# VidSum - Video Summarization using Deep Learning

Nishit Anand
*Department of Computer Science & Engineering and Information Technology*
*Jaypee Institute of Information Technology, Noida*
Noida, India
nishitanand01@gmail.com

Rupesh Kumar Koshariya
*Department of Computer Science & Engineering and Information Technology*
*Jaypee Institute of Information Technology, Noida*
Noida, India
rkoshariya@gmail.com

Varsha Garg
*Department of Computer Science & Engineering and Information Technology*
*Jaypee Institute of Information Technology, Noida*
Noida, India
varsha.garg@jiit.ac.in

2023 Second International Conference on Informatics (ICI) | 979-8-3503-4383-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICI60088.2023.10421339

***Abstract*—The aim of video summarization is to create a short summary video which captures the essence of the original video and contains all the important events of the original video. This is helpful because, now we don't have to go through the entire video and are able to get a gist of it from just a short summary video. Current Supervised learning video summarization methods, use Convolutional Neural Networks and some supervised learning techniques use Recurrent Neural Networks in addition to them. We propose VidSum, an architecture for Video Summarization using Deep Learning. We combine Long Short Term Memory (LSTM) Networks with Convolutional Neural Networks to solve the problem of Video Summarization. Our deep learning model is able to find the temporal importance of video frames and is able to generate video summaries which are temporally coherent and contain the important parts of a video clip. In our testing, our model outperforms other models on the famous TVSum and SumMe datasets for the task of Video Summarization.**

***Keywords— Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), Video Summarization, Supervised Learning, Deep Learning***

## I. INTRODUCTION

These days, due to the advent of social media and online video services like YouTube and Netflix, we consume and interact with a lot of video data. Every minute, hundreds of hours of video footage is uploaded to the internet. Other than that, due to cameras being present everywhere, ranging from our smartphones to cheap CCTV surveillance cameras, video data has grown multifold in the past decade and continues to grow at a tremendous rate. Therefore, it has become imperative to find suitable video summarization techniques, which capture the crucial segments of a video clip and thus help save human time and effort.

Video summarization helps in efficient browsing of large amounts of video data by presenting us with a short summary video. It helps in making videos more informative, impressive and interesting by shrinking the original video to a few significant frames.

Video Summarization has many useful real world applications. One very pertinent scenario is video surveillance. It is very time consuming and inefficient for humans to go through 24 hour or even 72 hour video surveillance footage. But using Video Summarization, investigating professionals only need to watch the summary video and don't need to watch hours of video footage to find who committed the crime or any important activity in the long surveillance footage as the Video Summarization model will find the important moments in the surveillance footage of, say, a parking lot for example, and extract the important scenes to include the in the summary video. This would be very helpful in legal matters and would expedite the investigation process, immensely helping the judicial system.

Video Summarization can also be used to create automatically generated movie trailers. This would reduce the load on directors and movie editors. It can also be used to create beautiful video summaries of birthday parties and marriages from their original 6-7 hour long videos. Apart from that, video summarization can also be used in creating image and video carousels/ combined short 1-2 minute videos directly on our smartphone from our smartphone's photo gallery, similar to how they are made on our smartphone's photo gallery currently, but the output summary video would be much better.

Thus, video summarization is very helpful in the real world and is an ongoing topic of research, with many researchers contributing to it and working on it.

## II. RELATED WORK

Many different approaches have been used for video summarization. Kanafani et al [2] have used Unsupervised Learning [16] for video summarization. Jadon et al [12] and Shemer et al [13] have also used Unsupervised Learning to find the important scenes in a video clip.

Apart from that, many research papers in the field of Video summarization have used Supervised Learning [20], where they have used Convolutional Neural Networks [17], Recurrent Neural Networks [18] etc. Zhu et al [4] and Elfeki et al [15] have used supervised learning for creating video summaries. Huang et al [14], Rochan et al [10] and Fajtl et al [11] have used Convolutional Neural Networks, while Vasudevan et al [9] have used Recurrent Neural Networks in addition to Convolutional Neural Networks.

Some very unique approaches have also been used for video summarization. Papalampidi et al [7] have used graphs, where graph nodes depict scenes in a video clip. Hohman et al [8] have extracted colors and text from famous TV shows videos in order to summarize them. Apostolidis et al [3] and Fu et al [1] have used General Adversarial Networks [22] for training their networks for creating video summaries.

