HUMAN VIOLENCE DETECTION USING VIDEOS

Abstract

In today's world, maintaining public safety in crowded or vulnerable spaces is of critical importance. Conventional surveillance systems place a significant emphasis on human monitoring, which can be unreliable and slow to identify violent incidents. The system is going to analyze video footage and use advanced machine learning techniques, such as convolutional neural networks (CNNs) and recurrent networks (RNNs), to identify patterns linked to physical violence, like fights or aggressive actions.

This system's integration with the current surveillance infrastructure can improve security in public areas like malls, schools, and transit hubs by cutting down on the amount of time it takes to respond to violent incidents. This technology could completely change video surveillance by offering an automated, scalable way to increase public safety without requiring constant human supervision.

1. Background and Motivation

Violence in public areas can seriously jeopardize public safety, including streets, malls, and schools. Conventional surveillance camera monitoring techniques depend on human operators, who might not always notice problems in a timely manner. Real-time alerts can be provided by AI-based systems that automatically scan video feeds, identify violent activity, and immediately notify the appropriate authorities.

2. Problem Identification

Manual surveillance struggles with real-time violence detection, leading to delays and errors. The task is to develop an automated system that can accurately classify violent and non-violent actions in short video clips.

- **Input**: a short video clips
- **Output**: A binary classification indicating whether the video contains violent or non-violent behaviour.

The system will use deep learning models to process spatial and temporal video data for efficient, real-time violence detection.

3. Related Work

3.1. Video Datasets for Violence Detection

Early datasets like Hockey Fight and Movies Fight: Limited in size and diversity, focusing on specific contexts (sports, movies).

3.2. Crowd Violence

- Highlighted crowded environments but suffered from low image quality.
- Recent datasets like CCTV-Fights and UCF-Crime: Introduced real-world surveillance videos but had issues like long durations and coarse annotations.

3.3. RWF-2000 Dataset

- Contains 2,000 trimmed clips of real-world violent incidents captured by security cameras.
- Balanced categories (violent/non-violent) and ensures minimal training/test overlap.

3.4. Methods for Violence Detection

❖ **Traditional Approaches**:

- Used handcrafted features (e.g., iDT, MoSIFT, STIPs) with shallow classifiers (e.g., SVM).
- Worked well for simple cases but lacked generalization.

❖ **Deep Learning Approaches**:

- Popular architectures: ConvLSTM, C3D, Two-Stream Networks.
- Combine spatial and temporal features, often using RGB and optical flow inputs.
- Require diverse datasets to mitigate overfitting but outperform traditional methods.

❖ **Gaps Identified**

- High computational costs (e.g., 3D CNNs, optical flow).
- Dependency on noisy data (e.g., raw video frames with background noise).
- Challenges in real-time processing.

❖ **How This Paper Advances the Field**

- Simplifies input by focusing on **skeletons** (essential body movements) and **frame differences** (inter-frame dynamics).
- Combines both pipelines using an **addition-based fusion** for robustness against pipeline failures.
- Uses **ConvLSTM** for efficient spatio-temporal aggregation with fewer parameters.
- Achieves real-time performance with superior accuracy compared to prior models.

3.5. Approaches

3.5.1. Classic Approaches

❖ **Feature-Based Methods:**

- **Spatio-Temporal Interest Points (STIP) :** Detects significant intensity variations in space and time.
- **MoSIFT:** Combines SIFT descriptors with motion information.

❖ **Violent Flow Descriptions:**

- **Violent Flows (ViF):** Analyzes optical flow changes over time.
- **Oriented Violent Flows (OViF):** Adds flow orientation for improved performance in uncrowded scenarios.

3.5.2. Deep learning Approaches

❖ **3D CNNs:**

- **Ding et al. (2014):** Extends 2D CNNs for spatio-temporal feature extraction but increases model complexity.
- **Li et al. (2019):** Introduces an efficient 3D CNN with reduced kernel size.

❖ **Hybrid models:**

- **CNN + LSTM:** Combines spatial feature extraction with temporal sequence analysis.
- **ConvLSTM:**Uses convolutional LSTMs for spatial and temporal aggregation.

