Learning Human Actions

Kanish Manuja
Vempati Anurag Sai
Advisor: Dr.Amitabh Mukherjee
{kanish, vanurag, amit}@iitk.ac.in
Dept. of Computer Science and Engineering

ABSTRACT

In spite of the wide applications of action detection, much work hasn't been done on it.In this project,we extended the use of the techniques existing in object recognition in 2D images to video sequences by determining the Spatio-temporal Interest points(corner points for 2D image) using an extension of Harris operator in the time dimension[2]. Features descriptors are computed on the cuboids around these interest points and further they are clustered and bag of features is built .SVM is used to classify the action class of the video to which it belongs. We have trained the agent for two classes of actions, namely handshakes and Standing up. The clips are taken from Hollywood movie database which have been automatically extracted using movie scripts in Laptev's work[1].

1. Introduction

Automatic categorization of human actions in video sequences is very interesting for a variety of applications: detecting activities in video surveillance, indexing video sequences, content based browsing etc. However, it remains a challenging problem because of factors like background cluttering, camera motion, occlusion and other geometric and photometric variances but some of these problems are surprisingly well handled with bag of words representation combined with machine learning techniques like SVM(Support Vector Machining). We would try to categorize only two classes of human actions, namely handshakes and standing up as the time taken to perform the computations for a video is significant.

a)Standing up b)Handshake

Fig 1.:Two classes of action taken for categorizing the actions.

2. The Layout

The project involves the following main stages:

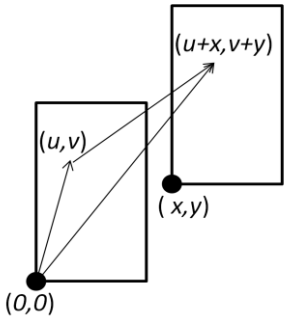
a) Identifying Space-Time interest points (STIP) in a video and collecting cuboid prototypes around them.
b) Transformations on the cuboids and building feature descriptors.
c) Building Spatio-temporal Bag of features (BoF).
d) Using Support Vector Machine (SVM) to train the agent to classify various classes of videos.

a) Detecting Space-Time interest points:

Space-Time interest points (**STIP**): Can either correspond to corners in a particular frame or instances of sudden variation in time occurring in the video. We assume that the useful information required for distinguishing between various actions lie across these STIPs.

The aim is to obtain robust, stable and well-defined image features for object tracking and recognition. Corner detection was being used for many years in the past for detecting interest points in images. We will be using Harris affine detector for our purpose. It relies heavily on both the Harris measure and a Gaussian scale-space representation. The algorithm basically tries to extend the principle of Harris Detector to detect STIPs in a video clip.

Harris Detector: The algorithm relies on a central principle that, at a corner, the image intensity will change largely in multiple directions.



Let this image be given by *I*. Consider taking an image patch over the area (*u*,*v*) and shifting it by(*x*,*y*). The weighted *sum of squared differences* (SSD) between these two patches, denoted ***S***, is given by:

$$S(x,y) = \sum_u \sum_v w(u,v)\ \left(I(u+x, v+y) - I(u,v)\right)^2$$

By using Taylor expansion for ***I* (*u* + *x*, *v* + *y*)** as,

$$I(u+x, v+y) \approx I(u,v) + I_x(u,v)x + I_y(u,v)y,$$

We can simplify the expression to,

$$S(x,y) \approx \sum_u \sum_v w(u,v)\ \left(I_x(u,v)x + I_y(u,v)y\right)^2,$$

$$S(x,y) \approx \begin{pmatrix} x & y \end{pmatrix} A \begin{pmatrix} x \\ y \end{pmatrix},$$

Where A is given by,

$$A = \sum_u \sum_v w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix},$$

I_x and I_y are the partial derivatives of *I*.

