Video Summarization Using Global and Segmented Local Attention

Submitted By-

Zadid Bin Azad
Student ID: 170042046

Wasif Azmaeen
Student ID: 170042055

Azad Al Fayad
Student ID: 170042075

Supervised by-

Dr. Md. Hasanul Kabir
Professor
Department of Computer Science and Engineering

Submitted To-

Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of B.Sc.

May, 2022

Department of Computer Science and Engineering(CSE)
Islamic University of Technology (IUT)
Gazipur, Dhaka-1704, Bangladesh

Declaration of Authorship

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by Zadid Bin Azad, Wasif Azmaeen and Azad al Fayad under the supervision of Dr. Md. Hasanul Kabir, Professor, Department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Dhaka, Bangladesh. It is also declared that neither of this thesis nor any part of this thesis has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others have been acknowledged in the text and a list of reference is given.

Zadid Bin Azad
Student ID: 170042046
B.Sc in Software Engineering
Department of Computer Science and Engineering(CSE),
Islamic University of Technology (IUT)

Wasif Azmaeen
Student ID: 170042055
B.Sc in Software Engineering
Department of Computer Science and Engineering(CSE),
Islamic University of Technology (IUT)

Azad al Fayad
Student ID: 170042075,
B.Sc in Software Engineering
Department of Computer Science and Engineering(CSE),
Islamic University of Technology (IUT)

Video Summarization Using Global and Segmented Local Attention

Approved and supervised by :

Dr. Md. Hasanul Kabir
Professor
Department of Computer Science and Engineering (CSE)
Islamic University of Technology (IUT), Gazipur.

ACKNOWLEDGMENTS

We would like to express our deepest gratitude to our supervisor Dr. Md. Hasanul Kabir for his constant guidance and support on this project showing us the path of conducting successful research and above all for always being there as our mentor. His enthusiasm, knowledge, and exacting attention to detail have been an inspiration and kept our work on track from the ideation of our research until the end. His suggestions drove us towards better ways of thinking, his reviews enriched us in solving problems, and his support gave us strength at the time of our disappointment. We shall forever cherish the memories of working with him. We deeply thank our friends and families for always believing in us even at the moment when we were losing our confidence.

ABSTRACT

Over the recent past years the amount of video data has grown exponentially. Video summarization has emerged as a process that can facilitate in areas like efficient storage, indexing and quick understanding of a large video. We take video summarization as a task of finding out visual cues from frames which lead to a sensible human understandable temporal order. As attention models are currently performing best for maintaining long range temporal orders, our research tends to find a better way to implement attention mechanism for the purpose of video summarization.We approach to solve the problem of video summarization with supervised method. For that, we propose a novel architecture using Global and Segmented Local Multi Head Attention mechanism and this has greatly helped us to maintain the temporal and contextual consistency in the summarized video. From our architecture, we get the insight that segment size should be determined based on the change points of videos inside a dataset and the number of heads in multi-head attention should be determined based on segment length. Our proposed methodology shows us superiority in results with respect to the existing state of the art methods and has achieved remarkable improvements from 2% to 3% on two benchmark data sets.

TABLE OF CONTENTS

List of Tables

List of Figures

CHAPTER 1

INTRODUCTION

1.1 Motivation

The ease of taking video footage and the expansion of video data uploading platform has taken us by a surge over the past few years. From mobile devices to CCTV cameras all multi-media devices capture videos at such an amount that we can barely keep up with consuming all of them. In the year 2020, sum of all videos uploaded to the internet will exceed the time frame of 500 million years in total according to Cisco Visual Networking Index.

Therefore efficiently storing, browsing, indexing and consuming this large amount of data calls for the technique of video summarization. The idea of video summarization in short is that this process can provide maximum amount of contextual information from the video while minimizing the time span with which a video can present the contextual information to the user.

1.2 Application Fields

Video summarization spans to different genre. Fields like sports, movies, surveillance feels the need of video summarization because of the amount of video information too large for a general user to consume and understand in a small amount of time. Some specific application can be depicted as:

- Efficient storage, fast retrieval, quick indexing of large videos without losing contextual information.
- Providing the user quick understanding of videos specially in the field of news and educational videos.
- Creating highlights of sports is of high interest now a days. Videos summarization can do this job efficiently.
- Creating movie summaries is another field where video summarization can play promising role.
- Finding events in a long surveillance video can also be an application of video summarization.

1.3 Overview

A video stores information in a spatio-temporal hierarchy. Briefly it means a video has a hierarchical structure and it comprises of frames, shots and scenes. Frames are the primary element of a video lying in the lowest tier in the hierarchy. Consecutive frame of uninterrupted time interval makes up a shot putting it on the 2^{nd} layer of the hierarchy.Lastly multiple shots make a a scene.To convey a meaningful story we have to combine all the scenes in a perfect timeline which ultimately makes up a video.

Let V a video with total frame count of n. If the summarized video is denoted as S with m frames which don't have to be consecutive but have to maintain temporal order then,

$$V = \{f_1, f_2, f_3, ..., f_n\} \tag{1.1}$$

$$S = \{f_{t_1}, f_{t_2}, f_{t_3}, ..., f_{t_m} | 1 \leq t_i \leq m \ and \ m \ll n\} \tag{1.2}$$

Figure 1.1: Example of a summarized video[5]

As video is a spatio-temporal data we have to maintain that in the summarized video too. So the summarized frames that we tend to achieve should be the most informative one. Previous works on automated video summarization works have been done on different methods and techniques. Most of them includes using some variants of RNN (LSTM, Bi-LSTM,GRU etc.) to extract the temporal feature and maintain the consistency[1]. But as these methods have grown they show some major weaknesses. Video length in today's time tend to be very long not to mention the 24/7 monitoring surveillance cameras and their video data length. In such cases, the RNN based methods suffer a lot to keep up as they have to traverse a long path in the network to calculate the dependencies with time. There have been studies with CNN based architecture to do the same but as CNNs are not made to conserve time dependent data they don't perform that well compared to the later studies[2].

To tackle the drawbacks these aforementioned methodologies proposed a different kind of approach was taken by the researchers. These works have been implemented with attention mechanism and has been improving since [5–8]. The attention mechanism has been evolving with time as there has been attention, self-attention, multi-head self attention etc. For supervised video summarization techniques VASNet[6] provides some key observation points. It shows that the frames which are temporally distant are less likely to be dependent on each other. As much as this observation helps in using attention, there should still be use of attention over a certain

shot or segment of the video and as well as the whole video itself. PGL-SUM[8] leverages the same idea of applying attention globally as well locally. But they provide no reasonable argument on how to chose the segment on which the local attention should be applied. Another architecture of supervised video summarization, DSNet[5] uses attention mechanism but doesn't leverage the proper multi-head attention mechanism technique which results as it is proven by the transformer[9]. Moreover, they use some fixed size anchors to generate proposals of segments in the summarized video. But the problem with fixed size anchors is that it does not fit videos of all length. So in our work we tackle these shortcomings using both local and global multi-head attention with relative positional encoding of the input video.

1.4 Research Challenges

Problems that deal with contextual information faces some general challenges. While a video holds contextual information it also holds spatial and temporal data. So while making a network which can deal with video summarization, we need to face both these challenges alongside some others.

1.4.1 Maintaining Temporal Consistency

A video is flow of frames with respect to time. Therefore if temporal consistency is broken there is no point of making video summarization. Then again in a certain shot which can be long up-to multiple seconds we need to pick one or more frames which maintains temporal order also picks up the best information.

1.4.2 Expressing Contextual Information

Each video has a primary subject that it tries to express to its consumer. A video summarizer has to ensure that it picks up all the necessary information the video is conveying to the user. If any contextual information is

left behind then the next frames that will be picked for the summary can not hope to express its information to the user properly.

1.4.3 Diversity

Videos in different domain can require different summarization technique. For example a summarizer for generating movie summary can not hope to do well in case of sports video. Then again some videos has no specific domain rather falls in the category of random user generated video which require another kind of approach to summarize properly.

1.4.4 Feature Set

A video can provide information in audio and video and even in textual subtitle form and all these can be used to summarize a video maintaining the proper quality. But these much feature is complex to process at a single time and more complicated to sync with one-another.

1.4.5 Inadequate Videos in Dataset

Till now mostly popular and benchmark datasets are TVSum[4] and SumMe[3]. But even though these datasets have been used in many cases the number of videos in these datasets are 50 and 10 respectively. We know that deep learning based methods tend to use a lot of data to train. So in such scenario we can't use pre-trained models as the models are not trained with very rich datasets.

1.5 Problem Statement

Considering all the mentioned challenges, we formulate our problem statement as -

Summarizing random user generated videos while maintaining temporal and contextual constraints with the help of global and segment local multi-head attention.

1.6 Objective of our work

One of the main objectives of our work is calculating the attention weights of each frame correctly as it will lead to pick important frames for the summarization. In order to achieve this, we are using two parallel attentions; Global multi-head attention that works on all video frames at once and Segmented Local attention that works on the segments of the source video.

Apart from that, we want to maintain the temporal consistency and contextual correctness of the summarized video with respect to the ground truth video. The use of positional encoding will help us to maintain temporal consistency in the correct order. It will also enable to reduce the computation cost of picking important frames in right order.

Our final objective is to establish that, in video summarization, fixed amounts of segments should not be used. [5] used fixed amount of segments after studying only one dataset. But different datasets have different types of videos and the change point of each video differs from each other. Thus flexible amount of segments should be chosen based of the change points of a video.

1.7 Organization of dissertation

For the rest of the dissertation, we have discussed about previous related works on video summarization and their existing limitations in chapter 2. We have briefly discussed about our proposed methodology in chapter 3. Our experimental setup and result analysis is depicted in chapter 4. In the final chapter, we have talked about the future scopes of this work.

CHAPTER 2

LITERATURE REVIEW

2.1 LSTM based Video Summarization

The technique Long Short-Term Memory (LSTM) was first used for video summarization in [1]. The goal of this study was to propose a new supervised learning technique for summarization of videos by automatically picking key frames or key sub-shots. They used LSTM to model the temporal dependency among video frames. They also improved the model's strength with a strategy called Determinantal Point Process in order to increase the diversity among the selected frames. They named the first model as **vsLSTM** and the second model as **dppLSTM**.

