A Survey on Deep Models for Video Summarization using Robust Metrics

Martin Saint-Jalmes
Georgia Institute of Technology
martin.saint-jalmes@gatech.edu

Alexis Navarian
Georgia Institute of Technology
anavarian3@gatech.edu

Sylvain Marchienne
Georgia Institute of Technology
sylvain.marchienne@gatech.edu

Abstract

Video Summarization consists in producing clips of a footage that highlights its main content. This paper focuses on providing a reliable benchmark framework for this challenging task. The complexity of this task stems from its subjectivity, which makes it both difficult for machine-generated summaries to achieve good results, but also to find how to measure these results in the first place. Researchers in the field have built numerous models inspired from novel concepts and paradigms in the broader Deep Learning community. Concurrently, there has been interest and theoretical research in producing better-suited metrics to evaluate fairly these models.

In this work, we propose a novel generic framework to benchmark models from different paradigms, using a common set of multi-purpose metrics that measure performance on both well-established and more recent and specialized datasets. We introduce a preprocessed dataset focused on esports lives, in contrast to the more general-purpose reference datasets. Finally, we use this framework to measure the impact of our own improvements on these models. Our experiments confirm that the traditional metrics used in the past were biased. Using the better-suited metrics, we finally show that good quantitative and qualitative results are reachable using advanced Deep Learning models on the introduced dataset.

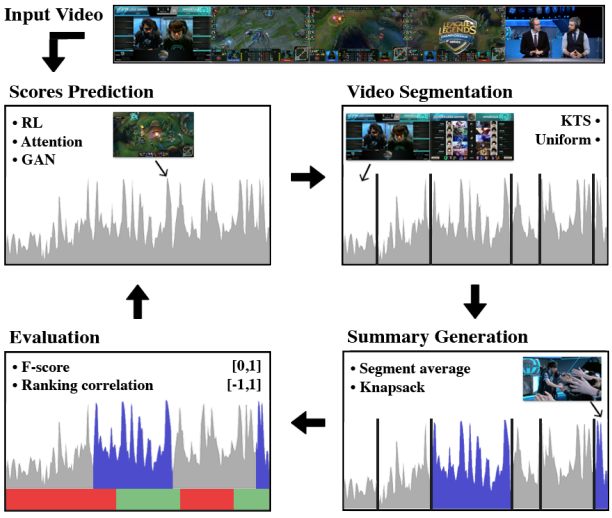


Figure 1: Overview of our video summarization learning pipeline. Using Deep Learning models, a score is predicted for each frame of the input video. Temporal segmentation is applied on the video and the most interesting segments are chosen for the summary using optimization techniques. Based on human annotations, the quality of the score predictions and the final summary are evaluated using robust metrics.

1. Introduction

The democratization and rapid growth of user-produced video content, including vlogs and game streaming, has introduced value in providing summaries of these videos. Shorter clips of such videos, especially regarding streaming content, allows to save storage, bandwidth and most importantly, view time. In the same way that human-crafted trailers can tease a movie, machine-generated video summaries could highlight the most important parts of a footage, so that a viewer may quickly grasp the core of the video and decide whether they want to watch it in full.

Many works and approaches to Video Summarization have been explored over the past few years, including a reinforcement-based approach in [20], attention-based [3], unsupervised adversarial LSTM-based [10] and many

others [13] [7] [5] [18] [4] [14] [19]. Despite this variety of works and ideas, the task of Video Summarization remains a complex one, and these models have yet to reach human performance. Furthermore, some other notable papers like [11] have demonstrated that the most commonly used metrics to assess models' efficiency aren't fully unbiased, reliable, or suitable. While some alternative metrics have been proposed by [11] to address these concerns, they haven't been used to assess the performances of most of recent works and models.

Hence, our main work is to propose a benchmarking environment and generic model evaluation framework, illustrated in Figure 1, that assesses the performance of established and new models on multiple Video Summarization datasets. Some are already routinely used for evaluation, like TVSum [15] or SumMe [7], but we include a more recent dataset, Twitch-LOL [4]. In order to evaluate performance, we rely on traditional evaluation metrics used in Video Summarization as well as the new ones proposed in [11] to address the originals' shortcomings.

Our objective in this work was to implement and select few models that achieved state-of-the-art results from [20], [3] and [10] in our model framework. Once implemented, these could be evaluated fairly on multiple datasets and normalized criteria. Our final task was adding possible improvements on these models, some of which suggested by the original authors.

