Indonesian Journal of Electrical Engineering and Computer Science
Vol. 35, No. 1, July 2024, pp. 274~283
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v35.i1.pp274-283

Feature fusion-based video summarization using SegNetSN

Sheetal Pravin Girase, Mangesh Bedekar
Department of Computer Engineering and Technology, Dr. Vishwanath Karad World Peace University, Pune, India

Article Info

Article history:

Received Jan 28, 2024
Revised Mar 5, 2024
Accepted Mar 14, 2024

Keywords:

Feature fusion
SegNetSN
Semantic segmentation
Sequence network
Video summarization

ABSTRACT

This paper addresses the video summarization problem. For the given video goal is to find the subset of frames that capture the important events of the input video and produce a small concise summary. We formulate video summarization as a sequence labeling problem, where for a given input video a subset of frames are selected as a summary video. Based on the principle of semantic segmentation, here each pixel within a frame is assigned to one of the labels, where each frame is assigned a binary label indicating whether it will be included in the summary video or not. We propose a SegNet sequence network (SegNetSN) for video summarization and further extend the work by applying various feature fusion techniques to enhance the input. We performed experiments on the benchmark dataset TVSum.

Corresponding Author:

Sheetal Pravin Girase
Department of Computer Engineering and Technology, Dr. Vishwanath Karad World Peace University
Pune, 411038, Maharashtra, India
Email: sheetal.girase@mitwpu.edu.in

1. INTRODUCTION

In today's world of technology, the creation and use of huge visual as well as video data has increased dramatically. For example, on YouTube for example, there are 400 hrs of videos are uploaded every minute, and over one billion hours of videos are watched every day in the year of 2018 [1]. Every day millions of videos are generated and shared by users. Efficient browsing and searching of data from these videos is ever challenging problem which is been addressed by the research community from time to time. The data generated by surveillance cameras for monitoring elderly people, traffic monitoring, and several other similar applications suffers from the requirement of huge storage devices, and analyzing these huge videos can be a time-consuming task affecting the efficiency of these video processing devices [2].

To address these issues solution is to construct algorithms that construct a summary of the videos which can provide relevant yet concise information about the topic of interest. Video summarization has evolved in recent years providing effective mechanisms to cope with this exponentially growing data. By making use of video summarization techniques, users can watch highlights of the video and they need not have to watch the whole lengthy videos making their lives easy [3]. The goal of video summarization is to create a short video of the given input video such that it conveys the context of the original video along with the important information. Video summarization is being used in security surveillance systems to detect and analyze suspicious activities and traffic monitoring. It is also used to generate sports highlights, especially soccer [4], make trailers of movies and serials, and video content indexing to facilitate fast browsing of huge videos through a video search engine, and so on [5], [6]. Video summarization is useful for many applications like analyzing the performance of the player by going through the long match videos. For humans watching these long videos can be time-consuming. Rather if we can provide a short summary of the long video it can reduce the human effort as well can give more insight to the player to improve his performance. These short summary videos are easier to store and consume less space as compared to the long videos.

Figure 1 depicts the general block diagram of supervised video summarization using deep learning which outputs the video summary as a sequence of frames. First, the input video is preprocessed such that it can be divided into frames/segments depending on the approach. Video summarization has input in the form of frames or depending upon the scene change it can also be divided into small chunks called segments. Likewise, output in the form of a summary is also a set of keyframes or key shots/key segments. In supervised learning, it is necessary to have huge data samples of videos and their summary to get effective results. The benchmark datasets come with the frame score assigned to each frame indicating their importance in the video. Based on the labeled data the deep neural network understands and generates the important frames called keyframes. The summary evaluator model evaluates the deep features and calculates the similarity score and accordingly key segments are generated and the video is reconstructed generating a summary of the input video.

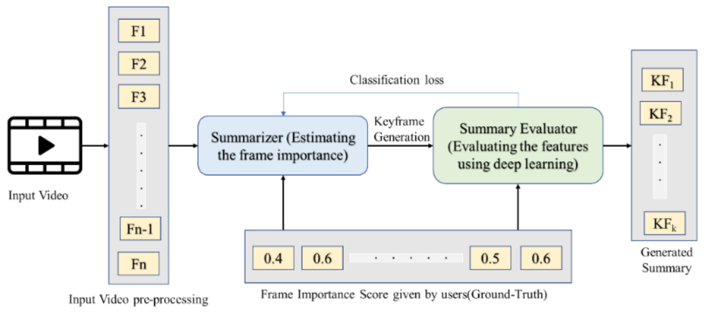

F1
F2
F3
Fn-1
Fn
Input Video
Input Video pre-processing
Classification loss
Summarizer (Estimating the frame importance)
Keyframe Generation
Summary Evaluator (Evaluating the features using deep learning)
0.4
0.6
0.5
0.6
Frame Importance Score given by users(Ground-Truth)
KF1
KF2
KFk
Generated Summary

Figure 1. General supervised video summarization block diagram

In this paper, we formulate video summarization as a sequence labeling problem, where for a given input video subset of frames are selected as a summary video. Based on the principle of semantic segmentation, where each pixel within a frame is assigned to one of the labels, here each frame is assigned with a binary label indicating whether it will be included in the summary video or not. The purpose of producing video summaries depends on the application domain. For example, in sports videos, the critical moments decide the outcome, in surveillance videos the unusual activities of the surroundings. Nowadays videos are shot by the person himself or video blogs which has led to different challenges on the characteristics that need to be considered as well as different user demands.

