Video Summarization with Long Short-term Memory

Ke Zhang[1⋆], Wei-Lun Chao[1⋆], Fei Sha[2], and Kristen Grauman[3]

[1]Dept. of Computer Science, U. of Southern California, United States
[2]Dept. of Computer Science, U. of California, Los Angeles, United States
[3]Dept. of Computer Science, U. of Texas at Austin, United States
{zhang.ke, weilunc}@usc.edu, feisha@cs.ucla.edu, grauman@cs.utexas.edu

Abstract. We propose a novel supervised learning technique for summarizing videos by automatically selecting keyframes or key subshots. Casting the task as a structured prediction problem, our main idea is to use Long Short-Term Memory (LSTM) to model the variable-range temporal dependency among video frames, so as to derive both representative and compact video summaries. The proposed model successfully accounts for the sequential structure crucial to generating meaningful video summaries, leading to state-of-the-art results on two benchmark datasets. In addition to advances in modeling techniques, we introduce a strategy to address the need for a large amount of annotated data for training complex learning approaches to summarization. There, our main idea is to exploit auxiliary annotated video summarization datasets, in spite of their heterogeneity in visual styles and contents. Specifically, we show that domain adaptation techniques can improve learning by reducing the discrepancies in the original datasets' statistical properties.

Keywords: Video Summarization, Long Short-Term Memory

1 Introduction

Video has rapidly become one of the most common sources of visual information. The amount of video data is daunting — it takes over 82 years to watch all videos uploaded to YouTube per day! Automatic tools for analyzing and understanding video contents are thus essential. In particular, *automatic video summarization* is a key tool to help human users browse video data. A good video summary would compactly depict the original video, distilling its important events into a short watchable synopsis. Video summarization can shorten video in several ways. In this paper, we focus on the two most common ones: *keyframe selection*, where the system identifies a series of defining frames [1,2,3,4,5] and *key subshot selection*, where the system identifies a series of defining subshots, each of which is a temporally contiguous set of frames spanning a short time interval [6,7,8,9].

There has been a steadily growing interest in studying learning techniques for video summarization. Many approaches are based on unsupervised learning, and

⋆ Equal contributions

define intuitive criteria to pick frames [1,5,6,9,10,11,12,13,14] without explicitly optimizing the evaluation metrics. Recent work has begun to explore supervised learning techniques [2,15,16,17,18]. In contrast to unsupervised ones, supervised methods directly learn from human-created summaries to capture the underlying frame selection criterion as well as to output a subset of those frames that is more aligned with human semantic understanding of the video contents.

Supervised learning for video summarization entails two questions: what type of learning model to use? and how to acquire enough annotated data for fitting those models? Abstractly, video summarization is a structured prediction problem: the input to the summarization algorithm is a sequence of video frames, and the output is a binary vector indicating whether a frame is to be selected or not. This type of sequential prediction task is the underpinning of many popular algorithms for problems in speech recognition, language processing, etc. The most important aspect of this kind of task is that the decision to select cannot be made locally and in isolation — the inter-dependency entails making decisions after considering all data from the original sequence.

For video summarization, the inter-dependency across video frames is complex and highly inhomogeneous. This is not entirely surprising as human viewers rely on high-level semantic understanding of the video contents (and keep track of the unfolding of storylines) to decide whether a frame would be valuable to keep for a summary. For example, in deciding what the keyframes are, *temporally close* video frames are often visually similar and thus convey redundant information such that they should be condensed. However, the converse is not true. That is, visually similar frames do not have to be temporally close. For example, consider summarizing the video "leave home in the morning and come back to lunch at home and leave again and return to home at night." While the frames related to the "at home" scene can be visually similar, the semantic flow of the video dictates none of them should be eliminated. Thus, a summarization algorithm that relies on examining visual cues only but fails to take into consideration the high-level semantic understanding about the video over a *long-range temporal span* will erroneously eliminate important frames. Essentially, the nature of making those decisions is largely sequential – any decision including or excluding frames is dependent on other decisions made on a temporal line.

Modeling variable-range dependencies where both short-range and long-range relationships intertwine is a long-standing challenging problem in machine learning. Our work is inspired by the recent success of applying long short-term memory (LSTM) to structured prediction problems such as speech recognition [19,20,21] and image and video captioning [22,23,24,25,26]. LSTM is especially advantageous in modeling long-range structural dependencies where the influence by the distant past on the present and the future must be adjusted in a data-dependent manner. In the context of video summarization, LSTMs explicitly use its memory cells to learn the progression of "storylines", thus to know when to forget or incorporate the past events to make decisions.

In this paper, we investigate how to apply LSTM and its variants to supervised video summarization. We make the following contributions. We propose vsLSTM, a LSTM-based model for video summarization (Sec. 3.3). Fig. 2 illustrates the conceptual design of the model. We demonstrate that the sequential modeling aspect of LSTM is essential; the performance of multi-layer neural networks (MLPs) using neighboring frames as features is inferior. We further show how LSTM's strength can be enhanced by combining it with the determinantal point process (DPP), a recently introduced probabilistic model for diverse subset selection [2,27]. The resulting model achieves the best results on two recent challenging benchmark datasets (Sec. 4). Besides advances in modeling, we also show how to address the practical challenge of insufficient human-annotated video summarization examples. We show that model fitting can benefit from combining video datasets, despite their heterogeneity in both contents and visual styles. In particular, this benefit can be improved by "domain adaptation" techniques that aim to reduce the discrepancies in statistical characteristics across the diverse datasets.

The rest of the paper is organized as follows. Section 2 reviews related work of video summarization, and Section 3 describes the proposed LSTM-based model and its variants. In Section 4, we report empirical results. We examine our approach in several supervised learning settings and contrast it to other existing methods, and we analyze the impact of domain adapation for merging summarization datasets for training (Section 4.4). We conclude our paper in Section 5.

2 Related Work

Techniques for automatic video summarization fall in two broad categories: unsupervised ones that rely on manually designed criteria to prioritize and select frames or subshots from videos [1,3,5,6,9,10,11,12,14,28,29,30,31,32,33,34,35,36] and supervised ones that leverage human-edited summary examples (or frame importance ratings) to *learn* how to summarize novel videos [2,15,16,17,18]. Recent results by the latter suggest great promise compared to traditional unupservised methods.

Informative criteria include relevance [10,13,14,31,36], representativeness or importance [5,6,9,10,11,33,35], and diversity or coverage [1,12,28,30,34]. Several recent methods also exploit auxiliary information such as web images [10,11,33,35] or video categories [31] to facilitate the summarization process.

Because they explicitly learn from human-created summaries, supervised methods are better equipped to align with how humans would summarize the input video. For example, a prior supervised approach learns to combine multiple hand-crafted criteria so that the summaries are consistent with ground truth [15,17]. Alternatively, the determinatal point process (DPP) — a probabilistic model that characterizes how a representative and diverse subset can be sampled from a ground set — is a valuable tool to model summarization in the supervised setting [2,16,18].

None of above work uses LSTMs to model both the *short-range and long-range dependencies* in the sequential video frames. The sequential DPP proposed in [2] uses *pre-defined temporal structures*, so the dependencies are "hardwired". In contrast, LSTMs can model dependencies with a data-dependent on/off switch, which is extremely powerful for modeling sequential data [20].

