Stacked Memory Network for Video Summarization

Junbo Wang[1,2], Wei Wang[1,2,*], Zhiyong Wang[3], Liang Wang[1,2], Dagan Feng[3], Tieniu Tan[1,2]
[1]Center for Research on Intelligent Perception and Computing (CRIPAC),
National Laboratory of Pattern Recognition (NLPR),
Institute of Automation, Chinese Academy of Sciences (CASIA)
[2]University of Chinese Academy of Sciences (UCAS)
[3]School of Computer Science, The University of Sydney
{junbo.wang, wangwei, wangliang, tnt}@nlpr.ia.ac.cn, {zhiyong.wang, dagan.feng}@sydney.edu.au

ABSTRACT

In recent years, supervised video summarization has achieved promising progress with various recurrent neural networks (RNNs) based methods, which treats video summarization as a sequence-to-sequence learning problem to exploit temporal dependency among video frames across variable ranges. However, RNN has limitations in modelling the long-term temporal dependency for summarizing videos with thousands of frames due to the restricted memory storage unit. Therefore, in this paper we propose a stacked memory network called SMN to explicitly model the long dependency among video frames so that redundancy could be minimized in the video summaries produced. Our proposed SMN consists of two key components: Long Short-Term Memory (LSTM) layer and memory layer, where each LSTM layer is augmented with an external memory layer. In particular, we stack multiple LSTM layers and memory layers hierarchically to integrate the learned representation from prior layers. By combining the hidden states of the LSTM layers and the read representations of the memory layers, our SMN is able to derive more accurate video summaries for individual video frames. Compared with the existing RNN based methods, our SMN is particularly good at capturing long temporal dependency among frames with few additional training parameters. Experimental results on two widely used public benchmark datasets: SumMe and TVsum, demonstrate that our proposed model is able to clearly outperform a number of state-of-the-art ones under various settings.

CCS CONCEPTS

• **Computing methodologies** → **Video summarization**; **Structured outputs**.

∗Corresponding Author.

MM '19, October 21–25, 2019, Nice, France

ACM ISBN 978-1-4503-6889-6/19/10…$15.00
https://doi.org/10.1145/3343031.3350992

KEYWORDS

video summarization, stacked memory network, temporal dependency, recurrent neural network

ACM Reference Format:
Junbo Wang, Wei Wang, Zhiyong Wang, Liang Wang, Dagan Feng, Tieniu Tan. 2019. Stacked Memory Network for Video Summarization. In *Proceedings of the 27th ACM International Conference on Multimedia (MM'19), Oct. 21-25, 2019, Nice, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3343031.3350992

1 INTRODUCTION

The amount of video data has increasingly dominated various internet computing and communication platforms, such as social media and mobile phones. For example, it has been reported that almost 5 billion videos are watched on YouTube every single day [20]. As a result, it has been very demanding to develop advanced computing techniques such as video summarization for processing, retrieving, and browsing video content effectively and efficiently. Video summarization aims to condense a given video into a short video summary (e.g., a collection of key video segments[23], keyframes[3], and key objects[12]). Due to its importance in a wide range of real world applications, many keyframe based video summarization methods have been proposed[1, 7, 15, 19, 30, 42].

While most existing summarization methods are unsupervised and do not require supervision information on keyframes, as inspired by the advances of deep learning techniques, many temporal deep learning based methods have been proposed by treating video summarization as a sequence-to-sequence learning problem, which translates a video frame sequence into a series of binary decisions or importance scores for the video frames. In [38], long short-term memory (LSTM) was first utilized to model temporal structure among frames for video summarization. In [40][41], hierarhical LSTM models were proposed to discover semantic structure within a video by using two LSTM layers. However, recent studies [21, 31] show that LSTM is not effective enough in dealing with temporal structures of videos longer than 80 frames (about 3 to 4 seconds). As a result, these recurrent neural network based methods have limitations to effectively model videos with several thousands of frames (about several minutes), which may eventually compromise the quality of the summaries produced.

To better model the long term temporal dependency among video frames, we propose a stacked memory network (SMN) for video summarization. Our SMN consists of two basic components: LSTM layer and memory layer, where each LSTM layer is augmented with an external memory layer. Within each layer, the LSTM layer interacts with its corresponding memory layer via one read head and one write head. In addition, we stacks multiple LSTM layers and memory layers hierarchically to integrate the learned representation from prior layers. To better fuse the learned representation from previous layers, we also explore different types of connections between two memory layers. As a result, each LSTM layer in our SMN receives the information from both the previous LSTM layers and memory layers to better derive long-term temporal context. By combining the hidden states of the LSTM layers and the read representations of the memory layers, our SMN is able to produce more accurate importance scores for video frames. Compared with the existing RNNs based methods, our SMN is also able to capture longer temporal dependency among frames with fewer additional training parameters.

To demonstrate the effectiveness of our proposed SMN, we conducted various experiments on two widely used public benchmark datasets: SumMe and TVsum. Experimental results indicate that our proposed method is able to achieve the best published performance to-date on both datasets under various settings.

Overall, the key contributions of our work are summarized as follows:

- We propose a novel stacked memory network to model the temporal dependency among video frames. As a result, long term temporal structures of videos can be exploited for producing video summaries with less redundancy.
- We explore different types of connections between two memory networks to fuse the learned representation from previous layers, and demonstrate that attention based fusion is the best for video summarization.
- We conduct comprehensive experiments on two widely used benchmark datasets for evaluating the performance of our proposed method and investigating its impact on different aspects. Experimental results demonstrate that our model outperforms the state-of-the-art methods by a large margin under various settings.

