Building Detectors to Support Searches on Combined Semantic Concepts

Robin Aly
University of Twente
P.O.Box 217
7500 AE Enschede,
the Netherlands
r.aly@
ewi.utwente.nl

Djoerd Hiemstra
University of Twente
P.O.Box 217
7500 AE Enschede,
the Netherlands
d.hiemstra@
ewi.utwente.nl

Roeland Ordelman
University of Twente
P.O.Box 217
7500 AE Enschede,
the Netherlands
ordelman@
ewi.utwente.nl

ABSTRACT

Bridging the semantic gap is one of the big challenges in multimedia information retrieval. It exists between the extraction of low-level features of a video and its conceptual contents. In order to understand the conceptual content of a video a common approach is building concept detectors. A problem of this approach is that the number of detectors is impossible to determine. This paper presents a set of 8 methods on how to combine two existing concepts into a new one, using a logical AND operator. The scores for each shot of a video for the combined concept are computed from the output of the underlying detectors. The findings are evaluated on basis of the output of the 101 detectors including a comparison to the theoretical possibility to train a classifier on each combined concept. The precision gains are significant, specially for methods which also consider the chronological surrounding of a shot promising.

1. INTRODUCTION

Bridging the semantic gap has been declared to be a key problem in multimedia information retrieval since long [7]. The gap is between the well understood extraction methods of low level features from media files and the high level concepts needed to satisfy the information needs of the user expressed in a query to a search system. That is why detecting the presence of semantic concepts in videos has come into research focus in the recent years [3]. The set of required concepts is hard to define and building programs to detect concepts is difficult and expensive. This paper evaluates several methods on how to combine two detectors for individual concepts to a detector for a combined concept. The number of available concepts is thus quadratically increased. If multiple combinations are allowed this growth is exponential.

In this paper we adopt the definition of a semantic concept from Snoek at al. [9]. There, a concept is defined as something which must appear clearly in the static key frame of a video shot. Thus the expression does not cover concepts which are only represented in the audio content nor purely abstract concepts like "World Peace".

The research community seems to have settled to create for each concept a so called concept detector programs which assign each shot of a video a certain score representing the estimated likelihood of the presence of this concept. Many research groups work on the question on how to best create these detectors.

The problem of how to determine the set of required detectors applies even for limited domains. The most commonly used metaphor for this problem is an indefinite "semantic space". An established approach is to create detectors for a certain set concepts which should allow to answer all possible queries [4]. Besides the problem of selecting the right concepts for a specific domain, another question is how to handle requests for concepts which are not directly present in the set of available concepts. Due to an unknown structure of the "semantic space" of the domain, it is not an option to simply increase the number of detectors, until the point where all requested concepts are covered. Thus, some concepts have to be expressed as a combination of concepts for which detectors exist. This problem has not been satisfactory addressed [9]. One of the subproblems is that techniques which use classifiers on feature vectors suffer from the few positive examples the combined concepts often have in the supplied training data.

An example where the combination of concepts are useful can be found in [8]. Here topics are always mapped to a single concept detector which is then used for searching the data set. As the user might have really meant the combined concept, the use of it would improve the precision of the answer.

Essentially the task of a detector for a combined concept is to come up with scores for the presence of this concept in a given shot. This has to be done by simple methods as complex methods like classification do not perform well with few positive examples in the training data, which is common for combined concepts. The score will be based on the output from the two underlying detectors and possibly findings from the results of these detectors on a training set. An example for the combination of two existing concepts would be, deriving a detector for "Indoor sports" from the detectors for "Indoor" and "Sports". Allowing an arbitrary number of these combinations would even result in exponen-

tial growth of the number of available concept detectors. A combination of three concepts could for example be "indoor sports referee".

The rest of this paper is structured as follows: In Section 2 we present work which has been done on concept detection and on searching on the resulting data. The following Section 3 describes the proposed methods in detail. Section 4 describe the experiments which have been carried out together with a interpretation of the results. The paper ends with a proposal for future work and a conclusion in Section 5.