LSTMs are used in [37] to model temporal dependencies to identify video highlights, cast as auto-encoder-based outlier detection. LSTMs are also used in modeling an observer's visual attention in analyzing images [38,39], and to perform natural language video description [23,24,25]. However, to the best of our knowledge, our work is the first to explore LSTMs for video summarization. As our results will demonstrate, their flexibility in capturing sequential structure is quite promising for the task.

3 Approach

In this section, we describe our methods for summarizing videos. We first formally state the problem and the notations, and briefly review LSTM [40,41,42], the building block of our approach. We then introduce our first summarization model vsLSTM. Then we describe how we can enhance vsLSTM by combining it with a determinantal point process (DPP) that further takes the summarization structure (e.g., diversity among selected frames) into consideration.

3.1 Problem Statement

We use $\boldsymbol{x} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_t, \cdots, \boldsymbol{x}_T\}$ to denote a sequence of frames in a video to be summarized while $\boldsymbol{x}_t$ is the visual features extracted at the t-th frame.

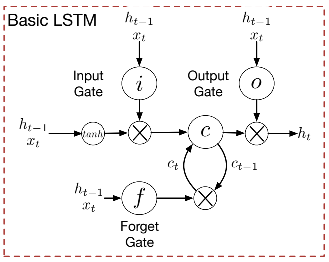
The output of the summarization algorithm can take one of two forms. The first is *selected keyframes* [2,3,12,28,29,43], where the summarization result is a subset of (isolated) frames. The second is *interval-based keyshots* [15,17,31,35], where the summary is a set of (short) intervals along the time axis. Instead of binary information (being selected or not selected), certain datasets provide frame-level importance scores computed from human annotations [17,35]. Those scores represent the likelihoods of the frames being selected as a part of summary. Our models make use of all types of annotations — binary keyframe labels, binary subshot labels, or frame-level importances — as learning signals.[1]

Our models use frames as its internal representation. The inputs are frame-level features $\boldsymbol{x}$ and the (target) outputs are either hard binary indicators or frame-level importance scores (i.e., softened indicators).

3.2 Long Short-Term Memory (LSTM)

LSTMs are a special kind of recurrent neural network that are adept at modeling long-range dependencies. At the core of the LSTMs are memory cells $\boldsymbol{c}$ which encode, at every time step, the knowledge of the inputs that have been observed up to that step. The cells are modulated by nonlinear sigmoidal gates, and

[1] We describe below and in the Supplementary Material how to convert between the annotation formats when necessary.



$$\begin{aligned}
\boldsymbol{i}_t &= \text{sigmoid}(\boldsymbol{W}_i[\boldsymbol{x}_t^{\mathrm{T}}, \boldsymbol{h}_{t-1}^{\mathrm{T}}]^{\mathrm{T}}) \\
\boldsymbol{f}_t &= \text{sigmoid}(\boldsymbol{W}_f[\boldsymbol{x}_t^{\mathrm{T}}, \boldsymbol{h}_{t-1}^{\mathrm{T}}]^{\mathrm{T}}) \\
\boldsymbol{o}_t &= \text{sigmoid}(\boldsymbol{W}_o[\boldsymbol{x}_t^{\mathrm{T}}, \boldsymbol{h}_{t-1}^{\mathrm{T}}]^{\mathrm{T}}) \qquad (1) \\
\boldsymbol{c}_t &= \boldsymbol{i}_t \odot \tanh(\boldsymbol{W}_c[\boldsymbol{x}_t^{\mathrm{T}}, \boldsymbol{h}_{t-1}^{\mathrm{T}}]^{\mathrm{T}}) \\
&\quad + \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t),
\end{aligned}$$

Fig. 1. The LSTM unit, redrawn from [21]. The memory cell is modulated jointly by the input, output and forget gates to control the knowledge transferred at each time step. $\odot$ denotes element-wise products.

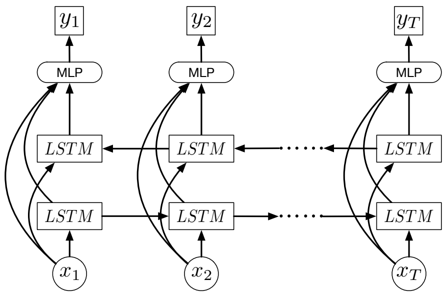


Fig. 2. Our vsLSTM model for video summarization. The model is composed of two LSTM (long short-term memory) layers: one layer models video sequences in the forward direction and the other the backward direction. Each *LSTM* block is a LSTM unit, shown in Fig. 1. The forward/backward chains model temporal inter-dependencies between the past and the future. The inputs to the layers are visual features extracted at frames. The outputs combine the LSTM layers' hidden states and the visual features with a multi-layer perceptron, representing the likelihoods of whether the frames should be included in the summary. As our results will show, modeling sequential structures as well as the long-range dependencies is essential.

are applied multiplicatively. The gates determine whether the LSTM keeps the values at the gates (if the gates evaluate to 1) or discard them (if the gates evaluate to 0).

There are three gates: the input gate ($\boldsymbol{i}$) controlling whether the LSTM considers its current input ($\boldsymbol{x}_t$), the forget gate ($\boldsymbol{f}$) allowing the LSTM to forget its previous memory ($\boldsymbol{c}_t$), and the output gate ($\boldsymbol{o}$) deciding how much of the memory to transfer to the hidden states ($\boldsymbol{h}_t$). Together they enable the LSTM to learn complex long-term dependencies – in particular, the forget date serves as a time-varying data-dependent on/off switch to selectively incorporating the past and present information. See Fig. 1 for a conceptual diagram of a LSTM unit and its algebraic definitions [21].

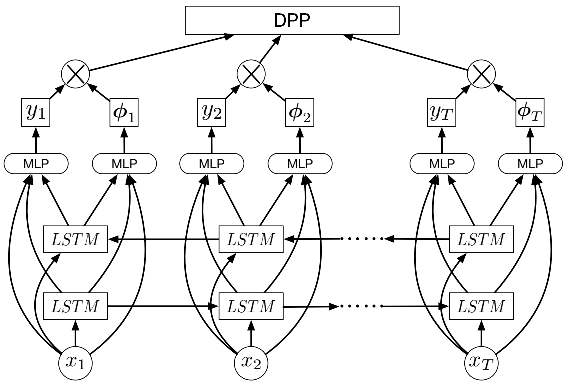


Fig. 3. Our dppLSTM model. It combines vsLSTM (Fig. 2) and DPP by modeling both long-range dependencies and pairwise frame-level repulsiveness explicitly.

3.3 vsLSTM for Video Summarization

Our vsLSTM model is illustrated in Fig. 2. There are several differences from the basic LSTM model. We use bidirectional LSTM layers [44] for modeling better long-range dependency in both the past and the future directions. Note that the forward and the backward chains do not directly interact.

We combine the information in those two chains, as well as the visual features, with a multi-layer perceptron (MLP). The output of this perceptron is a scalar

$$y_t = f_I(\boldsymbol{h}_t^{\text{forward}}, \boldsymbol{h}_t^{\text{backward}}, \boldsymbol{x}_t).$$

To learn the parameters in the LSTM layers and the MLP for $f_I(\cdot)$, our algorithm can use annotations in the forms of either the frame-level importance scores or the selected keyframes encoded as binary indicator vectors. In the former case, y is a continuous variable and in the latter case, y is a binary variable. The parameters are optimized with stochastic gradient descent.

