Video-Based Action Recognition Using Dimension Reduction of Deep Covariance Trajectories

Mengyu Dai and Anuj Srivastava

Department of Statistics, Florida State University

Abstract

Convolutional Neural Networks (CNNs) have proven very successful in extracting discriminative features from video data. These deep features can be summarized using spatial covariance descriptors for further analysis. However, due to large number of potential features, the covariance descriptors are often very high dimensional. To facilitate large scale data analysis, we propose a novel, metric-based dimension-reduction technique that reduces large covariances to small ones. Then, we represent videos as time series trajectories on the space of covariance matrices, or symmetric-positive definite matrices (SPDMs), and use a Riemannian metric on this space to quantify differences across these trajectories. These distance features can then be used for classification of video sequences. We illustrate this comprehensive framework using data from the UCF11 dataset for action recognition, with classification rates that match or outperform state-of-the-art techniques.

1. Introduction

Action recognition is an important area of research in computer vision. The goal here is to develop automated systems that can understand human actions common in daily life. Such systems should ideally be able to recognize and classify various human actions, and be applicable in areas such as surveillance systems, sports video analysis, healthcare systems, and human-computer interfaces.

CNN-based methods have led a tremendous progress in this area, especially in extracting powerful features from video frames [1, 2, 3, 4]. The deep features capture various characteristics of input images, and are further used for video classification or analysis. Since videos essentially form time-series data, the use of curves and trajectories parametrized by time to represent and quantitatively compare videos is natural [5, 6, 7, 8, 9]. Other approaches include a hierarchical clustering, multi-task learning method for joint human action grouping and recognition [10]; a deep progressive reinforcement learning (DPRL) method which aims to distill the most informative frames and discard ambiguous frames in sequences for recognizing actions [11]; and Long Short-Term Memory (LSTM) [12] related approaches that have shown great promise in action recognition tasks recently [13, 14, 15].

Despite a significant progress in recent years, it is still quite challenging to efficiently utilize high-dimensional features for comprehensive analysis of large-volume video data. One can summarize these features in the form of covariance descriptors. However, these covariances are often very high-dimensional. To handle this issue, we provide a metric-based, dimension-reduction method that helps reduce large covariance matrices into small ones, and significantly improves computational efficiency. Inspired by past works, we propose a framework that utilizes a trajectory-based representation of video clips on the space of symmetric positive-definite matrices (SPDM). Combined with dimension-reduction of SPDMs, this provides both performance and efficiency in the framework. The pipeline for the framework is shown in Fig 1 – we first extract features from video frames using existing CNN frameworks. Each frame is represented by a covariance matrix estimated from the extracted features and, thus, a video is represented as a sequence of covariance matrices. Next, we apply a dimension-reduction approach, based on the chosen Riemannian metric of SPDMs, to bring down dimensions of individual covariances, and then compare these reduced trajectories using a Riemmanian metric on the reduced space. Finally, the distance features acquired from these trajectories are used for video classification.

The novel aspects and contributions of this paper are as follows:

1. **Video Representation and Comparison with SPDM Trajectories Using Deep CNN Features**: We represent videos as SPDM trajectories constructed from ex-

tracted CNN features and compares these trajectories based on a Riemannian metric. The distance features from these trajectories are used for classification.

2. **Metric-Based SPDM Dimension Reduction**: We apply a metric-based unsupervised dimension-reduction from large SPDMs to small ones introduced in [16], making comparisons between covariance trajectories (with thousands of dimensions) practically feasible and significantly improving computational efficiency.

3. **Combining SPDM Trajectory Distance Features for Classification**: We combine distance vectors between trajectories, and use them as the last layer input for classification.

We evaluate the proposed framework on UCF11 dataset [17], and obtain results that outperform state-of-art methods.

2. Related Work

A large number of methods have been developed for video representation and classification. Here we focus on those methods that closely relate to our approach.

CNN-based: Karpathy et al. [18] studied the performance of convolutional neural networks in large-scale video classification. They found that CNN architectures are capable of learning powerful features from weakly-labeled data that surpass traditional feature-based methods in performance. Ji et al. [19] developd a 3D CNN model for automated recognition of human actions in surveillance videos. This model extracts features from both the spatial and the temporal dimensions by performing 3D convolutions, thereby capturing the motion information encoded in multiple adjacent frames. Zhou et al. [20] proposed the Mixed 2D/3D Convolutional Tube (MiCT) which enables 3D CNNs to extract deeper spatio-temporal features with fewer cost. Their deep network MiCT-Net based on the MiCT outperforms traditional 3D CNNs for action recognition in their experiments. Kong et al. [21] proposed a deep sequential context networks (DeepSCN) for action prediction, which utilize sequential context information to capture the appearance evolution and temporal structure of the full video observations. Ullah et al. [4] proposed an action recognition framework by utilizing frame level deep features of the CNN and processing it through bi-directional LSTM, which is capable of learning long term sequences and can process lengthy videos by analyzing features for a certain time interval. Acharya et al. [22] exploited the use of SPDNet on facial expression recognition problems. They leveraged covariance pooling to capture the temporal evolution of per-frame features for video-based facial expression recognition. After that they applied SPDNet on covariance of convolutional features. As in [22], we use deep features extracted from 2D CNN networks to calculate covariance descriptors, but instead of using them to train another neural network model, we propose a more explainable approach to quantify the difference between these descriptors and classify actions based on these differences.