❖ **Two stream models:**

- **Flow Gated Networks:** Combines spatial and temporal pipelines with optical flow.
- **Separable ConvLSTM:** Processes raw videos and frame differences independently.

3.5.3. Posed-based Approaches

❖ **Skeleton Points Interaction Learning (SPIL):** Proposed by Su et al. (2020): Uses Graph Neural Networks to analyze human skeletons and their interactions.

❖ **Pose-Driven Fusion:** Combines pose and temporal features for enhanced violence detection.

3.6. Challenges in existing work

Existing methods face challenges in effectively combining spatial and temporal features, which are crucial for detecting dynamic events like violence. This results in incomplete feature representation, reducing detection accuracy for complex scenes. Models trained on specific datasets often perform poorly in real-world scenarios with varying environments, lighting conditions, and camera angles. This lack of robustness limits the practical deployment of these models.

Many advanced methods have high computational demands, making them unsuitable for real-time applications. High latency and resource requirements hinder the adoption of such methods in live monitoring systems. Real-world surveillance data often contain incomplete or occluded information, such as missing skeletons or low-resolution frames. Models relying on clean data fail in handling noisy or partially corrupted inputs, reducing reliability.

4. Methodology

4.1 Dataset:

The Dataset: **RWF2000** - A Large Scale Video Database for Violence Detection

- This data was collected from YouTube, sliced them into clips with 5s at 30 fps, and labelled each clip as **Fight** or **NonFight**
- The dataset consists of 2000 videos, each video is 5 seconds long and shot at 30 frames (FPS)

4.2 Model Development:

Model Selection: We will experiment with state-of-the-art deep learning models such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM), and 3D CNNs for spatiotemporal analysis.

Here are 3 approaches:

- **CNN-LSTM**: using CNN to extract feature of each frame and feed them into an LSTM network for sequential analysis
- **Video ViT (ViViT)**: leverages the Transformer architecture, which has shown exceptional performance in vision tasks, particularly for video analysis. The ViViT model will be trained to classify entire video clips, treating each frame as a token in a sequence, allowing it to learn complex spatiotemporal relationships.
- **Human Skeleton and change detection (SOTA)**

4.3 CNN-LSTM:

CNN and LSTM, the two primary components of a CNN-LSTM model for video violence detection, collaborate to examine temporal and spatial patterns in video sequences.

4.3.1 Data Preparation

- **Frame Extraction**: Videos in the dataset are divided into sequences of frames, where each sequence represents a short segment of a video.
- **Labelling**: Each video segment is labelled as either "violence" or "non-violence," or possibly with more granular labels if there are multiple classes.
- **Data Augmentation**: To improve robustness, augmentations like cropping, flipping, and adjusting brightness can be applied to increase the diversity of the dataset.
- **Normalizing and Reshaping**: Each frame is resized and normalized to prepare it for input into the CNN layers. This often includes scaling pixel values between 0 and 1 or normalizing them around a mean.

Model Initialization
The model architecture is defined with CNN layers for spatial feature extraction, LSTM layers for temporal pattern learning, and fully connected layers for final classification.

Hyperparameters like learning rate, batch size, and number of epochs are set. These control the learning process and the number of passes through the entire dataset.

4.3.2. CNN-LSTM Approach

❖ **CNN Used**
The CNN models utilized in this approach include **MobileNetV2, Xception, and ResNet101V2**. All CNNs were initialized with **pre-trained weights** from ImageNet dataset, a large-scale dataset for image classification tasks, to leverage transfer learning and provide a strong baseline for feature extraction. To adapt the pre-trained models to the RWF-2000 dataset: During the initial training phase, the convolutional layers of the CNNs were frozen, ensuring the pre-trained weights remained intact for feature extraction.
Fine-tuning was applied to adapt the pre-trained models to the violence detection task, the last 40 layers of each CNN were unfrozen for fine-tuning. This allows task-specific learning while retaining the generalization capability of the initial layers.
About feature extraction: The CNNs are wrapped in a Time Distributed layer to process sequences of video frames individually. Each frame is resized to 224x224 and normalized (pixel values scaled to [0, 1]). The output of the CNN is a sequence of 2048-dimensional feature vectors, one for each frame.

