In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

Less is More: Learning Highlight Detection from Video Duration

Bo Xiong[1], Yannis Kalantidis[2], Deepti Ghadiyaram[2], Kristen Grauman[3*]
[1] University of Texas at Austin, [2] Facebook AI, [3] Facebook AI Research
bxiong@cs.utexas.edu,{yannisk, deeptigp, grauman}@fb.com

Abstract

Highlight detection has the potential to significantly ease video browsing, but existing methods often suffer from expensive supervision requirements, where human viewers must manually identify highlights in training videos. We propose a scalable unsupervised solution that exploits video duration *as an implicit supervision signal. Our key insight is that video segments from shorter user-generated videos are more likely to be highlights than those from longer videos, since users tend to be more selective about the content when capturing shorter videos. Leveraging this insight, we introduce a novel ranking framework that prefers segments from shorter videos, while properly accounting for the inherent noise in the (unlabeled) training data. We use it to train a highlight detector with 10M hashtagged Instagram videos. In experiments on two challenging public video highlight detection benchmarks, our method substantially improves the state-of-the-art for unsupervised highlight detection.*

1. Introduction

"I didn't have time to write a short letter, so I wrote a long one instead." – Mark Twain

With the increasing prevalence of portable computing devices (like smartphones, wearables) and promotion from social media platforms, it is seamless for Internet users to record and share massive amounts of video. According to Cisco [1], by 2021 video traffic will be 82% of all consumer Internet traffic, and every second a million minutes of video content will cross the network. Yet, indexing, organizing, and even browsing such massive video data is still very challenging.

As an attempt to mitigate the overload, *video highlight detection* has attracted increasing attention in the research community. The goal in highlight detection is to retrieve a moment—in the form of a short video clip—that captures a user's primary attention or interest within an unedited video. A well-selected highlight can accelerate browsing many videos (since a user quickly previews the most important content), enhance social video sharing (since friends become encouraged to watch further), and facilitate video recommendation (since systems can relate unedited videos in a more focused way). Highlight detectors are typically *domain-specific* [33, 40, 39, 28, 26, 20], meaning they are tailored to a category of video or keywords/tags like skiing, surfing, etc. This accounts for the fact that the definition of what constitutes a highlight often depends on the domain, e.g., a barking dog might be of interest in a dog show video, but not in a surfing video.

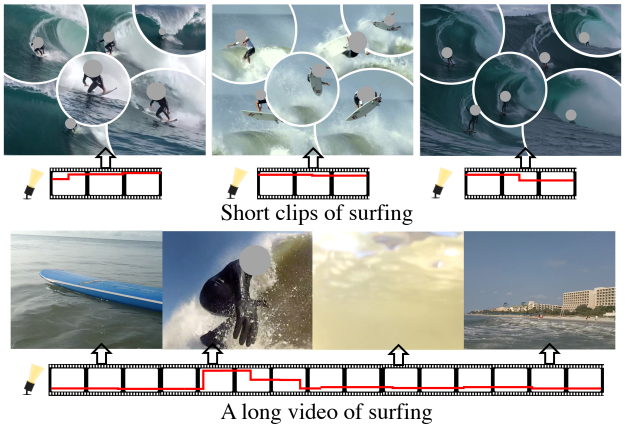

Figure 1: Video frames from three shorter user-generated video clips (top row) and one longer user-generated video (second row). Although all recordings capture the same event (surfing), video segments from shorter user-generated videos are more likely to be highlights than those from longer videos, since users tend to be more selective about their content. The height of the red curve indicates highlight score over time. We leverage this natural phenomenon as a free latent supervision signal in large-scale Web video.

Existing methods largely follow one of two strategies. The first strategy poses highlight detection as a supervised learning task [9, 33, 40]. Given unedited videos together with manual annotations for their highlights, a ranker is trained to score highlight segments higher than those else-

[*] *On leave from UT Austin (grauman@cs.utexas.edu).*

where in the video [9, 33, 40]. While the resulting detector has the advantage of good discriminative power, the approach suffers from heavy, non-scalable supervision requirements. The second strategy instead considers highlight learning as a weakly supervised recognition task. Given domain-specific videos, the system discovers what appears commonly among the training samples, and learns to detect such segments as highlights in novel videos for the same domain [39, 28, 26, 20]. While more scalable in supervision, this approach suffers from a lack of discriminative power. Put simply, repetition across samples does not entail importance. For example, while all dog show videos might contain moments showing the audience waiting in their seats, that does not make it a highlight.

We introduce a novel framework for domain-specific highlight detection that addresses both these shortcomings. Our key insight is that user-generated videos, such as those uploaded to Instagram or YouTube, carry a latent supervision signal relevant for highlight detection: their duration. We hypothesize shorter user-uploaded videos tend to have a key focal point as the user is more selective about the content, whereas longer ones may not have every second be as crisp or engaging. In the spirit of Twain's quote above, more effort is required to film only the significant moments, or else manually edit them out later. Hence duration is an informative, though implicit, training signal about the value of the video content. See Fig.1. We leverage duration as a new form of "weak" supervision to train highlight detectors with unedited videos. Unlike existing supervised methods, our training data requirements are scalable, relying only on tagged video samples from the Web. Unlike existing weakly supervised methods, our approach can be trained discriminatively to isolate highlights from non-highlight time segments.

