Supplementary material of
Self-Supervised Video Similarity Learning

Giorgos Kordopatis-Zilos Giorgos Tolias Christos Tzelepis Ioannis Kompatsiaris
Ioannis Patras Symeon Papadopoulos

	Retrieval			**Detection**		
λ	**VCDB**	**FIVR**	**EVVE**	**VCDB**	**FIVR**	**EVVE**
0	89.2	81.6	62.6	80.3	72.3	75.4
1	94.1	85.9	64.7	88.2	80.6	78.5
3	95.2	87.0	65.9	90.1	81.7	78.9
5	95.3	86.3	65.2	90.5	81.3	78.1
7	94.6	85.5	64.5	89.2	79.2	78.1

Table 1. Retrieval mAP (%) and detection μAP (%) for S^{2}VS with different values for scale factor λ.

	Retrieval			**Detection**		
τ	**VCDB**	**FIVR**	**EVVE**	**VCDB**	**FIVR**	**EVVE**
0.03	95.2	87.0	65.9	90.1	81.7	78.9
0.05	95.2	86.7	65.4	89.9	80.4	77.5
0.07	95.1	86.0	65.3	89.7	78.8	76.8
0.1	95.1	85.7	65.0	89.5	78.0	75.5

Table 2. Retrieval mAP (%) and detection μAP (%) for S^{2}VS with different values for temperature τ.

A. Additional ablations

In this section, we continue our ablations studying how various hyperparameters of the training processes affect the final performance of the proposed method. We conduct further experiments on the same datasets as in the main paper.

Impact of λ hyperparameter: In Table 1, we report the results of the proposed approach for different values of λ. We observe that the performance is not significantly affected for smaller than the default $\lambda = 3$ values. However, for larger values, it steadily decreases as the network focuses more on the SSHN loss than the InfoNCE one.

Impact of temperature τ: Table 2 presents the performance of S^{2}VS trained with different temperature values in the InfoNCE loss. The performance decreases for values other than the default $\tau = 0.03$, which is more noticeable on detection tasks where larger μAP drops are reported. We do not go lower due to numeric instability and overflow issues during training, causing our network to collapse.

	Retrieval			**Detection**		
T_B	**VCDB**	**FIVR**	**EVVE**	**VCDB**	**FIVR**	**EVVE**
8	86.8	70.1	52.6	78.5	57.0	66.7
16	95.1	86.9	65.2	89.6	82.0	78.8
32	95.2	87.0	65.9	90.1	81.7	78.9

Table 3. Retrieval mAP (%) and detection μAP (%) for S^{2}VS with different number of T_B frames.

		Retrieval			**Detection**		
$\mathcal{L}_{\text{ss}}$	$\mathcal{L}_{\text{hn}}$	**VCDB**	**FIVR**	**EVVE**	**VCDB**	**FIVR**	**EVVE**
✗	✗	89.2	81.6	62.6	80.3	72.3	75.4
✓	✗	91.9	84.0	61.8	79.5	68.1	64.5
✗	✓	94.5	84.8	64.6	88.1	78.9	75.1
✓	✓	95.2	87.0	65.9	90.1	81.7	78.9

Table 4. Retrieval mAP (%) and detection μAP (%) for S^{2}VS trained with different configurations for $\mathcal{L}_{\text{sshn}}$ loss.

task	**VCDB**	**FIVR**	**EVVE**
retrieval	95.2 ± 0.07	87.0 ± 0.20	65.9 ± 0.18
detection	90.0 ± 0.09	81.8 ± 0.21	78.8 ± 0.51

Table 5. Mean and standard deviation of retrieval mAP (%) and detection μAP (%) of S^{2}VS.

Impact of the number of T_B frames: Table 3 displays the results of our method trained with different number of T_B frames for the input videos. In almost all tasks, the larger the size of input videos, the better the performance. Also, when T_B is small, the network fails to learn anything useful. This is expected as comparing larger videos helps the model better capture temporal structures in the similarity matrices.

