VIDEO FEATURE SELECTION USING FAST-CONVERGING SORT-MERGE TREE

Yan Liu and John R. Kender

Department of Computer Science
Columbia University
New York, NY 10027
{liuyan, jrk}@cs.columbia.edu

ABSTRACT

High time complexity is a bottle-neck in video segmentation, classification, analysis, and retrieval. In this paper we use a heuristic method called Fast-converging Sort-Merge Tree (FSMT) to construct automatically a hierarchy of small subsets of features that are progressively more useful for video data exploration. The method combines the virtues of a wrapper model approach for high accuracy, with those of a filter method approach for deriving the appropriate features quickly. FSMT speeds up a more fundamental method, the Basic Sort-Merge Tree (BSMT) approach, while retaining its performance. We demonstrate FSMT's high accuracy: it has a 0.001 error rate in a frame classification task on 75 minutes of instructional video, and a 0.98 precision and 0.89 recall in a segment retrieval task on 30 minutes of sports video. Additionally, FSMT is more than 80% faster than its predecessor, BSMT.

1. INTRODUCTION

The rapid growth and wide application of digital video has led to a significant need for efficient video data management. To reach these semantic goals, some machine learning methods such as classification and boosting have been attempted, in order to find features that associate image properties with user labels. But due to the high volume of video data, the time complexity of these methods has been prohibitively high. Researchers therefore have worked on speeding up their algorithms. One way has been by seeking efficient ways of reducing the dimensionality of the data prior to processing from the view of image processing and computer vision. These researchers have assumed that some features, such as color histograms or texture energies, are more useful than others, based solely on their intuition. They provide theoretical analyses and empirical validations for their choices, but this approach is difficult to extend to other domains because of need for human interaction.

The problem of feature selection has received significant attention in the Artificial Intelligence literature recently, and various algorithms have been devised and applied to moderately large data sets in application domains like text categorization and genomic microarray analysis. However, learning research is not often carried out in the video domain since existing feature selection algorithms, designed for much smaller databases, run an inordinately long time. The heart of this paper is an automatic feature selection algorithm, called the Fast-converging Sort-Merge Tree (FSMT). It has low time complexity, and it does not require any manual definition or construction of features.

This paper is organized as follows. Some related work in feature selection is introduced in Section 2. Section 3 proposes the feature selection algorithm, FSMT, and provides a framework for video analysis using this algorithm. Section 4 presents empirical validation of the accuracy and efficiency of algorithm when applied to the particular genre of instructional videos. The paper closes with discussion in Section 5.

2. RELATED WORK

2.1. Filter methods and wrapper methods

There appears to be two major approaches to the feature selection problem. The first emphasizes the discovery of any relevant relationships between the features and the concept, whereas the second explicitly seeks a feature subset that minimizes prediction error of the concept. The first is referred to as a filter method, and the second approach is referred to as a wrapper method. In general, wrapper methods attempt to directly optimize the classifier performance so that they can perform better than filter algorithms, but they require more computation time. Seen in this context, this paper proposes a wrapper feature selection method with time cost considerably less than that of filter methods.

2.2. Feature selection algorithm design and evaluation

Feature selection methods are typically designed and evaluated with respect to the accuracy and cost of their

three components: their search algorithm; their statistical relationship method, in the case of filter methods, or their induction algorithm, in the case of wrapper methods; and their evaluation metric---which is simply prediction error in the case of wrapper methods. The dominating cost of any method, however, is that of the search algorithm, since feature selection is fundamentally a question of choosing one specific subset of features from the full power set of features. So far, at least three general kinds of heuristic search algorithms have been used: forward selection, backward elimination, and genetic algorithms.