The model is composed of two LSTM layers where one layer models the video frames in forward direction and the other layer models the video frames in backward direction. In each layer, the LSTM blocks are composed of single LSTM units. If we think the video as a a sequence of frames $x = \{x_1, x_2, ..., x_t, ..., x_T\}$, then x_t is the visual features extracted at the t_{th} frame which is used as the input of the LSTM layers. The LSTM layers' hidden states and the visual features x_t are then combined with a **Multi Layer Perceptron** and the final output y_t is produced.

$$y_t = \{h_{forward_t}, h_{backward_t}, x_t\} \tag{2.1}$$

We can see the architecture of vsLSTM in figure 2.1

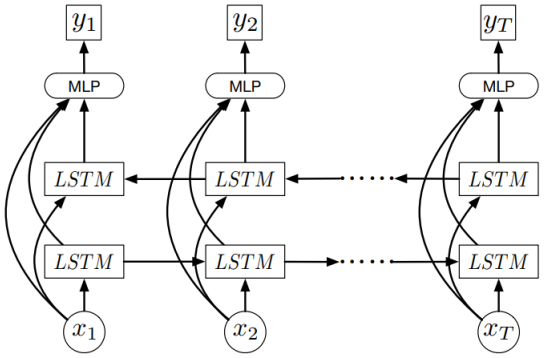

Figure 2.1: vsLSTM model pipeline [1]

Now as we can see in figure 2.2, in order to strengthen vsLSTM, they then introduced another model with **Determinantal Point Process** (DPP). DPP helped the model to find similarities between frames and eliminate redundant frames based on that. This led to produce a diverse set of frames. In this model, they used the visual features x_t, extracted at frame level and used that as the input of LSTM layers. Then they combined the output of these LSTM layers and visual features with two **Multi Layer Perceptron**. They claimed their empirical study showed that using two **Multi Layer Perceptron** helped to enhance the result. The output of the two MLP's were taken and the inner product of them were passed to DPP. DPP then determined the similarity among those frames and deleted the redundant frames.

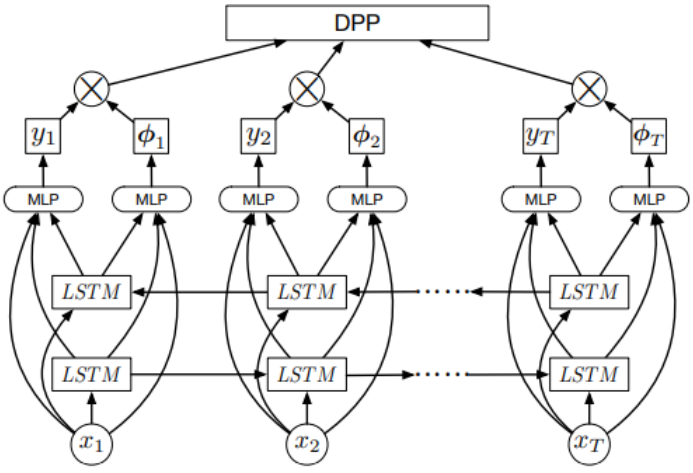

Figure 2.2: dppLSTM model pipeline [1]

The major contributions of this study is that they used Long Short-Term Memory to create a new supervised model for video summarization and then enhanced this technique with the help of Determinantal Point Process to select diverse set of frames.

They produced this result over the TVSum[4] and SumMe[3] datasets.

Dataset	Model	Results
SumMe	vsLSTM	37.6
	dppLSTM	38.6
TVSum	vsLSTM	54.2
	dppLSTM	54.7

Table 2.1: Video summarization results (F-Score) with vsLSTM and dppLSTM model [1]

2.2 Fully Convolutional Network based Video Summarization

Semantic Segmentation is a process of assigning a label to every pixel in the image. Fully Convolutional Network are popular in semantic segmentation. [2] proposed to adapt this technique for video summarization. Traditionally, video summarization and semantic segmentation are considered to be completely different problems. But [2] had the insight that this two problems can have similar techniques as a solution.

They presented that FCN can be used for video summarization by just changing the dimension of input (2D for semantic segmentation vs. 1D for video summarization) and the number of channels (3 or RGB for semantic segmentation vs. K - assuming each frame is represented as a K-dimensional vector).

They named their model as **SUM-FCN**. In **SUM-FCN**, the input is of $1 \times T \times D$ where, **T** is the total number of frames in a video and **D** is the dimension of the feature vector of a single frame.

The output of SUM-FCN is of dimension $1 \times T \times C$. As they classified all the frames as key-frames or non key-frames, here $C = 2$.

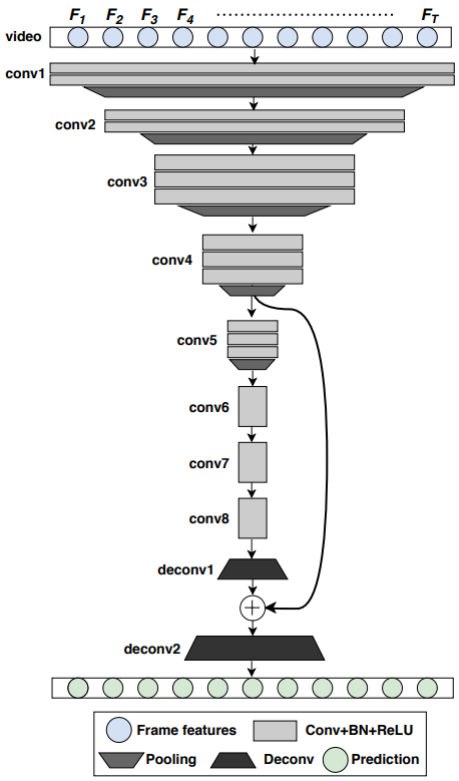

Figure 2.3: SUM-FCN model pipeline [2]

Figure 2.3 depicts the architecture of **SUM-FCN**. Here, the first five layers consist of multiple convolution layers, each one followed by a batch normalization and a ReLU activation. There's also a temporal max-pooling next to each convolution layer. Next two layers consist of one convolution layer followed by one ReLU and one dropout layer that helps to prevent over-fitting.

The next layer has a 1×1 convolution, one batch normalization and one deconvolution operation. It produces the desired output channel. In the next layer, they applied a 1×1 convolution with the output of pool4. It also has one batch normalization and then it was merged with deconv1 feature map. This is done to produce richer visual features. In the last layer, they applied a temporal deconvolution again and obtained the final prediction of length T.

For feature extraction, first they uniformly down sampled the videos to 2 fps. Then they took the output of the pool5 layer in the pretrained GoogleNet as the feature descriptor for each video frame. The dimension of this feature descriptor is 1024. They mainly used GoogleNet features because it is used in previous work and will allow fair comparison in the experiments.

The major contribution of this approach is that they presented a way to adapt popular semantic segmentation networks for video summarization. This indicates we can explore the usage of better and new semantic segmentation techniques on this field. They also showed the limitation of LSTM based approaches as it can't fully utilize the GPU with GPU parallelization, which SUM-FCN can do.

This results were also produced over the TVSum[4] and SumMe[3] datasets.

Dataset	**Result**
SumMe	47.5
TVSum	56.8

Table 2.2: Video summarization results (F-Score) with SUM-FCN [2]

2.3 Bi-LSTM and Attention based Video Summarization

This study presents attention based encoder-decoder network for supervised video summarization. Instead of aiming for basic encoder-decoder networks this paper tries to overcome the drawbacks of basic encoder-decoder network and incorporates Bi-LSTM in the encoder and attention mechanism in the decoded. According to the way attention is used in the decoder they have proposed two models named **A-AVS** and **M-AVS**. Lastly they apply keyshot selection which involves applying 0-1 knapsack prob-

lem solving technique over the importance score of the frames to take at most 15% of the frames in the original video.

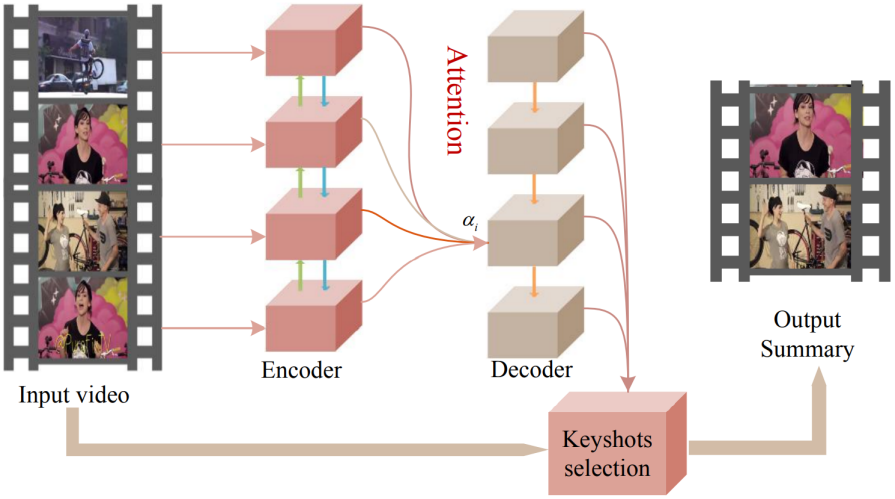

Figure 2.4: Overview of AVS architecture [7]

2.3.1 Encoder with Bi-LSTM

According to the study LSTM can hold the context of the video frames but the limitation is that it will only hold the context of the past but not the future. Therefore this study proposes Bi-LSTM to hold the sequential information from the past and the future for the current frame. The forward pass looks at the video from present towards the future but the series of backward LSTMs reads the video in reverse order. Then the features obtained from both pass are concatenated giving the final feature vector outputted from the encoder V. Each element of the vector V_t can focus on the frames around a certain frame x_t.

2.3.2 Decoder with Attention

The decoder works with the feature vector V produced by the encoder. Decoder calculates an attention score at each time step t.It creates a relevance score of each frame e_t^i based on the current encoder output V_i and its last hidden state s_{t-1}. The normalized relevance score is the attention weight α_t^i.

