ReHAR: Robust and Efficient Human Activity Recognition

Xin Li Mooi Choo Chuah
Department of Computer Science and Engineering, Lehigh University
xil915@lehigh.edu chuah@cse.lehigh.edu

Abstract

Designing a scheme that can achieve a good performance in predicting single person activities and group activities is a challenging task. In this paper, we propose a novel robust and efficient human activity recognition scheme called ReHAR, which can be used to handle single person activities and group activities prediction. First, we generate an optical flow image for each video frame. Then, both video frames and their corresponding optical flow images are fed into a Single Frame Representation Model to generate representations. Finally, an LSTM is used to predict the final activities based on the generated representations. The whole model is trained end-to-end to allow meaningful representations to be generated for the final activity recognition. We evaluate ReHAR using two well-known datasets: the NCAA Basketball Dataset and the UCFSports Action Dataset. The experimental results show that the proposed ReHAR achieves a higher activity recognition accuracy with an order of magnitude shorter computation time compared to the state-of-the-art methods.

1. Introduction

With technology advancement in embedded system design, powerful cameras have been embedded within smartphones, and wireless cameras can be easily deployed at street corners, traffic lights, big stadiums, train stations, etc. In addition, the growth of online media, surveillance and mobile cameras have resulted in explosion of videos being uploaded to social media sites such as Facebook, Youtube. For example, it is reported that over 300 hours of video are uploaded every minute to Youtube servers. The availability of such vast volume of videos has attracted the computer vision community to conduct much research on human activity recognition since people are arguably the most interesting subjects of such videos. Automatic human activity recognition allows engineers and computer scientists to design smarter surveillance systems, semantically aware video indexes and also more natural human computer interfaces.

Law enforcement tasked with monitoring a large crowd event, e.g., Macys Thanksgiving parade will appreciate having quick human activity recognition analysis on tons of videos captured by cameras deployed over the streets to quickly identify suspicious or criminal behaviors. Similarly, sport fans who may not be able to watch big games in real time will be thrilled if TV broadcasters can provide video-based sport highlights which they can enjoy without watching the whole 3-4 hours games. Image or video based social media sites such as Youtubes are also interested in automatically classifying millions of uploaded videos to provide semantic-aware video indexes to facilitate easy search by viewers. Furthermore, drones have been deployed to conduct surveillance after big natural disaster events, e.g., hurricanes. Having efficient human activity recognition from real-time videos allows emergency workers to quickly spot a small group of people waving on top of the roofs waiting to be rescued or criminals attempting to loot shops for goods while others' attentions are focused on rescue missions.

Despite the explosion of video data, the ability to automatically recognize and understand human activities is still rather limited. This is primarily due to multiple challenges inherent to the recognition task, namely large variability in human execution styles, complexity of the visual stimuli in terms of camera motion, background clutter, viewpoint changes, etc, and the number of activities that can be recognized. Much recent work has proposed deep architectures for activity recognition [1, 2, 3, 4, 5]. [4, 5] both propose convolutional networks which learn filters based on a stack of N input frames but such fixed length approaches cannot learn to recognize complex video sequences, e.g., cooking sequences as presented in [6]. [1] uses recurrent neural networks to learn temporal dynamics using traditional vision feature [1] while [2] uses deep features but both do not train their models end–to-end and hence may not perform well on more complex video sequences. A handcrafted video representation capturing short, medium, and long action dynamcis has also been proposed [7]. However, these approaches cannot infer group activities such as those found in sport related videos captured during volleyballs or basketball tournaments. Recent works have proposed hierarchical-based LSTM models [8, 9] for group activity recognition but these

approaches typically consume huge cloud resources and often run slowly. Faster schemes need to be designed for there are application scenarios that mandate real time requirements, e.g., sport highlights in big games. Computation time becomes more important when we run a deep learning model on mobile devices [10, 11].

In this paper, we propose a robust and efficient human activity recognition scheme, ReHAR, that can infer complex human activities from trimmed video clips and is trainable end-to-end.

In summary, our contributions of this paper include:

- design a robust and efficient human activity recognition scheme to recognize complex human activities, e.g., group activities in sport games.
- extensive evaluation using two popular activity datasets show that our scheme achieves higher accuracy and runs an order of magnitude faster than existing schemes.
- explore the visual explanation for our model to understand what it has learned.

The rest of this paper is organized as follows. In Section 2, we briefly discuss related work, followed by the introduction of some important building blocks in Section 3. In Section 4, we describe our proposed activity recognition scheme and implementation details. We report our experimental results in Section 5. Finally, we conclude this paper in Section 6.

2. Related work

Much work has been done on activity recognition so here we merely summarize the more recent work on group activity recognition which many existing activity recognition schemes cannot handle.

In recent years, researchers have started to work on group activity recognition. Most existing work on group activity recognition has used hand-crafted features in structured models to represent information between individuals in space and time domains [12, 13, 14]. All these approaches however merely use shallow hand crated features and typically adopt a linear model that suffers from representation limitation. In [8, 9], the authors propose a hierarchical model that uses a lower layer LSTM to track each individual and a higher layer LSTM that fuses information from the lower layer LSTMs to recognize group activities. Unfortunately, such approaches are computationally expensive. Thus, a more computationally efficient method must be designed to infer group activities for real time situation awareness applications.

