Contrastive Losses Are Natural Criteria for Unsupervised Video Summarization: Supplementary Material

Zongshang Pang
Osaka University
pangzs@is.ids.osaka-u.ac.jp

Yuta Nakashima
Osaka University
n-yuta@ids.osaka-u.ac.jp

Mayu Otani
CyberAgent, Inc.
otani-mayu@cyberagent.co.jp

Hajime Nagahara
Osaka University
nagahara@ids.osaka-u.ac.jp

1. Training Details

We have used two kinds of pre-trained features in our experiments, namely the GoogleNet [15] features for the video summarization datasets and the quantized Inception-v3 [16] features for the Youtube8M dataset, both with 1024 dimensions. The GoogleNet features are provided by [18] and the quantized Inception-v3 features by [1].

The model appended to the feature backbone for contrastive refinement is a stack of Transformer encoders with multi-head attention modules [17]. There are two scenarios for our training: 1. The training with TVSum [14], SumMe [6], YouTube, and OVP [4], which is again divided into the canonical, augmented, and transfer settings; 2. The training with a subset of videos from Youtube8M dataset [1]. We call the training in the first scenario as *Standard* and the second as *YT8M*.

The pre-trained features are first projected into 128 dimensions for training in both scenarios using a learnable, fully connected layer. The projected features are then fed into the Transformer encoders. The model architecture and associated optimization details are in Table A1.

The training for the $10,000$ Youtube8M videos takes around 6 minutes for 40 epochs on a single NVIDIA RTX A6000. The efficient training benefits from the straightforward training objective (contrastive learning), lightweight model, the low dimensionality of the projected features, and the equal length of all the training videos (200 frames).

Table A1: Model and optimization details used for the results in the main paper.

	# Layers	# Heads	d_{model}	d_{head}	d_{inner}	Optimizer	LR	Weight Decay	Batch Size	Epoch	Dropout
Standard	4	1	128	64	512	Adam	0.0001	0.0001	32 (TVSum) 8 (SumMe)	40	0
YT8M	4	8	128	64	512	Adam	0.0001	0.0005	128	40	0

The ablation results for the numbers of encoder layers and attention heads are conducted on the Youtube8M videos alone, as they are much more challenging than the videos used in the standard training, and the results are in Section 4.

2. The Full Evaluation Results for the Standard Training.

In Table 1 in the main text, we only provided Kendall's τ and Spearman's ρ for the canonical training setting, and we list the full results in Tables A2 and A3. For the analysis and the comparison with previous work, please refer to Table 1 in the main paper.

3. Hyperparameter Ablation Results.

We focus on ablating two hyperparameters: a for controlling the size of the nearest neighbor set $\mathcal{N}_t$ and λ_1 for balancing the alignment and uniformity losses. The ablation results are provided for when importance scores are produced by $\bar{\mathcal{L}}_{\text{align}}\&\bar{H}_{\hat{\theta}}$ and by $\bar{\mathcal{L}}_{\text{align}}\&\bar{H}_{\hat{\theta}}\&\bar{\mathcal{L}}_{\text{uniform}}$.

Table A2: The full results for TVSum in the standard training.

		Canonical			Augmented			Transfer		
		F1	τ	ρ	F1	τ	ρ	F1	τ	ρ
w.t. training	$\bar{\mathcal{L}}^*_{\text{align}}$	56.4	0.1055	0.1389	56.4	0.1055	0.1389	54.6	0.0956	0.1251
	$\bar{\mathcal{L}}^*_{\text{align}}$&$\bar{\mathcal{L}}^*_{\text{uniform}}$	58.4	0.1345	0.1776	58.4	0.1345	0.1776	56.8	0.1207	0.1589
w. training	$\bar{\mathcal{L}}_{\text{align}}$	54.6	0.1002	0.1321	55.1	0.1029	0.136	53.0	0.0831	0.1090
	$\bar{\mathcal{L}}_{\text{align}}$&$\bar{\mathcal{L}}_{\text{uniform}}$	58.8	0.1231	0.1625	**59.9**	0.1238	0.1631	57.4	0.1166	0.1529
	$\bar{\mathcal{L}}_{\text{align}}$&$\bar{H}_{\hat{\theta}}$	53.8	0.1388	0.1827	56	0.1363	0.1792	54.3	0.1173	0.1539
	$\bar{\mathcal{L}}_{\text{align}}$&$\bar{\mathcal{L}}_{\text{uniform}}$&$\bar{H}_{\hat{\theta}}$	**59.5**	**0.1609**	**0.2118**	59.9	**0.1623**	**0.2133**	**59.7**	**0.1405**	**0.1846**

Table A3: The full results for SumMe in the standard training.