Given a category (domain) name, we first query Instagram to mine public videos which contain the given category name as hashtags. We use a total of 10M Instagram videos. Since the hashtag Instagram videos are very noisy, and since even longer videos will contain some highlights, we propose a novel ranking model that is robust to label noise in the training data. In particular, our model introduces a latent variable to indicate whether each training pair is valid or noisy. We model the latent variable with a neural network, and train it jointly with the ranking function for highlight detection. On two public challenging benchmark datasets (TVSum [31] and YouTube Highlights [33]), we demonstrate our approach improves the state of the art for domain-specific unsupervised highlight detection.[1]

Overall, we make the following contributions:

- We propose a novel approach to unsupervised video highlight detection that leverages user-generated video duration as an implicit training signal.
- We propose a novel video clip deep ranking framework that is robust to noisily labeled training data.
- We train on a large-scale dataset that is one to two orders of magnitude larger than existing ones, and show that the scale (coupled with the scalablility of our model) is crucial to success.
- On two challenging public benchmarks, our method substantially improves the state of the art for unsupervised highlight detection, e.g., improving the next best existing method by 22%.

2. Related Work

Video Highlight Detection Many prior approaches focus on highlight detection for sports video [30, 37, 34, 35]. Recently, supervised video highlight detection has been proposed for Internet videos [33] and first-person videos [40]. These methods all require human annotated $\langle$highlight, source video$\rangle$ pairs for each specific domain. The Video2GIF approach [9] learns from GIF-video pairs, which are also manually created. All supervised highlight detection methods require human edited/labeled ranking pairs. In contrast, our method does not use manually labeled highlights. Our work offers a new way to take advantage of freely available videos from the Internet.

Unsupervised video highlight detection methods do not require video annotations to train. They can be further divided into methods that are domain-agnostic or domain-specific. Whereas a domain-agnostic approach like motion strength [24] operates uniformly on any video, domain-specific methods train on a collection of videos of the same topic. They leverage concepts like visual co-occurrence [5], category-aware reconstruction loss [44, 39], or collaborative sparse selection within a category [27]. Another approach is first train video category classifiers, then detect highlights based on the classifier scores [28] or spatial-temporal gradients from the classifier [26]. Like the domain-specific methods, our approach also tailors highlights to the topic domain; we gather the relevant training videos per topic automatically using keyword search on the Web. Unlike any existing methods, we leverage video duration as a weak supervision signal.

Video Summarization Whereas highlight detection (our goal) aims to score individual video segments for their worthiness as highlights, *video summarization* aims to provide a complete synopsis of the whole video, often in the form of a structured output, e.g., a storyline graph [15, 36], a sequence of selected keyframes [17] or clips [7, 43]. Video

[1] Throughout, we use the term *unsupervised* to indicate the method does not have access to any manually created summaries for training. We use the term *domain-specific* to mean that there is a domain/category of interest specified by keyword(s) like "skiing", following [28, 39, 26, 20].

summarization is often formalized as a structured subset selection problem considering not just importance but also diversity [6, 21] and coherency [21]. Supervised summarization methods focus on learning a visual interestingness/importance score [17, 7], submodular mixtures of objectives [8, 38], or temporal dependencies [42, 43]. Unsupervised summarization methods often focus on low-level visual cues to locate important segments. Recent unsupervised and semi-supervised methods use recurrent auto-encoders to enforce that the summary sequence should be able to generate a sequence similar to the original video [39, 23, 43]. Many rely on Web image priors [13, 31, 14, 15] or semantic Web video priors [3]. While we also leverage Web data, our idea about duration is novel.

Learning with Noisy Labels: Our work is also related to learning from noisy data, a topic of broad interest in machine learning [25, 19]. The proportion SVM [41] handles noisy data for training SVMs where a fraction of the labels per group are expected to be incorrect, with applications to activity recognition [16]. Various methods explore how to train neural networks with noisy data [32, 29, 18]. Recent work on attention-based Multiple Instance Learning (MIL) helps focus on reliable instances using a differentiable MIL pooling operation for bags of embeddings [12]. Inspired by this, we propose a novel attention-based loss to reliably identify valid samples from noisy training data, but unlike [12], 1) we have "bags" defined in the space of ranking constraints, 2) our attention is defined in the loss space, not in the feature space, 3) our model predicts scores at the instance level, not the "bag" level, and 4) our attention mechanism is extended with multiple heads to take into account a prior for the expected label noise level.

3. Approach

We explore domain-specific highlight detection trained with unlabeled videos. We first describe how we automatically collect large-scale hashtag video data for a domain (Sec. 3.1). Then we present our novel framework for learning highlights aided by duration as a training signal (Sec. 3.2). The results will show the impact of our method to find highlights in standard public benchmarks (Sec. 4).

3.1. Large-scale Instagram Training Video

First we describe our data collection process. We choose Instagram as our source to collect videos because it contains a large amount of public videos associated with hashtags. In addition, because Instagram users tend to upload frequently via mobile for social sharing, there is a natural variety of duration and quality—some short and eye-catching videos, others less focused. The duration of a video from Instagram can vary from less than a second to 1 minute.

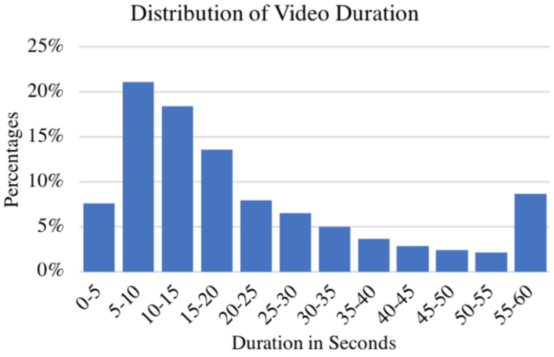

Figure 2: Durations for the 10M Instagram training videos.

