Retrospective Encoders for Video Summarization

Ke Zhang[1], Kristen Grauman[2], and Fei Sha[3](✉)

[1] Department of Computer Science, University of Southern California,
Los Angeles, CA 90089, USA
zhang.ke@usc.edu

[2] Facebook AI Research, 300 W. Sixth St., Austin, TX 78701, USA
grauman@fb.com

[3] Netflix, 5808 Sunset Blvd, Los Angeles, CA 90028, USA
fsha@netflix.com

Abstract. Supervised learning techniques have shown substantial progress on video summarization. State-of-the-art approaches mostly regard the predicted summary and the human summary as two sequences (sets), and minimize discriminative losses that measure element-wise discrepancy. Such training objectives do not explicitly model how well the predicted summary preserves semantic information in the video. Moreover, those methods often demand a large amount of human generated summaries. In this paper, we propose a novel sequence-to-sequence learning model to address these deficiencies. The key idea is to complement the discriminative losses with another loss which measures if the predicted summary preserves the same information as in the original video. To this end, we propose to augment standard sequence learning models with an additional "retrospective encoder" that embeds the predicted summary into an abstract semantic space. The embedding is then compared to the embedding of the original video in the same space. The intuition is that both embeddings ought to be close to each other for a video and its corresponding summary. Thus our approach adds to the discriminative loss a metric learning loss that minimizes the distance between such pairs while maximizing the distances between unmatched ones. One important advantage is that the metric learning loss readily allows learning from videos without human generated summaries. Extensive experimental results show that our model outperforms existing ones by a large margin in both supervised and semi-supervised settings.

Keywords: Video summarization · Sequence-to-sequence learning

K. Grauman—On leave from University of Texas at Austin (grauman@cs.utexas.edu)
F. Sha—On leave from University of Southern California (feisha@usc.edu).

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01237-3_24) contains supplementary material, which is available to authorized users.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11212, pp. 391–408, 2018.
https://doi.org/10.1007/978-3-030-01237-3_24

1 Introduction

The amount of online video data is staggering: hundreds of hours of videos are uploaded to YouTube every minute [2], video posts on Facebook have been increasing by more than 90% annually, and by Cisco's estimate, traffic from online videos will constitute over 80% of all consumer Internet traffic by 2020.

As such, there has been a growing interest in automatic video summarization. The main objective is to shorten a video while still preserving the important and relevant information it contains. A shortened video is more convenient and efficient for both interactive use (such as exploratory browsing), fast indexing and matching (such as responding to search queries). To this end, a common type of summary is composed of a selected set of frames (*i.e.* keyframes) [15,19,30,32,40,40,50,63], segments, or subshots (*i.e.* keyshots) [29,34,41,43]. Other formats are possible [12,14,25,46,51], though they are not the focus of this work.

Many methods for summarization have been proposed and studied. Among them, supervised learning based techniques have recently gained significant attention [6,15,18,19,49,54,64–66]. As opposed to unsupervised ones [21,25,26,30,33,34,36,38,43,63], supervised techniques explicitly maximize the correspondence between the automatically generated summary and the human created one. As such, those techniques often achieve higher performance metrics.

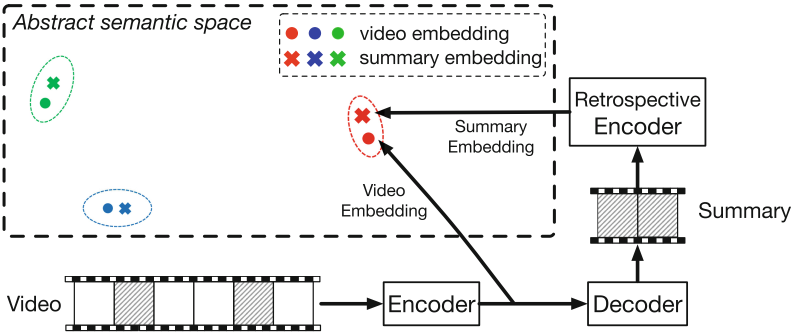


Fig. 1. Conceptual diagram of our approach. Our model consists of two components. First, we use SEQ2SEQ models to generate summaries. Secondly, we introduce an retrospective encoder that maps the generated summaries to an abstract semantic space so that we can measure how well the summaries preserve the information in the original videos, *i.e.* the summary (×) should be close to its original video (•) while distant from other videos ({•, •}) and summaries ({×, ×}). In this semantic space, we derive metric learning based loss functions and combine them with the discriminative loss by matching human generated summaries and predicted ones. See text for details. (Color figure online)

In particular, recent work on applying sequence-to-sequence learning techniques to video summarization has introduced several promising models [24,38,64–66]. Viewing summarization as a structured prediction problem, those techniques model the long-range dependency in video using the popular long short-term memory units (LSTM) and its variants [7,17,20]. The key idea is to maximize the accuracy of which frames or subshots are selected in the summary. Figure 2 shows the basic concepts behind these modeling techniques to which we collectively refer as SEQ2SEQ.

The conventional overlap accuracy is a useful surrogate for measuring how good the generated summaries are. However, it suffers from several flaws. First, it emphasizes equally the local correspondence between human and machine summaries on all frames or subshots. For instance, for a video of a soccer game, while arguably the moment around a goal-shot is likely in every human annotator's summary, whether or not other less critical events (before or after the shot) are included could be quite varying – for example, subshots showing running across different sections of the play-field are equally good (or bad). Thus modeling those subshots in the summary is not always useful and necessary. Instead, we ought to assess whether the summary "holistically" preserves the *most important and relevant* information in the original video.

The second difficulty of employing overlap accuracy (and thus to a large degree, supervised learning techniques) is the demand of time-consuming and labor-intensive annotation procedures, which has been a limiting factor of existing datasets, cf. [65]. Thus supervised techniques have limited applicability when the annotated data is scarce.

To address these flaws, we propose a new sequence learning model—retrospective sequence-to-sequence learning (*re*-SEQ2SEQ). The key idea behind *re*-SEQ2SEQ is to measure how well the machine-generated summary is similar to **the original video** in an abstract semantic space.

