Supplementary Materials

to Paper #1541

TL;DW? Summarizing Instructional Videos with Task Relevance & Cross-Modal Saliency

This section is organised as follows:

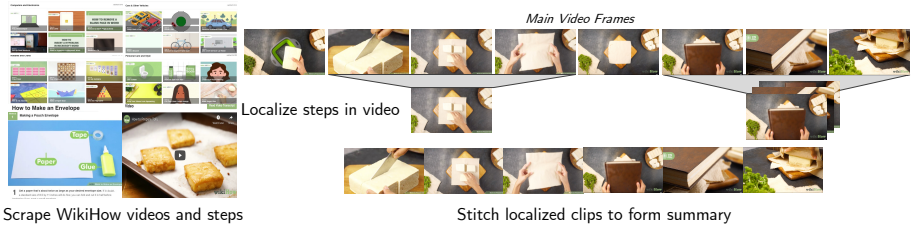

Fig. 1: ***WikiHow Summaries* Data Collection.** We first scrape all the main videos in the WikiHow articles, along with the images or video clips assosciated with each step. Next, the image/clip corresponding to each step is localized in the main video. The images are localized to $\pm$ 2.5 seconds(i.e. a 5 seconds window centered around the image). The localized clips are stitched together to form the summary.

1 *WikiHow Summaries* Data Collection

We provide more details on the WikiHow Summaries data collection process. As described in Sec. 4.2 of the main paper, these are the main stages of the dataset creation: (1) Scraping WikiHow videos (2) Localizing images/clips in video (3) Ground-truth summary from localized clips (4) Manual verification. Fig. 1 illustrates our data collection process. We show an example for the article *"Prepare Tofu"*. We localize each of the individual steps (images/clip) in the main video by comparing the ResNet features and obtain short localized clips. The

clips are stitched together to form the summary. A handful of summaries with spurious lengths (too long or too short) are manually verified and corrected.

We describe how we handle some edge cases in the articles, and the reasoning behind using ResNet features in stage (2).

Multiple methods. Sometimes the articles contain multiple methods of performing a task. If the video also contains multiple methods, as in this *"Draw a cow"* example, we localize each method in the video, and the summary is a compilation of all methods. The reasoning behind doing this is that users looking for a specific way of drawing a cow can take a quick glimpse of the summary and decide if they want to watch the whole video. However, if the article contains multiple methods but the video only contains one, as in *this* example, only the method depicted in the video is added in the summary.

Reason for using ResNet features instead of direct pixel comparison. As described in the main paper Sec 4.2, we compare ResNet features to localize the images/clips of the steps in the main video. The reason we compare ResNet features and not pixel values directly is because the images/clips associated with the steps aren't always extracted from the main video. For example, in this article on *"Making a pinwheel"*, the frames in the images/clips are from a different video and don't have exact matches in the *main video* for the article. Using ResNet features in place of pixels makes the localization robust to color/background changes, allowing us to localize steps despite an exact match of frames.

Table 1: Hyperparameters for training IV-Sum.

Hyperparamter	**Value**
Batch size	24
Epochs	300
Learning rate IV-Sum	1e-3
Learning rate S3D fine-tuning	1e-4
Weight decay	1e-4
Dropout	0.1
Learning rate decay	StepLR
t%	55%
#frames per segment	32
#frames per video during training	768
# Training FPS	8

2 Implementation Details

Video processing. For generating pseudo summaries and for training IV-Sum, the videos are down-sampled to 8 FPS, and divided into non-overlapping segments

of size 32 frames, which is the recommended segment size for MIL-NCE [3][1]. While training IV-Sum, we fix the number of segments sampled from a video to be 28 (i.e. 896 frames) which are selected as a contiguous sequence from a randomly chosen start location. If the video is shorter in duration, it is padded with zeros. During inference, we retain the original fps of the video and all the segments are passed to IV-Sum. For concatenating the text representations to the visual representations, we follow the approach in MIL-NCE and map each visual clip to the sentences a few seconds before, after, and during the clip. The text embedding is an average of all the sentence embeddings.

Hyperparameters. Tab. 1 shows detailed list of hyperparameters. For all baselines and our method, to ensure a fair comparison we generate the summary from scores by selecting the top t% of the highest scoring segments to be in the summary. t is set to be 55 based on the statistics in the validation set of WikiHow Summaries, where on average 55% of the original video appears in the summary.

Dimensions. We first describe the dimensions of each of the embeddings. The image embeddings are in $f_{\text{vid}}(s_i) \in \mathbb{R}^{512}$. The text embeddings for M transcript sentences using f_{text} are in $\mathbb{R}^{M\times 512}$ which are then fused using a 2 layer perceptron to $\mathbb{R}^{512}$. M is set to be the maximum number of sentences found in any ASR transcript. The image and text embeddings are concatenated and passed to the segment scoring transformer f_{trans}, the output dimension of this is in $\mathbb{R}^{512}$.

