Incident Detection in Traffic Video

Project-I (EC47003) report submitted to

Indian Institute of Technology Kharagpur

in partial fulfilment for the award of the degree of

Bachelor of Technology

in

Electronics and Electrical Communication Engineering

by

Ashutosh Naik

(20EC39049)

Under the supervision of

Prof. Pabitra Mitra

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

Autumn Semester, 2023

November 25, 2023

DECLARATION

I certify that

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Date: November 25, 2023 (Ashutosh Naik)
Place: Kharagpur (20EC39049)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Incident Detection in Traffic Video**" submitted by **Ashutosh Naik** (Roll No. 20EC39049) to Indian Institute of Technology Kharagpur towards partial fulfilment of requirements for the award of degree of Bachelor of Technology in Electronics and Electrical Communication Engineering is a record of bona fide work carried out by him under my supervision and guidance during Autumn Semester, 2023.

Pabitra Mitra

Prof. Pabitra Mitra
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721302, India

Date: November 25, 2023

Place: Kharagpur

Abstract

Name of the student: **Ashutosh Naik** Roll No: **20EC39049**

Degree for which submitted: **Bachelor of Technology**

Department: **Department of Computer Science and Engineering**

Thesis title: **Incident Detection in Traffic Video**

Thesis supervisor: **Prof. Pabitra Mitra**

Month and year of thesis submission: **November 25, 2023**

Understanding the environment is critical for Autonomous Vehicles (AVs). This report guides us to an online system, named MOVAD, designed to promptly respond to emerging anomalies around AVs using videos from a dash-mounted camera. MOVAD comprises two key modules: a Short-Term Memory Module utilizing a Video Swin Transformer (VST) to extract information about ongoing actions, and a Long-Term Memory Module, integrated into the classifier, which considers remote past information and action context through a Long-Short Term Memory (LSTM) network. MOVAD not only excels in performance but also stands out for its clear and modular architecture. Trained end-to-end exclusively with RGB frames and minimal assumptions, MOVAD is easily implementable and customizable. The system's effectiveness was assessed on the challenging Detection of Traffic Anomaly (DoTA) dataset, featuring dash-mounted camera videos capturing various accident scenarios.

Acknowledgements

I would like to thank my Thesis supervisor Prof. Pabitra Mitra for directing, guiding, and supporting me throughout the project.

Contents

List of Figures

Chapter 1

Literature Review

1.0.1

In the pursuit of enhancing the safety of Autonomous Vehicles (AVs), previous researchers have delved into the intricate domain of video anomaly detection, primarily leveraging the rich information provided by cameras in AVs. The challenges they encountered reflect the nuanced nature of real-world traffic scenarios and the complexities associated with defining and detecting anomalies.

One prominent avenue of exploration has been the use of Convolutional AutoEncoders (ConvAE) [1], a class of models trained solely on normal frames with the objective of frame reconstruction. While these models show promise in learning normal patterns, their sensitivity to the frequency of anomalies and the need for additional post-processing techniques pose significant challenges. The inherent difficulty lies in striking the right balance, as an overemphasis on anomaly frequency might lead to false positives, while neglecting it could result in overlooking crucial anomalous events.

Authors in [2, 3] proposed the use of Convolutional LSTM AutoEncoder, focusing on encoding both appearance and motion. Despite their efforts to capture temporal dependencies, these models face challenges in distinguishing between normal variations and true anomalies. The dynamic nature of traffic scenes and the variability

in normal traffic patterns make it challenging to discern anomalies solely based on appearance and motion.

AnoPred [4] introduced a multi-task loss approach for Video Anomaly Detection (VAD), incorporating image intensity, optical flow, gradient, and adversarial losses. While effective in certain contexts, AnoPred was primarily designed for video surveillance scenarios. The transition to videos acquired from within moving vehicles introduces a higher degree of dynamism and unpredictability, making anomaly prediction more challenging. The reliance on supplementary input information, such as optical flow and bounding boxes, further limits the applicability of such models in the context of AVs.

STFE [5] and FOL [6] proposed two-stage detectors, incorporating features like Histogram of Optical Flow (HOF) and ordinal features of frames to encode temporal and spatial relationships. However, their dependency on auxiliary information beyond RGB frames, such as optical flow and ego-motion information, restricts their adaptability to the diverse scenarios encountered by AVs. The limitations become evident when considering the presence of actors in the Field of View (FOV) and the need for additional input data beyond raw RGB frames.