Specifically, as the original video is processed by the encoder component of a SEQ2SEQ model, the encoder outputs a vector embedding which represents **the semantic meaning of the original video**. We then pass the outputs of the decoder, which should yield the desired summary, to a *retrospective* encoder to infer a vector embedding to represent **the semantic meaning of the summary**. If the summary preserves the important and relevant information in the original video, then we should expect that the two embeddings are similar (e.g. in Euclidean distance). Figure 1 schematically illustrates the idea. Besides learning to "pull" the summary close to the original video, our model also "pushes far away" the embeddings that do not form corresponding pairs.

The measure of similarity (or distance) is combined with the standard loss function (in SEQ2SEQ models) that measures how well the summary aligns locally on the frame/shot-level with what is provided by human annotators. However, the proposed learning of similarity in the abstract semantic space provides additional benefits. Since it does not use any human annotations, our measure can be computed on videos without any "ground-truth" summaries. This provides a natural basis for semi-supervised learning where we can leverage the large amount of unlabeled videos to augment training.

To summarize, our contributions are: (i) a novel sequence learning model for video summarization, which combines the benefits of discriminative learning by aligning human annotation with the model's output and semi-supervised/unsupervised learning which ensure the model's output is in accordance with the original video by embedding both in close proximity; (ii) an extensive empirical study demonstrating the effectiveness of the proposed approach on several benchmark datasets, and highlighting the advantages of using unlabeled data to improve summarization performance.

2 Related Work

Unsupervised video summarization methods mostly rely on manually designed criteria [8,11,21,25,27,30,31,33,34,36,43,45,50,63,67], e.g. importance, representativeness, and diversity. In addition, auxiliary cues such as web images [21, 26,27,50] or video categories [44,45] are also exploited in the unsupervised (weakly-supervised) summarization process.

Supervised learning for video summarization has made significant progress [6, 15,18,19,64]. Framing the task as a special case of structured prediction, Zhang *et al.* [65] proposed to use sequence learning methods, and in particular, sequence-to-sequence models [7,10,52] that have been very successful in other structured prediction problems such as machine translation [22,23,35,47,58], image or video caption generation [55,57,59], parsing [56], and speech recognition [4].

Several extensions of sequence learning models have since been studied [24,38,66,68]. Yang *et al.* [60] and Mahasseni *et al.* [38] bear a somewhat similar modeling intuition as our approach. In their works, the model is designed such that the video highlight/summary (as a sequence) can generate another (video) sequence similar to the original video. However, this desideratum is very challenging to achieve. In particular, the mapping from video to summary is lossy, making the reverse mapping almost unattainable: for instance, the objects that are in the discarded frames but missing from the summarization frames cannot be reliably recovered from the summary alone. In contrast, our model has a simpler architecture, and contains fewer LSTMs units (thus fewer parameters); our approach only desires that the *embeddings* of human created and predicted summaries be close. It attains better results than those reported in [38].

Zhou *et al.* [68] propose to use reinforcement learning to model the sequential decision-making process of selecting frames as a summary. While interesting, the design of reward functions with heuristic criteria can be as challenging as using unsupervised methods for summarization. It shows minor gains over the less competitive model in the fully supervised learning model of [65]. Both Zhao *et al.* [66] and Ji *et al.* [24] introduce hierarchical LSTMs and attention mechanisms for modeling videos. They focus on maximizing the alignment between the generated summaries and the human annotators'. Our approach also uses hierarchical LSTMs but further incorporates objectives to match the generated summaries and the original videos, *i.e.* aiming to preserve important information in the original videos. The experimental study demonstrates the advantages.

3 Approach

We start by stating the setting for the video summarization task, introducing the notations and (briefly) the background on sequence learning with LSTMs [20,39,52,61]. We then describe the proposed *retrospective encoder sequence-to-sequence (re-*SEQ2SEQ*)* approach in detail. The model extends the standard encoder-decoder LSTM by applying an additional encoder on the outputs of the decoder and introducing new loss functions.

3.1 Setting, Notations, and Sequence Learning Models

We represent a video as a sequence $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{\mathsf{T}}\}$ where $\boldsymbol{x}_t, t \in 1, \cdots, T$, is the feature vector characterizing the t-th frame in the video. We denote (sub)shots in the video by $\boldsymbol{B} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_{\mathsf{B}}\}$. Each of the B shots refers to a consecutive subset of $\boldsymbol{X}$, and doesn't overlap with others.

Video Summarization Task. The task is to select a subset of shots as the summary, denoted as $\boldsymbol{Y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_{\mathsf{L}}\}$ where $\boldsymbol{y}_l$ indicates the feature vector for the l-th shot in the summary. Obviously we desire $\mathsf{L} < \mathsf{B} < \mathsf{T}$. The ground-truth keyshots are denoted as $\boldsymbol{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_{\mathsf{L}}\}$.

When $\boldsymbol{B}$ is not given (which is common in most datasets for the task), we use a shot-boundary detection model to infer the boundaries of the shots. This leads to misaligned shot boundaries between what is given in the ground-truth keyshots and what is inferred. We discuss how we handle this in the experimental details and in the Suppl. For clarity, we assume $\boldsymbol{B}$ is known and given throughout this section.

Sequence Learning. Long short-term memory (LSTMs) are a special kind of recurrent neural networks that are adept at modeling long-range dependencies. They have been used to model temporal (and sequential) data very successfully [13,20,61]. An LSTM has time-varying memory state variables $\boldsymbol{c}_t$ and an output variable $\boldsymbol{h}_t$. The values of those variables depend on the current input, past outputs, and memory states. The details of the basic LSTM cells that we use in this paper are documented in the Suppl. and [16,65].

SEQ2SEQ typically consists of a pair of LSTMs—the encoder and the decoder—which are combined to perform sequence transduction [3,55]. Specifically, the encoder reads the input sequence $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_{\mathsf{T}}\}$ sequentially (or in any predefined order), and calculates a sequence of hidden states $\boldsymbol{H} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_{\mathsf{T}}\}$. Each hidden state $\boldsymbol{h}_t$ at step t feeds into the next step at $(t+1)$. The last hidden state $\boldsymbol{h}_{\mathsf{T}}$ feeds into the decoder. The decoder is similar to the encoder except for two changes: (1) the decoder has its own hidden state sequence $\boldsymbol{V} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_{\mathsf{L}}\}$; (2) the decoder does not have external input. Instead, its input (*i.e.* its version of $\boldsymbol{x}_t$) at step $(l+1)$ is now its own output at step l, given by $\boldsymbol{y}_l = g(\boldsymbol{y}_{l-1}, \boldsymbol{v}_l)$. Note that the function $g(\cdot)$ as well as other parameters in the encoder/decoder are learnt from data. Figure 2 illustrates these steps, though in the context of hierarchical LSTMs.

