Higher-Level Representation of Local Spatio-Temporal Features for Human Action Recognition Using Subspace Matching Kernels

Bisser Raytchev, Hideaki Kawamoto, Toru Tamaki and Kazufumi Kaneda
Department of Information Engineering, Graduate School of Engineering, Hiroshima University, Hiroshima, JAPAN

Abstract—Although the design of low-level local spatio-temporal features has recently led to significant improvement of performance in many action recognition applications, much less attention has been given to the equally important problem how to organize such low-level features extracted from the videos into a higher-level representation suitable to represent and discriminate between many different action classes. In this paper we propose an alternative approach to the widely-used Bag-of-Features (BoF), where instead of histograms of visual words, action sequences are represented as sets of low-dimensional linear subspaces. This results in richer and more discriminative models. In order to be able to calculate similarities between sets of subspaces we define a novel Subspace Matching Kernel. Experimental results are shown on the widely used KTH action dataset, which demonstrate the effectiveness of the proposed framework.

I. INTRODUCTION

Human action recognition from videos is an important research topic in computer vision and pattern recognition with numerous applications in areas like natural human-computer interfaces, video surveillance, robotic vision and many others. Although many different approaches have been proposed in the literature to address this problem [1], to date one of the most successful and widely used approach seems to be the one based on bag-of-features (or bag-of-words) models [2] which uses local spatio-temporal features [3]. Building on the success of local invariant features in generic object recognition from static images [4], numerous spatio-temporal local feature detectors and descriptors (extending local feature detectors and descriptors to the spatio-temporal domain) like the Harris3D [5], Cuboids [6] and Hessian [7] detectors, and the HOG/HOF [8], HOG3D [9], ESURF [7] and Motion Boundary Histograms (MBH) [10], [11] descriptors, etc., have been proposed and used in many action/behavior recognition related tasks. Wang et al. [12], [10] evaluate several of the most widely used and representative spatio-temporal detectors/descriptors on several action recognition video datasets and also discuss their advantages and limitations.

Although the design of such low-level features has led to significant improvement of performance in many important action recognition applications, much less attention has been given to the equally important problem how to organize such low-level spatio-temporal features extracted from the videos into a higher-level representation suitable to represent and discriminate between many different action classes. Still, the model predominantly used to represent video sequences relies on the same old bag-of-features framework, i.e. much less research has been done on finding alternative models through which to represent the sets of features.

In this paper, we investigate one possible alternative direction - instead of representing videos as histograms of visual words, we propose to use richer models, where the distributions of the low-level local spatio-temporal features are captured by a higher level representation through the bases of low-dimensional subspaces.

This idea is illustrated in Fig. 1. In the proposed framework, the set of local spatio-temporal features extracted from a video corresponding to a certain action class is represented as a low-dimensional linear subspace, and such a higher-level subspace-based representation is learned separately for each available class. The set of local features extracted from a test video is also represented by a low-dimensional subspace, representing a "query", and the distance between such a subspace-based query and a subspace-based vocabulary is defined as the distance between their corresponding subspaces. The crucial difference with the Bag-of-Features framework is that while BoF essentially matches lower-level local features (i.e. matches the query features to some other pre-learned representative features, the vocabulary), here we propose to match higher-level structures, in the form of subspaces, which could capture and represent better the overall character of each action sequence.

The natural way to represent distances between subspaces is by formulating the problem on the Grassmann manifold [13], the set of fixed-dimensional linear subspaces of a Euclidean...
space. Various distances which consider the geometric structure of the Grassmann manifold and can be represented in terms of the principal angles between subspaces have been given in [14]. A recent work in [15] shows how some such metrics can be used to define Grassmann kernels, which can be used with kernel-based learning algorithms. All this machinery for comparing a pair of subspaces based on metrics on Grassmann manifolds can be used in the proposed framework, which in the style of Bag-of-Features we call Subspaces-of-Features (SoF).

Although subspaces provide a richer model than histograms, this does not lead to problems with high-dimensional data (as in the case of BoF, where depending on the size of the vocabulary and the way the histograms are calculated, the resulting feature vectors can be very high-dimensional), as both the vocabularies and the queries are represented by low-dimensional subspaces (typically 20-50 dimensions). Additionally, the metric used, distance on Grassmann manifolds, is mathematically well-founded and is the only natural metric for the task, while in the case of BoF there is no obvious choice of metric both for the histogram calculation step and the histogram comparison step, and these usually need to be determined experimentally.

