LSTM Video Summarization utilizing continuous importance scores. LSTM-VS$_c$

Submitted To.

Prof. Walid Gomaa

Abstract

The research investigates video summarization, treating it as a sequential decision-making process. Given a sequence of video frames, the goal is to select a subset of frames that represent the essential content. While Long Short-Term Memory (LSTM) networks are commonly used for this task due to their ability to model temporal dependencies, The Multilayer Perceptron (MLP) follows the LSTM layer in the process. It consists of three fully connected layers, progressively reducing dimensionality.The final output represents an importance score, obtained through a modified Clamping function. The paper addresses the imbalanced class distribution in video summarization by designing a cost-sensitive loss function. The model improves the F-Score of Bi-LSTM by 26.2%, DPP-LSTM by 25%, DR-DSNsup by 21.5%, SUM-GANsup by 21.5%, H-RUN by 18.1%, SASUMsup by 17.3%, M-AVS by 12.1%, DHAVS by 12.5%.

1 Introduction

The propagation of video-generating devices, including smartphones, cameras, surveillance systems, and artificially generated content, has resulted in an unprecedented surge in video content across the internet. This exponential growth has sparked significant interest in the field of Video Summarization within the scientific community. Video Summarization works on providing a shorter comprehensive version of the original video which saves us the time of going through the whole videos.

Video Summarization is divided into two distinct categories, static summarization and dynamic summarization. Static summarization condenses lengthy videos into brief representations by selecting key frames or shots. These summaries provide an efficient way to browse content, optimize storage, and facilitate video retrieval.[2][5] Dynamic video summarization generates a temporally releated version of a given video, considering motion, transitions, and event flow. These summaries facilitate efficient browsing, real-time retrieval, and event detection.[4]

The two techniques used in video summarization research are unsupervised and supervised learning. The supervised approach employ human annotations to previously made datasets and use the annotations as a selection criteria for the shots. Unsupervised approach aims to automatically generate concise video summaries without relying on labeled training data. By analyzing visual content, motion patterns, and temporal coherence.[3][1]

2 Theoretical background

2.1 ResNext Introduction

In previous approaches, CNN networks were employed to catch features out of pictures and were later used in video classification and action recognition, this applied well for a certain number of layers, however as the layers increased the problem of vanishing/exploding gradient decent would become more prominent, meaning the GD would either become 0 or become too large, hence the training and test loss will also increase. The Recurrent Neural Networks solved this problem by employing the principle of not training the model to fit $G(x)$, but instead to train it on:

$$F(x) = G(x) - x \tag{1}$$

where by doing so we can obtain the initial goal by:

$$G(x) = F(x) + x[6] \tag{2}$$

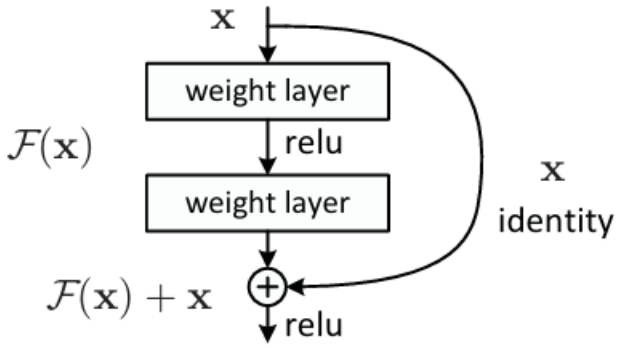

Figure 1: Residual Learning [6]

This analogy shines clearly in the problem of fitting models on up-sampling low-resolution images into high-resolution version, where we have a dataset of low-resolution and high-resolution version of the same images, and by implementing the explained approach, we can subtract the two images and obtain a residual of that image, where that residual is trained through the network over a series of convolutions. This approach solved the GD problem and introduced ResNets, which is implemented using residual blocks and skip connections that connect the activation layers of each block to the initial input of that residual block, by doing so the model is now able to fit the residual mapping instead of the original mapping of input.[7]

2.2 ResNeXt Advantage over ResNet

ResNext model introduces the concept of Cardinality, by which the width of a given layer in the architecture can be divided into parallel transformation paths (convolutional layer groups), this concept is introduced to replace the idea of increasing only the width or depth of an architecture, by doing so, and show experimentally that cardinality can be an essential dimension and can be more effective than the dimensions of width and depth.

