Deep Domain Adaptation in Action Space

Arshad Jamal[1]
arshad@iitk.ac.in
Vinay P Namboodiri[2]
vinaypn@iitk.ac.in
Dipti Deodhare[1]
dipti@cair.drdo.in
KS Venkatesh[2]
venkats@iitk.ac.in

[1] Centre for AI & Robotics
Bangalore, India

[2] Indian Institute of Technology,
Kanpur, India

Abstract

In the general settings of supervised learning, human action recognition has been a widely studied topic. The classifiers learned in this setting assume that the training and test data have been sampled from the same underlying probability distribution. However, in most of the practical scenarios, this assumption is not true, resulting in a sub-optimal performance of the classifiers. This problem, referred to as Domain Shift, has been extensively studied, but mostly for image/object classification task. In this paper, we investigate the problem of Domain Shift in action videos, an area that has remained under-explored, and propose two new approaches named Action Modeling on Latent Subspace (AMLS) and Deep Adversarial Action Adaptation (DAAA). In the AMLS approach, the action videos in the target domain are modeled as a sequence of points on a latent subspace and adaptive kernels are successively learned between the source domain point and the sequence of target domain points on the manifold. In the DAAA approach, an end-to-end adversarial learning framework is proposed to align the two domains. The action adaptation experiments were conducted using various combinations of multi-domain action datasets, including six common classes of Olympic Sports and UCF50 datasets and all classes of KTH, MSR and our own SonyCam datasets. In this paper, we have achieved consistent improvements over chosen baselines and obtained some state-of-the-art results for the datasets.

1 Introduction

Today, surveillance cameras are everywhere, be it city streets, market place, buildings or airports. These cameras operate 24×7, generating a massive amount of video data that needs to be processed for autonomous understanding of events and activities occurring in the scene. Recently, deep networks have shown remarkable progress in the area of event/activity recognition [4, 7, 10, 29, 30]. Most of these methods use a learning paradigm, where a large set of labeled or unlabeled data is used to learn a model of the underlying process and then use the same model to classify the unknown data. The learning paradigm works well when the training and test data are sampled from the same statistical distribution.

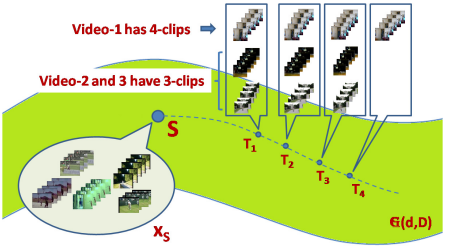


Figure 1: Concept of the proposed Action Modeling on Latent Subspace (AMLS) approach. For all the clips in the source domain, a subspace representation, shown as point $\mathbf{S} \in \mathbb{R}^{D\times d}$ on a Grassmann manifold $\mathcal{G}(d,D)$, is obtained using the Principal Component Analysis (PCA) method. Similarly, the target domain action videos are modeled as a sequence of points ($\mathbf{T}_i \in \mathbb{R}^{D\times d}$) on the manifold, where each point corresponds to a collection of features from each video as shown in the figure. In the end, adaptive kernels are learned between the successive points, which are then used for action domain adaptation.

However, in many practical scenarios, the semantically similar data captured in different contexts with different cameras introduces a shift in the statistical distribution. This makes the models learned using the data captured in one context to be less effective when tested on a data captured in another context. While this problem of *Domain Shift* has been an active area of research in recent years, most applications in vision have been restricted to recognition in images/objects [11, 12, 15, 21, 22]. One of the popular approaches for the image Domain Adaptation (DA) is based on the subspace modeling of the source and target domains [11, 14, 15], which works under the assumption that the two domains share a latent subspace where the domain shift can be reduced or eliminated. In this work, we seek to extend such ideas to events/actions for unsupervised domain adaptation in videos.

The videos, which are in general much more higher dimensional than the images, exhibit a larger degree of subspace-based dimensionality reduction due to its temporal smoothness. Thus, it is natural to model the video data on a subspace and perform domain adaptation in the latent space. In this paper, we use this idea to investigate Domain Adaptation (DA) in videos, an area that has remained under-explored. We propose two solutions to the action adaptation problem. First, we leverage the concept of image DA methods proposed in [8, 17] and formulate the problem as a sequence of adaptation on the latent subspace, as shown in Fig 1. There are two main motivations behind modeling the action video as a sequence of points on the subspace. First, it represents the temporal dynamics of the actions in a better way and second, it reduces the overall domain discrepancy between source and target domains, which has been empirically shown using the *Symmetrized KL Divergence* measure [15]. Together, they both enable better adaptation.

