# Hawk Eye: A Real-Time Violence Detection Using 3D-CNN and Jetson Nano for Surveillance

Fatin Ishrak*, Jannatun Nahar†, Zarin Tasnim Promi‡, Jannatul Ferdous§
Department of Computer Science and Engineering
Brac University, Dhaka,Bangladesh
Email: *fatin.ishrak@g.bracu.ac.bd, †jannatun.nahar@g.bracu.ac.bd, ‡zarin.tasnim.promi@g.bracu.ac.bd, §jannatul.ferdous1@g.bracu.ac.bd

***Abstract*—Violent action detection has become a trending topic and active research domain of computer vision and video processing within the past few years. In this paper, we detect violence in real-time using the Spatio-temporal feature with 3D CNN, which is a DL violence detection framework, specifically suited for crowded places. Here, we have used embedded devices in the form of Jetson Nano to feed it with the dataset and test our model to evaluate and eradicate the issues present in cloud-based systems. An alert is sent immediately to the local police station or security agency as soon as a violent activity is detected so that urgent preventive measures can be taken. We have used four different datasets in which we got an overall success rate of 96%-99%.**

***Index Terms*—Violence; DenseNet; 3D Bi-LSTM; Embedded Device; 3D CNN; Jetson Nano; Cloud Network**

## I. Introduction

Human activity detection for video surveillance systems is an automated method of analyzing video sequences and making intelligent decisions about the activities represented in the footage using an embedded device [1]. One of the many difficulties with human action recognition (HAR) is the classification of human action in real-time, almost instantaneously after the action has taken place. In South Korea, for example, about $954,261$ CCTVs were installed in public spaces in 2017, up 12.9% over the previous year [2]. With the expansion of technology and innovation in the field of computer vision in recent years, a large number of current ways for detecting 3D objects and recognizing violence have arisen, one of which is an edge device in the form of Jetson Nano [13]. Furthermore, for violent scene recognition to be helpful in the real-world the identification of violence must be swift in order to allow for prompt intervention and backbone.

Object detection in video surveillance systems is complicated by challenges such as image classification, frame segmentation, and object classification. High-performance embedded systems can help to solve these issues. When comparing two popular embedded devices, the Raspberry Pi 4 and the Jetson Nano, it became evident that the Jetson Nano had a far more competent AI performance [11]. Jetson Nano is a compact, customized AI machine that offers great performance for new AI applications, with 472 GFLOPs [12] for fast and parallel AI algorithms. It is an embedded device that supports various neural network algorithms and provides an efficient security system and low-cost real-time implementation.

Many enhanced cloud computing approaches have been presented in recent years to meet the current need for high-resolution video computing with deep learning techniques for surveillance systems. The cloud infrastructure is concerned with data loss in public clouds, costing issues in private clouds, and maintenance and distribution issues in hybrid clouds [4], which include, but are not limited to, relaying media streams at greater frame rates without skipping frames, cost-effective storage for extended media data retention, and storage elasticity. Surveillance systems over the cloud have several challenges, including disaster response [5], billing and cost structures [5], appropriate data protection policies, and vendor lock-in [6]. In a cloud system, media processing creates many issues [7], such as in the cloud choosing various sensory media streams, performing these activities to detect harmful security breaches and alerts when needed along with uncertainty in creating a real-time surveillance system. Furthermore, the fast reporting of incidents by surveillance clients and instantly driving multimedia surveillance systems are the prime issues whereas it is critical to determine the quality metrics such as event recognition rate, reaction time, CPU and memory usage, volume of work, cost-benefit analysis, typical operation wait period, and whether or not standard of service is assured.