3.4 Enhancing vsLSTM by Modeling Pairwise Repulsiveness

vsLSTM excels at predicting the likelihood that a frame should be included or how important/relevant a frame is to the summary. We further enhance it with the ability to model pairwise frame-level "repulsiveness" by stacking it with a determinantal point process (DPP) (which we discuss in more detail below). Modeling the repulsiveness aims to increase the diversity in the selected frames by eliminating redundant frames. The modeling advantage provided in DPP has been exploited in DPP-based summarization methods [2,16,18]. Note that diversity can only be measured "collectively" on a (sub)set of (selected) frames, not on frames independently or sequentially. The directed sequential nature in LSTMs is arguably *weaker* in examining *all the fames simultaneously* in the subset to measure diversity, thus is at the risk of having higher recall but lower precision. On the other hand, DPPs likely yield low recalls but high precisions. In essence, the two are complementary to each other.

Determinantal point processes (DPP) Given a ground set Z of N items (e.g., all frames of a video), together with an N $\times$ N kernel matrix $\boldsymbol{L}$ that records the pairwise frame-level similarity, a DPP encodes the probability to sample any subset from the ground set [2,27]. The probability of a subset $\boldsymbol{z}$ is proportional to the determinant of the corresponding principal minor of the matrix $\boldsymbol{L_z}$

$$P(\boldsymbol{z} \subset \mathsf{Z}; \boldsymbol{L}) = \frac{\det(\boldsymbol{L_z})}{\det(\boldsymbol{L}+\boldsymbol{I})}, \tag{2}$$

where $\boldsymbol{I}$ is the N $\times$ N identity matrix. If two items are identical and appear in the subset, $\boldsymbol{L_z}$ will have identical rows and columns, leading to zero-valued determinant. Namely, we will have zero-probability assigned to this subset. A highly probable subset is one capturing significant diversity (i.e., pairwise dissimilarity).

dppLSTM Our dppLSTM model is schematically illustrated in Fig. 3. To exploit the strength of DPP in explicitly modeling diversity, we use the prediction of our vsLSTM in defining the $\boldsymbol{L}$-matrix:

$$L_{tt'} = y_t y_{t'} S_{tt'} = y_t y_{t'} \boldsymbol{\phi}_t^{\mathrm{T}} \boldsymbol{\phi}_{t'}, \tag{3}$$

where the similarity between the frames x_t and x'_t are modeled with the inner product of another multi-layer perceptron's outputs

$$\boldsymbol{\phi}_t = f_S(\boldsymbol{h}_t^{\text{forward}}, \boldsymbol{h}_t^{\text{backward}}, \boldsymbol{x}_t), \ \boldsymbol{\phi}_{t'} = f_S(\boldsymbol{h}_{t'}^{\text{forward}}, \boldsymbol{h}_{t'}^{\text{backward}}, \boldsymbol{x}_{t'}).$$

This decomposition is similar in spirit to the quality-diversity (QD) decomposition proposed in [45]. While [2] also parameterizes $L_{tt'}$ with a single MLP, our model subsumes theirs. Moreover, our empirical results show that using two different sets of MLPs — $f_I(\cdot)$ for frame-level importance and $f_S(\cdot)$ for similarity — leads to better performance than using a single MLP to jointly model the two factors. (They are implemented by one-hidden-layer neural networks with 256 sigmoid hidden units, and sigmoid and linear output units, respectively. See the Supplementary Material for details.)

Learning To train a complex model such as dppLSTM, we adopt a stage-wise optimization routine. We first train the MLP $f_I(\cdot)$ and the LSTM layers as in vsLSTM. Then, we train all the MLPs and the LSTM layers by maximizing the likelihood of keyframes specified by the DPP model. Denote $\mathsf{Z}^{(i)}$ as the collection of frames of the i-th video and $\boldsymbol{z}^{(i)*} \subset \mathsf{Z}^{(i)}$ as the corresponding target subset of keyframes. We learn $\boldsymbol{\theta}$ that parameterizes (3) by MLE [27]:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_i \log\{P(\boldsymbol{z}^{(i)*} \subset \mathsf{Z}^{(i)}; \boldsymbol{L}^{(i)}(\boldsymbol{\theta}))\}. \tag{4}$$

Details are in the Supplementary Material. We have found this training procedure is effective in quickly converging to a good local optima.

Table 1. Key characteristics of datasets used in our empirical studies.

Dataset	# of video	Description	Annotations
SumMe	25	User generated videos of events	interval-based shots
TVSum	50	YouTube videos (10 categories)	frame-level importance
OVP	50	Documentary videos	selected keyframes as summarization
YouTube	39	YouTube videos (Sports, News, etc)	

3.5 Generating Shot-based Summaries from Our Models

Our vsLSTM predicts frame-level importance scores, i.e., the likelihood that a frame should be included in the summary. For our dppLSTM, the approximate MAP inference algorithm [46] outputs a subset of selected frames. Thus, for dppLSTM we use the procedure described in the Supplementary Material to convert them into keyshot-based summaries for evaluation.

4 Experimental Results

We first define the experimental setting (datasets, features, metrics). Then we provide key quantitative results demonstrating our method's advantages over existing techniques (Sec. 4.2). Next we analyze more deeply the impact of our method design (Sec. 4.3) and explore the use of domain adaptation for "homogenizing" diverse summarization datasets (Sec. 4.4). Finally, we present example qualitative results (Sec. 4.5).

4.1 Experimental Setup

Datasets We evaluate the performance of our models on two video datasets, **SumMe** [17] and **TVSum** [35]. **SumMe** consists of 25 user videos recording a variety of events such as holidays and sports. **TVSum** contains 50 videos downloaded from YouTube in 10 categories defined in the TRECVid Multimedia Event Detection (MED). Most of the videos are 1 to 5 minutes in length.

To combat the need of a large amount of annotated data, we use two other annotated datasets whuch are annotated with keyframe-based summarization, **Youtube** [28] and **Open Video Project (OVP)** [47,28]. We process them as[2] to create a ground-truth set of keyframes (then convert to a ground-truth sequence of frame-level importance scores) for each video. We use the ground-truth in importance scores to train vsLSTM and convert the sequence to selected keyframes to train dppLSTM.

For evaluation, both datasets provide multiple user-annotated summaries for each video, either in the form of keyshots (**SumMe**) or frame-level importance scores (**TVSum**, convertible to keyshot-based summaries). Such conversions are documented in the Supplementary Material.

Table 3 summarizes key characteristics of these datasets. We can see that these four datasets are heterogeneous in both their visual styles and contents.

Features For most experiments, the feature descriptor of each frame is obtained by extracting the output of the penultimate layer (pool 5) of the GoogLeNet model [48] (1024-dimensions). We also experiment with the same shallow features used in [35] (i.e., color histograms, GIST, HOG, dense SIFT) to provide a comparison to the deep features.