		Canonical			Augmented			Transfer		
		F1	τ	ρ	F1	τ	ρ	F1	τ	ρ
w.t. training	$\bar{\mathcal{L}}^*_{\text{align}}$	43.5	**0.0960**	**0.1173**	43.5	**0.0960**	**0.1173**	39.4	**0.0769**	**0.0939**
	$\bar{\mathcal{L}}^*_{\text{align}}$&$\bar{\mathcal{L}}^*_{\text{uniform}}$	47.2	0.0819	0.1001	46.07	0.0819	0.1001	41.7	0.0597	0.073
w. training	$\bar{\mathcal{L}}_{\text{align}}$	**46.8**	0.0942	0.1151	47.1	0.0872	0.1065	41.5	0.0756	0.0924
	$\bar{\mathcal{L}}_{\text{align}}$&$\bar{\mathcal{L}}_{\text{uniform}}$	46.7	0.0689	0.0842	**48.4**	0.0645	0.0788	41.1	0.0305	0.0374
	$\bar{\mathcal{L}}_{\text{align}}$&$\bar{H}_{\hat{\theta}}$	45.2	0.0585	0.0715	45	0.059	0.0721	**45.3**	0.0611	0.0747
	$\bar{\mathcal{L}}_{\text{align}}$&$\bar{\mathcal{L}}_{\text{uniform}}$&$\bar{H}_{\hat{\theta}}$	**46.8**	0.0358	0.0437	45.5	0.0353	0.0431	43.9	0.0306	0.0374

3.1. $\bar{\mathcal{L}}_{\text{align}}$ & $\bar{H}_{\hat{\theta}}$

As shown in Table A4 and Fig. 1, when a becomes larger, TVSum performance begins to be unstable in terms of both F1 and correlation coefficients, and SumMe performance is relatively more stable, but also shows a similar unstable pattern in terms of F1. We hypothesize that when a becomes larger, the nearest neighbor set becomes increasingly noisier, making both the alignment loss during training and the local dissimilarity metric (post-training alignment loss) for importance score generation less meaningful due to the semantically irrelevant neighbors. For λ_1, smaller values generally give better performance when a has a reasonable value, as larger values of λ_1 tend to make the uniformity loss suppress the alignment loss. Similarly, too small λ_1 will make the alignment loss suppress the uniformity loss, as we observed unstable training when further decreasing λ_1.

Table A4: The ablation results for a and λ_1 with $\bar{\mathcal{L}}_{\text{align}}$ & $\bar{H}_{\hat{\theta}}$ used for importance score calculation. See the text for analysis.

Hyper-params		TVSum			SumMe		
a	λ_1	F1	τ	ρ	F1	τ	ρ
0.025	0.25	54.7	0.1286	0.1683	47.0	0.0671	0.0828
	0.50	54.9	0.0936	0.1227	43.7	0.0446	0.0550
	1.00	54.8	0.0717	0.0946	43.1	0.0508	0.0628
	2.00	55.0	0.0743	0.0981	42.3	0.0459	0.0567
0.05	0.25	53.8	0.1240	0.1624	43.5	0.0830	0.1027
	0.50	54.7	0.1189	0.1556	44.2	0.0540	0.0666
	1.00	54.4	0.0841	0.1110	42.0	0.0577	0.0714
	2.00	54.3	0.0764	0.1010	40.4	0.0436	0.0543
0.1	0.25	46.9	0.0498	0.0645	44.4	0.0783	0.0971
	0.50	52.9	0.1263	0.1655	48.7	0.0780	0.0964
	1.00	53.2	0.0829	0.1094	44.1	0.0609	0.0755
	2.00	53.8	0.0689	0.0907	38.6	0.0470	0.0583
0.2	0.25	41.1	-0.0318	-0.0421	36.6	0.0639	0.0793
	0.50	52.4	0.0994	0.1299	46.2	0.0816	0.1010
	1.00	52.0	0.0536	0.0707	41.3	0.0630	0.0781
	2.00	54.3	0.0631	0.0830	44.5	0.0622	0.0771

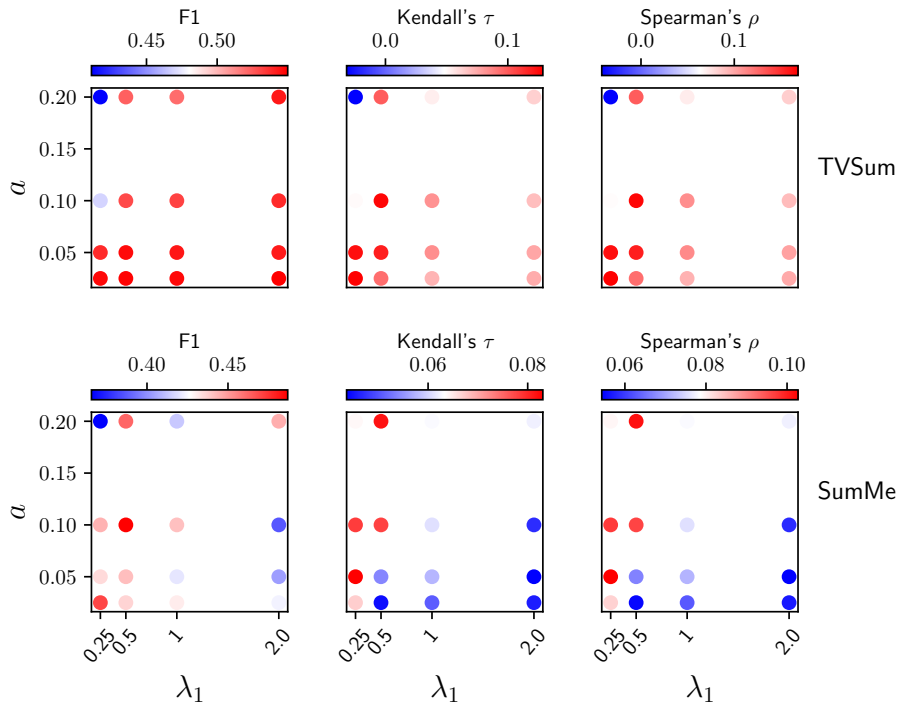


Figure 1: Accompanying figure for Table A4

3.2. $\bar{\mathcal{L}}_{\text{align}}$ & $\bar{H}_{\hat{\theta}}$ & $\bar{\mathcal{L}}_{\text{uniform}}$

