DETECTION OF ROBBERY-RELATED CONCEPTS USING DEEP LEARNING

BY

VIVAAINDREAN NG BIN SHAMIR NG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Kampar Campus)

JANUARY 2020

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: _DETECTION OF ROBBERY-RELATED CONCEPTS USING_____
_DEEP LEARNING_____
_____

Academic Session : JANUARY 2020

I ____VIVAAINDREAN NG BIN SHAMIR NG_____

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

_____ _____

(Author's signature) (Supervisor's signature)

Address:

No.44, Lorong Sri Tambun 2,
Taman Sri Tambun,
14100 Simpang Ampat,
Pulau Pinang.

Tan Hung Khoon
_____
Supervisor's name

Date: ___21/4/2020_____ **Date**: 22 April 2020

DETECTION OF ROBBERY-RELATED CONCEPTS USING DEEP LEARNING

BY

VIVAAINDREAN NG BIN SHAMIR NG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Kampar Campus)

JANUARY 2020

DECLARATION OF ORIGINALITY

I declare that this report entitled "**DETECTION OF ROBBERY-RELATED CONCEPTS USING DEEP LEARNING**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _____

Name : VIVAAINDREAN NG BIN SHAMIR NG

Date : 21/4/2020

ACKNOWLEDGEMENT

First and foremost, I would like thank my supervisor, Dr. Tan Hung Khoon, for willing to undertake me as his supervisee. I feel honoured to be given the opportunity in undertaking a complex project in the field of deep learning. His patience in guiding and aiding me is unequivocally helpful throughout my time in engaging my project, as I have learned a lot for the duration of my time in conducting this project.

Over and above that, I would also like to take this opportunity in thanking my parents for supporting me, in particular for providing a workstation equipped with GPU, which further helps a lot in accelerating the feature extraction and training process of this experiment. What's more, I am grateful for my friends and course mates in UTAR to have my back.

ABSTRACT

Detecting robbery-related concepts or any particular violent scenes in videos is one of the most fundamental on-going work in the world of computer vision. While it is evident that there are more discovery and improvements of such detection task especially in the realm of fully supervised settings, the acquisition of labelled training data at video's temporal-level is not sensible.

We instead tackle this problem by proposing two novel approaches – MIL-Ranking as well as TAL. At its very core, both aforementioned methods only necessitates ground-truth at video-level, instead of temporal-level. We show that the implementation of MIL and TAL approaches on the huge-scale UCF-Crime dataset demonstrates their capabilities in detecting violent-related concepts at video's temporal-level.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AUC	Area under ROC curve
C3D	3D ConvNet
CNN	Convolution Neural Network
FC	Fully-Connected
GPU	Graphical Processing Unit
I3D	Inflated 3D ConvNet
MIL	Multiple Instances Learning
OIC	Outer Inner Contrastive
OS	Operating System
R-C3D	Region Convolutional 3D Network
ReLU	Rectified Linear Unit
ROC	Receiver Operating Characteristic
ROI	Region of Interest
SSN	Structured Segment Networks
SVM	Support Vector Machine
TAL	Temporal Action Localization

CHAPTER 1: INTRODUCTION

1.1 Problem Statement and Motivation

Precise detection on the presence of violent scenes is getting more crucial now more than ever, especially in field of computer vision and deep learning. This is true thanks to the rise of increasing available data, coupled with discovery of state-of-the-art deep learning techniques, as well as the presence of hardware capable of enriching and speeding up computations required in deep learning tasks. In this paper, what we are interested in is the system's capability in predicting the violent category of the input video, as well as localizing the starting and ending timeframe of the violent event. The **primary problem** that motivates this experiment is the **impracticality of training phase which involves untrimmed videos**. For perspective, many state-of-the-art techniques like R-C3D [10] as well as SSN are able to predict and localize the starting and ending timeframe for specific actions. However, they necessitate the ground-truth labels at a more granular level during training. As such, these methods are regarded as *fully-supervised* because they rely on annotations at temporal-level, to indicate when a particular action starts and stops in a video.

In fully-supervised paradigm, it is extremely laborious especially for untrimmed input videos, because they generally contain multiple background scenes that are irrelevant to our interest. A robbery scene in an untrimmed video might only contain the robbing act in the final portion of a video, while the majority of it contains unwanted scenes. This demands the annotation of only wanted crime scenes at temporal-level of videos. However, this is too meticulous and time-consuming as it requires maximum supervision in order to fully annotate the datasets at temporal-level. Apart from that, providing ground-truth labels on temporal-level may be trickier than it seems, due to conflicting ideas. A normal video (without any violent scenes) may contain actions where people are running around, and thus assigned a negative label. However, other might presumed the act of somebody running and fleeing in any scene to strongly imply the presence of violent scenes due to victims fleeing away from a violent event, and thus be given positive label at that particular temporal portion, to indicate presence of violent scenes. It is sometimes complicated to exactly define what is considered as normal behaviors or violent acts without taking into account of the overall video or context, based on report by (Chandola, et al., 2009).

Second problem is concerning the **difficulty in predicting a particular violent scene fully, without annotations at temporal-level**. When it boils down to violent-related action, there are a wide array of sub-actions or lower-level concepts that can be uniquely identified to that particular violence. For example, a theft may employ the steal-and-run tactic (robbery), while some thefts occurred where the perpetrators lingering around in a store, right before shoplifting by concealing the stolen item underneath their clothes, making as less noticeable actions as possible before slowly walking away. Previously mentioned lower-level concepts like running or hastily snatching other person's belonging likely constitutes to a robbery scene, and this could be instantly learned by the model if it is supplied with temporal annotations. Unfortunately in the case of *weakly-supervised* techniques, this may seem tricky due to absence of ground-truth at temporal-level.

In order to address the complexity of predicting a specific kind of violence actions as previously mentioned, the general step would be to employ system that specifically detects this kind of events as being described by (Mohammadi, et al., 2015) where the work was aimed specifically on both violence in crowds and riots in prison. However, the main limitation with said work is that it would hardly generalize on other types of violence with different premises – as in, violence that occurs in non-crowded area, or crimes that are totally different in nature, such as a perpetrator vandalizing a property. As a matter of fact, a thief who shoplifted may have done so very subtly with minimal actions, which definitely poses a challenge to detection systems that solely relies on a single visual cues exhibited in that video when it comes to detecting shoplifting scene. To that end, a more robust deep learning architecture is needed where it is able to perform better generalization on various types of violent actions containing different visual cues, and not solely rely on a single particular action exhibited by subject(s). In other words, we would not want our system to only operate well on limited categories of violent events.

Moreover, in regards to this experiment, **another hindrance** in activity recognition of crime scene primarily lies in the **scarcity of a wide array of crime scenes samples**. It is a challenging task in acquiring large datasets which are rich in violent-based scenes as our training samples. Case in point, the Violence in Crowds dataset by (Hassner, et al., 2012) contains only 246 videos comprised of only two distinction, where half of it contains disruption among crowds, while the other half dataset contains normal scenes. Another variation of violence-related datasets, Hockey dataset [2] comprises 1000 clips, which can be categorized into 500 clips containing fights between

hockey players, while the remaining 500 clips are non-violence. For these datasets, both of them contains violence-related samples which are too specific to their particular category and lack in terms of variety. Classifiers that is being trained solely on these datasets may likely not generalize well on other types of violence samples that do not involve masses or hockey players. In this experiment, courtesy of (Sultani, et al., 2018), we will work on UCF-Crime datasets which contains 1,900 surveillance videos that can be broken down into 13 distinct violent categories as well as a non-violent category. The very fact that said dataset is based on recorded surveillance cameras without undergoing any trimming, makes it a very suitable and realistic candidate when it comes to violent scene detection.

The main motivation of this project is to address aforementioned problem statements by exploring two weakly-supervised methods in violent scene detection, namely by experimenting on **Multiple Instances Learning (MIL)** and **Temporal Action Localization (TAL)** proposed by (Sultani, et al., 2018) and (Liu, et al., 2019), respectively. Further inner workings of both aforementioned techniques shall be further delved in chapter 3.

1.2 Project Objectives

1. **To develop a Multiple Instances Learning (MIL) violence detection system**. In this experiment, we wish to detect the presence of any violence scenes present in an untrimmed video by only providing labels on video-level during training – without having to meticulously annotate at temporal-level within every video. MIL shall be able to temporally detect the occurrences of violent scenes with respect to the frames of a particular video by generating higher anomaly scores to indicate the presence of any violence-related events, and minimize anomaly scores for trivial background scenes.

2. **To develop a Temporal Action Localization (TAL) violence detection system**. TAL shares some similar traits with MIL, since both of these are weakly-supervised. In terms of capabilities however, TAL predominantly able to localize the boundary of a violence action by outputting the starting and ending timeframe, as well as classify the types of violence categories on every localized actions.

3. **To juxtapose the performances of MIL and TAL paradigms**. Since both MIL and TAL are weakly-supervised settings, we could gauge and compare both performances in detecting violent scenes. In short, both of these approaches have fundamentally different results – MIL aims to generate higher anomaly scores corresponding to temporal segments containing violent scenes and generate minimal scores for segments containing normal scenes. TAL on the other hand intends to localize violent scenes present in a video while simultaneously classify violent categories for each localized violent scenes.

1.3 Impact, Significance and Contribution

- The development of this experiment serves as a gateway for further exploration on the detection of violent scenes in videos through MIL and TAL. By capitalizing on weakly-annotated data, our system is capable of localizing actions for input videos, aside from avoiding possibly noisy data at temporal-level. This very fact is imperative, due to the difficulties in obtaining annotations and labels at temporal-level for each videos.

- Considering the huge amount of violent-based videos that can be further broken down into different categorization of violence, the UCF-Crime dataset is a prime candidate for further research when it comes to any violent-related detection, that are not bounded by specific elements such as location, or even the behaviors of perpetrators.

CHAPTER 2: LITERATURE REVIEW

2.1 Action Recognition in Video

Since the main gist of this project revolves around the detection of any violence or robbery-related concepts, we are really interested in detecting their presence throughout the entire duration of a given video. Given the fact that video is just a collection of frames that changes with respect to time, we can inherently regard detection in video as the expansion of aggregating 2D image classification task (2D tasks) on multiple input frames along the time dimension (3D tasks).

To that end, (Karpathy, et al., 2014), proposed three approaches – *Late Fusion*, *Early Fusion* and *Slow Fusion* in the context of classifying continuous video frames and fusing them temporally via 2D Convolution, with the aim of predicting video's classification.

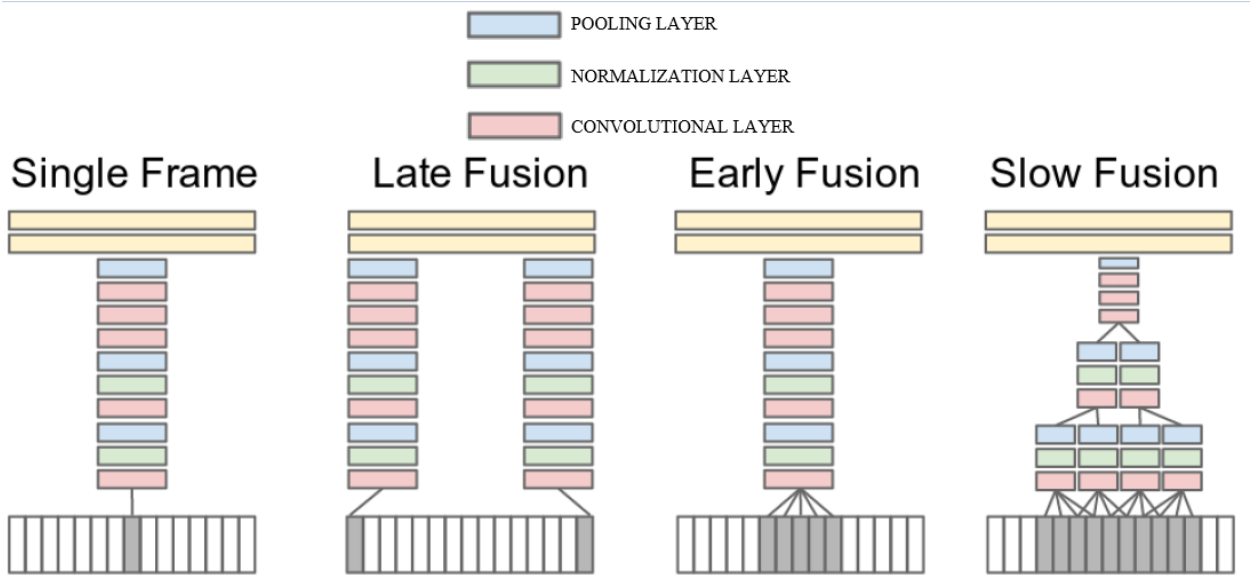


Figure 2.1.1: Fusion-based approaches explored in video classification.

Standard Single Frame CNN is derived from AlexNet architecture, where it takes a single input frame of a given video at a time, and only fuses information signal of entire frames (per video) at fully-connected (FC) layer, where it is linked to softmax classifier – resulting in computing probability distribution for multiple actions or activities. This approach is capable of detecting information spatially, but it could not temporally detect motions of objects or subjects throughout the frames. Instead of relying only on single CNN, *Late Fusion* is introduced, where it is made up

of two separated Single Frame CNN positioned at the first and last frame respectively, where both of them are 15 frames from each other. In short, two sampled frames are simultaneously fed into this *Late Fusion* scheme. Since the weights for both of these separated CNN are shared, the resulting predictions from both of these CNN will be merged in the FC layer, thereby enabling the detection of motion by computing the differences between outputs from both CNN components.

