Draft Title: Activity Recognition

Sameera Ramasinghe, *Member, IEEE,* John Doe, *Fellow, OSA,* John Doe, *Fellow, OSA,*
Ranga Rodrigo, *Member, IEEE,* and Ajith A. Pasqual, *Member, IEEE*

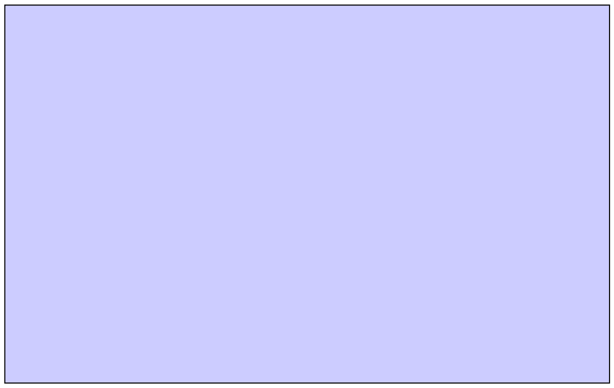

Fig. 1. Test Figure

***Abstract*—This is the abstract ...**

***Index Terms*—Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM).**

I. Introduction

THIS demo file is intended to serve as a “starter file” for IEEE journal papers produced under LaTeX using IEEEtran.cls version 1.8b and later.

Contribution of the motion and static features Total time-evolution Interaction between the actors and objects

Dense trajectories -¿ motion tubes

Earlier paper: no low-level feature gathering (just OF, and static frame -¿ CNN)

[**?**]

II. Method

This is part 1 in the methods section. Good for general method description. See the test figure **??**

A. Motion Features

1) Clustering: In order to isolate each sub area in a frame, which contains significant movement, DBScan clustering is applied on dense trajectory points. Our clustering algorithm is illustrated in the psudocode in figure. We use 8 and 10 as epsilon and minpoint parameters.

Eventhough there are many regions, which contain motion, in a video, some are not significant, nor discriptive. Those moving regions can be neglected without loss of performance. In order to achieve this, after clustering each trajectory point in to cluster groups, non-significant cluster groups are removed from consideration. This is done, in order to prevent the algorithm focusing on small random moving areas in a video, and to only create motion tubes along areas, which are significant and discriptive. Therefore, all the clusters, which are not atleast 50% of the largest cluster of the frame, are discarded.

After identifying the initial candidate clusters, for creating motion tubes, further processing is done to each cluster to

Algorithm 1 Clustering algorithm

```
function DBSCAN($Dataset, epsilon, MinPoints$)
    $Cluster = 0$
    for $each$ $P$ in $Dataset$ do
        if $P$ is $visited$ then
            $continue$
        end if
        mark $P$ as $visited$
        $NeighborPts = getAllPoints(P, epsilon)$
        if $size(NeighborPts) < MinPoints$ then
            mark $P$ as $NOISE$
        else
            $Cluster$ = next cluster
            $addToCluster(P, NeighborPts, Cluster, epsilon, MinP$
        end if
    end for
end function
function ADDTOCLUSTER($P, NeighborPts, Cluster, epsilon, MinPoints$)
    $addPtoclusterCluster$
    for $each$ $point$ $np$ in $NeighborPts$ do
        if $P$ is $not$ $visited$ then
            $mark$ $np$ as $visited$
            $NeighborPts' = getAllPoints(np, epsilon)$
            if $size(NeighborPts') >= MinPoints$ then
                $NeighborPts = NeighborPtsjoinedwithNeighborPts'$
            end if
        end if
        if $np$ is $not$ yet member of any cluster then
            $add$ $np$ to cluster $Cluster$
        end if
    end for
end function
function GETALLPOINTS($P, epsilon$)
    $return$ all points within $P$'s $epsilon$-neighborhood (including $P$)
end function
```

S. Ramasinghe was with the Department of Electronic and Telecommunication Engineering, the University of Moratuwa, 10400, Sri Lanka, e-mail: (see http://www.michaelshell.org/contact.html).