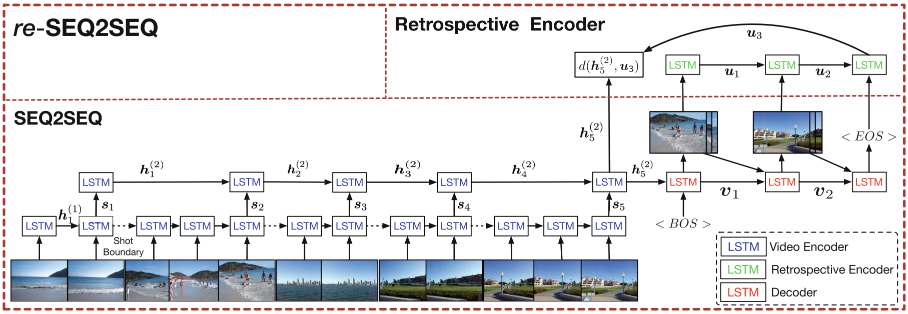


Fig. 2. Our proposed approach for video summarization. The model has several distinctive features. First, it uses hierarchical LSTMs: a bottom layer of LSTMs models shots composed of frames and an upper-layer LSTM models the video composed of shots. Secondly, the model has a *retrospective encoder* (green) that computes the embeddings of the outputs of the decoder (red). The training objective for the retrospective encoder is to ensure the embedding of the summary outputs matches the embedding computed from the original inputs, cf. Eq. (5). (Color figure online)

3.2 Retrospective-Encoder Sequence-to-sequence (*re*-seq2seq)

We propose several important extensions to the original SEQ2SEQ model: hierarchical modeling of frames and shots, and new loss functions for training.

Hierarchical Modeling. As shown in [42,55,66], the ideal length of video for LSTM modeling is less than 100 frames despite LSTMs' ability to model long-range dependencies. Thus, it is challenging to model long videos. To this end, we leverage the hierarchical structures in a video [66] to capture dependencies across longer time spans.

As shown in Fig. 2, there are two encoder layers made of LSTM units for modeling frames and shots, respectively. The first layer is responsible for modeling at the frame level and yielding a representation for all the frames in the current shot. This representation is then fed as input to the second LSTM layer. The final output of the second layer is then treated as the embedding for the whole video as it combines all the information from all the shots.

Concretely, for the first layer LSTMs, the input is $\boldsymbol{x}_t$, the feature vector for the tth frame. Assuming t is within a shot $\boldsymbol{b}_b$, the hidden state of this layer's LSTM unit is $\boldsymbol{h}_t^{(1)}$, encoding all frames from the beginning of the shot boundary by computing over the current feature $\boldsymbol{x}_t$ and the previous hidden state $\boldsymbol{h}_{t-1}^{(1)}$.

When t passes the shot's ending boundary, we denote the final hidden state of the LSTM unit as $\boldsymbol{s}_b$, the encoding vector for the current shot $\boldsymbol{b}_b$. The LSTM unit's memory $\boldsymbol{c}_t^{(1)}$ and the initial hidden state $\boldsymbol{h}_t^{(1)}$ are then reset to $\boldsymbol{c}_0^{(1)}$ and $\boldsymbol{h}_0^{(1)}$ which are both zero vectors in our model (one can also learn them).

After all the frames are processed, we have a sequence of encodings $\boldsymbol{S} = \{\boldsymbol{s}_b\}$ for $b = 1, 2, \cdots, \mathsf{B}$. We construct another LSTM layer over $\boldsymbol{S}$. The hidden states of this layer are denoted by $\boldsymbol{h}_b^{(2)}$. Of particular importance is $\boldsymbol{h}_{\mathsf{B}}^{(2)}$ which is regarded as the encoding vector for the whole video. (One can also introduce more layers for finer-grained modeling [9], and we leave that for future work.)

The decoder layer is similar to the one in a standard non-hierarchical LSTM (cf. Fig. 2). It does not have any input (except $\boldsymbol{h}_{\mathsf{B}}^{(2)}$ initializes the first LSTM unit in the layer), and its output is denoted by $\boldsymbol{y}_l$ for $l = 1, 2, \cdots, \mathsf{L}$. Its hidden states are denoted by $\boldsymbol{v}_l$, which is a function of the previous hidden state $\boldsymbol{v}_{l-1}$ and output $\boldsymbol{y}_{l-1}$. The output $\boldsymbol{y}_l$ is parameterized as a function of $\boldsymbol{v}_l$.

Regression Loss Function for Matching Summaries. In supervised learning for video summarization, we maximize the accuracy of whether a particular frame/subshot is selected. To this end, one would set the output of the decoder as a binary variable and use a cross-entropy loss function, cf. [65] for details.

In our model, however, we intend to treat the outputs as a shortened video sequence and regard $\boldsymbol{y}_l$ as visual feature vectors (of shots). Thus, the cross-entropy loss function is not applicable. Instead, we propose the following *regression loss*

$$\ell_{\text{SUMMARY}} = \sum_{i}^{\mathsf{L}} \|\boldsymbol{y}_l - \boldsymbol{g}_l\|_2^2, \tag{1}$$

where $\boldsymbol{g}_l$ is a target vector corresponding to the l-th shot in the ground-truth summary $\boldsymbol{Z}$. Suppose this shot corresponds to the b_l-th shot in the sequence of shots $\boldsymbol{B}$, where b_l is between 1 and B. We then compose $\boldsymbol{g}_l$ as the concatenation of two vectors: (1) the encoding $\boldsymbol{s}_{b_l}$ of the b_l-th shot as computed by the first LSTM layer, and (2) the average of frame-level vectors $\boldsymbol{x}_t$ within the b_l-th shot. We use $\bar{\boldsymbol{x}}_{b_l}$ to denote this average. $\boldsymbol{g}_l$ is thus given by

$$\boldsymbol{g}_l = [\boldsymbol{s}_{b_l} \quad \bar{\boldsymbol{x}}_{b_l}]. \tag{2}$$

This form of $\boldsymbol{g}_l$ is important. While the goal is to generate outputs $\boldsymbol{y}_l$ closely matching the encodings of shots, we could obtain trivial solutions where the LSTMs learn to encode shots as a constant vector and to output a constant vector as the summary. The inclusion of the video's raw feature vectors in the learning effectively eliminates trivial solutions.