In [15], the authors proposed a semantics based group activity recognition scheme that uses an LSTM model to generate a caption for each video frame and then use another LSTM to predict the final activity categories based on generated captions. Although it achieves a higher accuracy with a shorter running time, it has at least three weaknesses: (1) the caption generation model cannot always generate a perfect caption; (2) the caption generation model is trained only based on its own loss without getting any feedback from the final output the the model. This will result in the generated captions do not contain useful information for the second model to predict the final activities; (3) it is very difficult to access a large dataset which contain caption information, e.g. individual actions in datasets used in [15].

3. Important Building Blocks

We give a brief introduction about some building blocks before we discuss the details of our proposed scheme.

1. Optical Flow: Optical Flow was first proposed by Horn et. al [16]. It is used to describe how each point in the scene moves from a frame to the next. Many improvements have been introduced [17, 18]. Recently, machine learning methods [19, 20, 21, 22] have been used to estimate optical flow by taking two images as their inputs. Among all of these solutions, FlowNet 2.0 [22] achieved the most impressive results by using a stacked structure and fusion network.

2. Image Feature Extraction Via CNN: Convolutional Neural Network (CNN) [23] is a type of feed-forward artificial neural network. It has been widely used in solving different types of tough tasks, e.g. natural language processing [24], image recognition [25], etc. It has been proved that the CNN features contain more representative information of an image than other manually designed feature, e.g. SIFT, by Fischer et al. [26]. Furthermore, Donahue et al. [3] used CNN as a feature extractor to recognize human activities from videos, which showed that CNNs could extract useful information related to activities.

3. Global Average Pooling: In [27], Lin et al. first proposed the Global Average Pooling (GAP) layer. For a traditional image classification network, e.g. VGG16, they first changed the number of the channels in the final Max Pooling layer (block5_pool in VGG16), so that each feature map at this layer corresponded to one image category in the dataset. Then, they replaced the Flatten and the Fully Connected (FC) layers with a GAP layer. The GAP layer took the average of each feature map and fed the results directly into a softmax layer. Based on their experimental results, using GAP layer, a model achieved a slightly better performance than using FC layers. Comparing to the FC layer, the GAP layer has at least two advantages: (1) The GAP layer enforces correspondences between feature maps and categories, thus the feature maps can be easily interpreted as categories confidence maps. (2) There is no parameter to optimize in GAP, thus overfitting is avoided at this layer.

4. Long Short Term Memory: Long Short Term Mem-

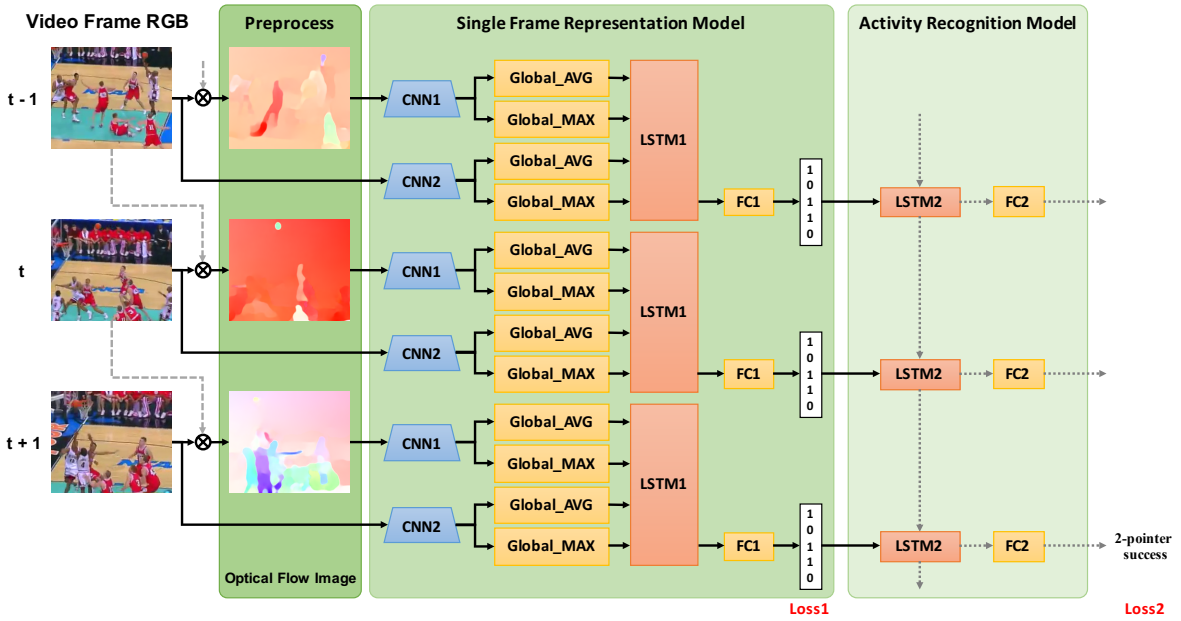


Figure 1: The architecture of the proposed Scheme. Symbol $\otimes$ indicates the operation of computing the dense optical flow using two continuous frames. CNN1 and CNN2 indicate VGG16 (layer "block1_conv1" to layer "block5_pool").

ory (LSTM) model is a particular type of Recurrent Neural Network (RNN) that was first proposed by Hochreiter et al. in [28]. Because of its more powerful update equations and appealing back-propagation dynamics, the LSTM Network works slightly better than the traditional RNN model in practice. Using an LSTM model, Donahue et al. [3] proposed a scheme that yielded a good performance in the tasks of activity recognition, image description, and video description. Moreover, a Neural Image Caption model based on LSTM was proposed by Vinyals et al. [29] to automatically describe the content of an image. Zhang et. al [30] use an LSTM model to count vehicles in city cameras. All of these works prove that the LSTM Network has the capability to extract useful information from its inputs and generate distinguishing representations.