Still, one possible problem with the SoF approach described above might be a lack of representation precision, if a relatively complex action sequence needs to be represented by only a single subspace. It would be desirable to provide more flexibility in the model by allowing action sequences to be represented by several subspaces, as shown in Fig. 2. Complex actions categories may consist of several different specific motions, which if put together in a single subspace might affect negatively the representative/discriminative abilities of the model. For this reason it is necessary to define a kernel able to match/compare sets of subspaces. One of the main contributions of the present paper is the definition of such a kernel, which we call Subspace Matching Kernel (SMK). By being able to handle sets of subspaces, SMK greatly extends the functionality of Grassmann kernels [15], [16] and can be also useful in many applications, even beyond action recognition.

The rest of the paper is organized as follows. In the next section we show how the sets of local spatio-temporal features extracted from videos can be represented as subspaces, and how learning and classification through Grassmann kernels defined on these subspaces can be accomplished. Section III shows how inter-class information can be used effectively to create more distinctive vocabularies by mutually orthogonalizing the subspaces corresponding to objects of different categories. Section IV introduces the Subspace Matching Kernel, and section V shows experimental results where the methods described in the previous sections are compared with BoF on one of the most popular and widely used for benchmarking video datasets, the KTH dataset [3].

II. SUBSPACES OF LOCAL SPATIO-TEMPORAL FEATURES

Consider two sets of local spatio-temporal features, extracted from two different videos, \( V_i = \{x_{i1}^1, x_{i2}^1, \ldots, x_{in}^i\} \) and \( V_j = \{x_{j1}^1, x_{j2}^1, \ldots, x_{jn}^j\} \), where \( x_{ni}^i \) is the \( n \)-th local feature from the \( i \)-th video, in vector form. We can represent these two videos by their corresponding subspaces in the Euclidean space \( \mathbb{R}^D \), \( \text{span}(V_i) = \text{span}\{u_1, \ldots, u_p\} \), and \( \text{span}(V_j) = \text{span}\{v_1, \ldots, v_p\} \), where \( \text{span}(V_i) \) denotes the subspace spanned by the column vectors of the \( D \times p \) matrix \( Y_i = [u_1, \ldots, u_p] \), and \( \{u_1, \ldots, u_p\} \) are orthonormal. The set of all \( p \)-dimensional linear subspaces in \( \mathbb{R}^D \) is called the Grassmann manifold \( G(p, D) \), and the subspaces \( \text{span}(V_i) \) and \( \text{span}(V_j) \) can be considered as two points on the manifold \( G(p, D) \). Various distances on \( G(p, D) \) have been defined [14], all of which can be represented in terms of the principal angles between subspaces [17]. The principal angles, or canonical angles, \( \{0 \leq \theta_1 \leq \cdots \leq \theta_p \leq \pi/2\} \), between the subspaces \( \text{span}(V_1) \) and \( \text{span}(V_2) \) can be defined recursively as

\[
\cos \theta_k = \max_{u_k \in \text{span}(V_1)} \max_{v_k \in \text{span}(V_2)} u_k^T v_k,
\]

subject to the constraints

\[
\begin{align*}
&u_k^T u_k = 1, \quad v_k^T v_k = 1, \\
&u_k^T u_i = 0, \quad v_k^T v_i = 0, \quad (i = 1, \ldots, k - 1)
\end{align*}
\]

The principal angles can be computed in a numerically stable way from the Singular Value Decomposition (SVD) of \( Y_1^T Y_2 \)

\[
Y_1^T Y_2 = U \Lambda A^T,
\]

\[
\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_p)
\]

where the orthonormal matrices \( Y_1 \) and \( Y_2 \) are the matrix representations of \( \text{span}(V_1) \) and \( \text{span}(V_2) \), and \( \lambda_i = \cos \theta_i \) are the cosines of the principal angles \( \theta_i \), also known as the canonical correlations.

