Forest of Fuzzy Decision Trees and their Application in Video Mining

Marcin Detyniecki
Université Pierre et Marie Curie-Paris6
CNRS UMR 7606, LIP6,
104 av. du Président Kennedy
Paris, F-75016, France
Marcin.Detyniecki@lip6.fr

Christophe Marsala
Université Pierre et Marie Curie-Paris6
CNRS UMR 7606, LIP6,
104 av. du Président Kennedy
Paris, F-75016, France
Christophe.Marsala@lip6.fr

Abstract

One of the great challenges today is to index videos with high-level semantic concepts or features. The basis of our approach is to use a fuzzy decision trees (FDT) to construct the heart of the system in order to reduce the need of human usage in the process of indexation. But when we address large, unbalanced, multiclass data sets, a single classifier - such as the FDT - is insufficient. Therefore we study the use of forests of fuzzy decision trees (FFDT): (a) its effectiveness for a high level feature detection task and (b) the effect on performance from number of classifiers point of view.

Keywords: Video mining, High level Features, Fuzzy Decisions Trees.

1 Introduction

The growth of multimedia data and in particular of video data has caused a corresponding growth in the need to analyze and to exploit them. One of the great challenges today is to be able to index these data with high-level semantic concepts (or features) such as "indoor/outdoor", "people", "maps", "military personnel", etc. Today, we are able to calculate accurately low level features as for instance colour or texture histograms. Unfortunately, between these two levels there is an unbridgeable gap.

One solution to bridge the semantic gap is, based on a set of examples, to learn or to extract a general rule that will allow classifying new examples. This type of approach is known as inductive reasoning. Inductive machine learning is a well-known research topic with a large set of methods, the most common being the decision trees (DT). However, robustness and threshold problems appear when considering classical DTs. The introduction of fuzzy set theory in a learning method

enables us to smooth out these negative effects. Thus, at the basis of our approach we use the Fuzzy Decision Trees (FDT).

Since 2001, each year, the National Institute of Standards and Technology (NIST) organizes the TREC Video Retrieval Evaluation. We will use this framework as basis for our comparison. Although decision trees are widely used as core technology in application using machine learning techniques, no other team participating to the challenge uses them. In fact, simple trees are not as effective when dealing with large unbalanced multiclass data sets [9]. Here, we not only propose to use them, but also to go farther by combining them in a boosting [5] manner, thus obtaining forests of fuzzy decision trees (FFDT).

In this paper, we describe the FFDT algorithm and we study (a) its effectiveness for a high level feature detection task and (b) the effect on performance from size point of view. Since today's approaches are evaluated based on the average precision, measure based on a ranking, we shortly discuss the fundamental difference between classification – what most of the learning algorithm do – and ranking optimization.

The paper is structured as follows: in Section 3, we describe our video processing method, which provides as result low level features. In Section 4 and Section 5 we present Fuzzy Decision Trees and Forests. Finally in Section 6 we detail the high level feature detection experiments and discuss the results.

2 Overview

To use the Fuzzy Decision Trees (FDT) learning method, a training set must be provided in which there are examples of keyframes *with* the high-level feature to be recognized and examples that do *not* possess that feature. For the sake of simplicity, presence of the feature is the *class descriptor*.

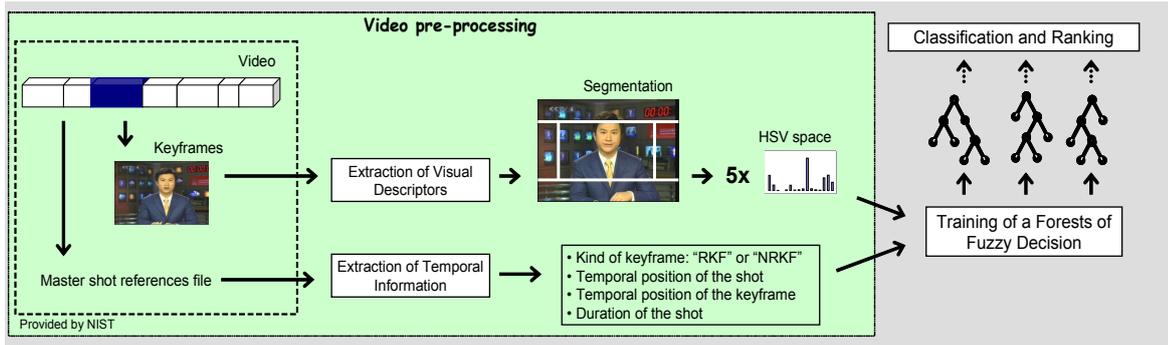The general schema of the application is illustrated in

