S-RAD SINGLE RUN ACTION DETECTOR OVER VIDEO STREAM

by

Anbumalar Saravanan

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Electrical Engineering

Charlotte

2020

Approved by:

_____
Dr. Hamed Tabkhi

_____
Dr. Chen Chen

_____
Dr. Andrew Willis

_____
Professor. Stephen Welch

ABSTRACT

ANBUMALAR SARAVANAN. S-RAD Single Run Action Detector over Video Stream .
(Under the direction of DR. HAMED TABKHI)

Vision based Activity Detection is concerned with the automatic extraction, analysis and understanding of useful information from a sequence of images. It involves the development of a theoretical and algorithmic basis to achieve automatic visual understanding. As a scientific discipline, Vision based systems is concerned with the theory behind Artificial systems that extract information from images. The image data can take many forms, such as video sequences, or views from multiple cameras. Cameras provide very rich information about persons and environments, and their presence is becoming more important in everyday environments like airports, train and bus stations, malls, elderly care and even streets. Therefore, reliable Vision-based action detection system is required for various application like healthcare assistance system, crime detection and sports monitoring system in real time scenarios. Our Approach takes initial strides at designing and evaluating a Vision-based system for privacy ensured human activity monitoring. The proposed technology utilizing Artificial Intelligence (AI)-empowered proactive systems offering continuous monitoring, behavioral analysis, and modeling of human activities. To this end, We presents Single Run Action Detector (S-RAD) which is a real-time privacy-preserving action detector that performs end-to-end action localization and classification. It is based on Faster-RCNN combined with temporal shift modeling and segment based sampling to capture the human actions. Results on UCF-Sports and UR-Fall dataset present comparable accuracy to State-of-the-Art approaches with significantly lower model size and computation demand and the ability for real-time execution on edge embedded device (e.g. Nvidia Jetson Xavier).

DEDICATION

I would like to dedicate this to my parents Saravanan Jayaprakasam, Gowri Saravanan for their financial support. I would also dedicate this to my kid Kiasha and my husband Deepan Dennisbabu for their emotional love and support.

ACKNOWLEDGEMENTS

With an immense pleasure, I would like to express my deep and sincere gratitude to my Advisor and Mentor Dr. Hamed Tabkhi for his encouragement and support throughout my Master study and thesis. This work would have not been possible without his motivation, patience and knowledge. Besides my advisor, I would like to thank the rest of my thesis committee, Dr. Chen Chen, Professor. Stephen Welch and Dr. Andrew Willis for their valuable time. I also want to thank my peers at TeCSAR (Transformative Computer Systems and Architecture Research) for constant support and encouragement where special thanks to Sam Rogers and Justin Sanchez for all those brainstorming sessions. I would also like to thank NSF (National science foundation) for the support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ECE An acronym for Electrical and Computer Engineering.

CNN An acronym for Convolutional Neural Network.

FLOP An acronym for Floating Point Operation.

FPS An acronym for Frame Per Second.

GFLOP An acronym for Giga Floating Point Operation.

mAP An acronym for Mean Average Precision.

R-CNN An acronym for Region based Convolutional neural network.

RGB An acronym for Red Green Blue images.

ROI An acronym for Region Of Interest.

S-RAD An acronym for Single Run Action Detector.

SoC An acronym for System on Chip.

SSD An acronym for Single Shot Detector.

TSM An acronym for Temporal shift module.

CHAPTER 1: INTRODUCTION

In recent years, deep learning has achieved success in fields such as computer vision and natural language processing. Compared to traditional machine learning methods such as support vector and random forest, deep learning has a strong learning ability from the data and can make better use of datasets for feature extraction. Because of this practicability, deep learning had become more and more popular to do research works.

Deep learning models usually adopt hierarchical structures to connect their layers. The output of a lower layer can be regarded as the input of a higher layer using linear or non-linear functions. These models can transform low-level features to high-level abstract features from the input data. Because of this characteristic, deep learning models are stronger than shallow machine learning models in feature representation. The performance of traditional machine-learning methods usually rely on user experiences and handcrafted methods, while deep learning approaches rely on the data.

Video data-based action detection has drawn considerable attention from the academic community recently [1, 2], owing to its applications in many areas such as security and video analytic. Consequently, the utility of video-based action detection has been explored in many applications , refer Figure 1.1. However, previous works either ignore human privacy [3, 4], or satisfy it with less optimal methods. For instance, the work of Chou et al [5] limits their input to low resolution depth images. While this method preserves privacy, it eliminates the possibility of fine-grained human analysis. The work of Asif et al [6] preserves privacy through utilizing synthetic training data. This method fails to address the actual human's privacy, and furthermore the usage of bulky keypoint and segmentation based models forces their approach to be run on a cloud server, further invalidating patient privacy.

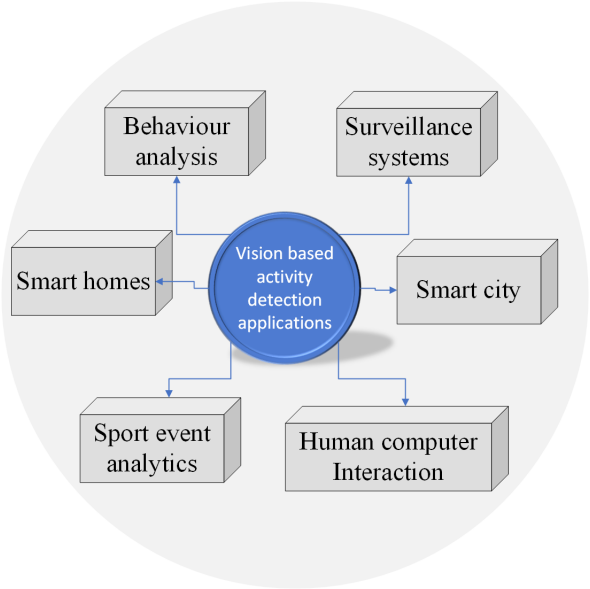


Figure 1.1: Vision based Activity Detection Application

In designing an AI system that supports real-time action detection in videos, there are two important cues that need to be taken into consideration. These include appearances and temporal information. The performance of the detection system depends, to a large extent, on whether it is able to extract and utilize relevant information about human behavior, physical movements, interactions among humans, and interactions between the human and the environment. However, extracting such information is difficult due to a number of challenges, such as scale variations, view point changes, and camera motions that are common when collecting data about behavioral health in uncontrolled environments. Therefore, it becomes crucial to design effective end-to-end systems to overcome these challenges while learning categorical information of human action classes.

The recent approaches in video analytic and deep learning algorithms like Convolutional Neural Network (CNN) provides the opportunity for real-time detection and analysis of human behaviors like walking,running or sitting down, which are part of Daily Living Activities (ADL) [7]. Cameras provide very rich information about persons and environments and their presence is becoming more important in everyday environments like airports, train and bus stations, malls, elderly care and even streets. Therefore, reliable vision-based ac-

tion detection systems is required for various application like healthcare assistance system, crime detection and sports monitoring system. In our research we explored two different domains (Sport and Healthcare), to prove the comprehensive nature of our proposed action detector algorithm. Approaches like [8, 9, 10, 11] use larger CNN models that impose huge computation demand and thus limit their application in real-time constrained systems, in particular on embedded edge devices. Additionally, these methods have not been designed to fulfill requirements of pervasive video systems including privacy-preserving and real-time responsiveness. Other works done in this area are based on the use of wearable sensors. These works used the tri-axial accelerometer, ambient/fusion, vibrations or audio and video to capture the human posture, body shape change. However, wearable sensors require relative strict positioning and thus bring along inconvenience especially in the scenario of healthcare unit where elderly seniors may even forget to wear them.

1.1 Problem Statement

Recent approaches like [11, 10, 12] in Figure [1.2b, 1.2d,1.2e] uses computationally heavy 3D kernels which would require the data to be moved to cloud server, hence looses its purpose for the real-time application and is not aiding in preserving the privacy. And approaches like [13, 1, 14, 8] in Figure[1.2c, 1.2d,1.2e] uses time intensive processing of optical flow method or saliency map generation method and indulges double/triple of the computation when using multi modal input rather than just using RGB images. In

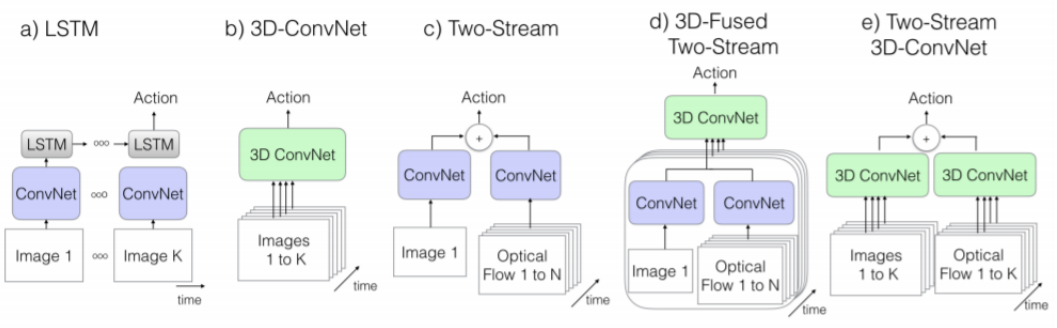


Figure 1.2: Other two stream/ 3D CNN based approaches

addition approaches like [13, 15, 16, 17] are more toward action classification and when applied in real-time, it will fail on the crowded scenes. Some works like [18] uses off-the-shelf external person detector to detect multiple human in the scene and then performs classification on top of it hence it no longer provide an unified approach for the real-time use cases.