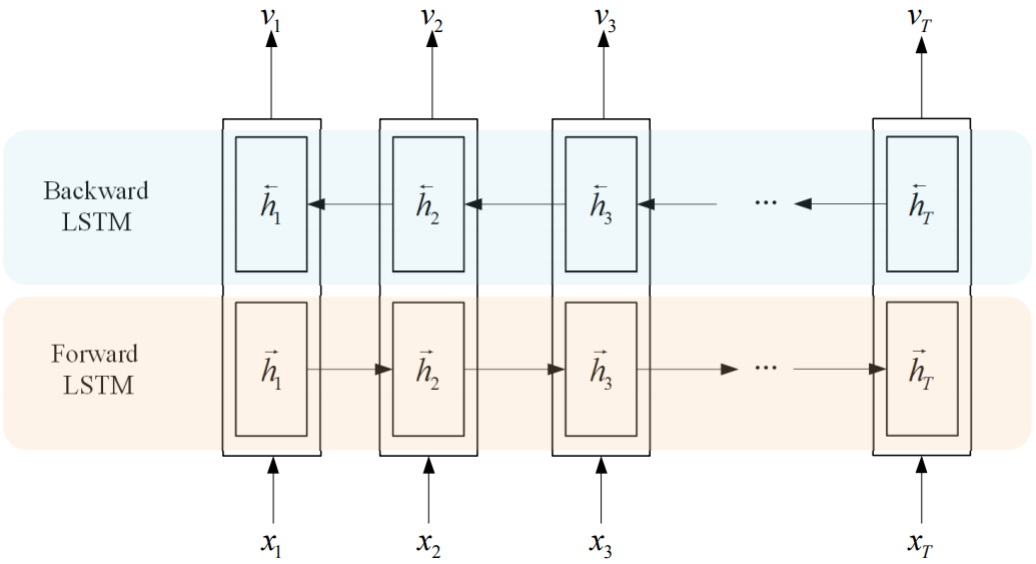

Figure 2.5: Bi-LSTM in AVS [7]

Based on how the relevance score is calculated two models are proposed naming **A-AVS** and **M-AVS**.

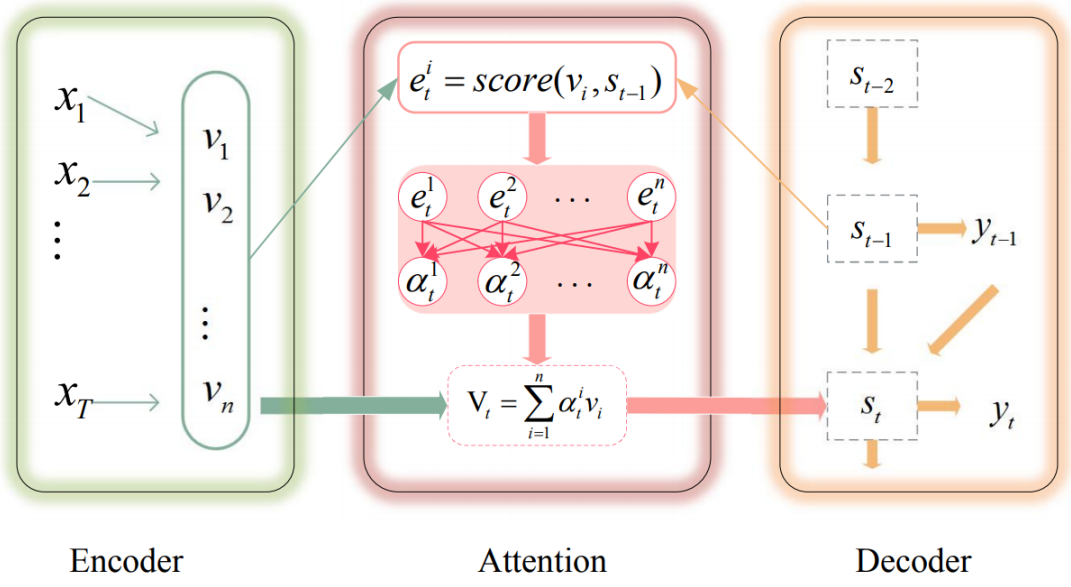

Figure 2.6: Working mechanism of attention decoder [7]

Kernel Temporal Segmentation is used to segment frames into shots. After that sum of scores of all the frames inside a shot is calculated. Then for each shot we can solve the 0-1 knapsack problem maximizing the total shot level score while maintaining the length of the summary be 15% of the original video.

The achievements of this study can be depicted as follows-

i. Use of Bi-LSTM for both past and future context of a frame.

ii. Use of attention mechanism along with attention to replicate how humans pick up contextual information from a video.

This study produced results over the TVSum[4] and SumMe[3] datasets.

Dataset	Model	Results
SumMe	A-AVS	43.9
	M-AVS	44.4
TVSum	A-AVS	59.4
	M-AVS	61.0

Table 2.3: Summarization results (F-Score) with AVS model

2.4 Attention based Video Summarization : DSNet

This study proposes a new model which is called **DSNet**. This model has two counterparts one is **anchor-based** and another one is **anchor-free**. The anchor-based method generates temporal interest proposals by itself and then predicts the representative contents of the video. On the other hand the anchor-free model doesn't care about the temporal proposals rather predicts the importance score and segment location for each frame.

2.4.1 Anchor Based Approach

In an overview an anchor based approach proposes interest proposals at each temporal location with multi-scale duration. This can handle the variation of length of different interest. Then location regression and importance score prediction is performed.

For the case of feature extraction this paper considers both spatial and long-range temporal features. First by applying GoogLeNet[10] without the last 3 layers to extract the visual feature vectors.Then to capture long

range temporal features self-attention[9] module is employed which outputs another feature vector.To get the final feature vector representation, both spatial and temporal feature vectors are concatenated.

The temporal interest proposal generation is inspired by the region proposal networks which try to generate interest segment of different sizes in the video. Here this study proposes a new theorem to propose the multi-scale proposals. After getting the interest proposals we label each interest proposals as positive or negative. That means we assign binary class labels to interest proposals.

The study shows that each proposal is assigned positive or negative based on temporal Intersection Over Union. ***tIOU***. As there will be way more negative classes than positive so the class imbalance problem has to be solved. In order to do that the proposals are sampled in the ratio of 1:3. When specifically explained, this study considers a proposal to be positive when the tIOU of a proposal is greater than 0.6 with any ground truth segment. But for negative label they assign interest proposals with tIOU lower than 0.3. The proposals whose tIOU are between 0.3 and 0.6 aren't assigned a negative label because they reduce the summarizer's performance. This method according to the authors can minimize the amount of irrelevant segments and select continuous frames which have higher tIOU with the ground truth segments.

As the proposals generated previously have variable length it can't be fed directly into fully connected layer. That is why a temporal average pooling layer has been used before they are fed to the classification and regression module. This module has two sibling output branches. The first branch predicts the important scores of the proposals and the second branch outputs offset length of the proposal segment and its center using regression.

Authors have adopt a multi-task loss function to jointly train the network.

During the testing phase, predicted offsets are generated using the

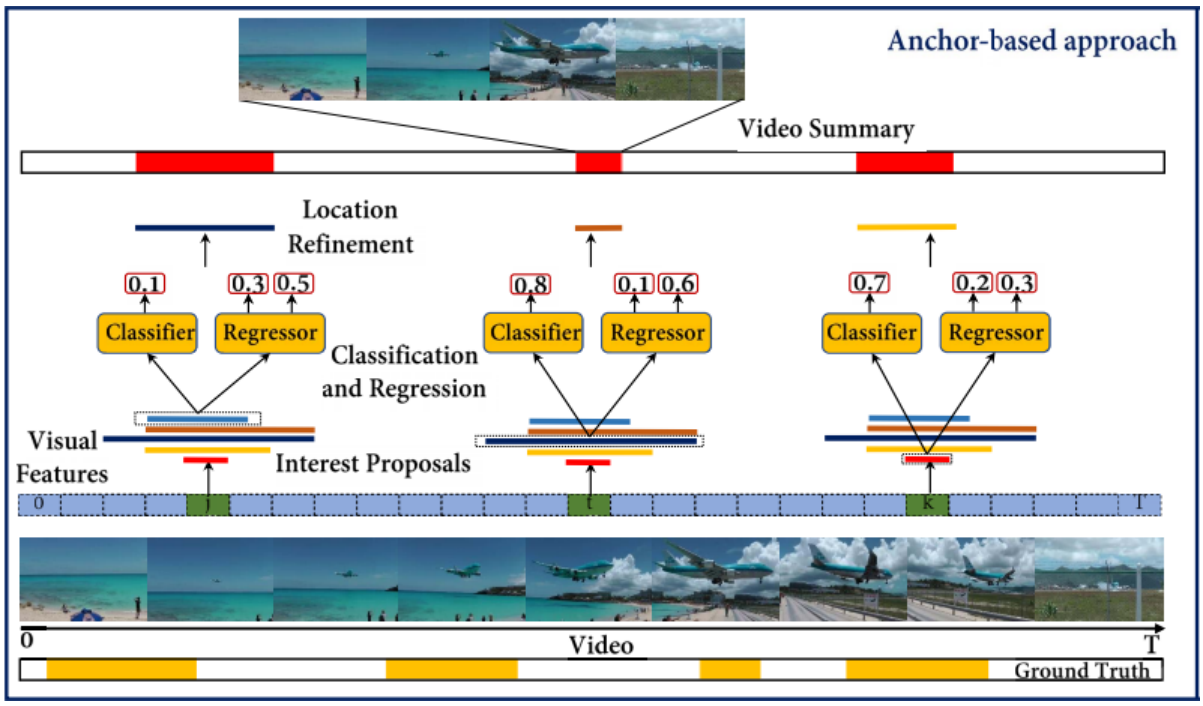

Figure 2.7: Anchor Based Model Pipeline [5]

model which is analogous to the training stage. Even though many predicted proposals have low confidence score they over high overlaps with each other. To solve this issue non-maximum suppression **NMS** is deployed. This process removes the segments which are redundant in nature. After that authors apply kernel temporal segmentation algorithm KTS to find out the shots in a video. To find each frame's importance score, the maximum value of the predicted segment at each temporal location is assigned. The shot level importance score is obtained by averaging the frame level importance score inside the same shot. Finally, frames are picked by maximizing the weight and maintaining the highest 15% length