Figure 1: General schema

Figure 1. Our approach is decomposed in three main stages. First, the video is pre-processed in order to obtain a set of descriptors to feed the machine learning algorithm. Afterwards, in the training step, the system learns how to recognize the presence of a high-level feature in a shot using manually indexed videos (the so-called *development data* set). Finally, in the classification step, the system is able to recognize the presence of a high-level feature in a shot for any additional video.

The training stage is composed of several steps (see Figure 3):

Step 1 the video is segmented into temporal shots; each shot is associated with a representative keyframe;

Step 2 a set of descriptors is extracted from each keyframe of the whole set of keyframes; keyframes from the development data set are also associated with some high-level features (classes) obtained by means of the manual indexation of the videos;

Step 3 several training sets of keyframe descriptors (and the associated class) are built;

Step 4 each of these training sets is used to construct a fuzzy decision tree (FDT) and thus obtaining a forest.

The classification stage is decomposed into the following steps (see Figure 4):

Step 1 each shot to be classified is associated with a representative keyframe;

Step 2 a set of descriptors is extracted from that keyframe;

Step 3 the set is classified by means of each FDT of the forest;

Step 4 the presence of a high level feature is valued by aggregating the classification results given by all the trees of the forest.

3 Video pre-processing

In order to feed the data mining algorithm, the video (MPEG file) has to be pre-processed. A set of numer-ical descriptors, on which the algorithm will learn, has to be defined to characterize the video.

3.1 Video segmentation

First, the video is segmented into a set of shots. We used the results of the segmentation of a video into shots provided by [12] for the TRECVID challenge. Each video of the corpus is described by:

- a keyframe for each detected shot.

- an XML file that provides the time codes of the detected shots, their duration, and the time codes of the keyframes.

To reduce the number of the shots, all shots shorter than 2 seconds are automatically merged with their neighbours to provide a unique shot. A representative keyframe (RKF) of this new shot is selected among the extracted keyframes. The other keyframes are kept and are called non representative keyframe (NRKF). Thus, at the end, each shot is always associated with a unique RKF, but can also be associated with several NRKF.

3.2 Visual Information Descriptors

Visual Information Descriptors are at the basis of the learning process. They are obtained directly and exclusively from the keyframes.

In order to obtain spatial-related information, the image is segmented into 5 regions (see Figure 2) in order to isolate important descriptors, thus helping the learning algorithm to focus on the discriminative variables.

Each of the regions corresponds to a spatial part of the keyframe: top, bottom, left, right, and middle. The five regions are not of the same size, thus reflecting a visual importance of the contained information based on its position.

Afterwards, for each region, the associated histogram in the HSV space is computed. Based on the importance of the region, the histogram is computed in a more or less precise way, by varying the number of bins: 6x3x3 for Middle, and Bottom, 4x3x3 for Right, and 4x2x2 for Left, and Top.

At the end of this procedure, the visual information descriptors are a set of numerical values (belonging to $[0, 1]$) that characterizes every keyframe.

The choice of the number of regions and the number of bins for the histogram deserve further optimization. Moreover, more complementary visual descriptors could be added in order to enhance the possibilities of choice of the learning algorithm (FDT) for its decisions.