Impact of $\mathcal{L}_{\text{sshn}}$ terms: Table 4 illustrates the results of the proposed approach trained with the different terms of $\mathcal{L}_{\text{sshn}}$ loss, with $\mathcal{L}_{\text{ss}}$ and $\mathcal{L}_{\text{hn}}$ standing for the self-similarity and hard negative part of the loss (cf. Equation 3, in the main paper). This highlights that both terms are necessary for the effective training of the system.

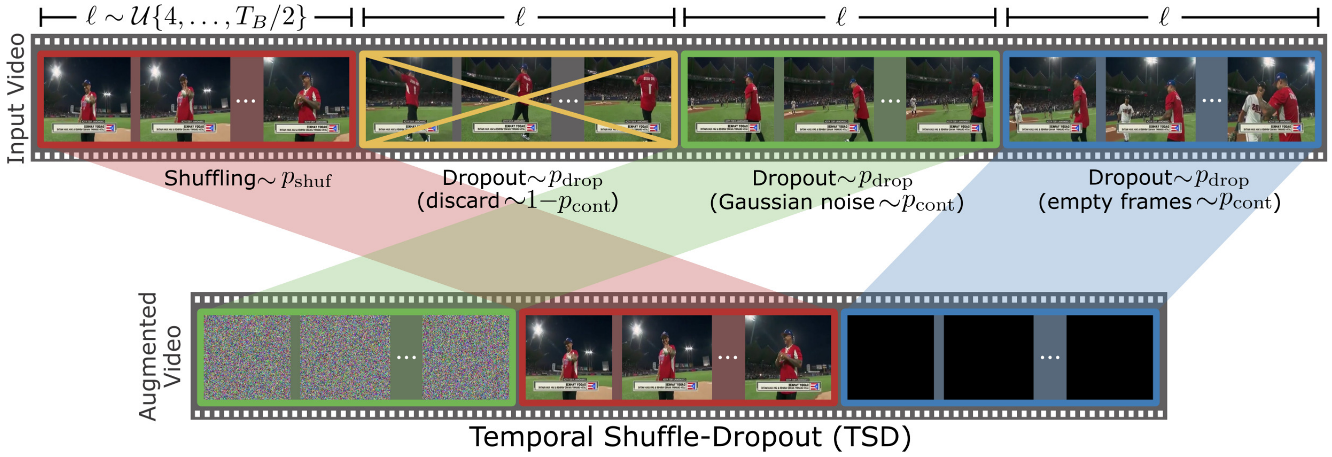

$\ell \sim \mathcal{U}\{4,\ldots,T_B/2\}$
ℓ
ℓ
ℓ
Input Video
Shuffling$\sim p_{\text{shuf}}$
Dropout$\sim p_{\text{drop}}$ (discard $\sim 1-p_{\text{cont}}$)
Dropout$\sim p_{\text{drop}}$ (Gaussian noise $\sim p_{\text{cont}}$)
Dropout$\sim p_{\text{drop}}$ (empty frames $\sim p_{\text{cont}}$)
Augmented Video
Temporal Shuffle-Dropout (TSD)

Figure 1. Illustration of the proposed Temporal Shuffle-Dropout (TSD) augmentation scheme, where an input video is split into short clips of fixed length $\ell \sim \mathcal{U}\{4,\cdots,T_B/2\}$, each one of them being shuffled with probability p_{shuf}, or dropped out with probability p_{drop} (i.e., filled with empty frames or Gaussian noise with probability p_{cont}, or entirely discarded with probability $1 - p_{\text{cont}}$). This guarantees the preservation of the local (clip-level) temporal order, but alters the global (video-level) temporal structure.

Mean and standard deviation: Table 5 shows the mean and standard deviation of mAP and μAP for retrieval and detection, respectively, of S^{2}VS trained and evaluated seven times with different seeds. Generally, the performance is steady with small fluctuations, especially for VCDB, where the standard deviation is less than 0.1%, and with the largest deviations reported on EVVE.

