

Multi-view Recognition Using Weighted View Selection

Scott Spurlock^(✉), Hui Wu, and Richard Souvenir

University of North Carolina at Charlotte, Charlotte, NC 28223, USA
{sspurloc,hwu13,souvenir}@uncc.edu

Abstract. In this paper, we present an algorithm for multi-view recognition in a distributed camera setting that learns which viewpoints are most discriminative for particular instances of ambiguity. Our method is built on top of 2D recognition algorithms and casts view selection as the problem of optimizing kernel weights in multiple kernel learning. The main contribution is a locality-sensitive meta-training step to learn a disambiguation function to select the relative weighting of available viewpoints needed to classify a 2D input example. Our method outperforms related approaches on benchmark multi-view action recognition data sets.

1 Introduction

Multi-view human action recognition is an active area of research with applications to surveillance, robotics, and human-computer interaction. Figure 1 shows a scene captured from multiple cameras with two tracked people, where the goal is to recognize the actions of each tracked person. In general, there are two main approaches to recognition from multiple viewpoints: integrate data from all the views to build a 3D model and solve a 3D recognition problem, or consider some combination of multiple 2D views. In general, 3D approaches tend to perform well, but require unobstructed views and, usually, a high computational cost for 3D model construction or finding correspondences. For the 2D case, a major challenge is caused by the unknown relative viewpoint or pose of the object, whereby instances from the same class may appear different from different viewpoints, while, from other viewpoints, instances from different classes may appear similar.

Our approach is applicable to distributed camera networks for human activity understanding (e.g., Fig. 1). In particular, we consider architectures with peer camera nodes, where for a given target, there is a *primary* camera and the rest of the camera nodes are *secondary*. This designation can be fixed, where a particular camera is primary for targets in a specified region, or dynamic, e.g., the primary camera is selected to track a target [1]. In either case, secondary views are represented by their *relative offset* from the primary camera.

Our focus is to only incorporate secondary views when necessary by learning a model that recognizes potentially confusing poses and determines the relative value of secondary viewpoints for disambiguation. That is, we not only determine



Fig. 1. In distributed camera networks, some viewpoints are more discriminative for action recognition. Our method learns *which* viewpoints are most discriminative for *particular instances* of ambiguity.

if information from additional viewpoints is needed, but *which* viewpoints would provide the most discriminative power. Our two-stage learning algorithm casts the problem of combining views as that of learning weights for multiple kernel learning (MKL), and we build off the efficient algorithms that have been developed for MKL.

2 Related Work

The idea of using information from one view to inform view selection is often framed as active vision [2], usually in the context of a mobile agent. For example, in [3], agents perform object recognition using entropy maps, which model the predicted suitability of potential viewpoints to help determine the object. This differs from the distributed camera setting, in that active vision approaches are typically carried out in a sequential fashion (during, for example, robot navigation), while all the views are available simultaneously in multi-camera networks. For multi-view action recognition [4], there are two broad categories of methods: (1) 3D methods, which explicitly build a 3D model, and (2) 2D methods, like ours, which may incorporate multiple 2D views, but do not build 3D models.

3D action descriptors include motion history volumes [5,6], a compact representation of the animated visual hull of a person performing an action. While complex 3D models often achieve high recognition rates, they also tend to be computationally expensive. A recent hybrid 2D/3D approach determines the

best viewpoint of an action in isolation after constructing a 3D model [7]. This approach differs from ours in that it is focused on finding the single best view from the perspective of a human observer, while our method learns how to combine views to improve recognition. Other hybrid approaches that include strong 3D correspondences during the training phase include methods designed for cross-view recognition (e.g., [8,9]), which is a version of multi-view recognition where synchronized views from all views are not available for both training and testing.

For 2D multi-view recognition, there are a variety of approaches. One method models 2D feature descriptors as a function of viewpoint [10]. Farhadi et al. distinguish between geometric and discriminative aspect, where geometric aspect corresponds to canonical views (front, side), while discriminative aspect encodes the ways in which different things look similar or different [11]. Their method learns the parameters governing these latent variables and weights features accordingly to improve classification. These approaches of specifically learning a viewpoint manifold or discriminative aspect model are very different from ours in that we do not learn to distinguish viewpoints, but instead seek to recognize when additional viewpoints are needed and their relative utility. A number of approaches seek to identify view-invariant features (e.g., [12]) or extract features that exhibit low intra-class, but high inter-class variation (e.g., [13]). Our method is agnostic to the base feature; rather than relying on view-invariance, our method learns cases where disambiguation is needed.