Figure 2: Spatial segmentation of a Keyframe

3.3 Temporal Information Descriptors

The *temporal information descriptors* are extracted from the shot detection process [12] and are linked to the sequence of frames in the video. Every keyframe is associated with the following information:

- the kind of the keyframe: representative (RKF) or non representative (NRKF);

- the temporal position of the shot containing the keyframe: time code of the beginning;

- the temporal position of the keyframe in the shot: relative time code since the beginning of the shot;

- the duration of the shot containing the keyframe.

3.4 Class Descriptor

The *class descriptor* is obtained from the manual indexation of the video. It corresponds to the feature(s) to be detected in a shot, as for instance sports, weather, desert, corporate leader, US flag, chart, etc.

The class descriptor is extracted from the file obtained from the collaborative work of indexation of the development video set. Note that a keyframe can be associated with more than one class descriptor depending on the result of the indexation process. We just choose the first class descriptor available, but clearly more refined selection should be done at this stage. Further research will address this point.

4 Fuzzy Decision Trees

Classical decision tree algorithms [3, 13] are one of the inductive learning algorithms that are most intensively used in data mining. Unfortunately they encounter technical problems when dealing with numerical attributes. That leads to the introduction of the fuzzy decision tree construction algorithms enabling the use of fuzzy values in the decision tree [10]. "Fuzziness" allows the decisions to be smoother, avoiding sharp thresholds. It also enables to have degrees of decision and of membership to a certain class.

Inductive learning rises from the *particular* to the *general*. A tree is built from its root to its leaves, by successive partitioning the training set into subsets. Each partition is done by means of a test on an attribute, which leads to the definition of a node of the tree.

Let us assume that a set of classes $C = \{c_1, ..., c_K\}$, representing a physical or a conceptual phenomenon, is considered. And that this phenomenon is described by means of a set of attributes $\mathcal{A} = \{A_1, ..., A_N\}$. In that case, a *description* is a N-tuple of attribute-value pairs (A_j, v_{jl}). Each description is linked with a particular class c_k from C to make up an *instance* (or *example*, or *case*) e_i of the phenomenon. Finally, the inductive learning is the process that generalizes from a *training set* $\mathcal{E} = \{e_1, ..., e_n\}$ of examples to a general law to bring out relations between descriptions and classes in C. In our case, each attribute A_j can take a fuzzy, numerical, or symbolic value v_{jl} in the set $\{v_{j1}, ..., v_{jm_j}\}$ of all possible values. We suppose that v_{jl} is associated with a membership function $\mu_{v_{jl}}$. Similarly, each c_k is supposed to be associated with a membership function μ_{c_k}.

4.1 Attribute Selection

Most algorithms designed for constructing decision trees proceed in the same way: the so-called *Top Down Induction of Decision Tree* (TDIDT) method. They build a tree from the root to the leaves, by successive partitioning the training set into subsets. Each partition is done by means of a test on one attribute and leads to the definition of a node of the tree. The attribute is selected by means of a *measure of discrimination H*. Such a measure enables us to order

the attributes according to increasing discrimination-accuracy when splitting the training set. The discrimination power of each attribute in $\mathcal{A}$ is valued with regard to the classes. The attribute with the highest discriminating power is selected to construct a node.

4.2 Construction of Fuzzy Partitions

The process of construction of FDT is based on the fuzzy partition for each numerical attribute. However, it is rare to know, a priori, such a fuzzy partition. Thus an automatic method of construction such a partition from a set of precise values was implemented. In this way we obtain a set of fuzzy values for each numerical attribute.

The algorithm [8] is based on the utilization of the mathematical morphology theory. Kernels of concordant values of a numerical attribute related to the values of the class can be found. Fuzzy values induced from a set of numerical values of an attribute are linked with the repartition of the values of the class related to the numerical attribute. Thus a contextual partitioning of an attribute is performed, enabling us to obtain the best partition related to that attribute with respect to the class.