4. Proposed Scheme

We propose a novel end-to-end model for recognizing activities in videos. The intuition of our model is that if we can generate a good representation for every single frame, then the model will be easier to infer the final activity label for the whole video based on these representations. The model, illustrated in Figure 1, consists of three components: (1) Input Preprocessing Model, (2) Single Frame Representation Model, and (3) Activity Recognition Model.

4.1. Preprocessing

We share the same position as Li and Chuah [15] that both the background context and the motion of people contribute towards the group activity recognition. Thus, we also use the original frames (contain environment information) and their corresponding optical flow images (provide motion information) in our scheme. During the preprocessing phase, we feed a video frame (at time t) and its previous one (at time $t-1$) to the FlowNet 2.0 [22] to compute optical flow, since FlowNet 2.0 provides the best performance for generating optical flow. Then, we use the method described in [31] [1] to visualize the optical flow information into a colorful image (3 channels), namely optical flow image. We generate an optical flow image for every frame (except the first one) in a video. The generated optical flow images are illustrated in Figure 1.

4.2. Single Frame Representation Model

The Single Frame Representation Model consists of two CNN feature extractors (one for video frame and another for optical flow image) and an LSTM model. Although any CNN model can be used as a feature extractor in our model, to simplify the explanation, VGG16[2] [32] is used in this section and the size of the video frames and the optical flow images are fixed to (224x224x3).

Once we get the optical flow image at time t, we feed it as well as the corresponding video frame to two CNN models ("CNN1" and "CNN2" correspondingly in Figure 1) to extract features. Instead of only removing the last

[1] http://vision.middlebury.edu/flow/

[2] VGG16: [block1_conv1, block1_conv2, block1_pool, block2_conv1, block2_conv2, block2_pool, block3_conv1, block3_conv2, block3_conv3, block3_pool, block4_conv1, block4_conv2, block4_conv3, block4_pool, block5_conv1, block5_conv2, block5_conv3, block5_pool, flatten, fc1, fc2, prediction]

prediction layer from VGG16 as in [3], we remove the last 4 layers (flatten, fc1, fc2, and prediction). Thus, the "CNN1" and "CNN2" model in Figure 1 include layers from "block1_conv1" to "block5_pool" of VGG16. The layer "block5_pool" has (7x7x512) output size. Then, we add a Global Average Pooling layer and a Global Maximum pooling layer ("Global_AVG" (1x512) and "Global_MAX" (1x512) respectively in Figure 1) to its end. The advantages of doing this have been discussed in Section 3. After that, we feed the output of these global pooling layers to an LSTM model (LSTM1 in Figure 1), which means the LSTM model has 4 input steps and each step has 512 dimensions. A fully-connected layer (FC1 in Figure 1) with a "softmax" activation function is added to the output of the final step of the LSTM1 to generate the representation for each input video frame.

4.3. Activity Recognition Model

The model predicts the final activity label based on a sequence of generated single frame representations. The Activity Recognition Model is an LSTM network (LSTM2 in Figure 1) that takes the single frame representations as its input. Thus, the input step $time\_step$ of LSTM2 equals to the number of the current input video frames. In Figure 1, $time\_step = 3$. Then, the output of the final step of the LSTM2 is fed into a fully-connected layer (FC2 in Figure 1) with a "softmax" activation function to predict the final activity label.

4.4. Implementation Details

Our scheme is implemented using Python Programming Language and Keras Library [33] with Tensorflow [34] backend. We report the implementation details of our scheme and the settings of important parameters as follows.

Optimization: We train our model as a multi-task learning. The overall loss can be computed as:

$$Loss = (\sum_{t=1}^{time\_step} loss_{1,t}) + \lambda * loss_2 \quad (1)$$

where $time\_step$ is the number of frames based on which the model predicts the final activity label (in Figure 1, $time\_step = 3$), $loss_{1,t}$ is the loss of the generated single frame representation at time t and $loss_2$ is the loss of prediction of the final activity. λ is the parameter that is used to balance the single frame representation generation loss and the final activity classification loss. In our experiment, we set $\lambda = 2$ to assign a higher weight to the final activity prediction, considering that the final activity prediction is our final purpose. The model is trained to minimize the $Loss$.

Single Frame Representation Model: The LSTM1 is a single layer LSTM with 200 hidden units. For FC1 layer, we set the dimension of its output to the number of the final activities and its training ground truth to be the one-hot vector of the final activity label. We train the Single Frame Representation Model as a classification task. In this case the representation is the probability distribution of each video frame over all activities. Thus, the $loss_1$ at time t, donated as $loss_{1,t}$, can be computed using categorical cross entropy loss:

$$loss_{1,t} = -\sum_{i=1} g_{t,i} \log(p_{t,i}) \quad (2)$$

where g are the ground truth and p are the predictions. During the testing phase, the model will generate a probability vector as a representation for each frame.

Activity Recognition Model: The LSTM2 is also a single layer LSTM with 200 hidden units. The output of the FC2 is set to the number of categories. To train the model for a classification task, we train the $loss_2$ using categorical cross entropy loss:

$$loss_2 = -\sum_{i=1} g_i \log(p_{ti}) \quad (3)$$

where g are the ground truth and p are the predictions.

Training Process: To speed up the training process and get a better performance, we load the pre-trained VGG16 weights on Imagenet dataset [35]. We train the model using "rmsprop" optimizer with 0.001 learning rate and 1e-8 fuzz factor until the loss becomes converged. Then, we switch the optimizer to SGD with 0.0001 learning rate. The "rmsprop" optimizer helps the model converge quickly, and the SGD with a small learning rate helps to tune the model.