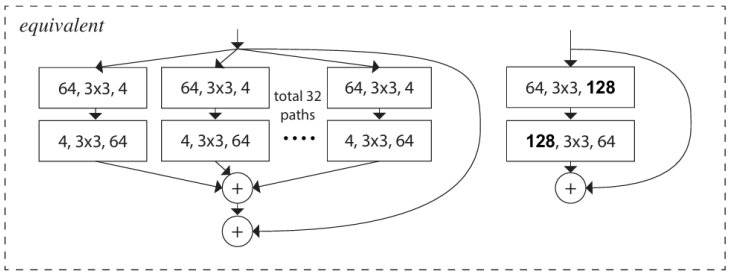

Figure 2: Equivalent Blocks of Aggregating Transformations[7]

	setting	top-1 error (%)
ResNet-50	1 × 64d	23.9
ResNeXt-50	2 × 40d	23.0
ResNeXt-50	4 × 24d	22.6
ResNeXt-50	8 × 14d	22.3
ResNeXt-50	32 × 4d	**22.2**
ResNet-101	1 × 64d	22.0
ResNeXt-101	2 × 40d	21.7
ResNeXt-101	4 × 24d	21.4
ResNeXt-101	8 × 14d	21.3
ResNeXt-101	32 × 4d	**21.2**

Figure 3: Experimentation Results on Different Settings of Cardinality[7]

We observe that the same width, with a higher cardinality model, can achieve better results.

2.3 LSTM Introduction

The standard Recurrent Neural Network (RNN) is challenging to train due to the gradient vanishing issue. To address this challenge, the Long Short-Term Memory (LSTM) network is presented, which extends the standard RNN. The dominant component of the LSTM is the memory cell c_t that can memorize the history of the inputs, where t denotes the tth time step. There are three gates in the cell: the input gate i_t, the forget gate f_t, and the output gate o_t.

The input gate i_t decides whether to consider the current input x_t, the forget gate f_t is applied to control whether to forget the previous memory c_{t-1}, and the output gate o_t decides how much information in the memory cell c_t will be transferred to the hidden state h_t. The cell c_t is modulated by the nonlinear gates f_t and g_t to selectively discard the previous memory cell or record the current input. In our approach, we utilize the LSTM unit employed in [8], which is formulated as follows:

$$i_t = \sigma(W_{ix}x_t + U_{ih}h_{t-1} + b_i) \tag{3}$$
$$f_t = \sigma(W_{fx}x_t + U_{fh}h_{t-1} + b_f) \tag{4}$$
$$o_t = \sigma(W_{ox}x_t + U_{oh}h_{t-1} + b_o) \tag{5}$$
$$g_t = \tanh(W_{gx}x_t + U_{gh}h_{t-1} + b_g) \tag{6}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{7}$$
$$h_t = o_t \odot \tanh(c_t) \tag{8}$$

Where h_t is the hidden state, σ denotes the sigmoid function, tanh denotes the hyperbolic tangent function, $\odot$ is the element-wise product, and W, U, and b are the network weights and biases[15].

3 Model Overview

We approach the video summarization as a sequence-to-sequence learning process involving the following steps:

1. **Feature Extraction:** For a given video, we utilize 3D ResNeXt-101 to extract spatial–temporal features denoted as $(x_1, x_2, \ldots, x_n)$, where n represents the number of video clips.

2. **Importance Scoring:** These extracted features are then fed into a Long Short-Term Memory (LSTM) network. The LSTM output is processed by a Multilayer Perceptron (MLP) to generate frame-level importance scores.

3. **Summary Generation:** Finally, the frame-level importance scores are converted into shot-level scores to produce the video summary.

