

Summarizing Videos with Attention^{*}

Jiri Fajtl¹, Hajar Sadeghi Sokeh¹, Vasileios Argyriou¹, Dorothy Monekosso²,
and Paolo Remagnino¹

¹ Robot Vision Team RoVit, Kingston University, London, UK

² Leeds Beckett University, Leeds, UK

Abstract. In this work we propose a novel method for supervised, keyshots based video summarization by applying a conceptually simple and computationally efficient soft, self-attention mechanism. Current state of the art methods leverage bi-directional recurrent networks such as BiLSTM combined with attention. These networks are complex to implement and computationally demanding compared to fully connected networks. To that end we propose a simple, self-attention based network for video summarization which performs the entire sequence to sequence transformation in a single feed forward pass and single backward pass during training. Our method sets a new state of the art results on two benchmarks TvSum and SumMe, commonly used in this domain.

Keywords: video summarization · self-attention · sequence to sequence

1 Introduction

Personal videos, video lectures, video diaries, video messages on social networks and videos in many other domains are becoming to dominate other forms of information exchange. According to Cisco Visual Networking Index: Forecast and Methodology, 2016-2021³, by 2019 video will account for 80% of all global Internet traffic, excluding P2P channels. Consequently, better methods for video management, such as video summarization, are needed.

Video summarization is a task where a video sequence is reduced to a small number of still images called keyframes, sometimes called storyboard or thumbnails extraction, or a shorter video sequence composed of keyshots, also called video skim or dynamic summaries. The keyframes or keyshots need to convey most of key information contained in the original video. This task is similar to a lossy video compression, where the building block is a video frame. In this paper we focus solely on the keyshots based video summarization.

^{*} This research was funded by the H2020 MONICA European project 732350 and by the NATO within the WITNESS project under grant agreement number G5437 and within the MIDAS G5381. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

³ <https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html>

Video summarization is an inherently difficult task even for us people. In order to identify the most important segments one needs to view the entire video content and then make the selection, subject to the desired summary length. Naturally, one could define the keyshots as segments that carry mutually diverse information while also being highly representative of the video source. There are methods that formulate the summarization task as a clustering with cost functions based on exactly these criteria. Unfortunately, to define how well chosen keyshots represent the video source as well as the diversity between them is extremely difficult since this needs to reflect the information level perceived by the user. Common techniques analyze motion features, measure the distance between color histograms, image entropy or in the 2/3D CNN feature space [25,18,1,2], reflecting semantic similarities. However, none of these approaches can truly capture the information in the video context. We believe that to automatically generate high quality summaries, similar to what we are capable of, a machine should learn from us humans by means of a behavioral cloning or supervision.

Early video summarization methods were based on unsupervised methods, leveraging low level spatio-temporal features and dimensionality reduction with clustering techniques. Success of these methods solely stands on the ability to define distance/cost functions between the keyshots/frames with respect to the original video. As discussed above, this is very difficult to achieve as well as it introduces a strong bias in the summarization given by the type of used features such as semantic and pixel intensities. In contrast, models trained with supervision learn the transformation that produces summaries similar to those manually produced. Currently, there are two datasets with such annotations, TvSum [32] and SumMe [12], where each video is annotated by 15-20 users. The annotations vary between users with consistency expressed by a pairwise F-score ~ 0.34 . This fact reveals that the video annotation is a rather subjective task. We argue that under these circumstances it may be extremely difficult to craft a metric that would accurately express how to cluster video frames into keyshots, similar to human annotation. On this premise, we decided to adopt the supervised video summarization for our work.

Current state of the art methods for video summarization are based on recurrent encoder-decoder architectures, usually with bi-directional LSTM [14] or GRU [6] and soft attention [4]. While these models are remarkably powerful in many domains, such as machine translation and image/video captioning, they are computationally demanding, especially in the bi-directional configuration. Recently A. Vaswani et al. [34] demonstrated that it is possible to perform sequence to sequence transformation only with the attention. Along similar lines, we propose a pure attention, sequence to sequence network VASNet for video keyshots summarization and demonstrate its performance on TvSum and SumMe benchmarks. Architecture of this model does not employ recurrent or sequential processing and can be implemented with conventional matrix/vector operations and run in a single forward/backward pass during inference/training, even for sequences with variable length. The architecture is centered around two

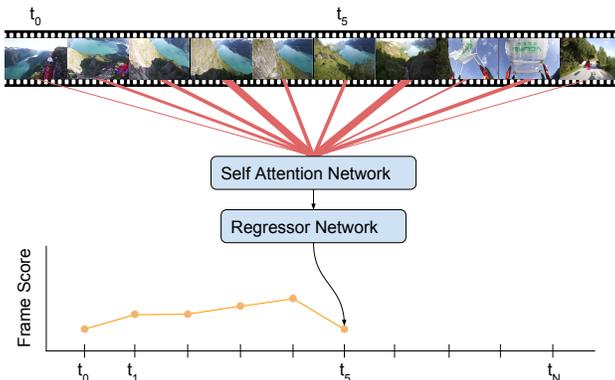


Fig. 1: For each output the self-attention network generates weights for all input features. Average of the input features, weighted by this attention, is regressed by a fully connected neural network to the frame importance score.