❖ **LSTM Parameters**
The temporal modeling is achieved through a Bidirectional LSTM with the following configurations:

- **Forward and Backward LSTMs**: Both forward and backward LSTMs are configured with 32 hidden units each. Bidirectional wrapping allows the model to capture dependencies from both past and future frames.
- **Dropout Regularization**: Dropout layers with a rate of 0.25 are applied after the LSTM layer and dense layers to mitigate overfitting.
- **Dense Layers**: A series of fully connected layers follow the LSTM, with dimensions of 256, 128, 64, and 32 units, each activated with ReLU. The final dense layer uses a sigmoid activation for binary classification (violence vs. non-violence).

- **Training Configuration**
- **Pipeline Design**: Frames are processed individually through the CNN layers wrapped in a TimeDistributed layer, ensuring consistent feature extraction across sequences. Extracted spatial features are flattened and passed as sequences to the Bidirectional LSTM, which learns temporal dependencies. Fully connected layers process the LSTM outputs to make the final prediction. The LSTM outputs are passed through the fully connected layers for classification into "violence" or "non-violence."
- **Optimization**: **Optimizer**: Adam with a learning rate of 0.0005. **Loss Function**: Binary cross-entropy to suit the binary classification task. **Learning Rate Schedule**: A custom learning rate scheduler adjusts the learning rate dynamically based on training epochs. **Batch Size:** The batch size was set to 8 to balance memory efficiency and model performance. **Callbacks:** Early stopping and learning rate reduction were utilized to halt training if validation performance stagnated, ensuring efficient use of resources.

4.3.3. Forward Pass

For each batch of video sequences:

- **CNN Processing**: Each frame in a sequence passes through the CNN layers, producing feature maps that represent spatial information.
- **LSTM Processing**: These feature maps are then fed sequentially to the LSTM, which processes the entire frame sequence and learns dependencies between frames.
- **Output Layer**: The final LSTM output goes through fully connected layers, ending with a softmax or sigmoid function to produce probabilities for each class.

4.3.4. Model Testing

Once training is complete, the model is evaluated on a test set to assess its performance on completely unseen data.

The test set should ideally represent real-world conditions for violence detection, helping to ensure that the model performs well outside of the training and validation environments.

4.3.5. Results

❖ **Accuracy vs Validation Accuracy**

- **Training Accuracy :** The training accuracy increased steadily over the epochs, reaching a final accuracy of 69.4%. This upward trend indicates that the model successfully learned patterns from the training data.
- **Validation Accuracy :** The validation accuracy fluctuated significantly during training. It peaked sharply at around 80% in epoch 2 but dropped significantly afterward, averaging around 55-65% in subsequent epochs. This inconsistency in validation performance suggests that the model struggled to generalize to unseen validation data.

❖ **Observations:**

- **Overfitting**: The noticeable gap between the training accuracy (69.4%) and the fluctuating validation accuracy suggests overfitting. This occurs when the model performs well on the training data but struggles to generalize to new, unseen data.
- **Validation Accuracy Peak and Drop:** The peak at epoch 2, followed by a significant decline, indicates that the model briefly captured useful patterns but lost its ability to generalize. This could be attributed to: Overfitting caused by insufficient regularization or dropout. An under-representation of certain patterns in the validation dataset.
- **Generalization Issues:** The inconsistent validation accuracy highlights potential generalization issues. These could stem from: Imbalanced or inadequate diversity in the training dataset. Suboptimal tuning of model hyperparameters.