In contrast, TRNmodel [7] couples action detection with temporal dependencies and anticipates future events through a temporal decoder. While commendable, this approach introduces complexities associated with predicting the future, especially in dynamic traffic scenarios. Speculating on future events may lead to inaccuracies, and the model's effectiveness depends heavily on the accuracy of the future event predictions.

Collectively, the challenges faced by previous researchers underscore the multifaceted nature of video anomaly detection for AVs. The need for models that can discern anomalies in real-time traffic scenarios without the burden of excessive assumptions or reliance on supplementary information remains a central concern. The subsequent section will delve into the proposed model, MOVAD, addressing these challenges and presenting a novel architecture that advances the state of the art in online video anomaly detection for AVs.

Chapter 2

Introduction

2.1 Introduction

In the realm of urban mobility and autonomous transportation, the effective detection of incidents in traffic videos stands as a pivotal challenge with far-reaching implications. As road networks continue to evolve and the deployment of autonomous vehicles becomes more commonplace, ensuring the swift and accurate identification of anomalous events, such as accidents or hazards, becomes paramount for enhancing overall traffic safety. Incident detection serves as the linchpin for proactive responses, enabling timely interventions to prevent collisions, protect pedestrians, and optimize traffic flow. Beyond its fundamental role in safeguarding lives and infrastructure, robust incident detection holds the key to unlocking the full potential of smart transportation systems, paving the way for efficient traffic management and the seamless integration of autonomous vehicles into our urban landscapes.

2.2 Motivation

The motivation behind incident detection in traffic videos is rooted in its profound impact on public safety and the well-being of individuals. Swift and accurate identification of incidents, such as accidents or hazards, plays a pivotal role in saving lives and preventing injuries. When incidents are promptly recognized, emergency

services can be mobilized efficiently, reducing response times and potentially minimizing the severity of outcomes.

Moreover, incident detection is instrumental in limiting property damage. By quickly identifying and responding to accidents, interventions can be initiated to prevent further destruction and protect valuable assets. This not only safeguards individuals but also contributes to the preservation of infrastructure and resources.

The implications extend beyond immediate response to incidents. In the realm of insurance, incident detection in traffic videos becomes a critical tool for evaluating claims. Insurance companies can use this information to determine fault, assess damages, and decide on appropriate compensation. This not only streamlines the claims process but also aids insurance companies in making informed decisions about their customers, helping to tailor coverage plans and premiums based on individual risk profiles.

In legal contexts, incident detection serves as valuable evidence for police investigations. The captured footage provides a factual and unbiased account of events, aiding law enforcement in understanding the sequence of incidents and establishing responsibility. This not only expedites legal proceedings but also ensures a fair and accurate representation of events.

Overall, the motivation for incident detection in traffic videos is deeply embedded in its potential to save lives, reduce injuries, limit property damage, facilitate insurance processes, and contribute to the effective resolution of legal cases. As technology continues to advance, harnessing the power of incident detection becomes increasingly vital for creating safer, more secure, and well-managed transportation ecosystems.

2.3 Objective

The core objective is to train the MOVAD architecture model for accurate accident detection in traffic videos, providing detailed incident descriptions. The study also aims to enhance the model's predictive capabilities, offering relative confidence

levels for each detection. This research contributes to the development of a transparent and effective incident detection system, crucial for advancing traffic safety and autonomous vehicle deployment.

Chapter 3

Related Work

3.1 Overview

The report highlights the contributions:

- **End-to-End Architecture** : MOVAD is an end-to-end architecture designed for Online Video Anomaly Detection, processing ongoing RGB frames with minimal assumptions about ongoing actions and without the need for extra auxiliary information.
- **Transformer-Based Short-Term Memory (STMM)**: The model incorporates a Short-Term Memory Module (STMM) using a Transformer as its backbone. This module processes current and near-past frames in a parallel fashion, leveraging the Transformer's ability to capture long-distance interactions in space and time. The chosen VST (Video Transformer) network serves as the backbone for STMM, outperforming alternatives like ViViT.
- **Long-Term Memory Module (LTMM)**: The paper introduces a Long-Term Memory Module (LTMM) to enrich the output latent space from STMM with contextual information from the distant past. It maintains focus on current events while accumulating a richer understanding of the scene. The design involves updating the representation of the past every time a new frame

is available, employing a stacked three-cell LSTM module to model long-term memory efficiently.

- **Ablation Study** : The paper conducts an exhaustive ablation study over the DoTA dataset, comparing MOVAD against the state-of-the-art models. The results demonstrate MOVAD's superior performance in terms of Area Under the Curve (AUC).
- **Model Training** : The classification head of MOVAD outputs anomaly classification scores for each frame, indicating the likelihood of an anomaly. The model is trained using a weighted cross-entropy loss, with higher weight assigned to the anomaly class. The weight is determined by the distribution of examples in the dataset.

