Multi-Annotation Attention Model for Video Summarization

Hacene Terbouche
Powder
Paris, France
hacene.terbouche@gmail.com

Maryan Morel
Powder
Paris, France
maryan@powder.gg

Mariano Rodriguez
Powder
Paris, France
rdiguez.mariano@gmail.com

Alice Othmani
LISSI, UPEC, Université Paris-Est
Vitry-sur-Seine, France
alice.othmani@u-pec.fr

Abstract

In the last decade, the supply of online video content exploded. Automatic video summarization has become necessary to allow content consumers to briefly glance at the video's content. However, the notion of video summary is subjective and thus requires multiple annotators to define the ground truth. Existing video summarization techniques are limited in many ways. First, existing summarization techniques aggregate multiple annotations using the average operation and use these estimates to train a learning model to make predictions on unseen videos. Second, the use of RNN-based architecture to model long-range dependencies. Third, the amount of annotated data available for general video summarization is too small to train visual models from scratch. To mitigate these issues, this work proposes a new end-to-end probabilistic framework called Multi-Annotation Attention Model (MAAM) optimized using the Expectation-Maximization algorithm where the true label is treated as a latent variable. The MAAM framework has several advantages: (i) it exploits multiple annotations from different human-labelers and thus combines model training with the label aggregation, (ii) it models the temporal dynamics representations of videos through an attention mechanism, and (iii) benefits from the power of pretrained visual encoders namely the Vision Transformer (ViT). The proposed approach is evaluated on two public datasets TVSum and SumMe. Our method significantly outperforms state-of-the-art methods on both datasets.

1. Introduction

Video summarization defines as reducing a (potentially long) video into a shorter form that includes the key scenes and thus enables the viewer to comprehend the video's content. A video summary should not only include the highlights from the video but also meet additional requirements like diversity, representativeness, and visual and semantic coherence [2]. Video summarization can be formulated as highlight detection, i.e. a subset selection problem on the basis of a learned model predicting the relevance of each frame or segment. It can be also be seen as a ranking problem ordering the importance of two or many segments/frames [2].

Current video summarization techniques are mainly based on deep neural networks. Usually the task is formulated as frame or segment-level importance score prediction. Several approaches have been proposed to model variable-range dependencies between frames using recurrent neural networks (RNN) [33, 35, 36]. However, RNNs suffer from vanishing gradient when treating long sequences as well as the lack of parallelization resulting in slow training. Other methods proposed attention mechanisms to address this, without taking into consideration the frame ordering [7, 8, 16, 17].

The annotation process for video summarization is a challenging problem due to the subjectivity inherent to the task. Obtaining reliable ground truth labels is then infeasible. A popular approach is to ask multiple human annotators to assess the importance of every part of the video [9, 27]. The performance of any video summarization system is computed in relation to all of the human annotators, transforming the problem into a multi-annotation learning problem. State-of-the-art methods aggregate multiple annotations into a single ground-truth annotation to train on. Encouraged by many works that train directly on multiple annotations [24–26, 31], this work follows this direction.

Motivated by limitations of the existing methods discussed above, this paper introduces a new supervised method for video summarization. First, we introduce the Averaged Annotation Attention Model (AAAM) that incorporates multi-head attention module enhanced with positional encoding and frames windowing. We then extend it by proposing a new end-to-end probabilistic framework called Multi-Annotation Attention Model (MAAM) optimized using the Expectation-Maximization algorithm where the true label is treated as a latent variable. In summary, our contributions are the following:

- A new end-to-end probabilistic framework called Multi-Annotation Attention Model (MAAM) designed to exploit labels produced by multiple annotators in the context of video summarization.
- Many issues found in the current evaluation protocols are presented and a more robust evaluation protocol is proposed
- Evaluation of the proposed methods on two public datasets (TVSum and SumMe). MAAM substantially outperforms state-of-the-art methods, showing genuine benefits of our contributions.

2. Related Work

2.1. Supervised Video Summarization

Video summarization techniques can be categorized into supervised and unsupervised methods. We focus on supervised video summarization techniques that train a deep learning model through human annotation. Readers are referred to [2] for a more comprehensive review of the literature.

Early works use RNN to model the temporal dependencies [33, 35, 36] within a video. For instance, bidirectional LSTM (Long-Short Term Memory) models are used to predict the frame level scores [33]. The use of Determinantal Point Process (DPP) was also proposed to enhance the summary diversity by to modeling pairwise frame-level repulsiveness [33]. Other methods have been proposed using convolutional LSTM for simultaneously modeling spatial and temporal structure of video for summarization [13, 32]. Another line of research introduced an attention mechanism by combining of RNN architectures [5, 10].

Recent methods are based on full attention models [3, 7, 8, 16, 17]. The VASNet model (Video Attention Summarization) proposed in [7] combines a dot-product attention layer and a regression network. Differently, the MSVA (Multi-Source Visual Attention) model [8] combines three types of features: image features, motion RGB features and motion flow features with a dot-product attention layer to build a summarization system. In order to model the video temporality at various levels of granularity, the PGL-SUM (Positional Global Local Summarization) [3] combines global and local multi-head attention mechanisms. More recent works incorporate video-text joint modeling to predict frame-level scores [15, 18].

