Online Multi-task Clustering for Human Motion Segmentation

Gan Sun[1,2,3], Yang Cong[1,2]*, Lichen Wang[4], Zhengming Ding[5], Yun Fu[4]
[1]State Key Laboratory of Robotics, Shenyang Institute of Automation,
Chinese Academy of Sciences, Shenyang, 110016, China. †
[2]Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang, China.
[3]University of Chinese Academy of Sciences, Beijing, 100049, China.
[4]Northeastern University, USA. [5]Indiana UniversityPurdue University Indianapolis, USA.
{sungan1412, congyang81, wanglichenxj}@gmail.com zd2@iu.edu yunfu@ece.neu.edu

Abstract

Human motion segmentation in time space becomes attractive recently due to its wide range of potential applications on action recognition, event detection, and scene understanding tasks. However, most existing state-of-the-arts address this problem upon an offline and single-agent scenario, while there are a lot of urgent requirements to segment videos captured from multiple agents for real-time application (e.g., surveillance system). In this paper, we propose an Online Multi-task Clustering (OMTC) model for an online and multi-agent segmentation scenario, where each agent corresponds to one task. Specifically, a linear autoencoder framework is designed to project motion sequences into a common motion-aware space across multiple collaborating tasks, while the decoder obtains motion-aware representation of each task via a temporal preserved regularizer. To tackle distribution shifts problem between each pair of tasks, the task-specific projections are further proposed to align representation across the motion segmentation tasks. By this way, significant motion knowledge can be shared among multiple tasks, and the temporal data structures are also well preserved. For the model optimization, an efficient and effective online optimization mechanism is derived to solve the large-scale formulation in real-time applications. Experiment results on Keck, MAD and our collected human motion datasets demonstrate the robustness, high-accuracy and efficiency of our OMTC model.

1. Introduction

One of the major challenges for video analysis/understanding [37, 15, 44, 29, 39, 34] is to localize and classify human motion among long, untrimmed videos, e.g., in security surveillance and health-care. In order to cluster a long sequential human motion into several short and non-overlapping fragments, many methods have been introduced to address this temporal data clustering problem, e.g., motion-based [38], temporal proximity based [13, 7] and representation based [12, 21, 17, 32, 33] methods. However, existing motion segmentation works are performed upon one agent. That means these methods segment each video individually, which cannot benefit from diverse and enriched knowledge from a network of multiple agents for the motion segmentation tasks. In this paper, our work investigates how to apply online knowledge transfer learning into multi-agent human motion segmentation task.

For the transfer learning framework, most recent works

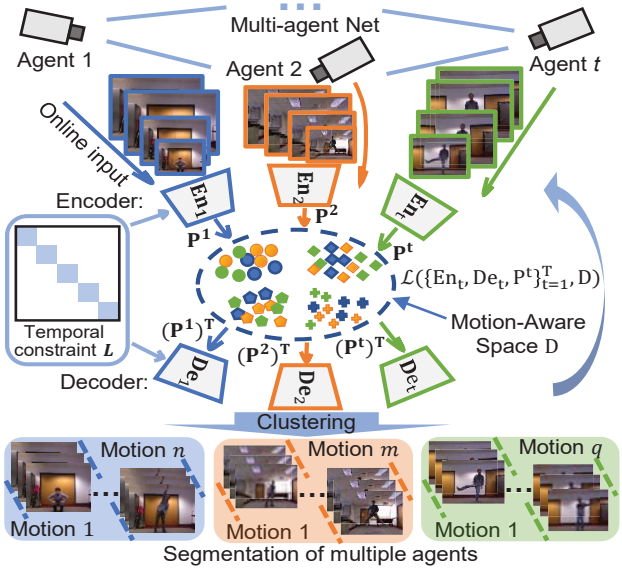

Figure 1. The framework of our proposed model. The motion knowledge among a network of multi-agent can be efficiently transferred via the motion-aware space D, where different shapes and colors denotes different motions and agents, respectively.

*The corresponding author is Prof. Yang Cong.

†This work is supported by NSFC (61821005, 61722311, U1613214, 61533015), and LiaoNing Revitalization Talents Program (XLY-C1807053).

[23, 8, 31, 19, 4] have witnessed superior performance via learning several effective domain-invariant features from two different domains (if each agent is treated as a domain), or improving the clustering performance by transferring knowledge among the related tasks [40, 26, 42] (if each agent is treated as a task). However, the adoptions of most recently-proposed transfer learning models are not suitable in the human motion segmentation task. Furthermore, there exist two challenges in this online multi-agent motion segmentation problem: 1) Segmentation Space Consistency: for transfer motion segmentation problem, different agents could involve different actions, e.g., full-body motion such as Running as well as upper-body motion such as Basketball Dribble. 2) Distribution Disparity: the distribution disparity among two different agents exists due to dynamic backgrounds, photometric variations, etc, which can further lead to the distribution shift problem. To this end, multi-agent motion segmentation is dramatically different when comparing with single-agent offline scenario.

To address above challenges, as shown in Figure 1, we propose an Online Multi-task Clustering (OMTC) model by incorporating linear autoencoder paradigm with general cross-task framework. Taking the encoder-decoder paradigm, an encoder is setup to consecutively project different tasks into a common motion-aware space with temporal regularizer, which can link all the learned motion segmentation tasks in different agents. Meanwhile, the decoder exerts an additional temporal constraint to reconstruct the original data, which ensure the motion-aware space is directly derived from all the agents. For the cross-task framework, we mitigate the distribution divergences between different agents by jointly learning task-specific projection for each agent, which can maintain the manifold structure of different agents. For model optimization, we derive a general online learning formulation for handling large-scale data, and further encode the optimization problem via an alternatingly direction strategy. After constructing an affinity graph using the encoding and decoding matrices, multiple segments of each agent can be grouped via spectral clustering. Finally, we evaluate our OMTC model on Keck, MAD and our collected datasets, and the experiment results demonstrate the effectiveness of our model when comparing with the state-of-the-art single and multi-task clustering models.

The novelty of our proposed model is threefold:

- We develop a novel model called online multi-task clustering model (OMTC) for multi-agent human motion segmentation task. To the best of our knowledge, this is an earlier work to consider the problem of consecutively unsupervised human segmentation task among a network of agents.
- A linear autoencoder is used to derive a motion-aware latent space across multiple agents. Meanwhile, two graph regularizers are used to capture the temporal information of encoder and decoder for better clustering.
- We present an efficient cross-task strategy to mitigate the distribution differences of data among individual agents. An efficient online learning algorithm is also designed to learn the motion-aware latent space and representation codings alternatingly.

2. Related Work

Our work mainly draws from subspace clustering, temporal data clustering and multi-task clustering, so we give a brief review on these three topics.

