Distributed Activity Recognition in Camera Networks via Low-Rank Matrix Recovery

Ehsan Adeli Mosabbeb
Iran Univ. of Science and Technology
Email: eadeli@iust.ac.ir

Kaamran Raahemifar
Ryerson University, Canada
Email: kraahemi@ee.ryerson.ca

Mahmood Fathy
Iran Univ. of Science and Technology
Email: mahfathy@iust.ac.ir

***Abstract*—Multi-view action recognition addresses many challenges like view-invariance and occlusion. But the huge amount of data makes it hard for use in real life applications. In this paper, we propose a distributed activity classification framework based on consensus matrix completion, where several smart cameras are observing the scene, each process their observations and come to an agreement about the activity class, through communication. We validate our approach using IXMAS dataset.**

I. Introduction

Action recognition has many applications, including vision based surveillance, human-computer interaction, patient monitoring systems [1]. With the development of the smart camera technologies, the huge amount of processing for such high level applications could be performed in a more robust and scalable way. Several previous works tackle developing many computer vision applications in such environments [2], [3], [4].

In this paper, we develop a method for the recognition of human activities portrayed in multi-view video sequences. Our method is based on low-rank matrix recovery. Rank Minimization has recently gained a lot of attention, due to the success in solving many problems, and dealing with noise and outliers [5]. Here, Each scene is represented with fixed length histograms of densely sampled features, which capture both the visual content and the temporal changes in the scene.

II. Matrix Completion for Classification

Assume that we want to recover a data matrix $\mathbf{D}$ from a matrix $\mathbf{D_0}$, in which we only observe a number of its entries. With sufficiently large and uniformly distributed measurements, we can assume that there is one low-rank matrix with these entries [5]. We know that a matrix of rank r has exactly r nonzero singular values. Thus, a simple estimate can be defined as $\|\mathbf{D}\|_* = \sum_{k=1}^{d} \sigma_k(\mathbf{D})$, which is called the nuclear norm. For a classification task, the goal is to learn a mapping from the space of features X to the space of labels Y, from N_{tr} training instances with m the number of different classes, n the dimensionality of the feature space and N the number of total instances. Concatenatng all labels/features into a single matrix, if a linear classification model holds, this matrix should be rank deficient:

$$\mathbf{D} = \begin{bmatrix} \mathbf{D_Y} \\ \mathbf{D_X} \\ \mathbf{D_1} \end{bmatrix} = \begin{bmatrix} \mathbf{Y_{tr}} & \mathbf{Y_{tst}} \\ \mathbf{X_{tr}} & \mathbf{X_{tst}} \\ \mathbf{1}^\top & \end{bmatrix} + \begin{bmatrix} \mathbf{E_{Y_{tr}}} & \mathbf{0} \\ \mathbf{E_{X_{tr}}} & \mathbf{E_{X_{tst}}} \\ \mathbf{0}^\top & \end{bmatrix} \quad (1)$$

where $\mathbf{Y_{tr}} \in \mathbb{R}^{m \times N_{tr}}$, $\mathbf{Y_{tst}} \in \mathbb{R}^{m \times N_{tst}}$, $\mathbf{X_{tr}} \in \mathbb{R}^{n \times N_{tr}}$ and $\mathbf{X_{tst}} \in \mathbb{R}^{n \times N_{tst}}$ are the training/testing labels, and the training/testing feature vectors, respectively. Therefore, the classification process would be to find the best $\mathbf{Y_{tst}}$ and $\mathbf{E}$ such that the rank of $\mathbf{D} = \mathbf{D_0} + \mathbf{E}$ is minimized [6], [7]:

$$\begin{aligned} \min_{\mathbf{D}} \quad & \mu\|\mathbf{D}\|_* + \\ & \frac{1}{|\Omega_X|} \sum_{ij \in \Omega_X} c_x(\mathbf{E_{X_{ij}}}) + \frac{\lambda_1}{|\Omega_Y|} \sum_{ij \in \Omega_Y} c_y(\mathbf{E_{Y_{ij}}}) \quad (2) \\ \text{s. t.} \quad & \mathbf{D} = \mathbf{D_0} + \mathbf{E},\ \mathbf{D_1} = \mathbf{1}^\top \end{aligned}$$

where Ω_X and Ω_Y are the set of known entries in $\mathbf{D_0}$, and μ and λ_1 are positive trade-off weights. $c_y(.)$ is a log loss function and $c_x(.)$ is a least squares error to avoid trivial solutions and to penalize large distortions of $\mathbf{D}$ [6], [7].

III. Distributed Activity Recognition

Let's assume that the network of the cameras is modeled with a connected undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V} = \{1, \ldots, N_c\}$ as the set of camera nodes and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ the nodes that can communicate with each other. To represent each video, for the activity recognition problem, we use Histogram of gradient (HoG) and histogram of optical flow (HoF) [8], with independent dictionaries. In order to make sure that the orientation of the activities with regard to the cameras does not strengthen noise, we employ a cycling approach as in [3].