5. Experiments

We run our scheme on a desktop running Ubuntu 14.04 with 4.0GHz Intel Core i7 CPU, 128GB Memory, and a NVIDIA GTX 1080 Graphics Card.

5.1. Datasets

We evaluate our scheme on two well known activity recognition datasets: NCAA Basketball Dataset [9] and UCF Sports Action Dataset [36].

NCAA Basketball Dataset: The NCAA Basketball Dataset[3] was collected by Ramanathan et al. [9] to evaluate the performance of activity recognition schemes on multi-person action videos. It is a subset (257 Basketball Game videos) of the 296 NCAA games available from YouTube[4]. All videos are randomly split into 212 training, 12 validation and 33 testing videos. Each of these videos are split into 4 second clips and sub-sampled to 6fps. They filter out clips which are not profile shots, which results in a total of 11436 training, 856 validation, and 2256 testing video clips. Each of these video clips is manually labeled as one

[3] http://basketballattention.appspot.com/

[4] https://www.youtube.com/user/ncaaondemand

of these 11 labels: 3-pointer success, 3-pointer failure, free-throw success, free-throw failure, layup success, layup failure, other 2-pointer success, other 2-pointer failure, slam dunk success, slam dunk failure or steal success. The Basketball Dataset also annotates the bounding boxes of all the players in a subset of 9000 frames from the training videos. In our scheme, we do not use this location annotation.

UCF Sports Action Dataset: The UCF Sports dataset[5] [36] consists of a set of actions collected from a wide range of stock footage websites including BBC Motion gallery and GettyImages. It consists of a total of 150 videos. Each video has one of these 10 cation categories: diving, golf swing, kicking, lifting, riding horse, running, skateboarding, swinging-bench, swinging-side, and walking.

5.2. Metrics

Mean Average Precision (mAP): Mean Average Precision is the mean of the average precision (AP) scores for each classification category. By computing a precision and recall, one can plot a precision-recall curve, plotting precision $p(r)$ as a function of recall r. Average precision computes the average value of $p(r)$ over the interval from $r = 0$ to $r = 1$ (please refer to wikipedia.org[6]):

$$AP = \int_0^1 p(r)dr \tag{4}$$

Confusion Matrix: A confusion matrix [37] contains information about actual and predicted classifications generated by a classification system. In a confusion matrix, each row represents the predicted classes, while each column represents the instances of an actual class.

5.3. Experiments on the NCAA Basketball Dataset

In this section, we report our experimental results on the NCAA Basketball Dataset. As described in [9], we classify isolated video clips into 11 classes without using any additional negative from other parts of the basketball videos. Each video clip has 24 frames (6fps for 4 seconds). The results are reported in Table 1. Among all 11 categories, our scheme achieves the highest accuracy at 8 categories compared to other baseline models. Overall, our scheme shows a 7.3% accuracy improvement compared to [9] (Atten. track in Table 1). We notice that all methods perform much poorer for categories such as "slam dunk failure". This is because we have very little data (47 training samples and 5 testing samples) belonging to "slam dunk failure" category in the Basketball dataset. The performance is much better for "free-throw" and "3-pointers", because these events have fixed and more obvious patterns (especially for "free-throw") and more training data in this dataset.

[5] http://crcv.ucf.edu/data/UCF_Sports_Action.php

[6] https://en.wikipedia.org/w/index.php?title=Information_retrieval

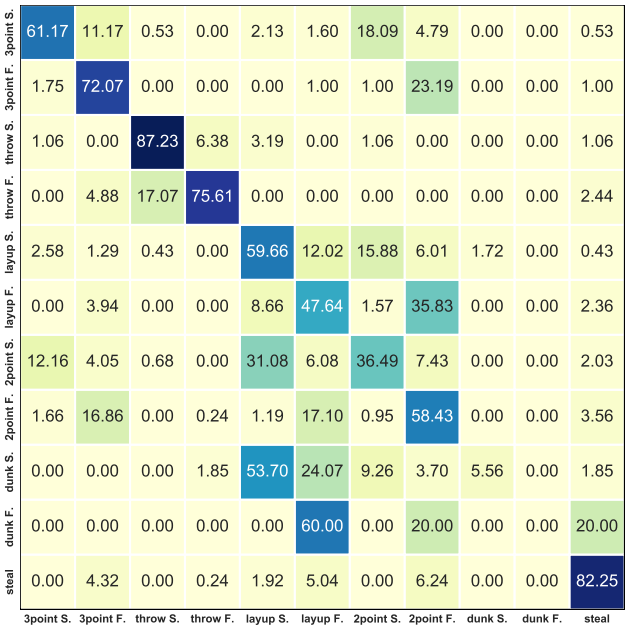


Figure 2: Confusion matrix of action recognition results on NCAA Basketball Dataset.

The confusion matrix for all 11 actions is shown in Figure 2. By analyzing this confusion matrix, one can see that: **(1)** 18.09% "3-pointer success" test samples are incorrectly labeled as "2-pointer success" and 23.19% "3-pointer failure" are labeled as "2-pointer failure". In contrast, 12.16% and 16.86 % "2-pointer success/failure" test samples are incorrectly labeled as "3-pointer success/failure" correspondingly. Based on the rule specification "A player's feet must be completely behind the three-point line at the time of the shot or jump in order to make a three-point attempt; if the player's feet are on or in front of the line, it is a two-point attempt.[7]", one can easily understand that sometime it is hard for a model (even for a person) to extract such detail information to distinguish between 3-pointers and 2-pointers. Although the authors in [9] designed a model to locate the "shooter", they still cannot extract useful enough features to achieve a better performance than our proposed scheme. **(2)** 53.7% and 60.0% "slam dunk success/failure" are predicted as "layup success/failure". The reason is two-fold: a. the training data for "slam dunk success/failure" are not enough; b. "layup" and "slam dunk" have similar action patterns (the shooter jumps under the net and sends the ball to the net).

