Characteristic Pattern Discovery in Videos

Mihir Jain*
Center for Visual Information Technology
International Institute of Information Technology
Hyderabad, PIN 500032, India
mihir@research.iiit.ac.in

C. V. Jawahar
Center for Visual Information Technology
International Institute of Information Technology
Hyderabad, PIN 500032, India
jawahar@iiit.ac.in

ABSTRACT

In this paper, we present an approach to discover characteristic patterns in videos. We characterize the videos based on frequently occurring patterns like scenes, characters, sequence of frames in an unsupervised setting. With our approach, we are able to detect the representative scenes and characters of movies. We also present a method for detecting *video stop-words* in broadcast news videos based on the frequency of occurrence of sequence of frames. These are analogous to stop-words in text classification and search. We employ two different video mining schemes; both aimed at detecting frequent and representative patterns. For one of our mining approaches, we use an efficient frequent pattern mining algorithm over a quantized feature space. Our second approach uses a Random Forest to first represent video data as sequences, and then mine the frequent patterns. We validate the proposed approaches on broadcast news videos and our database of 81 Oscar winning movies.

Keywords

Video mining, frequent pattern mining, random forest

1. INTRODUCTION

Large video repositories are becoming omnipresent. Content based analysis of such collections is challenging. Processing these videos is computationally costly, error prone and difficult to scale up. The necessity of content based access has triggered research in visual recognition with newer data-sets, categories, and computationally efficient methods [15, 20, 21]. Many approaches have been proposed for different problems of video analysis including activity recognition, visual search, movie/sitcoms analysis and visual mining. Most of these methods are supervised and requires labeled examples at some or other level. To make it feasible on large collection of videos, unsupervised and weakly supervised approaches are desired as argued in many of the recent works [9, 24].

In this work, our objective is to mine the videos in order to discover or detect important patterns. We discover characteristic patterns in videos based on frequency of occurrence of scenes, actors and sequence of frames, in an unsupervised setting. With our approach, we are able to detect the representative scene and main characters of movies. Going beyond objects and people, we extend our work to mine frequent video sub-sequences. We define *"video stop-words"* and present a method for detecting them in broadcast news videos. *Video stop-words* are analogous to stop-words in text classification and search. We define *video stop words* based on frequency of occurrence of sub-sequences in videos over the period of time and across different news channels in Section 3.3. Detecting them can assist in removing redundancy in videos.

Movies are fascinating data sets with significant visual variation and diversity. In movies, certain scenes, main characters or objects appear more frequently than others. Characteristic patterns in movies could convey a lot about the visual content and major theme of the video. We aim to discover these patterns directly from the video. The pattern of interest could be individuals, scenes etc. In Section 3.2, we do automatic labeling of characteristic scenes and main actors in movies. Our approach successfully extracts the characteristic patterns (scenes and people) from our movie database. The characteristic scene discovered from the movie database could vary significantly in visual content. Movie characterization through such mining or otherwise can help a great deal in building movie recommendation systems which to date are manual or semi-automatic requiring comprehensive human intervention. Another application is to mine patterns for sociological studies. The techniques are generic and are widely applicable in other category videos.

In broadcast news videos, many events like breaking news and commercials occur repeatedly. Such frequent *items* or *sequence* can be used for automatic characterization and understanding videos. Our goal is to efficiently detect frequently occurring sequence of frames in the news videos. This requires partial or complete matching of frames or sequences of frames of variable lengths from different parts of the videos. It is also desired that the method is robust enough to deal with the situations when sequences are repeated with few extra or fewer frames, but with ordering preserved. The challenge is to detect these sequences very efficiently.

Mining the visual content and thereby characterizing videos,

*Corresponding author

ICVGIP '10, December 12-15, 2010, Chennai, India

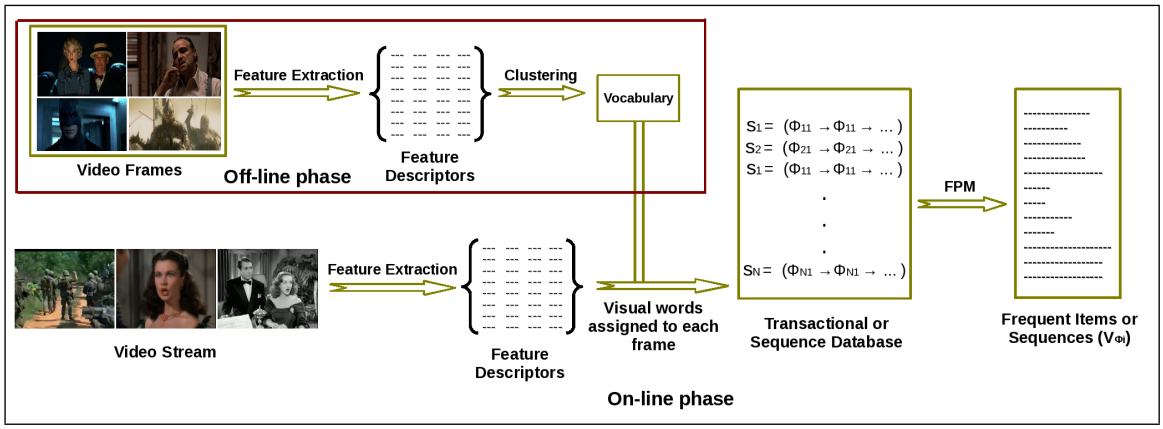

Figure 1: Frequent Pattern Mining in Video: Feature descriptors of frames are quantized to build vocabulary in offline phase. During online processing, video is represented as a transactional (or sequence) database, which is mined using Frequent Pattern Mining algorithms.

