Video Summarization Using Deep Action Recognition Features and Robust Principal Component Analysis

Daniel M. Claborne, Karl T. Pazdernik, Steven J. Rysavy, Michael J. Henry
Pacific Northwest National Laboratory - National Security Directorate

Abstract [1]

In an instance where desired pre-defined actions, behaviors, or other categories are known a priori, various video classification and recognition models can be trained to discover those classifications and their location within the video. Absent that information, one might still be tasked with identifying interesting portions within a video, a process which—if done manually—is onerous and time-consuming as it requires manual inspection of the video itself. Recognizing high-level interesting segments within a whole video has been a general area of interest due to the ubiquity of video data. However the size of the data makes storage, retrieval, and inspection of large collections of videos cumbersome. This problem motivates the task of generating shortened clips highlighting the primary content of a video, relieving the burden of having to watch the entire video. This paper presents an unsupervised method of creating shortened clips of videos, enabling the rapid review of the most interesting content within a video. Our method uses features extracted from pre-trained action recognition models as input to online moving window robust principal component analysis to generate summaries. The procedure is tested on a publicly available video summarization dataset and demonstrates comparable performance to state-of-the-art in an un-augmented setting while requiring no training.

1. Introduction and Related Works

Summarization of a video into a compact form is a difficult task that has been approached from several angles. These include extracting key-frames, defining shot boundaries, summarization by text, and creating continuous but shortened video. The method described here focuses on the last of these, with the specific goal of capturing the informative segments of the video.

The greatest challenge of this task is defining what constitutes 'informative' or 'interesting', which can depend on the application and perspective of the viewer. Two common criteria are *representativeness*, which can be thought of as how well each segment encapsulates the important content of the video, and *diversity*, which demands non-redundancy of segments of the summary. Evaluation is also difficult due to the lack of a large scale dataset that has frame-level ground truth, and the fact that 'ground truth' is far more ambiguous than in other vision tasks.

Low-level features such as GIST, HOG, color histograms, and optical flow have been used in frameworks that attempt to define an 'interestingness' metric [10][11], align themselves with content that draws human attention [6][13], or combine the features so as to capture semantic content [23]. Convolutional neural networks are now commonly used as feature extractors, and have shown modest improvements over using shallow features [25][26].

Extracted features are used in various schemes to select which frames are included in the summary, usually focusing on the mentioned goals of diversity and representativeness. In [15] a k-mediods problem is solved to select a diverse set of segments. Determinantal point processes (DPP) have been used to enforce the diversity of returned segments [9][14][24]. In [27], a reinforcement learning model was trained with the reward function directly trying to capture the notions of diversity and representativeness.

Due to the lack of a large labeled dataset, and the ambiguity of what constitutes ground truth, unsupervised approaches have historically been more common. These often include a step which segments the video into optimally diverse subsets and then assigns scores to each subset to produce a summary [11][15][16]. In [14] an adversarial framework is used to train a frame selector that produces maximally representative summaries. Highlights defined as outliers in the latent space of an autoencoder framework were used to create summaries in [21].

Supervised approaches based on recurrent neural networks can effectively capture the sequential nature of video content and have shown state-of-the-art results [25][26]. The unsupervised frameworks in [27] and [14] were extended to supervised versions that slightly improved performance. Human annotated summaries have also been used to fine-tune DPP methods to select optimal subsets [9][24][25].

We investigate an approach that requires no training (given pre-trained models), and makes use of robust

[1]The authors would like to thank Elizabeth Cary and Lee Burke for acting as peer-editors for this paper.

principal component analysis (RPCA)[4] to select summary segments. RPCA based methods have been used to perform foreground detection in video, that is to separate anomalous foreground elements from the background [3]. Like [21], our summary is a 'highlight reel', defined by anomalous events in the video, in our case as detected by an RPCA based algorithm. There are many difficulties to this approach, but most boil down to the lack of a globally consistent representation in a video [2]. Various adjustments have been made to RPCA based algorithms to account for one or more of these challenges [3]. Our method uses features extracted from action recognition models to create a more consistent and semantically meaningful input to this class of algorithms.

In **Section 2** we give a brief overview of RPCA and convolutional neural networks as applied to action recognition, the reasoning behind their use in this task, and the particular flavor of RPCA and action recognition model chosen. **Section 3** details the novel methodology, and **Section 4** covers the evaluation procedure, followed by a discussion of the results.

2. Preliminaries

2.1 Notation

We refer to matrices with upper case letters (A, L, S), vectors or lists with bolded lower case letters ($\boldsymbol{a}, \boldsymbol{l}, \boldsymbol{s}$), and scalars with unbolded lower case letters ($s_{t,j}, q_1$). The dimensions are given in subscripts when we feel it may be unclear. We use $t \in \{1, 2, ...\tau\}$ to index over matrix columns, sequences of video segments, or sequences of feature embeddings of video segments. The size of the feature space of action recognition model output is given by d.