As shown by [9] as long as the error matrix $\mathbf{E}$ is sufficiently sparse, we can recover the low-rank matrix $\mathbf{D}$ from $\mathbf{D_0} = \mathbf{D} + \mathbf{E}$ by solving (2). We can solve this problem using augmented Lagrangian method. According to the singular value thresholding (SVT) algorithm [10], problem (2) could be solved by updating each variable while keeping the others fixed, using a shrinkage operator, $\mathcal{S}_\epsilon[x]$. and with the the singular value decomposition of a matrix, $\mathbf{USV}^\top$, the augmented Lagrangian method is utilized [9], which applies the shrinkage operator to the singular values of the matrix $\mathbf{D_0} - \mathbf{E} + \mu_k^{-1}\mathscr{L}_\mathbf{k}$, in each iteration. In order to parallelize this algorithm, we need to distribute the $\mathbf{D_0}$ entries between the processing nodes. Suppose that we split the data matrix $\mathbf{D}$ into N_c parts, $\mathbf{D_i}$. We can assume that the original data matrix is formed as $\mathbf{D} = [\mathbf{D_1}^\top, \mathbf{D_2}^\top, \ldots, \mathbf{D_{N_c}}^\top]^\top \in \mathbb{R}^{(n+m) \times (N_{tr}+N_{tst})}$. The Lagrangian multipliers, $\mathscr{L}$, and the error matrix, $\mathbf{E}$, would also be split in a same manner. Now, we need to calculate the SVD of $\mathbf{J} = \mathbf{D_0} - \mathbf{E} + \mu_k^{-1}\mathscr{L}_\mathbf{k}$ matrix. First, suppose we want to compute $\mathbf{C} = \frac{1}{N_c}\mathbf{J}^\top\mathbf{J} = \frac{1}{N_c}\sum_{i=1}^{N_c} \mathbf{J_i}^\top\mathbf{J_i} = \frac{1}{N_c}\sum_{i=1}^{N_c} \mathbf{C_i}$. $\mathbf{C_i} = \mathbf{J_i}^\top\mathbf{J_i}$ could be denoted as the local correlation matrix. This is very easy to compute through consensus, since it is a simple averaging of data present in each node. Initially, each

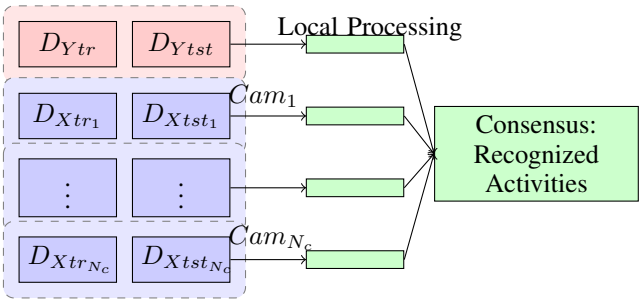

Fig. 1. A model for the data split between the smart cameras.

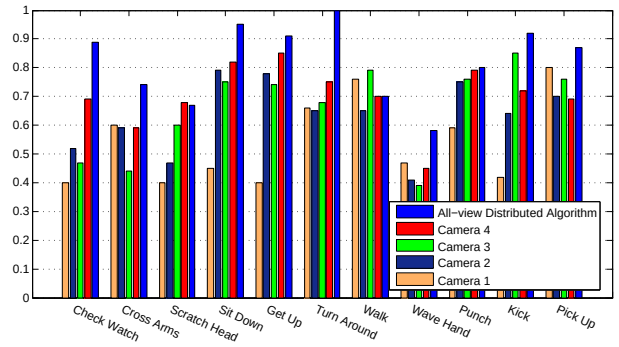

Fig. 2. Recognition results on IXMAS dataset.

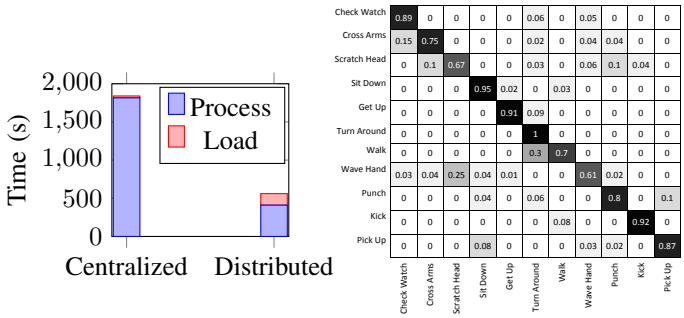

Fig. 3. Execution time comparisons and the confusion matrix on IXMAS.

node has a local state $\mathbf{c_i}(0) = \mathbf{C_i}$, in each iteration each node receives the internal state of its neighbors and updates its own by $\mathbf{c_i}(t+1) = \mathbf{c_i}(t) + (max_i\{d_i\})^{-1} \sum_{j \in \mathcal{N}_i} (\mathbf{c_j}(t) - \mathbf{c_i}(t))$. It is shown [2] that each state converges to the average of the initial values ($\lim_{t\to\infty} \mathbf{c_i} = \mathbf{C}$), regardless of the network configuration and the partial noise in the communications.