Subspace Clustering [20, 36, 18] is a classical topic has been introduced to automatically group the data into low-dimensional subspace. For instance, sparse subspace clustering (SSC) [5] adopts the sparse representation of data for clustering; instead of sparse constraint, low-rank representation (LRR) [22] can uncover the underlying data structure and remove the noisy by finding the lowest rank representation; least-square regression (LSR) [24] designs an efficient regression objective with Frobenius norm. Different from the above methods which are based on batch optimization and have to load all the clustering samples into memory during optimization, [6] designs an online algorithm for Robust Principal Component Analysis (RPCA) problem, which is just based on stochastic optimization of the batch RPCA. To online implement LRR, [27] develops a novel online algorithm of LRR, which can reduce the memory cost from $\mathcal{O}(n^2)$ to $\mathcal{O}(pd)$. However, these methods above can not efficiently tackle the time-series data or multi-agent task well since they ignore the temporal information and cross-task correlation among data.

Temporal Data Clustering methods aim to group/segment a long temporal data (e.g., action video) into several meaningful and non-overlapping parts. To achieve this, [43] adopts a dynamic time alignment kernel to group time-series data, and further use it in human motion segmentation. [9] learns a multi-class SVM to discriminate among temporal clusters, and determines the start and the end of each segment. To incorporate the temporal graph constraint, temporal subspace clustering (TSC) [21] proposes to learn expressive codings for temporal clustering by a dictionary learning method. However, these methods need to collect all the data before the optimization process, and are difficult to obtain an expected result.

Multi-task Clustering appeals a growing interest in artificial intelligence community due to its benefits of simultaneously clustering multiple tasks [28, 2]. Basically, multi-task clustering [40, 26, 42] allows transferring useful knowledge among different clustering tasks to improve the performance. In this paper, our OMTC model mainly adopts multi-task clustering technique to human motion segmentation task (e.g., security surveillance system), where each agent corresponding to one task receives its sequential data

in a batch manner, and collectively improve segmentation performance among a network of multi-agent.

3. The Proposed Model

3.1. Multi-agent Motion Segmentation Problem

Given a network of T agents with $\mathcal{T}^1, \ldots, \mathcal{T}^T$ tasks, where each agent faces an unsupervised consecutive motion segmentation task (we assume that each agent encounters an individual task), and each task has a set of n_t training data samples $X^t \in \mathbb{R}^{d\times n_t}$, and its probability distribution $\mathcal{P}(X^t)$. The intention of multi-agent motion segmentation problem is to uncover shared knowledge among all the segmentation tasks, and give the motion alignment matrix for each segmentation task.

3.2. Preliminaries

Consider the situation that we have human motion data $\{(X^t)_{i=1}^{n_t}\}_{t=1}^T$ for T agents, and the target is to improve the performance of motion segmentation via transfer learning. The existing transfer learning methods mostly learn a projection function $\mathcal{F} : X^i \mapsto X^j$ to uncover the common latent features. Basically, the common subspace may not well model the motion interactions across different tasks due to the existing of various structures. To mitigate this issue and inspired by [14], we propose an autoencoder based framework which connects multiple collaborating tasks by studying their common principle motion characteristics

In the encoding process, the input matrix X^t for the t-th task can be decomposed as the product of a code matrix $D \in \mathbb{R}^{d\times k}$ with a linear encoding matrix $V^t \in \mathbb{R}^{n_t\times k}$, i.e., $X^t \sim D(V^t)^\top$, where the code matrix D containing common principle motion characteristics is shared by multiple tasks, called as latent **motion-aware space**. For the decoding process, the code matrix can be decomposed into the product of the decoding matrix U^t with the original input matrix X^t, i.e., $D \sim X^tU^t$ with $U^t \in \mathbb{R}^{n_t\times k}$. Therefore, the encoder-decoder framework for the t-th task can be defined by minimizing the following formulation:

$$\min_{D,U^t,V^t} \left\|X^t - D(V^t)^\top\right\|_F^2 + \lambda_1 \left\|D - X^tU^t\right\|_F^2, \qquad (1)$$

where the first term decomposes the original input matrix X^t into an encoding matrix $(V^t)^\top$ and a motion-aware matrix D, which ensures the latent representative motions can well capture the original involved motions. The second term customizes the motion-aware matrix D can be derived from the original input matrix X^t via a decoding matrix U^t. In the following, we introduce how to link the segment task of specific agent accurately.

3.3. Online Multi-task Clustering (OMTC)

In addition to the principle motion characteristics among all the tasks, another challenge we concern about is that motion data in two tasks have different domains, e.g., two tasks X^i and X^j have two different distributions with $X^i \in \mathbb{R}^{d_i\times n_i}$, $X^j \in \mathbb{R}^{d_j\times n_j}$. Take action segmentation task in surveillance system as an example, we have a series of action shots captured from two different scenes. Directly transferring the segmentation knowledge from one task to another one, the segmentation performance will be unsatisfied. Thus, tt will be desirable that two projections can bring the data from two domains to a common low-dimensional space, and not lose too much information available in the original task. To facilitate this, we consider adding a task-specific projection to each task, and use the defined motion-aware matrix D as a bridge. Thus, we can reformulate our objective function as the following formulation:

$$\begin{aligned}\min_{\substack{D,\{P^t\}_{t=1}^T\\ \{U^t,V^t\}_{t=1}^T}} &\sum_{t=1}^T \Big(\left\|P^tX^t - D(V^t)^\top\right\|_F^2 + \lambda_1 \left\|(P^t)^\top D - X^tU^t\right\|_F^2\Big),\\ s.t.,\ & P^tX^tH^t(X^t)^\top(P^t)^\top = I,\ \ \forall t = 1,\ldots,T,\end{aligned} \qquad (2)$$

where $P^t \in \mathbb{R}^{d\times p_t}$ is the projection matrix corresponding to the t-th task, p_t is the dimension of the t-th task. The constraint $P^tX^tH^t(X^t)^\top(P^t)^\top = I$ can preserve the data variance after adaptation, where $H^t = I - \frac{1}{n_t}\mathbf{1}$, and $\mathbf{1}$ is all-one matrix. Thus, the distribution shift between different tasks could be well addressed via the bridge $D \in \mathbb{R}^{d\times k}$, and the discriminative knowledge across different tasks could be reused to facilitate the segmentation performance.

Temporal Laplacian Regularizer: Due to the fact that human motion data is a sequence of time-series data, it also needs to introduce a temporal constraint function to incorporate the temporal information in time series X^t. Different from the previous method [21] which just imposes temporal regularizer on the encoder matrix V^t, in this paper, we capture the sequential relationships among X^t on both encoding and decoding matrices, i.e.,

$$\begin{aligned} f(V^t) &= \frac{1}{2}\sum_{i=1}^{n_t}\sum_{j=1}^{n_t} w_{ij}\left\|v_i^t - v_j^t\right\|_2^2 = \mathrm{tr}((V^t)^\top L^t V^t),\\ f(U^t) &= \frac{1}{2}\sum_{i=1}^{n_t}\sum_{j=1}^{n_t} w_{ij}\left\|u_i^t - u_j^t\right\|_2^2 = \mathrm{tr}((U^t)^\top L^t U^t),\end{aligned} \qquad (3)$$

where L^t which is a temporal Laplacian matrix is defined as that in [21]. Thus, the target of multi-agent segmentation task can be reformulated as the following formulation:

$$\begin{aligned}\min_{\substack{D,\{P^t\}_{t=1}^T\\ \{U^t,V^t\}_{t=1}^T}} &\sum_{t=1}^T \Big(\left\|P^tX^t - D(V^t)^\top\right\|_F^2 + \lambda_1 \left\|(P^t)^\top D - X^tU^t\right\|_F^2\Big)\\ &+\lambda_2\big(\mathrm{tr}((U^t)^\top L^t U^t) + \mathrm{tr}((V^t)^\top L^t V^t)\big),\\ s.t.,\ & P^tX^tH^t(X^t)^\top(P^t)^\top = I,\ \ \forall t = 1,\ldots,T,\end{aligned} \qquad (4)$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are trade-off parameters to balance different constraint terms. Notice that with these temporal constraints, frame x_i^t and its sequential neighbors $\{x_{i-s/2}^t, \ldots, x_{i+s/2}^t\}$ are encouraged to hold the similar codes in encoder matrix $\{v_{i-s/2}^t, \ldots, v_{i+s/2}^t\}$ and decoder matrix $\{u_{i-s/2}^t, \ldots, u_{i+s/2}^t\}$.