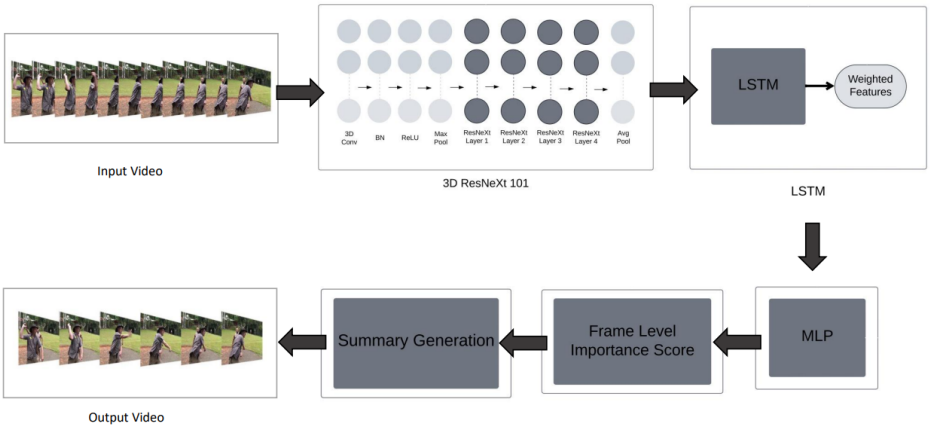

Figure 4: Model Architecture

4 ResNeXt Module

4.1 ResNeXt Implemntation

We use the predefined, non-trained ResNeXt-101 model[16], we begin by handling the video frame input data shapes to properly fit the model and train over the model's blocks consisting of convolutional layers, batch normalization layers, and ReLU layers. We utilize the cardinality feature of the model to separate feature maps into parallel transformational groups, with their number being cardinality. Then we use the output of the last average pooling layer, after some handling to be passed to the LStM layer explained in the next subsection, as the spatio-temporal features of the video to be weighted by LSTM.

4.2 ResNeXt-101 output Handling

Extracted features within the ResNext-101 are of length $2048 \times number\ of\ segments$, to properly pass the extracted features for LSTM for weighting according to long-range temporal dependencies, we opt to reshape the ResNext output to 3D by padding to the first integer divisible by 32, with 32 being the LSTM sequence length.

5 LSTM & MLP Module

In our video summarization model, the LSTM_MLP_Module illustrated in [Figure 5] leverages LSTM's strengths to capture temporal dependencies in video sequences. The structure and its usage in video summarization are described as follows:

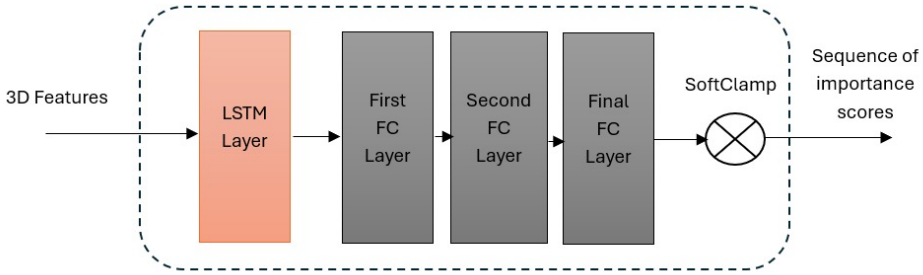

Figure 5: LSTM & MLP Module

The model receives a sequence of video frame features as input, the input sequence is processed by the LSTM layer to capture temporal dependencies. The LSTM outputs a sequence of hidden states, encoding the temporal context of the video. Fully Connected (FC) Layers transform the LSTM outputs into a more compact feature representation. The first FC layer reduces the dimension of the LSTM output to 128. The second FC layer further processes the features while maintaining the same dimension (128). The final FC layer transforms the features into a single scalar value, representing the importance score for each frame, after that, a custom activation function, SoftClamp, is applied to constrain the output values to a specified range, ensuring stability and reliability in the importance scores. The final output is a sequence of importance scores, indicating the significance of each video frame for summarization purposes. After the LSTM layer, the data is passed into a Multilayer perceptron (MLP) which consists of 3 layers. The first fully connected linear layer takes its input as the hidden size generated from the LSTM and it produces an output of size 128. The second fully connected layer takes the output from the previous layer (128 features) and produces another 128-dimensional output. The final fully connected layer reduces the dimensionality to a single output. The importance score is then passed onto a Clamp activation function to generate the output which represents the importance score. A modified Clamping function was used to perform a weighted average between SoftClamp and HardClamp to provide a closer representation to the intended output.