In order to compute SVD of the matrix $\mathbf{J}$, we need to calculate matrices $\mathbf{U} \in \mathbb{R}^{(n+m)\times r}$, $\mathbf{V} \in \mathbb{R}^{r\times(N_{tr}+N_{tst})}$ and $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$, with r as the rank of the matrix: $\mathbf{J} = \mathbf{U\Sigma V}^\top$. To do this, we can compute the SVD of $\mathbf{C} = \mathbf{V}(\frac{1}{N_c}\mathbf{\Sigma}^2)\mathbf{V}^\top$. After the distributed averaging each node can recover $\mathbf{V}$ and if they know N_c, they also can recover $\mathbf{\Sigma}$. These two matrices will be in common for all the nodes, and they can compute their own share of the matrix $\mathbf{U}$ as $\mathbf{U_i} = \mathbf{J_i V\Sigma}^{-1}$. As a result, the SVD operation could be calculated in a distributed fashion and each node can recover the complete matrix $\mathbf{\Sigma}$ to apply the shrinkage operator on and optimize for the data matrix rank.

We can model the distribution of the data matrix as shown in figure 1. The data matrix is split between the processing nodes, row-wise. The labels (upper row in figure 1) are also assigned to a single node. To solve the convex problem (2), let us for simplicity, replace the second and the third terms in the objective function with $f(\mathbf{E_{X_i}})$ and $g(\mathbf{E_{Y_i}})$, respectively. Now, we will have an equivalent problem, for each single processing node i. The only shared process is the minimization of the nuclear norm of the whole data matrix, as described above. This problem could be solved using the Alternating Direction Method (ADM), with the Lagrangian function: $\gamma\|\mathbf{D}\|_* + f(\mathbf{E_{X_i}}) + g(\mathbf{E_{Y_i}}) + \langle \mathscr{L}_\mathbf{i}, \mathbf{D_i} - \mathbf{D_{0_i}} - \mathbf{E_i} \rangle + \frac{\mu}{2}\|\mathbf{D_i} - \mathbf{D_{0_i}} - \mathbf{E_i}\|_F^2$,

IV. Experiments

We carried out experiments using the IXMAS dataset [11]. In order to be consistent with previous works [11], [3] we discard images from camera 5, and we use 10 subjects and 11 actions. We have simulated the network environment with a full topology. Figure 2 shows the classification results on individual cameras, compared with the distributed algorithm. Figure 3 outlines the confusion matrix of the distributed activity recognition, and the execution times of our distributed matrix completion compared to the centralized version. Table I also shows the overall recognition rate in comparisons with some state-of-the-art methods.

V. Conclusion

In this paper, we have described a distributed action recognition algorithm, based on low-rank matrix completion. We have proposed a simple distributed algorithm to minimize the nuclear norm of a matrix, and adapted a distributed optimization for the matrix completion problem. We have tested the algorithm on IXMAS dataset.

References

[1] J. Aggarwal and M. Ryoo, "Human activity analysis: A review," *ACM Comput. Surv.*, vol. 43, no. 3, pp. 16:1–16:43, Apr. 2011.

[2] R. Tron and R. Vidal, "Distributed computer vision algorithms through distributed averaging," in *CVPR*, 2011, pp. 57–63.

[3] G. Srivastava, H. Iwa., J. Park, and A. Kak, "Distributed and lightweight multi-camera human activity classification," in *ICDSC*, 2009.

[4] S. Ramagiri, R. Kavi, and V. Kul., "Real-time multi-view human action recognition using a wireless camera network," in *ICDSC*, 2011.

[5] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Found. Comput. Math.*, vol. 9, no. 6, pp. 717–772, 2009.

[6] A. B. Goldberg, X. Zhu, B. Recht, J. Xu, and R. Nowak, "Transduction with matrix completion: Three birds with one stone," in *NIPS*, 2010.

[7] R. S. Cabral, F. De la Torre, J. P. Costeira, and A. Bernardino, "Matrix completion for multi-label image classification," in *NIPS*, 2011.

[8] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *CVPR*, 2008.

[9] Z. Lin, M. Chen, L. Wu, and Y. Ma, "The ALM method for exact recovery of corrupted low-rank matrices," *UILU-ENG-09-2215*, 2009.

[10] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM J. on Opt.*, vol. 20(4), 2010.

[11] D. Weinland, E. Boyer, and R. Ronfard, "Action Recognition from Arbitrary Views using 3D Exemplars," in *ICCV*, 2007, pp. 1–7.

Approach	Method	Acc
Srivastava et al. [3]	Distributed	81.4%
Weinland et al. [11]	Multi-view	81.3%
Our Method	Distributed	82.1%

TABLE I. Accuracy results on IXMAS, using 4 cameras.