2.2 RPCA

Robust principal component analysis attempts to reconstruct an additive representation of a matrix in the presence of gross errors. Specifically, given a matrix $M = L + S$, where L is a low rank matrix and S is a sparse matrix of errors, we want to retrieve L and S. This is accomplished by solving the optimization problem:

$$\min_{L,S} \|L\|_* + \lambda \|S\|_1 \ : \ L + S = M$$

Where $\|L\|_*$ and $\|S\|_1$ are the nuclear norm and L1 norm of L and S respectively and λ is a tuning parameter. There are theoretical guarantees that the algorithm will recover L exactly under a wide range of conditions [4].

In our application to video summarization, we are most interested in the sparse matrix S. The elements of this matrix should be mostly zero, and columns for which this is not true indicate observations that are not representative of the underlying data generating process. Our approach favors these observations for inclusion in the produced summary.

The standard RPCA algorithm above has a particularly glaring drawback in that it cannot deal with inconsistent subspaces over time [20][3]. The observations (frames) in a video that have drastically different content across time are less dependent - in other words L, which represents the true underlying subspace, is poorly approximated by a low rank matrix.

A standard approach to overcoming this problem is moving window robust principal component analysis (MWRPCA), which updates the underlying subspace based on iterative computation of RPCA-PCP [4] on n_{win} samples in a sliding window. The limitation of this approach is that for large matrices, there is a severe computational bottleneck [20].

The solution proposed in [8] is to update the subspace in an online manner, however that algorithm loses the benefit of the sliding window. The online and sliding window approaches are combined in [20] to create an online moving window RPCA (OMWRPCA) algorithm and an overview of their implementation is given here. The goal remains the same: to recover the components L and S. To accommodate the online updating of L, it is further decomposed into a product of matrices $L = UV$, where U forms a basis for the low rank subspace, and the columns of V are the coordinates of the low rank portion of the observations with respect to the basis elements. The loss function to be minimized is:

$$\min_{U,V,S} \frac{1}{2}\|M - UV - S\|_F^2 + \frac{\lambda_1}{2}(\|U\|_F^2 + \|V\|_F^2) + \lambda_2 \|S\|_1$$

Where $\|U\|_F$ is the Frobenius norm of U. Now consider a sequence of observations, revealed one at a time $m_1, m_2, ..., m_T$. The values of L and S will be determined one column at a time as the observations are revealed by minimizing an empirical version of the loss function. Generally:

Decompose M into L and S by OMWRPCA

Input: $\boldsymbol{M}$, Matrix to be decomposed
$\boldsymbol{U_0}$, Initial estimate of U

for $t \in 1, 2, ..., T$ **do**
  Reveal observation $\boldsymbol{m}_t = M$[:,t]
  Find columns $\boldsymbol{v}_t$, $\boldsymbol{s}_t$ of V and S given $\boldsymbol{m}_t$
  Find U_t given U_{t-1}
  L[:,t]$\leftarrow U_t \boldsymbol{v}_t$
  S[:,t]$\leftarrow \boldsymbol{s}_t$
end for
return L, S

Where A[:,t] is code syntax for all rows of column t of matrix A

The steps to find $\boldsymbol{v}_t$, $\boldsymbol{s}_t$, and U_t in OMWRPCA are an extension of those in [8] but are calculated based on

the previous n_{win} samples instead of a running average of all the currently revealed samples. The advantage of this is that it can better handle quickly changing subspaces. This is desirable in our case since we want to detect abrupt changes in scene so as to identify a diverse set of content, but at the same time quickly update the estimated subspace to detect anomalous events *within* the new scene. OMWRPCA is the variant of RPCA we employ for video summarization.

2.3 Video Action Recognition Models

Determining actions within video is an ongoing area of research in computer vision, with successful models trained on very large datasets becoming available only in recent years [12]. The task is a classification problem, where we want to predict the action label of a short video clip. For example, the benchmark dataset UCF-101 [17] contains videos where the labels are categories such as 'applying eye makeup' and 'baseball pitch'.

The state-of-the-art models in [5][7] are extensions of convolutional neural network architectures for image recognition where the convolutions are applied over three dimensional volumes $(time \times width \times height)$. The convolution over the time dimension captures information about motion over short time periods in the final feature representation. Action recognition networks are also often trained on input that explicitly capture motion information[19]. In addition to visual and motion information separately, these models also capture the interaction between the two. We rely upon these semantically meaningful representations to provide input to OMWRPCA that is stable across video sections that have similar semantic content.

3. Methods

Our approach combines OMWRPCA and action recognition models to identify sections of a video which will be included in a summary; the procedure is outlined in Figure 1. The steps involved roughly separate into feature extraction, scoring, and frame selection.

3.1 Feature extraction

To create the input to OMWRPCA, we extract features from across an entire video using pre-trained action recognition models. In our experiments, we make use of the C3D [18] and i3D [5] architectures. The input dimensions for a single sample for each model used in the rest of this document are given below. i3D uses one or both of two possible inputs, raw RGB frames or depth 2 optical flow stacks.

$$\text{C3D} : (16 \times 112 \times 112 \times 3)$$

$$\text{i3D(rgb)} : (32 \times 224 \times 224 \times 3)$$

$$\text{i3D(flow)} : (32 \times 224 \times 224 \times 2)$$

For the C3D model, we take the approach described in the source paper, which is to obtain three model inputs of dimension $16 \times 112 \times 112 \times 3$, where each segment overlaps by 8 frames. For each of these inputs, the output of the second to last fully connected layer is extracted, yielding three vectors of dimension 4096. Finally, the three outputs are averaged and the result is L2-normalized to produce the final feature vector of dimension 4096. For the i3D model, a single input of dimension $32 \times 224 \times 224 \times 3$ is obtained, and the output of the final global average pooling layer is flattened to produce a feature vector of dimension 3072. The spatial (second and third) dimensions are obtained by resizing individual frames to 171×128 and 256×256 and then taking a center crop of dimension 112×112 and 224×224 for C3D and i3D respectively.

