VigilantEye (Violence Detection System)

Hesham Ayman♣ **Heba Mohamed**♣ **Maryam Ahmed**♣
Sama Hegazy♣ **Mohamed Nasr**♣
T.A.Mirna Al-Shetairy♣ **Prof. Dr. Abeer Mahmoud**♣

♣*Faculty of Computer and Information Sciences, Ain Shams University*

***Abstract*—** Due to the increasing worries about the general public's safety, many surveillance cameras have been installed over the years to increase security. As video surveillance has become more widespread, the amount of video data has become too immense to be tracked by humans efficiently. Additionally, in case video evidence is found by the authorities for previous crimes, a great amount of effort is exerted until the very vital few seconds that have the vital information are found, therefore making there a need for a system able to automatically detect violence. Computer vision algorithms have been getting increasingly effective at object detection- including moving objects, as well as human action recognition. In this paper, we developed VigilantEye, a user-friendly application that enables users to upload videos or start a livestream, for the purpose of being able to determine whether violent actions are being committed in said video or not. Our work was focused on achieving two main goals: building a model that detects violent actions in videos efficiently, and another that clusters similar types of violence seen in our dataset together, using unsupervised learning, to further classify the type of violence detected in videos. The proposed methods are tested on a publicly available RWF-2000 dataset, on which different deep learning algorithms were applied. Our solution achieves an accuracy of 89.25% using a Sep-ConvLSTM model. As for the clustering problem, we found that DBSCAN is the best fitting as it was successful in grouping the data into 2 groups**.**

I. INTRODUCTION

In recent years, due to the abundance of digital video capturing, storage, and processing technologies, as well as the rise in crime rates, the deployment of video surveillance cameras has become very common all over the globe. Despite being an accessible and efficient solution, it requires great human effort to keep track of all surveillance cameras 24/7, which makes their efficiency heavily dependent on the person monitoring them, whose attention could be lacking at the particular timeframe when the violent action occurs. In the aftermath, a massive amount of recorded footage would have to be manually reviewed, solely for the purpose of finding the frames necessary -which would only span a few seconds- to acquire the key evidence, making it an extremely time-consuming and labor-intensive process

Video classification using human action recognition is a popular research topic in recent times and is relevant to the field of violence detection [1]. While older research focused on various traditional machine-learning techniques for automated violence detection, recent advancements have shown that deep-learning methods yield far better results, as they can effectively extract spatio-temporal features - data that is collected across both space and time - from videos.

In addition to being able to determine whether the video footage is violent or not, building a model that can accurately cluster similar types of violence together can prove to be extremely useful, as different types of violence have different severities, and require different courses of action.

In the upcoming sections of the article, we review related work in violence detection. Then, we discuss the datasets used, the methodology implemented for each system module, the results achieved, what was gathered from our analysis of the dataset, and compare our proposed solution to previously implemented solutions.

II. LITERATURE REVIEW

Vidas Raudonis et al. [2] developed a computer vision and machine learning model to reduce reaction time in public health and safety cases. The model uses a U-Net-like network with a MobileNet V2 network model for static single-frame feature extraction, LSTM for sequential information, and a binary cross-entropy loss function for video classification. The model achieved an average accuracy of 82.0 ± 3% for the RWF-2000 dataset, 96.1 ± 1% for the Hockey Fight dataset, and 99.5 ± 2% for the Movie Fight dataset.

Juan C. San Miguel et al. [3] developed a deep-learning architecture for detecting violent crimes in surveillance videos using human pose extractors and change detectors. The architecture, using ConvLSTM for efficient two-stream architectures, achieved high model accuracy and computational efficiency, with the best model achieving 90.25% in the validation set and 92.37% in the training set after 50 epochs.

Islam, Zahidul et al. [4] proposed a two-stream deep learning architecture for detecting aggressive human behaviors like fighting, robbery, and rioting, suitable for unmanned security monitoring systems and internet video filtration. The architecture uses Separable Convolutional LSTM (which reduces computation and parameter count, producing localized Spatio-temporal features) and MobileNet (pre-trained on ImageNet dataset) as the CNN for extracting spatial features. The best model achieved 89.75% test accuracy for the RWF-2000 dataset.

Manan Sharma et al. [5] developed a deep learning model for video classification, using a combination of CNN and LSTMs. The ConvLSTM cell captures spatial features in multi-dimensional data, combining pattern recognition of ConvNets with memory properties of pure LSTM networks. Three datasets were used: KTH, Violent Flows, and Hockey. The model achieved 100% accuracy, 80% accuracy, and 87.5% accuracy, with four reducing points in the learning curve. The best CNN model was ResNet50 CNN with 89.9% accuracy.

