Learning to Recognize Activities from the Wrong View Point

Ali Farhadi[1] and Mostafa Kamali Tabrizi[2]

[1] Computer Science Department, University of Illinois at Urbana Champaign
afarhad2@uiuc.edu
[2] Institute for Studies in Theoretical Physics and Mathematics
kamali@ipm.ir

Abstract. Appearance features are good at discriminating activities in a fixed view, but behave poorly when aspect is changed. We describe a method to build features that are highly stable under change of aspect. It is not necessary to have multiple views to extract our features. Our features make it possible to learn a discriminative model of activity in one view, and spot that activity in another view, for which one might poses no labeled examples at all. Our construction uses labeled examples to build activity models, and unlabeled, but corresponding, examples to build an implicit model of how appearance changes with aspect. We demonstrate our method with challenging sequences of real human motion, where discriminative methods built on appearance alone fail badly.

1 Introduction

Human activity recognition is a core unsolved computer vision problem. There are several reasons the problem is difficult. First, the collection of possible activities appears to be very large, and no straightforward vocabulary is known. Second, activities appear to compose both across time and across the body, generating tremendous complexity. Third, the configuration of the body is hard to transduce, and there is little evidence about what needs to be measured to obtain a good description of activity.

There is quite good evidence that many activities result in strong spatio-temporal patterns of appearance, and most feature constructions are based on this observation. However, such constructions are bound to a view — if the camera rotates with respect to the body, a disastrous fall in discriminative performance is possible (see Section 6). One strategy is to learn a new model for each view; but it isn't practical to obtain several examples of each activity in each view. An alternative, which we expound in this paper, is to *transfer* models between views. We do so by building models in terms of features which are stable with change of view, yet discriminative.

2 Background

There is a long tradition of research on interpreting activities in the vision community (see, for example, the extensive surveys in [18,20]). Space allows touching only on most relevant points.

D. Forsyth, P. Torr, and A. Zisserman (Eds.): ECCV 2008, Part I, LNCS 5302, pp. 154–166, 2008.

Appearance features are widely used. Generally, such features encode (a) what the body looks like and (b) some context of motion. At low spatial resolution when limbs cannot be resolved, flow fields are discriminative for a range of motions [3]. At higher resolutions, appearance features include: braids [27]; characteristic spatio-temporal volumes [7]; motion energy images [9]; motion history images [9]; spatio-temporal interest points [23]; nonlinear dimensionality reduced stacks of silhouettes [38]; an extended radon transform [40]; and silhouette histogram of oriented rectangle features [21].

Primitives: Motions are typically seen as having a compositional character, with hidden Markov models used to recognize, among others: tennis strokes [45]; pushes [43]; and handwriting gestures [46]. Feng and Perona [17] call actions "movelets" and build a vocabulary by vector quantizing a representation of image shape. These codewords are then strung together by an HMM, representing activities; there is one HMM per activity, and discrimination is by maximum likelihood. Alternatively, Bregler fits a switching linear dynamical system ([10]; see also [37]); Ikizler and Forsyth build primitives explicitly by clustering motion capture [21].

Aspect: Appearance features are subject to aspect problems, which are not generally studied. One alternative is to model in 3D. There is a strong evidence that 3D configuration can be inferred from 2D images (e.g., [19,6,33]; see also discussion in [18]). At higher spatial resolutions one can recover body configuration and reason about it [31,21], and, though inferred body configuration can be quite noisy, there is some evidence this leads to aspect invariant methods [21], or learn in 3D and recognize in 2D [41]. Alternatively, one might use an implicit representation of 3D shapes as a set of silhouettes from multiple views [2,26], spatio-temporal volumes derived from such a silhouettes [42,47], or spatio-temporal curvature of 2D trajectory [32]. Things are more difficult when only one view is available, on which to compute feature values. View invariant motion representations offer one solution, but one must use model-based invariants [29].

Transfer Learning: Transferring knowledge across related tasks is a known phenomenon in human learning [11]. In machine learning, simply pooling the training data does not necessarily work because decision boundaries in the feature space may not be similar for similar tasks. For example, in Figure 2 corresponding frames in camera 1 and camera 4 are totally different.