J. Doe and J. Doe are with Anonymous University.

Manuscript received January XX, 201X; revised XXXX 26, 201X.

ensure that they contain only the important moving areas, according to the algorithm in figure. For each point, the chebychev distance from the centroid is calculated, and, furthest 20% of the points, with respect to the total number of points, are discarded from the cluster. The reason behind the choice of chebychev distance, over euclidian distance, is that, a symmetric square shaped cluster groups are obtained, rather than a circle. Which makes it easier to track moving areas, and create motion tubes.

$$D_{chebychev} = max(|X_2 - X_1|, |Y_2 - Y_1|) \tag{1}$$

Algorithm 2

```
1: function FORMATCLUSTER($inCluster, maxChebichevDistance$)
2:     $maxD_{chebichev}$ = $maxChebichevDistance$
3:     $totalPoints = getTotalPoints(inCluster, maxD_{chebichev})$
4:     $currentPoints$ = $totalPoints$
5:     while $true$ do
6:         if $number_of_count_currentPoints < number_of_count_totalPoints * 80/100$ then
7:             return $allthepointsbellongtocurrentPoints$
8:         end if
9:         $maxD_{chebichev}$ = $maxD_{chebichev}$ - 1
10:        $currentPoints = getTotalPoints(inCluster, maxD_{chebichev})$
11:    end while
12: end function
```

After identifying square shaped interest regions (action boxes), each of them is represented with a vector $b = (x, y, r)$, where, x, y are the coordinates of the top left corner of the box and r is the hight/width of the box.

2) Motion Tubes: Since our work models the time evolution of activities, within a video, we divide each input video V_i into temporal segments $V_i = [v_{i,1}, v_{i,2}, ...v_{i,n}]$, and create features for each individual segment seperately. Therefore, after creating the action boxes for each video segment, the action boxes within a segment V_i can be represented as per equation,

$$f(V_i) = \{[b_{i,1,1}, b_{i,1,2}, ..., b_{i,1,q}], [b_{i,2,1}, b_{i,2,2}, ..., b_{i,2,p}], ..., [b_{i,n,1}, b_{i,n,2}, ..., b_{i,n,k}]\} \tag{2}$$

Where $b_{i,j,k}$ is the k^{th} action box in j^{th} frame of the i^{th} video segment. Note that the number of action boxes diffe from frame to frame.

Therfore, Before linking the action boxes to create motion tubes, further pre processing is needed to ensure same number of action boxes exists in every frame, within a video segment. Otherwise, motion tubes could become joined halfway though the video, and capturing dynamics of each moving object seperately, is disturbed.

Therefore, mean action box number per a frame is calculated. And we iterate through each frame, until we come to a frame, which has the mean number action boxes.

Then from that frame, we propagate forward and back along the frames, to eliminate, or add action boxes.

If the action box count in a particular frame, is larger than the previous frame, the excess number of action boxes are removed, in the decending order of the size of the action box. If the case, where action box count is lower than the previous frame, linear regression for each value $x, yandr$ of vector $b = (x, y, r)$, up to that frame, is used to create artificial action boxes, until the number of action boxes match the mean number of action boxes, as follows.

$$\begin{bmatrix} Y_1 \\ Y_2 \\ .. \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ \multicolumn{2}{c}{......} \\ 1 & X_n \end{bmatrix} * \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ .. \\ \epsilon_n \end{bmatrix}$$

Where, X may be x, y or r. Then least square method is used to predict. Note that n increases as the observed action boxes increases, hence the prediction becomes more accurate. Note after this processing, equation 3 is transformed in to equation 4, which illustrates that the number of action boxes per frame is equal for all the frames within a video segment.

$$f(V_i) = \{[b_{i,1,1}, b_{i,1,2}, ..., b_{i,1,k}], [b_{i,2,1}, b_{i,2,2}, ..., b_{i,2,k}], ..., [b_{i,n,1}, b_{i,n,2}, ..., b_{i,n,k}]\} \tag{3}$$

The following procedure is used to link the action boxes within consequetive frames.

Assume $b_{t,1}$, $b_{t,2}$,...,$b_{t,k}$ and $b_{t+1,1}$, $b_{t+1,2}$,...,$b_{t+1,k}$ are action boxes in two consequetive frames at time $t, t+1$. Then, following distance matrix is caculated.

$$D = \begin{bmatrix} D_{11} & D_{12} & D_{13} & \dots & D_{1k} \\ D_{21} & D_{22} & D_{23} & \dots & D_{2k} \\ \multicolumn{5}{c}{\dots\dots\dots\dots\dots\dots\dots\dots} \\ D_{k1} & D_{k2} & D_{k3} & \dots & D_{kk} \end{bmatrix}$$

Where, $D_{i,j}$ is the euclidian distance, between the centroids of i^{th} action box in t^{th} frame and j^{th} action box in $(t+1)^{th}$ frame. Then, u^{th} action box at $t+1$, and 1_{st} action box at t is linked, where u is calculated using,

$$u = \underset{j \in J}{argmin_j}\{D_{1,j}\}, J = \{1, 2, ..., l\} \tag{4}$$

Then, the 1_{st} row and the j_{th} column is removed from the distance matrix, and the same process is applied repeatedly to link each of the action boxes at t with $t+1$. By this removal process, we avoid combining of motion tubes in halfway through the video segment, and keep them isolated from each other, which is vital to capture the dynamics seperately, for each moving object.