Given the difficulty in recognizing violence, we decided to use deep-learning (DL)-based 3D CNN models to learn complicated sequential patterns and reliably predict violence using Jetson Nano. Most violence detection algorithms are plagued by the difficulties of digesting huge volumes of meaningless frames. As a result, they consume a large amount of memory and take a long time to process. To address this limitation, we have used a pre-trained MobileNet CNN model to detect people in videos with the embedded device. Only critical feeds related to the event will be forwarded to the 3D CNN model for final prediction once the frames have been cleaned, in order to facilitate efficient processing. Various common algorithms for detecting violence are unable to establish usage patterns and offer low accuracy results due to a lack of data from bench-marked datasets but hockey fights (HFs), movie fights (MFs), violent flows (VFs), and real-life violence situations (RWF) are all publicly available benchmark datasets upon which 3D CNN has been fine-tuned. We applied the notion of transfer learning in this research to detect violence in both indoor and outdoor contexts, and we achieved a remarkable accuracy rate. Finally, we used Nvidia's Deepstream SDK to optimize the trained deep learning model. We are leveraging development boards like the Jetson Nano to deploy these types of devices using the HAR approach. The implementation on embedded devices will solve the problems of deployment architecture over clouds by ensuring that all processes within that architecture are being processed onboard and in real-time. Our methodology resolves the issues of stated frame rate, video pixel, and dynamic resource supply. While processing videos in real-time, it also provides a speedier response than offline video processing. The trained model translates into an alternative manner based on trained parameters and topologies using onboard processing techniques. We solved the resource allocation problems with the help of Jetson Nano which is briefly discussed in the Model on Jetson part (III-9). Furthermore, for cost-effective monitoring, we may adopt developing approaches that are lightweight in identifying activity and can be readily integrated into image sensors and IoT systems.

This report aims to construct a surveillance system that will detect anomalous behavior using a 3D CNN model and a real-time solution on Jetson Nano. In section II we focused on the background work of violent action detection. Section III unfolds the methodology, where the models, datasets, and architecture of the model are described that we have used to get our desired result. Section IV describes the result of our research, which shows improvement in accuracy and shorter inference time after measuring optical flow, and lastly, section V contains a conclusion and future work, that states in the future it is vital to integrate posture estimation or face identification in a generalization of violence affecting humans.

## II. Literature Review

The authors of the research [8] proposed a Mixed Convolutional Tube (MiCT), merging 2D CNNs with the 3D convolution module (C3D) and achieving less training complexity for Spatio-temporal fusion in each frame. Their model outperformed the C3D and C3D BN for the UCF101 dataset by increasing the accuracy by $6\%$ and $4.4\%$ and overall $89.1\%$. Again for the HMDB51 dataset, the model shows an improvement of $4.2\%$ to $5.4\%$ for C3D and $2.8\%$ to $3.9\%$ for C3D BN and overall $58\%$ accuracy, for a larger dataset Sport1M the model outperformed C3D with increased accuracy of $6.3\%$.

The spatio-temporal Texture (STT) Model is a high performing method and a sensitive model for crowd violence detection. In [9] the authors developed a decision making algorithm based on extracted spatio-temporal textures and compared the time consumption in real-time for a buffered video with other benchmark approaches, their algorithm worked faster by taking $1658ms$ ($18.4ms/frame$ based on 30fps video clip) for human activity detection. On the other hand, the authors of the research [10] presents a human action recognition system in sequential images. They extracted robust spatio-temporal features by silhouetting the whole human body on the image and focused on the essential characters. They used an Artificial Neural Network (ANN) to train their system and detected six key interactive characters from UT-Interaction dataset. They separated their data into two sets (Parking background & Lawn background) and obtained recognition accuracy of $83.5\%$ and $72.5\%$ respectively, outperforming other benchmark models such as BoW, HIR based on co-occurrence of visual words, STR Match kernel.