In Sect. 3, we describe our classification approach and how we cast the view combination problem as kernel weight learning. By applying a kernel-based approach, our method is applicable to a variety of feature transforms used in action recognition, and in Sect. 4, we show high recognition rates on benchmark datasets for multi-view action recognition.

3 Approach

Figure 2 presents an overview of our approach to example disambiguation.¹ Each example, $\mathbf{X}_i = \langle \mathbf{x}_{i,0}, \dots, \mathbf{x}_{i,V-1} \rangle$ represents V videos of a particular action instance captured by a primary camera, $\mathbf{x}_{i,0}$, and $V - 1$ secondary cameras, ordered by their offset to the primary camera (as shown in Fig. 3). As illustrated in Fig. 3, each instance of an action could represent up to V different training examples corresponding to each of the V cameras serving as the primary camera. Each example \mathbf{X}_i has class label y_i . In this section, we describe our approach to identifying ambiguous examples, learning to disambiguate, and training the combined classifier. Finally we describe the complete algorithm in Sect. 3.3.

¹ For illustration, Fig. 2 depicts a 2-dimensional feature space, but each step of our procedure can be kernelized, so a direct feature vector representation of the data is not necessary.

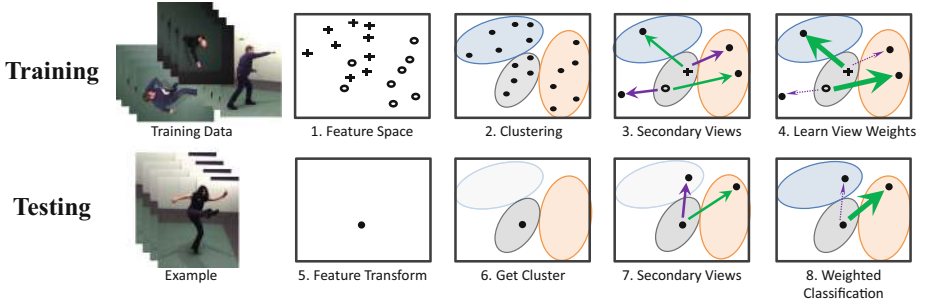


Fig. 2. The recognition procedure. For training, (1,2) the videos from all views are clustered. Some clusters (e.g., blue and orange) are homogeneous, while others (e.g., gray) contain examples from multiple classes. (3) In non-homogeneous clusters, secondary views are identified for each instance (green and purple arrows). (4) A classifier is trained that simultaneously learns the relative weights of each view for each cluster (represented by arrow thickness). For testing, (5,6) the input is assigned to a cluster. (7) If the cluster is non-homogeneous, secondary views are obtained, and (8) the input is classified using the weighted model learned for the cluster (color figure online).

3.1 Identifying Ambiguous Examples

The first step toward estimating view utility is identifying examples that are similar in appearance, but may represent different classes. Let $\kappa(\mathbf{x}, \mathbf{x}')$ be a kernel function, which we refer to as the *base kernel* that can be applied to the video feature representation. Using κ , we perform kernel k-means (KKM) [14] to partition the training data into C clusters, where each primary video, $\mathbf{x}_{i,0}$, receives cluster label $c_i \in [1, C]$. Even though KKM is unsupervised, with modern feature transforms designed for discriminative action recognition, a significant fraction of the clusters are *homogeneous*, containing only examples from a single class. These clusters demarcate regions of the feature space where input from a single, primary camera is sufficient for classification. However, the remaining *heterogeneous* clusters contain examples from multiple classes nearby in feature space, representing border cases that would typically be misclassified using standard classification methods. In each cluster, we will learn the relative utility of the secondary views for disambiguating the input (Sect. 3.2).

To disambiguate test examples, an efficient method to assign new examples to clusters is needed. Using the training examples, corresponding cluster labels $\{c_i\}$, and base kernel, κ , a support vector machine (SVM) is trained for *cluster assignment classification*. Methods exist for unsupervised support vector clustering that return support vectors suitable for cluster assignment classification. However, these methods tend to return large numbers of support vectors [15], so this two-stage (cluster-then-classification) approach can be more efficient for evaluating the SVM at run-time.

