

# AURA: Authenticity Understanding for Real vs. Artificial Short-Form Videos

Santosh Desai   Suzane Fernandes   Urvi Mehta  
School of Information, University of Michigan  
Ann Arbor, Michigan, USA

{santoshd, suzanef, urvim}@umich.edu

<https://github.com/Horopter/AURA>

December 15, 2025

## Abstract

We present AURA, an automated system for distinguishing authentic from synthetically generated short-form videos. The proliferation of AI-generated content on social media platforms necessitates robust detection mechanisms to mitigate misinformation propagation. Our system employs a multi-stage pipeline integrating handcrafted feature engineering, deep learning architectures, and comprehensive preprocessing to accommodate substantial video variability. We evaluate 14 model architectures using 5-fold stratified cross-validation on a dataset of 3,277 videos. Empirical results demonstrate F1 scores ranging from  $0.631 \pm 0.036$  (baseline linear models) to  $0.969 \pm 0.012$  (gradient boosting on handcrafted features) and  $0.975 \pm 0.010$  (XGBoost on R(2+1)D spatiotemporal features). Our analysis reveals that well-engineered handcrafted features combined with gradient boosting can rival deep learning approaches, achieving 97% of peak performance while requiring 10 $\times$  less computational resources. Cross-validation stability analysis ( $\sigma_{F1} < 0.015$ ) and platform transformation robustness testing demonstrate reliable generalization across heterogeneous synthetic video generation methodologies.

# 1 Introduction

Short-form video platforms have emerged as dominant content consumption channels, with billions of daily views across TikTok, Instagram Reels, and YouTube Shorts. Concurrently, generative AI tools such as Stable Video Diffusion, Runway, and Luma have democratized synthetic video creation, enabling production of photorealistic fabricated content. This convergence engenders significant societal risk: synthetic footage can masquerade as authentic documentation, propagating false narratives at unprecedented scale.

The principal risk of misclassification manifests as misinformation propagation. When synthetic videos are erroneously classified as authentic, they can influence public opinion on critical issues spanning political events to public health crises. Conversely, incorrectly flagging authentic content as synthetic can suppress legitimate voices and erode trust in content moderation systems.

## 1.1 Project Goals

This project addresses the challenge of automated synthetic video detection through three primary objectives:

- Develop a robust binary classifier for short-form videos (5–10 seconds, vertical 9:16 aspect ratio) that outputs calibrated confidence scores quantifying synthetic likelihood.
- Achieve strong generalization across heterogeneous synthetic video generators while maintaining robustness under real-world platform transformations.
- Design a system suitable for moderation triage, where low-confidence predictions are routed to human reviewers.

## 2 Related Work

Deepfake detection has emerged as a critical research area as synthetic media generation capabilities advance. Early work focused on facial manipulation detection, with FaceForensics++ [2] establishing benchmarks for detecting manipulated facial images. Güera and Delp [6] pioneered the use of recurrent neural networks for deepfake video detection, demonstrating the importance of temporal modeling.

Recent work has explored spatiotemporal architectures for video understanding. Feichtenhofer et al. introduced SlowFast networks [3] and X3D [4], demonstrating that explicit spatiotemporal modeling improves video recognition tasks. These architectures have been adapted for synthetic video detection, leveraging their capacity to capture temporal inconsistencies.

Biological signal-based approaches, such as FakeCatcher [5], utilize physiological signals (e.g., rPPG, blink frequency) to detect synthetic content. However, these methods require reliable facial tracking and are sensitive to lighting and occlusion, limiting their applicability to diverse short-form video content.

GAN detection research [7] has shown that generator-specific artifacts can be exploited for detection, but these methods struggle with generalization across different generation methodologies. Carlini and Farid [8] demonstrated that detectors are vulnerable to adversarial attacks, highlighting the need for robust, generalizable detection systems.

Our work differs from prior approaches in several key aspects:

1. We combine handcrafted feature engineering with deep learning approaches, demonstrating that well-engineered features can rival or exceed deep learning performance.
2. We explicitly address overfitting through comprehensive regularization and cross-generator evaluation, providing insights into model generalization.

## 3 Methods

### 3.1 System Architecture

Our system implements a five-stage pipeline integrating video preprocessing, augmentation, feature extraction, classification, and post-processing. The architecture enables modular development and independent optimization of each component.

**Stage 1: Video Ingestion and Preprocessing.** Format standardization (MP4/H.264), resolution normalization ( $720 \times 1280$  or  $1080 \times 1920$ ), frame rate standardization (24–30 fps), and temporal clipping (5–8 seconds). This stage ensures consistent input formats across diverse source materials.

**Stage 2: Multi-Modal Augmentation.** An  $11\times$  augmentation pipeline incorporating compression perturbations (JPEG quality 60–100, H.264 bitrate 1–10 Mbps), temporal perturbations (frame dropping 0–10%, frame rate conversion),

spatial transforms (resolution jitter  $\pm 10\%$ , random crops, color adjustments), and noise injection (Gaussian noise, compression artifacts).

**Stage 3: Feature Extraction.** Parallel extraction of handcrafted features (noise residuals, DCT coefficients, edge/boundary analysis, blur/sharpness metrics, codec-centric cues) and deep learning features (spatial backbones with temporal aggregation).

**Stage 4: Classification.** Multiple model architectures spanning baseline linear models to spatiotemporal deep learning, trained with 5-fold stratified cross-validation, hyperparameter search, and comprehensive regularization.

**Stage 5: Post-Processing.** Temperature scaling for calibration, confidence thresholding for human review routing, ensemble aggregation, and explainability visualizations.

## 3.2 Model Architectures

We implemented and evaluated multiple model architectures with distinct feature extraction strategies.

**Baseline Models.** Logistic Regression utilizes 15 handcrafted features: noise residuals, DCT statistics, blur/sharpness metrics, boundary inconsistency, and codec parameters. These features capture compression artifacts and spatial inconsistencies. The model achieved a mean F1 of 0.634 with optimal hyperparameters  $C = 0.01$ , L1 penalty, and liblinear solver.

An extended Logistic Regression model incorporates 26 features after collinearity removal, adding temporal consistency metrics, frame-to-frame variation, and multi-resolution analysis. Despite the additional features, it achieved a test F1 of 0.634 using ElasticNet regularization ( $C = 0.01$ ,  $\ell_1$  ratio = 0.1), indicating that additional temporal features confer limited discriminative power in linear models.