Here, we would like to highlight an interesting observation. If we group 10 shooting-related actions (except "steal") into two categories (success or failure), then we will get 717 success samples and 1122 failure samples in the testing subset. Based on the output of our model, 88% of

[7] https://en.wikipedia.org/wiki/Three-point_field_goal

	3point S.	3point F.	throw S.	throw F.	layup S.	layup F.	2point S.	2point F.	dunk S.	dunk F.	steal	Mean
IDT [38]	0.370	0.501	0.778	0.365	0.283	0.278	0.136	0.303	0.197	0.004	0.555	0.343
IDT [38] player	0.428	0.481	0.703	0.623	0.300	0.311	0.233	0.285	0.171	**0.010**	0.473	0.365
C3D[39]	0.117	0.282	0.642	0.319	0.195	0.185	0.078	0.254	0.047	0.004	0.303	0.221
MIL[40]	0.237	0.335	0.597	0.318	0.257	0.247	0.224	0.299	0.112	0.005	0.843	0.316
LRCN[3]	0.462	0.564	0.876	0.584	0.463	0.386	0.257	0.378	0.285	0.027	0.876	0.469
Atten. no track[9]	0.583	0.668	0.892	0.671	0.489	0.426	0.281	0.442	0.210	0.006	0.886	0.505
Atten. track[9]	0.600	0.738	0.882	0.516	0.500	**0.445**	0.341	0.471	**0.291**	0.004	0.893	0.516
Ours	**0.753**	**0.766**	**0.933**	**0.857**	**0.613**	0.435	**0.405**	**0.542**	0.232	0.007	**0.940**	**0.589**

Table 1: Mean average precision for event classification given isolated clips of Basketball Dataset. "S." stands for "success" and "F." stands for "failure". All results except ours are extracted from [9].

the test samples (583 success and 1035 failure) are correctly labeled to these two categories. This observation proves that our scheme has the capability to distinguish between shooting success and shooting failure. Sometime, it is hard for people to judge if a shooting is success or not only based on the relative location between the ball and the net, let alone a designed model. Thus, we believe that our scheme achieves a good performance for it benefits from its capability to analyze players' behaviors before and after shooting, and infer the final activity label based on these behaviors. We will discuss more later by visualizing our proposed model.

5.4. Experiments on the UCF Sports Action Dataset

In this subsection, we evaluate the performance of our scheme using the UCF Sports Action Dataset. We follow Lan et al. [41] to split the dataset into training (103 videos) and testing (47 videos) subsets[8]. Among all video clips, the minimum length is 2.2 seconds and the maximum length is 14.4 seconds. We down-sampling all video clips to 24 frames before feeding them into our model. In Table 2, we compare our scheme to other state-of-the-art solutions. Our scheme gets the highest prediction accuracy on 8 out of 10 categories. Comparing to [42], our scheme achieves 6.1% accuracy improvement. In addition, we want to highlight that our scheme performs a perfect prediction (1.0 average precision) on 6 categories. This proves that our model generates more distinguishing features that benefit our model performs better than other existing methods in the task of activity recognition.

Please refer to Figure 3 for more details about our results on the UCF Sports Dataset. One can see that our scheme performs very well on most categories. However, it incorrectly labels some "Walking" testing samples to "Golf" and "SkateBoarding". This is because these samples have some similiar features as samples in those incorrect categories. For example, in video "Walk-Front/006RF1-13902_70016.avi", there is a person walking on a golf course with a golf pole. The environment is definitely related to golf and the motion of the golf pole looks like a person is swinging the pole in front of him. More details will be discussed later by visualizing the model.

[8] http://cs.stanford.edu/~taranlan/other/train_test_split

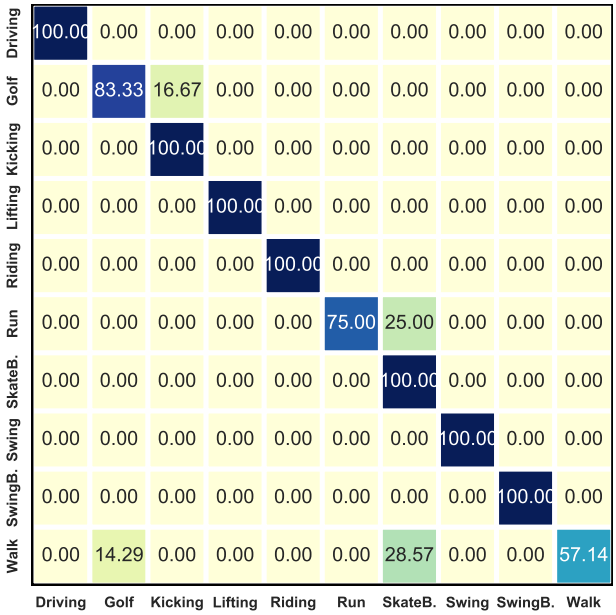


Figure 3: Confusion matrix of action recognition results on UCF Sports Action Dataset.