Video summaries can be of two types i.e. storyboard and video skim. Specifically, a storyboard is based on a set of important frames known as keyframes, and a video skim is composed of the number of representative video segments, called key shots [7]. Generation of storyboard includes three major components, selecting a set of frames, feature extraction, grouping frames, and skimming of keyframes [3]. The generation of video skims includes three components namely, segmentation of video into frames/shots, score prediction, and selection of segments [8].

Video summarization can be treated as a supervised, unsupervised, or semi-supervised problem depending on the nature of the domain and the available data. We are treating video summarization as a supervised learning problem. In recent years usage of the supervised approach has become more vital due to the release of baseline datasets like TVSum [9] and SumMe [10] as this approach is more suitable in the presence of plenty of video data and their summary. The densely annotated importance scores are widely used to train importance estimation models to generate effective summaries.

Due to the availability of computational capability and the complexity of video summarization problems, in recent years usage of deep learning-based approaches has increased exponentially. Taking this into account this literature survey mainly focuses on the research performed on video summarization using deep learning techniques. Figure 2 shows the usage of various deep learning techniques in eight years. For video summarization recurrent neural network (RNN), long short-term memory (LSTMs) are widely used whereas nowadays attention networks are quite popular. Deep neural networks have been explored to estimate the importance of frames i.e. keyframes or segments i.e. key shots within a video using temporal dependency. Zhang *et al.* [11] and Zhao *et al.* [12] used LSTM sequential modeling to predict the keyframes. Frames' importance is estimated using a multi-layer perceptron (MLP), and the diversity of the visual content of the generated summary is increased based on the determinantal point process (DPP) in an end-to-end manner. The remarkable work done by Zhao *et al.* [13] (H-RNN) includes a two-layer hierarchical structure of RNNs. The first layer is an LSTM, which encodes the intra-subshot temporal dependency whose output is given as input to the second layer, i.e. bi-directional LSTM. This layer uses the inter-subshot temporal dependency and determines whether a certain subshot is valuable to be a key subshot. In their subsequent work, Zhao *et al.* [14] hierarchical structure-adaptive RNN (HSA-RNN) integrated a component that is trained to identify the shot-level temporal structure of the video. This knowledge is then utilized for estimating importance at the shot level improving the summary quality by accurately capturing the redundancies in a video.

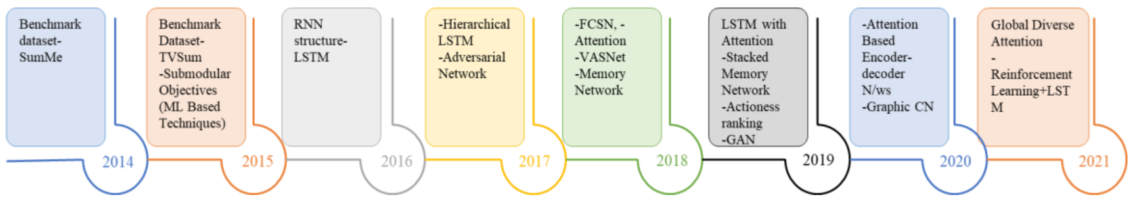

Benchmark dataset-SumMe
2014
Benchmark Dataset-TVSum -Submodular Objectives (ML Based Techniques)
2015
RNN structure-LSTM
2016
-Hierarchical LSTM -Adversarial Network
2017
-FCSN, -Attention -VASNet -Memory Network
2018
LSTM with Attention -Stacked Memory Network -Actioness ranking -GAN
2019
-Attention Based Encoder-decoder N/ws -Graphic CN
2020
Global Diverse Attention - Reinforcement Learning+LSTM
2021

Figure 2. Evolution of video summarization using deep learning techniques

To the existing hierarchical structure of RNN Zhao *et al.* [14] combined a tensor-train embedding layer to avoid large feature-to-hidden mapping matrices which is explained in Zhao *et al.* [15] (TTH-RNN). The output of these layers is used for determining the probability of each sub-shot being selected as a part of the video summary. Lebron *et al.* [16] incorporated the attention mechanism to model user interest along with the vsLSTM and dppLSTM architectures.

The challenge with the above-mentioned techniques is they give the same importance to all the frames within a video sequence irrespective of the output frames to be predicted which weakens the discrimination process of the summarizer. LSTMs also suffer from capturing long-term dependencies. To address this vanishing and exploding gradient problem of LSTM, video summarization needs to be addressed as a sequence-to-sequence learning problem. Ji *et al.* [17], Mahasenni *et al.* [18] and Zhou *et al.* [19] treated video summarization as a structure prediction problem in the Seq2Seq framework.

Ji *et al.* [7] proposed an LSTM-based encoder-decoder attention mechanism in the Seq2Seq framework. By acclimating the generative process in the decoder on the encoder's hidden states, different input frames are assigned different weights. This enhances the intrinsic relations between the input video sequence and the output keyframes. In the extension of the earlier work Ji *et al.* [7] integrate a semantic preserving embedding network and replace mean square error (MSE) loss with the Kullback–Leibler (KL) loss to mitigate the distribution inconsistency issue. To tackle the computationally complex and demanding network like BiLSTM Fajtl *et al.* [20] soft, self-attention mechanism with a two-layer fully connected network. These attention mechanisms are good at detecting the shot boundaries that face challenges while addressing complicated scene-changing videos [21]. Liu *et al.* [21] proposed the shot-level reconstruction model and multi-head attention model, which utilize multiple attention maps to improve the performance in summarization tasks. Li *et al.* [22] proposed a global diverse attention mechanism by adapting the self-attention mechanism. A pairwise similarity matrix is designed to model multi-scale temporal relations across frames by storing the context information. This framework requires less computational cost, which makes it quite efficient and suitable for GPU parallelization.