$$\text{Clamp}(x) = \frac{1}{2} \cdot \frac{(-|x - \max| + |x| + \max) + (\max \cdot \text{soft}) \left(\tanh\left(\frac{2x}{\max} - 1\right) + 1\right)}{\text{soft} + 1} \tag{9}$$

where soft is a parameter controlling the degree of softness of the clamping, and max is the maximum value for the output.

6 Loss Function

In the task of video summarization, accurately identifying keyframes is crucial. To achieve this, our loss function integrates multiple components to ensure robust training and effective summarization:

$$\text{loss} = (\text{Error\_term}) + \lambda \cdot (\text{Deviation\_term}) + \mu \cdot (\text{Correlation\_term}) \tag{10}$$

Where λ is the misclassification cost, μ is the correlation factor, and the mentioned terms are as follows:

$$\text{Error\_term} = -\sum_{i=1}^{n} \log\left(1 - \frac{|(\hat{y}_i) - y_i|}{5}\right) \tag{11}$$

This component measures the binary cross-entropy loss, crucial for minimizing prediction errors in distinguishing keyframes from non-keyframes. Where, $\hat{y}_i$ represents the predicted importance score for the ith frame, and y_i is the human label.

$$\text{Deviation\_term} = (\hat{\sigma} - \sigma)^2 \tag{12}$$

This term penalizes large deviations from the mean, helping maintain prediction stability. Where $\hat{\sigma}$ is the predicted standard deviation of the importance scores and σ is the true standard deviation.

$$\text{Correlation\_term} = \frac{1 - r}{1 + r} \tag{13}$$

This component encourages a high correlation between predicted scores and actual keyframe importance, aligning model predictions with ground truth. Where, r is the Pearson correlation coefficient between the predicted scores and the actual labels. By combining these terms, our loss function effectively balances error minimization, prediction stability, and alignment with actual keyframe importance, leading to more accurate video summarization.

7 Summary Generation

7.1 Kernel Temporal Segmentation

Using the retrieved features as an input from the ResNeXt model, this video summarization model applies Kernel Temporal Segmentation (KTS) [9] to divide the video into discrete temporal segments, each of which represents a unique activity or event. KTS uses kernel-based techniques, which are effective at handling non-linear data, to analyze changes in the distribution of video attributes over time. This method finds the borders,

or change points, when the video material significantly changes. The segments are defined by these change points, which provide a systematic depiction of the temporal flow of the movie. The average importance of the temporal segments had to be determined to later use the fractional knapsack method to generate the most important segments that made up the summary video. KTS plays a crucial role in this model since it successfully catches intricate temporal patterns, guaranteeing that the summary that results emphasize the most important and varied events in the video. This segmentation method improves the model's capacity to provide succinct yet thorough summaries, promoting improved comprehension and effective video content consumption.

7.2 Video Generation

Following a preset constraint (up to 40% of the original video length), the most important segments for creating the video summary were identified through the integration of the fractional knapsack algorithm with the outputs produced by the Kernel Temporal Segmentation (KTS) and the computed average importance of each segment. The final visual output was created by smoothly combining these chosen pieces with the MoviePy library[10].

8 Experiments and Results

8.1 Dataset

The dataset utilized in this study is TvSum, comprising 50 videos spanning across 10 distinct categories. Each video was sourced from YouTube and annotated by a panel of 20 users. The videos cover a diverse array of topics and genres, with durations typically ranging from 2 to 10 minutes. This broad spectrum of content ensures a comprehensive representation of real-world video material, facilitating robust experimentation and analysis. [11] For experimental purposes, the dataset was partitioned into an 80% training set and a 20% testing set. This partitioning strategy ensures a balanced distribution of data for model training and evaluation, enabling reliable performance assessment.