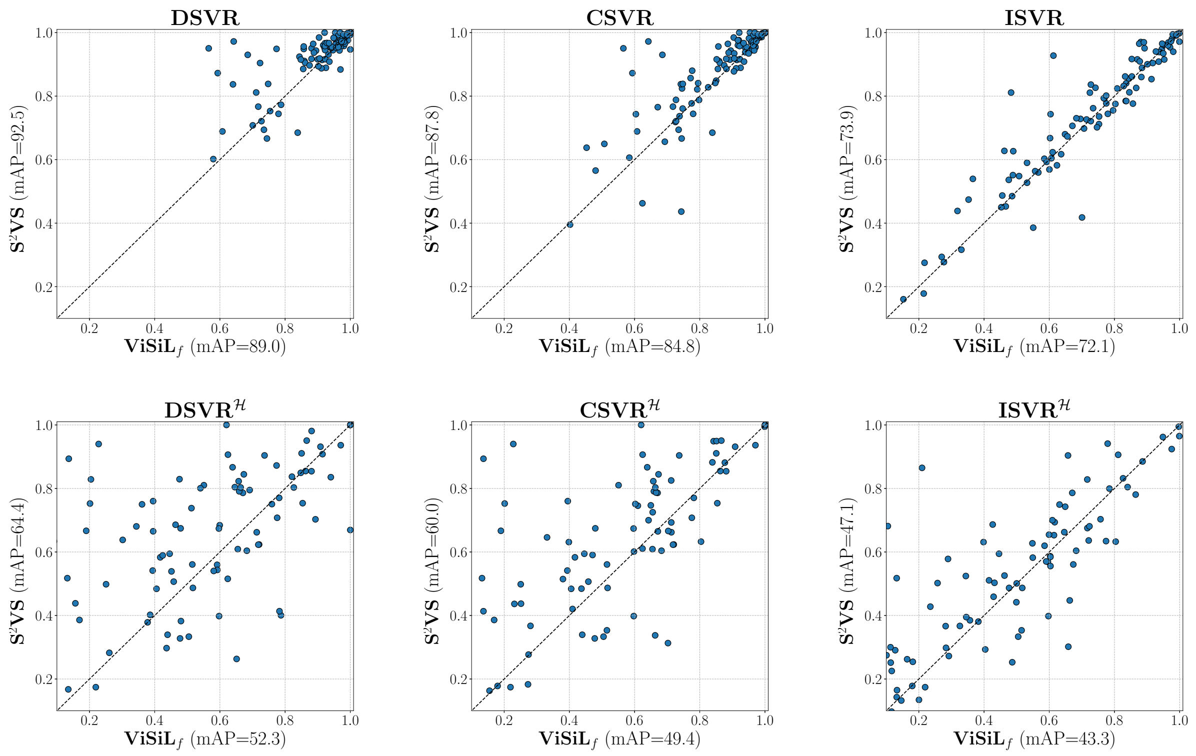
B. Comparison per video query

In this section, we compare the per query performance of the proposed approach with **ViSiL**$_f$ [1] and **DnS** [2]. Figure 2 illustrates our method's Average Precision (AP) for each query on the FIVR-200K dataset and its hard version in comparison to ViSiL$_f$ and DnS. The diagonal line indicates the cases where there is a tie performance between the two compared approaches. In the normal settings of the dataset, comparing S^{2}VS with ViSiL$_f$, most of the queries lie on the part above the diagonal line for all three tasks, indicating that the proposed method achieves better results on them, while a large number of points appear on the top-right corner indicating easy queries for all approaches. Additionally, comparing S^{2}VS with DnS, more queries are close to diagonal, which means that the two approaches have similar performance. Nevertheless, it is noteworthy that the vast majority of the queries are very close to one, dominating the results and making differences in performance less apparent in the final evaluation.

