A Least Squares Regression Framework on Manifolds and its Application to Gesture Recognition

Yui Man Lui
Department of Computer Science, Colorado State University
Fort Collins, CO 80521, USA
lui@cs.colostate.edu

Abstract

Least squares regression is a basic approach for statistical analysis. However, its simplicity has often led to researchers overlooking it for complex recognition problems. In this paper, we present a nonlinear regression framework on manifolds for gesture recognition. Our method is developed based upon two key attributes: underlying geometry and least squares fitting. The former attribute is vital since geometry characterizes the space for classification while the latter exhibits a simple estimation model. Considering geometric properties, we formulate least squares regression as a composite function. This gives a natural extension from Euclidean space to manifolds. Our experiments show that the proposed framework achieves state-of-the-art results on the standard hand gesture and body gesture datasets. Our method also generalizes well on the one-shot-learning CHALEARN gesture challenge.

1. Introduction

Human gestures are useful media for communication. We use gestures to depict sign language to deaf people, convey messages in noisy environments, and interface with computer games. Having human-machine gesture-based communication enriches our daily lives and makes mundane work effective. In recent years, many gesture recognition algorithms have been proposed [20, 4]. Despite these efforts, gesture recognition remains a challenging area due in part to the complexity of human movements. To champion the recognition performance, models are often complicated, causing difficulty for generalization. Consequently, heavy-duty models may not have substantial gains in overall gesture recognition problems.

Linear regression is one of the fundamental techniques in statistical analysis. It is simple and often outperforms complicated models when the number of training samples is small [9]. In this paper, we revisit least squares regression and emphasize the important aspect of underlying geometry [17]. Particularly for gesture recognition, patterns are collected from videos and intrinsically reside in non-Euclidean space. Our work is built upon the product manifold representation [18] by which an action is characterized as a point on a product manifold.

While product manifold representation has shown promising results, the classification is solely based on nearest neighbors. Thus, it has high variance and may be unstable in some cases. To balance the weight between variance and bias, we employ least squares regression as a statistical model yielding a smooth decision boundary [9]. The key element of our regression framework is that it accounts for underlying geometry on manifolds. In doing so, we formulate least squares regression as a composite function. As such, we ensure that both the domain values and the range values reside on a manifold through the regression process. The least squares fitted elements from a training set can then be exploited for pattern recognition, particularly for gesture recognition.

We demonstrate the proficiency of our framework on three gesture recognition problems including hand gestures, body gestures, and gestures collected from Kinect for the one-shot-learning CHALEARN gesture challenge. Our experimental results reveal that our method is competitive to the state-of-the-art methods, yet based on a simple statistical model.

The key contributions of the proposed work are 1) introducing a novel formulation for least squares regression on manifolds; 2) relating the proposed framework to gesture recognition; 3) achieving competitive performance; 4) using a simple pixel-based representation (no silhouette and no skeleton extraction); 5) no explicit assumption on data.

The remainder of this paper is organized as follows:

978-1-4673-1612-5/12/$31.00 ©2012 IEEE

Related work is reviewed in Section 2. The framework on regression in non-Euclidean space is introduced in Section 3. Experimental results are reported in Section 4. Finally, we conclude this paper in Section 5.

2. Related Work

Regression techniques have been widely used in statistical analysis as well as computer vision. Here, we review some related work.

Human detection is a crucial element for human action recognition. Schwartz *et al*. [22] combined HOG, color frequency, and co-occurrence matrices as a feature and employed Partial Least Squares (PLS) fitting for dimension reduction. As a consequence, low dimensional features were classified using quadratic discriminant analysis. Unlike PCA, PLS takes class labels into account while providing dimension reduction.

Group activities can be effectively identified from trajectories. Cheng *et al*. [6] applied Gaussian process regression and motion analysis for activity recognition. The fitted trajectories and motion patterns were characterized by a bag-of-feature model. The features were then grouped using multi-kernel learning followed by a support vector machine.

Liu *et al*. [16] proposed the use of ordinal regression via manifold learning. The ordinal regression learns the rating of data by maximizing the margin between two consecutive ranks. As such, the order information from a neighborhood graph is optimized. This framework is extended to a multilinear representation and applied to face and digit classification using a k-NN classifier.