has been attempted in the recent past. Sivic and Zisserman proposed a video mining approach in [21] to obtain principal objects, characters and scenes. Frequently occurring spatial configurations of features were found using clustering algorithms, rather than any frequent itemset mining schemes. Very few works have tried to adapt traditional data mining methods for visual data [15, 22]. The objective has been, often, to find the most frequent spatial configuration of points (eg. a building) from large number of video frames. Mining in visual data has been complemented by the processing of associated text (subtitles) [7] and speech [22] components. Our method relates to visual recognition as well as data mining. In this sense, the closest to our work is that of Quack and Gool in [15]. They use *frequent itemset mining* for finding frequently occurring configurations of features for mining frequently occurring objects and scenes from videos. They also apply frequent itemset mining: (a) on instances of a given object class to assist in object detection [16], and (b) for mining object and events from community photo collection [17]. Nowozin *et al.* in [11] introduced discriminative subsequence mining to find optimal discriminative subsequence patterns. Rather than focusing on objects or point configurations, our primary interest is in scene characterization, based on a global set of features. We also design the mining scheme to suite large video collection, as required in our case.

We employ two different video mining schemes; that are aimed at detecting frequent and representative patterns. For one of our mining approaches, we use an efficient frequent pattern mining algorithm over a quantized feature space, as in the case of visual bag of words methods. In our second approach we suggest a sequence representation of videos based on Random Forest [6] and propose to mine frequent sequences.

The remainder of this paper is organized as follows. We explain our two mining approaches in the next section. Then we evaluate and compare these approaches quantitatively in Section 3.1. In Section 3.2, we present our results on movies and show results for discovering characteristic scenes and main characters in movies. In Section 3.3, we define *video stop-words* and present the method to detect them. We demonstrate the accuracy and efficiency of the proposed approach by experimenting on a broadcast news video data.

2. OUR MINING APPROACHES

For mining videos, we represent features in quantized code-books. This has been popular for many recognition, retrieval and classification tasks [4, 14, 20]. Representing video frames using code-books helps in accommodating the uncertainty of the visual description while retaining the essential discriminative information. We employ *Frequent Pattern Mining* (FPM) [2, 3] to extract frequent sequence or items from videos.

In FPM, a set of patterns (transactional database) and minimum support threshold are given. Patterns are some or other form of collection of *items* such as itemsets [2], item sequences, sequences of itemsets [3]. The task is to find all the frequent patterns whose frequency of occurrence is no less than the minimum support threshold. *Frequent Itemset Mining* and *Frequent sequence mining* are special cases of *Frequent Pattern Mining*. In FIM transactions are set of items and in FSM they are sequence of items or itemsets. We say that a transaction supports an itemset (in case of FIM) or sequence (FSM), if itemset is sub-set or sequence is sub-sequence of the transaction. Transactional database is more popularly known as sequence database in case of FSM. Frequent sequence mining [3] has been successfully applied to several large-scale data mining problems such as market basket analysis or query log analysis [2]. Many algorithms have been proposed in the literature for solving FIM as well as FSM such as APriori [2], PrefixSpan [13], SPADE [25] etc.

In our first approach we use an FPM methods over video frames represented based on vocabulary built by K-means clustering. We then propose sequence representation for frames/images using nodes of randomized trees. We consider this representation using Random Forest for following reasons:

- Ensemble of clustering trees are able to find natural clusters in high dimensional spaces [10].
- Random Forest leads to more efficient clustering and less memory usage than k-means based algorithms.
- The existence of an implicit hierarchy in the trees can take care of partial matching of samples.

For large number of trees this sequence becomes very long and can not be mined efficiently using *PrefixSpan* algorithm.

Therefore we propose *Randomized Mining Forest* in Section 2.2 to mine frequent patterns from such sequences. We discuss them in detail in the rest of this section.

2.1 Visual Frequent Pattern Mining

Our approach of mining videos is illustrated in in Figure 1 as the online and offline phases. In offline phase, appropriate features (as per the application) are extracted from example frames and quantized by k-means to build vocabulary. During online phase, input data is assigned the visual words. Each frame or shot represented by visual words makes a transaction (or sequence) and thus transactional (or sequence) database is built. Frequent itemsets or frequent sequences are then mined from it using FIM or FSM algorithms.

We now state the problem of mining frequent sequences in video when frames are represented as items or set of items and shot as a sequence. In other similar cases, for example when frame is itself a sequence or transaction this can be modified accordingly. Let $\mathcal{V} = \{w_1, w_2, ..., w_k\}$ be the visual vocabulary of k visual words. A frame, ϕ , is represented as a visual word or unordered set of visual words, $\phi = (w_1, w_2, w_3, ..., w_m)$ and $\phi \subseteq \mathcal{V}$. A sequence is an ordered list of such frames.