key operations, attention weights calculation and frame level score regression. An overview of this model is shown in Fig. 1. Frame score at every step t is estimated from a weighted average of all input features. The weights are calculated with the self-attention algorithm. Given the generic architecture of our model we believe that it could be successfully used in other domains requiring sequence to sequence transformation. Our contributions are:

1. A novel approach to sequence to sequence transformation for video summarization based on soft, self-attention mechanism. In contrast, current state of the art relies on complex LSTM/GRU encoder-decoder methods.
2. A demonstration that a recurrent network can be successfully replaced with simpler, attention mechanism for the video summarization.

2 Related Work

Recent advancements in deep learning were rapidly adapted by researches focusing on video summarization, particularly encoder-decoder networks with attention for sequence to sequence transformation. In this section we will discuss several existing methods related to our work.

K. Zhang et al. [40] pioneered the application of LSTM for supervised video summarization to model the variable-range temporal dependency among video frames to derive both representative and compact video summaries. They enhance the strength of the LSTM with the determinantal point process which is a probabilistic model for diverse subset selection. Another sequence to sequence method for supervised video summarization was introduced by Ji et al. [15]. Their deep attention-based framework uses a bi-directional LSTM to encode the contextual information among input video frames. Mahasseni et al. [23] propose an adversarial network to summarize the video by minimizing the distance

between the video and its summary. They predict video keyframes distribution with a sequential generative adversarial network. A deep summarization network in an encoder-decoder architecture via an end-to-end reinforcement learning has been proposed by Zhou et al. [42] to achieve state of the art results in unsupervised video summarization. They design a novel reward function that jointly takes diversity and representativeness of generated summaries into account. A hierarchical LSTM is constructed to deal with the long temporal dependencies among video frames by [41], but it fails to capture the video structure information, where the shots are generated by fixed length segmentation.

Some works use side semantic information associated with a video along with visual features, like surrounding text such as titles, queries, descriptions, comments, unpaired training data and so on. Rochan et al. in [29], proposed deep learning video summaries from unpaired training data, which means they learn from available videos summaries without their corresponding raw input videos. Yuan et al. [39], proposed a deep side semantic embedding model which uses both side semantic information and visual content in the video. Similarly H. Wei et al. [35] propose a supervised, deep learning method trained with manually created text descriptions as ground truth. At the heart of this method is the LSTM encode-decoder network. Wei achieves competitive results with this approach, however, more complex labels are required for the training. Fei et al. [9] complemented visual features with video frame memorability, predicted by a separate model such as [16] or [8].

Other approaches, like the one described in [31], use an unsupervised method by clustering some features extracted from the video, delete the similar frames, and select the rest of the frames as keyframe of the video. In fact, they used a hierarchical clustering method to generate a weight map from the frame similarity graph in which the clusters can easily be inferred. Another clustering method is proposed by Otani et al. [26], in which they use deep video features to encode various levels of content including objects, actions, and scenes. They extract the deep features from each segment of the original video and apply a clustering-based summarization technique on them.

2.1 Attention Techniques

The fundamental concept of attention mechanism for neural networks was laid by Bahdanau et al. [4] for the task of machine translation. This attention is based on an idea that the neural network can learn how important various samples in a sequence, or image regions, are with respect to the desired output state. These importance values are defined as attention weights and are commonly estimated simultaneously with other model parameters trained for a specific objective. There are two main distinct attention algorithms, hard and soft.

Hard attention produces a binary attention mask, thus making a 'hard' decision on which samples to consider. This technique was successfully used by K. Xu et al. [37] for image caption generation. Hard attention models use stochastic sampling during the training; consequently, backpropagation cannot be employed due to the non-differentiable nature of the stochastic processes. REINFORCE

learning rule [36] is regularly used to train such models. This task is similar to learning an attention policy introduced by V. Mnih et al. [24].

In this work we exclusively focus on soft attention. In contrast to the hard attention, soft attention generates weights as true probabilities. These weights are calculated in a deterministic fashion using a process that is differentiable. This means that we can use backpropagation and train the entire model end-to-end. Along with the LSTM, soft attention is currently employed in the majority of sequence to sequence models used in machine translation [22], image/video caption generation [37],[38], addressing neural memory [11] and other. Soft attention weights are usually calculated as a function of the input features and the current encoder or decoder state. The attention is global if at each step t all input features are considered or local where the attention has access to only limited number of local neighbors.