Covariance descriptor for images: Tuzel et al. [23] introduced the use of covariance features and related algorithms for object detection and texture classification. They demonstrated the superior performance of the covariance features and algorithms on several examples. Sanin et al. [24] proposed an action and gesture recognition method based on spatio-temporal covariance descriptors and a weighted Riemannian locality preserving projection approach. The weighted projection was then used to create a classification algorithm that employed the most useful spatio-temporal regions. Tabia et al. [25] proposed a novel method for 3D shape analysis using the covariance matrices of the descriptors. They stated that covariance matrices enable efficient fusion of different types of features and modalities, which capture not only the geometric and the spatial properties of a shape region but also the correlation of these properties within the region. Liu et al. [22, 26] exploited the feasibility of representing such deep CNN image feature set with sample covariance matrix and the advantage of using such second-order statistics. Instead of using sample covariance matriix as the descriptor, in this paper we use the approach in [27] for covariance estimation from deep CNN features and represent each video clip as a covariance trajectory.

Trajectory-based: Papadopoulos et al. [7] proposed a human-tailored trajectory extraction scheme, in which trajectories are clustered using information from the human pose. Wang et al. [28] improved dense trajectories by explicitly estimating camera motion. They showed that performance can be significantly improved by removing background trajectories and warping optical flow with a robustly estimated homography approximating the camera motion. Peng et al. [29] proposed Stacked Fisher Vectors (SFV), a representation with multi-layer nested Fisher vector encoding, for action recognition. The combination of the traditional FV and SFV achieved high accuracies on various datasets. Zhang et al. [6] proposed a metric-based approach for simultaneous alignment and comparisons of covariance trajectories, and applied the framework to an application to the hand gesture recognition. Other trajectory-based representations have also been studied in video action recognition tasks as in [8, 9]. Trajectory-based approaches often encounter the problem of dealing with high dimensional representations of features.

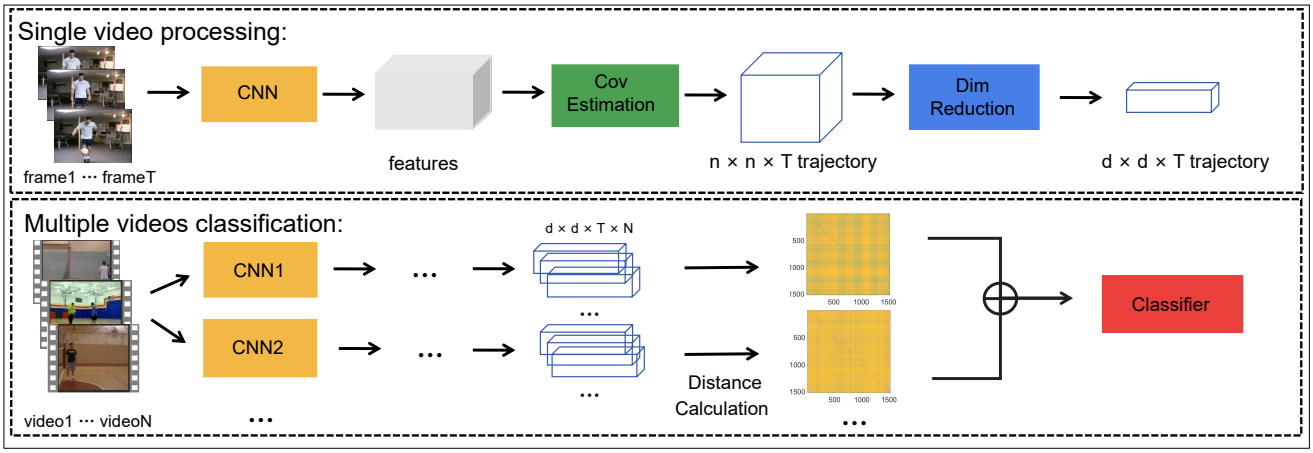

Figure 1. Pipeline of proposed framework.

3. Proposed Framework

In this section we introduce different components of our framework. After extracting deep features from video frames using existing CNN frameworks, we represent each frame as a covariance matrix estimated from the extracted features, and thus obtain a time series of covariance matrices or SPDM trajectory for each video clip. Next we apply a dimension-reduction approach based on the chosen Riemannian metric of SPDMs to bring individual covariances to smaller dimensions, and then compare these trajectories using a Riemmanian metric. We use the resulting distance features calculated from these trajectories for classification.