A common approach to handling multiple representations of the same object is feature vector concatenation. For multi-view recognition, this corresponds to

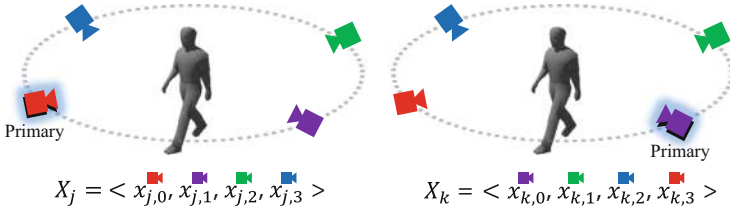


Fig. 3. For a given primary view, the offsets to other cameras are *relative*. The same instance of an action leads to different feature representations depending on which camera is selected as primary. On the left, selecting the red camera to be primary leads to \mathbf{X}_j , while, on the right, selecting the purple camera as primary leads to \mathbf{X}_k . With recorded data, during the training phase, each instance of an action could represent up to V different examples. For testing, the primary camera is determined as part of distributed network sensing. (Figure best viewed in color.) (color figure online)

concatenating the feature representations of the same action from different viewpoints. One drawback to using this combined representation directly in a standard supervised learning scheme is that multiple viewpoints are incorporated even if one of the viewpoints is highly discriminative, and, worse, the addition of a difficult, ambiguous feature vector to the joint representation can result in misclassification in some cases. Our approach, for a given ambiguity, is to learn the relative utility of secondary viewpoints for discrimination. The rest of this section explains how we cast the problem of estimating view offset utility as kernel weight learning.

3.2 Learning Discriminative View Combinations

For kernel-based methods, such as support vector machines (SVM), selecting an appropriate kernel function is a key step. There are a number of common choices (e.g., linear, RBF) and custom kernels (e.g., spatial pyramid kernel [16] for sets of local image features). Multiple kernel learning (MKL) has emerged as an alternative to simply selecting a single kernel function, with multiple approaches to learning weighted combinations of kernels [17]. MKL is often used to fuse different types of features, and has been used in this way for action recognition (e.g., [18]). By contrast, we propose to use multiple kernel learning for weighted view selection, rather than feature weighting.

Kernel functions can be combined; for example, a linear combination of kernels is itself a kernel. That is, $\kappa_\beta(\cdot) = \sum_{i=1}^M \beta_i \kappa_i(\cdot)$, where β is a vector of weights for M kernels, $\{\kappa_i\}$. In the case of convex β (i.e., $\beta \in \mathbb{R}_+^M$ and $\sum \beta_i = 1$), the weights represent the importance of each kernel and the combined kernel represents the similarity in a feature space defined by the concatenation of the individual feature vectors. While most of the focus in MKL has been on linear combinations of kernels, recent work [19] with nonlinear kernel combinations has shown they can be more effective. For our approach, we use the quadratic kernel:

$$\kappa_{\beta}(\mathbf{X}_i, \mathbf{X}_j) = \sum_{m=0}^{V-1} \sum_{l=0}^{V-1} \beta_m \beta_l \kappa(\mathbf{x}_{i,m}, \mathbf{x}_{j,m}) \kappa(\mathbf{x}_{i,l}, \mathbf{x}_{j,l}) \quad (1)$$

where each kernel represents one of the viewpoints in the system.

Many approaches to multiple kernel learning incorporate the problem of learning kernel weights with the optimization of the classification margin. However, as the complexity of the kernel combination increases, so does the complexity of the resulting optimization problem. For example, incorporating Eq. 1 into the usual SVM optimization leads to a difficult non-convex optimization. We apply an iterative approach [20], modified for SVM [17], to this MKL optimization:

$$\min_{\beta \in \mathcal{M}} \max_{\alpha \in \mathbb{R}^N} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \kappa_{\beta}(\mathbf{X}_i, \mathbf{X}_j) \quad (2)$$

where \mathcal{M} is the norm-1 bounded set which constrains β to be convex. This optimization can be solved iteratively, alternately finding α by solving the inner SVM problem with the current combined kernel, κ_{β} , and then finding the weights, β , using projection-based gradient descent.

3.3 Algorithm

Training Consider a set of N actions, each captured from V different viewpoints. This gives a training set of $N \times V$ labeled examples, where example \mathbf{X}_i has class label y_i . Let $\kappa(\mathbf{x}, \mathbf{x}')$ be the *base kernel*, which can be applied to the video feature representation. The two-stage training procedure is as follows:

1. Cluster the training videos to learn cluster labels $\{c_i\}$.
 - (a) Using the base kernel, κ , and cluster labels, train cluster assignment SVM, $\mathcal{C}_{cluster}$.
 - (b) For homogeneous clusters, no further work is needed.
2. For each non-homogeneous cluster, c_i , train multi-class SVM classifier (Eq. 2), \mathcal{C}_{c_i} , and learn cluster weights, β_{c_i} , with the classes represented in cluster c_i .