1.2 Contributions

In-order to have a unified network that does human action localization and classification together with a simple architecture and a single input, we introduce S-RAD single run action detector. Our contributions are as follows:

1. Contribution 1: We introduce S-RAD, an action detector that provides detection by regressing and classifying the sequence of frames in a single shot manner.

2. Contribution 2: We demonstrate that we can achieve comparable accuracy to the State-of-the-Art approaches (on the UCF-Sports and UR-Fall datasets) at much lower computation cost. We demonstrate our approach on two different dataset from Health-care and Sport domain to prove it's robustness and applicability to multiple action detection domains.

3. Contribution 3: We additionally provide possibility's of extending our network to real-time scenarios on an edge device.

1.3 Thesis Outline

The outline of this thesis is as follows. Background will give you the background of the technology used. It reviews the basic RCNN action detection model, Temporal shift module. Related-Work, briefly overviews the related works in the field of object detector based action detections and spatio-temporal action detections. Chapter1 proposes an overview of the S-RAD approach, temporal sampling strategy explains how the 8 framed input is

chosen from the video to capture action that has long range temporal structure , Base feature extractor explains the components of feature extraction in S-RAD base layer, it covers the backbone and Temporal shift module, proposal network and S-RAD head explains the process of Proposal generation and ROI Align . In Results, State-of-the-Art comparison on dataset UCF-Sport and UR-fall dataset with metrics details are provided . Additionally, Conclusion gives the summary of the approach and finally Future work concludes the area's of improvement in S-RAD for the future work.

CHAPTER 2: BACKGROUND AND RELATED WORK

This chapter covers the necessary background needed to understand our approach in the following chapter.

2.1 Background

2.1.1 Spatio-Temporal Action Localization And Classification

Spatial-temporal action localization and classification aims to recognize the actions of interest that is present in a video and localize them in both space and time. Action localization [19], is the task of classifying what action is being performed in a sequence of frames (or video) as well as localizing each detection both in space and time. The localization can be visualized using bounding boxes or masks. There has been an increased interest in this task in recent years due to the increased availability of computing resources as well as new advances in Convolutional Neural Network architectures. There are several approaches to tackle this task. Most of the approaches revolve around the following approaches: discriminative parts [20, 21], figure-centric models [21, 22], action proposals [23, 24], graph based [25], 3D convolutional neural networks [26], and more.

On the other hand several methods [11, 27] uses 3D CNN to capture spatial-temporal features from the frame sequence with the temporal dimension. However 3D CNN have high computation cost with more parameters than the 2D CNN counterparts, thus are more prone to over-fitting. Our framework has the same spatial temporal modeling ability as the 3D CNN while having the computation and parameter as 2D CNN.

There are several works to trade off between temporal modeling and computation cost, like post-hoc fusion [28, 17, 29, 30] and mid-level temporal fusion [31, 32, 33]. Such methods sacrifice the low-level temporal modeling and much of the useful information is

lost during the feature extraction before the temporal fusion happens.

2.1.2 R-CNN Based Action Detection

2.1.2.1 R-CNN Object Detector

The main idea is composed of two steps. First, using selective search, it identifies a manageable number of bounding-box object region candidates (region of interest or RoI). And then it extracts CNN features from each region independently for classification. To bypass the problem of selecting a huge number of regions, Ross Girshick et al. [34] proposed a method where selective search is used to extract just 2000 regions from the image and called them region proposals. Therefore, now, instead of trying to classify a huge number of regions, you can just work with 2000 regions refer Figure 2.1.

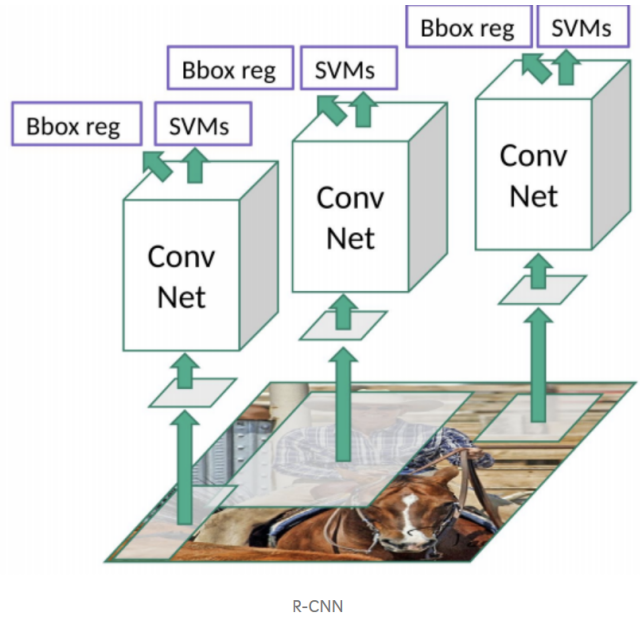


Figure 2.1: Slow R-CNN

Below are the problems of R-CNN:

1. It still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image.

2. It cannot be implemented real-time as it takes around 47 seconds for each test image.

3. The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals.

2.1.2.2 Fast R-CNN Object Detector

Drawbacks of R-CNN are used to build a faster object detection algorithm and it was called Fast R-CNN [35]. The approach is similar to the R-CNN algorithm. But, instead of feeding the region proposals to the CNN, input image is fed into the CNN to generate a convolutional feature map. From the convolutional feature map, the region of proposals are identified and warped into squares and by using a ROI pooling layer it is reshaped into a fixed size so that it can be fed into a fully connected layer refer Figure 2.2. The reason Fast R-CNN is faster than R-CNN is because you dońt have to feed 2000 region proposals to the convolutional neural network every time. Instead, the convolution operation is done only once per image and a feature map is generated from it. However, Fast R-CNN during

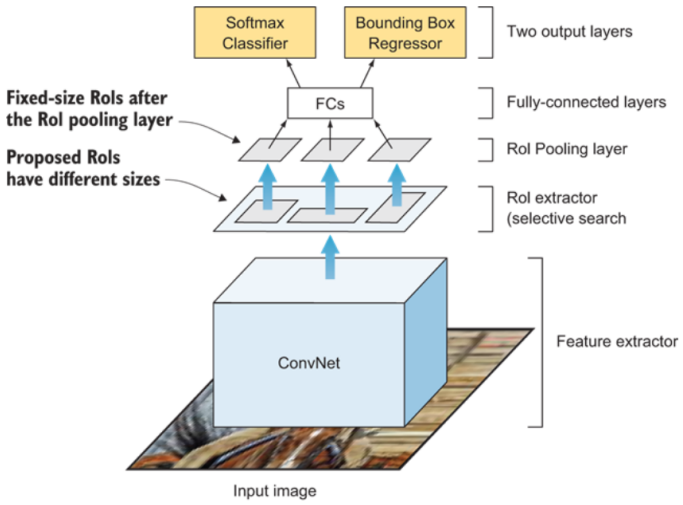


Figure 2.2: Fast R-CNN

testing time, including region proposals slows down the algorithm significantly than the one when compared to not using region proposals. Therefore, region proposals become bottlenecks in Fast R-CNN algorithm affecting its performance.

2.1.2.3 Faster R-CNN Object Detector

Both of the above algorithms (Slow R-CNN Fast R-CNN) uses selective search to find out the region proposals. Selective search is a slow and time-consuming process affecting the performance of the network. Similar to Fast R-CNN, the image is provided as an input to a convolutional network which provides a convolutional feature map. Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is used to predict the region proposals refer Figure 2.3. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.

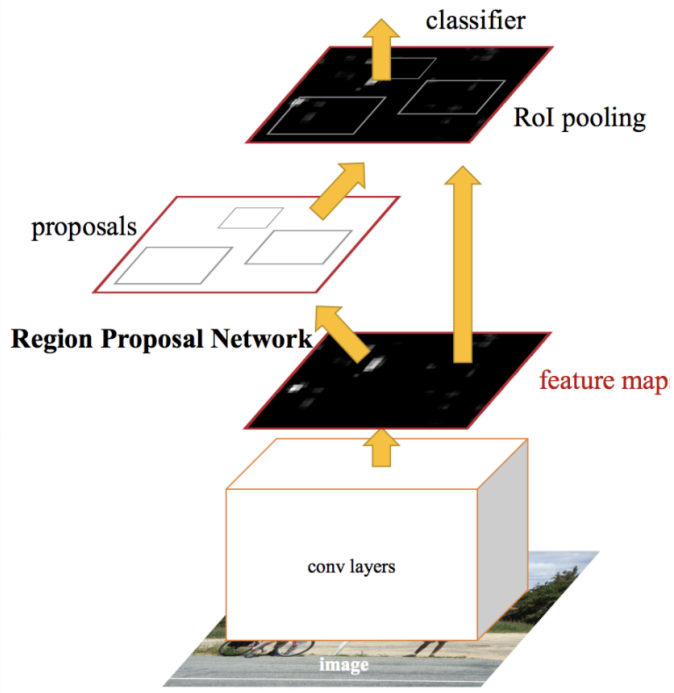


Figure 2.3: Faster R-CNN

2.1.2.4 Faster R-CNN based Action Detection

Inspired by the advances in the field of object detection [36],[37],[38], most recent work approaches the action detection task based on two-stage framework: where in the first stage action proposals are produced by a region proposal algorithm or densely sampled anchors, and in the second stage the proposals are used for action classification and

localization. Compared to object detection in images, spatial-temporal action detection in videos is however a more challenging problem. Unlike images, action detection in videos involve effective temporal modelling to capture all the temporal information to distinguish the class of action. For example activity standing up and activity siting down would require the temporal order to be maintained in the consequent frames. Many recent approaches[1, 39, 40, 41, 42, 19, 43] extended the above two stage framework with optical flow to capture the motion cues and used linking algorithm to connect the frame level detection results to video level. Although these methods have achieved prominent results they are computationally heavy and failed to exploit the temporal property of the videos since the detection's are performed on each frame independently. We surpass this limitation and treat the video as a sequence of frames.