Jetson boards are extremely efficient embedded computers for detecting violent acts in the real-time and fast system processing. The authors of [13] conducted a thorough analysis of 3D object identification frameworks and their performance when used with NVIDIA Jetson boards. They utilized YOLOv3, YOLOv4, and lesser variants of these frameworks, and ran all four on two Jetson boards with KITTI benchmark dataset. The four algorithms all achieve over $0.5mAP$ and up to $0.7IoU$ and Complex-YOLOv4 achieved the highest mAP in the experiment. They achieved considerably poorer performance on the Jetson boards in their study, which they could alleviate by utilizing Nvidia's TensorRT library. Another study [14] suggested a lightweight CNN-based single stream HAR model that can run in real time with the help of a Jetson Nano linked to a CCTV camera, demonstrating the model's potential in low-cost GPU-based embedded systems. They have achieved a higher action recognition accuracy of $93.69\%$ on the benchmark dataset UCF-101. The Jetson Nano's system environment includes a quad-core ARM A57 CPU, 128-core Maxwell GPU, and $4$ GB of main memory. Finally, the research found that data is processed $30\%$ quicker in Jetson Nano using a weighted mean-based model, which is superior to LSTM.

Cloud-based multimedia surveillance systems can readily solve surveillance situations on a large scale, in this regard, the research [3] proposed a framework highlighting several types of research, technical issues, and a prototype surveillance system. For the formation of their prototype, they used a public cloud platform Amazon EC2, OpenCV library for face detection, and an efficient task scheduling algorithm Min-Min. They accomplished 5 hours of simulation for $40\%$ optimization cost and $5\% - 7\%$ frame loss under 30 fps. On the other hand, [7] proposed another framework of media-edge cloud architecture to achieve high quality of service (QoS) for multimedia services. They used algorithms like PhotoSynth, dynamic programming model, genetic algorithm, and other fantastic techniques such as leveraging high QoS for various devices to optimize the proposed framework. They performed photosynthing code in an HPC cluster node with nine servers and achieved $1.65$ times and $4.25$ times computational gain over the traditional approach for the two-server case and nine-server case respectively.

## III. Dataset and Model specifications

*1) Dataset:* In our research, we have worked on four datasets that are widely used and accepted in violence detection. These datasets are Hockey Fights (HFs), Movies Fights (MFs), Violent Flows (VFs), and Real Life Violence Situations (RWF). These datasets cover indoor and outdoor scenarios as well as weather conditions. RWF-2000 is a Large Scale Video Database for Violence Detection that has a data preprocessor

TABLE I
SUMMARY OF REVIEWED PAPERS

| Ref. | Task | Device | classifier | Dataset | Accuracy |
|---|---|---|---|---|---|
| [8] | Mict for HAR | – | 2D CNN, C3D, C3D BN | UCF101, HMDB51, Sport1M | 89.1%, 58%, 6.3% increased |
| [9] | Real-time crowd anomaly detection | Host PC-64bit Core i7 CPU (4GB RAM) | Weighted multi-binary evaluation algorithm | UMN, UCSD | 18.4ms/frame |
| [10] | Human interaction recognition | – | ANN | UT-Interaction dataset | 83.5%, 72.5% |
| [13] | Benchmark Analysis of 3D Object Detectors | AGX Xavier, Jetson Nano | complex and tiny YOLO (v3,v4) | KITTI | 0.5 mAP up to 0.7 IoU (all four) |
| [14] | Real-time HAR on Low-cost embedded device | Jetson Nano | Single-stream CNN | UCF-101 | 93.69% |
| [3] | A cloud-based multimedia surveil-lance system | Intel Xeon Processor E5430, VIVOTEK camera | OpenCV, Min-Min | Amazon EC2 | Frame loss: 5% – 7% (30 fps) |
| [7] | Multimedia cloud computing | HPC cluster | PhotoSynth, dynamic programming model, genetic algorithm | Group of 9 server | computational gain: 1.65 times, 4.25 times |

that is done with python scripts. To extract the features, in images the color changed from RGB to Gray through optical flow. The same dimension is also used when we convert video to a python file which is of height, width $224 \times 224$, and only 1 channel is used as it is grey. After padding two channels are created: one is a normal channel and another is an empty channel.