Pham and Venkatesh [21] studied the use of multivariate lasso regression on Stiefel manifolds for face recognition. This method employed dual projections for dimension reduction and data fitting. Because of the orthogonality constraint of the projection matrix, the steepest descent method was applied to find the optimal projection on a Stiefel manifold.

Recently, Meyer *et al*. [19] have addressed a regression model under fixed-rank constraints. Since fixed-rank matrices do not reside on Euclidean space, quotient geometry is considered. Matrices were first factorized using balanced and polar factorizations. Line-search algorithms were then employed to seek the optimal projection matrices. This work was applied to collaborative filtering.

3. Regression in Non-Euclidean Spaces

Linear regression is one of the fundamental techniques in data analysis. The aim of this paper is to formulate a least squares regression model on manifolds.

3.1. Linear and Nonlinear Least Squares Regressions

Perhaps the most straightforward data prediction technique is least squares regression. In this paper, we demonstrate this by considering the underlying geometry and show its applicability on gesture recognition. Before we discuss the geometric extension, we will first review the standard form of least squares fitting.

Consider a regression problem $y = A\beta$ where $y \in \mathbb{R}^n$ is the regression value, $A([a_1|a_2|\cdots|a_k]) \in \mathbb{R}^{n\times k}$ is the training set, and $\beta \in \mathbb{R}^k$ is the fitting parameter. The residual sum-of-squares can be written as

$$R(\beta) = \| y - A\beta \|^2 \tag{1}$$

and the fitting parameter β can be obtained by minimizing the residual sum-of-squares error from Eq. (1). Then, we have

$$\hat{\beta} = (A^T A)^{-1} A^T y. \tag{2}$$

The fitted pattern from the training set has the following form

$$\hat{y} = A\hat{\beta} = A(A^T A)^{-1} A^T y. \tag{3}$$

We can further extend the linear least squares regression from Eq. (3) to a nonlinear form by incorporating a kernel function shown in the following

$$A(A \star A)^{-1}(A \star y) \tag{4}$$

where $\star$ is a nonlinear similarity operator. Obviously, $\star$ is equal to $x^T y$ in the linear case. In this paper, we employ the RBF kernel given as

$$x \star y \quad = \quad \exp(-\frac{\| \cdot \|}{\sigma}) \tag{5}$$

where $\| \cdot \|$ is a distance measure associated with a particular manifold discussed in the following subsection.

3.2. Nonlinear Least Squares Regression on Manifolds

The key contribution of this work is the characterization of least squares fitting on manifolds. In traditional applications, the patterns A from Eq. (4) generally represents a data matrix from a training set, i.e. a set of training instances. In gesture recognition, video data consist of spatio-temporal information usually represented by a high dimensional data structure. Particularly, in this work, we characterize a video as a third order tensor and a collection of videos as a set of third order tensors.

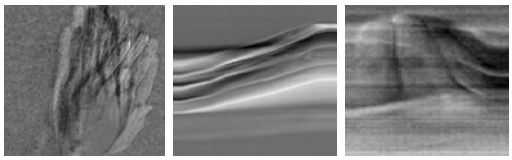

Figure 1. Factorized elements of HOSVD from a gesture video. (Left: appearance, Middle: horizontal motion, Right: vertical motion)

Data tensors are multi-dimensional data objects equipped with mathematical tools making multiple factor analysis possible. Considering a video represented as a third order tensor $\mathcal{A}$, we can apply HOSVD [13] to factorize the data tensor as

$$\mathcal{A} = \mathcal{S} \times_1 V^{(1)}_{\text{appearance}} \times_2 V^{(2)}_{\text{h-motion}} \times_3 V^{(3)}_{\text{v-motion}} \quad (6)$$

where $\times_k$ denotes the mode-k multiplication, $\mathcal{S}$ is the core tensor, and $V^{(k)}$ is the orthogonal matrix spanning the row space of the unfolded matrix $A_{(k)}$ associated with nonzero singular values. An example of the factorized elements from a hand gesture video is given in Figure 1.