Our goal is to build domain-specific highlight detectors. Given a category name, we query Instagram to mine for videos that contain the given category name among their hashtags. For most categories, this returns at least $200,000$ videos. Since we validate our approach to detect highlights in the public TVSum and YouTube Highlights benchmarks [31, 33] (see Sec. 4), the full list of hashtags queried are *dog, gymnastics, parkour, skating, skiing, surfing, changing vehicle tire, getting vehicle unstuck, grooming an animal, making sandwich, parade, flash mob gathering, beekeeping, attempting bike tricks,* and *dog show*. Thus the data spans a range of domains frequently captured for sharing on social media or browsing for how-to's online. Altogether we acquire more than 10M training videos.

Figure 2 shows the distribution of their durations, which vary from less than a second to 1 minute. We see there is a nice variety of lengths, with two modes centered around short ($\sim$ 10 s) and "long" ($\sim$ 60 s) clips.

Postprocessing hashtags, injecting word similarity models, or chaining to related keywords could further refine the quality of the domain-specific data [22]. However, our experiments suggest that even our direct hashtag mining is sufficient to gather data relevant to the public video datasets we ultimately test on. Below we will present a method to cope with the inherent noise in both the Instagram tags as well as the long/short video hypothesis.

3.2. Learning Highlights from Video Duration

Next we introduce our ranking model that utilizes large-scale hashtagged video data and their durations for training video highlight detectors.

Recall that a video highlight is a short video segment within a longer video that would capture a user's attention and interest. Our goal is to learn a function $f(x)$ that infers the highlight score of a temporal video segment given its feature x (to be specified below). Then, given a novel video, its highlights can be prioritized (ranked) based on each segment's predicted highlight score.

A supervised regression solution would attempt to learn $f(x)$ from a video dataset with manually annotated highlight scores. However, calibrating highlight scores collected from multiple human annotators is itself challenging. Instead, highlight detection can be formalized as a *ranking* problem by learning from human-labeled/edited video-highlight pairs [9, 33, 40]: segments in the manually annotated highlight ought to score more highly than those elsewhere in the original long video. However, such paired data is difficult and expensive to collect, especially for long and unconstrained videos at a large scale.

To circumvent the heavy supervision entailed by collecting video-highlight pairs, we propose a framework to learn highlight detection directly from a large collection of *unlabeled* video. As discussed above, we hypothesize that users tend to be more selective about the content in the shorter videos they upload, whereas their longer videos may be a mix of good and less interesting content. We therefore use the duration of videos as supervision signal. In particular, we propose to learn a scoring function that ranks video segments from shorter videos higher than video segments from longer videos. Since longer videos could also contain highlight moments, we devise the ranking model to effectively handle noisy ranking data.

Training data and loss: Let D denote a set of videos sharing a tag (e.g., *dog show*). We first partition D into three non-overlapping subsets $D = \{D_S, D_L, D_R\}$, where D_S contains shorter videos, D_L contains longer videos, and D_R contains the rest. For example, shorter videos may be less than 15 seconds, and longer ones more than 45 seconds (cf. Sec 4). Each video, whether long or short, is broken into uniform length temporal segments.[2]

Let s_i refer to a unique video segment from the dataset, and let $v(s_i)$ denote the video where video segment s_i comes from. The visual feature extracted from segment s_i is x_i. Since our goal is to rank video segments from shorter videos higher than those from longer videos, we construct training pairs (s_i, s_j) such that $v(s_i) \in D_s$ and $v(s_j) \in D_L$. We denote the collection of training pairs as $\mathcal{P}$. Since our dataset is large, we sample among all possible pairs, ensuring each video segment is included at least once in the training set. The learning objective consists of the following ranking loss:

$$L(D) = \sum_{(s_i,s_j)\in\mathcal{P}} \max\left(0, 1 - f(x_i) + f(x_j)\right), \qquad (1)$$

which says we incur a loss every time the longer video's segment scores higher. The function f is a deep convolutional network, detailed below. Note that whereas supervised highlight ranking methods [9, 33, 40] use rank constraints on segments from the *same* video—comparing those inside and outside the true highlight region—our constraints span segments from distinct short and long videos.

Learning from noisy pairs: The formulation thus far assumes that no noise exists and that D_s and D_L only contain segments from highlights and non-highlights, respectively. However, this is not the case when learning from unedited videos: some video segments from long videos can also be highlights, and some short segments need not be highlights. Furthermore, some videos are irrelevant to the hashtags. Therefore, only a subset of our pairs in $\mathcal{P}$ have *valid* ranking constraints (s_i, s_j), i.e., pairs where s_i corresponds to a highlight and s_j corresponds to a non-highlight. Ideally, a ranking model would only learn from valid ranking constraints and ignore the rest. To achieve this without requiring any annotation effort, we introduce binary latent variables w_{ij}, $\forall(s_i, s_j) \in \mathcal{P}$ to indicate whether a ranking constraint is valid. We rewrite the learning objective as follows:

$$\begin{aligned} L(D) = &\sum_{(s_i,s_j)\in\mathcal{P}} w_{ij} \;\; \max\left(0, 1 - f(x_i) + f(x_j)\right) \\ &\text{s.t.} \sum_{(s_i,s_j)\in\mathcal{P}} w_{ij} = p|\mathcal{P}|, \quad w_{ij} \in [0,1], \qquad (2) \\ &\text{and } \; w_{ij} = h(x_i, x_j) \end{aligned}$$

where h is a neural network, $|\mathcal{P}|$ is total number of ranking constraints, and p is the anticipated proportion of ranking constraints that are valid. In the spirit of learning with a proportional loss [41], this cap on the total weights assigned to the rank constraints represents a prior for the noise level expected in the labels. For example, training with $p = 0.8$ tells the system that about 80% of the pairs are a priori expected to be valid. The summation of the binary latent variable w_{ij} prevents the trivial solution of assigning 0 to all the latent variables.