*2) Distributed network and Dense optical flow:* As the software for the alert system and the NumPy array where the data segmentations are both stored on the Jetson Nano, the computing data stream process of more than one camera makes the system challenging. We tried to attempt to create a framework in which the CPU cores are uniformly distributed over the segments, however as we add more than one camera, various segments overlap and the computation speed decreases significantly, posing a serious threat to the framework. Finally, stacking the Jetson Nano devices raises the maintenance cost significantly because the upkeep of each device increases, as does the streaming procedure for each frame. Using a powerful development board such as AGX Xavier, TX2 may give a positive outcome as it has more CPU cores and computational power.

Dense optical flow is one of the inputs of our network. The generation of frame sequence is done in this algorithm where the most moved pixels between the consecutive frames are represented with greater intensity. Besides that, the main components are contact and speed. We chose dense OF over discrete OF because dense OF generates flow vectors for the entire frame, up to one flow vector per screen size, whereas sparse OF only generates flow vectors for certain features, such as some pixels that portray the edges/seams of an object within the frame.

*3) Model architecture & Model Justification:* Optical flow is encoded by a dense network as a segment of featuring maps. At first, the featured maps go through a multi- head self-attention layer. After that, it goes through a bidirectional ConvLSTM layer. After all these, attention mechanisms are applied in the forward and backward temporal direction respectively. This is basically an ST encoder that actually extracts the necessary spatial and temporal features from each and every video. Finally, the features which are encoded are inserted into a four-layer classifier which mainly signifies whether it is a violent video or not.

For video classification, the 3D DenseNet variant has been used. The Bi- RCNN block allows the feature analysis in the forward and backward temporal direction and thus the block improves the efficiency while recognizing the violent actions. The attention mechanism mainly recognizes three things, these are human action, Conv network combination and, Bi-Conv recurrent blocks. This model really helped us in developing our proposal as it is based on blocks that recognize human actions.

*4) Media Storage:* As the Jetson Nano is a resource-constrained device, after feeding the computing data and the software for the alert system, there is not much room left, therefore an event-driven mechanism to store data in the cloud is adopted. Only significant frames occurring 5 minutes before and after the occurrence are detected in an event-driven process in order to overcome the storage issue. Since data transfer to the cloud environment is limited in this approach, energy usage and bandwidth utilization are decreased. This mechanism will let the important data stored for further revision and investigation of the crime scene.

*5) DenseNet Convolutional 3D:* The DenseNet has a system of working in layer by layer and the layers are connected in feed-forward fashion. Instead of DenseNet the MaxPool3D and AveragePool3D were used which have the reduction layer of the size of $(2, 2, 2)$ and $(7, 7, 7)$. The basis of DenseNet structure is the dense blocks. The blocks are made of the feature maps of a layer with all of its product. The DenseNet structure can work in such a way that is more prosperous and it generates a lower number of screens and specifications in order to achieve high performance.

*6) Multi-Head Self-Attention:* The architecture of this procedure is showing the input data by applying different linear projections learned from the same data and finally executing the self-attention mechanism in every output. We selected the multi-head self-attention mechanism to determine which elements are common in both temporal directions by developing a weighted matrix that consists of more relevant past and future information.

*7) Bidirectional Convolutional LSTM 3D:* This system has two states: forward-state and future-state. The generated output can get data from both states. This module is well-known for its ability to look back in video. In order to avoid such a situation we proposed a model where we generated Conv layers instead of entirely connected layers, here the convLSTM is able to observe both spatial and temporal features and let us get data from both features. Bi-conv system is an advanced convLSTM which has the access to look backward and forward in a video,which gives the system an overall better outcome.

*8) Classifier:* Classifier is made up of connected layers. Each layer has nodes that are ordered in a definite manner. The ReLu function is used by the hidden layer, the Sigmoid function is engaged to verify whether an action category is violent or not and the output is a binary predictor which is of the last layer.