2.1.3 Temporal Shift Module

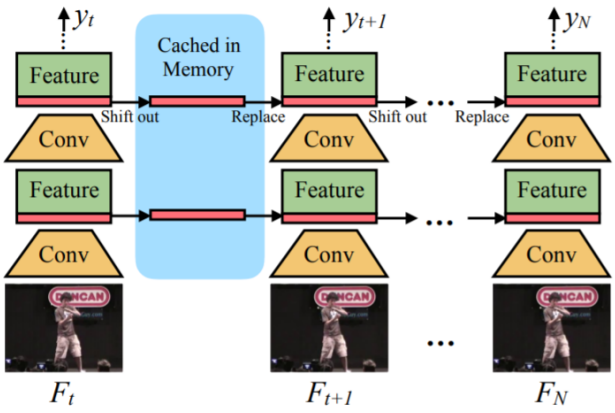


Figure 2.4: Temporal shift across the layers in N frames

The intuition behind TSM [44]: data movement and computation can be separated in a convolution.The whole idea is to shift part of the channels i.e feature maps between neighboring frames in the 8 framed input sample which in a way helps sharing features across frames in the temporal dimension. As in Figure 2.4 part of the features are shifted across the frames after every convolution into the next stage. Despite the zero-computation nature of the shift operation, it introduces two major issues for video understanding:

1. It is not efficient: shift operation is conceptually zero FLOP but incurs data movement. The additional cost of data movement is non-negligible and will result in latency increase. This phenomenon has been exacerbated in the video networks since they usually have a large memory consumption (5D activation).

2. It is not accurate: shifting too many channels in a network will significantly hurt the spatial modeling ability and result in performance degradation. To tackle the problems, we adhere to two main hyper-parameter from the Ablation study of [44].

So in order to overcome those problems:

1. We deploy temporal partial shift strategy: instead of shifting all the channels, we shift only a small portion of the channels for efficient temporal fusion. Such strategy significantly cuts down the data movement cost (Figure 2.4)

2. We insert TSM inside residual branch rather than outside so that the activation of the current frame is preserved, which does not harm the spatial feature learning capability of the 2D CNN backbone.

2.2 Related Work

2.2.1 Activity Recognition using Wearable Sensors

Most prior research focuses on using wearable and mobile devices (e.g., smartphones, smartwatches) for activity recognition . In designing efficient activity recognition systems, researchers have extensively studied various wearable computing research questions. These research efforts have revolved around optimal placement of the wearable sensors [45], automatic detection of the on-body location of the sensor [46], minimization of the sensing energy consumption [47], and optimization of the power consumption [48]. A limitation of activity monitoring using wearable sensors and mobile devices is that these technologies are battery-powered and therefore need to be regularly charged. Failure to charge the battery results in discontinuity of the activity recognition, which in turn may lead to important behavioral events remaining undetected.

2.2.2 Action Recognition in Video data

Action recognition is a long-term research problem and has been studied for decades. Existing State-of-the-Art methods mostly focus on modelling the temporal dependencies in the successive video frames [49, 15, 27]. For instance, [15] directly averaged the motion cues depicted in different temporal segments in order to capture the irregular nature of temporal information. [49] proposed a two-stream network, which takes RGB frames and optical flows as input respectively and fused the detection's from the two streams as the final output. This was done at several granularities of abstraction and achieved great performance. Beyond multi-stream based methods, methods like [27, 12] explored 3D ConvNets on video streams for joint spatio-temporal feature learning on videos. In this way, they avoid calculating the optical flow, keypoints or saliency maps explicitly. However all the above approaches are too large to fit in a real-time edge device. On the other hand [9] uses features calculated from variations in the human keypoints to classify falling and not falling actions, [8] uses VGG16 based on Multi-stream (optical flow, RGB, pose estimation) for human action classification. The above approaches only concentrate on the classification of single human action at scene level and will not perform well if multiple human's are present in an image, which is essential for the healthcare and other public place monitoring systems. Our proposed approach performs human detection and action classification together in a single shot manner where algorithm first localises the human's in an image and classifies his/her action.

2.2.3 Spatio-Temporal Human Action Detection

Spatio-temporal human action detection is a challenging computer vision problem, which involves detecting human actions in a video as well as localizing these actions both spatially and temporally.

2.2.3.1 Object Detector based Action Detection

Generally, 2D action regions are detected in each frame, and are stacked or linked together to generate 3D action volumes [50]. For example, Track Localization [41] tracks current proposals to obtain anchor in next frame by taking advantage of motion cues with optical flow, and selects the best regions in the neighborhood of anchors using a sliding window. However, distinguishing actions from single frame could be ambiguous. To address this issue, ACT [14] takes as input a sequence of frames and outputs tube proposals instead of operating on single frames. Also they uses object detectors like SSD [38] to generate spatio-temporal tubes by deploying high level linking algorithm on frame level detection's and also suffers from the foreground/background imbalance due to single stage training where Two-stage detectors easily handle this imbalance. The RPN narrows down the number of candidate object-locations, filtering out most background instances. Inspired by R-CNN approaches, [1] used Faster-R-CNN [36] to detect the human in an image by capturing the action motion cues with the help of optical flow and classify the final human actions based on the actionness score. [39] extracted proposals by using the selective search method on RGB frames and then applied the original R-CNN on per frame RGB and optical flow data for frame-level action detection's and finally link those detection's using the Viterbi algorithm to generate action tubes.

2.2.3.2 3D CNN

On the other hand Tube CNN[11] uses 3D CNN to generate spatio-temporal tubes by extending 2D Region-of-Interest pooling to 3D Tube-of-Interest (ToI) pooling with 3D convolution. It directly generates tube proposals on each fixed-length clip and then link the clip to represent the entire video. Approaches like [10] is a generalization of capsule network from 2D to 3D, which takes a sequence of video frames as input and the 3D generalization drastically increases the number of capsules in the network, making capsule routing computationally expensive. Also the routing-by-agreement in the network inherently models

the action representations and various action characteristics but pixel-wise localization of actions made the network computationally heavy to be deployed in the real-time embedded edge device. All these methods poses high processing time and computation cost due to the introduction of new dimension in the form of 3D kernels in the 3D CNN related works. As such, the aforementioned methods are unable to be applied in real-time monitoring system.

2.2.3.3 Multi-Stream Convolution Neural Network

Video can naturally be decomposed into spatial and temporal components. The spatial part, in the form of individual frame appearance, carries information about scenes and objects depicted in the video. The temporal part, in the form of motion across the frames, conveys the movement of the observer (the camera) and the objects. So in order to capture the spatial and temporal part , Few Approaches utilised two-stream networks [49, 16, 17, 1] that uses two inputs RGB to capture appearance cues and optical flow to capture the temporal cues into the network with two branched architecture. Firstly, computing Optical Flow from the RGB involves more processing time. Secondly, Two Branched architecture doubles the computation cost leading to heavy model and thus making it not deploy-able on embedded edge devices. Methods like [13] uses saliency map to capture the temporal cues along with Optical Flow to detection fall in the elderly care unit and the computation overhead makes it impossible to be deployed in real-time. Another disadvantages of multi-stream CNN based algorithm is the need of the cloud server where the human identifiable information has to be transferred to the cloud which in turn compromises the privacy of human data.

CHAPTER 3: S-RAD

We introduce S-RAD, an agile and real-time activity monitoring system. Our approach unifies spatio-temporal feature extraction and localization into a single network, allowing the opportunity to be deployed on edge device. This "on-the-edge" deployment eliminates the need for sending sensitive human data to privacy invalidating cloud servers, similar to [7]. Instead our approach can delete all video data after it is processed and can store only the high level activity analytics. Without stored images, S-RAD can be used to solely focus on differentiating between the human actions rather than identifying or describing the human.

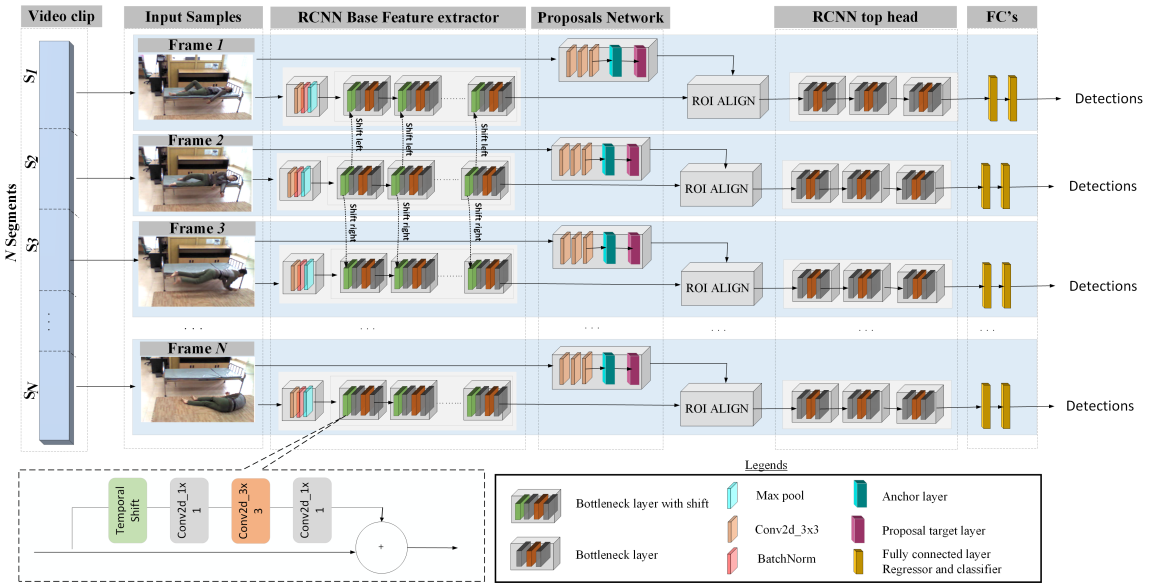


Figure 3.1: Overview of the activity detector. Given a sequence of frames we extract channel shifted convolutional features from the base feature extractor to derive the *activity proposals* in the action proposal network. We then ROI align the activity proposals to predict their scores and regress their co-ordinates.