To form the input to OMWRPCA, we need a sequence of feature vectors that cover the whole video. To accomplish this, we extract short, continuous segments of frames from across the entire length of an input video. The segments have width equal to the time dimension of the input for whichever feature extractor we are using, and begin at evenly spaced points throughout the video. In our experiments, this spacing is chosen to be half width of each segment, so that there is some overlap. Features are then extracted from each segment and are sequentially arranged as columns in a matrix which is the input to OMWRPCA:

$$F_{d\times\tau} = \begin{bmatrix} | & | & & | \\ \boldsymbol{f_1} & \boldsymbol{f_2} & \dots & \boldsymbol{f_\tau} \\ | & | & & | \end{bmatrix}$$

Where f_t is the feature representation of a video segment, indexed by t. It is worth noting the distinction between the feature indices and the frame indices. The features represent a chunk of frames at some position in the video.

3.2 Scoring

After we have obtained F, the matrix of video features, we want to identify which feature vectors we should choose to create a summary by assigning each of them a score, defined in **Procedure 1**. The score relies on the output of OMWRPCA, namely the low rank and sparse matrices L and S. Intuitively, non-sparse columns of S indicate segments which are noticeably different than the $n_{win} - 1$ segments leading up to that point. Additionally, we assume an unstable underlying subspace L, and so expect it to contain several independent representations of the various scenes in the video. We seek to form a video based on the criteria of diversity and representativeness.

To achieve representativeness, we select feature vectors whose corresponding component in S is non-sparse. These feature vectors hopefully correspond to new semantic content within the video. To identify non-sparse sections, we sum over the columns of the

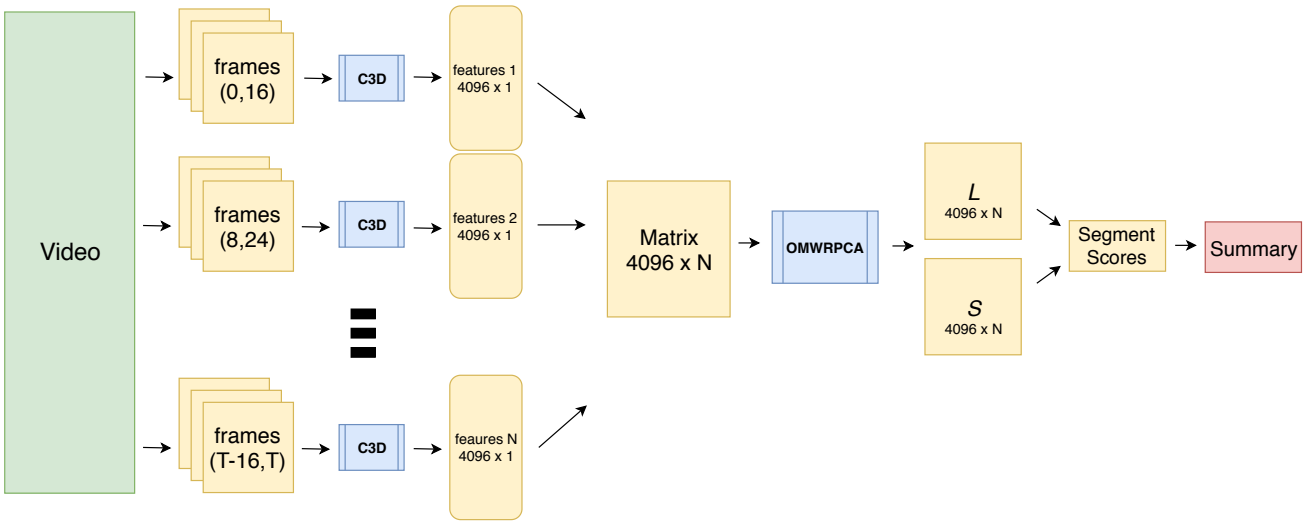

Figure 1: Overview of summary creation procedure using C3D model.

Procedure 1: Compute a vector of scores for each feature index.