Using HOSVD, the nonlinear regression given in Eq. (4) can operate on unfolded matrices; thus the regression model becomes three separate estimations as

$$\psi^{(k)}(y) = A^{(k)}(A^{(k)} \star A^{(k)})^{-1}(A^{(k)} \star y^{(k)}) \quad (7)$$

where k denotes the mode of unfolding, $A^{(k)}$ is a set of orthogonal matrices factorized from HOSVD, and $y^{(k)}$ is an orthogonal matrix from the unfolded matrix.

We now take a closer look at what the computation of a similarity map gives us from Eq. (7). Given p training videos, $(A^{(k)} \star A^{(k)})^{-1}$ produces a $p \times p$ matrix from the training set and $(A^{(k)} \star y^{(k)})$ would create a $p \times 1$ vector. Therefore, this similarity map provides a weighting vector characterizing the training data on a factor manifold as

$$\text{w} = (A^{(k)} \star A^{(k)})^{-1}(A^{(k)} \star y^{(k)}) \quad (8)$$

where the weighting vector is in a vector space, i,e, w $\in \mathcal{V}$.

To incorporate the weighting vector with the least squares fitting given in Eq. (7) , we make a simple modification and reformulate the regression as follows

$$\Psi^{(k)}(y) = A^{(k)} \bullet (A^{(k)} \star A^{(k)})^{-1}(A^{(k)} \star y^{(k)}) \quad (9)$$

where $\bullet$ is an operator mapping points from a vector space back to a factor manifold. By introducing an additional operator, we ensure that both the domain values $y^{(k)}$ and the range values $\Psi^k(y)$ reside on a manifold. From a function composition point of view, the proposed regression technique can be viewed as a composition map $\mathcal{G} \circ \mathcal{H}$ where $\mathcal{H} : \mathcal{M} \longrightarrow \mathcal{V}$ and $\mathcal{G} : \mathcal{V} \longrightarrow \mathcal{M}$ where $\mathcal{M}$ is a manifold and $\mathcal{V}$ is a vector space.

One possible way to accomplish the mapping $\mathcal{G} : \mathcal{V} \longrightarrow \mathcal{M}$ is to modify the Karcher mean [10]. The computation of Karcher mean considers the intrinsic geometry and iteratively minimizes the distance between the updated mean and all data samples. Since w is the weighting vector, it naturally produces the weight between training samples. All we need is to use the weighting vector to weight the training samples on a factor manifold. This is equivalent to computing the weighted Karcher mean, which is an element of a manifold.

Until now, the formulation of our least squares regression is very general. To make it specific for gesture recognition, we impose rotation invariance to the factorized element $V^{(k)}$ such that they are elements on a Grassmann manifold and the computation of the weighted Karcher mean can be realized. Here, we sketch the pseudo-code in Algorithm 1. As Algorithm 1 illustrates, the first step is to initialize a base point on a manifold. To do so, we compute the weighted average from the training samples in Euclidean space and project it back to the Grassmann manifold using

Algorithm 1 Weighted Karcher Mean Computation

1: Initialize a base point on a manifold
2: **while** not converged **do**
3: Apply the logarithmic map to the training samples to the base point
4: Compute the weighted average on the tangent space at the base point
5: Update the base point by applying the exponential map on the weighted average
6: **end while**

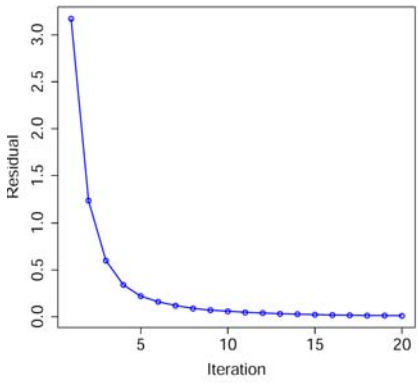

Figure 2. The residual values of tangent vectors.