In order to achieve this privacy preserving edge execution, it is important to have an algorithm able to perform in a resource constrained edge environment. Traditionally such constraints resulted in either accuracy reduction, or increased latency. The overview of S-

RAD is shown in Figure 3.1. S-RAD takes an input sequence of N frames $f_1, f_2, f_3, ..., f_N$ and outputs the detected bounding box and confidence score per each class of the proposals. The model consists of a base feature extractor integrated with temporal shift module (TSM) to capture low level spatio-temporal features. TSM [44] are highly hardware efficient. Temporal shift module are inserted into the bottleneck layer of Resnet-50 [51] of the feature extractor to sustain the spatial information using the identity mapping along with the temporal information using the shifted features. The base feature extractor is made up of the first 40 layers of the original ResNet-50 [51] backbone. The base feature maps are processed by the Proposal Network using a sliding window approach with handpicked anchors and generates action proposals for each frame. An Proposal Network is a fully convolutional network that simultaneously predicts action bounds and actionness scores at each position. The Proposal Network is trained end-to-end to localize and detect valid region action proposals (the foreground) from background. This sliding window approach to generate the proposals is the source of its accuracy as opposed to SSD's [38] rigid grid base proposal generation.

Following the first stage, the original spatio-temporal base features, in conjecture with the proposals are passed into the Region of interest Align (ROI-Align) layer which aligns the varying sized action proposals in to a fixed 7x7 spatial sized action proposals. The second stage of the action detector further classifies each valid action proposals to the action classes in that particular frame. The final classification layer outputs C+1 scores for each action proposal, one per each action class plus one for the background. The regression layer outputs 4 x K where K is the number of action proposals generated in each frame.

S-RAD goes beyond action classification to action detection. This is valuable for communal areas such as mesh halls, and for interactions with other human's and with objects. We chose Faster-R-CNN [36] as our detection baseline due to its fine-grained detection capabilities when compared to SSD [38]. This fine grained detection is especially applicable to the healthcare domain when dealing with wandering patients and fine-grain abnormal

behaviors. Despite the complexity of such tasks our utilization of TSM [44] enables the extraction of the necessary spatio-temporal features for human action localization and individual action classification, in a streaming real-time manner while maintaining privacy.

The components of S-RAD are explained in the following session in the order as given below:

1. **Temporal sampling strategy**: describes the input sampling technique to choose the frames from the video.

2. **Base Feature Extractor**: describes the backbone and feature extraction of S-RAD.

    - Temporal Shift Block: describes the TSM and its implementation in the base feature extractor.

3. **Proposal Network and S-RAD head** : describes the proposal generation network and R-CNN component of S-RAD head.

4. **Formulations** : describes the training loss in details.

3.1 Temporal Sampling Strategy

ConvNets(Convolution Neural Network) in their current forms has their inability in modeling long-range temporal structure. This is probably due to their limited access to temporal context since they are designed to operate only on a single frame or a single stack of frames in a short snippet. However, complex actions, such as sports action, comprise multiple stages spanning over a relatively long time in a video. It would be quite a loss failing to utilize long-range temporal structures in these actions into ConvNet training. To tackle this issue, we used temporal segment network, a video-level sampling strategy as shown in Figure 3.3, to enable to model dynamics throughout the whole video.

3.1.1 Dense Sampling Strategy

Recently there are a few attempts [52, 29] to deal with the problem of capturing long temporal structure. These methods mostly rely on dense temporal sampling as in Fig-

ure 3.2 with a pre-defined sampling interval, which would incur excessive computational cost when applied to long videos. More importantly, the limited memory space available severely limits the duration of video to be modeled. This poses a risk of missing important information for videos longer than the affordable sampling duration. For Example in Figure 3.2 if the action happens at the start,end of the frames in the video, the dense sampling does not capture those cues instead it is a bunch of redundant frames that does not have any useful information in it for the network to learn.

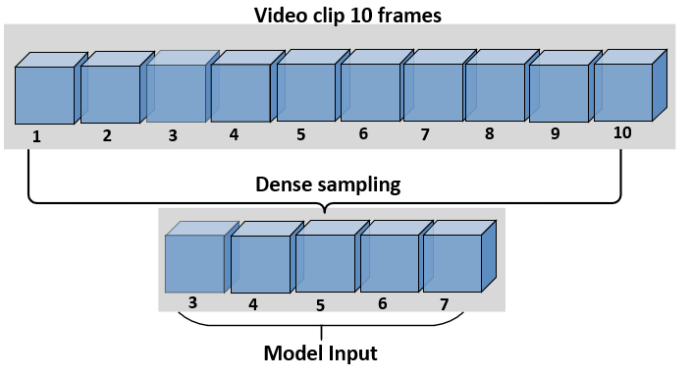


Figure 3.2: Dense Sampling Strategy

3.1.2 Temporal Segment Sampling Strategy

In terms of temporal structure modeling, a key observation is that consecutive frames are highly redundant. Therefore, dense temporal sampling, which usually results in highly similar sampled frames, is unnecessary. Instead a sparse temporal sampling strategy will be more favorable in this case. Motivated by this observation, we used the video-level framework, called temporal segment network (TSN) [15].

The TSN framework first extracts short snippets over a long video sequence with a sparse sampling scheme,where the video is first divided into a fixed number of segments and one snippet is randomly sampled from each segment as in Figure 3.3. By this means, temporal segment networks can model long-range temporal structures over the whole video, in a way that its computational cost is independent of the video duration. Moreover, this sparse sampling strategy preserves relevant information with dramatically lower cost, thus

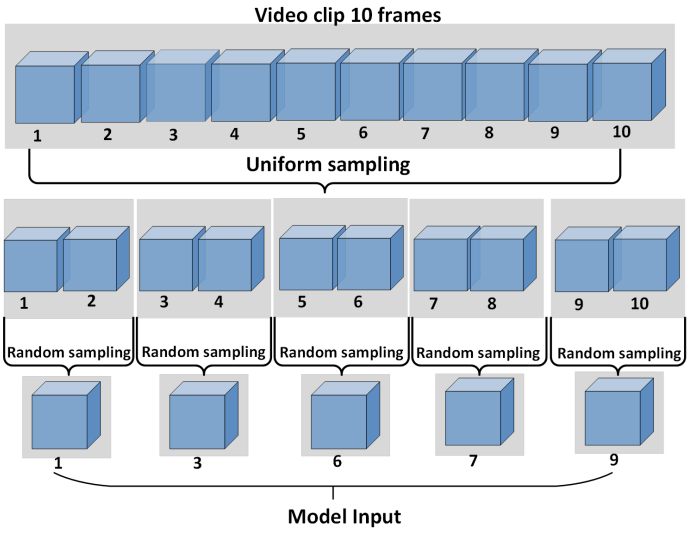


Figure 3.3: Temporal Segment Sampling Strategy

enabling end-to-end learning over long video sequences under a reasonable budget in both time and computing resources. In Figure 3.3 we sample 5 frames from the video in a sparse manner compared to the dense sampling . However the final frames(1,3,6,7,9) captures the entire video sequence rather than choosing the redundant frames. In our S-RAD, we used 8 segments to choose 8 images from the videos and resized the images to 300x400 resolution to have comparison with the State-of-The-Art approaches. Increasing the number of segments to 16 segments didn't improvise the accuracy as 40 layers of Resnet-50 is not deep enough to increase the receptive field of 16 frames, also it introduced additional computation constraint, so for the rest of the work we stick to 8-framed input, as 8-framed input produced comparable accuracy with other State-of-The-Art approaches.

