Collaborating Domain-shared and Target-specific Feature Clustering for Cross-domain 3D Action Recognition

Qinying Liu and Zilei Wang

University of Science and Technology of China, Hefei, China
lydyc@mail.ustc.edu.cn, zlwang@ustc.edu.cn

In this Supplementary, we provide more details about the optimal transport problem (Sec. 1), the comparison between datasets (Sec. 2), the visualization of datasets (Sec. 3), the evaluation metrics (Sec. 4), the experimental setup (Sec. 5), additional cross-domain tasks (Sec. 6).

1 Additional Details about Optimal Transport

In the paper, we propose that generating balanced pseudo labels can be transformed to an optimal transport problem [3,1], which explores the minimum cost for assigning N data points to K clusters. Here we provide more details about the problem transformation and the solution.

Specifically, based on the paper, we can express the original problem as

$$\arg\min_{\hat{\boldsymbol{Q}}\in\mathcal{U}} D(\hat{\boldsymbol{Q}}||\hat{\boldsymbol{P}}), \quad \text{subject to} \quad \hat{\boldsymbol{Q}}^{\top}\mathbf{1}_K = \mathbf{1}_N, \hat{\boldsymbol{Q}}\mathbf{1}_N = \frac{N}{K}\mathbf{1}_K, \tag{1}$$

To see the problem more clearly, we first define that

$$\boldsymbol{S} = \frac{1}{N}\hat{\boldsymbol{Q}}, \boldsymbol{T} = \frac{1}{N}\hat{\boldsymbol{P}}, \tag{2}$$

Then we can rewrite the constraint of Eq. (1) as

$$\boldsymbol{S}^{\top}\mathbf{1}_K = \frac{1}{N}\mathbf{1}_N, \boldsymbol{S}\mathbf{1}_N = \frac{1}{K}\mathbf{1}_K, \tag{3}$$

Besides, since we only consider one-hot pseudo labels, the objective function of Eq. (1) is transformed to

$$\begin{aligned} D(\hat{\boldsymbol{Q}}||\hat{\boldsymbol{P}}) &= \frac{1}{N}\sum_{n=1}^{N}\sum_{y=1}^{K} -\hat{Q}_{yn}\log\hat{P}_{yn} \\ &= \frac{1}{N}\sum_{n=1}^{N}\sum_{y=1}^{K} -(NS_{yn})\log(NT_{yn}) \\ &= \sum_{n=1}^{N}\sum_{y=1}^{K} -S_{yn}(\log N + \log T_{yn}) \\ &= -\log N\sum_{n=1}^{N}\sum_{y=1}^{K} S_{yn} - \sum_{n=1}^{N}\sum_{y=1}^{K} S_{yn}\log T_{yn} \end{aligned} \tag{4}$$

According to Eq. (3), $\sum_{y=1}^{K} S_{yn} = \frac{1}{N}$, $\forall n$, then

$$D(\hat{\boldsymbol{Q}}||\hat{\boldsymbol{P}}) = -\log N + \sum_{n=1}^{N}\sum_{y=1}^{K} S_{yn}(-\log T_{yn}) \tag{5}$$

Obviously, minimizing $D(\hat{\boldsymbol{Q}}||\hat{\boldsymbol{P}})$ is equivalent to minimizing $\sum_{n=1}^{N}\sum_{y=1}^{K} S_{yn}(-\log T_{yn})$. The $\sum_{n=1}^{N}\sum_{y=1}^{K} S_{yn}(-\log T_{yn})$ can be abbreviated as $\langle \boldsymbol{S}, -\log \boldsymbol{T}\rangle_F$, where $\langle\cdot,\cdot\rangle_F$ denotes the Frobenius dot-product. Thus, the problem of Eq. (1) is transformed to

$$\arg\min_{\boldsymbol{S}} \langle \boldsymbol{S}, -\log \boldsymbol{T}\rangle_F, \quad \text{subject to} \quad \boldsymbol{S}^\top \mathbf{1}_K = \frac{1}{N}\mathbf{1}_N, \boldsymbol{S}\mathbf{1}_N = \frac{1}{K}\mathbf{1}_K, \tag{6}$$