5.5. Computation Time

As we have already discussed in Section 1, in some application scenarios, predicting an activity label in real time is highly important. Thus, in this subsection, we report the computation time of ReHAR. Computing optical flow images takes FlowNet 2.0 [22] around 7ms (140 fps). We report the computation time of ReHAR (including optical flow images generation time) using different CNN models as base net in Table 3. In total, our model (using VGG16 as its base net) takes 103.65 ms to process 10 input frames and 239.04 ms for 24 input frames. In [15], the computation time of SBGAR model using InceptionV3 as feature extractor and 10 input frames was 108.53 ms. Using the same settings, ReHAR only takes 78.40 ms. Considering that both [8] and [9] predict the activities based on detecting and analyzing every single person and then infer the final activities based on individual actions, they have a sim-

	Diving	Golf	Kicking	Lifting	Riding	Run	SkateB.	Swing	SwingB.	Walk	mAP
Gkioxari et al. [43]	0.758	0.693	0.546	0.991	0.896	0.549	0.298	0.887	0.745	0.447	0.681
Weinzaepfel et al. [44]	0.607	0.776	0.653	1.000	0.995	0.526	0.471	0.889	0.629	0.644	0.719
Peng et al. [45]	0.961	0.805	0.735	0.992	0.976	0.824	0.574	0.836	0.985	0.760	0.845
Hou et al. [42]	0.844	0.908	0.865	0.998	1.000	**0.837**	**0.687**	0.658	0.996	0.878	0.867
Ours	**1.000**	**0.955**	**1.000**	**1.000**	**1.000**	0.806	0.626	**1.000**	**1.000**	**0.888**	**0.928**

Table 2: Mean average precision for event classification given isolated clips of UCF Sports Action Dataset. All results except ours are extracted from [42].

ilar computation time (4.2 seconds on a GTX 1080 reported in [15]). ReHAR runs an order of magnitude faster than [8] and [9]. Thus, our scheme will be more useful for real-time human activity recognition.

CNN base net	Time on 10 Frames (ms)	Time on 24 Frames (ms)
VGG16	103.65	239.04
InceptionV3	78.40	192.02

Table 3: Computation time of ReHAR using different CNN model as its base net (optical flow images generation time included).

5.6. Why does our scheme work?

In previous subsections, we report our comparable results on two well-known activity recognition datasets. In this subsection, we will try to explain the reason why our proposed model works.

First, we explore the necessity of the LSTM1 and the Global Pooling layers in our scheme by comparing baselines' results on UCFSports dataset. Our proposed model achieves 0.928 mAP (Table 2). **(1)** If we remove LSTM1, stack and feed the output of global layers to a Convolutional layer before "FC1" layer, the mAP reduces to 0.766. **(2)** We only get 0.702 mAP after replacing the LSTM1 with an element-wise sum operation. Baseline (1) and (2) are the best fusion methods discussed in [46]. One can see that our LSTM1 generates much better representations than a simple fusion method. **(3)** Replacing the Global Pooling layers with flattened layers, the mAP reduces from 0.928 to 0.889. Thus, using Global Pooling layers helps our model achieve a higher accuracy.

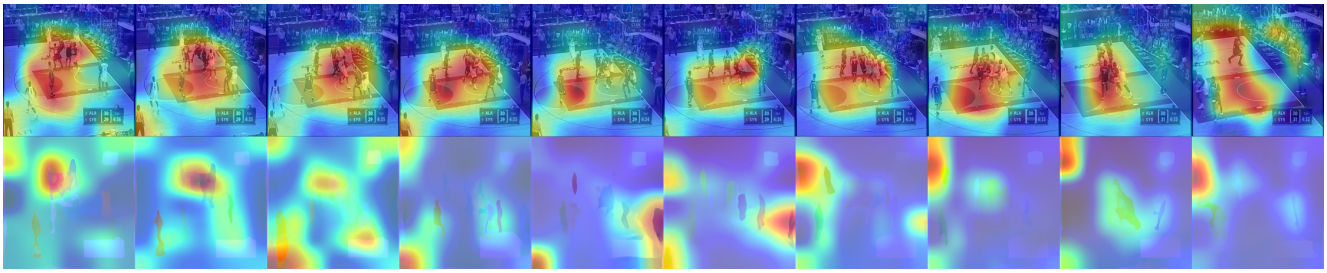
Then, we use the method proposed in [47] to compute the gradient of output category with respect to input image. This should tell us how the output category value changes with respect to a small change in input image pixels. We implement this function by modifying the keras-vis toolkit [48], so that the final class-specific information can be passed back through two LSTMs and fully-connected layers.

Figure 4 shows the visualized results using four samples from the Basketball Dataset and the UCF Sports Action Dataset. **Figure 4a** shows an "other 2-pointer success" event and our model correctly predicts it. One can notice that it is hard for a person to find and track the ball through all frames. The visualized result shows that our model focuses on analyzing the players instead of tracking the ball. At the first frame, the group of the players on the video frame draws the attention of our model, while the model pays more attention on the region of the shooter on the optical flow image. At the last frame, the model stares at the region under the net with only one player left. Based on these information as well as features extracted from intermediate frames, the model predicts a correct activity label. **Figure 4b** shows one player successfully steals the ball from another at sixth frame, and then all players are running towards the other side of the basketball court. The model focuses on a larger region of optical flow images after the sixth frame, because all players (including environment) are moving quickly. From **Figure 4c**, one can notice that the model has the capability to detect the key actor from video frames. There are two people on the images and the model highlights the shooter rather than the referee on most frames. In addition, the model highlights the location of the ball at the last 4 optical flow images. All of these prove that our designed model can focus on important and meaningful things. In **Figure 4d**, we visualize a sample that our model wrongly predicts a "Walking" event to "Golf". The visualized result shows that the model extracts features from the person and the background context on video frames. These features contributes toward "Golf" event. We also notice that at the last optical flow image, the model highlights the region of the golf pole which is located between the person's two legs. Maybe these are the reasons why the model has 97% confidence to label this sample as "Golf".