The proposed action adaptation method, named Action Modeling on Latent Subspace (AMLS) has been illustrated in Fig 1. The source and target videos are first segmented into fixed size action clips and then feature vectors are computed using the deep learning based 3D-CNN [29] method for each clip. The source domain feature vectors are projected as a single point on the latent subspace and the target domain features are represented as a sequence of points on the same subspace. Here, each point corresponding to the target domain data represents the collection of time aligned 3D-CNN features (see Fig 1) i.e. the i^{th} subspace point is learned using the i^{th} feature vectors from all the target videos. The

sequence of subspace points are learned using the incremental subspace learning approach [25] and a Geodesic path is constructed between the source point and sequence of target points. Adaptive kernels (transformation matrices) are computed between the source and sequence of target points. The kernels, in conjunction with the subspace based methods, such as [11, 15], are used to perform domain adaptation.

Further, we propose an end-to-end deep learning framework, named Deep Adversarial Action Adaptation (DAAA), that extend the idea of action modeling on the latent subspace, discussed above, to learn a domain invariant feature embedding for the action videos. This is achieved by training the network with mini-batch of target video clips corresponding to the sequence of points on the latent subspace, in the same temporal order. The sequence of clips in each action video are used together to ensure the learned embeddings are closed to each other in the latent space. Empirically, we show that the temporally ordered training with time-indexed samples results in improved adaptation performance.

2 Related Work

In a recent survey paper [5], domain adaptation and transfer learning techniques have been comprehensively discussed with a specific view on visual applications. It covers the historical shallow methods, homogeneous and heterogeneous domain adaptation methods and the deep domain adaptation methods that integrate the adaptation within the deep architecture. There are both semi-supervised and unsupervised domain adaptation methods. However, we mainly focus on the more challenging unsupervised domain adaptation approaches.

In the unsupervised domain adaptation approaches [2, 3, 11, 14, 28] etc., the main idea is to learn a domain invariant representation for both source and target domain in such a way that their distribution becomes as similar as possible. This is achieved by either a data sample re-weighting/selection based approach [2, 14] or subspace based approach [11, 16]. In the former, a linear combination of the source data samples is used to modify their distribution and bring it closer to the target distribution. In the subspace based approach [11, 16], a linear transformation is learned to align the source and target distribution.

Recently, deep domain adaptation methods [12, 20, 21, 22, 26] have shown significant performance gains over the prior shallow transfer learning methods. Many of these methods learn a feature representation in a latent space shared by the source and target domains. A popular approach among them is to minimize Maximum Mean Discrepancy (MMD) or its variant to effectively align the two distributions. In Deep Adaptation Network (DAN) [20], Multi-Kernel MMD is used to improve the transferability of the features from source to target domain. In Residual Transfer Network (RTN) [21], the assumption of shared classifier between source and target domain is relaxed. It combines MK-MMD with an adaptive classifier to further improve the performance. The classifier is adapted by learning a residual function with reference to the target classifier. In joint adaptation networks (JAN) [22], Joint-MMD (JMMD) is used to align the joint distributions of multiple domain-specific layers across two domains.

All the DA techniques found in the literature address the image/object classification problem. In fact, we could hardly find any work on the *video-to-video* domain adaptation problem. There are few studies [6, 19, 31] on cross-view action recognition and a few on heterogeneous domain adaptation [8, 9]. In that sense, to the best of our knowledge, this paper is one of the first few papers for the video-video domain adaptation.

3 Our Proposed Approach for Action DA

In this paper, we propose two solutions to the action adaptation problem. In our first approach, we extend the subspace based image/object DA methods (*e.g.* Subspace Alignment (SA) [11] and Geodesic Flow Kernel (GFK) [15]) for action space DA. We do this in conjunction with deep learning features (3D-CNN) [29]. This solution has been inspired by the image DA work in [8, 17]. In our second solution, we propose an end-to-end deep learning framework, in which target clips of each action videos are aligned together with the source domain video clips. This ensures that the feature embedding is close to each other on the latent subspace, enabling better adaptation. In both the approaches, we evaluate the segmented K-frame clips of source and target domain action videos. The details are given below.

In the AMLS approach, there are two main steps. First, we compute a subspace representation for the source and target data and then use the existing image adaptation methods such as SA or GFK to perform sequence of adaptations for the video clips. Since, there may not be sufficient data points available at every time instance (see Fig 1), the subspace representation is obtained using the incremental subspace learning method [25].