2.3. Basic Sort-Merge feature selection algorithm

Liu and Kender have proposed a Basic Sort-Merge Tree (BSMT) method, which exploits several properties unique to video data in order to induce appropriate but small feature sets with relatively low time cost [1]. BSMT combines the features of forward selection, backward elimination, and genetic algorithms. In order to avoid irrevocable adding or subtracting candidate features to or from the evolving feature set, it always operates on some representation of the original feature space. Therefore, at each step every feature has an opportunity to impact the selection. To avoid heuristic randomness, at each step a greedy algorithm is used to govern subset formation. Further, the recursive nature of the method provides an additional advantage, in that it enables the straightforward creation of near-optimal feature subsets of any or all desired cardinalities or accuracies, with little additional work.

Initialize level = 1
    N singleton feature subsets.
While level < $\log_2 N$
    Induce on every feature subset.
    Sort subsets based on their classification accuracy.
    Combine, pairwise, feature subsets.

Table 1. BSMT feature selection algorithm.

BSMT can be divided into two parts: the creation of a tree of feature subsets, and the manipulation of the tree to create a feature subset of desired cardinality or accuracy. Each part uses a heuristic greedy method. Table 1 shows the algorithm for setting up the tree. Table 2 shows the algorithm of cutting the tree, based on the application requirement. For example, Table 2 would be used to create a feature space with exactly r features, if r is not a power of 2. The performance of a wrapper feature selection algorithm not only depends on the search method, but also on the induction algorithm. For the induction method during the course of the learning, the BSMT method uses a novel combination of Fastmap for dimensionality reduction [2] with the Mahalanobis maximum likelihood measure for classification. Both of theses are well-suited to known statistical properties of video feature sets. Their time complexity are both linear in the number of training data m. We refer readers to the literature for a detailed explanation of these methods.

Select the leftmost branch of size $2^{\lfloor \log 2r \rfloor}$.
Initialize cutout = $2^{\lceil \log 2\, r \rceil}$ - r.
While cutout >0
    Let branch-size = $2^{\lfloor \log 2\ \text{cutout} \rfloor}$.
    For all remaining branches of this size, evaluate the induction result of removing those branches individually.
    Remove the branch with best result.
    Let cutout = cutout – branch-size.

Table 2. Algorithm to select exactly r features from BSMT.

3. VIDEO ANALYSIS USING FAST-CONVERGING SORT-MERGE FEATURE SELECTION TREE

It is not hard to show that the time cost of the search algorithm of BSMT is linear in the number of nodes in the Sort-Merge tree, i.e., $T = O(2*N*T_m)$, where T_m is the induction time complexity using m training data. It is therefore also clear that the time of induction dominates the total time cost. Therefore, we designed the Fast-converging Sort-Merge Tree (FSMT) method to reduce time complexity by dynamically reducing the number of nodes in the tree as it is created from the leaf level on upwards.

3.1. Fast-converging Sort-Merge Tree

The BSMT algorithm sets up the entire feature selection tree even if a full tree is not necessary to extract r features based on the user's requirement. Usually, r is small with respect to N. So, to save time without sacrificing accuracy, FSMT only sets up selected parts of the feature selection tree, based on several heuristic parameters: the number of selected features r, a vector of tree-trimming convergence rates for each level $\mathbf{V} = (v_1, v_2, \ldots v_{\lceil \log 2r \rceil + 1})$, and an evaluation metric G.

Although FSMT also produces a hierarchy of feature subsets with increasing cardinality as one nears the root,

in contrast to BSMT the root of the Fast-converging Sort-Merge Tree consists not of one feature subset with all N features, but of one feature subset with only the requested r features. FSMT does not therefore attempt to select feature subsets in full generality of cardinality, but is optimized for a specific user request. To do so, the vector V controls the convergence speed of FSMT, or, more precisely, the amount that each successive level of feature subsets is pruned. Usually this pruning rate is constant and usually each successive level is a fixed fraction of the one below it, although other convergence rates are also possible.