To model the dependency of video frames, along with the aforementioned approaches a different approach was proposed by Rochan *et al.* [23] who treated video summarization as a semantic segmentation task. Here the input video is seen as a 1D image (of size equal to the number of video frames) with K channels that correspond to the K dimensions of the frames' representation vectors (either containing raw pixel values or being precomputed feature vectors). Then, they used popular semantic segmentation models, such as fully convolutional networks (FCN) [24] and an adaptation of DeepLab [25], and built a network (called a fully convolutional sequence network) for video summarization. The latter consists of a stack of earlier research on video summarization focused on uniqueness, diversity, and interestingness [26]. Our contributions are as below: i) extensive experimentation of video summarization using various semantic segmentation architectures namely SegNet, ii) applying feature fusion techniques like concatenating raw image feature vectors with 5th layer of GoogleNet, the addition of raw image feature vectors with the 5th layer of GoogleNet, taking input feature vectors from the other pre-trained networks like ImageNet, ResNet150 and DCT.

2. METHOD

Our approach is based on the work done by Rochan *et al.* [23] where the author presents the video summarization as a key frame selection problem. They have established a correlation between semantic segmentation and video summarization. In the semantic segmentation task, 2D images with feature vectors are taken as input and the task is to identify labels for each pixel in the input image. Semantic segmentation attempts to apply a label to each pixel based on the object it belongs to. In video summarization, the aim is to reduce the size of the input video and generate shorter videos by identifying the important (key) frames from the input video and then using the keyframes to generate the smaller version of the input video. If we compare the task of identifying key and non-key frames with the semantic segmentation problem, we are trying to label each frame in the input video as a key or non-key frame. Although semantic segmentation and video summarization appear and are studied as very different problems in computer vision, there is a lot of similarity in these two problem domains. The semantic segmentation takes 2D images with 3D color channels (RGB) as input and the output is a 2D matrix with each cell representing the label for semantic segmentation. Similarly, we can consider the task of selecting key and non-key frames using semantic segmentation, we can look at each frame as a K-dimensional vector along the temporal dimension. Each frame can be a vector of raw image pixels or some computed features for the input image. In other words, each frame is a 1D image with K channels. The output will be the vector with a 1D matrix with length as the length of the input video and each value representing the frame as key or non-key.

Extending the approach of Rochan *et al.* [23] to the different network architectures of the semantic segmentation domain and evaluating the results of video summarization using these various models by following the architecture depicted in Figure 3. The original work of Rochan *et al*. [23] used the output of the fifth layer of GoogLeNet architecture to get the feature vector of the input video. Here we propose various other architectures to get the feature vector as well as feature fusion techniques to combine the image features in a multimodal fashion. It has been discussed in detail in the forthcoming sections. The basic architecture is shown as in Figure 4 and the modules are explained in the subsequent sections.

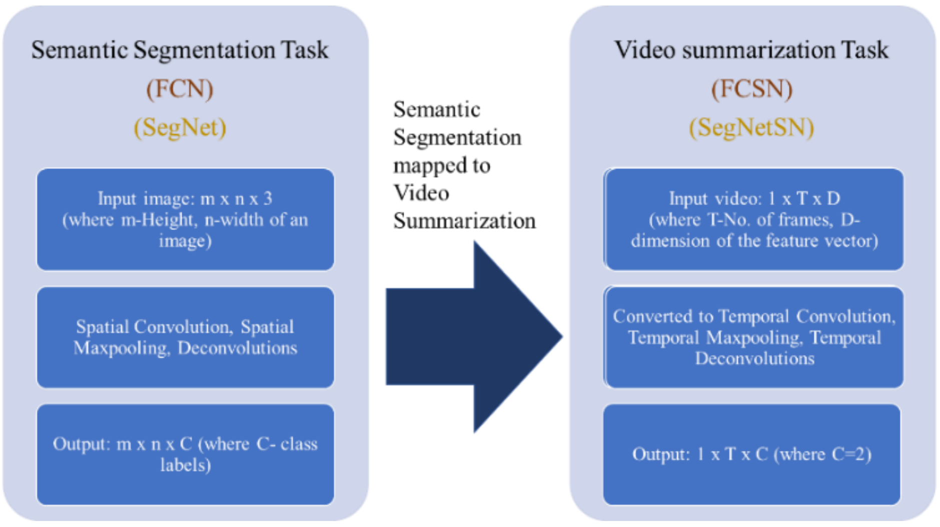

Semantic Segmentation Task
(FCN)
(SegNet)
Input image: m x n x 3 (where m-Height, n-width of an image)
Spatial Convolution, Spatial Maxpooling, Deconvolutions
Output: m x n x C (where C- class labels)
Semantic Segmentation mapped to Video Summarization
Video summarization Task
(FCSN)
(SegNetSN)
Input video: 1 x T x D (where T-No. of frames, D-dimension of the feature vector)
Converted to Temporal Convolution, Temporal Maxpooling, Temporal Deconvolutions
Output: 1 x T x C (where C=2)