Input: $\boldsymbol{F_{d\times\tau}}$, Matrix of feature vector columns

1: $L_{d\times\tau}, S_{d\times\tau} \leftarrow$ OMWRPCA(F)
With $\boldsymbol{l}_t$ and $\boldsymbol{s}_t$ as the length d columns of L and S respectively at index t:

2: $\boldsymbol{p}_{1\times\tau} \leftarrow [p_1, p_2, ...p_\tau] : p_t = \sum_{j=1}^{d} |s_{j,t}|$

3: $\boldsymbol{m}_{d\times 1} \leftarrow median(\{\boldsymbol{l}_1, \boldsymbol{l}_2, ..., \boldsymbol{l}_\tau\})$

4: $Q_{d\times\tau} \leftarrow [\boldsymbol{q}_1^*, \boldsymbol{q}_2^*, ...\boldsymbol{q}_\tau^*] : \boldsymbol{q}_t^* = |\boldsymbol{l}_t - \boldsymbol{m}|$

5: $\boldsymbol{q}_{1\times\tau} \leftarrow [q_1, q_2, ...q_\tau] : q_t = \sum_{j=1}^{d} q_{j,t}^*$

6: **return** $\boldsymbol{r} = \boldsymbol{p} \odot \boldsymbol{q}$ (element wise product)

absolute value of S (Proc 1. Step 2). As the sliding window of OMWRPCA moves forward in time, the new subspace will eventually become the new normal, the corresponding columns of S will tend to be sparse, and the column sums p_t will be relatively small, but not before several sections have been flagged as potential candidates, forming a representative chunk of novel video content.

This procedure involving the column sums of S works well for identifying video segments which are locally distinct, however it will still pick up on identical transitions at different points in the video (i.e. the same camera pan from foreground to sky). Though these similar transitions might still warrant inclusion, we penalize their score using information obtained from L. The penalization is done by computing the absolute distance from the median (vector) of all columns of L (Proc 1. Steps 3-5). Intuitively, segments which are closer to the median feature representation of the entire video are penalized for redundancy; this is the mechanism by which we attempt to enforce diversity.

The score vector $\boldsymbol{r}$ is the element wise product of the two resulting score vectors $\boldsymbol{p}$ and $\boldsymbol{q}$. Indices with high scores correspond to video segments that were both different than the previous $n_{win} - 1$ segments and far from the median video content, in terms of the action recognition feature representation of the video.

3.3 Frame Selection

To select frames from our scores for each feature index, we either consider the raw values of $\boldsymbol{r}$ or the difference between the current and previous indices $r_t \leftarrow r_t - r_{t-1} : t \in 1, 2, ...\tau$. We form a summary by selecting the feature indices with the highest scores and storing the corresponding video frames until we have frames equal to some desired proportion of the total frames in the video. The range of frames corresponding to a feature index t is given by: $(t \cdot w, t \cdot w + \Delta x)$ where w is the spacing (number of frames) between each video segment, and Δx is the time dimension (number of frames) of the input to our feature extractor. See Procedure 2.

4. Evaluation

4.1 Dataset and Criteria

To test the model, we use the publicly available SumMe dataset [11], which consists of 25 videos each with multiple human summaries. Specifically, subjects were asked to pick segments from a video that they believed best summarized its content, with lower and upper limits of 5% and 15% of the whole video. Our scoring method follows that of [11] and [25], which compare an automatic video summary against all human generated summaries for that video. Specifically, given n user summaries corresponding to video i: $\boldsymbol{b}_{ij} : j \in \{1, 2, ..n\}$, and a single automated video summary $\boldsymbol{a}_i$, compute the F1 measure f_{ij} between $\boldsymbol{a}_i$ and each user summary $\boldsymbol{b}_{ij}$:

$$f_{ij} = \frac{2 \cdot p_{ij} \cdot r_{ij}}{p_{ij} + r_{ij}} \; : \; p_{ij} = \frac{\boldsymbol{a}_i \cap \boldsymbol{b}_{ij}}{|\boldsymbol{a}_i|}, r_{ij} = \frac{\boldsymbol{a}_i \cap \boldsymbol{b}_{ij}}{|\boldsymbol{b}_{ij}|}$$

Then, within a video i either take the mean or maximum F1 measure across summaries j. Finally, take these maximum/mean scores and average across all videos i to get a final score.

As in [11][25] we constrain our summaries to be approximately 15% of the whole video (slightly more since the last segment appended will overflow). Several parameters were changed to test the effect on performance. These included the size of the OMWRPCA sliding window, the strength of the L1 penalty on S (λ_2 in the objective function), the spacing between segments used as input to the feature extractors, the model used (C3D vs i3D), whether to score a feature index by its raw value or difference from the previous value, and a slightly higher proportion of selected frames.

For the window size, n_{win}, several runs ranging from $n_{win} = 2$ to $n_{win} = 10$ found that a width of 5 is usually best. There was no appreciable difference in changing the λ_2 value from its default of $d^{-0.5}$. A spacing of 8 and 16 were best for the C3D and i3D models, respectively, and using the raw values of the score vector rather than the difference produced the highest F1-scores on SumMe.

Procedure 2: Create summary from feature scores.

Input: $\boldsymbol{r}$, Vector of scores from Procedure 1
γ, Size of summary, as a proportion of total video frames.