There is significant value to producing predictive models that are able to generate video summaries. Many applications can be envisioned, including producing automated highlights trailers in the entertainment industry. We hope that providing a reliable framework to evaluate models will allow to hint towards the concepts and architectures that are well suited for the summarization task and provide insights for future research directions.

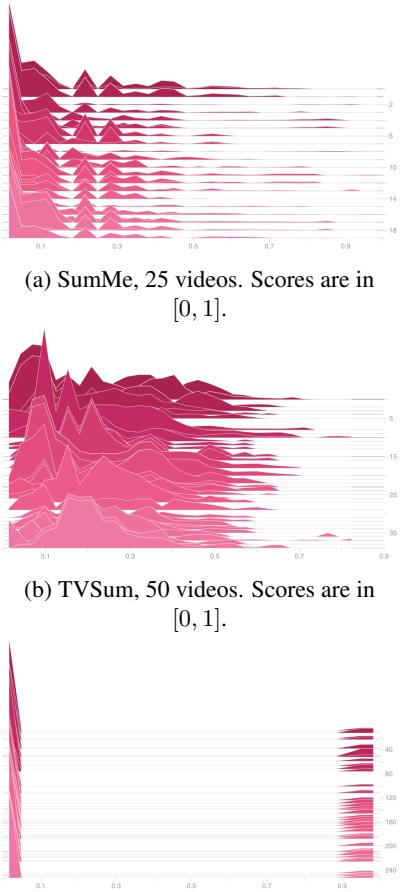
2. Benchmarking environment

2.1. Framework

With this paper we introduce Summarizer[1], a Video Summarization framework for research. Most of the literature now focuses on deep learning models experimenting on a set of reference datasets. We gathered the key assets to ease this research into a single Python framework.

The four main components are as follows: (1) centralized, preprocessed and documented datasets files, (2) open source PyTorch [12] implementations of the prominent deep learning models, (3) a robust set of evaluation metrics for fair quantitative comparison between models and datasets, (4) summary generation tools for qualitative evaluation of performances.

[1] https://github.com/sylvainma/Summarizer



(a) SumMe, 25 videos. Scores are in $[0, 1]$.

(b) TVSum, 50 videos. Scores are in $[0, 1]$.

(c) Twitch-LOL, 318 videos. Scores are either 0 or 1.

Figure 2: Datasets ground truth scores distribution per video. Each ridge represents the distribution of scores for a video in the dataset.

The implementations are either improved ones from the authors, or our own. The experiments shown in this paper were conducted with them. We hope that this framework will help to design the next generation of deep learning models for Video Summarization.

2.2. Datasets

The most popular datasets used in Video Summarization are SumMe [7] and TVSum [15]. These datasets contain a set of videos along with multiple human annotated reference summaries for each original video. We use preprocessed files in HDF5 format generated from these datasets to accelerate models training. We also generated cross-validation splits defining the training and testing

videos for a total of five folds. While previous authors had generated similar 5-fold cross-validation splits, they weren't shared across works. We chose not to reuse any of them, as to not favor a specific model. We release those new splits as part of our framework for future research and comparison of models. The splits files and datasets presented in this section are publicly available and documented in the code repository.

SumMe was released by [7] as a Video Summarization benchmark dataset, publicly available, human annotated, for frame importance score prediction. It contains a set of 25 general-purpose public videos from YouTube. Each video has 15 to 18 human annotations. They were asked to create a summary having the most important content with a limited time length. Thus, the annotations are 0/1 values for each frame of the videos indicating if a particular frame was picked by the annotator to be a part of his summary. As a target for the models, importance frame scores are derived from the proportion of annotators who picked the frames. The distribution of frame scores per video for SumMe is shown in Figure 2a. Most of the scores are between 0.0 and 0.5.

TVSum was later introduced by [15] directly for Video Summarization models validation. It contains 50 general-purpose videos from YouTube as well, annotated by 20 humans. In this dataset, the annotations were designed differently: annotators were asked to give a score between 1 (uninteresting) and 5 (interesting) to short video segments of two seconds. The importance frame scores to be predicted are derived by setting the score to all frame within the two seconds segment. They are also rescaled in $[0, 1]$ using min-max normalization. As depicted in Figure 2b, the scores are more scattered compared to SumMe.