3.1 Action Modelling on Latent Subspace

Let us assume that the source domain consists of N_S labelled actions videos $\mathcal{D}_\mathcal{S} = \{x^i_\mathcal{S}\}_{i=1}^{N_S}$, where each action video is segmented into L_i clips of size K-frame. These clips are then mapped to D-dimensional feature vectors using the 3D-CNN model [29]. The features are then stacked together to form a matrix $\mathbf{V}_\mathcal{S} \in \mathbb{R}^{D\times N}$, where $N(=\sum_{i=1}^{N_S} L_i)$ is the total number of video clips in source domain. A d-dimensional subspace representation, shown as point $\mathbf{S}$ in Fig 1, is learned using the PCA of the matrix $\mathbf{V}_\mathcal{S}$, where $d << D$ and the subspace point is an orthogonal matrix $\mathbf{S} \in \mathbb{R}^{D\times d}$.

Similarly, let us assume that the target domain has N_T unlabelled action videos, $\mathcal{D}_\mathcal{T} = \{x^i_\mathcal{T}\}_{i=1}^{N_T}$ and each action video $x^i_\mathcal{T}$ is temporally segmented into M_i clips of K-frame, which are then converted into a sequence of feature vectors using the 3D-CNN model. The target features are stacked together according to their time index, giving a sequence of matrices $\mathbf{V}_\mathcal{T} = \{\mathbf{v_1}, \mathbf{v_2}, ..., \mathbf{v_M}\}$, where $\mathbf{v_i} \in \mathbb{R}^{D\times P_i}$, P_i is the total number of the 3D-CNN features at i^{th} time index (e.g. $P = \{3,3,3,1\}$ in Fig 1) and $M = max(M_i), i \in [1, N_T]$ is the maximum number of clips across all the target videos. Since, the number of clips in $\mathbf{v_m}$ may be very small in some cases (e.g. only one clip for point T_4 in Fig 1), direct computation using the PCA technique is not possible in such cases. We use the incremental subspace learning approach [25] to compute the sequence of target points on the subspace. Once, the source and target domain points are obtained, we align the source domain with the sequence of target domains, by minimizing the Frobenius norm of the difference between the projected data from the source and target domains [11], which is defined as:

$$\min_{\mathbf{W}_{\mathcal{S},m}, \mathbf{W}_{\mathcal{T},m}} J(\mathbf{W}_{\mathcal{S},m}, \mathbf{W}_{\mathcal{T},m}) = \|\mathbf{S}\mathbf{W}_{\mathcal{S},m} - \mathbf{T}_m\mathbf{W}_{\mathcal{T},m}\|_F, \tag{1}$$

where, $\mathbf{W}_{\mathcal{S},m}$ and $\mathbf{W}_{\mathcal{T},m}$ are the transformation matrices for the source and target points.

Formally, the sequence of target points $\mathbf{T}_m$ are obtained by minimizing the re-projection error of the feature vector, defined as,

$$E(\mathbf{v_m}, \mathbf{T}_m) = \left\|\mathbf{v_m} - \mathbf{T}_m\mathbf{T}'_m\mathbf{v_m}\right\|_F \tag{2}$$

It has been mentioned in [15] that two domains adapt well if they are close to each other on the subspace. In order to ensure this, a regularizer term $r(\mathbf{T}_{m-1}, \mathbf{T}_m)$ is added to Eq (2) above. So, the overall cost function to be minimized is as follows:

$$\min_{\mathbf{T}'_m\mathbf{T}_m=\mathbf{I},\mathbf{W}_{\mathcal{S},m},\mathbf{W}_{\mathcal{T},m}} r(\mathbf{T}_{m-1}, \mathbf{T}_m) + E(\mathbf{v_m}, \mathbf{T}_m) + J(\mathbf{W}_{\mathcal{S},m}, \mathbf{W}_{\mathcal{T},m}) \tag{3}$$

The cost function in Eq (3) is non-convex and it can be solved in two steps. In the first step, for given transformation matrices, $\mathbf{W}_{\mathcal{S},m-1}$ and $\mathbf{W}_{\mathcal{T},m-1}$, the cost function is minimized for $\mathbf{T}_m$ and in the second step, for a given $\mathbf{T}_m$, Eq (3) is solved for $\mathbf{W}_{\mathcal{S},m}$ and $\mathbf{W}_{\mathcal{T},m}$. The two steps are detailed below.