3.2 Memory-Augmented Online VAD

Within our system, we systematically consider both recently observed frames and past frames, leveraging them as distinct sources of information pertaining to the ongoing action and the broader contextual understanding, respectively. To effectively handle the intricacies of Online Video Anomaly Detection (VAD), MOVAD incorporates two pivotal components: a dedicated Short-Term Memory Module (STMM) and a Long-Term Memory Module (LTMM), both seamlessly integrated within the classification head of the architecture. This strategic amalgamation ensures a comprehensive approach to processing both current and historical frames, thereby enhancing the model's ability to discern anomalies in real-time video streams.

3.2.1 Short-Term Memory Module

To preprocess the incoming frame f[t] and concurrently retain proximate spatio-temporal information from f[t − 1] to f[t − (NF − 1)], we opted for a transformer architecture instead of an RNN. This choice facilitates parallel processing, a crucial advantage in handling these data sequences. The preference for a transformer over 3D convolutional models stems from its well-documented efficacy in capturing

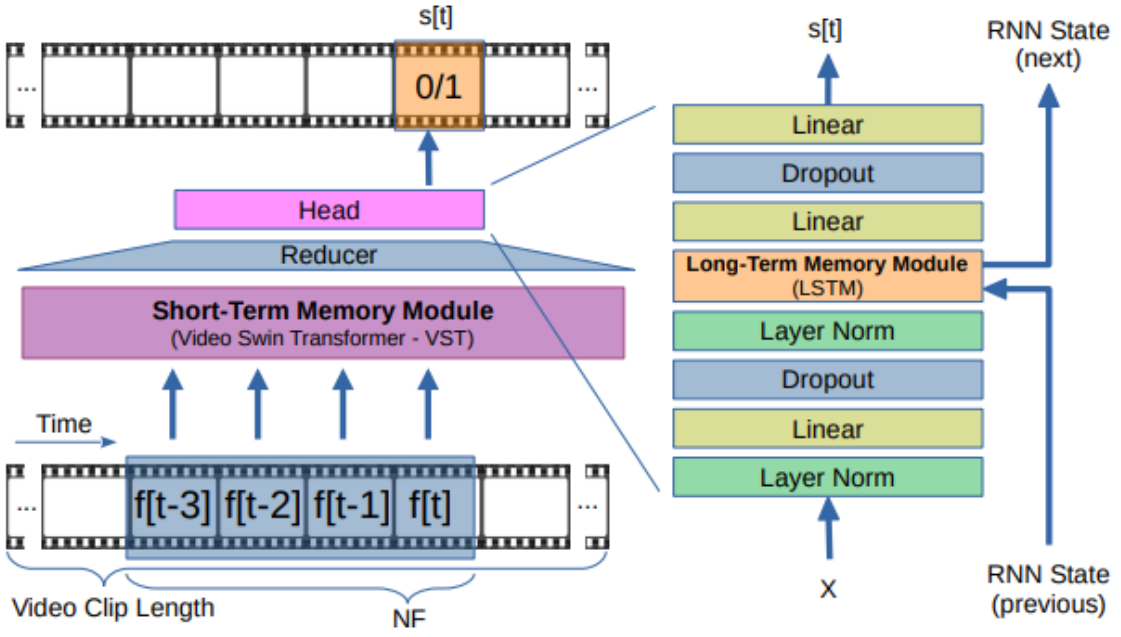

FIGURE 3.1: The online fram-level VAD architecture

long-distance interactions across both space and time . Within this framework, we specifically chose VST as the backbone network for the Short-Term Memory Module (STMM), favoring it over alternatives like ViViT due to its superior performance and more computationally efficient self-attention mechanism.

Despite the advantages of the transformer architecture, the inherent computational intensity of the self-attention mechanism, especially in the context of lengthy videos, prompted us to impose a constraint. The input was restricted to a concise temporal window of NF = 4 frames, spanning from the current frame at time t to the preceding frame at time t−(NF − 1). This deliberate limitation ensures the extraction of a condensed representation of the recent history, subsequently forwarded to the Long-Term Memory Module (LTMM). The VST model operates on video inputs with dimensions NF×H×W ×3, where NF, H, W, and 3 denote the number of frames, height, width, and RGB channels, respectively. Internally, the model segments frames into non-overlapping 3D patches, dividing the video into $\frac{NF}{2} \times \frac{H}{4} \times \frac{W}{4}$ 3D tokens. A 3D shifted window mechanism is employed to establish cross-window connections, effectively harnessing spatio-temporal information for enhanced model comprehension.