3.1. Mathematical Representation of Video Clips

Existing CNN frameworks such as ResNet [30] and VGGNet [31] have shown remarkable success in extracting powerful features from images. The covariance matrix provides a natural way of combining multiple correlated features. Liu et al. [22, 26] exploited the combination of deep CNN image feature set with sample covariance matrix and took advantage of using such second-order statistics. As they stated, one image frame can be treated as a set of feature vectors $F = \{f_1, f_2, ..., f_k\}$, where $f_i \in \mathcal{R}^n$ represents the i-th vector with an n-dimensional feature description. The sample covariance matrix is calculated as: $S = \frac{1}{k-1}\sum_{i=1}^{k}(f_i - \bar{f})(f_i - \bar{f})^T \in \tilde{\mathcal{P}}$, where $\tilde{\mathcal{P}}$ is the space of $n \times n$ SPDMs. Despite a simple form, the sample covariance matrix is not a good estimator in high dimensional space. In this paper we use the approach from [27] where the estimated Σ is an optimal convex linear combination of the sample covariance matrix S and the identity matrix I, i.e., $\Sigma = \rho_1 I + \rho_2 S \in \tilde{\mathcal{P}}$, where the optimal weights ρ_1 and ρ_2 are estimated from the data. The optimality is defined with respect to a quadratic loss function, asymptotically as the number of observations and the number of variables go to infinity together. Extensive Monte Carlo simulations confirm that the asymptotic results tend to hold well even in finite sample situations. Please refer to [27] for more details. After estimating a $n \times n$ covariance matrix for each frame, we obtain a $n \times n \times T$ covariance trajectory $\alpha : [0, T] \rightarrow \in \tilde{\mathcal{P}}$, for the full video clip with T frames.

3.2. Riemannian Structure on Symmetric Positive-Definite Matrices (SPDMs)

In order to quantify differences in covariance matrices and covariance trajectories, we need a metric structure on the manifold of SPDMs. While there are several Riemannian structures in the literature [6, 32, 33, 34, 35], we use the one introduced in [35] and [6], since it has the advantage of having closed forms for many operations we need on the SPDM manifold, e.g., geodesic distance, parallel transport, exponential map, inverse exponential map. Zhang et al. [6] also have demonstrated that this metric is superior over other metrics such as the log-Euclidean one [32] in some medical imaging contexts.

Let $\mathcal{P} \subset \tilde{\mathcal{P}}$ be subset of $n \times n$ SPDM matrices with determinant one. In our approach, we impose separate distances on the determinant one matrices and the determinants themselves. For any square matrix G with unit determinant, i.e. $G \in GL(n)$, we can write it as a product of two square matrices $G = PS$, where $P \in \mathcal{P}$ and $S \in SO(n)$ ($SO(n)$ is the set of all $n \times n$ rotation matrices). This is called the *polar decomposition*. It motivates us to analyze P by representing it as a G after removing S. More formally, we identify $\mathcal{P}$ with the quotient space $SL(n)/SO(n)$. This identification is based on the map $\pi : SL(n)/SO(n) \rightarrow \mathcal{P}$, given by $\pi([G]) = \sqrt{\tilde{G}\tilde{G}^t}$, for any $\tilde{G} \in [G]$, where the square-root is the symmetric, positive-definite square-root of a symmetric matrix. The notation $[G]$ stands for all possible rotations of the matrix G, given by $[G] = \{GS | S \in SO(n)\}$. The inverse map

of π is given by: $\pi^{-1}(P) = [P] \equiv \{PS | S \in SO(n)\}$. This establishes a one-to-one correspondence between the quotient space $SL(n)/SO(n)$ and $\boldsymbol{\mathcal{P}}$. Skipping further details, this process leads to the following geodesic distance between points in $\boldsymbol{\mathcal{P}}$. For any $P_1, P_2 \in \boldsymbol{\mathcal{P}}$:

$$d_{\boldsymbol{\mathcal{P}}}(P_1, P_2) = \|A_{12}\|_F \ , \tag{1}$$

where $A_{12} = \log(P_{12})$, $P_{12} = \sqrt{{P_1}^{-1} {P_2}^2 {P_1}^{-1}}$, and $\|\cdot\|_F$ denotes the *Frobenious norm* of a matrix. Since for any $\tilde{P} \in \tilde{\boldsymbol{\mathcal{P}}}$ we have $\det(\tilde{P}) > 0$, we can express $\tilde{P}$ as a pair $(P, \frac{1}{n}\log(\det(\tilde{P})))$ with $P = \frac{\tilde{P}}{\det(\tilde{P})^{1/n}} \in \boldsymbol{\mathcal{P}}$. Thus, $\tilde{\boldsymbol{\mathcal{P}}}$ is identified with the product space of $\boldsymbol{\mathcal{P}} \times \mathbb{R}_+$ and we take a weighted combination of distances on these two components to reach a metric on $\tilde{\boldsymbol{\mathcal{P}}}$:

$$d_{\tilde{\boldsymbol{\mathcal{P}}}}(I, \tilde{P})^2 = d_{\mathcal{P}}(I, P)^2 + \frac{1}{n}\left(\log(\det(\tilde{P}))\right)^2. \tag{2}$$

For any two arbitrary covariances $\tilde{P}_1$ and $\tilde{P}_2$, let $\tilde{P}_{12} = {\tilde{P}_1}^{-1}\tilde{P}_2 S_{12}$ for some optimal $S_{12} \in SO(n)$ (using Procrustes alignment). Also, note that for $\tilde{P}_{12} \in \tilde{\mathcal{P}}$, we have $\det(\tilde{P}_{12}) = \det(\tilde{P}_2)/\det(\tilde{P}_1)$. Therefore, the resulting squared geodesic distance between $\tilde{P}_1$ and $\tilde{P}_2$ is:

$$d_{\tilde{\boldsymbol{\mathcal{P}}}}(\tilde{P}_1, \tilde{P}_2)^2 = d_{\boldsymbol{\mathcal{P}}}(I, P_{12})^2 + \frac{1}{n}\left(\log(\det(\tilde{P}_2)) - \log(\det(\tilde{P}_1))\right)^2 \tag{3}$$

Note that in Eqn 3, the distance between covariance matrices is made up by two components – the determinant term and the unit symmetric matrix term. We can choose arbitrary relative weights on these terms to combine the two components. While, in some simple cases, it has been shown that one can obtain decent classification performance using only the determinant term, in general the other component provides important critical information about actual difference between covariance trajectories.