The second technique, *Early Fusion*, is also based on the adjustment of Single Frame CNN. Instead of feeding one frame at a time into the network, *Early Fusion* takes in a chunk of sampled frames (*N* in total). In this case, the first convolution layer has access to input data that temporally spans across *N* frames at a time, and this even allows the detection of a particular motion's characteristic, namely its speed and direction. *Slow Fusion* is considered to be the mixture of Early and Late Fusion and it performs the best out of the three Fusion schemes. Based on Figure 2.1.1, the first convolutional layers in *Slow Fusion* pipeline will convolve on 4 input frames temporally which are partially overlapped within a window size of *N* frames, at any given time. It works by having the fusion of temporally obtained motions right through the first layers, so that once we progress over to the subsequent higher convolutional layers in the network, it has already accumulated spatio-temporal information at a global level.

2.2 3D Convolutions

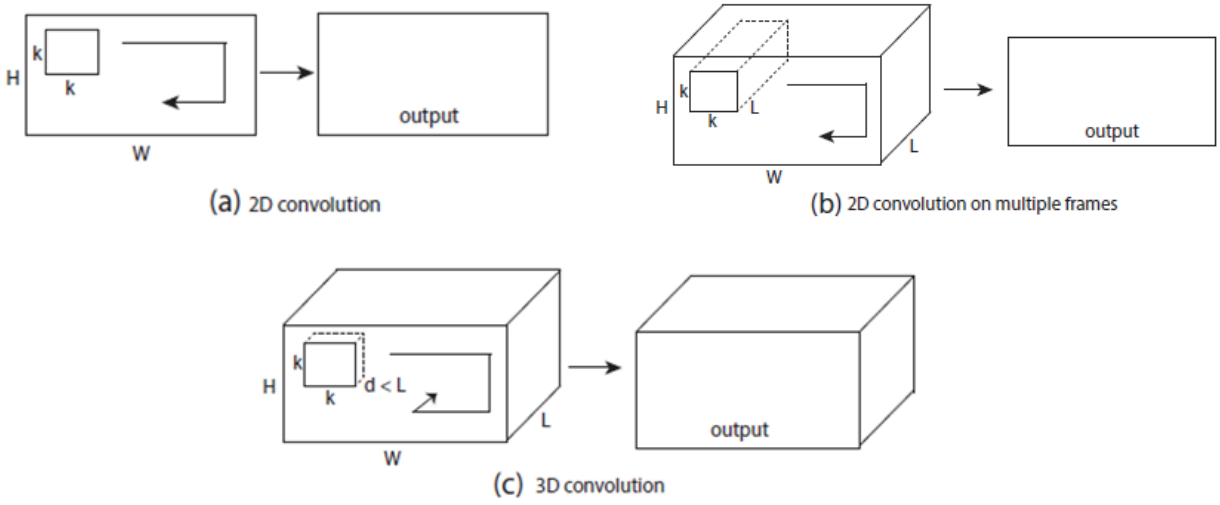


Figure 2.2.1: Comparison between 2D and 3D convolution.

While the previous work in activity recognition aims to delve deeper by taking into account of both spatial and temporal features of motion throughout a video, it is still relatively restricted when it comes to preserving its spatio-temporal features since all the convolution operations being done are in 2D.

It implies that even if 2D convolution is being done temporally on multiple frames, the resulting output will still be in the form of 2D activation map, as shown in Figure 2.2.1's case (b). This is because in 2D convolution, the depth of the filter that is being applied has to be the same as the depth of the input that is being fed, thereby making the convolution operates spatially on 2-dimensions, but not temporally throughout multiple frames. 3D convolution on the other hand leverages filters which having different depth than the ones in input image, thereby enabling the convolution being done throughout multiple frames in a video and resulting in 3-dimensional output volume. (Tran, et al., 2015), introduced *C3D* network which essentially relies on 3D convolution method in order to tackle the issues faced by the previous work. In fact, applying a 3D convolution across frames works better in preserving temporal information, as 2D convolution's temporal information will gradually diminish right after the initial convolution.

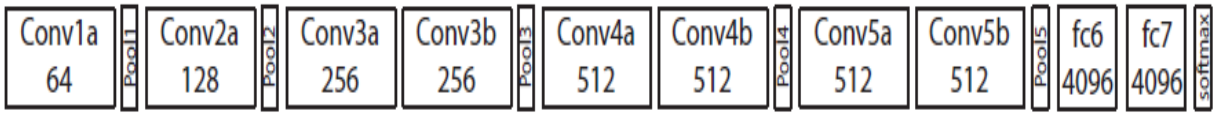


Figure 2.2.2: C3D network configurations.

In the same paper, (Tran, et al., 2015) also repurposed C3D as a refined feature extraction tool. Firstly, C3D network is being pre-trained on the Sports-1M datasets which contains 1.1 million videos of that can be categorized into 487 distinct sports. By doing so, it can be used to extract rich temporal features in UCF-Crime datasets via transfer learning, thereby potentially avoiding the need to train onto other UCF-Crime videos from scratch. To extract features, an input video is segregated into a number of clips (each containing a fixed predetermined number of frames, in this C3D's case is 16 frames), where appearances and motions will be encapsulated at fully-connected layer (fc6), since this *FC* layer progressively obtains high-level information from every element of previous layers. *C3D* is reportedly able to first detect spatial appearances in the first several frames, followed by tracking vital motion of said appearances subsequently. For instance, it

initially detects a person, and later on tracks the motion of aforementioned person doing specific kinds of activities, such as running or swimming.

2.3 Two-Stream Inflated 3D ConvNet (I3D)

I3D is complementary to the previously discussed *C3D* network. Instead of only extracting high-level features solely based on single spatial stream (RGB frames), I3D network also takes into account of temporal stream as well. Temporal stream comprised of optical flow frames, which extracts the motion flow of moving objects between contiguous frames along the horizontal and vertical axis – yielding a pair of optical flow frames.

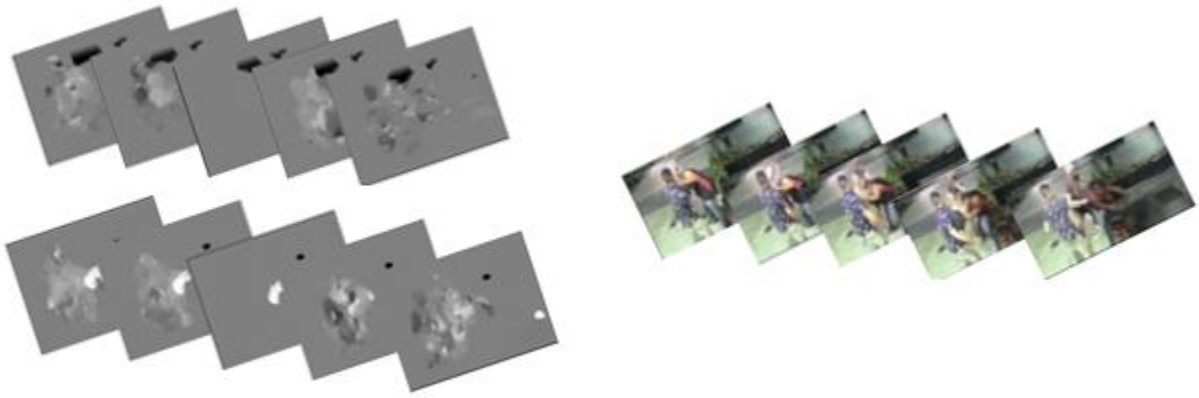


Figure 2.3.1: A pair of optical frames generated (left side) based on the consecutive RGB frames (right side).

While a single spatial-stream ConvNet is able to discern spatio-temporal patterns throughout video frames, temporal streams can be of a great aid in improving ConvNet's performance due to its explicit motion features captured through optical flows frames. In this paper, (Zisserman & Carreira, 2018) leverage the pre-trained 2D ConvNet and inflated it into 3D ConvNet by expanding every pooling kernels and filters, thereby "expanding" them by an extra dimension. The main take-away of this work is the effectiveness of leveraging pre-trained 2D ConvNet (on ImageNet dataset), and adapt it into spatio-temporal feature extraction tool as a 3D ConvNet via transfer learning, instead of training from scratch. This is helpful mainly because a conventional 3D ConvNet has way more weights and biases thanks to an extra kernel dimension, thereby making training more computationally expensive. For perspective, the 3D-ConvNet experimented by (Zisserman &

Carreira, 2018) contains 79 million parameters, while the two-stream I3D only contains 25 million parameters.

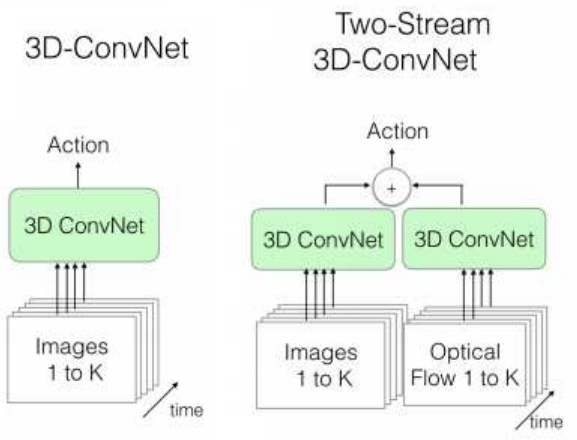


Figure 2.3.2: Side-by-side comparison between single-stream 3D *ConvNet* along with two-stream 3D-*ConvNet*.

Apart from that, I3D is capable of recognizing a wide array of activity and salient actions by pre-training it on Kinetics Human Action Video dataset, comprising of 400 human-action classes, spanning over 400 video clips for every class. In the context of detecting violent-related scenes in this project, **I3D network will be the foundation of our TAL's feature extraction** instrument.

2.4 Regional Convolutional 3D Network (R-C3D)

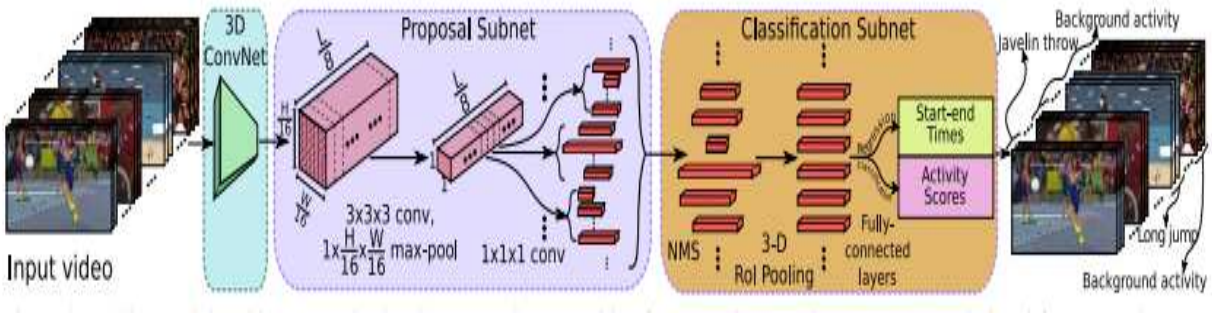


Figure 2.4.1: Architecture of R-C3D model.

Both C3D and I3D explored previously are primarily geared towards activity recognition tasks. They are capable of performing video classification, but not to the extent of temporally predict the

starting and ending timeframe where a particular action takes place in a video. To this end, (Xu, et al., 2017) came up with R-C3D model which works as follows: Feed an untrimmed input video into a 3D ConvNet feature extractor in order to yield feature map which holds spatio-temporal discriminating motions and actions. The extracted feature is then passed on to a **proposal subnet**. As its name suggests, this subnet will generate a number of proposal segments that may contain potential actions in them (proposals) based on pre-defined anchors. Localization task in video is tricky in nature, due to the difficulty in precisely predict the starting and ending boundary of an action, since actions have different durations where some lasts longer than others. Furthermore, actions could be located at either the beginning, the end or anywhere throughout a video clip. To resolve this matter, a number of *K* anchors with varying scales are incorporated to aid in the process of proposing localized actions throughout a video. As a matter of fact, during training, each of these anchor segments are assigned with binary labels – positive to indicate the presence of activity in the generated proposals, and negative to indicate otherwise. An anchor is assigned to positive label if its segment overlaps with ground truth's segment which exceeds a pre-defined threshold value. On the flip side, if an anchor segment has overlapping value below the threshold with all the ground truth's segment, said anchor will be tied to negative label. Hence, this subnet requires ground truth at temporal-level.

Since the proposals generated in proposal subnet has varying temporal length, they will be passed into the **classification subnet**, resulting in every proposals having fixed feature dimensions via ROI pooling layer, as shown in Figure 2.4.1. More importantly, this subnet carries out two main tasks: Classifying action categories as well as regressing the boundary of starting and ending time containing said action(s). Just as mentioned in proposal subnet, classification subnet also necessitates video-level label for each candidate proposals in order to acquire video-level classification prediction. All in all, R-C3D is capable of localizing the starting and ending timeframe containing an action, as well as predicting action classes for every localized segments.

2.5 Multiple Instances Learning (MIL)

While *R-C3D* is capable of localizing actions and classifying their categories, it is quite tedious since it necessitates ground truth at temporal-level during training. Compared with R-C3D, MIL is a form of weakly-supervised learning which does not require ground-truth at temporal-level.

