Supplementary Material
Summary Transfer: Exemplar-based Subset Selection for Video Summarization

Ke Zhang*, Wei-Lun Chao*
University of Southern California
Los Angeles, CA 90089
{zhang.ke,weilunc}@usc.edu

Fei Sha
University of California
Los Angeles, CA 90095
feisha@cs.ucla.edu

Kristen Grauman
University of Texas at Austin
Austin, TX 78701
grauman@cs.utexas.edu

In this Supplementary Material, we give out details omitted in the main text: learning parameters in section 1 (section 3.3 in the main text), datasets, features and evaluation metrics for our experimental studies in section 2, and 3 (section 4.1 in the main text), analysis and results in section 4 (section 4.2 in the main text), and further discussions in section 5.

1. Learning

1.1. Maximum Likelihood Estimation

We use gradient descent to maximize the log-likelihood $\sum_{q=1}^{\mathsf{R}} \log P(\boldsymbol{y}_q; \hat{\boldsymbol{L}}_q)$, defined in (13) of the main text, w.r.t. $\boldsymbol{\alpha}$ and $\boldsymbol{\Omega}$:

$$(\boldsymbol{\alpha}^*, \boldsymbol{\Omega}^*) = \arg\max_{\boldsymbol{\alpha},\boldsymbol{\Omega}} \sum_{q=1}^{\mathsf{R}} \log P(\boldsymbol{y}_q; \hat{\boldsymbol{L}}_q). \tag{1}$$

For brevity, we ignore the subscript q in the following. The corresponding partial derivatives w.r.t. $\boldsymbol{\alpha}$ and $\boldsymbol{\Omega}$ are shown below, according to the chain rule:

$$\frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial \alpha_r} = \sum_{m,n}\sum_{k,l} \frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial (\hat{\boldsymbol{L}})_{mn}} \frac{\partial (\hat{\boldsymbol{L}})_{mn}}{\partial \boldsymbol{L}_{r,kl}} \frac{\partial \boldsymbol{L}_{r,kl}}{\partial \alpha_r} \tag{2}$$

$$\frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial \boldsymbol{\Omega}} = \sum_{r=1}^{\mathsf{R}}\sum_{m,n}\sum_{k,l} \frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial (\hat{\boldsymbol{L}})_{mn}} \frac{\partial (\hat{\boldsymbol{L}})_{mn}}{\partial \boldsymbol{S}_{r,kl}} \frac{\partial \boldsymbol{S}_{r,kl}}{\partial \boldsymbol{\Omega}} \tag{3}$$

* Equal contributions

To ease the computation in what follows, we define $\boldsymbol{M}$ as a matrix with the same size of $\hat{\boldsymbol{L}}$, and $\boldsymbol{M}_{\boldsymbol{y}} = (\hat{\boldsymbol{L}}_{\boldsymbol{y}})^{-1}$. All the other entries of $\boldsymbol{M}$ are zero.

1.2. Gradients with respect to the scale parameters

$$\begin{aligned}&\frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial \alpha_r} \\ &= \sum_{m,n}\sum_{k,l} \frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial (\hat{\boldsymbol{L}})_{mn}} \frac{\partial (\hat{\boldsymbol{L}})_{mn}}{\partial \boldsymbol{L}_{r,kl}} \frac{\partial \boldsymbol{L}_{r,kl}}{\partial \alpha_r} \\ &= \mathbf{1}^\top \{ \Big(\boldsymbol{S}_r{}^\top (\boldsymbol{M} - (\hat{\boldsymbol{L}} + I)^{-1}) \boldsymbol{S}_r \Big) \circ \boldsymbol{I}_r \} \mathbf{1},\end{aligned} \tag{4}$$

where $\circ$ is the element-wise product. $\boldsymbol{I}_r$ is a diagonal matrix (of the same size as $\boldsymbol{L}_r$), with entries indexed by $\boldsymbol{y}_r$ as one; otherwise, zero. $\mathbf{1}$ is the all one vector with a suitable size.