Evaluation metrics Following the protocols in [15,17,35], we constrain the generated keyshot-based summary A to be less than 15% in duration of the original video (details in the Supplementary Material). We then compute the precision (P) and recall (R) against the user summary B for evaluation, according to the *temporal overlap* between the two:

$$\mathrm{P} = \frac{\text{overlapped duration of } \mathsf{A} \text{ and } \mathsf{B}}{\text{duration of } \mathsf{A}}, \mathrm{R} = \frac{\text{overlapped duration of } \mathsf{A} \text{ and } \mathsf{B}}{\text{duration of } \mathsf{B}}, \quad (5)$$

as well as their harmonic mean F-score,

$$\mathrm{F} = 2\mathrm{P} \times \mathrm{R}/(\mathrm{P} + \mathrm{R}) \times 100\%. \quad (6)$$

We also follow [35,15] to compute the metrics when there are multiple human-annotated summaries of a video.

Variants of supervised learning settings We study several settings for supervised learning, summarized in Table 2:

- Canonical This is the standard supervised learning setting where the training, validation, and testing sets are from the same dataset, though they are disjoint.
- Augmented In this setting, for a given dataset, we randomly leave 20% of it for testing, and augment the remaining 80% with the other three datasets to form an augmented training and validation dataset. Our hypothesis is that, despite being heterogeneous in styles and contents, the augmented dataset can be beneficial in improving the performance of our models because of the increased amount of annotations.
- Transfer In this setting, for a given dataset, we use the other three datasets for training and validation and test the learned models on the dataset. We are interested in investigating if existing datasets can effectively transfer summarization models to new unannotated datasets. If the transfer can be successful, then it would be possible to summarize a large number of videos in the wild where there is virtually no closely corresponding annotation.

4.2 Main Results

Table 3 summarizes the performance of our methods and contrasts to those attained by prior work. Red-colored numbers indicate that our dppLSTM obtains the best performance in the corresponding setting. Otherwise the best performance is bolded. In the common setting of “Canonical” supervised learning, on

Table 2. Supervision settings tested

Dataset	Settings	Training & Validation	Testing
SumMe	Canonical	80% SumMe	20% SumMe
	Augmented	OVP + Youtube + TVSum + 80% SumMe	20% SumMe
	Transfer	OVP + Youtube + TVSum	SumMe
TVSum	Canonical	80% TVSum	20% TVSum
	Augmented	OVP + Youtube + SumMe + 80% TVSum	20% TVSum
	Transfer	OVP + Youtube + SumMe	TVSum

Table 3. Performance (F-score) of various video summarization methods. Published results are denoted in ***bold italic***; our implementation is in normal font. Empty boxes indicate settings inapplicable to the method tested.

Dataset	Method	unsupervised	Canonical	Augmented	Transfer
SumMe	[30]	***26.6***			
	[17]		***39.4***		
	[15]		***39.7***		
	[16]		***40.9***†	41.3	38.5
	vsLSTM (ours)		37.6$\pm$0.8	41.6$\pm$0.5	40.7$\pm$0.6
	dppLSTM (ours)		38.6$\pm$0.8	42.9$\pm$0.5	41.8$\pm$0.5
TVSum	[34]	***46.0***			
	[11]‡	***36.0***			
	[35]‡	***50.0***			
	vsLSTM (ours)		54.2$\pm$0.7	57.9$\pm$0.5	56.9$\pm$0.5
	dppLSTM (ours)		54.7$\pm$0.7	59.6$\pm$0.4	58.7$\pm$0.4

†: build video classifiers using TVSum [35]. ‡: use auxiliary web images for learning.

TVSum, both of our two methods outperform the state-of-the-art. However, on SumMe, our methods underperform the state-of-the-art, likely due to the fewer annotated training samples in SumMe.

What is particularly interesting is that our methods can be significantly improved when the amount of annotated data is increased. In particular, in the case of Transfer learning, even though the three training datasets are significantly different from the testing dataset, our methods leverage the annotations effectively to improve accuracy over the Canonical setting, where the amount of annotated training data is limited. The best performing setting is Augmented, where we combine all four datasets together to form one training dataset.

The results suggest that with sufficient annotated data, our model can capture temporal structures better than prior methods that lack explicit temporal structures [11,15,17,30,35] as well as those that consider only pre-defined ones [2,16]. More specifically, bidirectional LSTMs and DPPs help to obtain diverse results conditioned on the whole video while leveraging the sequential nature of videos. See the Supplementary Material for further discussions.

Table 4. Modeling video data with LSTMs is beneficial. The reported numbers are F-scores by various summarization methods.

Dataset	Method	Canonical	Augmented	Transfer
SumMe	MLP-Shot	**39.8**±0.7	40.7±0.7	39.8±0.6
	MLP-Frame	38.2±0.8	41.2±0.8	40.2±0.9
	vsLSTM	37.6±0.8	41.6±0.5	40.7±0.6
TVSum	MLP-Shot	**55.2**±0.5	56.7±0.5	55.5±0.5
	MLP-Frame	53.7±0.7	56.1±0.7	55.3±0.6
	vsLSTM	54.2±0.7	57.9±0.5	56.9±0.5

4.3 Analysis

Next we analyze more closely several settings of interest.

How important is sequence modeling? Table 4 contrasts the performance of the LSTM-based method vsLSTM to a multi-layer perceptron based baseline. In this baseline, we learn a two-hidden-layer MLP that has the same number of hidden units in each layer as does one of the MLPs of our model.

Since MLP cannot explicitly capture temporal information, we consider two variants in the interest of fair comparison to our LSTM-based approach. In the first variant MLP-Shot, we use the averaged frame features in a shot as the inputs to the MLP and predict shot-level importance scores. The ground-truth shot-level importance scores are derived as the average of the corresponding frame-level importance scores. The predicted shot-level importance scores are then used to select keyshots and the resulting shot-based summaries are then compared to user annotations. In the second variant MLP-Frame, we concatenate all visual features within a K-frame ($K = 5$ in our experiments) window centered around each frame to be the inputs for predicting frame-level importance scores.

It is interesting to note that in the Canonical setting, MLP-based approaches outperform vsLSTM. However, in all other settings where the amount of annotations is increased, our vsLSTM is able to outperform the MLP-based methods noticeably. This confirms the common perception about LSTMs: while they are powerful, they often demand a larger amount of annotated data in order to perform well.

Shallow versus deep features? We also study the effect of using alternative visual features for each frame. Table 5 suggests that deep features are able to modestly improve performance over the shallow features. Note that our dppLSTM with shallow features still outperforms [35], which reported results on TVSum using the same shallow features (i.e., color histograms, GIST, HOG, dense SIFT).

What type of annotation is more effective? There are two common types of annotations in video summarization datasets: binary indicators of whether a frame is selected or not and frame-level importance scores on how likely a frame should be included in the summary. While our models can take either format, we

Table 5. Summarization results (in F-score) by our dppLSTM using shallow and deep features. Note that [35] reported ***50.0%*** on TVSum using the same shallow features.