The third dataset we use is Twitch-LOL, adapted from [4]. This dataset contains 318 videos corresponding to as many matches of the popular video game League of Legends (LOL). They are all issued from professional esports tournaments, specifically of the spring 2017 North American Championships and Master Series (Taiwan, Hong Kong, and Macau). Many of this dataset's characteristics differ from SumMe and TVSum. As full match retransmissions, the videos' length is comparatively longer (30 minutes to an hour), and the content is more homogeneous across videos.

By nature, this dataset is more specialized, which offers reference grounds if a video summarizer would be used for a specific sports or entertainment domain. In the specific case of LOL, we have high hopes that this dataset is somewhat representative of the much larger set (by several orders of magnitude) of all LOL game retransmissions. The original authors published this dataset as raw videos alongside annotation files with a single binary (0/1) ground truth sequence, stating if a particular frame was included in human-generated highlights and commentary videos on a YouTube channel. They used template matching to link summaries and original matches, and report good results. Out of the 330 original videos, we discarded those with uninformative (e.g. only zeros) or missing annotations and trimmed excess labels (when they exceeded the number of frames of the video). The distribution of frames found in the summaries for the remaining videos is shown in Figure 2c.

We adapted the same processing steps as the other datasets underwent: we generated a descriptive file in HDF5 format holding feature representations (extracted from the conv5 layer of GoogLeNet [16]) of every 15^{th} frame of each video. However, contrarily to the other datasets, we didn't use Kernel Temporal Segmentation (KTS) [13] to find optimal segments, as it has shown to be computationally intractable: this process took 6 hours and 120 GB of RAM for a single video. Therefore, as per [11], we applied uniform segmentation to retrieve parts of two seconds per segment.

2.3. Summary generation

The usual approach in video summarization is to decide on parts to include in a summary, not at the frame level, but at the segment level. This is motivated both by practical reasons (to save computations) and empirical observations.

Kernel Temporal Segmentation (KTS) [13] is a method designed to find optimal non-overlapping segments from a video, given a descriptive sequence of feature representations. It minimizes a loss measuring within-segment variances, mitigated by a penalty term on their number. While this algorithm has been used extensively for this task, it does have some limits and biases, as we detail in the next subsection 2.4. As this has been the method used for the SumMe and TVSum datasets, we kept it in this work.

An alternative for producing segments is uniform sampling. This fast partitioning method consists in splitting the video into same-length segments of a constant duration. This method was used for the Twitch-LOL dataset.

Common videos are shot at a rate of 24 to 60 frames per second, and this high rate leads to consecutive frames being very similar semantically. Therefore the inclusion of consecutive frames is more likely than not. To address this, the usual approach is to sample a couple of keyshot frames per second, acting as representatives of their temporal neighborhood.

A model assigns importance scores to every keyshot of a video. We run a Knapsack-solving algorithm to determine whether to include each sequence $s_i \in \mathcal{S}$ by summing the importance scores $h(\cdot)$ of the keyshots k it contains. Additionally, there is a constraint given by a maximum summary duration L, usually determined relative to the original video (we use 15% the of original lengths). This optimization problem is defined as follows:

$$\max \sum_{i=1}^{|\mathcal{S}|} b_i \sum_{k \in s_i} h(k) \quad \text{s.t.} \quad \sum_{i=1}^{|\mathcal{S}|} b_i |s_i| \leq L \qquad (1)$$

where $b_i \in \{0, 1\}$, indicating whether a sequence s_i of length $|s_i|$ should be included.

2.4. Metrics

The basic evaluation approach is to measure the agreement between the generated summary and the reference summaries using an F1-score. Most of recent papers on video summarization use F1-score metric to assess their model's performances on TVSum and SumMe datasets.

Nevertheless, the validity of reference summary-based evaluation and F1-score have been questioned because of [11]. The authors have shown that random scoring and summary generation methods achieve performances as high as most of state-of-the-art models on TVSum and SumMe using the F1-score metric. They have also demonstrated that F1-score is mostly determined by the distribution of video segment lengths. In order to avoid undesirable results where no learning model is useful, uniform segmentation should be preferred instead of KTS when using the F1-score metric.