Embedding Matching for the Summary and the Original. The intuition behind our modeling is that the outputs should convey the same amount of information as the inputs. For summarization, this is precisely the goal: a good summary should be such that after viewing the summary, users would get about the same amount of information as if they had viewed the original video.

How to measure and characterize the amount of information conveyed in the original sequence and the summary? Recall that the basic assumption about the encoder LSTMs is that they compress semantic information of their inputs into a semantic embedding, namely $\boldsymbol{h}_{\mathsf{B}}^{(2)}$, the final hidden state[1]. Likewise, if we add another encoder to the decoder output $\boldsymbol{Y}$, then this new encoder should also be

[1] Otherwise, we would not have expected $\boldsymbol{h}_{\mathsf{B}}^{(2)}$ to be able to generate the summary to begin with.

able to compress it into a semantic embedding with its final hidden state $\boldsymbol{u}_{\mathsf{L}}$. Figure 2 illustrates the new *re*-SEQ2SEQ model structure.

To learn this "retrospective" encoder, we use the following loss

$$\ell_{\text{MATCH}} = \|\boldsymbol{h}_{\mathsf{B}}^{(2)} - \boldsymbol{u}_{\mathsf{L}}\|_2^2. \tag{3}$$

Contrastive Embedding for Mismatched Summary and Original. We can further model the alignment between the summary and the original by adding penalty terms that penalize mismatched pairs:

$$\begin{aligned}\ell_{\text{MISMATCH}} = &\sum_{\boldsymbol{h}'}[m + \|\boldsymbol{h}_{\mathsf{B}}^{(2)} - \boldsymbol{u}_{\mathsf{L}}\|_2^2 - \|\boldsymbol{h}' - \boldsymbol{u}_{\mathsf{L}}\|_2^2]_+ \\ &+ \sum_{\boldsymbol{u}'}[m + \|\boldsymbol{h}_{\mathsf{B}}^{(2)} - \boldsymbol{u}_{\mathsf{L}}\|_2^2 - \|\boldsymbol{h}_{\mathsf{B}}^{(2)} - \boldsymbol{u}'\|_2^2]_+,\end{aligned} \tag{4}$$

where $\boldsymbol{h}'$ (or $\boldsymbol{u}'$) is the hidden state in the shot-level LSTM layer (or the re-encoder LSTM layer) from a video other than $\boldsymbol{h}_{\mathsf{B}}^{(2)}$ (or a summary other than $\boldsymbol{u}_{\mathsf{L}}$). $m > 0$ is a margin parameter and $[\cdot]_+ = \max(0, \cdot)$ is the standard hinge loss function. In essence this loss function aims to push apart mismatched summaries and videos.

Final Training Objective. We train the models by balancing the different types of loss functions

$$\ell = \ell_{\text{SUMMARY}} + \lambda \ell_{\text{MATCH}} + \eta \ell_{\text{MISMATCH}}, \tag{5}$$

where the λ and η are tradeoff parameters.

Note that neither ℓ_{MATCH} nor ℓ_{MISMATCH} requires human annotated summaries. Thus they can also be used to incorporate unannotated video data which are disjoint from the annotated data from which ℓ_{SUMMARY} is computed. Our empirical results will show that learning with these two loss terms noticeably improves learning with only the discriminative loss.

The specific forms of ℓ_{MATCH} and ℓ_{MISMATCH} are also reminiscent of metric learning (for multi-way classification) [5]. In particular, we transform both the summary and the video into vectors (through a series of encoder/decoder LSTMs) and perform metric learning in that abstract semantic space. However, different from traditional metric learning methods, we also need to learn to infer the desired representations of the structured objects (*i.e.* sequences of frames).

Implementation Details. Throughout our empirical studies, we use standard LSTM units. The dimensions of the hidden units for all LSTM units are 256. All LSTM parameters are randomly initialized with a uniform distribution in $[-0.05, 0.05]$. The models are trained to converge with Adam [28] with an initial learning rate $4e-4$, and mini-batch size of 10. All experiments are on a single GPU. Please refer to the Suppl. for more details.

4 Experiments

We first introduce the experimental setting (datasets, features, metrics) in Sect. 4.1. We then present the main quantitative results in the supervised learning setting to demonstrate the advantages of the proposed approach over existing methods in Sect. 4.2. We further validate the proposed approach in the semi-supervised setting in Sect. 4.3. We perform ablation studies and analyze strengths and weaknesses of our approach in Sects. 4.4 and 4.5.

4.1 Setup

Datasets. We evaluate on 3 datasets. The first two have been widely used to benchmark summarization tasks [24,38,65,68]: **SumMe** [19] and **TVSum** [50]. **SumMe** consists of 25 user videos of a variety of events such as holidays and sports. **TVSum** contains 50 videos downloaded from YouTube in 10 categories. Both datasets provide multiple user-annotated summaries per video.

Following [65], we also use **Youtube** [11] and **Open Video Project (OVP)** [1,11] as auxiliary datasets to augment the training data. We use the same set of features, *i.e.* each frame $\boldsymbol{x}_i$ is represented by the output of the penultimate layer (pool 5) of GoogLeNet [53] (1024 dimensions). As pointed out in [65], the limited amount of annotated data limits the applicability of supervised learning techniques. Thus, we focus on the **augmented setting** (if not specified otherwise) as in that work and other follow-up ones[2]. In this setting, 20% of the dataset is used for evaluation, 20% is used for validation (in our experiments) and the other 60% is combined with auxiliary datasets for training. We tune hyperparameters, e.g. λ and η, w.r.t. the performance on the validation set.