2 RELATED WORK

In general, there are two categories of video summarization methods: unsupervised and supervised ones, in terms of whether supervision information of a summary is utilized for developing summarization algorithms.

2.1 Unsupervised methods

Unsupervised video summarization methods aim to derive an importance score using heuristic rules for a video frame or a video segment (e.g., video shot), and choose those having high importance scores according to a predefined requirement (e.g., the number of keyframes or the duration of a summary). For example, clustering techniques have been used to group similar video frames into visually similar clusters and the cluster centres are chosen as keyframes form a final summary [30]. Recently, advanced representation and learning techniques are explored for video summarization. In [1, 16, 17, 19, 42], sparse representations techniques have been proposed to score the importance of each frame in terms of sparse represenation cost.

By following the adversarial nature of generative adversary network (GAN) [6], a generative adversarial framework [18] was proposed for video summarization with two components: summarizer and discriminator. The summarizer is trained to output frame selection and the discriminator is to differentiate an original video from the video reconstructed from the output of the summarizer. In [43], video summarization was also formulated under the framework of deep reinforcement learning with diversity and representativeness rewards.

2.2 Supervised methods

Supervised video summarization methods aim to utilize supervision information to train a classification or recognition model which classifies a given frame (or segment) into one of the two classes (i.e., keyframe class and non-keyframe class). For example, in [9], supervision information was utilized to learn a linear combination of multiple summarization objectives in the process of subset selection. In recent years, deep learning techniques have been increasingly utilized for video summarization. Similarly, in [13], a weighted score function was learned for the four aspects of ranking, importance, representativeness, diversity, and storyness. Therefore, in this section, we focus on reviewing supervised deep learning based methods which are closely relevant to our proposed method.

When supervision information is available at video or segment level only, video summarization is formulated as a task of classifying a video or segment. In [35], a recurrent auto-encoder model was proposed for video summarization by extending conventional auto-encoders with LSTM. The recurrent model is trained with highlight videos so that unseen non-highlight videos will produce large reconstruction error when going through the recurrent auto-encoder model. In [36], a deep convolutional neural network was trained with a pair-wise ranking function so that the trained deep network is able to produce a highlight probability score for a given video segment. In [24], video summarization was formulated as a two-class temporal segmentation problem using a fully convolutional network [14], which contains a series of convolution, pooling and deconvolution operations.

When frame level annotation is available for each video, supervised deep learning methods formulate video summarization as a sequential labelling task using recurrent neural networks. In [38], LSTM (long short-term memory) models were developed and trained to predict a label (i.e. keyframe or non-keyframe) for each video frame of a given video. In [39], a retrospective encoder was proposed to ensure that

the original video and its summary will be close enough in the embedded space. In [40], a hierarchical recurrent neural network was proposed to perform video summarization at two steps: the first layer RNN encodes a video segment and the second layer RNN derives the confidence score whether the video segment is chosen as a key segment. In [41], a hierarchical model which consists of two LSTM networks for video summarization, where one LSTM network is responsible for partitioning a video into segments by discovering video structures and the other LSTM network is responsible for producing keyframes for each segment.

However, most recurrent neural network based summarization methods have limitations in modeling long-term temporal structures due to the restricted memory storage unit. Inspired by the success of existing memory models in modelling long-term dependency in question answering [34] and video captioning [32], we propose a novel stacked memory network called SMN to explicitly model the long dependency among video frames. Compared the existing RNNs based methods, our SMN fuses different representations from LSTM layers and memory layers and can capture longer temporal dependency among frames with fewer additional training parameters.

3 OUR PROPOSED METHOD

Since keyframe based video summarization is to produce an importance score for each frame of a given video, we formulate it as a sequence-to-sequence learning problem. For a given video $X = \{x_1, x_2, \ldots, x_T\}$, where x_t $(t \in \{1, 2, \ldots, T\})$ denotes the feature representation of the t-th frame in the video, our method is to predict a set of frame level importance scores or binary labels $Y = \{p_1, p_2, \ldots, p_T\}$.

The overall framework of our method is illustrated in Figure 1. For an input video X, we first employ a pre-trained CNN network to extract feature representation for each frame, and feed the representation into our proposed SMN for producing importance scores or binary lables Y. Our proposed SMN consists of m LSTM layers and m Memory layers, where each LSTM layer is augmented with an external memory layer. Within each layer of our SMN, a LSTM layer interacts with its corresponding memory layer through one read head and one write head. At the same time, the hidden states in each LSTM layer or memory layer are forwarded into next LSTM layer or memory layer, respectively. After stacking m layers, we gather the representations from each LSTM layer and memory layer into the output layer to predict importance scores for the frames of the input video. In addition, we explore different types of connections between two memory networks to fuse the learned representation from previous layers.

In this section, we first describe the four major components of our proposed SMN method: CNN-based deep feature extraction (Section 3.1), LSTM-based temporal model (Section 3.2), memory model (Section 3.3) and stacked memory network (Section 3.4). Finally, we present the details of model training (Section 3.5).

3.1 CNN-based Deep Feature Extraction

Since CNN has achieved great success in image and video understanding tasks [4, 11], the high level activations are usually employed as representations of image and video data. Moreover, it has been reported that deep features are able to achieve better performance than shallow features (e.g, color histograms, GIST, HOG, dense SIFT) for video summarization [38]. In order to obtain effective visual representation for each frame in a video, we follow [38] to employ the activations from the penultimate layer of a deep CNN network as frame features. Given an input video X, we use GoogleNet [28] pre-trained on ImageNet [11] to encode the video into a set of feature vectors, denoted by $X = \{x_1, x_2, \ldots, x_t, \ldots, x_T\}$.