1.3. Gradients with respect to the transformation matrix

$$\begin{aligned}&\frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial \boldsymbol{\Omega}} \\ &= \sum_{r=1}^{\mathsf{R}}\sum_{m,n}\sum_{k,l} \frac{\partial \log P(\boldsymbol{y}; \hat{\boldsymbol{L}})}{\partial (\hat{\boldsymbol{L}})_{mn}} \frac{\partial (\hat{\boldsymbol{L}})_{mn}}{\partial \boldsymbol{S}_{r,kl}} \frac{\partial \boldsymbol{S}_{r,kl}}{\partial \boldsymbol{\Omega}} \\ &= 4\boldsymbol{\Omega}\big(\Phi \boldsymbol{C}_r {\Phi_r}^\top + \Phi_r \boldsymbol{C}_r^\top \Phi^\top \\ &- (\Phi \boldsymbol{C}_r^{(1)} \Phi^\top + \Phi_r \boldsymbol{C}_r^{(2)} {\Phi_r}^\top) \big),\end{aligned} \tag{5}$$

where $\boldsymbol{C}_r, \boldsymbol{C}_r^{(1)}, \boldsymbol{C}_r^{(2)}$ are defined as:

$$\begin{aligned}\boldsymbol{C}_r &= ((\boldsymbol{M} - (\hat{\boldsymbol{L}} + \boldsymbol{I})^{-1}) \boldsymbol{S}_r \boldsymbol{L}_r) \circ \boldsymbol{S}_r, \\ \boldsymbol{C}_r^{(1)} &= \mathsf{diag}(\boldsymbol{C}_r \mathbf{1}) \\ \boldsymbol{C}_r^{(2)} &= \mathsf{diag}(\mathbf{1}^\top \boldsymbol{C}_r).\end{aligned} \tag{6}$$

Φ_r and Φ are the column-wise concatenation (matrices) of $\{\boldsymbol{\phi}_r\}$ and $\{\boldsymbol{\phi}\}$, respectively.

1.4. Extension for sequential modeling

Prior work seqDPP [3] shows that the vanilla DPP is inadequate in capturing the sequential nature of video data. In particular, DPP models a bag of items such that permutation will not affect selected subsets. Clearly, video frames cannot be randomly permuted without losing the semantic meaning of the video. The authors propose the sequential DPP (seqDPP), which sequentially extracts summaries from video sequences.

Extending our approach to this type of sequential modeling is straightforward due to its non-parametric nature. Briefly, we segment each video in the training dataset into smaller chunks and treat each chunk as a separate "training" video containing a subset of the original summary. We follow the same procedure as described above to formulate the base summarization kernels for those shorter videos and their associated summaries. Likewise, we learn the parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\Omega}$ for each segment.

During testing time when we need to summarize a new video $\mathcal{Y}$, we segment the video $\mathcal{Y}$ in temporal order: $\mathcal{Y} = \mathcal{Y}_1 \cup \mathcal{Y}_2 \cdots \mathcal{Y}_T$. We then perform the sequential summarization procedure in the seqDPP method. Specifically, at time t, we formulate the ground set as $\boldsymbol{U}_t = \mathcal{Y}_t \cup \boldsymbol{y}_{t-1}$, i.e., the union of the video segment t and the *selected subset* in the immediate past. For the ground set $\boldsymbol{U}_t$, we calculate its kernel matrix using the (segmented) training videos and their summaries. We then perform the extraction. The recursive extraction process is exactly the same as in [3] and we hence omit it here.

In our experiments, we use sequential modeling for the Kodak, OVP, YouTube and SumMe datasets. For the MED dataset and subshot-based experiments on YouTube dataset, we directly perform our model on each video without sequential modeling as the number of subshots for each dataset is few. We perform KTS [9] to divide a video into segments.

2. Datasets

We validate our approach on five benchmark datasets. In this section we give detailed descriptions for each of them.

2.1. Kodak, OVP and YouTube dataset

We perform keyframe-based video summarization on three video datasets: the Open Video Project (OVP) [1, 2], the YouTube dataset [2], and the Kodak consumer video dataset [8]. They contain 50, 39, and 18 videos, respectively. In the main text, to investigate the effectiveness of category-specific prior, we use Category Sports (16 videos) and News (15 videos), i.e., 31 out of 39 videos. The OVP and YouTube datasets have five human-created frame-level summaries per video, while Kodak has one per video. Thus, for the first two datasets, we follow the procedure described in [3] to create an oracle summary per video. The oracle summaries are then used to optimize model parameters of various methods.