We also demonstrate our model on a third dataset **VTW** [62] which is large-scale and originally proposed for video highlight detection. In [66], it is re-targeted for video summarization by converting the highlights into keyshots. VTW collects user-generated videos which are mostly shorter than ones in SumMe and TVSum. The dataset is split into 1500 videos for training and 500 videos for testing as in [66]. We have not been able to confirm the split details so our results in this paper are not directly comparable to their reported ones.

Shot Boundary Generation. As mentioned in Sect. 3, shot boundaries for each video are required for both training and testing. However, none of the datasets used in the experiments are annotated with ground-truth shot boundaries: VTW is annotated only for keyshots, SumMe is annotated by multiple users for keyshots, and TVSum is annotated by fix-length intervals (2 s). To this end, we train a single-layer LSTM for shot-boundary detection with another *disjoint* dataset, CoSum [8]. CoSum has 51 videos with human annotated shot-boundaries. The LSTM has 256−dim hidden units and is trained for 150 epochs

[2] Zhao *et al.* [66] use a larger dataset MED [45] to augment the training, which results in a larger number of videos (235), as opposed to 154 video in [24,38,65,68]. Since their code is unavailable, we have re-implemented their method to the best of our knowledge from their paper and experimented in the same setting as ours and others.

with learning rate $4e-4$. We then threshold the predictions on any new videos to detect the boundaries. The best thresholds are determined by the summarization performance on the validation set. Please refer to details on shot boundary detection in the Suppl.

Evaluation. Given the heterogeneous video types and summary formats in these datasets, we follow the procedures outlined in [65,66] to prepare training, validation, and evaluation data. In particular, we set the threshold of the total duration of keyshots as 15% of the original video length (for all datasets), following the protocols in [19,50,66]. Then we compare the generated summary A to the user summary B for evaluation, by computing the precision (P) and recall (R), according to the *temporal overlap* between the two, as well as their harmonic mean F-score [18,19,38,50,65,66]. Higher scores are better. Please refer to the Suppl. for details on the performance and evaluation with user summaries.

Table 1. Performance (F-score) of various supervised video summarization methods on three datasets. Published results are denoted in *italic*; our implementations are in normal font. Nonzero values of λ and η represent contributions from relevant terms in our models.

	SumMe	TVSum	VTW
dppLSTM [65]	*42.9*	*59.6*	44.3
SUM-GAN [38]	*43.6*	*61.2*	-
DR-DSN [68]	*43.9*	*59.8*	-
H-RNN [66]	43.6	61.5	46.9
SEQ2SEQ (frame only)	40.8	56.3	-
re-SEQ2SEQ ($\lambda=0, \eta=0$)	43.2	61.9	45.1
re-SEQ2SEQ ($\lambda=\lambda^*, \eta=\eta^*$)	**44.9**	**63.9**	**48.0**

4.2 Supervised Learning Results

In Table 1, we compare our approach to several state-of-the-art supervised methods for video summarization. We report published results in the table as well as results from our implementation of [66]. Only the best variants of all methods are quoted and presented. We have implemented a strong baseline *re*-SEQ2SEQ ($\lambda=0, \eta=0$), trained with the objective function Eq. (5) described in Sect. 3. This baseline is different from the LSTM-based SEQ2SEQ models in [65] where the model is frame-based, to which we refer as SEQ2SEQ (frame only). *re*-SEQ2SEQ ($\lambda=0, \eta=0$) has the advantage of hierarchical modeling. Optimal λ^* and η^* are tuned over the validation set and $\lambda^*=0.1, 0.1, 0.2$ and $\eta^*=0.15, 0.1, 0.2$ for SumMe, TVSum, and VTW, respectively. The cells with red-colored numbers indicate the best performing methods in each column.

Main Results. Our approach *re*-SEQ2SEQ ($\lambda=\lambda^*, \eta=\eta^*$) performs the best on all 3 datasets. Hierarchical modeling is clearly advantageous, evidenced by the performance of all our model variations and [66]. Our model *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$) is slightly worse than [66], most likely due to the fact we use regression as summary loss while they use cross-entropy. Note that regression loss is needed in order to incorporate matching and mismatching losses in our model, cf. Eq. (5). The advantage of incorporating the retrospective encoder loss is clearly evidenced.

Table 2. F-scores on the TVSum dataset in the semi-supervised learning setting. n indicates the number of unannotated videos used for training.

	$n = 0$	$n = 150$	$n = 500$	$n = 1000$	$n = 1500$	$n = 1800$
pre-training	63.9	64.1	64.4	64.5	64.7	64.9
joint-training		64.1	64.7	64.9	65.1	**65.2**

Table 3. Performance of our model with different types of shot boundaries.

	SumMe	TVSum
re-SEQ2SEQ ($\lambda = \lambda^*, \eta = 0$) w/KTS	44.5	62.8
re-SEQ2SEQ ($\lambda = \lambda^*, \eta = 0$) w/LSTM	44.6	63.0
re-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) w/KTS	44.8	63.6
re-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) w/LSTM	**44.9**	**63.9**

4.3 Semi-Supervised Learning Results

Next we carry out experiments to show that the proposed approach can benefit from both unlabeled and labeled video data. For labeled data, we use the same train and test set as in the augmented setting, *i.e.* OVP + Youtube + SumMe + 80% TVSum, and 20% TVSum, respectively. For unlabeled data, we randomly sample n videos from the VTW dataset and ignore their annotations. We investigate two possible means of semi-supervised training:

(1) *pre-training*: the unlabeled data are used to pre-train *re*-SEQ2SEQ to minimize ℓ_{MATCH} and ℓ_{MISMATCH} only. The pre-trained model is further fine-tuned with labeled training data to minimize Eq. (5).

(2) *joint-training*: We jointly train the model with labeled training data and unlabeled data: we minimize Eq. (5) for labeled data, and minimize ℓ_{MATCH} and ℓ_{MISMATCH} for unlabeled data.

Note that the test set is only for testing and not used as either labeled or unlabeled data during training, which is different from the transductive setting [38]. Results are shown in Table 2. In general, both *pre-training* and *joint-training* show improvements over supervised learning, and joint-training seems slightly better with more unlabeled data. Results are also encouraging in showing that more unlabeled data can help improve more.