Note that we can also use deeper networks (e.g., ResNet [10] and C3D [29]) to obtain better representations of video frames. We use GoogleNet [28] for feature extraction in order to perform a fair comparison with previous works.

3.2 LSTM-based Temporal Model

The standard LSTM unit mainly contains an internal memory cell, an input gate, a forget gate and a output gate. The memory cell recurrently updates its hidden state by fusing the previous cell state and the current input with these gates. During each timestep t, we feed the input frame representation (or the hidden states in the previous layer), the previous hidden state at timestep $t-1$ and the read information r_{t-1} from the corresponding memory layer into the LSTM layer. The read information r_{t-1} will be described in next section. The transition formulas for a LSTM layer's forward pass are given below:

$$i_t = \sigma\left(W_i x_t + U_i h_{t-1} + R_i r_{t-1}\right), \tag{1}$$

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + R_f r_{t-1}\right), \tag{2}$$

$$o_t = \sigma\left(W_o x_t + U_o h_{t-1} + R_o r_{t-1}\right) \tag{3}$$

$$\widetilde{c}_t = \phi\left(W_c x_t + U_c h_{t-1} + R_c r_{t-1}\right), \tag{4}$$

$$c_t = i_t \odot \widetilde{c}_t + f_t \odot c_{t-1}, \tag{5}$$

$$h_t = o_t \odot \phi\left(c_t\right), \tag{6}$$

where $\odot$ denotes an element-wise multiplication, W, U, and R denote the weight parameters to be learned, and all the bias terms are omitted. σ denotes the element-wise logistic sigmoid function, and ϕ denotes hyperbolic tangent function tanh.

For the simplicity of illustration, the update procedure of the above-mentioned LSTM layer can be abbreviated as follows:

$$h_t = f_{lstm}\left(h_{t-1}, c_{t-1}, x_t, r_{t-1}\right) \tag{7}$$

3.3 Memory Model

The memory layer augmenting the j-th LSTM layer can be defined as a $N \times F$ matrix M_t^j at time t, where N denotes the number of memory locations and F denotes the vector length of each location. The memory layer interacts with its corresponding LSTM layer through one read head and one write head. The read and write heads perform selective read and write operations through an addressing mechanism. We