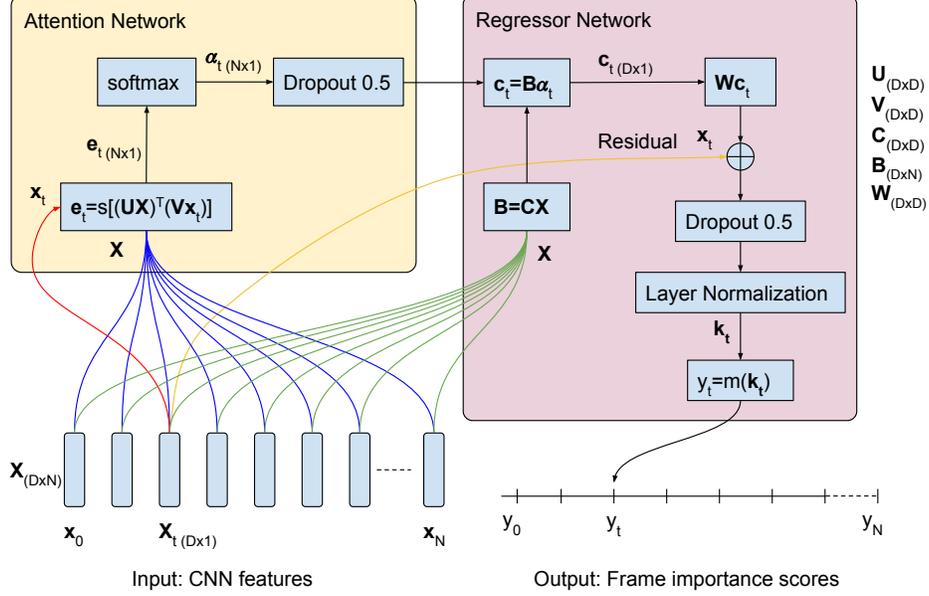
If the attention model does not consider the decoder state, the model is called self-attention or intra-attention. In this case the attention reflects the relation of an input sample t with respect to other input samples given the optimization objective. Self-attention models were successfully used in tasks such as reading comprehension, summarization and in general for task-independent sequence representations [5][27][20]. The self-attention is easy and fast to calculate with matrix multiplication in a single pass for entire sequence since at each step we do not need the result of past state.

3 Model Architecture

Common approach to supervised video summarization and other sequence to sequence transformations, is an application of a LSTM or GRU encoder-decoder network with attention. Forward LSTM is usually replaced with bi-directional BiLSTM since keyshots in the summary have relation to future video frames in the sequence. Unlike the RNN based networks, our method does not need to reach for special techniques, such as BiLSTM, to achieve non-causal behavior. The vanilla attention model has equal access to all past and future inputs. This aperture can be, however, easily modified and it can even be asymmetric, dilated, or exclude the current time step t .

The hidden state passed from encoder to decoder has always fixed length, however, it needs to encode information representing sequences with variable lengths. This means that there is a higher information loss for longer sequences. The proposed attention mechanism does not suffer from such loss since it accesses the input sequence directly without an intermediate embedding.

Architecture proposed in this work replaces entirely the LSTM encoder-decoder network with the soft, self-attention and a two layer, fully connected network for regression of the frame importance score. Our model takes an input sequence $\mathbf{X} = (\mathbf{x}_0, \dots, \mathbf{x}_N)$, $\mathbf{x} \in \mathbb{R}^D$ and produces an output sequence $\mathbf{Y} = (y_0, \dots, y_N)$, $y = [0, 1)$, both of length N . The input is a sequences of CNN feature vectors with dimensions D , extracted for each video frame. Fig. 2 shows the entire network in detail.

Fig. 2: Diagram of VASNet network attending sample \mathbf{x}_t .

Unnormalized self-attention weight $e_{t,i}$ is calculated as an alignment between input feature \mathbf{x}_t and the entire input sequence according to Luong et al. [21].

$$e_{t,i} = s[(\mathbf{U}\mathbf{x}_i)^T(\mathbf{V}\mathbf{x}_t)] \quad t = [0, N), \quad i = [0, N) \quad (1)$$

Here, N is the number of video frames, \mathbf{U} and \mathbf{V} are network weight matrices estimated together with other parameters of the network during optimization and s is a scale parameter that reduces values of the dot product between $\mathbf{U}\mathbf{x}_i$ and $\mathbf{V}\mathbf{x}_t$. We set the scale s to value 0.06, determined experimentally. Impact of the scale on the model performance was, however, minimal. Alternatively, the attention vector could be also realized by an additive function as shown by Bahdanou et al. [4].

$$e_{t,i} = \mathbf{M} \tanh(\mathbf{U}\mathbf{x}_i + \mathbf{V}\mathbf{x}_t) \quad (2)$$

where \mathbf{M} are additional network weights learned during training. Both formulas have shown similar performance, however, the multiplicative attention is easier to parallelise since it can be entirely implemented as a matrix multiplication which can be highly optimized. The attention vector \mathbf{e}_t is then converted to the attention weights α_t with softmax.