Finally, we create matrix M_i which encodes all the information in equation

$$M = \begin{bmatrix} 1 & 1 & x_{1,1} & y_{1,1} & r_{1,1} \\ 1 & 2 & x_{1,2} & y_{1,2} & r_{1,2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & n & x_{1,n} & y_{1,n} & r_{1,n} \\ 2 & 1 & x_{2,1} & y_{2,1} & r_{2,1} \\ 2 & 2 & x_{2,2} & y_{2,2} & r_{2,2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 2 & n & x_{2,n} & y_{2,n} & r_{2,n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ z & 1 & x_{z,1} & y_{z,1} & r_{z,1} \\ z & 2 & x_{z,2} & y_{z,2} & r_{z,2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ z & n & x_{z,n} & y_{z,n} & r_{z,n} \end{bmatrix}$$

3) Histogram Oriented Optic Flows - HOOF: After identifying the k number of motion tubes (n is a variable) for each video segment $v_{i,n}$, the optic flows along each motion tube, , are calculated, using Lucas et.al. After that, Histogram-Oriented-Optic-Flows, for each motion tube is created. We create HOOFs for every sub-frame within a motion tube. Each flow vector within a spatio-temporal sub-frame within a motion tube, is binned according to its primary angle from the horizontal axis and weighted according to its magnitude. Thus, all optical flow vectors, $v = [x, y]^T$ with direction, $\theta = tan^{-1}(\frac{x}{y})$ in the range,

$$-\frac{\pi}{2} + \pi\frac{b-1}{B} \leq \theta < -\frac{\pi}{2} + \pi\frac{b}{B} \quad (5)$$

will contribute by $\sqrt{x^2 + y^2}$ to the sum in bin b, $1 \leq b \leq B$ out of a total of B, bins. Finally, the histogram is normalized to sum up to 1. We choose 100 as the bin number.

4) Bag of HOOFs: We use a bag of features method to create a motion discriptor for each video segment. First, we create a code book for HOOF vectors. All the HOOF vectors of all the video segments in all video classes are clustered using k-means clustering, and 1000 cluster heads are identified. We choose the number of cluster heads as 1000, because, the dimensions final motion discriptors are needed to be same as static discriptors, which is explained later. Then for each video segment v_i. a histogram is calculated using

$$\forall k \in \{1, 2, ...n\} \quad (6)$$

$$H(p)_i = H(p)_i + 1 \quad (7)$$

where $H_i(p)$ is the p^{th} value of histogram of video segment v_i, and

$$p = argmin_j(Thf_j - hf_{i,k}) \quad (8)$$

This $H = [H_1, H_2, ...H_n]$ is the vector time series, which encodes the time evolution of motion information, in the video.

B. Static Features

In order to create static discriptors, we create a CNN of 1000 output classes, and train it on image-net dataset. The architecture is shown in figure. After 30 epochs, the training is stopped and used on each individual frame of each video segement. Then, the output vectors of the CNN, are averaged along indexes, and a static discritor s_i is obtained foe each video segment v_i. Following the procidure for every v_i, finally, we have a vector time series $S = [s_1, s_2, ...s_n]$, representing the static time evolution of the whole video.

III. Experiments and Results

Experiments, results, comparison, and interpretations.

IV. Conclusion

The conclusion goes here. This is the conclusion

Appendix A
Proof of the First Zonklar Equation

Appendix one text goes here.

Appendix B

Appendix two text goes here.

Acknowledgment

The authors would like to thank the National Research Council of Sri Lanka for support received under the Grant NRC/XX

PLACE PHOTO HERE

Sameera Ramasinghe Biography text here.

PLACE PHOTO HERE

Michael Shell Biography text here.

PLACE
PHOTO
HERE

Michael Shell Biography text here.

PLACE
PHOTO
HERE

Ranga Rodrigo Biography text here.

PLACE
PHOTO
HERE

Ajith A. Pasqual Biography text here.