Rather than optimize binary latent selection variables with alternating minimization, we instead use real-valued selection variables, and the function $h(x_i, x_i)$ directly predicts those latent variables w_{ij}. The advantages are threefold. First, we can simultaneously optimize the ranking function f and the selected training pairs. Second, the latent variable w_{ij} is conditioned on the input features so it can learn whether a ranking constraint is valid as a function of the specific visual input. Third, by relaxing w_{ij} to a continuous variable in the range from 0 to 1, we capture uncertainty about pair validity during training.

Finally, we parameterize the latent variables w_{ij}, which provide learned weights for the training samples, and refine our objective to train over batches while enforcing the noise level prior p. We split the training data into groups, each of which contains exactly n pairs. We then require that the latent variable w_{ij} for instances within a group sum up to 1. In particular, let $\mathcal{P}_1, \ldots, \mathcal{P}_m$ be a random split of the set of

[2] We simply break them up uniformly into 2-second segments, though automated temporal segmentation could also be employed [28, 31].

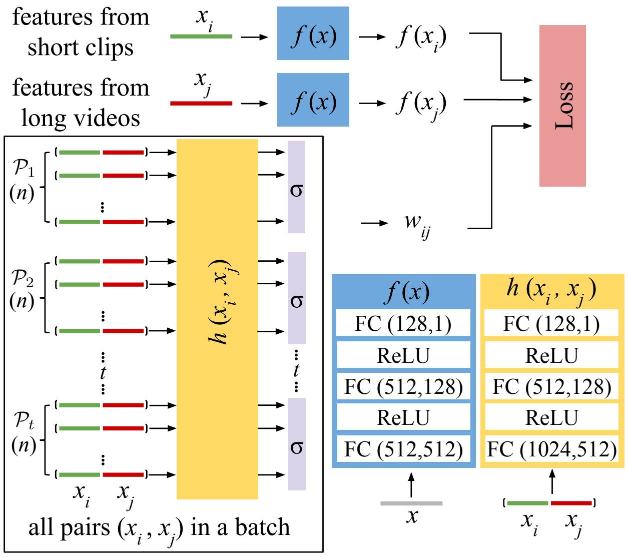

Figure 3: Network architecture details of our approach. The batch size is b. We group every n instances of training pairs and feed them to a softmax function. Each batch has t such groups ($b = nt$).

pairs $\mathcal{P}$ into m groups where each group contains exactly n pairs, then the final loss becomes:

$$\begin{aligned} L(D) = \sum_{g=1}^{m} \sum_{(s_i,s_j)\in\mathcal{P}_g} & \tilde{w}_{ij}\ \max\left(0, 1 - f(x_i) + f(x_j)\right) \\ \text{s.t.} \sum_{(s_i,s_j)\in\mathcal{P}_g} \tilde{w}_{ij} &= \sum_{(s_i,s_j)\in\mathcal{P}_g} \sigma_g(h(x_i,x_j)) = 1, \\ \tilde{w}_{ij} &\in [0,1], \end{aligned} \tag{3}$$

where σ_g denotes the softmax function defined over the set of pairs in group $\mathcal{P}_g$. Note that now the group size n, together with the softmax, serves to uphold the label noise prior p, with $p = \frac{1}{n}$, while allowing a differentiable loss for the selection function h. Intuitively, smaller values of n will speed up training at the cost of mistakenly promoting some invalid pairs, whereas larger values of n will be more selective for valid pairs at the cost of slower training. In experiments, we fix n to 8 for all results and datasets.

As f learns from training data, the function h helps f to attend to training pairs that are consistent. Starting with the prior that there are more valid than invalid pairs, it learns to assign low (high) weights to training pairs that violate (satisfy) ranking constraints, respectively. Please see Supp. for an ablation study with respect to n and results showing how h gradually concentrates more weight on valid pairs.

Network structure: We model both $f(x)$ and $h(x_i, x_j)$ with neural networks. We use a 3 hidden layer fully-connected model for $f(x)$. The function $h(x_i, x_j)$ consists of a 3 fully-connected layers, followed by a n-way softmax function, as shown in Eq.(3). See Fig. 3 for network architecture details.

Video segment feature representation: To generate features x_i for a segment s_i we use a 3D convolution network [10] with a ResNet-34 [11] backbone pretrained on Kinetics [4]. We use the feature after the pooling of the final convolution layer. Each video segment is thus represented by a feature of 512 dimensions.