As an alternative, the authors have also suggested new metrics to assess models performances: Spearman's correlation and Kendall's correlation. Both are evaluating the importance rankings using correlation between the predicted ordering and the ordering by human annotators. Hence, in our work we decided to implement and test our models using F1-score (for comparison with previous papers) and Spearman's correlation. We chose Spearman over Kendall for our experiments as it usually yields larger values than Kendall, which gives more flexibility for comparing models performances. Still, we leave users of our framework the choice of using Kendall's correlation if they desire.

3. Models

3.1. VASNet

Our first model is VASNet, an architecture developed by [3] relying on dot-product attention mechanisms, first introduced in [17]. Specifically, it uses what the authors describe as soft self-attention, warranted by the use of true probabilities as output instead of hard labels. The use of self-attention allows to consider a scoring mechanism for each frame that is a function of the others. Contrarily to the Transformer architecture developed in [17], the authors of VASNet use a single layer of encoder followed by fully connected feedforward layers, and do not rely on decoders at all. This is justified by the fact that we wish to output scores directly, not a sequence of feature representations analogous to the input.

VASNet fits into the supervised learning category, relying on a mean squared error loss computed between predicted probabilities and normalized ground truth scores from the highlights. Using any optimizer, the goal is to learn the parameters associated to the self-attention and feedforward layers to minimize this loss.

In this work, we bring several modifications to VASNet, in hopes of improving some of its characteristics. For clarity, we name the resulting model VASNet$_{\text{loc}}$. The first of these modifications is to consider local self-attention rather than global. This was actually suggested in the original paper, but they didn't implement the idea. In concrete terms, considering a frame $t \in \{1, \ldots, n\}$ and an attention aperture w, before applying softmax to self-attention logits $\{e_{t,i}\}_{i \in \{1,\ldots,n\}}$, we set:

$$e_{t,j} = -\infty \quad \forall j \in \{1, \ldots, n\} \backslash \{i - w, \ldots, i + w\} \qquad (2)$$

allowing us to focus self-attention score computation in a small window of neighboring frames.

We also add the option to include positional encodings. They can be attention-based as implemented in [17], representing positions in a periodical manner using sines or cosines. Since this addition was memory-intensive (requiring a matrix allocation in the order of the square of the number of frames in the videos), we implemented a simpler alternative to the attention-based encodings relying on creating embeddings of the position of the frames, from 1 to the length of the video. The choice of the positional encoding is left to the user as a model hyperparameter.

Aside from these two features, we also replaced VASNet's Xavier weights initialization [6] to the more recent Kaiming initialization [8] as this has proven to be somewhat effective in other works to tackle the problems of vanishing and exploding gradients.

In VASNet, the authors use a particular scaling factor for numerical stability in the self-attention matrix product. This factor of 0.06 has been determined empirically. We propose to replace it by the one used in [17]'s dot-product

attention, which is more generic in that it is a function of the feature vector dimension, rather than a constant.

Finally, as another alternative model, we implement an architecture based on the Transformer's encoder. It is conceptually very similar to VASNet, relying on soft self-attention. We bring very little modifications to this architecture, other than introducing an additional residual connection between before and after the encoder layers. We hope to test this model in close comparison to VASNet and $\text{VASNet}_{\text{loc}}$, while tuning hyperparameters linked to the number of attention heads and encoder layers.

3.2. SumGAN

As an unsupervised model, [10] is an architecture of combined LSTMs using adversarial training. There is an LSTM selector, a VAE LSTM generator and an LSTM discriminator. The selector inputs the videos and its role is to predict frame importance scores. It is trained in an adversarial manner. The generator is a VAE which inputs the output of the selector and reconstructs the original video. The discriminator is presented original videos and reconstructed ones. Its goal is to classify videos as either original or reconstructed. GAN optimization encourages the selector model to assign higher scores to important frames of the original video for which reconstruction is better.

The main challenge of SumGAN, as in most GANs, is to reach convergence. To address this issue, it has a supervised extension which maximizes the likelihood of the scores predicted by the selector given ground truth scores. We are able to use this version as the datasets we experiment are labeled. The supervised version is denoted hereafter as $\text{SumGAN}_{\text{sup}}$. It should be noted that, during the prediction phase, only the selector is used since we rely on the GAN itself just for training.

