Video Test-Time Adaptation for Action Recognition Supplementary

Wei Lin[†*1,2] Muhammad Jehanzeb Mirza[*1,3] Mateusz Kozinski[1] Horst Possegger[1]
Hilde Kuehne[4,5] Horst Bischof[1,3]

[1]Institute for Computer Graphics and Vision, Graz University of Technology, Austria
[2]Christian Doppler Laboratory for Semantic 3D Computer Vision
[3]Christian Doppler Laboratory for Embedded Machine Learning
[4]Goethe University Frankfurt, Germany
[5]MIT-IBM Watson AI Lab

For further insights into our ViTTA, we provide implementation details of TANet and Video Swin Transformer (Sec. 1), introduction of the corruptions used in evaluation (Sec.2) and additional experimental results (Sec. 3).

In Sec. 3.1, we compare the computation efficiency of ViTTA with baseline methods. In Sec. 3.2. we report adaptation performance on time-correlated data. In Sec. 3.3, we report the performance of adaptation to only a part of the test set to validation the adaptation efficiency of ViTTA. In Sec. 3.4, we compare the performance of adaptation to a test set using train statistics from different datasets. In Sec. 3.5, we perform an ablation study on the choice of momentum for moving average. Furthermore, we report standard deviation of evaluations with random distribution shift in Sec. 3.6, and report the results of all corruptions for the single distribution shift evaluation in Sec. 3.7.

1. Implementation Details

1.1. TANet

Training. The pre-trained `TAM-R50` models of TANet [8] based on ResNet50 [3] for K400 and SSv2 are from the model zoo of TANet[1]. We use the models that are trained with 8 frames sampled from each video for both datasets.

We train `TAM-R50` for UCF101 with weights initialized from the model pre-trained on K400. Following [8], we resize the shorter size of the frame to 256, and apply the multi-scale cropping and random horizontal flipping as data augmentation. The cropped frames are resized to 224×224 for training. We use a batch size of 24 and train for 50 epochs using SGD with momentum of 0.9, weight decay of $1e-4$, and initial learning rate of $1e-3$. We also the uniform sampling strategy and sample 16 frames per video.

Inference. During inference, we resize the shorter side of the frame to 256. For efficient implementation, we take the center crop of size 224×224 in the spatial dimensions. For frame sampling, we apply uniform sampling strategy to sample one clip from each test video. The clip length is 8 for K400 and SSv2, and 16 for UCF101.

1.2. Video Swin Transformer

Training. We use the pre-trained models of the Video Swin Transformer [7] from the model zoo of Video Swin Transformer[2]. The backbone is `Swin-B` from Swin Transformer [6]. For K400, we use the `Swin-B` model that is trained with weight initialization from pre-training on ImageNet-1K [1]. For SSv2, the model is trained with weight initialization from pre-training on K400.

For UCF101, we train `Swin-B` model with the weight initialization from K400. Following [7], the model is trained using an AdamW [5] optimizer with a cosine decay learning rate scheduler. The initial learning rate is set to $3e-5$. We train with batch size of 8 for 30 epochs. A clip of 32 frames (size 224×224) with a temporal stride of 2 is sampled from each video.

Inference. During inference, we take the center crop of size 224×224. For frame sampling, we apply uniform sampling strategy to sample one clip of 16 frames from each test video.

* Equally contributing authors.
† Correspondence: `wei.lin@icg.tugraz.at`

[1] https://github.com/liu-zhy/temporal-adaptive-module

[2] https://github.com/SwinTransformer/Video-Swin-Transformer

2. Corruption Types

We evaluate on 12 types of corruptions in video acquisition and video processing, proposed in [10, 13] that benchmark the robustness of spatio-temporal models. We prepare the corrupted videos for UCF101 [11], Something-something v2 (SSv2) [2] and Kinetics 400 (K400) [4] with implementations from these two works. We give short descriptions of the 12 corruptions in the following.