Paolo Sernani et al. [6] compared three deep learning models on the AIRTLab dataset, including 350 full HD clips at 30 frames per second. They proposed two transfer learning-based models and one "trained from scratch" model. The models achieved an average accuracy of 97.15 ± 1.5% for the AIRTLab dataset, 97.86 ± 0.8% for the Hockey Fight dataset, and 99.06 ± 0.5% for the Crowd Violence dataset. However, the research has limitations due to the lack of real violence in the dataset.

III. DATASETS

The definition of video-based violence detection is detecting violent behaviors in video data; the availability of a suitable dataset plays a vital role in training and evaluating a violence detection system.

There are two kinds of video datasets for violence detection: trimmed and untrimmed. The videos in trimmed datasets are all short clips with a length of several seconds, and each one of them has a video-level annotation. While the videos in untrimmed RWF-2000 Dataset

RWF-2000 dataset is composed of videos captured from real-life situations. The RWF (Real-World Fight) dataset was utilized for the development and evaluation of violence detection systems. The RWF dataset is a widely recognized benchmark dataset specifically designed for violence detection tasks in real-world scenarios. Figure 4.1 demonstrates some video examples in the RWF-2000 dataset.

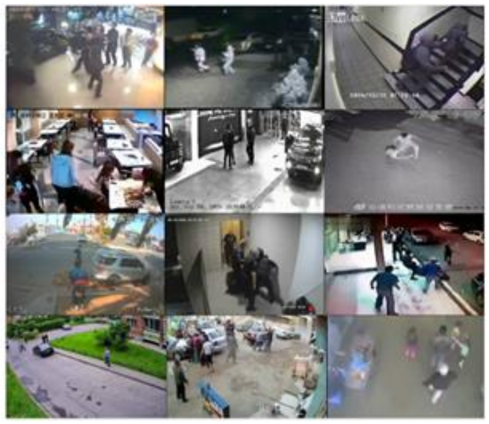

Figure 3.1. video examples in the RWF-2000 dataset.

It has 2000 videos captured by surveillance cameras in real-world scenes, 1000 of which are violent, while the other 1000 are non-violent. All the videos were downloaded from YouTube, and the types of violence captured weren’t specified, comprising many forms of identified violent activities, such as fighting, robbery, explosion, shooting, blood, and assault.

Following the data collection process, numerous raw videos are obtained. Each video is subsequently segmented into 5-second clips with a frame rate of 30 FPS, with various resolutions (e.g., 720P, 1080P, 2K, 4K). Noisy clips that contain unrealistic and non-monitoring scenes were removed. Additionally, each remaining clip is annotated as either "Violent" or "Non-Violent."

Data Analysis

The main objective was to categorize violent types within the RWF-20000 dataset. The feature extraction process involved loading and resizing video frames to a uniform size of (150x160x160). These frames were then fed into a 3DCNN model, and the outputs of each layer were obtained using functors. The focus was on the second-to-last layer's output, which served as the feature vector with 32 dimensions. Due to the high dimensionality, dimensionality reduction was performed before applying the clustering algorithm in the first experiment.

To evaluate the experiment's results, a two-person testing approach was employed to carefully examine the clustering outcomes and identify distinct groups representing different violence types. However, challenges arose in obtaining clear and meaningful results. The dimensionality reduction step was suspected of causing the loss of important information from the feature vector. Consequently, a second experiment was conducted where clustering was performed directly on the original feature vector without dimensionality reduction. Instead, dimensionality reduction techniques were applied later to visualize the clustering results in a more interpretable manner. We also evaluated this experiment by a two-person testing approach, In the case of DBSCAN clustering, which does not require a predefined number of clusters, the results indicated that the data was grouped into two clusters. However, with careful examination, we found that these clusters did not match with the known types of violence that we are familiar with.

IV. METHODOLOGY

Violence detection as a concept relies on a model's ability to detect human action in the first place, and then classify whether it is violent or not. While older research focused on various traditional machine-learning techniques, recent advancements have shown that deep-learning methods yield far better results [6]. For all methods, video frames were resized and normalized before the models were trained.

A. SepConvLSTM

In a traditional LSTM, an RNN, the model passes the previous hidden state to the next step of the sequence, therefore making it possible for old information to be retained to make future decisions. In order to utilize this advantage, as well as the CNN's abilities used for image processing, the ConvLSTM was designed to combine these two models to handle spatiotemporal data that involve both spatial and temporal dependencies.