Next we want to calculate distances between covariance trajectories. Let α denote a smooth trajectory on the Riemannian manifold of SPDMs $\boldsymbol{\mathcal{P}}$, where $\boldsymbol{\mathcal{P}}$ is endowed with the Riemannian distance in Eqn 3. Let $\mathcal{M}$ denote the set of all such trajectories: $\mathcal{M} = \{\alpha : [0,1] \to \boldsymbol{\mathcal{P}} | \alpha \text{ is smooth}\}$. Let α_1 and α_2 be two smooth trajectories in $\mathcal{M}$, a simple way to establish a metric between them is

$$d_{\mathcal{M}}(\alpha_1, \alpha_2) = \int_0^1 d_{\tilde{\boldsymbol{\mathcal{P}}}}(\alpha_1(t), \alpha_2(t))dt \ . \tag{4}$$

We will use this metric to generate inputs for machine learning classifiers.

3.3. Dimension Reduction for SPDMs

Comparing covariance trajectories for large values of n, say $n \approx 1000$, can be computationally very expensive. For such situations we seek a method for the data reduction, while preserving the symmetric, positive-definite nature of covariance matrices. The basic idea, introduced in [16], is to find a linear projection that maps high-dimensional SPDMs to low-dimensional SPDMs in a principled, near-optimal manner. In addition to providing computational simplification, the low-dimensional SPDMs also facilitate our analyses in the following ways:

1. Such a projection can bring trajectories associated with different dimensions of features to the same smaller dimension, and make comparisons between them possible.

2. In the case that not all image features carry the same amount of information, dimension reduction can help filter out some noise components.

The problem of dimension reduction of SPDMs has been studied and used in a variety of computer vision and pattern recognition problems, see e.g. [32, 36, 37, 38]. In this paper, we use the dimension reduction technique based on the Riemannian metric presented in Section 3.2. The reduced SPDMs are especially suitable for analyzing under the proposed Riemannian framework.

Given a set of $n \times n$ unit-determinant SPDMs $\{P_i\}$, where n is a large integer, our goal is to find orthogonal matrix $B \in \mathbb{R}^{n \times d}$, where $d << n$ and $B^T B = I_d$, to project P_i to Q_i in $\mathbb{R}^{d \times d}$ according to $Q_i = B^T P_i B$. The space of such orthogonal matrices is called a *Stiefel manifold*, often denoted as $\mathcal{S}_{n,d}$. The next question is: What should be the optimality criterion for defining an optimal B? A simple yet important idea is that the pairwise distances between the given SPDMs should be preserved as much as possible after the projection. That is, find $B \in \mathcal{S}_{n,d}$ such that $d_{\boldsymbol{\mathcal{P}_d}}(Q_i, Q_j) \approx d_{\boldsymbol{\mathcal{P}_n}}(P_i, P_j)$ for all i, j in the training set. This criterion can be formulated as:

$$\underset{B \in \mathcal{S}_{n,d}}{\operatorname{argmin}} \sum_{i,j} \left(d_{\boldsymbol{\mathcal{P}_n}}(P_i, P_j) - d_{\boldsymbol{\mathcal{P}_d}}(Q_i, Q_j)\right)^2 \ .$$

A direct optimization of this quantity over $B \in \mathcal{S}_{n,d}$ is complicated due to the complexity of the chosen Riemannian metric. Instead, [16] develops an approximation where the comparison of distances is replaced by the comparison of relevant matrices directly.

In the original space, $\boldsymbol{\mathcal{P}}$, the distance between matrices P_i and P_j is governed by the matrix $P_{ij} = P_i^{-1} P_j^2 P_i^{-1}$. Similarly, the distance in the smaller space is determined by the matrix $Q_{ij} = Q_i^{-1} Q_j^2 Q_i^{-1}$. In order to compare these matrices, we need to bring them to the same space. Let $\hat{P}_i$ denotes the reconstruction of P_i from its projection Q_i, i.e. $\hat{P}_i = B Q_i B^T \in \mathbb{R}^{n \times n}$. Our goal is to find $B \in \mathcal{S}_{n,d}$ that minimizes the quantity:

$$\underset{B \in \mathcal{S}_{n,d}}{\operatorname{argmin}} \sum_{i,j} \|P_i^{-1} P_j^2 P_i^{-1} - \hat{P}_i^{-1} \hat{P}_j^2 \hat{P}_i^{-1}\|^2. \tag{5}$$

However, this specification requires the following proviso. Since $\hat{P}_i$ is rank d, it is not invertible, and one needs to use its pseudoinverse instead. Let $\hat{P}_i^- = BQ_i^{-1}B^T$ denote the pseudoinverse of $\hat{P}_i$. Then, we have the following result.