Step 1: When $\mathbf{W}_{\mathcal{S},m-1}$ and $\mathbf{W}_{\mathcal{T},m-1}$ are fixed, the cost function would be minimum for $\mathbf{T}_m = \mathbf{T}_{m-1}$. In this case, $J(\mathbf{W}_{\mathcal{S},m}, \mathbf{W}_{\mathcal{T},m})$ acts as a regularization term and ensures that the neighbouring subspace points $\mathbf{T}_{m-1}$ and $\mathbf{T}_m$ are close to each other. It also means that one can combine the first and third term, in Eq (3), into single regularizer of $\mathbf{T}_m$ and solve the resulting cost function as:

$$\begin{aligned} \min_{\mathbf{T}_m} \quad & r(\mathbf{T}_{m-1}, \mathbf{T}_m) + E(\mathbf{v_m}, \mathbf{T}_m) \\ \text{s.t} \quad & \mathbf{T}'_m\mathbf{T}_m = I \end{aligned} \tag{4}$$

Depending on the choice of the regularizer function, the cost function has two possible solutions, which are as follows:

- If the regularizer r is constant, the solution for the subspace point $\mathbf{T}_m$ would be the d largest singular vector of $\mathbf{v_m}$, which can be easily computed by taking the SVD, if there are sufficient number of data points. In our case, as we want to capture the temporal dynamics of the videos, there would not be enough data at all time instance (see Fig 1) to compute the subspace points directly. Nonetheless, if all the action clips are considered together without bothering about the temporal dynamics, the problem reduces to the standard image DA formulation. We consider this as one of the baseline for our study, in which all the target domain clips are represented as a single point on the subspace and then GFK or SA methods are applied for adaptation.

- If the regularizer r varies with time, which is our case, the solution is obtained using the incremental subspace learning approach [25], which in turn is based on the R-SVD [13] method. In this work, we use a variant of the efficient sequential Karhunen-Loeve algorithm [18] to obtain the sequence of subspace points. The details are available in [25].

Step 2: In the second step, when $\mathbf{T}_m$ is fixed, the minimization does not depend on the first and second term of Eq (3). So, we are left with $J(\mathbf{W}_{\mathcal{S},m}, \mathbf{W}_{\mathcal{T},m})$, which can be easily solved using the SA or GFK methods, resulting into our two AMLS variants i.e. AMLS_SA and AMLS_GFK. The pseudo code has been given in the supplementary material.

3.2 Deep Adversarial Action Adaptation

The proposed AMLS formulation cannot be directly extended to an end-to-end deep learning framework because of two reasons. First, the cost function in Eq (3) is not differentiable and second, obtaining a constraint in a sub-space based criterion would require analysis of the

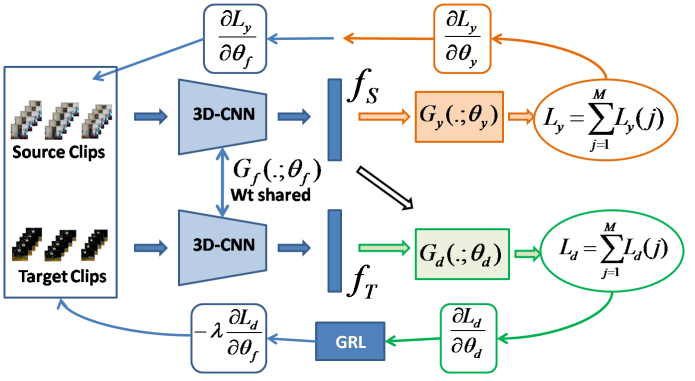


Figure 2: Architecture of the proposed Deep Adversarial Action Adaptation network. $G_f(.;\theta_f)$ is the feature mapping function, $G_y(.;\theta_y)$ is the class discriminator function and $G_d(.;\theta_d)$ is the domain discriminator function. L_y and L_d are the label and domain prediction losses. GRL is gradient reversal layer. All the M clips in an action video is simultaneously fed to the network and losses are accordingly computed. *Best viewed in color.*

whole dataset in each iteration. Hence, we adopt the next feasible approach to align the two domains using adversarial learning.

In this paper, we propose a new learning framework in which the target video clips are aligned with the source data as per their temporal ordering. Our network architecture named Deep Adversarial Action Adaptation (DAAA) has been shown in Fig 2 and it includes few layers of 3D-CNN for feature mapping, few discriminative layers for classification and adversarial layer for domain adaptation. In the network, the feature mapping layers share weight between source and target domains. In the DAAA method, an adversarial game is played between a domain discriminator $G_d(.;\theta_d)$, which is trained to distinguish the source and target domain samples, and the feature extractor $G_f(.;\theta_f)$, which is fine-tuned simultaneously to confuse the domain discriminator.