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{k=1}^N \exp(e_{t,k})} \quad (3)$$

The attention weights α_t are true probabilities representing the importance of input features with respect to the desired frame level score at the time t . Linear

transformation \mathbf{C} is then applied to each input and the results then weighted with attention vector α_t and averaged. The output is a context vector \mathbf{c}_t which is used for the final frame score regression.

$$\mathbf{b}_i = \mathbf{C}\mathbf{x}_i \quad (4)$$

$$\mathbf{c}_t = \sum_{i=1}^N \alpha_{t,i} \mathbf{b}_i \quad \mathbf{c}_t \in \mathbb{R}^D \quad (5)$$

The context vector \mathbf{c}_t is then projected by a single layer, fully connected network with linear activation and residual sum followed by dropout and layer normalization.

$$\mathbf{k}_t = \text{norm}(\text{dropout}(\mathbf{W}\mathbf{c}_t + \mathbf{x}_t)) \quad (6)$$

The \mathbf{C} and \mathbf{W} are network weight matrices learned during the network training. To regularize the network we also add a dropout for attention weights as shown in Fig. 2. We found it to be beneficial, especially for small training datasets such as in the canonical setting for TvSum (40 videos) and SumMe (20 videos).

By design, the attention network discards the temporal order in the sequence. This is due to the fact that the context vector \mathbf{c}_t is calculated as a weighted average of input features without any order information. The order of the output sequence is still preserved. The positional order for the frame score prediction is not important in the video summarization task, as has been shown in the past work utilizing clustering techniques that also discard the input frame order. For other tasks, such as machine translation or captioning, the order is essential. In these cases every prediction at time t , including attention weights, could be conditioned on state at $t - 1$. Alternatively, a positional encoding could be injected to the input as proposed by [34],[10].

Finally, a two layer neural network performs the frame score regression $y_t = m(\mathbf{k}_t)$. First layer has a ReLU activation followed by dropout and layer normalization [3], while the second layer has a single hidden unit with sigmoid activation.

3.1 Frame Scores to Keyshot Summaries

The model outputs frame-level scores that are then converted to keyshots. Following [40], this is done in two steps. First, we detect scene change points where each represents a potential keyshot segment. Second, we select a subset of these keyshots by maximizing the total frame score within these keyshots while constraining the total summary length to 15% of the original video length as per [12]. The scene change points are detected by Kernel Temporal Segmentation (KTS) method [28] as shown in Fig. 3. For each detected shot $i \in K$ we calculate score s_i .

$$s_i = \frac{1}{l_i} \sum_{a=1}^{l_i} y_{i,a} \quad (7)$$

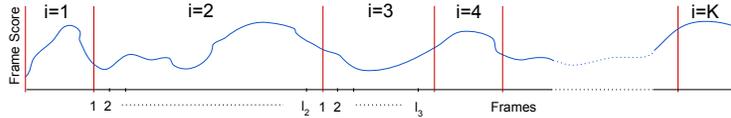


Fig. 3: Temporal segmentation with KTS.

where $y_{i,a}$ is score of a -th frame within shot i and l_i is the length of i -th shot. Keyshots are then selected with the Knapsack algorithm Eq. 8 according to [32].

$$\max \sum_{i=1}^K u_i s_i, \quad \text{s. t.} \quad \sum_{i=1}^K u_i l_i \leq L, u_i \in \{0, 1\} \quad (8)$$

Keyshots with $u_i = 1$ are then concatenated to produce the final video summary. For evaluation we create a binary summary vector where each frame in shot ($u_i = 1$) is set to one.

3.2 Model Training

To train our model we use the ADAM optimizer [17] with learning rate $5 \cdot 10^{-5}$. This low learning rate is used as a result of having a batch with single sample, where the sample is an entire video sequence. We use 50% dropout and $L2 = 10^{-5}$ regularization. Training is done over 200 epochs. Model with the highest validation F-score is then selected.

3.3 Computation Complexity

The self-attention requires a constant number of operations at each step for all input features N , each of size D . The complexity is thus $O(N^2D)$. The recurrent layer, on the other hand, requires $O(N)$ sequential operations, each of complexity $O(ND^2)$. Self-attention needs less computation when the sequence length N is shorter than the feature size D . For longer videos, a local attention would be used rather than the global one.

4 Evaluation

4.1 Datasets Overview

In order to directly compare our method with the previous work we conducted all experiments on four datasets, TvSum [32], SumMe [12], OVP [7] and YouTube [7]. OVP and YouTube were used only to augment the training dataset. TvSum and SumMe are currently the only datasets suitably labeled for keyshots video summarization, albeit still small for training deep models. Table 1 provides an overview of the main datasets properties.

Table 1: Overview of the TvSum and SumMe properties.

Dataset	Videos	User annotations	Annotation type	Video length (sec)		
				Min	Max	Avg
SumMe	25	15-18	keyshots	32	324	146
TvSum	50	20	frame-level importance scores	83	647	235
OVP	50	5	keyframes	46	209	98
YouTube	39	5	keyframes	9	572	196

The TvSum dataset is annotated by frame-level importance scores, while the SumMe with binary keyshot summaries. OVP and YouTube are annotated with keyframes and need to be converted to the frame-level scores and binary keyshot summaries, following the protocol discussed in the following section 4.2.