There are more issues aside from GAN convergence. Notably, the large number of (sequential) LSTMs in SumGAN makes it suboptimal to train on a GPU as apposed to other parallelizable models. We came up with two improvement suggestions. First, we optimized the GAN using the Wasserstein loss (WGAN) [2] and we added noise to the real and fake videos before being passed to the discriminator. Both are tricks supposed to stabilize the GAN optimization. Secondly, we replaced the selector LSTM by a Transformer's encoder and the VAE [9] LSTM by a full encoder-decoder Transformer [17]. By incorporating Transformer architectures in our model, we hope to leverage self-attention, which has previously shown promising results. They are also parallelizable and thus can speedup the training. We denote this new alternative model as $\text{SumGAN}_{\text{att}}$.

3.3. DSN

The Deep Summarization Network (DSN) model inspired from [20] is viewing Video Summarization as a sequential decision-making process. The model aims to predict a probability for each frame to be included in the summary and then decide on "actions" based on the obtained probability distribution to create the video summary.

DSN is designed as a reinforcement learning based mode which uses a newly created reward that accounts for diversity and representativeness of generated summaries. During the training step, the Deep Summarization Network aims to get higher rewards which would lead to more diverse and representative summaries.

Since labels are not required, this model can indeed be classified as an unsupervised learning model and previous works [20] have shown that this model outperforms other well-known unsupervised models and is comparable to state-of-the-art supervised models. However, this was only shown on the TVSum and SumMe datasets, using the F1-score metric.

An improvement idea to the Deep Summarization Network could be to use another algorithm than the commonly used Policy Optimization for the reinforcement learning part. For instance, we could use other simple reinforcement learning algorithms like Q-Learning or more complex ones like TRPO or A3C if it isn't considered as adding too much complexity for the results we could get. This improvements haven't been implemented yet and in our experimentations we will only work with the supervised and unsupervised versions of the classic Deep Summarization Network.

3.4. Logistic Regression

All previous models are computationally expensive and complex in terms of capacity. For comparison, we integrated in our experiment a simple logistic regression model which computes an importance score independently for each frame of an input video. This is a supervised model with the ground truth scores being required to maximize their predicted log-likelihood.

Such a model has low capacity but is very fast, especially on GPU as the computation is parallelizable. It is therefore a good and computationally cheap comparison to put other models in perspective.

3.5. Baselines

As baseline comparison, we define random and human performances as follows. They serve as expected lower and

	TVSum			SumMe			Twitch-LOL	
	Avg F1	Max F1	Avg Cor.	Avg F1	Max F1	Avg Cor.	F1	Cor.
DSN	0.570	0.801	0.156	**0.232**	**0.494**	0.057	0.154	0.086
DSN_{sup}	0.587	0.817	0.248	0.230	0.473	0.069	0.556	0.413
VASNet	0.582	0.803	0.233	0.229	0.479	0.112	0.568	0.402
$VASNet_{loc}$	0.580	0.812	**0.246**	0.231	0.483	**0.147**	**0.642**	**0.453**
Transformer Encoder	0.576	0.805	0.233	0.225	0.483	0.116	0.596	0.424
SumGAN	0.557	0.781	-0.019	0.230	0.467	0.058	0.331	0.264
$SumGAN_{sup}$	0.587	0.809	0.226	0.230	0.486	0.073	0.428	0.329
$SumGAN_{att}$	0.589	**0.827**	0.198	0.229	0.454	0.090	0.565	0.407
Logistic Regression	**0.592**	0.817	0.232	0.228	0.487	0.109	0.482	0.355
Random	0.546	0.743	0.002	0.215	0.450	0.003	0.125	0.003
Human	0.538	0.775	0.204	0.311	0.543	0.329	—	—

Table 1: F1-score and correlation measures for the various models and datasets we tested. The Twitch-LOL dataset having only one ground truth sequence, no per-user aggregation of these measures is possible, and the leave-one-out human evaluation is not defined.

high bounds respectively.

We set up a "random" model predicting random scores to each input frame of the video. There is no weight optimization but the F1-score and correlation metrics are computed as usual using these uniformly distributed scores.

The "human" performances are computed following a (reversed) leave-one-out strategy inspired from [11]. This is only defined on the datasets where we have multiple annotators expressing their interest in putting particular frames in their summary. We successively pick one of the annotators and consider their labels to be predictions that we want to evaluate. We compute the metrics on the ground truths that are established to be the labels from all other annotators except the one being tested. These scores are then averaged over all individuals, giving us an estimate of how a human would perform with regards to this task. We remind that as Twitch-LOL only has a unique sequence of ground truth annotations, this leave-one-out procedure cannot be applied to this dataset.