8.2 Evaluation Metrics

The efficacy of the model can be comprehensively assessed by comparing it to existing video summarization techniques using a variety of metrics. The following metrics were employed in this study:

F-score: This metric evaluates the model's performance by considering the lengths of both the ground truth summary and the predicted summary, as well as the intersection between the two. It provides a holistic measure of the summarization quality, considering

both precision and recall.[11]

$$\text{Recall} = \frac{\text{Length}_{\text{Intersection}}}{\text{Length}_{\text{Ground Truth}}} \tag{14}$$

$$\text{Precision} = \frac{\text{Length}_{\text{Intersection}}}{\text{Length}_{\text{Predicted}}} \tag{15}$$

$$F\text{-}Score = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \tag{16}$$

Kendall Tau: Kendall's Tau is a rank correlation coefficient that assesses the similarity of rankings between two variables. In the context of video summarization, Kendall Tau can quantify the concordance between the rankings of video segments in the ground truth summary and those in the predicted summary. [13]

$$\tau = \frac{2\left(n_{\text{concordant}} + n_{\text{discordant}}\right)}{n(n-1)} \tag{17}$$

Spearman's Rho: Spearman's Rho is another rank correlation coefficient that evaluates the monotonic relationship between two variables. Similar to Kendall Tau, Spearman's Rho can be employed to measure the agreement between the rankings of video segments in the ground truth summary and those in the predicted summary. [14]

$$\rho = \frac{6\sum_i d_i^2}{n(n^2-1)} \tag{18}$$

d_i: the difference in the ranking of the i^{th} element between the prediction and the ground truth spaces

By utilizing these evaluation metrics, a comprehensive and nuanced assessment of the model's performance can be obtained, facilitating meaningful comparisons with other video summarization techniques.

Model	F-Score
Bi-LSTM [19]	54.2
DPP-LSTM [19]	54.7
DR-DSN$_{sup}$ [18]	58.1
SUM-GAN$_{sup}$ [17]	56.3
H-RUN [21]	57.9
SASUM$_{sup}$ [22]	58.2
M-AVS [20]	61.0
DHAVS [12]	60.8
LSTM-VS$_c$ (ours)	**68.4**

Figure 6: Performance results (F-score) of LSTM-VSc compared to other video summarization approaches

Model	Kendall Tau	Spearman's Rho
DPP-LSTM [19]	0.020	0.026
DR-DSN [18]	0.042	0.055
DHAVS [12]	0.082	0.086
LSTM-VS$_c$ (ours)	**0.082**	**0.111**

Figure 7: Performance results (Correlation coefficients) of LSTM-VSc compared to other video summarization approaches

8.3 Implementation

In our implementation, we adopt a multi-step process to generate video summaries. Firstly, each input video is partitioned into non-overlapping 15-frame clips. Subsequently, we utilize the output of the last pooling layer of a 3D ResNext-101 model pre-trained on the Kinetics dataset as the input for an LSTM (Long Short-Term Memory) module, followed by a Multilayer Perceptron (MLP) to predict the importance of each clip.

We set the parameters λ and μ to 1.5 in the loss function to effectively train the model. For optimization, we employ a learning rate of 10^{-4} during the training process. Training proceeds until overfitting is detected, ensuring the model captures relevant patterns in the data.

To facilitate coherence in the generated summaries, we segment the video into coherent clips using Kernel Temporal Segmentation (KTS) before applying the summarization algorithm. The summarization itself is achieved by solving a knapsack problem, where the goal is to select a subset of clips that maximizes the overall importance while respecting constraints such as summary length.

By employing this methodology, we aim to produce concise and coherent video summaries that capture the salient content of the input videos effectively.

8.4 Quantitative results

Figures [Figure 6, Figure 7] show a comparable, if not better, performance to the other summarization models. The model improves the F-Score of Bi-LSTM by 26.2%, DPP-LSTM by 25%, DR-DSNsup by 21.5%, SUM-GANsup by 21.5%, H-RUN by 18.1%, SASUMsup by 17.3%, M-AVS by 12.1%, and DHAVS by 12.5%. There is also significant improvement in correlation coefficient, especially Spearman's ρ.