4.2 Ground Truth Preparation

Our model is trained using frame-level scores, while the evaluation is performed with the binary keyshot summaries. The SumMe dataset comes with keyshot annotations, as well as frame-level scores calculated as an average of the keyshot user summaries per frame. In the case of TvSum we convert the frame-level scores to keyshots following the protocol described in section 3.1. Keyframe annotations in OVP and YouTube are converted to frame-level scores by temporarily segmenting the video into shots with KTS and then selecting shots that contain the keyframes. Knapsack is then used to constrain the total summary length, however in this case the keyshot score s_i (Eq. 8) is calculated as a ratio of number of keyframes within the keyshot and the keyshot length.

To make the comparison even more direct, we adopt identical training and testing ground truth data used by [40], [42] and [23]. This represents CNN embeddings, scene change points, and generated frame-level scores and keyshot labels for all datasets. The preprocessed data are publicly available (K. Zhou et al. [42]⁴ and K Zhang et al.[40]⁵). CNN embeddings used in this preprocessed dataset have 1024 dimensions and were extracted from the pool5 layer of the GoogLeNet network [33] trained on ImageNet [30].

We use a 5-fold cross validation for both, canonical and augmented settings as suggested by [40]. In the canonical setting, we generate 5 random train/test splits for the TvSum and SumMe datasets individually. 80% samples are used for training and the rest for testing. In the augmented setting we also maintain the 5-fold cross validation with the 80/20 train/test, but add the other datasets to the training split. For example, to train the SumMe in the augmented setting

⁴ <http://www.eecs.qmul.ac.uk/~kz303/vsumm-reinforce/datasets.tar.gz>

⁵ <https://www.dropbox.com/s/yn14jsa2mxohs16/data.zip?dl=0>

we take all samples from TvSum, OVP and YouTube and 80% of the SumMe as the training dataset and the remaining 20% for evaluation.

4.3 Evaluation Protocol

To provide a fair comparison with the state of the art, we follow evaluation protocol from [40], [42] and [23]. To assess the similarity between the machine and user summaries we use the harmonic mean of precision and recall expressed as the F-score in percentages.

$$F = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \times 100 \quad (9)$$

True and false positives and false negatives for the F-score are calculated per-frame as the overlap between the ground truth and machine summaries, as shown in Fig. 4.

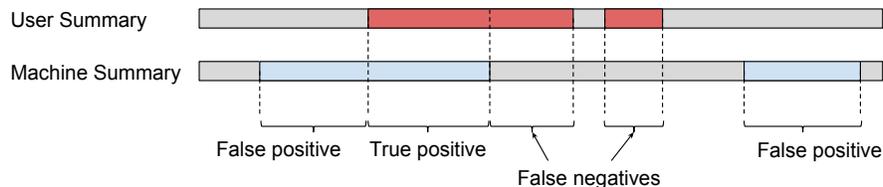


Fig. 4: True positives, False positives and False negatives are calculated per-frame between the ground truth and machine binary keyshot summaries.

Following [12], the machine summary is limited to 15% of the original video length and then evaluated against multiple user summaries according to [40]. Precisely, on the TvSum benchmark, for each video, the F-score is calculated as an average between the machine summary and each of the user summaries as suggested by [32]. Average F-score over videos in the dataset is then reported. On the SumMe benchmark, for each video, a user summary most similar to the machine summary is selected. This approach is proposed by [13] and also used in the work of Lin and Chin-Yew [19].

5 Experiments and Results

Results of the VASNet evaluation on TvSum and SumMe datasets, compared with the most recent state of the art methods are presented in Table 3. To illustrate how well the methods learned from the user annotations we show a human performance, which is calculated as pairwise F-scores between the ground truth and all user summaries. In Table 2 we also compare the human performance

Table 2: Average pairwise F-scores calculated among user summaries and between ground truth (GT) and users summaries.

Dataset	Pairwise F score	
	Among users annotations	Training GT w.r.t. users annotations (human performance)
SumMe	31.1	64.2
TvSum	53.8	63.7

with F-scores calculated among the user summaries themselves. We can see that the human performance is higher than the F-score among the user summaries which is likely caused by the fact that the training ground truth is calculated as an average of all user summaries and then converted to the keyshots, which are aligned on the scene change-points. These keyshots are likely to be longer than the discrete user summaries, thus having higher mutual overlap. The pairwise F-score 53.8 for TvSum dataset is higher than the F-score 36 reported by the authors [32]. This is because we convert each user summary to keyshots with KTS and limit the duration to 15% of the video length and then calculate the pairwise F-scores. Authors of the dataset [32] calculate the F-score from *gold standard labels*, that is, from keyshots of length 2 seconds, a length used by users during the frame-level score annotation. We chose to follow the former procedure which is maintained in all evaluations in this work to make the results directly comparable.