QR factorization. Then, we iteratively update the base point on the Grassmann manifold. The update procedure involves the standard logarithmic map and the exponential map on Grassmann manifolds [7] described as follows

$$\log_{\mu}(Y_i) \quad = \quad U_1 \Theta_1 V_1^T \tag{10}$$

where μ is the base point, Y_i is a training instance factorized on the Grassmann manifold, $\mu_\perp \mu_\perp^T Y_i (\mu^T Y_i)^{-1} = U_1 \Sigma_1 V_1^T$, $\Theta_1 = \arctan(\Sigma_1)$, and $\mu_\perp$ is the orthogonal complement to μ.

$$\exp_{\mu}(\Delta) \quad = \quad \mu V_2 \cos(\Sigma_2) + U_2 \sin(\Sigma_2) \tag{11}$$

where Δ is the weighted tangent vector at μ and $\Delta = U_2 \Sigma_2 V_2^T$. In addition, the weighted Karcher mean exhibits fast convergence [3]. A sample run is depicted in Figure 2. Typically, convergence can be reached within 10 iterations.

Now, let us get back to the construction of a kernel function. Since the factor manifold $V^{(k)}$ is Grassmannian, the similarity map should also relate to the geodesic distance on Grassmann manifolds. As such, the RBF kernel function is defined as

$$x \star y \quad = \quad \exp(-\frac{\sum_k \theta_k}{\sigma}) \tag{12}$$

where x and y are the elements on the factor manifold, θ_k is the canonical angle [5], and σ is set to 2 in all our experiments.

To perform gesture recognition, a set of training videos is collected. All videos are normalized to a standard size. During the test phase, the category of a query video is determined by

$$j^* = \underset{j}{\operatorname{argmin}} \mathcal{D}(Y, \Psi_j(Y)) \tag{13}$$

where Y is a query video, Ψ_j is the regression instance for the class j, and $\mathcal{D}$ is a geodesic distance measure. Because the query gesture Y and the regression instance are realized as elements on a product manifold [18], we employ the chordal distance for classification.

In summary, the least squares regression model applies HOSVD on a query gesture Y and factorizes it to three sub-regression models ($\Psi_j^{(1)}$, $\Psi_j^{(2)}$, $\Psi_j^{(3)}$) on three Grassmann manifolds where regressions are performed. The distance between the regression output and query is then characterized on a product manifold; gesture recognition is achieved using the chordal distance.

4. Experiments

This section summarizes our empirical results and demonstrates the proficiency of our framework on gesture recognition. We evaluate our method using two publicly available gesture datasets namely Cambridge Hand-Gesture [12] and UMD Keck Body-Gesture [15]. We further extend our method to the one-shot-learning CHALEARN gesture challenge [1]. Our experiments reveal that not only does our method achieve state-of-the-art performance on the benchmark datasets, but also it generalizes well on the one-shot-learning.

Figure 3. Hand gesture samples. Flat-Leftward, Flat-Rightward, Flat-Contract, Spread-Leftward, Spread-Rightward, Spread-Contract, V-Shape-Leftward, V-Shape-Rightward, and V-Shape-Contract.

Figure 4. Body gesture samples. First row: Turn Left, Turn Right, Attention Left, Attention Right, Attention Both, Stop Left, and Stop Right. Second row: Stop Both, Flap, Start, Go Back, Close Distance, Speed Up, and Come Near.

4.1. Cambridge Hand-Gesture Dataset

Our first experiment is conducted using the Cambridge Hand-Gesture dataset which has 900 video sequences with nine different hand gestures (100 video sequences per gesture class). The gesture data are further divided into five different illumination sets labeled as Set1, Set2, Set3, Set4, and Set5. Example gestures are shown in Figure 3.

We follow the experimental protocol in [12] where Set5 is the target set, and Set1, Set2, Set3, and Set4 are the query sets. The target Set5 is further partitioned into a training set and validation set (90 video sequences in the training set and 90 video sequences in the validation set). We employ five random trials in selecting the training and validation videos in Set5. The recognition results are summarized in Table 1 where the classification rates are the average accuracy obtained from five trial runs followed by the standard deviation. As Table 1 shows, our method performs very well across four different illumination sets. In particular, the least squares regression further improves the accuracy of the product manifold representation.

4.2. UMD Keck Body-Gesture Dataset

The UMD Keck body gesture dataset consists of 14 naval body gestures acquired from both static and dynamic backgrounds. In the static background, the subjects remain stationary whereas the subjects are

Table 1. Classification results on the Cambridge Hand-Gesture dataset (Five trial runs).