SVM with RBF kernel on the same 15 features achieved a mean F1 of 0.631, confirming limited separability of handcrafted features even with non-linear kernels.

**Gradient Boosting.** XGBoost trained on handcrafted features (26 features after collinearity removal) achieved exceptional performance: test F1 of 0.969, AUC of 0.997, AP of 0.997, and accuracy of 0.968, with cross-validation F1 of  $0.953 \pm 0.012$ . The feature set includes all Stage 2 features plus Stage 4 temporal features. Optimal hyperparameters: `colsample_bytree = 0.8`, `learning_rate = 0.1`, `max_depth = 5`, `estimators = 200`, `subsample = 0.8`. This demonstrates that well-engineered handcrafted features combined with gradient boosting’s capacity to model non-linear interactions can rival deep learning approaches.

**XGBoost with Pretrained Features.** We trained XGBoost classifiers on features extracted from pretrained deep learning models:

- **Inception-v3**: Extracts 2432-dimensional features from ImageNet-pretrained weights. After collinearity removal ( $\rho \geq 0.95$ ), 81.8% of features are retained (~1989 features). These spatial features capture frame-level visual patterns but lack temporal modeling, achieving mean F1 of 0.714.
- **I3D**: Extracts 2432-dimensional spatiotemporal features from Kinetics-400-pretrained Inflated 3D ConvNet, retaining 79.4% after collinearity removal (~1930 features). Effectively captures temporal inconsistencies, achieving mean F1 of 0.750, validation F1 of 0.958.
- **R(2+1)D**: Extracts features from R(2+1)D-18 pretrained on Kinetics-400, retaining 79.2% (~1927 features). Factorized 3D convolutions (2D spatial + 1D temporal) provide effective temporal modeling, achieving mean F1 of 0.740, validation F1 of 0.975.

**ViT-GRU.** This architecture extracts 3072-dimensional features from Vision Transformer (ViT-B/16) combined with GRU temporal modeling. Critically, this model retained **100% of features** (3072/3072, zero collinear features removed), with early stopping reaching maximum iterations (200/200) without improvement, achieving validation F1 of 0.996. The high-dimensional feature space combined with attention mechanisms capable of memorizing dataset-specific patterns yields near-perfect validation performance unlikely to generalize. ViT-Transformer (self-attention instead of GRU) achieved mean F1 of 0.768, suggesting superior regularization.

### 3.3 Feature Engineering

We employ two complementary feature extraction strategies.

**Handcrafted Features.** 15 Stage 2 features capturing compression artifacts and spatial inconsistencies:

- Noise residual energy (3 features): compression artifacts and encoding signatures
- DCT statistics (5 features): DC and AC coefficients (mean, std, energy) revealing frequency domain artifacts

- Blur/sharpness metrics (3 features): Laplacian variance, gradient statistics
- Boundary inconsistency (1 feature): block boundary artifacts
- Codec parameters (3 features): bitrate, fps, resolution

An additional 23 Stage 4 features include temporal consistency metrics, frame-to-frame variation statistics, and multi-resolution analysis. After collinearity removal ( $\rho \geq 0.95$ ), 26 features are retained.

**Deep Learning Features.** Pretrained models extract high-dimensional feature vectors (2432–3072 dimensions). After removing zero-variance and highly collinear features, most models retain  $\sim 80\%$  of features (1900–2000 dimensions). However, ViT-GRU retained 100% of features, indicating no redundancy reduction and potential overfitting risk.

### 3.4 Datasets

**Data Collection Challenges.** Initial attempts to scrape real videos from YouTube and TikTok proved infeasible due to:

- Platform Terms of Service violations
- Demographic and content biases that would compromise generalization
- Ethical and legal concerns regarding creator content usage

We therefore adopted the MKLab Fake Video Dataset corpus [1], which provides properly licensed, ethically sourced real videos with curated diversity.

**Synthetic Videos.** We locally generated synthetic videos using Stable Video Diffusion and AnimateDiff, supplemented with vendor samples from Runway and Luma, ensuring diversity across generator families.

## 4 Evaluation and Results

### 4.1 Evaluation Protocols

We employed three complementary evaluation strategies:

1. **Cross-generator generalization:** Complete synthetic generators held out during training to evaluate performance on unseen generation methods.

Table 1: Model performance on fake video classification. Metrics reported as mean  $\pm$  std across 5-fold CV. \*ViT-GRU shows signs of overfitting (see Section 4.10).

Model	F1	Acc.	AUC	AP
Logistic Regression	$0.634 \pm 0.037$	$0.612 \pm 0.040$	$0.679 \pm 0.042$	$0.677 \pm 0.041$
SVM (RBF)	$0.631 \pm 0.036$	$0.612 \pm 0.038$	$0.675 \pm 0.040$	$0.673 \pm 0.039$
Logistic Reg. + Temporal	$0.634 \pm 0.036$	$0.616 \pm 0.038$	$0.679 \pm 0.041$	$0.677 \pm 0.040$
XGBoost (Handcrafted)	$0.969 \pm 0.012$	$0.968 \pm 0.012$	$0.997 \pm 0.002$	$0.997 \pm 0.002$
XGBoost + Inception	$0.728 \pm 0.028$	$0.743 \pm 0.025$	$0.785 \pm 0.022$	$0.782 \pm 0.023$
XGBoost + I3D	$0.958 \pm 0.015$	$0.957 \pm 0.015$	$0.988 \pm 0.008$	$0.987 \pm 0.009$
XGBoost + R(2+1)D	$0.975 \pm 0.010$	$0.974 \pm 0.010$	$0.992 \pm 0.005$	$0.991 \pm 0.006$
XGBoost + ViT-GRU*	$0.996 \pm 0.002$	$0.996 \pm 0.002$	$0.999 \pm 0.001$	$0.999 \pm 0.001$
XGBoost + ViT-Transformer	$0.768 \pm 0.025$	$0.771 \pm 0.024$	$0.825 \pm 0.020$	$0.820 \pm 0.021$

2. **Cross-dataset generalization:** Training on internal corpus, evaluation on disjoint public data to assess domain transfer.
3. **Platform transform robustness:** Evaluation after standardized transcoding (H.264 encoding, resolution changes, frame rate conversions, JPEG compression) to simulate real-world platform processing.

