Supplementary Material: Unsupervised Video Domain Adaptation for Action Recognition: A Disentanglement Perspective

Anonymous Author(s)
Affiliation
Address
email

In this file, we supplement the following materials to support the findings and observations in the main body of this paper:

- Section A elaborates on additional implementation details to ensure reproduction.
- Section B provides additional quantitative and qualitative Results for better comparisons.
- Section C discusses the broader impact and some limitations of this work.
- Section D acknowledges the public resources used during the course of this work.

A Implementation Details

In this section, we provide more implementation details including the I3D feature extraction procedure, the concrete model architecture, and the hyperparameter selection in our proposed *TranSVAE* framework. We also provide detailed instructions for our anonymous live demo.

A.1 I3D Feature Extractions

We extract the I3D RGB features following the routine described in SAVA [1]. Given a video sequence, 16 frames along clips are sampled by sliding a temporal window with a temporal stride of 1. Specifically, for each frame in the video, the temporal window consists of its previous 7 frames and the following 8 frames. Zero padding is used for the beginning and the end of the video. We then feed the sliding windows to the I3D backbone to extract features, which results in a 1,024-dimensional feature vector for each frame of the video.

A.2 Model Architecture

We now provide the detailed model architecture of our *TranSVAE*. In Fig. A, we show the model with the *image* as the input, where the encoder and decoder are more complex convolutional and deconvolutional layers. For the model with the RGB *features* as the input, we can simply replace the encoder and decoder with fully-connected linear layers. Note that the dimensionality of all the modules shown above is uniformly applied in all the experiments.

A.3 Hyperparameter Selection

There are several hyperparameters used in *TranSVAE*, including the balancing weights λ_1, λ_2, λ_3, λ_4, the number of the video frames T, and the confidence threshold η for generating target pseudo-labels. For λ_1 to λ_4, we select from the value set $\{1e^{-3}, 1e^{-2}, 1e^{-1}, 0.5, 1, 5, 10, 50, 100, 1000\}$. For T, we select from $\{5, 6, 7, 8, 9, 10\}$. For η, we set its value range from 0.9 to 1.0 with a step of 0.01.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

Table A: Summary of the best-possible hyperparameter values for each UDA task in our experiments after an extensive grid search.

Task	Backbone	T	η	λ_1	λ_2	λ_3	λ_4
$\mathbf{U} \rightarrow \mathbf{H}$	I3D	8	0.93	50	1	1	1
$\mathbf{H} \rightarrow \mathbf{U}$	I3D	9	0.96	0.5	0.1	10	1
$\mathbf{J}_{\mathcal{S}} \rightarrow \mathbf{J}_{\mathcal{T}}$	I3D	6	0.95	0.001	10	100	10
$\mathbf{D}_1 \rightarrow \mathbf{D}_2$	I3D	9	0.96	50	10	5	100
$\mathbf{D}_1 \rightarrow \mathbf{D}_3$	I3D	10	1	1	0.5	0.1	100
$\mathbf{D}_2 \rightarrow \mathbf{D}_1$	I3D	8	0.93	100	10	5	100
$\mathbf{D}_2 \rightarrow \mathbf{D}_3$	I3D	10	0.91	0.5	10	50	100
$\mathbf{D}_3 \rightarrow \mathbf{D}_1$	I3D	8	0.91	100	10	50	100
$\mathbf{D}_3 \rightarrow \mathbf{D}_2$	I3D	9	0.91	1000	1	0.1	100

We set a high-value range of η to ensure a high confidence score on the correctness of the target pseudo-labels. Following the common protocol used in video-based UDA, we conduct an extensive grid search regarding these hyperparameters on the validation set of each transfer task. Tab. A summarizes the exact used values of these hyperparameters for the UCF-HMDB [2, 3], Jester [4], and Epic-Kitchens [5] UDA benchmarks. For the Sprites [6] dataset, we do not do a hyperparameter search as the data is quite simple. We simply set T as 8, which is the original length of the video sequence. The confidence threshold is set to be 0.99, and λ_1 to λ_4 are all set to be 1.