An extensive literature review is available at [22]. To summarize, two main categories of approach have been used in the literature. One is to use models that are robust under change of domain. One might: use similar tasks to reasonably estimate the prior [48,30,**?**]; use hierarchical bayesian models with hyper priors constrained to be similar for similar tasks [5,25,44]; consider a regularized multi task learning framework, when there is a trade-off between large margins and similar decision boundaries [15]; boost the prediction power of the most helpful out of domain data [12]; or transfer rules in a reinforcement learning framework [34,44].

The second category of approach is to design features which are well behaved under change of domain. The main idea is to have a common feature mapping in which similar objects in different domains look similar, and dissimilar objects look different, even in the same domain. To do this, one might introduce some auxiliary artificial tasks to

uncover the shared structure among similar tasks [4], or learn a distance function which behaves well under transfer [35].

Most similar to our work is the approach of Farhadi, Forsyth and White [16], who construct the space of comparative features. These features compare new words (in sign language) with base words for a view, and in turn, use the result of the comparison as a feature.They demonstrate that, once a semantically similar feature space has been constructed, models of sign language words learned from an animated dictionary alone can be used to recognize sign language words produced by a human signer in different aspects.They claim that since comparisons are transferable, the comparative features will be well behaved under change of domain. We believe that the random searches on the split space,used in [16], result in a noisy feature space. This, we think, is the main reason they needed a topic model on the top of their comparative features. We differ from their work in that: we have a more focused search mechanism for finding stable features and consequently we don't need to have a topic model; and our process appears to apply quite generally, rather than in the specific domain of ASL.

Many vision problems are naturally seen as transfer learning problems (for example, the standard problem of determining whether two face images match without ever having seen images of that individual; as another example, one might use cartoons to learn the location of object features, and very few real images to learn their appearance [14]).

3 Activity Model

We first describe frames of activities and their context. We then look at the long timescale. We describe frames by vector quantization of local appearance features to form the codewords. We match based on the frequencies with which codewords appear in a sequence. In the object recognition literature, local texture codewords are strongly discriminative [13], without spatial information. Similarly, we do not represent precise temporal orderings. Quantitative evaluations show strong discriminative performance for these activity descriptors. See Table 2 and 3 for detailed accuracies.

Descriptive Features: We employ the feature extraction technique of [36]. Their descriptor is a histogram of the silhouette and of the optic flow. Given the bounding box of the actor and the silhouette, the optic flow is computed using Lucas-Kanade algorithm [24]. They consider using a normalized size feature extraction window. Features consist of three channels, smoothed horizontal flow, smoothed vertical flow, and the silhouette. The normalized feature extraction window is divided into 2×2 sub-windows. Each sub-window is then divided into 18 pie slices covering 20 degrees each. The center of the pie is in the center of the sub-window, and the slices do not overlap. The values of each channel are integrated over the domain of every slice. The result is a 72-dimensional histogram. By concatenating the histograms of all 3 channels we get a 216-dimensional frame descriptor.

To encode local temporal structure of the activities we consider stacking features from previous and next frames. We pick the first 50 principal components of the descriptors of a window of size 5, centered at the frame we want to describe. For further frames in both directions, previous and next frames, we pick the first 5 principal components of the windows of size 5, centered at the $(i+5)^{th}$ and $(i-5)^{th}$ frames. This gives us a 60 dimensional descriptor for each frame, which we call descriptive features.

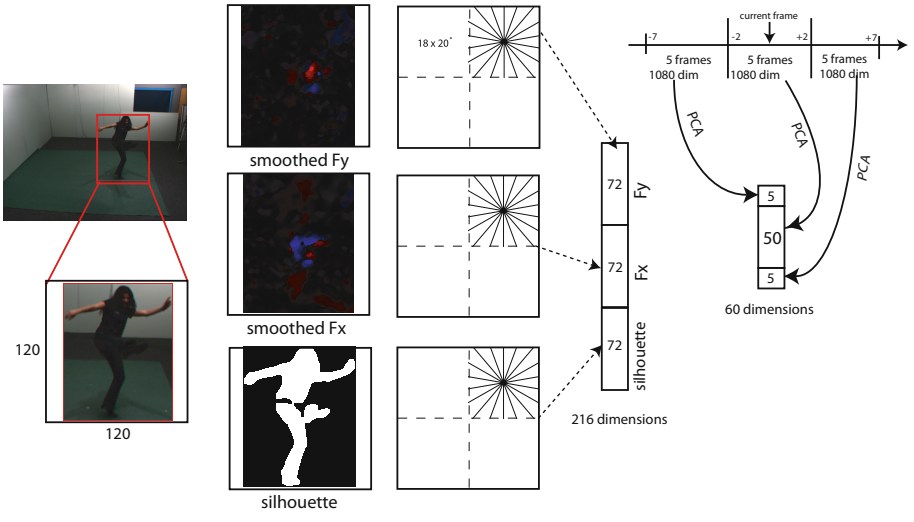

Fig. 1. Feature Extraction: Each frame is described by silhouette information, horizontal and vertical optical flow on a normalized size feature extraction window. We histogram all three channels of information by integration over angular slices of subwindows of the bounding box. We then stack information about previous and next frames to model local dynamics of activities.