Gaussian Noise could appear due to low-lighting conditions or sensor limitation during video acquisition. *Pepper Noise* and *Salt noise* simulates disturbance in image signal with densely occurring black pixels and white pixels. *Shot noise* captures the electronic noise caused by the discrete nature of light. In the implementation, it is approximated with a Poisson distribution. *Zoom Blur* occurs when the camera moves towards an object rapidly. *Impulse noise* simulates corruptions caused by the defect of camera sensor. *Defocus Blur* happens when the camera is out of focus. *Motion Blur* is caused by destabilizing motion of camera. *Jpeg* is a lossy image compression format which introduces compression artifacts. *Contrast* corruption is common in video acquisition due to changing contradiction in luminance and color of the scene that is captured. *Rain* corruption simulates a rainy weather condition during video acquisition. *H265 ABR Compression* corruption simulates compression artifacts when video compression is performed using the popular H.265 codec with an average bit rate.

We visualize some samples of action videos with corruptions from UCF101 in Fig. 5.

3. Additional Results

3.1. Computational Efficiency

We report Memory (MB), latency (s) and performance of TANet on UCF101 (Nvidia A6000 + AMD EPYC 7413) in Tab. 8. ViTTA offers an excellent performance/computation tradeoff.

3.2. Evaluation on Time-Correlated Data

In Tab. 9, we run experiments in adaptation to sequences of videos in the order given in the original validation lists, where videos of the same class are listed together. This makes them highly correlated. On UCF, even clips of the same scene/video are listed together. ViTTA still outperforms all baselines by a significant margin.

3.3. Adapting To Only A Part of Test Videos

During online adaptation, we estimate the test set statistics by continuous exponential moving average of statistics computed on a stream of test videos, and use these estimated statistics for alignment. Here we study the performance of our ViTTA when adapting to only the first portion of all test videos in the sequence.

method	batch	views	block	mem.	latency	acc
ViTTA	1	2	3,4	6500	0.159	**78.20**
ViTTA	1	2	4	6236	0.146	77.83
ViTTA	1	1	3,4	4601	0.110	75.57
ViTTA	1	1	4	4469	0.098	75.43
TENT	8	-	-	16482	0.456	**72.92**
NORM	8	-	-	3410	0.262	65.77
SHOT	8	-	-	16483	0.533	65.54
DUA	1	-	-	4785	0.830	55.34
T3A	8	-	-	3410	0.151	54.17
T3A	1	-	-	2702	0.042	54.17
NORM	1	-	-	2702	0.056	51.59
TENT	1	-	-	4339	0.081	51.58
SHOT	1	-	-	4339	0.089	51.20

Table 8. Computational efficiency.

model	batch	TANet			Swin		
dataset		UCF	SSv2	K400	UCF	SSv2	K400
source	-	51.35	24.31	37.16	78.48	42.18	47.17
NORM	1	17.14	3.77	8.72	-	-	-
DUA	1	24.83	6.45	13.23	-	-	-
TENT	1	17.14	3.79	8.77	79.35	33.22	45.64
SHOT	1	17.30	3.71	5.79	68.96	20.58	31.04
T3A	1	53.19	24.14	37.72	80.18	41.64	48.20
ViTTA	1	**71.40**	**31.45**	**45.04**	**83.48**	**42.98**	**48.97**
NORM	8	52.87	9.83	27.52	-	-	-
TENT	8	53.91	10.24	27.43	81.11	42.81	47.85
SHOT	8	52.90	9.85	26.27	77.83	39.52	46.05
T3A	8	53.47	24.17	37.75	80.31	41.69	48.22
ViTTA	8	**74.19**	**33.31**	**46.70**	**83.92**	**48.25**	**53.97**

Table 9. Mean Top-1 Classification Accuracy (%) over all corruption types on UCF101, SSv2 and K400 datasets. Adaptation on time correlated videos.

We denote the number of videos in the test set as N_T and the percentage of videos that we adapt to as $p \cdot 100\%$. The sequence of test videos is $\mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_{N_T}$, where i is the video index. We perform online adaptation only on the first $p \cdot 100\%$ of test videos in the sequence. Consequently, $N_{adapt} = p \cdot N_T$ denotes the absolute number of videos used for adaptation.