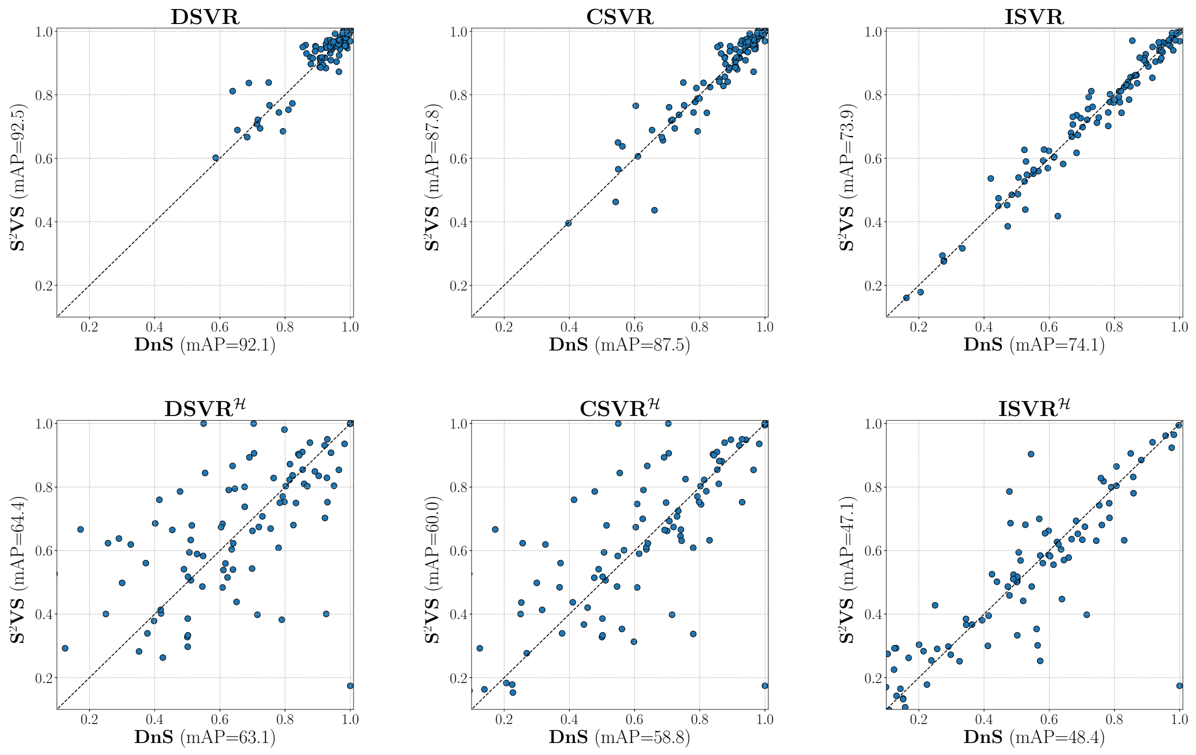
However, in the hard version of the dataset, the query APs are more spread out in the off-diagonal area, highlighting that the methods' performance is less correlated than in the initial settings. In comparison with ViSiL$_f$, more queries lie on the upper left part of the figure, indicating that our S^{2}VS performs better on them. Such queries dominate the mAP and result in a significant performance gap between the two approaches. On the other hand, in comparison with DnS, the queries appear more equally distributed in the off-diagonal areas, resulting in very similar mAP scores. Hence, one method is more effective than the other on a different set of queries. This highlights that there is still much room for improvement, *e.g.*, selecting the appropriate method for the corresponding queries.

C. Temporal Shuffle Dropout visualization

For a better understanding of the proposed Temporal Shuffle Dropout (TSD) transformation scheme, in Figure 1, we illustrate the various augmentation operations applied in an input video. In this example, the input video is first split into four clips (each being highlighted in a different colour) of $\ell \sim \mathcal{U}\{4,\cdots,T_B/2\}$ frames, as described in Section 3 of the main paper. Then, a separate operation is applied on each clip, with certain probabilities as shown in Figure 1, in order to form the augmented video that is then used during the training of our approach: (i) the red one is shuffled, i.e., it is moved between the green and the blue clips (shuffling phase), (ii) the yellow one is discarded entirely and it is not part of the augmented video (dropout phase), (iii) the content of the green clip is replaced with Gaussian noise (dropout phase), and (iv) the frames of the blue clip are replaced with empty frames in the augmented video (dropout phase). The proposed TSD encourages this way, the preservation of some local (clip-level) information and provides certain variations in the global temporal structure of the video, which is beneficial to the generalisation ability of the proposed framework.