4. Experiments

4.1. Experimental Setup

In video summarization, training duration is always a concern, especially for datasets with long videos like Twitch-LOL. Hence, we tried to mitigate the computations needed as much as possible. As KTS requires a significant amount of time and memory, we reused the preprocessed files in HDF5 format for the TVSum and SumMe datasets. There was significant compute associated with building the HDF5 file for Twitch-LOL, requiring multiple days to build the feature representations of the 318 videos.

We leveraged the capabilities of GPUs in building the deep learning models themselves, and distributed the training load across multiple machines, as much as the limits in dedicated Video RAM (VRAM) allowed. We trained most of our algorithms using 8 GPU-enabled (Nvidia P100) instances on Google Colaboratory, 2 instances on Google Cloud Platform (Nvidia T4), and one local computer (Nvidia GTX1060). Depending on the dataset, the trained model and its exact hyperparameters, performing training on a model took from 15 minutes to 24 hours.

We kept track of all our experiments by adapting our framework to use TensorBoard [1]. This allowed us to review past runs, compare models and hyperparamters, as well as to monitor and visualize our metrics in real time during all training processes. All training logs also appear in the associated log folders, ensuring us reliable bookkeeping, even when computations were distributed across multiple locations.

[2]The generated summary can be accessed at https://youtu.be/IGE3bpoNHLw

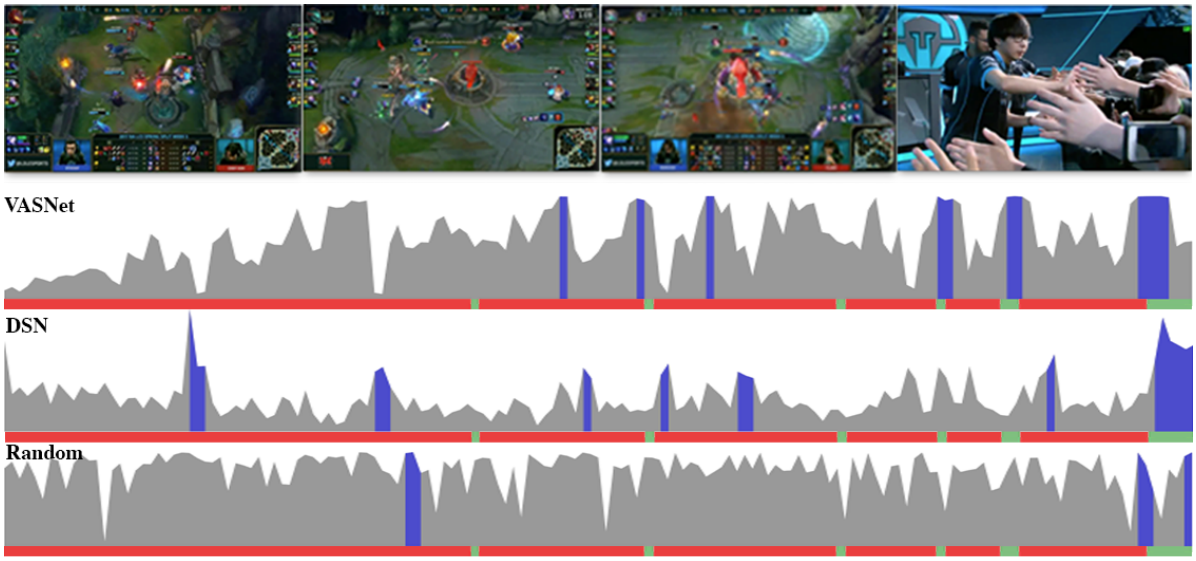


Figure 3: Different scores and summaries on the same video generated by the best model (VASNet$_{\text{loc}}$), the weakest (DSN) and the random one. Frame scores (logarithmic scale) are shown in gray. Predicted summaries are shown in blue. The ground truth summary is underlined in green & the rest in red. Frames (top) are from the summary[2]generated by VASNet$_{\text{loc}}$.

4.2. Quantitative results

As the shown in Table 1, the two Deep Summarization Network models achieved somewhat similar results as those described in [20], at least in terms of F1-score and on the TVSum and SumMe datasets. Nevertheless, the use of DSN as one of the reference models in video summarization could be questioned given its comparatively poorer performance in terms of Spearman correlation on the same datasets. We assume that these observations are linked to the $L_{\text{percentage}}$ loss described in [20] and a non-optimal tuning of the ϵ-parameter, since the DSN model is trying to produce scores close to ϵ. Therefore, some additional and more specific work on the matter could be envisioned to confirm or reject this hypothesis.