2. RELATED WORK

A lot of work on concept detection has been done in the context of the TRECVID Workshop [6]. There the expression "high-level feature" is used, which is treated as a synonym for the expression semantic concept, which is used here. A task for the participants is to come up with algorithms which compute a sequence of shots ordered by their likelihood to contain a certain concept. For each concept out of a predefined set the participants have to return a list of shots each attributed with a score whether this concept appears in each shot. The different search tasks could then use the output to improve their search performance.

The LSCOM workshop aims to build two sets of concept lexicons. A lexicon is defined as a set of concepts together with a short description of the shots which should qualify to the concept and truth judgment for each concept on each shot. The two lexicons are a light- and a large scale concept ontology. The LSCOM light ontology [4] consists of truth judgments for 39 concepts. Whereas the full LSCOM test environment [2] comprises nearly 1000 concepts of which 317 are already annotated.

There have been several approaches on building detector for a single concept. Among them are rule based schemes, Bayesian classifiers, neural network classification, Gaussian mixture models, finite state machines, support vector machines and hidden Markov models. Snoek et al. give a good overview of these methods in [11]. Most of these methods are classifiers algorithms which are commonly only used to put items in different classes, in this case "is present" or "not". In an information retrieval setting it is desirable to know which shot, out of two, is the more likely to contain a certain concept. With this criteria it is possible to create a ranking of a set of shots. Thus it is a strong requirement that the classifiers are also able to return some kind of score for each shot with which they can be compared. There is evidence that using a SVM classifier gives the best results and is most flexible in its application [10] [13].

The area of multi-concept relationships is related to the presented approach of creating detectors for combined concepts. The idea behind the multi-concept relationships is to let the scoring process for a semantic concept be influenced by the relationship with other related semantic concepts, for example the likelihood of observing the concept "bus" is less likely when also observing a "indoor" setting with a high score. Rong Yan gives a good overview of the available techniques to do this in [13]. The link between the two methods is that the multi-concept relationship approach tries to improve detectors by considering the presence of related concepts and the presented approach creates new concepts. Thus both considering multiple concepts.

The MediaMill Group [9] evaluated several ways of combining "low-level" features, namely associated text and color-histograms, into high-level concept detectors. Each strategy is based on a vector of certain low-level features. The detector is a SVM model [12] which gets trained on designated training data in order to accurately assign scores to shots from other data sources. The evaluated methods differ in what the vectors, used for training and estimation, consist of. The following variations have been researched in the their experiments: experiment 1) consider only video features, experiment 2) only take text features from automatic speed recognition into account, experiment 3) use both kinds of low-level features (early fusion), experiment 4) combine the output of 1) and 2) (late fusion) and finally 5) combining the output of methods 1-4. For TRECVID 2005 they trained and evaluated 101 concept detectors on approximately 30,000 shots. On average method 1) was performing the best on the TRECVID dataset. That means the textual features were less beneficial. The reason for this were not explicitly researched.

3. COMBINATION METHODS

This section describes the proposed methods to combine two detectors into a detector for a combined concept. Only the combination by a logical "AND" (i.e. c_1 AND c_2) is considered here. This resembles the case that there have to be two concepts present at the same time. Studies on cases like c_1 AND NOT c_2 are planned for the future.

It is required to be able to calculate the scores for the shots in an online fashion. The reason for this is the big number of shots and large number of concepts in a realistic scenario - making it impossible to keep all scores stored on disk, due to the exponential growth in number of the combinations.

The rest of the section is structured as follows: To lay a good formal foundation Subsection 3.1 defines a number of important terms. Afterwards, in Subsection 3.2, the proposed combination methods are defined and discussed.

3.1 Basic Definitions

In the following the most important terms for the problem of building a combined concept are formally defined. This should help reason about and defining the methods presented later.