Method	Set1	Set2	Set3	Set4	Total
Graph Embedding [23]	-	-	-	-	82%
TCCA [12]	81%	81%	78%	86%	82±3.5%
DCCA+SIFT [11]	-	-	-	-	85±2.8%
RLPP [8]	86%	86%	85%	88%	86.3±1.3%
Product Manifold [18]	89%	86%	89%	87%	88±2.1%
Our Method	93%	89%	91%	94%	91.7±2.3%

Table 2. Classification results on the UMD Body-Gesture dataset (Static background and Dynamic background).

Method	Static-Bg	Dynamic-Bg
HOG3D [4]	-	53.6%
Shape Manifold [2]	82%	-
Prototype-Tree [15]	95.2%	91.1%
Product Manifold [18]	92.9%	92.3%
Our Method	94.4%	92.3%

moving in the dynamic environment during the performance of the gesture. There are 126 videos collected from the static background and 168 videos taken from the dynamic background. Example gestures are given in Figure 4.

We follow the experimental protocol in [15] for both static and dynamic settings. In the static background, the protocol is leave-one-subject-out (LOSO) cross-validation. The gestures acquired from the static scene are used for training while the gestures collected from the dynamic background are employed as test videos. The recognition results for both static and dynamic backgrounds are reported in Table 2. We can see that our method is competitive to the current state-of-the-art methods. Unlike the prototype-tree approach depending on the extraction of silhouette images, our method works directly on the raw pixels making our method more generic.

4.3. One-Shot-Learning Gesture Challenge

Microsoft Kinect has recently revolutionized gesture recognition by providing both RGB and depth images. To facilitate the adaptation to new gestures, Kaggle is organizing a one-shot-learning challenge for gesture recognition.

The key aspect of one-shot-learning is to perform training with a single training example. We extend our least squares framework by synthesizing training examples from the original training instance. Consequently, we are able to apply the same algorithm for the one-shot-learning CHALEARN gesture challenge.

Figure 5. Gesture samples on the one-shot-learning gesture challenge (devel03, devel10, and devel19).

Table 3. Results on the development data for the one-shot-learning challenge where TeLev is the sum of the Levenshtein distance divided by the true number of gestures and TeLen is the average error made on the number of gestures.

	Baseline		Our Method	
Batch	TeLev%	TeLen%	TeLev%	TeLen%
devel01	53.33	12.22	13.33	4.44
devel02	68.89	16.67	35.56	14.44
devel03	77.17	5.43	71.74	20.65
devel04	52.22	30.00	10.00	2.22
devel05	43.48	10.87	9.78	7.61
devel06	66.67	17.78	37.78	14.44
devel07	81.32	19.78	18.68	3.30
devel08	58.43	12.36	8.99	5.62
devel09	38.46	9.89	13.19	1.10
devel10	75.82	21.98	50.55	1.10
devel11	67.39	18.48	35.87	2.17
devel12	52.81	5.62	22.47	4.49
devel13	50.00	17.05	9.09	2.27
devel14	73.91	22.83	28.26	3.26
devel15	50.00	8.70	21.74	0.00
devel16	57.47	17.24	31.03	6.90
devel17	66.30	32.61	30.43	4.35
devel18	70.00	28.89	40.00	11.11
devel19	71.43	15.38	49.45	3.30
devel20	70.33	36.26	35.16	12.09
Average	62.32	18.01	28.73	6.24

We note that our method does not depend on any third party computer vision software other than some standard MATLAB functions.

We assess the effectiveness of the proposed framework on the development dataset for the one-shot-learning CHALEARN gesture challenge. The development dataset consists of 20 batches of gestures. Each batch is made of 47 gesture videos and split into a training set and a test set. The training set includes a small set of vocabulary spanning from 8 to 15 gestures. Every test video contains gestures from 0 to 5 gestures.

The recognition performance is evaluated using the Levenshtein distance [14]. Table 3 shows the average errors over 20 batches. As Table 3 reveals, our method significantly outperforms the baseline algorithm [1] and achieves 28.73% average Levenshtein distance per ges-

ture. This illustrates that our method can be effectively adopted for one-shot-learning from the traditional supervised learning scheme.