Computation resources. The training time is approximately 2 days using Distributed Data Parallel to train for 300 epochs on 8 NVIDIA RTX 2080 GPUs. The model inference time for a single video at its original fps is 1.5 minutes on average.

3 Additional Results

Table 2: **Evaluating on generic video summarization datasets.** We compare F-Score of IV-Sum and CLIP-It on the instructional videos in TVSum.

Method	F-Score
CLIP-It [4]	0.72
IV-Sum	0.73

Evaluating on instructional videos in generic video summarization datasets. Here, we consider the existing generic video summarization datasets, in particular, those videos that fall under "instructional" domain, in order to validate our model further. Generic video summarization dataset TVSum [9] has 15 videos pertaining to the categories *changing a car tire*, *getting a car unstuck*,

[1] We use the implementation of MIL-NCE available here https://github.com/antoine77340/MIL-NCE_HowTo100M

and *making a sandwich* while SumMe [2] has no instructional videos. We follow the evaluation protocol described in CLIP-It [4]. For a fair comparison to CLIP-It [4], we curate a test set by randomly selecting 7 of these 15 videos, while the remaining 8 are added to the training set, so as to ensure that the CLIP-It model sees instructional videos during training. The augmented training set is curated by combining the 8 videos with those in SumMe (25 videos), TVSum (45 videos), OVP [5] (50 videos), and YouTube [1] (39 videos). CLIP-It is trained on this augmented training set consisting of 168 videos (including 8 instructional videos) and evaluated on the held out 7 instructional videos. Our method is trained on pseudo summaries (built on top of CrossTask and COIN) and evaluated in a *zero-shot* way on the test set of 7 videos.

As seen in Tab. 2, our method IV-Sum, although trained with noisy / weakly labeled pseudo summaries from a different data distribution, achieves an F-Score comparable to the CLIP-It [4], trained on human annotated summaries.

Table 3: **Comparing step-recall.** We report step-recall on our method and 2 baselines.

Method	**Step-recall**
Step Cross-Modal Similarity	0.68
CLIP-It with ASR	0.70
IV-Sum	0.94

Step recall. We define an additional metric, *step-recall* to be the average percentage of steps present in the ground-truth summary which were successfully picked by the generated summary. Our *WikHow Summaries* dataset contains annotations of frames pertaining to each step, and if any of the frames from a step are present in the summary, we assume the step is covered. Using this logic, we generate a list of steps in the generated summary Y'_{step}, and a list of steps in the ground-truth Y_{step}. We compute step-recall as follows,

$$\text{Step-recall} = \frac{\text{overlap between } Y_{\text{step}} \text{ and } Y'_{\text{step}}}{\text{total duration of } Y_{\text{step}}}$$

In Tab. 3 we report the step-recall for Step Cross-Modal Similarity, CLIP-It with ASR (trained on generic video summarization datasets) and IV-Sum. Both Step Cross-Modal Similarity and CLIP-It with ASR baselines miss 30% of steps found in the ground truth summary while our method on average only misses 6% of the steps.

Loss Ablations. In Table 5, we explore additional loss functions as in prior video summarization works [8,7,6,4]. Diversity loss ensures diversity among the summary segments and the reconstruction loss enforces similarity in representations of the reconstructed summary and the input video. Adding diversity reduced the recall and we notice no improvement on adding reconstruction loss. We believe this may be because frames corresponding to different steps are not always diverse but are still important for the summary.

Table 4: **Instructional Video Summarizer Ablations.** We perform ablations on different components of the video summarizer network and report results on the *WikiHowTo Summaries* validation set.

(a) **IVSum S3D backbone Ablations.** We compare fixing the pre-trained weights of the S3D model to fine-tuning a part of it.

Method	**F-Score**
S3D fixed	65.8
S3D fine-tuned	67.9

(b) **IVSum Segment Scoring Transformer Ablations.** We compare different architecture configurations of the segment scoring transformer.

Method			**F-Score**
	#heads	#layers	
SST	8	16	63.1
SST	16	6	63.5
SST	8	12	66.7
SST	8	24	67.9
MLP	-	-	32.1

Table 5: **Loss Ablations.** We ablate different losses in the IV-Sum model and show results on validation set of *WikiHow Summaries*

Method	**F-Score**	**Recall**
MSE	67.9	84.5
MSE + Diversity	61.2	63.4
MSE + Reconstruction	67.6.	85.8

Model ablations. Table 4a shows the performance comparisons between freezing the video and text encoding backbone (S3D) vs. fine-tuning part of the network. In Table 4b, we ablate the segment scoring transformer (SST) in our model and change the number of encoder layers, heads, and also replace the transformer with an MLP. We report the F-Score on the validation set of *WikiHowTo Summaries*.