Vector quantization: There is evidence that motions are composites of short timescale primitives [17]. Temporal order over short to medium timescales does not seem to be an important discriminative cue, so we can use a bag of features model. This suggests vector quantizing frame descriptors to form codewords. We use K-means clustering.

Matching: Each activity is described by a histogram of codewords, which we call codeword-based description. We use a 1 Nearest Neighbor (1NN) classifier to recognize an activity. We use hamming distance for matching the histograms.

Figure 1 depicts the feature extraction procedure. These features are discriminative, 98.92% average accuracy on Weizman10 dataset [8] is reported in [36] using a 1NN classifier on these features, comparing to 97.78% average accuracy of [39] and 82.6% average accuracy of [28] on the same dataset.

4 Transferable Activity Model

We consider two types of activities: *shared* activities, that are observed in both *source* and *target* views, and *orphan* activities that are observed only in the source view. Shared activities in both the source and the target view are not labeled, while orphan activity labels are available in the source view. We would like to transfer orphan activity models from the source to the target view. This means that we would like to learn a model for an orphan activity in the source view and test it in the target view.

Activities look different from different view points. Each column in Figure 2 shows how much a particular frame in an activity looks different across view points. This

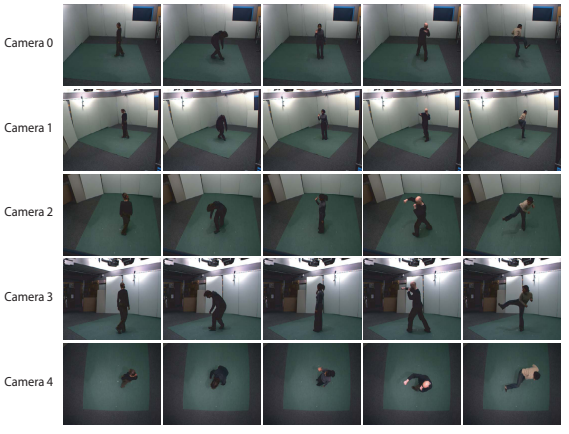

Fig. 2. The IXMAS dataset, Each column shows changes across the views. Because of these differences learning a model of descriptive features in one view and testing it in another view fails badly. Camera 4 has a weird configuration, introducing ambiguities in action classification for this view.

suggests that, under transfer circumstances, appearance based description will work poorly. It does; a 1NN classifier gets 14% average accuracy (leave one action out) when trained on one camera and tested in another one (for detailed accuracies see Table 2).

To be able to transfer activity models from the source to the target view, we need discriminative features that tend to be similar in different views. Codeword-based representation of activities inside each view seems to be discriminative. We could cluster both the source and the target views. But this is not enough because we do not know which cluster in the target view corresponds to which one in the source view, so we could not transfer a model. We need a method to describe each view such that correspondence is straightforward. For this reason, we build a cell structure on top of the clusters in the source view.

Shared activities are valuable because they demonstrate what the body will look like in the target view, for a given appearance in the source view. Our strategy is to build a discriminative problem around the clusters in the source view. We construct a structure of cells within which clusters lie. These cells are produced by an arrangement of hyperplanes, which we call *splits*. We use the shared activities to train this discriminative task. For shared activities we *choose* which side of each hyperplane in the source view a frame lies. We can now force the corresponding frame in the target view to be on the same side of the corresponding hyperplane in the target view. Figure 3 shows the procedure for making these new features, which we call *split-based* descriptors. If we have enough shared activities, we can establish correspondences between views.

The core idea here is that splits are transferable because we learn them to be. This means that the split-based features of a frame in the source view are similar to the split-based features of the corresponding frame in the target view. As a result, an orphan activity model can be applied to both views, if it is built using split-based features.