3.2 Base Feature Extractor

Base feature extractor consists of 40 layers of Resnet-50 [51] as the backbone. Resnet-50 alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

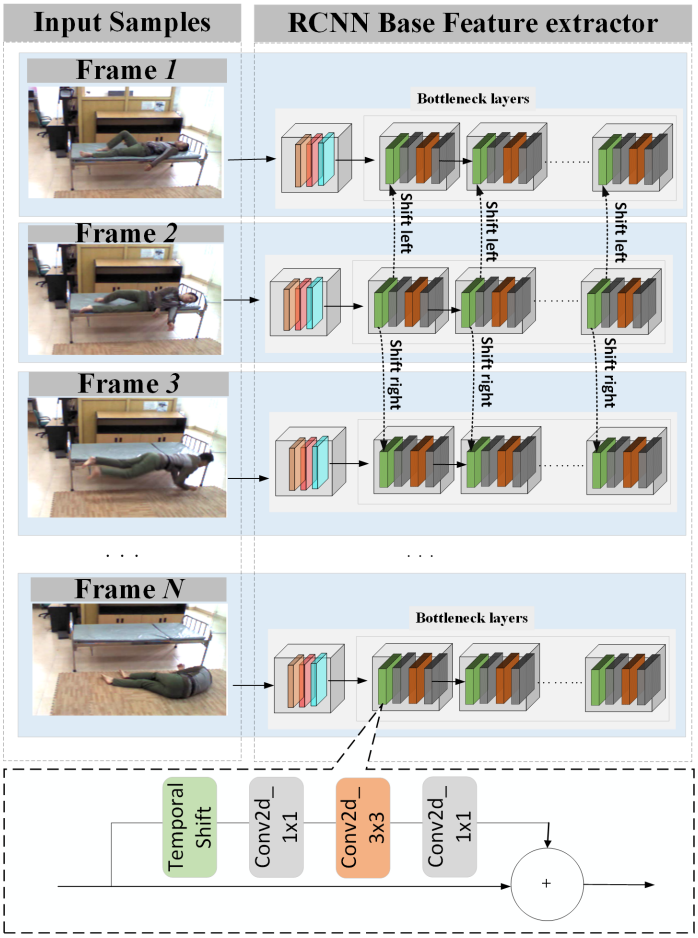


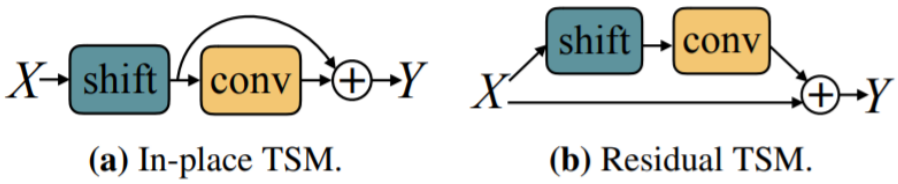
Figure 3.4: Base feature extractor

3.2.1 Base Feature Extractor

ResNet-50 architecture proposed Residual connection, from previous layers to the current one which helps us eradicate the problem of combining spatial and temporal modelling and it will be covered in the following section. Features extracted from the input images are passed into the Proposal Network to generate the Proposal Regions based on the pixel wise location. The Base feature extractor accepts any input resolution making it suitable for multi-scale experiments. Base feature extractor in Figure 3.4 consists of the bottleneck layers with Temporal shift module in the residual connection to capture the motion cues and the output channel of the feature map is matched to align to the Proposal Network.

3.2.2 Temporal Shift Module

Temporal shift block TSM [44] are highly hardware efficient. Temporal shift module are inserted into the bottleneck layer of Resnet-50 [51] based feature extractor to sustain the spatial information using the identity mapping along with the temporal information using the shifted features. We tried two approaches as shown in the Figure 3.5 as they are the hyper-parameters mentioned in the [44] work. The Inplace TSM as in Figure 3.5a affected



(a) In-place TSM. **(b)** Residual TSM.

Figure 3.5: In-place Vs Residual TSM

the spatial modelling capablity and performed worse than the Residual TSM as in Figure 3.5b. Residual TSM in turn captures the temporal modelling without loosing the spatial modelling through identity mapping.

As shown in Figure 3.6, each shift receives the C channels from the previous layer. We shift 1/8th of the channels from the past frame to the current frame and shift 1/8th of the channels from current frame to the future frame, while the other part of the channels remain unchanged. The new features (channels are referred to as features) $\hat{x}_2$, have the information of both the past x_1 and future x_2 frames after the "shift" operation. The features are convoluted and mixed into new spatio-temporal features. The shift block coupled to the next layer will do the same operation. Each shift block increases the temporal receptive field by a magnitude of 2 neighbor frames until N frames. For our work we choose $N = 8$ since features are in the magnitude of 8 in Resnet-50 architecture [51].

3.3 S-RAD Head

S-RAD Head consists of Proposal Network, ROI Align and R-CNN top. The convolutional features with the temporal information shifted and shared among the 8 frames are passed into the Proposal Network to get the proposal regions in the form of 4 box co-

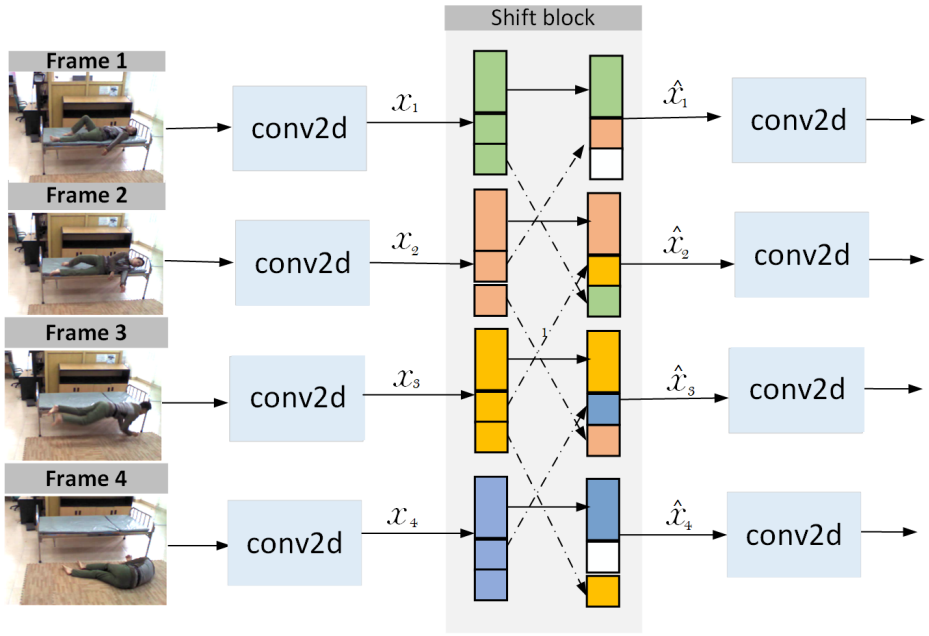


Figure 3.6: Temporal shift block

ordinates. The Proposal Region in turn with the base features are fed in the ROI Align layer to crop features of the proposal in the fixed spatial size 7x7 and is in turn passed into the R-CNN top to convolute for the final classification and regression. Each component is described in detail in below sections.

3.3.1 Proposal Network

A Proposal Network takes an image (of any size) as input and outputs a set of rectangular action proposals, each with an actioness score. Proposal Network is a fully convolutional network [53], which share a common set of convolutional layers. To generate action proposals, a small network is slided over the convolutional feature map output by the last shared convolutional layer. This small network takes as input an n x n spatial window of the input convolutional feature map. Each sliding window is mapped to a lower-dimensional feature (256-d for Resnet-50 and 512-d for VGG [54], with ReLU [55] following). This feature is fed into two sibling fully connected layers regression layer (reg) and classification layer (cls) which classifies foreground from background. At each sliding-window location, we simultaneously predict multiple region proposals, where the number of maximum pos-

sible proposals for each location is denoted as k. So the reg layer has $4k$ outputs encoding the coordinates of k boxes, and the cls layer outputs $2k$ scores that estimate probability of action or no action for each proposal. The k proposals are parameterized relative to k reference boxes, which we call anchors. An anchor is centered at the sliding window in question, and is associated with a scale and aspect ratio. In our approach we use 5 scales and 3 aspect ratios, yielding k = 15 anchors at each sliding position. For a convolutional feature map of a size W x H (typically appr.2,400), there are WxHk anchors in total. Anchors are the hyper parameters and are highly dependent on the dataset. We explored the options handpicked anchors and data driven anchors by k-means clustering algorithm on the training dataset. The hand-picked anchors boosted the accuracy by 2 % in UCF-sport Action dataset, so rest of the approach we stick to the hand-picked anchors.

3.3.2 ROI Align

Region of Interests (ROI) are obtained either by pooling or align operations from the input convolutional features. When Proposal Network return region proposals, all proposals are the offsets for each anchors. Using the offsets, proposed bounding boxes coordinates are obtained which in turn are based on original image size. Region of interest is obtained by cropping the feature map from the predicted bounding box. So, first the given feature map was decreased k times from the original image (via convolutions). It means that each coordinate can be decreased k times. ROI pooling proposes to divide each coordinate by k and take an integer part: [x / k]. To get fixed size output from ROI pooling, cropped feature part is divided into bins. Such kind of division gives n x n grid. And from each bin can be taken maximum or average value. However ROI pooling looses large amount of data due to the above quantisation process. To avoid this we used the better technique ROI Align from [37]. ROI align divides each coordinate by k: x / k and do NOT take integer part. It means that there is no definite pixel in grid that can be taken, because new coordinates are float values. Nevertheless, cropped part is also divided into grid, but for defining concrete values in these bins ROI align choose regularly 4 points in each bin using bilinear

interpolation. And from these 4 points maximum or average value from each bin is taken. Since this proved to have better accuracy than ROI warp or ROI Pooling we chose to use ROI-Align in S-RAD. The cropped feature are reshaped to 7x7 spatial sized feature maps for the S-RAD R-CNN based top.

3.3.3 R-CNN S-RAD Top

The Proposal Network and R-CNN S-RAD top network act as one unified network during training. In each SGD iteration, the forward pass generates region proposals which are treated just like fixed, pre-computed proposals when training a Fast R-CNN detector. The backward propagation takes place as usual, where for the shared layers the backward propagated signals from both the Proposal Network loss and the R-CNN loss are combined. The details about the Loss are described in the following section briefly. The last 10 layers of Resnet-50 are used as R-CNN top and coupled along with a regression and classification layer. Unlike the Proposal network which classifies foreground/background the purpose of these layers is classify the action class like walking,standing from the input ROI-Aligned features. The last 10 layers of Resnet-50 are converted to R-CNN top which cleverly eradicate the need of introducing the computationally heavy "fully connected layers" from being modelled at the end.