Specifically, for videos with index $i \leq N_{adapt}$, we test right after adapting to them. For videos with index $i > N_{adapt}$, we test directly without any further adaptation. We vary p from 0% (*Source-Only*, $N_{adapt} = 0$) to 100% (full adaptation, $N_{adapt} = N_T$), and report the performance of TANet and Video Swin Transformer on three datasets in Fig. 4.

We notice that on all datasets, when N_{adapt} is below 1000, the adaptation performance improves fast with

Test Set	Source-Only	Train Statistics		
		UCF101	SSv2	K400
UCF101	51.35	**78.20**	65.64	74.53
SSv2	24.31	30.34	**37.97**	29.41
K400	37.16	46.94	38.46	**48.69**

Table 10. Mean Top-1 Classification Accuracy (%) over all corruptions of adaptation to a test set with train statistics from different datasets. We perform adaptation on TANet in combination of different train and test sets across UCF101, SSv2 and K400.

N_{adapt} increasing. After N_{adapt} reaches 1000, increasing N_{adapt} leads to stable and saturated adaptation performance. Adapting only to the first 1000 videos leads to performance drop of less than 3% in comparison to the full adaptation case. Note that $N_{adapt} = 1000$ corresponds to p of 26.4% on the small UCF101 and only 4% $\sim$ 5% on the large-scale SSv2 and K400. This indicates the adaptation efficiency of our ViTTA.

3.4. Train Statistics From A Different Dataset

For feature distribution alignment, our ViTTA requires feature means and variances computed on the training data. When training data is no longer available, these statistics could be computed on other data with similar distribution. We compare the performance of adaptation to a test set using train statistics from different datasets in Table 10. Intuitively, using the training statistics from the same dataset leads to the best adaptation performance. Adaptation with train statistics on clean data leads to performance boost upon *Source-Only*, even if the train statistics are from a different dataset. As UCF101 and K400 are both generic action datasets in user video style and have similar statistics, using the train statistics from one to adapt to the test set of the other leads to slight performance drop. SSv2 is a egocentric motion-based dataset and has significantly different distribution than UCF101 and K400. Adaptation across SSv2 and UCF101 or K400 leads to less satisfying performance improvement.

3.5. Momentum of Moving Average

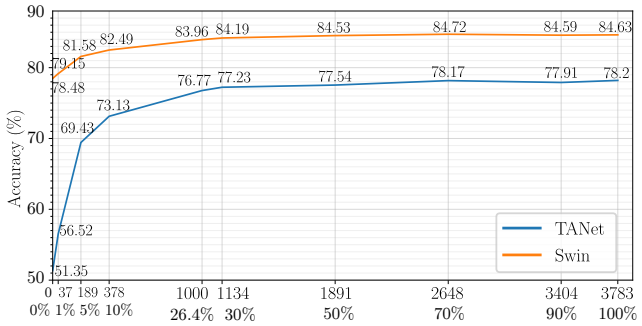
We approximate the test set statistics by exponential moving averages of statistics computed on consecutive test videos. Recall Eqs. (4) and (5) of the main manuscript:

$$\mu_l{}^{(i)}(\theta) = \alpha \cdot \mu_l(\mathbf{x}_i;\theta) + (1-\alpha)\cdot \mu_l{}^{(i-1)}(\theta), \quad (1)$$

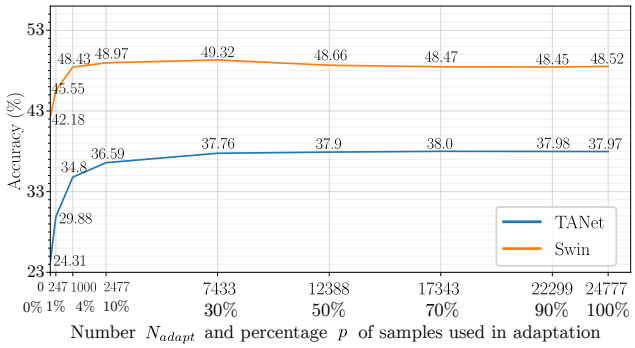
$$\sigma_l^{2\,(i)}(\theta) = \alpha \cdot \sigma_l^2(\mathbf{x}_i;\theta) + (1-\alpha)\cdot \sigma_l^{2\,(i-1)}(\theta). \quad (2)$$