MIL played an indispensable role for exploring the capabilities of weakly-supervised techniques since it served as the main backbone of our preliminary work (Chapter 3). In contrast to conventional fully-supervised learning that assign labels at instances, we only provide labelling at a group of instances using MIL – we refer this group as "bag". With reference to Figure 2.5.1, negative bag will only contain negative instances. Conversely, any positive bags is assumed to contain at the very least a single positive instance in order to easily distinguish positive bags apart from negative bags. In many applications, this concept works well since some of the causes for the labelling occurs at bag-level. A straightforward analogy would be the case of classifying cancerous organ – an organ will be labelled as positive if it contains any presence of cancer, regardless if it's in the form of cancerous cells/tissue, or the variation cancer stages.

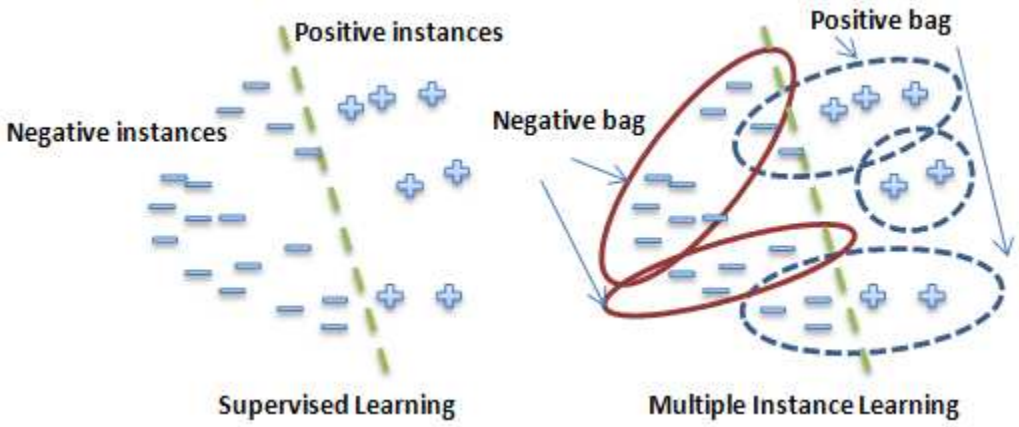


Figure 2.5.1: Core concepts of MIL.

Since our preliminary work is based on the pipeline propounded by (Sultani, et al., 2018), in-depth descriptions behind the intuition as well as implementation details of MIL will be further discussed in the following chapter 3.

2.6 AutoLoc

AutoLoc also shares the same modus operandi with MIL, in the sense that it is a form of weakly-supervised setting on activity detection task. During training, only the video-level label is provided along with extracted input features. (Shou, et al., 2018) configured AutoLoc into having two main branches: Classification branch and localization branch, as shown in Figure 2.6.1. After extracting discriminative features of untrimmed videos using pre-trained model, the feature sequences are

being fed into the classification branch. Firstly, this branch generates classification score for all the input snippets to indicate snippet's activation for *K* classes in total – this is referred to as Class Activation Sequences (CAS). Meanwhile, the extracted input features is being passed into the localization branch as well. This branch is responsible for generating anchor proposals directly in order to obtain prediction at temporal-level. Subsequently, aforementioned anchors generated would be regressed based on the anchor's center location and anchor's length (duration), resulting in an inner boundary (x_1 and x_2 corresponds to starting and ending snippets of inner boundary, respectively). Having obtained inner boundary, both $x_1$and x_2 will be inflated to obtain $X_1$and X_2, where X_1 precedes x_1 and x_2 precedes X_2.

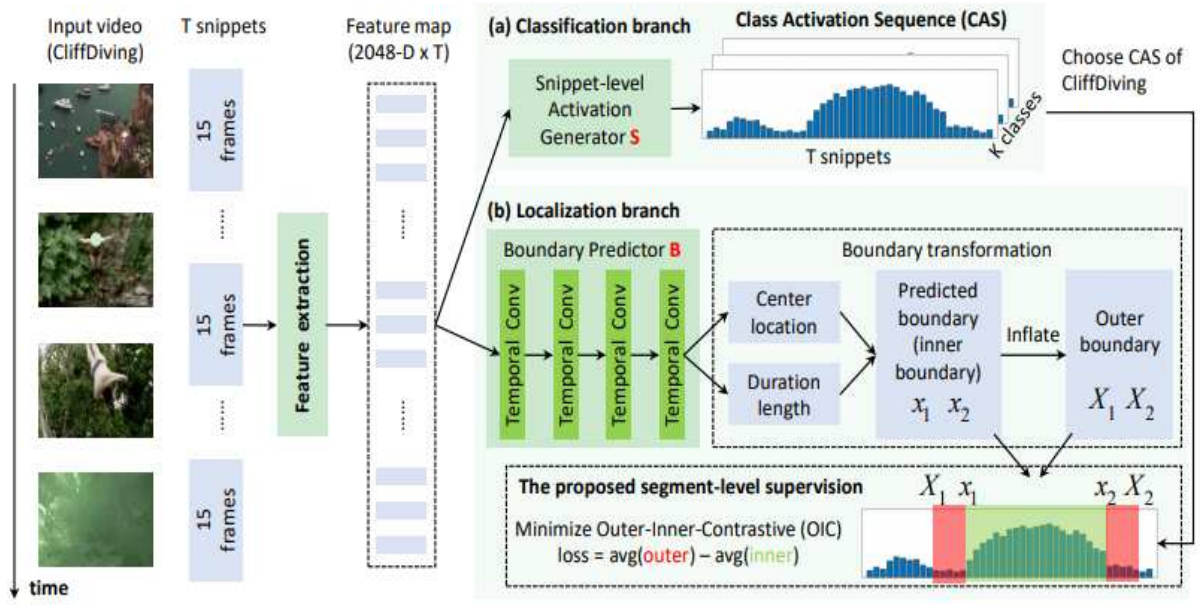


Figure 2.6.1: System architecture of AutoLoc.

Based on the CAS obtained from classification branch, only the ones similar to the video-level label is being chosen to supervise prediction at temporal-level. In order to achieve this feat, ***OIC* (Outer Inner Contrastive) Loss** is being introduced and it works as follows:

$$\boldsymbol{L_{OIC} = A_{outer}(\phi) - A_{inner}(\phi)}$$

where $A_{outer}(\phi)$ represents the average activation on surrounding outer area (denoted with red region on the CAS in Figure 2.6.1), while the inner area (denoted with green region on the CAS in

Figure 2.6.1) is referred to as $A_{inner}(\phi)$. The $A_{outer}(\phi)$ is obtained by inflating the $A_{inner}(\phi)$. The main gist of $\boldsymbol{L_{OIC}}$ is to ensure that the average activation on the inner area exceeds the ones in the outer area, because it's more likely that the inner boundary's activation is aligned with ground truth at temporal-level. Based on the above equation, the loss incurred by $\boldsymbol{L_{OIC}}$ will be minimal if the value of $A_{inner}(\phi)$ is larger than its outer counterpart. And by minimizing $\boldsymbol{L_{OIC}}$, it can be used to determine which temporal-level sequences is needed to supervise the training of the boundary predictor, for more accurate localized prediction.

CHAPTER 3: METHODOLOGY

This chapter is made up of two main sections – Section 3.1 details the workflow of MIL, followed by description of TAL pipeline in Section 3.2. In the case of MIL, it was trained and tested via Keras framework (running on top of Theano as back-end), while TAL was experimented using Pytorch framework. We utilized NVIDIA GTX1070ti GPU to accommodate the feature extraction and training process in both mentioned pipelines. (Liu, et al., 2019) experimented TAL paradigm using THUMOS'14 validation datasets, which is made up of 20 categories of action classes (soccer penalty, diving, high jump) for localizing and action classification tasks. This paper will examine both MIL and TAL technique onto the UCF-Crime dataset (as shown in Table 3) which comprises a wide multitude of violent scenes. MIL aims to generate higher anomaly scores for temporal segments containing crime scenes. Meanwhile for TAL, instead of providing the label at temporal-level in every video samples during training, the goal of TAL is to (1) localize violent actions apart from the background scenes, as well as (2) classifying these localized actions, given the label at video-level only during training.

Class	Training	Testing
Abuse	48	2
Arrest	45	5
Arson	41	9
Assault	47	3
Burglary	87	13
Explosion	29	21
Fighting	45	5

Class	Training	Testing
Road Accidents	127	23
Robbery	145	5
Shooting	27	23
Shoplifting	29	21
Stealing	95	5
Vandalism	45	5
Normal	800	150

Table 3: Distribution of UCF-Crime dataset based on category of videos.

3.1 Multiple Instances Learning (MIL)

Feature Extraction. To ensure clarity in terminologies for MIL's section, we will regard 'instances' to signify temporal segments, and 'bags' to denote videos (containing several temporal segments) in this following sub-section.

To start off, we need to obtain snippets for every input videos where each snippet consisted of 16 frames. We do so by firstly using a C3D network as the basis of our feature extraction tool, specifically the layer at FC6. Through transfer learning, the pre-trained C3D network will extract high-level features of every 16 consecutive frames as an individual clip. Since the duration of video varies from one another, this will result in some videos to contain smaller, or larger number of clips than the others. To remedy this differences, we will average out the extracted clips and normalize them via L2-normalization, thereby making every video to contain fixed number of segments (each segment comprised of 4096 dimension of features), regardless of the number of clips extracted earlier, as can be seen in following Figure 3.1.1.

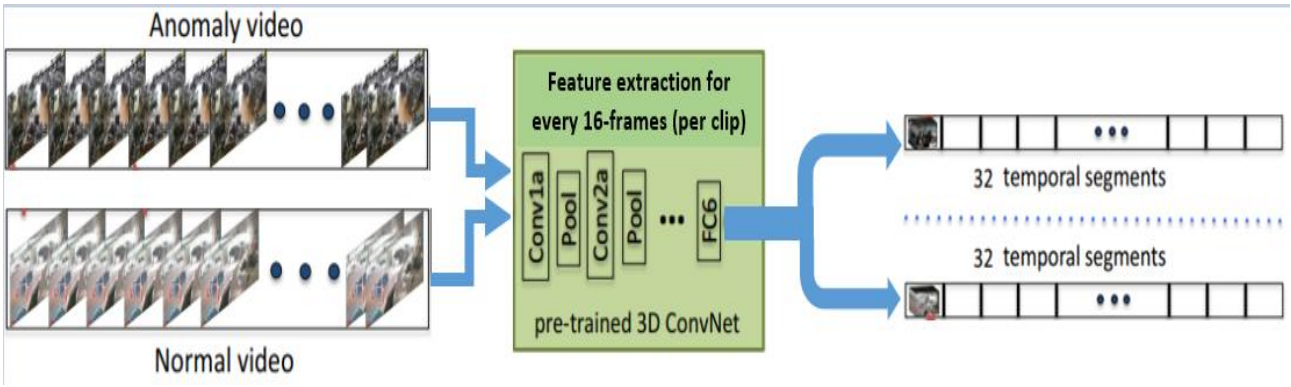


Figure 3.1.1 Flow of system prior to training.

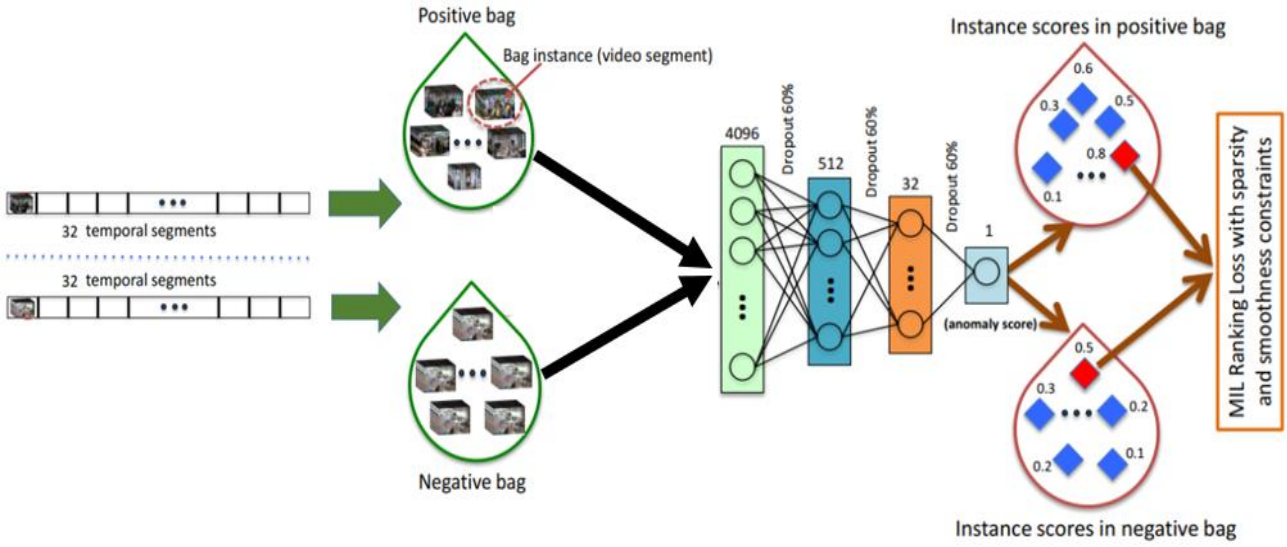


Figure 3.1.2 Rundown of system during training.