3.4 Formulations

3.4.1 Proposal Network Loss

For training Proposal Networks , we assign a binary action class label (of being an action or not i.e foreground vs background) to each anchor. We assign a positive action class label to two kinds of anchors:

1. The anchors with the highest Intersection-over Union (IoU) overlap with a ground-truth box

2. An anchor that has an IoU > 0.7 with any ground-truth box. We assign a negative action class label to a non-positive anchor if it's IoU < 0.3 for all ground-truth boxes

Anchors that are neither positive nor negative do not contribute to the training. With these definitions, our loss function for Proposal Networks is defined as:

$$\mathrm{L}_{rpn}(\{p_i\},\{bb_i\}) = \frac{1}{K}.\sum_{i=1}^{K} L_{cls}(p_i,p_i^*) + \frac{1}{K}.\sum_{i=1}^{K} p_i^* L_{reg}(bb_i,bb_i^*) \qquad (3.1)$$

Here, i is the index of an anchor in a mini-batch and p_i is the predicted probability of anchor i belonging to an action class. The ground-truth label p_i^* is 1 if the anchor is positive, and 0 if the anchor is negative. The vector representing the 4 coordinates of the predicted bounding box is bb_i, and bb_i^* is the ground-truth box associated with a positive anchor. The term p_i^* L_{reg} dictates the smooth L1 regression loss is activated only for positive anchors ($p_i^* = 1$) and is disabled otherwise ($p_i^* = 0$). L_{cls} is log loss(cross-entropy) over two classes (action vs. no action) and is averaged over K frames.

3.4.2 R-CNN Loss

The seconds stage of the detector assigns the action class label to the region of interest or foreground proposals from the RPN training. It involves classification loss and regression loss. The classification layer here includes detecting the correct action class label for the proposals from ROI align layer and regression layer is to regress the detected box with ground truth.

The R-CNN loss is defined as :

$$\mathrm{L}_{rcnn}(\{p_i\},\{bb_i\}) = \frac{1}{K}.\sum_{i=1}^{K} L_{cls}(p_i,p_i^*) + \frac{1}{K}.\sum_{i=1}^{K} L_{reg}(bb_i,bb_i^*) \qquad (3.2)$$

where i is the index of proposals or region of interests with spatial dimension 7x7 and p_i is the predicted probability of the action class label, with p_i^* being the ground truth class label. The vector representing the 4 coordinates of the predicted bounding box is bb_i, and bb_i^* is that of the ground-truth box. L_{cls} is log loss (cross-entropy) over multi-classes, L_{reg} is the smooth L1 regression loss and is averaged over K frames. In training mode we set the network to output 256 proposals and in inference mode network outputs 300 proposals.

3.4.3 Total loss

Total loss of S-RAD is given below and the network is trained together as a single network with Total loss. Total loss is defined as sum of R-CNN and RPN loss:

$$\text{Total\_loss= } \mathrm{L}_{rpn}(\{p_i\}, \{bb_i\}) + L_{rcnn}(\{p_i\}, \{bb_i\}) \tag{3.3}$$

where i is the index of the proposals or region of interest, during training the number is set to 256. p_i is the predicted probability of the action class label and the vector representing the 4 coordinates of the predicted bounding box is bb_i. L_{rpn} is the summation of L_{reg} and L_{cls} from Stage 1 as explained in Section 3.4.1 and L_{rcnn} is the summation of L_{reg} and L_{cls} from Stage 2 from Section 3.4.2 . Total loss is summation of four different loss from Stage-1 and Stage-2. Total loss is propagated back to the network to train all the independent network as single unified network and both the Stage1 and Stage 2 learns together based on this fact.

CHAPTER 4: EXPERIMENTAL RESULTS

4.1 Results on UCF-Sport Action Dataset

The UCF-Sports dataset [56] consists of 150 videos from 10 action classes. All videos have spatio-temporal annotations in the form of frame-level bounding boxes and we follow the same training/testing split used by [39]. On average there are 103 videos in the training dataset and 47 videos in the testing dataset. Videos are truncated to the action and bounding boxes annotations are provided for all frames.

To quantify our results, we report the mean Average Precision (mAP) at the frame level (frame mAP). Frame-level metrics allow us to compare the quality of the detection's independently. We use the Precision-Recall AUC (Area under curve) to calculate the average precision per class. We compute the mean of the average precision per class to see how our algorithm is able to differentiate the features of images between different action classes. We followed the same procedure as in the PASCAL VOC detection challenge [57] to have an apple to apple comparison with the State-of-the-Art approaches in the detection task.

We first evaluate S-RAD on the widely used UCF-Sports dataset. Table 4.1 indicates frame level Average Precision per class for an intersection-over-union threshold of 0.5. Our approach achieves a mean AP of 85.04% .

While obtaining excellent performance on most of the classes, "Walking" is the only action for which the framework fails to detect the humans (40.71% frame-AP). This is possibly due to several factors, the first being that the test videos for "walking" contain multiple actors in close proximity, which results in false detections due to occlusions. Additionally, walking is a very slow action with fine grained features and potentially lacks enough temporal displacement in 8 frames to be picked up by our detector due to sparse temporal sampling strategy. Ultimately, our approach is off by only 2% when compared to

Table 4.1: State-of-the-Art comparison per class frame mAP at IOU threshold 0.5 in UCF-Sports

Action Class	Action tubes[39]	Learning to Track[41]	Multi-region[1]	Tube CNN[11]	S-RAD
Diving	75.79	60.71	96.12	84.37	**99.90**
Golf	69.29	77.54	80.46	90.79	87.20
Kicking	54.60	65.26	73.48	86.48	76.00
Lifting	99.09	100.00	99.17	99.76	**99.96**
Riding	89.59	99.53	97.56	100.0	**99.90**
Run	54.89	52.60	82.37	83.65	**89.79**
Skate Boarding	29.80	47.14	57.43	68.71	67.93
Swing1	88.70	88.87	83.64	65.75	**88.78**
Swing2	74.50	62.85	98.50	99.71	**99.9**
Walk	44.70	64.43	75.98	87.79	40.71

the State-of-the-Art approaches that utilize either multi-modal, 3-dimensional, or complex proposal architecture solutions.The State-of-the-Art comparison in terms of mean Average precision (mAP) is summarised in Table 4.2. The Precision Recall AUC is ploted in Figure

Table 4.2: Overall frame mAP at IOU 0.5 threshold comparison in UCF-Sports Action dataset

	Action tubes[39]	Learning to Track[41]	Multi-region[1]	Tube CNN[11]	ACT[14]	Videocapsulenet[10]]	S-RAD
mAP	68.09	71.90	84.51	86.70	**87.7**	83.9	85.04

4.1 shows the capability of our algorithm to separate different classes.

We also provided the confusion matrix to better understand the detections with the original ground truth in Figure 4.2. The confusion matrix is calculated considering both the detection and classification tasks. Here the grids in the diagonal are the true positive's whose IOU$>$0.8 and the detected action class label match with the ground truth action class label. Other columns are the false positive whose IOU$>$0.8 but the detected action class label does not match the ground truth action class label. The last column contains false negatives with detections with an IOU$<$0.8. We used 0.8 as IOU threshold only in confusion matrix for demo purpose to understand why and where the algorithm is struggling to find the patterns.