2.2. Multi-Annotation Learning

It is very difficult if not impossible to obtain a single objective and reliable label for each sample in a video summarization dataset. To improve data quality, practitioners often need to ask several domain experts or possibly a large number of non-experts to contribute to the labeling effort [29]. As a result, a single sample is associated with multiple labels, which may be conflicting in some instances.

A common approach (called *two-stage* approaches [14]) rely on aggregating the annotations before the training, using a majority vote for a classification problem or averaging for a regression problem. However, this process may yield noisy *ground truth* labels and hinder the performance of supervised machine learning models.

On the other hand, *one-stage* approaches train the the aggregation algorithm along with the model. Several works proposed probabilistic models [24–26, 31] to represent the aggregated label, that could be learned using the Expectation-Maximization algorithm [24]. Among these methods, Rayker et al. [25] model the true labels as hidden variables and evaluate the expertise of different annotators.

3. Proposed Method

Section 3.1 describes the problem and set the notation. Next, section 3.2 presents the Averaged Annotation Attention Model (AAAM), the proposed framework is trained using the averaged annotations. Section 3.3 describes the Multi-Annotation Attention Model (MAAM), our multi-annotation framework.

3.1. Problem Statement

Given a dataset with N human-annotated videos $\mathcal{D} = \{(V_i, Y_i)\}_{i=1}^{i=N}$, where V_i is the video and Y_i is the annotation. Videos can be of different lengths $\{T_i\}_{i=1}^{i=N}$, where T_i is the length in terms of the number of frames for video V_i. The extracted RGB frames from video V_i are $V_i = \{F_t\}_{t=1}^{t=T_i}$.

Each video is annotated by M human annotators, where the annotations for the video V_i can be written as $Y_i = \{Y_i^j\}_{j=1}^{j=M}$. As a video is a sequence of frames, then Y_i^j is a sequence of annotations along the time dimension $Y_i^j = \{y_{i,t}^j\}_{t=1}^{t=T_i}$. Therefore, $y_{i,t}^j \in \mathcal{R}$ is a real value assigned to the t^{th} frame by the j^{th} annotator for th i^{th} video in the dataset, indicating the importance of the frame in the sequence.

The presented setting is a multi-annotation problem as each sample of the dataset has multiple ground-truth annotations. To mitigate that, we can reduce the annotators dimension by averaging across it. Then, for each video we get

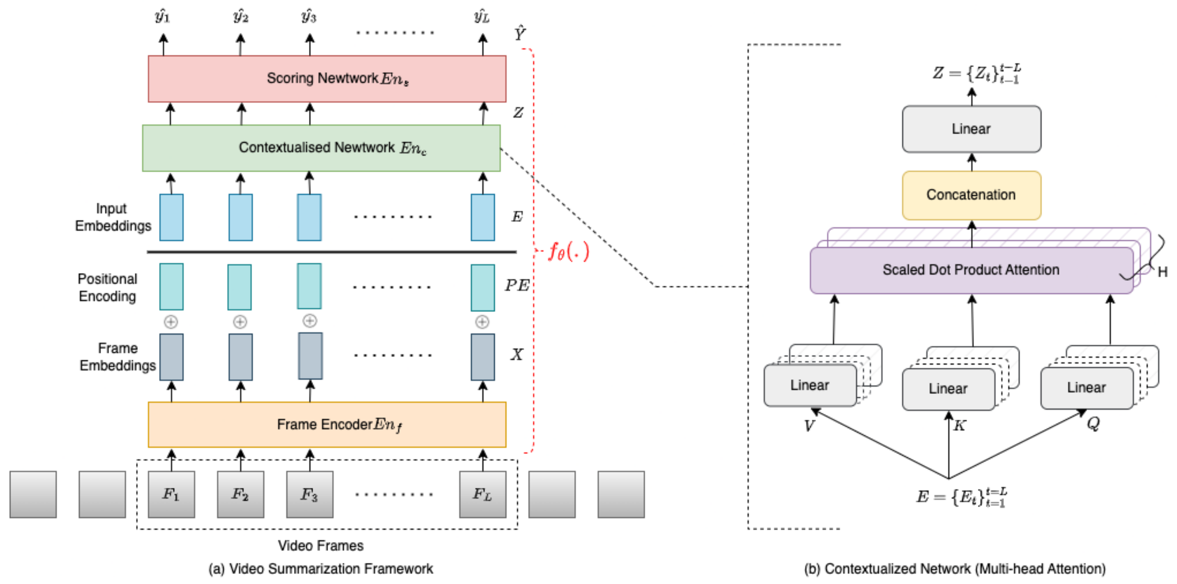

Figure 1. Illustration of the proposed Averaged Annotation Attention Model (AAAM). The input to the framework is a window of L frames from a given video. Figure (a) depicts the proposed architecture and the interconnection between the three networks (E_f, E_c and E_s). Figure (b) shows the details of the multi-head attention module.

the average annotation as $\overline{Y_i} = \{\overline{y_{i,t}}\}_{t=1}^{t=T_i}$, where:

$$\overline{y_{i,t}} = \frac{1}{M} \sum_{j=1}^{j=M} y_{i,t}^{j} \tag{1}$$

This aggregation method gives the same level of confidence for each annotator despite the fact that annotators might have biases in their labeling process.