As shown in Table A5, the analysis is in general similar to that made in Table A4. However, we can observe that the performance has been obviously improved for TVSum but undermined for SumMe due to incorporating $\bar{\mathcal{L}}_{\text{uniform}}$. We will explain this phenomenon in Section 7.

The results in the main paper for the YT8M training were produced for TVSum with $(a, \lambda_1) = (0.05, 0.25)$ and for SumMe with $(a, \lambda_1) = (0.1, 0.5)$, which are the best settings for the two datasets, respectively. This was done for avoiding the effect of the hyperparameters when we ablate the different components, *i.e.*, $\bar{\mathcal{L}}_{\text{align}}$, $\bar{H}_{\hat{\theta}}$, and $\bar{\mathcal{L}}_{\text{uniform}}$. The results in the main paper for the standard training were produced with $(a, \lambda_1) = (0.1, 0.5)$, which is stable for both TVSum and SumMe when only few videos are available for training.

Table A5: The ablation results for a and λ_1 with $\bar{\mathcal{L}}_{\text{align}}$ & $\bar{H}_{\hat{\theta}}$ & $\bar{\mathcal{L}}_{\text{uniform}}$ used for importance score calculation. See the text for analysis.

Hyper-params		TVSum			SumMe		
a	λ_1	F1	τ	ρ	F1	τ	ρ
0.025	0.25	58.2	0.1403	0.1842	40.2	0.0341	0.0421
	0.50	57.3	0.0548	0.0716	38.7	0.0083	0.0101
	1.00	55.9	-0.0058	-0.0078	40.6	-0.0191	-0.0239
	2.00	54.5	-0.0087	-0.0112	40.3	-0.0069	-0.0087
0.05	0.25	58.5	0.1564	0.2050	42.7	0.0618	0.0765
	0.50	57.2	0.1205	0.1577	38.7	0.0182	0.0223
	1.00	55.5	0.0427	0.0563	39.7	0.0094	0.0114
	2.00	54.1	0.0074	0.0098	40.1	-0.0027	-0.0034
0.1	0.25	56.0	0.0743	0.0971	42.4	0.0737	0.0914
	0.50	57.8	0.1421	0.1866	43.2	0.0449	0.0553
	1.00	54.7	0.0446	0.0587	41.6	0.0130	0.0160
	2.00	54.7	0.0097	0.0127	41.3	-0.0142	-0.0176
0.2	0.25	50.9	-0.0267	-0.0355	41.8	0.0597	0.0740
	0.50	56.4	0.1178	0.1540	46.6	0.0626	0.0775
	1.00	50.7	0.0053	0.0067	39.0	0.0087	0.0109
	2.00	54.7	0.0162	0.0212	41.1	-0.0064	-0.0079