Dataset	Feature type	Canonical	Transfer
SumMe	deep	38.6±0.8	**41.8**±0.5
	shallow	38.1±0.9	40.7±0.5
TVSum	deep	54.7±0.7	**58.7**±0.4
	shallow	54.0±0.7	57.9±0.5

Table 6. Results by vsLSTM on different types of annotations in the Canonical setting

dataset	Binary	Importance score
SumMe	37.2±0.8	37.6±0.8
TVSum	53.7±0.8	54.2±0.7

suspect the frame-level importance scores provide richer information than the binary indicators as they represent relative goodness among frames..

Table 6 illustrates the performance of our vsLSTM model when using the two different annotations, in the Canonical setting. Using frame-level importance scores has a consistent advantage.

However, this does not mean binary annotation/keyframes annotations cannot be exploited. Our dppLSTM exploits both frame-level importance scores and binary signals. In particular, dppLSTM first uses frame-level importance scores to train its LSTM layers and then uses binary indicators to form objective functions to fine tune (cf. Section 3 for the details of this stage-wise training). Consequently, comparing the results in Table 3 to Table 6, we see that dppLSTM improves further by utilizing both types of annotations.

4.4 Augmenting the Training Data with Domain Adaptation

While Table 3 clearly indicates the advantage of augmenting the training dataset, those auxiliary datasets are often different from the target one in contents and styles. We improve summarization further by borrowing the ideas from visual domain adaptation for object recognition [49,50,51]. The main idea is first eliminate the discrepancies in data distribution before augmenting.

Table 7 shows the effectiveness of this idea. We use a simple domain adaptation technique [52] to reduce the data distribution discrepancy among all four datasets, by transforming the visual features linearly such that the covariance matrices for the four datasets are close to each other. The "homogenized" datasets, when combined (in both the Transfer and Augmented settings), lead to an improved summary F-score. The improvements are especially pronounced for the smaller dataset SumMe.

4.5 Qualitative Results

We provide exemplar video summaries in Fig. 4. We illustrate the temporal modeling capability of dppLSTM and contrast with MLP-Shot.

Table 7. Summarization results by our model in the Transfer and Augmented settings, optionally with visual features linearly adapted to reduce cross-dataset discrepancies

Dataset	Method	Transfer		Augmented	
		w/o Adaptation	w/ Adaptation	w/o Adaptation	w/ Adaptation
SumMe	vsLSTM	40.7±0.6	41.3±0.6	41.6±0.5	42.1±0.6
	dppLSTM	41.8±0.5	43.1±0.6	42.9±0.5	44.7±0.5
TVSum	vsLSTM	56.9±0.5	57.0±0.5	57.9±0.5	58.0±0.5
	dppLSTM	58.7±0.4	58.9±0.4	59.6±0.4	59.7±0.5

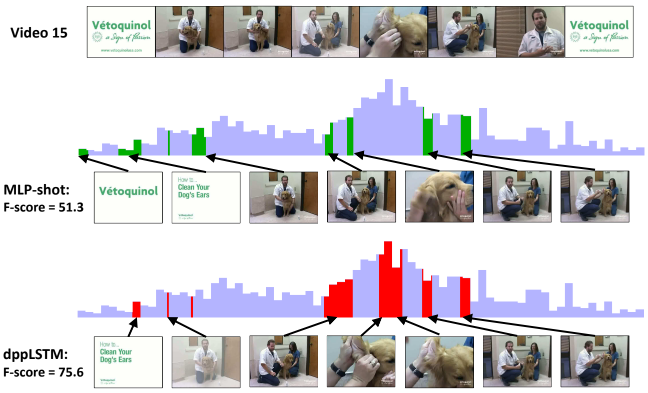


Fig. 4. Exemplar video summaries by MLP-Shot and dppLSTM, along with the ground-truth importance (blue background). See texts for details. We index videos as in [35].

The height of the blue background indicates the ground-truth frame-level importance scores of the video. The marked red and green intervals are the ones selected by dppLSTM and MLP-Shot as the summaries, respectively. dppLSTM can capture temporal dependencies and thus identify the most important part in the video, i.e. the frame depicting the cleaning of the dog's ears. MLP-Shot, however, completely misses selecting such subshots even though those subshots have much higher ground-truth importance scores than the neighboring frames. We believe this is because MLP-Shot does not capture the sequential semantic flow properly and lacks the knowledge that if the neighbor frames are important, then the frames in the middle could be important too.

It is also very interesting to note that despite the fact that DPP models usually eliminate similar elements, dppLSTM can still select similar but important subshots: subshots of two people with dogs before and after cleaning the dog's ear are both selected. This highlights dppLSTM's ability to adaptively model long-range (distant states) dependencies.

Fig. 5 shows a failure case of dppLSTM. This video is an outdoor ego-centric video and records very diverse contents. In particular, the scenes change among a sandwich shop, building, food, and the town square. From the summarization results we see that dppLSTM still selects diverse contents, but fails to capture

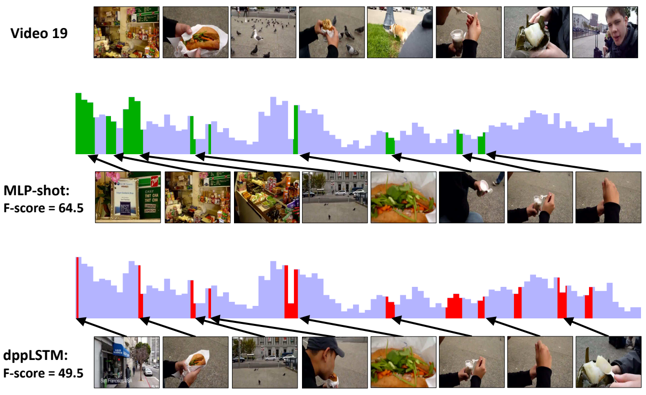


Fig. 5. A failure case by dppLSTM. See texts for details. We index videos as in [35].

the beginning frames — those frames all have high importance scores and are visually similar but are *temporally clustered crowdedly*. In this case, dppLSTM is forced to eliminate some of them, resulting in low recall. On the other hand, MLP-Shot needs only to predict importance scores without being diverse, which leads to higher recall and F-scores. Interestingly, MLP-Shot predicts poorly towards the end of the video, whereas the repulsiveness modeled by dppLSTM gives the method an edge to select a few frames in the end of the video.

In summary, we expect our approaches to work well on videos whose contents change smoothly (at least within a short interval) such that the temporal structures can be well captured. For videos with rapid changing and diverse contents, higher-level semantic cues (e.g., object detection as in [5,9]) could be complementary and should be incorporated.

5 Conclusion

Our work explores Long Short-Term Memory to develop novel supervised learning approaches to automatic video summarization. Our LSTM-based models outperform competing methods on two challenging benchmarks. There are several key contributing factors: the modeling capacity by LSTMs to capture variable-range inter-dependencies, as well as our idea to complement LSTMs' strength with DPP to explicitly model inter-frame repulsiveness to encourage diverse selected frames. While LSTMs require a large number of annotated samples, we show how to mediate this demand by exploiting the existence of other annotated video datasets, despite their heterogeneity in style and content. Preliminary results are very promising, suggesting future research directions of developing more sophisticated techniques that can bring together a vast number of available video datasets for video summarization. In particular, it would be very productive to explore new sequential models that can enhance LSTMs' capacity in modeling video data, by learning to encode semantic understanding of video contents and using them to guide summarization and other tasks in visual analytics.