Using the notion of [3], we rewrite the problem as

$$\arg\min_{\boldsymbol{S}\in\mathcal{U}} \langle \boldsymbol{S}, -\log \boldsymbol{T}\rangle_F, \tag{7}$$

where

$$\begin{aligned} \mathcal{U} := \{\boldsymbol{S} \in \mathbb{R}_+^{K\times N} | \boldsymbol{S}\mathbf{1}_N = \boldsymbol{r}, \boldsymbol{S}^\top \mathbf{1}_K = \boldsymbol{c}\}, \\ \boldsymbol{r} = \frac{1}{K}\mathbf{1}_K, \boldsymbol{c} = \frac{1}{N}\mathbf{1}_N \end{aligned} \tag{8}$$

here $\boldsymbol{r}$ and $\boldsymbol{c}$ are the marginal projections of $\boldsymbol{S}$ onto its rows and columns, respectively. This is called an optimal transport problem [3,1] between $\boldsymbol{r}$ and $\boldsymbol{c}$ given the cost matrix '$-\log \boldsymbol{T}$'. Traditional algorithms are difficult to solve it, since it involves the data points of the whole dataset. Thus, we adopt the fast version of Sinkhorn-Knopp algorithm [3] to address this issue. This amounts to adding a regularization term to Eq. (6)

$$\arg\min_{\boldsymbol{S}\in\mathcal{U}} \langle \boldsymbol{S}, -\log \boldsymbol{T}\rangle_F + \frac{1}{\xi} D(\boldsymbol{S}||\boldsymbol{r}\boldsymbol{c}^\top), \tag{9}$$

here ξ is a parameter that controls the balance between the convergence speed and problem approximation [1]. Then, the solution to Eq. (9) is

$$\boldsymbol{S}^* = \mathrm{diag}(\mathbf{u})\boldsymbol{T}^{\xi}\mathrm{diag}(\mathbf{v}), \tag{10}$$

here $\mathbf{u} \in \mathbb{R}^K$, $\mathbf{v} \in \mathbb{R}^N$ are initially set as $\mathbf{c}$ and $\mathbf{r}$ respectively and then iteratively updated by

$$\forall y : u_y \leftarrow \left[\boldsymbol{T}^{\xi}\mathbf{v}\right]_y^{-1} \quad \forall n : v_n \leftarrow \left[\mathbf{u}^\top \boldsymbol{T}^{\xi}\right]_n^{-1}. \tag{11}$$

Since the $\boldsymbol{S}$ is relaxed to be continuous in Eq (8), we apply a rounding procedure on $\boldsymbol{S}^*$ to obtain the integral solution [1]. Besides, according to Eq. (2), we can see that the solution of the problem,*i.e.*, Eq. (10), is exactly consistent with the formula expressed in the paper.

2 Additional Details about Datasets

The datasets used in the paper include: NTU-RGBD 60 (NTU-60) [10], NTU-RGBD 120 (NTU-120) [10], PKU Multi-Modality (PKUMMD) [8], Skeletics [5]. For NTU-60 and PKUMMD, two evaluation protocols are generally used in previous methods: 1) Cross-Subject (xsub): training data and validation data are collected from different subjects. 2) Cross-View (xview): training data and validation data are collected from different camera views. For NTU-120, two different protocols are recommended in previous methods: 1) Cross-Subject (xsub): training data and validation data are collected from different subjects. 2) Cross-Setup (xset): training data and validation data are collected from different setup IDs. Different protocols split the training and validation sets differently. As for Skeletics, there is only one protocol to split the training and validation sets. Besides, Skeletics is much larger than other datasets and some classes are common with the classes of other datasets, thus we sample 30 classes that don't overlap with the classes of other datasets. Three cross-domain tasks are evaluated in the paper: NTU-60 → Skeletics, Skeletics → PKUMMD, and NTU-60+ → PKUMMD. In all the tasks, we train the models using the training split of source and target datasets, and evaluate it on the validation split of the target dataset. The comparison of these tasks is shown in Table 1.

Table 1. The datasets used in different tasks.

	NTU-60→Skeletics			Skeletics→PKUMMD			NTU-60+→PKUMMD		
	Train		Test	Train		Test	Train		Test
	Source	Target	Target	Source	Target	Target	Source	Target	Target
Dataset	NTU-60	Skeletics	Skeletics	Skeletics	PKUMMD	PKUMMD	NTU-60+	PKUMMD	PKUMMD
Protocol	xview	-	-	-	xview	xview	xsub	xsub	xsub
Split	train	train	val	train	train	val	train	train	val
Camera	fixed	moving	moving	moving	fixed	fixed	fixed	fixed	fixed
Scenario	indoor	wild	wild	wild	indoor	indoor	indoor	indoor	indoor
Extractor	Kinect V2	VIBE	VIBE	VIBE	Kinect V2	Kinect V2	Kinect V2	Kinect V2	Kinect V2
Noise	small	large	large	large	small	small	small	small	small
No. of Setups	17	–	–	–	3	3	15	3	3
No. of Subjects	40	–	–	–	66	66	33	57	9
No. of Classes	60	30	30	30	51	51	60	41	41
No. of Samples	37, 646	24, 265	2, 341	24, 265	14, 357	7, 188	22, 935	18, 841	2, 704