SepConvLSTM [4] is a deepwise separable convolution method that produces robust long-range spatiotemporal features using fewer parameters. The proposed pipeline consists of two streams of CNN and SepConvLSTM modules, each composed of a condensed version of the MobileNet module. The outputs from each stream are combined to produce robust spatiotemporal features. The model uses MobileNetV2 (α = 0.35) [7] as a CNN to extract spatial features, and the last 30 layers of MobileNet models were shortened. Local spatiotemporal features are generated from CNN's output feature maps. SepConvLSTM has been used in the past to speed up video segmentation tasks. Shape 224×224 ×3 frames are fed into the model, and CNN collects spatial characteristics of shape 7 × 7 × 56 from each stream. The outputs are fused using a window size (2,2) Max-Pooling layer and the binary cross-entropy loss is computed. Three fusion algorithms are developed to combine the output feature maps from the two routes, making up SepConvLSTM-M, SepConvLSTM-C, and SepConvLSTM-A versions of the suggested model.

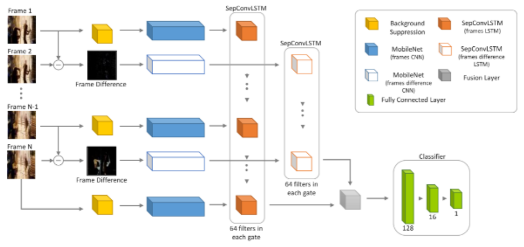

Figure 3.1. Structure of the two CNN-LSTM streams that comprise the proposed model

- SepConvLSTM-M

The LeakyRelu activation layer is used to process frame streams' output, while a Sigmoid activation layer processes feature maps from the frame difference stream, then multiplying each element independently.

$$F_{fused} = LeakyRelu(F_{frames}) \oplus Sigmoid(F_{diff}) \quad \textbf{(1)}$$

Here, $Fframes$ and $Fdiff$ denotes the feature maps from frames stream and frame difference stream respectively. $Ffused$ is the output feature map of the Fusion layer.

- SepConvLSTM-C

To send the data to the classification layers, we just concatenate the two output features of the two streams.

$$F_{fused} = Concat(F_{frames}\,,\, F_{diff}) \quad \textbf{(2)}$$

Here, the Concat function concatenates $Fframes$ and $Fdiff$ along the channel axis.

- SepConvLSTM-A

The output feature maps of the two streams are added element by element

$$F_{fused} = (F_{frames}) \oplus (F_{diff}) \quad \textbf{(3)}$$

Here, $\oplus$ refers to element-wise addition operation combining the output feature maps of the two streams.

The SepConvLSTM-C model was used because it had the best result. It was trained for approximately 120 epochs or until the model began to overfit. The CNNs were initialized using weights previously trained on the ImageNet dataset. Used the Xavier initialization [8] of the SepConvLSTM kernel. The Hockey [9] and Movie [9] datasets are very small, which can cause overfitting. Therefore, we first trained on the RWF-2000 dataset. Next, the weights of this trained model will start training on the other two datasets. To improve the model, we used the AMSGrad variant of the Adam optimizer [10]. Start training with a learning rate of 4 × 10^−4. After every 5 epochs, we halved the learning rate until it reached 5 × 10^−5. Keep it unchanged from that era. The model is optimized to minimize the sigmoid loss between the ground truth and the predicted label.

B. C3D [11]
A popular 3D-CNN architecture for learning spatiotemporal (data has information about the location (space) and the time (temporal)) features from video data. C3D is specialized for action recognition and can be applied to various data analysis tasks.

The aim is to design a temporal pooling mechanism achieved by network self-learning. The structure of our proposed model with four parts: the RGB channel, the Optical Flow channel, the Merging Block, and the Fully Connected layer, as shown in Figure 3.1.

Optical flow is a video task that estimates per-pixel motion between frames, calculating the shift vector for pixels. It estimates the displacement vector caused by motion or camera movements. Dense optical flow computes the vector for every pixel, providing more accurate results for detecting motion, segmenting, and learning structure from motion.

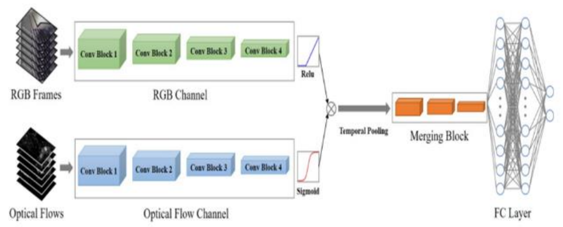

Figure 3.2. The structure of the Flow Gated Network

The RGB and Optical Flow channels utilize cascaded 3DCNNs for fused output, while the Merging Block processes information after self-learned temporal pooling and fully connected layers generate output.