Classification. Given a new query example (action captured from V different views), \mathbf{X}_q , the classification procedure follows an analogous two-stage approach.

1. Using SVM, $\mathcal{C}_{cluster}$, get cluster, c_q , for $\mathbf{x}_{q,0}$.
2. Classify $\mathbf{x}_{q,0}$:
 - (a) If c_q is homogeneous, \mathbf{X}_q is classified immediately.
 - (b) Otherwise, \mathbf{X}_q is classified using SVM \mathcal{C}_{c_q} , and cluster-specific weights, β_{c_q} .

Using this locality-sensitive approach, similar to [21], kernel combinations can be applied in a data-dependent fashion rather than learning a global combination across the whole input space.

4 Results

We evaluated our method on three publicly-available multi-view recognition datasets. Our algorithm was implemented in Matlab on a standard PC, using `libsvm` [22] for support vector classification. For multi-class classification, we use the one-versus-all (OVA) approach. The SVM cost parameter, C_{SVM} , was selected using cross-validation for each experiment. For clustering, we initialized kernel k -means 25 times and selected the cluster assignment with minimum energy, as measured by average intra-class similarity.

4.1 Action Recognition

Our approach is neither specific to particular camera configurations nor video feature representations. We evaluated our algorithm using two widely-used multi-view human action datasets with two different feature representations.

Data Sets. For these experiments, we used the i3DPost multi-view human action data set [23] and the INRIA Xmas Motion Acquisition Sequences (IXMAS) data set [5]. i3DPost includes 10 actions performed by 8 actors recorded by 8 synchronized cameras. IXMAS, which is commonly used as a benchmark for multi-view action recognition, contains multiple actors performing various actions, captured by five synchronized cameras. Compared with i3DPost, IXMAS is more challenging, containing fewer cameras, more easily confused actions, and more instances of self occlusion. Figure 4 shows sample frames from both datasets.

Features. To demonstrate the ability of our method to handle different underlying motion features, we used the Motion Context (MC) transform [24] and the \mathcal{R} Transform Surface (RT) video descriptor [10]. MC computes frame-based histograms of the localized optic flow and silhouette occupancy. The base kernel for an image sequence is the histogram intersection kernel on quantized

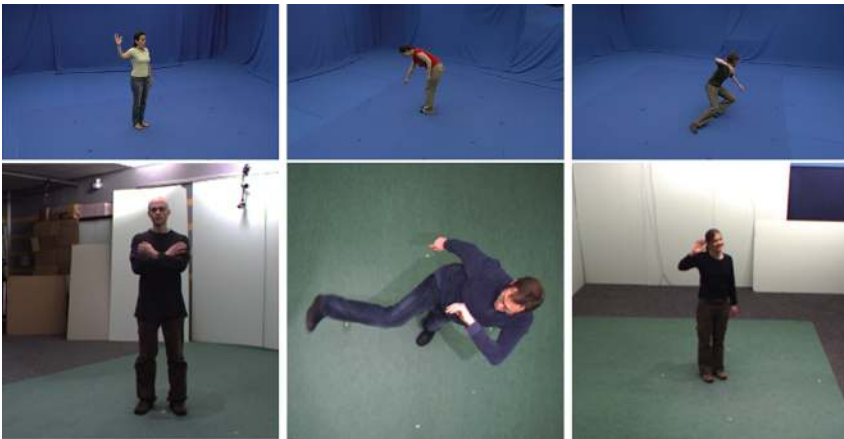


Fig. 4. Example frames from the i3Dpost (top) and IXMAS (bottom) data sets.

features, with a dictionary size, $|D| = 800$. RT is a silhouette-based descriptor that extends the Radon transform. The base kernel is the Gaussian radial basis function of the diffusion distance [25] between the base features.

Methods. Using the same features, base kernel, and free parameters, we compare the performance of our multi-view learning (mv1) approach to several other kernel-based schemes for multi-view classification:

View-Insensitive SVM (svm). The basic classifier where the SVM is trained with the base kernel without respect to viewpoint.

SVM Voting (vote). An example is classified from each viewpoint and the majority decision serves as the final classification.

Weighted SVM Voting (wvote). Similar to *vote*, but SVM posterior probabilities [26] are used to weight each vote.

Winner Take All (wta). Using SVM posterior probabilities as a proxy for classifier confidence, the most confident classifier provides the final classification.

Uniform-Weighting SVM (uwsvm). Each example is represented as the concatenation of all the views. This is equivalent to using the uniformly-weighted convex combination of base kernels for each view representation.