The group of attention-based supervised methods perform comparatively well on all three datasets across all metrics. This seems to confirm experiments from [3] who presented state-of-the-art results, besting SumGAN and DSN. With the newly generated cross-validation split however, we notice that VASNet doesn't excel in all situations, with poorer F1-scores on the SumMe dataset than DSN. Our transformer encoder, with tuned hyperparameters, has shown to be competitive with VASNet, especially on the Twitch-LOL dataset. Contrarily to initial hypotheses, more capacity hasn't translated into better performance for Transformer encoders: for example, the best results were attained for 2 layers of encoders and 2 attention heads on Twitch-LOL. Finally, our optimized model, VASNet$_{\text{loc}}$ has shown very promising results on all datasets, with the best correlation values. It also is the best-performing model in terms of F1-score on Twitch-LOL. While we presented the results under the same model, there are differences with regards to the exact best-performing hyperparameters for VASNet$_{\text{loc}}$ on each dataset. In detail, a 30-second aperture of local attention worked best for Twitch-LOL, but positional encodings worsened performance (-0.05 in correlation). It was also the case that the model didn't benefit from positional encodings when trained on SumMe, while attention-based positional encodings significantly helped the model predictions on TVSum (+0.1 in correlation).

Overall, the three different SumGAN models did not obtain good performances on the datasets. As mentioned earlier, GANs are hard to train. In SumGAN and SumGAN$_{\text{sup}}$, there is a clear capacity imbalance between the generator and the discriminator which might be explained by the very high number of model parameters (around 350 million). We also noticed that the convergence of the GAN was not directly correlated with an increase in performance metrics. SumGAN$_{\text{att}}$ experienced the same issues, but the replacement of LSTMs by Transformer layers led to an increase of speed when trained on GPU, as expected. However the GAN convergence stability attempt was not successful. Moreover, just as for DSN, we noticed that the sparsity loss in the original SumGAN model [10] pushes the predicted scores to a fixed ϵ on average. This is not the intended behavior explained by the authors but it sometimes leads the predictions far from the ground truth.

Student Name	Contributed Aspects	Details
Martin Saint-Jalmes	VASNet Twitch-LOL processing	VASNet implementation, training, improvements and analyses. Twitch-LOL dataset HDF5 generation and processing.
Alexis Navarian	DSN Report & results structure	DSN implementation, training, improvements and analyses. Led the structuring bases for the report
Sylvain Marchienne	SumGAN Framework & visualizations	SumGAN implementation, training, improvements and analyses. Structuring bases for the framework + report visualizations

Table 2: Contributions of team members.

On the other hand, as shown in [11], a completely random model also performs well in terms of F1-score on the two datasets TVsum and SumMe. Hence, [11]'s assumptions are once more verified. Furthermore, we can also add that the Logistic Regression model used as a baseline is performing quite well in terms of correlation since its performances are around 90% of those obtained with other models on the three datasets.

4.3. Qualitative results

We perform a qualitative analysis of our results by visually inspecting some of the summaries suggested by our models. As our models reason in terms of assigning scores to a few keyshots of the video, there are some steps needed to relate the output of our deep learning models to the initial task at hand, that of generating summary videos.

For any given video, we first use the method described in subsection 2.3, solving a knapsack problem to determine which segments to include in the summary, based on the highlight scores. We then leverage FFmpeg to generate frames from the original video. Finally, given a set of segments boundaries to include, we concatenate all frames that we chose among the set of stills, and output a single video of our predicted highlights back-to-back.

Figure 3 represents the output scores obtained by DSN, VASNet and Random models on a video sampled from the Twitch-LOL dataset. As we can see, while the Random model expectedly predicted completely random results, VASNet and DSN models both have quite interesting scores in the sense that they both have picked frames in the middle and at the end of the video file. In League of Legends high level games, the first minutes are often the same and most of the events occur at the middle or at the end of the match as the ground truth supports this. Furthermore, we can also notice that VASNet seems to be more precise than the DSN model in that it did predict high scores for frames labeled as important by the ground truth. In the other hand, DSN didn't really predict high scores which fit with the ground truth except at the end of the video.

