Activity Detection in Untrimmed Surveillance Videos

Seunghwan Cha

CMU-RI-TR-21-61

August 05, 2021

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Abstract

Accurately detecting activities in untrimmed videos is a challenging task as systems need to handle variance in object scales, multiple viewpoints, and multiple types of activities. Furthermore, in a real-world scenario, activity detectors are often required to detect novel kinds of activities when the need arises from end-users. To address these issues, we first build an activity classifier on the known activities using detection-based proposals. Then, we propose a retrieval-based solution that utilizes both visual and textual queries for detecting novel types of activities.

For known activity classification, a sequence of object detection, optical flow, and hierarchical clustering are run to obtain spatiotemporal proposals. Then, a multilabel loss is used for optimizing the TSM model. Our trained action classifier demonstrates classification and scene generalization capability by performing competitively on the public MEVA test set and the Known Activity Leaderboards from ActEV Challenge.

For the vision-based retrieval, the penultimate features from the trained TSM are extracted on both query and gallery proposals. The averaged features from the query proposals are compared against the pool of gallery proposals to select the top-ranked proposals as detected activity instances. We also explore a language-based retrieval system that can utilize the textual descriptions of the unseen activities. An image-text model called CLIP is used to extract textual and visual features from the given examples. The same retrieval technique from the vision-based approach is applied for final predictions. Our proposed system ranked 1st place on the Surprise Activity Leaderboard from ActEV Challenge. We hope that the proposed system can help facilitate the successful deployment of activity detection in the real world.

Acknowledgments

I would like to thank my advisors, Deva and Aswin, for their guidance in the past two years. It has been a pleasure working with you on this very challenging problem. You gave me valuable lessons in research and life. I would also like to thank all of my colleagues who have helped me through my Master's Program, especially Krishna, Vijay, Gautam, and Pavel. Thank you for being part of this fruitful journey.

I would also like to thank my thesis committee members: Professor Kris Kitani and Xiaofang Wang for agreeing to take part in my thesis presentation. I also appreciate the discussions with fellow RI members in Smith Hall. You guys gave me a lot of inspiration for my research and career.

Additionally, deep gratitude to our UMD partners who generously provided their codebase for us to develop a working system. Special thanks to Josh, Carlos, and Rama for their insightful feedback during our monthly meeting.

Finally, I wish to send my sincere appreciation to my mother, Sunhwa, and my sister, Seungmi, who gave me emotional support during COVID-19. Thank you for always having faith in me.

Funding

This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/ Interior Business Center (DOI/IBC) contract number D17PC00345. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied of IARPA, DOI/IBC or the U.S. Government.

Contents

List of Figures

List of Tables

Chapter 1

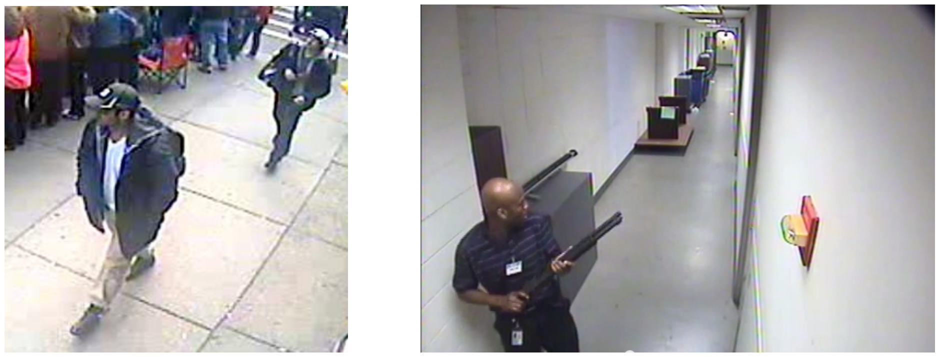
Introduction

April 15, 2013, on the day of the Annual Boston Marathon, Tsarnaev Brothers detonated two homemade pressure-cooker bombs, killing three people and seriously injuring hundreds. Because the site was crowded, it was very difficult to spot the suspect. FBI agents watched several hundreds of hours of security footage near the bombing to pinpoint any suspicious individuals. In the end, as shown in Figure 1.1 (a), FBI agents found two males entering the event with backpacks full but leaving the event without their bags. Six months later, a 34-year-old male named Aaron Alexis walked into the Washington Navy Base with his shotgun and massacred 12 people. Even if we had an automatic computer vision system that can detect guns, it would not have been flagged as dangerous as the shooting occurred in a military base where carrying a shotgun isn't an abnormal situation.

These two disasters motivated the US government to develop a system that can automatically understand human activities from static cameras. MEVA is a human activity detection dataset from surveillance cameras.[1] The dataset contains a total of 9,300 hours of continuous, untrimmed videos and annotations of 37 types of activities on 144 hours videos [5]. As shown in Figure 1.2, MEVA was collected at a government facility with more than 100 actors performing scripted activities. The data contains many different realistic scenes including bus station, gym, cafeteria, park, and etc.

MEVA dataset is very challenging as it contains multiple actors, viewpoints, and activities in a single frame. Hence, developing an automatic activity detection

[1] https://mevadata.org/



(a) Tsarnaev Brothers entering the Boston marathon site

(b) Aaron Alexis just before Washington Navy Base mass shooting

Figure 1.1: **Security camera footage of two incidents that happened in 2013.** Suspects of both incidents were spotted in the CCTV just before the disastrous massacres.

system requires several state-of-the-art modules to work simultaneously to localize the activities of interest. Furthermore, we are also required to detect novel activities that were not seen during training using only a few examples when the need arises. It is essential that our system can accept a new set of activities and still successfully localize those activities. Therefore, we develop a retrieval-based system that utilizes both the visual and textual examples of the given novel activities.

We lay our foundation in activity understanding by looking at various benchmarks in Chapter 2. In Chapter 3, we explain the detailed pipeline of the proposal generation, network architecture, and loss functions to train our activity classification network. We further describe the methodology of retrieving cuboids that contain novel activities by using exemplar video clips and corresponding textual descriptions in Chapter 4. Finally, we introduce post-processing techniques on the retrieved cuboids to achieve better performance on the evaluation metric. Chapter 5 introduces the empirical observations of different methods of training the backbone. In addition, the qualitative and quantitative results of our visual and textual retrieval systems are discussed.

Our system ranked 1st place in P_{miss} and 2nd place in terms of $nAUDC$ metric on 2021 Unknown Facility Surprise Activity ActEV Sequestered Data Leaderboard (SDL).[2] The system completes all the required tasks in real-time and does not require

[2]https://actev.nist.gov/sdl#tab_activities

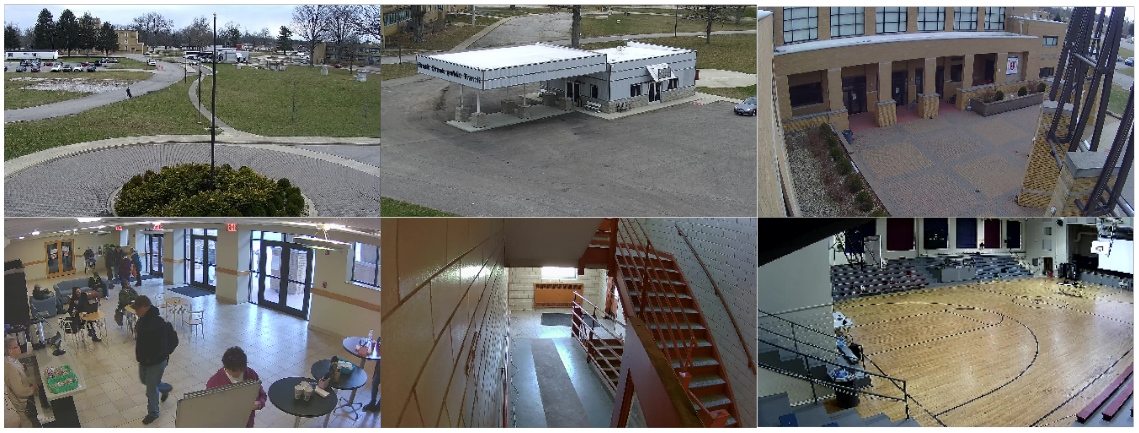


Figure 1.2: **Example scenes in the MEVA dataset.** Approximately 100 actors were hired to perform different types of activity at the Muscatatuck Urban Training Center (MUTC) for three weeks resulting in a total of 9,300 hours of video. MEVA dataset contains various viewpoints with 9 indoor and 13 outdoor scenes. Each video is approximately 5 minutes long with several videos not having any relevant activity for the entire duration of the video.

any additional finetuning on the set of novel activities. We hope that our system can be deployed in a real-world scenario to help improve activity understanding in the wild.

Chapter 2

Related Work

2.1 Video Representation Learning

Video representation learning has been limited to building hand-crafted features to encode motion information [42] before large video datasets like Kinetics [16] and computational resources became widely available. Initial deep learning methods tried to use both RGB and optical flow to embed temporal information through two stream networks [3, 36, 43] but many subsequent architectures attempted adopting different versions of 3D convolutions to better incorporate motion understanding from only RGB input [40]. SlowFast [9] tried to better mimic the human brain's way of recognizing activities through implementing two pathways network that accepts clips with different frame rates. TSM [18] is an efficient yet effective adaptation of TSN [43] through shuffling 2D channel-wise features across time.

Motivated by the success of Transformer architecture in NLP [41], many works looked into ways to replace attention with convolution completely in image recognition [6]. More recently, researchers have built upon ViT which is an image-based Transformer that splits images into 16x16 patches to serve as tokens [6] and have applied a similar network architecture for video representation learning [1, 2]. However, these networks require much larger data and complex data augmentation in order to achieve similar performance with the CNN counterpart. Hence, we have adopted TSM [18] as our major video backbone network.