Acknowledgements KG is partially supported by NSF IIS-1514118 and a gift from Intel. Others are partially supported by USC Graduate Fellowships, NSF IIS-1451412, 1513966, CCF-1139148 and A. P. Sloan Research Fellowship.

References

1. Zhang, H.J., Wu, J., Zhong, D., Smoliar, S.W.: An integrated system for content-based video retrieval and browsing. Pattern recognition **30**(4) (1997) 643–658 1, 2, 3
2. Gong, B., Chao, W.L., Grauman, K., Sha, F.: Diverse sequential subset selection for supervised video summarization. In: NIPS. (2014) 1, 2, 3, 4, 6, 7, 8, 10, 20, 23
3. Mundur, P., Rao, Y., Yesha, Y.: Keyframe-based video summarization using delaunay clustering. International Journal on Digital Libraries **6**(2) (2006) 219–232 1, 3, 4
4. Liu, D., Hua, G., Chen, T.: A hierarchical visual model for video object summarization. IEEE transactions on pattern analysis and machine intelligence **32**(12) (2010) 2178–2190 1
5. Lee, Y.J., Ghosh, J., Grauman, K.: Discovering important people and objects for egocentric video summarization. In: CVPR. (2012) 1, 2, 3, 14
6. Ngo, C.W., Ma, Y.F., Zhang, H.: Automatic video summarization by graph modeling. In: ICCV. (2003) 1, 2, 3
7. 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) 1
8. Nam, J., Tewfik, A.H.: Event-driven video abstraction and visualization. Multimedia Tools and Applications **16**(1-2) (2002) 55–77 1
9. Lu, Z., Grauman, K.: Story-driven summarization for egocentric video. In: CVPR. (2013) 1, 2, 3, 14
10. Hong, R., Tang, J., Tan, H.K., Yan, S., Ngo, C., Chua, T.S.: Event driven summarization for web videos. In: SIGMM Workshop. (2009) 2, 3
11. Khosla, A., Hamid, R., Lin, C.J., Sundaresan, N.: Large-scale video summarization using web-image priors. In: CVPR. (2013) 2, 3, 10, 11
12. Liu, T., Kender, J.R.: Optimization algorithms for the selection of key frame sequences of variable length. In: ECCV. (2002) 2, 3, 4
13. Kang, H.W., Matsushita, Y., Tang, X., Chen, X.Q.: Space-time video montage. In: CVPR. (2006) 2, 3
14. Ma, Y.F., Lu, L., Zhang, H.J., Li, M.: A user attention model for video summarization. In: ACM Multimedia. (2002) 2, 3
15. Gygli, M., Grabner, H., Van Gool, L.: Video summarization by learning submodular mixtures of objectives. In: CVPR. (2015) 2, 3, 4, 9, 10, 11, 21
16. Zhang, K., Chao, W.l., Sha, F., Grauman, K.: Summary transfer: Exemplar-based subset selection for video summarization. In: CVPR. (2016) 2, 3, 6, 10, 11, 22, 23
17. Gygli, M., Grabner, H., Riemenschneider, H., Van Gool, L.: Creating summaries from user videos. In: ECCV. (2014) 2, 3, 4, 8, 9, 10, 11, 20, 21
18. Chao, W.L., Gong, B., Grauman, K., Sha, F.: Large-margin determinantal point processes. In: UAI. (2015) 2, 3, 6

19. Deng, L., Hinton, G., Kingsbury, B.: New types of deep neural network learning for speech recognition and related applications: An overview. In: ICASSP. (2013) 8599–8603 2
20. Graves, A., Mohamed, A.r., Hinton, G.: Speech recognition with deep recurrent neural networks. In: ICASSP. (2013) 6645–6649 2, 4
21. Graves, A., Jaitly, N.: Towards end-to-end speech recognition with recurrent neural networks. In: ICML. (2014) 1764–1772 2, 5
22. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: CVPR. (2015) 2625–2634 2
23. Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., Courville, A.: Describing videos by exploiting temporal structure. In: ICCV. (2015) 4507–4515 2, 4
24. Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., Saenko, K.: Sequence to sequence-video to text. In: ICCV. (2015) 4534–4542 2, 4
25. Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., Saenko, K.: Translating videos to natural language using deep recurrent neural networks. CVPR (2014) 2, 4
26. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. In: CVPR. (2015) 3128–3137 2
27. Kulesza, A., Taskar, B.: Determinantal point processes for machine learning. Foundations and Trends in Machine Learning **5**(2–3) (2012) 3, 6, 8, 21, 22
28. 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 Recognition Letters **32**(1) (2011) 56–68 3, 4, 8, 20, 23
29. Furini, M., Geraci, F., Montangero, M., Pellegrini, M.: Stimo: Still and moving video storyboard for the web scenario. Multimedia Tools and Applications **46**(1) (2010) 47–69 3, 4
30. Li, Y., Merialdo, B.: Multi-video summarization based on video-mmr. In: WIAMIS Workshop. (2010) 3, 10, 11
31. Potapov, D., Douze, M., Harchaoui, Z., Schmid, C.: Category-specific video summarization. In: ECCV. (2014) 3, 4, 18, 19, 20, 21, 23
32. Morere, O., Goh, H., Veillard, A., Chandrasekhar, V., Lin, J.: Co-regularized deep representations for video summarization. In: ICIP. (2015) 3165–3169 3
33. Kim, G., Xing, E.P.: Reconstructing storyline graphs for image recommendation from web community photos. In: CVPR. (2014) 3
34. Zhao, B., Xing, E.P.: Quasi real-time summarization for consumer videos. In: CVPR. (2014) 3, 11
35. Song, Y., Vallmitjana, J., Stent, A., Jaimes, A.: Tvsum: Summarizing web videos using titles. In: CVPR. (2015) 3, 4, 8, 9, 10, 11, 12, 13, 14, 19, 20, 21
36. Chu, W.S., Song, Y., Jaimes, A.: Video co-summarization: Video summarization by visual co-occurrence. In: CVPR, IEEE (2015) 3
37. 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) 4633–4641 4
38. Xu, K., Ba, J., Kiros, R., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. ICML (2015) 4
39. Jin, J., Fu, K., Cui, R., Sha, F., Zhang, C.: Aligning where to see and what to tell: image caption with region-based attention and scene factorization. arXiv preprint arXiv:1506.06272 (2015) 4

40. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: Continual prediction with lstm. Neural computation **12**(10) (2000) 2451–2471 4
41. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8) (1997) 1735–1780 4
42. Zaremba, W., Sutskever, I.: Learning to execute. arXiv preprint arXiv:1410.4615 (2014) 4
43. Wolf, W.: Key frame selection by motion analysis. In: ICASSP. (1996) 4
44. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm networks. In: IJCNN. (2005) 2047–2052 5
45. Kulesza, A., Taskar, B.: Learning determinantal point processes. In: UAI. (2011) 7
46. Buchbinder, N., Feldman, M., Seffi, J., Schwartz, R.: A tight linear time (1/2)-approximation for unconstrained submodular maximization. SIAM Journal on Computing **44**(5) (2015) 1384–1402 8, 21
47. : Open video project. `http://www.open-video.org/` 8, 20
48. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR. (2015) 1–9 9
49. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: ECCV. (2010) 213–226 12
50. Gong, B., Shi, Y., Sha, F., Grauman, K.: Geodesic flow kernel for unsupervised domain adaptation. In: CVPR. (2012) 2066–2073 12
51. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. In: ICML. (2014) 647–655 12
52. Sun, B., Feng, J., Saenko, K.: Return of frustratingly easy domain adaptation. AAAI (2016) 12