3.2.2 Long-Term Memory Module

As emphasized in the ablation analysis in Section 3.1, an excessive amount of historical information (NF $>$ 4) has the potential to mislead the Short-Term Memory Module (STMM), resulting in a performance decline. Our hypothesis attributes this effect to the equal placement of all frames at the same level, lacking a mechanism to weigh them based on their relevance to the past. To address this, we devised an alternative approach to integrate and enhance the output latent space from the STMM with contextual information extracted from the distant past. Upon the availability of a new frame, we transfer the compact representation obtained from the STMM to our Long-Term Memory Module (LTMM), updating its understanding of the past. This dual-focus strategy enables the system to remain attentive to the current situation while accumulating a more nuanced comprehension of the scene, facilitating more precise classifications.

In detail, the output of the Video Spatio-Temporal Transformer (VST) undergoes processing through an Adaptive Average Pool 3D layer (Reducer in Fig. 1) before entering the classification head. The head comprises a series of normalization layers, linear layers, and dropout, arranged in an alternating fashion. Following the last normalization layer, a stacked three-cell Long Short-Term Memory (LSTM) module is introduced to model long-term memory. The LSTM's state, composed of three hidden h[t] and cell c[t] states, undergoes updates whenever a new frame becomes available. The LSTM processes an input features block of [B, 1024], where B represents the batch size, returning a block of the same size along with the state of the cells. Given the relatively small size of the state, this module is highly efficient, incurring a fixed and limited additional computational cost.

For each frame f[t], the classification head produces the anomaly classification score s[t] [0, 1], where 0 signifies no anomaly and 1 indicates an anomalous frame. Training the model involves the use of a weighted cross-entropy loss, assigning higher weight to the anomaly class based on the distribution of data. The weight for each class, denoted as wi, follows the formula wi = e/ei, where e represents the total number of examples in the dataset, and ei is the number of examples for class i.

3.3 Main Function Code

```
import torch
import argparse
import yaml
import numpy as np
import random
import os
import utils

from easydict import EasyDict

from dota import setup_dota, Dota
from metrics import evaluation, print_results
from models import build_cls, build_model_cfg
from optim import build_optimizer
from play import play
from test import test
from train import train


def set_deterministic(seed):
    torch.manual_seed(seed)
    torch.cuda.manual_seed(seed)
    torch.cuda.manual_seed_all(seed)  # if you are using multi-GPU.
    np.random.seed(seed)  # Numpy module.
    random.seed(seed)  # Python random module.
    torch.manual_seed(seed)
    torch.backends.cudnn.benchmark = True
    torch.backends.cudnn.deterministic = True


def parse_configs():
    parser = argparse.ArgumentParser(description='MOVAD implementation')
    # For training and testing
    parser.add_argument('--config',
                        default="cfgs/v1.yml",
                        help='Configuration file.')
    parser.add_argument('--phase',
                        default='train',
                        choices=['train', 'test', 'play'],
                        help='Training or testing or play phase.')
    help_num_workers = 'The number of workers to load dataset. Default: 0'
    parser.add_argument('--num_workers',
                        type=int,
                        default=0,
                        metavar='N',
                        help=help_num_workers)
    parser.add_argument('--seed',
                        type=int,