As suggested in [2], we pre-process the video frames as follows. We uniformly sample one frame per second for OVP and YouTube, two frames per second for Kodak (as the videos are shorter) to create the ground set $\mathcal{Y}$ for each video. We then prune away obviously uninformative frames, such as transition frames close to shot boundaries and near-monotone frames. On average, the resulting ground set per video contains 84, 128, and 50 frames for OVP, YouTube, and Kodak, respectively. We replace each frame in the human-annotated summary with the temporally closest one from the ground set, if that is not already contained in the ground set.

2.2. SumMe dataset

The SumMe dataset [4, 5] consists of 25 videos, with the average length being 2m40s. Similar to [9], each video is cut and summarized by 15 to 18 people, and the average length of the ground-truth (shot-based) summary is 13.1% of the original video. We thus perform subshot-based summarization on this dataset.

The video contents in SumMe dataset is heterogeneous and it does not come with pre-defined categories. However, some of them have various degrees of relatedness. For example, videos with title 'Kids playing in leaves' and 'Playing on water slide' are about children playing. Thus, it is interesting to investigate if category prior can help improve summarization on this dataset.

Instead of manually labeling each video, we propose two synthetic categories and split videos into two partitions. We first collapse the 10 video categories in the dataset TVSum[1] [10] into two super-categories, denoted as Super-category I and II, respectively. Super-category I includes activities such as attempting bike tricks, flash mob gathering, park tour and parade, which mostly have raw videos with crowds, and Super-category II includes the rest types of activities with much less people interaction. These two super-categories are semantically similar within each other, though they do not have obvious visual similarity to videos in the SumMe. We then build a binary classifier trained on TVSum videos but classify the videos in SumMe as Super-category I and Super-category II, and then proceed as if they are ground-truth categories, as in MED and YouTube.

Since the shot boundaries given by users vary even for the same video, to create an oracle summary of each video for training, we first score each *frame* by the number of user summaries containing it. We then obtain our shot boundaries based on KTS [9] and compute the shot-level scores by averaging the frame scores within each shot. Finally we

[1] We choose this one as it has raw videos for us to extract features and have a larger number of labeled videos for us to build a category classifier.

rank the shots and select the top ones that are combined to have around 15% of the original video to be the oracle summary for each video.

2.3. MED dataset

The MED dataset, developed in [9], is a subset of the TRECVid 2011 MED dataset. The dataset has 12,249 videos: 2,389 videos in the following 10 categories (namely Birthday party, Changing a vehicle tire, Flash mob gathering, Getting a vehicle unstuck, Grooming an animal, Making a sandwich, Parade, Parkour, Repairing an appliance, and Working on a sewing project) and 9,860 from none of those (labeled as 'null'). 160 videos from the above 2,389 videos are annotated with summaries. For each video in the dataset, an annotator is asked to cut and score all the shots in a video (from 0 to 3). We thus perform subshot-based summarization on this dataset.

There are several constraints in using this dataset: the dataset does not provide the original videos, only boundaries of previously segmented shots and pre-computed features (on average 27 shots per video and their Fisher vectors). Those shot boundaries are often different from the shots determined by human annotators.

To use this dataset for subshot-based summarization, we first need to combine all the human annotators' shot scores into oracle summaries for training. To this end, we first map each users' annotations to the pre-determined shots — in particular, each frame in its pre-determined shot "inherents" the importance scores from the human annotators' scores of the subshots if the subshots contain the frame. We then compute shot-level importance scores by taking an average of the frame importances with in each pre-determined subshot. We then select shots having the top $K\%$ importance scores as ground-truth. We then use this ground-truth for training. In this paper, we test our model under different summary length, i.e. K = 15 and 30.

2.4. Features

For the Kodak, OVP, and YouTube datasets, we extract Fisher vectors, color histograms, and CNN features (based on GoogLeNet [11]). In our experiments, we use Fisher vectors and color histogram for these three dataset, except the comparison in section 4.4. For the MED dataset, as the author didn't provide the raw video, we use the Fisher vectors given in the dataset. For SumMe dataset, we use the output of layer-6 of AlexNet [6] as features.