*9) Model on Jetson:* Our proposed model is based on Jetson Nano. For high performance with modern AI workload Jetson Nano has been used for the implementation. For our model power consumption is 10 watt. The $80mm$ x $100mm$ dimension consisting 128 core Maxwell GPU, quad core Arm Cortex-A57 CPU and primary memory of 4GB made us to use it with CCTVs. For better performance, we used 4GB of SWAP memory. Our model is finely optimized and Jetson worked much faster due to its dedicated GPU that is backed up with Nvidia's TensorRT. We used TensorRT's accelerator library JetPack 4.6.1 for the DL platform as it has DL optimizer and it achieved higher throughput and lower latency at run time for the DL applications.

Jetson has Linux based operating system called Linux 4 Tegra OS. For real time video capturing, we used CHCPE07 USB camera with recording measure $1080p$ $30fps$. With few additional libraries been installed, the entire pre-trained model has been implemented with the transfer learning methods to detect the real time violence. The main challenge was the device has low resource and many computations needs to be processed simultaneously for IoT connectivity, alert process and feed transfer and receiving. To overcome the challenge our model has optimized minimizing unnecessary computation, only pre-processed data are being sent to detect the violence in order to operate the system perfectly.

## IV. Implementation and Results Analysis

In this section, the implementation of our proposed model has been described for detecting violence for the surveillance system. This model used Python, Tensorflow, Keras and OpenCV for implementation and used to run the test on unclassified input data and to generate the result.

*1) Training Methodology:* In this stage, the weights of all neurons in the model were randomly initialized. Here, Each frame's pixel values were normalized to be in the range of 0 to 1. To calculate the input video sequence from all of the videos from every dataset the average of all sequence was calculated. Moreover, the frames were enlarged to the standard size for Keras pre-trained models, which is $224 \times 224 \times 3$. The chosen parameters were: a base learning rate of 104, a batch size of 12 films, and 100 epochs where the weight decay was set to 0.1. Furthermore, the Adam optimizer's default setup was employed. For the last layer of the classifier, loss function determiner the Binary Cross Entropy has been taken and for activation function the Sigmoid function has been chosen. A random permutation cross-validator method was used to test the datasets' performance and decided to use a ten-fold cross validation method. The studies were conducted with optical flow. On each pair of the adjacent frames, a matrix subtraction has been performed in a sequence of frames

$$\forall n < k \in [0, k] : sn = f(n) - f(n) + 1 \qquad (1)$$

Any variation between the pixels of two successive frames was represented using this way. If the same pixel did not change in value in both frames, removing both frames turned that pixel black.

*2) Ablation Study & results:* A ten-fold experiment was developed for the ablation investigation. The direct connection between the Bi-ConvLSTM and the DenseNet avoided the self-attention process.

Despite the fact that a very powerful core network has been used here than in any prior studies. We thought it would be really amazing to watch the improvement of performance through the modification of the input of the network and also engaging the attention mechanism. When other were compared to our model two primary benefits were found: improved accuracy and lower inference time.

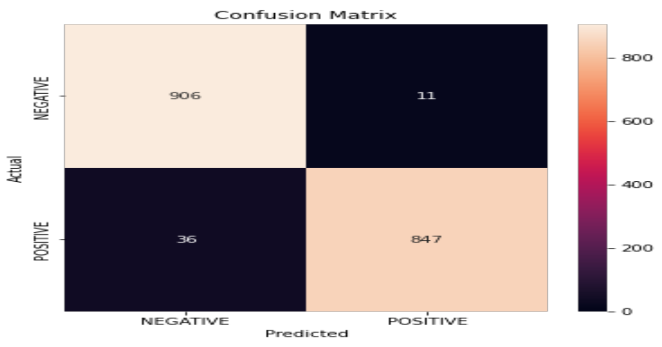

Fig. 1. Confusion Matrix

The variation accuracy was least when the dataset consisted of fifty frames on the average (HF and MF), but when it reached a hundred frames on the average (VF and RWF-2000), the model with self-attention outperformed the others