Lemma 1. *Under the conditions specified above, we have, for all i, j,*

$$\|P_i^{-1}P_j^2P_i^{-1} - \hat{P}_i^-\hat{P}_j^2\hat{P}_i^-\|^2 = \|P_{ij} - BQ_{ij}B^T\|^2.$$

To prove this, one only needs to show that $\hat{P}_i^-\hat{P}_j^2\hat{P}_i^- = BQ_{ij}B^T$ and that proof is left out. This lemma essentially provides another interpretation of the objective function.

Lemma 2. *The optimization of quantity in Lemma 1 can be rephrased as follows.*

$$B^* = \underset{B\in\mathcal{S}_{n,d}}{\operatorname{argmin}} \sum_{i,j=1}^{N} \|P_{ij} - BQ_{ij}B^T\|^2$$
$$= \underset{B\in\mathcal{S}_{n,d}}{\operatorname{argmax}} \left(\sum_{i,j=1}^{N} tr(B^TP_{ij}BB^TP_{ij}B) \right) .$$

We solve the optimization problem on the Stiefel manifold using the Matlab toolbox *Manopt* [39].

4. Experiments

In this section, we present experiments using video datasets to illustrate the proposed methodology.

4.1. Dataset

We use an extension of Youtube Action dataset [17] that contains 11 action categories: basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog. This data set is very challenging due to large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc. For each category, the videos are grouped into 25 groups with more than 4 action clips in each group. The video clips in the same group share some common features, such as the same actor, similar background, similar viewpoint, and so on. In the previous YouTube Action dataset, most groups contain 4 video clips, resulting a total of 1168 video clips. In our experiment we used all video clips of the 11 acitivies from UCF101 dataset [40], with a total of 1523 video clips.

4.2. Implementation Details

For each video clip, we extracted image frames using the OpenCV [41]. We used full images to extract features with pre-trained ResNet50 [30] and VGG16 [31] networks using Keras [42] with Tensorflow [43] Backend. Outputs from final convolutional layers were flattened to estimate covariance matrices using the approach described in Section 3.1. Each video then was represented as a sequence of $n \times n$ covariance matrices with sequence length T_i, where T_i is the number of frames in video i; thus, each sequence is size $n \times n \times T_i$. For example, covariance descriptors estimated using deep features from ResNet50 have $n = 2048$.

Comparison between trajectories in such high dimensions are almost computationally impossible. Thus, we implement the proposed dimension reduction technique and reduced the ith trajectory to dimension $d \times d \times T_i$. Since T_i differs from video to video, here we resampled all trajectories to $T = 50$, and the final dimension for each trajectory is then $d \times d \times T$. In Fig 2, we present an example to demonstrate the effect of dimension reduction. Here we calculate pairwise distances between trajectories using first 5 videos in each category and VGG16 model for feature extraction. In this example we present results for $d = 100,\ 50,\ 20$, and 5, respectively. We can see that our dimension-reduction successfully preserves the block structure of distance pattern within classes, except when d gets very small.

Since the space complexity of storing a $d \times d \times T$ trajectory is $O(d^2 \times T)$, and it would use too much space to store high dimensional trajectories $(d > 100)$. Thus, we did not run the full experiment in high dimensions given limited memory and time. One could try to find an optimal d given an evaluation metric for a specific problem, but also need to consider the tradeoff between its performance and cost. In the following experiments we set $d = 20$ for consistency. After obtaining such trajectories from multiple video clips, we calculated paiwise distances between them using Eqn 4. The average time for computing distances between two $20 \times 20 \times 50$ trajectories is 0.11s using Intel i7-6700HQ CPU in Matlab. The distances features were used as input to eventual classifiers.

To further illustrate the effectiveness of our framework, we performed simple experiments comparing with dimension reduction using PCA, and, classifications using trajectories as input instead of their distance features. The dimension reduction step using PCA was performed on the flattened CNN features. Table 1 shows average classification rates from different setups using VGG16 features with SVM classifier. Exploring various classifiers could potentially improve results, especially in the case where one uses trajectories as input, but in this paper we mainly focus on showing our framework and use SVM classifier as an example.

Since different CNN models capture features with different characteristics [44], the resulting distance patterns between covariance trajectories naturally differ from network to network. In this paper we have explored combining dis-

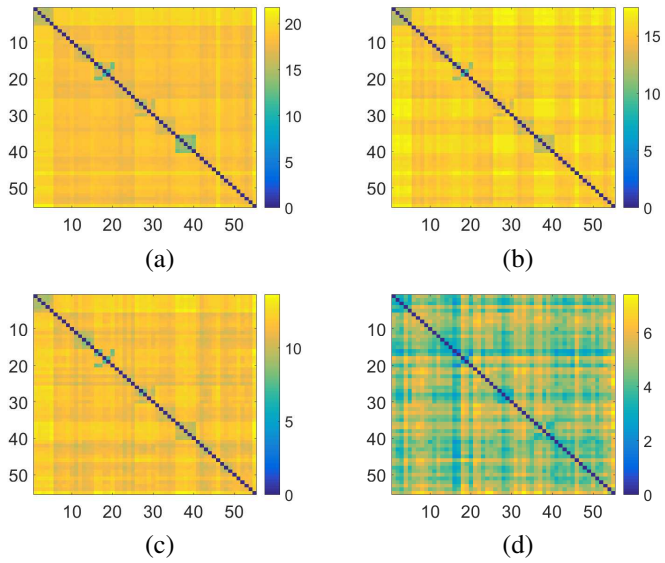

Figure 2. Pairwise distances between 55 covariance trajectories after dimension reduction. The trajectories were calculated from first 5 videos in each category of UCF11 dataset with VGG16 features. (a), (b), (c) and (d) show the distance matrices after reducing the dimension to d=100, 50, 20 and 5 respectively.