Training. Having obtained the averaged 32 segments per video, these segmented features will be segregated into training and testing set, as illustrated in Table 3. The training set will be made up of 2 different directories – abnormal directory which contains 810 samples of abnormal videos (videos containing violent scenes), and normal directory which contains 800 samples of normal videos with non-violent scenes. The remaining 290 samples will be populated in the test set. As for the network's architecture, it is made up of 3 fully-connected layers, with first, second and third layer having 512, 32 and 1 neuron respectively, as can be seen in Figure 3.1.2. The single neuron in the final layer of the model will generate a score for every single segment via linear regression. Dropout rate of 60% is enforced in order to avoid overfitting during training. Adagrad optimizers is being implemented, with its learning rate and epsilon value set to 0.01 and 1×10^{-8}, respectively. Training is being done for 20000 iterations, with each mini-batch having 60 randomly sampled videos (30 abnormal and another 30 normal videos will be fed into the network). The main key idea revolving the training of these features is the custom **MIL ranking loss function** proposed by (Sultani, et al., 2018), which will shortly be discussed in-depth. During training, only the ground-truth for each corresponding video will be assigned, instead of labelling the entire segment within each video. Regardless of the category for videos containing either abusive, explosive, or robbery scenes, these videos will be labelled as positive as long as they contain some violence scenes. Normal videos conversely will be labelled as negative during training.

In order for our model to learn and perform better in computing scores for each segments, it has to have a lower loss function. The main point of this loss function is to quantify on how bad our classifier or system performs, and its parameter will be updated at each iterations in order to minimize the loss being incurred. For this experiment, the loss function will be based on the (SVM) hinge-loss function:

$$L = \sum_{y_i \neq j} \max(0, 1 - s_{y_i} + s_j)$$

where s_{y_i} represents the score for the true class, while s_j denotes the score of the non-true class. In this experiment however, we are training our model via regression manner and not classification, since the output of our predicted model is not which category/class does our input video belongs to, but rather what are the computed scores for all the segments within our input video. We then reframe s_{y_i} to be s_{abn}^i, which denotes the score for every segment within an abnormal video. Likewise s_j will be s_{nor}^i, which indicates the score for all segments within a normal video, whereby superscript i denotes the segment within each video. Thus our hinge-loss function is represented as such:

$$L = \max\left(0, 1 - s_{abn}^i + s_{nor}^i\right).$$

Essentially, we want s_{abn}^i to have larger values than s_{nor}^i by a margin of 1. If not, then this loss function will incur some "penalty" on our model. Since we do not provide the ground-truth label at segment-level during training, we could not right away implement $s_{abn}^i > s_{nor}^i$ in our loss function, as we do not exactly know which segments in an abnormal video that truly contains scores higher than the ones in normal video's counterpart. A workaround to this matter is to enforce **ranking** by expressing $\max(s_{abn}^i)$ to represent the maximum score for each abnormal video, and $\max(s_{nor}^i)$ to signify the maximum score per normal video, without having to know which segments in either of these videos that contains the maximum score. Another reasoning of doing so is because $\max(s_{nor}^i)$ contains maximum score which are likely to be false positive (false alarm - where our model predicted the segment to contain violent scenes, where in actual truth it does not), while $\max(s_{abn}^i)$ contains maximum anomaly scores which are true positive in nature, since these predicted scores corresponds to segments containing violent scenes. Ultimately, our wish is

that the model is able to learn in outputting higher anomaly scores for segments within abnormal videos, and lower scores for segments within normal videos. By imposing ranking on maximum scores, we are able to set a distinction between true positive scores and false positive scores by further widening the gap between maximum scores in both abnormal and normal videos, such that:

$$\max_{i \in B_{abn}}(s_{abn}^{i}) > \max_{i \in B_{nor}}(s_{nor}^{i}).$$

When considering B_{abn} and B_{nor} to represent abnormal video and normal video respectively, our **MIL ranking loss function** now becomes:

$$L(B_{abn}, B_{nor}) = \max\left(0, 1 - \max_{i \in B_{abn}}(s_{abn}^{i}) + \max_{i \in B_{nor}}(s_{nor}^{i})\right).$$

Subsequently, we will also take into consideration of **temporal smoothness** and **sparsity** for our MIL ranking loss. Since our model outputs the scores for every segments within a video, the scores between contiguous segments should not be too far apart. In an abnormal video, if a violent scene occurs within several adjacent segments, then these neighboring segments should not be too uneven in terms of predicted scores, since segments in videos are continuous in manner. Aside from that, it is quite rare for a video to have a longer scenes of anomalies, especially true in the case of this experiment's dataset where the surveillance videos are untrimmed, thereby making the anomalies to be thinly scattered throughout the video. To emphasize on this, we shall include the sparsity term as well, thereby making our loss function to be:

$$L(B_{abn}, B_{nor}) = \max\left(0, 1 - \max_{i \in B_{abn}}(s_{abn}^{i}) + \max_{i \in B_{nor}}(s_{nor}^{i})\right) + \lambda \sum_{i}^{(n-1)} \left(s_{abn}^{i} - s_{abn}^{i+1}\right)^2 + \lambda \sum_{i}^{n}(s_{abn}^{i}) + \|W\|,$$

where W represents the weights of our model, $\lambda \sum_{i}^{(n-1)} \left(s_{abn}^{i} - s_{abn}^{i+1}\right)^2$ corresponds to the **temporal smoothness**, $\lambda \sum_{i}^{n}(s_{abn}^{i})$ denotes our **sparsity constraints**, λ is set to 8×10^{-5}, and finally value n is set to be the total number of segments per video. After having computed the loss of every training samples per batch via the equation above, the summed up loss acquired will be

averaged over the number of samples per batch, and compute gradient during backpropagation in order to perform gradient update on to the weights of our model.

Testing. The model and weight that has been trained will subsequently be used to predict the scores of every segment of unseen video during testing phase. The generated scores indicates the probability of violent scenes being detected for a particular segment. These scores will then be distributed across the entire frames once we determine the position of clips in their respective segment. Aside from that, the temporal annotations for each test sample will also be used as ground truth. These annotations contain the number of both starting and ending frame that indicates the presence of violent scenes. Having acquired both the predicted scores and ground truth, we shall be able to compute area under the ROC on frame-level.

3.2 Temporal Action Localization (TAL)

Data preparation and preprocessing. Prior to extracting features, the optical flows (temporal) and RGB (spatial) frames for every single video has to be acquired. Unlike RGB frames which can be generated almost instantly for all videos, the optical flows frames for every video has to be saved on the disk firstly, since it is very time consuming to generate them. The TVL1 Optical Flow algorithm will be utilized in extracting the temporal motion information between adjoining RGB frames. Spatial and temporal I3D networks that has been pre-trained on the Kinetics Human Action Video Dataset will be utilized in extracting high-level discriminative motion features for every non-overlapping snippets, with each snippet containing 16 frames. Since the default size of RGB frames and optical flow frames generated is 320 x 240 during prior to feature extraction, these input frames will be resized into 224 x 224 and subsequently fed into I3D network. With that, every input video will contain both spatial (X_{rgb}) and temporal (X_{flow}) extracted features, indicated by $X \in \mathbb{R}^{T \times D}$, where T is number of snippets per video and D stands for the feature dimensions per snippet.

Generating static clips. One key ideas in activity recognition for untrimmed videos lies in the capability of the system in **differentiating between violent actions and background scenes**.

Violent events tend to contain more vigorous actions or salient motions, compared to background events. To that end, we would be preparing static clips which comprises minimal intensity of actions, based on the optical flows frames. Only optical flow frames that belongs to the training set shall be included in the formation of static clips. Furthermore, not every training sample's static clips would be generated, as those which are too short or too long would be discarded. The resulting violent static clips along with those static clips generated from *Normal* (non-violent) videos shall then be grouped and categorized as a new class – *Background*. The ratio of selecting static clips is set as 30%, while *Normal* static clips are sampled at 50% rate.

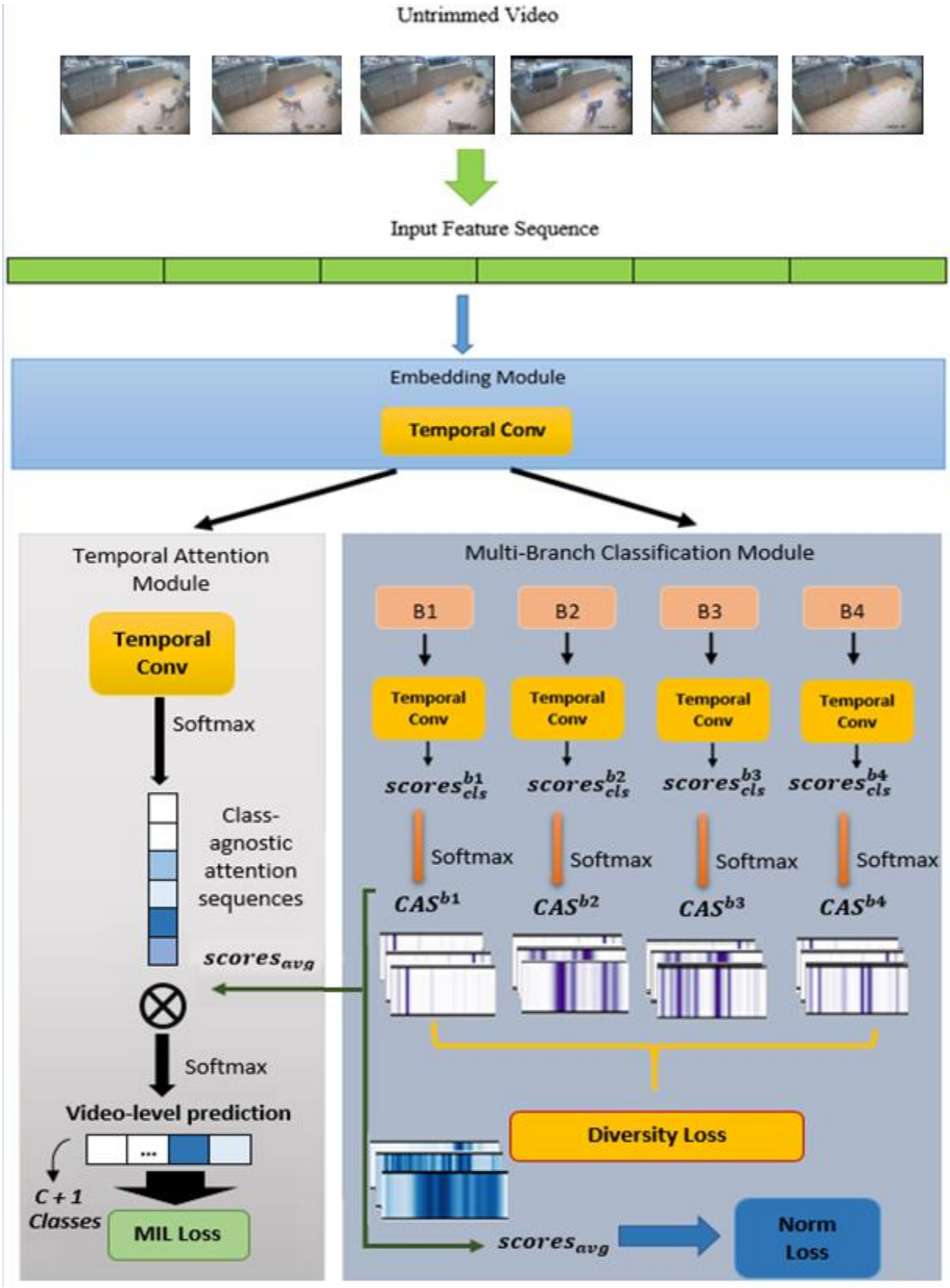


Figure 3.2.1: TAL architecture is mainly comprised of 3 main modules.

Training. Just as detailed in weakly-supervised MIL's training phase, we only supply ground-truth in the form of video-level label during training phase for 3 types of modalities: RGB, flow and both. For brevity's sake, we will regard extracted RGB and flow features, X_{rgb} and X_{flow} respectively as *X*. As for input features for *both* modality, X_{both}, it undergoes the same procedure during training as well, with the slight difference in that it stacks both X_{rgb} and X_{flow} along the feature dimensions axis.

During forward propagation, the extracted features *X* will firstly be fed into the embedding module. The **embedding module** will apply a 1D-convolution over the input features *X*, followed by ReLU activation layer, as follows:

$$\boldsymbol{X_{conv} = Weight_{emb} * X + bias_{emb}}$$

$$\boldsymbol{X_{emb} = \max(X_{conv}, 0)}$$

where $Weight_{emb}$, $bias_{emb}$ and $X_{emb} \in \mathbb{R}^{T \times F}$ denotes the weight and biases values for the embedding module, as well as the output of embedding module (where *T* signifies number of snippets, *F* indicates number of filters), respectively. X_{emb} encompasses temporal information from neighboring timeframe after it undergoes 1D temporal convolution. Subsequently, X_{emb} will be feed-forward into the **multi-branch classification module**. As its name implies, this module comprises of *K* number of classification branches, working in tandem. Each of these classification branches have exact similar structures, in such as way:

$$\boldsymbol{score_{cls}^{k} = Weight_{cls}^{k} * X_{emb} + bias_{cls}^{k}}$$

$$\boldsymbol{CAS^{k} = softmax(score_{cls}^{k})}$$

where $score_{cls}^{k} \in \mathbb{R}^{T \times (C+1)}$ represents the classification scores at *k*-th branch, while $Weight_{cls}^{k}$ and $bias_{cls}^{k}$ represents the weights and biases for classification module in the *k*-th branch as well. The raw classification scores, $score_{cls}^{k}$ is computed by feeding X_{emb} into a temporal convolutional layer. Note that since we have included an additional *Background* class through mining static clips, classification scores now amounts to $C + 1$ classes throughout *T* snippets in a video. In addition, CAS^{k} refers to Class Activation Sequence (CAS) at *k*-th branch after applying softmax onto the

classification scores, $score_{cls}^{k}$ – resulting in a probabilistic distributions for the activation of classes throughout a video.

