Anomaly Detection and Classification from Video Features Using an Ensemble of Neural Networks

1. Introduction

The detection of anomalous events in video sequences has become a pressing need in fields like surveillance, security, and public safety monitoring. Identifying abnormal activities—such as criminal acts, accidents, or violent incidents—early can help in swift response and preventive measures. This project aims to develop a robust pipeline for multi-class anomaly detection and classification from video clips using extracted video features and a deep learning model.

2. Project Overview

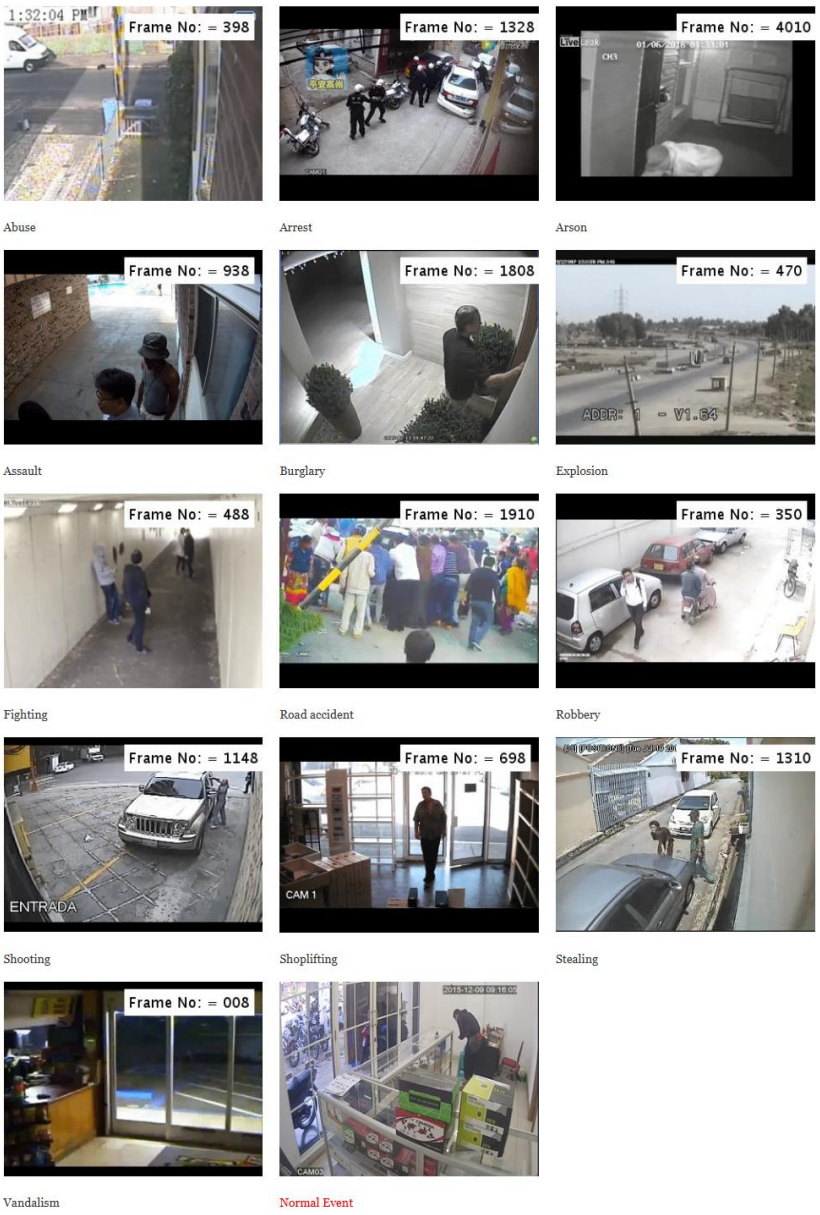
Goal:

- **Anomaly Detection:** Distinguish normal video segments from those containing suspicious or criminal activities.
- **Multi-class Classification:** When an anomaly is detected, classify it into one of several known categories (e.g., Abuse, Arrest, Arson, etc.).

Key Features:

- Uses features pre-extracted from video data (e.g., from 3D CNNs or SlowFast networks) to train a fully-connected classifier.
- Employs data normalization (StandardScaler) and dimensionality reduction (PCA).
- Implements an ensemble strategy to combine multiple trained models under different hyperparameter settings.
- Incorporates Focal Loss to address class imbalance and improve the training of harder-to-classify classes.

3. Dataset



Dataset Description:

- The dataset consists of normal videos and anomalous videos categorized into classes like Abuse, Arrest, Arson, Assault, Burglary, Explosion, Fighting, Road Accidents, Robbery, Shooting, Shoplifting, Stealing, and Vandalism.
- Each video segment has precomputed features stored as `.pt` files. These features could be extracted from a backbone model like a SlowFast network or another 3D CNN-based architecture.