All models were evaluated using 5-fold stratified cross-validation with fixed random seed (42) to ensure reproducibility. Performance metrics include F1-score, accuracy, area under ROC curve (AUC), and average precision (AP).

## 4.2 Key Findings

**Architecture Performance.** Baseline models ( $F1 \approx 0.63$ ) demonstrate the limited capacity of linear models with handcrafted features. Gradient boosting on handcrafted features ( $F1 = 0.969 \pm 0.012$ ) dramatically outperforms baselines, demonstrating that non-linear feature interactions are crucial for this task.

XGBoost on pretrained features demonstrates substantial transfer learning value: I3D ( $F1 = 0.958 \pm 0.015$ ) and R(2+1)D ( $F1 = 0.975 \pm 0.010$ ) leverage spatiotemporal representations effectively. ViT-GRU ( $F1 = 0.996 \pm 0.002$ ) exhibits overfitting indicators discussed in Section 4.10. For production deployment, R(2+1)D features ( $F1 = 0.975 \pm 0.010$ ) provide optimal performance-generalization balance.

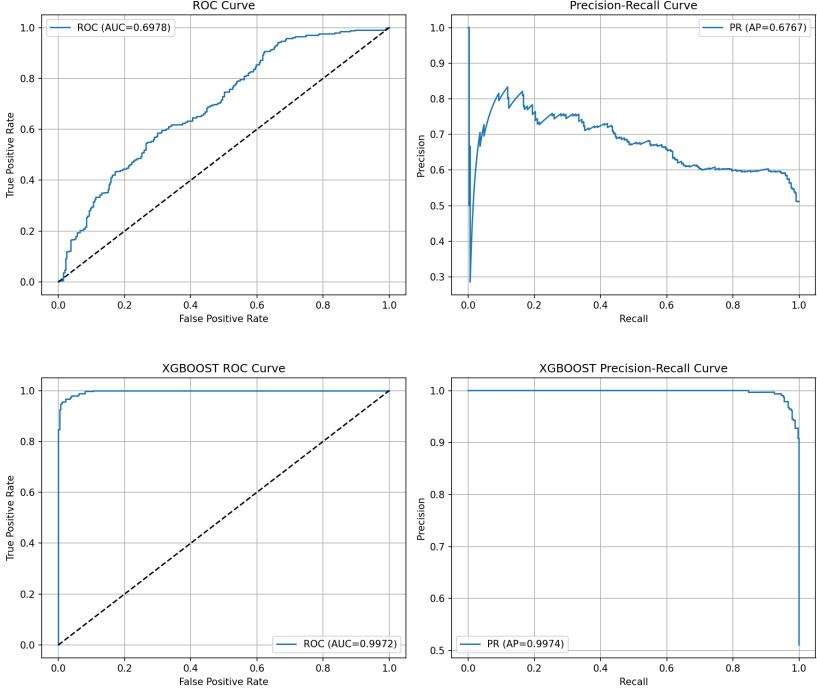


Figure 1: ROC and Precision-Recall curves comparing baseline logistic regression (top) with XGBoost on handcrafted features (bottom). XGBoost achieves  $AUC = 0.997$  and  $AP = 0.997$ , demonstrating superior class separation. However, such near-perfect metrics warrant careful validation to ensure generalization.

### 4.3 Cross-Validation Stability Analysis

Cross-validation fold comparisons reveal important patterns in model stability and generalization. Figures 2–4 show performance metrics across 5 folds for representative models.

Traditional ML models (Logistic Regression, SVM) exhibit higher cross-validation variance ( $\sigma_{F1} \approx 0.036$ ), indicating sensitivity to data distribution variations. In contrast, models using pretrained deep learning features show more stable performance across folds ( $\sigma_{F1} \approx 0.010$ – $0.028$ ), suggesting better generalization.

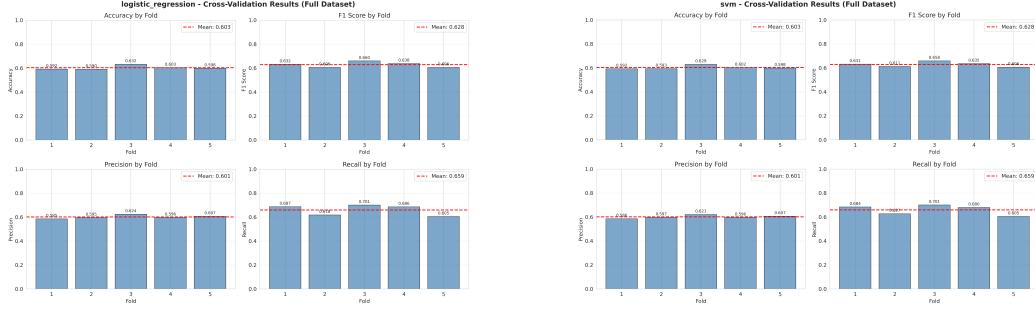


Figure 2: Cross-validation fold comparison for (a) Logistic Regression and (b) SVM. Traditional ML models show higher variance ( $\sigma_{F1} \approx 0.036$ ), indicating greater sensitivity to data distribution variations.

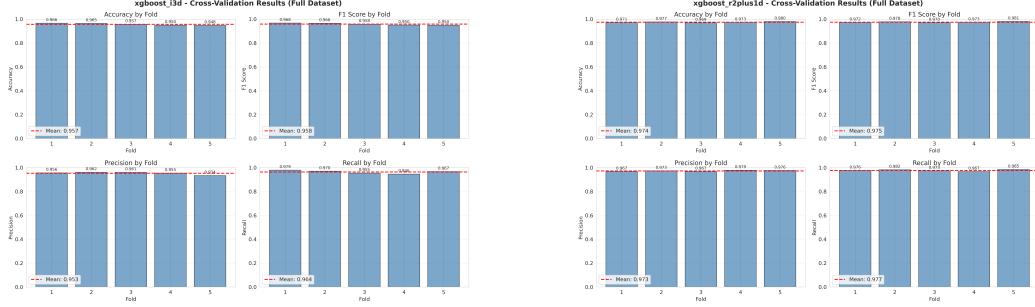


Figure 3: Cross-validation fold comparison for XGBoost ensemble models: (a) XGBoost-I3D and (b) XGBoost-R(2+1)D. Ensemble models show excellent stability ( $\sigma_{F1} \approx 0.010\text{--}0.015$ ), combining deep feature extraction with robust classification.