                        default=123,
```

```
                        metavar='N',
                        help='random seed (default: 123)')
    parser.add_argument('--epochs',
                        type=int,
                        default=200,
                        metavar='N',
                        help='number of epoches (default: 50)')
    help_snapshot = 'The epoch interval of model snapshot (default: 10)'
    parser.add_argument('--snapshot_interval',
                        type=int,
                        default=10,
                        metavar='N',
                        help=help_snapshot)
    help_epoch = 'The epoch to restart from (training) or to eval (testing).'
    parser.add_argument('--epoch',
                        type=int,
                        default=-1,
                        help=help_epoch)
    parser.add_argument('--output',
                        default='./output/v1',
                        help='Directory where save the output.')
    parser.add_argument('--num_videos',
                        type=int,
                        default=20,
                        metavar='N',
                        help='Number of video to play (phase = play)')
    parser.add_argument('--no_make_video',
                        action='store_true',
                        default=False)
    parser.add_argument('--machine_reading',
                        '-mr',
                        action='store_true',
                        default=False)
    args = parser.parse_args()

    with open(args.config, 'r') as f:
        cfg = EasyDict(yaml.safe_load(f))
    cfg.update(vars(args))
    device = torch.device('cuda') if torch.cuda.is_available() \
        else torch.device('cpu')
    cfg.update(device=device)

    return cfg


if __name__ == "__main__":
    # parse input arguments
    cfg = parse_configs()

    # fix random seed
    set_deterministic(cfg.seed)
```

```
traindata_loader, testdata_loader = setup_dota(
        Dota, cfg, num_workers=cfg.num_workers,
        VCL=cfg.get('VCL', None),
        phase=cfg.phase)

checkpoint = None
epoch = 0

if cfg.phase != 'play':
    t_model, mod_kwargs, shape_input = build_model_cfg(cfg)
    model = build_cls(
        cfg, t_model(**mod_kwargs),
        shape_input,
        1 if cfg.phase == 'test' else None
    )

    try:
        checkpoint = utils.load_checkpoint(cfg)
        if cfg.phase != 'play':
            model.load_state_dict(checkpoint['model_state_dict'])

        epoch = checkpoint['epoch'] + 1
    except FileNotFoundError:
        print('no checkpoint found')
        # save info about no checkpoint has been loaded
        cfg._no_checkpoint = True
        if cfg.epoch != -1:
            epoch = cfg.epoch
        # load pretrained if available
        utils.load_pretrained(model, cfg)

if cfg.phase == 'train':
    optimizer, lr_scheduler = build_optimizer(cfg, model, checkpoint)
    index_loss = 0
    index_guess = 0
    if checkpoint is not None:
        index_guess = checkpoint.get('index_guess', 0)
        index_loss = checkpoint.get('index_loss', 0)
    train(cfg, model, traindata_loader,
          optimizer, lr_scheduler, epoch, index_guess, index_loss)

elif cfg.phase == 'test':
    filename = utils.get_result_filename(cfg, epoch)
    if not os.path.exists(filename):
        if cfg.get('_no_checkpoint', False):
            # in case you don't have a checkpoint to test
            raise Exception('no checkpoint to test')
        with torch.no_grad():
            test(cfg, model, testdata_loader,
                 epoch, filename)

    content = utils.load_results(filename)
```

```
        outputs = content['outputs']
        targets = content['targets']
        toas = content['toas']
        teas = content['teas']

        print_results(cfg, *evaluation(FPS=cfg.FPS, **content))

    elif cfg.phase == 'play':
        play(cfg, testdata_loader)
```