Supplementary Material: Video Summarization with Long Short-term Memory

In this Supplementary Material, we provide details omitted in the main text:

– Section A: converting between different formats of ground-truth annotations (Section 3.1 in the main text)
– Section B: details of the datasets (Section 4.1 in the main text)
– Section C: details of our LSTM-based models, including the learning objective for dppLSTM and the generating process of shot-based summaries for both vsLSTM and dppLSTM (Section 3.4 and 3.5 in the main text)
– Section D: comparing different network structures for dppLSTM (Section 3.4 in the main text)
– Section E: Other implementation details
– Section F: Additional discussions on video summarization

A Converting between different formats of ground-truth annotations

As mentioned in Section 3.1 of the main text, existing video summarization datasets usually provide the ground-truth annotations in (one of) the following three formats — (a) selected keyframes, (b) interval-based keyshots, and (c) frame-level importance scores. See Table 1 for illustration.

In order to combine multiple datasets to enlarge the training set, or to enable any (supervised) video summarization algorithm to be trained under different ground-truth formats, we introduce a general procedure to convert between different formats. *Note that we perform this procedure to the ground truths only in the training phase.* In the testing phase, we directly compare with the user-generated summaries in their original formats, unless stated otherwise (see Section B). Also note that certain conversions require *temporal segmentation* to cut a video into disjoint time intervals, where each interval contains frames of similar contents. Since none of the datasets involved in the experiments provides *ground-truth temporal segmentation*, we apply the kernel temporal segmentation (KTS) proposed by Potapov *et al.* [31]. The resulting intervals are around 5 seconds on average.

Table 1. Illustration of different formats of ground-truth annotations for video summarization. We take a 6-frame sequence as an example.

Format	Description
(a) keyframes	{frame 2, frame 6} or [0 1 0 0 0 1]
(b) interval-based keyshots	{frames 1–2, frames 5–6} or [1 1 0 0 1 1]
(c) frame-level importance scores	[0.5 0.9 0.1 0.2 0.7 0.8]

A.1 keyframes → keyshots and frame-level scores

To covert keyframes into keyshots, we first *temporally segment* a video into disjoint intervals using KTS [31]. Then if an interval contains at least one keyframe, we view such an interval as a keyshot, and mark all frames of it with score 1; otherwise, 0.

To prevent generating too many keyshots, we rank the candidate intervals (those with at least one keyframe) in the descending order by the number of key frames each interval contains divided by its duration. We then select intervals in order so that the total duration of keyshots is below a certain threshold (e.g., using the knapsack algorithm as in [35]).

A.2 keyshots → keyframes and frame-level scores

Given the selected keyshots, we can randomly pick a frame, or pick the middle frame, of each keyshot to be a keyframe. We also directly mark frames contained in keyshots with score 1. For those frames not covered by any keyshot, we set the corresponding importance scores to be 0.

A.3 frame-level scores → keyframes and keyshots

To convert frame-level importance scores into keyshots, we first perform *temporal segmentation*, as in Section A.1. We then compute interval-level scores by averaging the scores of frames within each interval. We then rank intervals in the descending order by their scores, and select them in order so that the total duration of keyshots is below a certain threshold (e.g., using the knapsack algorithm as in [35]). We further pick the frame with the highest importance score within each keyshot to be a keyframe.

Table 2 summarizes the conversions described above.

Table 2. Illustration of the converting procedure described in Section A.1–A.3. We take a 6-frame sequence as an example, and assume that the temporal segmentation gives three intervals, {frames 1–2, frames 3–4, frames 5–6}. The threshold of duration is 5.

Conversion	Description
Section A.1	(a) [0 1 0 0 0 1] → (b) [1 1 0 0 1 1], (c) [1 1 0 0 1 1]
Section A.2	(b) [1 1 0 0 1 1] → (a) [0 1 0 0 0 1], (c) [1 1 0 0 1 1]
Section A.3	(c) [0.5 0.9 0.1 0.2 0.7 0.8] → (b) [1 1 0 0 1 1], (a) [0 1 0 0 0 1]

(a) keyframes (b) interval-based keyshots (c) frame-level importance scores

B Details of the datasets

In this section, we provide more details about the four datasets — **SumMe** [17], **TVSum** [35], **OVP** [47,28], and **Youtube** [28] — involved in the experiments. Note that **OVP** and **Youtube** are only used to augment the training set.

B.1 Training ground truths

Table 3 lists the training and testing ground truths provided in each dataset. Note that in training, we require a single ground truth for each video, which is directly given in **SumMe** and **TVSum**, but not in **OVP** and **Youtube**. We thus follow [2] to create a single ground-truth set of keyframes from multiple user-annotated ones for each video.

Table 4 summarizes the formats of training ground truths required by our proposed methods (vsLSTM, dppLSTM) and baselines (MLP-Shot, MLP-Frame). We perform the converting procedure described in Section A to obtain the required training formats if they are not provided in the dataset. We perform KTS [31] for temporal segmentation for all datasets.

B.2 Testing ground truths for TVSum

TVSum provides for each video multiple sequence of frame-level importance scores annotated by different users. We follow [35] to convert each sequence into a keyshot-based summary for evaluation, which is exactly the one in Section A.3. We set the threshold to be 15% of the original video length, following [35].

Table 3. Training and testing ground truths provided for each video in the datasets.

Dataset	Training ground truths	Testing ground truths
SumMe	a sequence of frame-level scores	multiple sets of keyshots
TVSum	a sequence of frame-level scores	multiple sequences of frame-level scores†
OVP	multiple sets of keyframes‡	-
Youtube	multiple sets of keyframes‡	-

† following [35], we convert the frame-level scores into keyshots for evaluation.
‡ following [2], we create a single ground-truth set of keyframes for each video.

C Details of our LSTM-based models

In this section, we provide more details about the proposed LSTM-based models for video summarization.

Table 4. The formats of training ground truths required by vsLSTM, dppLSTM, MLP-Shot, and MLP-Frame.

Method	Training ground truths
MLP-Shot	shot-level importance scores[†]
MLP-Frame	frame-level importance scores
vsLSTM	frame-level importance scores
dppLSTM	keyframes, frame-level importance scores[‡]

[†] The shot-level importance scores are derived as the averages of the corresponding frame-level importance scores. We perform KTS [31] to segment a video into shots (disjoint intervals).
[‡] We pre-train the MLP $f_I(\cdot)$ and the LSTM layers using frame-level scores.