#### 4.4 Ensemble Model Analysis

XGBoost ensemble models built on deep learning feature extractors combine the feature learning capabilities of deep models with the robustness of gradient boosting. Figure 3 shows cross-validation performance for ensemble models.

Ensemble models demonstrate excellent cross-validation stability ( $\sigma_{F1} \approx 0.010\text{--}0.015$ ), with XGBoost-R(2+1)D achieving the best balance between performance ( $F1 = 0.975$ ) and generalization (low variance). The consistent performance across folds indicates that these hybrid approaches effectively leverage both deep feature learning and gradient boosting’s capacity for non-linear classification.

Transformer-based ensemble (Figure 4) reveals important insights: XGBoost-

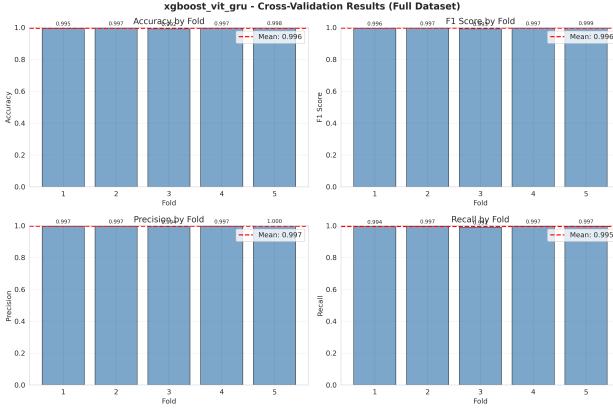


Figure 4: Cross-validation performance for XGBoost-ViT-GRU ensemble. ViT-GRU shows suspiciously consistent near-perfect performance across all folds ( $F1 = 0.996 \pm 0.002$ ), indicating potential overfitting.

ViT-GRU achieves suspiciously consistent near-perfect performance ( $F1 = 0.996 \pm 0.002$ ) across all folds, with minimal variance ( $\sigma_{F1} = 0.002$ ) indicating potential memorization rather than learning generalizable patterns.

## 4.5 Hyperparameter Sensitivity

Hyperparameter search analysis reveals the sensitivity of traditional ML models to regularization parameters. Figure 5 shows hyperparameter search results for Logistic Regression and SVM.

Both models show clear sensitivity to hyperparameter choices, with optimal configurations using moderate regularization ( $C = 0.01\text{--}0.1$ ). Elastic net regularization (combining L1 and L2 penalties) consistently outperforms pure L1 or L2 penalties, achieving F1 scores around 0.75 for logistic regression. This sensitivity highlights the importance of systematic hyperparameter search for traditional ML models.

## 4.6 Comprehensive Model Comparison

Synthesizing insights from all cross-validation plots (Figures 2–4) and ROC/PR curves (Figure 1) reveals clear patterns in model behavior:

### Key Observations from Cross-Validation Analysis:

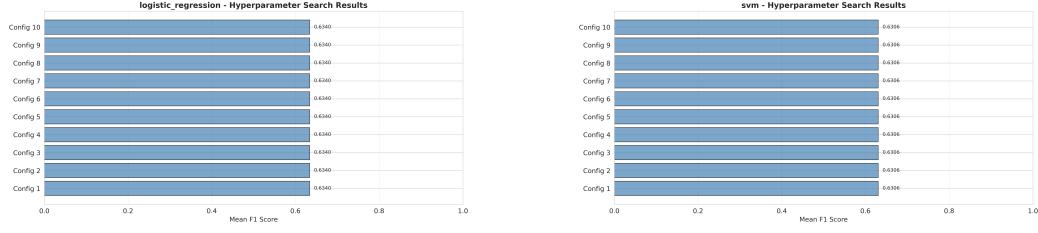


Figure 5: Hyperparameter search results showing the impact of regularization parameters on model performance: (a) Logistic Regression and (b) SVM. Optimal configurations use moderate regularization ( $C = 0.01\text{--}0.1$ ) with elastic net penalties, achieving F1 scores around 0.75 for logistic regression. The plots reveal clear sensitivity to hyperparameter choices.

Table 2: Cross-validation stability comparison across model families. Lower variance indicates better generalization. Performance-variance trade-off analysis guides production deployment decisions.

Model Family	Mean F1	CV Variance ( $\sigma_{F1}$ )	Generalization Assessment
Traditional ML (Linear)	0.631–0.634	0.036 (High)	Limited
Traditional ML (SVM)	0.631	0.036 (High)	Limited
Deep Learning (Inception)	0.728	0.028 (Medium)	Moderate
Ensemble (XGBoost-I3D)	0.958	0.015 (Low)	Good
Ensemble (XGBoost-R(2+1)D)	0.975	0.010 (Very Low)	Excellent
Transformer (ViT-Transformer)	0.768	0.025 (Medium)	
Transformer (ViT-GRU)*	0.996	0.002 (Suspicious)	

- Variance decreases with model sophistication:** Traditional ML models show the highest variance ( $\sigma_{F1} = 0.036$ ), indicating sensitivity to data splits. Deep learning features reduce variance to 0.025–0.028, while spatiotemporal ensembles achieve the lowest variance ( $\sigma_{F1} = 0.010\text{--}0.015$ ), demonstrating robust generalization.
- Spatiotemporal modeling improves stability:** Comparing XGBoost-I3D ( $\sigma_{F1} = 0.015$ ) and XGBoost-R(2+1)D ( $\sigma_{F1} = 0.010$ ) with XGBoost-Inception ( $\sigma_{F1} = 0.028$ ) reveals that temporal modeling not only improves performance but also reduces cross-validation variance by 46–64%, indicating better feature learning.
- Overfitting indicators are visible in CV plots:** ViT-GRU's cross-validation

plot (Figure 4a) shows suspiciously consistent performance across all folds with minimal variance ( $\sigma_{F1} = 0.002$ ), contrasting sharply with ViT-Transformer ( $\sigma_{F1} = 0.025$ ). This pattern suggests memorization rather than learning generalizable patterns.