This matrix is a Harris matrix, and angle brackets denote averaging (i.e. summation over (*u*,*v*)). If a circular window (or circularly weighted window, such as a Gaussian) is used, then the response will be isotropic.
A corner (or in general an interest point) is characterized by a large variation of ***S*** in all directions of the vector (*x*,*y*). By analyzing the eigenvalues of ***A***, this characterization can be expressed in the following way:

A should have two "large" eigenvalues for an interest point. Based on the magnitudes of the eigenvalues, the following inferences can be made based on this argument:

- If $\lambda_1 \approx 0$ and $\lambda_2 \approx 0$ then this pixel (*x,y*) has no features of interest.
- If $\lambda_1 \approx 0$ and λ_2 has some large positive value, then an edge is found.
- If λ_1 and λ_2 have large positive values, then a corner is found.

But since evaluating eigenvalues takes a lot of computations, an alternative was proposed. A corner point is a point where the function '***Mc***' is locally maximum and above some threshold.

$$M_c = \lambda_1 \lambda_2 - \kappa \left(\lambda_1 + \lambda_2\right)^2 = \det(A) - \kappa \ \mathrm{trace}^2(A)$$

Analogous to spatial domain, [2] considers a Spatio-temporal second-moment matrix '***A'*** which is a 3x3 matrix with first order spatial and temporal derivatives averaged using a Gaussian weighting function.

$$A = \begin{bmatrix} L_x^2 & L_x L_y & L_x L_t \\ L_x L_y & L_y^2 & L_y L_t \\ L_x L_t & L_y L_t & L_t^2 \end{bmatrix},$$

L(x,y,t) is the 3-D matrix with pixel values of a video.

And the Harris corner function was also be extended as,

$$\boldsymbol{M_c} = det\ (A) - k * trace^3\ (A)$$

A value of *k* = 0.005 has been used.

This corresponds to $\frac{\lambda_2}{\lambda_1}$ and $\frac{\lambda_3}{\lambda_1}$ both greater than 23 at the point of interest.

Smoothing out the videos: Before we go about detecting STIPs, we need to smooth out the entire video sequence. The reason being that the lighting, rate of actions (the speed at which they are taking place) may vary from one video to another. Let's consider a video sequence as a 3-D matrix with each frame spread out in x-y plane and the 3rd dimension corresponding to temporal domain. We convolve this matrix with a gaussian kernel with independent spatial variance (σ_l) and temporal variance (τ_l) as shown in Fig.2(a) to smooth out the sequence.

As an illustration suppose the Spatio-temporal terrain varies as in Fig.2(a,b). As we can see, the detected peaks depend on the spatial and temporal variances of the gaussian kernel. So, for every video we must use optimal choice of these variances to be able to detect the correct STIPs. But adapting these variances according to the sequence takes significant computation time. So, we will be detecting STIPs at various scales of Spatial and temporal variances.

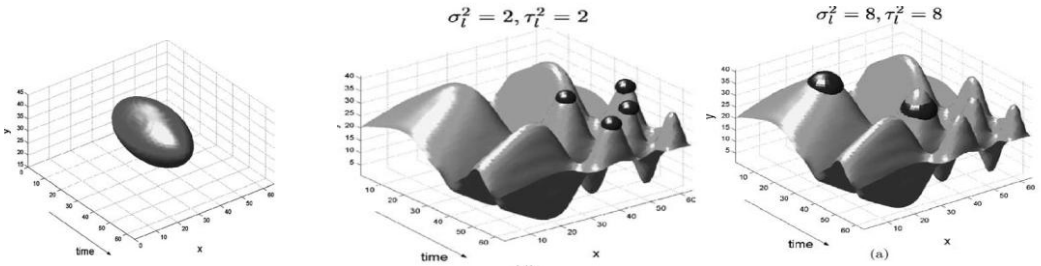


Fig 2. a) Ellipsoid Gaussian b) STIP for σ_l=2, τ_l =2 c) STIP for σ_l =8, τ_l =8