Table 3: Comparison of our method VASNet with the state of the art methods for canonical and augmented settings. For a reference we add human performance measured as pairwise F-score between training ground truth and user summaries.

Method	SumMe		TvSum	
	Canonical	Augmented	Canonical	Augmented
dppLSTM [40]	38.6	42.9	54.7	59.6
M-AVS [15]	44.4	46.1	61.0	61.8
DR-DSN _{sup} [42]	42.1	43.9	58.1	59.8
SUM-GAN _{sup} [23]	41.7	43.6	56.3	61.2
SASUM _{sup} [35]	45.3	-	58.2	-
Human	64.2	-	63.7	-
VASNet (proposed method)	49.71	51.09	61.42	62.37

In Table 3 we can see that our method outperforms all previous work in both canonical and augmented settings. On the TvSum benchmark the improvement is by 0.7% and 1% in the canonical and augmented settings respectively and

2% lower than the human performance. On the SumMe this is 12% and 11% in the canonical and augmented settings respectively and 21% below the human performance. In Fig. 5 we show this improvements visually.

The higher performance gain on the SumMe dataset is very likely caused by the fact that our attention model can extract more information from the ground truth compared to the TvSum, where most methods already closely approach the human performance. It is conceivable to assume that the small gain on the TvSum is caused by the negative effect of the global attention on long sequences. TvSum videos are comparatively longer than the SumMe as seen in Table 1. At

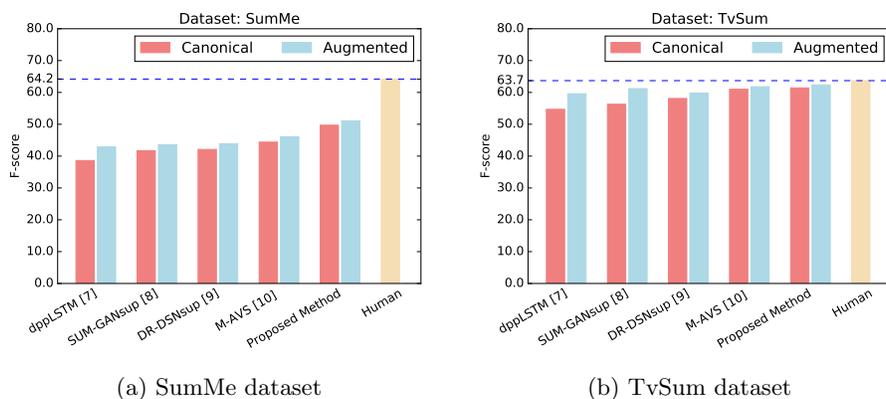


Fig. 5: VASNet performance gain compared to the state of the art and human performance.

every prediction step the global attention ‘looks’ at all video frames. For long video sequences frames from temporally distant scenes are likely less relevant than the local ones, but the global attention still needs to explore them. We believe that this increases variance in the attention weights, which negatively impacts the prediction accuracy. We hypothesize that this could be mitigated by the introduction of local attention.

5.1 Qualitative Results

To show the quality of the machine summaries produced by our method we plot the ground truth and predicted scores for two videos from TvSum in Fig. 6. We selected videos 10 and 11, since they are also used in previous work [42], thus enabling a direct comparison. We can see a clear correlation between the ground truth and machine summary, confirming the quality of our method. Original videos and their summaries are available on YouTube.⁶

⁶ <https://www.youtube.com/playlist?list=PLEdpjt8KmmQMfQEat4HvuIx0Rwi09q9DB>

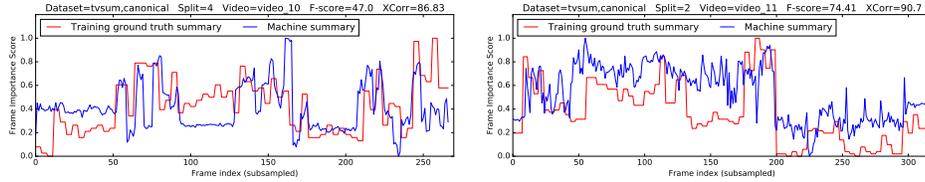


Fig. 6: Correlation between ground truth and machine summaries produced by VASNet for test videos 10 and 11 from TvSum dataset, also evaluated in [42].

We also compare the final, binary keyshot summary with the ground truth. In Fig. 7 we show machine generated keyshots in light blue color over the ground truth importance scores shown in gray. We can see that the selected keyshots align with most of the peaks in the ground truth and that they cover the entire length of the video. The confusion matrix in Fig. 8 shows attention weights

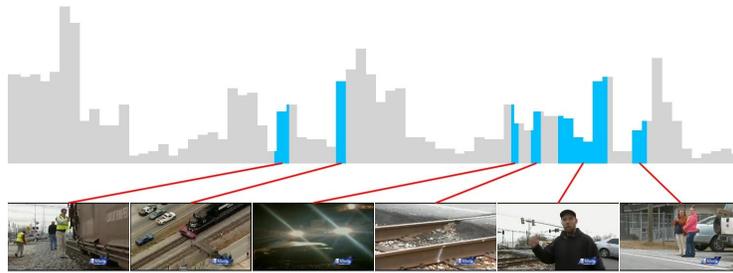


Fig. 7: Ground truth frame scores (gray), machine summary (blue) and corresponding keyframes for test video 7 from TvSum dataset.

produced during evaluation of TvSum video 7. We can see that the attention strongly focuses on frames either correlated with low frame scores (top and bottom image in Fig. 8, attention weights for frames ~ 80 and ~ 190) or high scores (second and third image, frames ~ 95 and ~ 150). It is conceivable to assume that the network learns to associate every video frame with other frames of similar score levels.