A sequence of frames, $\Phi_\alpha = (\phi_{\alpha 1} \rightarrow \phi_{\alpha 2} \rightarrow ... \rightarrow \phi_{\alpha p})$, is said to be sub-sequence of another sequence $\Phi_\beta = (\phi_{\beta 1} \rightarrow \phi_{\beta 2} \rightarrow ... \rightarrow \phi_{\beta q})$, $\Phi_\alpha \preceq \Phi_\beta$, if there exist integers $1 \leq j_1 < j_2 < ... < j_p \leq q$ such that $\phi_{\alpha 1} \subseteq \phi_{\beta j_1}$, $\phi_{\alpha 2} \subseteq \phi_{\beta j_2}$ ..., $\phi_{\alpha p} \subseteq \phi_{\beta j_p}$. A video is represented as a database of shots. Any sequence Φ is valid if $\Phi \preceq s_i$, $i = 1...N$, where s_i is a shot and N is the number of shots in the video. The relative support of a sequence, Φ, in a video or shot database, D, is the ratio of number of sequences containing Φ to the number sequences present in the database.

$$support_S(\Phi) = \frac{|\{(s_i \in D) | (\Phi \preceq s_i)\}|}{|D|} \in [0, 1] \qquad (1)$$

A frame sequence Φ is called frequent in D if $support_S(\Phi) \geq min\_sup$ where min_sup is a threshold for the relative minimal support.

We obtain frequent sequences of frames by using PrefixSpan method. It is more efficient than APriori based methods for mining sequential patterns [23] and particularly for lower min_sup values. In case of videos even if a sequence of frames repeats for only a few times it would be considered frequent. Therefore we use PrefixSpan algorithm for our purpose of mining frequent sequences of frames in videos in Section 3.3. Corresponding to each frequent sequence, Φ, we have an ordered set of visual words, $V_\Phi = \{w_1, w_2, ..., w_n\}$ and a set of tuples, $M = \{(sid, F) | \Phi \preceq s_{sid}\}$, where sid is shot-id, F is ordered set of frames in the shot and $|M|$ is the absolute support of Φ in the video.

When frame is itself a set of items (or transaction) i.e. no sequential information is used (as in Section 3.2.1) then it is a problem of FIM. The above formulation can be modified accordingly by representing Φ_α as set of items/itemsets and replacing $\preceq$ by $\subseteq$ in equation 1. *APriori* algorithm is used to get frequent itemsets, Φ, and V_Φ and F are orderless.

Now we discuss an alternative and more efficient approach using randomized trees.

2.2 Randomized Trees for Mining Videos

Random Forest was introduced in Machine Learning literature by Breiman [6] for classification and regression, and is shown to be comparable with boosting and support vector machines. They have become very popular in the computer vision community. Many papers have applied them to various classification, segmentation and clustering tasks [5, 10, 18]. Moosmann *et al.* [10] proposed an efficient clustering scheme using randomized decision tree. Shotton *et al.* [18] simultaneously exploit both classification and clustering for segmentation and categorization.

We use ensemble of randomized trees for fast clustering and also make use of tree hierarchies in the way similar to [10, 18]. Each tree is built from root to leaf nodes in an unsupervised manner using a randomly selected subset of the training data. At each node a function that most evenly divides the data is used i.e., each sample is considered to belong to a different class. Entropy at any node N_i with X_i number of sample is given as, $E(N_i) = log(X_i)$.

The node-functions are selected as follows:

- At each internal node (including root) F node-functions are randomly selected. Here we consider 3 types of node-functions: (a) single feature component, (b) difference of two components and (c) linear combination of a few components of the descriptor ([5]).
- For each node-function, we need to determine a threshold that best splits (most evenly) the data reached to this node.
- The combination of node-functions and threshold that gives maximum information gain is the final choice.

When a sample (say a keyframe) is pushed through a tree its path from root to leaf node makes a sequence of nodes. Such sequences of nodes from all the trees are concatenated (Figure 2) to represent the frame as an item sequence. Applying FSM on such a sequential database would give us the frequent set of paths across the trees. Similar frames may not reach the same leaf node or may not have long enough common path or prefix sequence in some trees. Using ensemble of T (around 50) trees handles this as similar frames are expected to have enough number of common paths across the trees. With the proposed sequences (using path not just leaf) partial matching can be taken care of when FSM is applied because of tree hierarchies.

Since we use each node as an item, set of all nodes becomes our vocabulary ($\mathcal{V}$). Each frame is represented as a sequence of such nodes, $n \in \mathcal{V}$. Consider a frequent sequence extracted by FSM from this sequential database,

$$seq_{freq(i)} = \{n^i_{11} \rightarrow n^i_{12} \rightarrow ... \rightarrow n^i_{1L_1} \rightarrow n^i_{21} \rightarrow ... \rightarrow n^i_{TL_T}\} \qquad (2)$$

where n^i_{tj} is j^{th} node in $seq_{freq(i)}$ from t^{th} tree and $n^i_{1L_1}$ is the last node coming from t^{th} tree in $seq_{freq(i)}$. Note that $n^i_{1L_1}$ need not be a leaf node. It is equivalent to represent these frequent sequences by only last nodes:

$$seq_{freq(i)} = \{n^i_{1L_1} \rightarrow n^i_{2L_2} \rightarrow ... \rightarrow n^i_{TL_T}\}$$