In the adversarial training, the parameters θ_f are learned by maximizing the domain discriminator loss L_d and the parameters θ_d are learned by minimizing the domain loss. In addition, the label prediction loss L_y is also minimized. The overall loss function for the DAAA is:

$$\begin{aligned} L(\theta_f,\theta_y,\theta_d) = & \frac{1}{N_s} \sum_{x_i \in \mathcal{D}_S} \sum_{j=1}^{M} L_y(G_y(G_f(x_i^j)), y_i^j) \\ & - \frac{\lambda}{N_S + N_T} \sum_{x_i \in \mathcal{D}_S \cup \mathcal{D}_T} \sum_{j=1}^{M} L_d(G_d(G_f(x_i^j)), d_i^j), \end{aligned} \tag{5}$$

where x_i^j is the j^{th} clip of i^{th} video and λ is a trade-off parameter between the two objectives that shape the features during learning. At the end of the training, the parameters $\hat{\theta}_f$, $\hat{\theta}_y$, $\hat{\theta}_d$ will give the saddle point of the loss function (5): $(\hat{\theta}_f, \hat{\theta}_y) = \min_{\theta_f,\theta_y} L(\theta_f,\theta_y,\theta_d)$ and $(\hat{\theta}_d) = \max_{\theta_d} L(\theta_f,\theta_y,\theta_d)$.

Table 1: Symmetrized KL Divergence for the **UO** and **KMS** datasets. The first two columns are for UO datasets and the last four columns are for the KMS datasets. Lower SKLD is better for the adaptation. In the first two cols, the subscript U, O depicts the base 3D-CNN model with U for UCF50 subset net and O for Olympic Sports subset net.

Methods	$(\mathbf{U} \rightarrow \mathbf{O})_U$	$(\mathbf{O} \rightarrow \mathbf{U})_O$	$\mathbf{K} \rightarrow \mathbf{M}$	$\mathbf{K} \rightarrow \mathbf{S}$	$\mathbf{M} \rightarrow \mathbf{S}$	$\mathbf{M} \rightarrow \mathbf{K}$
GFK/SA	0.901	0.207	0.159	0.061	0.089	0.1289
AMLS	0.262	0.064	0.092	0.05	0.07	0.066

3.3 Domain Discrepancy Measure

One of the motivation behind modeling the action videos as a sequence of points is to reduce the overall domain discrepancy. It has been shown in [11, 15] that two domains lend themselves well for the adaptation task if the domain discrepancy between them is lower. In [15], *Symmetrized KL Divergence (***SKLD***)* was proposed to measure the adaptability of two domains. Here, we use this measure to evaluate the effect of action modeling using sequence of points on a subspace.

Let $\mathbf{V}_{\mathcal{S}}$, $\mathbf{V}_{\mathcal{T}}$ be the features for the source and target datasets and $\mathbf{S}$, $\mathbf{T}$ be the basis of the two subspaces. We can define the **SKLD** between the source and target domain as $\frac{1}{d^*}\sum_i^{d^*}\theta_i\{KL(\mathcal{S}_i||\mathcal{T}_i)+KL(\mathcal{T}_i||\mathcal{S}_i)\}$, where d^* is the optimal dimensionality of the subspace and θ_i is the i^{th} principal angle. In order to obtain a closed form solution for the SKLD measure, we approximate $\mathcal{S}_i$ and $\mathcal{T}_i$ as two one-dimensional Gaussian distribution of $\mathbf{V}'_{\mathcal{S}}\mathbf{s}_i$ and $\mathbf{V}'_{\mathcal{T}}\mathbf{t}_i$ respectively with mean zero and variances $\sigma^2_{i\mathcal{S}}$ and $\sigma^2_{i\mathcal{T}}$. The principle angles θ_i between two subspaces are efficiently computed using the SVD of matrix $\mathbf{S}'\mathbf{T} = \mathbf{U}\Gamma\mathbf{V}'$ and they are $\theta_i = \arccos(\gamma_i)$, where γ_i is the i^{th} singular value in the diagonal matrix Γ. The principle vectors $\mathbf{s}_i = (\mathbf{SU})_{.,i}$ and $\mathbf{t}_i = (\mathbf{TV})_{.,i}$ are the i^{th} basis vector of the source and target points on the subspace.

The SKLD measure for the approximate domain distribution is.

$$\mathbf{SKLD}(\mathcal{S},\mathcal{T}) = \frac{1}{d^*}\sum_i^{d^*}\theta_i\left\{\frac{1}{2}\frac{\sigma^2_{i\mathcal{S}}}{\sigma^2_{i\mathcal{T}}}+\frac{1}{2}\frac{\sigma^2_{i\mathcal{T}}}{\sigma^2_{i\mathcal{S}}}-1\right\} \tag{6}$$

Further details have been given in the supplementary material.

4 Experiments

We evaluate the two proposed approaches against five baselines, including **Baseline-S**, **Baseline-T**, **3D-CNN**, **GFK_Action** and **SA_Action**. The last two baselines are the extension of image adaptation methods for action space. In **Baseline-S (T)**, we project the source and target data points on the subspace defined by the PCA of the source (target) data. The experimental setup, implementation details and results have been described below.