3 Visualization of Datasets

To vividly show the style differences between datasets, we provide some visualized examples in Fig. 1. As can be seen from the comparison of RGB images, compared to the NTU RGBD and PKUMMD, the Skeletics is a harder dataset for skeleton extraction owing to the fast movement of humans and cameras, the incomplete body parts, the complex backgrounds, *etc.* Hence, the skeletons of NTU RGBD and PKUMMD are generally of high quality, while that of the Skeletics dataset contains more noises, *e.g.*, missed detection, deformed body parts, as shown in Fig. 1. Furthermore, we present some specific action classes for each dataset in Fig. 2.

The joints in Microsoft Kinect V2 and VIBE are somewhat different. According to the positions of joints on the human body, we select 15 joints as shared joints, as shown in Fig. 3. Note that even these shared joints are defined slightly differently in Microsoft Kinect V2 and VIBE, which is also a kind of style gap.

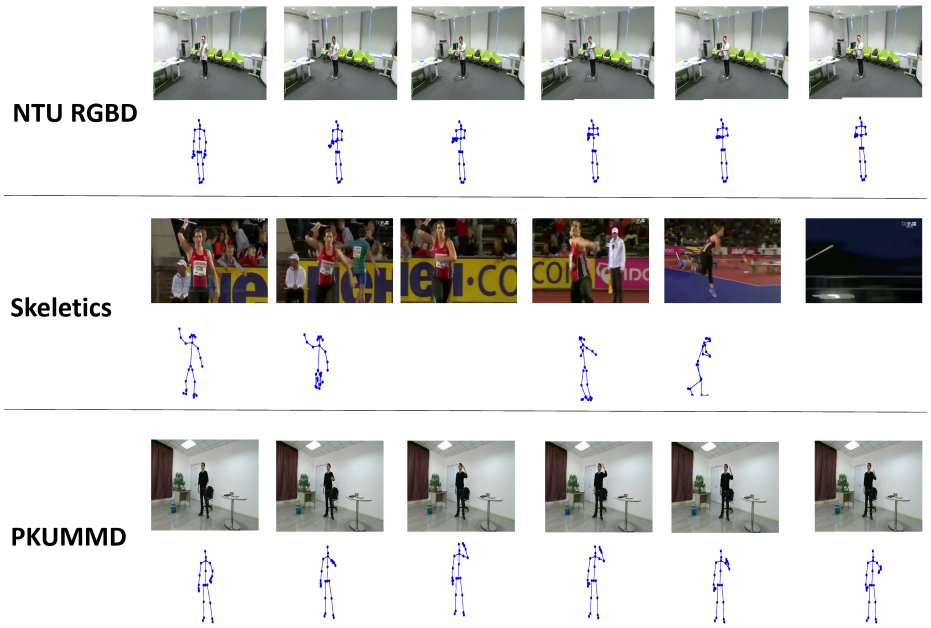


Fig. 1. Visualization of the image and skeleton sequences from NTU RGBD, Skeletics, PKUMMD, respectively.

4 Additional Details about Evaluation Metrics

Accuracy (ACC) is computed by assigning each cluster with the dominating class label and taking the average correct classification rate as the final score. Normalised Mutual Information (NMI) quantifies the normalised mutual dependence between the predicted labels and the ground-truth. Adjusted Rand Index (ARI) evaluates the clustering result as a series of decisions and measures its quality according to how many positive/negative sample pairs are correctly assigned to the same/different clusters. All of the metrics scale from 0 to 1 and a higher value is better.