The goal of a video summarization module is to learn the parameters θ of a mapping function $f_\theta(.)$ that produces the highlight scores from the sequence of frames for a given video:

$$\tilde{Y}_i = \{\tilde{y}_i\}_{t=1}^{t=T_i} = f_\theta(V_i) = f_\theta(\{F\}_{t=1}^{t=T_i}) \tag{2}$$

3.2. Averaged Annotation Attention Model

Our AAAM framework is based on a multi-head attention module that captures the frames' dependencies and produces contextualized representations of the input sequence. The output of this attention module is fed to a score prediction module that outputs the importance of each frame in the sequence. The predicted scores are then compared to the ground-truth scores (average annotations) and thus the model is trained in a supervised setting.

Based on the notation of the previous section, each video V_i of T_i frames is split into one or more windows of L frames, where L is the sequence length of the model. This procedure is accomplished for two reasons. First, videos have different lengths and so they have to be padded to the longest length or truncated to the smallest length during training in order to stack them into batches (tensors) . Second, the model might find difficult to predict highlight score of the frames when the video is too long.

Considering the size of the available videos annotated by humans, it is difficult to train from scratch both a visual extractor for RGB frames and the highlight prediction model. We, therefore, investigate using pretrained vision models that can be used to encode separate frames. Then, each frame in the window is encoded using the RGB encoder En_f (Frame Encoder) to get a set of deep feature representations $X = \{X_t\}_{t=1}^{t=L}$, where $X_t \in \mathcal{R}^d$, and d is the dimension of the feature space.

Attention modules [30] are permutation-invariant by construction and thus do not consider the ordering of the input frames of the sequence. However, the temporal coherence in a video is essential, specially in the context of video summarization. To alleviate this issue, we augment the frames' features with position encoding components. Due to the fact that fixed encoding adds no new parameters to learn, it is preferable to use it over a learned one in our problem. In this work, we use sine and cosine functions of different frequencies as used in [30].

As a result, the positional encoding and the visual features have the same dimension d, allowing the two to be added. The result of this step are the embeddings that are fed to the multi-head attention module:

$$E = \{E_t\}_{t=1}^{t=L} = X + PE \tag{3}$$

The goal of the Contextualized Network En_c (multi-head module) is to discover diverse representations of the interdependence between the frames of the sequence using different heads. The architecture is the same as proposed in the original paper [30] as presented in Figure 1 (b). Queries, keys and values are all formed using the input embeddings E, and projected using h heads of simple attention layers. The concatenated attention values from all the attention heads are then fed to a linear layer to produce the contextualized vectors $Z = \{Z_t\}_{t=1}^{t=L}$. The resulting representations are given to the Scoring Network En_s that estimates the frames' importance in the sequence $\tilde{Y}$. Figure 1 (a) depicts the interconnection between the three networks. The mapping function $f_\theta(.)$ is a composition of the three networks and can be written as follows:

$$f_\theta(.) = En_s \circ En_c \circ En_f \tag{4}$$

At training time, the described framework is optimized using backpropagation and gradient descent. We use the mean square error between the predicted scores and the averaged ground-truth annotations. The loss function for a batch B of sequences can be written as follows:

$$\mathcal{L} = \frac{1}{B \times L} \sum_{i=1}^{i=B} \sum_{t=1}^{t=L} (\tilde{y_{i,t}} - \overline{y_{i,t}})^2 \tag{5}$$

In the inference phase, the list of frames of a given video is divided into multiple non-overlapping windows of size L, then the resulting windows are stacked into a single batch. The batch that results is fed to the model $f_\theta(.)$ to get a batch of highlight scores, which are then fattened into a single sequence. With this method, we can obtain the prediction for a given video in a single forward step, which cuts down on the inference time.

3.3. Multi-Annotation Attention Model

We propose to use the Expectation-Maximization (EM) algorithm to solve this problem: both estimating the gold ground-truth for training videos with multiple annotations (ie, optimize the model using all the annotations) and training the model to predict the highlight scores for unseen videos.

Following the same notation in the section 3.1, let $y_{i,t}^j$ the highlight score assigned to the t^{th} frame in the i^{th} video by the j^{th} annotator. In our modeling, the annotator gives a noisy version of the true value $y_{i,t}$. The true label is considered as a latent variable. To facilitate the calculation, we assume a Gaussian noise with mean $y_{i,t}$ and inverse-variance τ^j,

$$y_{i,t}^j \sim \mathcal{N}(y_{i,t}, 1/\tau^j)\,. \tag{6}$$

where the Gaussian distribution for a random variable z with a mean μ and a standard deviation σ is defined as follows:

$$\mathcal{N}_z(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(z-\mu)^2}{2\sigma^2}) \tag{7}$$

For simplicity, we assume that the unknown precision τ^j does not depend on the frame (t, i) and only depends on the annotator. This is a practical assumption, but it is not totally accurate as the annotators might behave differently according to the instance they are labeling.