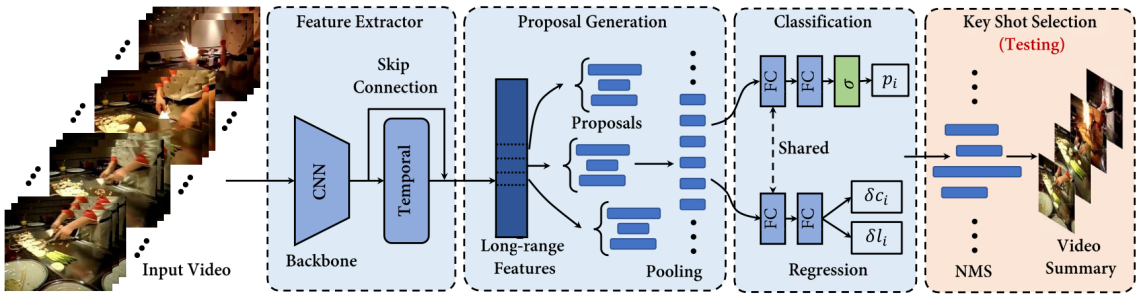

Figure 2.8: Anchor based DSNet architecture [5]

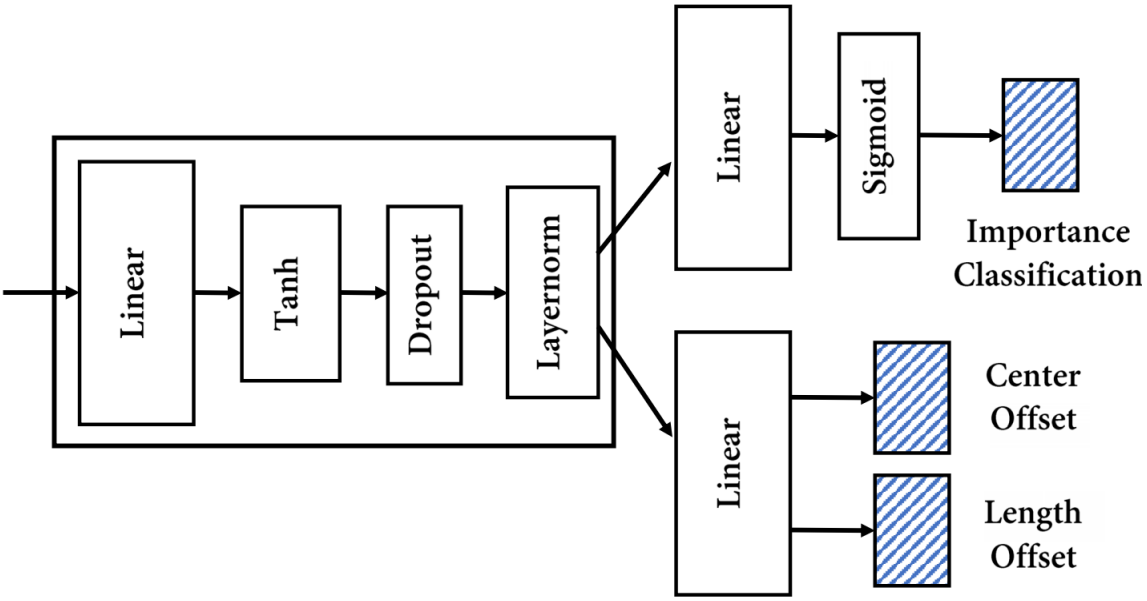

Figure 2.9: Components of the classification and regression module [5]

of the original video.

2.4.2 Anchor Free Approach

Generating interest proposal is a resource hungry computation. Because there can be a huge number of proposal count in a large size of video and labeling each of the proposals that are generated can be computationally not viable. More than that it might not be convenient to find out interest proposals for complex and dynamic scenes. Finally the hyper-parameters that are used to find the interest proposals might need tuning in many different scenarios. As a remedy of all this limitations this study proposes another method which is anchor-free approach. The basic methodology is divided into 3 parts for this architecture.

The feature extraction part is exactly the same as the anchor-based method which is finding the visual cues using GoogLeNet[10] and then applying self-attention[9] to find the temporal features. Final feature vector is achieved by concatenating these two vectors.

In anchor-free approach authors avoid finding proposals rather they propose to work with each video frame where they find out the segment location and the importance score of each frame.

The positive and negative weight assignment of each frame is simple. If the current frame is selected in the ground-truth summaries then it is assigned as a positive frame whereas the frame which is not present in the ground truth summary is assigned a negative weight in this process. To find out the exact location of a frame authors have maintained a ground-truth 2D vector that keeps the interval between current location and left-right boundaries of the segment the frame lies in. Authors also use a center-ness score to ensure the temporal location is close to the center of the predicted segment.

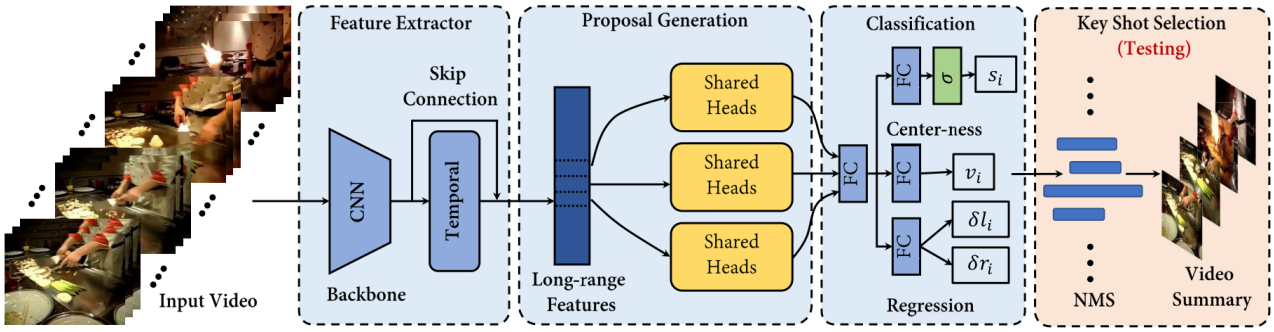

Figure 2.10: Anchor free model pipeline [5]

Figure 2.10 shows that anchor-free architecture contains a fully connected layer with two sibling output branches for importance classification and location regression. The location regression branch is responsible for finding the left-right boundary segment as well as the center-ness score. Again the authors have used a multi-task loss function to optimize all the parameters.

At each temporal location we get the importance score, the location prediction and the center-ness score by employing the training model. After that confidence score of each segment is calculated. Higher confidence score indicates a better segment. Also by using NMS the high overlapping and low confidence segments are discarded. Then frame level importance score is calculated just like anchor-based method and rest of the task of this portion is a complete replication of the anchor-based model.

This study has contributed in the ares of -

i. Applying region proposal network for generating interest proposals in videos.

ii. Making video summaries more temporally consistent and contextually sound.

iii. Using self-attention[9] for temporal feature extraction.

The best results of this study are produced by the anchor-based model in the case of the TVSum [4] dataset and for[3] dataset the anchor-free model was the better one.

Dataset	Model	Results
SumMe	Anchor Based	50.2
	Anchor Free	51.2
TVSum	Anchor Based	62.1
	Anchor Free	61.9

Table 2.4: Summarization results (F-Score) with DSNet

2.5 Attention based Video Summarization : VasNet

Due to the limitations of encoders and decoders in long sequences, the researchers at Google Research and Google Brains came up with the proposed method of attention mechanism [11]. By using an attention mechanism, the output can give attention to inputs in order to get the output. However, this study [12] proposes a model called VasNet which uses a simple self attention mechanism to summarize the video on a complete sequence to sequence basis. The self Attention mechanism is a technique that makes it possible for the inputs to interact with all the other inputs. This is basically an intra - attention mechanism.
The main goal was to substitute the other existing computational demanding methods that follow the RNN and encoder decoder based architecture and propose a model with soft self attention which follows simple architecture and is less computationally demanding.

The architecture can work with CNN feature vectors and can run in a single forward or backward pass for even sequences of different sizes.

The main objective of the architecture is to perform two key tasks which is firstly generating attention weights and finally calculating frame level importance score.

The architecture mainly consists of two modules in order to carry out each of the two above mentioned key operation -

- Self Attention Network
- Regressor Network (a Neural Network)

As input , CNN feature vectors are extracted from the sequences of video by using GoogleNet model and are passed to the self Attention Network Layer in order to generate weights of all the input features which is then averaged. The weighted average is then carried forward to the Regressor Network to perform regression and finally calculate the importance score of the frames.

As mentioned earlier, the main replacement of the RNN based models is with a self attention mechanism that follows soft attention. Therefore, the model does not need to traverse again and again forward and backward like the bi-direction BiLSTM as the attention mechanism allows it to obtain both the previous and next inputs in parallel. Also, the attention mechanism allows it to have no loss in information for sequences with longer range as the video size increases. This is because there is no method of embedding in between as the mechanism can work on the sequences of input directly.