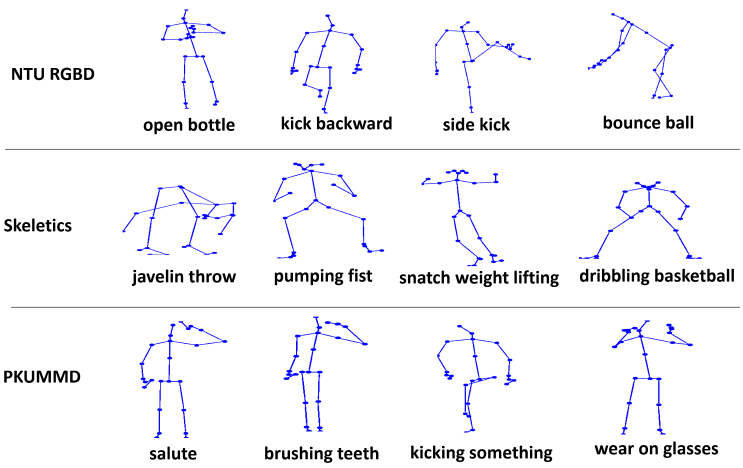
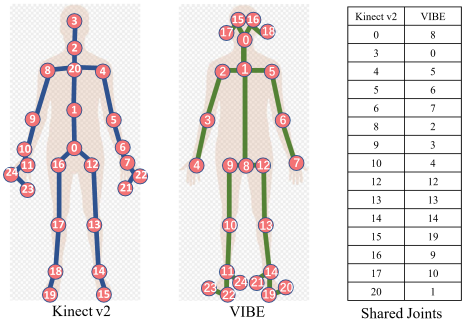


Fig. 2. Visualization of some specific classes in NTU RGBD, Skeletics, and PKUMMD.

5 Additional Implementation Details

5.1 Implementation Details of CoDT

The ST-GCN [13] is adopted as the encoder. All the classifiers take the encoded feature vectors as input, while the decoder D_0 takes the 1D feature map before the temporal pooling layer as input. The classifiers are comprised by a full-connected layer, while the decoder is implemented by a two-layer MLP to only regress the spatial locations of joints. For data pre-processing, we resize each skeleton sequence to the length of 50 frames by linear interpolation. Following [14], *Shear* (*i.e.*, spatial transformation depends on a shear amplitude β) and *Crop* (*i.e.*, temporal distortion depends on a padding ratio $1/\gamma$) are used for data augmentation. For weak augmentation, β is set as 0.5 and γ is set as 6. As for strong augmentation, we set β and γ as 1 and 3. Besides, we randomly mask each part of skeletons in each frame with a probability of 0.3. The strong augmentation is only used for the student models during finetuning stage. The joint definitions are different in the Microsoft Kinect V2 and VIBE, thus in the former two tasks, we only use the shared 15 joints of source and target skeletons in $\mathcal{B}_0$ while keeping all joints of target skeletons in $\mathcal{B}_1$. For the implementation of CoDT, we use SGD with momentum 0.9 as the optimizer. In pretraining stage, the models are trained with the learning rate 0.1. After pretraining, we first apply the spherical k-means on the features of all target samples and then use the centroids to initialize weights of target classifiers. The models are then trained with the learning rate 0.01. The batch size n^t and n^s are set to 128. The loss weights λ_{sup}, λ_{dec}, λ_{cont}, λ_{cls} and λ_{cot} are set to 1, 20, 1, 5 and 10, respectively. The decay rate α of EMA is set as 0.999. The ξ is set as 10. The temperature ρ is set to 7. All the experiments are conducted on the PyTorch [9].



Kinect v2	VIBE
0	8
3	0
4	5
5	6
6	7
8	2
9	3
10	4
12	12
13	13
14	14
15	19
16	9
17	10
20	1

Shared Joints

Fig. 3. Illustration of the 15 shared joints of Kinect V2 and VIBE.

5.2 Implementation Details of CCM-*FP* and CCM-*PF*

In CCM-*FP*, we measure the pairwise feature similarity by using the indices of feature elements rank ordered according to their magnitudes. If two samples share the same top-k ($k = 5$ is used) indices in their respective lists of rank ordered feature elements, the paired samples belong to a positive pair, otherwise, a negative pair. Note that, the above operations are performed on the features from the teacher model. Then the constructed binary pair-wise pseudo labels of each branch are used to optimize the pair-wise prediction similarities for the other branch, which is the same as CCM.

In CCM-*PF*, we construct the binary pair-wise pseudo labels in the same way as CCM. But different from CCM which treats the pair-wise prediction similarity as the similarities of samples, CCM-*PF* uses the cosine similarities of features as the pair-wise similarities of samples. That's, the pair-wise pseudo labels of each branch are used to optimize the pairwise feature similarities for the other branch. For the sake of fairness, we adopt our proposed formula of the supervised contrastive loss function to optimize both CCM-*FP* and CCM-*PF*.