Table 4. Performances of transductive setting on SumMe and TVSum.

	SumMe	TVSum
SUM-GAN [38]	*43.6*	*61.2*
re-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$)	**45.5**	**65.4**

Table 5. Performance of the proposed approach with different choices of λ and η

	SumMe	TVSum	VTW
re-SEQ2SEQ ($\lambda = 0, \eta = 0$)	43.2	61.9	45.1
re-SEQ2SEQ ($\lambda = \lambda^*, \eta = 0$)	44.6	63.0	47.7
re-SEQ2SEQ ($\lambda = 0, \eta = \eta^*$)	44.6	63.2	47.8
re-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$)	**44.9**	**63.9**	**48.0**

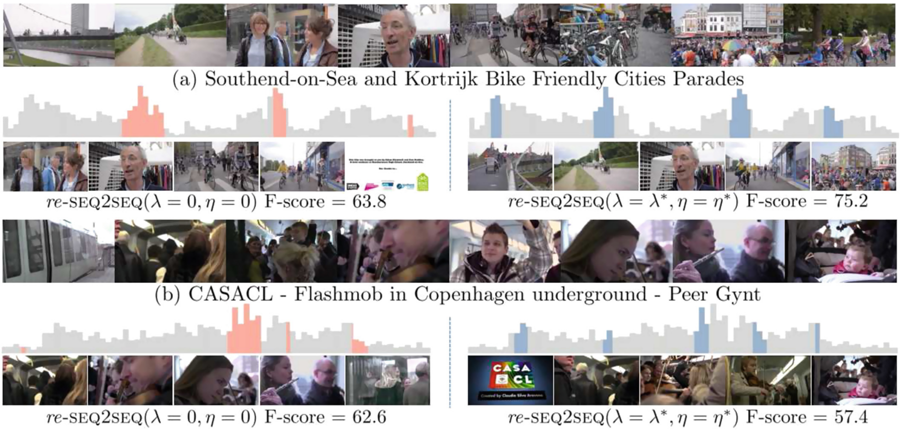
4.4 Ablation Study

Shot-Boundaries. Shot boundaries play an important role in keyshot-based summarization methods. In this paper we learn an LSTM to infer the shot boundaries, while in [65] an unsupervised shot boundary detection method, KTS [45], is applied. Table 3 reports the performances of our model with shot boundaries generated by KTS and the learned LSTM, respectively. The main observation is better shot boundary detection in general improves summarization.

Transductive Setting. To make a fair comparison to [38], we next perform our model in the transductive setting, where the testing data are included in computing the two new loss terms ℓ_{MATCH} and ℓ_{MISMATCH}. The results are shown in Table 4 and they are clearly stronger than those in the supervised setting (Table 1). One possible interpretation for this case is that our model maps the video and its summary in close proximity while the reconstruction from a summary to the original video in [38] may be lossy or even unattainable.

Contributions of Each Loss Term. Table 5 reports experiments of the proposed approach with different combinations of ℓ_{MATCH} and ℓ_{MISMATCH} through their balancing parameters, *i.e.* λ and η. To summarize, jointly minimizing both loss terms brings the state-of-the-art performance on all datasets. Furthermore, the performances of different combinations of loss terms are consistent across the 3 datasets: using ℓ_{MISMATCH} alone gets the same or slightly better performances compared to using ℓ_{MATCH} alone, while combining both of them always obtains the best performance. Please refer to the Suppl. for model details.

Other Detailed Analysis in Suppl. We summarize additional discussions as follows. We show that summaries by our approach obtain comparable diversity to ones by dppLSTM [65]. We also show that our approach outperforms the autoencoder-based method [60]. We further analyze the correlation between our approach and recent works [48,49] on the query-focused summarization.



(a) Southend-on-Sea and Kortrijk Bike Friendly Cities Parades

re-SEQ2SEQ($\lambda = 0, \eta = 0$) F-score = 63.8

re-SEQ2SEQ($\lambda = \lambda^*, \eta = \eta^*$) F-score = 75.2

(b) CASACL - Flashmob in Copenhagen underground - Peer Gynt

re-SEQ2SEQ($\lambda = 0, \eta = 0$) F-score = 62.6

re-SEQ2SEQ($\lambda = \lambda^*, \eta = \eta^*$) F-score = 57.4

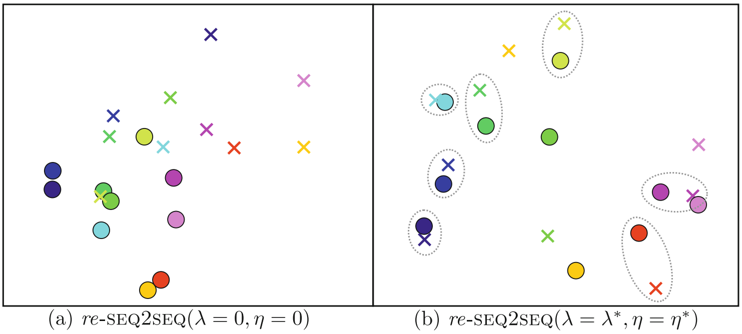
Fig. 3. Exemplar videos and predicted summaries by *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) (blue) and *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$) (red). Pictures on the top are sampled from the video and ones in the bottom are sampled from the corresponding summary. The ground-truth importance scores are shown as gray background. See text for details. (Color figure online)

4.5 Qualitative Results and Analysis

How Does *re*-seq2seq Summarize Differently than Regular seq2seq? We examine a few exemplar video summarization results in Fig. 3 to shed light on how the learning objective of *re*-SEQ2SEQ affects summarization results. Our approach aims to reduce the difference in semantic embeddings for both input videos and output summaries, cf. Eq. (5). Balancing the need to match between the outputs and human summaries, it would be sensible for our approach to summarize broadly. This will result in comprehensive coverage of visual features, and thus increase the chance of more similar embeddings. The opposite strategy of selecting from a concentrated area is unlikely to yield high similarity as the selected frames are unlikely to provide a sufficient coverage of the original.

