Supplemental Material
Representation Learning via Global Temporal Alignment and Cycle-Consistency

Isma Hadji, Konstantinos G. Derpanis, Allan D. Jepson
Samsung AI Centre Toronto
{isma.hadji, allan.jepson}@samsung.com   k.derpanis@partner.samsung.com

Our supplementary material is organized as follows: Sec. 1 derives several properties of the two smooth minimum approximations introduced in the main manuscript. Sec. 2 provides details of our evaluation baselines. Sec. 3 further explains our re-organization of FineGym to satisfy our training requirements. For completeness, Secs. 4 and 5 provide additional fine-grained action recognition results. Similarly, Sec. 6 provides additional alignment results on PennAction. Finally, Sec. 7 probes our learned representations via visualizations to ascertain what information is captured. A supplemental video highlighting the different applications of the proposed loss is also provided.

1. Smooth Minimum Properties

We derive several properties of the two smooth minimum approximations, smoothMin and $\min^{\gamma}$. For more general applicability, we consider any vector-valued input, $\mathbf{a} \in \mathbb{R}^N$.

smoothMin properties. The smoothMin function is defined by

$$\text{smoothMin}(\mathbf{a};\gamma) = \begin{cases} \min\{a_i \mid 1 \leq i \leq N\}, & \gamma = 0 \\ \frac{\sum_{i=1}^N a_i e^{-a_i/\gamma}}{\sum_{j=1}^N e^{-a_j/\gamma}}, & \gamma > 0 \end{cases}, \tag{1}$$

where γ denotes a temperature hyper-parameter. Define a_0 to be the minimum coefficient in $\mathbf{a}$, that is, $a_0 = \min(\mathbf{a})$. Then for $\gamma > 0$ it follows that

$$\begin{aligned} \text{smoothMin}(\mathbf{a};\gamma) &= \frac{\sum_{i=1}^N a_i e^{-(a_i - a_0)/\gamma}}{\sum_{j=1}^N e^{-(a_j - a_0)/\gamma}} \\ &= a_0 + \frac{\sum_{i=1}^N (a_i - a_0) e^{-(a_i - a_0)/\gamma}}{\sum_{j=1}^N e^{-(a_j - a_0)/\gamma}} \\ &= a_0 + \gamma \frac{\sum_{i=1}^N ((a_i - a_0)/\gamma) e^{-(a_i - a_0)/\gamma}}{\sum_{j=1}^N e^{-(a_j - a_0)/\gamma}} \\ &= \min(a) + \gamma \, \text{smoothMin}((\mathbf{a} - a_0)/\gamma; 1). \end{aligned} \tag{2}$$

Therefore, the smoothness penalty satisfies

$$\begin{aligned} d(\mathbf{a};\gamma) &:= \text{smoothMin}(\mathbf{a};\gamma) - \min(\mathbf{a}), \\ &= \gamma \, \text{smoothMin}((\mathbf{a} - a_0)/\gamma; 1). \end{aligned} \tag{3}$$

Note that from (1) it follows that $d(\mathbf{a};\gamma) \geq 0$, so we have $\text{smoothMin}(\mathbf{a};\gamma) \geq \min(\mathbf{a})$, *i.e.*, it provides an upper bound on the true minimum.

The maximum possible value of the penalty $d(\mathbf{a};\gamma) = \gamma\text{smoothMin}(\mathbf{a} - a_0)/\gamma; 1)$ is also of interest. Given (3) it is sufficient to maximize $\text{smoothMin}(\mathbf{r}; 1)$ subject to $r_1 = 0$ and $r_i \geq 0$ for $i \in \{2, \dots N\}$. By setting $\frac{\partial}{\partial r_i}\text{smoothMin}(\mathbf{r}; 1) = 0$ and simplifying, we find

$$1 - r_i + \text{smoothMin}(\mathbf{r}; 1) = 0 \tag{4}$$

for $i \geq 2$. This implies all the r_i's for $i > 1$ are equal at the maximum. Using (1) in (4), setting $r_1 = 0$ and $r_i = x$ for $i > 1$, and simplifying, we find x must be the solution of

$$x - 1 = (N - 1)e^{-x}. \tag{5}$$

For $N \geq 1$, (5) has a unique solution $x(N) \geq 1$. This implies that the penalty function (3) only has maxima, $\mathbf{a}$, for which there is exactly one distinct minimum value $a_j = \min(\mathbf{a})$, and all the other values are in an $(N-1)$-way tie for second largest at $a_i = a_j + x\gamma$ for $i \neq j$, and where x is as in (5).