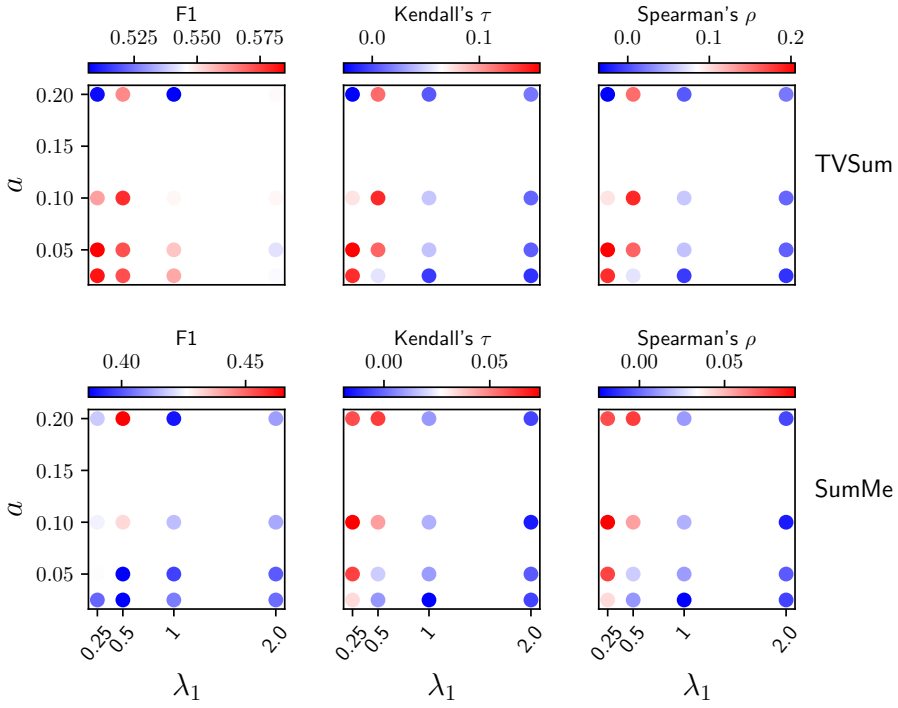


Figure 2: Accompanying figure for Table A5

4. Ablation on Model Size and Comparison with DR-DSN on Youtube8M.

In this section, we show the ablation results in Table A6 for different sizes of the Transformer encoder [17], where the number of layers and the number of attention heads are varied. Meanwhile, we compare the results with those obtained from DR-DSN [19] trained on the same collected Youtube8M videos, as DR-DSN has the best τ and ρ among past unsupervised methods and is the only one that has a publicly available official implementation.

As can be observed in Table A6, the model performance is generally stable with respect to the model sizes, and we choose 4L8H as our default as reported in Section 1. Moreover, the DR-DSN has a hard time generalizing well to the test videos when trained on the Youtube8M videos.

We also recorded the training time on Youtube8M for our model and DR-DSN to show that our training objective based on the contrastive losses is much more efficient than DR-DSN's reinforcement learning-based one that bootstraps online-generated summaries. We chose DR-DSN for comparison in this regard because it is the most lightweight model (a single layer of bi-directional LSTM) and the only one that has officially released code among all the unsupervised methods. Our heaviest model (4L8H) only took around 10s for one pass through our Youtube8M dataset, and only took 40 epochs to complete the training. However, DR-DSN took 140s per epoch and 100 epochs to reach the performance in Table A6. The performance stayed stable if we kept training DR-DSN for more epochs. Both experiments were done on a single NVIDIA RTX A6000.

Table A6: Ablation results for the model size together with comparison with DR-DSN [19] trained on the same Youtube8M videos, where 2L2H represents "2 layers 2 heads" and the rest goes similarly. All the three components $\bar{\mathcal{L}}_{\text{align}}$ & $\bar{H}_{\hat{\theta}}$ & $\bar{\mathcal{L}}_{\text{uniform}}$ are used with $(a, \lambda_1) = (0.05, 0.25)$ for both SumMe and TVSum for fair comparison with DR-DSN, which also uses a representativeness-based training objective.

	TVSum			SumMe		
	F1	τ	ρ	F1	τ	ρ
DR-DSN [19]	51.62	0.0594	0.0788	39.82	-0.0142	-0.0176
2L2H	58.0	0.1492	0.1953	42.9	0.0689	0.0850
2L4H	58.1	0.1445	0.1894	42.8	0.0644	0.0794
2L8H	58.8	0.1535	0.2011	44.0	0.0584	0.0722
4L2H	57.4	0.1498	0.1963	45.3	0.0627	0.0776
4L4H	58.3	0.1534	0.2009	43.1	0.0640	0.0790
4L8H	58.5	0.1564	0.2050	42.7	0.0618	0.0765

5. Comparing Different Pre-trained Features

As our method can directly compute importance scores using pre-trained features, it is also essential for it to be able to work with different kinds of pre-trained features. To prove this, we computed and evaluated the importance scores generated with 2D supervised features, 3D supervised features and 2D self-supervised features in Table A7.

In general, different 2D features, whether supervised or self-supervised, all deliver decent results. Differences exist but are not huge. The conclusion made in the main text that $\bar{\mathcal{L}}_{\text{unif}}$ helps TVSum but undermines SumMe also holds for most of the features. Based on this, we conclude that as long as the features contain decent semantic information learned from either supervision or self-supervision, they are enough for efficient computation of the importance scores for video summarization. The performance of these features transferred to different downstream image tasks does not necessarily generalize to our method for video summarization, as the latter only requires reliable semantic information (quantified as dot products) to calculate heuristic metrics for video frames. After all, it may be reasonable to say that the linear separability of these features for image classification and transferability for object detection and semantic segmentation are hardly related to and thus unable to benefit a task as abstract as video summarization.

However, one interesting observation is that our method does not work well with 3D supervised video features. This is understandable since these 3D features were trained to encode the information of video-level labels, thus encoding less detailed semantic information in each frame on which our method is built. Still, such 3D features contain part of the holistic information of the associated video and may be a good vehicle for video summarization that can benefit from such information. We consider it interesting to incorporate 3D features into our approach and will explore it in our future work.

Table A7: Evaluation results with different pre-trained features. The results were produced under the transfer setting with $a = 0.1$.