1: $\boldsymbol{r} \leftarrow$ argsort($\boldsymbol{r}$)
2: Initialize empty list of frames $\boldsymbol{F}$
3: **while** len($\boldsymbol{F}$) $< \gamma \cdot$ len($Video$) **do**
4: $t \leftarrow$ pop($\boldsymbol{r}$) (Index of next highest score)
5: $frames \leftarrow [t \cdot w, t \cdot w + \Delta x] \cap \mathbb{N}$
6: $\boldsymbol{F} \leftarrow \boldsymbol{F} \cup frames$
7: **end while**
8: **return** $\boldsymbol{F}$

Where len($\boldsymbol{x}$) is the number of elements in $\boldsymbol{x}$, $t \leftarrow$pop($\boldsymbol{x}$) removes the first element in $\boldsymbol{x}$ and assigns it to t, and argsort($\boldsymbol{x}$) arranges the *indices* of $\boldsymbol{x}$ in descending order by the *value* of $\boldsymbol{x}$. See 3.3 for descriptions of w and Δx.

4.2 Results

Performance against other reported results on SumMe is shown in Table 1. Parentheses indicate the use of augmented training data, [27] and [14] report both supervised and unsupervised approaches. We did not investigate any uses of augmented data as our method does not share any updated parameters across videos, and so we compare scores in the un-augmented setting using maximum F1-score as the comparison metric: **Our method reports the highest scores among all unsupervised approaches, and beats all but the method of [27] among supervised competitors, where our procedure has the advantage of requiring no training**.

Method (†= supervised)	Mean F1-score	Max F1-score
CNN + Cluster [15]	0.182	–
Interestingness [11]	0.234	0.393
Reinforcement Learning [27]	–	0.414 (0.428)
Adversarial [14]	–	0.391 (0.434)
DPP [24] †	–	0.409
LSTM + DPP [25] †	–	0.386 (0.429)
Adversarial [14] †	–	0.417 (0.436)
Reinforcement Learning [27] †	–	**0.421** (0.439)
Hierarchical RNN [26] †	–	(0.443)
C3D + OMWRPCA	0.210	0.403
i3D + OMWRPCA	0.220	**0.419**

Table 1: F1-scores for ours and other results. Mean and maximum columns correspond to taking the mean or maximum F1-score across summaries within a video before averaging across all videos. Parentheses indicate scores achieved using training data augmented from other videos. Best scores for supervised and unsupervised methods in bold.

On such a dataset as SumMe, where there are multiple ground truths for a single input, a metric such as maximum F1-score is useful, but difficult to interpret. To further evaluate performance, a visual inspection of the results can show areas where the model performs well, and where it fails, as determined by its ability to pick up on obvious peaks.

Figure 2 shows the segments selected by the procedure against the proportion of users that included a particular frame in their summary for all 25 videos. Remember that the F1-score is not computed against the top values of the red line shown, but pairwise against every user summary for a given video, which are aggregated in Figure 2 for visualization purposes.

For content such as 'Paintball' and 'car over camera' in which objects and humans make periodic and abrupt appearances in the video, the model performs well, as seen in Figure 2, highlighted green. The model fails to pick up on subtle changes, such as the middle spike in 'Cooking' (Figure 2, row 1 column 3), where indeed there is not enough signal in either L or S to warrant inclusion in the summary. The produced summary for 'St Maarten Landing' is another example of this type of failure (Figure 2, row 2 column 1).

Another type of failure can be seen in 'paluma jump' (Figure 2, row 4 column 1), where the procedure focuses too heavily on the end of the video and misses two important spikes. The last section of 'paluma jump' has a substantially different background than the first 80% of the video. In terms of the scoring, the last section is far from the global median feature representation, and its segments are penalized much less severely than the previous 80% of video segments.

