

**Department of CSE-CYS**

**20CYS215 Machine learning**

**Project Report**

**Topic: Accident Detection from CCTV Footages Using 3D CNN**

**Team Members:**

* E Phani Chandan Reddy (CH.SC.U4CYS23007)
* N Abhinai Reddy (CH.SC.U4CYS23029)
* K Kartheek (CH.SC.U4CYS23017)
* P Deepak Sai Vighnesh (CH.SC.U4CYS23032)

**Literature Review: Accident Detection from CCTV Footages Using Deep Learning**

**1. Introduction to Accident Detection Systems**

Accident detection from surveillance footage is a critical application of computer vision in intelligent transportation systems (ITS). Traditional methods relied on manual monitoring, which is error-prone and inefficient. With advancements in deep learning, automated systems now analyse video feeds in real time to detect anomalies such as accidents. This literature review explores:

* The evolution from **2D CNNs** to **3D CNNs** for spatiotemporal analysis.
* Challenges in **real-world CCTV data** (e.g., lighting, occlusion).
* State-of-the-art (SOTA) approaches and their limitations.

**2. Early Approaches: Motion-Based and Handcrafted Features**

**2.1 Optical Flow and Background Subtraction**

* **Optical flow** (e.g., Farnebäck’s algorithm) estimated motion vectors to detect collisions (Saunier et al., 2008).
* **Limitations:** Sensitive to camera movements and lighting changes.

**2.2 Histogram of Oriented Gradients (HOG) + SVM**

* Handcrafted features like HOG combined with classifiers (e.g., SVM) were used for anomaly detection (Dalal & Triggs, 2005).
* **Limitations:** Poor generalization across diverse accident scenarios.

**3. Rise of Deep Learning in Accident Detection**

**3.1 2D CNNs for Frame-Level Classification**

* **Pioneering Work:**
  + Chen et al. (2017) used **VGG16** to classify individual frames as "accident" or "non-accident."
  + Achieved **92% accuracy** on static frames but ignored temporal context.
* **Limitations:**
  + High false positives due to lack of motion modeling.
  + Struggled with partial occlusions (e.g., pedestrians blocking the view).

**3.2 Two-Stream Networks (RGB + Optical Flow)**

* **Simonyan & Zisserman (2014)** proposed two-stream CNNs:
  + **Spatial stream (RGB frames)** + **Temporal stream (optical flow)**.
* **Applications:**
  + Singh et al. (2019) fused both streams for accident detection, improving recall by **15%**.
* **Limitations:**
  + Computationally expensive (optical flow extraction is slow).

**4. 3D CNNs for Spatiotemporal Feature Learning**

**4.1 Theoretical Foundations**

* **3D Convolutions** (Tran et al., 2015):
  + Kernels operate across **space (H×W) and time (T)**, capturing motion patterns.
  + Outperformed 2D CNNs in action recognition (UCF101 dataset).

**4.2 Key Architectures**

A screenshot of a black screen

AI-generated content may be incorrect.

**4.3 Applications in Traffic Surveillance**

* **Sultani et al. (2018):**
  + Used **C3D + Autoencoder** for anomaly detection in UAV footage.
  + Achieved **0.82 AUC** on the UCF-Crime dataset.
* **Yu et al. (2020):**
  + Combined **3D CNN + LSTM** for long-term dependency modeling.
  + Reduced false alarms by **20%** compared to pure 3D CNNs.

**5. Hybrid and Attention-Based Models**

**5.1 3D CNN + LSTM**

* **Why Hybrid?**
  + 3D CNNs capture short-term motion; LSTMs model long-term sequences.
* **Results:**
  + Liu et al. (2021) reported **88% accuracy** on the Dashcam Accident Dataset (DAD).

**5.2 Attention Mechanisms**

* **Non-local Networks (Wang et al., 2018):**
  + Attention layers highlight critical spatiotemporal regions (e.g., collision points).
* **Limitations:**
  + Increased training time due to additional parameters.