4.3 Classification using Fuzzy Decision Tree

It is well-known that the path from the root to a leaf in a decision tree is equivalent to a production rule [7]. The premises for such a rule r are tests on attributes values, and the conclusion is the value of the class that labels the leaf of the path:

$$\texttt{if } A_{l_1} = v_{l_1} \texttt{ and ...and } A_{l_p} = v_{l_p} \texttt{ then } C = c_k$$

In a FDT, a leaf can be labelled by a set of values $\{c_1, ..., c_K\}$ for the class, each value c_j associated with a weight computed during the learning phase. Thus, a path of a fuzzy decision tree is equivalent to the following rule:

$$\begin{aligned}&\texttt{if } A_{l_1} = v_{l_1} \texttt{ and ...and } A_{l_p} = v_{l_p} \texttt{ then}\\ &C = c_1 \texttt{ with the degree } P^*(c_1|(v_{l_1}, v_{l_2}, ...v_{l_p}))\\ &\texttt{and ... and } C = c_K \texttt{ with the degree}\\ &\qquad P^*(c_K|(v_{l_1}, v_{l_2}, ...v_{l_p}))\end{aligned}$$

In a FDT, each value v_i can be either precise or fuzzy, and is described by means of a membership function μ_{v_i}. When a keyframe k, described by means of a set of values $\{A_1 = w_1; ...; A_n = w_n\}$, must be classified, its description is compared with the premises of the rule r, by looking to the degree with which the observed value w is near to the edge value v. This proximity is valued as $\texttt{Deg}(w, v)$. In our case, the value w is a precise value and we have $\texttt{Deg}(w, v) = \mu_v(w)$.

For each premise, $\texttt{Deg}(w_{l_i}, v_{l_i})$ is valued for the corresponding value w_{l_i}. Finally, given the rule r, the keyframe k is associated with the class c_j with a *final degree* $\texttt{Fdeg}_r(c_j)$ that is valued as the aggregation of all the degrees $\texttt{Deg}(w_{l_i}, v_{l_i})$ by means of the *minimum*:

$$\texttt{Fdeg}_r(c_j) = \min_{i\,=\,1...p} \texttt{Deg}(w_{l_i}, v_{l_i}).P^*(c_j|(v_{l_1}, v_{l_2}, ...v_{l_p}))$$

For each class c_j, the keyframe k is associated with the membership degree $\texttt{Fdeg}(c_j)$, from $[0, 1]$, computed based on the whole set of rules. If n_ρ is the number of rules given by the fuzzy decision tree:

$$\texttt{Fdeg}(c_j) = \max_{r\,=\,1...n_\rho} \texttt{Fdeg}_r(c_j)$$

The predicted class c_k associated with k can be chosen as the class with the highest $\texttt{Fdeg}(c_k)$.

4.4 The Salammbô Software

The construction and the use of the FDT was done by means of the Salammbô software. This software was developed for building FDT efficiently and it enables to test several kinds of parameters of the FDT [10]. Moreover, the automatic method to build a fuzzy partition on the set of values of a numerical attribute, mentioned above, was implemented enabling us to avoid the prior definition of fuzzy values of attributes.
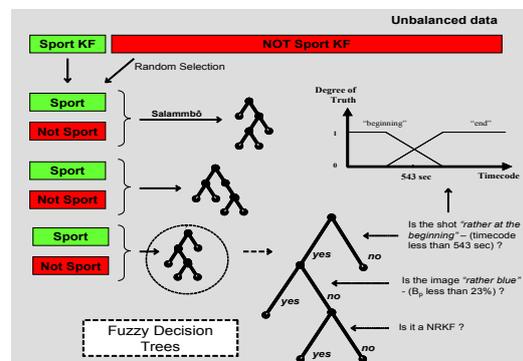

Figure 3: Growing a Forest of Fuzzy Decision Trees

5 Forests of Fuzzy Decision Trees

The use of Forests of Fuzzy Decision Trees (FFDT) is crucial when we have large, unbalanced, multiclass data sets [9]. In fact, in these cases a unique decision tree is extremely dependent on the data chosen for its construction and thus the result can be unstable and the tree generalises poorly.