		TVSum						SumMe					
		$\bar{\mathcal{L}}^*_{\text{align}}$			$\bar{\mathcal{L}}^*_{\text{align}}$ & $\bar{\mathcal{L}}^*_{\text{unif}}$			$\bar{\mathcal{L}}^*_{\text{align}}$			$\bar{\mathcal{L}}^*_{\text{align}}$ & $\bar{\mathcal{L}}^*_{\text{unif}}$		
		F1	τ	ρ	F1	τ	ρ	F1	τ	ρ	F1	τ	ρ
Supervised (2D)	VGG19 [13]	50.62	0.0745	0.0971	55.91	0.1119	0.1473	45.16	0.0929	0.1151	43.28	0.0899	0.1114
	GoogleNet [15]	54.67	0.0985	0.1285	57.09	0.1296	0.1699	41.89	0.0832	0.1031	40.97	0.0750	0.0929
	InceptionV3 [16]	55.02	0.1093	0.1434	55.63	0.0819	0.1082	42.71	0.0878	0.1087	42.30	0.0688	0.0851
	ResNet50 [10]	51.19	0.0806	0.1051	55.19	0.1073	0.1410	42.30	0.0868	0.1076	43.86	0.0737	0.0914
	ResNet101 [10]	51.75	0.0829	0.1081	54.88	0.1118	0.1469	42.32	0.0911	0.1130	44.39	0.0736	0.0913
	ViT-S-16 [5]	53.48	0.0691	0.0903	56.15	0.1017	0.1332	40.30	0.0652	0.0808	40.88	0.0566	0.0701
	ViT-B-16 [5]	52.85	0.0670	0.0873	56.15	0.0876	0.1152	42.10	0.0694	0.0860	41.65	0.0582	0.0723
	Swin-S [11]	52.05	0.0825	0.1082	57.58	0.1120	0.1475	41.18	0.0880	0.1090	41.63	0.0825	0.1022
Supervised (3D)	R3D50 [7]	52.09	0.0590	0.0766	53.35	0.0667	0.0869	37.40	0.0107	0.0138	41.03	0.0150	0.0190
	R3D101 [7]	49.77	0.0561	0.0727	52.15	0.0644	0.0834	33.62	0.0173	0.0216	34.96	0.0212	0.0264
Self-supervised (2D)	MoCo [9]	51.31	0.0797	0.1034	55.97	0.1062	0.1390	42.01	0.0768	0.0953	43.19	0.0711	0.0882
	DINO-S-16 [3]	52.50	0.0970	0.1268	57.57	0.1200	0.1583	42.77	0.0848	0.1050	42.67	0.0737	0.0913
	DINO-B-16 [3]	52.48	0.0893	0.1170	57.02	0.1147	0.1515	41.07	0.0861	0.1066	44.14	0.0679	0.0843
	BEiT-B-16 [2]	49.64	0.1125	0.1468	56.34	0.1270	0.1665	36.91	0.0554	0.0686	38.48	0.0507	0.0629
	MAE-B-16 [8]	50.40	0.0686	0.0892	54.58	0.1013	0.1327	40.32	0.0560	0.0695	39.46	0.0484	0.0601

6. Unstable F1 scores

We observed that F1 values could be highly unstable for TVSum. When there are zeros in the predicted importance scores, the F1 values will be unreasonably low for TVSum but relatively stable for SumMe. This happens because when all segments have non-zero scores, the knapsack algorithm favors selecting short segments, which is how the ground-truth summaries for TVSum were created [18]. With such ground-truth summaries, the predicted importance scores without zero values can already have a good starting point for the F1 value since they basically go through the same knapsack algorithm with the short-segment bias used to create the ground-truth. Similar observations were made in [12]. Segments with zero scores will not be selected by the knapsack algorithm and make the problem harder to solve, thus leading to low F1 values for TVSum. The SumMe summaries are not created from the knapsack and directly come from the users. Therefore, the F1 values on SumMe are less affected.

To make a fair comparison with previous work, which usually does not have zeros in the predicted scores, we chose to add a small value ϵ (*e.g.*, 0.05) to the predicted scores. However, this does not address the unstableness of the F1 values.

Let us consider another choice to avoid zero scores: $\exp(p-1)$, where $p \in [0, 1]$ is the predicted importance score for a frame. We obtain the results in Table A8 by using these two choices without changing any other part of the method.

As can be observed in Table A8, merely shifting or scaling the importance scores without changing their relative magnitude can lead to different F1 values for both TVSum and SumMe. At the same time, the correlation coefficients are much more stable. Thus, we reinforced the conclusion in [12] that the F1 value is not as stable as the correlation coefficients.

Table A8: The results obtained with different strategies for avoiding zero values in the predicted importance scores. ϵ stands for the operation $p + \epsilon$ and exp for $\exp(p-1)$, where $p \in [0, 1]$ is the importance score for a frame.

	TVSum						SumMe					
	F		τ		ρ		F1		τ		ρ	
	exp	ϵ	exp	ϵ	exp	ϵ	exp	ϵ	exp	ϵ	exp	ϵ
$\bar{\mathcal{L}}_{\text{align}}$&$\bar{H}_{\hat{\theta}}$	58.1	53.8	0.123	0.124	0.1612	0.1624	46	48.7	0.0776	0.078	0.0958	0.0964
$\bar{\mathcal{L}}_{\text{align}}$&$\bar{H}_{\hat{\theta}}$&$\bar{\mathcal{L}}_{\text{uniform}}$	59.4	58.5	0.1563	0.1564	0.2048	0.205	42.86	43.2	0.0441	0.0449	0.0544	0.0553

7. The Behavior of $\bar{\mathcal{L}}^*_{\text{uniform}}$ on SumMe.

In the main text, we saw that $\bar{\mathcal{L}}^*_{\text{uniform}}$ improved performance for TVSum but hurt for SumMe. We now discuss why this happened.