While our method performs well on the one-shot-learning CHALEARN gesture challenge, it is not a complete system yet. There are three particular batches that cause difficulties on our algorithm. These batches are devel03, devel10, and devel19 where the example frames are shown in Figure 5. These three batches share a common characteristic that the gesture is only distinguishable by identifying the hand positions. Since we do not have a hand detector, the gross motion dominates the whole action causing it to be confused with other similar gestures.

Another source of errors is made by the sequence segmentation. It is supposed that the actor will return to the rest position before performing a new gesture. However, this rule has not always been observed resulting in a mismatch between the micro-gesture and the target gesture. Currently, the large error in devel03 is caused by the need for hand positions and sequence segmentation.

Nevertheless, our method is encouraging since we are capable of recognizing both hand gestures and body gestures. Once we have a reliable hand detector, we expect to further improve gesture recognition from a single training example.

5. Conclusions

We have presented a least squares regression framework on manifolds and underscored its applicability on gesture recognition. The principle of our regression framework is based upon latent geometry in the data. Taking geometric properties in the least squares regression framework, we formulate it as a composite function. Consequently, least squares regression is performed on a manifold. Experimental results demonstrate that our method is effective for gesture recognition and generalizes well in one-shot-learning.

Our proposed framework is not limited to gesture recognition. Future work will focus on exploring other visual applications and regression models on manifolds.

References

[1] Chalearn gesture dataset (cgd 2011), chalearn, california, 2011.

[2] M. F. Abdelkader, W. Abd-Almageeda, A. Srivastavab, and R. Chellappa. Silhouette-based gesture and action recognition via modeling trajectories on riemannian manifolds. *CVIU*, 115(3):439–455, 2011.

[3] P.-A. Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematicae*, 80:199–220, 2004.

[4] P. Bilinski and F. Bremond. Evaluation of local descriptors for action recognition in videos. In *ICVS*, 2011.

[5] A. Björck and G. Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of Computation*, pages 579–594, 1973.

[6] Z. Cheng, L. Qin, Q. Huang, S. Jiang, and Q. Tian. Group activities recognition by gaussian processes estimation. In *ICPR*, 2010.

[7] A. Edelman, R. Arias, and S. Smith. The geometry of algorithms with orthogonal constraints. *SIAM J. Matrix Anal. Appl.*, (2):303–353, 1999.

[8] M. T. Harandi, C. Sanderson, A. Wiliem, and B. C. Lovell. Kernel analysis over riemannian manifolds for visual recognition of actions, pedestrians and textures. In *WACV*, 2012.

[9] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

[10] H. Karcher. Riemannian center of mass and mollifier smoothing. *Comm. Pure Appl. Math.*, 30(5):509–541, 1977.

[11] T.-K. Kim and R. Cipolla. Gesture recognition under small sample size. In *ACCV*, 2007.

[12] T.-K. Kim and R. Cipolla. Canonical correlation analysis of video volume tensors for action categorization and detection. *PAMI*, 31(8):1415–1428, 2009.

[13] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21:1253–1278, 2000.

[14] V. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710, 1966.

[15] Z. Lin, Z. Jiang, and L. Davis. Recognizing actions by shape-motion prototype trees. In *ICCV*, 2009.

[16] Y. Liu, Y. Liu, and K. C. C. Chan. Ordinal regression via manifold learning. In *AAAI*, 2011.

[17] Y. M. Lui. Advances in matrix manifolds for computer vision. *Image and Vision Computing (to appear)*.

[18] Y. M. Lui, J. R. Beveridge, and M. Kirby. Action classification on product manifolds. In *CVPR*, 2010.

[19] G. Meyer, S. Bonnabel, and R. Sepulchre. Linear regression under fixed-rank constraints: a riemannian approach. In *ICML*, 2011.

[20] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, Cybernetics - Part C: Applications and Reviews*, 37:311–324, 2007.

[21] D.-S. Pham and S. Venkatesh. Robust learning of discriminative projection for multicategory classification on the stiefel manifold. In *CVPR*, 2008.

[22] W. R. Schwartz, A. Kembhavi, D. Harwood, and L. S. Davis. Human detection using partial least squares analysis. In *ICCV*, 2009.

[23] Y. Yuan, H. Zheng, Z. Li, and D. Zhang. Video action recognition with spatio-temporal graph embedding and spline modeling. In *ICASSP*, 2010.