In addition, experiments had been conducted to study the effect of λ and μ values on the results quality. We tried the values of 0.5, 1.0, 1.5, 2.0 for both parameters as shown in Figures [Figure 8, Figure 9]

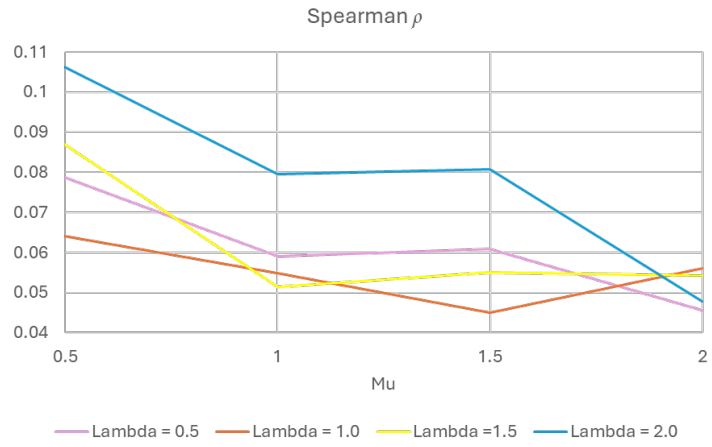

Figure 8: Effects of λ and μ values on correlation

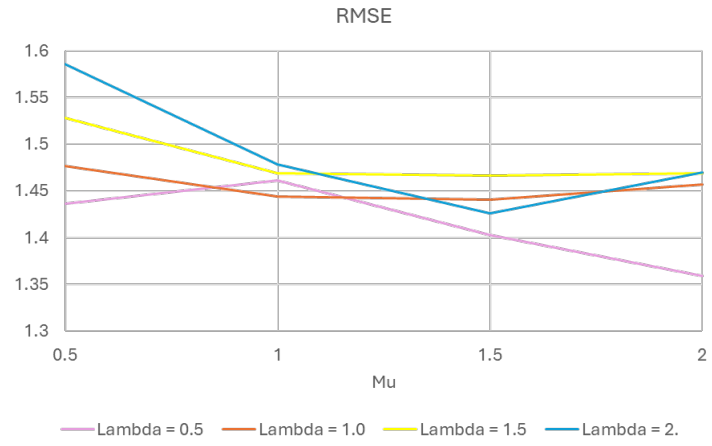

Figure 9: Effects of λ and μ values on mean error.

The collected data suggests that the high values of λ is generally preferable to both increase correlation and decrease mean error, while the higher values of μ leads to a decreased correlation coefficient and error.

To find the optimal number of iterations we observed the loss across the testing data sets.

Figure 10: Observed loss across iterations

Overfitting is observed at about 35 iterations. As illustrated in [Figure 10].

For video summary generation, we observed the effect of changing the summary length on the F-Score at 50 clips and 100 clips segmentations. The results for 5%, 10%, 15%, 20%, 25% are shown in [Figure 11]. The F-score can be observed to generally increase with the clip count and summary length.

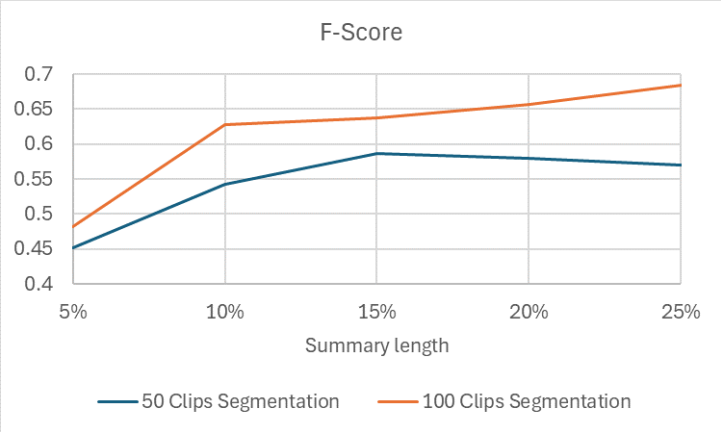

Figure 11: Variation of F-Score with Summary length.