4.1 Setup

The DA experiments require multiple distinct action datasets having the same action categories. Unfortunately, there are hardly any benchmark action datasets available for this experiment. We specifically created three multi-domain datasets and evaluated the proposed

Table 2: Action Classification Accuracy (%) for the **UO** and **KSM** datasets. The best results are shown in bold. All the methods, except 3D-CNN and DAAA, use 4096-dimensional *fc7* features given by the fine-tuned 3D-CNN model and the classification is done using SVM. In DAAA method, end-to-end training is done, with softmax classifier. The subscript U, O depicts the base 3D CNN model i.e. UCF50 subset or Olympic Sports subset.

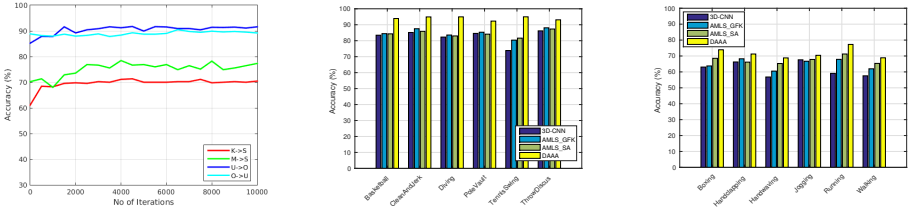
Methods	$(\mathbf{U} \rightarrow \mathbf{O})_U$	$(\mathbf{O} \rightarrow \mathbf{U})_O$	$\mathbf{K} \rightarrow \mathbf{M}$	$\mathbf{K} \rightarrow \mathbf{S}$	$\mathbf{M} \rightarrow \mathbf{S}$	$\mathbf{M} \rightarrow \mathbf{K}$
Baseline-S	80.72	83.1	57.89	63.01	72.13	70.11
Baseline-T	80.76	83.02	54.49	62.86	72.06	71.22
3D-CNN [29]	82.13	83.16	49.8	61.11	70.22	71.89
GFK_Action	84.04	86.21	61.16	63.71	73.27	72.9
AMLS_GFK (ours)	84.65	86.44	66.63	64.46	74.96	74.61
SA_Action	84.10	85.67	62.13	64.71	76.7	74.5
AMLS_SA (ours)	83.92	86.07	64.15	67.26	76.21	73.27
DAAA (ours)	**91.6**	**89.96**	**73.2**	**70.44**	**77.33**	**86.85**

approaches with them. Two of the datasets are discussed below and the third is included in the supplementary material.

UCF50 and Olympic Sports Datasets: In the first series of experiments, we use a subset of six common classes from UCF50 [24] and Olympic Sports [23] datasets (denoted by **U** for UCF50 subset and **O** for Olympic Sports subset). The classes are *Basketball, Clean and Jerk, Diving, Pole Vault, Tennis* and *Discus Throw*. For the UCF50 dataset, we use 70%-30% train-test split suggested in [27], which results into $432-168$ train/test action videos for training and testing. Similarly, for Olympic Sports dataset, the number of videos in training and testing set are 260 and 55 respectively. We fine-tune the publicly available Sport 1M 3D-CNN model using the two datasets independently. This gives us two different models, which are used in the two DA problems ($(\mathbf{U} \rightarrow \mathbf{O})_U$ and $(\mathbf{O} \rightarrow \mathbf{U})_O$). For the end-to-end training, we start with the fine-tuned model and train it for the alignment of two domains.

KTH, MSR Action II and Sonycam Datasets: In the second series of experiments, a combination of three datasets has been used, which is referred to as **KMS** dataset collection. It consists of two benchmark datasets i.e. KTH [1] and MSR Action II [1] (denoted by **K** and **M** respectively) along with our own six class SonyCam dataset (denoted by **S**) captured using a **hand-held** Sony camera. In KTH and SonyCam datasets, there are six classes, namely, *Boxing, Handclapping, Handwaving, Joging, Running* and *Walking*. In the MSR Action II dataset, only the first three classes from the KTH dataset are available. For the **KMS** dataset collection, there are four adaptation problems (K $\rightarrow$ M, K $\rightarrow$ S, M $\rightarrow$ S and M $\rightarrow$ K). The SonyCam dataset is only used as target domain owing to its small size (180 clips across 6 action classes). In case of KTH dataset, we use training data partition of 1530 clips spread almost equally across six classes for source domains and testing data partition of 760 clips for target domain. In the MSR dataset, there are 202 clips for three classes (Boxing-80, Handclapping-51 and Handwaving-71).