Another interesting observation to make is that the transitions between the high and low attention weights in the confusion matrix highly correlate with the scene change points, shown as green and red horizontal and vertical lines. It is important to note that the change points, detected with KTS algorithm, were not provided to the model during learning or inference, nor were used to process the training GT. Thus, we believe that this model could be also applied to scene segmentation, removing the need for the KTS post-processing step. We will explore this possibility in our future work.

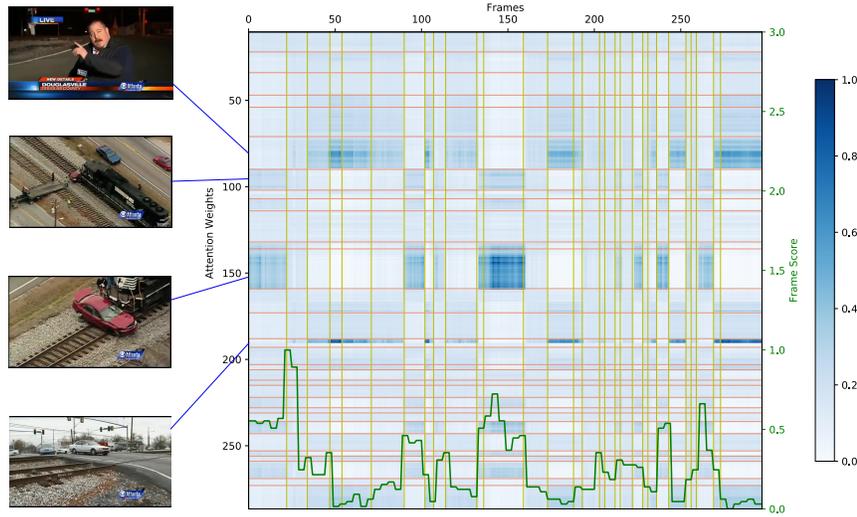


Fig. 8: Confusion matrix of attention weights for TvSum video 7 from test split 2. Green plot at the bottom shows the GT frame scores. Green and red horizontal and vertical lines show scene change points. Values were normalized to range 0-1 across the matrix. Frames are sub-sampled to 2fps.

6 Conclusions

In this work propose a novel deep neural network for keyshot video summarization based on pure soft, self-attention. This network performs a sequence to sequence transformation without recurrent networks such as LSTM based encoder-decoder models. We show that on the supervised, keyshot video summarization task our model outperforms the existing state of the art methods on the TvSum and SumMe benchmarks. Given the simplicity of our model it is easier to implement and less resource demanding to run than LSTM encoder-decoder based methods, making it suitable for application on embedded or low power platforms.

Our model is based on a single, global, self-attention layer followed by two, fully connected network layers. We intentionally designed and tested the simplest architecture with global attention, and without positional encoding to establish a baseline method for such architectures. Limiting the aperture of the attention to a local region as well as adding the positional encoding are simple modifications that are likely to further improve the performance. We are considering these extensions for our future work.

The complete PyTorch 0.4 source code to train and evaluate our model, as well as trained weights to reproduce results in this paper, will be publicly available on <https://github.com/ok1zjf/VASNet>.