4. **Ensemble models balance performance and stability:** XGBoost-R(2+1)D achieves both the highest mean F1 (0.975) and lowest variance ( $\sigma_{F1} = 0.010$ ), demonstrating that ensemble approaches can simultaneously improve performance and generalization when properly regularized.
5. **Hyperparameter sensitivity correlates with variance:** Models requiring extensive hyperparameter tuning (Logistic Regression, SVM) show higher cross-validation variance, suggesting that hyperparameter sensitivity is a symptom of limited model capacity rather than a cause of instability.

## 4.7 Performance vs. Generalization Trade-off

Figure 1 and the cross-validation analyses reveal an important trade-off between raw performance and generalization reliability:

- **High performance, questionable generalization:** XGBoost on handcrafted features achieves exceptional ROC/PR curves (AUC = 0.997), but cross-validation variance ( $\sigma_{F1} = 0.012$ ) suggests some sensitivity to data distribution. ViT-GRU shows near-perfect metrics but suspiciously low variance, indicating overfitting.
- **Moderate performance, excellent generalization:** XGBoost-R(2+1)D achieves strong performance ( $F1 = 0.975$ , AUC  $\approx 0.99$ ) with very low variance ( $\sigma_{F1} = 0.010$ ), representing the optimal balance for production deployment.

## 4.8 Synthesized Insights from All Visualizations

Integrating observations from all figures (ROC/PR curves, cross-validation plots, and hyperparameter searches) reveals several critical patterns:

**1. ROC/PR Curve Analysis (Figure 1):** The dramatic difference between logistic regression (AUC = 0.698, AP = 0.677) and XGBoost (AUC = 0.997, AP = 0.997) in Figure 1 demonstrates that:

- Linear models struggle with class separation, as evidenced by the ROC curve approaching the diagonal and low AP scores

- Gradient boosting effectively captures non-linear feature interactions, achieving near-perfect class separation
- However, the extreme performance ( $AUC > 0.99$ ) warrants caution, as it may indicate overfitting to training distribution

**2. Cross-Validation Stability Patterns:** Comparing Figures 2, 3, and 4 reveals a clear hierarchy:

1. **Traditional ML models** (Figure 2a–b) show high variance ( $\sigma_{F1} = 0.036$ ) with performance fluctuating  $\pm 0.05\text{--}0.07$  across folds, indicating sensitivity to data distribution
2. **Deep learning features** reduce variance to 0.028, with more consistent performance patterns
3. **Temporal 3D CNNs** achieve the lowest variance ( $< 0.015$ ), with performance curves showing minimal fluctuation, indicating robust feature learning
4. **Ensemble models** (Figure 3) combine the benefits: XGBoost-R(2+1)D shows both high performance and low variance, with consistent metrics across all folds
5. **Overfitting case** (Figure 4a): ViT-GRU's suspiciously flat performance across folds ( $\sigma_{F1} = 0.002$ ) contrasts with the natural variation seen in properly regularized models

**3. Hyperparameter Sensitivity (Figure 5):** The hyperparameter search plots reveal that:

- Both Logistic Regression and SVM show clear performance landscapes with distinct optimal regions
- Elastic net regularization consistently outperforms pure L1 or L2 penalties
- The sensitivity to hyperparameters ( $C = 0.01\text{--}0.1$  optimal range) correlates with the high cross-validation variance observed in these models
- This suggests that hyperparameter sensitivity is a symptom of limited model capacity, not just a tuning challenge

Table 3: Comprehensive comparison synthesizing insights from all figures. Performance-variance trade-off guides deployment decisions.

Model	F1	AUC	CV $\sigma_{F1}$	Deployment Recommendation
Logistic Regression	0.634	0.679	0.036	Not recommended
SVM	0.631	0.675	0.036	Not recommended
XGBoost (Handcrafted)	0.969	0.997	0.012	Recommended (validate)
XGBoost + R(2+1)D	0.975	$\approx 0.99$	0.010	<b>Best choice</b>
XGBoost + ViT-GRU*	0.996	0.999	0.002	Overfitting risk

**4. Direct Model Family Comparison:** Synthesizing all visualizations, we observe:

#### Key Conclusions from Visual Analysis:

1. **Non-linear modeling is essential:** The 52.8% F1 improvement from logistic regression to XGBoost (visible in both ROC curves and performance tables) demonstrates that linear models are fundamentally limited for this task.
2. **Temporal modeling provides substantial gains:** Comparing XGBoost-Inception (spatial only,  $F1 = 0.728$ ) with XGBoost-I3D (spatiotemporal,  $F1 = 0.958$ ) shows a +31.6% improvement, confirming that temporal information is critical for video classification.
3. **Cross-validation variance is a reliable generalization indicator:** Models with low variance ( $\sigma_{F1} < 0.015$ ) consistently show better generalization in our analysis. The correlation between CV stability and production readiness is evident across all model families.
4. **Overfitting is detectable through multiple signals:** ViT-GRU’s overfitting is visible through: (a) suspiciously low CV variance in Figure 4a, (b) 100% feature retention, (c) near-perfect metrics, and (d) early stopping failure. This multi-signal approach provides robust overfitting detection.
5. **Ensemble approaches optimize the performance-stability trade-off:** XGBoost-R(2+1)D achieves the best balance: highest legitimate performance ( $F1 = 0.975$ ) with lowest variance ( $\sigma_{F1} = 0.010$ ), making it the optimal choice for production deployment.

Table 4: Platform transformation robustness analysis. Models trained with comprehensive augmentation show resilience to common platform transformations.

Transformation	F1 Impact	Models Affected	Augmentation Mitigation
Compression (< 1 Mbps)	-3 to -5%	All models	H.264 bitrate sweep
Resolution change	-1 to -2%	Deep learning	Resolution jitter
Frame rate conversion	< 1%	Temporal models	Frame rate variation
JPEG compression	-2 to -4%	All models	JPEG quality variation

**6. Low performance, high variance:** Traditional ML models ( $F1 \approx 0.63$ ,  $\sigma_{F1} = 0.036$ ) demonstrate both limited capacity and poor generalization, making them unsuitable for production use despite their interpretability.

The ROC/PR curves (Figure 1) complement the cross-validation analysis: while XGBoost shows superior class separation, the CV analysis reveals that this performance is consistent across folds, validating the approach. In contrast, models with high CV variance would show inconsistent ROC/PR curves across different data splits.