2.2 Activity Detection

Similar to how advances in image recognition led to improvement in localization tasks like object detection, advances in activity recognition, especially works on Kinetics [16] helped advances in activity detection. There are different types of activity detection datasets that have different objectives. UCF-101 [37] requires models to predict both spatial and temporal boundaries of activities. Due to its small size, the dataset quickly became saturated and the best performing model now achieves almost perfect accuracy [11].

There are datasets that concentrate on localizing the temporal duration of activities [7, 15]. Videos from these datasets are long and often untrimmed so directly building an end-to-end solution is not feasible. As a result, many works tried to build extra layers on top of the precomputed features from action recognition backbones to accurately detect the start and end timestamp of each activity [19, 20]. AVA [12] focuses more on the spatial localization of actors and objects involved in movies and most models on this benchmark directly apply models trained on Kinetics [9, 44] with an object detection module like Faster RCNN [33].

VIRAT [30] and MEVA [5] are surveillance activity detection datasets that have fixed camera viewpoints and long duration of videos. These datasets are very different from the aforementioned datasets for three reasons: 1) Kinetics and UCF-101 are crawled from web sources and are often artificially trimmed, 2) THUMOS14 and ActivityNet have less than 5 activities happening in each clip while MEVA can have more than 100 activities happening in a 5-minute clip, and 3) surveillance videos only have limited number of available scenes so deep learning model quickly becomes overfitted to the distribution of the training data. Therefore, works that tried to solve this task employ various different modules rather than end-to-end solutions [10, 22, 34]. Argus [22] uses the detect-and-track method to generate initial proposals and filter them through a heuristic-based mechanism. Gabriella [34] applies action segmentation module using I3D-based encoder-decoder structure to directly localize objects that are performing actions of interest. TRI-I3D [10] uses object detection and clustering to generate a large pool of proposals and feeds to the Flow I3D network with an additional temporal regression layer at the end. Our work builds upon the work of TRI-I3D for generating proposals but our focus was mainly on detecting novel activities using retrieval rather than detecting known activities from TRI-I3D classification. As a result, we did not make use of the two linear layers in TRI-I3D: softmax classification and temporal regression layers.

2.3 Vision-Language Learning

With the recent success of GPT-2 [31] in language modeling, learning visual and textual representation jointly from a large dataset has gained a lot of attention and some works tried to directly apply the Transformer-based model for joint learning [38]. However, due to the difficulty of collecting annotated video-text data, multimodal models were not able to scale nicely on small datasets like MSR-VTT [45]. HowTo100M [25] is the first large video-text dataset that was collected using YouTube instructional videos. Instead of annotating the texts manually, the dataset utilized automatically generated captions as weak signals for learning joint embeddings. In particular, MIL-NCE [26] investigates Multiple Instance Loss from noisy instructional videos to update weights of video and text encoders jointly. Finally, CLIP was a first attempt to apply Transformer to both the image and text encoders by training in a self-supervised manner on a large web-crawled joint dataset. Directly applying weights of CLIP on video retrieval tasks in a zero-shot manner has proven to be successful [8, 23] and outperformed many targeted network architecture designs. We follow this line of research and try to apply CLIP directly for retrieving unseen activities from untrimmed videos.

Chapter 3

Action Classification

In order to retrieve novel types of activities with only a few given exemplars, it is essential to train the activity backbone network well so that the learned representation outputs features that contain useful information. As a result, we implemented a training pipeline that accepts a sequence of untrimmed videos as input and generates activity detection results. Our pipeline which is shown in Figure 3.1 contains three main modules: 1) proposal generation using objection detection and hierarchical clustering (Section 3.1, 3.2, and 3.3), 2) optical flow and RGB frame extractions (Section 3.4), and 3) Temporal Shift Module (TSM) Network and Binary Cross Entropy Loss (BCE) for activity classification of the proposals (Section 3.5). Our proposal generation module is from the baseline work [10]. The details of each module will be discussed below.

Algorithm 1 Proposal Generation

```
1: procedure GENERATE_PROPOSALS(video, gt)
2:     detections ← $Mask\_RCNN$(video)
3:     props ← $hierarchical\_clustering$(detections)
4:     final_props ← $assign\_activities$(props, gt)
5:     return final_props
6: end procedure
```

3.1 Object Detection

We first run an object detector called Mask-RCNN with Feature Pyramid Network (FPN) [14] that was trained on MSCOCO dataset [21]. Mask-RCNN is an improved version of two stage object detector, Faster-RCNN [33] with a separate instance segmentation head and is one of the most widely used object detectors in the domain. We experimented with more advanced networks such as Hybrid Task Cascade [4] but considering the trade-off between speed and accuracy, we chose Mask-RCNN R101 as our detector. In addition, we ran the detector on every 30th frame to reduce the computations and found that such sampling rate doesn't hurt the overall performance significantly. Finally, we only stored bounding boxes that were predicted as either *person* or *vehicle* (COCO ids: *car, truck, bus, train*).[1]

3.2 Hierarchical Clustering

We believe that having high recall is more important than high precision as our main goal is to detect given activities in a pool of videos. As a result, we use hierarchical, agglomerative clustering [27] to generate clusters. We feed (x, y, t), 3-dimensional features with x and y corresponding to the center coordinate of the bounding box and

[1]Implementation from https://github.com/roytseng-tw/Detectron.pytorch

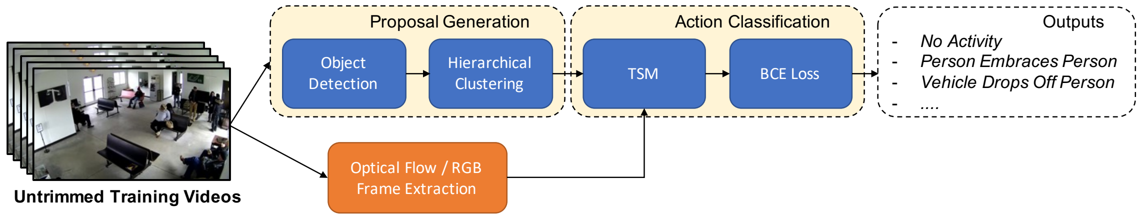


Figure 3.1: **Overview of training our backbone representation.** Given a series of untrimmed videos, we first run object detection on frames using Mask-RCNN. Then, hierarchical clustering is used to generate a pool of different proposals that sufficiently overlap with the ground truth. Optical flow and RGB frames are concurrently extracted and further pre-processed based on the generated cuboids. These inputs are fed to the TSM model to output 38 different known activities including *No_Activity*. BCE loss is used to update the weights of the network.

Algorithm 2 Temporal Jittering

```
1: procedure JITTER_PROPOSALS(prop, vid_length)
2:     anchor, frame_stride, padding ← $45, 60, 450$
3:     start ← max(0, prop.start - padding)
4:     end ← min(vid_length, prop.end + padding)
5:     Initialize new_props
6:     for t in range(start, end, frame_stride) do
7:         new_p.start ← t - anchor + 1
8:         new_p.end ← t + anchor
9:         Append new_p to new_props
10:    end for
11:    return new_props
12: end procedure
```

t corresponding to the frame number of the detection. Then, the generated linkage trees are dynamically split at different levels to create k resulting clusters. To reduce the computational burden, if there are more than 44,000 detections in a single video, we randomly sub-sample 44,000 *person* and *vehicle* detections. The output of the stage is a list of cuboids which are represented as $(x_{min}, y_{min}, x_{max}, y_{max}, t_{start}, t_{end})$.

3.3 Proposal Generation and Temporal Jittering

After clustering the detections, we can generate proposals necessary for training activity classifier backbone. The overview of our proposal generation module is demonstrated in Algorithm 1.

We split the proposals into multiple overlapping smaller chunks as shown in Algorithm 2. The reasoning behind this step is closely related to the evaluation protocol of P_{miss}. By temporally jittering proposals with a lot of overlaps, we reduce the probability of missing the ground truths (further details in Section 5.1.2). Given a single proposal, we start generating proposals of 3 seconds (90 frames) length at every 2 seconds resulting in smaller chunks that overlap at least 1 second.

The next step in this pipeline is assigning activity labels to jittered cluster proposals by comparing with the ground truth annotations. As shown in Algorithm 3, we compute the spatial and temporal Intersection over Union (IoU) between the proposals and ground truth cuboids. If spatial IoU is larger than 0.3 and temporal

Algorithm 3 Activity Assignment for Generated Proposals

```
procedure ASSIGN_ACTIVITIES(props, gt)
    Initialize final_props
    spatial_thres, pos_thres, neg_thres $\leftarrow 0.3, 0.5, 0.15$
    for p in props do
        p $\leftarrow$ $jitter\_proposal$(p)
        temp_ious, acts $\leftarrow$ $iou$(p, gt)
        for temp_iou, act in zip(temp_ious, acts do
            if temp_iou $\geq$ pos_thres then
                p.act_type $\leftarrow$ act.act_type
                Append p to final_props
            else
                p.act_type $\leftarrow$ 0
                Append p to final_props
            end if
        end for
    end for
    return final_props
end procedure
```

IoU is larger than 0.5, we consider the proposal as true positive and assign the corresponding activity label of the given ground truth activity. IoU calculation can be generalized as following:

$$sIoU = \frac{|R_{xy} \cap S_{xy}|}{|R_{xy} \cap S_{xy}|}$$

$$tIoU = \frac{|R_t \cap S_t|}{min(|R_t|, |S_t|)}$$

R refers to the reference or ground truth activity and S refers to the system or proposal activity. For temporal IoU, we only consider the overlap divide by the duration of either reference or system activity. Note that multiple different ground truth cuboids can overlap with a single proposal which is quite common in the MEVA dataset with activities such as *person_exits_through_structure* and *person_opens_facility_door* that are happening at the almost same spatial and temporal location.