A.4 Demo Instruction

As mentioned in the main body, we include an anonymous live demo for our *TranSVAE* framework. This demo can be accessed via: `https://huggingface.co/spaces/anonymous-demo/Anonymous-TranSVAE-Demo`. Here we include the detailed instructions for playing with this demo.

This demo is built upon Hugging Face Spaces[1], which provides concise and easy-to-use live demo interfaces. Our demo consists of one input interface and one output interface as shown in Fig. B. Specifically, the appearances of the Sprites avatars are fully controlled by four attributes, *i.e.*, body, hair color, top wear, and bottom wear. We construct two domains, $\mathbf{P}_1$ and $\mathbf{P}_2$. $\mathbf{P}_1$ uses the "Human" body while $\mathbf{P}_1$ uses the "Alien" body. The attribute pools of "Human" and "Alien" are non-overlapping across domains, resulting in completely heterogeneous $\mathbf{P}_1$ and $\mathbf{P}_2$. Each video sequence contains 8 frames in total.

For conducting domain disentanglement and transfer with *TranSVAE*, users are free to choose the action and the appearance of the avatars on the left-hand side of the interface. Next, simply click the "Submit" button and the adaptation results will display on the right-hand side of the interface in a few seconds. The outputs include:

- The **1st** column: The original input of the "Human" and "Alien" avatars, *i.e.*, $\{\mathbf{x}_1^{\mathbf{P}_1}, ..., \mathbf{x}_8^{\mathbf{P}_1}\}$ and $\{\mathbf{x}_1^{\mathbf{P}_2}, ..., \mathbf{x}_8^{\mathbf{P}_2}\}$;
- The **2nd** column: The reconstructed "Human" and "Alien" avatars $\{\widetilde{\mathbf{x}}_1^{\mathbf{P}_1}, ..., \widetilde{\mathbf{x}}_8^{\mathbf{P}_1}\}$ and $\{\widetilde{\mathbf{x}}_1^{\mathbf{P}_2}, ..., \widetilde{\mathbf{x}}_8^{\mathbf{P}_2}\}$;
- The **3rd** column: The reconstructed "Human" and "Alien" avatars using only $\{\mathbf{z}_1^{\mathcal{D}}, ..., \mathbf{z}_8^{\mathcal{D}}\}$, $\mathcal{D} \in \{\mathbf{P}_1, \mathbf{P}_2\}$, which are domain-invariant;
- The **4th** column: The reconstructed "Human" and "Alien" avatars by exchanging $\mathbf{z}_d^{\mathcal{D}}$, which results in two sequences with the same actions but exchanged appearance, *i.e.*, domain disentanglement and transfer.

B Additional Experimental Results

In this section, we provide additional quantitative and qualitative results for our *TranSVAE* framework.