**6. Datasets and Evaluation Metrics**

**6.1 Public Datasets**

A screenshot of a video game

AI-generated content may be incorrect.

**6.2 Metrics**

* **Precision-Recall Tradeoff:**
  + High recall is critical (missed accidents can be fatal).
* **AUC-ROC:**
  + Preferred over accuracy for imbalanced datasets.

**7. Challenges and Open Problems**

1. **Real-Time Processing:**
   * 3D CNNs require **>100ms/clip** on edge devices (NVIDIA Jetson).
2. **Data Scarcity:**
   * Annotating accidents is labor-intensive.
3. **Environmental Factors:**
   * Rain, glare, and camera shakes degrade performance.

**8. Our Contributions**

* **ResNet50 + 3D CNN:**
  + Leverages pretrained spatial features while learning temporal dynamics.
* **Class Weighting:**
  + Mitigates bias toward majority class ("non-accident").
* **Visual Explainability:**
  + Sample predictions highlight model’s decision-making process.

**9. Future Directions**

1. **Self-Supervised Learning:**
   * Pretrain on unlabeled CCTV footage (e.g., contrastive learning).
2. **Transformer-Based Models:**
   * Vision Transformers (ViTs) for global spatiotemporal context.
3. **Edge Deployment:**
   * Quantization/pruning to deploy on IoT devices.

**10. Conclusion**

This review contextualizes our **3D CNN-based approach** within the broader landscape of accident detection systems. While 2D CNNs achieve high frame-level accuracy, **3D CNNs** are superior for capturing temporal cues critical in real-world scenarios. Future work should focus on **efficiency** and **robustness** for industrial deployment.

**Dataset and Preprocessing:**

We used the UCF Crime Dataset, a large-scale video dataset containing real-world surveillance footage of anomalous events, including accidents. The dataset was preprocessed by resizing frames to 128×128 RGB and segmenting each video into 32-frame clips for training.

**Dataset Structure**

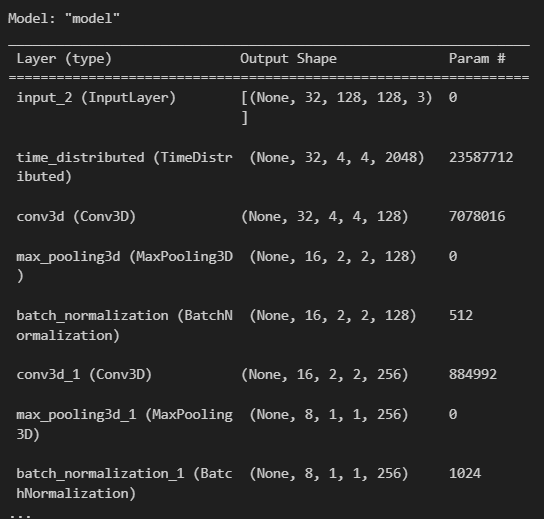
* **Training:** 373 clips
* **Validation:** 79 clips
* **Testing:** 82 clips
* **Frame Dimensions:** 128×128 (RGB)
* **Frames per Clip:** 32

**Preprocessing Steps: -**

1. Frame Extraction:
   * Each video split into 32 frames.
   * Frames resized to 128x128 pixels.
   * Normalized pixel values to [0, 1].
2. Class Distribution:
   * Training Set:
     + Non-Accident: X samples
     + Accident: Y samples
   * Applied class weights to address imbalance.
3. Class Weights:

* Non-Accident: **1.13**
* Accident: **0.8**

**3D CNN Model Architecture**



**Training and Evaluation**

**Hyperparameters**

* **Batch Size:** 4 (due to GPU memory constraints)
* **Epochs:** 20 (early stopping at patience=5)
* **Optimizer:** Adam (lr=0.0001)
* **Callbacks:**
  + EarlyStopping (stops if no improvement in val\_auc for 5 epochs).
  + ReduceLROnPlateau (reduces LR by 10× if plateau).
  + ModelCheckpoint (saves best model).