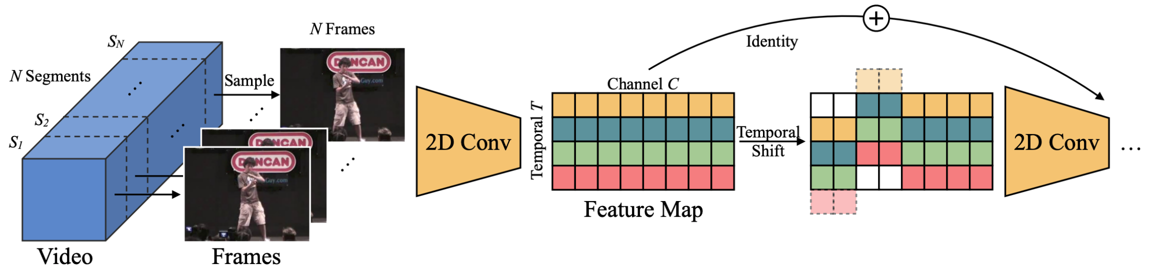


Figure 3.2: **Architecture of TSM.** TSM learns spatiotemporal information through 2D convolutions instead of conventional 3D convolutions by shuffling the features from timestamps T-n to T+n. Due to the efficiency of shifting operation in GPUs, TSM can achieve fast inference speed without experiencing a drop in performance.

3.4 Optical Flow and Frame Extraction

Now that we have spatial and temporal information of the generated cuboids, we need to obtain the raw data necessary for pre-processing. In order to do so, we extract RGB frames from the raw *.avi* MEVA videos using *FFmpeg* [39] with the highest image quality possible.

In addition, we make use of GPU accelerated optical flow extraction from *OpenCV* for fast computation. Specifically, TV-L1 method [46] was applied to every 4th frame in the video and each frame was reduced to 75% of its resolution while maintaining the original aspect ratio to improve the speed of computation. Note that either RGB frame or optical flow extraction was run depending on the input modality of the TSM model and multiple videos were processed concurrently to speed up the process. Finally, for fast data loading, we store the pre-processd frames to RAM prior to training by cropping the cuboids from the frame and resizing the cropped images to (144, 144).

3.5 Activity Classifier

We use TSM [18] that is widely used for activity recognition. As shown in Figure 3.2, TSM is based on a TSN backbone that segments a video into non-overlapping chunks [43]. For each chunk, 2D ResNet is run on every frame and the final prediction is made using a majority vote. Although TSN is very efficient as it does not incorporate computationally heavy 3D convolutions that are common in activity recognition, it is not good at handling temporal information essential for video recognition. TSM is a novel architecture that can embed motion understanding into a 2D network through shifting nearby feature maps mimicking temporal convolutions. Compared to 3D convolution that requires much more parameters and FLOPs, the shifting operation can be implemented efficiently in GPUs and does not require any additional parameters. Hence, we initialize the weights from TSM ResNet50 pretrained on Kinetics-400 dataset [16].

For training, we initially used standard categorical cross entropy loss with softmax activation layer with $f(s_i) = \frac{e^{s_i}}{\sum_{j=0}^{C+1} e^{s_j}}$ where t_i is the ground truth and s_i is the score from the last linear layer of TSM.

$$CE = -\sum_{i=0}^{C+1} t_i log(f(s_i))$$

However, we realized the limitations of using multi-class setting and switched to using sum of multi-label binary cross entropy loss with sigmoid activation layer with $f(s_i) = \frac{1}{1+e^{-s_i}}$.

$$BCE = -\sum_{i=0}^{C+1} (t_i log(f(s_i)) - (1 - t_i) log(1 - f(s_i)))$$

BCE Loss with sigmoid activation allows a single proposal to be assigned to multiple different activities if spatial IoU and temporal IoU are above a certain threshold. With a multiclass set-up, we only assigned one of the ground truth activities so that each proposal only had one activity label.

For training, class balanced sampling is used so that 50% of instances in every batch are negatives (*No_Activity*) and each positive activity label appears approximately uniformly. We train for 6 epochs using ADAM optimizer [17] with an initial learning rate of $5e-4$. The batch size was set to 200 per GPU and 10 2080Ti NVIDIA GPUs were used for training. We adopted PyTorch implementation of TSM.[2]

[2]Implementation from https://github.com/mit-han-lab/temporal-shift-module

Chapter 4

Action Retrieval

Figure 4.1 illustrates the overall pipeline of our visual retrieval. The ActEV Unknown Facility Surprise Activity (UF SA) Leaderboard provides several (exact number is unknown) example clips that are spatially and temporally cropped and the corresponding text descriptions of each novel activity that are not part of 37 known activities. In addition, we are not given any information on what the set of novel activities are as the submission system masks out the activity names to be meaningless like *act51*. As a result, we believe that applying few-shot learning methods is not applicable as we are not even sure of how many training instances we have for each activity. In conclusion, we believe that providing a few exemplars and asking the system to look for similar instances in the pool of untrimmed video is more practical in a real-world scenario.

4.1 Visual Exemplar based Retrieval

Initially, we run the same proposal generation that we used for training our backbone network on the example clips which are spatially and temporally trimmed to only include the given novel activity. By running proposal generation on these clips, we are augmenting the number of generated query proposals. In addition to generated cluster proposals, we also include the ground truth cuboids in case the detection missed the object of interest due to low resolution. Then, we pass the cuboids into the TSM model that was trained on KF MEVA data and obtain its penultimate features with a vector size of $(\#of cuboids, 2048)$. Since there are multiple proposals in each clip and multiple clips for each activity, we average the features so that we get a single feature vector denoted as the query descriptor, d_Q^{id} for each activity.

Once we extract features for all the novel activities, we run a similar pipeline for the test videos. Only difference is that we do not go through the ground truth assignment stage as we do not have any ground truth annotations present for these videos. The extracted features in this stage are denoted as the gallery descriptors, d_G. Finally, we run cosine similarity score between d_Q^{id} and d_G by computing $\frac{d_Q^{id} \cdot d_G}{||d_Q^{id}||||d_G||}$ for every activity id. We rank the gallery proposals based on the similarity score and output the top 10,000 proposals as the given activity id. Finally, we run post-processing steps which will be explained in Section 4.2. Details of retrieval can be found in Algorithm 4.

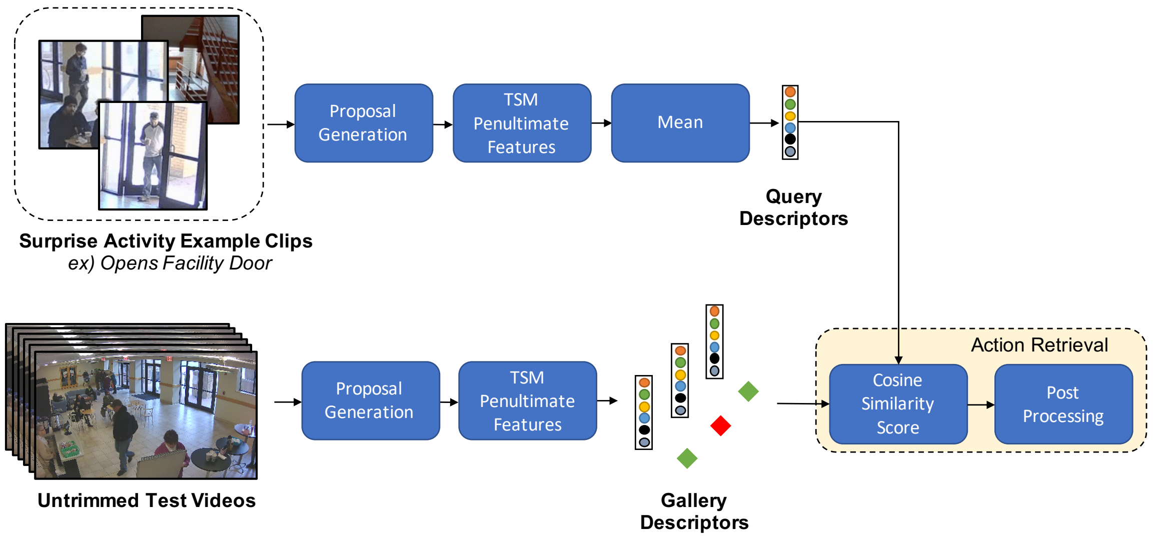


Figure 4.1: **Overview of our visual retrieval system.** Given several example clips of the novel activity, we extract a query descriptor from the penultimate layer of the TSM trained on the MEVA dataset. We also run the same pipeline that we used for training on the pool of untrimmed test videos to get gallery descriptors to calculate the cosine similarity scores with the query descriptor. The top 10,000 ranked proposals are selected and filtered through post-processing.

Algorithm 4 Retrieval using Cosine Similarity

```
1:  procedure RUN_RETRIEVAL($d_Q$, $d_G$, props, act_ids)
2:      Initialize final_acts & final_scores
3:      k ← $10,000$
4:      for $id$ in act_ids do
5:          $d_Q^{id} \leftarrow mean(d_Q^{id})$
6:          sims ← $cosine\_similarity(d_Q^{id}, d_G)$
7:          indices ← $argsort$(sims)
8:          acts, scores ← props[indices], sims[indices]
9:          acts.act_type ← $id$
10:         acts ← $shorten\_activity$(acts)
11:         acts, scores ← $3d\_nms$(acts, scores)
12:         Append acts[:k] and scores[:k] to final_acts and final_scores
13:     end for
14:     return final_acts and final_scores
15: end procedure
```

4.2 Post-Processing

We apply two different post-processing methods before generating the final system predictions. 1) We shorten each proposal to be 2 seconds long from the original 3 seconds chunks and 2) we run 3D Non-Maximum Suppression (NMS) [29] to prune overlapping action proposals that were predicted as the same class. These steps can help improve our performance by preventing our system from getting penalized on T_{FA} for duplicate and overlapping predictions (more details in Section 5.1.2).