Firstly, when frames with zero annotated scores (recall the 15% constraint for SumMe) have high $\bar{\mathcal{L}}^*_{\text{align}}$, $\bar{\mathcal{L}}^*_{\text{uniform}}$ will reinforce such false confidence as long as it is not low enough to almost zero out $\bar{\mathcal{L}}^*_{\text{align}}$. Such frames may not necessarily be highly irrelevant and have low $\bar{\mathcal{L}}^*_{\text{uniform}}$ to begin with because most of them, though still informative, are eliminated by the 15% constraint. As shown in Table A9, keeping the most confidence values of $\bar{\mathcal{L}}^*_{\text{align}}$ by simple thresholding, which may help remove false predictions of $\bar{\mathcal{L}}^*_{\text{align}}$, can mitigate this issue. The results in the main text are all without such thresholding for a fair comparison with previous work.

Though the results in Table A9 show improvement for $\bar{\mathcal{L}}^*_{\text{uniform}}$, it is not as striking as that for TVSum in terms of τ and ρ. Compared to TVSum videos, many videos in SumMe already contain quite consistent frames due to their slowly evolving properties. Such slowly evolving features can be visualized by T-SNE plots shown in Fig. 4a under the comparison against Fig. 4b. For videos with such consistent contents, the $\mathcal{L}^*_{\text{uniform}}$ (before normalization) tend to be high for most of the frames. We show the normalized histogram of $\mathcal{L}^*_{\text{uniform}}$ for both TVSum and SumMe videos in Figure 3. As can be observed, SumMe videos have distinctly higher $\mathcal{L}^*_{\text{uniform}}$ than those of TVSum videos.

Consequently, for videos already possessing consistent contents, $\mathcal{L}^*_{\text{uniform}}$ normalized to 0 and 1 ($\bar{\mathcal{L}}^*_{\text{uniform}}$) will filter out frames that are relatively the least consistent. However, such frames are still relevant and may well likely possess a certain level of diversity, which the annotators of SumMe favor (see the results for $\bar{\mathcal{L}}^*_{\text{align}}$ for SumMe in Table 1, 2 and 3 in the main text). Thus, removing such frames using the $\bar{\mathcal{L}}^*_{\text{uniform}}$ will cause harm to the correlations. For some videos with truly noisy frames, $\bar{\mathcal{L}}^*_{\text{uniform}}$ retains its functionality of attenuating their importance, leading to better correlations. Thus, the pros and cons brought by $\bar{\mathcal{L}}^*_{\text{uniform}}$ may cancel out on average for the whole SumMe dataset, eventually yielding minor improvement in correlation (*i.e.* τ and ρ). $\bar{\mathcal{L}}^*_{\text{uniform}}$ thus suits better complex videos such as those in TVSum, since such videos may inevitably contain noisy frames that can be well detected by $\bar{\mathcal{L}}^*_{\text{uniform}}$.

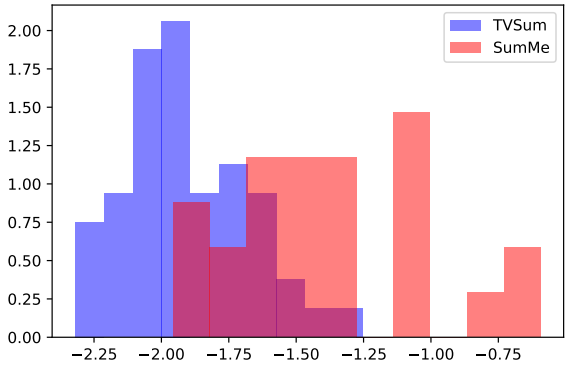
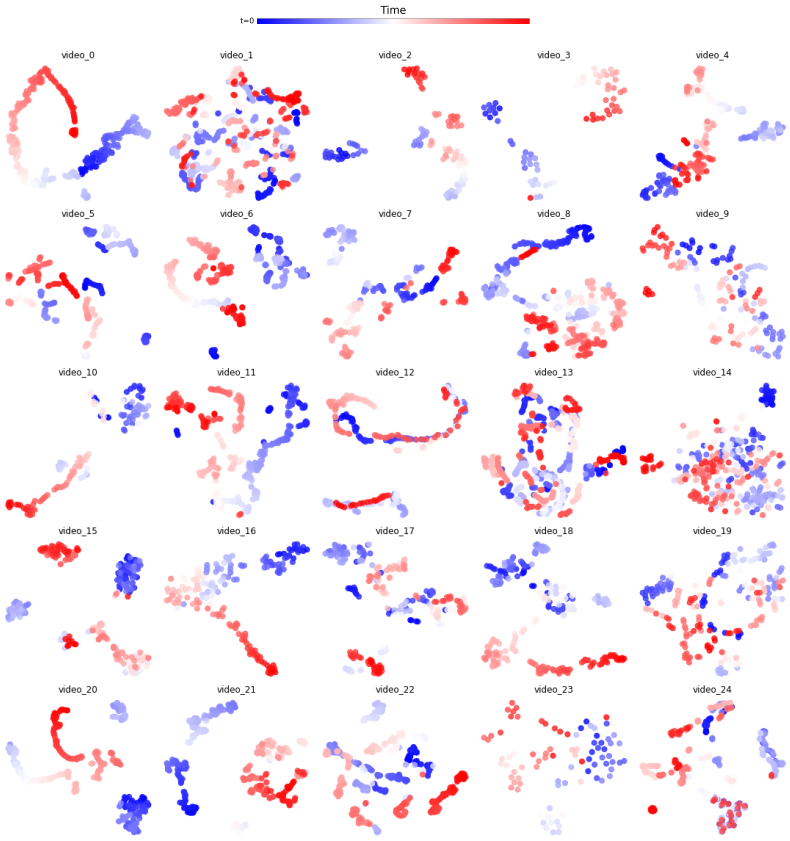