With the obtained matrices V^t and U^t for the t-th task, a graph matrix G^t is then generated for the sequential clustering procedure. Note that U^t and V^t are symmetric and can be expressed by the same parameters, i.e., $U^t = V^t$. Such a design can make $G^t(i,j) = u_i^t(u_j^t)^\top / \|u_i^t\|_2 \|u_j^t\|_2$ or $G^t(i,j) = v_i^t(v_j^t)^\top / \|v_i^t\|_2 \|v_j^t\|_2$, where u_i^t and v_j^t are the corresponding coding of x_i^t and x_j^t, respectively. However, this strategy can ignore the differences between the representability of the encoder V^t for original space X^t and the selectablity of decoder U^t from X^t, respectively. So we define the graph G^t as $u_i^t(v_j^t)^\top / \|u_i^t\|_2 \|v_j^t\|_2$ in this paper. With the generated G, an effective clustering algorithm (i.e., Normalized Cuts/Spectral Clustering [25]) is then used to generate the final clustering results.

4. Model Optimization

In general, most motion segmentation models focus on the offline setting, and need to access every frame in each iteration of the optimization. However, for many important applications (e.g., health-care and surveillance system), this optimization process is quite inefficient. Moreover, it has to be re-performed on all collected samples when a new sample/action is added, which is not practical in these applications. To this end, we present an online optimization algorithm for the objective function in Eq. (4), which will have lower computational cost in terms of memory space and CPU time than offline algorithm. The basic ideas are:

- The motion data for each task are given in mini-batches. Formally, we use $m \in \{1, 2, \ldots\}$ to index each time stamp when a batch of new motion data arrives, and the data in each batch is from the same task.
- We introduce several statistical records to store the previous knowledge, so that batches of coming motion data can be used iteratively to update variable P^t's, and further refine the motion-aware space D.

Based on the basic ideas, we have the following objective function at each time stamp m:

$$\begin{aligned}\min_{\substack{D,\{P^t\}_{t=1}^T,\\ \{U_{(1:m)}^t,V_{(1:m)}^t\}_{t=1}^T}} & \sum_{t=1}^{T}\Big(\left\|P^t X_{(1:m)}^t - D(V_{(1:m)}^t)^\top\right\|_F^2 \\ & +\lambda_1 \left\|(P^t)^\top D - X_{(1:m)}^t U_{(1:m)}^t\right\|_F^2\Big) \\ +\lambda_2(\mathrm{tr}((U_{(1:m)}^t)^\top L_{(1:m)}^t U_{(1:m)}^t) & +\mathrm{tr}((V_{(1:m)}^t)^\top L_{(1:m)}^t V_{(1:m)}^t)) \\ s.t.,\ P^t X_{(1:m)}^t H^t (X_{(1:m)}^t)^\top (P^t)^\top = I,\ & \forall t = 1, \ldots, T,\end{aligned} \tag{5}$$

where the subscript m denotes the value of a variable in the time stamp m, and the subscript $(1:m)$ denotes the cumulative values of a variable from time stamp 1 to m. For instance, X_m^t denotes the motion data arriving at time stamp m for the t-th task, and $X_{(1:m)}^t$ denotes all the motion data has been received for the t-th task until m. Basically, if we directly adopt an offline alternating method to this objective function, the computational complexity will dramatically increase as more data come. To reduce the computational complexity, in the next section, we introduce how to efficiently solve our proposed model when we receive a batch of data X_m^t at time stamp m.

4.1. Solving for $\{U_m^t, V_m^t\}_{t=1}^T$:

With the fixed projection P^t and matrix D, U_m^t and V_m^t for each task are the variables in this subproblem, and the model in Eq. (5) can be expressed as:

$$\begin{aligned}\min_{\{U_m^t,V_m^t\}_{t=1}^T} \sum_{t=1}^{T}\Big(& \left\|P^t X_m^t - D(V_m^t)^\top\right\|_F^2 \\ & + \lambda_1 \left\|(P^t)^\top D - X_m^t U_m^t\right\|_F^2\Big) \\ & + \lambda_2\big(\mathrm{tr}((U_m^t)^\top L_m^t U_m^t) + \mathrm{tr}((V_m^t)^\top L_m^t V_m^t)\big).\end{aligned} \tag{6}$$

After setting the derivative of Eq. (6) with respect to V_m^t's and U_m^t's to zero, we can have the following equations:

$$\begin{aligned}\lambda_2 L_m^t V_m^t + V_m^t D^\top D &= (X_m^t)^\top (P^t)^\top D, \\ \lambda_2 L_m^t U_m^t + \lambda_1 (X_m^t)^\top X_m^t U_m^t &= \lambda_1 (X_m^t)^\top (P^t)^\top D.\end{aligned} \tag{7}$$

From the equation above, we can notice that the optimization equation of V_m^t is a standard Sylvester equation, which could be effectively optimized by adopting existing tools, e.g., Bartels-Stewart algorithm [1]. The solution of U_m^t can be given as follows:

$$U_m^t = ((X_m^t)^\top X_m^t + \lambda_2/\lambda_1)^{-1} (X_m^t)^\top (P^t)^\top D. \tag{8}$$

4.2. Solving for $\{P^t\}_{t=1}^T$:

With the obtained encoding V_m^t's, U_m^t's and motion-aware matrix D, the optimization problem of variable P^t for the t-th task can be denoted as:

$$\begin{aligned}P^t = & \arg\min_{P^t} \left\|P^t X_{(1:m)}^t - D(V_{(1:m)}^t)^\top\right\|_F^2 \\ & + \lambda_1 \left\|(P^t)^\top D - X_{(1:m)}^t U_{(1:m)}^t\right\|_F^2 \\ = & \arg\min_{P^t} \mathrm{tr}\big(P^t (X_{(1:m)}^t (X_{(1:m)}^t)^\top + \lambda_1 (P^t)^\top D D^\top P^t \\ & - 2D(V_{(1:m)}^t + \lambda_1 U_{(1:m)}^t)^\top (X_{(1:m)}^t)^\top)(P^t)^\top\big), \\ s.t.,\ & P^t X_{(1:m)}^t H^t (X_{(1:m)}^t)^\top (P^t)^\top = I.\end{aligned} \tag{9}$$