4.3 Textual Exemplar based Retrieval

Although we can already build a great activity detector by only utilizing the visual exemplars, we are not fully taking advantage of the information that is available to us. By developing a system that can retrieve novel activities with only textual descriptions, we can further improve the performance of our retrieval pipeline and make our system more deployable to the real world. As shown in Figure 4.2, we employ the similar overall pipeline for retrieval but instead make use of the state-of-the-art multimodal model called Contrastive Language-Image Pretraining (CLIP) [32]. The

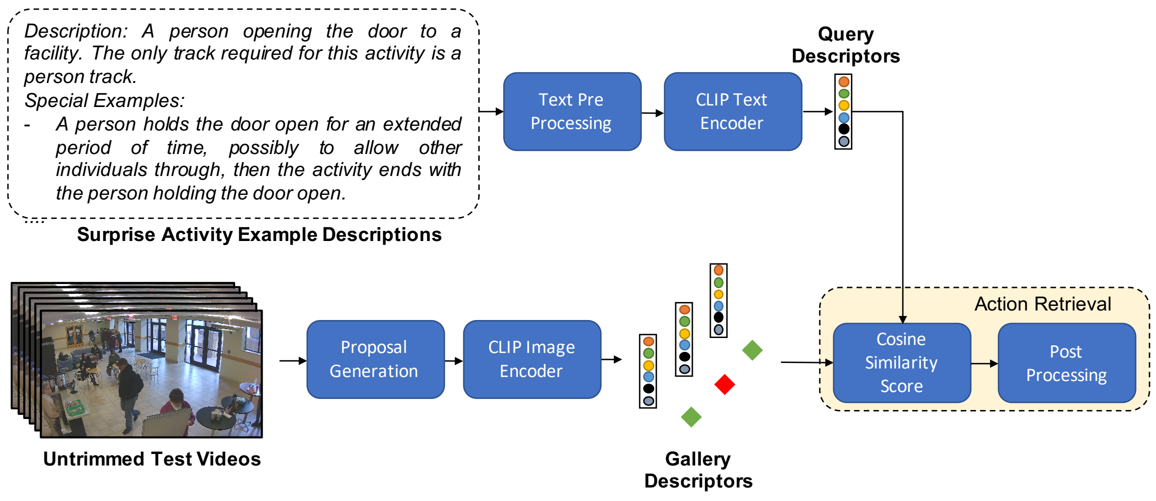

Description: A person opening the door to a facility. The only track required for this activity is a person track.
Special Examples:
- A person holds the door open for an extended period of time, possibly to allow other individuals through, then the activity ends with the person holding the door open.
Surprise Activity Example Descriptions
Text Pre Processing
CLIP Text Encoder
Query Descriptors
Untrimmed Test Videos
Proposal Generation
CLIP Image Encoder
Gallery Descriptors
Action Retrieval
Cosine Similarity Score
Post Processing

Figure 4.2: **Overview of our textual retrieval system.** Instead of utilizing the example clips of the novel activity, we use text descriptions that outline the specific guidelines for annotating the activities. We tokenize the sentences and feed them to CLIP text encoder to obtain a query descriptor. Gallery descriptors are generated by running CLIP image encoder on every frame in cuboids.

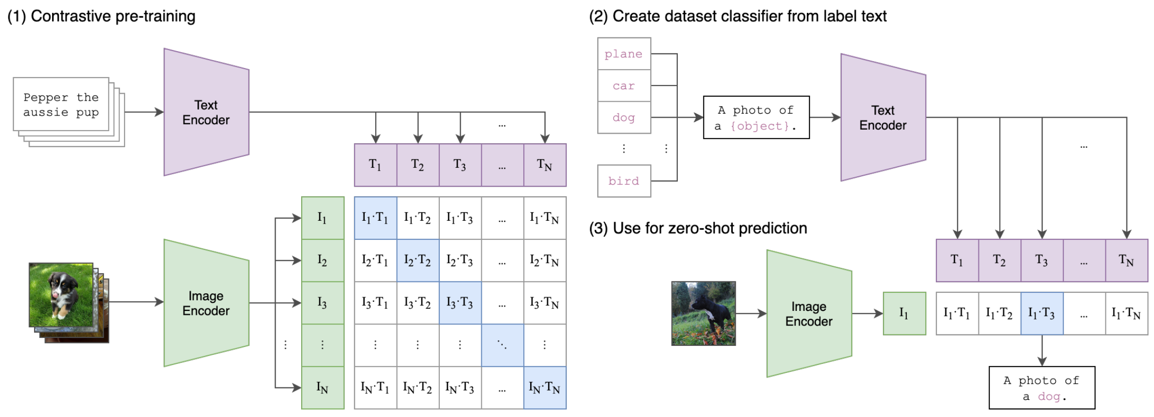


Figure 4.3: **Architecture of CLIP.** CLIP is a multimodal architecture where a set of images are encoded through ViT and the corresponding class names are encoded through language Transformers. Both encoders are pretrained in self-supervised manner on a very large image-text dataset crawled from the web. At test times, cosine similarity between the output of the image encoder and that of the text encoder is calculated for zero-shot predictions.

details of the CLIP architecture are displayed in Figure 4.3. CLIP has an image encoder that is based on either ViT [6] or ResNet [13] and a text encoder that is based on GPT-2 [31]. CLIP was trained on a large web crawled dataset that contains 400 million pairs of images and texts in a self-supervised manner. Given a batch of N image and text pairs, the task of CLIP during training is to correctly identify N correct image-text pairs by maximizing the cosine similarity between the image and the text embeddings and minimizing the cosine similarity of embeddings from $N^2 - N$ incorrect pairs. CLIP shows amazing generalization ability by performing well on various vision datasets including ImageNet and Kinetics through zero-shot linear probing. As a result, we use CLIP for our text-based retrieval experiments.

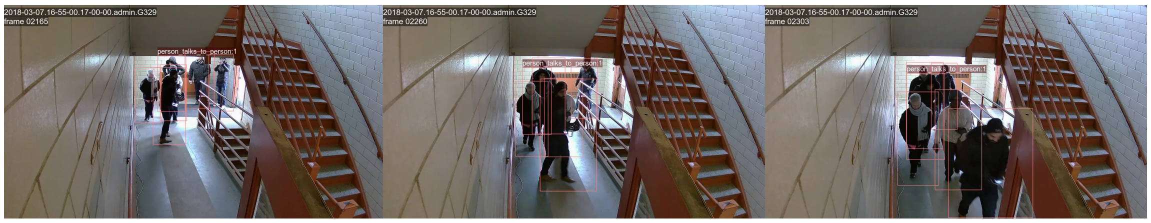
We pre-process each sentence in the activity descriptions by tokenizing them through lower-cased byte pair encoding (BPE) [35]. We pass the tokenized inputs to CLIP text encoder and average them so that we get a single query descriptor d_Q^{id} for each activity. For test videos, we run CLIP image encoder on the generated proposals to obtain the gallery descriptors d_G.[1] Since CLIP image encoder only accepts image input, we pass every frame separately as image inputs and average the features across all frames. We have found that ViT/B32 shows better performance than ResNet image encoder and using all frames is better than subsampling the frames uniformly. For action retrieval, cosine similarities between the text embedding and the averaged image embeddings are computed.

[1]Implementation from `https://github.com/openai/CLIP`

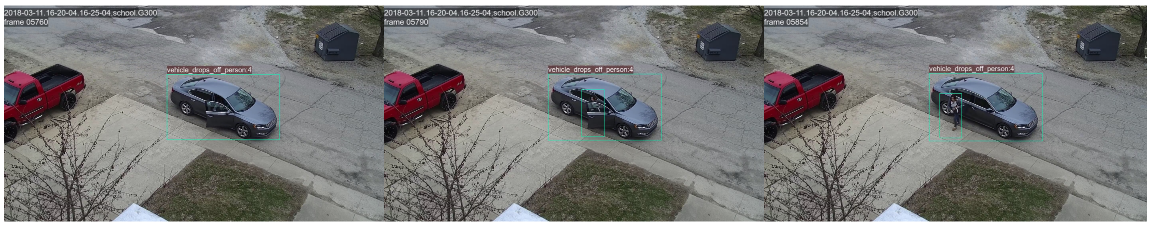
Chapter 5

Experiments

We explain the main dataset we used for our experiments and the evaluation metrics used to evaluate the performances of the activity detector in Section 5.1. Then, ablation studies of different settings to train the activity representation network are provided in Section 5.2. Finally, we validate the accuracy and efficiency of our



(a) Annotation Example of *person_talks_to_person*



(b) Annotation Example of *vehicle_drops_off_person*

Figure 5.1: **Example cuboid annotations of MEVA dataset.** MEVA provides both track-based and cuboid-based annotations. Bounding boxes and timestamps of the actors and activities are provided. For our experiment, we only used cuboid-based annotations.

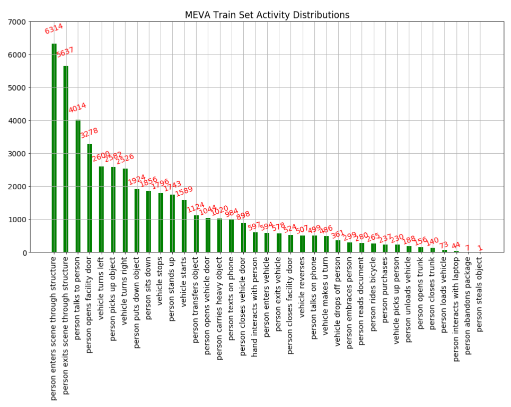
retrieval system by demonstrating the qualitative and quantitative results in Section 5.3.