Implementation Details: For the feature embedding, we used the architecture given in [29]. All the subspace based domain adaptation experiments have been conducted using the 4096-dimensional *fc7* features computed for 16-frame clips, obtained by segmenting the action videos. The sequence of points on the subspace are obtained using the clip level deep features computed with the fine-tuned model. All the source domain video features are stacked together to form a matrix of dimension $N \times 4096$, where N is the total number



(a) (b) (c)

Figure 3: Action Adaptation Empirical Analysis: (a) Accuracy vs. Iterations; (b) $U \rightarrow O$: Accuracy vs Class; (c) $K \rightarrow S$: Accuracy vs Class. *Best viewed in color.*

of video clips in the source domain. The subspace embedding for the source domain is computed using the PCA of the matrix. For the target dataset, a sequence of points is obtained on the same subspace by two operations: (i) the features across all the videos are collected as per their time index; and (ii) the incremental subspace learning method [25] is used to find the sequence of subspace points. These target domain points on the subspace are then successively aligned to the source domain point using either GFK or SA method [11, 15], resulting into our two approaches i.e. AMLS_GFK and AMLS_SA. The final classification is performed using the SVM method, which predicts labels for each 16-frame clips.

In the case of deep adversarial action adaptation, we first obtain a base 3D-CNN model by fine-tuning the publicly available Sports 1M model using the source domain data. The domain adaptation layers are added to the feature mapping layers of the 3D-CNN. During the adaptation, the inputs to the network are provided according to the sequence of target points on the latent subspace. The training stops after several epochs are completed. The classification is done using the softmax layer. We fine-tune all the convolutional and pooling layers and train the classifier layer via back propagation. Since the classifier is trained from scratch, we set its learning rate to be 10 times that of the lower layers. We employ the mini-batch stochastic gradient descent (SGD) with momentum of 0.9 and the learning rate strategy implemented in RevGrad [12].

4.2 Results and Discussions

In this section, we first present the results for the Symmetrized KL Divergence (SKLD) and then discuss the performance of the AMLS algorithms and the deep action adaptation method.

Symmetrized KL Divergence (SKLD) Measure: We computed the SKLD values for both the datasets and found that the AMLS approach has consistently lower values when compared with the baseline adaptation method. In Table 1, the SKLD values have been given for the **UO** and **KMS** datasets. The results show that the action modeling on latent subspace reduces the domain discrepancy and hence gives better adaptation performance.

Domain Adaptation in Action Spaces: In this paper, we have evaluated our action domain adaptation approaches for **UO**, **KMS** and **HU** (results given in supplementary material) datasets. In majority of the cases, improvements have been observed over all the baselines. The two subspace based domain adaptation methods (GFK_Action and SA_Action) have been found to be generally better than the 3D-CNN method and other two baselines and the proposed AMLS approaches are better than all the five baselines. However, the deep action adaptation method outperforms all the other methods. For example, in the first four columns

of Table 2, it can be seen that the GFK_Action and SA_Action methods give better result than the top three rows, the AMLS method further improves the performance and the DAAA method is the best.

In the last four columns of Table 2, results for the **KMS** dataset has been given. In this case also, we observed that the subspace based adaptation methods were better than the three baselines shown in top three rows, which was further improved by the two AMLS methods. However, in all the cases, the DAAA approach was significantly better than all the other methods. These results show that the subspace based image adaptation methods also work for action adaptation and the proposed explicit modeling on a latent subspace further improves the results. The results also confirm the earlier finding [12] that the end-to-end adversarial learning do help in bridging the gap between source and target domains.

The upper bound for the adaptation algorithms are obtained by testing the classifiers trained on the same dataset. The accuracy obtained for UCF50 subset, Olympic Sports subset, KTH and MSR are 92.76%, 92.54%, 97.5% & 89.83% respectively. It can be observed in Table 2 that the performance of the deep adaptation method is close to the upper bound for the UO datasets. However, for the KTH and MSR datasets, there is still significant scope of improvement.

Analysis of Deep Adversarial Learning: The improvement in the accuracy of the DAAA over the course of training is shown in Fig 3(a). It includes results for four adaptation problems, two for each datasets. The training starts with a fine-tuned model and stops after 10000 iterations. The significant improvement in the accuracy is evident in the three of four cases shown in the figure. For $O \rightarrow U$ DA problem, the accuracy does not significantly change during the course of training. In Fig 3(b)&(c), we show the class wise classification performance of four methods for $U \rightarrow O$ and $K \rightarrow S$ adaptation problems. The figures show consistent improvement by the adaptation methods across all classes. The results of AMLS_GFK and AMLS_SA are better than the No Adaptation baseline but the DAAA method gives substantial improvement over the other methods.

5 Conclusions

In this paper, we formulated the problem of domain adaptation for human action recognition as a sequence of points on a smoothly varying latent subspace, capturing the temporal dynamics of the action videos. We proposed two solutions to the problem, including an end-to-end adversarial learning framework. In our experiments, we obtained consistent and significant performance improvements over various baselines. Particularly, the deep action adaptation method substantially outperformed all the other methods. Our experiments also validate that by embedding the domain-adaptation modules into 3D-CNN architecture, more transferable features can be learned. In future, we would like to study the concept of continuous domain adaptation on the streaming action videos. In addition, we would like to study other deep learning frameworks for action domain adaptation.