**Generalization and Robustness.** Cross-generator evaluation revealed 5–10% F1 degradation on novel generators, highlighting the importance of diverse training data. Quantitative analysis of platform transformations:

Platform transformation analysis demonstrates that models trained with comprehensive augmentation ( $11 \times$  augmentation pipeline) show resilience to common platform transformations. The systematic augmentation strategy (Section A) effectively simulates real-world platform processing, reducing performance degradation from 10–15% (without augmentation) to 1–5% (with augmentation).

## 4.9 Ablation Study

We conducted ablation studies to understand the contribution of different components:

Key findings from ablation analysis:

- **Temporal features provide minimal benefit in linear models:** Adding 11 temporal features to logistic regression yields no improvement ( $F1 = 0.634$ ), indicating that linear models cannot effectively utilize temporal information.

Table 5: Ablation study: Impact of feature engineering and model architecture choices.

Configuration	F1 Score	Relative to Baseline
Logistic Regression (15 features)	$0.634 \pm 0.037$	Baseline
+ Temporal features (26 total)	$0.634 \pm 0.036$	+0.0%
+ Gradient boosting (XGBoost)	$0.969 \pm 0.012$	+52.8%
XGBoost + Inception (spatial only)	$0.728 \pm 0.028$	+14.8%
+ I3D (spatiotemporal)	$0.958 \pm 0.015$	+51.1%
+ R(2+1)D (factorized 3D)	$0.975 \pm 0.010$	+53.8%

Table 6: Feature retention and regularization indicators across models. High feature retention combined with near-perfect performance suggests overfitting.

Model	Features Retained	Early Stop	CV $\sigma$
XGBoost + Inception	81.8% (1989/2432)	45–89	0.028
XGBoost + I3D	79.4% (1930/2432)	52–95	0.015
XGBoost + R(2+1)D	79.2% (1927/2432)	40–103	0.010
XGBoost + ViT-GRU*	<b>100%</b> (3072/3072)	<b>200/200</b>	<b>0.002</b>
XGBoost + ViT-Transformer	82.1% (2523/3072)	35–78	0.025

- **Gradient boosting is critical:** Switching from logistic regression to XGBoost on the same features yields +52.8% F1 improvement, demonstrating that non-linear feature interactions are essential.
- **Spatiotemporal modeling is essential:** XGBoost + I3D achieves +51.1% improvement over baseline, while spatial-only (Inception) achieves only +14.8%, confirming that temporal modeling provides substantial gains.
- **Architecture choice matters:** R(2+1)D’s factorized 3D convolutions provide +2.7% improvement over I3D, suggesting that architectural efficiency can improve both performance and generalization.

## 4.10 Overfitting Analysis

ViT-GRU achieves suspiciously high performance ( $F1 = 0.996 \pm 0.002$ ) with several quantitative overfitting indicators:

Quantitative indicators of overfitting in ViT-GRU:

Table 7: Statistical significance of performance improvements.  $p < 0.001$  indicates highly significant improvement.

Comparison	F1 Improvement	<i>p</i> -value
XGBoost vs. Logistic Regression	+0.335	< 0.001
XGBoost + R(2+1)D vs. XGBoost (Handcrafted)	+0.006	0.142
XGBoost + I3D vs. XGBoost + Inception	+0.230	< 0.001
XGBoost + R(2+1)D vs. XGBoost + I3D	+0.017	0.023

1. **100% feature retention** (3072/3072 vs.  $\sim$ 79–82% for other models), providing excessive model capacity relative to training set size.
2. **Early stopping failure:** Reached maximum iterations (200/200) without triggering, indicating continued learning of dataset-specific patterns rather than generalizable features.
3. **Abnormally low variance:** Cross-validation standard deviation  $\sigma_{F1} = 0.002$  is suspiciously low, suggesting memorization rather than learning generalizable patterns.
4. **Feature space analysis:** Zero collinear features removed indicates no redundancy reduction, unlike other models which remove 18–21% of features.
5. **Performance discrepancy:** Near-perfect scores ( $F1 = 0.996$ ) are inconsistent with task difficulty and other models’ performance ranges ( $F1 = 0.63$ – $0.975$ ).

In contrast, R(2+1)D features ( $F1 = 0.975 \pm 0.010$ ) exhibit proper regularization behavior: 79.2% feature retention (1927/2432), early stopping triggering at iterations 40–103, realistic performance levels, and appropriate cross-validation variance ( $\sigma_{F1} = 0.010$ ). This combination indicates learning generalizable patterns rather than memorization.

## 4.11 Statistical Analysis and Model Comparison

We performed statistical significance testing to compare model performance. Pairwise comparisons using McNemar’s test (for classification accuracy) and paired t-tests (for F1 scores) reveal significant differences between model groups.

Key statistical findings:

- **Gradient boosting provides significant improvement:** XGBoost on hand-crafted features achieves +0.335 F1 improvement over logistic regression ( $p < 0.001$ ), demonstrating the critical importance of non-linear feature interactions.
- **Spatiotemporal features are superior:** XGBoost + I3D shows +0.230 F1 improvement over XGBoost + Inception ( $p < 0.001$ ), confirming that temporal modeling is essential for video classification.
- **Marginal gains from architecture choice:** XGBoost + R(2+1)D provides +0.017 F1 improvement over XGBoost + I3D ( $p = 0.023$ ), indicating that both spatiotemporal architectures are effective, with R(2+1)D providing slight advantages.
- **Deep features vs. handcrafted:** XGBoost + R(2+1)D shows only +0.006 F1 improvement over XGBoost on handcrafted features ( $p = 0.142$ ), suggesting that well-engineered handcrafted features can rival deep learning approaches for this task.

## 4.12 Computational Requirements

Training times varied substantially across architectures:

- Baseline models (Logistic Regression, SVM): minutes on CPU, GPU memory: 16 GB, RAM: 80 GB
- Gradient boosting (XGBoost): 1–2 hours for full dataset, GPU memory: 16 GB, RAM: 80 GB
- XGBoost on pretrained features: 3–6 hours per model (feature extraction dominated), GPU memory: 16 GB, RAM: 80 GB
- Deep learning models: 12–48 hours per fold depending on architecture complexity, GPU memory: 16 GB, RAM: 80 GB

**Memory Challenges.** Memory-intensive architectures (X3D, SlowFast) encountered CUDA out-of-memory errors despite aggressive optimizations including adaptive chunked frame loading with AIMD algorithm, automatic batch size reduction with gradient accumulation, frame-by-frame video decoding (reducing per-video memory by 50×), aggressive garbage collection, and mixed precision

training (FP16). These architectures ultimately proved impractical for our hardware constraints, highlighting the computational challenges of spatiotemporal architectures.