On the other hand, the true annotation $y_{i,t}$ is assumed to be the output of our mapping function f_θ (which is supposed to be deterministic) with additive Gaussian noise, and can be written as follows:

$$y_{i,t} = f_\theta(F_i)[t] + \epsilon \tag{8}$$

The notation $[t]$ means that we take the t^{th} index in the output sequence of highlight scores to get the prediction for the t^{th} frame, and F_i is the list of frame for the video V_i: $F_i = \{F_{i,t}\}_{t=1}^{t=L}$. The noise ϵ is a zero-mean Gaussian variable with inverse-variance γ, then $y_{i,t}$ can be written as follows:

$$y_{i,t} \sim \mathcal{N}(f_\theta(F_i), 1/\gamma) \tag{9}$$

When the two equations of both the annotator (equation 6) and the model (equation 9) are combined, the following equation is obtained:

$$y_{i,t}^j \sim \mathcal{N}(f_\theta(F_i), 1/\lambda^j) \tag{10}$$

We also define λ^j as the grouping of the two precision terms (γ and τ) that indicates the precision of each annotator:

$$\frac{1}{\lambda^j} = \frac{1}{\gamma} + \frac{1}{\tau^j} \tag{11}$$

At this point, the learning problem consists of estimating the latent variables $\lambda = [\lambda_1, ..., \lambda_M]$ as well as the parameters θ of the neural network $f_\theta(.)$. This type of problems can be solved using the well-known algorithm Expectation-Maximization [24].

The likelihood of the parameters $\omega = \{\theta, \lambda\}$ given the dataset $\mathcal{D} = \{(V_i, Y_i)\}_{i=1}^{i=N}$ can be written as follows if the instances of $\mathcal{D}$ are independent:

$$P(\mathcal{D}|\{\theta, \lambda\}) = \prod_{i=1}^{i=N} P(Y_i|F_i, \omega) \tag{12}$$

As human-labelers annotate data independently, we will assume annotations $\{Y_i^j\}_{j=1}^{j=M}$ are independent conditional on the video V_i. So, equation 12 can be written as:

$$P(\mathcal{D}|\{\theta, \lambda\}) = \prod_{i=1}^{i=N} \prod_{j=1}^{j=M} P(Y_i^j|F_i, \omega) \tag{13}$$

In order to simplify the computation, we will make a rough assumption that the scores of different timestamps are independent. This allows the likelihood to be factored as a product of Gaussian distributions, and thus to be derived easily later. The likelihood is then written as:

$$P(\mathcal{D}|\omega) = \prod_{i=1}^{i=N} \prod_{j=1}^{j=M} \prod_{t=1}^{t=L} P(y_{i,t}^{j}|F_i, \omega) \tag{14}$$

Although the last assumption is approximate, the trained model shows an improved performance. Also, note that we still model the dependencies between the frames as the proposed model is sensitive to the temporal order.

Given our model, we estimate the set of all parameters $\omega = \{\theta, \lambda\}$ using the Maximum Likelihood Estimator (MLE) by maximizing the log-likelihood function. Equivalently:

$$\hat{\omega} = \{\hat{\theta}, \hat{\lambda}\} = arg \max_{\omega} P(\mathcal{D}|\omega) \tag{15}$$

Since we have missing variables λ, the optimization problem can be simplified when utilizing the Expectation-Maximization (EM) algorithm [24]. In the context of missing or latent data, the EM algorithm is an effective iterative algorithm for computing the maximum-likelihood solution. The Expectation (E) and Maximization (M) steps make up each iteration of the EM algorithm. The M-step involves maximizing a lower bound on the log-likelihood that is improved by the E-step with each iteration.

- **E-step**: The gradient of the log-likelihood is equated to zero to obtain the equations to update the parameters $\{\lambda\}$:

$$\frac{1}{\hat{\lambda}^j} = \frac{1}{N \times L} \sum_{i=1}^{i=N} \sum_{t=1}^{t=L} [y_{i,t}^{j} - f_\theta(F_i)[t]]^2 \tag{16}$$

  Given the samples of the dataset $\mathcal{D}$ and the current value of the model parameters θ, we update the precision parameters $\{\lambda\}$ using equation 16. For a smoother updating, the exponential moving average is also proposed to be used instead:

$$\hat{\lambda^j}_{it} = \alpha \times \hat{\lambda^j}_{new} + (1-\alpha) \times \hat{\lambda^j}_{it-1} \tag{17}$$

  where it is the iteration counter of the EM algorithm, and $\hat{\lambda^j}_{new}$ is computed using equation 16.

- **M-step**: Since, there is no closed-form solution for maximizing the log-likelihood with respect to the parameters θ of the neural network $f_\theta(.)$, gradient descent with backpropagation is used to minimize the negative log-likelihood.

  Based on current value of the parameters $\{\lambda\}$ and given the dataset $\mathcal{D}$, the parameters θ are updated as follows:

$$\nabla_\theta \propto \frac{\partial}{\partial_\theta} \sum_i \sum_j \sum_t \lambda^j (y_{i,t}^{j} - f_\theta(F_i)[t]])^2 \tag{18}$$

The two steps (E-step and M-step) are repeated with alteration until convergence as described in Algorithm 1.