3. Evaluation protocols

In all our experiments, we evaluate automatic summarization results (A) by comparing them to the human-created summaries (B) and reporting the F-score (F), precision (P), and recall (R), shown as follows:

$$\text{Precision} = \frac{\#\text{matched pairs}}{\#\text{frames (shots) in A}} \times 100\%, \tag{7}$$

$$\text{Recall} = \frac{\#\text{matched pairs}}{\#\text{frames (shots) in B}} \times 100\%, \tag{8}$$

$$\text{F-score} = \frac{P \cdot R}{0.5(P+R)} \times 100\%. \tag{9}$$

For datasets with multiple human-created summaries, we average or take the maximum over the number of human-created summaries to obtain the evaluation metrics for each video. We then average over the number of videos to obtain the metrics for the datasets.

3.1. Kodak, OVP and YouTube datasets

We utilize the VSUMM package [2] to evaluate video summarization results. Given two sets of summaries, one formed by an algorithm and the other by human annotators, this package outputs the maximum number of matched pairs of frames between them. Two frames are qualified to be a matched pair if their visual difference[2] is below a certain threshold, while each frame of one summary can be matched to at most one frame of the other summary. We then can measure F-score based on these matched pairs.

3.2. SumMe

We follow the protocol in [4, 5]. As they constrain the summary length to $\leq 15\%$ of the video duration, we take the shots selected by our model (and if their duration exceeds 15%, we further rank them by their values on the diagonal of $\boldsymbol{L}$-kernel matrix — higher values imply more important items, in our probabilistic framework for subset selection). We then take the highest ranked shots and stop when their total length exceeds 15% of the video duration.

In computing the F-score, [4, 5] use the same formulation in comparing the summarization result to a user summary. However, when dealing with multiple user summaries on one video, instead of taking the average as in [4], [5] reports the highest F-score compared to all the users. In this paper, we follow [5].

3.3. MED

For MED, we conduct subshot-based summarization. We follow the same procedure to align each user's annotation with pre-determined shot boundaries in the dataset (cf. section 2.3 in this Suppl. Material) to create user summaries. We then evaluate similarly as we have done on the Kodak, OVP and YouTube datasets, treating subshots as "frames". Since the KVS algorithm [9] cannot decide the summarization length but only scores each shot, we simply

[2] VSUMM computes the visual difference automatically based on the color histograms.

Table 1. For our method, better frame-based visual similarities improve summarization quality. We report 100-round results for Kodak, OVP and 5-round results for the remaining.

	sim^1	sim^2	sim^3
Kodak	76.6±0.6	80.2±0.3	**82.3**±0.3
OVP	71.4±0.3	75.3±0.3	**76.5**±0.4
YouTube soft	58.9±1.7	60.6 ±1.4	**60.9**±1.3
YouTube hard	59.6±1.4	**61.5**±1.5	**61.5**±1.5

Table 2. Results on YouTube dataset (39 videos), all based on 100 rounds.

	VSUMM$_1$ [2]	seqDPP [3]	**Ours**
YouTube	56.9±0.5	60.3±0.5	60.2±0.7

take the top 15% or 30% shots (ordered by scores) to be the summary of KVS.

4. Detailed experimental results and analysis

4.1. Benefits of learning visual similarity

We further examine the effect of employing different measures of visual similarity. As shown in Table 1, non-linear similarity with the Gaussian-RBF kernel (sim^2) generally outperforms linear similarity (sim^1), while learning a non-identity metric $\boldsymbol{\Omega}$ (sim^3) improves only marginally the performance. In the following experiments, we use sim^2.

4.2. Results on the complete YouTube dataset

The original YouTube dataset [2], after excluding the cartoon videos as in [3], contains 39 videos with 8 of them in neither Sports nor News category. In the main text, we have used 31 videos mainly for the purpose to compare to summarization results exploiting category prior.

For maximum comparability to prior work on this dataset, we provide our summarization results on the 39 videos and compare to previous published results, as in Table 2.

4.3. Detailed results with category prior

The results for the YouTube, SumMe and MED datasets are shown in Table 3, 4 and 5, respectively.

We can see that both the soft category-specific and the hard category-specific outperform the no category-specific setting in most cases. For example, a 'birthday party' video is likely to share structures with an 'outdoor activity' video, thus helping to summarize each other.

Additionally, even when the ground-truth categories for the testing videos are unknown and need to be determined with a category classifier, the category prior can still help.