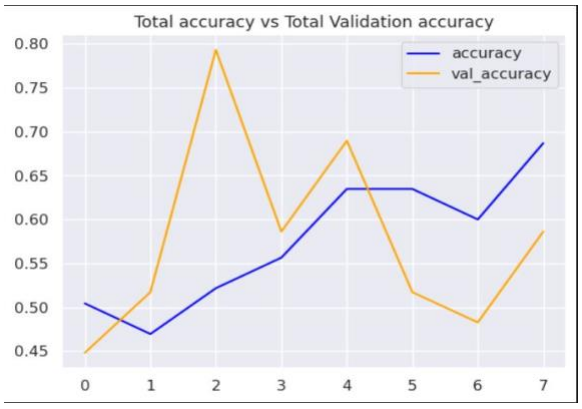

Figure 1. Training Accuracy vs Validation Accuracy

❖ **Loss vs Validation Loss**

- **Training Loss** : The training loss consistently decreases, which is a good sign that the model is learning from the training data and adjusting weights to minimize errors.
- **Validation Loss**: The validation loss also decreases over time but at a slightly slower rate and it briefly increases around epoch 3 before continuing to decline.

Observations:

- The decrease in both training and validation loss over time is positive, showing that the model is improving.
- However, the initial fluctuation in validation loss could be an early sign of overfitting, and the fact that the validation loss remains higher than the training loss suggests that the model might not be fully capturing the generalizable patterns required for violence detection.

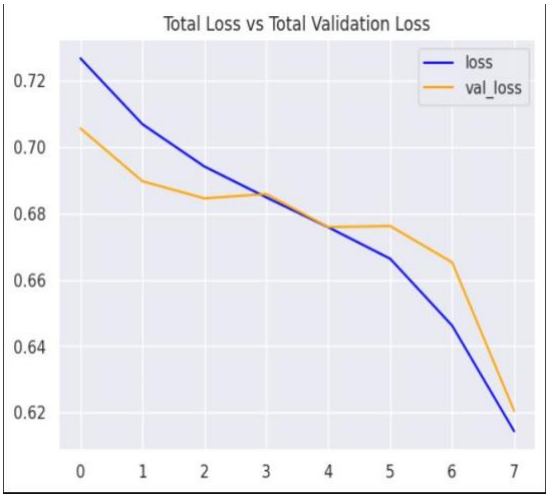

Figure 2. Training Loss vs Validation Loss

4.4 Video ViT:

The Video Vision Transformer (ViViT) model was chosen due to its ability to capture complex spatiotemporal relationships within video data.

***4.4.1 Model details*:** A Video Vision Transformer (pure- Transformer based model) was chosen for its ability to process both spatial and temporal information.

- **Patch Extraction**: Each video frame was split into small patches, which were then flattened and used as tokens for the transformer.
- **Embedding and Temporal Positional Encoding**: Each patch was encoded into an embedding vector. Temporal positional encodings were added to maintain the order of frames within a sequence.
- **Self-Attention Mechanism:** Multi-headed self-attention layers enabled the model to capture relationships across patches within and between frames, allowing it to identify violent actions based on motion and context.

4.4.2 Training Process

Data Preprocessing: Video frames were extracted and resized to a fixed resolution. Each sequence consisted of a fixed number of frames (e.g., 15 frames per sequence).

Data augmentation: We applied data augmentation methods to help decrease overfitting by artificially increasing the diversity of the training data, allowing the model to learn more generalized patterns instead of memorizing specific details from the limited dataset. This increased variation in the data helps improve the model's robustness and performance on unseen data.

- Some data augmentation methods applied: Resize, RandomResizedCrop, HorizontalFlip, VerticalFlip, ShiftScaleRotate, CLAHE.

Training and Optimization:

The model was trained with labelled data, using cross-entropy loss as the objective function.

- **Pre-trained Model: google/vivit-b-16x2-kinetics400** (pre-trained on Kinetics-400 dataset).
- **Optimizer:** Adam optimizer with a learning rate scheduler.

Evaluation Metrics: The primary metric for evaluation was accuracy, with additional consideration of precision, recall, and F1-score for more granular performance insights.

4.4.3 Results

Throughout the training epochs, the model demonstrated steady progress in terms of accuracy:
The training accuracy increased rapidly, reaching close to 100% accuracy after a few epochs. By the 19th epoch, the model's training accuracy was effectively 100%, indicating that the model could almost perfectly classify the training data.