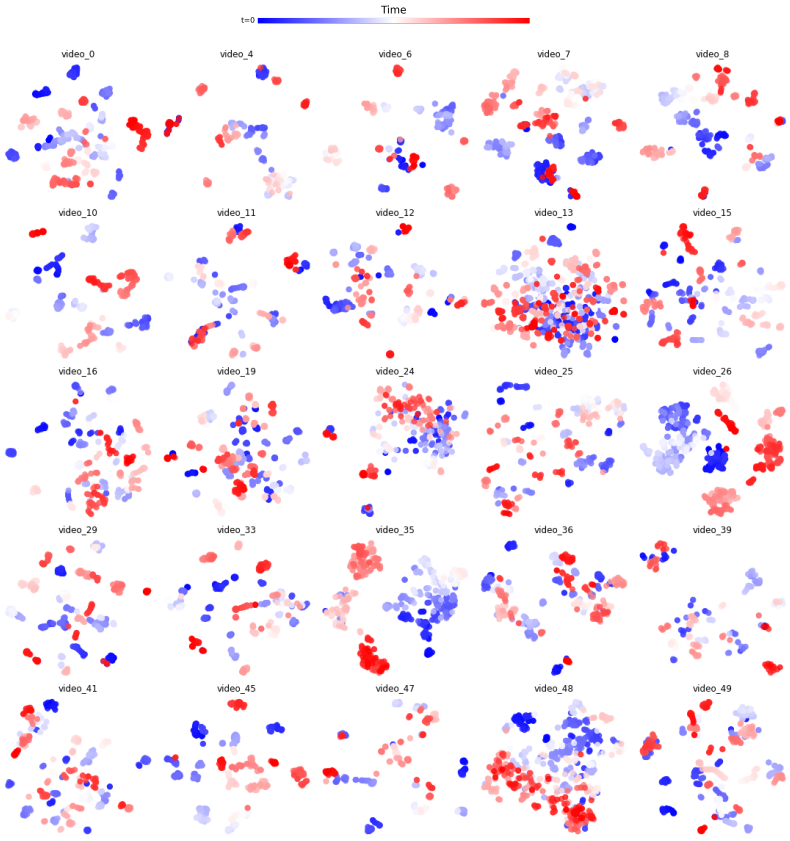
Figure 3: The histogram (density) of $\bar{\mathcal{L}}^*_{\text{uniform}}$ (before normalization) for TVSum and SumMe videos. It is clear that SumMe videos have distinctly higher values than those for TVSum videos.

Table A9: The results were obtained from the transfer setting on SumMe directly with pre-trained features with $a = 0.1$. Thresholding means simply setting $\bar{\mathcal{L}}^*_{\text{align}}[\bar{\mathcal{L}}^*_{\text{align}} < \alpha\text{AVG}(\bar{\mathcal{L}}^*_{\text{align}})] = 0$, where AVG means taking average over the whole video and $\alpha = 1.1$.

	Before thresholding			After thresholding		
	F	τ	ρ	F	τ	ρ
$\bar{\mathcal{L}}^*_{\text{align}}$	39.4	0.0769	0.0939	40.7	0.0969	0.1119
$\bar{\mathcal{L}}^*_{\text{align}}$&$\bar{\mathcal{L}}^*_{\text{uniform}}$	41.7	0.0597	0.073	44.5	0.0982	0.1134



(a) T-SNE plots for all the SumMe videos.



(b) T-SNE plots for randomly selected 25 TVSum videos.

Figure 4: TSNE plots for SumMe and TVSum videos. See text for analysis.

8. Complete Figures for the Qualitative Analysis

In Fig.5, we provide a larger version of the figure for the qualitative analysis in the main paper (the top and the middle) together with another one showing how $\bar{H}_{\hat{\theta}}$ improves $\bar{\mathcal{L}}_{\text{align}}$&$\bar{\mathcal{L}}_{\text{uniform}}$ (bottom). Please refer to the main paper for the analyses of the first two figures.