4.1 Building Splits in the Source View

We start by clustering the source view to form the codewords. We can now describe activities in the source view by a histogram of the codewords. Our split-based features take the form

$$f_i(x_j^v) = sign(\alpha_i^v \cdot \Phi(x_j^v)). \quad (1)$$

Where $f_i(x_j^v)$ is the i^{th} component of the feature vector of the j^{th} frame in view v, α_i^v is the i^{th} split coefficient in view v and $\Phi(x_j^v)$ is the appearance feature for the j^{th} frame in view v . We would like these features to be discriminative, in the sense that different clusters in one view should have distinct features. These features should also be transferable, meaning that

$$f_i(x_j^s) = f_i(x_{\hat{j}}^t). \quad (2)$$

Where $\hat{j}$ in the target view, is the corresponding frame to frame j in the source view. When we see activities simultaneously from two different views, we get temporal correspondences between frames in different views for free. We can now choose split-based features that are discriminative for the source view. We expect these features, and so orphan models, to transfer because we can produce training data to train on the target view. Training data is the appearance feature in the target view and training labels are, in fact, the split-based descriptors in the source view.

How to choose splits? A random choice of splits was used in [16]. There is a better way. Generally, clusters form blobby structures in the feature space, so cluster members are usually close together. We want splits to form a cell structure that respects the clusters. This implies that splits that do not break up clusters are better. This suggests choosing splits that have large margins with respect to cluster centers. We obtain them by Maximum Margin Clustering (MMC).

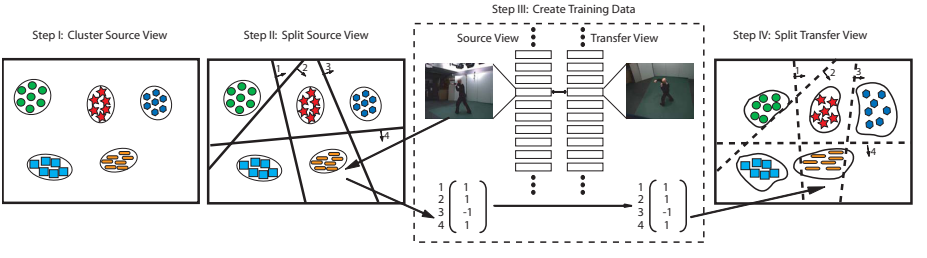

Fig. 3. We first cluster the source view using the original descriptive features. We then use Maximum Margin Clustering to choose informative splits in the source view. Split-based features are constructed by checking which side of each split a particular frame lies. We can directly transfer the split values to the corresponding frames in the target view. Using unlabeled shared activities, which have established implicit correspondences, splits are transferred from the source to the target view. We now can learn splits in the target view, using the descriptive features as training data and transferred split values as labels. These split-based features are transferable, and one could simply use an already learned model in the source view to recognize orphan activities in the target view.

Feature selection is a problem here. Similar to [16], we use several different random projections of the data. We apply MMC to each random projection to get several splits. Different projections usually yield different splits. We discard redundant splits and score the remainder by how well they can be predicted from the data.

Max Margin Clustering. For each random projection we look for the maximum margin hyperplane separating cluster centers in the source view. This lowers the computational burden of finding splits.

One can find these maximum margin hyperplanes by solving a Semi-Definite Programming problem, which is computationally expensive. However, [49] introduce an alternating optimization approach on the original non-convex problem of:

$$\begin{aligned} \min_y \min_{\omega,b,\xi_i} \; & \| \omega \|^2 + 2C\xi^T e \\ s.t. \quad & y_i(\omega^T \varphi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0 \\ & y_i = \{\pm 1\}, -\ell \leq e^T y \leq \ell \end{aligned} \tag{3}$$

where ω, b are SVM parameters, ξ is the slack variable, φ is the mapping induced by the kernel, $C > 0$ is the regularization parameter, y_i is the label for x_i, $\ell \geq 0$ is a constant controlling the class imbalance and e is the vector of ones. There, one fixes y and optimizes over ω, b, ξ then fixes ω, b, ξ and optimizes over y, and so on. They suggest using SVR with the Laplacian loss, instead of SVM with the hinge loss, in the inner optimization subproblem, to avoid premature convergence.

4.2 Building Splits in the Target View

So far, we have chosen different splits of the data in the source view. Since we want the features to be similar in different views, we directly transfer the splits in the source view to the target view. For each random projection, we learn a classifier in the target view. We use appearance features in the target view as training data and split values, transferred from the source view, as labels for training. Figure 3 depicts the whole procedure for constructing the split-based features.