Eq. (9) can then be solved by using generalized Eigen-decomposition as follows:

$$\left(\mathcal{C}_m^t + \lambda_1 (P^t)^\top D D^\top P^t\right)\rho = \gamma X_{(1:m)}^t H^t (X_{(1:m)}^t)^\top \rho, \tag{10}$$

where γ denotes the eigenvalue of the corresponding eigenvector ρ for the generalized Eigen-decomposition formulation. $\mathcal{C}_m^t$ is a statistical record to store the previous knowledge and denoted as the following formulation:

$$\mathcal{C}_m^t = \sum_{i=1}^{m} X_i^t (X_i^t)^\top - 2D(V_i^t + \lambda_1 U_i^t)^\top (X_i^t)^\top. \tag{11}$$

Each P^t can then be defined as $[\rho_0, \ldots, \rho_{p-1}]$, where the vectors ρ_i $(i = 0, 1, \ldots, p-1)$ are provided by the minimum eigenvalue solutions of Eq. (10), which can minimize the objective function in Eq. (10).

4.3. Solving for D:

After ignoring other variables, D is the single variable in this subproblem, and the formulation can be expressed as:

$$\begin{aligned} \min_D \sum_{t=1}^{T} \left\| P^t X_{(1:m)}^t - D(V_{(1:m)}^t)^\top \right\|_F^2 \\ + \lambda_1 \left\| (P^t)^\top D - X_{(1:m)}^t U_{(1:m)}^t \right\|_F^2. \end{aligned} \tag{12}$$

Thus, the solution of D can be given as:

$$D = \arg\min_D \operatorname{tr}(D^\top D(\mathcal{A}_m + \lambda_1 I)) - 2\operatorname{tr}(D^\top \mathcal{B}_m), \tag{13}$$

where $\mathcal{A}_m \in \mathbb{R}^{k\times k}$ and $\mathcal{B}_m \in \mathbb{R}^{d\times k}$ are statistical records, and are defined as $\sum_{i=1}^{m}\sum_{t=1}^{T}(V_i^t)^\top V_i^t$, $k \times k$ and $\sum_{i=1}^{m}\sum_{t=1}^{T} P^t X_i^t (U_i^t + V_i^t)$, respectively.

Finally, the optimization algorithm for our model is summarized in **Algorithm 1**.

4.4. Computational Complexity

For the computational complexity of our model, the main cost of learning new batch samples in **Algorithm 1** involves several subproblems, and begins by Eq. (7). The computational complexity for Eq. (7) is $O(T(n_m^2(k + n_m^3)) + \sum_{t=1}^{T} d_t n_m^2$, where n_m is the sample number. Next, updating P^t in Eq. (10) contains Eigen-decomposition, which costs $O(\sum_{t=1}^{T} d_t d^2)$. The computational complexity of solving D begins by computing $\mathcal{A}_m$ and $\mathcal{B}_m$, which takes $O(T(n_m^2 k) + \sum_{t=1}^{T} d_t n_m (p + k)))$. Then computational complexity of solving D is $O(T(n_m^2 k) + \sum_{t=1}^{T} d_t n_m (p + k) + dk^2)$. Finally, the time complexity of each iteration part is $O(T(n_m^2(k + n_m^3) + \sum_{t=1}^{T} d_t n_m (p + k + n_m) + dk^2)$. Since k is usually small when comparing with the dimension of feature and number of sample, our proposed OMTC model is thus computationally efficient.

Algorithm 1 Online Multi-task Clustering (OMTC)

Input: Observed samples $\{X^t\}_{t=1}^T$, parameters $d \le d_t$, $\lambda_1 > 0, \lambda_2 > 0$, zero matrix $\{\mathcal{C}_0^t\}_{t=1}^T$, $\mathcal{A}_0$ and $\mathcal{B}_0$.
Output: D, $\{U^t, V^t\}_{t=1}^T$

```
for i = 1, ..., m do
  for t = 1, ..., T do
    if isFirstAccess the sample X_i^t then
      Initialize P^t via PCA;
    else
      Compute V_i^t via Eq. (7);
      Compute U_i^t via Eq. (8);
      Compute C_i^t via Eq. (11);
      Compute P^t via Eq. (10);
    end if
  end for
  Compute A_i via Σ_{j=1}^{i} Σ_{t=1}^{T} (V_j^t)^⊤ V_j^t;
  Compute B_i via Σ_{j=1}^{i} Σ_{t=1}^{T} P^t X_j^t (U_j^t + V_j^t);
  Compute D via Eq. (12).
end for
```

5. Experiment

This section evaluates the performance of our proposed model via comparing with several state-of-the-art clustering models on three human motion datasets. We firstly introduce the used competing models. Then several adopted datasets and experiment results are provided. Finally, some analyses about our model are reported.

5.1. Comparison Models and Evaluation

In our experiment, we evaluate our proposed model with eight representative offline clustering (single and multi-task) models and two online clustering methods. The brief introductions are below:

Offline single-task clustering models:

- Spectral Clustering (SPE) [25], single task clustering, which is a baseline clustering model.
- Low-Rank Representation (LRR) [22], which seeks the best representation using a low-rank constraint.
- Ordered Subspace Clustering (OSC) [30], which can achieve better performance for clustering sequential data by enforcing the consecutive columns of coding matrix to be similar.
- Structured Sparse Subspace Clustering (S^3C) [16], which combines the sparse representation and spectral clustering. We adopt its $SoftS^3C$ version, since this model yields slightly better results on the motion segmentation task [17].
- Temporal Subspace Clustering (TSC) [21], which considers the sequential information in time-series data by constraint of a temporal Laplacian regularization term.
- Transfer Subspace Segmentation (TSS) [32], which incorporates useful information from source data to en-

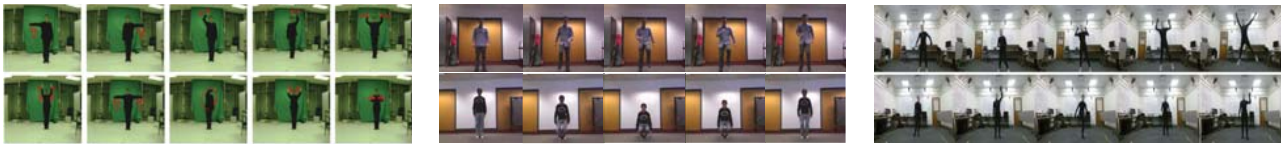

Figure 2. Example frames of Keck dataset (Left), MAD dataset (Middle) and EV-Action dataset (Right).

Table 1. Segmentation comparisons between our model and state-of-the-arts in terms of ACC and NMI on Keck dataset: mean and standard errors over ten random runs. Models with the best performance are bolded, and the runner-up are underlined. M denotes MAD dataset.