Hamm and Lee have shown [15] that two of the Grassmann distances in [14], namely the projection and the Binet-Cauchy metric can be used to define the following positive definite Grassmann kernels:

The projection kernel

\[
k_p(Y_1, Y_2) = ||Y_1^T Y_2||_F^2
\]

The Binet-Cauchy kernel

\[
k_{BC}(Y_1, Y_2) = \det(Y_1^T Y_2)^2 = \det(Y_1^T Y_2 Y_2^T Y_1) = \prod_i \cos^2 \theta_i
\]

3852
Therefore, in the proposed Subspaces-of-Features (SoF) framework, a subspace is formed from the local spatio-temporal features extracted from each video sequence, and the Grassmann kernels obtained from the training/test video sequences are used in conjunction with any of the available kernel-based algorithms [18] to learn/classify action classes. For example, a Kernel Discriminant Analysis can be used, as in [15] (called GDA there), where a Grassmann kernel is used, applying the kernel trick to the Rayleigh quotient \( L(w) = w^T S_b w / w^T S_w w \), used in Linear Discriminant Analysis to find the discriminant direction \( w \), where \( S_b \) and \( S_w \) are the between-class and within-class covariances matrices. If \( \phi \) is the feature map and \( \Phi = [\phi_1, \ldots, \phi_N] \) the feature matrix of the training data (each training data is a subspace in this case), then by representing \( w \) as a linear combination of the feature vectors \( w = \Phi \alpha \), the Rayleigh quotient can be expressed in terms of \( \alpha \) as
\[
L(\alpha) = \frac{\alpha^T \Phi^T S_B \Phi \alpha}{\alpha^T \Phi^T S_W \Phi \alpha} = \frac{\alpha^T K (V - 11^T / N) K \alpha}{\alpha^T \sigma^2 I \alpha}
\]
where \( K \) is the kernel matrix obtained by applying one of the Grassmann kernels on the training data, \( I \) is an \( N \)-vector of all-ones, \( V \) is a block-diagonal matrix whose \( c \)-th block (corresponding to the \( c \)-th class) is an \( N_c \times N_c \) all-ones matrix divided by \( N_c \) (the number of training samples from the \( c \)-th class), and \( \sigma^2 I \) is a regularizer. GDA proceeds by finding through eigen-decomposition the values of \( \alpha \) which maximize Eq. (6), and then classification is done by nearest neighbor classification (or k-NN) using the Euclidean distance between \( F_{\text{train}} = \alpha^T K \) and \( F_{\text{test}} = \alpha^T K_{\text{test}} \), where \( K_{\text{test}} \) is the kernel matrix obtained from both training and test samples.

### III. Orthogonalization of the Subspaces

In the previous section, the subspaces were formed directly from the local features extracted from each video, without taking into consideration information across different action classes. The subspace representation can be made more discriminative in terms of classification by orthogonalization, i.e. a transformation can be found which makes the subspaces corresponding to different action classes orthogonal to each other [19]. This can be achieved through the Orthogonal Subspace Method (OSM) [20] in the following way. First, the correlation matrix \( C_0 \) of the mixture of all action classes
\[
C_0 = \sum_{i=1}^{c} \pi_i C_i
\]
is formed, where \( C_i \) are the correlation matrices for each of the \( c \) classes, and \( \pi_i \) the corresponding a priori class probabilities. Then matrix \( C_0 \) is diagonalized by \( B C_0 B^T = \Lambda \) and the whitening transform \( W = \Lambda^{-1/2} B \) is formed (which whitens \( C_0 \), i.e. \( W C_0 W^T = I \)). From
\[
W C_0 W^T = \sum_{i=1}^{c} \pi_i W C_i W^T
\]
\[
= \sum_{i=1}^{c} D_i = I
\]
where \( D_i = \pi_i W C_i W^T \), it follows that \( D_i \) and \( \sum_{j \neq i} D_j \) have the same eigenvectors, but the corresponding eigenvalues \( \lambda(i) \) and \( \mu(i) \) are related by \( \lambda(i) = 1 - \mu(i) \). This means that the eigenvectors of \( D_i \) corresponding to the largest eigenvalues would be at the same time the eigenvectors of \( \sum_{j \neq i} D_j \) corresponding to its smallest (least important) eigenvalues.

Then, if we choose the basis for the subspace corresponding to the \( i \)-th action class to be formed from the eigenvectors in matrix \( B_i \) which diagonalizes \( D_i \), i.e. \( B_i^T D_i B_i = \Lambda_i \), it would follow that
\[
\sum_{j \neq i} B_j^T D_j B_i = I - \Lambda_i.
\]

Since \( D_i \) are positive semidefinite matrices, \( B_i^T D_i B_i \) is a diagonal matrix with elements smaller than 1, and if \( D_j \) is represented by \( B_j \Lambda_j B_j^T \) then \( B_j^T B_i \Lambda_i B_j^T B_i \) would have small diagonal elements, and \( B_j^T B_j \) would be close to a zero matrix (would be exactly a zero matrix if the eigenvalues of \( D_j \) are exactly equal to one). In the latter case, the columns of \( B_i \) and \( B_j \) would be orthogonal and the subspaces they span would also be orthogonal (or almost orthogonal in reality).