Here we have the hyperparameter α and, consequently, $1-\alpha$ is the momentum (erratum for line 408 in the main paper: α is a hyperparameter, not the momentum). As we evaluate with the small batch size of 1, a large momentum will result in a steady update of statistics. We set α to 0.1, which corresponds to a common choice of large momentum of 0.9. We study other choices of the momentum value and report the performance in Table 11.

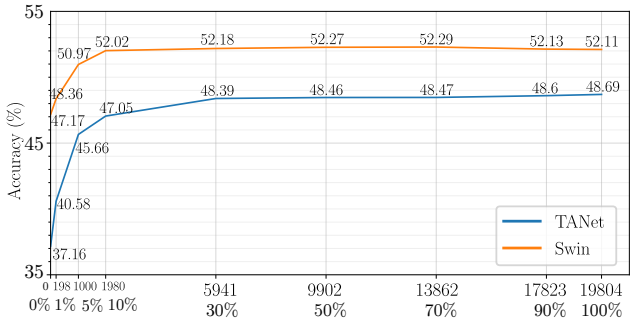
When the momentum is too large, there is a slow convergence of statistics. The moving average is dominated by statistics of features in the past, which are outdated as the model evolves [12]. As shown in Table 11, this slow



(a) UCF101



(b) SSv2



(c) K400

Figure 4. Mean Top-1 Classification Accuracy (%) over all corruptions of adaptation to only the first N_{adapt} test videos (p%) in the sequence. For videos with index $i \leq N_{adapt}$, we test right after adapting to it. For videos with index $i > N_{adapt}$, we test directly without any further adaptation. We vary p from 0% (*Source-Only*, $N_{adapt} = 0$) to 100% (full adaptation, $N_{adapt} = N_T$), and report the performance of TANet and Video Swin Transformer on three datasets in (a) UCF101 (b) SSv2 and (c) K400.

Momentum $1-\alpha$	0.99	0.95	0.9	0.85	0.8
UCF101	56.94	64.70	**78.20**	77.53	76.80
SSv2	34.81	37.53	**37.97**	37.53	36.13
K400	45.68	48.15	**48.69**	47.47	46.13

Table 11. Mean Top-1 Classification Accuracy (%) over all corruptions with different momentum values ($1-\alpha$) for moving average. We report adaptation performance of TANet on three action datasets.

convergence has a big impact on a small video dataset like UCF101 (3783 validation videos), resulting in a drastic performance drop. In the meanwhile, on large video datasets like SSv2 (24K validation videos) and K400 (20K validation videos), the performance drop due to slow convergence is less severe. When the momentum is too small, the estimated statistics are dominated by the recent samples and do not represent the whole population, leading to performance degradation.

3.6. Evaluation with Random Distribution Shift

In Sec.4.3.2 in the main paper, we evaluated in the scenario where we assume that each video received has a random type of distribution shift. For each of the videos in a sequence, we randomly selected one of the 13 distribution shifts (12 corruption types, plus the case of *no corruption*). We ran the experiments 3 times while shuffling the order of the videos. In addition to the average results (reported in Table 3 in the main paper), here we also report the standard deviation in Table 12. Shuffling the video and corruption order leads to only minor variation in the performance. Considering the standard deviation, our ViTTA still constantly outperforms baseline methods in all settings.

3.7. Evaluation with Single Distribution Shift: Results of All Corruptions

In Table 1 of the main paper, for evaluation with single distribution shift, we report the average performance on all corruption types. Here we report the detailed results of all the 12 corruption types for TANet (Table 13) and Video Swin Transformer (Table 14) on the three datasets.