Table 4. Category-specific experimental results on SumMe dataset. SP_I and SP_II stands for super-category I and II, respectively. We report 5-round results.

Setting	Testing	Category determined by a classifier	
SumMe	SP_I	38.6±0.5	39.2±0.7
	SP_II	40.7±0.7	
SumMe soft	SP_I	39.2±0.6	40.2±0.7
	SP_II	41.9±0.7	
SumMe hard	SP_I	39.8±0.8	40.9±0.9
	SP_II	43.3±0.9	

Table 6. The comparison of shallow and deep features for summarization. We report 5-round results.

	Shallow	Deep
Kodak	**80.2**±0.3	79.1±0.4
YouTube	**60.0**±1.3	59.3±1.5
YouTube soft	**60.6**±1.4	59.6±1.6
YouTube hard	**61.5**±1.5	61.0±1.6

4.4. Comparison between deep and shallow features

Here we present results on contrasting deep to shallow features to show that deep features for visual recognition do not help much over shallow features. As shown in Table 6. We compare Fisher vector and color histogram with state-of-the-art CNN features by GoogLeNet [11].

4.5. Comparison to seqDPP

Our method outperforms seqDPP [3] on Kodak and YouTube (cf. Table 2 in the main text), whose contents are more diverse and more challenging than OVP. We suspect that since videos in OVP are edited and thus less redundant, methods with higher precision such as seqDPP might be able to perform better than methods with higher recall (such as the proposed approach).

At testing time, our model requires more computation. On YouTube, it takes 1 second on average per video, slower than 0.5 second by seqDPP. However, our model is far advantageous in training. First, it learns much fewer parameters (cf. in the main text, about 9,000 for $\boldsymbol{\alpha}$ in eq. (5) and diagonal $\boldsymbol{\Omega}$ in eq. (4)), while seqDPP requires tuning 80,000 parameters. Our model thus learns well even on small training datasets, shown on Kodak (cf. in section 4.3 of the main text). Secondly, learning seqDPP is computationally very intensive — according to its authors, a great deal of hyperparameter tuning and model selection was performed. Learning our model is noticeably faster. For example, on Youtube, our model takes about 1 minute per configuration of hyperparameters, while seqDPP takes 9 minutes.

4.6. Qualitative comparisons

We provide exemplar video summaries in Fig. 1 (and attached movies), along with the human-created summaries.

Table 3. Exploiting category prior with YouTube dataset, which contains totally 31 videos in Sport and News categories. We report F-score in each category as well as on the whole dataset, averaging over 5 rounds of experiments.

Setting	Testing video's category	Category prior not used	
YouTube	Sports	53.5±1.5	
	News	66.9±1.2	
	Combined	60.0±1.3	
		Ground-truth category is used	Category determined by a classifier
YouTube soft	Sports	53.4±1.5	53.4±1.5
	News	68.2±1.4	67.5±1.6
	Combined	60.6±1.4	60.2±1.6
YouTube hard	Sports	54.4±1.6	54.4±1.6
	News	69.1±1.4	68.3±1.8
	combined	61.5±1.5	61.1±1.8

Table 5. Category-specific experimental results on MED summaries dataset with 10 categories. We consider two settings, $K = 15$ and $K = 30$, as mentioned in section 2.3 and 3.3. We report 5-round results.

(a) User summaries at $K = 15$

Setting	Category prior not used	
MED	28.9±0.8	
	Ground-truth category is used	Category determined by a classifier
MED soft	30.7±1.0	29.4±1.2
MED hard	30.4±1.0	28.5±1.3

(b) User summaries at $K = 30$

Setting	Category prior not used	
MED	47.2±0.7	
	Ground-truth category is used	Category determined by a classifier
MED soft	48.6±0.8	45.9±1.0
MED hard	48.1±1.1	44.5±1.1

Since OVP and YouTube have five human-created summaries per video, we display the merged oracle summary (see section 2.1) for brevity.