Table 1. Multi-view classification rates on the i3DPost and IXMAS data sets. (Top) Rates for our approach, mv1, and kernel-based variants using two different feature representations (RT & MC). (Bottom) Representative recognition rates reported in the literature on IXMAS using the same experimental protocol.

Method	i3Dpost		IXMAS	
	RT	MC	RT	MC
mv1	73.75%	96.25%	90.91%	93.58%
uwsvm	72.50 %	95.00 %	87.27 %	92.42 %
vote	66.56 %	93.75 %	81.82 %	89.15 %
wvote	66.25 %	92.50 %	75.09 %	82.79 %
wta	73.75%	93.75 %	74.18 %	79.39 %
svm	63.75 %	91.25 %	71.88 %	77.15 %
Wu et al. [18]	-		88.2 %	
Zhu et al. [27]	-		88.0 %	
Parrigan and Souvenir [28]	-		84.0 %	

Results. Table 1 shows the multi-view classification rates for both data sets with both feature representations. In general, our method, mv1, outperformed competing approaches, except for an unusual case where an otherwise underperforming method, wta, matched our performance for a particular feature-dataset combination.

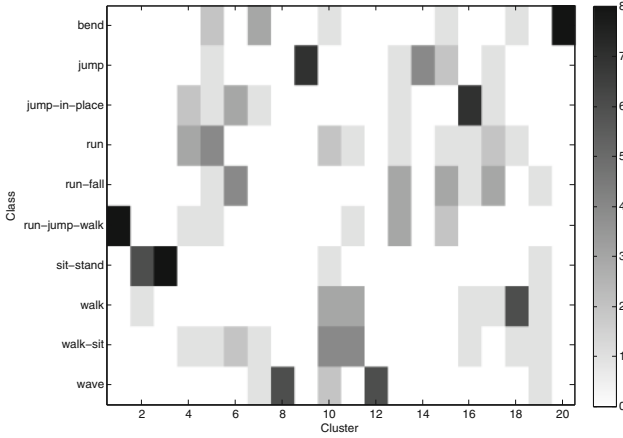


Fig. 5. Distribution of class labels in clusters for i3DPost. In most cases, clusters were either homogeneous (e.g., clusters 8 and 12 contain only wave actions) or represented specific instances of ambiguity (e.g., run and run-fall tend to co-cluster).

For i3DPost, we employed 2-fold cross-validation, with half the actors in each set, and included 4 views and 10 actions. For both features, *mv1* returned the highest classification rates (except as noted above), improving on the baseline kernel classification by up to 10%. Figure 5 shows an example from an experiment with 20 clusters and the MC features. Several of the clusters are homogeneous. For example, cluster 1 contains only run-jump-walk actions, while clusters 8 and 12 contain only wave actions. Overall in this experiment, 34% of test examples fell into homogeneous clusters, which results in immediate (single-camera) classification and computational savings compared to the other aggregation approaches that always incorporate each viewpoint.

There does not appear to be a commonly-used experimental protocol in the literature for i3Dpost. To our knowledge, the best reported accuracy is 97.50% using a full 3D (not 2D multi-view) approach [29]. In addition to the inherent complexity of 3D reconstruction, this method uses leave-one-actor-out (LOAO) cross-validation, where for each fold, a single actor is used for testing, while the remainder are used for training. With LOAO on i3DPost, our *mv1* method also resulted in 97.50% accuracy. However, the other multi-view methods also performed well, so this may be a function of the complexity of the dataset rather than the effectiveness of the classification approaches.

Compared to i3Dpost, IXMAS contains more self-occlusion and similar-looking actions. For IXMAS, we followed the LOAO protocol, which is the experimental setup most commonly found in the literature (see, e.g., [30]). The experiment uses 10 actors, each performing one of 11 actions three times. As shown in Table 1, *mv1* far outperforms the baseline *svm* and voting approaches. While the MC feature outperformed RT on each test, our method boosted the performance of both feature descriptors. The 93.58% classification accuracy

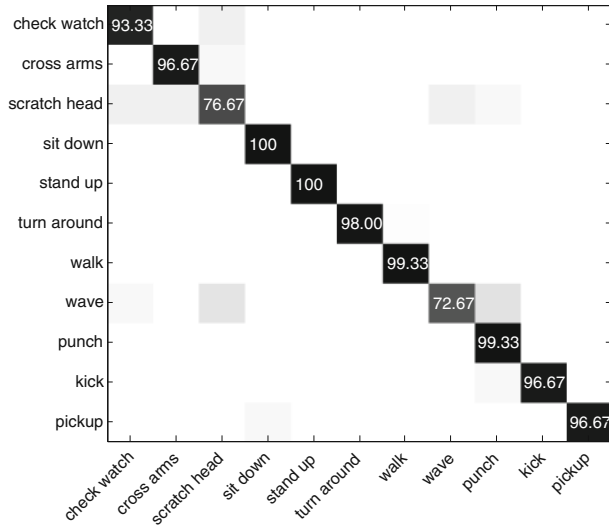


Fig. 6. Confusion matrix for mv1 classification on IXMAS data. Each row represents the actual class and each column represents the predicted class. Our method achieved 93.58 % accuracy.