Training loss consistently decreased with each epoch, achieving values near zero by the end of training, which aligns with the high training accuracy.
However, the nearly perfect training accuracy suggests that the model might have overfitted on the training data. The introduction of data augmentation appears to have had a positive impact initially, but it may not have been enough to fully prevent overfitting.

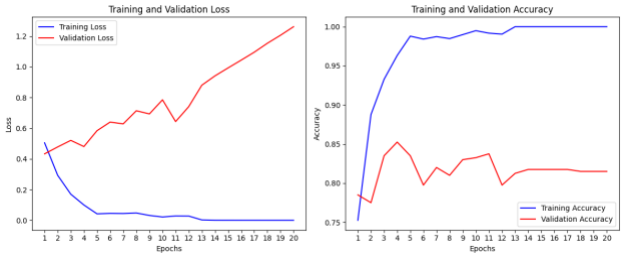

Figure 3. Training and Validation with Data Augmentation - ViViT

Figure 4. Training and Validation without Data Augmentation - ViViT

4.4.4 Discussion

Without Data Augmentation

- **Training Accuracy**: Rapidly reached high accuracy, but showed signs of overfitting, with near-perfect accuracy on the training data.
- **Validation Performance**: Lower validation accuracy with early plateauing and increasing validation loss in later epochs, indicating poor generalization.

With Data Augmentation

- **Training Accuracy**: Initially lower but improved gradually as the model adapted to the variability introduced by augmentation.
- **Validation Accuracy**: Reached a higher peak of 84.25%, showing better

generalization, though some overfitting persisted at later stages.

Impact

Data augmentation improved validation accuracy and generalization, delaying overfitting and making the model more robust. While not entirely preventing overfitting, augmentation proved effective in enhancing performance on unseen data. Future work could explore additional regularization for further gains.

4.5. Human Skeleton and Change Detection

This method focuses on **pose estimation (skeletons)** and **frame difference analysis** to detect violence in videos. By focusing on **human movement patterns** rather than raw pixel data, it reduces noise and focuses on **action-specific dynamics**.

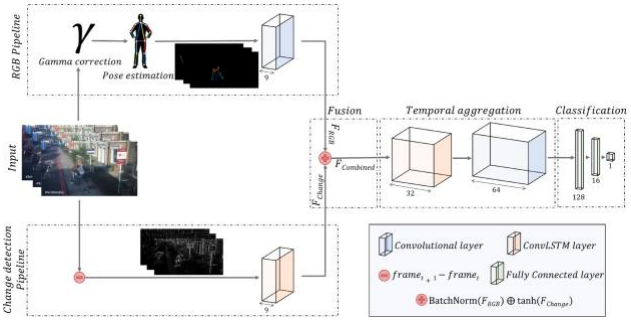

Figure 5. Model Architecture. It comprises two pipelines: one for **extracting people's skeletons** in the scene and

another for **estimating their dynamic temporal changes between frames**.

4.5.1 Human Skeleton Extraction

Human skeleton extraction focuses on capturing key joint points of the human body using pose estimation models. These techniques reduce noise by isolating skeletal structures and avoiding distractions from background elements.

- **Pose Estimation Models** (OpenPose) extract **key skeletal joints** from each video frame.

Openpose was chose because of high accuracy in the multi-skeleton extraction task

- **Skeletons are rendered**: Extracted skeletons are rendered on simplified backgrounds (black screens) to focus on structural details and minimize irrelevant visual information.
    - **With background:** For context awareness.
    - **Without background:** To focus solely on human body structure.

These features are all done by Openpose, including extracting multiple skeletons in 1 frame, outputting as backgroundless (black backround)

Figure 6. Skeletons' representation with and without background

Enhancing Skeleton Extraction

To enhance pose estimation accuracy and violence detection, preprocessing addresses challenges such as poor lighting, occlusions, and low-resolution footage.

(a) Original image (b) Contrast increase

(c) Gamma correction ($\gamma = 0.4$) (d) Histogram equalization

Figure 7. Some preprocessing method to enhance the pose estimation performance

Contrast increase: involves multiplying each pixel by a factor, while accounting for variations in brightness across different regions of the image.
Gamma Correction: Adjusts pixel intensities to enhance contrast, improving visibility in darker frames.
Histogram Equalization: Redistributes pixel intensities for clearer images, aiding pose estimation.
➔ Gamma correction is preferred for its lower computational cost and comparable results.