To compute the value of this maximum, we substitute the resulting vector $\mathbf{r} = \begin{bmatrix} 0 & x(N) & \dots & x(N) \end{bmatrix}^\top$ into $\text{smoothMin}(\mathbf{r}; 1)$ and, by simplifying, we find

$$\begin{aligned} \max d(\mathbf{r}; 1) &= \frac{(N-1)x(N)e^{-x(N)}}{1 + (N-1)e^{-x(N)}} \\ &= \frac{(N-1)x(N)\left[(x(N)-1)/(N-1)\right]}{1 + (N-1)\left[(x(N)-1)/(N-1)\right]} \\ &= x(N) - 1, \end{aligned} \tag{6}$$

where we used (5) in the second line above.

For concrete examples, we find $x(2) \approx 1.2785$, $x(3) \approx 1.4631$ for $N = 2$ and 3, respectively. Also, for $N \geq 4$ it follows from (5) that $x(N) < \log(N+1)$ (since the left hand side less minus the right is negative at $x = 0$, positive for $x = \log(N+1)$, and the derivative of this difference with respect to x is positive). Therefore, (6) implies that the maximum smoothness penalty when $N = 2$ is roughly 0.2785γ, and 0.4631γ for $N = 3$, which are in agreement with the plot in Fig. 3 of the main manuscript. Moreover, for large N the maximum penalty is $O(\gamma \log(N))$.

$\min^{\gamma}$ properties. Similarly we examine $\min^{\gamma}$, which is defined by

$$\min{}^{\gamma}(\mathbf{a}) = \begin{cases} \min\{a_i\}, & \gamma = 0 \\ -\gamma \log \sum_{i=1}^{N} e^{-a_i/\gamma}, & \gamma > 0 \end{cases}. \tag{7}$$

For $a_0 = \min(\mathbf{a})$ and $\gamma > 0$ we have

$$\begin{aligned} \min{}^{\gamma}(\mathbf{a}) &= -\gamma \log \sum_{i=1}^{N} \left[e^{-(a_i - a_0)/\gamma} e^{-a_0/\gamma} \right], \\ &= -\gamma \log \left[e^{-a_0/\gamma} \sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma} \right], \\ &= -\gamma \left[-a_0/\gamma + \log \sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma} \right], \\ &= \min(\mathbf{a}) - \gamma \log \sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma}. \end{aligned} \tag{8}$$

Recall that $a_0 = \min(\mathbf{a})$ so $a_i = a_0$ for at least one i. Given that all the terms in the sum $\sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma}$ are non-negative, and one of them is 1, we conclude that $\sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma} > 1$. Therefore, we have $\min^{\gamma}(\mathbf{a}) < \min(\mathbf{a})$, *i.e.*, it provides an lower bound on the true minimum.

Moreover, the maximum sum (8) occurs when $a_i = a_0$ for all i (*i.e.*, there is an N-way tie for the minimum). In this case $\sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma} = N$. Therefore, we have

$$\sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma} \in (1, N]. \tag{9}$$

Finally, we see that the smoothness penalty when $\min^{\gamma}$ is used is given by

$$\begin{aligned} d^{\,\gamma}(\mathbf{a}; \gamma) &:= \min{}^{\gamma}(\mathbf{a}; \gamma) - \min(\mathbf{a}), \\ &= -\gamma \ \log \sum_{i=1}^{N} e^{-(a_i - a_0)/\gamma} \\ &\in [-\gamma \log(N), 0). \end{aligned} \tag{10}$$

Therefore, we have shown that $d^{\,\gamma}(\mathbf{a})$ is always negative and achieves its most negative value of $-\gamma \log(N)$ if and only if the input vector $\mathbf{a}$ represents an N-way tie, *i.e.*, $\mathbf{a} = \begin{bmatrix} a_0 & \dots & a_0 \end{bmatrix}^{\top}$.

2. Baselines

We compare our approach to other weakly supervised [3, 5, 2] and self-supervised [1, 4] methods that entail temporal understanding in their definition.