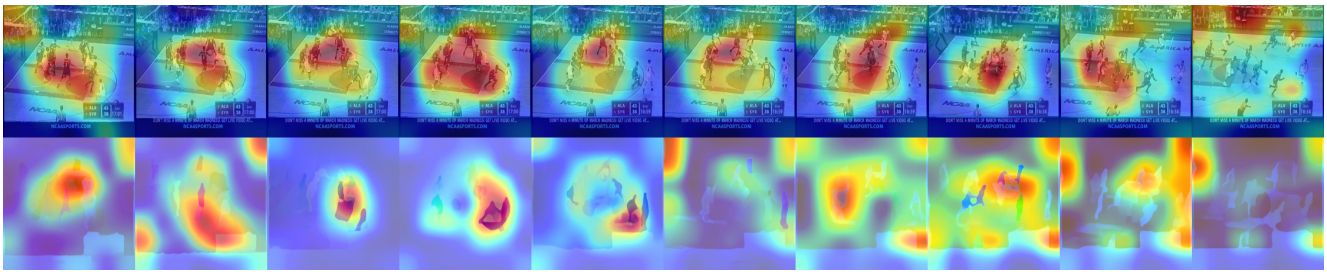
Please refer to our Appendix for more visualized results.

6. Conclusion

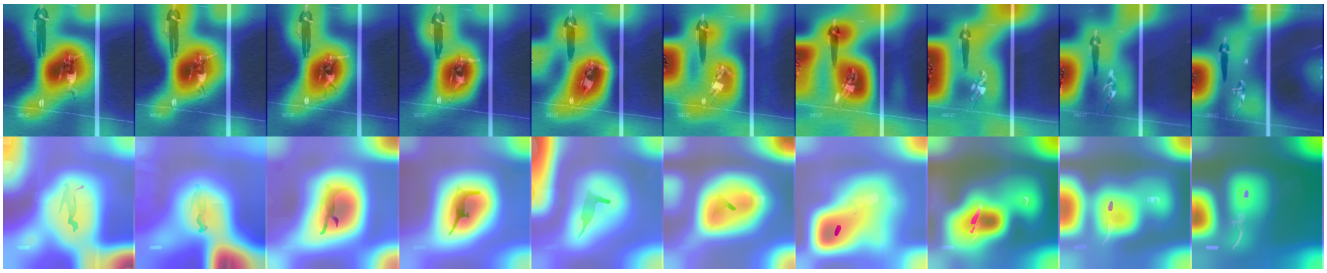
In this paper, we propose a novel scheme (ReHAR) to recognize human activities in videos. The proposed model is trainable end-to-end and achieves a higher accuracy than the existing state-of-the-art solutions on both single person activity and group activity datasets. The experimental results also show that ReHAR runs an order of magnitude faster than other schemes. By visualizing the proposed model, we understand what ReHAR learns and notice that it has the potential capability to detect key actors.



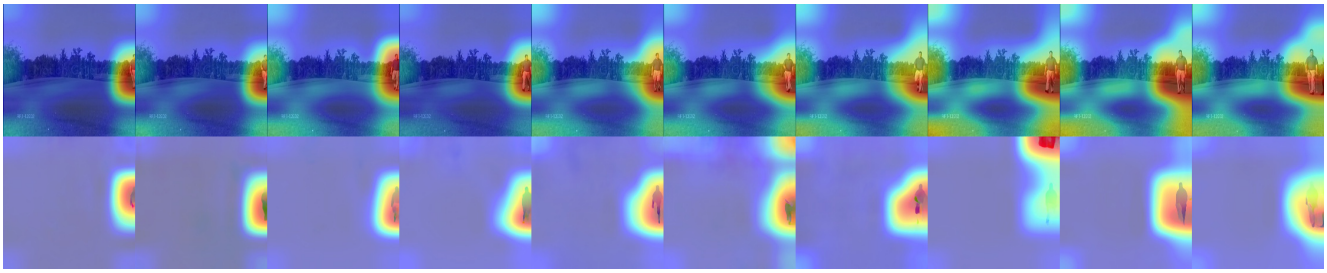
(a) Correctly predict an "other 2-pointer success" event on Basketball Dataset.



(b) Correctly predict a "Steal Success" event on Basketball Dataset.



(c) Correctly predict a "Kicking" event on UCF Sports Action Dataset.



(d) Incorrectly predict a "Walking" event as "Golf" on UCF Sports Action Dataset.

Figure 4: Visualized class-specific important regions on input video frames and optical flow images. The top 2 samples are from Basketball Dataset and the bottom 2 are from UCF Sports Dataset. Each sample has two rows of visualized results. The first row shows the results of video frames, while the second row illustrates the results of optical flow images. Because of the limitation of the space, we only visualize 10 frames of each event.

In the near future, we would like to evaluate the impact of using different CNN models, e.g. C3D [39] or MobileNet, as base net. C3D can extract spatiotemporal features from videos, thus using C3D as our base net may achieve higher accuracy. Comparing to other CNN models, MobileNet runs much faster while maintaining accuracy. Thus, using MobileNet as our base net should speed up our model without losing too much accuracy. Moreover, we are planning to explore the performance of our proposed ReHAR on larger datasets, e.g. UCF101 [49] and THUMOS[50], and in other tasks, e.g. activity detection or event proposal generation in untrimmed videos.