In general, our approach provides summaries most similar to those created by humans. We attribute this to two factors. Firstly, employing supervised learning helps identify representative contents. VSUMM$_1$ [2], though achieving diversity via unsupervised clustering, fails to capture such a notion of representativeness, resulting in a poor F-score and a drastically different number of summarized frames compared to the oracle/human annotations. Secondly, through non-parametrically transferring the underlying criteria of summarization, the proposed approach arrives at better summarization kernel matrices $\boldsymbol{L}$ than seqDPP, further eliminating several uninformative frames from the summaries (thus improving precision). The bottom failure case, however, suggests how to improve the method further, in the direction of increasing recall rate. We believe those missing frames (thus, shorter summaries) can be recalled if we could consider jointly the kernel computed explicitly on the test videos (such as seqDPP) and the kernel computed by our method.

5. Detailed discussions

5.1. Computational complexity and practicality

Section 3.5 of the main text discusses the computational cost and ways for reducing it. Specifically, the cost depends on (1) the length of the training and testing videos; (2) the number of training videos.

For (1), we can down-sample the videos to reduce the number of frames (cf. section 3.5 of the main text and section 2.1 for details), and exploit subshot-based transfer (section. 3.4 of the main text). For the latter, the size of $\boldsymbol{L}$ (eq. (11) of the main text) depends only on the number of subshots (not frames), and the subshot-to-subshot similarity further reduces computational cost, without worsening the performance (cf. Table 5 of the main text). Moreover, the sequential modeling trick in [3] can also be used to reduce the cost (see section 1.4).

For (2), the proposed hard category-specific transfer (cf. section 3.4 of the main text) enables transferring from fewer training videos with improved performance.

Figure 1. Exemplar video summaries generated by VSUMM$_1$ [2], seqDPP [3], and ours (sim3), along with the (merged) human-created summaries. We index videos following [2]. On two videos, our approach reaches the highest agreement with the oracle/human-created summaries compared to the other methods. The failure case on the bottom hints the limitation, however. See texts for details.

5.2. Applicability to long egocentric videos

Egocentric video is challenging due to its length and diverse content (e.g., multiple events). One possible strategy is to apply existing techniques to detect and segment the videos into shorter events and then perform our approach on shorter segments, or use methods described in [7] to zoom into frames surrounding the occurrence of important people and objects.

5.3. Mechanisms to check for failure

It is interesting to investigate when our approach will fail in practice. As our approach is non-parametric by transferring the summarization structures from annotated videos, the visual similarities between the testing and annotated videos might be a useful cue. We conduct analysis (cf. Fig. 2) and show that there is indeed positive correlation between visual similarity and summarization quality. A preliminary fail-safe mechanism thus would be thresholding on the similarity.

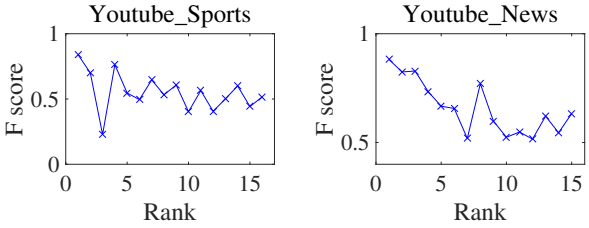

Figure 2. Visual similarity (x-axis, smaller ranks imply stronger similarity) between the testing and annotated videos is positively correlated to summarization quality (F-score on y-axis). Results are from summarizing YouTube via hard transfer.

References

[1] Open video project. http://www.open-video.org/. 2

[2] S. E. F. de Avila, A. P. B. Lopes, A. da Luz, and A. de Albuquerque Araújo. Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011. 2, 3, 4, 5, 6

[3] B. Gong, W.-L. Chao, K. Grauman, and F. Sha. Diverse sequential subset selection for supervised video summarization. In *NIPS*, 2014. 2, 4, 5, 6

[4] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool. Creating summaries from user videos. In *ECCV*, 2014. 2, 3

[5] M. Gygli, H. Grabner, and L. Van Gool. Video summarization by learning submodular mixtures of objectives. In *CVPR*, 2015. 2, 3

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In

NIPS, 2012. 3

[7] Y. J. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012. 6

[8] J. Luo, C. Papin, and K. Costello. Towards extracting semantically meaningful key frames from personal video clips: from humans to computers. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(2):289–301, 2009. 2

[9] D. Potapov, M. Douze, Z. Harchaoui, and C. Schmid. Category-specific video summarization. In *ECCV*, 2014. 2, 3

[10] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes. Tvsum: Summarizing web videos using titles. In *CVPR*, 2015. 2

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 3, 4