Other than that, Reinforcement Learning [21] has also been used by a few researchers to solve the problem of video summarization. Lan et al [5] and Zhou et al [6] have used Reinforcement Learning for training their deep learning models.

All of them, unsupervised, supervised and reinforcement learning methods have been applied to video summarization. But the most effective are supervised learning methods. We have used supervised learning because the model is able to learn much better in this scenario as compared to other methods. Supervised Learning has shown some of the best results till now in the field of video summarization and is one of the most widely used approaches to solve the problem of video summarization. We have used Long Short Term Memory (LSTM) Networks [19] in addition to Convolutional Neural Networks in our model. We chose Long Short Term Memory (LSTM) Networks over other options because:

A. *Recurrent Neural Networks suffer from the problem of vanishing gradients, due to which they are not able to learn on long data sequences. LSTM solves the Vanishing Gradient Problem as they have a direct connection to the activation function of the forget gate.*

B. *LSTM Networks perform much better than Gated Recurrent Unit (GRU) [23] on large datasets. They have high accuracy on datasets having huge amounts of data.*

C. *LSTM Networks, due to their inherent design are able to process sequential data and make predictions on them as compared to Convolutional Neural Networks (CNN) which can only find spatial features and find spatial patterns in image and speech data.*

## III. Proposed Approach

Our proposed solution is to first divide a video into its constituent video frames. These video frames are then entered into the Convolutional Neural Network, where Convolutional Layers extract spatial features from these frames. These extracted features are then fed to the LSTM network, in order to determine time based importance of these frames. After that, each of these images are classified into a yes or no, whether they will be included in the final summary or not by computing their Temporal Intersection over Union with the ground truth. In the last step, all the images which were classified as yes are compiled to create the summary video.

Our proposed solution consists of 5 steps:

### A. *Extracting Frames from video file:*

Any given video consists of a number of video sequences and each video sequence consists of a number of images. So we first divide the video into sequences. Then we extract images from the video. Thus, we get an image representation of a video in the form of all the frames, which the video is made up of. For this we use the OpenCV library of Python.

Most CCTV videos are 24fps (frames per second) which means that 1 second of video footage consists of 24 images. Thus 24 images are extracted from every second of video, and are thus put into the model. Our model supports other frame rates like 30fps and 60fps as well.

### B. *Feature Extraction from the extracted frames:*

Our model needs to learn the features from these extracted frames. For this, we use convolutional neural networks to extract spatial features from the images.

Now, our model starts learning from the frames extracted from the video in the above step. We have also added dropout layers to stop the model from overfitting. The output of these layers gives us the features that our model has extracted from the image frames.

### C. *Temporal interest proposals:*

Now we need to determine which frames are the most important and need to be included in the summary. We use LSTM (Long Short Term Memory) cells in this case. We use LSTM because they are able to pick up information from the previous LSTM cells also. This is important because in a video or in a movie, time based interest video portions are very important.

In order to determine whether a scene is important or not, we need to know what scene happened before that scene. Thus, we have used LSTM and so, our model has "memory" and can remember the previous scenes, so it can better determine which scenes in the video have a higher interest value in the summary. The model then proposes which frames are interesting and important in a time coherent manner with the help of LSTM.

### D. *Classification on the basis of Temporal Intersection over Union:*

Finally we classify if a frame is going to be put in the final summary or not. The above frames are compared with the Ground Truth summary frames given in the dataset and if their Temporal Intersection over Union (tIOU) is more than 0.6, then we consider that proposal to be positive and is given a score of 1 and thus gets included in the final summary.

If the Temporal Intersection over Union is less than 0.6, then that frame is considered negative and is given a score of 0 and thus is not included in the final summary. Thus, for training our model, we have kept the Temporal Intersection-Over-Union Threshold at 60%.

### E. *Amalgamation of Frames:*

All the frames which are positive and given a score of 1, need to be combined to make the summary video. We use the OpenCV library to combine these frames together and thus the output is the final video summary.

The Flowchart for our Proposed Approach of VidSum - Video Summarization is shown in Fig. 1.