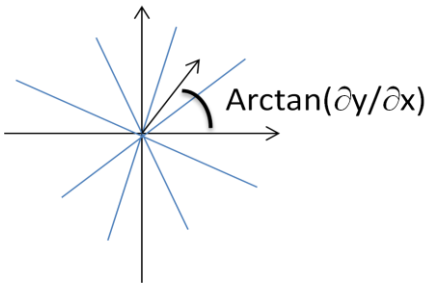
[Image Courtesy: [2]]

b) Building Feature Descriptors (HoG):

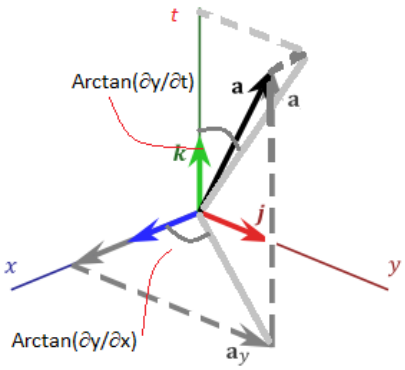
Cuboid Prototypes: Once we have STIPs of a video, we need to extract useful information from them. So we take small chunks of data around these interest points, called the cuboid prototypes. These cuboids contain key features of that particular video sequence.

The size of each chunk (Δ_x , Δ_y , Δ_t) is related to the detection scales by $\Delta_x = \Delta_y = 18\sigma$ and $\Delta_t = 18\tau$. Then, each volume is further divided into $n_x = n_y = 3$ and $n_t = 2$ grids of cuboids. Several transformations are performed on these cuboids to make them invariant to difference in lighting, appearance, motion etc. yet retaining their discriminative power. These might include normalization of pixel values, brightness gradient, windowed optical flow etc. Once the transformations have been performed, a feature vector is generated by stretching out the pixel values of the cubiod into a vector (sensitive to perturbations) or histogramming the values (loss of positional information). In this project, for each cuboid a histogram of oriented gradient (*HoG*) is computed. Normalized histograms are then concatenated into a single HoG feature descriptor.

HoG Descriptor: The essential thought behind the Histogram of Oriented Gradient descriptors is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. This involves Gradient computation and Orientation binning. Gradient computation requires filtering the image with the kernels [-1,0,1] and $[-1,0,1]^T$. In the second phase, each pixel in the cell casts a weighted vote for an orientation-based histogram channel based on the values found in the gradient computation. The orientation (unsigned) of the gradient is quantized into 9 bins in the range of 0^o-180^o as shown in the Fig. The magnitude of the gradient at a pixel is used as the weight of the vote that particular pixel casts.



In the similar fashion we have extended this concept to get feature descriptors for space-time volumes. For the gradient computation the cuboids are filtered with the kernels [-1,0,1] oriented along 3 different dimensions. Specifying the orientation of a vector in 3-D requires 2 angles as opposed to 1 angle in a 2-D case. So, we will be quantizing the orientations of the "projection" of the gradient vector onto 2 orthogonal planes (x-y and y-t) as shown in the Fig.



Refer to [3] for an alternative approach on Spatio-temporal HoG descriptor. They use dot product of the gradient vector with the vectors from centre to corners of a dodecahedron and compute histogram from those projections.

c) Building Feature Descriptors (HoG):

Although two instances of the same behaviour may vary significantly in terms of their overall appearance and motion, many of the interest points they give rise to are similar. So, though the number of possible cuboids is unlimited, the number of different "types" of cuboids is limited. A huge number of samples are obtained from the dataset and feature descriptors for each video are evaluated. Given a set of Spatio-temporal features, we build Spatio-temporal bag of features (BoF).

This requires construction of visual vocabulary. For this purpose, we clustered several such features using K-means clustering algorithm to find "types" of features in human behaviour. Each behaviour might have many such features and each feature might be present in many behaviours. But, the "*distribution*" of these features varies significantly for behaviours belonging to different classes. The BoF representation assigns each feature to the nearest cluster and computes the histogram of visual word occurrences over space-time volume to obtain distribution of the features in a video.

d) Support Vector Machine (SVM):

SVM is a supervised learning technique that analyzes data and recognizes patterns, used for classification and regression analysis. It represents the training examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. The test data is then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. SVM essentially tries to find a hyper plane in higher dimension that maximises the separation between examples of various classes.