TABLE II
CLASSIFICATION REPORT

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| NEGATIVE | 0.96 | 0.99 | 0.97 | 917 |
| POSITIVE | 0.99 | 0.96 | 0.97 | 883 |
| accuracy | _ | _ | 0.97 | 1800 |
| macro avg | 0.97 | 0.97 | 0.97 | 1800 |
| weighted avg | 0.97 | 0.97 | 0.97 | 1800 |

by two points. It lessened inference from 4% (VF) to 16%(HF) without attention. The following table shows the accuracies achieved with our proposed model for each of the datasets.

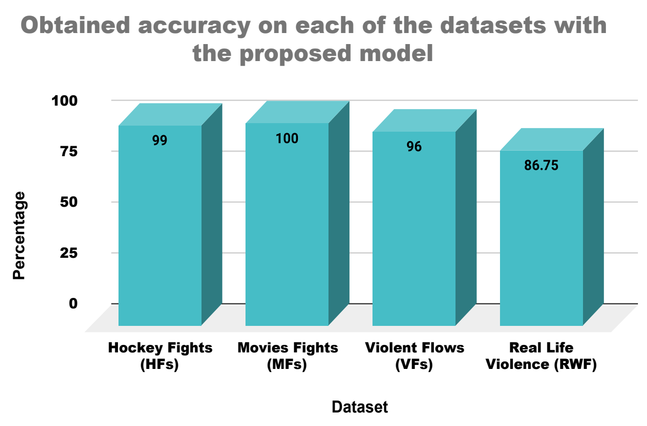

Fig. 2. Accuracy for each of the datasets with the proposed model

*3) Implementation on Jetson Nano:* For implementing on Jetson Nano we used 64GB microSD card containing minimum of 120MB/s bus interface speed and flashed the Jetpack 4.6.1 on the memory. Jetson inference has set up as a base container. As it is a small sized embedded device 4GB of the swap memory has been initialized to support RAM. Next we ran the Jetson-inference docker container with shell scripts. Then we used transfer learning approach to retrain the model of Jetson Nano with our previously trained model.

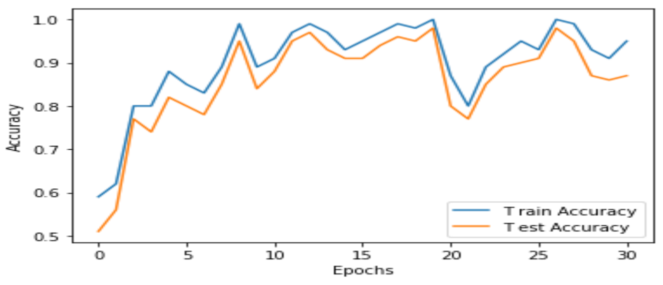

Fig. 3. Accuracy curve

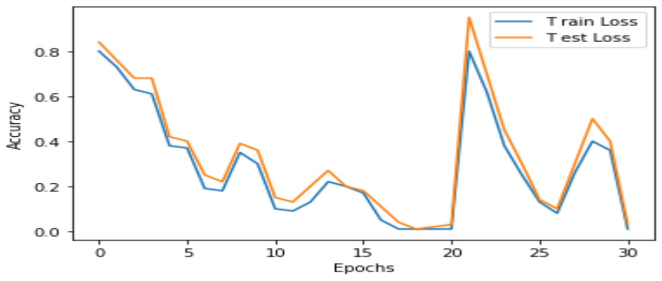

Fig. 4. Loss curve

For better performance, we resumed the training until the loss and accuracy meets the requirement. After the learning process finished the model is exported as $onnx$ format that can be run on many major ML framework.