Now let's look at the step by step features in the architecture.The input is sequences of CNN feature vectors extracted for each video frame, which is then passed onto the attention network layer where self attention mechanism is performed. In the Attention Network Layer, the attention vector et is then converted to the attention weights at with softmax function. Linear transformation C is applied to each input, weighted with attention vector at and averaged. This is then fed to the Regressor Network as its input.The Regressor Network mainly consists of two layers. The first layer consists of ReLU activation, dropout, and layer normalization. The

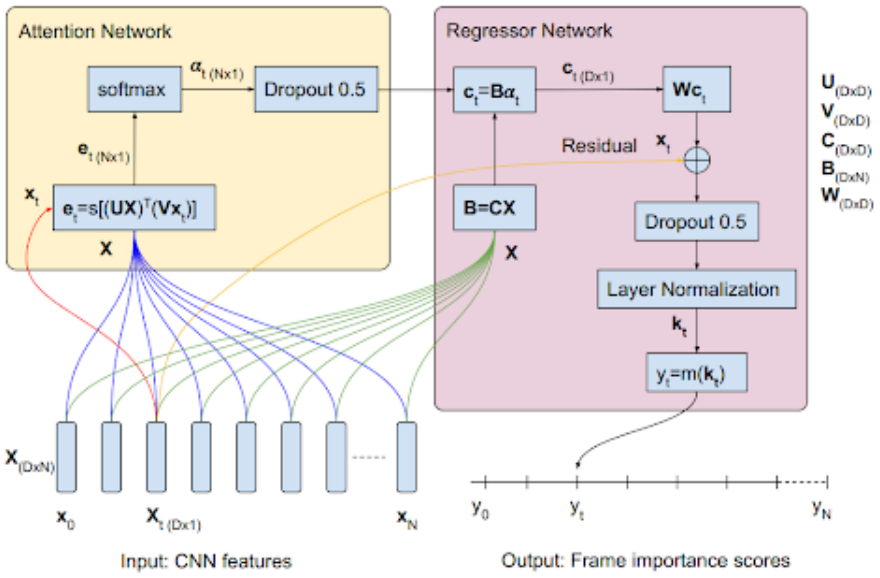

Figure 2.11: VasNet model Architecture [6]

second layer is of hidden unit sigmoid activation that performs the frame score regression.**C** and **W** are the weight matrices from the network which was learned and optimized during the training. A dropout for the weights generated by the attention is included for the purpose of regularizing the network. The f - score or the frame importance score calculated after performing regression on the data by the Regressor Network is then used to determine the keyshots. The following process happens mainly in two operations :

1. Potential Keyshot segment points are determined by analyzing the points from where there is a significant change in scene. The changes in the scene are determined by using the Kernel Temporal Segmentation (KTS) method

2. Among all the keyshots, some are selected while assuring that the total frame score is maximum within those set of keyshots and the total summarization should always be less than 85% of the original video length.

Keyshots are selected with Knapsack algorithm and then concatenated to a final video summarization.

The main contributions from the following study is that this model is easier to implement and uses less resource to run than LSTM encoder-decoder based methods and performs a sequence to sequence transformation without recurrent networks. Thus, the model outperforms the existing state of the art methods on the TvSum and SumMe benchmarks and also does not suffer from information loss for longer sequences

Two data sets SumMe and TvSum were used to find the f score using this VasNet model and the result is shown at Table 2.5

Dataset	Results
SumMe	49.7
TVSum	61.42

Table 2.5: Summarization results (F-Score) with VasNet

However, this model has some shortcomings in it as it Uses a single Regressor network to calculate the final frame level importance score and it has no clear idea on how to calculate the frame position.

2.6 Attention based Video Summarization : MSVA

The study [13] comprehends a model with an attention mechanism but instead of only working with visual features as input , it uses motion features as well in parallel to calculate the frame importance score. The architecture utilizes three feature sets from visual contents and combines it with the motion feature in parallel as input. Attention mechanism is used just before this feature set addition. The study [13] shows that using both static and motion features combinedly along with attention mechanism has proven improved results.The architecture model here used for video summarization is a supervised deep learning model called Multi-Source Visual Attention (MSVA).

The whole architecture consists of mainly four key operations -

1. Use of both Motion and Visual Features combinedly
2. Use of Attention mechanism

3. Use of different fusion techniques

4. Generating importance Score for frames

After applying uniform sub sampling to all frames of a video (two frames per second), the selected frames are passed as input to the pre-trained models to extract the following features GoogleNet [10] is used to extract the visual features from the pool 5 layer dimensions which is the image content.And for the motion content,the pretrained I3D (Inflated 3D ConvNet) model on Kinetics dataset, which is composed of human actions is used for extracting motion features. Basically we are extracting two types of features from motion content, i.e RGB and Optical flow. RGB feature mainly consists of the channel wise color information according to the diversity of the scenes. And, optical flow consists of the motion based features according to the frames in sequence.Then separate attention mechanisms are linked individually to each of the feature types.Attention mechanism is performed and the output from here is fed into two linear layers and one normalization layer to obtain a latent vector representation.All the latent vector results obtained from each of the feature layers are then fused together as one single vector representation of input. Addition fusion technique is used to combine all the three features latent representation.The result is passed to a linear layer (L3), followed by ReLU activation , dropout, normalization and another linear layer (L4). Finally,the output vector is fed into a sigmoid function that outputs importance scores for the input frames.

They have used several combinations of fusion techniques such as early , intermediate and late . And it was found that the intermediate fusion technique gives the best result.

The major key contributions from this paper were the use of the attention mechanisms along with multiple sources of visual features. Moreover, problems in the past experimental setups are detected and mitigated on valid cross validation folds. Thus, a better outcome for the SumMe dataset, while achieving similar results in comparison with other models on the

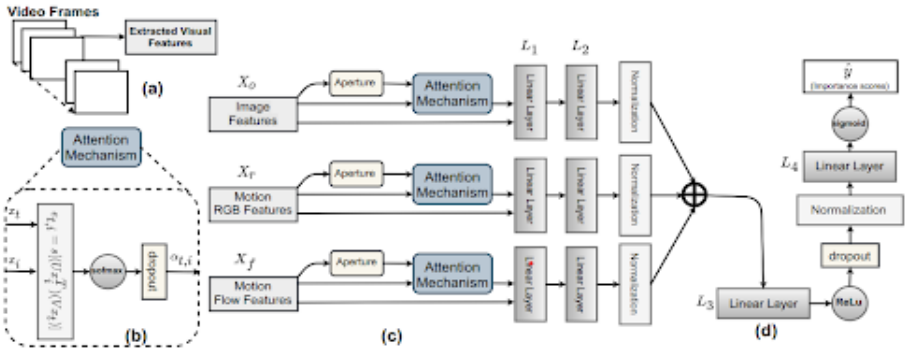

Figure 2.12: Multi-Source Visual Attention (MSVA) model with parallel self-attention [13]

TVSum dataset

This paper also used SumMe and TVSum data set and has shown furter improved f score result than the previously used VasNet model.Thw result is shown in table 2.6

Dataset	Results
SumMe	54.5
TVSum	62.8

Table 2.6: Summarization results (F-Score) with MSVA

2.7 Attention based Video Summarization : PGLSum

To address the limitations of previous RNN-based summarization architectures, such as modeling frames correlation in videos of longer lengths and the potential to run the training in parallel, self attention mechanism techniques were applied. However, the importance of maintaining temporal consistency was totally ignored even though temporal consistency is a very important aspect in video summarization.

Therefore, in the paper, a new supervised video summarizing approach [8] called the PGL Sum model has been developed. This architectural model uses both the Global and Local multihead attention layer in parallel to model the frames dependencies in different levels of abstraction in smaller parts of the video segments and also uses the technique called the

absolute positional encoding to encode the temporal positions of the video frames. Thus, solving all the earlier issues.

The entire architecture consists of the use of both Global Multihead Attention layer and Local Multihead Attention Layer in parallel along with a positional encoder to keep track of the temporal dependency. Moreover,fusion of Global and Local attention weights were done and lastly a Regressor Network Layer was also present to generate the final f score.

Using the GoogleNet model, multiple CNN feature vectors are extracted from the video and [10]the features are fed to the following two layers in parallel. The features are directly fed to the Global multihead attention layer where the frames,Äô dependencies with each other are modeled and within the Global multihead attention layer , absolute positional encoding happens. In parallel, the features are divided into several non-overlapping segments and each of which is connected to an individually different local multihead attention layer. Based on the pipeline working in parallel, the most important video frames within each segment are extracted according to the overall frames' dependencies within that specific part of the video. Next, feature addition is performed between the weights generated from both the Global multihead attention layer and the Local multihead attention layers, so that a new feature vector can be stated which determines each of the frames importance level according to its dependent relationship with the local and global multihead. Then the resultant output is added with the original deep feature vectors and fed into the Regressor Network followed by a dropout layer and normalization layer Finally, regression is done and we get the importance scores as output

From figure 2.14 , we can see, Q, K , V are feature vector inputs that are fed to the dot product attention layer to produce the attention weights of the vectors , which are then concatenated together and passed on into linear layers , through which we get the final output of the Global multihead Attention layer.

Absolute positional encoding is embedded within this layer to label the position of each of the frames in the sequence in order to keep track of the

temporal order. This encoding method uses sine and cosine functions of different frequencies.

The study has contributed the following -

1. To address the challenge of video summarization, the use of absolute positional encoding as part of the self-attention method in order to keep temporal consistency

2. To learn better modeling of frames' dependencies from human annotations, a novel architecture that embeds an absolute positional encoding component into global and local multi-head attention techniques.