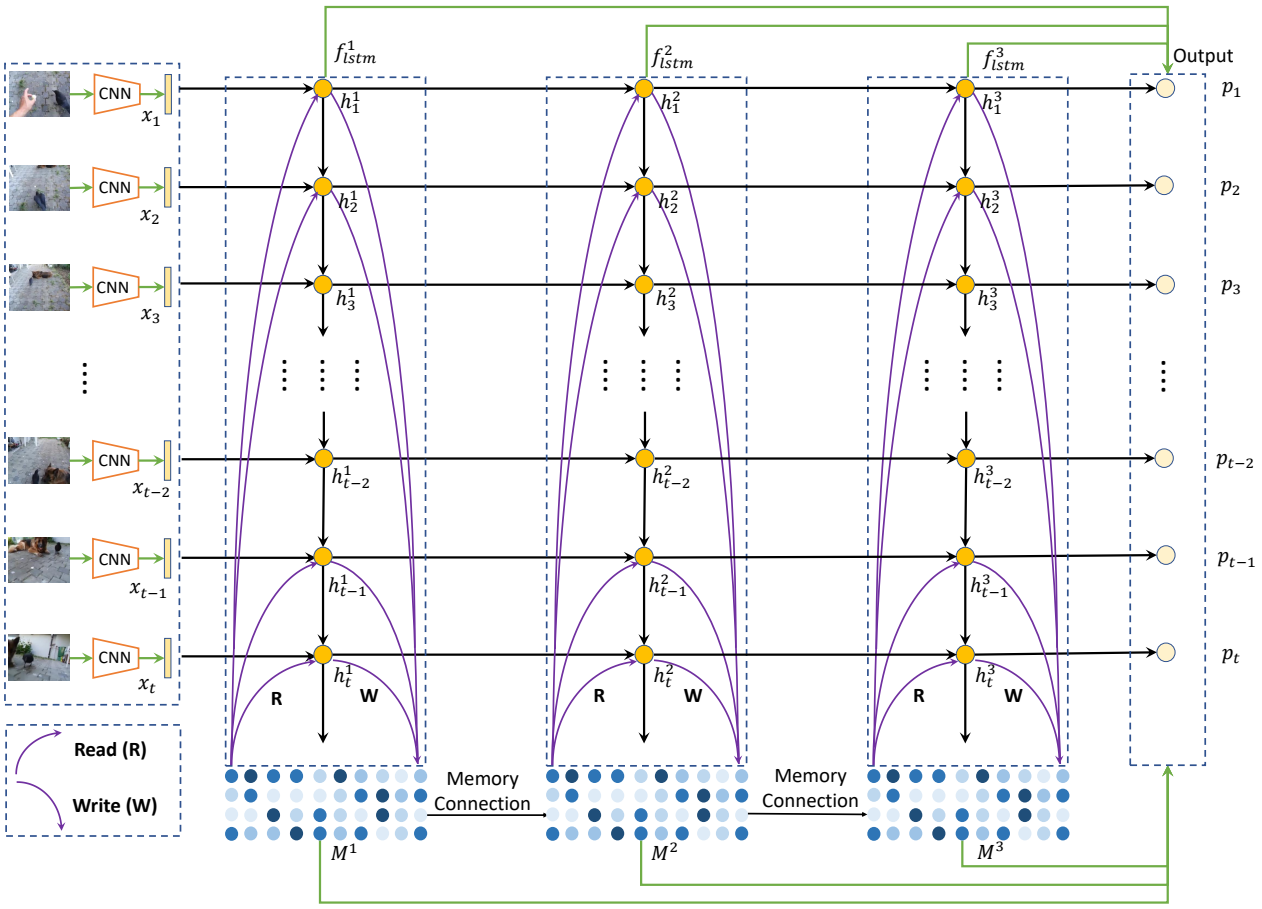

Figure 1: The overall framework of our proposed Stacked Memory Network (SMN) based video summarization method (to be best viewed in color). Given a video, we first employ the pretrained CNN network to extract video frame features. Then, we forward these features into our stacked memory networks to update the states of LSTM layers and memory layers. After combining the states from these LSTM layers and memory layers, we employ a fully-connected layer to predict each frame an importance score. In addition, We also explore different types of connections between two memory networks to fuse the learned representation from previous layers.

will describe the three basic operations of the memory model and different types of connections between two memory layers in detail below.

3.3.1 *Reading.*

Given the weighting vector w_t^j emitted by the j-th read head over the N locations at time t, which needs to be constrained as follows:

$$\sum_{i=1}^{N} w_t^j(i) = 1, 0 \leq w_t^j(i) \leq 1, \forall \mathrm{i} \in [1, N]. \tag{8}$$

Then the read vector r_t^j returned by the j-th read head is calculated as a linear weighting of the row-vectors $M_t(i)$:

$$r_t^j = \sum\nolimits_{i=1}^{N} w_t^j(i) M_t^j(i). \tag{9}$$

3.3.2 *Writing.*

The writing operation is divided into two parts: erase and add. Here we define the weighting vector, the erase vector and the add vector as w_t^j, e_t^j and a_t^j, respectively, all of which are emitted by the j-th write head. The elements of erase vector e_t^j lie in the range of (0,1). The length of both the erase vector e_t^j and the add vector a_t^j is M. Since both the erase vector and add vector have M independent elements, the elements in each memory location can be erased or added in a fine-grained way. Then the memory state can be updated as follows:

$$M_t^j(i) = M_{t-1}^j(i) \left[1 - w_t^j(i) e_t^j\right] + w_t^j(i) a_t^j, \tag{10}$$

where $\mathrm{i} \in [1, N]$ denotes the i-th memory location.

3.3.3 *Memory Addressing.*

We use a combination of content-based addressing and location-based addressing to update the above read/write weighting vector. During content-based addressing, each read/write head first produces a key vector k_t and a sharpening factor β_t. The key vector k_t is mainly used for comparing with each memory vector $M_t(i)$ using a similarity measure function K, and the sharpening factor β_t is employed for regulating the precision of the focus. Then they all can be computed as follows:

$$K(x, y) = \frac{x \cdot y}{\|x\| \cdot \|y\| + \varepsilon}, \tag{11}$$

$$d_t(i) = \beta_t K(k_t, M_t(i)), \tag{12}$$

$$w_t(i) = softmax(d_t(i)). \tag{13}$$

The location-based addressing mainly focuses on simple iterations among the locations of the memory and random-access jumps, which extends the content-based addressing with a scalar interpolation gate. Before calculating the read/write weighting vector, the read/write head first produces a scalar interpolation gate g_t, a shift weighting vector s_t and a sharpen weighting scalar γ_t. The scalar interpolation gate g_t is used for blending previously generated weighting vector w_{t-1} and the weighting vector w_t^c produced by the content-based addressing. The shift weighting vector s_t is defined as a normalised distribution across the fixed integer range. The sharpen weighting scalar γ_t is employed to sharpen the final weighting vector. The final weighting vector can be formulated as follows:

$$w_t^l = g_t w_t^c + (1 - g_t) w_{t-1}, \tag{14}$$

$$\widetilde{w}_t(i) = \sum_{j=0}^{N-1} w_t^l(j) s_t(i - j), \tag{15}$$

$$w_t(i) = \frac{\widetilde{w}_t(i)^{\gamma_t}}{\sum_{j=0}^{N-1} \widetilde{w}_t(j)^{\gamma_t}}. \tag{16}$$

3.3.4 Connections between Memory Layers.

To figure out what is the best way of connecting two memory layers, we propose five different types of connection in our SMN. The five different connections are described as follows:

1) Scalar based Addition (SA):

$$M_t^j = aM_t^j + bM_t^{j-1}, \tag{17}$$

where a and b are scalar mixture weights, and the default operation between two memory layers is element-wise addition.

2) Global Gated Addition (GGA):

$$\alpha_t^j = \sigma\left(z^{\mathrm{T}} \tanh\left(W_j M_t^j + U_j M_t^{j-1}\right)\right), \tag{18}$$

$$M_t^j = M_t^j + \alpha_t^j M_t^{j-1}, \tag{19}$$

where W_j, U_j and z are the weight parameters to be learned, σ denotes the element-wise logistic sigmoid function and α_t^j is a scalar gate.

3) Location-wise Gated Addition (LGA):

$$\alpha_t^j = \sigma\left(W_j M_t^j + U_j M_t^{j-1}\right), \tag{20}$$

$$M_t^j = M_t^j + \alpha_t^j \odot M_t^{j-1}, \tag{21}$$

where W_j and U_j are the weight parameters to be learned, $\odot$ denotes an element-wise multiplication, σ denotes the element-wise logistic sigmoid function and α_t^j is a vector gate over N locations.