Additionally, we do not simply use the prediction error of the resulting classifier as an evaluation metric of feature subset quality as in BSMT; such a metric is coarse and results in many ties. We use instead the information gain of the resulting classifier of the feature subsets at each level, a method that naturally breaks ties in a useful way. (Other tie-breaking metrics are also possible). This metric, G(C,F), calculates the reduction of entropy in classifying C categories using the feature subset F. Table 3 shows the algorithm for setting up the FSMT. Figure 1 illustrates a tree with N=1800, r=16, a constant convergence rate of 0.5, and information gain as the tie-breaker. The number of inductions is only 682 using FSMT, compared with 4095 using BSMT.

```
Initialize level = 0
        N singleton feature subsets.
        Calculate R: number of features of each level
        based on V and r.
While level <  log2 r +1
        Induce on every feature subset.
        Sort subsets based on G(C, F).
        Prune the level based on R.
        Combine, pair-wise, feature subsets form
        those remaining.
```

Table 3. FSMT feature selection algorithm.

3.2. Framework of video analysis using FSMT

We apply the FSMT method to video data in the following way. First, we calculate the number of features subsets on each level, based on V and r, and store them in the vector R; if the convergence is uniform, this is straightforward. Then, we down-sample the video temporally and spatially to get the original feature space. Each feature is first placed into its own subset to initialize the sort-merge process. Next, using Fastmap, the dimensionality of each feature subset is reduced to a pre-specified small number, c, of dimensions. Then, for each feature subset at this level, using the reduced dimensionality representation, the training frames of the video train a Mahalanobis classifier to classify the test frames of the video. Feature subsets are sorted by performance on the test data, but since the number of subsets is often much larger than the number of test frames, many subsets tie in performance. So, the G(C, F) of each feature subset is computed and feature subsets with identical performance are further sorted by their information gain. The fully sorted feature subsets are now merged pair-wise and in order. The process repeats again, starting at the Fastmap step, for each level until the root of the tree, is reached, where there is one feature subset with r features. This feature subset can then be used as the basis for classification, segmentation, or retrieval.

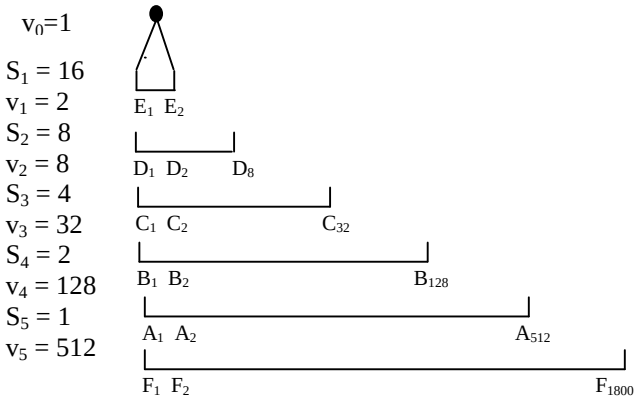
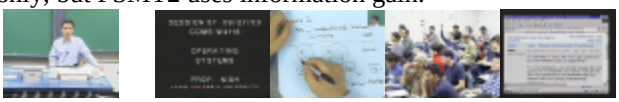


Figure 1. FSMT for N=1800,r=16, constant convergence of 0.5, and levels j, from 5 down to 1. S_j is the size of the feature subset at level j, and v_j is the number of such features subsets.

4. EXPERIMENT

First, we illustrate FSMT on an extended instructional video of 75 minutes duration in MPEG-1 format. Our task is to retrieve ``announcement" frames like those of Figure 2(a) from amongst all other frames, like those in Figure 2(b). To begin, we down-sample the video temporally using only every other I frame, and we spatially subsample by only using the DC terms of each macroblock of the I frame. This gives a total 4500 data vectors, each with 1800 features. Figure 3 displays the video retrieval error rate on this task when FSMT is set to terminate with r=16 features. For a comparison task, we used random feature selection, since virtually all other published feature selection algorithms ran interminably long on this data. We repeated the random selection

alternative 100 times, and show their average error rate. The difference of FSMT1 and FSMT2 is only the evaluation metric: FSMT1 like BSMT uses error rate only, but FSMT2 uses information gain.