Listing 3.1: Main function code

Chapter 4

Results

4.1 Dataset

We performed our training and test on Task 1, the frame-level VAD, of DoTA dataset [9], using only the anomaly class and its temporal boundaries, strictly in the online scenario

4.2 Evaluation Metrics

We use the well-known Area Under the Curve (AUC) metric at frame-level, to evaluate how well the model is able to temporally locate the anomaly in the videos

4.3 Training Details

The training process is conducted on a single machine equipped with an A100 GPU, utilizing the Stochastic Gradient Descent (SGD) optimization algorithm with a learning rate set to 0.0001 and a momentum of 0.9. We opt for SGD over Adam due to its superior stability in our experiments, as the latter tended to lead to more erratic training behavior, resulting in model divergence after a few epochs. Unless explicitly stated otherwise, the batch size is set to 8, and the input video size is

320×240. The Video Clip Length (VCL), representing the number of frames within each video batch, is configured as 8. Additionally, we set the LSTM cell number to 2, NF (number of frames) to 3, and initialize linear weights using a uniform distribution. The LSTM cells are initialized with a (semi-)orthogonal matrix, while bias parameters are initialized to zero. The Visual spatio-temporal Transformer (VST) is initialized with a model pre-trained on Something-Something v2 [3].

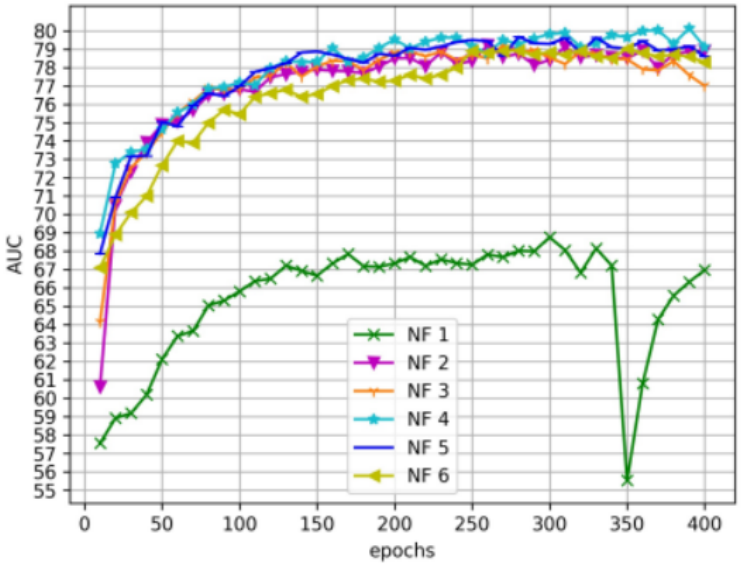

FIGURE 4.1: Performance Comparison, changing the number of frames(NF) in input to STMM (from 1 to 6)

During training, we employ a weighted Cross-Entropy loss to address the challenge of imbalanced data within the Detection of Traffic Anomaly (DoTA) dataset. Specifically, we assign weights $w_{\mathrm{n}} = 0.3$ and $w_{\mathrm{a}} = 0.7$ to the normal and anomaly classes, respectively, as per the equation provided at the end of Section 2.

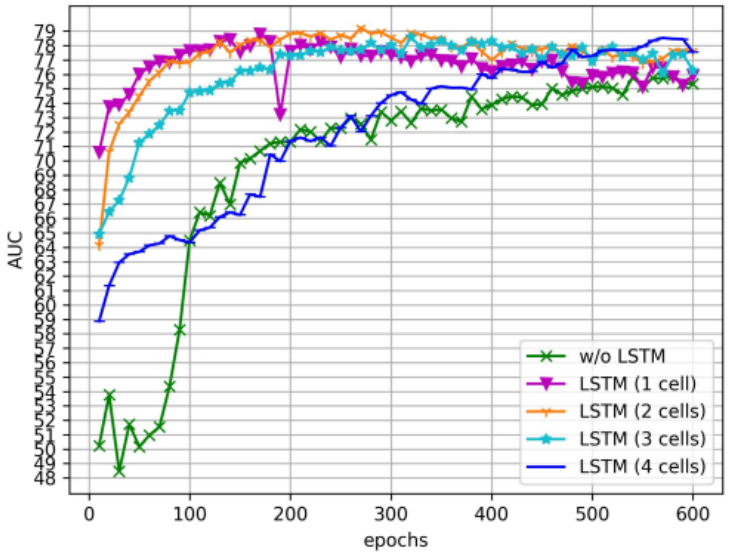

FIGURE 4.2: Performance Comparison, changing the LSTM cells (from 0 to 4)

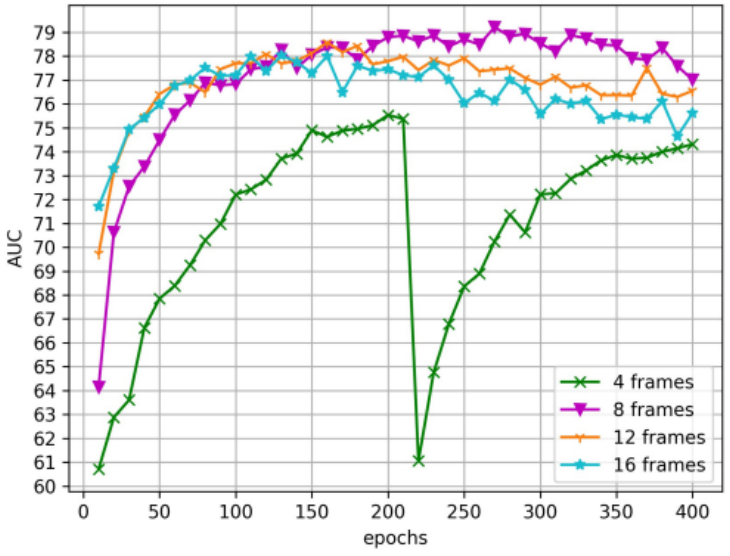

FIGURE 4.3: Performance Comparison, changing the VCL

#	Short-term	Long-term	*AUC*
1			66.53
2	✓		74.46
3		✓	68.76
4	✓	✓	**79.21**

FIGURE 4.4: Performance comparison with and w/out short and long-term memory. Short-term: with (NF = 3) and w/out (NF = 1). Long-term: with (2 cells) and w/out (0 cells).

Chapter 5

Ablation Study

5.1 Memory module effectiveness

We first tested the effect of memories. STMM and LTMM both contribute to enhance the general performance, obtaining the best AUC when both are active, highlighting their importance.