Keck	Criteria	SPE[25]	LRR[22]	OSC[30]	SoftS3C[17]	TSC[21]	TSS(M)[32]	ESC-FFS[41]	MTCMRL[42]	OLRSC[6]	ORPCA[27]	Ours
Task 1	ACC(%)	40.46±1.5	36.12±0.9	43.03±2.4	47.93±2.6	51.45±1.1	52.54±2.4	47.00±3.8	47.61±1.6	52.61±0.2	40.78±2.7	**60.68±2.0**
	NMI(%)	48.23±1.1	50.12±1.3	59.79±0.2	64.59±1.8	76.79±0.6	**80.75±0.8**	52.90±2.5	47.54±4.2	51.98±0.1	53.54±1.5	78.63±1.4
Task 2	ACC(%)	41.93±3.3	40.28±2.2	42.57±2.1	48.69±1.6	44.06±1.9	47.74±2.4	45.79±1.7	48.09±1.7	53.04±0.5	35.72±3.6	**56.83±0.9**
	NMI(%)	53.08±2.4	48.13±2.1	50.97±0.1	66.91±2.3	71.03±2.4	74.68±2.9	53.39±2.3	45.87±3.1	48.35±0.2	51.49±0.3	**77.52±1.7**
Task 3	ACC(%)	40.97±2.2	41.90±2.5	43.70±1.9	45.73±2.5	49.18±1.9	51.91±2.4	53.63±4.4	52.44±4.3	45.01±1.4	38.34±2.5	**62.86±1.1**
	NMI(%)	48.48±0.5	47.54±2.5	58.37±0.1	64.53±2.2	71.25±1.0	75.65±2.7	58.04±2.8	47.97±4.6	45.97±0.3	48.49±0.2	**78.57±1.1**
Task 4	ACC(%)	33.94±0.8	32.38±2.6	43.98±2.1	42.01±1.2	53.57±1.8	53.36±2.4	49.06±1.0	45.60±2.1	40.83±0.6	35.10±2.0	**60.13±0.9**
	NMI(%)	37.49±2.5	38.43±2.7	51.51±0.1	54.65±0.4	**77.93±1.7**	77.36±2.4	53.18±3.4	41.21±1.3	41.21±0.7	43.24±0.3	77.39±1.6
Average	ACC(%)	39.32±1.6	37.41±2.8	43.32±1.7	46.09±1.0	49.57±1.6	51.39±2.5	48.87±2.7	48.44±2.4	47.87±0.5	37.48±0.5	**59.63±1.5**
	NMI(%)	46.83±0.4	46.09±2.2	55.16±0.1	62.67±1.0	74.39±1.4	77.67±2.7	54.38±2.7	45.65±3.3	46.87±0.1	49.19±0.5	**78.02±1.5**

hance clustering performance in the target sequential data, and can achieve the state-of-the-art results.

- Exemplar-based Subspace Clustering (ESC) [41], which is exemplar-based subspace clustering method to tackle the problem of large-scale and imbalanced datasets. We use its ESC-FFS version in this paper.

Offline multi-task clustering model:

- Multi-Task Clustering with Model Relation Learning (MTCMRL) [42], which is a multi-task clustering method via automatically learning the model parameters relatedness between each pair of tasks.

Online subspace clustering models:

- Online Robust PCA (ORPCA) [6], which is based on stochastic optimization of batch PCA. We use $R_0 R_0^T$ (R_0 is the row space of $Z_0 = L_0 \Sigma_0 R_0^T$, and Z_0 is the clean matrix recovered by ORPCA) as the similarity matrix to perform clustering.
- Online Low-Rank Subspace Clustering (OLRSC) [27], which is a novel online implementation of LRR [22], and aims to reduce the memory cost.

For those competing methods above, we compare with them utilizing the source codes given by the authors. For the evaluation, we adopt two performance measures: accuracy (ACC) and normalized mutual information (NMI). The bigger the value of ACC and NMI is, the better the clustering performance of the corresponding model will be. Besides, all the models are implemented in Matlab, and the parameters of all the competing models are tuned in $\{10^{-3} \times i\}_{i=1}^{10} \cup \{10^{-2} \times i\}_{i=2}^{10} \cup \{10^{-1} \times i\}_{i=2}^{10} \cup \{2 \times i\}_{i=1}^{10} \cup \{40 \times i\}_{i=1}^{20}$.

5.2. Real-world Human Motion Datasets

In this section, we use three human motion datasets to evaluate our proposed model. More specifically, the introduction of these three datasets are as follows:

Keck Gesture dataset (Keck) [11] consists of 14 different actions from military signals, where each gesture can be performed by four subjects, and the same gesture is repeated three times by each person. Therefore, the total number of video sequences is $4 \times 3 \times 14 = 168$. The frame samples are shown in Figure 2 (Left). To this end, we have 4 tasks in total by treating each person as a motion segmentation task.

Multi-model Action Detection dataset (MAD) [10] is composed of 40 sequences captured from 20 subjects (i.e., 2 sequences per subject). Each subject performs 35 activities (e.g., *Jumping*, *Throw*, *Dribble*, and *kicking*) continuously. In this experiment, we randomly select 5 subjects, and treat each subject as one task. Frame samples of this dataset are shown in Figure 2 (Middle).

Electromyography-Vision dataset (EV-Action) dataset [35] is a novel multi-modality action dataset collected by ourselves. It contains 20 actions preformed by 50 subjects with 4 different modalities (e.g., RGB, EMG, Depth, Skeleton) simultaneously which could provide comprehensive information for human motion related research. The RGB and depth data have the resolution of 1080×1920 and 424×512 respectively. In this paper, we also randomly select 5 subjects, and treat each subject as one task. Frame samples are illustrated in Figure 2 (Right).

For each used dataset, we extract low-level HoG features [3] and obtain a 324-dimension feature vector from each frame. Due to the fact that different datasets contain different number of motions/actions, we randomly choose 10 motions performed by the same task from each dataset. The parameters λ_1 and λ_2 are empirically set to $0.1, 1000$, while the input batch size for our **Algorithm 1** is 80 in this paper.

The experimental results averaged over ten random repetitions are provided in Table 1, 2, 3 and Figure 3. From the obtained results, we can notice that: **1)** From Table 1 and Table 3, we can find that our model achieves the best performance among these competing models on the Keck and

Table 2. Segmentation comparisons between our model and state-of-the-arts in terms of ACC and NMI on MAD dataset: mean and standard errors over ten random runs. Models with the best performance are bolded, and the runner-up are underlined. M denotes MAD dataset.