Next, the python program has structured to use the model for real-time violence detection. The program contains jetson.inference, jetson.utils, time and datetime modules. Then input and output source and network has introduced. Then required variables has been declared and model started running under main function with Capture, Render and SetStatus. To store the required data, we used two global variable for time control such that when the violence detected it will fetch the start violence time and the previous 5 minutes and after entering the violence block it includes another timer to get the next 5 minutes after the violence ends. Finally, it saves the status of the program in a $.txt$ file for further data analysis.

*4) Outcome:* In the figure, we can see the first violent act video feed captured in the camera however not been detected yet.

Our approach was not limited to detecting violence but also we wanted to implement it in smart cities. Therefore, we created feeds for CCTV so that later we can identify the culprit. The real-time CCTV feed version of our model with detection of violence is shown in the figures.

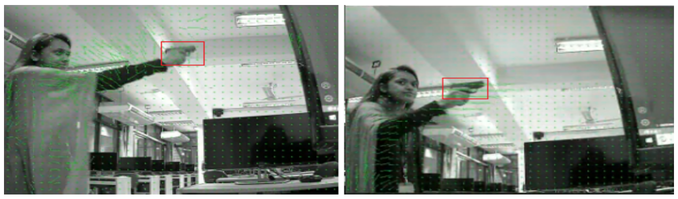

Fig. 5. Violent Action Detected

For a better understanding of our approach, the figure shown below is the side-by-side comparison of the original CCTV footage which will be monitored by the authorities, and the underlying process of the Jetson Nano for detecting the violence. Here, we can see that Jetson Nano has kept only guns in red and removed all other features of the picture as per our model which increases the violence detection rate and also increases the fps.

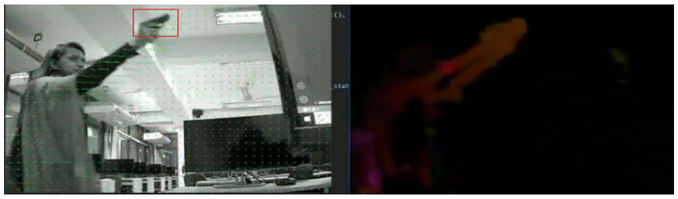

Fig. 6. side by side comparison

Here, the comparison of our model with other models after implementing on Jetson Nano is shown in terms of performance. To get a clear idea, the comparison is shown based on the FPS of the video. In the figure, we can see that our model outperformed the rest of the model in terms of the fps.

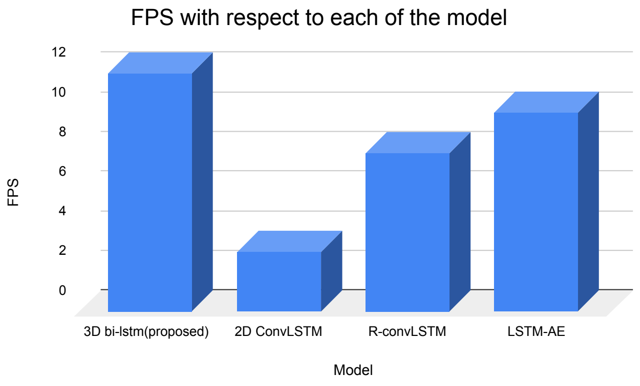

Fig. 7. Comparison based on FPS

The figure shows our model has achieved 12 fps while detecting the violence which is the highest among others. The second-best model with 10 fps is the LSTM AutoEncoder which has a lower accuracy rate. The RNN-based approach that has convLSTM cells in between Encoder and Decoder has the fps of 7 to 9 which is lower than our proposed method of ours. After a few trade-offs with accuracy to gain performance, our model outperformed other models to detect real-time violence with satisfactory accuracy and performance.

*5) Detection Process in CCTV:* Our model operated in a CCTV system with video pieces of similar duration. Each of these fragments was preprocessed using the dense optical flow technique, which generated the input to the ViolenceNet model, which was in charge of classifying these fragments as violent or non-violent. The model identifies each fragment based on the light flow provided by a camera in a CCTV system. The violent segments are represented by the red box, while the non-violent segments are represented by the blue box.