Several previous works have explored the use of forests of decision trees. Generally, the decision trees of the forest are constructed classically, and they are used
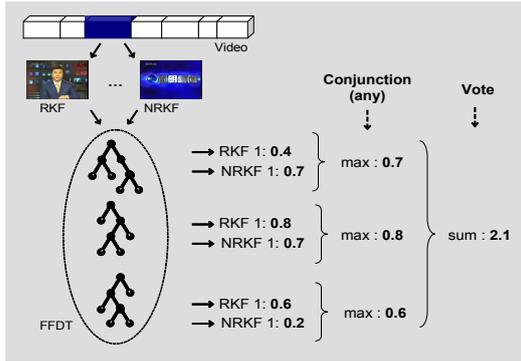
Figure 4: Fuzzy Classification and ranking using a Forest of FDT

to classify cases either classically [2, 6], or by means of the fuzzy set theory [4]. Forests of fuzzy decision trees with a fuzzy-based construction of the trees and a fuzzy classification of new cases have been proposed in [9].

A forest is composed of a given number n of Fuzzy Decision Trees. Each FDT F_i of the forest is constructed from a training set T_i. Each training set T_i is a random sample of the whole training set, as described hereafter.

Although the FDT can handle several classes simultaneously, in domains where a great number of classes exist, we decompose the problem by constructing a forest for each single class. The number of forest is thus the number of classes. For instance, if the aim is to associate each case e with one of the three classes c_1, c_2, and c_3, we construct a forest to recognize if e can be associated with c_1 or not, another forest to recognize if e can be associated with c_2 or not, an a last one to recognize if e can be associated with c_3 or not. Thus, here a forest is dedicated to the recognition of a single high level feature class and is composed of fuzzy decision trees that classify into a binary class (yes or no).

In order to use the Fuzzy Decision Trees learning method we need two training sets, one with keyframes that contain the feature to be recognized and another one with keyframes that do *not* possess that feature (see Figure 3). Since the FDTs are based on a measure estimating the quality of a decision, the two classes need to be equal in number of cases. For example, if one class outnumbers the other one, the best decision will be to always classify an example as being part of the majority class.

Thus, to have a valid training set for the construction of a FDT, we have to balance the number of keyframes of each class by (randomly) selecting a subset of the whole development data set with an equal number of

cases in each class.

By repeating the random selection of examples and each time building a FDT, we obtain a robust forest classifier that are able to cover the description space of all training examples.

5.1 Classification with a forest

After the construction of the FDT as explained previously (Section 4.3), each FDT is used to classify the whole test set of keyframes. Then, by means of the classification, each keyframe k from the test set is associated with a membership degree $\mathtt{FDeg}(c)$ to each class c.

With a forest of n FDTs, corresponding to a single class to be recognized, the classification of a keyframe k is performed in two steps (see Figure 4):

1. classification of k by means of the n FDT of the forest: k is classified with each FDT F_i in order to obtain a degree $\mathtt{Fdeg}(c)$ of k to belong to the class c. We denote $d_i(k) = \mathtt{Fdeg}(c)$ the degree given by F_i for k.

2. sum (or average) of the $d_i(k)$, $i = 1, ..., n$ degrees for each k in order to obtain a single value $d(k) = \sum_{i=1}^{n} d_i(k)$, which corresponds to the degree for the forest to believe that the k contains the feature. The higher $d(k)$, the higher it is believed that k contains the corresponding feature.

5.2 Ranking of the test shots

Finally, shots are ranked according to a degree $D(S)$: the higher $D(S)$, the higher the FFDT believed that S contains the corresponding feature.

Since only high-level features in shot are considered, the degrees of all the keyframes (RKF and NRKF) that pertained to the same shot S are aggregated to obtain the degree $D(S)$. We considers that a shot contains a given feature if at least one of its keyframes contains the feature. Therefore, the degree $D(S)$ for the shot S containing the feature is $D(S) = \max_{k \in S}(d(k))$.