5.2 Short-Term Memory Module

We opted to structure our Short-Term Memory Module (STMM) based on a Video Swin-B architecture, characterized by an embedding dimension of C=128 after the linear projection of the patches. In Fig. 4.1, the results are depicted when varying the number of frames (NF) processed by the STMM at each step. As anticipated, considering only the current frame is deemed the least favorable scenario, resulting in the loss of temporal information and rendering the training process unstable. This is understandable: lacking any awareness of the near past, the STMM predisposes the Long-Term Memory Module (LTMM) to overfitting, given the high similarity between consecutive frames. Conversely, an increase in the number of frames generally enhances performance, up to the point of processing 5 or more frames, where the effect becomes counterproductive. As outlined in Section 2, in accordance with our

hypotheses, overloading the transformer becomes detrimental, as it lacks a mechanism to differentiate the weighting of the remote and recent past. Overall, the highest Area Under the Curve (AUC) is achieved with 4 frames."

#	Method	Input	*AUC*
1	ConvAE [6] (*)	Gray	64.3
2	ConvAE [6] (*)	Flow	66.3
3	ConvLSTMAE [2] (*)	Gray	53.8
4	ConvLSTMAE [2] (*)	Flow	62.5
5	AnoPred [7] (*)	RGB	67.5
6	AnoPred [7] (*)	Masked RGB	64.8
7	FOL-Ensemble [15] (*)	RGB + Box + Flow + Ego	73.0
8	STFE [16]	RGB + Flow	79.3
9	**Our (MOVAD)**	**RGB** (320×240)	**80.09**
10	**Our (MOVAD)**	**RGB** (640×480)	**82.17**

FIGURE 5.1: Benchmarks of VAD methods on the DoTA dataset. Both MOVAD models are trained with the best configuration of: VCL of 8, 3 LSTM cells and NF = 4.

5.3 Long-Term Memory Module

In Fig. 4.2, we assess the capabilities of the Long-Term Memory Module (LTMM) by varying the number of cells from zero (absence of LSTM) to four. Interestingly, having no cells results in slower training, saturating performance with the lowest achieved Area Under the Curve (AUC). Generally, an increase in the number of cells leads to a slower attainment of the maximum AUC, but the performance surpasses that without LSTM. With 1 cell, the performance saturates rapidly, with a gradual decline during the subsequent epochs. On the contrary, 4 cells lead to a very slow saturation without reaching the optimal performance. The global maximum AUC is achieved with 2 cells, albeit only slightly higher (+0.02) than with 3 cells. Despite this marginal difference, we favor the latter configuration due to its ability to enhance training quality in a more gradual and continuous manner, which we believe has broader benefits. Our hypothesis is substantiated, and in conjunction with NF =

4 (the optimal configuration from the previous experiment), it results in a higher AUC compared to the 2-cell scenario. We speculate that this occurs because both configurations (3 cells in Fig. 4.2 and NF = 4 in Fig. 4.1) achieve the best AUC around the same time, approximately 400 epochs.

Model	*ST*	*AH*	*LA*	*OC*	*TC*	*VP*	*VO*	*OO*	*UK*
AnoPred [7]	69.9	73.6	75.2	69.7	73.5	66.3	N/A	N/A	N/A
AnoPred [7] + Mask	66.3	72.2	64.2	65.4	65.6	66.6	N/A	N/A	N/A
FOL-STD [15]	67.3	77.4	71.1	68.6	69.2	65.1	N/A	N/A	N/A
FOL-Ensemble [15]	73.3	81.2	74.0	73.4	75.1	70.1	N/A	N/A	N/A
STFE [16]	75.2	84.5	72.1	77.3	72.8	71.9	N/A	N/A	N/A
Our (MOVAD)	85.6	85.1	83.9	82.2	85.3	**86.2**	**79.3**	86.7	**77.1**
Our (MOVAD) †	**86.6**	**86.3**	**84.9**	**83.7**	**85.5**	81.6	77.4	**87.9**	73.8
Model	*ST∗*	*AH∗*	*LA∗*	*OC∗*	*TC∗*	*VP∗*	*VO∗*	*OO∗*	*UK∗*
AnoPred [7]	70.9	62.6	60.1	65.6	65.4	64.9	64.2	57.8	N/A
AnoPred [7] + Mask	72.9	63.7	60.6	66.9	65.7	64.0	58.8	59.9	N/A
FOL-STD [15]	75.1	66.2	66.8	74.1	72.0	69.7	63.8	69.2	N/A
FOL-Ensemble [15]	77.5	69.8	68.1	76.7	73.9	71.2	65.2	69.6	N/A
STFE [16]	**80.6**	65.6	69.9	76.5	74.2	N/A	75.6	70.5	N/A
Our (MOVAD)	72.1	71.6	72.3	76.5	75.7	74.1	77.9	71.7	69.1
Our (MOVAD) †	72.2	**74.0**	**74.8**	**80.2**	**79.6**	**76.8**	**82.2**	**78.3**	**72.9**

FIGURE 5.2: Detection accuracy (AUC) for individual accident categories. "*" non-ego anomaly categories. "†" if input resolution is 640×480 instead of 320×240.