4.3 Natural Transfer of Activity Models

At this point we know the splits in the source view and we have learned them in the target view, so we can describe each frame using the split-based features in both views. For shared activities, the split-based features are the same in both views. For a given orphan activity in the target view, we first construct the split-based features. To do this, we project the original features, using the same random projections used in learning each split. Then, by checking which side of each hyperplane each frame lies we can form the split-based features. Split-based features, as depicted in Figure 3, are binary features explaining which side of each split a particular frame lies.

With enough training data, we will have similar split-based features for orphan activities in both views. This means that, we can simply use the model learned on the source view and test it in the target view. Note that the orphan activities have not ever been observed in the target view. See Table 1 for the general framework on building transferable models.

Table 1. General framework of learning transferable models. One can choose his/her choice of descriptive features and classifier and plug them in to this general framework to build transferable models. All one needs is a pool of unlabeled, but corresponding, data in both views and labeled data for orphan objects only in the source domain.

General framework for building transferable models :
Learning:
1. Extract suitable descriptive features for your problem.
2. Select your choice of classifier.
3. Choose splits using the MMC on random projections of descriptive features in the source domain.
4. Transfer the splits to the target domain, using an established correspondences between unlabeled shared objects in both domains.
5. Learn the transferred splits in the target domain
Recognition:
1. Extract descriptive features for a query orphan object.
2. Construct the split-based descriptors in the target view, using the already learned splits in the target view.
3. Recognize the query orphan object using the model that you learned in the source domain.

5 Experimental Setup

To examine the performance of our transferable activity model we need to choose a dataset with multiple views of multiple actions. We picked the IXMAS dataset [42] because there are 5 different views of 11 actions performed 3 times by 10 actors, sequences are time aligned, and silhouette information is also provided. There is a drawback in using the IXMAS dataset to test our method. The actors were not instructed to have a fixed angle to each cameras. This introduces some noise to our model. Fortunately, there is no large aspect change in each view, probably because the actors needed to see the instructor. See Figure 2 for example frames from the IXMAS dataset.

We try all 20 combinations of transfer among views (there are 5 views in IXMAS dataset). We cluster the source view to 40 clusters using k-means clustering on the descriptive features. We describe activities by histograms of codewords. 1000 different 30-dimensional random projections of descriptive features are used. For each random projection we find the MMC split on the cluster centers in the source view. Redundant splits are discarded and the best 25 splits are picked. Splits are scored based on how well they can be predicted from the data(train set accuracies). We learn splits in the target view using SVMs with Gaussian kernels on $1/10$ of data, due to the computational limitations. This doesn't affect the quality of the splits dramatically. For a given orphan action, we predict which side of each split all frames lie. We then construct the codeword-based description of that action by matching the split-based description of each frame in the target view to the closest one in the source view. We then use a 1NN classifier to predict the action label for the orphan action, using the source view. Due to the limited number of actions in the dataset, we follow the leave-one-action-out strategy for choosing an orphan action. This means that we exclude all examples of the selected orphan action, performed by all the actors, in the target view. We averaged over all possible combinations of selecting orphan actions to report the average transfer accuracy.

Table 2. Multi Class Transfer Results: The row labeled FO gives the accuracy with which activities are discriminated when both trained and tested on that camera. Columns give the result of testing on the camera heading the column (in bold, the target), when trained on the camera given in the row (normal, the source). There are two cases: first, simply training on appearance features (WT) and second, using our transfer mechanism (Tr). For example, training on camera 4 and testing on camera 0 gives an accuracy of 12% if one uses appearance features only, and 51% if one uses our method. Notice that our transfer mechanism significantly improves performance; there is a cost for training on the wrong view, but it is not particularly high. Camera 4 is generally difficult. Accuracy numbers are computed in leave-one-orphan-activity-out fashion. This means that for a given transfer scenario we average over all possible selections of an orphan activity.