achieved by our mv1 approach not only out-performs other recent 2D methods, it approaches the performance of 3D approaches, but without the computational expense of building 3D models.

For mv1, Fig. 6 shows the confusion matrix for the classification experiment on IXMAS with the MC feature, where each row represents the actual class and each column represents the predicted class. For many actions (e.g., sit, stand, turn, walk, pickup), accuracy was above 95 %. The most challenging cases involved confusions between waving, punching, and scratching head. These results are expected as the base descriptors are primarily silhouette-based, and these motions include self-occlusion from most viewpoints. Additionally, by comparing the per-class accuracies between mv1 and the best-performing 2D method from the literature [18], we find that the achieved per-class accuracy for eight of the actions is similar (and relatively high). However, for three actions (check watch, cross arms, scratch head), our method shows noticeable gains: from 78 % to 93 % for check watch, 83 % to 97 % for cross arms, and 72 % to 77 % for scratch head. This suggests that the superior performance of mv1 is mainly due to correctly classifying the most challenging cases of confusion, which, for our approach, tend to fall in heterogeneous clusters.

As the same base kernel was used for both clustering and classification, the clustering step provides an initial partition of the data into groups that are either mostly homogeneous with regards to the class of the examples or require disambiguation. For the IXMAS data with $C = 50$ clusters and RT feature, Fig. 7 shows the distribution of class labels in each cluster. It can be seen that

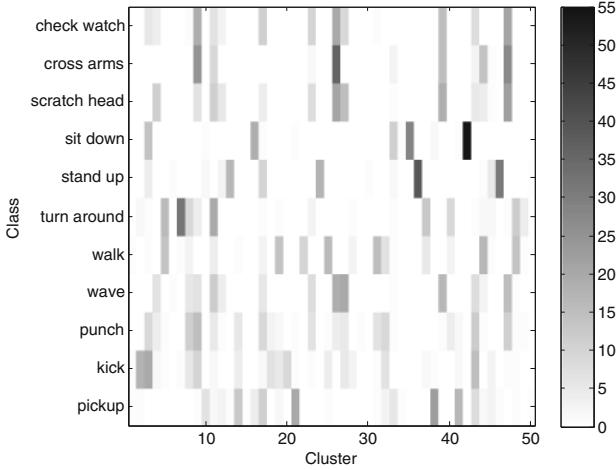


Fig. 7. Distribution of class labels in clusters for IXMAS. In most cases, clusters were either homogeneous (e.g., clusters 35 and 42 contain only sit actions) or represented specific instances of ambiguity (e.g., check watch, cross arms, wave, and scratch head tend to co-cluster).

the classes that resulted in the highest recognition rates (e.g., sit, stand, turn) tend to lie in homogeneous clusters. For example, clusters 35 and 42 contain only sit actions, while cluster 7 contains only turn. By contrast, classes that were more likely to be confused by our method tend to lie in more heterogeneous clusters. The most frequently confused action, punch, is present in 28 of the 50 clusters in this experiment.

Figure 8 shows examples views of actions that were co-clustered during an experiment. The top row contains examples from a cluster containing primarily kick and walk motions. From the given viewpoint, these actions are visually similar and indistinguishable for most feature transforms. The bottom row shows examples from a cluster containing punches and kicks from a viewpoint directly overhead. In this case, the silhouette-based descriptors are similar for these semantically different actions. However, incorporating almost any of the other viewpoints would serve to disambiguate this confusion and correctly classify these actions. Figure 9 shows two examples of the weighting learned by our method for secondary views. For each row, the left image is a frame from an action that was not immediately classified. The next three images show frames from secondary viewpoints, sorted in order of weights learned by our method. The highest weighted viewpoint often corresponds to views that are highly discriminative, with minimal self-occlusion.