Algorithm 1 Multi-Annotation Highlight Algorithm

```
Require: the dataset $\mathcal{D} = \{(V_i, Y_i)\}_{i=1}^{i=N}$
Require: the updating parameter $\alpha$
Require: max_iteration
Ensure: the final estimation of the model parameters $\{\theta\}$
  Initialize: the annotators' precision $\{\lambda\}$
  Initialize: the model parameters $\{\theta\}$
  while not convergence do
      E-step: estimate $\{\lambda\}$ using equations 16 and 17.
      M-step: Loop over the dataset for one epoch using the gradient of equation 18.
  end while
```

4. Experiments & Results

4.1. Datasets

TvSum TvSum dataset [27] contains 50 videos from Youtube that belong to 10 categories of the TRECVid MED task (5 videos per category). Each video is annotated by 20 human labelers. Annotations are in the form of frame-level importance ranging from 1 (not important) to 5 (very important).

SumMe This dataset [9] consists of 25 raw or minimally edited user videos covering holidays, events and sports. Their length ranges from about 1 to 6 minutes. Each video is annotated by 15 to 18 different people, where a total of 41 subjects have participated. The annotation is binary and indicates if a frame should be part of the summary or not.

4.2. Data preprocessing

RGB frames are extracted from videos at the rate of 2 FPS compared to the raw rate which varies between 25 to 30 FPS. Images are then resized to 224 × 224. For feature extraction, we chose to use the ViT-B/32 model [6] as our frame encoder. The model was trained using the CLIP (Contrastive Language-Image Pre-Training) [23] method which contrasts the representations of images with that of natural language sequences. It produces 512 dimensional features vector that describes the semantic content of the input image. The feature extraction is performed for all the frames of the videos of the dataset before the training as the frame encoder is frozen during the training.

In section 3.2, the sequence length of the model L was described as the number of frames in a single window. As a result, we divide the training videos into consecutive windows with or without overlapping to create windows of length L. The overlapping ratio is controlled by the hop size hyperparameter. This allows us to train the model using bigger batches.

On the other hand, the ground truth importance scores (the target) are standardized to have a zero mean and unit variance. Note that the statistics (mean and variance) are computed on the training part of each split and not globally to avoid any data leakage.

4.3. Issues in Current Evaluation Protocols

In this section, we aim to highlight several issues related to existing evaluation approaches for video summarization using TvSum and SumMe datasets as no standard benchmark is given. Examining the current evaluation of the proposed approaches revealed many problems that we explain in the following before outlining solutions. The next section 4.4 describes our proposed evaluation protocol.

4.3.1 Data Splits

The first issue is related to how to generate training and testing sets as no standard benchmark is given with both TVSum and SumeMe datasets. Different data splitting schemes were found in the literature ranging from 1 random split (80% for train, 20% for test), five random (5 Rand) splits (80% for train, 20% for test) and 5-fold cross-validation (5 FCV). This makes the benchmarking challenging as some videos will be never part of the test set depending on the splitting strategy. The list of videos to test on could be critical as videos are not equally difficult. Particularly, 5-fold cross validation provides five non-overlapping splits where videos are equally divided across splits without any repetition or exclusion.

4.3.2 Validation Set

Most existing approaches report using $(80\% - 20\%)$ proportions for training and testing sets respectively. However, we failed to find any information to the validation set. This raises the question of how they did hyperparameter tuning, stopping the training and choosing the best model. In [11], authors plotted the performance on the test splits during the training epochs and used the resulting plot to validate the model selection criterion. In this circumstance, the test splits are playing the role of both validation and test data, which results in an unfair evaluation.

4.3.3 Experiment Repeats

Neural networks are trained using stochastic gradient descent where the convergence is non-deterministic process that depends on the initial weights. Moreover, the size of the public dataset is small (50 videos for TVSum and 25 videos for SumMe). This causes the final performance to be quite sensitive to the random initialization. This phenomena is particularly observed with SumMe dataset. It is uncertain if researchers report their best run or a random one. Then, it would be preferable to perform multiple runs of the same experiment and report the average performance.

4.3.4 Model Size

Most developed approaches in the literature used pre-trained models to extract one or many features: visual, motion, optical flow, audio and text. Additionally, Some recent methods used a text generator to produce a sentence description of the frames [18]. Those modules are generally not trained and thus the number of learnable parameters includes only that of the frame scoring system as done in [4]. However, the comparison of different summarization techniques in terms of the model size should also includes the scale of non-learnable parameters. The reason is that for a new unseen video, all the modules (both trainable and non-trainable) are used to perform the inference.

4.3.5 Metric Evaluation

Two main evaluation families have been proposed and used to assess the performance of video summarization systems: F1-score and rank-based metrics. As videos have multiple ground truth annotations, the proposed metrics are computed against each reference annotator, and then aggregated (mean for TVSum and max for SumMe) to get the performance on a single video. The overall performance is obtained by averaging over all the test videos.

F1-score To determine how well a summarization system based on the F1-score performs, both a pre-processing step and a post-processing step are required. The pre-processing step consists in temporal segmentation, where a video is divided into consecutive fragments, then fragment-level highlight score is calculated by averaging the predicted scores of the frames within each fragment. The machine summary is produced during the post-processing stage based on a shot selection algorithm (such as the Knapsack algorithm [1]), with a restriction that it should not exceed 15% of the video's duration. The similarity between the machine generated summary S_M and the user summary S_U is computed based on their temporal overlap ($\cap$) to obtain the F1-score. The precision (P), the recall (R) and the F1-score are computed as follows, where $\|\|$ is the temporal duration:

$$P = \frac{S_M \cap S_U}{\| S_M \|}, R = \frac{S_M \cap S_U}{\| S_U \|}, F1 = 2 \times \frac{P \times R}{P + R} \quad (19)$$

However, many concerns could be raised regarding the use of this metric to benchmark the proposed systems. First, the performance of any proposed video summarization system is highly dependent of not only the efficiency of the trained model, but also on both the temporal segmentation algorithm and the shot selection algorithm. As described in [19], changing only the type of the segmentation algorithm changes drastically this metric. Actually, the Kernel Temporal Segmentation (KTS) algorithm [22] utilizes the representations of the frames to estimate the fragments' boundaries based on the correlation matrix. Then, it is enough to use a *different feature extractor* to have different boundaries, and thus change the machine summary which alters the performance measure.