Def 1 (Video) *A Video v is a combination of a stream of pictures and optionally sound. V is the set of videos on which the concept detection should be done. Formally: $V = \{v_1, ..., v_m\}$.*

Def 2 (Shot) *A shot is a sequence of pictures of a video being recorded in one go by a camera. $s_{i,j}$ stands for the jth shot of video v_i. Following notation is used:*

Number of shots *Function $ns()$ returns the number of shots contained by video v. Thus $ns(v) \Rightarrow \mathbb{N}$ where $v \in V$.*

Shots of a video *S_i is the temporal ordered sequence of all shots of video v_i. $S_i = [s_{i,1}, ..., s_{i,ns(v_i)}]$*

All shots *S is the set of all shots from all videos. Formally defined this reads as: $S = \bigcup_{i=1}^{m} \{s_{i,1}, ..., s_{i,ns(v_i)}\}$*

Def 3 (Set of Concepts) *The set of concepts $C = \{c_1, ..., c_n\}$ is the set of semantic concepts which a system has appropriate detectors for.*

Def 4 (Rankingfunction) *A ranking function r calculates real number, a score, for a shot s. The score symbolizes the likelihood that in shot s a certain concept c is present. For convenience here $r_i(s)$ with $s \in S$ is used which means $r(c_i, s)$ with $c_i \in C$ and $s \in S$. Formally: $r(c, s) \Rightarrow \mathbb{R}$ with $c \in C$ and $s \in S$;*

As each of the two rankings for the individual concepts does not necessarily contain all shots. Again we define for each ranking function that does not define a score the to a shot this score to be 0.

3.2 Proposed Methods

In the following the methods for creating a ranking for a combined concept are described. The task is to find a ranking function for the combined concept $c = c_k$ AND c_l in a way that it yields the highest precision given relevance judgments. For convenience a ranking function r for the above combined concepts c is written as $r_{k,l}(s)$ which means $r(c_k$ AND $c_l, s)$,. All the following methods require that the score of both detectors for a individual shot have to be efficiently accessible if the score of this shot should be computed.

Add.

The add rankingfunction simply adds the scores of the base ranking functions together. This way shots get favored which have a high rank for both concepts.

$$add_{k,l}(s_{i,j}) = r_k(s_{i,j}) + r_l(s_{i,j})$$

Multiply (mult).

The mult rankingfunction multiplies the ranking values, thus assuming the independence between the occurrences of both concepts.

$$mult_{k,l}(s_{i,j}) = r_k(s_{i,j}) r_l(s_{i,j})$$

Combine weighted (cbw).

The combine weighted method uses the average precision of the underlying detectors on the training data as weighting factors for each score (here $ap(c)$.

$$cbw_{k,l}(s_{i,j}) = \frac{ap(k) r_k(s_{i,j}) + ap(l) r'_l(s_{i,j})}{ap(k) + ap(l)}$$

Neighbor functions.

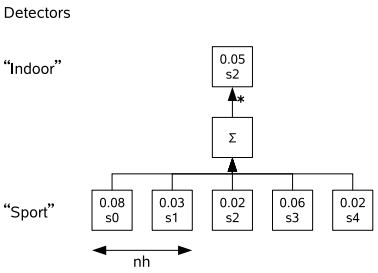
The family of Neighbor methods do not only consider both scores of the shot in question but also the scores from the shots chronological before and after the shot. The neighborhood nh is the number of shots to consider before and after the actual shot. This method is based on the assumption that a shot is more likely about "indoor sports" if the surrounding shots also have a higher score. A reason for this might be that a concept might be presented in a unusual form in the current shot, not allowing the concept detector to recognize it properly in this shot. If the concept is detected in the adjacent shots the probability that it was present in the current shot is considered high. For this family of combination methods it is required that the previous and next shots of a certain base shot can be accessed efficiently. Once the output of the two detectors can be accessed by video and chronological sorted shots, an efficient algorithm using a sliding window of length $(2nh) + 2$ to calculate the combined score exists.

Neighbor left (nl).

The Neighbor left method considers the score for s_i and multiplies it with the summed score of the neighboring shots of s_j.

$$nl_{k,l}(s_{i,j}) = r_k(s_{i,j}) \sum_{m=max(j-nh,0)}^{min(j+nh,ns(v_i))} r_l(s_{i,m})$$

The following diagram should visualize the working. Mathematically the score can be calculated as follows $nh = 2$: $r(s_2) = in(s_2)(\sum_{i=0}^{4} s_i)$;



Neighbor right (nr).