Advantages of Skeleton-Based Methods

- **Noise Reduction:** Removes background clutter, focusing on body movements.
- **Efficient Representation:** Reduces computational overhead compared to raw pixel-based video processing.
- **Adaptability:** Performs well even in low-resolution or noisy environments.

4.5.2 Change Detection Techniques

Frame difference analysis captures inter-frame motion dynamics by identifying significant changes between consecutive video frames. These temporal changes often correspond to aggressive movements indicative of violence.

Three methods commonly used for change detection in violence detection systems:

- **Frame Difference**: Detects inter-frame motion by capturing changes between consecutive frames, offering rich temporal information with low computational cost.
- **Frame Distance**: Computes the L2 norm between frames, summarizing temporal changes efficiently but losing directional motion details.
- **Optical Flow**: Estimates pixel-level motion vectors for detailed representation but is computationally intensive and sensitive to noise in low-resolution or occluded videos.

➔ **Frame difference** offers a balanced trade-off between computational efficiency and accuracy, providing meaningful temporal cues without requiring significant computational resources.

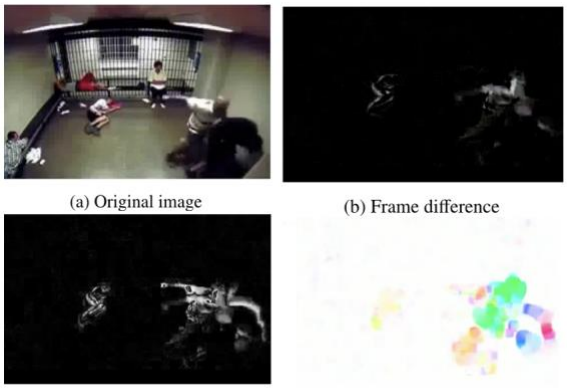

Figure 8. Comparison of change detectors

4.5.3 Feature Fusion

Feature fusion is a critical step in integrating **spatial** (**skeleton features**) and **temporal** (**frame difference dynamics**) information to create a **unified represen-tation** for violence detection. The chosen method for feature fusion is **Addition-Based Fusion** combined with **Temporal Aggregation** using **ConvLSTM**.

$$F_{combined} = \text{ReLU}\left(F_{RGB}\right) \odot \text{sigmoid}\left(F_{Change}\right),$$

where F_{RGB} and F_{Change} are the corresponding feature maps.

- **ReLU for Skeleton Features**: ReLU introduces non-linearity, avoids vanishing gradients, and accelerates convergence, enabling the model to capture complex structural patterns in skeleton data efficiently.
- **Sigmoid for Temporal Features**: Sigmoid norm-alizes inputs to [0, 1], preserving subtle temporal changes, making it ideal for emphasizing temporal dynamics during feature fusion.

➔ ReLU excels at capturing spatial features from skele-ton data, while Sigmoid highlights temporal dynamics, ensuring complementary spatial-temporal feature representation.

4.5.4 Temporal Aggregation with ConvLSTM

The fused features are processed using **Convolutional LSTMs (ConvLSTMs)**. ConvLSTM is specifically designed to handle spatio-temporal data and effectively preserves spatial patterns across time, capturing temporal dependencies and long-term relationships in video data.

4.5.5 Classifier Head

Fully connected dense layers perform final classification (binary: violent/non-violent).