5.1 MEVA

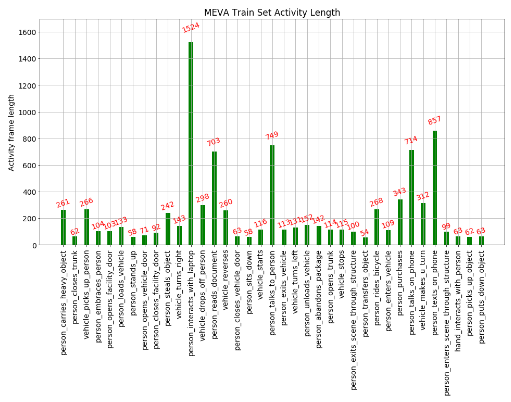
This section describes the characteristics of the MEVA dataset [5] which contains 328 hours of raw ground camera videos and 4.6 hours of aerial videos from drones. For this report, we only consider ground-level videos. About 100 actors were instructed to perform various types of activities based on scripts provided by NIST at MUTC which is denoted as Known Facility (KF). There are about 20 different camera views that include both indoor and outdoor scenes. In addition to raw videos, NIST has also annotated 37 different known activity types (the list is shown in Appendix Figure 1) for more than 2,500 videos. There are various types of activities that can be broadly grouped as person-only, vehicle-only, person-person, and person-vehicle activities. For our experiment, we selected 2,221 videos for train, 100 videos for validation, and 185 videos for the test set. Example annotations of MEVA are shown in Figure 5.1. MEVA provides both geometric and temporal information with bounding boxes of every object involved for every frame and start and end timestamps of each activity. We can either use track-based annotation with a list bounding boxes of every frame for the duration of the activity or cuboid-based annotation with a single bounding box that combines all the bounding boxes but we chose to only use cuboid-based annotation throughout our experiments as we are only evaluated on the temporal accuracy and not on the spatial accuracy of the predictions.

5.1.1 Dataset Statistics

We analyze the MEVA dataset characteristics by looking at the annotations of the training set. Figure 5.2 a) illustrates the distribution of classes based on their occurrences in the annotation. It is clear that the MEVA dataset is heavily imbalanced and follows long-tail distribution similar to public action detection datasets such as AVA [12]. Activities like entering and exiting are very common while a rare activity like *person_steals_object* occur only once in the entire annotations. Furthermore, we have plotted the average lengths of each activity in Figure 5.2 b). We can also see



(a) Activity class distribution



(b) Activity frame length of each activity type

Figure 5.2: **Statistics of the MEVA training set.** a) We can see that the data which contains 46,995 ground truth instances of 37 known activities is heavily imbalanced and follows a long-tail distribution. b) There are 2,221 untrimmed 5 minutes videos in the training set. Static activities such as *person_texts_on_phone* last for about 30 seconds while atomic activities like *person_sits_down* happen within 2 seconds.

that there is a huge gap in terms of activity frame length among different activities. For instance, *person_interacts_with_laptop* on average lasts for 50.8 seconds while *person_stands_up* lasts for less than 2 seconds. Such discrepancy in the duration of activities demonstrates how challenging the MEVA dataset is. Standard action recognition benchmark like Kinetics is composed of atomic activities that all last for less than 10 seconds and the distribution across different activities is also uniform.

5.1.2 Evaluation

In this section, we explain the common evaluation metrics for the MEVA dataset which is also used to evaluate performance on ActEV Challenge. The primary metric used for ActEV is Normalized partial Area Under the DET Curve (nAUDC) which is based on Detection Error Tradeoff curve (DET) commonly used for detection [24]. DET curve is preferred over Reciever Operating Characteristics curve (ROC) as the resulting plots generally have a linear shape as shown in Figure 5.3. The $nAUDC$

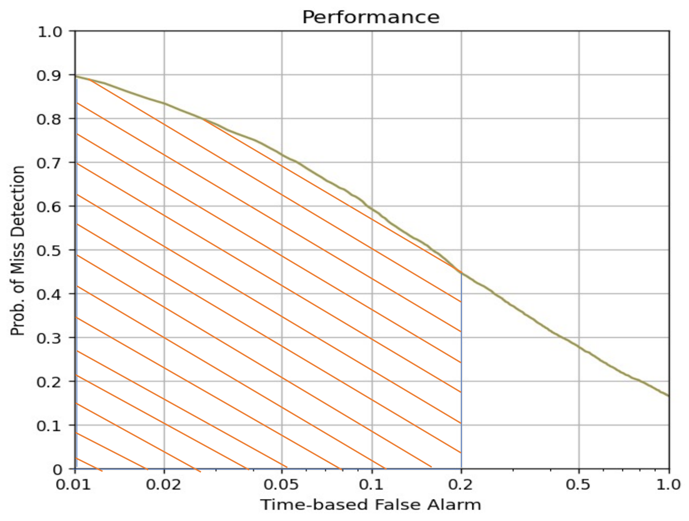


Figure 5.3: **Example DET curve.**$nAUDC@0.2TFA$ is computed by calculating the area displayed in orange.

can be computed as follows:

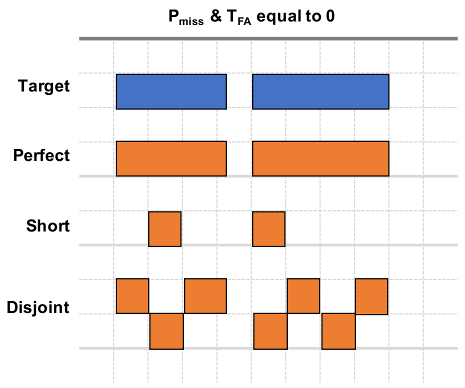
$$nAUDC_a = \frac{1}{a}\int_{x=0}^{a} P_{miss}(x)dx, x = T_{FA}$$

In order to understand the above equation, we need to introduce two concepts which are Time-based False Alarm (T_{FA}) and Probability of miss (P_{miss}). Firstly, T_{FA} is calculated as following:

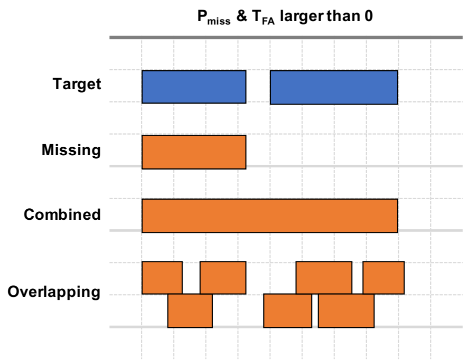
$$T_{FA} = \frac{1}{NR}\sum_{i=1}^{N_{frames}} max(0, S'_i - R'_i)$$

NR means the duration of the video without the target activity occurring, S'_i is the total count of system instances for frame i, and R'_i is the total count of reference instances for frame i. T_{FA} penalizes for additional overlapping predictions and is only affected by frames with more system instances than the reference instances. An example of such case is shown in the last row of Figure 5.4 b). P_{miss} is the fraction of reference instance not detected by the given system and is calculated as $\frac{N_{md}(x)}{N_{ti}}$ where N_{md} is the number of missing detection and N_{ti} is the total number of reference instances.

Before calculating P_{miss}, the Hungarian algorithm to the Bipartite Graph Matching



(a) Good Cases



(b) Bad Cases

Figure 5.4: **Examples of different output types for evaluation.** a) Cases where P_{miss} and T_{FA} are both 0. b) Cases where P_{miss} and T_{FA} are larger than 0.

problem [28] is run to align the system and reference instances and each prediction can only match up to one GT instance. One important thing is that the temporal overlap between reference and system instance must be larger than 1 second. Perfect P_{miss} cases are illustrated in Figure 5.4 a). To achieve low P_{miss} and T_{FA}, predictions can either be a single short prediction that overlaps with GT for at least 1 second or multiple short disjoint predictions. Such a matching mechanism is the reason why we have temporal jittering to produce multiple overlapping proposals for higher recall and post-processing steps to shorten our predictions to be 2 seconds so that they are disjoint. Lastly, submitted systems to ActEV Leaderboard must be within the real-time constraint to be fully acceptable and the real-time speed is calculated as $\frac{SysTime}{VideoDuration}$. For all metrics introduced in this section, a lower score means better.

5.2 Known Activities Classification Results

5.2.1 Experiments on the Internal Set

This section introduces different ablation studies we have conducted to find the best performing TSM network trained and validated on the internal set. Initially, we tested two different input modalities which are Flow and RGB with two different loss functions which are CE and BCE. For this experiment, we used the same hyperparameters and ResNet50 backbone trained on Kinetics with the number of

Table 5.1: Performance comparison of multiclass and multilabel trained models on the internal test set.

Loss	Modality	$nAUDC@0.2T_{FA} \downarrow$	$P_{miss}@0.04T_{FA} \downarrow$
Multiclass CE	RGB	0.5204	0.5841
	Flow	0.5137	0.6008
Multilabel BCE	RGB	0.4049	0.5092
	Flow	**0.4014**	**0.4887**

Table 5.2: Ablations of different backbone settings on the internal test set.

Modality	# of Segments	Architecture	$nAUDC@0.2T_{FA} \downarrow$	$P_{miss}@0.04T_{FA} \downarrow$
Flow	8	TSM-R50	**0.4014**	**0.4887**
RGB	8	TSM-R50	0.4049	0.5092
	16	TSM-R50	0.4073	0.521
	8	TSM-R101	0.4148	0.502

segments fixed to 8. As shown in Table 5.1, Flow TSM performed better than RGB TSM in $nAUDC$ but slightly worse in P_{miss} for multiclass setting. Surprisingly, switching the loss function to BCE resulted in a dramatic performance improvement for both modalities by almost 10% in both $nAUDC$ and P_{miss}. This ablation study demonstrates that assigning multiple ground truth activity labels to a single proposal helps the model to learn a better representation of the activities. Multilabel Flow TSM performed better than multilabel RGB TSM on both metrics. It is an interesting observation as the Flow TSM trained on Kinetics performed far worse than RGB TSM on the Kinetics test set with a Top-1 accuracy gap of 13.1%.

We conducted another ablation where we tested different architecture settings for RGB TSM to reduce the gap between Flow and RGB as shown in Table 5.1. We tested a larger backbone with ResNet101 and tried doubling the number of frames fed to the network but only experienced minor improvement. We believe the main reason why Flow performs better than RGB is the limited number of scenes in MEVA. As a result, the RGB model often overfits to correlate a particular set of activities with a particular scene. Furthermore, optical flow removes background semantics and inherently conducts background subtraction which can help the model to not focus on the scene semantics. In the end, we submitted TSM Flow and TSM RGB both

Table 5.3: Performance on the KF KA Full Set as of 07/20/2021.