C.1 The learning objective of dppLSTM

As mentioned in Section 3.4 of the main text, we adopt a stage-wise optimization routine to learn dppLSTM — the first stage is based on the prediction error of importance scores; the second stage is based on the maximum likelihood estimation (MLE) specified by DPPs. Denote Z as a ground set of N items (e.g, all frames of a video), and $\boldsymbol{z}^* \subset \mathsf{Z}$ as the target subset (e.g., the subset of keyframes). Given the N $\times$ N kernel matrix $\boldsymbol{L}$, the probability to sample $\boldsymbol{z}^*$ is

$$P(\boldsymbol{z}^* \subset \mathsf{Z}; \boldsymbol{L}) = \frac{\det(\boldsymbol{L}_{\boldsymbol{z}^*})}{\det(\boldsymbol{L} + \boldsymbol{I})}, \tag{1}$$

where $\boldsymbol{L}_{\boldsymbol{z}^*}$ is the principal minor indexed by $\boldsymbol{z}^*$, and $\boldsymbol{I}$ is the N $\times$ N identity matrix.

In dppLSTM, $\boldsymbol{L}$ is parameterized by $\boldsymbol{\theta}$, which includes all parameters in the model. In the second stage, we learn $\boldsymbol{\theta}$ using MLE [27]

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \sum_i \log\{P(\boldsymbol{z}^{(i)*} \subset \mathsf{Z}^{(i)}; \boldsymbol{L}^{(i)}(\boldsymbol{\theta}))\}, \tag{2}$$

where i indexes the target subset, ground set, and $\boldsymbol{L}$ matrix of the i-th video. We optimize $\boldsymbol{\theta}$ with stochastic gradient descent.

C.2 Generating shot-based summaries for vsLSTM and dppLSTM

As mentioned in Section 3.1 and 3.5 of the main text, the outputs of both our proposed models are on the frame level — vsLSTM predicts frame-level importance scores, while dppLSTM selects a subset of keyframes using approximate MAP inference [46]. To compare with the user-annotated keyshots in **SumMe** and **TVSum** for evaluation, we convert the outputs into keyshot-based summaries.

For vsLSTM, we directly apply the conversion in Section A.3. We set the threshold of the total duration of keyshots to be 15% of the original video length (for both datasets), following the protocols in [35,17,15].

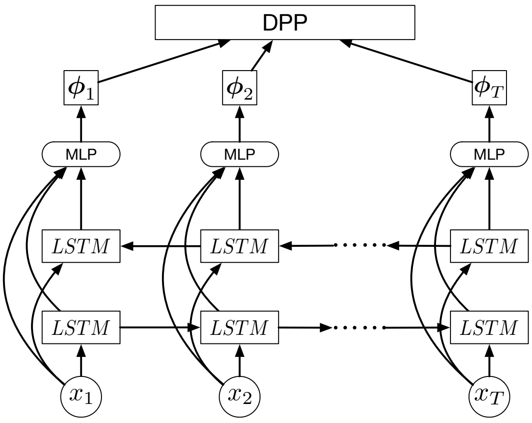


Fig. 1. Our dppLSTM-single model. It is similar to dppLSTM (Fig. 3 in the main text) but learns only a single MLP $f_S(\cdot)$ and then stacks with a DPP.

For dppLSTM, we apply the conversion in Section A.1. In practice, DPP inference usually leads to *high precision* yet *low recall*; i.e., the resulting total duration of keyshots may be far below the threshold (on average, 10%). We thus add in few more keyshots by utilizing the scalar output of the MLP $f_I(\cdot)$, following the procedure in Section A.3. The MLP $f_I(\cdot)$ is pre-trained using the frame-level importance scores (cf. Section 3.4 of the main text) and conveys a certain notion of importance even after fine-tuning with the DPP objective.

D Comparing different network structures for dppLSTM

The network structure of dppLSTM (cf. Fig. 3 of the main text) involves two MLPs — the MLP $f_I(\cdot)$ outputting y_t for frame-level importance and the MLP $f_S(\cdot)$ outputting $\boldsymbol{\phi}_t$ for similarity.

In this section, we compare with another LSTM-based model that learns only a single MLP $f_S(\cdot)$ and then stacks with a DPP. We term this model as dppLSTM-single. See Fig. 1 for illustration. dppLSTM-single also outputs a set of keyframes and is likely to generate a keyshot-based summary of an insufficient duration (similar to dppLSTM in Section C.2). We thus add in few more keyshots by utilizing the diagonal values of $\boldsymbol{L}$ as frame-level scores, following [16].

Table 5 compares the performance of the two network structures, and dppLSTM obviously outperforms dppLSTM-single. As a well-learned DPP model should capture the notions of both quality (importance) and diversity [27], we surmise that separately modeling the two factors would benefit, especially when the model of each factor can be pre-trained (e.g, the MLP $f_I(\cdot)$ in dppLSTM).

Table 5. Comparison between dppLSTM and dppLSTM-single on different settings.

Dataset	Method	Canonical	Augmented	Transfer
SumMe	dppLSTM	38.6±0.8	42.9±0.5	41.8±0.5
	dppLSTM-single	37.5±0.9	41.4±0.8	40.3±0.9
TVSum	dppLSTM	54.7±0.7	59.6±0.4	58.7±0.4
	dppLSTM-single	53.9±0.9	57.5±0.7	56.2±0.8

E Other implementation details

In this section, we provide the implementation details for both the proposed models (vsLSTM, dppLSTM) and baselines (MLP-Frame, MLP-Shot).

E.1 Input signal

For vsLSTM, dppLSTM, and MLP-Frame, which all take frame features as inputs, we uniformly subsample the videos to 2 fps[1]. The concatenated feature (of a 5-frame window) to MLP-Frame is thus equivalent to taking a 2-second span into consideration. For MLP-Shot, we perform KTS [31] to segment the video into shots (disjoint intervals), where each shot is around 5 seconds on average.

E.2 Network structures

$f_I(\cdot)$ and $f_S(\cdot)$ are implemented by one-hidden-layer MLPs, while MLP-Shot and MLP-Frame are two-hidden-layer MLPs. For all models, we set the size of each hidden layer of MLPs, the number of hidden units of each unidirectional LSTM, and the output dimension of the MLP $f_S(\cdot)$ all to be 256. We apply the sigmoid activation function to all the hidden units as well as the output layer of MLP-Shot, MLP-Frame, and $f_I(\cdot)$. The output layer of $f_S(\cdot)$ are of linear units. We run for each setting and each testing fold (cf. Section 4.2 of the main text) 5 times and report the average and standard deviation.

E.3 Learning objectives

For MLP-Frame, MLP-Shot, vsLSTM, and the first stage of dppLSTM, we use the square loss. For dppLSTM-single and the second stage of dppLSTM, we use the likelihood (cf. (2)).

E.4 Stopping criteria

For all our models, we stop training after K consecutive epochs with descending summarization F-score on the validation set. We set $K = 5$.

[1] For videos with slow varying contents such as SumMe/TVSum, this scheme seems adequate. For OVP and YouTube, even 1 fps is sufficient for fairly good summarization [28,2,16].

F Additional discussions on video summarization

Video summarization is essentially a structured prediction problem and heavily relies on how to model/capture the sequential (or temporal) structures underlying videos. In this work, we focus on modeling the structures making sequentially inter-dependent decisions at three levels: (a) realizing boundaries of subevents/shots; (b) removing redundant nearby shots/frames; (c) retaining temporally distant events despite being visually similar (cf. the motivating example of “leave home” in Section 1 of the main text). Essentially, any decision including or excluding frames is dependent on other decisions made on a temporal line.