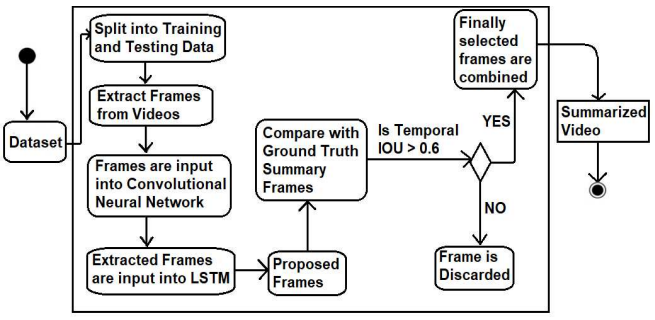


Fig. 1. Proposed Approach Flowchart of VidSum - Video Summarization using Deep Learning

## IV. Implementation

These are the steps we implemented for our proposed solution:

### A. Collecting the datasets:

For training our model we need training data. For this we need video files along with human created ground truth summaries for the video files. We need this, so that the model can learn from it and then be able to identify which frames are important in a video and also for checking Temporal Intersection over Union of the proposed frames with the ground truth frames. So we collected the TVSum [24] and SumMe [25] Datasets which are very famous datasets used for video summarization which contain videos of various genres along with human summaries for each video.

For validation purposes, we also need testing data, so that we can check the performance of our model and see how well it can propose summary frames and create a temporally coherent summary. We divided the TVSum and SumMe Datasets into training and testing data for training and testing purposes. We checked if any video files were corrupt and sorted all the video files along with their human created ground truth user summaries. This is important to be done, so that the model can learn and get trained on the input videos.

### B. Preprocessing the input video files:

We wrote a python program to extract image frames from the video. We used the OpenCV python library for this. This is done, so that the Convolutional Neural Network can learn from these extracted frames.

### C. Detecting and Extracting Features:

We wrote a custom Machine learning Model made up of Convolutional layers and Maxpool2D layers in between them. The Convolutional Neural Network extracts spatial features from the images and trains on them, so that it can better predict the right frame to be selected for the summary video.

Fig. 2 shows our CNN training and extracting features from the input video frames.

```
20:18:52] Epoch: 18/300 Loss: 0.6001/0.6685/1.2686
20:18:55] Epoch: 19/300 Loss: 0.5809/0.6502/1.2311
20:18:57] Epoch: 20/300 Loss: 0.5701/0.6431/1.2132
20:19:00] Epoch: 21/300 Loss: 0.5741/0.6385/1.2127
20:19:02] Epoch: 22/300 Loss: 0.5608/0.6241/1.1849
20:19:05] Epoch: 23/300 Loss: 0.5562/0.6175/1.1737
20:19:07] Epoch: 24/300 Loss: 0.5497/0.6143/1.1640
20:19:09] Epoch: 25/300 Loss: 0.5456/0.5866/1.1322
20:19:12] Epoch: 26/300 Loss: 0.5324/0.6024/1.1348
20:19:14] Epoch: 27/300 Loss: 0.5167/0.6271/1.1438
20:19:17] Epoch: 28/300 Loss: 0.5318/0.5948/1.1265
20:19:19] Epoch: 29/300 Loss: 0.5243/0.6057/1.1300
20:19:21] Epoch: 30/300 Loss: 0.5178/0.5773/1.0951
20:19:24] Epoch: 31/300 Loss: 0.5020/0.5733/1.0753
20:19:26] Epoch: 32/300 Loss: 0.4968/0.5636/1.0604
20:19:28] Epoch: 33/300 Loss: 0.5022/0.5620/1.0641
20:19:31] Epoch: 34/300 Loss: 0.4909/0.5665/1.0574
20:19:33] Epoch: 35/300 Loss: 0.4924/0.5656/1.0579
```

Fig. 2. Convolutional Neural Network training and extracting features from the frames

### D. Reducing Overfitting:

We noticed that our model was overfitting, so we added dropout layers with a chance of 0.5 in between the convolutional layers. This helps reduce overfitting. The output of these layers gives us the spatial features that our model has extracted from the image frames.

### E. Feeding these frames into LSTM:

After extracting Spatial Features via Convolutional layers, we have used LSTM to better understand which frames are temporally coherent. As LSTM has "Memory", it will know which scene occurred before the current scene and will thus be able to determine whether the current frame is important or not. The LSTM layers give output frames which it thinks should be included in the summary video. These we call as temporal interest proposal frames.