The goal of having *K* parallel branches in the classification module is to resolve the issue of capturing a variety of different sub-actions throughout a video, in the hopes of capturing violent actions fully. In other words, for every *k*-th branch, we want each of them to yield different activations of classes. As a result, we implement a cosine similarity formulation:

$$\boldsymbol{cosine\_similarity} = \frac{\boldsymbol{CAS^i \cdot CAS^j}}{\| \boldsymbol{CAS^i} \| \times \| \boldsymbol{CAS^j} \|}$$

Based on the equation, cosine similarity measure the gap between CAS from *i*-th branch and CAS from *j*-th branch. The smaller the gap between CAS^i and CAS^j, the higher the cosine similarity. To put it simply, if both *i*-th and *j*-th branch in the classification module focuses on the same action parts in a video and therefore have almost very high similarities, it will incur a higher cosine similarity value. Hence, we can express this equation as a **diversity loss**, L_{div}:

$$\boldsymbol{L_{div}} = \frac{\boldsymbol{1}}{\boldsymbol{Z}} \sum_{\boldsymbol{c=1}}^{\boldsymbol{C+1}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{K-1}} \sum_{\boldsymbol{j=i+1}}^{\boldsymbol{K}} \frac{\boldsymbol{CAS_c^i \cdot CAS_c^j}}{\| \boldsymbol{CAS_c^i} \| \times \| \boldsymbol{CAS_c^j} \|}$$

$$\boldsymbol{L_{div}} = \boldsymbol{Weight_{div}} * \boldsymbol{L_{div}}$$

Z signifies the normalization factor, where $Z = \frac{1}{2}K(K-1)(C+1)$. $CAS_c^i \in \mathbb{R}^T$ denotes the CAS for a particular class *c* from *i*-th branch, while $Weight_{div}$ represents the diversity weight, with the value of 0.2. By plugging in the above functions, we have diversity loss that incurs higher loss if any pair of branches have very similar CAS values. By enforcing such formulation, we can expect that each branch in the classification module to generate higher activations on different parts or sub-actions across a single video. The goal of minimizing L_{div} is to further diversify the CAS generated across all *K* branches, by not ignoring other minor regions throughout the video. By establishing this first loss function, we can obtain the overall CAS by averaging the classification scores, $score_{cls}^{k}$ across the entire *K* branches:

$$\boldsymbol{scores_{avg}} = \frac{\boldsymbol{1}}{\boldsymbol{K}} \sum_{\boldsymbol{k=1}}^{\boldsymbol{K}} \boldsymbol{score_{cls}^{k}}$$

$$\boldsymbol{CAS_{avg} = softmax(scores_{avg})}$$

At this point, $CAS_{avg} \in \mathbb{R}^{T \times (C+1)}$ represents average CAS which encapsulates the different class activations throughout a video, across the average scores of all branches. Apart from that, we also have to ensure that none of the branch has more dominance than the others. To avoid a particular branch in dominating the overall CAS, the other branches has to have similar importance as well. Failure in doing so would likely render CAS_{avg} to solely depend on a single dominant branch, resulting in activations only on a single action. Thus, this is where the second loss function comes into the equation:

$$\boldsymbol{L_{norm} = \frac{1}{K(C+1)} \sum_{c=1}^{C+1} \sum_{i=1}^{K} | \, \| \, scores_c^i \, \| - \| \, scores_{avg_c} \, \| \, |}$$

$$\boldsymbol{L_{norm} = Weight_{norm} * L_{norm}}$$

$Weight_{norm}$ is assigned to value 0.2, $scores_c^i$ represents the scores for a particular class $\boldsymbol{c}$ from the i-th branch, while $scores_{avg_c}$ corresponds to the average scores (without softmax) of all branches for a particular class $\boldsymbol{c}$. The main principle in this **normalization loss** is such that it penalizes the model if the $scores_c^i$ deviates from $scores_{avg_c}$ by a large margin. This ensures that every branch shall carry equal importance in contributing to the overall average CAS. Subsequently, the output from embedding module, X_{emb} shall be fed into the **temporal attention module**:

$$\boldsymbol{\mathcal{A} = softmax(Weight_{att} * X_{emb} + bias_{att})}$$

where $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{\boldsymbol{T}}$ represents attention sequence after applying temporal convolutional operations on the embedded features, followed by softmax operations. The temporal attention's weight and bias are represented by $Weight_{att}$ and $bias_{att}$, respectively. The purpose of obtaining class-independent attention sequence, $\boldsymbol{\mathcal{A}}$ is that it captures on every distinctive action parts of a particular video regardless of violent classes, and yields class-agnostic parameters. Following this, we obtain video-level classification prediction through the weighted sum of attention sequence with average scores of all branches:

$$\boldsymbol{\rho = softmax\left(\sum_{t=1}^{T} \mathcal{A}_t * scores_{avg_t}\right)}$$

where $\rho \in \mathbb{R}^{C+1}$ stands for video-level prediction across all classes, including background class. To obtain MIL loss, we enforce Cross-Entropy loss as follows:

$$\boldsymbol{L_{mil} = -\sum_{c=1}^{C+1} y_c \log(\rho_c)}$$

where L_{mil} computes the distance between distributions of video-level classification predictions ρ and its corresponding video-level ground truth, y. Having established the third loss function, we can now incorporate all aforementioned loss functions into a single summation loss, $\boldsymbol{L_{sum}}$:

$$\boldsymbol{L_{sum} = L_{mil} + L_{div} + L_{norm}}$$

In training phase, the accumulated $\boldsymbol{L_{sum}}$ computed based on the output of forward passes will then be backpropagated from the back to earlier layers within each modules while computing the gradient with respect to the weights. In terms of the experimental settings and configuration, this experiment ran for 9000 iterations, with a batch size of 24. The dropout rate is set to be 0.5, with learning rate and weight decay value set to be 0.0004 and 0.001, respectively. In the embedding module, a total of 32 filters are being used, with each filter having the size of 1. As for the multi-branch classification module, 16 filters are being allocated, with every filter size set as 3, and the number of branches, *K* is assigned to 4. Finally in the temporal attention module, 16 filters are being used, with every filter having the kernel size of 1. Both the strides and paddings for every convolutional layer is set as 1.

Testing. In order for the model to be able to classify action class at each localized instances, we feed the input features into the model, which would generate the average Class Activation Sequence across all *K* branches, CAS_{avg} as well video-level classification prediction, ρ. However, at this stage the model would exclude the prediction of the *Background* class, and only consider violent classes *C*. To achieve the prediction of a localized actions, the model must be able to detect the starting and ending timeframe that bounds the localized action. Coupled with OIC loss function proposed by (Shou, et al., 2018), we also implemented it, but with a slight twist. In AutoLoc, the

OIC loss function is implemented during its training stage where it determines which predicted instances at segment-level is needed to supervise the boundary predictor. As an alternative, OIC loss is only implemented during this paper's testing phase for outputting confidence score for a localized predictions in regards to its predicted violent class:

$$\boldsymbol{Inner_{mean} = mean\left(CAS_{avg|start:end}\right)}$$

$$\boldsymbol{Outer_{mean} = mean\left([CAS_{avg|start-\vartheta:start}, CAS_{avg|end:end+\vartheta}]\right)}$$

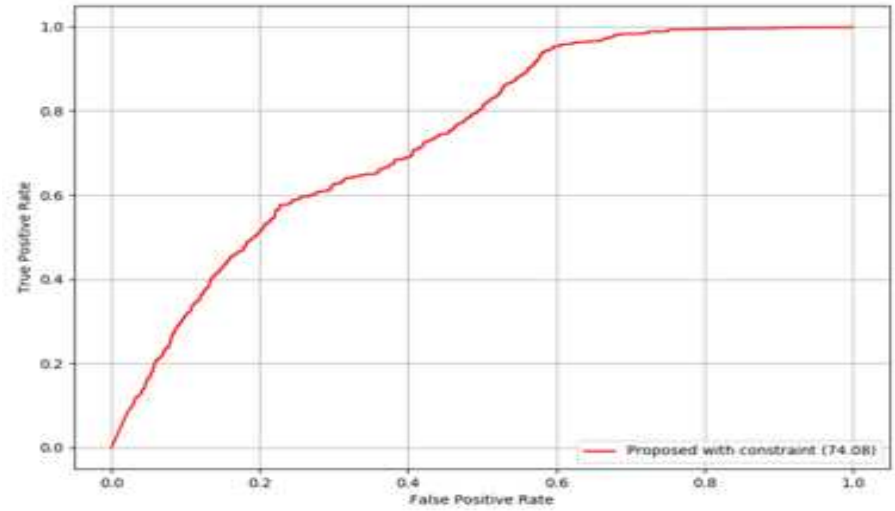
$$\boldsymbol{conf = Inner_{mean} - Outer_{mean} + \gamma \rho_c}$$

The $Inner_{mean}$ is acquired by getting the mean of CAS_{avg} right from the starting and ending time frame localized by the system. Let ϑ denote the inflation length, where $\vartheta = (start - end)/4$. We then obtain the surrounding activations of inner mean, $Outer_{mean}$ via the inflation length. Subsequently we computed $conf$, which signifies the confidence score of a particular localized boundary with respect to the probability score of its predicted class, ρ_c. All things considered at this point, for every test sample with ρ_c above or equal to the threshold value (set to 0.1), the model is capable of outputting $[start, end, conf, class]$ which represents starting time of localized action, ending time of localized action, confidence score as well as types of violent class pertaining to the particular localized action. In addition, four modalities will be evaluated based on AUC score: spatial stream, temporal stream, both (early fusion) as well as late fusion. The evaluation results for MIL and TAL shall be showcased in the next chapter.

CHAPTER 4: EVALUATION RESULTS

In order to gauge the performance between MIL and TAL, we evaluated their area under the ROC curve (AUC) at frame-level across all the testing samples in the first place. Following that, we also compare the detection and localization output for both weakly-supervised pipelines based on a number of selected testing set in order to gain some insights as to how both of them perform qualitatively. Subsequently, we presented ablation studies on TAL by visualizing the Class Activation Sequences (CAS) generated by the branches in TAL's Multi-Branch Classification module, as well as examining the significance of implementing static background clips prior to TAL's training phase.

4.1 Comparison between MIL and TAL



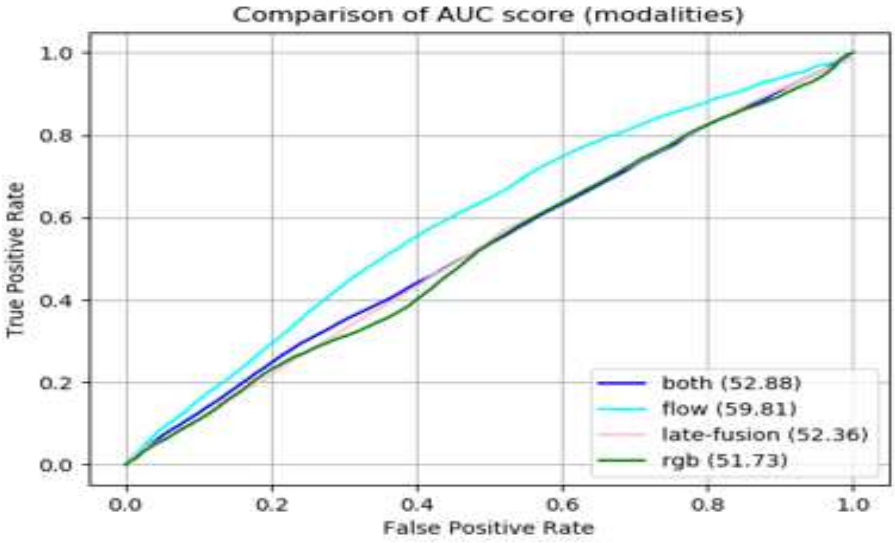


Figure 4.1.1: The ROC graphs obtained based on the evaluation of our model on test sets. The top-most ROC graph (denoted in red color) corresponds to MIL's, while the bottom chart depicts the TAL's evaluation result across all 4 modalities.

There are two folds in evaluating the performance of TAL's testing set. Firstly, we evaluated the area under the ROC curve (AUC) at frame-level for the entire testing samples. Unlike in the case of MIL's quantitative evaluation, we would only evaluate ROC for non-normal testing samples only. The ROC-AUC is an evaluation metric whereby it assess the performance of detection systems based on its ability in classification predictions. Essentially, the higher the AUC score, the better the system is in distinguishing between different classes. In the case of MIL, all the violence classes are treated as a singular positive class, while the normal videos are treated as negative class. MIL system managed to achieve 74.08% for the area under curve (AUC) by proposing the smoothness and sparsity constraints on our MIL loss function.

Detection System Type	**Modality**	**Area under ROC curve (AUC)**
MIL	-	**74.08**
TAL	Both	52.88
TAL	RGB	51.73
TAL	Flow	59.81
TAL	Late-Fusion	52.36

Table 4.1.1: Comparison of experimental setups in regards to AUC score.

Based on the above reported results, TAL has a very low quantitative score compared to MIL. This could potentially be attributed to the system's localization tasks which involves the classification of various different violence scenes, instead of simply grouping them as a single positive "bag" as discussed in MIL. Furthermore, investigation should be done in comparing between different feature extraction tools in order to further find out if different pre-trained models may have some effect on the high-level features being extracted from UCF-Crime dataset, prior to training.