6 Learning High-level Features

In other to compare our approach to others we participated to the high-level feature extraction task at TRECVID 2006 [11]. The aim of that task is to propose, for each high-level feature, a ranking of at most 2000 shots that contain it. The addressed features (and their identification number) are: sports (1), weather (3), office (5), meeting (6), desert (10), mountain (12), waterscape/waterfront (17), corporate leader (22), police security (23), military personnel
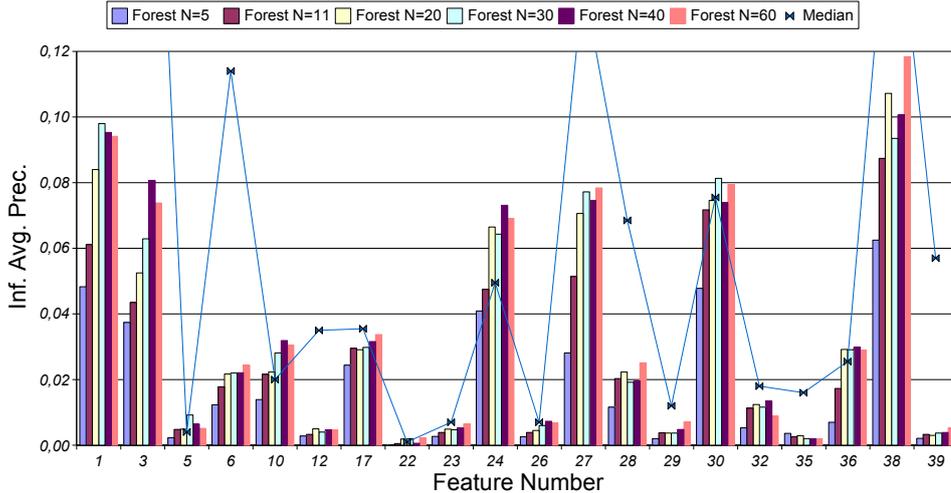
Figure 5: Importance of the size of the forest

(24), animal (26), computer TV screen (27), US flag (28), airplane (29), car (30), truck (32), people marching (35), explosion fire (36), maps (38), and charts (39).

For this study forests of 5, 11, 20, 30, 40, and 60 Fuzzy Decision Trees were built and compared between each other for each high-level feature.

6.1 Experimental roadmap

The TRECVID video corpus [11] is composed of:

- the *development data*: used to construct the system. It is composed of 137 video news (recorded in November 2004), 30mn length in average. These videos were segmented into shots and are associated with an XML file that gives information about the shots (time code, duration, etc.). Each shot is associated with a representative keyframe and, possibly, a set of non representative keyframes. Each keyframe has been (manually) indexed by one or more high level feature. The development data is composed of around 74500 keyframes (*devel* keyframes).

- the *test data*: used to evaluate the system. It is composed of 259 video news (recorded in November and December 2005), 30mn length in average. As in the development data, these videos have also been segmented into shots and are associated with an XML file that gives information about the shots (timecode, duration, etc.). Each shot is associated with a representative keyframe and, possibly, a set of non representative keyframes. None of the keyframes is indexed. The test data is constituted by around 146000 keyframes (*test*

keyframes).

The set of training keyframes is highly unbalanced. For instance, for the Sports feature there are 60066 keyframes without the Sports feature, while only 1570 keyframes with it. As stated in Section 5 this problem can be solved by sampling the data set. The size of the training set was limited to 5000 keyframes at most, with as many keyframes in each class.

All runs are evaluated by means of the Inferred Average Precision (InfAP) [11], and the number of hits at several depths of the ranking (first 100, first 1000, all 2000). These values were computed by means of the software provide by the NIST and the reference file published after the competition. In order to provide an idea of the degree of complexity for the detection of each feature, the median InfAP from the 88 submitted results that were sent for evaluation to TRECVID 2006 is also shown.

6.2 Fuzzy Decision Forest vs other methods

Even though our descriptors are rather simple and incomplete, by looking Figure 5, we observe that the performance in average of the FFDT is around the median of all submitted runs.