Systems	$nAUDC@0.2T_{FA} \downarrow$	$P_{miss}@0.02T_{FA} \downarrow$	RT (Relative Time) $\downarrow$
CMU-DIVA	**0.1903**	**0.3823**	0.415
UMD	0.2914	0.4636	0.373
UMD-Columbia	0.3055	0.4716	0.516
Ours (Flow)	0.3236	0.5297	0.464
Purdue	0.3378	0.5848	**0.126**
UCF	0.3386	0.4767	0.693
Ours (RGB)	0.3726	0.6101	0.243
IBM-Purdue	0.3799	0.6045	0.130
MINDS_JHU	0.4834	0.6649	0.967
BUPT-MCPRL	0.7985	0.9281	0.123

with TSM-R50 and 8 frames as input.

5.2.2 Performance on ActEV KA Leaderboards

To test the generalization capability of our trained TSM, we made submissions of our best trained TSM models to the Known Facility Known Activity (KF KA) Leaderboard. The test videos come from the same set of camera views from MEVA training data and approximately contain 108 hours long videos. The performance is displayed in Table 5.3. We have placed 4th out of 9 teams that have participated in the challenge as of 07/20/2021. We can see that our pipeline is competitive with only 3% gap between the second best performing system. The purpose of the submission was to prove that our trained model is generalizable to unseen test videos for the set of 37 known activities it was trained on.

In addition, we also made submissions to the Unknown Facility Known Activity (UF KA) Leaderboard to test the generalization capability of TSM on unseen scenes. UF KA includes 35 out of 37 known activities and contains 116 hours long videos. As shown in Table 5.4, our system does not perform as well on UF KA compared to KF KA as the model has never seen the scene semantics of UF. Furthermore, the gap between Flow and RGB TSM is very large with more than %7 difference in $nAUDC$. It can be hypothesized that the RGB model overfits to correlate activities with a set of scenes in KF and has difficulties generalizing to unseen semantics. Adding

Table 5.4: Performance on the UF KA Micro Set as of 07/20/2021.

Systems	$nAUDC@0.2T_{FA} \downarrow$	$P_{miss}@0.02T_{FA} \downarrow$	RT (Relative Time) $\downarrow$
CMU-DIVA	**0.3723**	**0.5823**	0.527
UCF	0.3893	0.5938	0.817
IBM-Purdue	0.3998	0.6105	0.701
UMD-Columbia	0.4198	0.6129	0.534
Visym Labs	0.4336	0.6254	1.041
UMD	0.4405	0.6292	0.616
Ours (Flow)	0.5142	0.7122	0.684
Purdue	0.5160	0.7523	**0.234**
Ours (RGB)	0.5828	0.7893	0.278
MINDS_JHU	0.6560	0.7957	0.924

techniques like background subtraction may help boost the performance of the RGB model in the future.

5.3 Surprise Activities Retrieval Results

5.3.1 Qualitative Analysis

Before making a submission to Unknown Facility Surprise Activity (UF SA) Leaderboard, we tested the power of our retrieval system by visualizing the top retrieved cuboids from the simulated surprise activity set.[1] NIST provides a simulated surprise activity data using the 35 known activities (excluding *person_steals_object* and *person_abandons_package*) to aid challenge participants to develop surprise activity system. There are total of 60 test videos and 2-6 exemplar clips for every activity along with text descriptions. For visualization purposes, we only demonstrate the retrieval results on 32 test videos in which our proposal generation module produced a total of 102,853 proposals.

Figure 5.5 shows the visualization results of our TSM retrieval system of four activities. On the left, frames from the query clips are displayed with blue bounding boxes corresponding to the ground truth annotations and red bounding boxes corresponding

[1]https://gitlab.kitware.com/actev/actev-data-repo/-/tree/master/partitions/ActEV-Eval-CLI-Validation-Set7

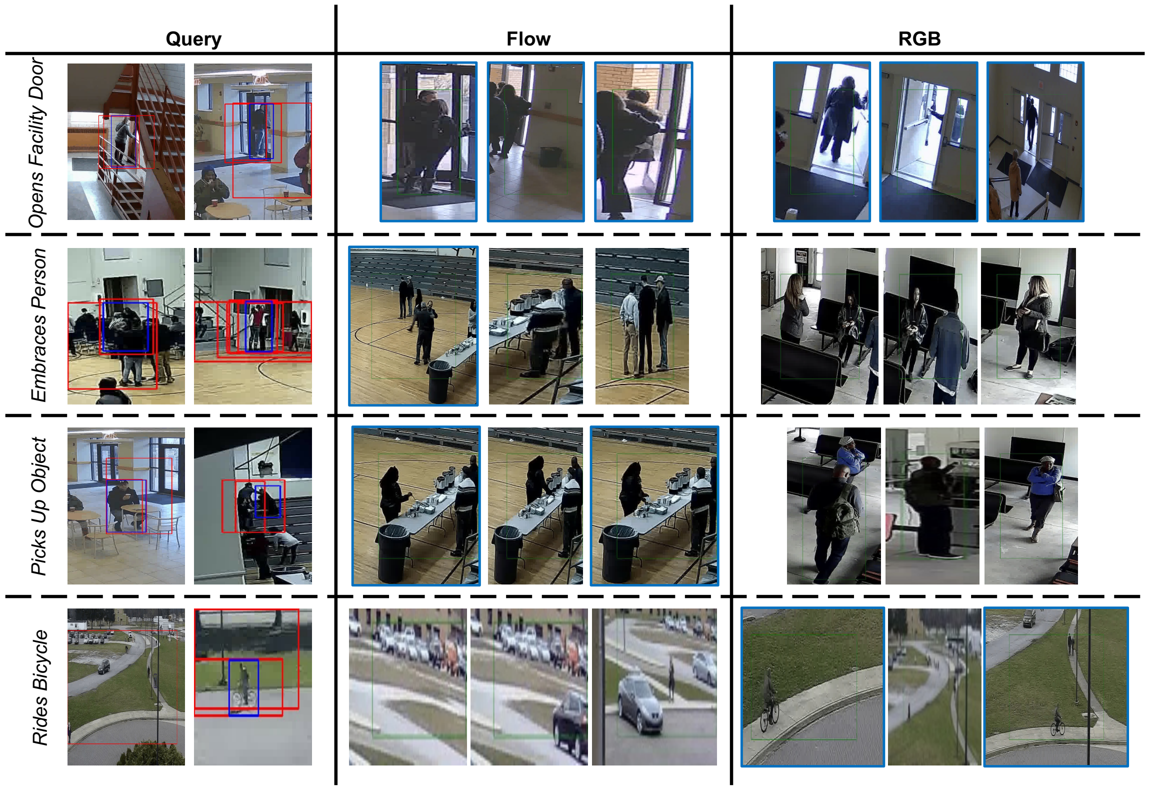
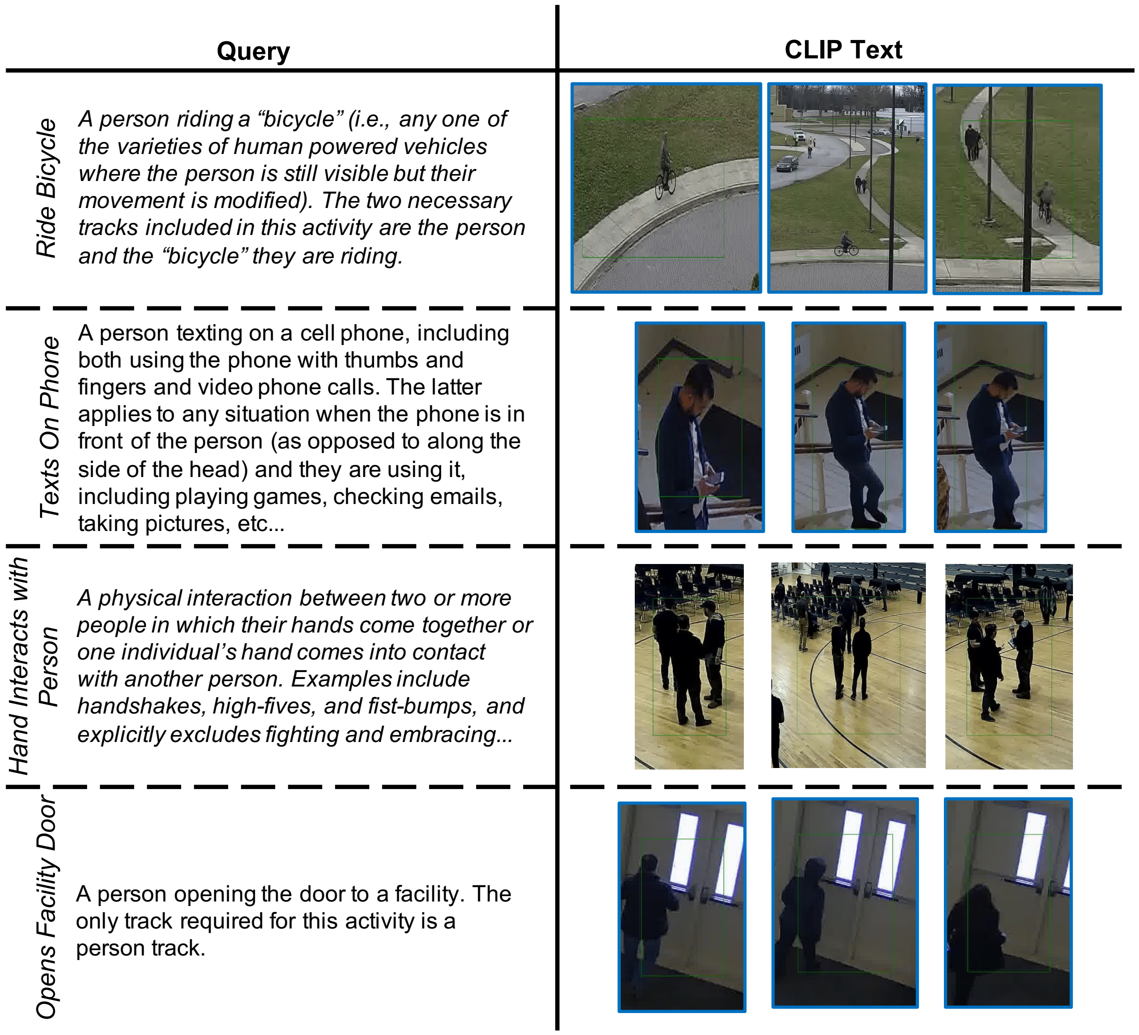