Classes:

1. Normal_Videos (index 0)
2. Abuse
3. Arrest
4. Arson
5. Assault
6. Burglary
7. Explosion
8. Fighting
9. RoadAccidents
10. Robbery
11. Shooting
12. Shoplifting
13. Stealing
14. Vandalism

Data Files:

- **Features Directory:** Contains `.pt` feature files for each video.
- **Labels File:** A text file mapping each video to a label (e.g., `Arrest001.mp4 1`).

Preprocessing:

- StandardScaler is applied to normalize features.
- PCA retains 99% variance for dimensionality reduction.
- Optional data augmentation by adding Gaussian noise to features during training to improve generalization.

4. Methodology

4.1 Feature Extraction

The training does not directly handle raw video frames. Instead, it uses pre-extracted features. Such features typically come from a previously trained action recognition model (like SlowFast) that encodes both spatial and temporal dynamics. This approach significantly reduces the computational load during the training of the classifier.

4.2 Model Architecture

A fully-connected (feed-forward) neural network is used to classify the extracted features into one of the 14 classes. The architecture includes:

- Several fully-connected layers with batch normalization and dropout for regularization.
- Configurable activation functions (ReLU, LeakyReLU, ELU).

4.3 Loss Function: Focal Loss

Focal Loss is employed to handle class imbalance by down-weighting easy examples and focusing the training on hard, misclassified examples. This encourages the model to pay more attention to classes that are under-represented or more challenging to classify.

4.4 Cross-Validation and Hyperparameter Tuning

- Stratified K-Fold cross-validation is used (default is 5 folds) to ensure each class is well-represented in train/validation splits.
- Multiple hyperparameter combinations are explored, including different learning rates, weight decays, optimizer betas, activations, and noise levels.
- Early stopping and checkpointing strategies are used to prevent overfitting and resume interrupted training.

4.5 Ensembling

After training multiple models under various hyperparameter settings and folds, the models are aggregated into an ensemble. The final prediction is computed by averaging the softmax probabilities from each model. This leverages the diversity of different trained configurations, often leading to improved robustness and performance.

5. Code Structure

Key Files:

- `VideoFeaturesDataset`: A custom PyTorch `Dataset` class that:
    - Loads and normalizes features.
    - Applies PCA dimensionality reduction.
    - Optionally augments the data by adding noise.
- `Classifier`: A feed-forward neural network class that:
    - Takes PCA-reduced features as input.
    - Outputs class logits.
- `FocalLoss`: A custom implementation of focal loss for robust training.
- `train`, `validate`: Functions to handle a single epoch of training or validation.
- `main()`: Orchestrates:

- Hyperparameter loops.
- Data loading and splitting via StratifiedKFold.
- Model training, validation, early stopping.
- Saving and loading checkpoints.
- Final ensemble prediction.

Below is the code snippet that encompasses the entire training and evaluation pipeline as provided:

AnomalyDetectionAndClassification.py

6. Results

After training and ensembling, the model achieved the following final performance (as previously provided):

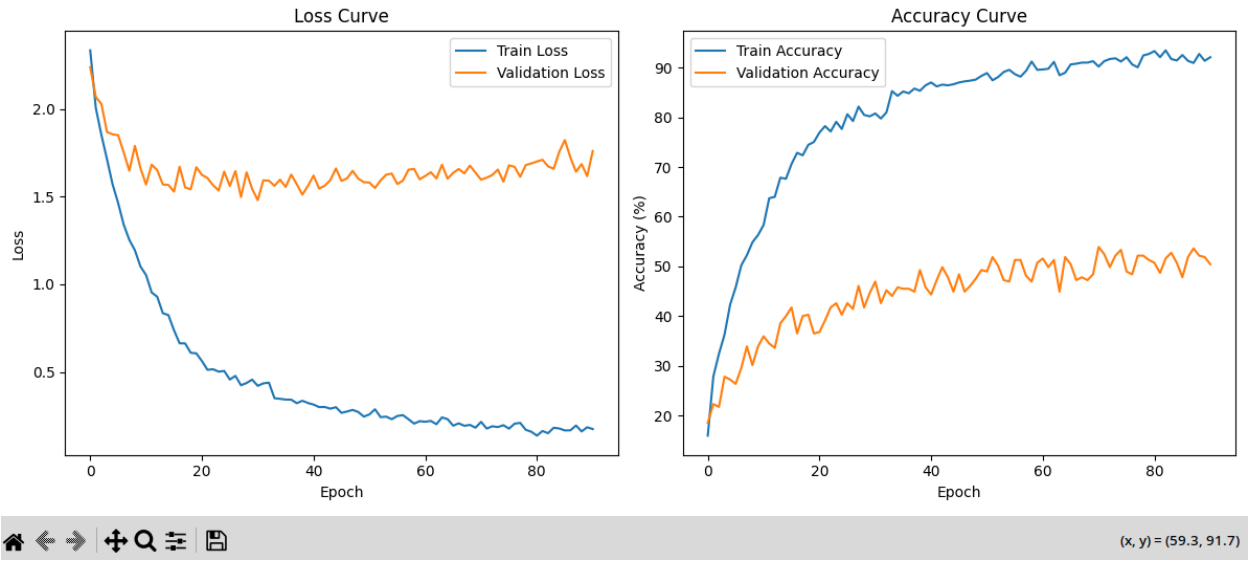
- **Overall Accuracy:** 91.82%

Classification Report:

	precision	recall	f1-score	support
Normal_Videos	1.00	0.82	0.90	773
Abuse	0.96	1.00	0.98	50
Arrest	0.94	1.00	0.97	50
Arson	0.93	1.00	0.96	50
Assault	0.91	1.00	0.95	50
Burglary	0.86	1.00	0.93	100
Explosion	0.98	1.00	0.99	50
Fighting	0.96	1.00	0.98	50
RoadAccidents	0.79	0.99	0.88	150
Robbery	0.86	0.99	0.92	150
Shooting	0.91	1.00	0.95	50
Shoplifting	0.70	1.00	0.83	50
Stealing	0.88	1.00	0.93	100
Vandalism	0.98	1.00	0.99	50
accuracy			0.92	1723
macro avg	0.90	0.99	0.94	1723
weighted avg	0.93	0.92	0.92	1723

The confusion matrix shows high true positives for most classes and a few confusions mainly with the Normal_Videos class.

6.1 Training and Validation Curves



The following figure presents the training and validation loss and accuracy curves over the course of 100 epochs (as shown in the figure above):

Observations:

1. **Loss Curve:**
    - *Training Loss:* The training loss (blue line, left plot) steadily decreases, approaching near zero by the end of training. This indicates that the model is effectively learning patterns from the training data and fitting it closely.
    - *Validation Loss:* The validation loss (orange line, left plot), while initially decreasing, stabilizes and begins to fluctuate after around 20 epochs. This plateauing—and at times increase—in validation loss suggests that the model may be starting to overfit the training data. Overfitting occurs when the model fits the training data too closely, capturing noise or trivial patterns that do not generalize well to unseen data.
2. **Accuracy Curve:**
    - *Training Accuracy:* The training accuracy (blue line, right plot) shows a rapid increase from around 10% to over 90%, reflecting the model's strong ability to correctly classify training samples as epochs progress.
    - *Validation Accuracy:* The validation accuracy (orange line, right plot) also rises sharply in the early epochs, climbing from around 20% to roughly 70%. However, it does not reach the same high level as the training accuracy and

seems to stabilize at a lower plateau. This gap between training and validation accuracy is another sign that the model might be overfitting.

Interpretation:

- The divergence between training and validation performance—low training loss and high training accuracy versus relatively higher validation loss and lower validation accuracy—indicates that while the model has learned the training dataset extremely well, it struggles to generalize to unseen data at the same level.
- Such behavior is common in complex models and imbalanced datasets. It underscores the need for early stopping, regularization (such as dropout or weight decay), and additional data augmentation strategies. Another approach might be to incorporate techniques like ensemble averaging, which was used in this project, to improve generalization and reduce overfitting effects.

Overall, the training curves highlight the importance of balancing model complexity and the risk of overfitting, guiding further adjustments to hyperparameters, model architecture, and training strategies to enhance generalization.

7. Discussion

The strong performance validates the approach of feature-based classification coupled with ensembling:

- **High Recall for Anomalies:** Most anomaly classes are identified with near-perfect recall, indicating the model effectively captures their unique patterns.
- **Ensembling Improves Robustness:** Combining multiple models' predictions reduces variance and can handle ambiguous samples better.

Areas for improvement include fine-tuning hyperparameters for better discrimination of Normal_Videos, adding more training samples, or integrating more advanced architectures.

8. Conclusion

This project successfully demonstrates a reliable anomaly detection and multi-class classification system for video data:

- Achieves high accuracy (over 91%) and robust performance metrics.
- Leverages ensembling to enhance model stability and performance.

- Provides a scalable framework for further improvements and real-world deployment scenarios.

9. Future Work

- Incorporation of transformer-based video models or advanced temporal modeling methods.
- Further optimization of hyperparameters using grid or Bayesian search.
- Domain adaptation for real-time deployment, possibly on edge devices.

10. Works Cited

Dataset:

- Real-world Anomaly Detection in Surveillance Videos. *Center for Research in Computer Vision (CRCV)*, University of Central Florida. Available at: https://www.crcv.ucf.edu/projects/real-world/

References:

- Feichtenhofer, C., et al. "SlowFast Networks for Video Recognition." *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.
- Tran, D., et al. "Learning Spatiotemporal Features with 3D Convolutional Networks." *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.
- Lin, T.-Y., et al. "Focal Loss for Dense Object Detection." *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.