5.4 Video Clip Length

In Fig. 4.3, we investigate the impact of different values of Video Clip Length (VCL). The least favorable and most unstable training occurs with 4 frames, potentially because this quantity is insufficient to fully leverage the long-term memory effect of LSTM cells. Increasing VCL facilitates faster performance saturation, but excessively high values (such as 12 or 16) tend to result in a lower overall Area Under the Curve (AUC). The optimal AUC is achieved with 8 frames as VCL, striking a balance between enlarging clip size to exploit LSTM cells and reducing it to prevent overfitting, considering that consecutive frames may be overly similar.

Chapter 6

Conclusion

This paper introduces MOVAD, a novel architecture designed for the frame-level Video Anomaly Detection (VAD) task. MOVAD operates seamlessly in an online fashion, addressing the most challenging VAD scenarios through end-to-end training and relying solely on RGB frames. The architecture consists of the Short-Term Memory Module (STMM), which captures information pertinent to ongoing actions using a Visual Spatiotemporal Transformer (VST), and the Long-Term Memory Module (LTMM), incorporating considerations of the remote past through the integration of LSTM within the classifier. The performance evaluation conducted on the DoTA dataset, comprising dash-mounted camera videos of accidents, demonstrates an impressive 82.17% Area Under the Curve (AUC), surpassing the State-of-the-Art (SOTA) by +2.87 AUC.

6.1 Future Work

A lot of work is to be done in the incident detection problem in traffic video:

- Description of the video : We need to identify what is happening in the video so that we can take the right course of action to prevent such incidents.
- Uncertainty involved : We aim to provide not only accurate predictions but also a measure of our confidence in those predictions

Bibliography

[1] Hasan, M., Choi, J., Neumann, J., Roy-Chowdhury, A.K., Davis, L.S.: Learning temporal regularity in video sequences. In: Procs. of the IEEE Conf. on computer vision and pattern recognition. available: https://openaccess.thecvf.com/content_cvpr_2016/papers/Hasan_Learning_Temporal_Regularity_CVPR_2016_paper.pdf.

[2] Luo, W., Liu, W., Gao, S.: Remembering history with convolutional lstm for anomaly detection. In: 2017 IEEE Intl. Conf. on Multimedia and Expo. available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8019325.

[3] Wang, L., Zhou, F., Li, Z., Zuo, W., Tan, H.: Abnormal event detection in videos using hybrid spatio-temporal autoencoder. available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8451070.

[4] Liu, W., Luo, W., Lian, D., Gao, S.: Future frame prediction for anomaly detection–a new baseline. In: Proc. of Conf. on Computer Vision and Pattern Recognition available: https://arxiv.org/pdf/1712.09867.pdf.

[5] Zhou, Z., Dong, X., Li, Z., Yu, K., Ding, C., Yang, Y.: Spatio-Temporal Feature Encoding for Traffic Accident Detection in VANET Environment. IEEE Tran. on Intelligent Transportation Systems. available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9714213.

[6] Yao, Y., Wang, X., Xu, M., Pu, Z., Wang, Y., Atkins, E., Crandall, D.: Dota: Unsupervised detection of traffic anomaly in driving videos. IEEE Tran. on Pattern Analysis and Machine Intelligence. available: http://vision.soic.indiana.edu/papers/dota2022pami.pdf.

[7] Xu, M., Gao, M., Chen, Y.T., Davis, L.S., Crandall, D.J.: Temporal recurrent networks for online action detection. In: Procs. of the IEEE/CVF Intl. Conf. on Computer Vision. available: https://arxiv.org/pdf/1811.07391.pdf.

[8] Leonardo R., Vittorio B., Tomasao F., Massimo B., Andrea P.: Memory-Augmented online video anomaly detection. available: https://arxiv.org/pdf/2302.10719v2.pdf.

[9] Dataset. available : https://drive.google.com/drive/folders/1IVCedrlPg03Fsg4tqDA2cWYlcdrsKUsp?usp=sharing