(a) (b)

Figure 2. Task: Retrieve "announcements"(a) from an entire video with competing image types(b).

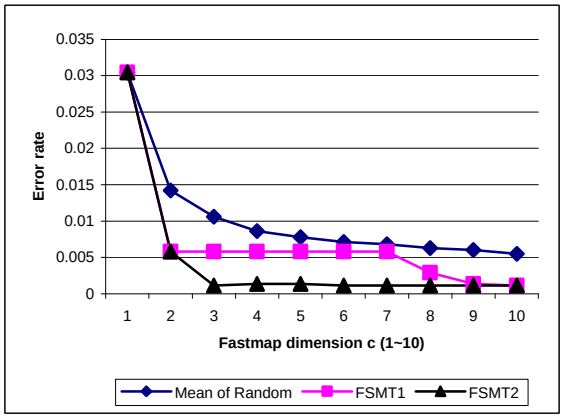
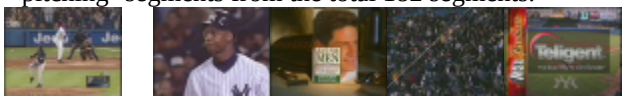


Figure 3. Error rate with same feature subset size (r=16) and different Fastmap dimensions c (from 1 to 10): random, BSMT, FSMT.

The second task applies FSMT to a task of sports video retrieval. As shown in Figure 4, we are interested in defining ``pitching frames" that look like Figure 4(a). The rest of the video has many different competing image types, some of which are shown in Figure 4(b). The data is sampled somewhat more finely, with every I-frame extracted as a data frame, giving 3600 frames for the half-hour. The video has been segmented to 182 segments; a segment is a shots except that commercials are considered to be a single segment. We then attempt to retrieve the 45 ``pitching" segments from the total 182 segments.



(a) (b)

Figure 4. Task: Retrieve "Pitching"(a) from an entire video with competing image types(b).

As mentioned by Lin and Hauptman in [3], simple accuracy is often an insufficient measure of performance, so we compared the feature selection algorithms on this task based on their recall and precision. Table 4 shows the performance of retrieval using features selected randomly, against FSMT with information gain as tie-break. FSMT performs much better than random feature selection, especially under sparse features. For example, only using a feature subset with 2 features from the 1800-dimensional original feature space, the precision is nearly perfect and the recall is near 0.9.

Number of Features	Precision		Recall	
	Random	FSMT	Random	FSMT
2	0.5983	0.9756	0.3067	0.8889
4	0.7117	0.8519	0.2711	0.5111
8	0.7104	0.8205	0.2547	0.7111
16	0.8008	0.9024	0.2689	0.8222
32	0.8648	0.9667	0.2804	0.6444

Table 4. Retrieval performance using different feature selection algorithms (c=2)

5. CONCLUSION

We have presented a low-cost feature selection algorithm that is well-suited for video data. FSMT is able to prune the feature subset tree according to the user's need for exactly r features. Information gain is introduced as an additional evaluation metric to further increase performance. We have demonstrated the results on two different retrieval tasks in extended videos. We intend to investigate the utility of the FSMT method across a larger library of videos of this kind, and also across other genres, such as situation comedies, which share a similar categorization structure of scene classifications.

6. REFERENCE

[1] Yan Liu and John R. Kender. Video frame categorization using Sort-Merge feature selection. In *Proceedings IEEE Workshop on Motion and Video Computing, pages 72--77, 2002.*

[2] Christons Faloutsos and king-Ip (David) Lin"FastMap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets", *Proceedings of ACM SIGMOD*,1995, pp 163-174.

[3] Wei-Hao Lin and Alexander Hauptman, "News video classification using SVM-based multimodal classifiers and combination strategies", *Proceedings of ACM Multimedia 2002, Juan-les-Pins, France, December 1-6*, 2002.

Acknowledgments: This research was supported by NSF grant EIA-00-71954.