MAD	Criteria	SPE[25]	LRR[22]	OSC[30]	SoftS3C[17]	TSC[21]	TSS(M)[32]	ESC-FFS[41]	MTCMRL[42]	OLRSC[6]	ORPCA[27]	Ours
Task 1	ACC(%)	35.18±0.3	31.81±0.6	40.03±2.2	35.58±2.8	52.39±0.6	53.83±0.7	53.60±3.8	40.67±1.0	43.02±2.6	45.74±3.3	**59.66±0.3**
	NMI(%)	43.75±0.8	35.76±0.8	51.73±1.1	51.07±0.6	74.55±0.3	**81.96±0.1**	56.46±2.8	39.53±2.5	61.25±0.5	69.58±0.9	78.82±0.7
Task 2	ACC(%)	34.60±1.5	33.11±0.1	37.31±0.9	40.36±1.5	52.38±1.3	56.59±0.7	47.45±1.5	39.54±1.6	43.78±2.4	47.38±0.4	**64.51±2.8**
	NMI(%)	43.94±0.8	36.95±0.4	53.11±0.1	51.94±0.6	74.34±0.5	82.37±0.3	55.42±1.3	35.47±1.9	58.44±1.0	71.55±0.5	**83.59±1.9**
Task 3	ACC(%)	31.35±0.4	35.97±3.2	39.51±3.0	40.68±0.3	47.56±0.5	59.22±0.5	42.82±2.1	39.18±1.1	42.11±2.7	45.65±1.9	**63.65±2.2**
	NMI(%)	36.97±0.7	36.67±2.5	52.15±0.1	53.05±0.6	73.18±0.3	**83.79±0.1**	49.18±1.6	40.54±2.7	57.47±1.7	63.79±0.7	82.69±1.2
Task 4	ACC(%)	30.96±1.4	32.96±0.6	40.66±1.6	37.82±0.1	53.15±0.2	53.99±0.6	45.29±3.1	42.80±2.1	44.97±1.6	53.37±0.3	**65.97±1.8**
	NMI(%)	33.49±1.3	34.10±1.0	47.82±0.1	41.43±0.5	74.09±0.8	**79.46±0.4**	51.61±2.9	36.78±1.7	58.60±0.2	71.29±0.9	78.15±0.3
Task 5	ACC(%)	46.96±2.8	45.26±4.2	51.38±0.3	46.79±2.6	51.74±1.2	55.88±0.7	57.34±3.5	54.76±2.2	45.12±2.5	54.37±0.4	**61.06±0.9**
	NMI(%)	47.27±1.2	44.66±1.8	60.51±0.1	55.95±2.2	73.02±0.5	79.56±0.1	61.96±2.8	48.34±3.6	56.56±0.6	69.36±0.1	**80.72±1.6**
Average	ACC(%)	35.81±1.3	35.82±1.7	41.77±1.6	40.25±1.6	51.45±0.7	55.90±0.6	49.30±2.8	40.55±1.5	43.80±2.4	49.30±1.2	**62.67±1.8**
	NMI(%)	41.09±0.9	37.63±1.6	53.06±0.2	50.69±1.1	73.84±0.5	**81.43±0.2**	52.45±2.4	38.08±2.2	58.47±0.8	69.12±0.6	80.79±1.0

Table 3. Segmentation comparisons between our model and state-of-the-arts in terms of ACC and NMI on our dataset: mean and standard errors over ten random runs. Models with the best performance are bolded, and the runner-up are underlined. M denotes MAD dataset.

EV-Action	Criteria	SPE[25]	LRR[22]	OSC[30]	SoftS3C[17]	TSC[21]	TSS(M)[32]	ESC-FFS[41]	MTCMRL[42]	OLRSC[6]	ORPCA[27]	Ours
Task 1	ACC(%)	42.54± 2.0	44.19±3.6	49.06±1.6	47.87±1.2	51.24±2.9	48.08±2.2	44.70±1.6	49.41±2.3	45.41±2.1	43.87±3.8	**67.88±3.1**
	NMI(%)	49.81±2.4	52.73±3.2	59.78±2.5	53.49±1.4	74.26±0.8	77.12±0.1	51.81±3.1	49.64±1.7	57.75±0.8	65.83±1.2	**81.52±1.8**
Task 2	ACC(%)	49.01±3.0	38.55±2.5	46.31±3.2	52.22±4.4	55.21±3.8	56.73±2.9	50.49±2.9	52.89±3.2	46.55±4.6	56.40±4.9	**66.99±3.2**
	NMI(%)	60.37±2.6	52.30±3.4	62.26±0.1	66.21±0.8	79.36±1.9	80.87±0.9	50.26±2.4	54.21±2.8	59.56±1.2	71.12±0.7	**81.51±1.8**
Task 3	ACC(%)	48.11±2.5	41.63±3.5	43.22±4.0	52.87±0.6	55.01±2.5	48.58±1.7	57.00±2.3	53.34±3.3	47.64±5.2	53.61±5.2	**60.41±2.9**
	NMI(%)	56.86±1.0	47.62±3.2	65.57±0.5	59.41±0.3	**78.47±0.6**	78.22±1.1	61.17±1.6	52.57±2.2	60.66±1.3	77.25±1.3	77.71±0.2
Task 4	ACC(%)	39.22±3.5	35.78±0.9	45.93±3.5	38.32±2.2	49.96±1.4	47.82±3.6	53.83±2.2	51.85±0.5	42.12±6.9	54.70±3.9	**63.36±3.2**
	NMI(%)	53.36±2.6	53.42±1.4	64.84±0.1	59.86±0.4	79.12±0.6	81.55±0.3	57.62±1.2	52.66±1.4	55.21±2.5	75.01±2.2	**82.35±1.5**
Task 5	ACC(%)	43.35±3.4	43.45±3.2	44.29±4.2	45.86±2.4	52.29±1.6	52.91±3.1	51.77±5.6	57.72±3.4	48.52±1.0	53.94±2.2	**65.09±3.4**
	NMI(%)	52.36±2.1	50.05±1.4	58.04±0.1	61.89±1.3	79.18±2.4	78.90±0.9	58.20±2.9	56.19±3.1	58.22±1.8	72.07±0.8	**79.63±0.2**
Average	ACC(%)	44.45±3.7	40.72±2.7	45.76±3.7	47.42±2.6	52.74±2.5	50.82±2.7	51.56±2.9	51.87±2.3	46.05±3.9	52.51±3.9	**64.75±3.6**
	NMI(%)	54.55±2.1	51.23±2.5	62.10±0.6	60.17±0.8	78.08±1.3	79.33±0.7	53.44±1.7	52.27±2.0	58.28±1.5	72.05±1.0	**80.54±1.1**

EV-Action datasets. Specifically, our model has 8.24% and 0.35% improvement in terms of ACC and NMI for Keck dataset, and 12.01% and 1.21% improvement in terms of ACC and NMI for EV-Action dataset, respectively; From Table 2, notice that our model can obtain 6.77% in ACC on the MAD datasets, respectively. It is because that (1) we adopt the autoencoder structure in the subspace clustering, which not only contains the motion-aware latent space, but also adopts encoder and decoder to link the motion-aware latent space with original data; (2) the useful information among different tasks can be fully used to segment these motions/actions. **2)** For the NMI on the MAD dataset, our model can just achieve the runner-up performance when comparing with TSS [32] model. The main reason for this observation is that our model is based on the online learning algorithm, i.e., we can just achieve a batch of motions in each time stamp and notice some local temporal information on the coming actions. **3)** The reason why our model outperforms the multi-task clustering model MTCMRL is that even though MTCMRL incorporates the within-task clustering and cross-task relatedness learning, it do not consider the temporal information. **4)** From the presented results in Figure 3 (different motions are marked as different colors), we can notice that the competing methods can not achieve useful segment performance when they do not consider the temporal information, e.g., SPE, LRR, MTCMRL and OLRSC. Moreover, even though our model just adopts the local temporal graph information, our model can also achieve clearer sequential structure. It is because that we use the motion-aware space with multiple tasks.