Model	TANet			Swin		
Dataset	UCF101	SSv2	K400	UCF101	SSv2	K400
source-only	55.41	26.93	39.98	79.62	44.31	49.48
NORM	33.32± 0.09	10.63± 0.09	27.22± 0.12	-	-	-
DUA	41.94± 0.40	12.53± 0.19	30.89± 0.18	-	-	-
TENT	31.32± 0.06	10.68± 0.11	27.25± 0.09	81.35± 0.25	44.58± 1.96	49.46± 0.08
SHOT	32.91± 0.16	9.02± 0.12	22.89± 0.12	78.66± 0.21	32.93± 0.40	42.37± 1.77
T3A	53.32± 0.21	24.74± 0.12	38.86± 0.14	81.02± 0.14	43.54± 0.09	49.59± 0.08
ViTTA	**66.94**± 0.31	**32.87**± 0.08	**42.76**± 0.07	**83.11**± 0.35	**46.32**± 0.11	**49.67**± 0.01

Table 12. Top-1 Classification Accuracy (%) with random distribution shifts. For each video in the test set, we randomly select 1 out of 13 distribution shifts (12 corruption types + *original test set*). We run these experiments 3 times while shuffling the order of the videos and report the mean and standard deviation.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 1

[2] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, pages 5842–5850, 2017. 2

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1

[4] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[6] 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. 1

[7] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *CVPR*, pages 3202–3211, 2022. 1

[8] Zhaoyang Liu, Limin Wang, Wayne Wu, Chen Qian, and Tong Lu. Tam: Temporal adaptive module for video recognition. In *ICCV*, pages 13708–13718, 2021. 1

[9] M Jehanzeb Mirza, Jakub Micorek, Horst Possegger, and Horst Bischof. The norm must go on: Dynamic unsupervised domain adaptation by normalization. In *CVPR*, pages 14765–14775, 2022. 5

[10] Madeline C. Schiappa, Naman Biyani, Shruti Vyas, Hamid Palangi, Vibhav Vineet, and Yogesh Rawat. Large-scale robustness analysis of video action recognition models. *CoRR*, abs/2207.01398, 2022. 2

[11] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 2

TANet UCF101													
corruptions	gauss	pepper	salt	shot	zoom	impulse	defocus	motion	jpeg	contrast	rain	h265.abr	avg
Source-Only	17.92	23.66	7.85	72.48	76.04	17.16	37.51	54.51	83.40	62.68	81.44	81.58	51.35
NORM	45.23	42.43	27.91	86.25	84.43	46.31	54.32	64.19	89.19	75.26	90.43	83.27	65.77
DUA	36.61	33.97	22.39	80.25	77.13	36.72	44.89	55.67	85.12	30.58	82.66	78.14	55.34
TENT	58.34	53.34	35.77	89.61	**87.68**	59.08	64.92	75.59	90.99	82.53	92.12	**85.09**	72.92
SHOT	46.10	43.33	29.50	85.51	82.95	47.53	53.77	63.37	88.69	73.30	89.82	82.66	65.54
T3A	19.35	26.57	8.83	77.19	79.38	18.64	40.68	58.61	86.12	67.22	84.0	83.45	54.17
ViTTA	**71.37**	**64.55**	**45.84**	**91.44**	**87.68**	**71.90**	**70.76**	**80.32**	**91.70**	**86.78**	**93.07**	84.56	**78.33**
TANet SSv2													
corruptions	gauss	pepper	salt	shot	zoom	impulse	defocus	motion	jpeg	contrast	rain	h265.abr	avg
Source-Only	14.29	16.36	7.83	34.73	45.72	14.39	27.83	26.92	25.21	23.24	41.10	14.07	24.31
NORM	15.99	16.95	12.87	40.14	45.30	17.06	34.53	32.47	41.77	23.05	42.18	15.89	28.18
DUA	7.45	8.95	6.45	24.0	29.92	8.0	20.71	19.48	27.62	6.38	25.0	10.48	16.20
TENT	20.52	20.74	15.45	44.01	47.11	21.34	38.10	35.87	45.26	27.42	45.70	17.36	31.57
SHOT	17.67	18.16	13.85	37.49	43.25	17.85	32.59	31.71	39.37	22.28	39.0	16.41	27.47
T3A	15.23	16.49	7.69	33.89	44.26	15.48	27.69	27.17	29.08	23.34	39.71	13.37	24.45
ViTTA	**34.69**	**26.69**	**16.15**	**48.58**	**49.26**	**35.54**	**44.05**	**40.93**	**48.37**	**42.05**	**50.95**	**19.57**	**38.07**
TANet K400													
corruptions	gauss	pepper	salt	shot	zoom	impulse	defocus	motion	jpeg	contrast	rain	h265.abr	avg
Source-Only	28.52	26.14	15.97	57.89	39.66	29.51	47.01	52.79	60.60	25.69	48.23	13.91	37.16
NORM	32.55	30.09	21.67	58.22	43.44	33.25	43.99	50.74	60.28	28.19	57.14	12.29	39.32
DUA	24.26	22.25	14.68	50.74	35.10	25.04	38.15	43.41	53.29	18.32	48.04	9.31	31.88
TENT	34.16	31.54	22.64	60.29	46.03	34.88	45.54	52.50	61.87	32.50	59.21	12.44	41.13
SHOT	32.24	29.91	21.76	55.72	40.26	32.95	42.77	47.93	56.44	23.59	51.05	**14.55**	37.43
T3A	29.38	27.19	16.24	58.02	40.43	30.33	46.50	53.05	60.39	26.50	49.16	13.93	37.59
ViTTA	**47.95**	**43.90**	**34.73**	**64.07**	**50.08**	**49.23**	**52.80**	**57.69**	**63.89**	**46.77**	**61.88**	14.24	**48.94**