Neighbor right does the equivalent to the Neighbor left method.

$$nr_{k,l}(s_{i,j}) = r_l(s_{i,j}) \sum_{m=max(j-nh,0)}^{min(j+nh,ns(v_i))} r_k(s_{i,m})$$

Neighbor (n).

The general Neighbor method combines the both single sided Neighbor methods by averaging Neighbor left and Neighbor right.

$$n_{k,l}(s_{i,j}) = \frac{(nl_{k,l,nh}(s_{i,j}) + nr_{k,l,nh}(s_{i,j}))}{2}$$

Neighbor Multiply (nm).

Neighbor Multiply multiplies instead of adding the both components of the combination together. The rational behind this is to get a more robust ranking. That means in case a detector sometimes recognizes something wrong there is still the hope that the other might successful do it's job.

$$nm_{k,l}(s_{i,j}) = nl_{k,l,nh}(s_{i,j}) * nr_{k,l,nh}(s_{i,j})$$

Neighbor Weighted (nw).

The Neighbor Weighted method is equal to the Neighbor method only that it does the weighting of both parts of the Neighbor function by the average precision the source detector had on the training data with the same meaning of $ap()$ as in add.

$$nw_{k,l}(s_{i,j}) = \frac{ap(k) nl_{k,l,nh}(s_{i,j}) + ap(l) nr_{k,l,nh}(s_{i,j})}{ap(k) + ap(l)}$$

4. EXPERIMENTS

This section presents the experiments, which were done to evaluate the presented methods. All the experiments were based on the MediaMill data [9] which provides truth judgments, features for each shot and scores for both a train and a test dataset. For the proposed methods only the scores and truth judgments from the training set were used to help creating the scoring for the combined concept on the test data..

The needed relevance judgments to evaluate the performance of a combined concept detector is derived from the judgments of the underlying concepts c_i and c_j. It can be computed from the judgments for c_i and c_j using the logical "AND" operator. That means the concept $c = (c_i \text{ AND } c_j)$ is only present if c_i and c_j are present.

The models for the single concept detectors are trained using training data. The evaluation of the methods was done purely on the test data. The cbw and nw methods and the built SVM used the training data for the calculation of the average precision or to train a model, in case of the SVM.

Besides self created scripts which automated the execution of the experiments, we used two software packages: The software libsvm version 2.83 [1] was used for support vector classification. It is important to note that the software needs specifically to be told to produce probabilities for the single classes. The second package was trec_eval version 8.1 [5]. This package was used to produce the various quality parameters used in the evaluation.

In both experiments all Neighbor methods were using a neighborhood constant $nh = 5$. Runs on train data confirmed that this was a good value to use since it increased the average precision and was still computational cheap enough to use it on big datasets.

From the theoretical $\frac{101*(101-1)}{2} = 5050$ possible combined concepts 443 were selected as possible combinations. The reason for this selection was that the other concepts had either in the training or test dataset less than 30 positive examples. This few positive examples would introduce too much randomness in the precision calculations.

Two separate experiments were performed, which are described in the following sections. Afterwards in Subsection 4.1 a experiment run is described which compares the presented combination methods against each other and a preliminary baseline. Subsection 4.2 presents the results on comparing the best of the previous evaluated combination methods with the theoretical reasonable but practically infeasible possibility of building a SVM classifier for the combined concept.

4.1 Experiment 1: Method Comparison

The first experiment considered the performance of the available 443 combined concepts. The aim of this experiment was to compare the proposed methods among themselves and against a baseline on the test data. As a baseline the average of both single concept detectors was used ('single'). As a quality measure the mean average precision of all the produced rankings of a method was chosen. The left and right alternative of the Neighbor methods were left out, because it was not expected that they would have better performance than the other methods of this class. The experiment was conducted the examination for the base experiments 1 to 4 see Section 2.