This model uses a branch of the optical flow channel to build a pooling mechanism, adopting relu activation at the end of the RGB channel and a sigmoid function at the end of the optical flow channel. The outputs from both channels are multiplied and processed by temporal max pooling, with the sigmoid function acting as a scaling factor. This self-learned pooling strategy uses a branch of optical flow as a gate to determine which information to preserve or drop [11]. The model was trained for 30 epochs for all trials, as we tried to focus on maximizing the accuracy by adding optimization techniques.

C. 3DCNN [11]
CNN is a class of artificial neural networks that has become dominant in various computer vision tasks and is used to identify and extract features from images through backpropagation, by building blocks, which consist of convolution layers, pooling layers, and fully connected layers. One key feature is spatio (Captures features like edges, textures, and shapes within individual frames or slices), and temporal (Detects motion patterns, changes, and temporal dependencies across frames or slices) features. The structure of our proposed model with blocks of convolution layers followed by batch normalization layers, followed by pooling layers. Then the fully connected layers, as shown in Figure 3.3.

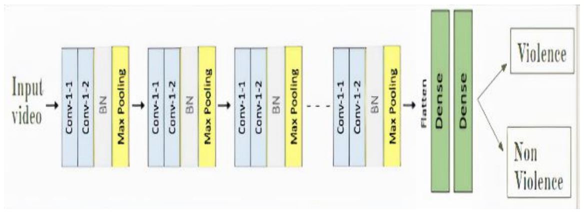

Figure 3.3. Architecture of the 3DCNN Model

As Figure 3.3 shows, the model architecture is like C3D architecture, explained in the previous section, without the branch of the optical flow channel. For all trials, the model was trained for 40 epochs. For later trials, various optimization techniques were added, including additional batch normalization layers and an L2 Kernel Regularizer, in an attempt to maximize accuracy. In some cases, we faced problems like being unable to increase the frame size over 160×160 or the batch size over 6 for the data generator used, due to lack of computational resources.

D. Transformer-based Model
Transformers are a type of neural network architecture that transforms or changes an input sequence into an output sequence, through learning context and tracking relationships between sequence components.

To extract features from the videos, DenseNet-12, a type of convolutional neural network that uses dense connections between layers, is used, ensuring a feed-forward nature. Each layer receives inputs from previous layers and passes on its feature maps to subsequent layers.

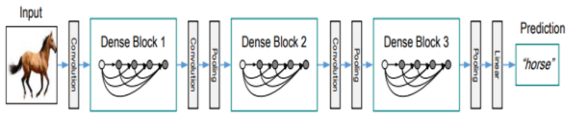

Figure 3.4. DenseNet Architecture

Our transformer model is built based on the architecture of the original transformer developed by Google [12], but is composed only of the encoder part and not the decoder, as shown in the architecture shown in Figure 3.4.

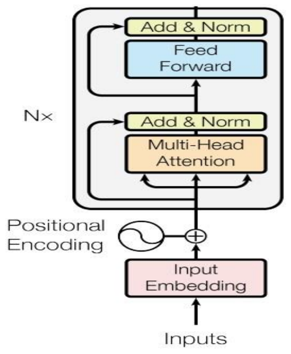

Figure 3.5. Structure of the Encoder

Positional encoding helps understand the order of elements within a sequence. It compensates for the lack of order in self-attention mechanisms. The architecture involves initializing an embedding layer, building it according to the input shape, and adding positional embeddings during the forward pass. **Transformer encoders** are used in various tasks involving sequential data, such as NLP and time series analysis. The key components include multi-head self-attention, a feed-forward neural network, and layer normalization, as shown in Figure 3.4. Different trials were conducted using different hyperparameters and optimizers to extract 1024 features from videos. The model was trained for 20 epochs using an Adam optimizer in a trial, and for 50 epochs using the SGD optimizer in another trial which achieved higher accuracy than other trials. Various preprocessing techniques were used to improve accuracy, but these often resulted in lower test accuracy and model overfitting.

Table 5.1. Results of all models of the RWF-2000 dataset

V. RESULTS

Accuracy and Confusion Matrix are used to evaluate the models. As shown in Table 5.1, we combined classification reports for the best trials of each model.

As shown in Table 5.2., we achieved high accuracy with the ConvLSTM model compared with other models, which is 89% test accuracy.