Figure 3. Correlation between the semantic segmentation and video summarization

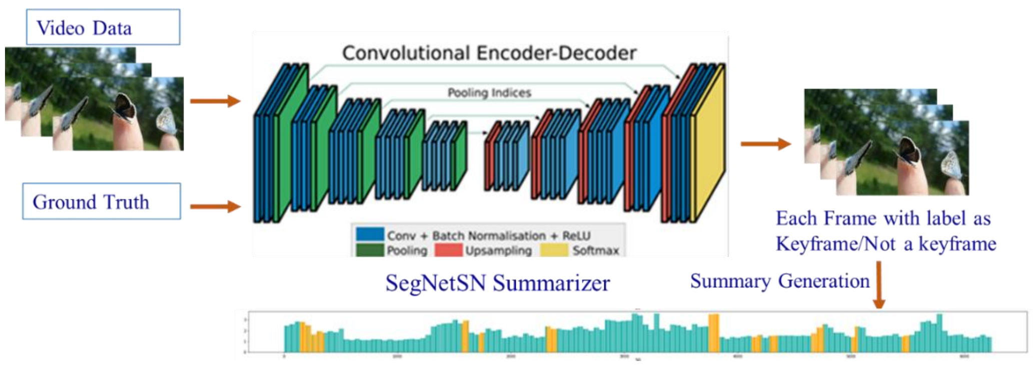

Video Data
Ground Truth
Convolutional Encoder-Decoder
Pooling Indices
Conv + Batch Normalisation + ReLU
Pooling
Upsampling
Softmax
SegNetSN Summarizer
Each Frame with label as Keyframe/Not a keyframe
Summary Generation

Figure 4. Video summarization using SegNetSN architecture diagram

2.1. Data preparation and model training

To generate the input feature vectors for training, the frames of the input video are passed through pre-trained networks like GoogLeNet/Densenet. GoogLeNet, is also known as Inception Net. It is a convolutional neural network (CNN) with 22-deep layer architecture and was trained on the ImageNet dataset. It can classify objects into 1,000 different categories. Densely Connected Convolutional Networks (DenseNet) is a feed-forward CNN architecture that links each layer to every other layer. A feed-forward CNN design called densely connected convolutional networks (DenseNet) connects each layer to every other layer. This lowers the number of parameters and improves the gradient flow during training by enabling the network to learn more efficiently by reusing features. Dataset is prepared in the form of a .h5 file which acts as an input for training and testing of the SegNetSN Model. This .h5 file contains multiple groups, each group represents one video. Each group comprises the input features, ground truths, Change points, user summary, and frames per segment data. The sequence of steps executed to get this .h5file is described below. For feature fusion, one additional step is to apply the feature fusion techniques like addition/concatenation or Discrete Cosine Transform (DCT) on extracted feature vectors i.e. Algorithm1 and Algorithm2 below. Once the data has been prepared it is fed to the training model of SegNetSN for summarization purposes. Algorithm 3 returns a trained model with evaluation matrix.

Algorithm 1. Data preparation without feature fusion
```
Inputs: Video V, Ground Truth G, Change Points (CPs),
Output: .h5 file
Start Process:
For each video:
    Down sample video to 2fps
   For each frame in input Video
      Extract feature Vector (F) using GoogLeNet or DenseNet feature extractor
     Read the ground truth g, Read CPs, User Summary, Frames per segment etc.
    Calculate Oracle Summary from user summary data
Select 320 frames out of the total frames retrieved after down sampling to 2 fps
       Write extracted feature vector, Ground Truths, Oracle summary, CPs, frames per segment
etc to .h5 file
```

Algorithm 2. Data preparation with feature fusion
```
Inputs: Video V, Ground Truth G, Change Points (CPs), Feature Fusion Operation,
       Output: .h5 file
Start Process:
For each video:
    Down sample video to 2fps
   For each frame in input Video
          Extract feature Vector (F) using GoogLeNet or DenseNet feature extractor
         Apply the Feature fusion (addition or concatenation or DCT) to frame feature and
output of GoogLeNet/DenseNet
         Read the ground truth g, Read CPs, User Summary, Frames per segment, etc.
        Calculate Oracle Summary from user summary data
        Select 320 frames out of the total frames retrieved after downsampling to 2 fps
       Write extracted feature vector, Ground Truths, Oracle summary, CPs, frames per segment
etc to the .h5 file
```

Algorithm 3. Training and validation loop using SegNetSN model
```
       Inputs: .h5 file containing the details of frame features, Ground Truth, Change Points
(CPs) for the selected dataset, optimizer to be used, hyperparameters like number of epochs,
batch size, learning rate, and decay,
       Output: trained model, training and validation metrics like loss, accuracy,
precision, recall and F1-Score.
Start Process:
Load the train and validation dataset with an 80:20 split
For each epoch:
   Load the batch of training data and pass it through the network
      Calculate the loss using the loss function
     Calculate the metrics loss, accuracy, precision, recall, and F1-score
  Load the batch of validation data and pass through the network in validation mode
     Using the predicted values for the frames, we select the key shots
    Calculate the metric specific to the Video Summarization using the user ratings,
predicted values, and selected key shots.
```