An overview of the results of this experiment is shown in Table 1. One can see that in all cases the proposed methods are better than the baseline of a single detector. The finding from [9], that the usage of detectors considering only visual features (experiment 1), can also found in the combined concept detectors. It is expected that this means that better performing input detectors also yield better combined detectors. From the two basic combination methods, add and mult, the later performs better in all experiments. The reason is probably that it gives a more stable scoring for varying precision of the input detectors. An explanation for this can be that the probabilities from underlying detectors are independent. The cbw method improved the add method in experiment 2 by 2.3% and worsens it in experiment 2. Otherwise it stays the same. Thus it is not clear whether it helps boosting the performance of it base method. The Neighbor methods n,nm and nw all perform nearly identical. The multiplying version nm slightly performs better on the visual only experiment 1 and worse on the text only experiment 2. This phenomenon is expected to be bound to the specific instances of the concepts and is not further investigated. Thus the Neighbor methods n, nm and nw are assumed to perform the same. The average gain compared to the base line was 7% absolute which corresponds to a relative improvement of 78%. A general improvement was also determined by a Student-t test with a significance level of 5%.

method	ex1	ex2	ex3	ex4
single	0.093	0.050	0.088	0.088
add	0.112	0.058	0.101	0.109
mult	0.142	0.065	0.121	0.127
cbw	0.135	0.056	0.101	0.109
n	0.166	0.074	0.131	0.143
nm	0.167	0.072	0.131	0.143
nw	0.166	0.074	0.131	0.143

Table 1: Mean average precision of all 443 combined concepts of experiment one to four (ex1-4)

In Figure 1 the precision-recall graph for experiment 1 is shown. As the Neighbor methods nearly had the same performance only the first presented Neighbor method is drawn. This graph is supporting the previous statements. The plots show that the improvement of precision mainly originates from lower recalls levels. One can also see that the Neighbor methods are performing best over all recall levels. The precision-recall graphs from other base experiments do not reveal any other findings.

4.2 Experiment 2: SVM Classifier

The second experiment was done to evaluate the theoretical alternative of creating classifiers for combined concepts online. This method is theoretical, as it is not possible to store and train all models for each combined concept thus the classifier need to be trained online, which is impractical. It was benchmarked against the best method from the previous experiment (n) and the use of the average of the single detectors 'single'. Due to the limitation of CPU-time resources only random samples of concept combinations were considered. The not available computer cluster and the tremendous optimization costs for the classifier models, resulted in a random sample of 20 concept combinations. For

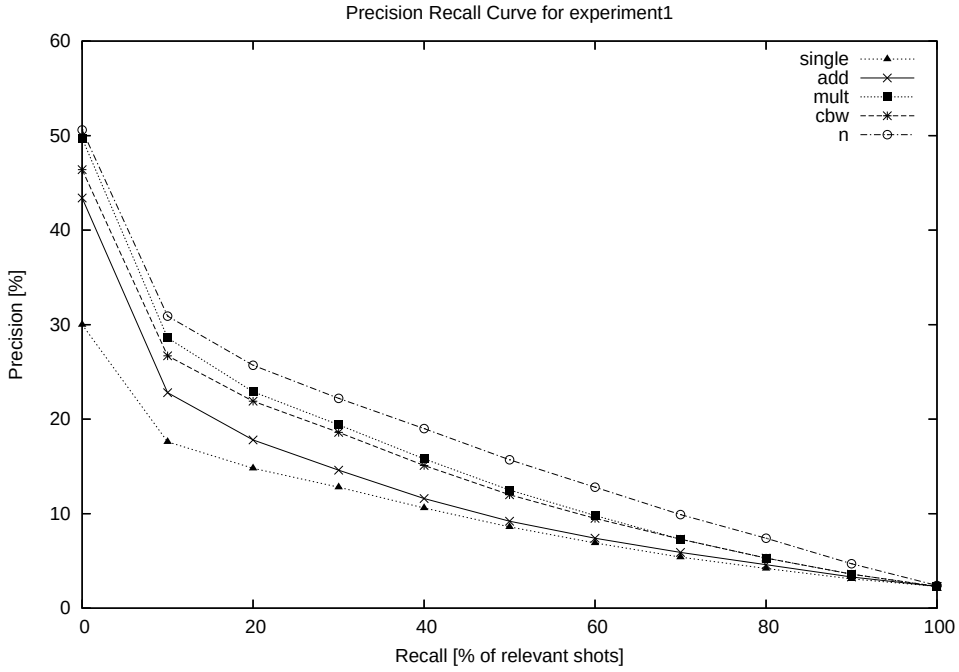


Figure 1: Precision recall curve for experiment 1

the same reason the experiment was also limited on the base data of experiment 4 from [9].