So, set of frames supporting frequent sequence $seq_{freq(i)}$ can be given as:

$$F_{Freq}(seq_{freq(i)}) = \{C(n^i_{1L_1}) \cap C(n^i_{2L_2}) \cap ... \cap C(n^i_{TL_T})\} \qquad (3)$$

where $C(n)$ is the set of frames passing through node n. We extract set of all *maximal frequent sequences*, F_{max}. A frequent sequence is maximal if it is not a subsequence of

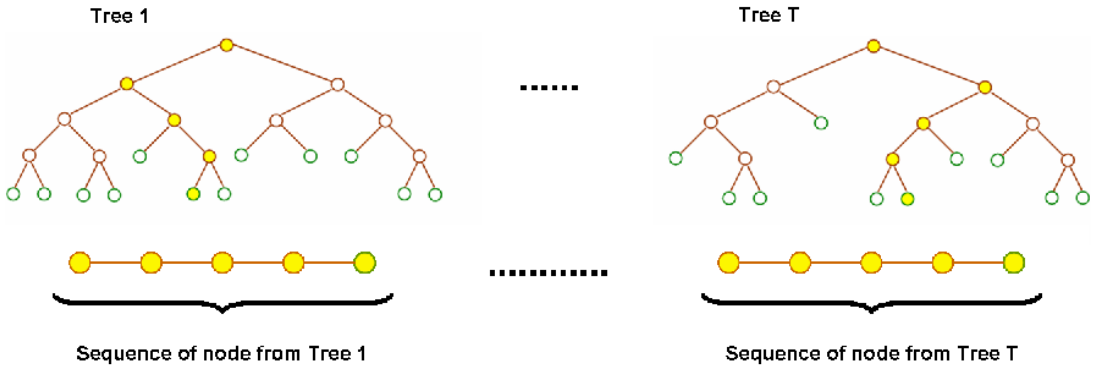

Figure 2: Randomized Mining Forest of T trees built without supervision. Each sample while descending updates the counts of the nodes in each tree. The paths traversed by a sample in each tree, shown in yellow, are concatenated and used as a sequence representation of the sample.

any other frequent sequence. Support of $seq_{freq(i)}$ according to the definition of FSM would be $suppT(seq_{freq(i)}) = \frac{|F_{Freq}(seq_{freq(i)})|}{|D|}$. Support of all sequences in F_{max} has to be greater than minimum support threshold, *minsupp*. Frames supporting any frequent sequence from F_{max} are frequent or characteristic frames. Set of such frames is given by:

$$\Gamma = \bigcup_{F_{max}} F_{Freq}(seq_{freq(i)}) \tag{4}$$

2.2.1 Randomized Mining Forest

With T in range of $50 - 100$, the sequences become too long for PrefixSpan algorithm to compute FSMs efficiently. Computational time exponentially increases with number of trees or length of sequence. We suggest Random Forest based solution to find Frequent frames in a given movie. Keyframes (or features) of a video to be mined are passed through the built forest. The paths followed by each frame and number of samples reaching at each internal and leaf node are stored. We call this ensemble of randomized trees with above details of a given video as a *Randomized Mining Forest* (RMF). Figure 2 illustrates an example of RMF. The node tests are learned from a sub-set of dataset consisting of several videos. Such a set can be thought to have a number of complex classes (of say scenes). When a frame reaches to a node in RMF it belongs to some hypothetical class with some probability. Therefore, the item-sequence generated by RMF can be seen as a sequence of probabilistic items.

We here suggest a method to mine videos and find frequent frames approximately as given by equation 4. In each frequent sequence $seq_{freq(i)}$ we go down the trees after last node ($n^i_{tL_t}$) to some node, $n^i_{tEx_t}$ that has high *depth normalized frequency* (explained below). Thus we get a new sequence $seq^{ext}_{freq(i)} = \{n^i_{1Ex_1} \rightarrow n^i_{2Ex_2} \rightarrow ... \rightarrow n^i_{TEx_T}\}$, where $n^i_{tEx_t} \geq n^i_{tL_t}$. By extending like this or going further down the trees, frames left in the deeper nodes are mutually more similar. Set of these extended sequences are extended maximal frequent sequence. *Depth normalized frequency* of a node n is given as $depF(n) = \frac{|C(n)|}{2^{H-d}}$, where H is height of the tree and d depth of node n. To find n^{ext}, nodes with high $depF$ we start from leaf nodes. In each tree t, M leaf nodes with highest $depF$ are selected at first as a member of set of frequent nodes. Mean *depth normalized frequency* $MdepF$ of the selected nodes is computed. Then we move to the parent of each of the selected nodes and if any of the parent has *depth normalized frequency* greater than $MdepF$ then child is replaced by the parent. This is done iteratively until no parent satisfies the above criteria to get final set of extended frequent nodes $\mathcal{N}$ from all the trees in RMF.