Figure 3 precisely highlights the strategy adopted by our approach. The video of Fig. 3(a) is about bike parades. *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$) summarizes the mid-section of the video, but completely misses the important part in the beginning, which tells us the parades actually start from suburb via a bridge to downtown. In contrast, *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) selects broadly from video shots, which show much better consensus with the video. In Fig. 3(b), however, *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) underperforms (slightly) *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$). The video depicts a flash mob in Copenhagen. *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$) gets a better F-score by focusing on the mid-section of the video where there are a lot of human activities and is able to correctly get the major events in that region. *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$), on the other hand, spreads out its selection and takes only a small part of the major event compared to *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$).

While more error analysis is desirable, these preliminary evidences seem to suggest that *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) would work well for videos that depict various scenes and activities that follow a storyline. In particular, it might not work well with "bursty videos" where there are interesting but short shots of videos scattered in the middle of a large number of frames with non-essential information likely to be discarded when summarized.



(a) *re*-SEQ2SEQ($\lambda = 0, \eta = 0$)

(b) *re*-SEQ2SEQ($\lambda = \lambda^*, \eta = \eta^*$)

Fig. 4. t-SNE visualization of semantic encodings of videos (denoted as •) and their summaries (denoted as ×). Corresponding pairs are in the same color. The closer they are the better. Each dashed ellipsoid indicates that a video is the nearest neighbor to its summary after embedding. See text for details.

Can *re*-seq2seq Lead to Semantically Similar Embeddings Between the Video and Summary? Here we evaluate how well the video and summary can be embedded in close proximity. Videos used here are sampled from the TVSum dataset. For *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$), we input the summary to the same encoder as for videos and obtain the outputs of the encoder as the embedding, and in *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$), we collect the embedding for the summary from the retrospective LSTM encoder. We then use t-SNE [37] to visualize the embeddings in $2d$ space as shown in Fig. 4. We use circles to denote video embeddings, and crosses for summary embeddings. Each video-summary pair is marked by the same color.

We can clearly observe that the video and its summary are mostly embedded much closer by *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) (Fig. 4(b)) than ones by *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$) (Fig. 4(a)). In particular the video embedded by *re*-SEQ2SEQ ($\lambda = \lambda^*, \eta = \eta^*$) mostly has its corresponding summary as the nearest neighbor, while this is usually not the case in ones by *re*-SEQ2SEQ ($\lambda = 0, \eta = 0$). Moreover, embeddings of videos and summaries in Fig. 4(a) are 'clustered' together compared to ones in Fig. 4(b), where different pairs of video and summary are relatively far away from each other. This shows the proposed approach embeds a summary and its original video into similar locations, while pushing apart mismatched summaries and original videos.

5 Conclusion

We propose a novel sequence-to-sequence learning model for video summarization that not only minimizes the discriminative loss for matching the generated and target summaries, but also embeds corresponding video and summary pairs in close proximity in an abstract semantic space. The proposed approach exploits both labeled and unlabeled videos to derive semantic embeddings. Extensive experimental results on multiple datasets show the advantage of our approach over existing methods in both supervised and semi-supervised settings. In the future, we plan to explore more delicate strategies to combine unlabeled data during training to improve summarization performance.

Acknowledgments. We appreciate the feedback from the reviewers. KG is partially supported by NSF IIS-1514118 and an AWS Machine Learning Research Award. Others are partially supported by USC Graduate Fellowships, NSF IIS-1065243, 1451412, 1513966/1632803/1833137, 1208500, CCF-1139148, a Google Research Award, an Alfred P. Sloan Research Fellowship, gifts from Facebook and Netflix, and ARO# W911NF-12-1-0241 and W911NF-15-1-0484.

References

1. Open video project. http://www.open-video.org/
2. Youtube statistics. https://www.youtube.com/yt/press/statistics.html
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: ICLR (2015)
4. Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., Bengio, Y.: End-to-end attention-based large vocabulary speech recognition. In: ICASSP (2016)
5. Bellet, A., Habrard, A., Sebban, M.: Metric learning. Synth. Lect. Artif. Intell. Mach. Learn. **9**(1), 1–151 (2015)
6. Chao, W.L., Gong, B., Grauman, K., Sha, F.: Large-margin determinantal point processes. In: UAI (2015)
7. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint (2014)
8. Chu, W.S., Song, Y., Jaimes, A.: Video co-summarization: video summarization by visual co-occurrence. In: CVPR (2015)
9. Chung, J., Ahn, S., Bengio, Y.: Hierarchical multiscale recurrent neural networks. arXiv preprint (2016)
10. Dai, A.M., Le, Q.V.: Semi-supervised sequence learning. In: NIPS (2015)
11. De Avila, S.E.F., Lopes, A.P.B., da Luz, A., de Albuquerque Araújo, A.: VSUMM: a mechanism designed to produce static video summaries and a novel evaluation method. Pattern Recognit. Lett. **32**(1), 56–68 (2011)
12. Furini, M., Geraci, F., Montangero, M., Pellegrini, M.: STIMO: still and moving video storyboard for the web scenario. Multimed. Tools Appl. **46**(1), 47–69 (2010)
13. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with LSTM. Neural Comput. **12**(10), 2451–2471 (2000)
14. Goldman, D.B., Curless, B., Salesin, D., Seitz, S.M.: Schematic storyboarding for video visualization and editing. ACM Trans. Graph. **25**(3), 862–871 (2006)