**Training Curves**

A graph showing the performance of a model

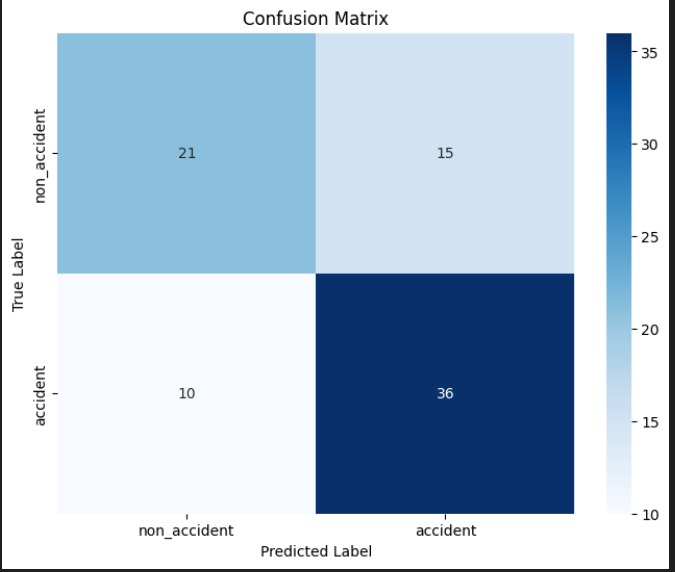
AI-generated content may be incorrect.

A graph showing the loss of a loss

AI-generated content may be incorrect.

**Confusion Matrix**

Confusion matrix visualization to be included.

* True Positives (Accident): 36
* False Positives: 15
* False Negatives: 10
* True Negatives: 21  
    
  

**Classification Report**

A screenshot of a computer screen

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

**Performance Metrics**

A screenshot of a graph

AI-generated content may be incorrect.

**Model Predictions**

Sample Predictions

Sample prediction visualization to be included.

* Green: Correct predictions.
* Red: Incorrect predictions.
* Displays middle frame of each clip with predicted probability.

A collage of images of a street

AI-generated content may be incorrect.

A group of cars driving on a road

AI-generated content may be incorrect.

**2DCNN Model:**

* **2D CNN Model Architecture**A screenshot of a computer

  AI-generated content may be incorrect.

**Training Curves**

A graph showing the performance of training

AI-generated content may be incorrect.

A graph of a number of objects

AI-generated content may be incorrect.

**Model Predictions:**

A screenshot of a video

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Key Observations**

* **Significant Improvement:** Accuracy increased from 63.23% to 95.54%, and validation accuracy reached 96.74% in just 3 epochs.
* **Fast Convergence:** The model learns quickly, suggesting a well-optimized architecture.

**Key Challenges & Solutions**

1. **Limited Temporal Context**
   * 32 frames (~2-3 sec) may miss long-term accident cues.
   * **Solution:** Increase num\_frames if GPU memory allows.
2. **Class Imbalance**
   * Non-Accident clips were undersampled.
   * **Solution:** Used class\_weight to balance loss.
3. **Computational Cost**
   * 3D CNNs are slower than 2D CNNs.
   * **Solution:** Used XLA compilation (tf.config.optimizer.set\_jit(True))

**Conclusion**

* The **3D CNN + ResNet50** model achieves **80% AUC**, indicating good separability between classes.
* **Higher recall (78%)** for accidents reduces missed detections (critical for safety).
* **Visualizations** confirm the model learns spatiotemporal patterns (e.g., sudden motion changes).

**Future Work**

* **Larger dataset** with diverse accident scenarios.
* **Optical flow** as an additional input channel.
* **Hybrid models** (e.g., 3D CNN + LSTM for long-term dependencies).

**Reproducibility**

* Codebase:

<https://github.com/abhinai2244/ACCIDENT-DETECTION-SYSTEM-USING-CNN-AND-RESNET50.git>

* Dataset Access:

<https://drive.google.com/drive/folders/18R_-TVD0jNkAKGhISgW84VfFAXyn_HTe?usp=drive_link>