	Camera 0		**Camera 1**		**Camera 2**		**Camera 3**		**Camera 4**	
FO	76		76		68		73		51	
	WT	Tr	WT	Tr	WT	Tr	WT	Tr	WT	Tr
Camera 0	NA	NA	35	**72**	16	**61**	8	**62**	10	**30**
Camera 1	38	**69**	NA	NA	15	**64**	8	**68**	11	**41**
Camera 2	16	**62**	16	**67**	NA	NA	6	**67**	11	**43**
Camera 3	8	**63**	8	**72**	8	**68**	NA	NA	8	**44**
Camera 4	12	**51**	11	**55**	15	**51**	9	**53**	NA	NA

6 Results

We consider two general experiments. Transfer of binary classifiers (Table 3), and transfer of multi class classifiers (Table 2). We demonstrate the possibilities of transferring activity models from one view to another view. One could reasonably use more sophisticated activity models than what we are using in these experiments. What we care about is the price that we are paying for transfer, given a classifier. Note that (a) we are not observing orphan activities in the target domain; (b) we don't know the labels for shared activities in both views (c) we are classifying activities in the target view using a classifier learned on the source view.

Table 2 and 3 show average accuracies in all 20 possible transfer scenarios. For each case we report three different accuracies: 1) Fully Observed (FO) accuracies, in which we use 1NN classifier and all the actions are observed in that view, 2) Without Transfer (WT) accuracies, in which we don't observe orphan activities in the target view and we try to classify them using 1NN classifier on the source view, and 3) Transfer accuracies (Tr).

Transfer is hard, as the "without transfer" (WT) columns in Table 2 show. For example, without transfer learning we get an accuracy of 8% for classifying activities in camera 3 using classifiers learned in camera 1. Transfer learning classifies the activities with the accuracy of 72%. This is a big gain, considering that *our classifier has not ever seen an example of orphan activities in the target view*. If the classifier fully observes the activities in the target view, we get an accuracy of 76%, comparing to 72% accuracy under transfer. Note that transferring from (or to) camera 4 is not as good as the other 4 cameras, because we can't build discriminative enough split-based descriptors. The fully observed accuracies in camera 4 shows that due to the odd geometrical configuration of camera 4 (See Figure 2 for some pictures from camera 4), some of the actions look the same, in that view (also reported in [41]).

Table 3. One vs All Transfer Results: The row labeled FO gives the accuracy with which activities are discriminated when both trained and tested on that camera. Columns give the result of testing on the camera heading the column (in bold, the target), when trained on the camera given in the row (normal, the source). There are two cases: first, simply training on appearance features (WT) and second, using our transfer mechanism (Tr). For example, training on camera 4 and testing on camera 0 gives an accuracy of 32% if one uses appearance features only and 87% if one uses our method. Notice that our transfer mechanism significantly improves performance; there is a cost for training on the wrong view, but it is not particularly high. Camera 4 is generally difficult. Accuracy numbers are computed in leave-one-orphan-activity-out fashion. This means that for a given transfer scenario we average over all possible selections of an orphan activity.

	Camera 0		**Camera 1**		**Camera 2**		**Camera 3**		**Camera 4**	
FO	95		96		93		93		87	
	WT	Tr	WT	Tr	WT	Tr	WT	Tr	WT	Tr
Camera 0	NA	NA	69	**96**	35	**88**	33	**86**	26	**75**
Camera 1	67	**95**	NA	NA	35	**91**	35	**81**	26	**74**
Camera 2	41	**86**	42	**87**	NA	NA	28	**89**	31	**75**
Camera 3	27	**87**	32	**84**	20	**91**	NA	NA	23	**76**
Camera 4	32	**87**	32	**88**	30	**84**	27	**83**	NA	NA

Table 4. View Estimation Results. If we don't know the target view, we can identify the view discriminatively using the descriptive features and a 1NN classifier. This table shows classification results for view classification problem. If the target view is not specified, then we should multiply accuracies in Table 2 and 3 by accuracies in this table. Since these numbers are reasonably close to 1, we shouldn't expect a major change in accuracies.

Camera 0	Camera 1	Camera 2	Camera 3	Camera 4
98	95	96	95	85

Estimating view: Our discussion has assumed that the view is known. This means that we know the source and the target view, for computing accuracies in Table 2 and 3. But it is not crucial to know the target view. We can identify the view discriminatively using the descriptive features and a 1NN classifier. Table 4 shows classification results for view classification problem. If the target view is not specified, then we should multiply accuracies in Table 2 and 3 by accuracies in Table 4. Since the numbers in Table 4 are reasonably close to 1, we shouldn't expect a major change in accuracies.