We observe that the features that relatively to others seem to work better, in decreasing order, are: maps (38), sports (1), car (30), computer TV screen (27), military personnel (24), and weather (3). It is interesting to notice that (a) while map detection performs well, chart detection performs poorly and (b) that besides military personnel detection these features seem to be easily characterized by visual descriptors, which is coherent with our approach.
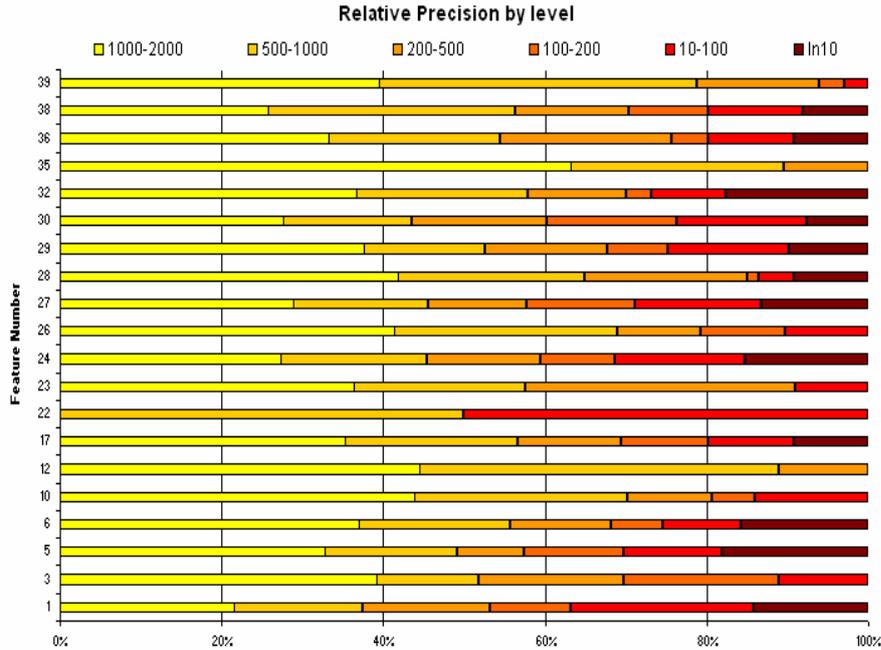
Figure 6: Classification vs ranking

If we compare our approach to (the median of) the others, and only by looking the features where some reasonable results were obtained by all the participants, we observe that our approach is particularly interesting for the detection of the features military personnel (24), desert (10), car (30) and waterscape/waterfront (17). And that our approach is relatively weak for weather (3), meeting (6) and sport (1). We guess that the reason for this is, in the one hand the existence of specialized systems for these features and, in the other hand, the simplicity of our visual descriptors. Further works will address these precises questions.

6.3 Size of the Forests

By looking on the overall performance (Figure 5) and to the per feature results, we clearly remark that by increasing the number of FDTs of a forest we improve the results. This confirms the hypothesis that FFDT is a suitable technique for covering large, unbalanced, multiclass data sets.

However, we observe that there is a limit in the number of FDT for a forest. In fact, too many classifiers lead to an over specialization to the development data, implying a loss in the generalization power.

6.4 Classification vs Ranking

Our approach performs better (relatively to others), when looking at large recall list (e.g. all the 2000 ranked shots), than when looking at the top of the

list (e.g. shots ranked within the first 100). In Figure 6, we show the distribution of good classified shots in relation with their ranking.

Although the use of a forest increases the accuracy of the values and the overall ranking with respect to a single FDT classifier, it appears that the improvement is still not sufficient and uniform enough. In fact, classification algorithms like decision trees do not optimize the ranking, but only the decision concerning the class.

In other words, a classification algorithm like decision trees will try to keep all degrees above a certain (decision) threshold rather that focusing on the fact that the higher the degrees have to be more certainly in the class. In terms of error, in classification, it should be avoided that correct examples are classified under a certain threshold, while in ranking optimisation and error at high a the degree of certainty should cost more than at a lower one.

The presence of false positives in the classification (examples wrongly classified as being in the class) will degrade the ranking because both true positives (examples perfectly classified in the class) and false positives will have the same degree of classification, the sorting will not garantee that true positives will be on the top of the ranking. However, the classification garantees that "in a sufficiently large amount" of examples, the precision will be good.