Figure 5.5: **Visualization of the query clips and the corresponding top-3 retrieved results from TSM Flow and RGB.** We only display two exemplar queries for visualization. Blue bounding boxes correspond to GT and red ones are generated proposals. Gallery proposals were enlarged by 50% and rescaled for visualization purposes (Best viewed in color).

to the generated proposals. Short and atomic activities like *person_opens_facility_door* are simple for both Flow and RGB TSM in which both networks correctly detected the cuboids that contain people opening a door. However, when an activity requires more motion understanding such as *person_embraces_person* and *person_picks_up_object*, RGB TSM is unable to retrieve correct cuboids in the top-3 ranked list while Flow TSM successfully retrieves one correct cuboid. On the contrary, activities that are more semantically distinctive such as *person_rides_bicycle* can be more easily retrieved by RGB TSM. Hence, we can see the complementary nature of the two different input modalities.

We also visualize retrieval results of our CLIP text-based retrieval system in Figure 5.6 using ViT B/32 backbone. Considering that we only feed textual de-



A person riding a "bicycle" (i.e., any one of the varieties of human powered vehicles where the person is still visible but their movement is modified). The two necessary tracks included in this activity are the person and the "bicycle" they are riding.

A person texting on a cell phone, including both using the phone with thumbs and fingers and video phone calls. The latter applies to any situation when the phone is in front of the person (as opposed to along the side of the head) and they are using it, including playing games, checking emails, taking pictures, etc...

A physical interaction between two or more people in which their hands come together or one individual's hand comes into contact with another person. Examples include handshakes, high-fives, and fist-bumps, and explicitly excludes fighting and embracing...

A person opening the door to a facility. The only track required for this activity is a person track.

Figure 5.6: **Visualization of CLIP text encoder from activity descriptions and corresponding top-3 retrieved results.** CLIP was originally trained on a large web dataset that mainly contains noun-based objects. As a result, activities that contain distinctive objects like *person_rides_bicycle*, can be retrieved well even with only textual descriptions. For activities that require motion understanding such as *hand_interacts_with_person*, CLIP is unable to retrieve correctly (Best viewed in color).

scriptions to produce the query descriptors, our text-based system demonstrates a promising result. Especially, CLIP is able to retrieve activities that contain distinctive objects such as phones or bicycles. For instance, CLIP text does better at retrieving *person_rides_bicycle* than RGB TSM by getting all three predictions correct.

Furthermore, although CLIP failed to understand the temporal characteristics of *hand_interacts_with_person* which includes actions like high-five or patting the back of a person, it was able to identify cuboids that contain hand gestures. Since CLIP was only trained on web crawled image-text pairs that include mostly objects, it is unable to generalize to actions. Nevertheless, CLIP was never finetuned on the MEVA data but it still shows promising results for several activities.

5.3.2 Performance on ActEV UF SA Leaderboard

Table 5.5: Performance on the UF SA as of 07/20/2021.

Dataset	Systems	$nAUDC@0.2T_{FA} \downarrow$	$P_{miss}@0.02T_{FA} \downarrow$	RT $\downarrow$
Micro Set	UMD	**0.6626**	**0.8666**	0.518
	UCF	0.6642	0.8876	**0.256**
	Ours (Flow-S)	0.7030	0.8789	0.613
	Ours (Flow)	0.7103	0.8825	0.658
	Ours (RGB)	0.7248	0.9099	0.271
	Ours (ViT/B32)	0.7855	0.9512	0.404
Full Set	UMD	**0.6151**	0.8428	0.511
	Ours (Flow-S)	0.6162	**0.8204**	0.589
	Ours (Flow)	0.6276	0.8345	0.589
	UCF	0.6337	0.8557	**0.273**

Table 5.5 illustrates our submissions on the Unknown Facility Surprise Activity (UF SA) Leaderboard. UF SA has two leaderboards: Micro Set and Full Set. Micro Set is a subset of the Full Set that contains 116 hours of test videos to facilitate the fast processing of submissions. UF SA videos were collected from different sites than that of KF so the scene distribution is expected to vary from the MEVA training data. There are 10 novel activities that are provided on the fly which are known to be different from the 37 known activity types. On the Micro Set, we can see that TSM Flow performs better than TSM RGB. TSM Flow-S stands for TSM Flow with shortening activity post-processing which brings about 0.007 and 0.004 gain on $nAUDC$ and P_{miss} respectively. As a result, we place second place next to the UMD system on P_{miss}. On the Full Set, our best performing system performs achieves 0.6162 in terms of $nAUDC$ and only has about 0.0011 gap with the best performing

UMD system. Furthermore, it is ranked first in terms of P_{miss} metric and outperforms the UMD system by 0.0224. This indicates the robustness of our vision-based retrieval system on unseen data without needing to finetune the network.

We also made a submission using our language-based retrieval system with CLIP ViT B/32 backbone. The submitted system has about 5 to 6% gap with our TSM RGB model. However, considering that textual descriptions are given as the sole source of information to the query descriptor, the result seems promising. In addition, CLIP was never finetuned on video or MEVA dataset so it is making decent predictions through "zero-shot". We believe that the use of different modalities to retrieve a novel set of activities can help facilitate the deployment of activity detectors in the wild.

Chapter 6

Conclusions

We propose visual-based and text-based retrieval systems that can successfully detect novel activities of interest with only few examples. We first point out how challenging the activity detection required by the MEVA dataset is and propose a comprehensive framework that can localize activities from untrimmed videos through several modules. Our pipeline includes object detection, hierarchical clustering, optical flow extraction, TSM network, and post-processing modules that can run in real-time. Our system achieves competitive results on both on Known Facility Known Activity and Unknown Facility Surprise Activity Leaderboard.

We also demonstrates possibility of incorporating multimodal learning into activity detection through visualizing the retrieval results on the simulated surprise activity data and showing performance on UF SA Leaderboard. For our future work, we hope to develop a joint retrieval system that can incorporate different modalities like RGB and Flow for better performance. Furthermore, devising a method that can find the correlations between activities and input modalities can help further improve the performance by optimizing each activity retrieval with specific modality.

Bibliography

[1] A. Arnab, M. Dehghani, G. Heigold, Chen Sun, Mario Lucic, and C. Schmid. Vivit: A video vision transformer. *ArXiv*, abs/2103.15691, 2021. 2.1

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding?, 2021. 2.1

[3] João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733, 2017. 2.1

[4] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. Hybrid task cascade for instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 4974–4983. Computer Vision Foundation / IEEE, 2019. 3.1

[5] Kellie Corona, Katie Osterdahl, Roderic Collins, and Anthony Hoogs. Meva: A large-scale multiview, multimodal video dataset for activity detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1060–1068, January 2021. 1, 2.2, 5.1

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`. 2.1, 4.3

[7] Bernard Ghanem Fabian Caba Heilbron, Victor Escorcia and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015. 2.2

[8] Han Fang, Pengfei Xiong, Luhui Xu, and Yu Chen. Clip2video: Mastering video-text retrieval via image clip, 2021. 2.3

[9] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2.1, 2.2

[10] Joshua Gleason, Rajeev Ranjan, Steven Schwarcz, Carlos Castillo, Jun-Cheng Chen, and Rama Chellappa. A Proposal-Based Solution to Spatio-Temporal Action Detection in Untrimmed Videos. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 141–150, January 2019. doi: 10.1109/WACV.2019.00021. ISSN: 1550-5790. 2.2, 3

[11] Shreyank N. Gowda, Marcus Rohrbach, and Laura Sevilla-Lara. Smart frame selection for action recognition. In *AAAI*, 2021. 2.2

[12] C. Gu, Chen Sun, Sudheendra Vijayanarasimhan, C. Pantofaru, David A. Ross, G. Toderici, Yeqing Li, Susanna Ricco, R. Sukthankar, C. Schmid, and J. Malik. Ava: A video dataset of spatio-temporally localized atomic visual actions. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2018. 2.2, 5.1.1

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90. 4.3

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017. doi: 10.1109/ICCV.2017.322. 3.1

[15] 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. 2.2

[16] Will Kay, João Carreira, K. Simonyan, B. Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, T. Back, A. Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *ArXiv*, abs/1705.06950, 2017. 2.1, 2.2, 3.5

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`. 3.5

[18] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 2.1, 3.5

[19] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. BSN: Boundary Sensitive Network for Temporal Action Proposal Generation.

arXiv:1806.02964 [cs], September 2018. URL http://arxiv.org/abs/1806.02964. arXiv: 1806.02964 version: 3. 2.2

[20] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. BMN: Boundary-Matching Network for Temporal Action Proposal Generation. *arXiv:1907.09702 [cs]*, July 2019. URL http://arxiv.org/abs/1907.09702. arXiv: 1907.09702 version: 1. 2.2

[21] Tsung-Yi Lin, M. Maire, Serge J. Belongie, James Hays, P. Perona, D. Ramanan, Piotr Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 3.1

[22] Wenhe Liu, Guoliang Kang, Po-Yao Huang, Xiaojun Chang, Lijun Yu, Yijun Qian, Junwei Liang, Liangke Gui, Jing Wen, Peng Chen, and Alexander G. Hauptmann. Argus: Efficient activity detection system for extended video analysis. In *2020 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pages 126–133, 2020. doi: 10.1109/WACVW50321.2020.9096929. 2.2