3. Implementation Details:

The entire project was implemented in MATLAB. We have used Hollywood dataset available on Laptev's [1] website for our project. Firstly, the videos from the dataset are read as 3-D matrices and are converted to greyscale. Smoothing action is performed on these matrices with various scales of spatial and temporal variances chosen as $\sigma_i = 2^{(i+1)/2}$, i = 1,2,3 and $\tau_j = 2^{j/2}$, j = 1,2. After that, the objective function ***Mc*** is evaluated at each pixel and its local maxima are detected to get STIPs. Then, cuboid prototypes were extracted around these STIPs of size (Δ_x , Δ_y , Δ_t) which are related to the detection scales by $\Delta_x = \Delta_y = 18\sigma$ and $\Delta_t = 18\tau$. We then went about modifying 2-D HOG descriptor code [4] to get feature descriptors for each volume. These descriptors are clustered using fast k-means code [5] to obtain feature histogram for each video. We than used "LibSVM" [6] to learn the histograms of videos belonging to 2 different classes ("handshake" and "stand-up"). We used a radial basis kernel for classification in the SVM. Our code was taking 1 hour to process each video but, we hope the time can be drastically brought down when implemented in OpenCV.

4. Results

The following figure shows the spatio-temporal interest points in a video sequence of a person standing up. The vertical axis is the time axis and a horizontal plane corresponds to a frame in the video. The person is located around (350,100).

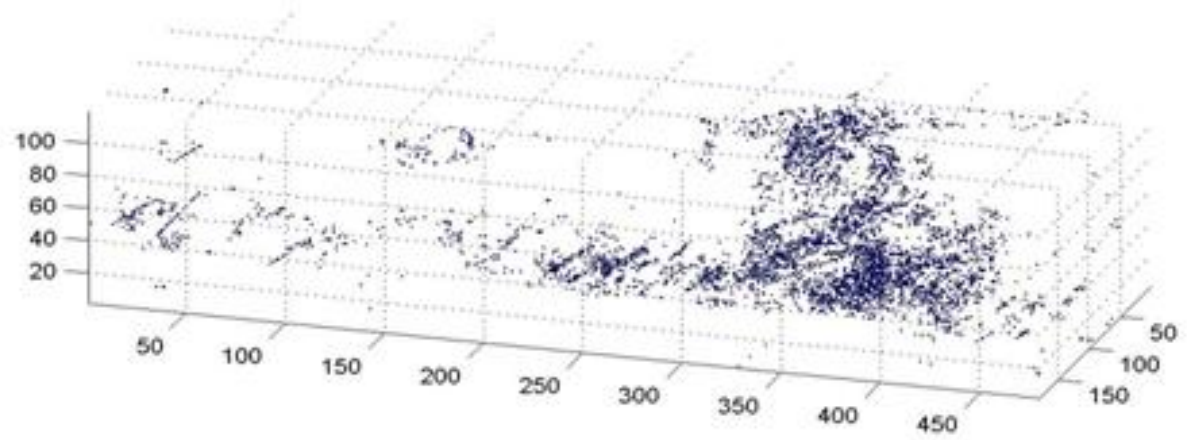
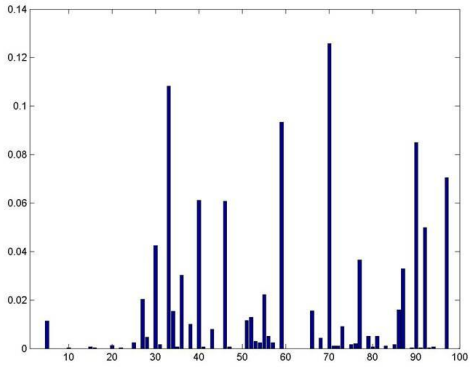


Fig. 3 STIPs of a video in which a person stands up around the (350,100) location.

Fig. 4

Figure 4 shows the Interest points for 3 frames in a video of action class “Standing up”.