An adjustment was made to Procedures 1 and 2 in an

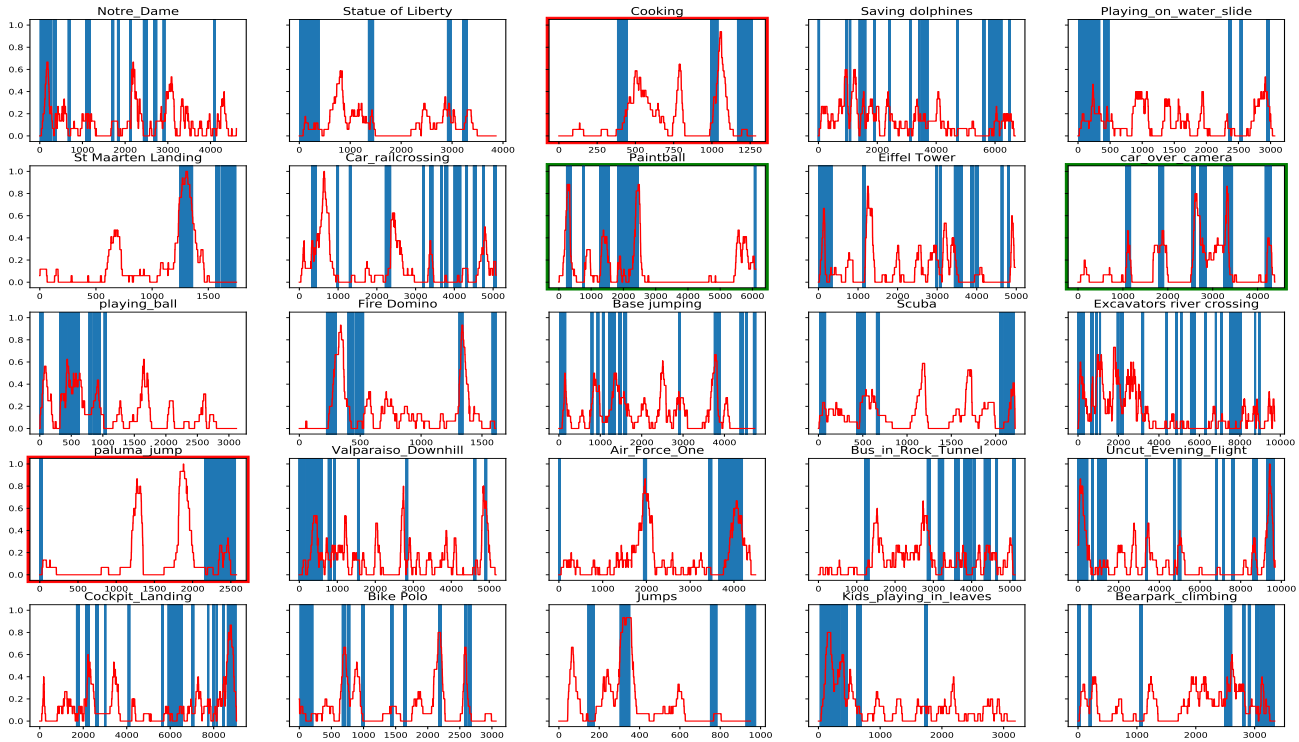

Figure 2: Segments selected by i3d + OMWRPCA (blue) against proportion of users who included a segment in the summary (red) for all 25 videos in SumMe [11]. Successful and unsuccessful identification examples bordered in green and red respectively.

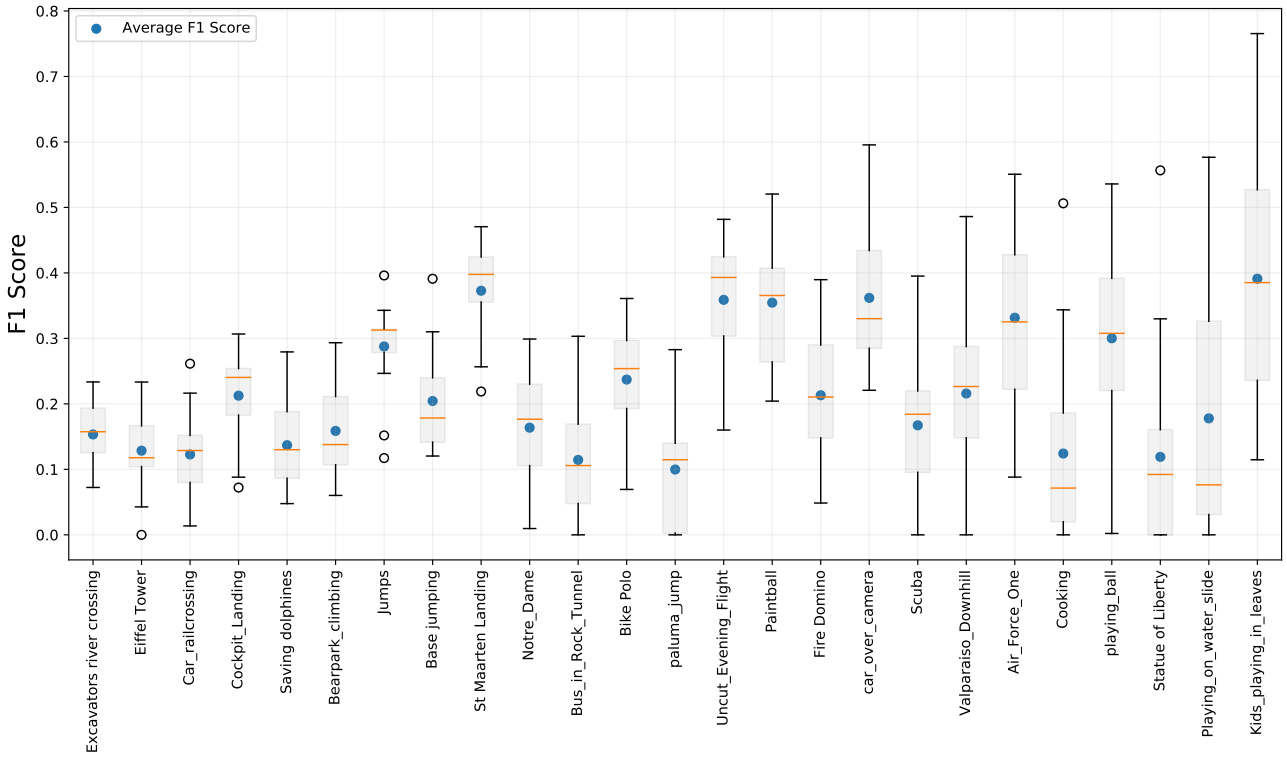

Figure 3: Boxplots of pairwise F1-scores between automated and user summaries for each video.

attempt to remedy this type of failure. When iteratively selecting segments, we compute a different score vector r at each iteration instead of a static score vector. The update is described in Procedure 3. At each selection step, the median vector $\boldsymbol{m}$ is computed as the median of the global median *and* all columns of L corresponding to already selected feature indices ($\boldsymbol{I}$). Intuitively, we are dissuading the algorithm from selecting a new segment whose underlying representation is close to that of already selected segments, hopefully improving the diversity of the resulting summary. This does not actually improve F1-score, but 'fixes' the type of failure seen in videos such as 'paluma jump' and 'St. Maarten landing', i.e. we see frame selection at the spikes where previously there were none.