Table 1. Average classification rates on UCF11 dataset using VGG16 features when d = 20.

DR method	Distances as input	Accuracy(%)
PCA	No	24.6
PCA	Yes	33.8
Ours	No	80.6
Ours	Yes	88.9

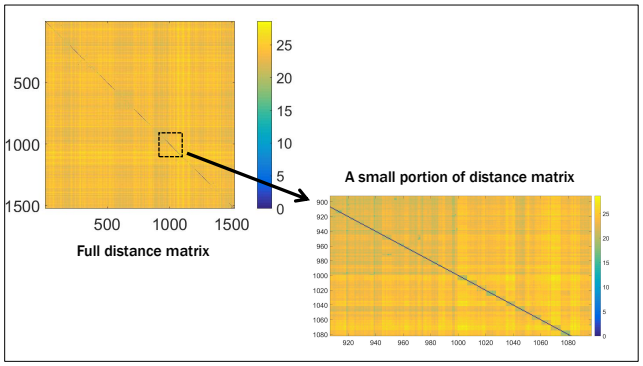

Figure 3. Sum of two distance matrices between 1523 covariance trajectories. One distance matrix is calculated using ResNet50 features and the other is calculated using VGG16 features. Left: the full distance matrix. Right: A small portion of distance matrix to show the block patterns within groups.

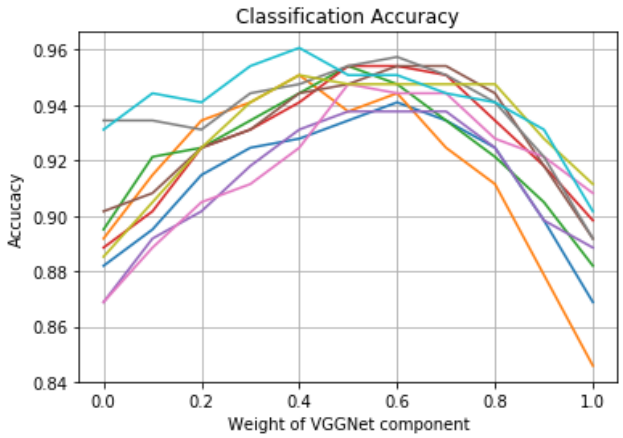

Figure 4. Overall classification rates vs distance feature components in 10 random splits of UCF11 dataset. For each line the weight of distance feature calculated from VGGNet goes up from left to right. The weight of distance feature calculated from ResNet goes down from left to right.

tance features, calculated using deep features from different CNN models, by adding the distance vectors with different weights together, and compared with results from individual CNN model. Denoting distance vector from VGG16 as F_1, with weight w_1, and distance vector from ResNet50 as F_2, with weight w_2, the final distance vector F is calculated as $F = w_1F_1 + w_2F_2$, where $w_1 + w_2 = 1$. We search the weight combinations from different data splits and try to find the pattern of a good weight combination for this dataset based on overall accuracy. At last step, we use SVM classifier with RBF kernel for classification. In the experiments we have used all 1523 video clips and randomly split the data into 80% training set and 20% test set. The experiments were mainly performed with GPU GeForce GTX 1080 Ti.

4.3. Results and Comparisons

Classifications of videos directly utilize distance features calculated from the corresponding covariance trajectories. In Fig 3 we display the full pairwise distance matrix as an image to show the clustering of videos from various groups and activities. As shown in Fig 3, videos within each group reveal explicit block structure. In most cases distances within a block are smaller than those outside the blocks, which makes it easy for activity classification. A few trajectories are found to have large distances from other samples.

To find an optimal weight between the two CNN models, we display overall accuracy *vs* weight of distance vectors from VGG16 for 10 random splits in Fig 4. From the picture we see that despite different data splits, overall results show common pattern across lines, i.e., a good combination should consider both components with similar weights. In the following experiments, we use $w_1 = 0.5$ for consistency.

When a new video comes to the system for recognition, one only needs to implement the following steps:

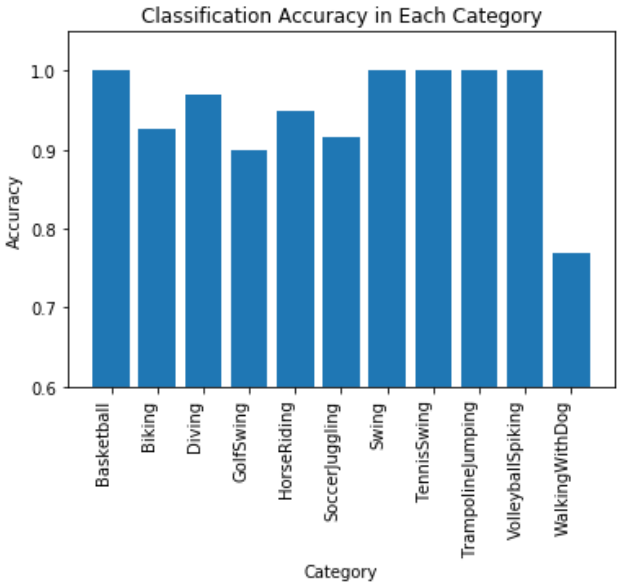

Figure 5. Classification accuracy for each class in UCF11 dataset.