Second, the computation described in equation 19 is suitable for the videos of SumMe dataset, as the provided annotations are in the form of key-fragments. However, for TVSum dataset, a processing step is required before applying them, in which ground truth frame-level annotations are converted to key-fragments as described in [27, 33]. Surprisingly, this processing step involves the utilization of a temporal segmentation algorithm and thus a set of temporal boundaries as well as a shot selection technique. Consequently, it is quite likely that the *ground truth user summaries* will vary depending on the method.

Third, Otani et al. [19] explained that randomly produced summaries were able to get comparable or even higher performance scores than the most advanced techniques. The authors showed that the obtained F1-score was primarily determined by the video segmentation algorithm (KTS), specifically the distribution of segment lengths. This was mostly caused by the frequently employed shot selection method (Knapsack algorithm). The metric computation typically gave almost no consideration to the importance scores' contribution.

Rank-based Metrics Motivated by the facts above, rank-based metrics [19] have been used to assess the performance the proposed methods. The advantage of this evaluation method is that it eliminates the use the pre-processing and post-processing steps explained earlier. It is no longer impacted by the utilized video segmentation and shot selection algorithms. Kendall's (τ) [12] and Spearman's (ρ) [37] rank correlation coefficients are used to measure the similarities directly between the human importance scores and the predicted scores by the trained model at the frame level.

4.4. Proposed Evaluation Protocol and Implementation Details

In order to ensure the robustness and the generalization of the proposed models, we evaluate the model on all the videos not only some them as discussed earlier in section 4.3.1. We use 5-fold cross-validation technique as the evaluation method for all our proposed models. At each cross-validation split, 80% of the available data is involved in the training process, while the remaining 20% is used for testing. Following the discussion in section 4.3.2, 90% of the data associated to the training data in assigned to the training set, while 10% is assigned to the validation set. We use the validation set to monitor the evaluation metrics as the criteria for early stopping, where after each training epoch, those metrics are calculated with respects to the validation data to check if the network starts to overfit the training set. Model checkpointing is also used to save the best model during training. Each experiment is repeated 10 times and the average performance is reported (section 4.3.3).

The Contextualized Network (E_c) is formed by a multi-head attention module whose the number of heads is 16, and the latent dimension is 1024. The Scoring Network (E_s) is composed of a dropout layer followed by a linear layer that maps the sequence of representations into a sequence of importance scores. The number of trainable parameters is $2.1M$, while the number of non-trainable parameters is $86M$.

Models are trained for 50 epochs using the Adam optimizer using an initial learning rate of 6×10^{-5} and a weight decay of 5×10^{-5}. Experiments were performed on a single NVIDIA V100 (32GB) GPU using a batch size of 128.

4.5. Performance Comparisons

Method	**Kendall** (τ)	**Spearman** (ρ)	Data Splits
Random	0.000	0.000	-
Human	0.177	0.204	-
vsLSTM [33]	0.042	0.055	5 Rand
GLRPE [11]	0.070	0.091	5 Rand
RSGN [34]	0.083	0.090	5 Rand
SumGraph [21]	0.094	0.138	5 Rand
PGL-SUM [3]	0.157	0.206	5 Rand
Clip-it [18]	0.108	0.147	5 Rand
SSPVS [15]	0.177	0.233	-
SSPVS + Text [15]	0.181	0.238	-
MSVA [8]	0.190	0.210	5 FCV
MFST [20]	**0.222**	0.224	5 Rand
AAAM	0.193	0.254	5 FCV
MAAM	0.207	**0.271**	5 FCV

Table 1. Experimental results (Spearman's (ρ) and Kendall's (τ) coefficients) on TVSum. Results are computed using 5-fold cross-validation (5 FCV) and averaged over 10 runs.

4.5.1 Results on TVSum

We train the two proposed frameworks AAAM and MAAM as described in section 4.4. We compare them to the state of the art supervised methods using the correlation coefficients namely: Spearman's (ρ) and Kendall's (τ). Results are presented in Table 1. The first row indicates the performance for a random summary. The second row shows the performance for human annotations. These numbers are computed following the method described in [19].

We find that both proposed methods (AAAM and MAAM) outperform by far the existing methods in terms of the Spearman's coefficient. For the Kendall's metric, our MAAM framework is the second-best performing method among numerous supervised summarization methods. The best performing one [20] combines visual, text and audio modalities with a pretrained model for each modality. The MFST model [20] has many more parameters than our model that relies only on visual features with a little drop in the performance. Also, according to [20], this method has been evaluated using 5 random splits whereas our method is evaluated using 5-fold cross-validation. Moreover, a significant performance improvement is provided by the MAAM compared to the AAAM model.