In the experiments shown in section V the columns \( B_i \) corresponding to the largest \( p \) eigenvalues are used as a basis for the \( p \)-dimensional subspace representing the \( i \)-th action class (the value of \( p \) is determined experimentally).

### IV. The Subspace Matching Kernel

Although Grassmann kernels and Grassmann distances provide a method to compare two subspaces, they are not sufficient to compare two sets of subspaces, as in the case we are interested. As shown in Figure 2, we would like to represent a single action sequence as a group of motions, and each motion within the group as a separate subspace. Then we need to have a mechanism to compare two sets of subspaces (the number of subspaces in each set can be different, as different action categories can be represented best by a different number of constituting motions/subspaces). To this end, we define the Subspace Matching Kernel (SMK), which is able to provide a measure of similarity between two sets of subspaces.

Let \( V_i = \{ M^i_1, \ldots, M^i_n \} \) and \( V_j = \{ M^j_1, \ldots, M^j_m \} \) be two action video sequences represented as sets of their specific constituting Motions, where \( M^i_k \) denotes the \( k \)-th motion/subspace in video sequence \( V_i \), represented by the matrix spanning the subspace of the local features defining that particular motion.

The easiest way to define a matching kernel between two sets of subspaces might be to use the \( \text{max} \) operator, i.e. just match each element in each set to the element in the other set to which it is most similar, as for instance is done in the kernel recipe for matching two sets of local features given in [21]. Unfortunately, it has been pointed out in [22] that the \( \text{max} \) of a Mercer kernel is not necessarily a Mercer kernel.

For this reason, we define the Subspace Matching Kernel (SMK) using a soft-max rather than a \( \text{max} \) operation, in a similar way as done in [23] for two sets of local features:
\[
k_{\text{SMK}}(V_i, V_j) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{k=1}^{m} [kP(M^i_j, M^j_k)]^\beta,
\]
where \( k_P(\ldots) \) is some basic Grassmann kernel (the Projection Kernel from equation 4 has been used here and in the experiments, but it can be substituted by any other Grassmann kernel). Here \( \beta \geq 1 \) is a kernel parameter, determining the “softness” of the soft-max operator. For low values of \( \beta \) most of the elements in the two sets will contribute to the similarity score, while for high values of \( \beta \) only the most similar elements would have influence, leading in the extreme to a global \( \max \) operator. The most suitable value of the parameter \( \beta \) would depend on the data, but in general taking the soft-max instead of the \( \max \) allows to quantify more subtle similarities between the sets of regions. Finally, note that the soft-max operation has been proved to produce a positive definite kernel in [24], i.e. the kernel in equation (11) is a valid Mercer kernel.

Another question which needs to be answered is how to “segment” the original action sequence into its constituting motions (each one of which in turn would be represented by a subspace). One extreme solution would be to construct a spatio-temporal pyramid, as shown for ease of visualization in a somewhat simplistic way in Fig. 2, i.e. just cut along the temporal axis (and the spatial ones too if necessary) and continue this process until all desired levels of granularity are covered (a few resulting partitions are shown in different color in Fig. 2). A better way though would be to automatically segment the original action sequence by using some clustering algorithm, and indeed using a \( k \)-means clustering led to better results in the experiments (using more sophisticated segmentation algorithms, like [25], for instance, to segment the action sequence would undoubtedly lead to better results, but even a simple algorithm like the \( k \)-means shows significant increase in recognition rate and is sufficient to illustrate the concept). However, using a predefined number of clusters for all sequences is not likely to be optimal, and for that reason we adopt the following strategy. We run the \( k \)-means algorithm for different consecutive values of \( k \) (say, \( k = 1, 2, 3, \ldots, K \)); the value of \( K \), largest number of clusters, would be small if the dataset consists of relatively simple action categories, and may grow larger for more complex datasets), each time obtaining \( k \) different motion clusters for each sequence. Then we represent each of the resulting clusters as subspaces (i.e. we use all clusters, obtained for all values of \( k \)), and let the matching kernel select automatically the most relevant ones. A similar procedure can be used with different number of partitions, in the case when the partitions are formed by cutting across the spatio-temporal axes rather than clustering. For larger values of the parameter \( \beta \) in equation (11) the irrelevant clusters/partitions would simply add too small similarity values to influence the final similarity score and thus would be effectively ignored, while the influence of the relevant ones (which can happen to be obtained for different values of \( k \)) would be significantly amplified, and would effectively determine the final similarity score. In this way, the matching kernel has the very desirable property to allow automatic selection of scale (or relevant detail) of the set of motions to represent each action sequence.