2.2. Loss function

A video can be represented as a sequence of frames as $V=\{v_1, v_2, \ldots, v_t, \ldots, v_T\}$, where $t \in 1, 2, \ldots, T$ and v_t is the frame at time t. The sequence of frames of V, can be represented as raw features for each frame as $I=\{i_1, i_2, i_3\ldots, i_t, \ldots., i_T\}$. The sequence of frames of V, can be represented by high level features as $F=\{f_1, f_2, \ldots,$

f_t, …, f_T}, where $f_t \in \mathbb{R}^D$ is a vector of dimension D, that represents the extracted features from the t-th frame based GoogleNet's pool five layer/ DenseNet's adaptive average pool 2D layer. For GoogleNet the value of D=1,024 while for the DenseNet D=2,048. The task of the model is to produce binary classification result as key or non-key for every frame. $\hat{Y}=\{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_t, \ldots, \hat{y}_T\}$ as an output that represents the binary classification result i.e. either 1 (key) or 0 (non-key). The ground truth is represented as $y=\{y_1, y_2, \ldots y_t, \ldots, y_T\}$.

Video summarization task is a binary classification task (key vs non-key frames). Binary cross entropy error has been used for calculating the loss during each batch of training. The losses are averaged over the multiple batches in a single epoch of execution.

$$L(y,\hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}(y_i \,.\log(\hat{y}_i) + (1-y_i).\log(1-\hat{y}_i)) \quad (1)$$

Where, y is the ground truth label, ŷ is the predicted label, N is the number of samples.

3. RESULTS AND DISCUSSION

3.1. Datasets

In the domain of video summarization, the benchmark datasets that are used are TVSum [9] and SumMe [27]. These datasets are been used for training, testing, and experimental purposes. TVSum dataset has 50 videos of various genres (news, sports, how-to, vlog, egocentric, and documentary). The videos are from 1 to 5 minutes in duration and each video is annotated by 20 users to provide the shot level importance scores for each frame (ranging from 1 to 5) in the video. SumMe dataset has 25 videos of 1 to 6-minute duration with 15 user summaries for each video (390 in total). The datasets are randomly shuffled and for training and testing 80:20 ratio is considered. i.e. 80% videos for training and 20% videos for testing purposes.

3.2. Experimental setup

The input videos are down-sampled to 2 fps and 320 frames are selected equally from each video. As part of the experimentation, the feature, vector F is generated using GoogLeNet and DenseNet for each video. The hyperparameters that are used for the experiments include optimizers i.e. Adam and SGD both, learning rate=0.001, and exponential decay=0.96 with 1,000 steps. Also, the batch size of 5 and the number of epochs used are 250. Further, the testing dataset is used to evaluate the trained models for 20 runs each and we report the average and the maximum value of the evaluation metrics.

3.3. Metrics

The evaluation module executes the test data over the trained model. By following the approach of [8], [21] predicted key frames for the test data have been converted to the key shots so that fair comparison can be done with existing approaches. Video can be viewed as a series of shots identified by the change point vectors. Change points divide the complete video into multiple video segments. The predicted score is upsampled over the complete length of the video. Using the upsampled predicted score, if any of the frames in the video segment(shot) is predicted as key-frame then all the frames within that video segment are marked as key-frames. Out of the total number of frames of the input video, 15% of the frames are selected using the knapsack algorithm [9] to get the selected frames. For each video, the JSON file contains the predicted score and the user summary. Using this data, evaluation metrics like loss, accuracy, precision, recall and, F1-score are calculated. For the quantitative evaluation we, use F1-score as the metric as per [10], [18], [26] which are defined as follows:
Accuracy is defined as the ratio of correctly identified frames to the total number of frames in a video:

$$Accuracy = \frac{\#\, of\ correctly\ identified\ key\ frames + \#\, of\ correctly\ identified\ nonkey\ frames}{Total\ number\ of\ frames} \quad (2)$$

Precision is defined as:

$$Precision = \frac{\#\, of\ correctly\ identified\ key\ frames}{\#\, of\ correctly\ identified\ key\ frames + \#\, of\ incorrectly\ identified\ non-key\ frames} \quad (3)$$

Recall is defined as:

$$Recall = \frac{\#\, of\ correctly\ identified\ keyframes}{\#\, of\ correctly\ identified\ keyframes + \#\, of\ incorrectly\ identified\ key\ frames} \quad (4)$$

F1-score is defined as:

$$F1\ Score = 2 * \frac{Precision*Recall}{Precision+Recall} \tag{5}$$

As mentioned in the Table 1, the experiments are performed taking into consideration input video in the form of a feature vector extracted from GoogleNet and/or DenseNet architecture. Also, the raw image has been given as an input to the video summarization module. We implemented and tested the fully convolution sequential network (FCSN) model proposed by Rochan *et al.* [23] for the validation purpose. Later, we have gone ahead with the implementation of SegNetSN architecture for video summarization. Table 2, shows the observed average and max Fscore for both the networks i.e. SegNetSN and the FCSN.

For each test, we have taken 20 runs and reported the average and max F-score. SegNetSN performs better with SGD optimizer whereas FCSN performs better with Adam optimizer. From Figure 5, it is evident that grayscale images alone cannot perform better than the feature vectors from GoogleNet and/or DenseNet.