[23] Huaishao Luo, Lei Ji, Ming Zhong, Yang Chen, Wen Lei, Nan Duan, and Tianrui Li. Clip4clip: An empirical study of clip for end to end video clip retrieval, 2021. 2.3

[24] Alvin F. Martin, G. Doddington, Teresa M. Kamm, M. Ordowski, and Mark A. Przybocki. The det curve in assessment of detection task performance. In *EUROSPEECH*, 1997. 5.1.2

[25] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*, 2019. 2.3

[26] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, I. Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncurated instructional videos. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9876–9886, 2020. 2.3

[27] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *ArXiv*, abs/1109.2378, 2011. 3.2

[28] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of The Society for Industrial and Applied Mathematics*, 10:196–210, 1957. 5.1.2

[29] A. Neubeck and L. Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 850–855, 2006. doi: 10.1109/ICPR.2006.479. 4.2

[30] Sangmin Oh, A. Hoogs, A. Perera, Naresh P. Cuntoor, Chia-Chih Chen, J. T. Lee, S. Mukherjee, J. Aggarwal, Hyungtae Lee, L. Davis, E. Swears, Xiaoyang Wang,

Q. Ji, K. Reddy, M. Shah, Carl Vondrick, H. Pirsiavash, D. Ramanan, Jenny Yuen, A. Torralba, B. Song, Anesco Fong, A. Roy-Chowdhury, and M. Desai. A large-scale benchmark dataset for event recognition in surveillance video. *CVPR 2011*, pages 3153–3160, 2011. 2.2

[31] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. 2.3, 4.3

[32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. URL `https://arxiv.org/abs/2103.00020`. 4.3

[33] Shaoqing Ren, Kaiming He, Ross B. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 2015. 2.2, 3.1

[34] M. N. Rizve, U. Demir, Praveen Tirupattur, A. J. Rana, Kevin Duarte, I. Dave, Y. Rawat, and M. Shah. Gabriella: An online system for real-time activity detection in untrimmed security videos. *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4237–4244, 2021. 2.2

[35] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL `https://aclanthology.org/P16-1162`. 4.3

[36] K. Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 2.1

[37] 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. 2.2

[38] Chen Sun, Austin Myers, Carl Vondrick, K. Murphy, and C. Schmid. Videobert: A joint model for video and language representation learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7463–7472, 2019. 2.3

[39] Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006 (146):10, 2006. 3.4

[40] Du Tran, Heng Wang, L. Torresani, Jamie Ray, Y. LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 2.1

[41] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017. 2.1

[42] Heng Wang, Alexander Kläser, C. Schmid, and C. Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103:60–79, 2012. 2.1

[43] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11):2740–2755, 2019. doi: 10.1109/TPAMI.2018.2868668. 2.1, 3.5

[44] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krähenbühl, and Ross Girshick. Long-Term Feature Banks for Detailed Video Understanding. In *CVPR*, 2019. 2.2

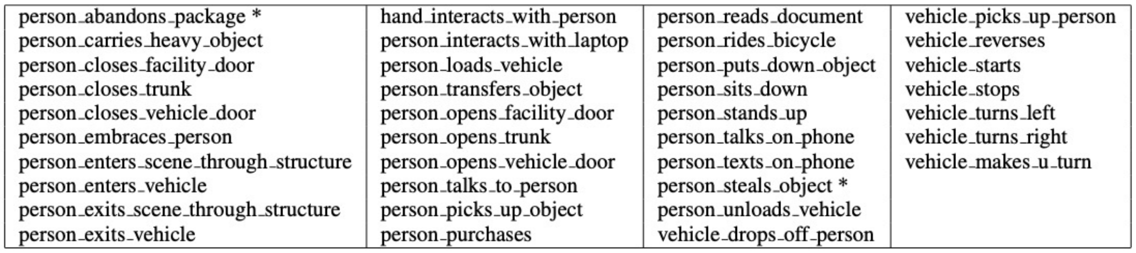
[45] J. Xu, Tao Mei, Ting Yao, and Y. Rui. Msr-vtt: A large video description dataset for bridging video and language. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5288–5296, 2016. 2.3

[46] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l1 optical flow. In *DAGM-Symposium*, 2007. 3.4

Appendix

Dataset Information

Figure 1 illustrates the list of known activity names in MEVA. During training, we add an activity named as *No_Activity* which is treated as a background activity.



person_abandons_package *	hand_interacts_with_person	person_reads_document	vehicle_picks_up_person
person_carries_heavy_object	person_interacts_with_laptop	person_rides_bicycle	vehicle_reverses
person_closes_facility_door	person_loads_vehicle	person_puts_down_object	vehicle_starts
person_closes_trunk	person_transfers_object	person_sits_down	vehicle_stops
person_closes_vehicle_door	person_opens_facility_door	person_stands_up	vehicle_turns_left
person_embraces_person	person_opens_trunk	person_talks_on_phone	vehicle_turns_right
person_enters_scene_through_structure	person_opens_vehicle_door	person_texts_on_phone	vehicle_makes_u_turn
person_enters_vehicle	person_talks_to_person	person_steals_object *	
person_exits_scene_through_structure	person_picks_up_object	person_unloads_vehicle	
person_exits_vehicle	person_purchases	vehicle_drops_off_person	

Figure 1: **List of 37 Known Activities in MEVA.**

Additional Results on ActEV UF Leaderboard

Figure 2 illustrates the per activity scores of the two submissions. Flow consistently outperforms RGB on most of the known activities.

Furthermore, we display the DET curve of models submitted to the UF SA Leaderboard in Figure 3. Our system consistently outperforms other two submissions on both P_{miss} in low T_{FA} (0 0.05) and in high T_{FA} (0.3 1.0)

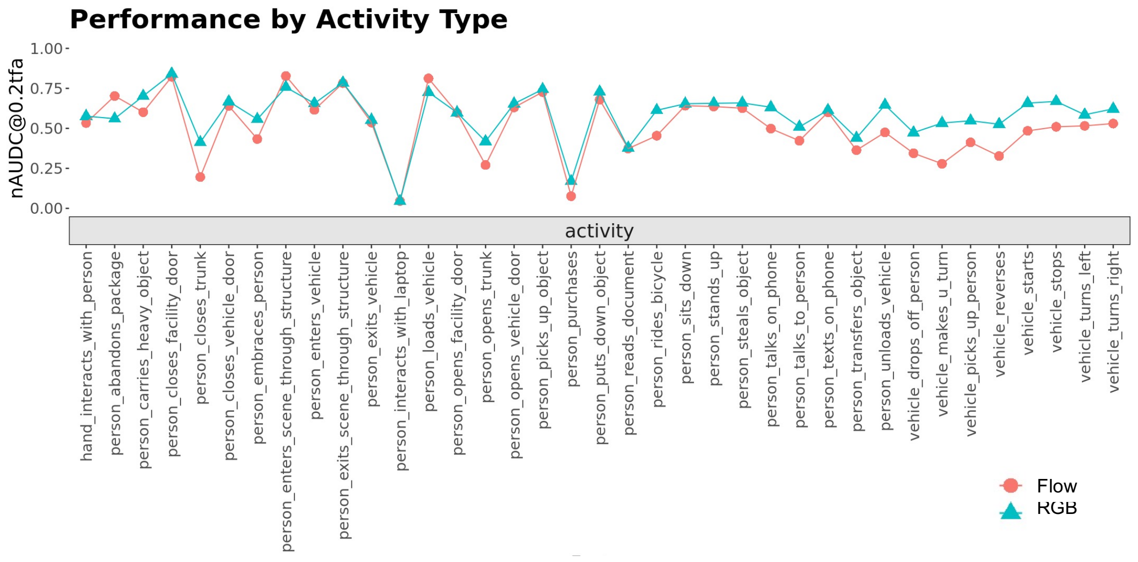


Figure 2: **Per-activity performance of Flow and RGB on UF KA.**

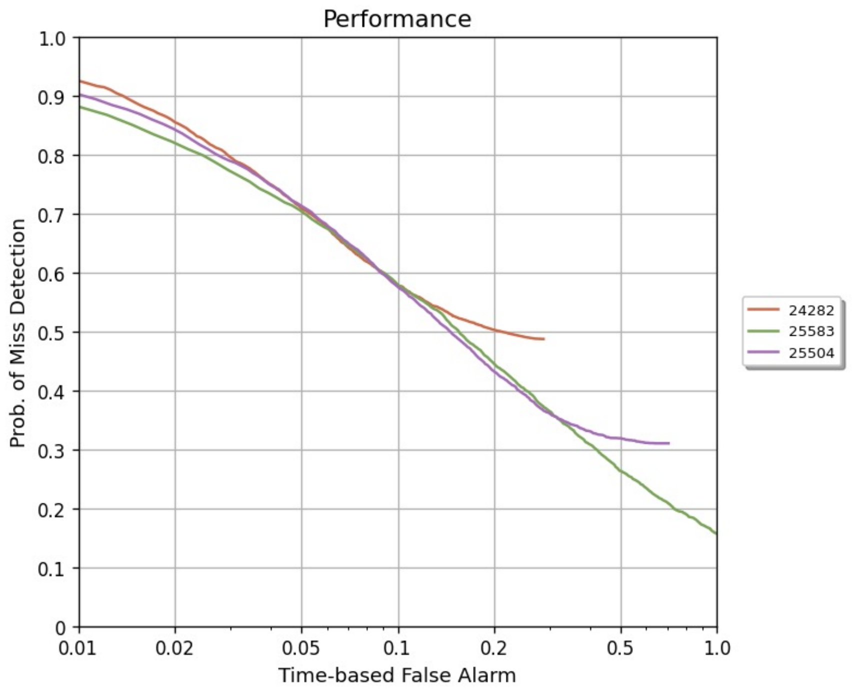


Figure 3: **DET curve of systems submitted to UF SA Leaderboard.** *24282*: UCF, *25504*: UMD, and *25583*: Ours. It is clear that the DET curve of our system is almost linear and outputs activity even in high T_{FA}.