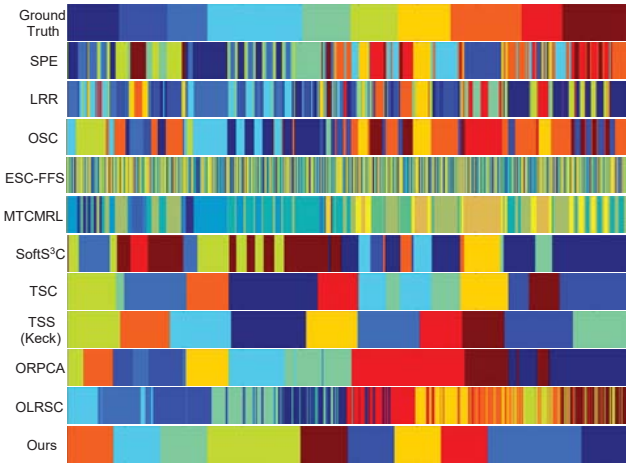

Figure 3. Visualization demonstration of clustering result on MAD (Subject 2) data. Ten motions are denoted by ten different colors.

5.3. Discussions

Time Consumption Generally, one of the major advantage of our model is the time consumption when comparing with offline subspace clustering methods. In order to investigate the runtime of our model, we utilize the Keck dataset by selecting the action dataset of the first task. The runtimes are presented in Figure 4, where we also include the time cost of spectral clustering or k-means. The size of each input batch for our model is 80. Notice that our model significantly outperforms most offline single-task cluster-

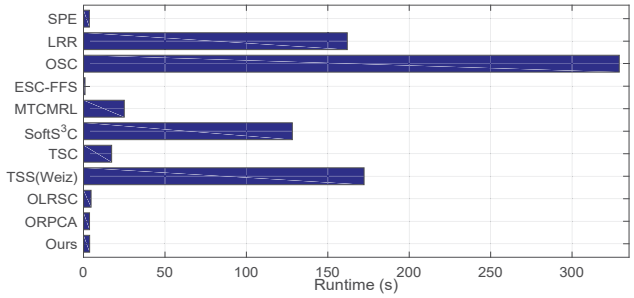

Figure 4. The demonstration of computational time (seconds) among the ten competing clustering models, where each bar denotes different competing models.

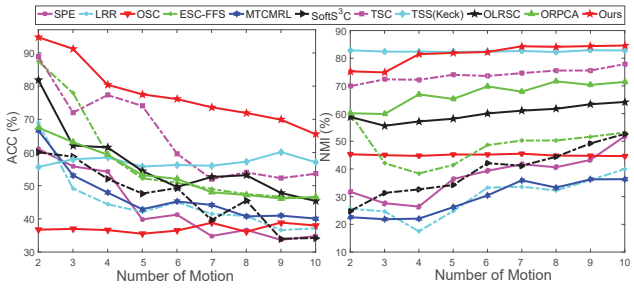

Figure 5. Clustering results: ACC (Left) and NMI (Right) on MAD dataset with different number of motion.

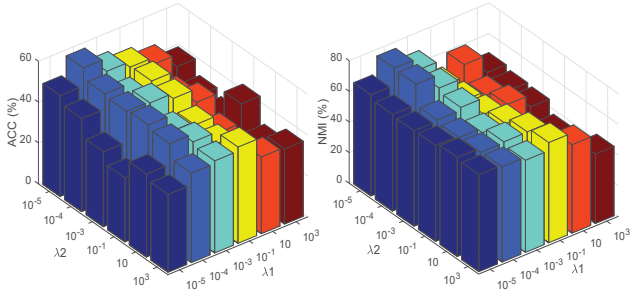

Figure 6. The effect of parameters λ_1 and λ_2 on Keck dataset in terms of ACC (left) and NMI (right), respectively.

ing methods, such as LRR and TSS, and offline multi-task clustering model, MTCMRL. Besides, our model can also achieve comparable runtime when comparing with other two online methods, i.e., OLRSC and ORPCA. However, our model always identifies more correct samples as well as consumes comparable running time. All the experiments are executed on the computer with 8G RAM, Intel i7 CPU.

Effect of Motion Number By removing some motions and set the motion number of MAD video from 2 to 10, we justify the effect of motion number on all competing models. As depicted in Figure 5, we can observe that our model can consistently achieves much better results than other models. In addition, the performance of our model is more stably when the number of motions is changing. This observation also demonstrates the effectiveness of our model.

Parameter Investigation In order to study how the parameters λ_1 and λ_2 affect the clustering performance of our model, we use Keck dataset in this subsection. By varying the parameters λ_1 and λ_2 in range $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-1}, 10^{1}, 10^{3}\}$, we present the clustering performance of our model in terms of ACC and NMI in Figure 6. Notice that the performance of both ACC and NMI change with different values of parameters, which verify that the appropriate temporal regularizer can make the clustering performance of our model better.

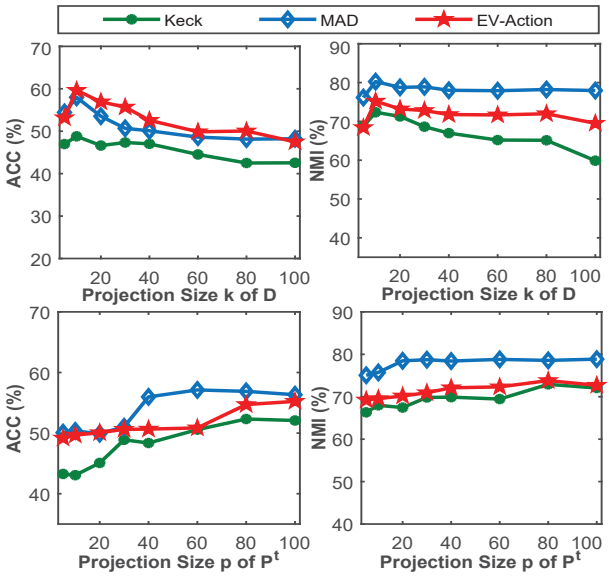

Figure 7. The effect of projection size k and p of D and P^t in terms of ACC and NMI, respectively, where different lines denote the clustering results on different datasets.

In addition, we also show the clustering performance on these three datasets when the projection sizes of motion-aware matrix D and projections P^t vary in range $\{5, 10, 20, 30, 40, 60, 80, 100\}$. As the results shown in Figure 7, our model can achieve better results when the size of motion-aware matrix approaches the motion number. Whereas the result of our model is relatively stable when the size p of projection P^t is larger than 40. This observation also indicates that our proposed model can get an accurate result when $p \geq 40$, and k around 10.

6. Conclusion

In this paper, we propose an online multi-task clustering model for multi-agent human motion segmentation task by integrating a linear encoder-decoder framework. The encoder-decoder paradigm can learn a latent motion-aware space via maximizing both the reconfigurability of the original motion data from the motion-aware space and the predictability of the motion-aware space from the original motion data. To address cross-task problem, projection for each task is jointly learned to mitigate the distribution divergences. We also derive a general online learning algorithm for dynamic environments. Experiment results show the effectiveness and efficiency of our model.