[1] `https://huggingface.co`.

```
TranSVAE(
  (encoder): encoder(
    (c1): dcgan_conv(
      (main): Sequential(
        (0): Conv2d(3, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
        (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (2): LeakyReLU(negative_slope=0.2, inplace=True) ) )
    (c2): dcgan_conv( (main): Sequential(
        (0): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
        (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (2): LeakyReLU(negative_slope=0.2, inplace=True) ) )
    (c3): dcgan_conv( (main): Sequential(
        (0): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
        (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (2): LeakyReLU(negative_slope=0.2, inplace=True) ) )
    (c4): dcgan_conv( (main): Sequential(
        (0): Conv2d(256, 512, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
        (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (2): LeakyReLU(negative_slope=0.2, inplace=True) ) )
    (c5): Sequential(
        (0): Conv2d(512, 1024, kernel_size=(4, 4), stride=(1, 1))
        (1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (2): Tanh())
  )
  (decoder): decoder_woSkip
    (upc1): Sequential(
      (0): ConvTranspose2d(1024, 512, kernel_size=(4, 4), stride=(1, 1))
      (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): LeakyReLU(negative_slope=0.2, inplace=True) )
    (upc2): dcgan_upconv( (main): Sequential(
      (0): ConvTranspose2d(512, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
      (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): LeakyReLU(negative_slope=0.2, inplace=True) ) )
    (upc3): dcgan_upconv( (main): Sequential(
      (0): ConvTranspose2d(256, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
      (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): LeakyReLU(negative_slope=0.2, inplace=True) ) )
    (upc4): dcgan_upconv( (main): Sequential(
      (0): ConvTranspose2d(128, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
      (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (2): LeakyReLU(negative_slope=0.2, inplace=True) ) )
    (upc5): Sequential(
      (0): ConvTranspose2d(64, 3, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
      (1): Sigmoid() )
  )
  (relu): LeakyReLU(negative_slope=0.1)
  (dropout_f): Dropout(p=0.5, inplace=False)
  (dropout_v): Dropout(p=0.5, inplace=False)
  (z_prior_lstm_ly1): LSTMCell(512, 512)
  (z_prior_lstm_ly2): LSTMCell(512, 512)
  (z_prior_mean): Linear(in_features=512, out_features=512, bias=True)
  (z_prior_logvar): Linear(in_features=512, out_features=512, bias=True)
  (z_lstm): LSTM(1024, 512, batch_first=True, bidirectional=True)
  (f_mean): Linear(in_features=1024, out_features=512, bias=True)
  (f_logvar): Linear(in_features=1024, out_features=512, bias=True)
  (z_rnn): RNN(1024, 512, batch_first=True)
  (z_mean): Linear(in_features=512, out_features=512, bias=True)
  (z_logvar): Linear(in_features=512, out_features=512, bias=True)
  (fc_feature_domain_frame): Linear(in_features=512, out_features=512, bias=True)
  (fc_classifier_domain_frame): Linear(in_features=512, out_features=2, bias=True)
  (TRN)
  (bn_trn_S): BatchNorm1d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (bn_trn_T): BatchNorm1d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc_feature_domain_video): Linear(in_features=256, out_features=256, bias=True)
  (fc_classifier_domain_video): Linear(in_features=256, out_features=2, bias=True)
  (relation_domain_classifier_all): ModuleList(
    (0): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Linear(in_features=256, out_features=2, bias=True) )
    (1): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Linear(in_features=256, out_features=2, bias=True) )
    (2): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Linear(in_features=256, out_features=2, bias=True) )
    (3): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Linear(in_features=256, out_features=2, bias=True) )
    (4): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Linear(in_features=256, out_features=2, bias=True) )
    (5): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Linear(in_features=256, out_features=2, bias=True) )
    (6): Sequential(
      (0): Linear(in_features=256, out_features=256, bias=True)
      (1): ReLU()
      (2): Linear(in_features=256, out_features=2, bias=True) )
  )
  (pred_classifier_video): Linear(in_features=256, out_features=15, bias=True)
  (fc_feature_domain_latent): Linear(in_features=512, out_features=512, bias=True)
  (fc_classifier_doamin_latent): Linear(in_features=512, out_features=2, bias=True)
)
```