Implementation details: We implement our model with PyTorch, and optimize with stochastic gradient with momentum for 30 epochs. We use a batch size of 2048 and set the base learning rate to 0.005. We use a weight decay of 0.00005 and a momentum of 0.9. With a single Quadro GP100 gpu, the total feature extraction time for a one-minute-long video is 0.50 s. After extracting video features, the total training time to train a model is one hour for a dataset of 20,000 video clips of total duration 1600 hours. At test time, it takes 0.0003 s to detect highlights in a new one-minute-long video after feature extraction.

4. Results

We validate our approach for highlight detection and compare to an array of previous methods, focusing especially on those that are unsupervised and domain-specific.

4.1. Experimental setup

Datasets and metrics: After training our model on the Instagram video, we evaluate it on two challenging public video highlight detection datasets: YouTube Highlights [33] and TVSum [31]. YouTube Highlights [33] contains six domain-specific categories: *surfing, skating, skiing, gymnastics, parkour*, and *dog*. Each domain consists of around 100 videos and the total accumulated time is 1430 minutes. TVSum [31] is collected from YouTube using 10 queries and consists of 50 videos in total from domains including *changing vehicle tire, grooming an animal, making sandwich, parade, flash mob gathering*, and others. Since the ground truth annotations in TVSum [31] provide frame-level importance scores, we first average the frame-level importance scores to obtain the shot-level scores, and then select the top 50% shots (segments) for each video as the human-created summary, following [27, 26]. Finally, the highlights selected by our method are compared with 20 human-created summaries. We report mean average precision (mAP) for both datasets.

Baselines: We compare with nine state-of-the-art methods as reported in the literature. Here we organize them based on whether they require shot-level annotation (supervised) or not (unsupervised). Recall that our method is unsupervised and domain-specific, since we use no annotations and compose the pool of training video with tag-based queries.

- **Unsupervised baselines:** We compare with the following unsupervised methods: RRAE [39], MBF [5], KVS [28], CVS [27], SG [23], DeSumNet(DSN) [26], and VESD [3]. We also implement a baseline where we train classifiers (CLA) with our hashtagged Instagram videos. The classifiers use the same network structures (except the last layer is replaced with a K-way classification) and video features as our method. We then use the classifier score for highlight detection. CLA can be seen as a deep network variant of KVS [28]. We also implemented k-means and spectral clustering baselines, but found them inferior to the more advanced clustering method [5] reported below.

- **Supervised baselines:** We compare with the latent-SVM approach [33], which trains with human-edited video-highlight pairs, and the Video2GIF approach [9], a domain-agnostic method that trains with human-edited video-GIF pairs. Though these methods require annotations—and ours does not—they are of interest since they also use ranking formulations.

We present results for two variants of our method: **Ours-A**: Our method trained with Instagram data in a domain-*agnostic* way, where we pool training videos from all queried tags. We use a single model for all experiments; **Ours-S**: Our method trained with domain-*specific* Instagram data, where we train a separate highlight detector for each queried tag. For both variants, our method's training data pool is generated entirely automatically and uses no highlight annotations. A training video is in D_S if its duration is between 8 and 15 s, and it is in D_L if its duration is between 45 and 60 s. We discard all other videos. Performance is stable as long as we keep a large gap for the two cut off thresholds. Our networks typically converge after 20 epochs, and test performance is stable ($\pm 0.5\%$) when we train multiple times with random initializations. See Supp.

4.2. Highlight Detection Results

Results on YouTube Highlights dataset: Table 1 presents the results on YouTube Highlights [33]. All the baseline results are as reported in the authors' original papers. Our domain specific method (Ours-S) performs the best—notably, it is even better than the *supervised* ranking-based methods. Compared to the unsupervised RRAE approach [39], our average gain in mAP is 18.1%. Our method benefits from discriminative training to isolate highlights from non-highlight video segments. Our method also outperforms the CLA approach that is trained on the same dataset as ours, indicating that our advantage is not due to the training data alone. CLA can identify the most discriminative video segments, which may not always be highlights. On average our method outperforms the LSVM approach [33], which is trained with domain-specific manually annotated data. While the supervised methods are good at leveraging high quality training data, they are also limited by the practical difficulty of securing such data at scale. In contrast, our method leverages large-scale tagged Web video at scale, without manual highlight examples.

	RRAE	GIFs	LSVM	CLA	Ours-A	Ours-S
	(unsup) [39]	(sup) [9]	(sup) [33]	(unsup)	(unsup)	(unsup)
dog	0.49	0.308	**0.60**	0.502	0.519	0.579
gymnast.	0.35	0.335	0.41	0.217	**0.435**	0.417
parkour	0.50	0.540	0.61	0.309	0.650	**0.670**
skating	0.25	0.554	**0.62**	0.505	0.484	0.578
skiing	0.22	0.328	0.36	0.379	0.410	**0.486**
surfing	0.49	0.541	0.61	0.584	0.531	**0.651**
Average	0.383	0.464	0.536	0.416	0.505	**0.564**

Table 1: Highlight detection results (mAP) on YouTube Highlights [33]. Our method outperforms all the baselines, including the supervised ranking-based methods [33, 9].

Our method trained with domain specific data (Ours-S) performs better than when it is trained in a domain-agnostic way (Ours-A). This is expected since highlights often depend on the domain of interest. Still, our domain-agnostic variant outperforms the domain-agnostic Video2GIF [9], again revealing the benefit of large-scale weakly supervised video for highlight learning.

Fig. 4 and the Supp. video show example highlights. Despite not having explicit supervision, our method is able to detect highlight-worthy moments for a range of video types.