References

1. Argyriou, V.: Sub-hexagonal phase correlation for motion estimation. *IEEE Transactions on Image Processing* **20**(1), 110–120 (Jan 2011)
2. Athiwaratkun, B., Kang, K.: Feature representation in convolutional neural networks. *arXiv preprint arXiv:1507.02313* (2015)
3. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. *arXiv preprint arXiv:1607.06450* (2016)
4. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014)
5. Cheng, J., Dong, L., Lapata, M.: Long short-term memory-networks for machine reading. In: *Proceedings of the EMNLP*. pp. 551–561 (2016)
6. Cho, K., Merriënboer, B., Gulcehre, C., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. In: *Proceedings of the EMNLP* (2014)
7. De Avila, S.E.F., Lopes, A.P.B., da Luz Jr, A., de Albuquerque Araújo, A.: Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters* **32**(1), 56–68 (2011)
8. Fajtl, J., Argyriou, V., Monekosso, D., Remagnino, P.: Amnet: Memorability estimation with attention. In: *Proceedings of the IEEE CVPR*. pp. 6363–6372 (2018)
9. Fei, M., Jiang, W., Mao, W.: Memorable and rich video summarization. *J. Vis. Commun. Image Represent.* **42**(C), 207–217 (Jan 2017)
10. Gehring, J., et al.: Convolutional sequence to sequence learning. In: *Proceedings of the ICML*. pp. 1243–1252 (06–11 Aug 2017)
11. Graves, A., et al.: Hybrid computing using a neural network with dynamic external memory. *Nature* **538**(7626), 471 (2016)
12. Gygli, M., Grabner, H., Riemenschneider, H., Van Gool, L.: Creating summaries from user videos. In: *Proceedings of the ECCV*. pp. 505–520. Springer (2014)
13. Gygli, M., et al.: Video summarization by learning submodular mixtures of objectives. In: *Proceedings of the IEEE CVPR*. pp. 3090–3098 (2015)
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural computation* **9**(8), 1735–1780 (1997)
15. Ji, Z., Xiong, K., Pang, Y., Li, X.: Video summarization with attention-based encoder-decoder networks. *arXiv preprint arXiv:1708.09545* (2017)
16. Khosla, A., Raju, A.S., Torralba, A., Oliva, A.: Understanding and predicting image memorability at a large scale. In: *Proceedings of the IEEE ICCV*. pp. 2390–2398 (2015)
17. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: *Proceedings of the ICLR*. vol. 5 (2015)
18. Larkin, K.G.: Reflections on shannon information: In search of a natural information-entropy for images. *CoRR* **abs/1609.01117** (2016)
19. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. In: *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. pp. 74–81. Association for Computational Linguistics, Barcelona, Spain (July 2004)
20. Lin, Z., Feng, M., dos Santos, C.N., Yu, M., Xiang, B., Zhou, B., Bengio, Y.: A structured self-attentive sentence embedding. In: *Proceedings of the ICLR* (2017)
21. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015)
22. Luong, T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: *Proceedings of the EMNLP* (2015)

23. Mahasseni, B., Lam, M., Todorovic, S.: Unsupervised video summarization with adversarial lstm networks. *Proceedings of the IEEE CVPR* pp. 2982–2991 (2017)
24. Mnih, V., Heess, N., Graves, A., et al.: Recurrent models of visual attention. In: *Proceedings of the NIPS*. pp. 2204–2212 (2014)
25. Novak, C.L., Shafer, S.A.: Anatomy of a color histogram. In: *Proceedings of the IEEE CVPR*. pp. 599–605. IEEE (1992)
26. Otani, M., et al.: Video summarization using deep semantic features. In: *Proceedings of the ACCV*. pp. 361–377 (2016)
27. Parikh, A., et al.: A decomposable attention model for natural language inference. In: *Proceedings of the EMNLP*. pp. 2249–2255 (2016)
28. Potapov, D., Douze, M., Harchaoui, Z., Schmid, C.: Category-specific video summarization. In: *Proceedings of the ECCV*. pp. 540–555. Springer (2014)
29. Rochan, M., Wang, Y.: Learning video summarization using unpaired data. *arXiv preprint arXiv:1805.12174* (2018)
30. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Others: Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* **115**(3), 211–252 (2015)
31. dos Santos Belo, L., Caetano Jr, C.A., do Patrocínio Jr, Z.K.G., Guimarães, S.J.F.: Summarizing video sequence using a graph-based hierarchical approach. *Neurocomputing* **173**, 1001–1016 (2016)
32. Song, Y., Vallmitjana, J., Stent, A., Jaimes, A.: Tvsum: Summarizing web videos using titles. In: *Proceedings of the IEEE CVPR*. pp. 5179–5187 (2015)
33. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: *Proceedings of the IEEE CVPR*. pp. 1–9 (2015)
34. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: *Proceedings of the NIPS*, pp. 5998–6008. Curran Associates, Inc. (2017)
35. Wei, H., Ni, B., Yan, Y., Yu, H., Yang, X., Yao, C.: Video summarization via semantic attended networks. In: *Proceedings of the AAAI* (2018)
36. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. In: *Reinforcement Learning*, pp. 5–32. Springer (1992)
37. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: *Proceedings of the ICML*. pp. 2048–2057 (2015)
38. Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., Courville, A.: Describing videos by exploiting temporal structure. In: *Proceedings of the IEEE ICCV*. pp. 4507–4515 (2015)
39. Yuan, Y., Mei, T., Cui, P., Zhu, W.: Video summarization by learning deep side semantic embedding. *IEEE Transactions on Circuits and Systems for Video Technology* (2017)
40. Zhang, K., Chao, W.L., Sha, F., Grauman, K.: Video summarization with long short-term memory. In: *Proceedings of the ECCV*. pp. 766–782. Springer (2016)
41. Zhao, B., Li, X., Lu, X.: Hierarchical recurrent neural network for video summarization. In: *Proceedings of the ACM Multimedia Conference*. pp. 863–871 (2017)
42. Zhou, K., Qiao, Y., Xiang, T.: Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In: *Proceedings of the AAAI* (2018)