SpeedNet [1] learns a video representation by learning to distinguish between videos played at different speeds.

Time Contrastive Networks (TCN) [5] makes fine-grained temporal distinctions by optimizing a contrastive loss that encourages embeddings of an anchor image and an image taken simultaneously from a different camera viewpoint to be similar, while the embedding of an image taken from the same sequence of the anchor but at a different time instant to be distant from that of the anchor.

Shuffle and Learn (SaL) [4] learns to predict whether a triplet of frames are in the correct order or shuffled.

Discriminative Differentiable Time Warping (D^{3}TW) [2] was originally introduced to learn alignments between video frames and action labels. Its definition includes a discriminative component requiring explicit definitions of positive and negative pairs. A pair is considered positive if all action labels are used in the alignment table, whereas negative pairs are constructed by dropping some of the action labels. We adapt this loss to our video pair alignment setting and define **D^{3}TW*** as follows: a positive pair is constructed by sampling frames from the entire duration of both *video-1* and *video-2* in any pair, whereas we randomly drop portions of *video-2* to constitute a negative pair, thereby mimicking the dropping of action labels as proposed in the original paper.

Temporal Cycle Consistency (TCC) [3] learns fine-grained temporal correspondences between individual video frames by imposing a soft version of cycle consistency on the individual matches.

3. FineGym re-organization

As mentioned in the main mansucript, each video in FineGym is annotated according to a three-level hierarchy denoting the event being performed in the video, the different sets involved in performing the event, and the frame-wise elements (i.e., action phases) involved in each set. To perform any event-level action, a gymnast may perform the different sets in any order. To train our embedding network using our alignment-based method, we re-organize the FineGym dataset such that all sets belonging to the same event appear in the same order in any given video. For example, given floor exercise events, gymnasts can perform four different sets of exercises in any order, we re-organize the clips in each video according to a selected prototype order, as shown Figure 1. These organized event-level videos are used during training and testing.

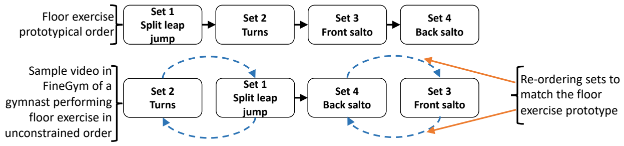

Floor exercise prototypical order
Set 1 Split leap jump
Set 2 Turns
Set 3 Front salto
Set 4 Back salto
Sample video in FineGym of a gymnast performing floor exercise in unconstrained order
Set 2 Turns
Set 1 Split leap jump
Set 4 Back salto
Set 3 Front salto
Re-ordering sets to match the floor exercise prototype

Figure 1. Illustration of our re-organization of FineGym.

4. Fine-grained action recognition

In Sec. 4.4 of the main manuscript, we compared our fine-grained action recognition performance to our baselines using FineGym with two training settings for the backbone framewise encoder. Here, we provide our complete comparison. In addition, to training from scratch (**train-all**) and fine-tuning the batch norm layers (**only-bn**), we include a third experiment consisting of fine-tuning all layers of a ResNet50 model pre-trained on ImageNet (**train-all**).

The full results of all experiments are summarized in Table 1. Consistent with our conclusions in Sec. 4.4, we outperform all the weakly and self-supervised baseline methods by significant margins under all training settings with the best results obtained under the *only-bn* setting.

Method	Training	FineGym101	FineGym290
SpeedNet [1]	scratch	30.40	29.87
TCN [5]		36.52	37.40
SaL [4]		40.25	37.98
D^3TW* [2]		32.10	32.15
TCC [3]		41.78	40.57
Ours		**45.79**	**43.49**
SpeedNet [1]	train all	28.27	30.95
TCN [5]		30.97	35.68
SaL [4]		36.76	39.76
D^3TW* [2]		35.75	33.63
TCC [3]		44.29	40.31
Ours		**47.78**	**44.77**
SpeedNet [1]	only bn	34.38	35.92
TCN [5]		41.75	39.93
SaL [4]		42.68	41.58
D^3TW* [2]		38.21	34.04
TCC [3]		45.62	43.40
Ours		**49.51**	**46.54**

Table 1. Fine-grained action recognition accuracy on both organizations of FineGym.