4) Recurrent Learning based Addition (RLA):

$$M_t^j = \phi\left(W_j M_t^j + U_j M_t^{j-1}\right), \tag{22}$$

where W_j and U_j are the weight parameters to be learned, and ϕ denotes hyperbolic tangent function tanh.

5) Attention based Addition (AA):

$$\alpha_t^j = \text{softmax}\left(\frac{M_t^j \cdot M_t^{j-1^T}}{\sqrt{F}}\right), \tag{23}$$

$$M_t^j = M_t^j + \alpha_t^j M_t^{j-1}, \tag{24}$$

where α_t^j is calculated by a scaled dot-product attention model and it is a $N \times N$ matrix.

3.4 Stacked Memory Network

Given an input video frame sequence $x = \{x_1, x_2, \ldots, x_T\}$ and the number m of stacked LSTM layers, our SMN predicts the importance scores via stacked memory networks. In particular, we first feed the input frame sequence or the hidden state at previous layer into the next LSTM layer to update the hidden states for each LSTM layer. The hidden state for the j-th LSTM layer can be calculated as follows:

$$h_t^j = f_{lstm}\left(h_{t-1}^j, c_{t-1}^j, h_t^{j-1}, r_{t-1}^j\right), \tag{25}$$

where the read vector r_0^j is initialized with random float number to be learned, h_0^j and c_0^j are initialized with zeros.

Next, the j-th LSTM layer performs selective read and write operations on the corresponding memory layer according to previous introduction as follows:

$$r_t^j = f_{read}\left(wr_{t-1}^j, M_{t-1}^j, h_t^j\right), \tag{26}$$

$$M_t^j = f_{write}\left(ww_{t-1}^j, M_{t-1}^j, h_t^j\right), \tag{27}$$

where f_{read} denotes the read operation and f_{write} denotes the write operation.

Next, we transform the states from the low-layer memory into the high-layer memory as follows:

$$M_t^j = f_{connect}\left(M_t^j, M_{t-1}^j\right), \tag{28}$$

where $f_{connect}$ denotes one of the five connection functions. Based on the hidden states of all the LSTM layers and the read information from all the memory layers, we fuse them to predict the corresponding importance score for each video frame as follows:

$$p_t = \left[h_t^1, \cdots, h_t^m, r_t^1, \cdots, r_t^m\right] W^o, \tag{29}$$

where [] denotes the concatenation operation and W^o denotes the weights to be learned.

3.5 Model Training

Our SMN model is trained in terms of two loss functions: prediction loss and diversity loss. Since the importance scores of the input video's frames are of continuous values, we employ mean-square loss in our experiments as prediction loss:

$$L_{sum} = \frac{1}{T}\sum_{t=1}^{T} \|y_t - p_t\|_2^2, \tag{30}$$

where p_t is the predicted importance score and y_t is the ground-truth importance score.

Assuming that the set Y is the selected frames in the summary, we define the mean of the pairwise similarity between these selected frames as follows:

$$L_{div} = \frac{1}{|Y|\,(|Y|-1)} \sum_{t\in Y} \sum_{t'\in Y, t'\neq t} d\,(x_t, x_{t'}), \tag{31}$$

$$d\,(x_t, x_{t'}) = \frac{{x_t}^T x_{t'}}{\|x_t\|\,\|x_{t'}\|}. \tag{32}$$

Finally, we train our model by balancing the two loss functions as follows:

$$L = L_{sum} + \lambda L_{div}, \tag{33}$$

where λ denotes tradeoff hyperparameter.

4 EXPERIMENTAL RESULTS

4.1 Datasets

We evaluate and compare our proposed method with other state-of-the-art methods on two public benchmark datasets: SumMe [8] and TVSum [27].

SumMe Dataset contains 25 user videos covering a variety of events (e.g., cooking and sports). Each video in this dataset varies from 1.5 to 6.5 minutes in length and was annotated with a sequence of frame-level importance scores by 15 to 18 persons. In this dataset, there are many first-person videos and third-person videos.

TVSum Dataset consists of 50 videos collected from YouTube. All videos in this dataset were selected from 10 different categories (e.g. *animal grooming*, *making sandwich*, *changing vehicle tire*, etc.) from the TRECVid Multimedia Event Detection (MED) task [26]. Each video varies from 1 to 5 minutes in length and was annotated by 20 users. Similar to SumMe, these videos are provided with frame-level importance scores and include first-person camera and third-person camera.

To increase the amount of annotated data, we follow [38] to use other two datasets as auxiliary datasets. In particular, we took 39 videos from the YouTube dataset [2] and 50 videos from the Open Video Project (OVP) dataset [2, 22] for augmented setting and transfer setting [38]. The videos in YouTube dataset cover a variety of events including news, sports and cartoon, and the videos in OVP dataset cover different types of content, such as documentary. Since each video in the two datasets was annotated with multiple sets of keyframes, we follow [5] to create a single ground-truth set of keyframes for each video. To make a full comparison with the state-of-the-art methods, we follow [38] to train and test our models on three different dataset settings: canonical setting, augmented setting and transfer setting.