Table 1. Details of experiments performed with following settings

Optimizer	SGD	Adam	
Dataset	TVSum	SumMe	
Model	FCSN	SegNetSN	
Feature extraction model	GoogleNet	DenseNet	Gray scale image of 32 by 32 (RAW image)

Table 2. Comparison with other input features

Model	SegNetSN				FCSN			
Optimizer	SGD		Adam		SGD		Adam	
	Avg	Max	Avg	Max	Avg	Max	Avg	Max
Gray scale image of 32 by 32	53.95	57.37	54.02	57.15	53.05	56.10	53.86	57.05
Feature vector from DenseNet	52.92	57.73	8.39	8.39	52.77	55.17	57.12	58.85
Feature vector from pool5 of GoogleNet	54.13	57.92	38.93	57.03	41.40	54.87	57.63	58.06

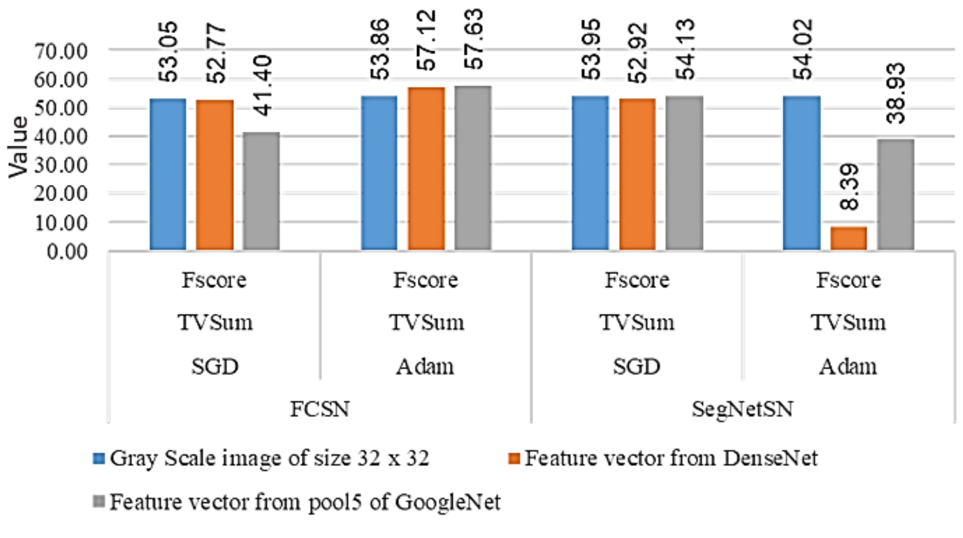


Figure 5. Comparison of FCSN and SegNetSN for various input features

We studied earlier approaches w.r.t. training, testing, validation, and splits used for experimentation, as well as the feature set as shown in Table 3. The clustering-based technique was proposed by De Avila *et al.* [28]. Zhang *et al.* [29] proposed vsLSTM and extended it to dppLSTM which does not show much improvement in the F1-score. Zhang *et al.* [12] combined the TVSum dataset with SumMe and MED and changed the split still it did not show a major improvement in the F1-score. For the performance improvement feature fusion techniques can be used which combines the shallow and deep features of the video frames.

Table 3. Comparison of F1-score with earlier approaches

Authors	Existing models	TVSum	Training	Testing	Validation	Split	Feature set used
De Avila *et al.* [28]	VSUMM (2010)	39.1	-	-	-	-	Color features, color histogram in HSV color space
Zhang *et al.* [29]	vsLSTM (2016)	54.0	√	√	√	80:20	Shallow features, color histogram, optical flow and SIFT features. Deep features, GoogLeNet is employed to extract the frame features.
Zhang *et al.* [12]	dppLSTM (2016)	54.7	√	√	-	80:20	Shallow features, color histogram, optical flow, and SIFT features. Deep features, GoogLeNet is employed to extract the frame features
Zhao *et al.* [13]	H-RNN (2017)	54.9	√	√	-	77:23	Shallow features, color histogram, optical flow and SIFT features. Deep features, GoogLeNet is employed to extract the frame features.
Rochan *et al.* [23]	FCSN (2018)	56.9	√	√	-	80:20	Shallow features, color histogram, optical flow and SIFT features. Deep features, GoogLeNet is employed to extract the frame features.

3.4. Feature fusion techniques

Fusion is a process of combining the specific extracted features that are stored in a dictionary to obtain a single feature file, which is very informative [30]. Concatenation, addition, and discrete cosine transform (DCT) techniques are employed to improve the feature vectors. We performed experimentations by the fusion of shallow features from raw images along with deep features from GoogleNet/DenseNet and also, DCT transformation. The combinations of feature fusion techniques and their results are discussed:

a. Concatenation: In this technique, raw image features are concatenated with pool5 of GoogleNet. This can be defined as concatenation of feature vector F and raw image vector I as outlined (6),

$$X = concat(F, I) \tag{6}$$

– GoogleNet+raw image (i.e. grayscale image of 32×32).
– DenseNet+raw image (i.e. grayscale image of 32×32).

b. Addition: This technique applies as element-wise addition of feature vector F and raw image feature vector I as outlined (7).

$$X = F + I \tag{7}$$

– GoogleNet+raw image (i.e. grayscale image of 32×32).
– DenseNet+raw image (i.e. grayscale image of 32×32).

c. DCT: This technique makes use of the DCT transforms on raw image features as outlined (8).

$$X = DCT(F, I) \tag{8}$$

– DCT with GoogleNet.
– DCT with DenseNet.