Procedure 3: Updated scoring procedure with running median.

Input: γ Size of summary, as a proportion of total video frames.
$\boldsymbol{F}$ Matrix of feature vector columns

$L_{d\times\tau}, S_{d\times\tau} \leftarrow$ OMWRPCA($F_{d\times\tau}$)

With $\boldsymbol{l}_t$ and $\boldsymbol{s}_t$ as the length d columns of L and S respectively at index t:

$\boldsymbol{m}_{d\times 1} \leftarrow median\{\boldsymbol{l}_1, \boldsymbol{l}_2, ..., \boldsymbol{l}_\tau\}$
$\boldsymbol{p}_{1\times\tau} \leftarrow [p_1, p_2, ...p_\tau] : p_t = \sum_{j=1}^{d} |s_{j,t}|$
Initialize empty list of frames $\boldsymbol{U}$, list of medians $\boldsymbol{M} = \{\boldsymbol{m}\}$, empty list of already selected indices $\boldsymbol{I}$
while len($\boldsymbol{U}$) $< \gamma \cdot$ len($Video$) **do**
    $\boldsymbol{m^*_{d\times 1}} \leftarrow median(\boldsymbol{M})$
    $Q_{d\times\tau} \leftarrow [\boldsymbol{q}^*_1, \boldsymbol{q}^*_2, ...\boldsymbol{q}^*_\tau] : \boldsymbol{q}^*_t = |\boldsymbol{l}_t - \boldsymbol{m^*}|$
    $\boldsymbol{q}_{1\times\tau} \leftarrow [q_1, q_2, ...q_\tau] : q_t = \sum_{j=1}^{d} q^*_{j,t}$
    $\boldsymbol{r}_{1\times\tau} \leftarrow \boldsymbol{p} \odot \boldsymbol{q}$ (element wise product)
    $\boldsymbol{r}[i] \leftarrow -\infty, \forall i \in \boldsymbol{I}$
    $t \leftarrow argmax(\boldsymbol{r})$
    $frames \leftarrow [t \cdot w, t \cdot w + \Delta x] \cap \mathbb{N}$ (See 3.3)
    $\boldsymbol{U} \leftarrow \boldsymbol{U} \cup frames$
    $\boldsymbol{I} \leftarrow \boldsymbol{I} \cup t$
    $\boldsymbol{M} \leftarrow \boldsymbol{M} \cup L[:, t]$
end while
return $\boldsymbol{U}$

See 3.3 for descriptions of w and Δx.

4.3 Further Discussion

All this raises the question about how we might redefine what constitutes a 'good' summary or evaluate performance. One approach could be to pass our summarized video to another machine learning task that requires a fixed length video representation. This is a common requirement in the task of video captioning [1][22]. Another approach could be to define a threshold for how much agreement between human raters is necessary to define a segment as important. Performance would then be the algorithm's ability to select such segments given some budget of frames as a proportion of the total video frames.

The SumMe dataset is very small, and the difficulty of creating a large dataset that has frame-level labeling remains a challenge to effective evaluation. The evaluation metric of pairwise F1-score is also problematic. Boxplots of the F1-scores between our manually created summary and each user summary for all videos are shown in Figure 3, and illustrate the variance in agreement. For videos with high variance in F1-scores, the difference between the average and maximum F1-score can be rather egregious. Depending on the application, a method which achieves the highest possible average pairwise-within-video-average F1-score (approximately .454) may still not produce the most desirable summary. Algorithms attempting to achieve high F1-score will necessarily ignore segments that are selected by only a few human raters, as these are 'wasted frame guesses' in terms of F1 with a cap of 15% of the video length, though someone might still be interested in capturing these odd occurrences.

5. Conclusion

This work investigates the combination of deep action recognition model features with the sparse and low rank decomposition methods of robust principal component analysis as an unsupervised approach to video summarization that shows competitive results to state-of-the-art on a standard benchmark dataset. The model is simple to implement, employing two well defined procedures of action recognition model feature extraction and online moving window robust principal components decomposition with easily tunable hyperparameters to adjust the length and compactness of the produced summary.

Generated summaries perform well in maintaining diversity in most cases and pick up on semantically meaningful content. Evaluation on larger datasets, on different objectives rather than diversity and representativeness, and on different measures of success rather than F1-score are all directions for future exploration. Finally, it may be of interest to extend this framework to a trainable version that can take advantage of augmented training data.

6. Acknowledgements

The authors would like to thank Elizabeth Cary and Lee Burke for peer review. The research at Pacific Northwest National Laboratory (PNNL) was funded by PNNL's Open Source Data Analytics Program to complete the work (released under PNNL-SA-152604). PNNL is a multi-program national laboratory operated by Battelle for the U.S. Department of Energy.