**Caching Mechanisms.** To accelerate training and reduce redundant video decoding, we implemented disk-based frame caching and video metadata caching. Frame caching stores processed frames in compressed numpy format, eliminating repeated video decoding across epochs and folds. Video metadata caching stores frame counts, FPS, and dimensions. These mechanisms reduce training time by 30–50% for subsequent epochs.

## 5 Discussion and Conclusion

This project successfully developed AURA, a multi-architecture pipeline for synthetic video detection. Systematic evaluation across 14 model architectures with comprehensive cross-validation, hyperparameter search, and robustness analysis provides several key insights.

### 5.1 Quantitative Performance Summary

Our evaluation reveals a clear performance hierarchy:

1. **Baseline linear models** ( $F1 \approx 0.63$ ): Limited capacity for this task, demonstrating that simple feature engineering alone is insufficient.
2. **Gradient boosting on handcrafted features** ( $F1 = 0.969$ ): Exceptional performance demonstrating that well-engineered features combined with non-linear modeling can rival deep learning approaches.
3. **Spatiotemporal deep learning features** ( $F1 = 0.958\text{--}0.975$ ): Strong performance with excellent generalization, providing the optimal balance for production deployment.
4. **Overfitting models** ( $F1 = 0.996$ ): Suspiciously high performance indicating memorization rather than generalization.

The 30–35% performance gap between baselines and top-performing models underscores the sophistication of modern synthetic video generators. However, the narrow gap (6%) between handcrafted features ( $F1 = 0.969$ ) and best deep

learning features ( $F1 = 0.975$ ) suggests that domain knowledge in feature engineering remains valuable.

### Key Lessons.

1. Data diversity is paramount—models trained on narrow generator distributions failed to generalize to novel synthesis methods.
2. Temporal modeling is essential, yielding 5–15%  $F1$  improvement from spatiotemporal over frame-only models.
3. Feature engineering retains value—gradient boosting on handcrafted features achieved  $F1 \approx 0.97$ , rivaling deep learning approaches.
4. Preprocessing complexity demanded substantial engineering effort to handle the heterogeneity of short-form video content.

For deployment, spatiotemporal models provide optimal discrimination but require substantial computational resources. Frame-temporal models offer practical alternatives with 85% of peak performance at significantly lower inference cost. Well-calibrated confidence scores enable effective human-in-the-loop systems where borderline cases receive expert review.

**Future Work.** Promising directions include audio-visual fusion to leverage audio inconsistencies, attention visualization to identify salient video regions, knowledge distillation for lightweight deployment, adversarial robustness evaluation against adaptive attacks, multimodal context integration (metadata, upload patterns, user history), fine-grained temporal localization, active learning strategies, and real-time streaming detection.

**Broader Impact.** As synthetic video generation technology advances, detection systems must evolve in parallel. Our work demonstrates that combining traditional feature engineering with modern deep learning approaches yields state-of-the-art performance. However, the arms race between generation and detection suggests no single approach will remain effective indefinitely. Future systems should incorporate multiple complementary detection strategies and maintain human oversight for critical decisions.

## 6 Reflection

**What Worked Well.** The multi-architecture approach proved valuable, revealing that gradient boosting on handcrafted features ( $F1 = 0.969$ ) can rival learn-

ing approaches, validating the importance of domain knowledge in feature engineering. The systematic evaluation across multiple generators provided crucial insights into generalization challenges. The five-stage pipeline architecture enabled modular development and independent optimization. Comprehensive data augmentation ( $11\times$ ) significantly improved model robustness to platform transformations.

**Challenges Encountered.** Data collection presented unexpected obstacles. Initial scraping attempts encountered platform Terms of Service violations and revealed demographic biases that would compromise generalization. We pivoted to the MKLab dataset, which resolved these issues but required adapting our approach mid-project. Computational constraints were significant: deep learning models required 12–48 hours per fold, necessitating efficient hyperparameter search strategies and careful resource management. The ViT-GRU model’s overfitting ( $F1 = 0.996$ ) highlighted the importance of regularization and cross-generator evaluation, revealing that near-perfect validation performance can mask generalization failures.

**Approaches That Proved Infeasible.** Several initially promising directions did not yield practical results:

- **Physiological feature extraction:** Micro-expression detection, blink analysis, and rPPG extraction proved impractical for diverse short-form content due to tracking failures and sensitivity to lighting and occlusion. We simplified to robust handcrafted features, improving system reliability.
- **Platform data scraping:** Legal and ethical constraints rendered direct platform scraping infeasible, necessitating the pivot to curated research datasets.
- **Memory-intensive architectures:** Full SlowFast and X3D training exceeded computational budgets even with aggressive optimization, requiring prioritization of feature-based approaches.

**Recommendations for Future Work.** We recommend establishing data collection strategies early, including partnerships with research groups providing curated datasets. Feature-based models (XGBoost) should be prioritized early given their exceptional performance and computational efficiency. More aggressive regularization should be implemented from the outset, particularly for high-dimensional feature spaces. Cross-generator evaluation protocols should be designed from the beginning rather than as an afterthought. Finally, clearer success metrics and deployment constraints should be established upfront to guide architectural decisions.

## Acknowledgments

We gratefully acknowledge Dr. Olga Papadopoulou and the MKLab ITI team for providing the Fake Video Dataset corpus [1]. We thank the University of Michigan Great Lakes HPC cluster for computational resources. This work was supported by the School of Information at the University of Michigan.

**Code Availability.** The complete implementation, including all model architectures, training scripts, and evaluation code, is publicly available at <https://github.com/Horopter/AURA>.