8.5 Qualitative results

In figure [Figure 12] we show the summery generated by the ground truth against the data from our model which scored 66 F-Score

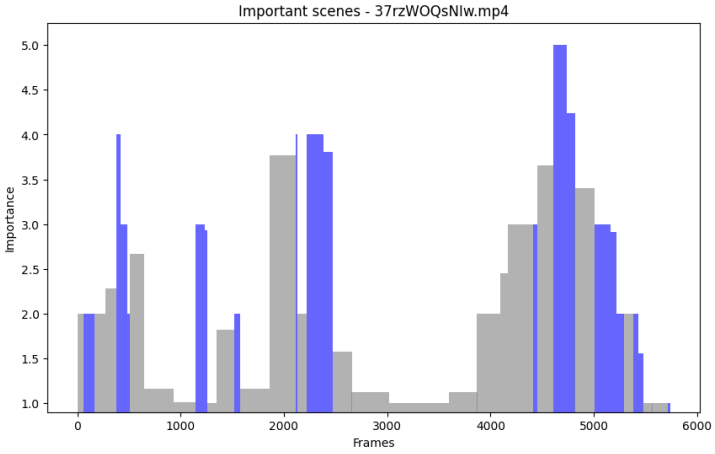

Figure 12: The ground truth summery

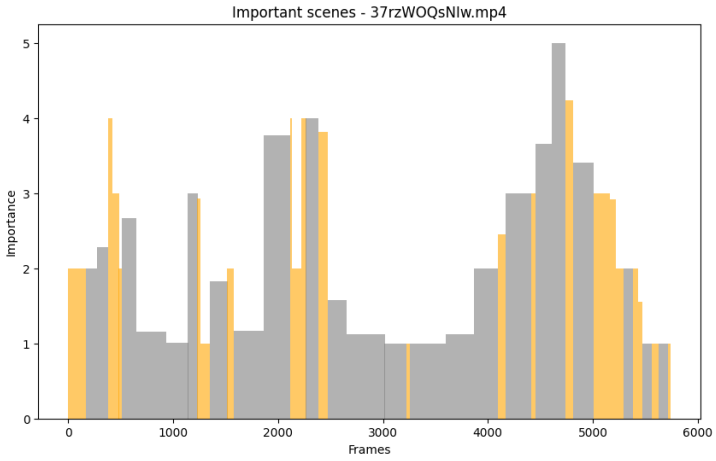

Figure 13: The predicted summery

And it can be seen that the results of proposed model closely resembles the ground truth.

9 Conclusion

In this research, we present a novel approach to video summarization, treating it as a sequential decision-making process. By leveraging a combination of Long Short-Term Memory (LSTM) networks and Multilayer Perceptrons (MLP), our method effectively captures both temporal dependencies and frame-level importance scores. We address the issue of imbalanced class distribution with a cost-sensitive loss function, leading to significant improvements in F-score and correlation coefficients compared to several state-of-the-art methods.

Our experimental results demonstrate that our model improves the F-score over Bi-LSTM by 26.2%, DPP-LSTM by 25%, DR-DSNsup by 21.5%, SUM-GANsup by 21.5%, H-RUN by 18.1%, SASUMsup by 17.3%, M-AVS by 12.1%, and DHAVS by 12.5%. We achieve this by employing a 3D ResNeXt-101 model for spatial-temporal feature extraction, followed by LSTM and MLP layers to generate importance scores, which are refined using a modified Clamping function. The final video summary is generated by integrating Kernel Temporal Segmentation (KTS) and a fractional knapsack algorithm to select the most significant segments.

Our future work will focus on extending our model to handle longer videos and further improving the summarization process by incorporating sparse attention and position embedding. Additionally, we aim to enhance our method's ability to learn from unpaired video summaries available on the internet, addressing the limitation of labeled video summaries. This research provides a robust foundation for efficient and accurate video summarization, facilitating improved video content consumption and retrieval.