V. EXPERIMENTAL RESULTS

In this section we illustrate the proposed Subspace-of-Features (SoF) framework by comparing the performance of the algorithms described in the previous sections to that of the conventional Bag-of-Features (BoF) framework on one of the most popular human action recognition benchmarks, the KTH dataset [3]. The KTH dataset consists of six human action categories (“boxing”, “hand clapping”, “hand waving”, “jogging”, “running”, and “walking”), which are performed in four different scenarios (outdoors, outdoors with scale variation, outdoors with different clothes and indoors) by 25 different subjects, resulting in 2,391 video sequences in total. We follow the standard experimental setup for this dataset which splits the data into a training set (8 subjects), validation set (8 subjects) and a test set (9 subjects). The data from the validation set is used to tune all relevant parameters for each method, after which these parameters are fixed and the final recognition results are reported on the test set. From each video sequence local features were extracted and used as input to the different representation frameworks, both BoF and SoF. We used combined HOG, HOF, MBH and Trajectory motion-based descriptors obtained from the improved dense trajectories introduced in Ref. [11], as these have been shown to be very successful for human action recognition (we used the code provided by the authors).

<table>
<thead>
<tr>
<th>Method</th>
<th>Recognition Rate [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>BoF (intersection kernel + SVM)</td>
<td>93.7</td>
</tr>
<tr>
<td>SoF (single subspace)</td>
<td>93.0</td>
</tr>
<tr>
<td>SoF (single orthogonalized)</td>
<td>94.4</td>
</tr>
<tr>
<td>SoF (partition)</td>
<td>96.5</td>
</tr>
<tr>
<td>SoF (k-means clustering)</td>
<td>97.5</td>
</tr>
</tbody>
</table>

As a baseline method for comparison we used a standard implementation of the Bag-of-Features (BoF) algorithm, using an SVM as a classifier. Table I compares the best results obtained for both BoF and the different variants of SoF shown in an order of using increasingly richer models. The results show, as anticipated, that representing a whole action sequence by a single subspace does not provide a sufficiently rich model, and performance is similar to that of BoF. Using additional inter-class information by orthogonalizing the subspaces (as explained in section III) improves things a little bit, but still does not result in a sufficiently rich model. The most significant increase in performance is achieved when each action sequence is represented by multiple subspaces which are matched using the Subspace Matching Kernel from section IV. Again, as expected, using a more flexible way to segment the sequences into constituting motions (through clustering vs. the more rigid partitioning) leads to better results.

Below we show how the methods are influenced by the relevant parameters and design choices. First, Figure 3 shows how the performance in the BoF framework is influenced by the number of visual words and the kernel type. For the KTH dataset and dense trajectories-based features, best recognition rates are obtained for 2000 visual words (using less than 1000 visual words seems not enough, then performance saturates around few thousands) and the intersection kernel (for comparison results for the linear kernel are also shown – use of other kernels like RBF and \( \chi^2 \) did not result in better performance).

Next we turn to the SoF framework. First we show in more detail how the selection of subspaces influences the results (this will be shown for the clustering version; things are similar in
Fig. 3. Influence of the number of visual words and kernel type on the recognitions rates for the BoF method.