To our knowledge, transferring activity models between views is novel, and so we can not compare directly with other methods. The most similar experiment is [41]. In this case, learning is in 3D, using 3D exemplars, and recognition is in 2D. Authors report an average accuracy of 57.8%, compared to our average transfer accuracy of 58.1%, on the same dataset. In our experiments we use the same experimental setting as in [41]. Although we appear to have only a tiny lead in classification accuracy, the comparison is rough because, to train, they observe all the activities in all views and we do not. Similarly, it is worth mentioning that view invariant approaches [42,2,26,23,32] need to observe all of the activities and labels in all views. *Here, we neither observe orphan activities in the target view, nor know the labels for shared activities in either views.*

7 Discussion

We have shown that a discriminative appearance model of activity can be trained on one view and tested on a second, different view, *as long as one uses the right features*. Our split-based representation requires knowing the view, but this can be determined by a simple matching procedure. These split-based features can be seen as view invariant features, or as a representation of the *structure shared between different tasks (recognize from a fixed view), using a pool of unsupervised data*.

One could cluster activity frames in each view, then build correspondence between clusters explicitly. However, this might require quite small clusters, because a large volume of feature space in one view might correspond to a small volume in another. Our discriminative strategy for building features has the advantage that it implicitly follows these changes in the underlying feature metric.

Our procedure is implemented using activity data, but we believe that the underlying problem — use data from one aspect to be able to recognize something seen at another aspect — is pervasive in vision. Our solution appears to be quite general: ***if one has corresponding, but unlabeled, data, one can learn and account for the distortion of the feature space caused by changes of view***.

Acknowledgements

The authors would like to thank David Forsyth for his insightful comments and helpful discussions. This work was supported in part by the National Science Foundation under IIS - 0534837 and in part by the Office of Naval Research under N00014-01-1-0890 as part of the MURI program. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation or the Office of Naval Research.

References

1. Niculescu-Mizil, R.C.A.: Inductive transfer for bayesian network structure learning (2007)
2. Aloimonos, Y., Ogale, A.S., Karapurkar, A.P.: View invariant recognition of actions using grammars. In: Proc. Workshop CAPTECH (2004)
3. Mori, G., Efros, A.A., Berg, A.C., Malik, J.: Recognizing action at a distance. In: IEEE International Conference on Computer Vision (ICCV 2003) (2003)
4. Ando, R.K., Zhang, T.: A framework for learning predictive structures from multiple tasks and unlabeled data. J. Mach. Learn. Res. 6, 1817–1853 (2005)
5. Bakker, T.H.B.: Task clustering and gating for bayesian multitask learning. Journal of Machine Learning, 83–99 (2003)
6. Barron, C., Kakadiaris, I.: Estimating anthropometry and pose from a single uncalibrated image. Computer Vision and Image Understanding 81(3), 269–284 (2001)
7. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. In: ICCV, pp. 1395–1402 (2005)
8. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. In: ICCV (2005)