Aside from quantitatively evaluate the performance of AUC, we further observe the contrast between MIL and TAL's performance onto several testing samples. The results on these test

samples will illuminate on how well both detection system works in the context of temporally detect violence actions. In these following figures, brown color indicates ground-truth annotations at frame-level (along the x-axis), while the green color chart/graph indicates prediction results. Specifically for TAL, it also outputs the predicted violent class for each of the modalities within the parenthesis. Best viewed in color.

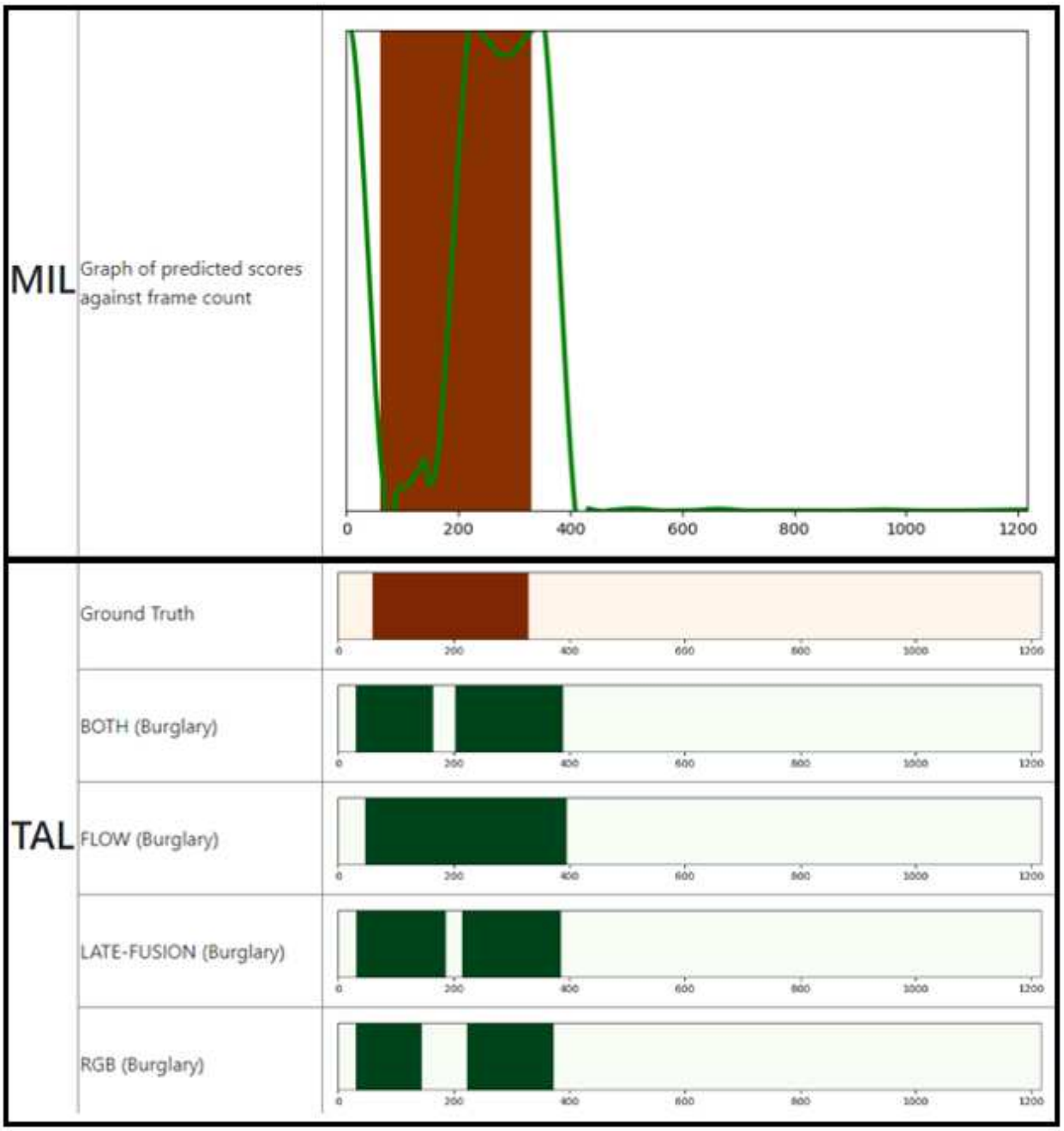


Figure 4.1.2: The contrast of detection results between MIL and TAL onto a test set containing Burglary scene.

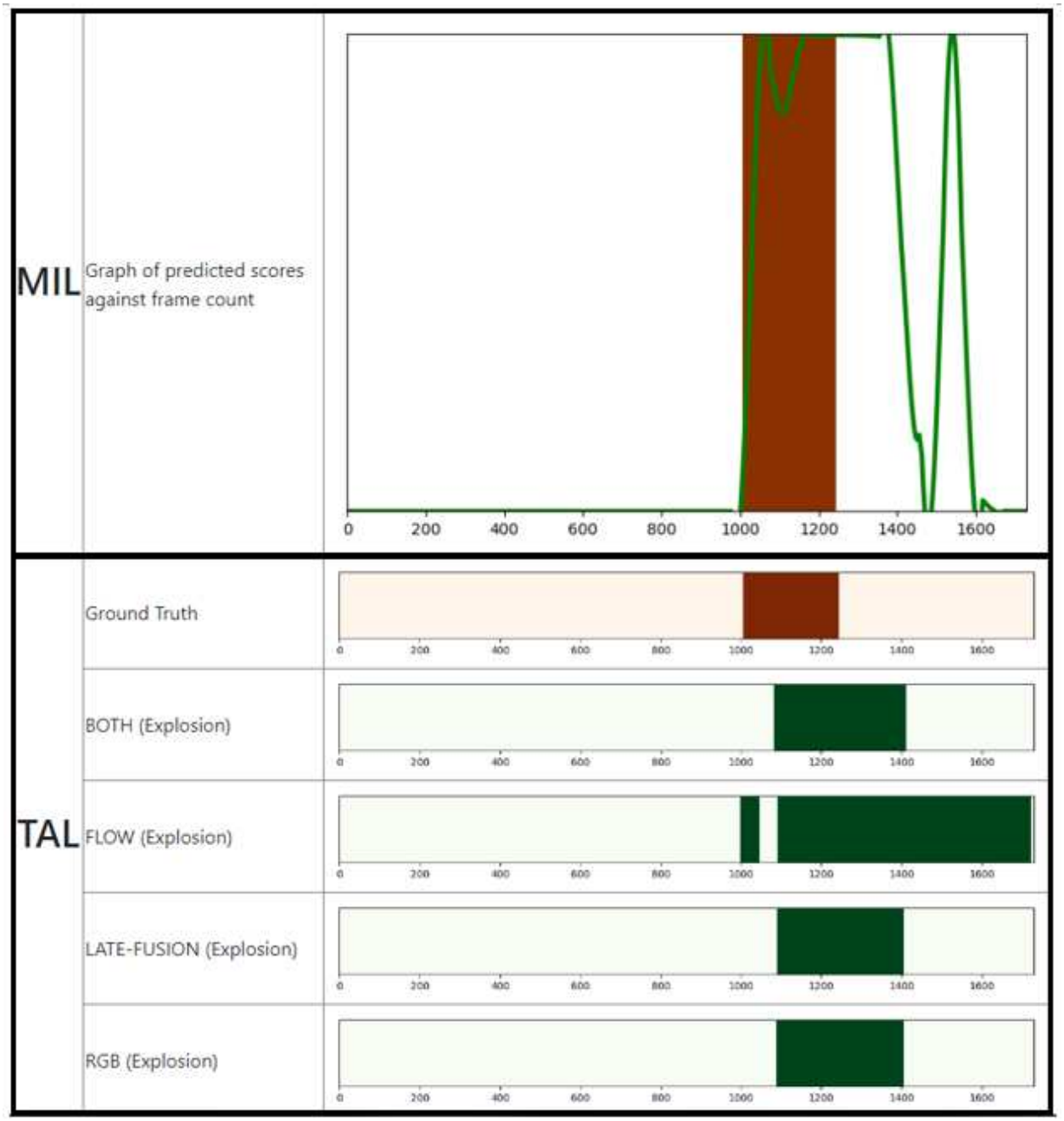


Figure 4.1.3: Comparison between MIL and TAL on a particular test set containing Explosion scene.

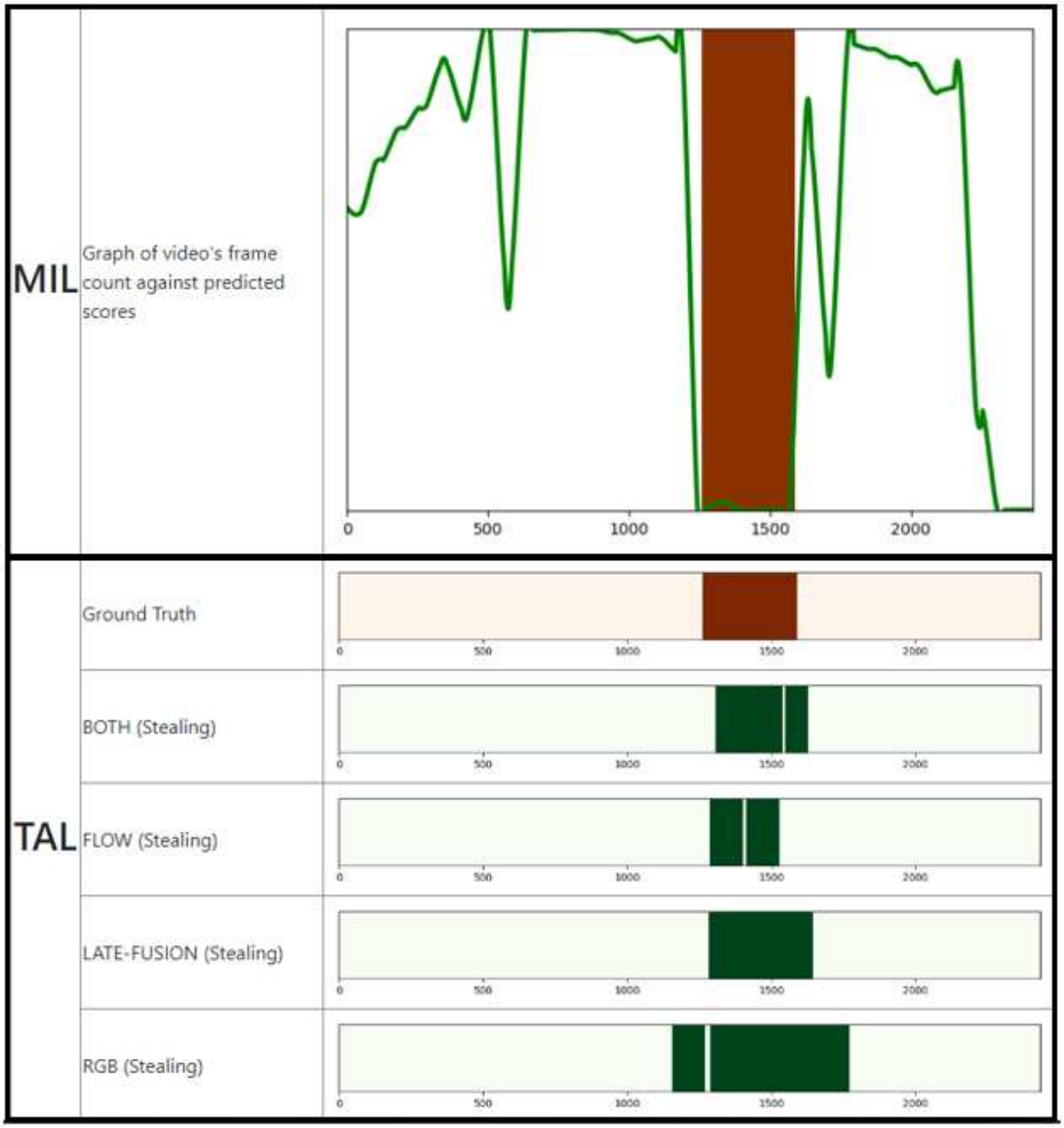


Figure 4.1.4: Comparison between MIL and TAL on a particular test set containing Stealing action.

4.2. Visualization of Class Activation Sequences (CAS)

By visualizing the CAS generated for each branch in TAL's Multi-Branch Classification module, we can conceptualize which temporal frames throughout a video that are responsible for invoking higher activations for a particular class. The coloured region in each branches denotes the degree of activation for a particular action class. Starting at Figure 4.2.1 to Figure 4.2.3, from top to bottom – Each horizontal grid charts in the ensuing figures corresponds to ground truth (GT), full localization prediction of TAL (Full), average CAS of all branches (Average), followed by CAS in each individual branch (Branch 1 to Branch 4). Best viewed in color.