### F. Comparing with ground truth summary video frames:

These proposed frames are compared with the frames from the ground truth summary video from the dataset, and their Temporal Intersection over union is calculated as shown in Fig. 3.


Fig. 3. Comparing Proposed Frames with Ground Truth Summary Frames

### G. Classifying these frames on the basis of Temporal IOU:

If the Temporal Intersection over Union of a frame is >0.6, then we label it as positive and it gets included in the final summary video. If the Temporal Intersection over Union of a frame is <0.6, then we label it as negative and it does not get included in the final summary video. Calculation of Temporal IOU (Intersection Over Union) is shown in Fig. 4.

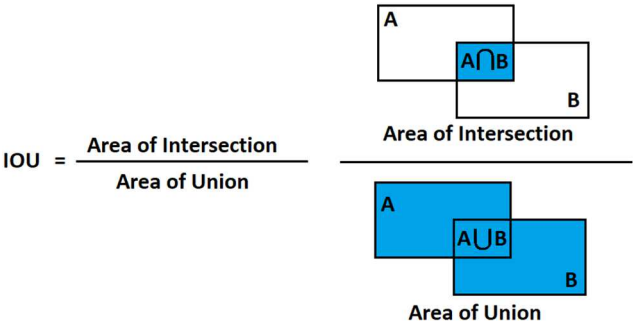


Fig. 4. Calculating Temporal Intersection over Union

### H. Creating Summary Video:

All the frames which had Temporal Intersection Over Union (tIOU) >0.6 are put into the summary video while the frames which had Temporal Intersection Over Union (tIOU)< 0.6 are discarded. After this we combine the frames. All the selected frames are combined together in the correct order to make the summary video. We use the OpenCV python library for this.

### I. *Training the model:*

We trained the model for 10,000 iterations for 30 hours. We made changes to the model multiple times and trained the model again after making changes to increase its accuracy.

We added a few dropout layers to stop overfitting. This was done in order to make the model more generalized and so that it is better able to learn the important segments in a video clip.

Fig. 5 and Fig. 6 show our model getting trained on SumMe and TVSum Dataset respectively.

```
nishit@nishit-ubuntu: ~/Downloads/src
21:00:01] Start training on summe:
21:00:02] Epoch: 0/300 Loss: 0.9570/1.1018/2.0588
21:00:03] Epoch: 1/300 Loss: 0.9430/0.9692/1.9122
21:00:04] Epoch: 2/300 Loss: 0.8813/0.8663/1.7475
21:00:04] Epoch: 3/300 Loss: 0.7956/0.8545/1.6501
21:00:05] Epoch: 4/300 Loss: 0.7778/0.8236/1.6014
21:00:06] Epoch: 5/300 Loss: 0.7550/0.8230/1.5780
21:00:07] Epoch: 6/300 Loss: 0.7383/0.7884/1.5267
21:00:08] Epoch: 7/300 Loss: 0.6949/0.8191/1.5140
21:00:09] Epoch: 8/300 Loss: 0.6799/0.7514/1.4313
```

Fig. 5. Training our model on SumMe Dataset

```
nishit@nishit-ubuntu: ~/Downloads/src
20:55:40] Start training on tvsum:
20:55:41] Epoch: 0/300 Loss: 1.0968/1.1236/2.2204
20:55:42] Epoch: 1/300 Loss: 0.9517/0.9833/1.9351
20:55:42] Epoch: 2/300 Loss: 0.9074/0.8053/1.7126
20:55:43] Epoch: 3/300 Loss: 0.8436/0.8475/1.6911
20:55:44] Epoch: 4/300 Loss: 0.8505/0.7720/1.6225
20:55:45] Epoch: 5/300 Loss: 0.7988/0.8043/1.6032
20:55:46] Epoch: 6/300 Loss: 0.7301/0.7322/1.4623
20:55:47] Epoch: 7/300 Loss: 0.7228/0.7744/1.4972
```

Fig. 6. Training our model on TVSum Dataset

## V. Testing and Observations

### A. *Testing:*

We originally split the TVSum and SumMe datasets for testing and training purposes, so now we evaluated the model by testing it against the testing dataset splits of the original datasets.