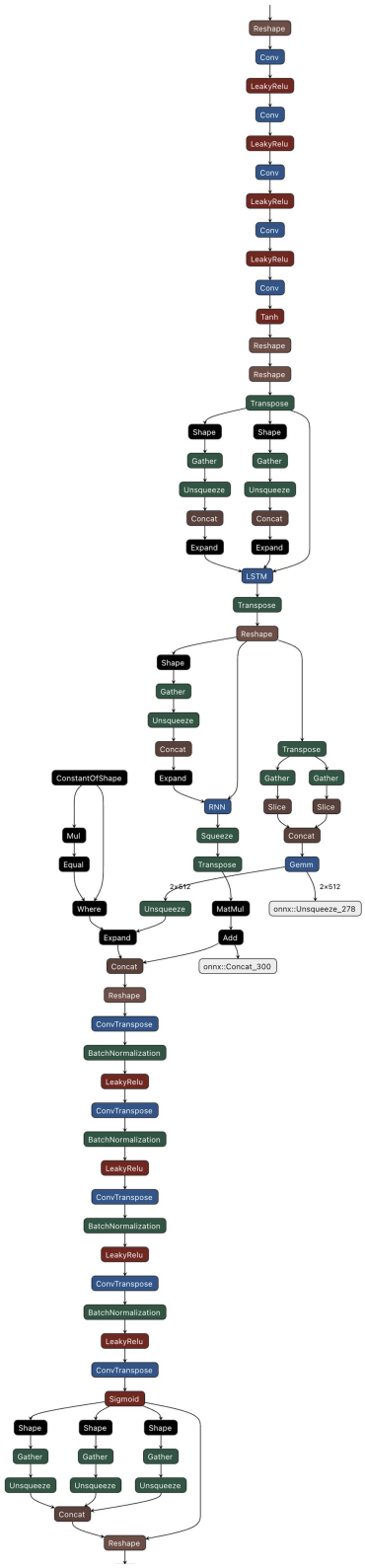

Figure A: The neural network structure (left) and a Netron graph (right) of the proposed *TranSVAE* framework. Zoom-ed in for the details.

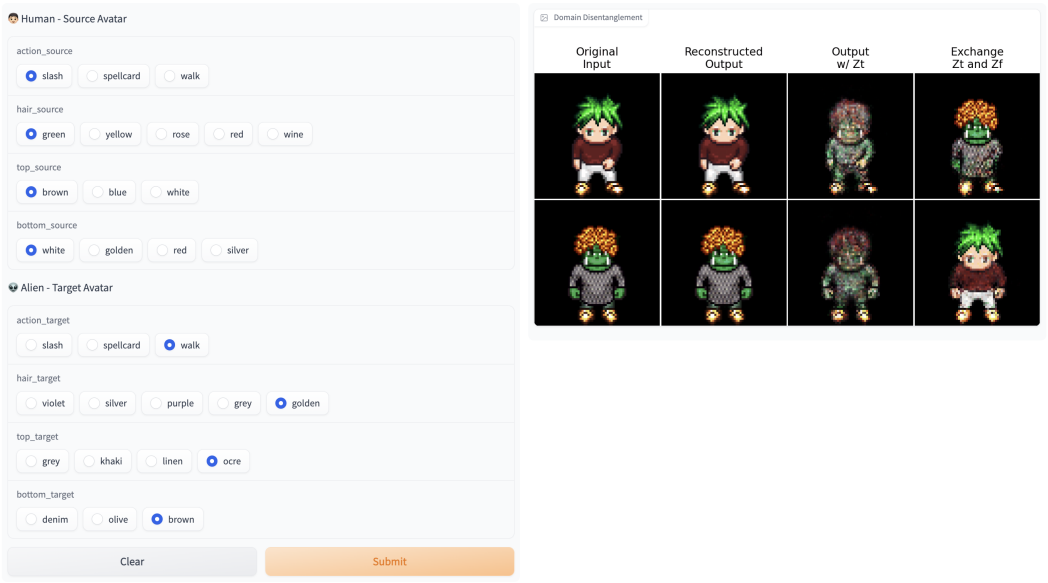

Figure B: The input (left) and output (right) interfaces of our live demo. Users are free to customize the actions and appearances of the source and target inputs (*i.e.*, the "Human" and "Alien" avatars) in the left-hand side and use them for domain disentanglement and transfer as shown in the right-hand side.

Table B: Ablation results for the frame number T.

Task	5	6	7	8	9	10	12	14	16	20
$\mathbf{U} \rightarrow \mathbf{H}$	84.44	86.11	84.17	**87.78**	84.72	85.28	83.89	84.44	82.78	85.00
$\mathbf{H} \rightarrow \mathbf{U}$	96.85	94.22	98.42	94.75	**98.95**	93.70	93.87	94.40	94.05	94.40

B.1 Additional Ablation Studies

Number of Frames T. Tab. B shows the transfer performance with the variation of the number of frames T on the UCF-HMDB dataset. As can be seen, the two tasks achieve the optimal performance with different T, specifically $T = 8$ for $\mathbf{U} \rightarrow \mathbf{H}$ and $T = 9$ for $\mathbf{H} \rightarrow \mathbf{U}$. Based on this observation, we apply a grid search on the validation set to obtain the optimal T for each task in our experiments.