The input parameters for the SVMs were optimized as follows. The optimization's aim is not the classification accuracy but the precision of the produced ranking. This is important as the classification disregards position in a ranking and could therefore produce different results. The needed vector for the training and optimization was build from the two scores of both underlying detectors for concepts c_1 and c_2 associated with a certain shot. Thus $\vec{x_{s_i}} = (r_1(s_i), r_2(s_i))$. This resembles a kind of late fusion considering the detectors output as a trained output based on other features.

Following Snoek et al. the used parameters were not γ, a kernel parameter, an C, a penalty factor for the error term. Instead the weights of the positive (w+1) and negative (w-1) examples. First all the input vectors were scaled to the range of [0..1]. Afterwards a coarse search in steps of three in the interval [0..100] was performed. Around the best point a fine search with step size 1 is done. The set of training vectors is then split in three. On two sets the model is build with the parameters to test. Afterwards a ranking from the third set of feature vectors is produced using the generated model. This is done for all three possible combinations. Then the produced rankings are concatenated, sorted by their estimated probability and tested for the precision. The parameter combination with the highest precision is used to train the final model.

In Table 2 the mean average precisions of the examined methods are shown. One can see clearly that the Neighbor method (n) performs the best. The SVM method is worse than the single method. The large number of combined concepts with only few positive examples, which are known to problematic for SVMs, could be a explanation. That is why the training on train data does not reflect on the execution on the test data. Figure 2 (a) shows the (average) precision recall curve for this experiment. This shows that the Neighbor method achieves higher precision in the range of lower recall levels. A reason for this could be that in this sample of combined concepts the correctly, highly scored shots lie chronological close to each other, thus increasing each other's score.

method	ex4
single	0.155
svm	0.068
n	0.208

Table 2: Mean average precision over all 20 combined concepts for experiment four

Figure 2 (b) shows a overview of the achieved precision of the combined concept detectors using the SVM and Neighbor method in relation to the number of positive examples in the training data. One can see that the Neighbor method always performs better than the SVM method. Specially in the regions with very few positive examples it is still able to deliver some precision.

5. CONCLUSION & FUTURE WORK

This work presented nine methods on how to create a detector for a semantic concept, logically combined from two other semantic concepts using the "AND" operator. For the two base concepts the detectors were assumed to exist. The main problem was the inability to create an indefinite amount of detectors, let alone training them at search time. Nevertheless the presented methods were compared against a baseline of taking the average of both base scores and