Results on TVSum dataset: Table 2 presents the results on TVSum [31].[3] We focus the comparisons on unsupervised and domain-specific highlight methods. TVSum is a very challenging dataset with diverse videos. Our method outperforms all the baselines by a large margin. In particular, we outperform the next best method SG [23] by 10.1 points, a relative gain of 22%. SG learns to minimize the distance between original videos and their summaries. The results reinforce the advantage of discriminatively selecting segments that are highlight-worthy versus those that are simply representative. For example, while a close up of a bored dog might be more *representative* in the feature space for dog show videos, a running dog is more likely to be a highlight. Our method trained with domain specific data (Ours-S) again outperforms our method trained in a domain-agnostic way (Ours-A).

Instagram vs. YouTube for training: Curious whether an existing large-scale collection of Web video might serve equally well as training data for our approach, we also trained our model on videos from YouTube8M [2]. Training on 6,000 to 26,000 videos per domain from YouTube8M, we found that results were inferior to those obtained with the Instagram data (see Supp. for details). We attribute this to two factors: 1) the YouTube-8M was explicitly curated

[3] Results for CVS [27], DeSumNet [26] and VESD [3] are from original papers. All others (MBF [5], KVS [28] and SG [23]) are as reported in [3].

	MBF [5]	KVS [28]	CVS [27]	SG [23]	DSN [26]	VESD [3]	CLA	Ours-A	Ours-S
Vehicle tire	0.295	0.353	0.328	0.423	-	-	0.294	0.449	**0.559**
Vehicle unstuck	0.357	0.441	0.413	0.472	-	-	0.246	**0.495**	0.429
Grooming animal	0.325	0.402	0.379	0.475	-	-	0.590	0.454	**0.612**
Making sandwich	0.412	0.417	0.398	0.489	-	-	0.433	0.537	**0.540**
Parkour	0.318	0.382	0.354	0.456	-	-	0.505	0.602	**0.604**
Parade	0.334	0.403	0.381	0.473	-	-	0.491	**0.530**	0.475
Flash mob	0.365	0.397	0.365	**0.464**	-	-	0.430	0.384	0.432
Beekeeping	0.313	0.342	0.326	0.417	-	-	0.517	0.638	**0.663**
Bike tricks	0.365	0.419	0.402	0.483	-	-	0.578	0.672	**0.691**
Dog show	0.357	0.394	0.378	0.466	-	-	0.382	0.481	**0.626**
Average	0.345	0.398	0.372	0.462	0.424	0.423	0.447	0.524	**0.563**

Table 2: Highlight detection results (Top-5 mAP score) on TVSum [31]. All methods listed are unsupervised. Our method outperforms all the baselines by a large margin. Entries with "-" mean per-class results not available for that method.

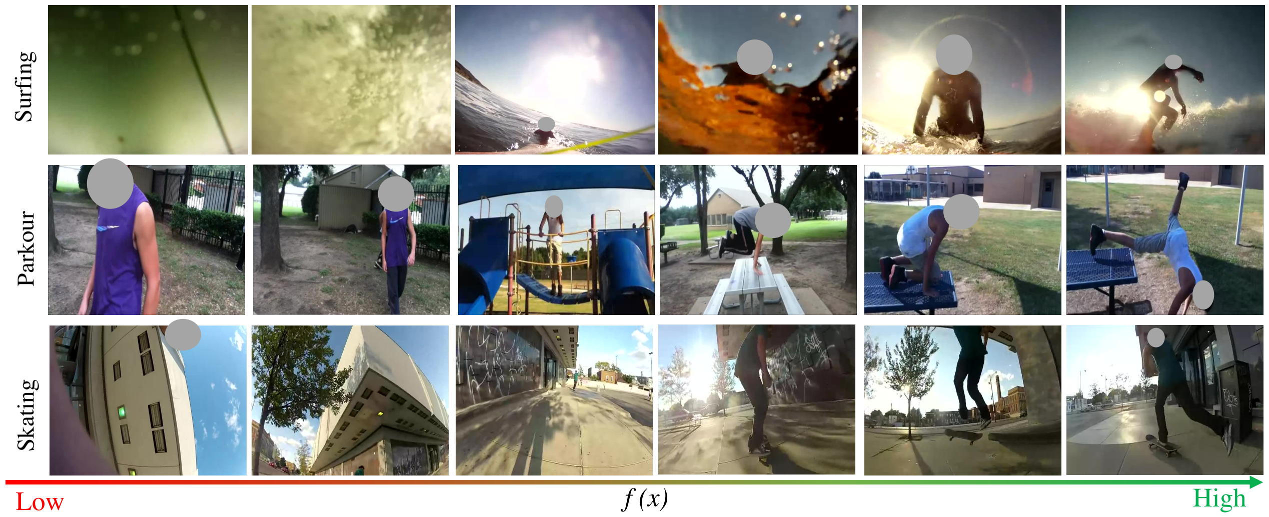

Figure 4: Example highlight detection results for the YouTube Highlights dataset [33]. We show our method's predicted ranking from low (left) to high (right) and present one frame for each video segment. Please see Supp. video for examples.

to have fairly uniform-length "longer" (120-500 s) clips [2], which severely mutes our key duration signal, and 2) users sharing videos on Instagram may do so to share "moments" with family and friends, whereas YouTube seems to attract a wider variety of purposes (e.g., instructional videos, edited films, etc.) which may also weaken the duration signal.