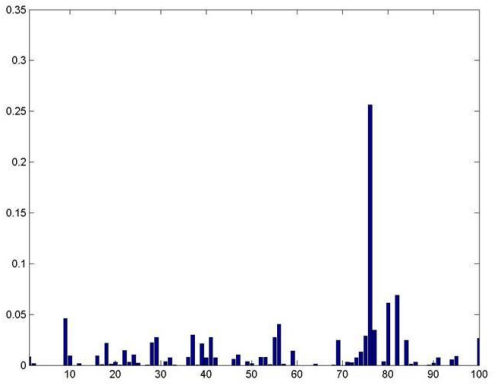


Fig.5 a) Feature Histogram for a Handshake video b) Feature Histogram for a Standing up video

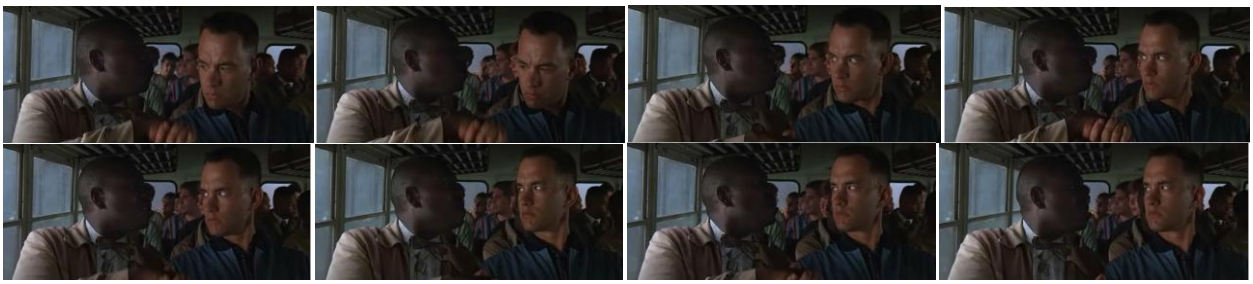
S.V.M. was then used to classify the video sequence into handshakes and Standing up.

The accuracy for standing up was 100% (9 out of 9) and for handshakes, it was 77.8% (7 out of 9). This level of accuracy is better than expected, it may be due to small number of test cases. Since it took a significant amount of time for a video sequence, the number of videos involved in accuracy estimation was on the lower side. The following are a few frames from 2 handshake videos that got misclassified:

1:

As we can see the handshake was getting obstructed by the person which could be the reason for misclassification.

2:



In this video the handshake is taking place at the bottom of the frame and there is a lot of background activity going on, which could be the reason for misclassification.

5. Further work

The accuracy obtained for the two classes is very good , the S.V.M. can be trained for more action classes. The codes were written in MATLAB, hence the time for computing spatio-temporal interest points, Hog, and K means clustering was inconveniently large. The same code can be written in OpenCV which would help in further analysis. Joint learning has been implemented in object recognition to detect objects that the agent hasn't been trained before. The agent basically tries to take inferences from various known objects to get information about the new object. On the similar lines joint learning can be worked upon for video sequences. Joint Learning hasn't been explored for video sequences till date and with minimal training, it is expected to determine a variety of action classes.

References :

[1] Ivan Laptev, Marcin Marszałek, Cordelia Schmid, Benjamin Rozenfeld. Learning realistic human actions from movies. IEEE Conference on Computer Vision & Pattern Recognition, 2008.
[2] I. Laptev. On space-time interest points. IJCV, 64(2/3):107–123, 2005.
[3] Alexander Klaser, Marcin Marszałek, Cordelia Schmid. A Spatio-Temporal Descriptor Based on 3D-Gradients. In BMVC, 2008.
[4] O. Ludwig, D. Delgado, V. Goncalves, and U. Nunes, 'Trainable Classifier-Fusion Schemes: An Application To Pedestrian Detection,' In: 12th International IEEE Conference On Intelligent Transportation Systems, 2009.
[5] Mark Everingham's code on k-means implementation in C,2003.
[6] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.