Table 13. Top-1 Classification Accuracy (%) for all corruptions for TANet on three datasets. *DUA* is evaluated with batch size of 1, following the setting in the original work [9]. All the other methods are evaluated with batch size of 8. Highest accuracy is shown in bold, while second best is underlined.

[12] Yuxin Wu and Justin Johnson. Rethinking" batch" in batchnorm. *arXiv preprint arXiv:2105.07576*, 2021. 3

[13] Chenyu Yi, Siyuan Yang, Haoliang Li, Yap-Peng Tan, and Alex C. Kot. Benchmarking the robustness of spatial-temporal models against corruptions. In *NeurIPS*, 2021. 2

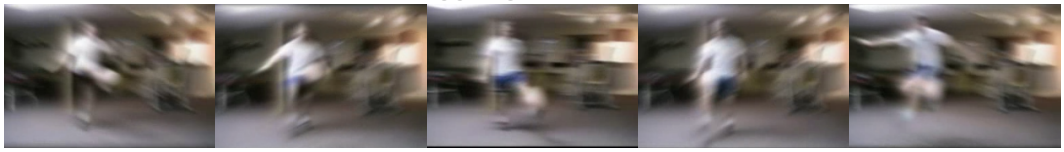
Swin UCF101													
corruptions	gauss	pepper	salt	shot	zoom	impulse	defocus	motion	jpeg	contrast	rain	h265.abr	avg
Source-Only	73.01	69.57	53.32	90.38	82.66	73.83	78.19	79.21	84.03	81.58	90.69	85.33	78.48
TENT	77.66	75.47	60.90	91.75	85.96	78.75	**81.29**	82.86	88.13	85.81	92.20	87.42	82.35
SHOT	72.98	69.39	53.21	90.35	82.58	73.80	78.17	79.19	84.03	81.42	90.64	85.33	78.42
T3A	75.89	72.56	57.34	91.22	83.87	76.79	80.02	81.28	86.52	84.17	91.59	86.89	80.68
ViTTA	**81.60**	**81.52**	**70.90**	**92.65**	**86.07**	**82.47**	79.41	**84.55**	**90.40**	**86.89**	**93.21**	**87.23**	**84.74**
Swin SSv2													
corruptions	gauss	pepper	salt	shot	zoom	impulse	defocus	motion	jpeg	contrast	rain	h265.abr	avg
Source-Only	39.79	33.22	22.26	56.47	56.01	40.07	49.74	43.36	51.08	45.81	48.96	19.41	42.18
TENT	35.83	36.61	11.92	**58.13**	**57.59**	40.52	53.15	46.58	53.93	51.57	52.67	15.57	42.84
SHOT	40.42	33.82	24.07	56.31	56.01	40.67	49.59	43.79	51.31	45.53	49.46	19.66	42.55
T3A	39.54	33.16	22.06	56.02	55.97	39.67	50.15	44.67	52.56	45.63	49.57	19.97	42.41
ViTTA	**48.0**	**43.10**	**40.09**	58.04	57.28	**48.87**	**54.20**	**50.88**	**58.18**	**53.17**	**59.94**	**24.22**	**49.66**
Swin K400													
corruptions	gauss	pepper	salt	shot	zoom	impulse	defocus	motion	jpeg	contrast	rain	h265.abr	avg
Source-Only	42.91	35.52	28.64	65.37	42.15	44.43	57.56	58.92	66.09	45.84	57.12	21.56	47.17
TENT	43.94	35.59	26.38	65.90	43.67	45.76	**58.60**	59.65	66.64	50.04	56.94	20.94	47.84
SHOT	43.73	36.15	29.38	66.26	43.02	45.27	58.42	59.86	67.0	46.08	58.20	22.39	47.98
T3A	43.55	36.49	29.25	65.73	43.93	45.04	58.55	59.78	66.96	47.43	59.17	**22.51**	48.20
ViTTA	**54.45**	**51.17**	**44.15**	**67.92**	**51.99**	**55.92**	57.57	**60.97**	**67.68**	**54.47**	**65.97**	22.34	**54.55**