4.3. Ablation Studies

Next we present an ablation study. All the methods are trained with domain-specific data. We compare our full method (Ours-S) with two variants: 1) **Ranking-D**, which treats all the ranking constraints as valid and trains the ranking function without the latent variables. This is similar to existing supervised highlight detection methods [9, 40]. 2) **Ranking-EM**, which introduces a binary latent variable and optimizes the ranking function and binary latent selection variable in an alternating manner with EM, similar to [33]. Note that unlike our approach, here the binary latent variable is discrete and it is not conditioned on the input.

Table 3 shows the results. Our full method outperforms the alternative variants. In particular, our average gain in mAP over *Ranking-D* is 13.9% and 16.3% for Youtube and TVSum, respectively. This supports our hypothesis that ranking constraints obtained by sampling training pairs (s_i, s_j) such that $v(s_i) \in D_s$ and $v(s_j) \in D_L$ are indeed noisy. By modeling the noise and introducing the latent selection variable, our proposed method improves performance significantly. Our method also significantly outperforms *Ranking-EM*, which also models noise in the training samples. In contrast to *Ranking-EM*, our method directly predicts the latent selection variable from input. In addition, we benefit from joint optimization and relaxation of the latent selection variable, which accounts for uncertainty.

Fig. 6 shows highlight detection accuracy as a function of training set size. We report this ablation for YouTube Highlights only, since the videos sharing tags with some TVSum categories max out at 24,000. As we increase the number of videos in each domain, accuracy also improves.

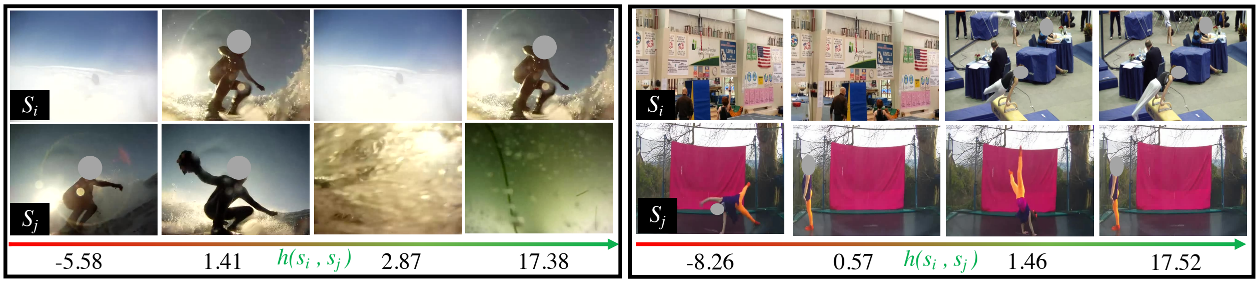

Figure 5: Predicted latent values (before softmax) for video segment pairs from YouTube Highlights. Higher latent value indicates higher likelihood to be a valid pair. The predicted latent value is high if s_i (top row) is a highlight and s_j (bottom row) is a non-highlight. See Supp. for more.

Dataset	Ranking-D	Ranking-EM	Ours-S
YouTube	0.425	0.458	**0.564**
TVSum	0.400	0.444	**0.563**

Table 3: Accuracy (mAP) in ablation study.

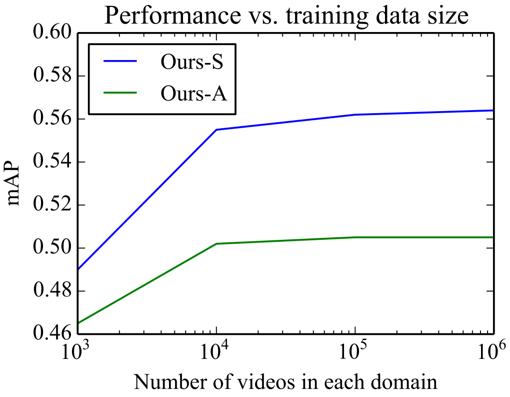

Figure 6: Accuracy vs. training set size on YouTube [33].

The performance improves significantly (6.5% for Ours-S and 3.7% for Ours-A) when the training data is increased from $1,000$ to $10,000$ in each domain, then starts to plateau.

4.4. Understanding Learning from Duration

Finally, we investigate what each component of our model has learned from video duration. First, we test whether our model can distinguish segments from shorter videos versus segments from longer videos. This is essentially a validation of the main training objective, without the additional layer of highlight accuracy. We train our model and reserve 20% novel videos for testing. Each test pair consists of a randomly sampled video segment from a novel shorter video and one from a novel longer video. We use $f(x)$ to score each segment and report the percentage of successfully ranked pairs. Without the proposed latent weight prediction, our model achieves a 58.2% successful ranking rate. Since it is higher than chance (50%), this verifies our hypothesis that the distributions of the two video sources are different. However, the relatively low rate also indicates that the training data is very noisy. After we weight the test video pairs with $h(x_i, x_j)$, we achieve a 87.2% success rate. The accuracy improves significantly because our latent value prediction function $h(x_i, x_j)$ identifies discriminative pairs.

Second, we examine video segment pairs constructed from the YouTube Highlights dataset alongside their predicted latent values (before softmax). See Fig. 5. Higher latent values indicate higher likelihood to be a valid pair. Video segments (s_i) from the top row are supposed to be ranked higher than video segments (s_j) from the second row. When s_i corresponds to a highlight segment and s_j a non-highlight segment, the predicted latent value is high (last columns in each block). Conversely, the predicted latent value is extremely low when s_i corresponds to a non-highlight segment and s_j a highlight segment (first column in each block). Note if we group all the examples in each block into a softmax, all the training examples except the last will have negligible weights in the loss. This demonstrates that the learned $h(x_i, x_j)$ can indeed identify valid training pairs, and is essential to handle noise in training.