4.5.2 Results on SumMe

Experiments are done likewise for SumMe dataset. Results are presented in Table 2. The same tendency has been observed for this dataset.

Method	**Kendall** (τ)	**Spearman** (ρ)	Data Splits
Random	0.000	0.000	-
Human	0.205	0.213	-
RSGN [34]	0.083	0.085	5 Rand
SSPVS [15]	0.178	0.240	-
SSPVS+Text [15]	0.192	0.257	-
MSVA [8]	0.200	0.230	5 FCV
MFST [20]	**0.229**	0.229	5 Rand
AAAM	0.223	0.273	5 FCV
MAAM	0.227	**0.278**	5 FCV

Table 2. Experimental results (Spearman's (ρ) and Kendall's (τ) coefficients) on SumMe. Results are computed using 5-fold cross-validation and averaged over 10 runs.

4.6. Ablation Study

4.6.1 Impact of the sequence length

This study investigates the impact of the sequence length L on the performance of the trained model. As described in section 4.2, training videos are divided into multiple windows of size L. The study includes experimenting with the following values for $L = \{32, 64, 128\}$, as well as training with full length videos (ie, without any windowing and a batch size equal to 1). The study is performed using the MAAM framework. Evaluation protocol is the same described in section 4.4 and results are presented in Table 3. We find that increasing the sequence length tends to decrease the performance even so the performance for 32 and 64 are comparable to each other. A significant drop in performance is observed when training the model with full-length sequences. The model is then required to adjust its parameters to different temporal dynamics.

Sequence length	**Kendall** (τ)	**Spearman** (ρ)
32	0.204	0.267
64	0.202	0.265
128	0.188	0.247
full length	0.163	0.215

Table 3. Ablation study on sequence length using MAAM framework applied to TVSum dataset. Spearman's (ρ) and Kendall's (τ) coefficients are computed using 5-fold cross validation and averaged over 10 runs. The full length setting indicates using full-length video sequences (ie, batch size equal to 1 video)

4.6.2 Impact of the visual features

The proposed models utilize visual features extracted using the ViT model as explained in section 4.2. The question of whether the obtained performance is only due to the quality of the features may be raised. For instance, we trained both AAAM and MAAM frameworks with visual features obtained by taking the output of the pool5 layer of GoogleNet [28] as used by many previous works [3, 33, 34]. The proposed methods outperform the best performing method [3] that uses the same features among the existing approaches.

Model	**Kendall** (τ)	**Spearman** (ρ)
vsLSTM [33]	0.042	0.055
GLRPE [11]	0.070	0.091
RSGN [34]	0.083	0.090
SumGraph [21]	0.094	0.138
PGL-SUM [3]	0.157	0.206
AAAM	0.169	0.223
MAAM	0.179	0.236

Table 4. Evaluation performance of AAAM and MAAM frameworks on TVSum using GoogleNet features. Spearman's (ρ) and Kendall's (τ) coefficients are computed using 5-fold cross validation and averaged over 10 runs

5. Conclusion

This paper introduces a high-performing supervised learning approach for video summarization based on an end-to-end probabilistic framework and attention mechanism. The developed AAAM framework integrate a number of multi-head attention augmented with the temporal encoding of the position of the frames in the video. We have also demonstrated that merging the annotations aggregation step with the model training provides better performance. Reliable evaluation protocol on both datasets TVSum and SumMe showed that our methods are the best performing compared to state of the art methods. For future work, we plan to apply self-supervised learning methods to pretrain the contextualized network.

References

[1] *The Knapsack Problem*, pages 439–448. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. 6

[2] Evlampios Apostolidis, Eleni Adamantidou, Alexandros I. Metsai, Vasileios Mezaris, and Ioannis Patras. Video Summarization Using Deep Neural Networks: A Survey. *Proceedings of the IEEE*, 109(11):1838–1863, Nov. 2021. Conference Name: Proceedings of the IEEE. 1, 2

[3] Evlampios Apostolidis, Georgios Balaouras, Vasileios Mezaris, and Ioannis Patras. Combining Global and Local Attention with Positional Encoding for Video Summarization. In *2021 IEEE International Symposium on Multimedia (ISM)*, pages 226–234, Nov. 2021. 2, 7, 8

[4] Evlampios Apostolidis, Georgios Balaouras, Vasileios Mezaris, and Ioannis Patras. Summarizing videos using concentrated attention and considering the uniqueness and diversity of the video frames. In *Proceedings of the 2022 International Conference on Multimedia Retrieval*, ICMR '22, page 407–415, New York, NY, USA, 2022. Association for Computing Machinery. 6

[5] Luis Casas and Eugenia Koblents. *Video Summarization with LSTM and Deep Attention Models: 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8–11, 2019, Proceedings, Part II*, pages 67–79. 01 2019. 2

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. 5

[7] Jiri Fajtl, Hajar Sadeghi Sokeh, Vasileios Argyriou, Dorothy Monekosso, and Paolo Remagnino. Summarizing videos with attention. *CoRR*, abs/1812.01969, 2018. 1, 2