Fig. 4. Different ways to segment a video sequence into its constituting motions by using k-means clustering (see text for details).

principle for the partition version) As mentioned toward the end of section IV we perform k-means clustering for each action sequence for different consecutive values of k. This is illustrated in Fig. 4. For k = 1 (case A in Fig. 4), actually no clustering needs to be done and we form a single subspace from the whole sequence. This corresponds to SoF (single subspace) and SoF (orthogonalized) if orthogonalization is used. Case B shows that two clusters are formed from the sequence (in the partition case this would correspond to taking the first half and second half of the sequence), and further 3 clusters are found in case C. For the KTH dataset we do not expect that more than 3 constituting motion would be needed, and for that reason do not continue further to obtain more clusters. Now, as we mentioned in section IV using a pre-defined number of clusters might not be optimal for all sequences (some might be better represented by a single cluster, some might need 2 or 3 and so on). For that reason in case D in Fig. 4 we supply all clusters obtained in cases A and B (i.e. 1 + 2 clusters being used) and let the matching kernel choose automatically which clusters are relevant for each different sequence. Similarly, in case E we use the clusters from cases A, B and C (i.e. 1 + 2 + 3 clusters), and this process can be continued further if necessary. The horizontal lines in cases D and E indicate that the clusters/subspaces separated by the lines have been obtained at different scales, i.e. a pyramidal structure is formed and each further level of the pyramid corresponds to more detailed representation (the number of constituting motions is increased). The pyramidal structure in cases D and E allows to match the resulting subspaces in two different ways: (a) match all subspaces on equal terms, irrespective of the pyramid level from which they come; and (b) match only subspaces coming from the same pyramid level (i.e. don’t mix levels).

The recognition results for cases A to E from Fig. 4 are given in Fig. 5, where SoF_A corresponds to case A and so on. For cases D and E “all” means that the subspaces from all levels are matched together (irrespective of the pyramid level), while “pyramid” means that only subspaces from the same pyramid level are matched together. As expected, the results show that matching all subspaces across all scales provides best results.

Figure 6 shows how recognition rate is affected by the dimension of the subspaces. The results are obtained on the validation set for case D when changing the subspace dimension from 10 to 70. The performance peak is at 20 dimensions, after which recognition rates slightly decrease, however no significant or abrupt drop in performance is evident, which shows that the algorithm is not very sensitive to the actual value of this parameter.

Finally, Tables 2 and 3 compare the confusion matrices for the cases when a single subspace and multiple subspaces are used respectively. As can be seen from the confusion
matrices, in both cases only relatively similar action categories, like hand-waving and hand-clapping, are being predominantly confused. However, the richer model using multiple subspaces results in more precise recognition, evidenced by much fewer mistakes of hand-clapping with boxing and no mistakes of running with jogging.

<table>
<thead>
<tr>
<th></th>
<th>clapping</th>
<th>waving</th>
<th>jogging</th>
<th>running</th>
<th>walking</th>
</tr>
</thead>
<tbody>
<tr>
<td>boxing</td>
<td>141</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>clapping</td>
<td>12</td>
<td>134</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>waving</td>
<td>0</td>
<td>13</td>
<td>113</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>jogging</td>
<td>0</td>
<td>0</td>
<td>140</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>running</td>
<td>0</td>
<td>0</td>
<td>31</td>
<td>113</td>
<td>0</td>
</tr>
<tr>
<td>walking</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>142</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>clapping</th>
<th>waving</th>
<th>jogging</th>
<th>running</th>
<th>walking</th>
</tr>
</thead>
<tbody>
<tr>
<td>boxing</td>
<td>179</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>clapping</td>
<td>4</td>
<td>159</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>waving</td>
<td>0</td>
<td>11</td>
<td>135</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>jogging</td>
<td>0</td>
<td>0</td>
<td>139</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>running</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>144</td>
<td>0</td>
</tr>
<tr>
<td>walking</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>142</td>
</tr>
</tbody>
</table>

VI. CONCLUSION

In this paper we have proposed a novel framework for representation and recognition of human actions in video sequences. In the proposed framework, the low-level local spatio-temporal features extracted from each video are represented as sets of low-dimensional linear subspaces to enhance discriminability by building richer models (experimental results confirm that the proposed framework is able to achieve higher recognition rates than the popular BoF framework). In order to be able to calculate similarities between sets of subspaces we have defined a novel Subspace Matching Kernel which extends and generalizes conventional Grassmann kernels to be able to handle multiple sets of subspaces. We expect this to be also very useful in other problems beyond action recognition, where the underlying modalities can be naturally represented as sets of subspaces. We expect that significant further improvements can also be achieved by using more sophisticated video segmentation techniques to obtain the constituting motions from which the subspace are learned, or using alternative techniques from the rich literature on subspace methods to learn the subspace representations, like constrained mutual subspaces or non-linear subspaces, etc.

ACKNOWLEDGMENT

This work has been supported in part by JSPS KAKENHI Grants Numbers 25330337 and 16K00394 to the first author.

REFERENCES