Set of frequent or characteristic frames from the given video can be approximately given as:

$$\Gamma_{Ext} = \bigcup_{n \in \mathcal{N}} C(n) - \{freq(f_j) < \tau | f_j \in \bigcup_{n \in \mathcal{N}} C(n)\} \tag{5}$$

where $freq(f_j)$ number of occurrences of frame f_j in $\mathcal{N}$ which should be greater than τ. We define support of a frequent frame $f_j \in \Gamma_{Ext}$ as:

$$suppT(f_j) = \frac{\bigcup_{n \in \mathcal{N}_{f_j}} C(n)}{|\Gamma_{Ext}|} \tag{6}$$

where $\mathcal{N}_{f_j}$ are those frequent nodes through which f_j passes.

In summary, we approximate equation 4 by equation 5. We go further down from a last node ($n^i_{tL_t}$) to some node with high enough *depF*. This node will be traversed by a subset of frames that reached node, $n^i_{tL_t}$. The frames reaching to the extended node are expected to be mutually more similar and also frequent. The idea is that when we take union of set of frames reaching the extended nodes we get a set of nodes approximately similar to that given by equation 4. We of course take out those frames that do not occur frequently in set of extended nodes ($\mathcal{N}$) in equation 5. We quantitatively evaluate RMF in Section 3.1 and apply for mining characteristic scenes from the movies in Section 3.2.

3. EXPERIMENTS AND RESULTS

In this section, we first quantitatively evaluate the effectiveness of our approaches and compare them on the grounds of accuracy and efficiency. Then we present our experiments on movie and news videos with results for identifying characteristic scenes and main actors, and *video stop-word* detection.

3.1 Quantitative Evaluation of Mining Approaches

The goal of this experiment is to find the frequent object class categories in a given database. Since we do not have the ground-truth for large movie and news datasets, we use

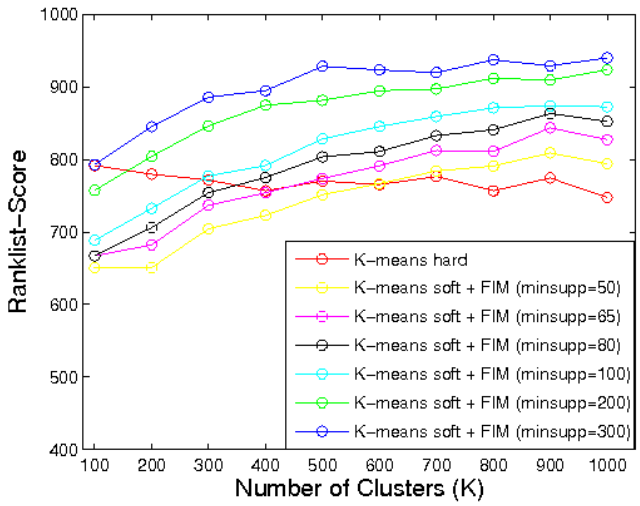

Figure 3: Performance of different approaches for ranking.

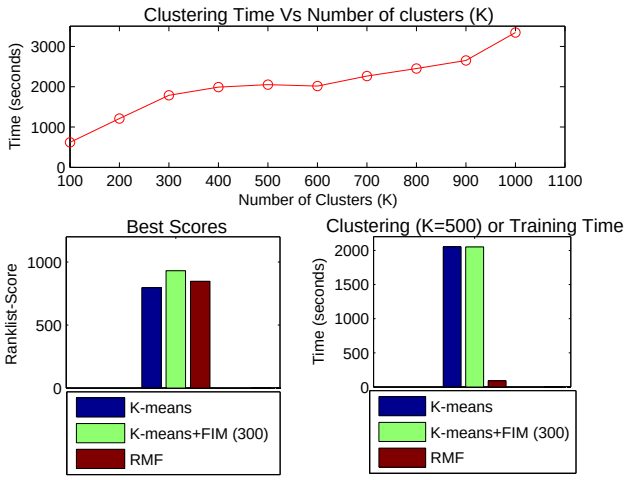

Figure 4: Top: Clustering time is too high compared to the time taken to build forests, which takes only about 90 seconds to built 20 trees; Bottom left: Best scores by the baseline and our two methods; and Bottom right: training RMF is about 20 times faster than clustering for k=500, when k-means+FIM reaches its best range of ranking score.

VOC 2007 [8] as database for the quantitative evaluation. It has 20 object categories such as person, boat, car etc. The dataset also provides bounding box level ground truth for each object instance. The task is to automatically rank these object instances such that the more frequent categories are higher in the ranked list. For representation of object bounding boxes we use PHOG [5] descriptor. Here we analyze and compare our mining approaches experimentally for finding frequent patterns. We also compare them with k-means as a simple baseline.

K-means: In our baseline method we quantize the PHOG descriptors by k-means and represent each sample by the cluster ID. Now the samples are ranked based on the size of the cluster they belong to and the ones belonging to larger clusters are ranked higher. Samples belonging to same cluster are ranked based on their distance from the cluster center.

K-means (soft assignment) + FIM: Here each sample is represented by a set of cluster IDs. Set includes the nearest cluster and the clusters having distance not more 1.05 times of the distance from nearest cluster. These sets can be seen as transactions and cluster IDs as items. We apply FIM on such a transactional database to find frequent cluster ID sets. Each sample is assigned to the largest cluster ID set which it supports. When a sample supports more than one sets of same size then it is assigned to that set of clusters which have least mean distance from it.