4.2 Experimental Settings

Similar to [38], we downsample each video in all the datasets to 2 fps, and extract the output of the penultimate layer (pool5) of GoogleNet [28] pretrained on ImageNet [25] as the frame representation. For the LSTM layers in our SMN, we set the input size and hidden size of first LSTM layer to 1024 and 256 respectively, the input size and hidden size of other LSTM layers to 256, 256 respectively, and the number of the stacked LSTM layers m to 5. For the memory layers in our SMN, we set the number of memory locations N and the vector length of each location F to 256 and 32 respectively. For the scalar based addition between two memory layers, we set scalar weight a and scalar weight b to 1 and 1 respectively. In our experiments, we employ sigmoid activation function to predict the final importance score for each video frame. Since the ground-truth scores in some datasets are greater than 1, we normalize all the scores to between 0 and 1 before training. In addition, we set the batch size, learning rate, number of training epochs, balance factor λ to 1, 0.001, 200, 0.01, respectively. To prevent overfitting, we also add a dropout with rate 0.15 to the final output layer. We train our model with the Adam optimizer and decay the learning rate by 0.5 every 30 epoches. To make a fair comparison with existing approaches, we follow [38] to run each testing fold for 5 times and report the average result for these datasets.

4.3 Evaluation Metrics

To make a fair comparison with existing approaches, we follow [38] to use F-score as the evaluation metric to measure similarity between a generated summary and a ground truth summary. Given the predicted summary A and the ground truth summary B, we use the temporal overlap between the two sets to define the following metrics:

$$P = \frac{\text{overlap between A and B}}{\text{duration of A}} \tag{34}$$

$$R = \frac{\text{overlap between A and B}}{\text{duration of B}} \tag{35}$$

$$F = \frac{2P \times R}{P + R} \times 100\% \tag{36}$$

Since a video usually has multiple human-annotated summaries, we follow [9, 27, 38] to compute the average or maximum values of the metric across multiple summaries.

4.4 Quantitative Analysis

Table 1 shows the performance comparison between our models and other 13 state-of-the-art methods on the SumMe dataset. From these results, we can observe that our model outperforms other thirteen state-of-the-art methods by a large margin under all the three different settings. For example, our Stack-LSTM which employs stacked LSTM layers without memory layers also achieves better performance than other state-of-the-art methods such as the closest competitor SUM-FCN [24], which proves the effectiveness of

Method	Canonical	Augmented	Transfer
CSUV [8]	39.4	-	-
LSMO [9]	39.7	-	-
SMTF [37]	40.9	41.3	38.5
vsLSTM [38]	37.6	41.6	40.7
dppLSTM [38]	38.6	42.9	41.8
US-LSTM [18]	41.7	43.6	-
HRNN [40]	-	43.6	-
AFER [13]	43.1	-	-
HSA-RNN [41]	-	44.1	-
DR-DSN [43]	42.1	43.9	42.6
SASUM [33]	-	45.3	-
SUM-FCN [24]	47.5	51.1	44.1
Seq2seq [39]	-	44.9	-
Stack-LSTM	49.2	52.3	45.7
Our-SMN	**58.3**	**60.1**	**50.2**

Table 1: Performance comparison with thirteen state-of-the-art methods on the SumMe dataset. The results of the baseline model (Stack-LSTM) and our full model (Our-SMN) are shown at the bottom of the table. Stack-LSTM employs five stacked LSTM layers without memory layers and Our-SMN employs five stacked LSTM layers and memory layers.

our stacked structure. By incorporating the stacked memory layers, Our-SMN achieves F-scores 58.3, 60.1 and 50.2 under three different settings respectively, making the absolute improvement over the baseline by 9.1%, 7.8% and 4.5% respectively, which indicates that Our-SMN is very good at modelling long-term temporal dependency for video summarization. Although these methods (vsLSTM [38], HRNN [40] and HSA-RNN [41]) also attempt to model long-term temporal dependency among video frames, our model can make a large improvement over them in terms of three different settings, which further demonstrates modelling long-term temporal dependency across video frames is helpful for video summarization. When augmenting training data (augmented setting) during training, we notice that our proposed model SMN can achieve even better performance, which indicates that adding more data is very necessary for training a better model.

Table 2 shows the results of our models against other ten state-of-the-art methods in terms of three different settings on the TVSum dataset. Similarly, it can be seen that Our-SMN outperforms other state-of-the-art methods in terms of three settings by large margins. In the canonical setting and transfer setting, Our-SMN substantially outperforms the closest competitor DR-DSN [43] by 6.4% and 6.3% respectively. In the augmented setting, Our-SMN also outperforms the closest competitor Seq2seq [39] by 4.5%. Similar to the observations on the SumMe dataset, Our-SMN also outperforms Stack-LSTM by a large margin, especially in the augmented setting by modelling the long-term temporal dependency

Method	Canonical	Augmented	Transfer
vsLSTM [38]	54.2	57.9	56.9
dppLSTM [38]	54.7	59.6	58.7
US-LSTM [18]	56.3	61.2	-
HRNN [40]	-	61.5	-
AFER [13]	52.7	-	-
HSA-RNN [41]	-	59.8	-
DR-DSN [43]	58.1	59.8	58.9
SASUM [33]	-	58.2	-
SUM-FCN [24]	56.8	59.2	58.2
Seq2seq [39]	-	63.9	-
Stack-LSTM	60.8	63.6	62.1
Our-SMN	**64.5**	**68.4**	**65.2**

Table 2: Performance comparison with ten state-of-the-art methods on the TVSum dataset. Similarly, the results of the baseline model (Stack-LSTM) and our full model (Our-SMN) are shown at the bottom of the table.