The result is shown in table 2.7

Dataset	Results
SumMe	57.1
TVSum	62.7

Table 2.7: Summarization results (F-Score) with PGL SUM

However, the limitations were that the architecture did not show any clear method on how to choose segment size and didn‚Äôt change the

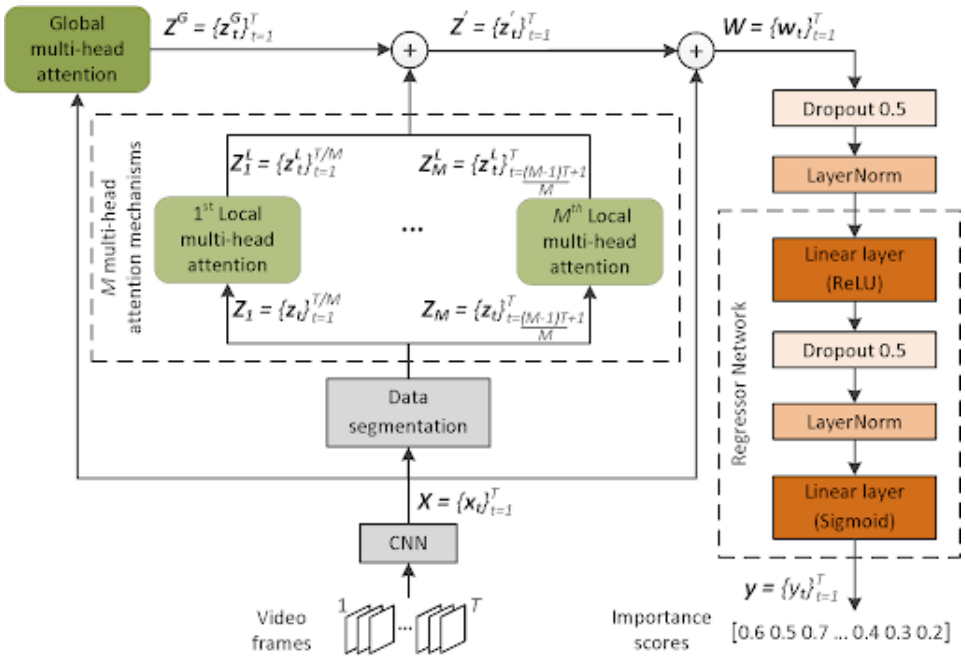

Figure 2.13: PGL SUM Architecture [8]

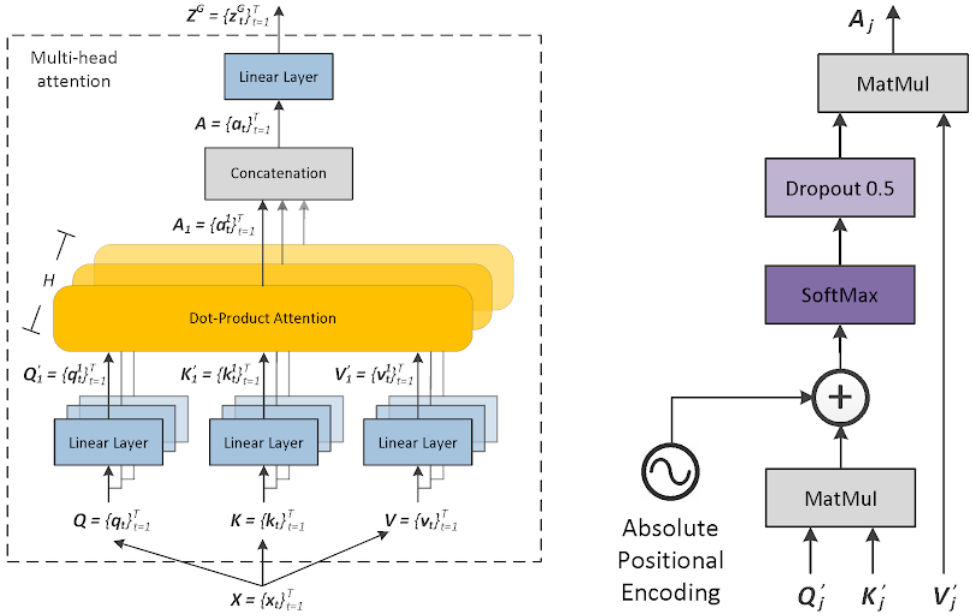

Figure 2.14: The applied dot-product attention that integrates information about the temporal position of the video frames [8]

attention head based on video size of the data set.

2.8 Issues with Existing Methods

2.8.1 Problems with RNN based Models

In every RNN based models, the forward and backward feeds have to traverse the whole network multiple times. This negatively affects the network's ability to model long-range dependencies as well as limits the amount of parallel operations during training. So the training procedure becomes slow, complex and resource hungry. More than that, RNN based networks can be parallelized to a limited extend. One common architecture which is modeled using RNNs are the encoder-decoder type of architecture. The basic limitation to such a network is that a single vector which holds all the context of all the frames of the video need to be passed. So the encoder-decoder architecture fixes the rest of the transitional code length. This disables the architecture to put different weight to different frames in the input sequence. So, there exists the high chance of missing the in depth temporal consistency of a video. From [1] and [2], we are able to find

this problems to be true.

2.8.2 Problems with Attention Based Models

The existing self-attention based summarization approaches ignore the temporal order of the video frames, that is the order of occurrence of frames in summarized video. Without temporal consistency, the result can never improve. Although the frame level importance score can be easily calculated with the attention based models to pick a frame in summarized video, the absence of temporal consistency makes the gap between ground truth summary and generated summary larger. So the frame sequence in generated summary needs to be handled separately. Also some model used fixed sized change points to segment the videos which definitely can't work well for all video sizes.

CHAPTER 3

PROPOSED METHOD

In our work, we propose a new architecture which is based on multi-head attention mechanism. Our work is greatly inspired by the architecture of DSNet[5] and PGLSum[8]. Our whole architecture can be divided into 3 major parts :

Figure 3.1: Modules in Architecture

3.1 Basic Pipeline

As mentioned before one of the major challenges of video summarization is maintaining the temporal consistency of the frames. Our objective is to overcome this challenge and with that objective in mind we have designed our pipeline. In our proposed pipeline we have worked with the feature extraction module and specifically improved the architecture of temporal feature extractor.

Then we took used proposal generation module which is based on DSNet[5]. This module basically has two different configuration. One

is the Anchor Based configuration and another one is the Anchor Free configuration. Lastly, based on the configuration of the Proposal Generation module we use the classification and the regression module to find out the proposals' importance score as well as calculate the frame level importance score using the anchor based method and the anchor free method accordingly.

So, in our architecture we have used local attention module to find the temporal consistency and important frames inside the segments or shots of an specific video. Which then works together with the global attention to help the architecture take a better decision thinking about the overall shot sequences inside that video. For the task of combining two attention weights produced by the local and global attention modules we have used additive techniques which in short is basically element wise adding two vectors produced by the respective modules.

3.2 Feature Extractor

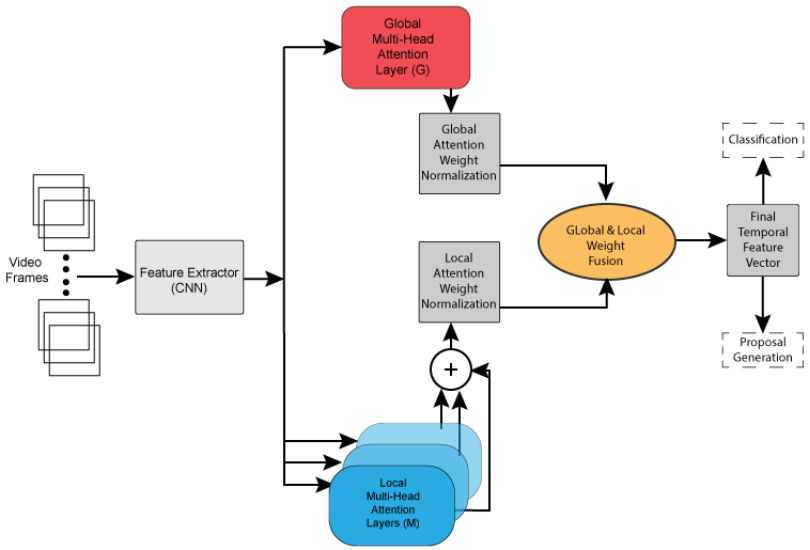

Figure 3.2: Feature Extraction Module

3.2.1 Spatial Feature Extractor

In our proposed architecture, the visual features are extracted using the same method as previous works.So we use GoogLeNet[10] to extract the visual features of the video .We pass the whole video in the GoogLeNet architecture and take the output from the 4^{th} last layer which excludes the last three layers. These extracted features comes in the form of a vector which has the same size as the GoogleNet's 4^{th} last layer. The size of vector we get is 1024×1024 in shape. Lets assume that we have a video V which has t frames in it. So the visual feature vector after using GoogLeNet will be $V_i, i\epsilon\{1,...,t\}$.

3.2.2 Temporal Feature Extractor

For temporal feature extraction, our unique architecture comes into place. We use the visual feature vector extracted from previous step as our input for the next temporal feature extraction layer. Here we use our local and global attention attention module to get the final temporal feature vector $\{T_j\}_1^t$ which is then added with the original spatial feature vector V. The temporal feature extractor works in a parallel manner. The input to this module goes into two different streams: the local attention blocks and the global attention block. Both the attention blocks have multiple heads inside them. Which in short means we use multi-head attention mechanism for calculating the attention scores. Though the global attention module has only one attention block to calculate the scores the local attention module has multiple attention blocks. We call these attention blocks segmented localized attention because these work with segment of the video. That means the shots or the change points the video has, we try to apply one local attention block to each of the shots of the video the model is trying to learn from. The main goal of applying the local attention block is to understand the most important frame inside a shot itself. Rather than allowing the global attention to decide for all the frames and shots inside a video, local attention blocks help to find

out the important frames or features more independently. For selecting the number of segments in the local attention module which give the best results for in our work, we have extensively studied the videos' nature of the dataset and the change points in the videos. Upon our study we have come to an conclusion that the segment number should be around the max number of change points of the video. Then again the number of segments we can choose practically will also depend on the output vector size of the spatial feature vector. As the number of segments must be a divisor of the output vector size.

3.2.3 Attention Mechanism

The basic attention mechanism which was introduced by [11] has been applied in previous works of video summarization. But scaled dot product attention was introduced in transformer[9] which gives better results in various fields of deep learning application. We have also used the scaled dot product multi-head attention to calculate the attention score for the input vector. The multi-head attention structure is given in 3.3. Here we can see that all the vectors **Query**(Q), **Key**(K) and **Value**(V) go through a linear layer first. After that the actual attention value is calculated.

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.1}$$

Difference between scaled dot product attention and multi-head scaled dot product attention is that in multi-head attention the attention is calculated h times.

$$head_i = Attention\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{3.2}$$

$$MultiHead(Q,K,V) = Concat(head_1, head_2....., head_h)W^o \tag{3.3}$$

The result of the h iterations of Scaled-Dot Transformation Attention is then spliced, and the value produced by additional linear transformation

is the Multi-Head Attention result.