## V. Conclusion

Experimentation with four datasets that are benchmarks in the field of violence detection shows that our proposed method outperforms the other state-of-the-art violence detection methods in terms of performance and accuracy, and successfully detect violence in resource constraint device like Nvidia Jetson Nano. The model described here, can capture temporal information successfully in both directions, and is a good solution to cope with increasingly heterogeneous datasets. Our proposed model has been implemented with Jetson Nano on CCTV cameras to detect real-time violence and it is able to notify the nearest authority with server-less; on device platform which is one of our motivations for the study and very new to this field mitigating all problems of cloud-based systems. Finally, while our model does not include any human features and performs appropriately given the input dataset, it would be necessary to include pose estimation or at the very least face identification in future work to accomplish a generalization of violence involving people. Despite the strong solution we provided with 12 FPS there is still room for more improvement in terms of FPS that we hope to see in the future. Additionally, implementing a new data structure to process the data segment parallelly with multi-threading and without overlapping frames on GPU cores can reduce cost by which it will be possible to run multiple cameras with one GPU. We feel that more research into this topic will yield positive outcomes.

## References

[1] Jeong, C. Y., & Kim, M. (2019). An energy-efficient method for human activity recognition with segment-level change detection and deep learning. Sensors, 19(17), 3688.
[2] Shakhnoza, M., Sabina, U., Sevara, M., & Cho, Y. I. (2021). Novel video surveillance-based fire and smoke classification using attentional feature map in capsule networks. Sensors, 22(1), 98.
[3] Hossain, M. A. (2014). Framework for a cloud-based multimedia surveillance system. International Journal of Distributed Sensor Networks, 10(5), 135257.
[4] Mell, P., & Grance, T. (2011). The NIST definition of cloud computing.
[5] Zeng, W., Zhao, Y., Ou, K., & Song, W. (2009, November). Research on cloud storage architecture and key technologies. In Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human (pp. 1044-1048).
[6] Wang, C., Chow, S. S., Wang, Q., Ren, K., & Lou, W. (2011). Privacy-preserving public auditing for secure cloud storage. IEEE transactions on computers, 62(2), 362-375.
[7] Zhu, W., Luo, C., Wang, J., & Li, S. (2011). Multimedia cloud computing. IEEE Signal Processing Magazine, 28(3), 59-69.
[8] Zhou, Y., Sun, X., Zha, Z. J., & Zeng, W. (2018). Mict: Mixed 3d/2d convolutional tube for human action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 449-458).
[9] Wang, J., & Xu, Z. (2016). Spatio-temporal texture modelling for real-time crowd anomaly detection. Computer Vision and Image Understanding, 144, 177-187.
[10] Mahmood, M., Jalal, A., & Sidduqi, M. A. (2018, December). Robust spatio-temporal features for human interaction recognition via artificial neural network. In 2018 International Conference on Frontiers of Information Technology (FIT) (pp. 218-223).
[11] Tan, C. (2021, July 27). Jetson Nano vs Raspberry Pi 4: The differences. All3DP. Retrieved June 9, 2022, from https://all3dp.com/2/raspberry-pi-vs-jetson-nano-differences/
[12] UZUN, F. N., KAYRICI, M., & AKKUZU, B. (2021). Nvidia Jetson Nano Development Kit. Programmable Smart Microcontroller Cards, 82.
[13] Choe, M., Lee, S., Sung, N. M., Jung, S., & Choe, C. (2021, October). Benchmark Analysis of Deep Learning-based 3D Object Detectors on NVIDIA Jetson Platforms. In 2021 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 10-12).
[14] Kim, J., & Cho, J. (2021). Low-cost embedded system using convolutional neural networks-based spatiotemporal feature map for real-time human action recognition. Applied Sciences, 11(11), 4940.