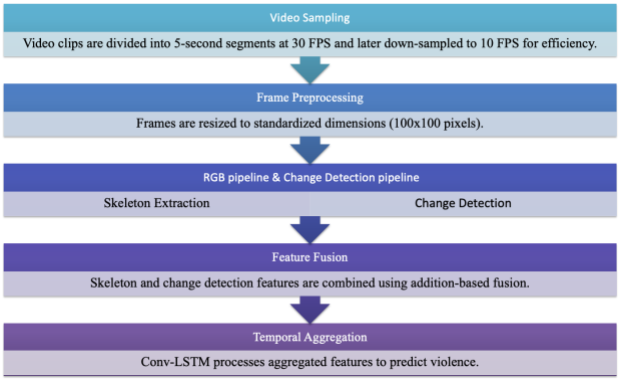

Figure 9. Implementation Workflow

4.5.6 Experiments

We tried to experimentally test the results from the paper, we tested on 2 tests, 1 is experimenting with methods to improve skeleton extraction, 2 is experimenting with the whole pipeline

- **GPU Model:** NVIDIA GeForce RTX 3090
- **Memory Utilization:** 24GB

Table 1: Results for different inputs using RGB pipeline

Model	Train Accuracy	Validation Accuracy	Validation Accuracy (Reproduce)
RGB frames	75.06 %	78.5%	78.12%
Skeletons with background	73.5%	80%	84.9 %
Skeletons without background	87.88 %	84.75 %	85.68 %

Experiments were conducted to compare the impact of di-fferent skeleton extraction enhancement techniques (table 1):

- **Gamma Correction:** Improved model accuracy by **2.5%**.
- **Histogram Equalization:** Delivered moderate improvement but increased computational load.
- **Background Simplification:** Significantly re-duced noise interference, improving model clarity and focus.

Observation: Gamma correction and background simpli-fication offered the best trade-off between computational efficiency and performance improvement.

Table 2: Comparison with state of the art in RWF-2000 from the paper

Comparison with state of the art in RWF-2000. '–' indicates that the number was not published.'*' retrained from scratch using our random seed following their public code and specifications.

Model	Year	Validation accuracy	Trainable parameters
Flow Gated Net (Cheng et al., 2021)	2020	87.30%	272 690
SPIL (Su et al., 2020)	2020	89.30%	–
SepConvLSTM (Islam et al., 2021)	2021	89.75%	333 057
Spatio-Temporal Modeling (Kang et al., 2021)	2021	92.00%	1 300 000
Spatio-Temporal Modeling (Kang et al., 2021) (retrained*)	2021	88.00%	1 300 000
Lightweight 2D+3D CNN (Wang et al., 2022)	2022	81.98%	352 000
Person detector + CNN (Choqueluque-Roman and Camara-Chavez, 2022)	2022	88.70%	–
U-Net + LSTM (Vijeikis et al., 2022)	2022	82.00%	3 457 219
Ours	2022	**90.25%**	**62 583**

Table 3: Compare between the authors' results and our reproduce results

Model	Train Accuracy	**Test Accuracy**
Author results	92.37 %	90.25 %
Our reproduce results	92.68 %	87.75 %

Experiments were conducted to compare the results obtained from reproducing the author's model and the results reported in the original study:

- **Author's Results**: The original authors reported an overall accuracy of 90.25% on the RWF-2000 validation set with significantly fewer parameters compared to other state-of-the-art models.
- **Reproduced Results:** Our reproduced model achieved an accuracy of 87.75% on the same dataset, showing minor variations likely due to differences in random seed initialization, hardware configurations, or slight variations in preprocessing steps.

Observation: Despite the slight accuracy gap, the overall performance, efficiency, and robustness of the model remain consistent with the original findings, validating the effectiveness of the proposed architecture.
This comparison demonstrates that the architecture is reproducible and maintains its advantages in terms of both computational efficiency and accuracy.

4.5.7 Runtime Performance

- **Pose Estimation Time:** Pose estimation using OpenPose takes approximately **1.562 seconds per 5-second video clip**, representing **94% of the total runtime**.
- **Prediction Time:** The violence detection model takes an average of **0.106 seconds per 5-second video clip**, representing **6% of the total runtime**.
- **Total Processing Time:** The total time to process a 5-second video is **1.668 seconds**.

Observation: The majority of the computational cost is incurred during pose estimation, indicating that optimizing this step could significantly reduce runtime overhead

4.6.8 Insight and Comments on the Paper

Innovative Dual-Pipeline Design:

- Combines human skeleton features with temporal dynamics using frame difference, addressing both spatial and temporal aspects of violence detection.
- Provides a significant balance between accuracy and computational efficiency, making it suitable for real-time applications.