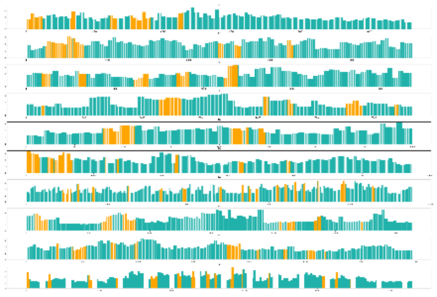
Table 4, shows the observed F1-score for the SegNetSN architecture with all feature fusion techniques. Here DenseNet outperforms than the other techniques. Table 4, shows the observed F1-score for the SegNetSN architecture with all feature fusion techniques using TVSum dataset. Raw image features concatenated with the Deep features extracted from GoogleNet and/or DenseNet do not show major improvements. As demonstrated by Apostolidis *et al.* [31] the performance of the F-score is significantly affected by how the dataset is split into the training and testing splits. Since random data splits are created and evaluated in each paper it results in varying scores of evaluation metrics (namely F-score).

Table 4. Feature fusion: SegNetSN observations using SGD optimizer

Input	F1 score	
	Avg.	Max
TVSum	56.50	57.10
TVSum_Raw	56.45	56.80
TVSum_Raw_ Concat_DenseNet	56.50	57.40
TVSum_Raw_Concat_GoogleNet	56.26	56.40

3.5. Qualitative analysis

For testing purposes, the summary is generated for randomly selected 10 testing videos as shown in Figure 6, Figures 6(a) and 6(b) for FCSN and SegNetSN architectures respectively. The score bar shows the frame index on the x-axis and the user's score on the y-axis. The columns on the y-axis are marked as orange denoting the selected key shots by the respective algorithm. From the results, it is clear that SegNetSN performs equivalent to FCSN.


(a)

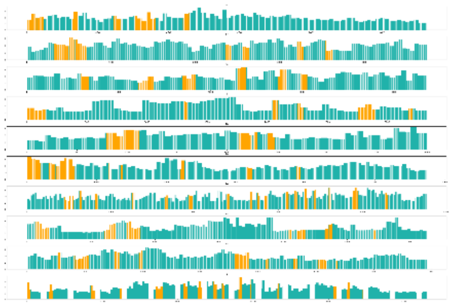

(b)

Figure 6. Score bar for 10 testing videos of TVSum (a) FCSN and (b) SegNetSN

4. CONCLUSION

We have introduced here the SegNetSN which is lightweight as compared to FCSN. This addresses the relation between the two separate problems of computer vision i.e. semantic segmentation and video summarization. SegNetSN can be used as an alternative for LSTM. Our model achieves performance very close to FCSN. We also experimented with various feature fusion techniques and proposed various fusion techniques to improve the video input data. With various fusion experiments though the F-score has not shown drastic improvement. Comparing results reported in the literature with the results reported in this paper vary drastically owing to varying degrees of data splits namely training and testing.

After the pandemic, video conferencing, vlogging, creation, and usage of video have increased for domestic as well as for business purposes. In the future developing video summarization for emerging applications and evaluating summaries with appropriate evaluation metrics is still a challenge. The output of a video summary is more subjective than objective based on user perception on the application domain.

REFERENCES

[1] Z. Ji, F. Jiao, Y. Pang, and L. Shao, “Deep attentive and semantic preserving video summarization,” *Neurocomputing*, vol. 405, pp. 200–207, Sep. 2020, doi: 10.1016/j.neucom.2020.04.132.

[2] R. Panda and A. K. Roy-Chowdhury, “Multi-view surveillance video summarization via joint embedding and sparse optimization,” *IEEE Transactions on Multimedia*, vol. 19, no. 9, pp. 2010–2021, Sep. 2017, doi: 10.1109/TMM.2017.2708981.

[3] M. Kini M. and K. Pai, “A survey on video summarization techniques,” in *2019 Innovations in Power and Advanced Computing Technologies (i-PACT)*, Mar. 2019, pp. 1–5. doi: 10.1109/i-PACT44901.2019.8960003.

[4] S. Jai-Andaloussi, A. Mohamed, N. Madrane, and A. Sekkaki, “Soccer video summarization using video content analysis and social media streams,” in *2014 IEEE/ACM International Symposium on Big Data Computing*, Dec. 2014, pp. 1–7. doi: 10.1109/BDC.2014.20.

[5] V. Sharma and M. Singh, IEEE 2018 international conference on advances in computing, communication control and networking (ICACCCN): On 12th & 13th Oct. 2018 : Proceedings. IEEE, 2018

[6] A. Bora and S. Sharma, “A review on video summarization approcahes: recent advances and directions,” in *2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN)*, Oct. 2018, pp. 601–606. doi: 10.1109/ICACCCN.2018.8748574.

[7] Z. Ji, K. Xiong, Y. Pang, and X. Li, “Video summarization with attention-based encoder-decoder networks,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1709–1717, Jun. 2020, doi: 10.1109/TCSVT.2019.2904996.

[8] A. G. Money and H. Agius, “Video summarisation: a conceptual framework and survey of the state of the art,” *Journal of Visual Communication and Image Representation*, vol. 19, no. 2, pp. 121–143, Feb. 2008, doi: 10.1016/j.jvcir.2007.04.002.

[9] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes, “TVSum: summarizing web videos using titles,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5179–5187.

[10] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, “Creating summaries from user videos,” in *ECCV 2014: Computer Vision – ECCV 2014*, 2014, pp. 505–520. doi: 10.1007/978-3-319-10584-0_33.

[11] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, “Video summarization with long short-term memory,” in *ECCV 2016: Computer Vision – ECCV 2016*, 2016, pp. 766–782. doi: 10.1007/978-3-319-46478-7_47.