5. Conclusion

To sum-up, we implemented a framework to assess Video Summarization models' performances. The implemented framework is also well equipped to handle every step of the Video Summarization pipeline since methods to generate preprocessed dataset files and final video summaries are also provided. The current state of the framework contains the models described above as well as their improvements ideas described in section 3 and those improvements were successful for the most part.

Furthermore, we performed a benchmark that shows the F1-score's inefficiency as a metric when combined with Kernel Temporal Segmentation. The results of the Deep Summarization Network model, which is referenced as the most effective unsupervised Video Summarization model, were effectively comparable to those obtained with the random model according to the F1-score on TVSum and SumMe datasets.

On the other hand, the experiments led on Twitch-LOL dataset raised a few interesting observations. The major one is that Video Summarization Models seem to work best on specialized datasets and less on general-purpose ones. Further assessment with human qualitative evaluation would be a next step. We think that fot this particular dataset, it wouldn't be too difficult to collect more examples (from other game seasons) and more ground truths (from different commentary channels). The improvements tied to such additions would make for an interesting research direction.

To the best of our knowledge, we are the first to present a large benchmark ranking Video Summarization models with robust metrics on these three datasets. We hope that our work will set a new basis for further research on the topic.

6. Work Division

There was a fair and equal contribution from the three team members. Exact details are listed in Table 2.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. 3 2016. 6

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *34th International Conference on Machine Learning, ICML 2017*, 1 2017. 5

[3] Jiri Fajtl, Hajar Sadeghi Sokeh, Vasileios Argyriou, Dorothy Monekosso, and Paolo Remagnino. Summarizing Videos with Attention. *Asian Conference on Computer Vision*, 11367 LNCS:39–54, 12 2018. 1, 2, 4, 7

[4] Cheng Yang Fu, Joon Lee, Mohit Bansal, and Alexander C. Berg. Video highlight prediction using audience chat reactions. In *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 972–978, 7 2017. 2, 3

[5] Tsu Jui Fu, Shao Heng Tai, and Hwann Tzong Chen. Attentive and adversarial learning for video summarization. In *Proceedings - 2019 IEEE Winter Conference on Applications of Computer Vision, WACV 2019*, pages 1579–1587. Institute of Electrical and Electronics Engineers Inc., 3 2019. 2

[6] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Journal of Machine Learning Research*, volume 9, pages 249–256, 2010. 4

[7] 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, *ECCV - European Conference on Computer Vision*, volume 8695 LNCS, pages 505–520. Springer International Publishing, Cham, 2014. 2, 3

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 4

[9] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 2014. 5

[10] Behrooz Mahasseni, Michael Lam, and Sinisa Todorovic. Unsupervised video summarization with adversarial LSTM networks. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-January, pages 2982–2991. Institute of Electrical and Electronics Engineers Inc., 11 2017. 1, 2, 5, 7

[11] Mayu Otani, Yuta Nakashima, Esa Rahtu, and Janne Heikkila. Rethinking the evaluation of video summaries. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 7588–7596. IEEE Computer Society, 3 2019. 2, 3, 4, 6, 8

[12] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, pages 8024–8035, 12 2019. 2

[13] 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*, pages 540–555, Cham, 2014. Springer International Publishing. 2, 3

[14] Mrigank Rochan and Yang Wang. Video Summarization by Learning from Unpaired Data. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June:7894–7903, 5 2018. 2

[15] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. TVSum: Summarizing web videos using titles. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 5179–5187, 2015. 2, 3

[16] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 1–9. IEEE Computer Society, 10 2015. 3

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *Advances in Neural Information Processing Systems*, 2017-Decem(Nips):5999–6009, 6 2017. 4, 5

[18] Ting Yao, Tao Mei, and Yong Rui. Highlight detection with pairwise deep ranking for first-person video summarization. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-December, pages 982–990. IEEE Computer Society, 12 2016. 2

[19] Ke Zhang, Wei Lun Chao, Fei Sha, and Kristen Grauman. Video summarization with long short-term memory. In *ECCV - European Conference on Computer Vision*, volume 9911 LNCS, pages 766–782. Springer Verlag, 5 2016. 2

[20] Kaiyang Zhou, Yu Qiao, and Tao Xiang. Deep Reinforcement Learning for Unsupervised Video Summarization with Diversity-Representativeness Reward. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 7582–7589, 12 2017. 1, 2, 5, 7