Acknowledgment

The authors would like to thank Director, Centre for AI & Robotics, Bengaluru, India for supporting the research.

References

[1] KTH and MSR action II dataset. URL http://www.cs.utexas.edu/~chaoyeh/web_action_data/dataset_list.html.

[2] Rahaf Aljundi, Rémi Emonet, Damien Muselet, and Marc Sebban. Landmarks-based kernelized subspace alignment for unsupervised domain adaptation. In *CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 56–63, 2015.

[3] Mahsa Baktashmotlagh, Mehrtash Tafazzoli Harandi, and Mathieu Salzmann. Learning domain invariant embeddings by matching distributions. In *Domain Adaptation in Computer Vision Applications*, Advances in Computer Vision and Pattern Recognition, pages 95–114. Springer, 2017.

[4] J Carreira and A Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR 2017*.

[5] Gabriela Csurka. A comprehensive survey on domain adaptation for visual applications. In *Domain Adaptation in Computer Vision Applications.*, pages 1–35. 2017.

[6] N Faraji Davar, T E deCampos, D Windridge, J Kittler, and W Christmas. Domain adaptation in the context of sport video action recognition. In *Domain Adaptation Workshop, in conjunction with NIPS*, 2011.

[7] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[8] Lixin Duan, Dong Xu, and Ivor W. Tsang. Domain adaptation from multiple sources: A domain-dependent regularization approach. *IEEE Transactions on Neural Networks and Learning Systems*, 23(3):504–518, March 2012.

[9] Lixin Duan, Dong Xu, Ivor W. Tsang, and Jiebo Luo. Visual event recognition in videos by learning from web data. *PAMI*, 34(9):1667–1680, September 2012.

[10] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. pages 1933–1941.

[11] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV 2013*, pages 2960–2967.

[12] Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML 2015, Lille, France, 6-11 July 2015*, pages 1180–1189, 2015.

[13] Gene H. Golub and Charles F. Van Loan. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996. ISBN 0-8018-5414-8.

[14] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML 2013, Atlanta, GA*, June 2013.

[15] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR 2012*, pages 2066–2073, 2012.

[16] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Unsupervised adaptation across domain shifts by generating intermediate data representations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(11):2288–2302, 2014.

[17] Judy Hoffman, Trevor Darrell, and Kate Saenko. Continuous manifold based adaptation for evolving visual domains. In *CVPR 2014*, pages 867–874, 2014.

[18] Avraham Levy and Michael Lindenbaum. Efficient sequential karhunen-loeve basis extraction. In *ICCV*, page 739, 2001.

[19] Ruonan Li. Discriminative virtual views for cross-view action recognition. In *CVPR 2012*, pages 2855–2862, 2012.

[20] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *ICML 2015*, pages 97–105, 2015.

[21] Mingsheng Long, Jianmin Wang, and Michael I. Jordan. Unsupervised domain adaptation with residual transfer networks. *CoRR*, abs/1602.04433, 2016. URL http://arxiv.org/abs/1602.04433.

[22] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 2208–2217, 2017.

[23] Juan Carlos Niebles, Chih-Wei Chen, and Li Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. 2010.

[24] Kishore K. Reddy and Mubarak Shah. Recognizing 50 human action categories of web videos. *Mach. Vision Appl.*, 24(5):971–981, July 2013.

[25] David A. Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. *Int. J. Computer. Vision*, 77(1-3):125–141, May 2008.

[26] Rui Shu, Hung H. Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-T approach to unsupervised domain adaptation. *CoRR*, abs/1802.08735, 2018. URL http://arxiv.org/abs/1802.08735.

[27] Waqas Sultani and Imran Saleemi. Human action recognition across datasets by foreground-weighted histogram decomposition. In *CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 764–771, 2014.

[28] Baochen Sun and Kate Saenko. Subspace distribution alignment for unsupervised domain adaptation. In *BMVC 2015, Swansea, UK, September 7-10, 2015*, pages 24.1–24.10, 2015.

[29] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 4489–4497, 2015.

[30] G Varol, I Laptev, and S Cordelia. Long-term temporal convolutions for action recognition. In *PAMI*, 2017.

[31] Zhong Zhang, Chunheng Wang, Baihua Xiao, Wen Zhou, Shuang Liu, and Cunzhao Shi. Cross-view action recognition via a continuous virtual path. In *CVPR 2013, Portland, OR, USA, June 23-28, 2013*, pages 2690–2697, 2013.