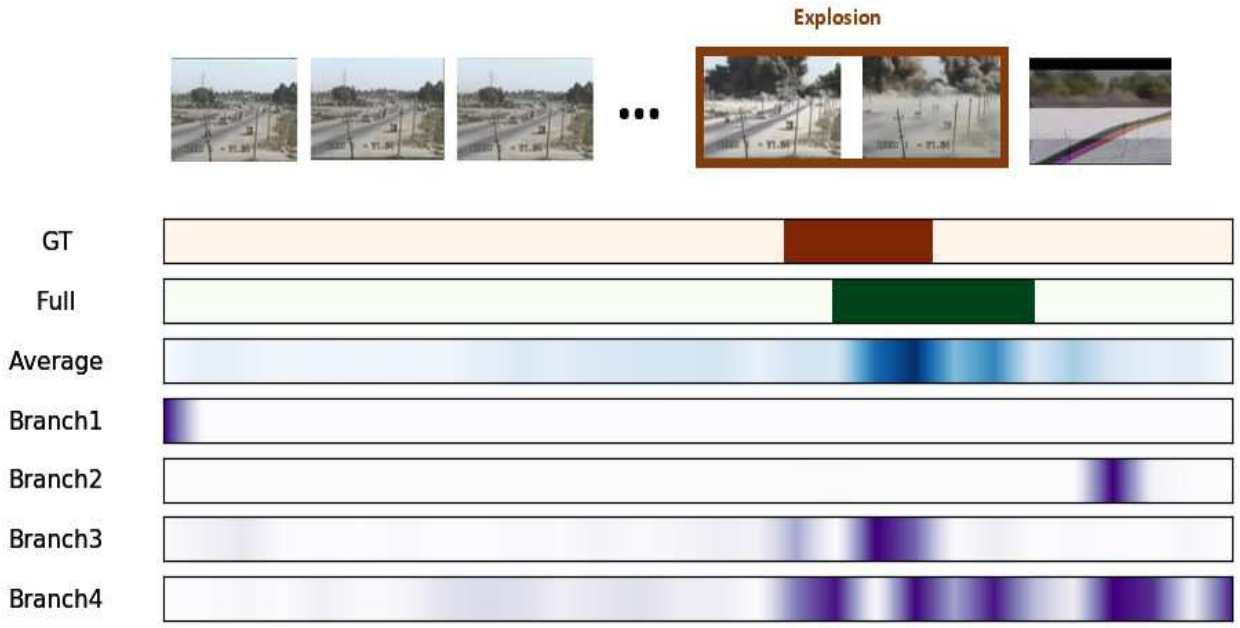


Figure 4.2.1: Visualization on a test sample containing explosion scene.

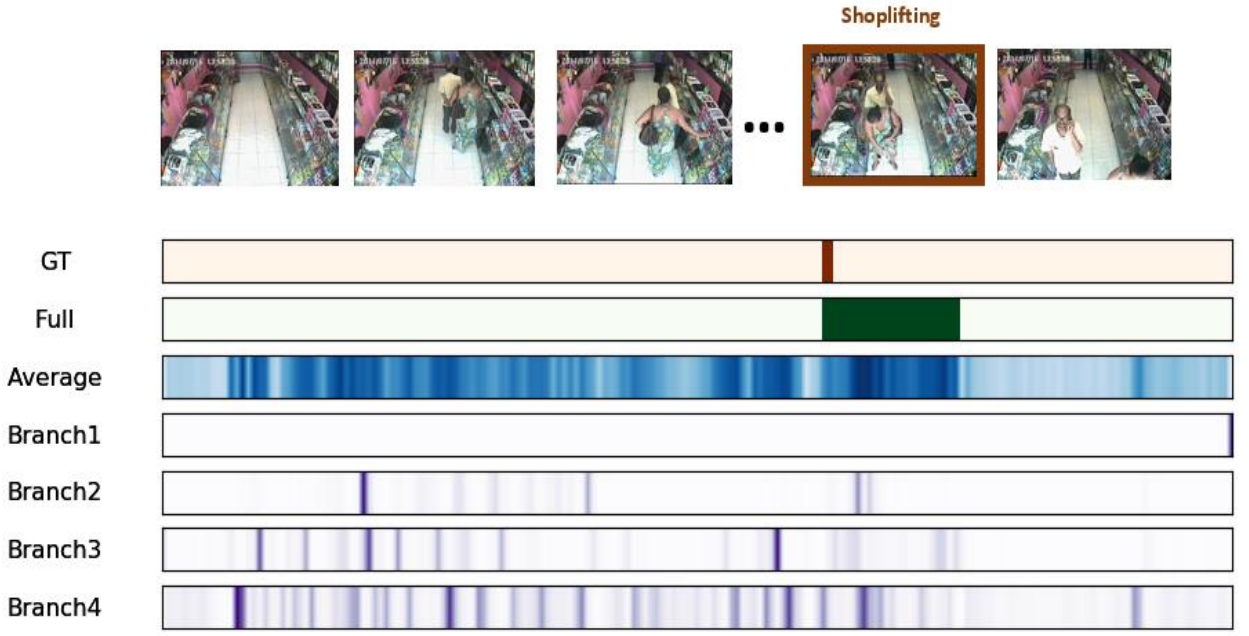


Figure 4.2.2: Visualization on a test sample containing shoplifting activity.

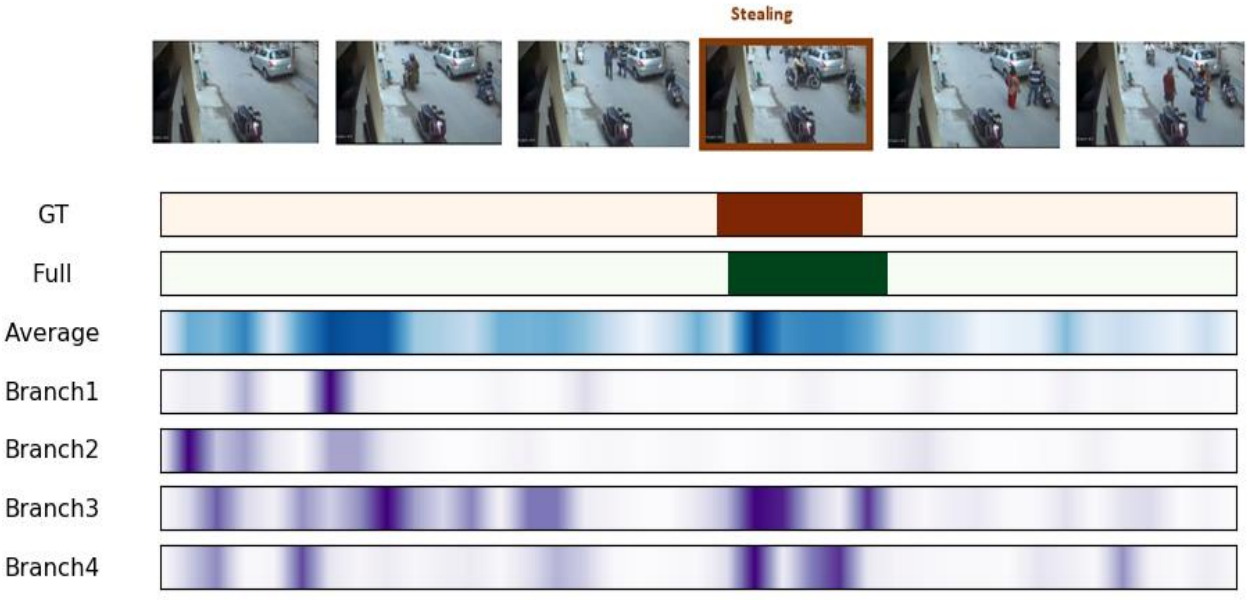


Figure 4.2.3: Visualization on a test sample containing stealing scene.

4.3. Analysis on Static Clips

In this section, we take a closer look at the performance of localized prediction in regards to implementing with and without static clips in TAL's experimentation. Two observations were being made, with each observation corresponds to a particular video in the test set. Based on both ensuing tables, localized detections without incorporating static clips generation tends to localized longer predicted boundary, largely because the model have a hard time distinguishing between distinct violent actions with its surrounding background noise. The absence of *Background* class from not including static clips confuses the model and thereby causing models to even include some fragments of unrelated background events as violent. It is also worthwhile to note that the AUC scores across all modalities which implemented static clips in TAL are a tad higher than its counterpart (without static clips) as shown in Table 4.3.3.

Observation I:

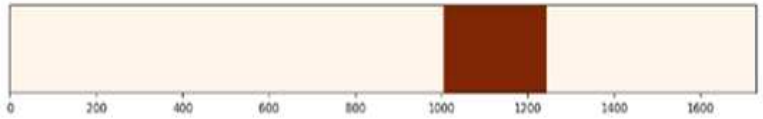
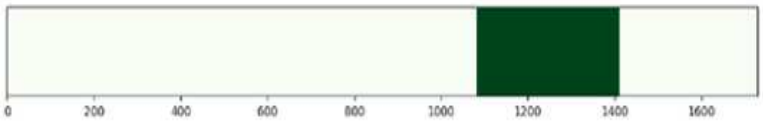
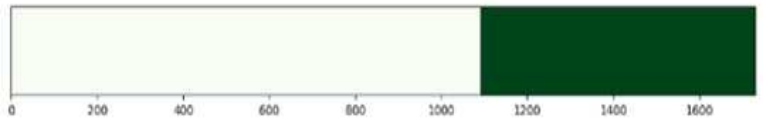
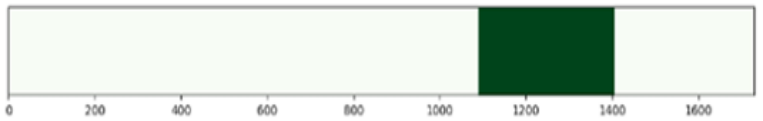
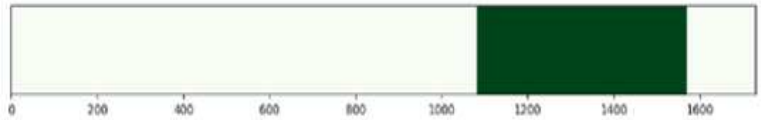
	**GT**	
	**Both**	**With static clips**
	**Both**	**Without static clips**
	**Late-Fusion**	**With static clips**
	**Late-Fusion**	**Without static clips**

Table 4.3.1: Table above depicts the performance of TAL model in regards to availability of static clips.

Observation II:

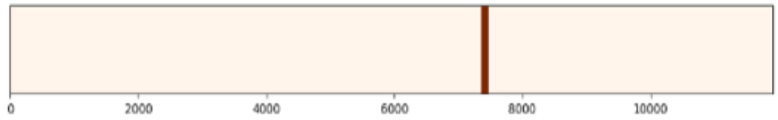
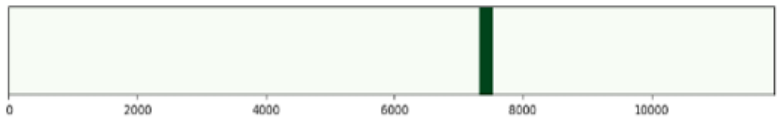
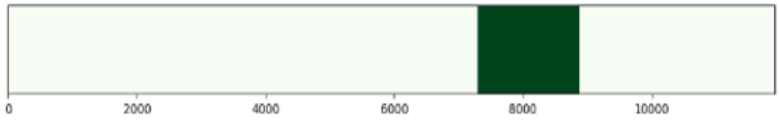
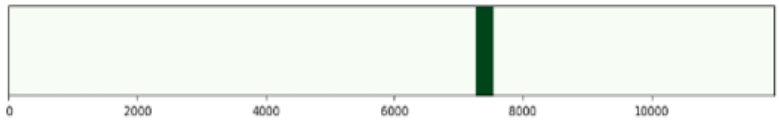
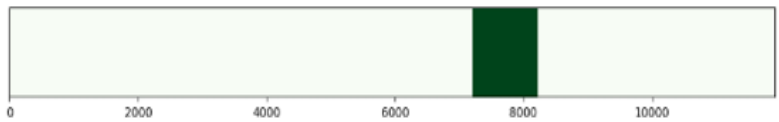
	**GT**	
	**Both**	**With static clips**
	**Both**	**Without static clips**
	**Late-Fusion**	**With static clips**
	**Late-Fusion**	**Without static clips**

Table 4.3.2: Another observation on localized events.

Modality	Area under ROC curve (AUC)
Both	52.88
RGB	51.73
Flow	59.81
Late-Fusion	52.36
Both *	49.18
RGB *	50.60
Flow *	56.90
Late-Fusion *	49.41

Table 4.3.3: Comparison of AUC between modalities which implemented training with static clips and without static clips. The * notation denotes modalities which did not include static clips for TAL's training.

CHAPTER 5: CONCLUSION

In this paper, we demonstrated the capability of predicting violent-related concepts in general by using weakly-supervised setting. This project gives us an insight on the minimal amount of supervision needed by providing annotations at video-level, and it is especially paramount in mitigating the laborious task of providing annotations at segment-level. This goes to show the robustness of both MIL and TAL paradigm when it comes to violent-scene detection even with the absence of temporal-level ground truth. MIL is limited to assigning higher anomaly scores temporally where segments containing violent-related concepts, while trying to suppress and minimize anomaly scores for irrelevant background scenes. On the flip side, TAL is able to detect the starting and ending boundary containing a violent scene, as well as classifying the violent categories pertaining to the predicted boundary.

As a further matter, the evaluation results in TAL leaves a lot to be desired, due to its low AUC in comparison with MIL's AUC. This means that while TAL is capable of precisely localize and classify violent categories for several testing samples, it has weaker detection performance and misclassify a lot of testing samples as a whole. Alternatively, the TAL paradigm can be a great foundation in further improving its evaluation benchmark, by possibly incorporating MIL's "bag" formulation to train the model in contrasting between violence actions and normal actions, instead of solely relying on mining static clips.