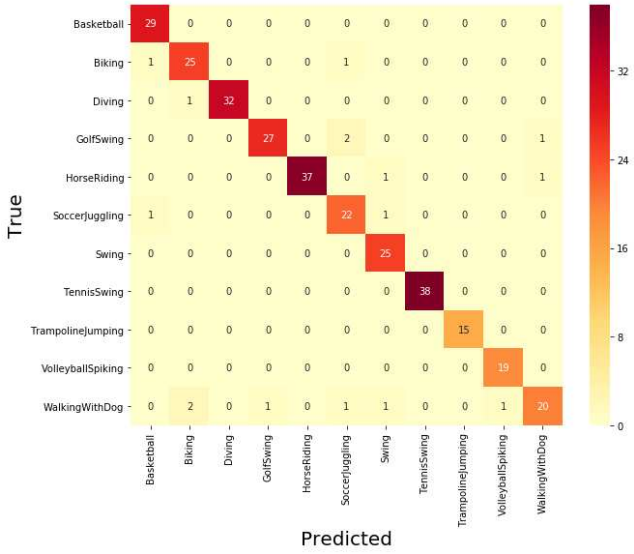

Figure 6. Confusion matrixes of UCF11 dataset for the proposed framework for classification.

1. Transform the video into a covariance trajectory and project it to the same dimension as trajectories in the database.

2. Calculate distances between this trajectory and others in the database.

3. Feed the distance vector into a pre-trained classifier.

For Step 2, one can use the mean of trajectories or a few typical videos in each class as templates to save time and storage cost. Time cost for calculating distance between two low dimensional trajectories using Eqn 4 is small, which makes the system computationally very efficient.

Experimental outcomes showing average overall accuracies, and using deep features extracted from different CNN models based on 5-fold cross-validation, are shown in Table 2. Classification rates for all categories are shown in Fig 5. Success rates for most classes are over 90%, with some classes reaching 100% accuracy. The only class with low (77%) success rate is *walking with a dog*. This can be due to very different backgrounds and angles at which the videos were recorded. The confusion matrix for classification is presented in Fig 6.

Table 2. Average classification rates on UCF11 dataset by using deep features extracted from different CNN models for the proposed framework for action recognition.

Network	Feature size	Accuracy(%)
ResNet50	$7 \times 7 \times 2048$	89.5
VGG16	$7 \times 7 \times 512$	88.9
ResNet50+VGG16		94.2

In the literature, some researchers have used a total of 1168 video clips from YouTube Action Dataset [17] in their studies. In our experiments, we used an extension with a total of 1523 video clips. With different amounts of data and experiment setups, it is hard to compare our results with other methods directly. Still, we present performances comparing various works in Table 3 for reference.

Table 3. Result comparison on YouTube Action dataset and the extended UFC11 dataset.

Method	Accuracy(%)
Liu et al. [17]	71.2
DT + BoVW [45]	85.4
Discriminative Parts [46]	84.5
Fisher Vectors [29]	93.4
Hierarchical clustering [10]	89.7
Bi-directional LSTM [4]	92.8
Proposed	94.2

5. Conclusion and Future Work

In this paper we present a comprehensive framework for action recognition tasks. We represent video frames as covariance descriptors estimated using deep CNN features and classify videos based on distances between the corresponding covariance trajectories. We also apply a method for SPDM dimension reduction, which saves significant computational costs while preserving pairwise distances as much as possible. The propsed ensemble model provides a way of combining different deep CNN features and use the distances of their descriptors for classification. Experimental results achieve high classification accuracies in a

multi-class classification problem.

In the future we aim to explore the performance involving features from more powerful models, and implement the proposed method on more challenging datasets to validate the adaptivity of our framework.

6. Acknowledgment

Some of the computing for this project was performed on the High Performance Computing (HPC) cluster at the Research Computing Center at the Florida State University (FSU).

References

[1] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, pages 568–576, Cambridge, MA, USA, 2014. MIT Press.

[2] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, 2016.

[3] J. Y. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4694–4702, 2015.

[4] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik. Action recognition in video sequences using deep bi-directional lstm with cnn features. *IEEE Access*, 6:1155–1166, 2018.

[5] A. Ben Tanfous, H. Drira, and B. Ben Amor. Coding Kendall's Shape Trajectories for 3D Action Recognition. In *IEEE Computer Vision and Pattern Recognition 2018*, Salt Lake City, United States, June 2018.

[6] Z Zhang, J Su, E Klassen, H Le, and A Srivastava. Rate-invariant analysis of covariance trajectories. *Journal of Mathematical Imaging and Vision*, published online, 2018.

[7] K. Papadopoulos, M. Antunes, D. Aouada, and B. Ottersten. Enhanced trajectory-based action recognition using human pose. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 1807–1811, Sep. 2017.