[8] Junaid Ahmed Ghauri, Sherzod Hakimov, and Ralph Ewerth. Supervised Video Summarization Via Multiple Feature Sets with Parallel Attention. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6s, July 2021. ISSN: 1945-788X. 1, 2, 7, 8

[9] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating Summaries from User Videos. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, Lecture Notes in Computer Science, pages 505–520, Cham, 2014. Springer International Publishing. 1, 5

[10] Zhong Ji, Kailin Xiong, Yanwei Pang, and Xuelong Li. Video summarization with attention-based encoder-decoder networks. *CoRR*, abs/1708.09545, 2017. 2

[11] Yunjae Jung, Donghyeon Cho, Sanghyun Woo, and In So Kweon. Global-and-local relative position embedding for unsupervised video summarization. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV*, page 167–183, Berlin, Heidelberg, 2020. Springer-Verlag. 6, 7, 8

[12] M. G. Kendall. The treatment of ties in ranking problems. *Biometrika*, 33(3):239–251, 1945. 7

[13] Shamit Lal, Shivam Duggal, and Indu Sreedevi. Online video summarization: Predicting future to better summarize present. *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 471–480, 2019. 2

[14] Khiem H. Le, Tuan V. Tran, Hieu H. Pham, Hieu T. Nguyen, Tung T. Le, and Ha Q. Nguyen. Learning from multiple expert annotators for enhancing anomaly detection in medical image analysis, 2022. 2

[15] Haopeng Li, Qiuhong Ke, Mingming Gong, and Rui Zhang. Video summarization based on video-text modelling. *CoRR*, abs/2201.02494, 2022. 2, 7, 8

[16] Ping Li, Qinghao Ye, Luming Zhang, Li Yuan, Xianghua Xu, and Ling Shao. Exploring global diverse attention via pairwise temporal relation for video summarization. *CoRR*, abs/2009.10942, 2020. 1, 2

[17] Yen-Ting Liu, Yu-Jhe Li, Fu-En Yang, Shang-Fu Chen, and Yu-Chiang Frank Wang. Learning hierarchical self-attention for video summarization. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 3377–3381, 2019. 1, 2

[18] Medhini Narasimhan, Anna Rohrbach, and Trevor Darrell. CLIP-It! Language-Guided Video Summarization. In *Advances in Neural Information Processing Systems*, volume 34, pages 13988–14000. Curran Associates, Inc., 2021. 2, 6, 7

[19] Mayu Otani, Yuta Nakashima, Esa Rahtu, and Janne Heikkila. Rethinking the Evaluation of Video Summaries. pages 7596–7604, 2019. 7

[20] Jeiyoon Park, Kiho Kwoun, Chanhee Lee, and Heuiseok Lim. Multimodal frame-scoring transformer for video summarization, 2022. 7, 8

[21] Jungin Park, Jiyoung Lee, Ig-Jae Kim, and Kwanghoon Sohn. Sumgraph: Video summarization via recursive graph modeling. In *ECCV*, 2020. 7, 8

[22] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. Category-specific video summarization. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *ECCV - European Conference on Computer Vision*, volume 8694 of *Lecture Notes in Computer Science*, pages 540–555, Zurich, Switzerland, Sept. 2014. Springer. 7

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. 5

[24] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Anna Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: Whom to trust when everyone lies a bit. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 889–896, New York, NY, USA, 2009. Association for Computing Machinery. 1, 2, 4, 5

[25] Vikas C. Raykar, Shipeng Yu, Linda H. Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11(43):1297–1322, 2010. 1, 2

[26] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. Learning from multiple annotators: Distinguishing good

from random labelers. *Pattern Recognition Letters*, 34(12):1428–1436, 2013. 1, 2

[27] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. TVSum: Summarizing web videos using titles. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5179–5187, June 2015. ISSN: 1063-6919. 1, 5, 7

[28] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 8

[29] Brian Vander Schee. Crowdsourcing: Why the power of the crowd is driving the future of business [book review] 2009 jeff howe. new york, ny: Crown business. *Journal of Consumer Marketing*, 26:305–306, 06 2009. 2

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. 3, 4

[31] Yan Yan, Romer Rosales, Glenn Fung, Mark Schmidt, Gerardo Hermosillo, Luca Bogoni, Linda Moy, and Jennifer Dy. Modeling annotator expertise: Learning when everybody knows a bit of something. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 932–939, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. 1, 2

[32] Yuan Yuan, Haopeng Li, and Qi Wang. Spatiotemporal Modeling for Video Summarization Using Convolutional Recurrent Neural Network. *IEEE Access*, 7:64676–64685, 2019. Conference Name: IEEE Access. 2

[33] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. Video Summarization with Long Short-Term Memory. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 766–782, Cham, 2016. Springer International Publishing. 1, 2, 7, 8

[34] Bin Zhao, Haopeng Li, Xiaoqiang Lu, and Xuelong Li. Reconstructive sequence-graph network for video summarization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 7, 8

[35] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. Hsa-rnn: Hierarchical structure-adaptive rnn for video summarization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7405–7414, 2018. 1, 2

[36] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. Hierarchical recurrent neural network for video summarization. *CoRR*, abs/1904.12251, 2019. 1, 2

[37] Daniel Zwillinger and Stephen Kokoska. *CRC Standard Probability and Statistics Tables and Formulae*. CRC Press, Dec. 1999. 7