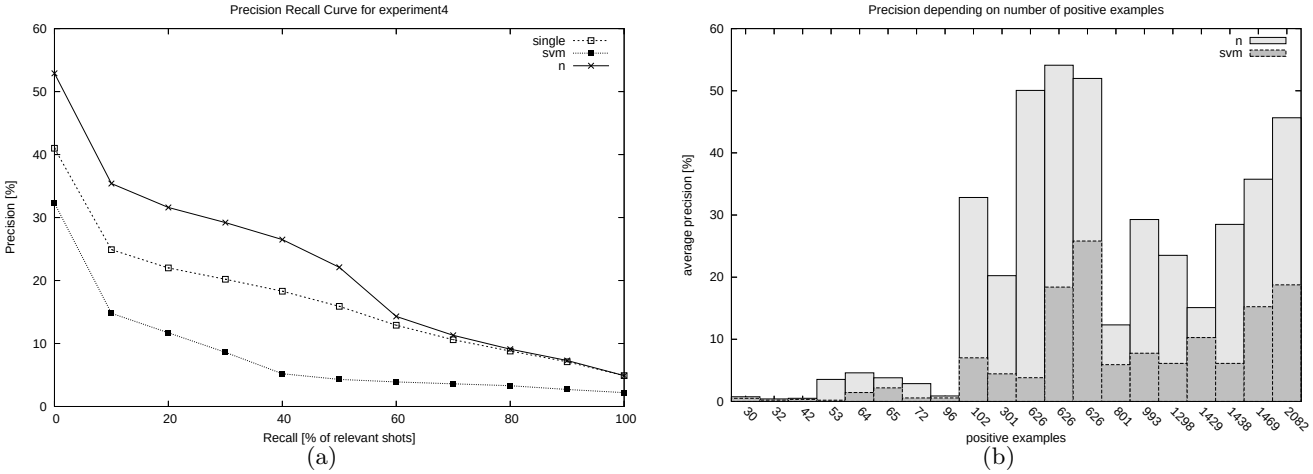


Figure 2: Precision-Recall Curve and Influence of Positive Examples

creating a classifier online. The presented Neighbor methods were performing the best. With mean average precision gains of about absolute 7% against the base line which is a relative gain of about 78%. This is regarded as a positive result.These methods require an access structure that allows to enumerate the shots in an chronological order.

Another big advantage of the methods presented here is that, even if it was feasible to train a classifier for each possible combined concept, the quality would not be optimal as the number of positive examples are much smaller than for the single concept and therefore, as this is a major problem of classifiers, their performance degrades.

Future work in this direction should include investigation on i) using also other operations like negating the presence of a detector, ii) combine more than just two concepts and iii) alternative formulas which do not require scores of the data structures to be enumerated chronologically.

6. REFERENCES

[1] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[2] Milind Naphade, John R. Smith, Jelena Tesic, Shih-Fu Chang, Winston Hsu, Lyndon Kennedy, Alexander Hauptmann, and Jon Curtis. Large-scale concept ontology for multimedia. *IEEE MultiMedia*, 13(3):86–91, 2006.

[3] Milind R. Naphade and John R. Smith. On the detection of semantic concepts at trecvid. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 660–667, New York, NY, USA, 2004. ACM Press.

[4] M.R. Naphade, L. Kennedy, J.R. Kender, S.-F. Chang, J.R. Smith, P. Over, and A. Hauptmann. A light scale concept ontology for multimedia understanding for trecvid 2005. Technical report, IBM T.J. Watson Research Center, 2005.

[5] NIST. *Trec eval 2.81*, 2006. Software available at http://trec.nist.gov/trec_eval/.

[6] W. Kraaij A. F. Smeaton P. Over, T. Ianeva. Trecvid 2005 an overview, 2005.

[7] Nicu Sebe. The state of the art in image and video retrieval. In *Image and Video Retrieval*, volume Volume 2728/2003, pages 1–8. Springer Berlin / Heidelberg, 2003.

[8] Cees G. M. Snoek, Jan C. van Gemert, Theo Gevers, Bouke Huurnink, Dennis C. Koelma, Michiel van Liempt, Ork de Rooij, Koen E. A. van de Sande, Frank J. Seinstra, Arnold W. M. Smeulders, Andrew H. C. Thean, Cor J. Veenman, and Marcel Worring. The MediaMill TRECVID 2006 semantic video search engine. In *Proceedings of the 4th TRECVID Workshop*, Gaithersburg, USA, November 2006.

[9] Cees G. M. Snoek, Marcel Worring, Jan C. van Gemert, Jan-Mark Geusebroek, and Arnold W. M. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, pages 421–430, New York, NY, USA, 2006. ACM Press.

[10] Cees G.M. Snoek and Marcel Worring. Multimedia event-based video indexing using time intervals. *IEEE Transactions on Multimedia*, 7(4):638–647, 2005.

[11] Cees G.M. Snoek and Marcel Worring. Multimodal video indexing: A review of the state-of-the-art. *Multimedia Tools and Applications*, 2005.

[12] V. N. Vapik. *Learning Theory: Inference from Small Samples*. Wiley, 1998.

[13] Rong Yan. *Probabilistic Models for Combining Diverse Knowledge Sources in Multimedia Retrieval*. PhD thesis, Canegie Mellon University, 2006.