Efficiency and Scalability: Processes videos at a reduced frame rate (10 FPS) without losing critical temporal information, ensuring fast runtime.

Robust Preprocessing: Gamma correction and black-background skeleton rendering improve noise resistance and focus the model on essential body movements.
Impressive Results: Achieves state-of-the-art accuracy (90.25%) on the RWF-2000 dataset, demonstrating the effectiveness of fusing pose estimation with temporal changes.
Generalization Capability: The proposed method generalizes well across different datasets, showing adaptability to varied violence scenarios.

Weakness
Dependency on Pose Estimation:

- Relies heavily on the accuracy of pose extraction. Any inaccuracies in skeleton detection (e.g., due to occlusion, crowding, or lighting) can adversely affect the performance.

False Positives/Negatives:

- False positives may arise in non-violent but dynamic movements (e.g., sports activities).
- False negatives are common in occlusion-heavy or low-contrast scenes, where pose estimation or temporal dynamics detection might fail.

Figure 10. Failure cases

Qualitative analysis reveals that false positives arise from ambiguous non-violent motions, while false negatives occur with occluded violence.

5. Conclusion

While the initial approach struggled with generalization and capturing temporal features, integrating pose estimation addressed these issues by emphasizing action-specific and sequential dynamics. Human skeletons and change detection were utilized for efficient violence detection in surveillance videos, combining pose information with video processing techniques to deliver a more effective and reliable solution.

❖ **Performance Comparison:**

- **CNN-LSTM**: Achieved 69.4 % test accuracy with 34.2M parameters. It effectively extracted spatial and temporal features but struggled with generalization and accuracy on noisy datasets.
- **ViViT (Video Vision Transformer)**: Reached 84.75% test accuracy with 86.5M parameters, excelling in modeling complex spatio-temporal dependencies. Data augmentation boosted performance but required high computational resources.
- **Human Skeleton and Change Detection**: Delivered the highest test accuracy of 90.25% (87.75% in reproduced experiments) with 62.6M parameters. It effectively leveraged spatial (skeleton) and temporal (motion) features, proving efficient and robust in real-world scenarios.

❖ **Key Insights**

- **Human Skeleton Model**: Balanced accuracy and efficiency, ideal for real-time violence detection.
- **ViViT**: Excellent at capturing complex relationships but resource intensive.
- **CNN-LSTM**: Struggled with overfitting and generalization, limiting its adaptability.

❖ **Challenges**

- **Data availability**: Acquiring diverse datasets of violent and non-violent acts in various settings (indoors, outdoors, crowd density, lighting conditions).
- Some other challenges in dataset such as:
  - Only part of the person appears in the picture
  - Crowds and chaos
  - Transient Action
  - Low resolution
  - **False Positives**: Minimizing the number of false alerts triggered by non-violent actions.
  - **Computational Resources**: Training complex models for Video and transformers requires substantial computational power.

Reference

[1] Y. Li, Y. Su, and C. Fu, “An Efficient Three-Dimensional Convolutional Neural Network for Inferring Interaction Force,” Sensors, vol. 19, no. 16, p. 3579, 2019, https://doi.org/10.3390/s19163579.

[2] 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, 2023, doi: https://doi.org/10.1016/j.cviu.2023.103739.

[3] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lucic, and C. Schmid, “ViViT: A Video Vision Transformer,” *CoRR*, vol. abs/2103.15691, 2021, [Online]. Available: https://arxiv.org/abs/2103.15691

[4] A. Dosovitskiy *et al.*, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” *CoRR*, vol. abs/2010.11929, 2020, [Online]. Available: https://arxiv.org/abs/2010.11929.

[5] J. Kitskerkin, "HuggingFace Blogs," [Online]. Available: https://huggingface.co/docs/transformers/en/model_doc/vivit.

[6] M. Cheng, “RWF-2000: A Video Database for Violence Detection,” GitHub repository, 2020. [Online].Available: https://github.com/mchengny/RWF2000-Video-Database-for-Violence-Detection.