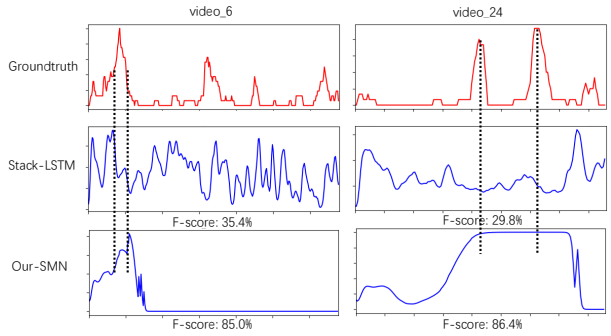

Figure 2: Illustration of ground truth and importance scores predicted by Our-SMN and Stack-LSTM on two test videos (video_6 and video_24). The first row, the second row and the last row denote the importance score curve generated by groundtruth, Stack-LSTM and Our-SMN, respectively. Our-SMN can capture the most important part (peak regions) in the video compared with Stack-LSTM.

with stacked memory layers. These results strongly prove the effectiveness of our method.

4.5 Qualitative Analysis

To learn more about our model, as shown in Figure 2, we visualize the ground truth and importance scores predicted by Our-SMN and Stack-LSTM on two test videos, respectively. It can be see that Our-SMN captures the most important part in the video (peak regions) since it can model the long-term temporal dependency across video frames. For Stack-LSTM, it completely misses selecting the most important part for the test video video_24 due to lacking stacked memory

layers. More importantly, these regions conform to human understanding on which regions are important. This strongly demonstrates that modelling long-term temporal dependency can well imitate the human-decision process and effectively train our model to select important frames from a long video.

5 MODEL ANALYSIS

Computational Complexity To demonstrate that Our-SMN only introduces few additional parameters compared with Stack-LSTM, we report the number of model parameters and the corresponding performance in Table 3. It can be seen that Our-SMN outperforms Stack-LSTM which employs the same stacked LSTM layers by large margins (58.3 and 49.2) while using the similar parameter number (3.76M and 3.42M). The comparison between them further demonstrates the advantage of modelling long-term temporal dependency of our model.

Method	F-score	Parameter Size
Stack-LSTM	49.2	3.42M
Our-SMN	**58.3**	3.76M

Table 3: Comparison of computational complexity between Stack-LSTM and Our-SMN on the SumMe dataset in the canonical setting. Here the F-score corresponds to the model performance. With the similar parameter size, Our-SMN clearly outperforms Stack-LSTM.

Number of Stacked Layers To investigate the impact of the number of stacked layers in our SMN on model performance, we perform extensive experiments on the SumMe dataset by varying the number of stacked layers and fixing the others. Table 4 shows the experimental results of our SMN with different number of stacked layers. It can be seen that the model performance will clearly increase when adding more layers. However, the model performance comes to decrease when the number of stacked layers is larger than 5. When increasing the number of stacked layers, the model is prone to overfitting due to the increase of model parameters.

Connection Types To investigate the impact of different connection types on model performance, we perform extensive experiments on the SumMe dataset under the same parameter setting (the number of stacked layers is 3). Table 5 shows the results of our SMN with different connection ways. When two memories are not connected, we can see that the model SMN-no performs the worst among these models, which demonstrates that the connection between two memories is helpful for modelling long-term temporal dependency. Moreover, when using attention-based connection way, our SMN-AA can achieve clear improvements over other models (SMN-GGA, SMN-LGA and SMN-RLA). It is worth noting that both SMN-SA and SMN-AA without training parameters achieves better results than other models, which demonstrates that parameter learning between two memories may be not very necessary for video summarization.

Number of Stacked Layers	F-score
m-1	49.1
m-2	51.5
m-3	54.5
m-4	56.2
m-5	**58.3**
m-6	57.9

Table 4: Performance comparison of our model with different number of stacked layers on the SumMe dataset in the canonical setting. Here the number "m" denotes the number of stacked layers in Our-SMN. The second column shows the experimental results of the number "m" with different values.

Connection Type	F-score
SMN-no	50.3
SMN-SA	53.8
SMN-GGA	51.6
SMN-LGA	51.7
SMN-RLA	51.1
SMN-AA	**54.5**

Table 5: Performance comparison of our SMN under different connection types between two memory layers on the SumMe dataset in the canonical setting. Here the connection "SMN-no" denotes our stacked memory networks without any connection between two memory layers. The other five connection types are presented in the previous section.

6 CONCLUSIONS

In this paper, we present a novel model called stacked memory network (SMN) for video summarization by capturing long term temporal dependency among video frames with multiple LSTM layers and memory layers. In our proposed SMN, multiple LSTM layers and memory layers are stacked hierarchically and different connection types can be employed for connecting two adjacent memory layers. The experimental results on the two public datasets indicate that our proposed SMN can clearly boost summarization performance under various settings and outperform more than 10 state-of-the-art methods by a large margin.

ACKNOWLEDGMENTS

This work is jointly supported by National Key Research and Development Program of China (2016YFB1001000), National Natural Science Foundation of China (61420106015,61572504). In addition, this work is also supported by Science and Technology Project of SGCC Research on feature recognition and prediction of typical ice and wind disaster for transmission lines based on small sample machine learning method.

REFERENCES

[1] Yang Cong, Junsong Yuan, and Jiebo Luo. 2012. Towards scalable summarization of consumer videos via sparse dictionary selection. *IEEE Transactions on Multimedia* 14, 1 (2012), 66–75.

[2] Sandra Eliza Fontes De Avila, Ana Paula Brandão Lopes, Antonio da Luz Jr, and Arnaldo de Albuquerque Araújo. 2011. VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters* 32, 1 (2011), 56–68.

[3] Ehsan Elhamifar, Guillermo Sapiro, and Rene Vidal. 2012. See all by looking at a few: Sparse modeling for finding representative objects. In *CVPR*. 1600–1607.

[4] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. 2016. Convolutional two-stream network fusion for video action recognition. In *CVPR*. 1933–1941.