(a) ViSiL$_f$ – S^{2}VS



(b) DnS – S^{2}VS

Figure 2. Average Precision (AP) per query for the proposed S^{2}VS in comparison with the **ViSiL$_f$** [1], and **DnS** [2] on the three subtasks of FIVR-200K and on its hard subset FIVR-200K$^{\mathcal{H}}$ (cf. Section 5.1, in the main paper).

D. Additional implementation details

All of our models are implemented with the PyTorch [3] library. Table 6 displays all hyperparameters used for the network training and augmentations.

For global transformations, we use $N_{RAug} = 2$ consecutive transformations with $M_{RAug} = 9$ magnitude in RandAugment. The frame transformations are applied with probability $p_{overlay} = 0.3$ for each text and emoji overlay and with $p_{blur} = 0.5$ for blurring. For the temporal transformations, we select TSD with probability $p_{tsd} = 0.5$ and the rest transformations with probability 0.1 each. For TSD, shuffling and dropout are applied with probabilities $p_{shuf} = 0.5$ and $p_{drop} = 0.3$, respectively, with the latter discarding clips or removing their content with $p_{cont} = 0.5$. Finally, the video-in-video transformation is applied with probability $p_{viv} = 0.5$, and the donor video is downsampled with a factor λ_{viv} ranging in (0.3, 0.7).

To enable vectorization during training and facilitate augmentations, we load the original videos with 1 fps, resized to 256 pixels according to their smaller side. Also, to assert that the two augmentation versions overlap, we select a clip of consecutive frames equal to $2T_B$. If the initial video is shorter, it is repeated in the time axis until it reaches the necessary length. We follow the literature and resize/crop the videos to 256/224 pixels.

All experiments were conducted on a Linux machine with an Intel i9-7900X CPU and two Nvidia 3090 GPUs. Times and storage requirements are the same as the fine-grained attention student reported on the DnS paper [2].

Parameter	**Notation**	**Value**
Training process		
Iterations	-	30,000
Batch size	-	64
Optimizer	-	AdamW
Learning rate	-	$5 \cdot 10^{-5}$
Learning rate decay	-	cosine
Warmup iterations	-	1,000
Weight decay	-	0.01
# of input frames in batch	T_B	32
Frame size in batch	H_B,W_B	224
InfoNCE temperature	τ	0.07
SSHN loss factor	λ	3
Sim. regularization factor	r	1
Global transformations		
RandAug., # of transf.	N_{RAug}	2
RandAug., magnitude	M_{RAug}	9
Frame transformations		
Overlay prob.	$p_{overlay}$	0.3
Blur prob.	p_{blur}	0.5
Temporal transformations		
TSD prob.	p_{tsd}	0.5
Fast forward prob.	p_{ff}	0.1
Slow motion prob.	p_{sm}	0.1
Reverse prob.	p_{rev}	0.1
Pause prob.	p_{pau}	0.1
TSD shuffle prob.	p_{shuf}	0.5
TSD dropout prob.	p_{drop}	0.3
TSD content drop prob.	p_{cont}	0.5
Video-in-video		
ViV prob.	p_{viv}	0.3
ViV factor range	λ_{viv}	(0.3, 0.7)

Table 6. Implementation details of the training process and the augmentations parameters.

References

[1] Giorgos Kordopatis-Zilos, Symeon Papadopoulos, Ioannis Patras, and Ioannis Kompatsiaris. ViSiL: Fine-grained spatio-temporal video similarity learning. In *ICCV*, 2019. 2, 3

[2] Giorgos Kordopatis-Zilos, Christos Tzelepis, Symeon Papadopoulos, Ioannis Kompatsiaris, and Ioannis Patras. DnS: Distill-and-Select for Efficient and Accurate Video Indexing and Retrieval. *IJCV*, 2022. 2, 3, 4

[3] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 4