Target Pseudo-Label Threshold η. We show the sensitivity analyses of the transfer performance with respect to the target pseudo label threshold η on the UCF-HMDB dataset in Tab. C. The results show that different tasks yield the best transfer result with different η, specifically $\eta = 0.93$ for $\mathbf{U} \rightarrow \mathbf{H}$ and $\eta = 0.96$ for $\mathbf{H} \rightarrow \mathbf{U}$. Thus, we also apply the grid search on the validation set to obtain the optimal η for each task.

B.2 Additional Qualitative Results

For those who cannot access our live demo, we have included more qualitative examples for domain disentanglement and transfer in Fig. C. We also provide an anonymous link[2] for GIFs demonstrating various disentanglement and reconstruction results.

In the GIFs link, we show two cases, first two columns for the first one and last two columns for the second one. Each case contain a source and a target cartoon character performing an action. For each row, we have the following remark:

- The second row shows the reconstructed results of *TranSVAE*, i.e., $\mathbf{z}_d^{\mathcal{D}}$ and $\mathbf{z}_t^{\mathcal{D}}$. As can be seen, *TranSVAE* reconstructs the image with a high quality.

[2] `https://github.com/anonymousRevievv/TranSVAE`

Table C: Ablation results for the pseudo-label threshold η.

Task	*w/o*	0.90	0.91	0.92	0.93	0.94	0.95	0.96	0.97	0.98	0.99
U → H	87.22 (−0.56)	86.94	87.22	87.50	**87.78**	86.11	86.67	86.94	86.39	86.11	86.39
H → U	94.40 (−4.55)	98.42	97.37	98.77	98.60	98.25	98.25	**98.95**	97.90	97.72	95.27

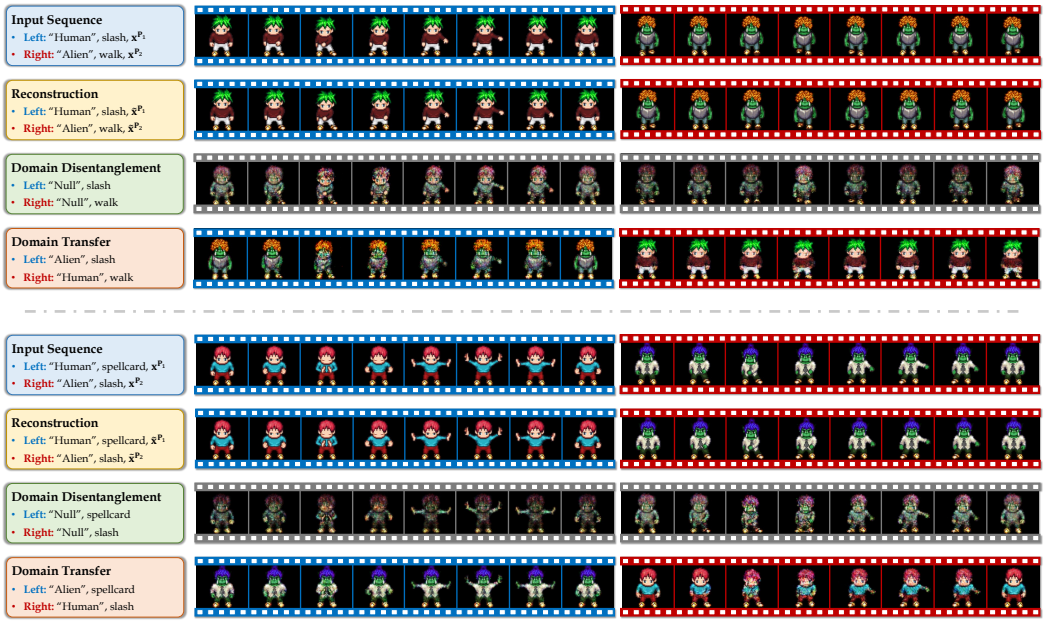

Figure C: Additional qualitative results for illustrating the domain disentanglement and transfer properties in our *TranSVAE* framework.