7. Acknowledgement

This work is partially supported by a Qualcomm gift and a GPU donated by NVIDIA.

References

[1] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Action classification in soccer videos with long short term memory recurrent neural networks. *Proceedings of ICANN*, 2010.

[2] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential deep learning for human action recognition. *Human Behavior Understanding*, 2011.

[3] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[4] A. Karpathy, G. Goderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li. Large-scale video classification with convolutional neural networks. In *Proceedings of CVPR*, 2014.

[5] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in video. In *arXiv preprint arxiv:1406.2199*, 2014.

[6] P. Das, C. Xu, R. F. Doell, and J. J. Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[7] Y. Li, W. Li, V. Mahadevan, and V. Vasconcelos. Vlad3: Encoding dynamics of deep features for action recognition. In *Proceeedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[8] Mostafa S. Ibrahim, Srikanth Muralidharan, Zhiwei Deng, Arash Vahdat, and Greg Mori. A hierarchical deep temporal model for group activity recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] Vignesh Ramanathan, Jonathan Huang, Sami Abu-El-Haija, Alexander Gorban, Kevin Murphy, and Li Fei-Fei. Detecting events and key actors in multi-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3043–3053, 2016.

[10] Dawei Li, Xiaolong Wang, and Deguang Kong. Deeprebirth: Accelerating deep neural network execution on mobile devices. *arXiv preprint arXiv:1708.04728*, 2017.

[11] Dawei Li, Theodoros Salonidis, Nirmit V Desai, and Mooi Choo Chuah. Deepcham: Collaborative edge-mediated adaptive deep learning for mobile object recognition. In *Edge Computing (SEC), IEEE/ACM Symposium on*, pages 64–76. IEEE, 2016.

[12] T. Lan, L. Sigal, and G. Mori. Social roles in hierarchical models for human activity recognition. *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[13] Tian Lan, Yang Wang, Weilong Yang, Stephen N Robinovitch, and Greg Mori. discriminativeminative latent models for recognizing contextual group activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1549–1562, 2012.

[14] V. Ramananthan, B. Yao, and F. F. Li. Social role discovery in human events. *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[15] Xin Li and Mooi Choo Chuah. Sbgar: Semantics based group activity recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2876–2885, 2017.

[16] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[17] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. *Computer Vision-ECCV 2004*, pages 25–36, 2004.

[18] Andreas Wedel, Daniel Cremers, Thomas Pock, and Horst Bischof. Structure-and motion-adaptive regularization for high accuracy optic flow. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1663–1668. IEEE, 2009.

[19] Dan Rosenbaum, Daniel Zoran, and Yair Weiss. Learning the local statistics of optical flow. In *Advances in Neural Information Processing Systems*, pages 2373–2381, 2013.

[20] Marius Leordeanu, Andrei Zanfir, and Cristian Sminchisescu. Locally affine sparse-to-dense matching for motion and occlusion estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1721–1728, 2013.

[21] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. *arXiv preprint arXiv:1504.06852*, 2015.

[22] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *arXiv preprint arXiv:1612.01925*, 2016.

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[24] Phil Blunsom, Nando de Freitas, Edward Grefenstette, Karl Moritz Hermann, et al. A deep architecture for semantic parsing. In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, 2014.

[25] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE Transactions on Neural Networks*, pages 98–113, 1997.

[26] Philipp Fischer, Alexey Dosovitskiy, and Thomas Brox. Descriptor matching with convolutional neural networks: a comparison to sift. *arXiv preprint arXiv:1405.5769*, 2014.

[27] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[28] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[29] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: Lessons learned from the 2015 mscoco image captioning challenge. *IEEE transactions on pattern analysis and machine intelligence*, 2016.

[30] Shanghang Zhang, Guanhang Wu, Joao P Costeira, and José MF Moura. Fcn-rlstm: Deep spatio-temporal neural networks for vehicle counting in city cameras. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3687–3696. IEEE, 2017.

[31] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[33] François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[34] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[35] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[36] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[37] Ron Kohavi and Foster Provost. Glossary of terms. editorial for the special issue on applications of machine learning and the knowledge discovery process. *Machine Learning*, 30(2-3):271–274, 1998.

[38] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169–3176. IEEE, 2011.

[39] Du Tran, Lubomir D Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. C3d: generic features for video analysis. *CoRR, abs/1412.0767*, 2(7):8, 2014.

[40] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *Advances in neural information processing systems*, pages 577–584, 2003.

[41] Tian Lan, Yang Wang, and Greg Mori. Discriminative figure-centric models for joint action localization and recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2003–2010. IEEE, 2011.

[42] Rui Hou, Chen Chen, and Mubarak Shah. Tube convolutional neural network (t-cnn) for action detection in videos. *arXiv preprint arXiv:1703.10664*, 2017.

[43] Georgia Gkioxari and Jitendra Malik. Finding action tubes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 759–768, 2015.

[44] Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. Learning to track for spatio-temporal action localization. In *Proceedings of the IEEE international conference on computer vision*, pages 3164–3172, 2015.

[45] Xiaojiang Peng and Cordelia Schmid. Multi-region two-stream r-cnn for action detection. In *European Conference on Computer Vision*, pages 744–759. Springer, 2016.

[46] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1933–1941, 2016.

[47] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *See https://arxiv.org/abs/1610.02391 v3*, 2016.

[48] Raghavendra Kotikalapudi and contributors. keras-vis. `https://github.com/raghakot/keras-vis`, 2017.

[49] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. In *CRCV-TR-12-01*, 2012.

[50] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`, 2014.