10 References

[1] Evlampios Apostolidis et al. "Unsupervised Video Summarization via Attention-Driven Adversarial Learning". In: *Conference on Multimedia Modeling*. 2019. url: https://api.semanticscholar.org/CorpusID:209655133.

[2] Wu Liu et al. "Multi-task deep visual-semantic embedding for video thumbnail selection". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 3707–3715. url: https://api.semanticscholar.org/CorpusID:14495525.

[3] Behrooz Mahasseni, Michael Lam, and Sinisa 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.

[4] Antonio Tejero-de-Pablos et al. "Summarization of User-Generated Sports Video by Using Deep Action Recognition Features". In: *IEEE Transactions on Multimedia* PP (Sept. 2017). doi: 10.1109/TMM.2018.2794265.

[5] Jiaxin Wu et al. "A novel clustering method for static video summarization". en. In: *Multimedia Tools and Applications* 76.7 (May 19, 2016), pp. 9625–9641. issn: 1380-7501. doi: 10.1007/s11042-016-3569-x. url: http://dx.doi.org/10.1007/s11042-016-3569-x.

[6] He, K., Zhang, X., Ren, S., & Sun, J. (2015, December 10). Deep residual learning for image recognition. arXiv.org. https://arxiv.org/abs/1512.03385

[7] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2016, November 16). Aggregated residual transformations for deep neural networks. arXiv.org. https://arxiv.org/abs/1611.05431

[8] Zaremba, W., & Sutskever, I. (2014). Learning to execute. arXiv preprint arXiv:1410.4615. https://arxiv.org/abs/1410.4615

[9] TatsuyaShirakawa. (n.d.). GitHub - TatsuyaShirakawa/KTS: Kernel Temporal Segmentation. GitHub. https://github.com/TatsuyaShirakawa/KTS

[10] moviepy. (2020, May 7). PyPI. https://pypi.org/project/moviepy/

[11] Zhang, K., Chao, WL., Sha, F., Grauman, K. (2016). Video Summarization with Long Short-Term Memory. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science(), vol 9911. Springer, Cham. https://doi.org/10.1007/978-3-319-46478-7_47

[**12**] Deep hierarchical LSTM networks with attention for video summarization. (n.d.). Retrieved from ScienceDirect website: https://www.sciencedirect.com/science/article/pii/S2090447919303778

[**13**] The treatment of ties in ranking problems. (n.d.). Retrieved from JSTOR website: https://www.jstor.org/stable/2332303

[**14**] CRC Standard Probability and Statistics Tables and Formulae, Student Edition. (n.d.). Retrieved from ResearchGate website: https://www.researchgate.net/publication/264605249_CRC_Standard_Probability_and_Statistics_Tables_and_Formulae_Student_Edition

[**15**] LSTM — PyTorch 2.3 documentation. (n.d.). Retrieved from https://pytorch.org/docs/stable/generated/torch.nn.LST M.html

[**16**] Kensho Hara. (n.d.). GitHub - TatsuyaShirakawa/KTS: Kernel Temporal Segmentation. GitHub. https://github.com/kenshohara/3D-ResNets-PyTorch

[**17**] Mahasseni B, Lam M, Todorovic S. Unsupervised video summarization with adversarial lstm networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2017. p. 202–11.

[**18**] Zhou K, Qiao Y, Xiang T. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In: Thirty-second AAAI conference on artificial intelligence; 2018. p. 7582–9..

[**19**] Zhang K, Chao W-L, Sha F, Grauman K. Video summarization with long short-term memory. In: European conference on computer vision. Springer; 2016, p. 766–82.

[**20**]] Ji Z, Xiong K, Pang Y, Li X. Video summarization with attention-based encoder–decoder networks. IEEE Trans Circuits Syst Video Technol 2019;30(6):1709–17.

[**21**] Zhao B, Li X, Lu X. Hierarchical recurrent neural network for video summarization. In: Proceedings of the 25th ACM international conference on multimedia; 2017. p. 863–71.

[**22**] Wei H, Ni B, Yan Y, Yu H, Yang X, Yao C. Video summarization via semantic attended networks. In: Thirty-second AAAI conference on artificial intelligence; 2018. p. 216–223.