Randomized Mining Forest (RMF): This is the approach described in Section 2.2. Each sample is represented by the sequences of nodes traversed till leaf node in each tree. We build Random forest of 20 trees with 50 features, 15 thresholds at each node and maximum depth is set to 20. Therefore, the length of each transaction is about 400. Samples are ranked according to their support given by equation 6.

Let class of r^{th} sample in the ranklist (i.e. sample having rank r) is given as $C(r)$. Frequency of an object classes c is given as

$$Freq(c) = \frac{number\ of\ samples\ of\ class\ c}{N}$$

where N is total number of samples in the dataset. The ranklist-score is computed as,

$$ranklist\text{-}score = \frac{1}{N}\sum_{r=1}^{N} Freq(C(r)) \times (N - r + 1) \qquad (7)$$

There are 12839 object instances in VOC 2007 *trainval+test* data (we do not include truncated examples). In addition we randomly take bounding boxes (6642 samples) from background which do not overlap with any object instance. Frequency of these samples is set to zero ($Freq(c) = 0$). According to equation 7 for 19481 samples the expected ranklist-score of a random ranking would be 702.

Figure 3 shows the results of applying *K-means* and *K-means+FIM* for ranking these samples. With minimum support greater than 100, *K-means + FIM* method achieves higher *ranklist-score* at almost all values of K. We get better results with larger number of clusters with *K-means+FIM* method. Using *RMF* we get scores ranging in 750 to 847 with different values of M. Figure 4 compares the baseline and our two methods for efficiency and performance. Both our methods perform better than the baseline. Best score by *K-means+FIM* is higher than that of *RMF*. But it requires quantization into large number of clusters, which takes significantly more time than that for training RMF. The advantage of *RMF* comes in efficiency as building random forest is much faster (approx. 4.5 seconds per tree). The speed is critical while processing on large datasets as we do in the next section.

3.2 Movie Characterization

Figure 5: Some examples from the dataset

Figure 6: Some examples of characteristic scenes retrieved from movies Braveheart, LOTR: The Return of The King, Sixth Sense and Chicago (from top to bottom).

The dataset for the experiments in these section includes 81 Oscar winning and nominated best movies over the last 60 years ranging from 1950 to 2008. These movies form a good mixture and subset of the huge number of movies available. The genres, directors and other movie details of the dataset were taken from Internet Movie Database (IMDB, [1]). Figure 5 shows some example keyframes from the database.

3.2.1 Characteristic Scenes of the Movie

In this experiment we apply our method to identify characteristic scenes of the given movie. We extract GIST [12] features from the key-frames to encode the global information. About 50 thousand keyframes are selected at random from the dataset of 81 movies for feature extraction. Using the extracted features the feature space is quantized into 1000 bins by K-means clustering algorithm. In movies, frames belonging to same shot are generally very similar so we do not represent frame as a transaction. Here we represent a shot as a transaction of frames (items) and do frequent itemset mining to find frequent keyframes. Support of a visual word is computed as number of shots it occurs in divided by total numbers of shots in the movie. Certain scenes such as people speaking, road, room etc. are very common in the movies. So support computed from a movie is taken as *term-frequency (TF)* and the *inverse document frequency (IDF)* is computed by applying FIM on all the movies together. So the *TF/IDF* support of any visual word (W) in movie m from dataset M is given by

$$support(W) = \frac{support_m(W)}{support_M(W)}$$

Another experiment is done to detect characteristic scenes of the movie but with each frame represented as a sequence of items as explained in Section 2.2. The items are the IDs of the nodes traversed by the frame when pushed down the RMF. Each frame is represented as a sequence of items or nodes. We used ensemble of 100 randomized trees, number of features and thresholds tried to create node-test at each node are 100 and 15 respectively.

Figure 6 shows some examples of characteristic scenes retrieved from our movie database. First two rows show some examples of results by our first approach using FIM. Corresponding to each visual word we have many frames and shots. The figure shows six keyframes for a movie each representing one of the top 6 words from that particular movie.

Figure 7: Main character discovered from the movies *Rocky1*, *300*, *All About Eve*, *Slumdog Millionaire* and *A Beautiful Mind*.

Last two rows in Figure 6 shows some examples of characteristic scenes retrieved by RMF.

Saying that a given frame is characteristic frame of the movie is highly subjective. But the frames discovered by our approaches do match with the theme and genre of the movie. *Braveheart* is an action, drama movie which has many war scenes in it, this can be seen in the retrieved keyframes. Similarly for the *Lord of The Rings: The Return of The King* which is again an action, adventure, fantasy movie. Genre for *Sixth Sense* is a drama, mystery, thriller and *Chicago* is a musical, drama and crime movie. The characteristic scenes retrieved by our approach also suggests the same. We conducted the experiment for all 81 movies and got relevant keyframes as characteristic scenes. In films without much action, adventure or music mostly main characters are visible in the characteristic frames.

3.2.2 Identifying main characters in the Movie

The characteristic scenes can be used to predict the genre of the movie, in movie recommendation systems etc. Here we use these keyframes to predict the main character of the movie. Everingham *et al.* [7] have investigated the problem of automatically naming the characters in TV or film material. They do this by aligning subtitles and transcripts, and complement these cues by visually detecting which character in the video corresponds to the speaker.