6 Additional Cross-Domain Tasks

Apart from the above-mentioned cross-domain tasks, here we introduce two more tasks: FineGym (VIBE) $\rightarrow$ Skeletics (VIBE), Kinetics-400 (OpenPose) $\rightarrow$ NTU-60 (HRNet). The FineGym dataset [11] is a fine-grained action recognition dataset with $29K$ videos of 99 fine-grained action classes. The Kinetics-400 dataset [6] contains around $300,000$ video clips retrieved from YouTube. The videos cover as many as 400 action classes. The VIBE [7] is a 3D pose estimator. The Openpose [2], HRNet [12] are two 2D pose estimators which produce 2D locations and confidence scores for the joints. That's, each joint is represented by a (pseudo) 3D tensor containing two coordinates and a score. The poses of FineGym (VIBE) and NTU-60 (HRNet) are provided by [4], and the poses of Kinetics-400 (OpenPose) are provided by [13]. These tasks can simulate

the transfer learning between datasets extracted by different 2D or 3D pose estimation algorithms. In the task of FineGym (VIBE) → Skeletics (VIBE), the FineGym dataset only contains the fine-grained gymnastic action categories, while the categories of Skeletics are much more coarse and unconstrained. And in the task of Kinetics-400 (OpenPose) → NTU-60 (HRNet), the extracted joints in two domains are extensively different in qualities and styles. Intuitively, these two tasks are extremely difficult.

We conduct the ablation study on these two tasks to further examine the effectiveness of our method. The results are shown in Table 2. It can be seen that the new tasks are very hard. For example, the ACC on the task of Kinetics-400 → NTU-60 is only around 10%. However, even in such cases, our proposed OCM and CCM can significantly improve the performances of base modules, proving the generalization ability of our method.

Table 2. Results on the additional tasks. The 'F → S' and 'K → N' represent 'FineGym (VIBE) → Skeletics (VIBE)' and 'Kinetics-400 (OpenPose) → NTU-60 (HRNet)', respectively.

Methods	$\mathcal{B}_*$	F → S			K → N		
		ACC	NMI	ARI	ACC	NMI	ARI
BM†	$\mathcal{B}_0$	16.6	19.1	5.0	12.2	26.2	5.2
	$\mathcal{B}_1$	18.7	23.3	6.4	10.4	18.6	3.7
BM + OCM	$\mathcal{B}_0$	17.7	20.1	5.5	13.6	29.4	6.2
	$\mathcal{B}_1$	21.8	26.4	8.7	12.1	22.9	5.2
BM + OCM + CCM	$\mathcal{B}_0$	22.6	25.4	10.7	14.6	30.9	7.2
	$\mathcal{B}_1$	23.4	26.3	11.0	13.9	29.8	6.5

References

1. Asano, Y.M., Rupprecht, C., Vedaldi, A.: Self-labelling via simultaneous clustering and representation learning. In: ICLR (2020)
2. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR (2017)
3. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. In: NIPS (2013)
4. Duan, H., Zhao, Y., Chen, K., Shao, D., Lin, D., Dai, B.: Revisiting skeleton-based action recognition. arXiv preprint arXiv:2104.13586 (2021)
5. Gupta, P., Thatipelli, A., Aggarwal, A., Maheshwari, S., Trivedi, N., Das, S., Sarvadevabhatla, R.K.: Quo vadis, skeleton action recognition? International Journal of Computer Vision **129**(7), 2097–2112 (2021)
6. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)

7. Kocabas, M., Athanasiou, N., Black, M.J.: Vibe: Video inference for human body pose and shape estimation. In: CVPR (2020)
8. Liu, C., Hu, Y., Li, Y., Song, S., Liu, J.: Pku-mmd: A large scale benchmark for continuous multi-modal human action understanding. arXiv preprint arXiv:1703.07475 (2017)
9. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
10. Shahroudy, A., Liu, J., Ng, T.T., Wang, G.: Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In: CVPR (2016)
11. Shao, D., Zhao, Y., Dai, B., Lin, D.: Finegym: A hierarchical video dataset for fine-grained action understanding. In: CVPR (2020)
12. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: CVPR (2019)
13. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: AAAI (2018)
14. Yao, F.: Cross-domain few-shot learning with unlabelled data. arXiv preprint arXiv:2101.07899 (2021)