Here we encounter another major contribution of our work that is the use of different positional encoding of our input vector. Positional encoding was first introduced in the original work of transformer[9]. The idea of encoding was to include the position information of words in a sentence as attention models does not work like RNN models and can't hold positional information by default. So the authors of transformer[9] introduced a d-dimensional vector which carries information about a word's specific position within a sentence. For our work we use the same idea but here we encode the position of frames inside a video with the help of the encoding vector. However, we don't apply encoding on the representative vector which was proposed in transformer, rather we apply it on the elements of the original sequence.

So, if we have a video which has t frames in it and the representational feature vector has d dimension then we calculate a positional encoding vector of size $t \times t$ [8] but not of size $t \times d$ which is proposed by transformer.

$$f(t)^{(i)} = \begin{cases} sin(\omega_k.t), & i = 2k \\ cos(\omega_k.t), & i = 2k+1 \end{cases} \tag{3.4}$$

where,

$$\omega_k = \frac{1}{10000^{2k/d}}$$

This provides us the absolute positional encoding vector to calculate the attention score from the self-attention block. To further extend our model's robustness we have used relative positional encoding in our work which is basically inspired by [14]. The idea of relative positional encoding says that during attention computation, relative positional information is added to keys(K) and values(V) on the fly, rather than merely mixing semantic embedding with absolute positional ones. Simply put, relative positional encoding uses pairwise distances for creating positional encoding. The relative positional information is given as a sub-component of the original Value(V) vector.

$$z_i = \sum_{j=1}^{n} \alpha_{ij} \left(x_j W^V + a_{ij}^V \right) \tag{3.5}$$

But the softmax operation remains unchanged from the original transformer.

$$\alpha_{ij} = \frac{expe_{ij}}{\sum_{k=1}^{n} expe_{ik}} \tag{3.6}$$

$$e_{ij} = \frac{x_i W^Q \left(x_j W^K + a_{ij}^K \right)^T}{\sqrt{d_z}} \tag{3.7}$$

If we represent two input elements as x_i and x_j then the temporal relation between them is represented by vectors a_{ij}^V and a_{ij}^K. These matrices are used in equation 3.5 and equation 3.7. Using relative positional encoding makes the model more efficient while the model is learning from a long range input feature vector.

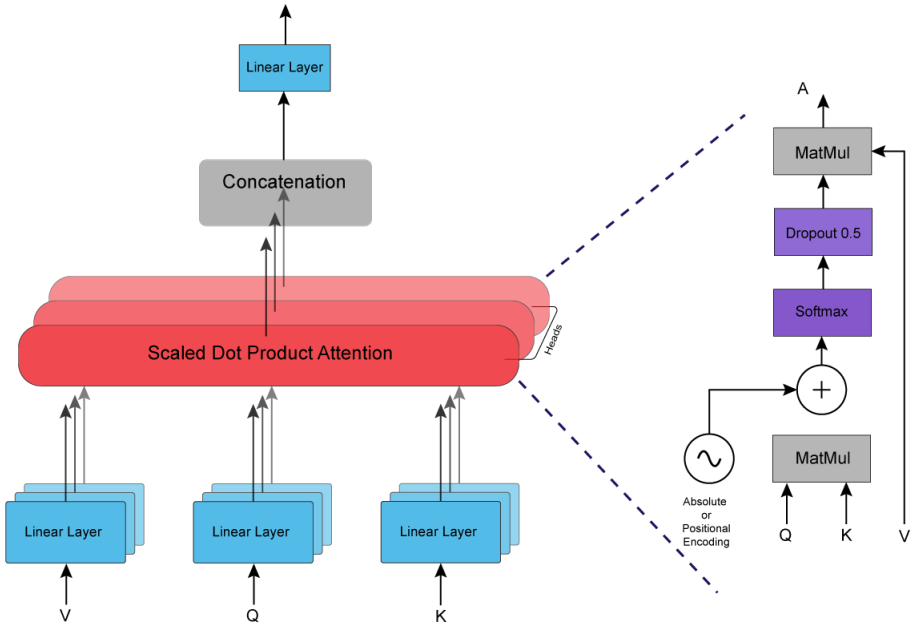

Figure 3.3: Inside Multi-Head Attention Layers

3.2.4 Attention Weight Fusion

After the local and global attention weights are calculated separately but in parallel, both of the local weights and the global attention weights

are normalized to contain them in a certain range. As our architecture works with 2 parallel streams of attention modules we preserve information from both of the attention using additive fusion. Simply explained, we perform a element-wise vector addition for both the attention weight of normalized local attention block and normalized global attention block.

3.3 Classification and Regression

The classification and regression module is inspired from the previous work of DSNet[5]. As mentioned in our objective we try not to restrain our model to use fixed size proposals for summarized videos. So we perform our classification and regression task in both way to prove that fixed size proposals are not a good idea for generalized video length. So, we design our model so that we can perform the classification and regression with fixed anchor size and with frame level importance score. In the classification module we use the sigmoid activation function to find out either a proposal or a frame can be included in the summarized video. The input which the classification and regression module takes is the final feature vector which is summation of the final temporal vector and the spatial vector. The regression module basically determines the position of the proposal or the frame and where it should be placed inside the summarized video. For the anchor based configuration which works with fixed sized segments of proposals the regression module tries to fix the position of a segment like a sliding window. On the other hand for frame level we try to find out the centerness score as described in DSNet[5].

CHAPTER 4

EXPERIMENTAL SETUP AND ANALYSIS

4.1 Dataset

We assessed our proposed method with two benchmark datasets, SumMe[3] and TVSum[4]. The video lengths in these datasets are 1 to 10 minutes. SumMe [3] has 25 raw videos showing a diverse number of events such as holidays and sports. TVSum [4] has 50 videos downloaded from YouTube. These videos can be categorized into 10 types that shows day to day activities, such as changing vehicle tire, grooming an animal etc. Both of SumMe and TVSum datasets provide frame-level importance scores for each video. We will use these frame-level importance scores as ground-truth values.

4.2 Evaluation Metrics

F-Score is the most popular and widely used metrics of evaluating the performance of a video summarizer. Suppose a generated summary of a video is S and the ground truth summary of that video is annotated as GT.To get the F-Score we first compute the precision P and the recall R for each pair of S and GT based on the temporal overlaps between them, as follows:

$$P = \frac{\text{overlapped time of } S \text{ and } GT}{\text{time of } S} \tag{4.1}$$

$$R = \frac{\text{overlapped time of } S \text{ and } GT}{\text{time of } GT} \tag{4.2}$$

Positional Encoding Type	Segments	Local Attention Heads	Global Attention Heads	Learning Rate
Relative	64	16	16	2e-07
Absolute	64	16	16	1e-0

Table 4.1: Configurations for TVSum Dataset

Finally, the F-Score is calculated with:

$$F = \frac{2 \times P \times R}{P + R} \times 100\% \tag{4.3}$$

4.3 Experimental Settings

For training, we used a GeForce RTX 3080 GPU. We were limited to segment our feature vector input due this. We used 32 and 64 segments. If we were provided with more powerful GPU, we could've segmented our feature vector input upto 128 segments. We ran our training based on either Anchor Based or Anchor Free approach first. Then based on our dataset and positional encoding type, we segmented our input vector between 32 and 64 segments. We ran our training for 300 epochs. Our learning rate varied based on configuration. The learning rates based on configuration is given in table 4.3

Positional Encoding Type	Segments	Local Attention Heads	Global Attention Heads	Learning Rate
Relative	64	8	8	2e-07
Relative	32	8	8	5e-08
Absolute	32	8	8	1e-0

Table 4.2: Configurations for SumMe Dataset

4.4 Result and Analysis

From table 4.3 and table 4.4, we can see that we beat the state of the art for the TVSum dataset in Anchor Based approach with relative positional encoding of 64 segments, 16 Local Attention Heads and 16 Global Attention Heads. We also beat the state of the art for SumMe dataset in Anchor Free Method with relative positional encoding of 64 segments, 8 Local Attention Heads and 16 Global Attention Heads.

Model Type	Data Set	Positional Encoding Type	Segments	Local Attention Heads	Global Attention Heads	F-Score
Anchor Based	TVSum	relative	64	16	16	65.31
		absolute	64	16	16	62.42
	SumMe	relative	64	8	16	55.29
		relative	32	8	8	53.14
		absolute	32	8	8	51.46
Anchor Free	TVSum	relative	64	16	16	64.13
		absolute	64	16	16	64.61
	SumMe	relative	64	8	16	59.62
		relative	32	8	8	53.14
		absolute	32	8	8	51.46

Table 4.3: Results with configuration

Method	SumMe	TVSum
vsLSTM [1]	37.6	54.2
dppLSTM [1]	38.6	54.7
ActionRanking [15]	40.1	56.3
vsLSTM + Attention [16]	43.2	-
dppLSTM + Attention [16]	43.8	-
H-RNN [17]	42.1	57.9
A-AVS [7]	43.9	59.4
M-AVS [7]	44.4	61.0
SF-CVS [18]	46.0	58.0
SUM-FCN [2]	47.5	56.8
CRSum [19]	47.3	58.0
SUM-DeepLab [2]	48.8	58.4
SUM-GDA [20]	52.8	58.9
SMLD [21]	47.6	61.0
H-MAN [22]	51.8	60.4
SMN [23]	58.3	64.5
DS-Net [5]	50.2	62.1
PGLSUM [8]	55.6	61.0
Ours (GSLA)	**59.62**	**65.31**

Table 4.4: Comparison between our approach and other supervised approaches in SumMe[3] and TVSum[4] Dataset

4.4.1 Performance based on encoding

From figure 4.1, we can have some key intuitions.

