d) **S-query**

Figure 7: The Precision-Recall curves comparing HME and text retrieval for every query class.

6. CONCLUSION

This paper presented an automatic video retrieval approach using query-class dependent weights within a hierarchial mixture of expert framework to combine multiple retrieval results. We first classify each user query into one of the 4 predefined query classes, i.e. finding named persons, named objects, general objects and scenes. Then we apply the associated weights, which are learned from the development data off-line, to fuse the outputs of multiple retrieval results. The combined probabilistic outputs are translated into a rank list of video shots as the final retrieval output.

The experimental results on TRECVID manual retrieval task demonstrate that applying query-class dependent weights can considerably improve the retrieval performance over the query-independent weights. Our experiments also show the feasibility of accurately mapping queries into some predefined classes. Moreover, by observing the similar performance between using query-class dependent weights and query dependent weights, we conclude that it is reasonable to learn the combination weights based on pre-defined classes because they can be learned from a tractable number of training data and generalized to other unseen queries easily without significant performance decline. Future work includes evaluating our query classification approach with a larger pool of queries, refining our definition of query classes and using image examples to aid query classification.

Acknowledgement

This research is partially supported by Advanced Research and Development Activity (ARDA) under contract number MDA908-00-C-0037 and MDA904-02-C-0451.

7. REFERENCES

[1] A. Amir, M. Berg, S.-F. Chang, and et al. IBM research TRECVID-2003 video retrieval system. In *NIST TRECVID-2003*, Nov 2003.

[2] M. Beigi, A. B. Benitez, and S.-F. Chang. Metaseek: A content-based meta-search engine for images. In *Proc. of SPIE*, 1997.

[3] D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. Nymble: a high-performance learning name-finder. In *Proc. 5th Conf. on Applied Natural Language Processing*, 1997.

[4] E. Brill. Some advances in transformation-based part of speech tagging. In *Proc. of the 12th National Conf. Artificial Intelligence*, volume 1, 1994.

[5] P. Clarkson and R. Rosenfeld. Statistical language modeling using the CMU-Cambridge toolkit. In *Proc. Eurospeech'97*, 1997.

[6] M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, adaboost and bregman distances. In *COLT'00*, pages 158–169, 2000.

[7] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[8] G. Gaughan, A. F. Smeaton, C. Gurrin, H. Lee, and K. McDonald. Design, implementation and testing of an interactive video retrieval system. In *Proc. of 11th ACM MM Workshop on MIR*, Nov 2003.

[9] D. Grinberg, J. Lafferty, and D. Sleator. A robust parsing algorithm for link grammars. In *Proc. of the 4th Int'l Workshop on Parsing Technologies*, 1995.

[10] A. G. Hauptmann and et al. Informedia at TRECVID 2003: Analyzing and searching broadcast news video. In *Proc. of TRECVID 2003*, Gaithersburg, MD, 2003.

[11] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

[12] I.-H. Kang and G. Kim. Query type classification for web document retrieval. In *Proc. of the 26th ACM SIGIR*, pages 64–71. ACM Press, 2003.

[13] X. Li and D. Roth. Learning question classifiers. In *COLING'02*, Aug 2002.

[14] A. Merlino, D. Morey, and M. Maybury. Broadcast news navigation using story segmentation. In *Proc. ACM Multimedia*, 1997.

[15] L. Ramshaw and M. Marcus. Text chunking using transformation-based learning. In *Proc. of the ACL Third Workshop on Very Large Corpora*, 1995.

[16] S. E. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau. Okapi at TREC4. In *Text REtrieval Conference*, pages 21–30, 1992.

[17] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, 1983.

[18] TRECVID: TREC Video Retrieval Evaluation. http://www-nlpir.nist.gov/projects/trecvid.

[19] T. Westerveld, T. Ianeva, L. Boldareva, A. P. de Vries, and D. Hiemstra. Combining infomation sources for video retrieval: The lowlands team at TRECVID 2003. In *NIST TRECVID-2003*, Nov 2003.

[20] R. Yan and A. Hauptmann. Co-retrieval: A boosted reranking approach for multimedia retrieval. In *Proc. of Intl. Conf. on Image and Video Retrieval*, 2004.

[21] H. Yang, L. Chaisorn, Y. Zhao, S.-Y. Neo, and T.-S. Chua. VideoQA: question answering on news video. In *Proc. of the 11th ACM MM*, pages 632–641, 2003.

[22] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proc of the 14th ICML*, pages 412–420, 1997.

[23] D. Zhang and W. S. Lee. Question classification using support vector machines. In *Proc. of the 26th ACM SIGIR*, pages 26–32. ACM Press, 2003.