[12] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, “Supplementary material: video summarization with long short-term memory,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 1–7.

[13] B. Zhao, X. Li, and X. Lu, "Hierarchical recurrent neural network for video summarization," in *Proceedings of the 25th ACM international conference on Multimedia*, Oct. 2017, pp. 863–871. doi: 10.1145/3123266.3123328.
[14] B. Zhao, X. Li, and X. Lu, "HSA-RNN: hierarchical structure-adaptive RNN for video summarization," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp. 7405–7414. doi: 10.1109/CVPR.2018.00773.
[15] B. Zhao, X. Li, and X. Lu, "TTH-RNN: Tensor-train hierarchical recurrent neural network for video summarization," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 4, pp. 3629–3637, Apr. 2021, doi: 10.1109/TIE.2020.2979573.
[16] L. Lebron Casas and E. Koblents, "Video summarization with LSTM and deep attention models," in *MMM 2019: MultiMedia Modeling*, 2019, pp. 67–79. doi: 10.1007/978-3-030-05716-9_6.
[17] Z. Ji, K. Xiong, Y. Pang, and X. Li, "Video summarization with attention-based encoder–decoder networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 6, pp. 1709–1717, Jun. 2020, doi: 10.1109/TCSVT.2019.2904996.
[18] 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)*, Jul. 2017, pp. 2982–2991. doi: 10.1109/CVPR.2017.318.
[19] K. Zhou, Y. Qiao, and T. Xiang, "Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018, doi: 10.1609/aaai.v32i1.12255.
[20] J. Fajtl, H. S. Sokeh, V. Argyriou, D. Monekosso, and P. Remagnino, "Summarizing videos with attention," in *ACCV 2018: Computer Vision – ACCV 2018 Workshops*, 2019, pp. 39–54. doi: 10.1007/978-3-030-21074-8_4.
[21] Y.-T. Liu, Y.-J. Li, and Y.-C. F. Wang, "Transforming multi-concept attention into video summarization," in *ACCV 2020: Computer Vision – ACCV 2020*, 2021, pp. 498–513. doi: 10.1007/978-3-030-69541-5_30.
[22] P. Li, Q. Ye, L. Zhang, L. Yuan, X. Xu, and L. Shao, "Exploring global diverse attention via pairwise temporal relation for video summarization," *Prepr. arXiv.2009.10942*, Sep. 2020.
[23] M. Rochan, L. Ye, and Y. Wang, "Video summarization using fully convolutional sequence networks," in *ECCV 2018: Computer Vision – ECCV 2018*, 2018, pp. 358–374. doi: 10.1007/978-3-030-01258-8_22.
[24] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 3431–3440. doi: 10.1109/CVPR.2015.7298965.
[25] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, Apr. 2018, doi: 10.1109/TPAMI.2017.2699184.
[26] M. Gygli, H. Grabner, and L. Van Gool, "Video summarization by learning submodular mixtures of objectives," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 3090–3098. doi: 10.1109/CVPR.2015.7298928.
[27] M. Rochan, L. Ye, and Y. Wang, "Video Summarization Using Fully Convolutional Sequence Networks," in ECCV 2018: 15th European Conference, ACM Digital Library Proceedings, Part XII, Munich, Germany, Aug. 2018, pp. 358–374. doi: https://doi.org/10.1007/978-3-030-01258-8_22.
[28] S. E. F. de Avila, A. P. B. Lopes, A. da Luz, and A. de Albuquerque Araújo, "VSUMM: a mechanism designed to produce static video summaries and a novel evaluation method," *Pattern Recognit Lett*, vol. 32, no. 1, pp. 56–68, Jan. 2011, doi: 10.1016/j.patrec.2010.08.004.
[29] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, "Video summarization with long short-term memory," in ECCV 2016: Computer Vision – ECCV 2016, 2016, pp. 766–782. doi: 10.1007/978-3- 319-46478-7_47.
[30] D. Sudha and M. Ramakrishna, "Comparative study of features fusion techniques," in *2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT)*, Mar. 2017, pp. 235–239. doi: 10.1109/ICRAECT.2017.39.
[31] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris, and I. Patras, "Video summarization using deep neural networks: a survey," *Proceedings of the IEEE*, vol. 109, no. 11, pp. 1838–1863, Nov. 2021, doi: 10.1109/JPROC.2021.3117472.

BIOGRAPHIES OF AUTHORS

Sheetal Pravin Girase is an Assistant Professor at the School of Computer Engineering and Technology at Dr. Vishwanath Karad MIT World Peace University. She graduated from SSVPsCE, Dhule, Maharashtra, and completed her master's from VIT Pune, India. Her primary research interests include recommendation systems, databases, and computer vision. She has authored numerous national and international publications. She can be contacted at email: sheetal.girase@mitwpu.edu.in.

Mangesh Bedekar received his Ph.D. in Computer Science and Engineering from BITS Pilani, India. Prior to the Ph.D. program, he graduated from SSGMCE, Shegaon, and completed his master's at BITS Pilani, India. He currently works as Dean and Professor in the School of Computer Engineering and Technology at Dr. Vishwanath Karad MIT World Peace University, Pune, India. His primary research interests include web data mining, web personalization, user interface design, user interface improvements, browser customization, affective computing and information visualization. He has authored numerous national and international publications. He can be contacted at email: mangesh.bedekar@mitwpu.edu.in.