We changed hyperparameters and fine tuned the model in order to increase its accuracy. We made changes to the model multiple times and optimized it further to increase the accuracy of the model. Finally we achieved testing F-Scores of 60.3 % on the TVSum Dataset and 48.9% on the SumMe Dataset as shown in Fig. 7 and Fig. 8 respectively.

```
nishit@nishit-ubuntu: ~/Downloads/src
20:55:10] Epoch: 287/300 Loss: 0.1135/0.0254/0.1390
20:55:13] Epoch: 288/300 Loss: 0.1116/0.0244/0.1360
20:55:15] Epoch: 289/300 Loss: 0.1197/0.0235/0.1432
20:55:18] Epoch: 290/300 Loss: 0.1114/0.0246/0.1360
20:55:20] Epoch: 291/300 Loss: 0.1144/0.0239/0.1383
20:55:23] Epoch: 292/300 Loss: 0.1259/0.0246/0.1505
20:55:25] Epoch: 293/300 Loss: 0.1437/0.0241/0.1678
20:55:28] Epoch: 294/300 Loss: 0.1249/0.0239/0.1488
20:55:30] Epoch: 295/300 Loss: 0.1219/0.0242/0.1461
20:55:33] Epoch: 296/300 Loss: 0.1039/0.0229/0.1268
20:55:35] Epoch: 297/300 Loss: 0.1092/0.0235/0.1327
20:55:37] Epoch: 298/300 Loss: 0.1048/0.0236/0.1284
20:55:40] Epoch: 299/300 Loss: 0.1048/0.0228/0.1276
20:55:40] Testing done on tvsum. F-score: 0.6030
```

Fig. 7. Testing our model on TVSum Dataset

```
nishit@nishit-ubuntu: ~/Downloads/src
21:17:02] Epoch: 290/300 Loss: 0.1517/0.2907/0.4424
21:17:03] Epoch: 291/300 Loss: 0.1404/0.3141/0.4545
21:17:04] Epoch: 292/300 Loss: 0.1450/0.2959/0.4409
21:17:05] Epoch: 293/300 Loss: 0.1614/0.2837/0.4451
21:17:06] Epoch: 294/300 Loss: 0.1435/0.2835/0.4271
21:17:07] Epoch: 295/300 Loss: 0.1562/0.2933/0.4494
21:17:07] Epoch: 296/300 Loss: 0.1419/0.2966/0.4385
21:17:08] Epoch: 297/300 Loss: 0.1567/0.3053/0.4620
21:17:09] Epoch: 298/300 Loss: 0.1513/0.2942/0.4455
21:17:10] Epoch: 299/300 Loss: 0.1346/0.2852/0.4198
21:17:10] Testing done on summe. F-score: 0.4890
```

Fig. 8. Testing our model on SumMe Dataset

### B. *Observations:*

Varying Temporal IOU Threshold:

We tested and changed the Temporal IOU Threshold from 0.6 to 0.7 and 0.4, but the summary video obtained in this case was not temporally coherent, and thus produced bad results. We found that the best summarized videos were produced when the Temporal IOU Threshold was set at 0.6 . Also, the highest F-Score was achieved when Temporal Intersection Over Union threshold was set to 0.6 as shown in Fig. 9. Thus, 0.6 was the most suitable value of Temporal IOU Threshold.

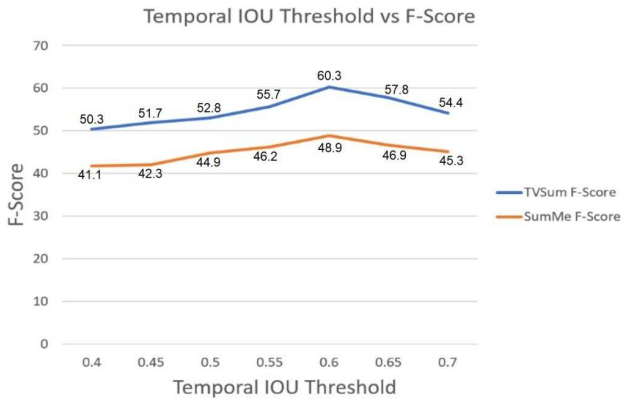


Fig. 9. Temporal IOU Threshold vs F-Score

## VI. Results

In the end, after training our model for 10,000 iterations for 30 hours and fine tuning the model, we achieved testing F-Scores of 60.3 % on the TVSum Dataset and 48.9% on the SumMe Dataset. These are higher than previous techniques in the field of Video Summarization. Table I shows our F-Score as compared to the work of some previous authors on the topic of Video Summarization on TVSum and SumMe Datasets.