5. Conclusions

We introduce a scalable unsupervised solution that exploits *video duration* as an implicit supervision signal for video highlight detection. Through experiments on two challenging public video highlight detection benchmarks, our method substantially improves the state-of-the-art for unsupervised highlight detection. The proposed framework has potential to build more intelligent systems for video preview, video sharing, and recommendations. It could also benefit applications like auto-captions for the visually impaired or more accurate detection of policy-violating content. Future work will explore how to combine multiple pretrained domain-specific highlight detectors for test videos in novel domains. Since the proposed method is robust to label noise and only requires weakly-labeled annotations like hashtags, it has the potential to scale to an unprecedented number of domains, possibly utilizing predefined or learned taxonomies for reusing parts of the model.

References

[1] https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html#_Toc484813989.

[2] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[3] Sijia Cai, Wangmeng Zuo, Larry S Davis, and Lei Zhang. Weakly-supervised video summarization using variational encoder-decoder and web prior. In *ECCV*, 2018.

[4] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017.

[5] Wen-Sheng Chu, Yale Song, and Alejandro Jaimes. Video co-summarization: Video summarization by visual co-occurrence. In *CVPR*, 2015.

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

[7] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. In *ECCV*, 2014.

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

[9] Michael Gygli, Yale Song, and Liangliang Cao. Video2gif: Automatic generation of animated gifs from video. In *CVPR*, 2016.

[10] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *CVPR*, 2018.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[12] Maximilian Ilse, Jakub M Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *ICML*, 2018.

[13] Aditya Khosla, Raffay Hamid, Chih-Jen Lin, and Neel Sundaresan. Large-scale video summarization using web-image priors. In *CVPR*, 2013.

[14] G. Kim, L. Sigal, and E. Xing. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. In *CVPR*, 2014.

[15] Gunhee Kim and Eric P Xing. Reconstructing storyline graphs for image recommendation from web community photos. In *CVPR*, 2014.

[16] Kuan-Ting Lai, Felix X Yu, Ming-Syan Chen, and Shih-Fu Chang. Video event detection by inferring temporal instance labels. In *CVPR*, 2014.

[17] Yong Jae Lee, Joydeep Ghosh, and Kristen Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012.

[18] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *ICCV*, 2017.

[19] T. Liu and D. Tao. Classification with noisy labels by importance reweighting. *PAMI*, 2016.

[20] W. Liu, T. Mei, Y. Zhang, C. Che, and J. Luo. Multi-task deep visual-semantic embedding for video thumbnail selection. In *CVPR*, 2015.

[21] Zheng Lu and Kristen Grauman. Story-driven summarization for egocentric video. In *CVPR*, 2013.

[22] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018.

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

[24] Engin Mendi, Hélio B Clemente, and Coskun Bayrak. Sports video summarization based on motion analysis. *Computers & Electrical Engineering*, 2013.

[25] N. Natarajan, I. Dhillon, P. Ravikumar, and A. Tewari. Learning with noisy labels. In *NIPS*, 2015.

[26] Rameswar Panda, Abir Das, Ziyan Wu, Jan Ernst, and Amit K Roy-Chowdhury. Weakly supervised summarization of web videos. In *ICCV*, 2017.

[27] Rameswar Panda and Amit K Roy-Chowdhury. Collaborative summarization of topic-related videos. In *CVPR*, 2017.

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

[29] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

[30] Yong Rui, Anoop Gupta, and Alex Acero. Automatically extracting highlights for tv baseball programs. In *ACM Multimedia*, 2000.

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

[32] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

[33] Min Sun, Ali Farhadi, and Steve Seitz. Ranking domain-specific highlights by analyzing edited videos. In *ECCV*, 2014.

[34] Hao Tang, Vivek Kwatra, Mehmet Emre Sargin, and Ullas Gargi. Detecting highlights in sports videos: Cricket as a test case. In *ICME*, 2011.

[35] Jinjun Wang, Changsheng Xu, Chng Eng Siong, and Qi Tian. Sports highlight detection from keyword sequences using hmm. In *ICME*, 2004.

[36] Bo Xiong, Gunhee Kim, and Leonid Sigal. Storyline representation of egocentric videos with an applications to story-based search. In *ICCV*, 2015.

[37] Ziyou Xiong, Regunathan Radhakrishnan, Ajay Divakaran, and Thomas S Huang. Highlights extraction from sports

video based on an audio-visual marker detection framework. In *ICME*, 2005.

[38] Jia Xu, Lopamudra Mukherjee, Yin Li, Jamieson Warner, James M Rehg, and Vikas Singh. Gaze-enabled egocentric video summarization via constrained submodular maximization. In *CVPR*, 2015.

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

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

[41] Felix X Yu, Dong Liu, Sanjiv Kumar, Tony Jebara, and Shih-Fu Chang. $\propto$ svm for learning with label proportions. In *ICML*, 2013.

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

[43] Ke Zhang, Kristen Grauman, and Fei Sha. Retrospective encoders for video summarization. In *ECCV*, 2018.

[44] Bin Zhao and Eric P Xing. Quasi real-time summarization for consumer videos. In *CVPR*, 2014.