- Almost all the configuration produces better results with relative positional encoding
- For SumMe dataset in Anchor Free configuration, the result difference is significantly high than the previous researches

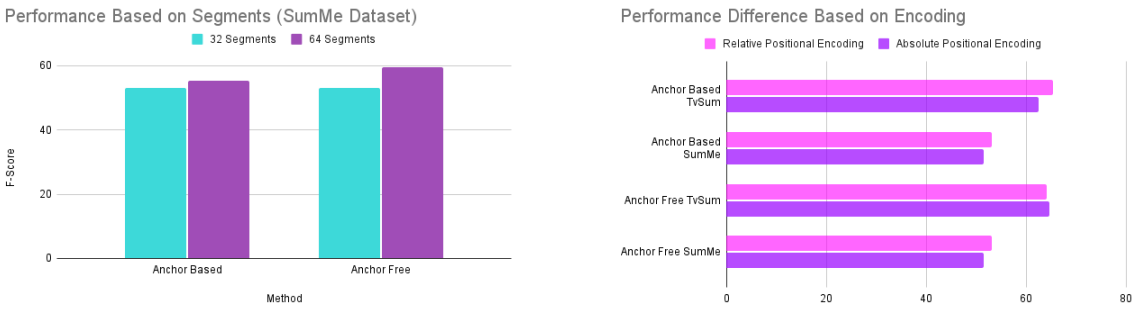

Figure 4.1: Comparison based on number of segments and encoding type

- Relative positional encoding almost solves the issue of fewer video numbers with shorter length in SumMe dataset as we gain F-score close to TvSum dataset, which doesn't occur in previous researches.

4.4.2 Performance based on segmentaion in SumMe dataset

Particularly in SumMe dataset, we can get some key intuitions from our observation in figure 4.1 -

- In both case of anchor-based and anchor-free method we see a higher number of segments give better F-Score.

This happens because segment numbers are determined after studying the change point number distribution of the videos in a dataset rather than using fixed and fewer number of segments.

4.4.3 Comparison with DSNet and PGL SUM

We can observe some key comparisons between DSNet, PGLSUM and our proposed method from figure 4.2 -

- Our Anchor free configuration for SumMe beats DSNet by a landslide and almost gets close to the result of TVSum which rarely occured in researches before.

- Also the result produced by our architecture is far better than DSNet in SumMe for anchor free proving that fixed sized anchors are not a good choice for all sizes of videos in different datasets.

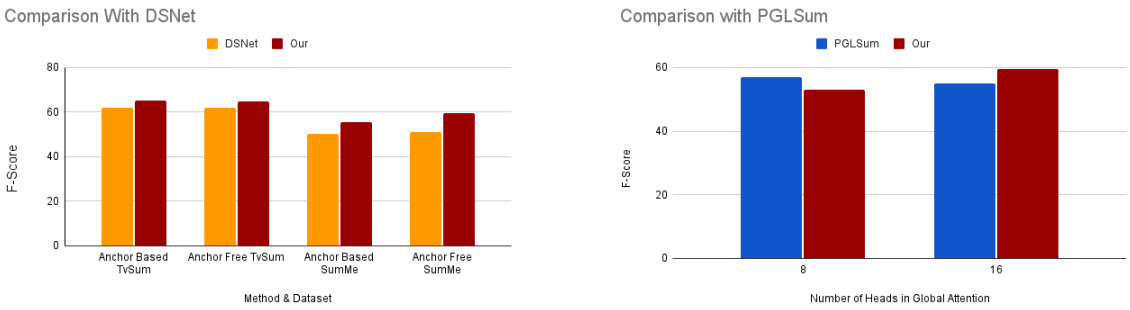

Figure 4.2: Comparison with DSNet and PGLSum

- For PGLSUM, we can see that when the number of Heads in Global and Local Attention is increased, we get a better result than PGLSUM. This proves that the increasing number of attention head has a huge impact in our result although this increases computational complexity.

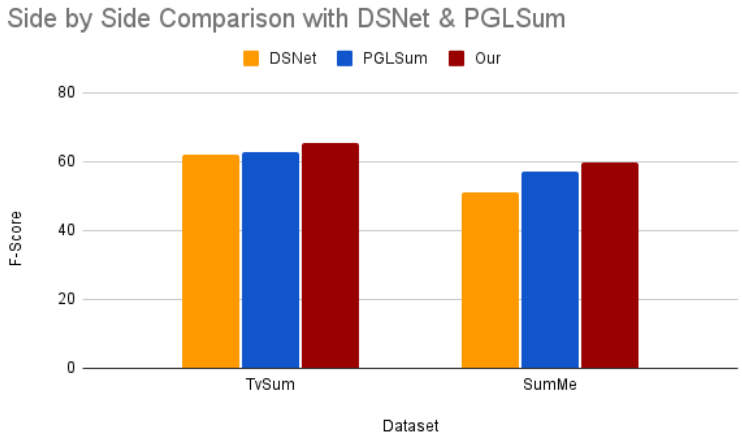

Figure 4.3: Overall Comparison

4.4.4 Overall Observation

If we compare our best result for each configuration with DSNet and PGLSUM, we get to conclude that -

- For TVSum dataset, we get best result with Anchor Based approach having relative positional encoding with 64 segments, 16 Local and 16 Global Multi Head attention.

- For SumMe dataset, we get best result with Anchor Free approach having relative positional encoding with 64 segments, 8 Local and 16 Global Multi Head attention.

So, our key results induce some important observations. The relative positional encoding produces better results in all configuration compared to absolute positional encoding. The segment number plays a significant role in improving the result. For both datasets we get the best results for 64 segments. Lastly for the case of fixed size anchor, even though we have got the best result for TVSum Dataset with the anchor based results the anchor sizes don't act cohesively in SumMe dataset. The anchor free method produces the best results for SumMe dataset.

CHAPTER 5

CONCLUSION AND FUTURE WORK

Based on our experimental results we can conclude that we do not have to use a fixed size anchor to get better results in video summarization. Results for both TVSum and SumMe dataset have improved significantly compared to the past state-of-the-art methods used by previous studies. We have been able to do that even without the help of fixed size proposals. Our architecture solve the problem of finding fixed size proposal lengths for the task of video summarization. We have also used attention in an elegent way so that both our computational complexity and attention weight calculation can keep a balance. We have used positional encoding to alleviate the problem of computational cost in attention blocks. Again using the idea of local attention and global attention helped us achieve the attention weights perfectly. The number of segments should be determined based on the change points of videos inside a dataset. As the number of segments go up in the videos, our model should be trained with higher segment number to get the full benefit of our architecture. Also the number of heads in multi-head attention should be determined based on segment length as the heads gives us attention weights based on the correlation between frames.

For our future works, firstly we will try to implement our architecture on a new benchmark dataset. We will explore the usage of multiple Fusion technique which include multiplicative fusion, average fusion and argMax fusion. Apart from that, we also think using better CNN architecture to extract visual feature other than GoogleNet[10]. We also aim to apply

more or less segmentation based on the video size of the dataset.

BIBLIOGRAPHY

[1] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, "Video summarization with long short-term memory," in *Computer Vision – ECCV 2016* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), (Cham), pp. 766–782, Springer International Publishing, 2016.

[2] M. Rochan, L. Ye, and Y. Wang, "Video summarization using fully convolutional sequence networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 347–363, 2018.

[3] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, "Creating summaries from user videos," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), (Cham), pp. 505–520, Springer International Publishing, 2014.

[4] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes, "Tvsum: Summarizing web videos using titles," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[5] W. Zhu, J. Lu, J. Li, and J. Zhou, "Dsnet: A flexible detect-to-summarize network for video summarization," *IEEE Transactions on Image Processing*, vol. 30, pp. 948–962, 2021.

[6] J. Fajtl, H. S. Sokeh, V. Argyriou, D. Monekosso, and P. Remagnino, "Summarizing videos with attention," in *Computer Vision – ACCV 2018 Workshops* (G. Carneiro and S. You, eds.), (Cham), pp. 39–54, Springer International Publishing, 2019.

[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, 2020.

[8] E. Apostolidis, G. Balaouras, V. Mezaris, and I. Patras, "Combining global and local attention with positional encoding for video summarization," in *2021 IEEE International Symposium on Multimedia (ISM)*, pp. 226–234, 2021.

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, pp. 5998–6008, 2017.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[11] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[12] J. Fajtl, H. S. Sokeh, V. Argyriou, D. Monekosso, and P. Remagnino, "Summarizing videos with attention," in *Asian Conference on Computer Vision*, pp. 39–54, Springer, 2018.

[13] J. A. Ghauri, S. Hakimov, and R. Ewerth, "Supervised video summarization via multiple feature sets with parallel attention," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6s, IEEE, 2021.

[14] P. Shaw, J. Uszkoreit, and A. Vaswani, "Self-attention with relative position representations," 2018.

[15] M. Elfeki and A. Borji, "Video summarization via actionness ranking," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 754–763, IEEE, 2019.

[16] L. Lebron Casas and E. Koblents, "Video summarization with lstm and deep attention models," in *International Conference on MultiMedia Modeling*, pp. 67–79, Springer, 2019.

[17] B. Zhao, X. Li, and X. Lu, "Hierarchical recurrent neural network for video summarization," in *Proceedings of the 25th ACM international conference on Multimedia*, pp. 863–871, 2017.

[18] C. Huang and H. Wang, "A novel key-frames selection framework for comprehensive video summarization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 2, pp. 577–589, 2019.

[19] Y. Yuan, H. Li, and Q. Wang, "Spatiotemporal modeling for video summarization using convolutional recurrent neural network," *IEEE Access*, vol. 7, pp. 64676–64685, 2019.

[20] P. Li, Q. Ye, L. Zhang, L. Yuan, X. Xu, and L. Shao, "Exploring global diverse attention via pairwise temporal relation for video summarization," *Pattern Recognition*, vol. 111, p. 107677, 2021.

[21] W.-T. Chu and Y.-H. Liu, "Spatiotemporal modeling and label distribution learning for video summarization," in *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6, IEEE, 2019.

[22] Y.-T. Liu, Y.-J. Li, F.-E. Yang, S.-F. Chen, and Y.-C. F. Wang, "Learning hierarchical self-attention for video summarization," in *2019 IEEE international conference on image processing (ICIP)*, pp. 3377–3381, IEEE, 2019.

[23] J. Wang, W. Wang, Z. Wang, L. Wang, D. Feng, and T. Tan, "Stacked memory network for video summarization," in *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 836–844, 2019.