TABLE I. OUR FINAL F-SCORE COMPARED TO PREVIOUS WORKS

| Model | TVSum F-Score | SumMe F-Score |
|---|---|---|
| FCSN (Rochan et al.) [10] | 52.7 | 41.5 |
| VASNet (J. Fajtl et al.) [11] | 59.8 | 47.7 |
| DR-DSN (Zhou et al.) [6] | 57.6 | 41.4 |
| Ours | 60.3 | 48.9 |

Final Testing F-Scores achieved by our model on the TVSum and SumMe Dataset are shown in Fig. 10 and Fig. 11 respectively.

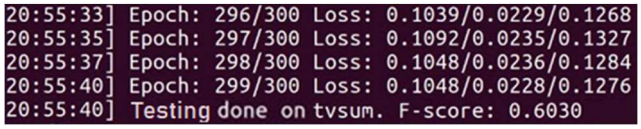


Fig. 10. Final Testing F-Scores of our model on TVSum Dataset

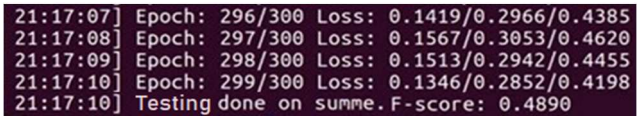


Fig. 11. Final Testing F-Scores of our model on SumMe Dataset

F-Scores achieved by our model as compared to other Video Summarization techniques on the SumMe and TVSum Dataset are shown in Fig. 12 and Fig. 13 respectively.

Fig. 14 shows screenshots from a summary video created by our model on CCTV surveillance footage.

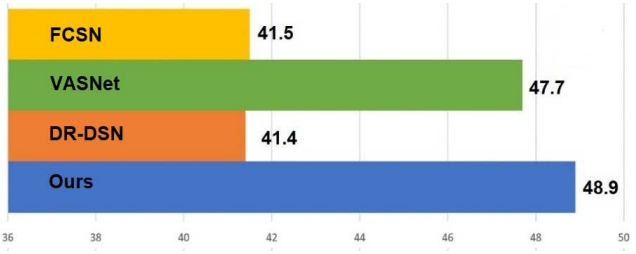


Fig. 12. Our Final F-Score as compared to other video summarization techniques on SumMe Dataset

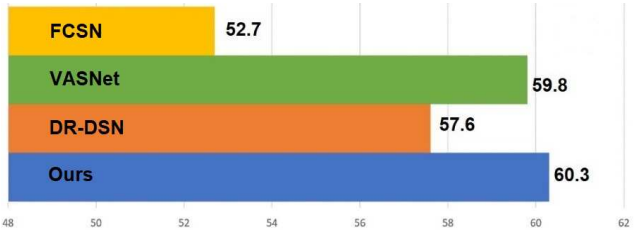


Fig. 13. Our Final F-Score as compared to other video summarization techniques on TVSum Dataset

## VII. CONCLUSION

Thus, this paper presents our framework for video summarization. Contrary to current video summarization models that learn every video frame's importance score only, our video summarization model thinks of video summarization as a temporal-interest detection problem. This helps it to learn temporal coherence between the frames of a video and thus it performs much better than current supervised models at the task of video summarization.

In the end, our model became robust and was able to successfully determine which segments of a video footage are important to be included in the summary video and the model was able to separate them out from the rest of the video clip. We were able to improve accuracy for the problem of video summarization as our model was able to create a good summary video containing all the important parts in a video clip.

In the future, we will attempt to incorporate more interest-based coherence into our unified framework. We could also attempt to increase the temporal coherence between scenes in a video, which could improve the performance of our model.

Fig. 14. Screenshots from summary video created by our model on CCTV surveillance footage

## REFERENCES

[1] T. -J. Fu, S. -H. Tai and H. -T. Chen, "Attentive and Adversarial Learning for Video Summarization," 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2019, pp. 1579-1587, doi: 10.1109/WACV.2019.00173.

[2] Kanafani, Hussain, et al. Unsupervised Video Summarization via Multi-Source Features. arXiv, 26 May 2021. arXiv.org, https://doi.org/10.48550/arXiv.2105.12532.