7. References

[1] Lorenzo Baraldi, Costantino Grana, and Rita Cucchiara. "Hierarchical Boundary-Aware Neural Encoder for Video Captioning". In: *CoRR* abs/1611.09312 (2016). arXiv: 1611.09312.

[2] Thierry Bouwmans and El Hadi Zahzah. "Robust PCA via Principal Component Pursuit: A review for a comparative evaluation in video surveillance". In: *Computer Vision and Image Understanding* 122 (2014), pp. 22 –34. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2013.11.009.

[3] Thierry Bouwmans et al. "Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset". In: *Computer Science Review* 23 (2017), pp. 1 –71. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2016.11.001.

[4] Emmanuel J. Candès et al. "Robust principal component analysis?" In: *Journal of the ACM* 58.3 (2011), pp. 1–37. ISSN: 0004-5411. DOI: 10.1145/1970392.1970395.

[5] J. Carreira and A. Zisserman. "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4724–4733. DOI: 10.1109/CVPR.2017.502.

[6] Naveed Ejaz, Irfan Mehmood, and Sung Wook Baik. "Efficient visual attention based framework for extracting key frames from videos". In: *Signal Processing: Image Communication* 28.1 (2013), pp. 34–44. ISSN: 0923-5965. DOI: 10.1016/j.image.2012.10.002.

[7] Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. "Spatiotemporal Residual Networks for Video Action Recognition". In: *CoRR* abs/1611.02155 (2016). arXiv: 1611.02155.

[8] Jiashi Feng, Huan Xu, and Shuicheng Yan. "Online Robust PCA via Stochastic Optimization". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 404–412.

[9] Boqing Gong et al. "Diverse Sequential Subset Selection for Supervised Video Summarization". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 2069–2077.

[10] M. Gygli et al. "The Interestingness of Images". In: *2013 IEEE International Conference on Computer Vision*. 2013, pp. 1633–1640. DOI: 10.1109/ICCV.2013.205.

[11] Michael Gygli et al. "Creating Summaries from User Videos". In: *ECCV*. 2014.

[12] Will Kay et al. "The Kinetics Human Action Video Dataset". In: *CoRR* abs/1705.06950 (2017). arXiv: 1705.06950.

[13] Jie-Ling Lai and Yang Yi. "Key frame extraction based on visual attention model". In: *Journal of Visual Communication and Image Representation* 23.1 (2012), pp. 114 –125. ISSN: 1047-3203. DOI: https://doi.org/10.1016/j.jvcir.2011.08.005.

[14] B. Mahasseni, M. Lam, and S. Todorovic. "Unsupervised Video Summarization with Adversarial LSTM Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2982–2991. DOI: 10.1109/CVPR.2017.318.

[15] Mayu Otani et al. "Video Summarization Using Deep Semantic Features". In: *Computer Vision – ACCV 2016*. Ed. by Shang-Hong Lai et al. Cham: Springer International Publishing, 2017, pp. 361–377.

[16] Danila Potapov et al. "Category-specific video summarization". In: *European conference on computer vision*. Springer. 2014, pp. 540–555.

[17] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild". In: *CoRR* abs/1212.0402 (2012). arXiv: 1212.0402.

[18] Du Tran et al. "C3D: Generic Features for Video Analysis". In: *CoRR* abs/1412.0767 (2014). arXiv: 1412.0767.

[19] H. Wang and C. Schmid. "Action Recognition with Improved Trajectories". In: *2013 IEEE International Conference on Computer Vision*. 2013, pp. 3551–3558. DOI: 10.1109/ICCV.2013.441.

[20] W. Xiao et al. "Online Robust Principal Component Analysis with Change Point Detection". In: *IEEE Transactions on Multimedia* (2019), pp. 1–1. DOI: 10.1109/TMM.2019.2923097.

[21] H. Yang et al. "Unsupervised Extraction of Video Highlights via Robust Recurrent Auto-Encoders". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 4633–4641. DOI: 10.1109/ICCV.2015.526.

[22] L. Yao et al. “Describing Videos by Exploiting Temporal Structure”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 4507–4515. DOI: 10.1109/ICCV.2015.512.

[23] J. You et al. “A Multiple Visual Models Based Perceptive Analysis Framework for Multilevel Video Summarization”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 17.3 (2007), pp. 273–285. ISSN: 1051-8215. DOI: 10.1109/TCSVT.2007.890857.

[24] Ke Zhang et al. “Summary Transfer: Exemplar-based Subset Selection for Video Summarization”. In: *CoRR* abs/1603.03369 (2016). arXiv: 1603.03369.

[25] Ke Zhang et al. “Video Summarization with Long Short-Term Memory”. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 766–782.

[26] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. “Hierarchical Recurrent Neural Network for Video Summarization”. In: *Proceedings of the 2017 ACM on Multimedia Conference - MM ’17* (2017). DOI: 10.1145/3123266.3123328.

[27] Kaiyang Zhou and Yu Qiao. “Deep Reinforcement Learning for Unsupervised Video Summarization with Diversity-Representativeness Reward”. In: *CoRR* abs/1801.00054 (2018). arXiv: 1801.00054.