For a fair comparison of MIL-NCE vs CLIP features, we retrain our IV-Sum model replacing video segments with frames and replacing MIL-NCE features with CLIP image and text features (same as the ones used in the CLIP-It baseline). We report results in Tab. 6. We see that IV-Sum with CLIP performs at par with CLIP-It with ASR but falls short of IV-Sum, indicating the need to use video segments and MIL-NCE features pre-trained on HowTo100M.

Table 6: **Instructional Video Summarization results on *WikiHow Summaries.*** All models were trained on pseudo summaries.

Method	F-Score		τ (Kendall)	ρ (Spearman)
	Val	Test	Test	Test
CLIP-It with ASR	62.5	61.8	0.093	0.191
IV-Sum with CLIP	61.8	62.0	0.094	0.201
IV-Sum	**67.9**	**67.3**	**0.101**	**0.212**

4 Additional Quantitative Results

Please watch the video on our website for qualitative results.
Comparison to baselines. We show video results comparing the ground-truth summary to that from IV-Sum (our method) and baselines Step Cross-Modal Similarity and CLIP-It with ASR trained on generic video summarization datasets. Our method picks all frames in the ground-truth and assigns high scores to salient frames. Step Cross-Modal Similarity misses the crucial step *"fold and tuck"* at the end as it assigns higher scores to irrelevant frames at the start of the video. This is because it has no knowledge of task-relevance. CLIP-It with ASR (trained on generic video summarization datasets) misses steps (like *"fold into a triangle"*) and assigns lower scores to the key frames in a step as it optimizes for diversity. **Evaluating Pseudo Summary generation procedure for WikiHow Summaries.** We found that there are 15 tasks which are shared between the Pseudo Summary training set and the WikiHow Summaries test set. We applied the method used to construct pseudo summaries to these 15 task videos in the WikiHow Summaries by fetching videos of the same task from our training set. We compare this to IV-Sum and report results in Tab 7. We notice a slight improvement on all three metrics, indicating that our model is able to learn above the noise in the pseudo summaries.

Table 7: **Evaluating pseudo summary generation on subset of WikiHow Summaries**

Method	F-Score	τ	ρ
	Test	Test	Test
Pseudo Summary Generation	38.0	0.03	0.36
IV-Sum	**42.0**	**0.04**	**0.38**

Pseudo summary vs IV-Sum summary. IV-Sum is trained on weakly labeled pseudo summaries that may sometimes be noisy. However, since the training loss is not 0, we check if our model learns despite the noise and produces summaries

of a better quality. In this example, we show summaries for *"this"* video from the Pseudo Summaries dataset. As seen the pseudo summary contains an irrelevant segment where results from a web search are shown in Korean for nearly 10 seconds (9^{th} second to the 19^{th} second). The IV-Sum model trained on pseudo summaries yields a resulting summary without this segment, as it is able to learn *"task-relevance"* and *"cross-modal saliency"*.

Pseudo summary vs step-localization annotation. We compare pseudo summaries generated using our method to the step-localization summary. In the example *"Make a pumpkin spice latte"*, the input video can be found *here*. Step localization only localizes two main steps, *"boil milk"* and *"add coffee and blend"* whereas our pseudo summary contains all the main steps necessary to do the task.

Failure case. Since we always select the top 55% of the segments to be in the summary (i.e. t=55%), the summary chosen by our method is sometimes much longer/shorter than the ground-truth summary. This is a failure case of the baseline methods as well. For example, for this 38 second video on *"How to ripen a cantaloupe"*, the ground-truth summary is a brief 15 seconds whereas our summary covers the steps in more detail and is 21 seconds long.

References

1. De Avila, S.E.F., Lopes, A.P.B., da Luz Jr, A., de Albuquerque Araújo, A.: Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method. Patt. Rec. Letters (2011) 4
2. Gygli, M., Grabner, H., Riemenschneider, H., Gool, L.V.: Creating summaries from user videos. European Conference on Computer Vision (ECCV) (2014) 4
3. Miech, A., Alayrac, J.B., Smaira, L., Laptev, I., Sivic, J., Zisserman, A.: End-to-end learning of visual representations from uncurated instructional videos. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020) 3
4. Narasimhan, M., Rohrbach, A., Darrell, T.: Clip-it! language-guided video summarization. Advances in Neural Information Processing Systems (NeurIPS) (2021) 3, 4
5. Open video project. https://open-video.org/ 4
6. Park, J., Lee, J., Kim, I.J., Sohn, K.: Sumgraph: Video summarization via recursive graph modeling. European Conference on Computer Vision (ECCV) (2020) 4
7. Rochan, M., Wang, Y.: Video summarization by learning from unpaired data. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 4
8. Rochan, M., Ye, L., Wang, Y.: Video summarization using fully convolutional sequence networks. European Conference on Computer Vision (ECCV) (2018) 4
9. Song, Y., Vallmitjana, J., Stent, A., Jaimes, A.: Tvsum: Summarizing web videos using titles. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015) 3