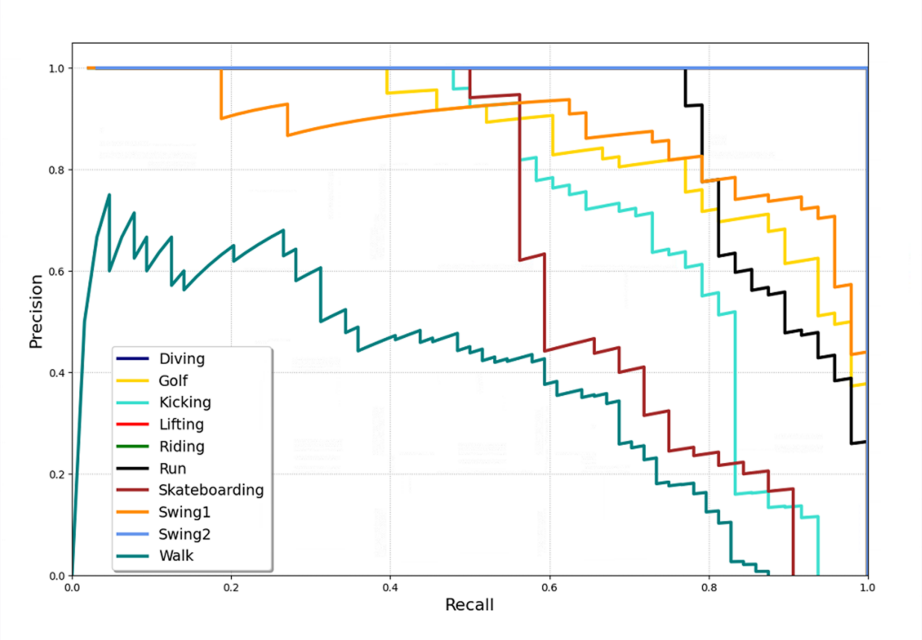


Figure 4.1: Precision-Recall curve per Action class in UCF-Sports

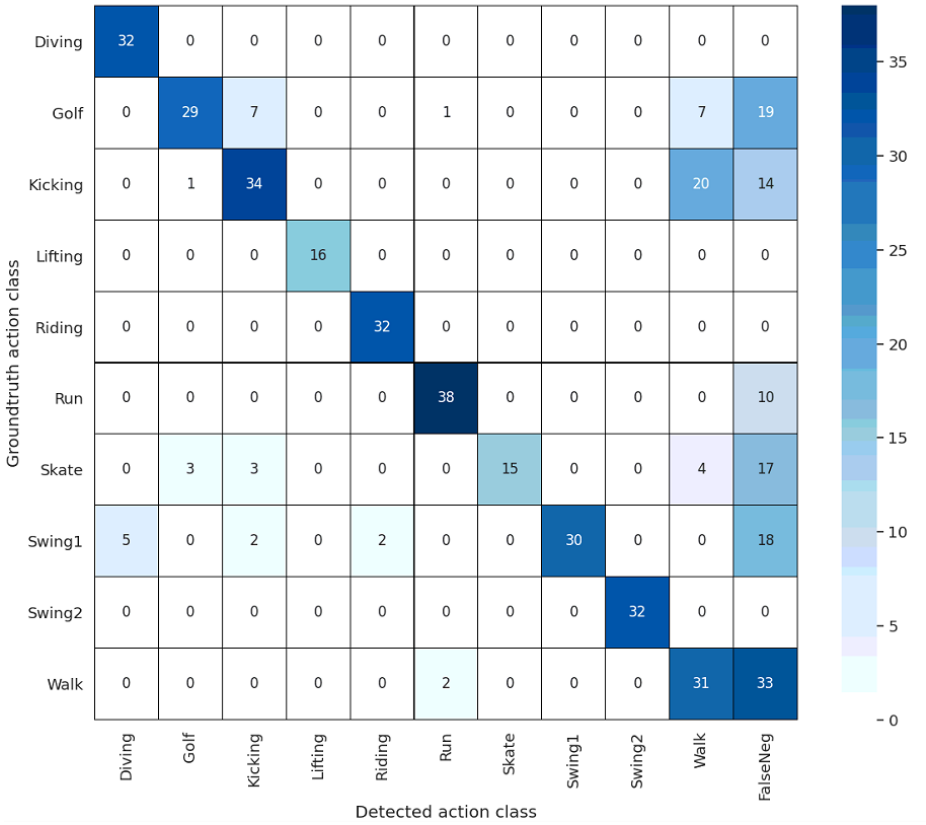


Figure 4.2: Confusion matrix of S-RAD on UCF-Sports

4.2 Results on UR-Fall Dataset

We have also evaluated our framework on the healthcare extensive dataset [58]. The UR-fall dataset is composed of 70 videos: (i) 30 videos of falls; and (ii) 40 videos displaying diverse activities. We used [59] pre-trained only on the person class in the coco dataset to obtain the bounding box annotations for the ground truth. On average there are 56 videos in the training and 14 videos are in the testing dataset.

For the UR-fall dataset we calculate specificity, sensitivity and accuracy along with mAP for comparison.

1. Sensitivity: A metric to evaluate detecting falls. And compute the ratio of trues positives to the number of falls.

$$Sensitivity = \frac{TP}{TP + FN} * 100 \tag{4.1}$$

2. Specificity: A metric to evaluate how much our algorithm detects just "fall" and avoids misclassification with the "not fall" class.

$$Specificity = \frac{TN}{TN + FP} * 100 \tag{4.2}$$

3. Accuracy: Metric to compute how much our algorithm can differ between falls and non-fall videos.

$$Accuracy = \frac{TP + TN}{TN + FP + TP + FN} * 100 \tag{4.3}$$

True positive (TP) means that the frame has a fall and our algorithm has detected fall in those frames. True negative (TN) refers to the frames that donât contain fall and our algorithm does not detect fall in those frames. False negative (FN) designates the frames containing falls, however our algorithm fails to detect the fall in those frames. Finally, false positive (FP) indicates the frames dont contain a fall, yet our algorithm claims to detect a fall. For the sake of comparison with the other classification based State-of-the-Art approaches we take the detection with the highest confidence score from the output

of S-RAD and compare it's class label with the ground truth class label to calculate the above mentioned parameters. Since our approach is based on frame level detection, the classification task on UR-fall dataset is also done in frame level. We achieved a competitive score of 96.54 % in mAP (detection task at frame level). It is important to note, other State-of-the-Art approaches on this dataset relied solely on classification, hence our comparison is concentrated on the classification metrics. The Results are shown on Table 4.3, showing S-RAD's true capabilities in the field of healthcare.

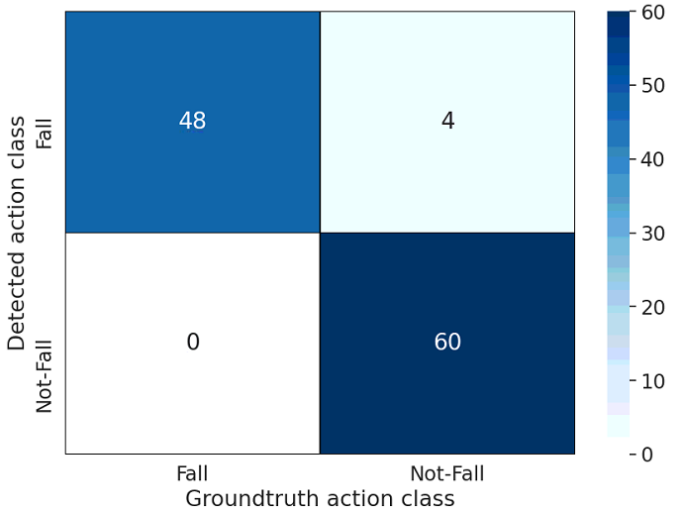


Figure 4.3: Confusion matrix of S-RAD on UR-fall dataset

The confusion matrix on Figure 4.3 shows the ability of the S-RAD to distinguish Fall and Not Fall with only 4 instances being misclassified as Fall.

Table 4.3: State-of-the-Art per frame comparison in UR-fall dataset

	Keypoint Based[9]	**3DCNN Based [12]**	**Multi-stream Based[8]**	**Three-stream Based[13]**	**S-RAD**
Sensitivity	100	-	100	100	**100**
Specificity	95	-	98.61	98.77	**93.75**
Accuracy	97.5	99.27	98.77	98.84	**96.46**

4.3 Real-Time Execution Results

4.3.1 Server Class Execution Results of S-RAD

The S-RAD framework has the advantage of reduced inference time and less number of parameters, enabling us to perform real-time on the edge activity monitoring in a privacy-aware manner. We compare our framework with others in terms of FPS (Frame-Per-Second) and mAP in Table 4.4 on the UCF-Sports Action dataset.

Table 4.4: Comparison on Server Class Execution on Nvidia Titan platform

Approach	Input	Resolution	Param # (M)	FPS	mAP
Multi-stream [1]	RGB+Flow	600x1067	274	11.82	84.51
CapsuleNet[10]	RGB	112x112	103.137	78.41	83.9
TubeCNN[11]	RGB	300x400	245.87	17.391	86.7
ACT[14]	RGB+Flow	300x300	50	12	87.7
S-RAD	**RGB**	**300x400**	**28.35**	**41.64**	**85.04**

We tested our models on one Titan V GPU (except the work of TubeCNN [11], which was reported on a Titan X). The trade-off is between accuracy and inference FPS, as well as parameters. Among the State-of-the-Art approaches, our method has the second fastest run time and can process 41 frames per second which is three times faster than [11] and [1]. Moreover, the number of parameters of our framework is the smallest, about 28.36 M in Table 4.4, although works like [10] have better FPS with their models, their features are too heavy to fit into a real-time edge device, additionally our work maintains a higher mAP at a high resolution when compared to their work. We were unable to provide performance comparisons with the State-of-the-Art approaches on the UR-fall dataset as most of the approaches are not publicly available to run on the edge device, and do not provide performance metrics of their own.

4.3.2 Embedded Edge Execution Results of S-RAD

We additionally evaluated our work on an edge platform, the Nvidia Xavier to test its performance on an resource constrained edge platform. We compare the work of Video-

CapsuleNet [10] with our approach, and despite their initial performance advantage on the Titan V in terms of FPS, our work is the only model capable of running on the memory constrained edge device. S-RAD, as opposed to VideoCapsuleNet folds temporal data into the channel dimension, and as a result avoids introducing another dimension to the tensor sizes. VideoCapsuleNet not only process 3D spatial-temporal feature maps, but they also introduce another dimension of complexity in the form of capsules. We also observed 6.0 FPS with only 5.21W of total SoC (on chip) power consumption and it is summarised in the Table 4.5 where other methods like [10, 1, 11] were not able to fit into the edge devices.

Table 4.5: Comparison on Embedded Edge device platform on Nvidia Xavier

Approach	**Input**	**Resolution**	**Param # (M)**	**FPS**	**Power**	**mAP**
CapsuleNet[10]	RGB	112x112	103.137	-	-	83.9
S-RAD	**RGB**	**300x400**	**28.35**	**6.0**	**5.21**	**85.04**

CHAPTER 5: CONCLUSION and RELATED WORK

5.1 Conclusion

We introduced a novel Single Run Action detector (S-RAD) for activity monitoring. S-RAD provides end-to-end action detection without the use of computationally heavy methods with the ability for real-time execution of embedded edge device. S-RAD is a privacy-preserving approach and inherently protects Personally Identifiable Information (PII). Results on UCF-Sports and UR-Fall dataset presented comparable accuracy to State-of-the-Art approaches with significantly lower model size and computation demand and the ability for real-time execution on edge embedded device.