We only use the key frames corresponding to most frequent visual words to find the main characters of the movie. Face detection and facial feature localization is done on these characteristic key frames. We start from the visual word with highest support and detect no more than 100 faces. Only faces larger than 100×100 are considered. The face descriptors are extracted from detected faces using Oxford VGG face processing code [7]. These face descriptors are clustered into 8-15 clusters by k-means. Then each of the cluster is pruned by removing all the faces which are at a distance greater than D from its cluster center. D is computed as the mean of the distances of faces from their cluster centers. Clusters having faces only from nearby shots are rejected as the main character should be present at many points throughout the movie. It is desired to have smaller clusters with many members. We compute the cluster density as a summation of inverse of distances of all cluster

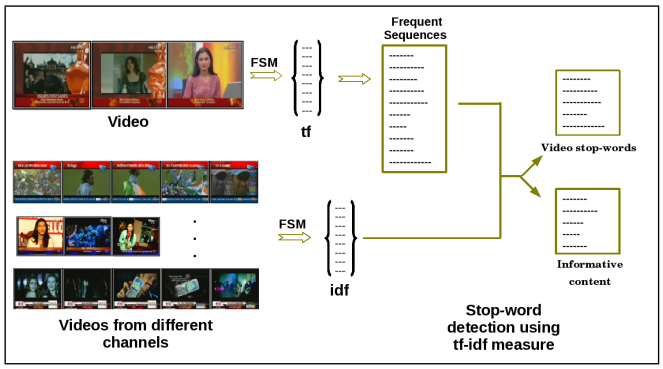

Figure 8: Stop word detection from video using Frequent Sequence Mining

members from the center. Cluster density for cluster c is given as: $\delta(C) = \sum_{i \in C} \frac{1}{d_i}$.

Only the most dense clusters are returned as the set of instances of main characters. Figure 7 shows the keyframes from the most dense clusters for movies *Rocky1*, *300*, *All About Eve*, *Slumdog Millionaire* and *A Beautiful Mind*. This works well as the characteristic scene mostly include many instances of main character's close-up face.

3.3 Video Stop Word Detection

Stop words in a language are words that many search engines do not stop for when searching for texts and titles on the web. These are common words, such as "the", "are", "is" etc. Similarly in text classification, elimination of stop words potentially increases the accuracy, reduces the feature space and helps to speed up the computation [19].

In videos too, certain sequence of frames repeatedly occur, motivating us to address the similar problems in videos. In case of videos, repetition could be either absolute or approximate. Some examples are commercials in TV programs and news or routine events in surveillance videos. It is desired to detect/remove such redundancy for many applications like video summarization, and search.

We define *video stop word* as the frequently occurring *sequence of frames* that are not informative. For example advertisement can be a *video stop word*. The frequency of occurrence of a sequence in a video gives us the *TF* part based on which we select frequent sequences or potential *video stop-words*. We use *IDF* to classify it as *video stop-word*. To compute *IDF* for any sequence, we use frequency with which it occurs in news videos across different channels. Higher the *IDF* measure more is the probability that the frequent sequence is a commercial. This works fine as same set of commercials occur frequently in many channels and consistently for large intervals.

This analogy between commercials (*video stop-words*) and stop-words fits well in case of news videos. A typical news video would mostly contain important news or commercials in its set of frequent sequences. Therefore when detected *video stop-words* are removed from the set of frequent sequences and we get the *informative content* of the video. This is shown in Figure 8, based on *IDF* measure frequent sequences extracted from video are classified as *video stop-word* or informative part of video.

Thus the *video stop-word* detection results can be used to: (a) block undesirable content and (b) summarize video with only important content. The latter can be done by computing the *information measure* of each shot, S_{sid} as:

$$\Gamma_{S_{sid}} = \sum_{F \in F_{freq}} sign(F) * support(F) * |F| \quad (8)$$

$$sign(F) = \begin{cases} -1 & \text{if } F \in stopword \\ 1 & \text{otherwise} \end{cases}$$

where F_{freq} is a set of frequent sequences in shot S_{sid} and F is a frequent sequence in F_{freq}. Therefore, more informative shots can be kept in the summary by keeping a threshold on Γ_{sid}.

Vocabulary size	Frequent sequence		Video stop-word		Informative Content	
	Prec	Rec	Prec	Rec	Prec	Rec
100	0.63	0.57	0.67	0.59	0.55	0.52
200	0.81	0.68	0.87	0.71	0.67	0.61
500	0.87	0.78	0.93	0.79	0.71	0.74
1000	0.97	0.91	0.98	0.94	0.90	0.83
2000	0.98	0.90	0.98	0.93	0.95	0.83

Table 1: Precisions and recalls for frequent sequence, *video stop-word* and informative content detection with different vocabulary sizes.

3.3.1 Experiments

Experiments are done on news videos obtained from eight broadcast news channels. For estimating *IDF* measure, videos from these channels are mined over five days (100 hours) and extracted frequent sequences with their frequency of occurrence (*IDF*) are stored. While testing these *IDF* values are used to separate commercials from the rest of the frequent sequences obtained in the set of test videos. For testing 1000 news video clips of total duration of 10 hours are used. News videos have lot of overlaid text in its lower one-third part, which affects the feature descriptor and similar frames are assigned different visual word or item. To handle this we only extract features from the upper two-third part of the frame.