- The third row shows the reconstructed results only using the static latent factors, i.e., $\mathbf{z}_d^{\mathcal{D}}$ and $\mathbf{0}_t^{\mathcal{D}}$, where we replace $\mathbf{z}_t^{\mathcal{D}}$ with zero vectors. As can be seen, the reconstructed results are basically static containing the appearance of the character which is the main domain difference in the Sprite dataset. Specifically, we find they generally lack arms. This is reasonable as the target action is slashing or spelling cards with arms moving, and such dynamic information on the arm is captured by $\mathbf{z}_t^{\mathcal{D}}$.
- The fourth row shows the reconstructed results only using the dynamic latent factors, i.e., $\mathbf{0}_d^{\mathcal{D}}$ and $\mathbf{z}_t^{\mathcal{D}}$, where we replace $\mathbf{z}_d^{\mathcal{D}}$ with zero vectors. As can be seen, the reconstructed results are performing the right action but the appearance is mixed-up. This shows that $\mathbf{z}_t^{\mathcal{D}}$ are indeed domain-invaraint and contain the semantic information.
- The last row shows the reconstructed results of exchanging the dynamic latent factors between domains, i.e., $(\mathbf{0}_d^{\mathcal{S}}, \mathbf{z}_t^{\mathcal{T}})$ and $(\mathbf{0}_d^{\mathcal{T}}, \mathbf{z}_t^{\mathcal{S}})$. As can be seen, the reconstructed results are with the original appreearance but performing the transferred action. This indicates the potential of *TranSVAE* for some style-transfer tasks.

C Broader Impact

This paper provides a novel transfer method to use cross-domain video data, which effectively helps reduce the annotation efforts in related video applications. Although the main empirical evaluation is on the video action recognition task, the model structure proposed in this paper is also applicable to other video-related tasks, such as action segmentation, video semantic segmentation, *etc*. More generally, the idea of disentangling domain information sheds the light on other data modality style transfer tasks, *e.g.*, voice conversion. The negative impacts of this work are difficult to predict. However, as a deep model, our method shares some common pitfalls of the standard deep learning models, *e.g.*, demand for powerful computing resources, and vulnerability to adversarial attacks.

D Public Resources Used

We acknowledge the use of the following public resources, during the course of this work:

- UCF_{101}[3] . . . . . Unknown
- HMDB_{51}[4] . . . . . CC BY 4.0
- Jester[5] . . . . . Unknown
- Epic-Kitchens[6] . . . . . CC BY-NC 4.0
- Sprites[7] CC-BY-SA-3.0
- I3D[8] . . . . . Apache License 2.0
- TRN[9] . . . . . BSD 2-Clause License
- Netron[10] . . . . . MIT License

References

[1] Jinwoo Choi, Gaurav Sharma, Samuel Schulter, and Jia-Bin Huang. Shuffle and attend: Video domain adaptation. In *European Conference on Computer Vision*, pages 678–695. Springer, 2020.

[2] 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.

[3] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *IEEE/CVF International Conference on Computer Vision*, pages 2556–2563. IEEE, 2011.

[4] Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. The jester dataset: A large-scale video dataset of human gestures. In *IEEE/CVF International Conference on Computer Vision Workshops*, pages 1–12, 2019.

[5] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision*, pages 720–736, 2018.

[6] Yingzhen Li and Stephan Mandt. Disentangled sequential autoencoder. In *International Conference on Machine Learning*, pages 5670–5679. PMLR, 2018.

[3] `https://www.crcv.ucf.edu/data/UCF101.php`
[4] `https://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database`
[5] `https://20bn.com/datasets/jester`
[6] `https://epic-kitchens.github.io/2021`
[7] `https://github.com/YingzhenLi/Sprites`
[8] `https://github.com/piergiaj/pytorch-i3d`
[9] `https://github.com/zhoubolei/TRN-pytorch`
[10] `https://github.com/lutzroeder/netron`