References

[1] Richard H. Bartels and George W Stewart. Solution of the matrix equation ax+ xb= c [f4]. *Communications of the ACM*, 15(9):820–826, 1972. 4

[2] Yang Cong, Gan Sun, Ji Liu, Haibin Yu, and Jiebo Luo. User attribute discovery with missing labels. *Pattern Recognition*, 73:33–46, 2018. 2

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. IEEE, 2005. 6

[4] Jiahua Dong, Yang Cong, Gan Sun, and Dongdong Hou. Semantic-transferable weakly-supervised endoscopic lesions segmentation. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2

[5] Ehsan Elhamifar and René Vidal. Sparse subspace clustering. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 2790–2797. IEEE, 2009. 2

[6] Jiashi Feng, Huan Xu, and Shuicheng Yan. Online robust pca via stochastic optimization. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, pages 404–412, 2013. 2, 6, 7

[7] Behnam Gholami and Vladimir Pavlovic. Probabilistic temporal subspace clustering. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 3066–3075, 2017. 1

[8] Samitha Herath, Mehrtash Tafazzoli Harandi, and Fatih Porikli. Learning an invariant hilbert space for domain adaptation. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 3956–3965, 2017. 2

[9] Minh Hoai and Fernando De la Torre. Maximum margin temporal clustering. In *Artificial Intelligence and Statistics*, pages 520–528, 2012. 2

[10] Dong Huang, Shitong Yao, Yi Wang, and Fernando De La Torre. Sequential max-margin event detectors. In *Proc. European conference on computer vision (ECCV)*, pages 410–424. Springer, 2014. 6

[11] Zhuolin Jiang, Zhe Lin, and Larry Davis. Recognizing human actions by learning and matching shape-motion prototype trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):533–547, 2012. 6

[12] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Sigmod Record*, 30(2):151–162, 2001. 1

[13] Eamonn Keogh and Shruti Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and knowledge discovery*, 7(4):349–371, 2003. 1

[14] Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning. *arXiv preprint arXiv:1704.08345*, 2017. 3

[15] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 706–715, 2017. 1

[16] Chun-Guang Li and Rene Vidal. Structured sparse subspace clustering: A unified optimization framework. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 277–286, 2015. 5

[17] Chun-Guang Li, Chong You, and René Vidal. Structured sparse subspace clustering: A joint affinity learning and subspace clustering framework. *IEEE Transactions on Image Processing*, 26(6):2988–3001, 2017. 1, 5, 6, 7

[18] Jun Li and Hongfu Liu. Projective low-rank subspace clustering via learning deep encoder. In *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. 2

[19] Jingjing Li, Ke Lu, Zi Huang, Lei Zhu, and Heng Tao Shen. Transfer independently together: A generalized framework for domain adaptation. *IEEE Transactions on Cybernetics*, 2018. 2

[20] Jun Li and Handong Zhao. Large-scale subspace clustering by fast regression coding. In *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. 2

[21] Sheng Li, Kang Li, and Yun Fu. Temporal subspace clustering for human motion segmentation. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 4453–4461, 2015. 1, 2, 3, 5, 6, 7

[22] Guangcan Liu, Zhouchen Lin, and Yong Yu. Robust subspace segmentation by low-rank representation. In *Proc. International Conference on Machine Learning (ICML)*, pages 663–670, 2010. 2, 5, 6, 7

[23] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 2200–2207, 2013. 2

[24] Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. Robust and efficient subspace segmentation via least squares regression. In *Proc. European Conference on Computer Vision (ECCV)*, pages 347–360. Springer, 2012. 2

[25] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Proc. Advances in neural information processing systems (NeurIPS)*, pages 849–856, 2002. 4, 5, 6, 7

[26] Bo Peng, Jianjun Lei, Huazhu Fu, Changqing Zhang, Tat-Seng Chua, and Xuelong Li. Unsupervised video action clustering via motion-scene interaction constraint. *IEEE Transactions on Circuits and Systems for Video Technology*, 2018. 2

[27] Jie Shen, Ping Li, and Huan Xu. Online low-rank subspace clustering by basis dictionary pursuit. In *Proc. International Conference on Machine Learning (ICML)*, pages 622–631, 2016. 2, 6, 7

[28] Gan Sun, Yang Cong, Dongdong Hou, Huijie Fan, Xiaowei Xu, and Haibin Yu. Joint household characteristic prediction via smart meter data. *IEEE Transactions on Smart Grid*, 10(2):1834–1844, 2017. 2

[29] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Movieqa: Understanding stories in movies through question-answering. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 4631–4640, 2016. 1

[30] Stephen Tierney, Junbin Gao, and Yi Guo. Subspace clustering for sequential data. In *Proc. IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1019–1026, 2014. 5, 6, 7

[31] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 5018–5027, 2017. 2

[32] Lichen Wang and Zhengming Ding. Learning transferable subspace for human motion segmentation. In *Proc. Association for the Advancement of Artificial Intelligence (AAAI)*, 2018. 1, 5, 6, 7

[33] Lichen Wang, Zhengming Ding, and Yun Fu. Low-rank transfer human motion segmentation. *IEEE Transactions on Image Processing*, 28(2):1023–1034, 2018. 1

[34] Lichen Wang, Zhengming Ding, Zhiqiang Tao, Yunyu Liu, and Yun Fu. Generative multi-view human action recognition. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2019. 1

[35] Lichen Wang, Bin Sun, Joseph Robinson, Taotao Jing, and Yun Fu. Ev-action: Electromyography-vision multi-modal action dataset. *arXiv preprint arXiv:1904.12602*, 2019. 6

[36] Qianqian Wang, Zhengming Ding, Zhiqiang Tao, Quanxue Gao, and Yun Fu. Partial multi-view clustering via consistent gan. In *Proc. IEEE International Conference on Data Mining (ICDM)*, pages 1290–1295. IEEE, 2018. 2

[37] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning for video understanding. *arXiv preprint arXiv:1712.04851*, 2017. 1

[38] Yimin Xiong and Dit-Yan Yeung. Mixtures of arma models for model-based time series clustering. In *Proc. IEEE International Conference on Data Mining (ICDM)*, pages 717–720. IEEE, 2002. 1

[39] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 5288–5296, 2016. 1

[40] Xiaoqiang Yan, Shizhe Hu, and Yangdong Ye. Multi-task clustering of human actions by sharing information. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6401–6409, 2017. 2

[41] Chong You, Chi Li, Daniel P Robinson, and René Vidal. A scalable exemplar-based subspace clustering algorithm for class-imbalanced data. In *Proc. European Conference on Computer Vision (ECCV)*, pages 68–85. Springer, 2018. 6, 7

[42] Xiaotong Zhang, Xianchao Zhang, Han Liu, and Jiebo Luo. Multi-task clustering with model relation learning. In *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3132–3140, 2018. 2, 6, 7

[43] Feng Zhou, Fernando De la Torre, and Jessica K Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(3):582–596, 2013. 2

[44] Yizhou Zhou, Xiaoyan Sun, Zheng-Jun Zha, and Wenjun Zeng. Mict: Mixed 3d/2d convolutional tube for human action recognition. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, pages 449–458, 2018. 1