4.2 Object Recognition

While our method was originally intended for action recognition from video, we also applied it to object recognition from images. The 3D object data set

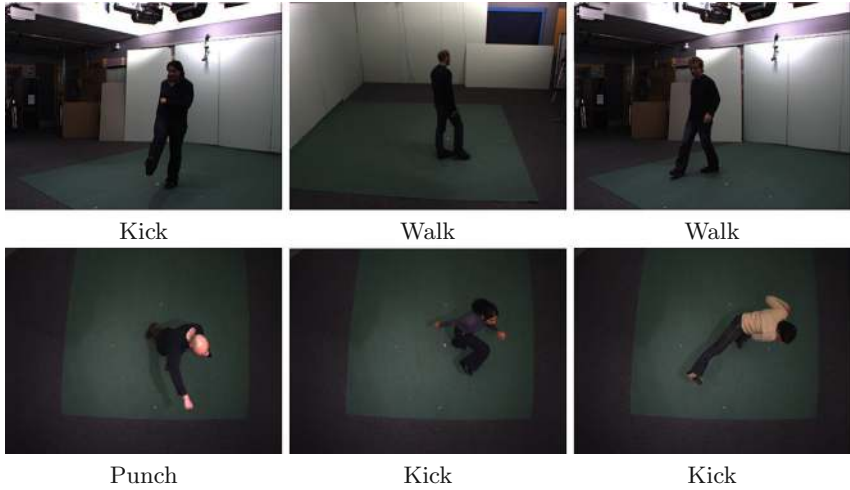


Fig. 8. Each row shows a set of three example frames from viewpoints that were co-clustered by our algorithm. The label indicates the action represented.



Fig. 9. For each row, the left image is a frame from an action that was not immediately classified. The following images show frames from secondary viewpoints, sorted in order of weights learned by our method.

(3DO) [31] consists of examples of 10 object categories from several different views, elevations, and scales. Figure 10 shows some example images from the data set. We selected a subset of seven categories for which 10 examples were available with four different views at the same elevation and scale (bicycle, car, cellphone, head, iron, monitor, and mouse). For each image, we calculated SIFT descriptors for 16×16 patches on a regular grid, with a spacing of 8, and used the Spatial Pyramid [16] (number of levels, $L = 3$, and the vocabulary size, $M = 200$) as the base kernel. For evaluation, we divided the data in half and performed 2-fold testing. The basic svm approach achieved 81.43% accuracy.



Fig. 10. Example images from the 3DO data set.



Fig. 11. Example images that were co-clustered by our method.

Integrating multiple views was beneficial as *vote* achieved 87.50% accuracy. Our approach, *mv1*, achieved 90.71% accuracy.² Like the action recognition data, classification ambiguity in this data set appears to be tied to ambiguous viewpoints. Figure 11 shows an example of co-clustered object views. Unlike the examples for action recognition, for this data set, it is not always apparent that co-clustered images should share similar feature representations. Nonetheless, the results demonstrate that incorporating weighted view aggregation provides additional discriminative power.

5 Conclusions and Future Work

In this paper, we presented a multi-view recognition method for identifying when input is ambiguous and learning the relative utility of secondary views as a function of the particular ambiguity. We cast the problem of viewpoint weighting as kernel weight optimization for multiple kernel learning, localized in feature space, and took advantage of efficient solutions in the MKL domain. A significant amount of research effort in the action recognition domain focuses on devising new, more discriminative, feature representations. However, by applying a strategy to account for the inherent ambiguity in certain poses from certain viewpoints, our method boosted the performance of existing features and achieved high recognition accuracy on a benchmark action recognition data set, outperforming other recent 2D multi-camera methods, and similar to 3D approaches,

² We were unable to directly compare these results with those previously reported. Unlike with the IXMAS set, there is little agreement in the literature on an experimental protocol for 3DO.

with reduced computational effort. For future work, we plan to investigate early identification of potentially ambiguous actions and learn the *single* view shift that would best disambiguate the input to allow for dynamic camera switching in distributed camera networks on untrimmed video.