We partition news video clips into shots and pick four frames per second within each shot. The ground truthing is done at the frame sequence level, each manually annotated frequent sequence is marked as *video stop-word* or *informative content*. This resulted in 91 frequent sequences out of which 68 were marked as visual stop-word and remaining 23 as informative content.

We use precision and recall to evaluate the detection performance. Precision and recall for *video stop-word* detection are computed as follows:

$$Precision = \frac{\#\ true\ visual\ stopwords\ detected}{\#\ total\ visual\ stopwords\ detected}$$

$$Recall = \frac{\#\ true\ visual\ stopwords\ detected}{\#\ total\ true\ visual\ stopwords}$$

Sometimes due to disturbance in the telecast, few frames get modified. A detection is considered to be true if the overlap between detected sequence F_{Det} and annotated sequence F_{GT} is more than 90%. Overlap is given by $\frac{|F_{Det} \cap F_{GT}|}{|F_{Det} \cup F_{GT}|}$. Figure 9 shows some examples of detected *video stop-words*.

For the experiments we set $min\_sup = 0.005$. The performances for detecting frequent sequence, *video stop-word* and informative content in terms of precision and recall are reported in Table 1. The precisions are high as it is difficult to

Figure 9: Some examples of *video stop-word* detection

get false frequent sequences when large enough vocabulary is chosen. With vocabulary size of 1000 and above, recall and precisions are high for all three cases. Precision always increases with vocabulary size. However recall starts decreasing with too much quantization. This is because with more number of clusters a slight variation in a frame can assign it to a different cluster, which may lead to false negatives. Some of the examples of detected commercials as *video stop-word* is shown in Figure 9.

Also the method is very efficient. In the online phase features are extracted at 150 fps and visual words are assigned at 250 fps with vocabulary size 1000. For mining 76,000 sequences of average length 100 it takes about 40 seconds. This shows that the proposed method is scalable and can be effectively used for real-time on-line applications.

4. CONCLUSIONS

We have presented an approach to discover characteristic patterns in videos in an unsupervised fashion. Our method is based on finding frequently occurring patterns. First of our approaches employs frequent pattern mining for efficient characterization. We also propose to use randomized trees to represent frame as sequence of nodes and mine frequent sequences. Our second approach provides efficiency which is key while operating on large databases. To evaluate the proposed methods we compare them with each other and with a simple baseline. The approach is validated by experiments over a large movie dataset to discover characteristic scenes and main actors in the video. We also define *video stop-words* and detect them using frequent sequence mining. Stop words are identified using *TF-IDF* type measure for frame sequences. By adopting appropriately, traditional methods from data mining can been successfully used in computer vision. Such techniques can result in fast, efficient algorithms for large scale video processing.

5. REFERENCES

[1] In *Internet Movie Database, http://www.imdb.com.*

[2] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD*, 1993.

[3] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. ICDE*, 1995.

[4] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via pLSA. In *Proc. ECCV*, 2006.

[5] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Proc. ICCV*, 2007.

[6] L. Breiman. Random forests. *ML Journal, 45(1)*, 45:5–32, 2001.

[7] M. Everingham, J. Sivic, and A. Zisserman. Hello! my name is... buffy - automatic naming of characters in tv video. In *Proc. BMVC*, 2006.

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.

[9] Y. J. Lee and K. Grauman. Object-graphs for context-aware category discovery. In *Proc. CVPR*, 2010.

[10] F. Moosman, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Proc. NIPS*, 2006.

[11] S. Nowozin, G. Bakir, and K. Tsuda. Discriminative subsequence mining for action classification. In *Proc. ICCV*, 2007.

[12] A. Oliva and A. B. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[13] J. Pei, J. Han, B. Mortazavi-asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proc. ICDE*, 2001.

[14] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007.

[15] T. Quack, V. Ferrari, and L. J. V. Gool. Video mining with frequent itemset configurations. In *Proc. ACM CIVR*, 2006.

[16] T. Quack, V. Ferrari, B. Leibe, and L. V. Gool. Efficient mining of frequent and distinctive feature configurations. In *Proc. ICCV*, 2007.

[17] T. Quack, B. Leibe, and L. V. Gool. World-scale mining of objects and events from community photo collections. In *Proc. ACM CIVR*, 2008.

[18] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Proc. CVPR*, 2008.

[19] C. Silva and B. Ribeiro. The importance of stop word removal on recall values in text categorization. *Neural Networks*, pages 320–324, 2003.

[20] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003.

[21] J. Sivic and A. Zisserman. Video data mining using configurations of viewpoint invariant regions. In *Proc. CVPR*, 2004.

[22] V. S. Tseng, J.-H. Su, J.-H. Huang, and C.-J. Chen. Semantic video annotation by mining association patterns from visual and speech features. In *PAKDD*, 2008.

[23] S. Vijayalakshmi, V. Mohan, and S. S. Raja. Mining constraint-based multidimensional frequent sequential pattern in web logs. *European Journal of Scientific Research*, 36(3):480–490, 2009.

[24] B. Yao and L. Fei-Fei. Grouplet: a structured image representation for recognizing human and object interactions. In *Proc. CVPR*, 2010.

[25] M. J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, pages 31–60, 2001.