9. Bobick, A., Davis, J.: The recognition of human movement using temporal templates. PAMI 23(3), 257–267 (2001)
10. Bregler, C., Malik, J.: Tracking people with twists and exponential maps. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 8–15 (1998)
11. Perkins, G.S.D.N.: Transfer of learning, 2nd edn. International Encyclopedia of Education (1992)
12. Dai, W., Yang, Q., Xue, G.-R., Yu, Y.: Boosting for transfer learning. In: ICML 2007 (2007)
13. Dance, C., Willamowski, J., Fan, L., Bray, C., Csurka, G.: Visual categorization with bags of keypoints. In: ECCV International Workshop on Statistical Learning in Computer Vision (2004)
14. Elidan, G., Heitz, G., Koller, D.: Learning object shape: From drawings to images. In: CVPR 2006, Washington, DC, USA, pp. 2064–2071. IEEE Computer Society, Los Alamitos (2006)
15. Evgeniou, T., Pontil, M.: Regularized multi–task learning. In: KDD 2004 (2004)
16. Farhadi, A., Forsyth, D.A., White, R.: Transfer learning in sign language. In: CVPR (2007)
17. Feng, X., Perona, P.: Human action recognition by sequence of movelet codewords. In: Proceedings of First International Symposium on 3D Data Processing Visualization and Transmission,2002, pp. 717–721 (2002)
18. Forsyth, D., Arikan, O., Ikemoto, L., O'Brien, J., Ramanan, D.: Computational aspects of human motion i: tracking and animation. Foundations and Trends in Computer Graphics and Vision 1(2/3), 1–255 (2006)
19. Howe, N.R., Leventon, M.E., Freeman, W.T.: Bayesian reconstruction of 3d human motion from single-camera video. In: Solla, S., Leen, T., Müller, K.-R. (eds.) Advances in Neural Information Processing Systems 12, pp. 820–826. MIT Press, Cambridge (2000)
20. Hu, W., Tan, T., Wang, L., Maybank, S.: A survey on visual surveillance of object motion and behaviors. IEEE Trans. Systems, Man and Cybernetics - Part C: Applications and Reviews 34(3), 334–352 (2004)
21. Ikizler, N., Forsyth, D.: Searching video for complex activities with finite state models. In: CVPR (2007)
22. Kaski1, S., Peltonen, J.: Learning from Relevant Tasks Only. Springer, Heidelberg (2007)
23. Laptev, I., Lindeberg, T.: Space-time interest points (2003)
24. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stero vision. IJCAI (1981)
25. Rosenstein, L.K.M.T., Marx, Z.: To transfer or not to transfer (2005)
26. Niu, F., Abdel-Mottaleb, M.: View-invariant human activity recognition based on shape and motion features. In: ISMSE 2004 (2004)
27. Niyogi, S., Adelson, E.: Analyzing and recognizing walking figures in xyt. In: Media lab vision and modelling tr-223. MIT, Cambridge (1995)
28. Scovanner, M.S.P., Ali, S.: A 3-dimensional sift descriptor and its application to action recognition. ACM Multimedia (2007)
29. Parameswaran, V., Chellappa, R.: View invariants for human action recognition. In: IEEE Conf. on Computer Vision and Pattern Recognition (2003)
30. Raina, D.K.R., Ng, A.Y.: Transfer learning by constructing informative priors (2005)
31. Ramanan, D., Forsyth, D.: Automatic annotation of everyday movements. In: Advances in Neural Information Processing (2003)
32. Rao, C., Yilmaz, A., Shah, M.: View-invariant representation and recognition of actions. IJCV 50(2), 203–226 (2002)
33. Taylor, C.: Reconstruction of articulated objects from point correspondences in a single uncalibrated image. Computer Vision and Image Understanding 80(3), 349–363 (2000)
34. Taylor, M.E., Stone, P.: Cross-domain transfer for reinforcement learning. In: ICML 2007 (2007)

35. Thrun, S.: Is learning the n-th thing any easier than learning the first? In: NIPS (1996)
36. Tran, D., Sorokin, A.: Human activity recognition with metric learning. In: ECCV (2008)
37. Turaga, P.K., Veeraraghavan, A., Chellappa, R.: From videos to verbs: Mining videos for activities using a cascade of dynamical systems. In: IEEE Conf. on Computer Vision and Pattern Recognition (2007)
38. Wang, L., Suter, D.: Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model. In: IEEE Conf. on Computer Vision and Pattern Recognition (2007)
39. Wang, L., Suter, D.: Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model. CVPR (2007)
40. Wang, Y., Huang, K., Tan, T.: Human activity recognition based on r transform. Visual Surveillance (2007)
41. Weinland, D., Boyer, E., Ronfard, R.: Action recognition from arbitrary views using 3d exemplars. In: ICCV, Rio de Janeiro, Brazil (2007)
42. Weinland, D., Ronfard, R., Boyer, E.: Free viewpoint action recognition using motion history volumes. Computer Vision and Image Understanding (2006)
43. Wilson, A., Bobick, A.: Learning visual behavior for gesture analysis. In: IEEE Symposium on Computer Vision, pp. 229–234 (1995)
44. Wilson, A., Fern, A., Ray, S., Tadepalli, P.: Multi-task reinforcement learning: a hierarchical bayesian approach. In: ICML 2007 (2007)
45. Yamato, J., Ohya, J., Ishii, K.: Recognising human action in time sequential images using hidden markov model. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 379–385 (1992)
46. Yang, J., Xu, Y., Chen, C.S.: Human action learning via hidden markov model. IEEE Transactions on Systems Man and Cybernetics 27, 34–44 (1997)
47. Yilmaz, A., Shah, M.: Actions sketch: A novel action representation (2005)
48. Marx, L.K.Z., Rosenstein, M.T.: Transfer learning with an ensemble of background tasks (2005)
49. Zhang, K., Tsang, I.W., Kwok, J.T.: Maximum margin clustering made practical. In: ICML 2007 (2007)