5.2 Future Work

Our Future work would include introducing the two pathway to utilise the sparse sampling strategy to capture the long temporal structure and dense sampled strategy to capture the short fine grained temporal structure. Also improvement in that area would improvise the fine grained action like "Walking". Different from grid-shaped structures of images/videos, human skeleton, consisting of a series of joints and bones, has an irregular geometric structure. Human action may be regarded as a consecutive dynamic sequence of such irregular structures. Considering the motion complexity of the entire body skeleton, the sophisticated strategy is to separately model the trajectory of each joint or clique of joints through skeletal based information and they are less computationally intensive than the heavy pixel wise RGB based method. Additionally leveraging skeletal action detection just like [60, 61] with graph convolution [62] will reduce the GFLOP of pixel wise calculation of RGB images and also will reduce the appearance noise introduced by RGB images in the scene level detection .

REFERENCES

[1] X. Peng and C. Schmid, "Multi-region two-stream R-CNN for action detection," vol. 9908 of *Lecture Notes in Computer Science*, (Amsterdam, Netherlands), pp. 744–759, Springer, Oct. 2016.

[2] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," *CoRR*, vol. abs/1505.04868, 2015.

[3] P. V. K. Borges and N. Nourani-Vatani, "Vision-based detection of unusual patient activity.," in *HIC*, pp. 16–23, 2011.

[4] B. Ni, C. D. Nguyen, and P. Moulin, "Rgbd-camera based get-up event detection for hospital fall prevention," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1405–1408, IEEE, 2012.

[5] E. Chou, M. Tan, C. Zou, M. Guo, A. Haque, A. Milstein, and L. Fei-Fei, "Privacy-preserving action recognition for smart hospitals using low-resolution depth images," *arXiv preprint arXiv:1811.09950*, 2018.

[6] U. Asif, B. Mashford, S. Von Cavallar, S. Yohanandan, S. Roy, J. Tang, and S. Harrer, "Privacy preserving human fall detection using video data," in *Machine Learning for Health Workshop*, pp. 39–51, 2020.

[7] C. Neff, M. Mendieta, S. Mohan, M. Baharani, S. Rogers, and H. Tabkhi, "Revamp2t: Real-time edge video analytics for multicamera privacy-aware pedestrian tracking," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2591–2602, 2020.

[8] S. A. Cameiro, G. P. da Silva, G. V. Leite, R. Moreno, S. J. F. GuimarÃ£es, and H. Pedrini, "Multi-stream deep convolutional network using high-level features applied to fall detection in video sequences," in *2019 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 293–298, 2019.

[9] A. Y. Alaoui, S. El Fkihi, and R. O. H. Thami, "Fall detection for elderly people using the variation of key points of human skeleton," *IEEE Access*, vol. 7, pp. 154786–154795, 2019.

[10] K. Duarte, Y. S. Rawat, and M. Shah, "Videocapsulenet: A simplified network for action detection," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPSâ18, (Red Hook, NY, USA), p. 7621â7630, Curran Associates Inc., 2018.

[11] R. Hou, C. Chen, and M. Shah, "Tube convolutional neural network (t-cnn) for action detection in videos," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[12] N. Lu, Y. Wu, L. Feng, and J. Song, "Deep learning for fall detection: Three-dimensional cnn combined with lstm on video kinematic data," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 1, pp. 314–323, 2019.

[13] G. Leite, G. Silva, and H. Pedrini, "Fall detection in video sequences based on a three-stream convolutional neural network," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pp. 191–195, 2019.

[14] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid, "Action tubelet detector for spatio-temporal action localization," *CoRR*, vol. abs/1705.01861, 2017.

[15] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool, "Temporal segment networks for action recognition in videos," *CoRR*, vol. abs/1705.02953, 2017.

[16] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *CoRR*, vol. abs/1406.2199, 2014.

[17] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," *CoRR*, vol. abs/1604.06573, 2016.

[18] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," *CoRR*, vol. abs/1812.03982, 2018.

[19] Y. Ye, X. Yang, and Y. Tian, "Discovering spatio-temporal action tubes," *CoRR*, vol. abs/1811.12248, 2018.

[20] S. Ma, J. Zhang, N. Ikizler-Cinbis, and S. Sclaroff, "Action recognition and localization by hierarchical space-time segments," in *2013 IEEE International Conference on Computer Vision*, pp. 2744–2751, 2013.

[21] Y. Ke, R. Sukthankar, and M. Hebert, "Event detection in crowded videos," in *2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8, 2007.

[22] A. Prest, V. Ferrari, and C. Schmid, "Explicit modeling of human-object interactions in realistic videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 835–848, 2013.

[23] M. Jain, J. v. Gemert, H. JÃ©gou, P. Bouthemy, and C. G. M. Snoek, "Action localization with tubelets from motion," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 740–747, 2014.

[24] G. Yu and J. Yuan, "Fast action proposals for human action detection and search," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1302–1311, 2015.

[25] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," *CoRR*, vol. abs/1801.07455, 2018.

[26] A. Diba, V. Sharma, and L. V. Gool, "Deep temporal linear encoding networks," *CoRR*, vol. abs/1611.06678, 2016.

[27] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "C3D: generic features for video analysis," *CoRR*, vol. abs/1412.0767, 2014.

[28] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. C. Russell, "Actionvlad: Learning spatio-temporal aggregation for action classification," *CoRR*, vol. abs/1704.02895, 2017.

[29] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," *CoRR*, vol. abs/1411.4389, 2014.

[30] B. Zhou, A. Andonian, and A. Torralba, "Temporal relational reasoning in videos," *CoRR*, vol. abs/1711.08496, 2017.

[31] M. Zolfaghari, K. Singh, and T. Brox, "ECO: efficient convolutional network for online video understanding," *CoRR*, vol. abs/1804.09066, 2018.

[32] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning for video understanding," *CoRR*, vol. abs/1712.04851, 2017.

[33] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," *CoRR*, vol. abs/1711.11248, 2017.

[34] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013.

[35] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015.

[36] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[37] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017.

[38] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, 2015.

[39] S. Saha, G. Singh, M. Sapienza, P. H. S. Torr, and F. Cuzzolin, "Deep learning for detecting multiple space-time action tubes in videos," *CoRR*, vol. abs/1608.01529, 2016.

[40] "Online real time multiple spatiotemporal action localisation and prediction on a single platform," *CoRR*, vol. abs/1611.08563, 2016. Withdrawn.

[41] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," *CoRR*, vol. abs/1506.01929, 2015.

[42] Z. Yang, J. Gao, and R. Nevatia, "Spatio-temporal action detection with cascade proposal and location anticipation," *CoRR*, vol. abs/1708.00042, 2017.

[43] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid, "Joint learning of object and action detectors," in *ICCV - IEEE International Conference on Computer Vision*, (Venice, Italy), pp. 2001–2010, IEEE, Oct. 2017.

[44] J. Lin, C. Gan, and S. Han, "Temporal shift module for efficient video understanding," *CoRR*, vol. abs/1811.08383, 2018.

[45] L. Atallah, B. Lo, R. King, and G.-Z. Yang, "Sensor positioning for activity recognition using wearable accelerometers," *IEEE transactions on biomedical circuits and systems*, vol. 5, no. 4, pp. 320–329, 2011.

[46] R. Saeedi, J. Purath, K. Venkatasubramanian, and H. Ghasemzadeh, "Toward seamless wearable sensing: Automatic on-body sensor localization for physical activity monitoring," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 5385–5388, IEEE, 2014.

[47] J. Pagan, R. Fallahzadeh, M. Pedram, J. L. Risco-Martin, J. M. Moya, J. L. Ayala, and H. Ghasemzadeh, "Toward ultra-low-power remote health monitoring: An optimal and adaptive compressed sensing framework for activity recognition," *IEEE Transactions on Mobile Computing (TMC)*, vol. 18, no. 3, pp. 658–673, 2018.

[48] S. I. Mirzadeh and H. Ghasemzadeh, "Optimal policy for deployment of machine learning modelson energy-bounded systems," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, 2020.

[49] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *CoRR*, vol. abs/1406.2199, 2014.

[50] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," *CoRR*, vol. abs/1710.03958, 2017.

[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[52] J. Y. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," *CoRR*, vol. abs/1503.08909, 2015.

[53] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *CoRR*, vol. abs/1411.4038, 2014.

[54] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," in *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pp. 730–734, 2015.

[55] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (J. FÃŒrnkranz and T. Joachims, eds.), pp. 807–814, 2010.

[56] K. Soomro and A. R. Zamir, “Action recognition in realistic sports videos,” 2014.

[57] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *Int. J. Comput. Vision*, vol. 88, p. 303â338, June 2010.

[58] B. Kwolek and M. Kepski, “Human fall detection on embedded platform using depth maps and wireless accelerometer,” *Computer methods and programs in biomedicine*, vol. 117 3, pp. 489–501, 2014.

[59] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, “MMDetection: Open mmlab detection toolbox and benchmark,” *arXiv preprint arXiv:1906.07155*, 2019.

[60] C. Li, Z. Cui, W. Zheng, C. Xu, and J. Yang, “Spatio-temporal graph convolution for skeleton based action recognition,” *CoRR*, vol. abs/1802.09834, 2018.

[61] C. Li, Q. Zhong, D. Xie, and S. Pu, “Skeleton-based action recognition with convolutional neural networks,” *CoRR*, vol. abs/1704.07595, 2017.

[62] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *CoRR*, vol. abs/1609.02907, 2016.