15. Gong, B., Chao, W.L., Grauman, K., Sha, F.: Diverse sequential subset selection for supervised video summarization. In: NIPS (2014)
16. Graves, A., Jaitly, N.: Towards end-to-end speech recognition with recurrent neural networks. In: ICML (2014)
17. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. **18**(5), 602–610 (2005)
18. Gygli, M., Grabner, H., Van Gool, L.: Video summarization by learning submodular mixtures of objectives. In: CVPR (2015)
19. Gygli, M., Grabner, H., Riemenschneider, H., Van Gool, L.: Creating summaries from user videos. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8695, pp. 505–520. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10584-0_33
20. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
21. Hong, R., Tang, J., Tan, H.K., Yan, S., Ngo, C., Chua, T.S.: Event driven summarization for web videos. In: SIGMM Workshop (2009)
22. Jean, S., Cho, K., Memisevic, R., Bengio, Y.: On using very large target vocabulary for neural machine translation. In: ACL (2015)
23. Jean, S., Firat, O., Cho, K., Memisevic, R., Bengio, Y.: Montreal neural machine translation systems for WMT'15. In: WMT (2015)
24. Ji, Z., Xiong, K., Pang, Y., Li, X.: Video summarization with attention-based encoder-decoder networks. arXiv preprint (2017)
25. Kang, H.W., Matsushita, Y., Tang, X., Chen, X.Q.: Space-time video montage. In: CVPR (2006)
26. Khosla, A., Hamid, R., Lin, C.J., Sundaresan, N.: Large-scale video summarization using web-image priors. In: CVPR (2013)
27. Kim, G., Xing, E.P.: Reconstructing storyline graphs for image recommendation from web community photos. In: CVPR (2014)
28. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint (2014)
29. Laganière, R., Bacco, R., Hocevar, A., Lambert, P., Païs, G., Ionescu, B.E.: Video summarization from spatio-temporal features. In: ACM TRECVid Video Summarization Workshop (2008)
30. Lee, Y.J., Ghosh, J., Grauman, K.: Discovering important people and objects for egocentric video summarization. In: CVPR (2012)
31. Li, Y., Merialdo, B.: Multi-video summarization based on video-MMR. In: WIAMIS Workshop (2010)
32. Liu, D., Hua, G., Chen, T.: A hierarchical visual model for video object summarization. IEEE Trans. Pattern Anal. Mach. Intell. **32**(12), 2178–2190 (2010)
33. Liu, T., Kender, J.R.: Optimization algorithms for the selection of key frame sequences of variable length. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 403–417. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-47979-1_27
34. Lu, Z., Grauman, K.: Story-driven summarization for egocentric video. In: CVPR (2013)
35. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. arXiv preprint (2015)
36. Ma, Y.F., Lu, L., Zhang, H.J., Li, M.: A user attention model for video summarization. In: ACM MM (2002)
37. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**, 2579–2605 (2008)

38. Mahasseni, B., Lam, M., Todorovic, S.: Unsupervised video summarization with adversarial LSTM networks. In: CVPR (2017)
39. Mueller, J., Gifford, D., Jaakkola, T.: Sequence to better sequence: continuous revision of combinatorial structures. In: ICML, pp. 2536–2544 (2017)
40. Mundur, P., Rao, Y., Yesha, Y.: Keyframe-based video summarization using delaunay clustering. Int. J. Digit. Libr. **6**(2), 219–232 (2006)
41. Nam, J., Tewfik, A.H.: Event-driven video abstraction and visualization. Multimed. Tools Appl. **16**(1–2), 55–77 (2002)
42. Ng, J.Y.H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: deep networks for video classification. In: CVPR (2015)
43. Ngo, C.W., Ma, Y.F., Zhang, H.: Automatic video summarization by graph modeling. In: ICCV (2003)
44. Panda, R., Das, A., Wu, Z., Ernst, J., Roy-Chowdhury, A.K.: Weakly supervised summarization of web videos. In: ICCV (2017)
45. Potapov, D., Douze, M., Harchaoui, Z., Schmid, C.: Category-specific video summarization. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 540–555. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10599-4_35
46. Pritch, Y., Rav-Acha, A., Gutman, A., Peleg, S.: Webcam synopsis: peeking around the world. In: ICCV (2007)
47. Ramachandran, P., Liu, P.J., Le, Q.V.: Unsupervised pretraining for sequence to sequence learning. arXiv preprint (2016)
48. Sharghi, A., Gong, B., Shah, M.: Query-focused extractive video summarization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 3–19. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_1
49. Sharghi, A., Laurel, J.S., Gong, B.: Query-focused video summarization: dataset, evaluation, and a memory network based approach. In: CVPR (2017)
50. Song, Y., Vallmitjana, J., Stent, A., Jaimes, A.: TVSum: summarizing web videos using titles. In: CVPR (2015)
51. Sun, M., Farhadi, A., Taskar, B., Seitz, S.: Salient montages from unconstrained videos. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8695, pp. 472–488. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10584-0_31
52. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS (2014)
53. Szegedy, C., et al.: Going deeper with convolutions. In: CVPR (2015)
54. Vasudevan, A.B., Gygli, M., Volokitin, A., Van Gool, L.: Query-adaptive video summarization via quality-aware relevance estimation. In: ACM MM (2017)
55. Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., Saenko, K.: Sequence to sequence-video to text. In: ICCV (2015)
56. Vinyals, O., Kaiser, Ł., Koo, T., Petrov, S., Sutskever, I., Hinton, G.: Grammar as a foreign language. In: NIPS (2015)
57. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: CVPR (2015)
58. Wu, Y., et al.: Google's neural machine translation system: bridging the gap between human and machine translation. arXiv preprint (2016)
59. Xu, K., et al.: Show, attend and tell: neural image caption generation with visual attention. In: ICML (2015)
60. Yang, H., Wang, B., Lin, S., Wipf, D., Guo, M., Guo, B.: Unsupervised extraction of video highlights via robust recurrent auto-encoders. In: ICCV (2015)

61. Zaremba, W., Sutskever, I.: Learning to execute. arXiv preprint (2014)
62. Zeng, K.-H., Chen, T.-H., Niebles, J.C., Sun, M.: Title generation for user generated videos. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 609–625. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_38
63. Zhang, H.J., Wu, J., Zhong, D., Smoliar, S.W.: An integrated system for content-based video retrieval and browsing. Pattern Recognit. **30**(4), 643–658 (1997)
64. Zhang, K., Chao, W.L., Sha, F., Grauman, K.: Summary transfer: exemplar-based subset selection for video summarization. In: CVPR (2016)
65. Zhang, K., Chao, W.-L., Sha, F., Grauman, K.: Video summarization with long short-term memory. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 766–782. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_47
66. Zhao, B., Li, X., Lu, X.: Hierarchical recurrent neural network for video summarization. In: ACM MM (2017)
67. Zhao, B., Xing, E.P.: Quasi real-time summarization for consumer videos. In: CVPR (2014)
68. Zhou, K., Qiao, Y.: Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. arXiv preprint (2017)