For completeness, we also provide results of training the SVM classifier for framewise fine-grained action recognition on the original FineGym99 and 288 short clips. The results of this experiment, summarized in Table 2, once again demonstrate the superiority of the proposed approach even under this more challenging setting. Notably, comparison between our method and those reported in [6] is not direct for two main reasons. First, we are targeting framewise accuracy, whereas [6] focuses on clip-level accuracy. Second, we are the first to report weakly supervised results on FineGym.

Method	Training	FineGym99	FineGym288
SpeedNet [1]	only bn	16.86	15.57
TCN [5]		20.02	17.11
SaL [4]		21.45	19.58
D^3TW* [2]		15.28	14.07
TCC [3]		25.18	20.82
Ours		**27.81**	**24.16**

Table 2. Fine-grained action recognition accuracy on the original clips of FineGym99 and FineGym288.

5. Details of fine-grained action recognition

To further investigate the utility of the learned embeddings, we also consider classification results of each event separately. In particular, using the embeddings learned on the entire FineGym101 dataset, we train a separate SVM classifier for each event. The results summarized in Table 3, further confirm the superiority of our approach. These results also show that classifying the sub-actions in the floor exercise (FX) event is the most challenging for all methods. Careful examination of videos in this class revealed wide variations in the way gymnasts perform each sub-action in the floor exercise event, which makes learning a proper alignment especially challenging.

Method	VT, *8cls*	FX, *35cls*	BB, *33cls*	UB, *25cls*
SpeedNet [1]	61.63	12.10	24.25	24.42
TCN [5]	68.82	13.49	29.57	27.14
SaL [4]	70.53	15.16	37.73	33.06
D^3TW* [2]	65.79	13.57	31.96	26.61
TCC [3]	69.21	20.01	40.95	37.08
Ours	**70.69**	**20.06**	**43.81**	**40.12**

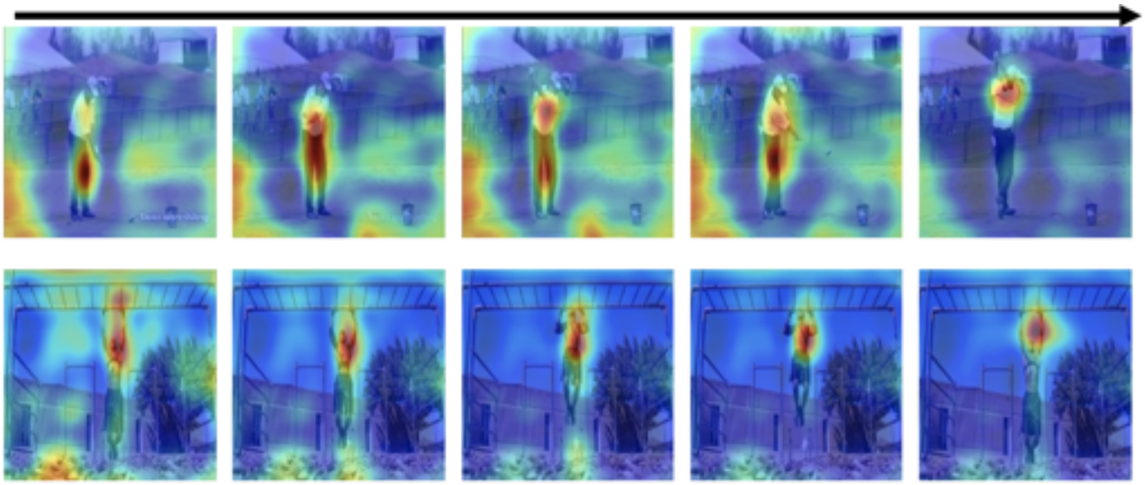
Table 3. Detailed evaluation of fine-grained action recognition performance by looking at elements within each event separately.