As shown in the bottom figure of Fig. 5, the green bar selects a frame with nontrivial local dissimilarity and global consistency but low uniqueness. It turns out that the frame contains only texts and has semantic neighbors with different texts, which can lead to high local dissimilarity. As the video is an instruction video, such frames with only textual illustrations should be prevalent, thus also incurring nontrivial global consistency. However, such textual frames are common in nearly all kinds of instruction videos or other videos with pure textual frames, hence the low uniqueness score. The red bar selects a frame where some treatment is ongoing for a dog, which is specific to the current video and satisfies the local dissimilarity and the global consistency. The **black** bar selects a frame with low local dissimilarity and global consistency but high uniqueness, which is a frame showing off the vet medicine. Though quite specific to the video, the frame is not favored by the other two metrics.

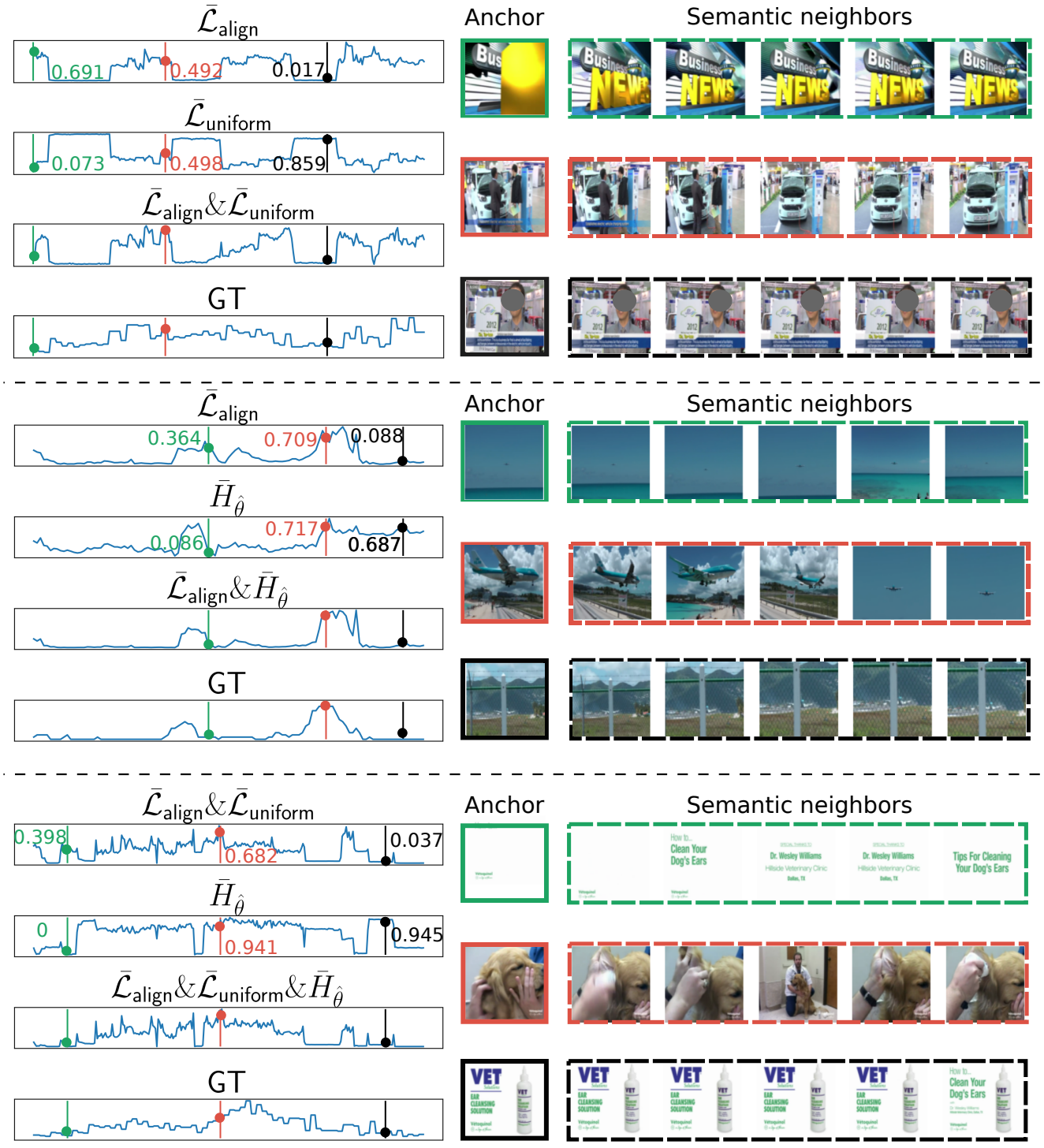


Figure 5: The complete figure for the qualitative analysis in the main paper.All the important scores are scaled to $[0, 1]$ for visualization.

References

[1] 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.

[2] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[3] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *CVPR*, 2021.

[4] Sandra Eliza Fontes De Avila, Ana Paula Brandao Lopes, Antonio da Luz Jr, and Arnaldo 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.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

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

[7] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *CVPR*, pages 6546–6555, 2018.

[8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pages 16000–16009, 2022.

[9] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[11] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021.

[12] Mayu Otani, Yuta Nakashima, Esa Rahtu, and Janne Heikkila. Rethinking the evaluation of video summaries. In *CVPR*, 2019.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

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

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

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

[19] Kaiyang Zhou, Yu Qiao, and Tao Xiang. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In *AAAI*, 2018.