BIBLIOGRAPHY

Chandola, V., Banerjee, A. & Kumar, V., 2009. *Anomaly Detection : A Survey.* s.l., Association for Computing Machinery, pp. 1-72.

Gracia, I. S., Suarez, O. D., Garcia, G. B. & Kim, T.-K., 2015. *Fast Fight Detection.* [Online] Available at: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0120448 [Accessed 18 July 2019].

Hassner, T., Itcher, Y. & Kliper-Gross, O., 2012. *Violent Flows: Real-Time Detection of Violent Crowd Behavior.* [Online] Available at: https://www.openu.ac.il/home/hassner/data/violentflows/violent_flows.pdf [Accessed 20 June 2019].

Karpathy, A. et al., 2014. *Large-scale Video Classification with Convolutional Neural Networks.* [Online] Available at: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/42455.pdf [Accessed 15 July 2019].

Liu, D., Jiang, T. & Wang, Y., 2019. *Completeness Modeling and Context Separation for Weakly Supervised Temporal Action Localization.* [Online] Available at: http://openaccess.thecvf.com/content_CVPR_2019/papers/Liu_Completeness_Modeling_and_Context_Separation_for_Weakly_Supervised_Temporal_Action_CVPR_2019_paper.pdf [Accessed 10 December 2019].

Mohammadi, S., Kiani, H., Perina, A. & Murino, V., 2015. *Violence Detection in Crowded Scenes using Substantial Derivative.* [Online] Available at: http://www.hamedkiani.com/uploads/5/1/8/8/51882963/camera_ready_approved_final.pdf [Accessed 25 June 2019].

Shou, Z. et al., 2018. *AutoLoc: Weakly-supervised Temporal Action Localization in Untrimmed Videos.* [Online] Available at: https://arxiv.org/pdf/1807.08333.pdf [Accessed 22 December 2019].

Sultani, W., Chen, C. & Shah, M., 2018. *Real-world Anomaly Detection in Surveillance Videos.* [Online] Available at: https://arxiv.org/pdf/1801.04264.pdf [Accessed 25 May 2019].

Tran, D. et al., 2015. *Learning Spatiotemporal Features with 3D Convolutional Networks.* [Online] Available at: https://arxiv.org/pdf/1412.0767.pdf [Accessed 6 July 2019].

Xu, H., Das, A. & Saenko, K., 2017. *R-C3D: Region Convolutional 3D Network for Temporal Activity Detection.* [Online] Available at: https://arxiv.org/pdf/1703.07814.pdf [Accessed 6 March 2020].

Zisserman, A. & Carreira, J., 2018. *Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset.* [Online] Available at: https://arxiv.org/pdf/1705.07750.pdf [Accessed 15 March 2020].

POSTER

UNIVERSITI TUNKU ABDUL RAHMAN

Faculty of Information and Communication Technology

DETECTION OF ROBBERY-RELATED CONCEPTS USING DEEP LEARNING

INTRODUCTION

Precise detection on the presence of violent scenes is getting more crucial now more than ever, especially in deep learning. In this paper, what we are interested in is the system's capability in predicting the violent category of the input video, as well as localizing the starting and ending timeframe of the violent event.

- **Primary problem**: Impracticality of training phase which involves untrimmed videos**.**
- **Secondly**: Difficulty in predicting a particular violent scene fully, without annotations at temporal-level.
- **Another hindrance**: Scarcity of a wide array of crime scenes samples.

OBJECTIVES

- **To develop a Multiple Instances Learning (MIL) violence detection system.**
- **To develop a Temporal Action Localization (TAL) violence detection system.**
- **To juxtapose the performances of MIL and TAL paradigms.**

METHODOLOGY

- MIL paradigm: Re-purposed crime videos as 'positive' bags, normal videos as 'negative' bag.
- TAL formulation: Find diverse Class Activation Sequences (CAS) throughout a video via 3 modules.

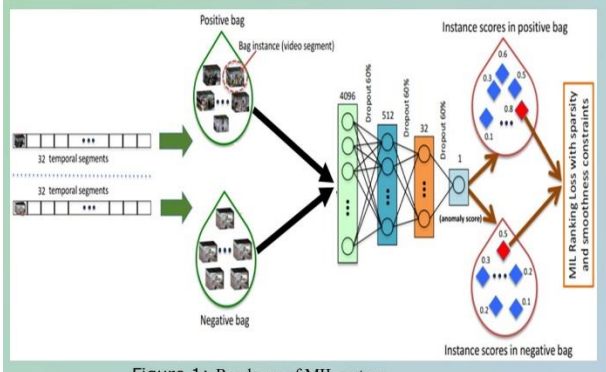


Figure 1: Rundown of MIL system.

EVALUATION ANALYSIS

Table 1: Performance of MIL and TAL based on area under the ROC curve (AUC) at frame-level.

MIL	TAL
74.08	RGB: 51.73
	FLOW: 59.81
	BOTH: 52.88
	LATE-FUSION: 52.36

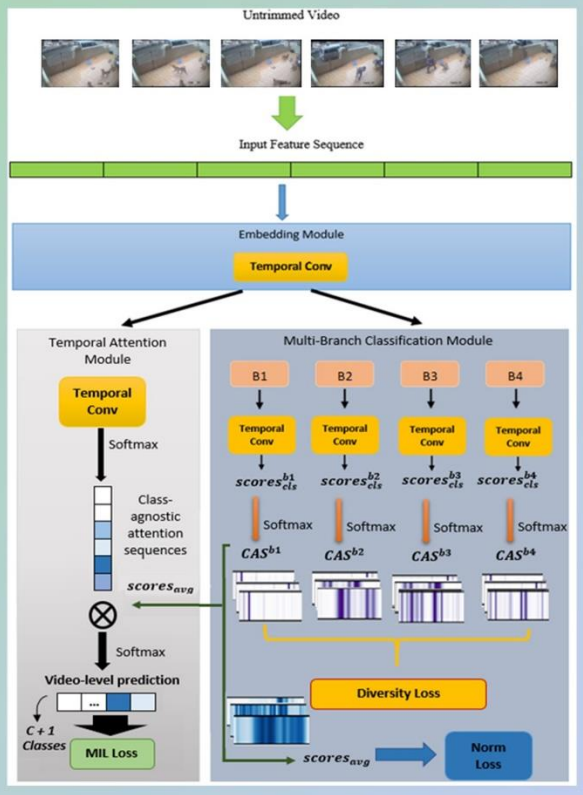


Figure 2: TAL paradigm for detecting robbery related concepts

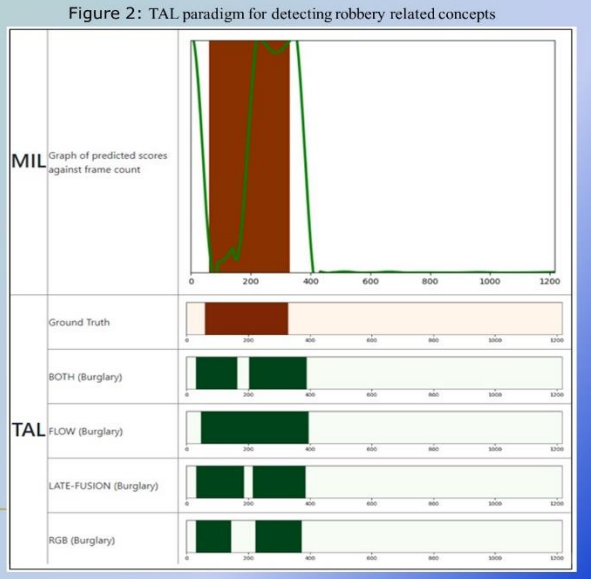
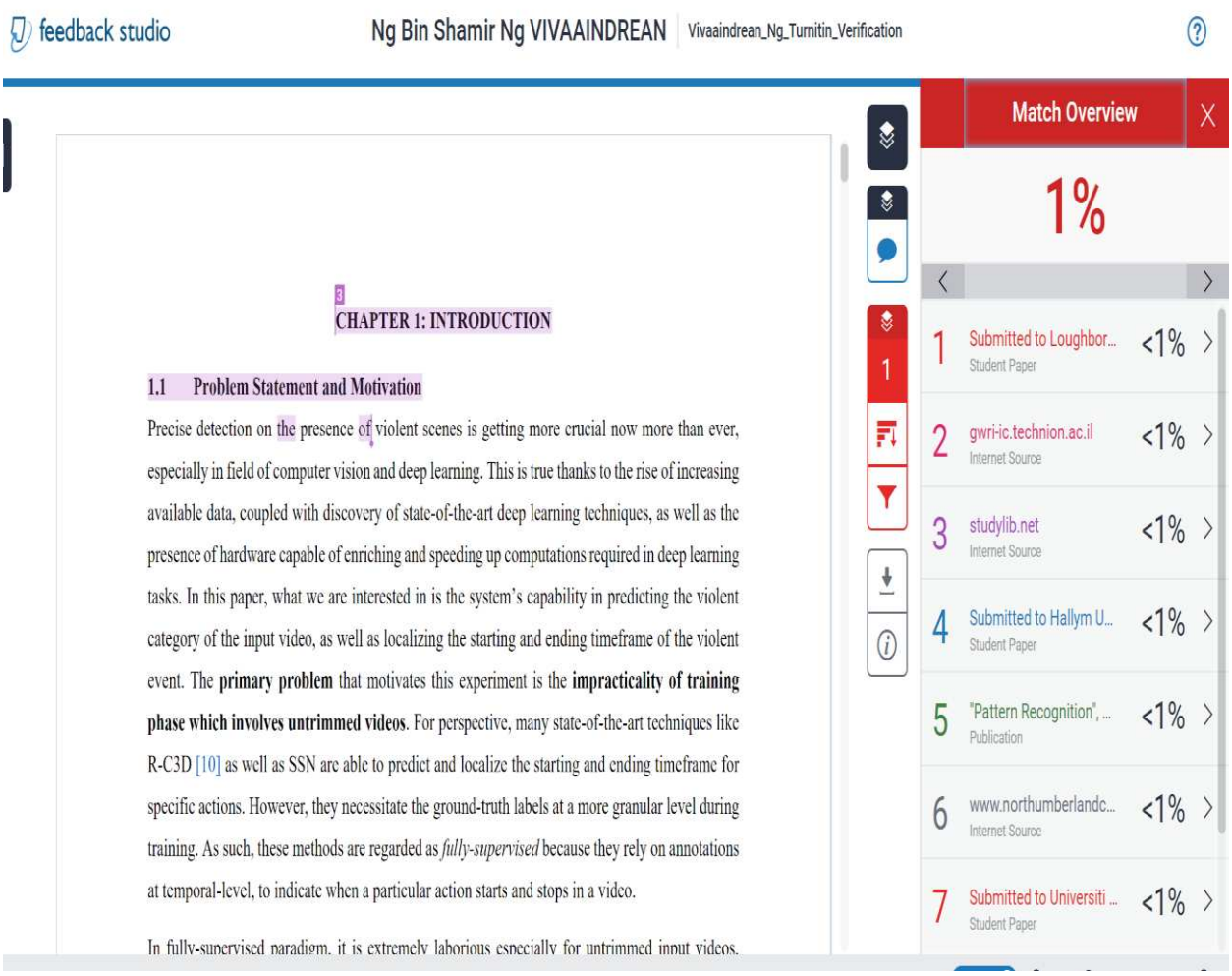


Figure 3: Comparison between MIL (top) vs TAL (bottom)

Student: Vivaaindrean Ng bin Shamir Ng *Supervisor: Dr. Tan Hung Khoon* *Moderator: Dr. Ooi Boon Yaik*

PLAGIARISM CHECK RESULT



Vivaaindrean_Ng_Turnitin_Verification

ORIGINALITY REPORT

1%	0%	0%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Loughborough University Student Paper	<1%
2	gwri-ic.technion.ac.il Internet Source	<1%
3	studylib.net Internet Source	<1%
4	Submitted to Hallym University Ilsong Memorial Library Student Paper	<1%
5	"Pattern Recognition", Springer Science and Business Media LLC, 2020 Publication	<1%
6	www.northumberlandcounty.ca Internet Source	<1%
	Submitted to Universiti Teknologi Malaysia	1

turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Ng Bin Shamir Ng VIVAAINDREAN
Assignment title:	FYP submissions
Submission title:	Vivaaindrean_Ng_Turnitin_Verificat...
File name:	17ACB00362_FYP2.docx
File size:	1.78M
Page count:	36
Word count:	8,101
Character count:	44,163
Submission date:	21-Apr-2020 11:21PM (UTC+0800)
Submission ID:	1302892911

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date:	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	VIVAAINDREAN NG BIN SHAMIR NG
ID Number(s)	1700362
Programme / Course	CS
Title of Final Year Project	DETECTION OF ROBBERY-RELATED CONCEPTS USING DEEP LEARNING

Similarity	**Supervisor's Comments** **(Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 1 % **Similarity by source** Internet Sources: 0 % Publications: 0 % Student Papers: 1 %	
Number of individual sources listed of more than 3% similarity: 0	
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Tan Hung Khoon

Date: 22 April 2020

_____-_____

Signature of Co-Supervisor

Name: -

Date: -

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	17ACB00362
Student Name	Vivaaindrean Ng bin Shamir Ng
Supervisor Name	Dr. Tan Hung Khoon

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Front Cover
√	Signed Report Status Declaration Form
√	Title Page
√	Signed form of the Declaration of Originality
√	Acknowledgment
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result – Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.  _____ (Signature of Student) Date: 21/4/2020	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction.  _____ (Signature of Supervisor) Date: 22/4/2020