However, in Figure 6, for each feature, the relative precision by level is presented. Ideally, a good rank-

ing system should provide 100% of the true positives within the top positions of the ranking. With a forest, it can be seen that in general, the global precision for a feature is given mostly by the true positives ranked after the 1000th rank in the ordered shots. Supporting the hypothesis that a classifier is able to detect the presence of a feature in keyframes, does not automatically imply that it can sort them on the presence criteria.

For some rare features (truck (32), military personnel (24), meeting (6), office (5), sports (1)), the number of relative true positives found within the first 10 ranked is very high. On the contrary, for the features people marching (35) and mountain (12), the true positives are never ranked within the 500 first of the list.

The direct learning of ranking is an interesting alternative and is today a new field of research [1].

7 Conclusion

Our aim is to build a system that can automatically recognize a set of high level features for each shot in a video. The basis of our approach is to use a forest of fuzzy decision trees to construct the heart of an automatic system, in order to reduce the need of human usage in the process of indexation.

Although decision trees are very popular as key technology in several application domains, video indexation seems to be an exception. And even if our low level features are rather simple, when comparing our approach to others on the basis of the TRECVID 2006 challenge, we notice that the FFDTs have performance close to the median.

Combining several classifiers (here FDTs into a FFDT) improve the results and this especially since we address a large, unbalanced, multiclass video data collection. There is a performance limit due to overfitting.

Some high level features are "easier" to learn than others. It seems to be a strong correlation between the quality of the results and how well a feature can be described visually. But further research should be done in this direction, in particular by studying the homogeneity of the visual classes.

On the final analysis, there is strong difference between classifications – what most machine learning algorithms do – and ranking optimization. In fact, in the classification case we maximise the accuracy of the decision and not the degree of confidence of this decision. While combining several classifiers through a vote (sum of the degrees) seems to improve the exactness of the confidence degree, the direct learning of the ranking is an alternative promising research field.

References

[1] M. R. Amini, A. Tombros, N. Usunier, and M. Lalmas. Learning-based summarisation of xml documents. *Jour. of Information Retrieval*, 2007, *to appear*.

[2] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[3] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification And Regression Trees*. Chapman and Hall, New York, 1984.

[4] K. Crockett, Z. Bandar, and D. McLean. Growing a fuzzy decision forest. In *Proc. of the 10th IEEE Int. Conf. on Fuzzy Systems*, pages 614–617, 2001.

[5] Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Jour. of Machine Learning Research*, 4:933–969, 2003.

[6] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[7] C. Marsala. Fuzzy decision trees to help flexible querying. *Kybernetika*, 36(6):689–705, 2000.

[8] C. Marsala and B. Bouchon-Meunier. Fuzzy partioning using mathematical morphology in a learning scheme. In *Proc. of the 5th IEEE Int. Conf. on Fuzzy Systems*, pages 1512–1517, New Orleans, USA, September 1996.

[9] C. Marsala and B. Bouchon-Meunier. Forest of fuzzy decision trees. In M. Mareš, R. Mesiar, V. Novák, J. Ramik, and A. Stupňanová, editors, *Proc. of the 7th IFSA World Congress*, pages 369–374, Prague, Czech Republic, June 1997.

[10] C. Marsala and B. Bouchon-Meunier. An adaptable system to construct fuzzy decision trees. In *Proc. of the NAFIPS'99*, pages 223–227, New York, USA, 1999.

[11] NIST. Guidelines for the TRECVID 2006 evaluation - National Institute of Standards and Technology, 2006. http://www-nlpir.nist.gov/projects/tv2006/tv2006.html.

[12] C. Petersohn. Fraunhofer HHI at TRECVID 2004: Shot boundary detection system. Technical report, TREC Video Retrieval Evaluation Online Proceedings, TRECVID, 2004. http://www-nlpir.nist.gov/projects/tvpubs/tvpapers04/fraunhofer.pdf.

[13] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):86–106, 1986.