6. Video synchronization

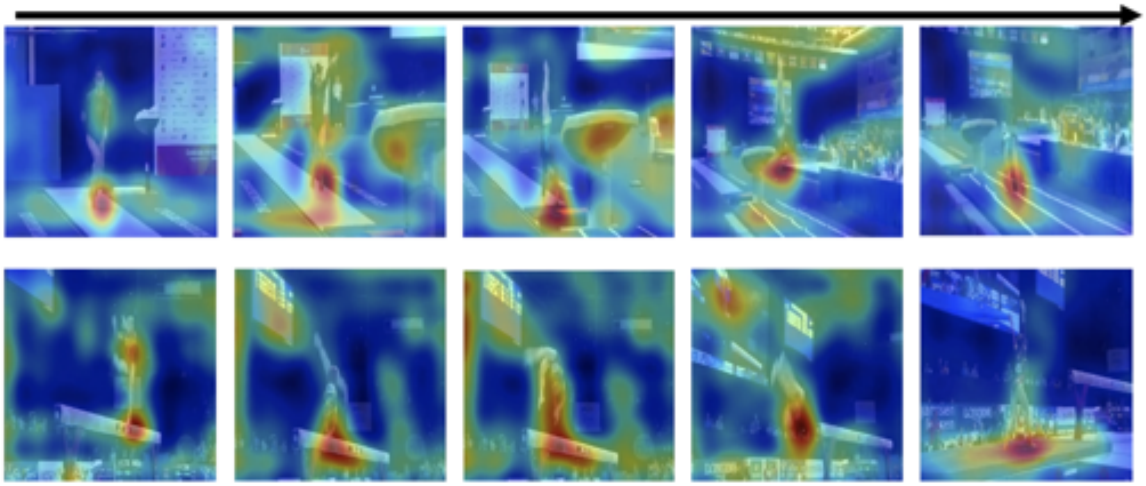
In the main manuscript, we provide results of video synchronization under the challenging setting of training a single network for all classes in PennAction, whereas [3] trained a different network for each class. For completeness, we provide additional results here where we also trained a network per class using our loss on PennAction to directly compare with [3]. The results summarized in Table 4 speak decisively in favor of our approach where we outperform all approaches with a sizeable margin.

7. Visualizing learned features

To investigate what our learned representation captures, we adapt the Class Activation Map (CAM) method [7] to visualize the learned features. In particular, we extract feature maps from the last convolutional layer of our embedding network and simply average them along the channel di-



(a) Example from PennAction



(b) Example from FineGym

Figure 2. Visualization of the features learned with the proposed alignment loss using the CAM method.

Method	Kendall's Tau
TCN [5]	73.28
SaL [4]	63.36
TCC [3]	73.53
Ours	**78.29**

Table 4. Video alignment results using Kendall's Tau metric on PennAction. *Consistent* with previous work, these results were obtained by training a separate network for each class separately.

mension. The resulting activation maps are then normalized between 0 and 1 framewise, upsampled to match the input dimensions, and superimposed on the input video frames. For PennAction, the heatmaps in Figure 2 (a) shows that our embeddings are selective to body parts most involved in performing an action. This can explained by the fact that videos in PennAction are carefully curated with relatively clean, similar actions with no repetitions. On the other hand, we can see from Figure 2 (b), that our embeddings are tuned to human contact with surfaces to learn alignments in FineGym. This is an especially desired behaviour as while there can be significant variations in the way gymnasts perform different phases of an action, they generally share some commonality in the manner that they make contact with surfaces. More generally, these visualizations suggest that the proposed loss learns to adapt and identify the most reliable cues to learn the alignments. Additional activation visualizations are provided in the supplemental video.

8. Downstream applications

Please see accompanying supplemental video for various downstream application results.

References

[1] Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, William T. Freeman, Michael Rubinstein, Michal Irani, and Tali Dekel. SpeedNet: Learning the speediness in videos. In *CVPR*, pages 9919–9928, 2020. 2, 3

[2] Chien-Yi Chang, De-An Huang, Yanan Sui, Li Fei-Fei, and Juan Carlos Niebles. D3TW: Discriminative differentiable dynamic time warping for weakly supervised action alignment and segmentation. In *CVPR*, pages 3546–3555, 2019. 2, 3

[3] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal cycle-consistency learning. In *CVPR*, pages 1801–1810, 2019. 2, 3, 4

[4] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In *ECCV*, pages 527–544, 2016. 2, 3, 4

[5] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. In *ICRA*, pages 1134–1141, 2018. 2, 3, 4

[6] Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. FineGym: A hierarchical video dataset for fine-grained action understanding. In *CVPR*, pages 2613–2622, 2020. 3

[7] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929, 2016. 3