[3] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris and I. Patras, "AC-SUM-GAN: Connecting Actor-Critic and Generative Adversarial Networks for Unsupervised Video Summarization," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 8, pp. 3278-3292, Aug. 2021, doi: 10.1109/TCSVT.2020.3037883.

[4] W. Zhu, J. Lu, J. Li, and J. Zhou. 2021. DSNet: A Flexible Detect-to-Summarize Network for Video Summarization. Trans. Img. Proc. 30 (2021), 948–962. https://doi.org/10.1109/TIP.2020.3039886

[5] S. Lan, R. Panda, Q. Zhu, A.K. Roy-Chowdhury. (2018). "FFNet: Video Fast-Forwarding via Reinforcement Learning". 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 6771-6780.

[6] K. Zhou, Y. Qiao, and T. Xiang. 2018. "Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward". In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'18/IAAI'18/EAAI'18). AAAI Press, Article 929, 7582–7589.

[7] P. Papalampidi, F. Keller, and M. Lapata, “Movie Summarization via Sparse Graph Construction”, AAAI 2021, 2021

[8] Fred Hohman, Sandeep Soni, Ian Stewart, and John Stasko, 2017, A Viz of Ice and Fire: Exploring Entertainment Video Using Color and Dialogue, VIS4DH Workshop 2017

[9] Arun Balajee Vasudevan, Michael Gygli, Anna Volokitin, and Luc Van Gool, 2017, Query-adaptive Video Summarization via Quality-aware Relevance Estimation, MM '17: Proceedings of the 25th ACM International Conference on Multimedia, pp. 582–590, https://doi.org/10.1145/3123266.31232972017

[10] Mrigank Rochan, Linwei Ye and Yang Wang, 2018, Video Summarization Using Fully Convolutional Sequence Networks, ECCV 2018, 2018

[11] Jiri Fajtl, Hajar Sadeghi Sokeh, Vasileios Argyriou, Dorothy Monekosso, and Paolo Remagnino, 2019, Summarizing Videos with Attention

[12] Shruti Jadon, and Mahmood Jasim, 2020, Unsupervised video summarization framework using keyframe extraction and video skimming, 2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA), 2020, doi: 10.1109/ICCCA49541.2020.9250764

[13] Yair Shemer, Daniel Rotmanand, and Nahum Shimkin, 2019, ILS-SUMM: Iterated local search for unsupervised video summarization, 2020 25th International Conference on Pattern Recognition (ICPR), 2020, doi: 10.1109/ICPR48806.2021.9412068

[14] Jia-Hong Huang, and Marcel Worring, 2019, Query controllable video summarization, ICMR '20: Proceedings of the 2020 International Conference on Multimedia Retrieval, June 2020, pp. 242–250, doi: 10.1145/3372278.3390695

[15] Mohamed Elfeki, Liqiang Wang, and Ali Borji, 2019, Multi stream dynamic video summarization, WACV 2022, pp. 185-195

[16] Muhammad Usama, Junaid Qadir, Aunn Raza, and Hunain Arif, 2019, Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges, IEEE Access (Volume: 7), doi: 10.1109/ACCESS.2019.2916648

[17] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi, 2017, Understanding of a convolutional neural network, 2017 International Conference on Engineering and Technology (ICET), doi: 10.1109/ICEngTechnol.2017.8308186

[18] Alex Sherstinsky, 2018, Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network, 2018

[19] Greg Van Houdt, Carlos Mosquera, Gonzalo Nápoles, 2020, A Review on the Long Short-Term Memory Model, Springer Artificial Intelligence Review, Dec 2020, doi: 10.1007/s10462-020-09838-1

[20] Vladimir Nasteski, 2017, An overview of the supervised machine learning methods, Dec 2017, doi: 10.20544/HORIZONS.B.04.1.17.P05

[21] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath, 2017, A Brief Survey of Deep Reinforcement Learning, 2017

[22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, 2014, Generative Adversarial Nets, NeurIPS, 2014

[23] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio, Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, NeurIPS 2014, 2014

[24] TVSum Dataset, [Online], Available: http://people.csail.mit.edu/yalesong/tvsum/

[25] SumMe Dataset, [Online], Available: https://zenodo.org/record/4884870#.YorKa6hByUl