Table 5.2. comparison between previous work and our work

Method	Accuracy
Related Work	
SepConvLSTM-C[4]	**89%**
C3D [11]	85.75%
P3D [11]	87.25%
3DCNN (RGB only) [11]	85%
Our Work	
ConvLSTM	**89%**
C3D	80%
3DCNN	78%
Transformer	79%

	ConvLSTM			**C3D**			**3DCNN**			**Transformer**			**support**
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	
0	0.87	0.91	0.89	0.74	0.92	0.82	0.74	0.85	0.79	0.76	0.84	0.80	**200**
1	0.91	0.86	0.88	0.89	0.68	0.77	0.83	0.69	0.76	0.82	0.74	0.78	**200**
Accuracy	**0.89**			0.80			0.78			0.79			**400**
Macro avg.	0.89	0.89	0.88	0.82	0.80	0.79	0.78	0.77	0.77	0.79	0.79	0.79	**400**
Weighted avg.	0.89	0.89	0.88	0.82	0.80	0.79	0.78	0.78	0.77	0.79	0.79	0.79	**400**

The study compared the results of using the ConvLSTM model and fine-tuning achieving the same results. However, the C3D model's results differed due to different preprocessing techniques, layer changes, and resource limitations. The 3DCNN model, which deals with RGB channels only, achieved 78% test accuracy. The transformer model, which was not used in the study due to a lack of resources, was used and trained from scratch with different hyperparameters and preprocessing techniques, resulting in different results.

VI. Dataset Analysis

VII. Conclusion and Future Work

The project aimed to facilitate the detection of widespread violence by aiding government institutions and individuals like parents, and facilitating prompt action from competent authorities The study uses innovative methods, such as SepConvLSTM, to detect violent activity in surveillance footage. The network effectively learns discriminative spatiotemporal features, achieving high recognition accuracy. It's computationally efficient, suitable for time-sensitive applications and low-end devices, and offers a compact alternative to ConvLSTM.. We performed data analysis to identify and categorize the violent types within RWF-20000 dataset. We extracted meaningful features from a trained 3D Convolutional Neural Network (3DCNN) model to obtain a feature vector. We performed dimensionality reduction using techniques such as t-SNE and UMAP, then applied clustering algorithms, including K-means and DBSCAN. However, with careful examination, we found that these clusters did not match with the known types of violence that we are familiar with.

VIII. References

[1] V. Huszar, V. Adhikarla, I. Negyesi, and C. Krasznay, "Toward Fast and Accurate Violence Detection for Automated Video Surveillance Applications," *IEEE Access*, vol. PP, p. 1, Jun. 2023, doi: 10.1109/ACCESS.2023.3245521.

[2] R. Vijeikis, V. Raudonis, and G. Dervinis, "Efficient Violence Detection in Surveillance," *Sensors*, vol. 22, no. 6, Mar. 2022, doi: 10.3390/s22062216.

[3] G. Garcia-Cobo and J. C. SanMiguel, "Human skeletons and change detection for efficient violence detection in surveillance videos," *Computer Vision and Image Understanding*, vol. 233, p. 103739, Aug. 2023, doi: 10.1016/J.CVIU.2023.103739.

[4] Z. Islam, M. Rukonuzzaman, R. Ahmed, Md. H. Kabir, and M. Farazi, "Efficient Two-Stream Network for Violence Detection Using Separable Convolutional LSTM," Feb. 2021, doi: 10.1109/IJCNN52387.2021.9534280.

[5] M. Sharma and R. Baghel, "Video Surveillance for Violence Detection Using Deep Learning," 2020, pp. 411–420. doi: 10.1007/978-981-15-0978-0_40.

[6] P. Sernani, N. Falcionelli, S. Tomassini, P. Contardo, and A. F. Dragoni, "Deep Learning for Automatic Violence Detection: Tests on the AIRTLab Dataset," *IEEE Access*, vol. 9, pp. 160580–160595, 2021, doi: 10.1109/ACCESS.2021.3131315.

[7] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2018, pp. 4510–4520. doi: 10.1109/CVPR.2018.00474.

[8] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks." [Online]. Available: http://www.iro.umontreal.

[9] E. B. Nievas, O. Deniz Suarez, G. Bueno García, and R. Sukthankar, "Violence Detection in Video Using Computer Vision Techniques." [Online]. Available: http://visilab.etsii.uclm.es/

[10] S. J. Reddi, S. Kale, and S. Kumar, "On the Convergence of Adam and Beyond," Apr. 2019.

[11] M. Cheng, K. Cai, and M. Li, "RWF-2000: An Open Large Scale Video Database for Violence Detection," Nov. 2019.

[12] A. Vaswani *et al.*, "Attention Is All You Need," Jun. 2017.