References

1. Wang, X.: Intelligent multi-camera video surveillance: A review. *Pattern Recogn. Lett.* **34**(1), 3–19 (2012)
2. Dutta Roy, S., Chaudhury, S., Banerjee, S.: Active recognition through next view planning: a survey. *Pattern Recogn.* **37**, 429–446 (2004)
3. Arbel, T., Ferrie, F.: Viewpoint selection by navigation through entropy maps. In: *Proceedings of the International Conference on Computer Vision*, vol. 1, pp. 248–254 (1999)
4. Weinland, D., Ronfard, R., Boyer, E.: A survey of vision-based methods for action representation, segmentation and recognition. *Comput. Vis. Image Underst.* **115**, 224–241 (2011)
5. Weinland, D., Ronfard, R., Boyer, E.: Free viewpoint action recognition using motion history volumes. *Comput. Vis. Image Underst.* **104**, 249–257 (2006)
6. Turaga, P., Veeraraghavan, A., Chellappa, R.: Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8 (2008)
7. Rudoy, D., Zelnik-Manor, L.: Viewpoint selection for human actions. *Int. J. Comput. Vis.* **97**, 243–254 (2012)
8. Li, R., Zickler, T.: Discriminative virtual views for cross-view action recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2855–2862 (2012)
9. Zhang, Z., Wang, C., Xiao, B., Zhou, W., Liu, S., Shi, C.: Cross-view action recognition via a continuous virtual path. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2690–2697 (2013)
10. Souvenir, R., Babbs, J.: Learning the viewpoint manifold for action recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–7 (2008)
11. Farhadi, A., Tabrizi, M., Endres, I., Forsyth, D.: A latent model of discriminative aspect. In: *Proceedings of the International Conference on Computer Vision*, pp. 948–955 (2009)
12. Ali, S., Shah, M.: Human action recognition in videos using kinematic features and multiple instance learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**, 288–303 (2010)
13. Sharma, A., Kumar, A., Daume, H., Jacobs, D.: Generalized multiview analysis: A discriminative latent space. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2160–2167 (2012)
14. Dhillon, I., Guan, Y., Kulis, B.: Kernel k-means: spectral clustering and normalized cuts. In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 551–556. ACM (2004)
15. Zhang, K., Tsang, I., Kwok, J.: Maximum margin clustering made practical. *IEEE Trans. Neural Netw.* **20**, 583–596 (2009)

16. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 2169–2178. IEEE (2006)
17. Gönen, M., Alpaydm, E.: Multiple kernel learning algorithms. *J. Mach. Learn. Res.* **12**, 2211–2268 (2011)
18. Wu, X., Xu, D., Duan, L., Luo, J.: Action recognition using context and appearance distribution features. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 489–496 (2011)
19. Levinboim, T., Sha, F.: Learning the kernel matrix with low-rank multiplicative shaping. In: Proceedings of the National Conference on Artificial Intelligence (2012)
20. Cortes, C., Mohri, M., Rostamizadeh, A.: Learning non-linear combinations of kernels. In: Advances in Neural Information Processing Systems 22, pp. 396–404 (2009)
21. Yang, J., Li, Y., Tian, Y., Duan, L., Gao, W.: Group-sensitive multiple kernel learning for object categorization. In: Proceedings of the International Conference on Computer Vision, pp. 436–443 (2009)
22. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2**, 27:1–27:27 (2011)
23. Gkalelis, N., Kim, H., Hilton, A., Nikolaidis, N., Pitas, I.: The i3dpost multi-view and 3d human action/interaction database. In: Conference for Visual Media Production, pp. 159–168. IEEE (2009)
24. Tran, D., Sorokin, A.: Human activity recognition with metric learning. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 548–561. Springer, Heidelberg (2008)
25. Ling, H., Okada, K.: Diffusion distance for histogram comparison. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 246–253. IEEE Computer Society, Washington, DC (2006)
26. Lin, H.T., Lin, C.J., Weng, R.C.: A note on platts probabilistic outputs for support vector machines. *Mach. Learn.* **68**, 267–276 (2007)
27. Zhu, F., Shao, L., Lin, M.: Multi-view action recognition using local similarity random forests and sensor fusion. *Pattern Recogn. Lett.* **24**, 20–24 (2012)
28. Parrigan, K., Souvenir, R.: Aggregating low-level features for human action recognition. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Chung, R., Hammoud, R., Hussain, M., Kar-Han, T., Crawfis, R., Thalmann, D., Kao, D., Avila, L. (eds.) ISVC 2010, Part I. LNCS, vol. 6453, pp. 143–152. Springer, Heidelberg (2010)
29. Holte, M.B., Chakraborty, B., Gonzalez, J., Moeslund, T.B.: A local 3-d motion descriptor for multi-view human action recognition from 4-d spatio-temporal interest points. *IEEE J. Sel. Top. Sig. Process.* **6**, 553–565 (2012)
30. Weinland, D., Boyer, E., Ronfard, R.: Action recognition from arbitrary views using 3d exemplars. In: Proceedings of the International Conference on Computer Vision, pp. 1–7 (2007)
31. Savarese, S., Fei-Fei, L.: 3d generic object categorization, localization and pose estimation. In: Proceedings of the International Conference on Computer Vision, pp. 1–8 (2007)