[8] I. Atmosukarto, B. Ghanem, and N. Ahuja. Trajectory-based fisher kernel representation for action recognition in videos. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 3333–3336, Nov 2012.

[9] H. A. Abdul-Azim and E. E. Hemayed. Human action recognition using trajectory-based representation. *Egyptian Informatics Journal*, 16(2):187 – 198, 2015.

[10] A. Liu, Y. Su, W. Nie, and M. Kankanhalli. Hierarchical clustering multi-task learning for joint human action grouping and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1):102–114, Jan 2017.

[11] Y. Tang, Y. Tian, J. Lu, P. Li, and J. Zhou. Deep progressive reinforcement learning for skeleton-based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[12] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[13] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1110–1118, June 2015.

[14] L. Sun, K. Jia, K. Chen, D. Y. Yeung, B. E. Shi, and S. Savarese. Lattice long short-term memory for human action recognition. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2166–2175, Oct 2017.

[15] F. Baradel, C. Wolf, J. Mille, and G. W. Taylor. Glimpse clouds: Human activity recognition from unstructured feature points. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[16] Mengyu Dai, Zhengwu Zhang, and Anuj Srivastava. Analyzing Dynamical Brain Functional Connectivity As Trajectories on Space of Covariance Matrices. *arXiv e-prints*, page arXiv:1904.05449, 2019.

[17] J. Liu, , and M. Shah. Recognizing realistic actions from videos in the wild. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1996–2003, June 2009.

[18] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, June 2014.

[19] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, Jan 2013.

[20] Y. Zhou, X. Sun, Z. Zha, and W. Zeng. Mict: Mixed 3d/2d convolutional tube for human action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[21] Y. Kong, Z. Tao, and Y. Fu. Deep sequential context networks for action prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3662–3670, July 2017.

[22] D. Acharya, Z. Huang, D. P. Paudel, and L. V. Gool. Covariance pooling for facial expression recognition. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 480–4807, 2018.

[23] O. Tuzel, F. M. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. In *ECCV*, 2006.

[24] M. T. Harandi, C. Sanderson, A. Sanin, and B. C. Lovell. Spatio-temporal covariance descriptors for action and gesture recognition. In *Proceedings of the 2013 IEEE Workshop on Applications of Computer Vision (WACV)*, WACV '13, pages 103–110. IEEE Computer Society, 2013.

[25] H. Tabia, H. Laga, D. Picard, and P. Gosselin. Covariance descriptors for 3d shape matching and retrieval. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4185–4192, June 2014.

[26] M. Liu, R. Wang, S. Li, S. Shan, Z. Huang, and X. Chen. Combining multiple kernel methods on riemannian manifold for emotion recognition in the wild. In *Proceedings of the 16th International Conference on Multimodal Interaction*, ICMI '14, pages 494–501, New York, NY, USA, 2014. ACM.

[27] O. Ledoit and M. Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365 – 411, 2004.

[28] H. Wang and C. Schmid. Action recognition with improved trajectories. In *2013 IEEE International Conference on Computer Vision*, pages 3551–3558, Dec 2013.

[29] X. Peng, C. Zou, Y. Qiao, and Q. Peng. Action recognition with stacked fisher vectors. In *Computer Vision – ECCV 2014*, pages 581–595, Cham, 2014. Springer International Publishing.

[30] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[32] X. Pennec, P. Fillard, and N. Ayache. A riemannian framework for tensor computing. *International Journal of Computer Vision*, 66(1):41–66, Jan 2006.

[33] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Log-Euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine*, 56(2):411–421, August 2006.

[34] I. L. Dryden, A. A. Koloydenko, and D. Zhou. Non-Euclidean statistics for covariance matrices with applications to diffusion tensor imaging. *Annals of Applied Statistics*, 3(3):1102?1123, 2009.

[35] J Su, I.L. Dryden, E Klassen, H Le, and A Srivastava. Fitting optimal curves to time-indexed, noisy observations on nonlinear manifolds. *Journal of Image and Vision Computing*, 30(6-7):428–442, 2012.

[36] Cristani-Murino Tosato, Spera. Characterizing humans on riemannian manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(08):1972–1984.

[37] M. T. Harandi, M. Salzmann, and R. I. Hartley. Dimensionality reduction on spd manifolds: The emergence of geometry-aware methods. *IEEE transactions on pattern analysis and machine intelligence*, 2017.

[38] Y. Li and R. Lu. Locally preserving projection on symmetric positive definite matrix lie group. *CoRR*, abs/1703.09499, 2017.

[39] N.s Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a matlab toolbox for optimization on manifolds. *J. Mach. Learn. Res.*, 15(1):1455–1459, January 2014.

[40] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, page 2012.

[41] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[42] F. Chollet et al. Keras, 2015.

[43] M. Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[44] C. Feichtenhofer, A. Pinz, R. P. Wildes, and A. Zisserman. What have we learned from deep representations for action recognition? *2018 IEEE/CVF Conference*

on Computer Vision and Pattern Recognition, pages 7844–7853, 2018.

[45] H. Wang, A. Kläser, C. Schmid, and C. Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103(1):60–79, May 2013.

[46] M. Sapienza, F. Cuzzolin, and P. H.S. Torr. Learning discriminative space–time action parts from weakly labelled videos. *International Journal of Computer Vision*, 110(1):30–47, Oct 2014.