[5] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. 2014. Diverse sequential subset selection for supervised video summarization. In *NIPS*. 2069–2077.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[7] Genliang Guan, Zhiyong Wang, Shaohui Mei, Max Ott, Mingyi He, and David Dagan Feng. 2014. A top-down approach for video summarization. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 11, 1 (2014), 4.

[8] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. 2014. Creating summaries from user videos. In *ECCV*. Springer, 505–520.

[9] Michael Gygli, Helmut Grabner, and Luc Van Gool. 2015. Video summarization by learning submodular mixtures of objectives. In *CVPR*. 3090–3098.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*. 1097–1105.

[12] Xuelong Li, Zhigang Wang, and Xiaoqiang Lu. 2016. Surveillance video synopsis via scaling down objects. *IEEE Transactions on Image Processing* 25, 2 (2016), 740–755.

[13] Xuelong Li, Bin Zhao, and Xiaoqiang Lu. 2017. A general framework for edited video and raw video summarization. *IEEE Transactions on Image Processing* 26, 8 (2017), 3652–3664.

[14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3431–3440.

[15] Mingyang Ma, Shaohui Mei, Jingyu Ji, Shuai Wan, Zhiyong Wang, and Dagan Feng. 2017. Exploring the influence of feature representation for dictionary selection based video summarization. In *ICIP*. IEEE, 2911–2915.

[16] Mingyang Ma, Shaohui Mei, Shuai Wan, Zhiyong Wang, and Dagan Feng. 2019. Video Summarization via Nonlinear Sparse Dictionary Selection. *IEEE Access* 7 (2019), 11763–11774.

[17] Mingyang Ma, Shaohui Mei, Shuai Wan, Zhiyong Wang, and David Dagan Feng. 2018. Robust video summarization using collaborative representation of adjacent frames. *Multimedia Tools and Applications* (2018), 1–21.

[18] Behrooz Mahasseni, Michael Lam, and Sinisa Todorovic. 2017. Unsupervised video summarization with adversarial lstm networks. In *CVPR*, Vol. 1.

[19] Shaohui Mei, Genliang Guan, Zhiyong Wang, Shuai Wan, Mingyi He, and David Dagan Feng. 2015. Video summarization via minimum sparse reconstruction. *Pattern Recognition* 48, 2 (2015), 522–533.

[20] MerchDope 2019. 37 Mind Blowing YouTube Facts, Figures and Statistics C 2019. https://merchdope.com/youtube-stats/.

[21] Joe Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. 2015. Beyond short snippets: Deep networks for video classification. In *CVPR*. 4694–4702.

[22] OVP 2011. Open video project. https://open-video.org/.

[23] Danila Potapov, Matthijs Douze, Zaid Harchaoui, and Cordelia Schmid. 2014. Category-specific video summarization. In *ECCV*. 540–555.

[24] Mrigank Rochan, Linwei Ye, and Yang Wang. 2018. Video Summarization Using Fully Convolutional Sequence Networks. *arXiv preprint arXiv:1805.10538* (2018).

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision* 115, 3 (2015), 211–252.

[26] Alan F Smeaton, Paul Over, and Wessel Kraaij. 2006. Evaluation campaigns and TRECVid. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*. ACM, 321–330.

[27] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. 2015. Tvsum: Summarizing web videos using titles. In *CVPR*. 5179–5187.

[28] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *CVPR*. 1–9.

[29] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*. 4489–4497.

[30] Ba Tu Truong and Svetha Venkatesh. 2007. Video abstraction: A systematic review and classification. *ACM transactions on multimedia computing, communications, and applications* 3, 1 (2007), 3.

[31] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. 2015. Sequence to Sequence – Video to Text. In *2015 IEEE International Conference on Computer Vision (ICCV)*. 4534–4542.

[32] Junbo Wang, Wei Wang, Yan Huang, Liang Wang, and Tieniu Tan. 2018. M3: Multimodal memory modelling for video captioning. In *CVPR*. 7512–7520.

[33] Huawei Wei, Bingbing Ni, Yichao Yan, Huanyu Yu, Xiaokang Yang, and Chen Yao. 2018. Video Summarization via Semantic Attended Networks.. In *AAAI*.

[34] Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *ICML*. 2397–2406.

[35] Huan Yang, Baoyuan Wang, Stephen Lin, David Wipf, Minyi Guo, and Baining Guo. 2015. Unsupervised extraction of video highlights via robust recurrent auto-encoders. In *ICCV*. 4633–4641.

[36] Ting Yao, Tao Mei, and Yong Rui. 2016. Highlight detection with pairwise deep ranking for first-person video summarization. In *CVPR*. 982–990.

[37] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. 2016. Summary transfer: Exemplar-based subset selection for video summarization. In *CVPR*. 1059–1067.

[38] Ke Zhang, Wei-Lun Chao, Fei Sha, and Kristen Grauman. 2016. Video summarization with long short-term memory. In *ECCV*. 766–782.

[39] Ke Zhang, Kristen Grauman, and Fei Sha. 2018. Retrospective Encoders for Video Summarization. In *ECCV*. 383–399.

[40] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. 2017. Hierarchical recurrent neural network for video summarization. In *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 863–871.

[41] Bin Zhao, Xuelong Li, and Xiaoqiang Lu. 2018. HSA-RNN: Hierarchical Structure-Adaptive RNN for Video Summarization. In *CVPR*. 7405–7414.

[42] Bin Zhao and Eric P Xing. 2014. Quasi real-time summarization for consumer videos. In *CVPR*. 2513–2520.

[43] Kaiyang Zhou, Yu Qiao, and Tao Xiang. 2018. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In *AAAI*.