Table 14. Top-1 Classification Accuracy (%) for all corruptions for Video Swin Transformer on three datasets. *NORM* and *DUA* cannot be evaluated on transformer without batch norm layers. All the methods are evaluated with batch size of 8. Highest accuracy is shown in bold, while second best is underlined.

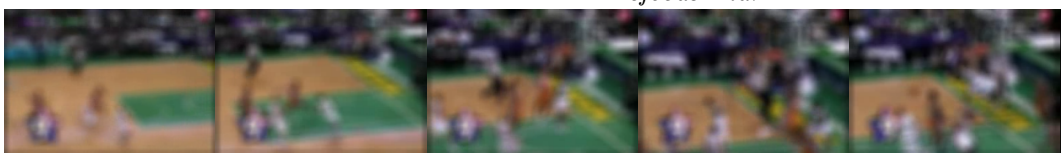
GolfSwing with *Impulse Noise*

Skiing with *Contrast*

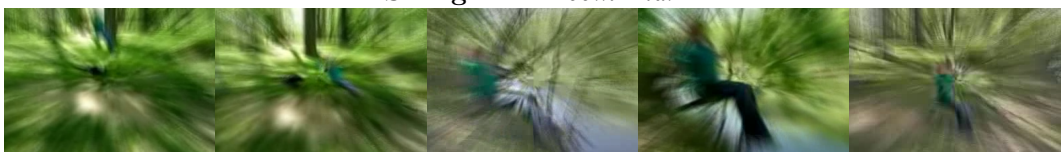
SoccerJuggling with *Motion Blur*



BasketBallDunk with *Defocus Blur*



Swing with *Zoom Blur*



HorseRiding with *Jpeg compression*

Kayaking with *Rain*

Punch with *H265.ABR compression*

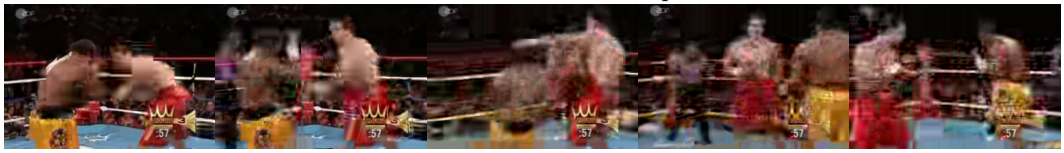


Figure 5. Samples of action videos with corruptions. Best viewed on screen.