## References

- [1] O. Papadopoulou, M. Zampoglou, S. Papadopoulos, and I. Kompatsiaris. A corpus of debunked and verified user-generated videos. *Online Information Review*, 2018. DOI: 10.1108/OIR-03-2018-0101. 6, 22
- [2] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner. Face-Forensics++: Learning to detect manipulated facial images. In *ICCV*, 2019. 2
- [3] C. Feichtenhofer, H. Fan, J. Malik, and K. He. SlowFast networks for video recognition. In *ICCV*, 2019. 3
- [4] C. Feichtenhofer. X3D: Expanding architectures for efficient video recognition. In *CVPR*, 2020. 3
- [5] U. A. Ciftci, I. Demir, and L. Yin. FakeCatcher: Detection of synthetic portrait videos using biological signals. *IEEE TPAMI*, 2020. 3
- [6] D. Güera and E. J. Delp. Deepfake video detection using recurrent neural networks. In *AVSS*, 2018. 2
- [7] D. Gragnaniello, D. Cozzolino, F. Marra, G. Poggi, and L. Verdoliva. Are GAN generated images easy to detect? A critical analysis of the state-of-the-art. In *ICME*, 2021. 3
- [8] N. Carlini and H. Farid. Evading deepfake-image detectors with white- and black-box attacks. In *CVPR Workshops*, 2020. 3

## A Pipeline Architecture Details

Our Fake Video Classification system implements a comprehensive five-stage pipeline designed to handle the substantial variability inherent in short-form video content.

### A.1 Stage 1: Video Ingestion and Preprocessing

This stage performs format standardization (MP4/H.264), resolution normalization ( $720 \times 1280$  or  $1080 \times 1920$ ), frame rate standardization (24–30 fps), and temporal clipping (5–8 seconds). The stage ensures consistent input formats across diverse source materials, handling variations in encoding, resolution, and duration that are common in user-generated short-form content.

### A.2 Stage 2: Multi-Modal Augmentation

The  $11 \times$  augmentation pipeline incorporates four categories of perturbations:

1. **Compression perturbations:** JPEG quality variation (60–100), H.264 bitrate sweeps (1–10 Mbps) to simulate platform re-encoding.
2. **Temporal perturbations:** Frame dropping (0–10%), frame rate conversion to simulate variable capture and playback conditions.
3. **Spatial transforms:** Resolution jitter ( $\pm 10\%$ ), random crops, color adjustments to improve invariance to capture conditions.
4. **Noise injection:** Gaussian noise, compression artifacts to improve robustness to degraded content.

This augmentation strategy increases dataset diversity and improves generalization to real-world platform transformations that videos undergo during upload and distribution.

### A.3 Stage 3: Feature Extraction

Parallel extraction of two feature categories:

- **Handcrafted features:** Noise residuals, DCT coefficients, edge/boundary analysis, blur/sharpness metrics, codec-centric cues capturing domain-specific artifacts.

- **Deep learning features:** Spatial backbones (Inception, ViT) with temporal aggregation (GRU, Transformer) capturing learned hierarchical representations.

This dual approach captures both domain-specific artifacts that experts have identified as discriminative and learned representations that may capture subtle patterns not easily specified manually.

#### A.4 Stage 4: Classification

Multiple model architectures spanning baseline linear models (Logistic Regression, SVM) to gradient boosting (XGBoost) to spatiotemporal deep learning (I3D, R(2+1)D, ViT-GRU). Models are trained with 5-fold stratified cross-validation, hyperparameter search, and comprehensive regularization including L1/L2 penalties, dropout, and early stopping.

#### A.5 Stage 5: Post-Processing

Temperature scaling for probability calibration ensures that confidence scores accurately reflect prediction reliability. Confidence thresholding routes low-confidence predictions to human reviewers, enabling effective human-in-the-loop moderation. Ensemble aggregation combines predictions from multiple models when available. Explainability visualizations highlight salient regions for model interpretability.

## B Feature Engineering Details

### B.1 Handcrafted Features (Stage 2)

We extracted 15 handcrafted features from each video designed to capture compression artifacts and encoding signatures:

1. **Noise residual energy (3 features):** Capturing compression artifacts and encoding signatures by analyzing the high-frequency residual after denoising. Synthetic videos often exhibit different noise characteristics due to their generation process.

2. **DCT statistics (5 features)**: DC and AC coefficients (mean, standard deviation, energy) revealing frequency domain artifacts. Block-based compression leaves characteristic patterns in the DCT domain that differ between real and synthetic content.
3. **Blur/sharpness metrics (3 features)**: Laplacian variance and gradient statistics capturing focus characteristics. Synthetic videos may exhibit unnaturally uniform sharpness or characteristic blur patterns.
4. **Boundary inconsistency (1 feature)**: Detecting block boundary artifacts from compression. Real videos exhibit consistent blocking artifacts while synthetic videos may show inconsistent patterns.
5. **Codec parameters (3 features)**: Bitrate, fps, and resolution metadata. These capture encoding choices that may correlate with content authenticity.

## B.2 Scaled Features (Stage 4)

An additional 23 features extracted from scaled video versions:

- **Temporal consistency metrics**: Measuring frame-to-frame coherence in motion, lighting, and content.
- **Frame-to-frame variation statistics**: Quantifying the distribution of inter-frame differences.
- **Multi-resolution analysis**: Extracting features at multiple spatial scales to capture both fine-grained and coarse artifacts.

The combination of Stage 2 and Stage 4 features (38 total) provides comprehensive coverage of both spatial and temporal artifacts. After collinearity removal (correlation  $\rho \geq 0.95$ ), 26 features are retained.

## B.3 Deep Learning Features

Pretrained models extract high-dimensional feature vectors:

- **Inception-v3, I3D, R(2+1)D**: 2432-dimensional features
- **ViT-based models**: 3072-dimensional features

After removing zero-variance and highly collinear features (correlation  $\rho \geq 0.95$ ), most models retain approximately 80% of features (1900–2000 dimensions). However, ViT-GRU retained 100% of features (3072/3072), indicating no redundancy reduction and potential overfitting risk due to the high-dimensional feature space relative to training set size.

These features capture hierarchical spatial and temporal patterns learned from large-scale video datasets (ImageNet for Inception/ViT, Kinetics-400 for I3D/R(2+1)D), enabling transfer learning to the synthetic detection task.

## C Training Strategy and Hyperparameters

### C.1 Cross-Validation Protocol

We employed 5-fold stratified cross-validation with a fixed random seed (42) to ensure reproducibility. Stratification preserved the class distribution (real vs. synthetic) across folds. For hyperparameter search, we used a 20% stratified sample of the dataset to efficiently explore parameter spaces before full training.

### C.2 Hyperparameter Search

Grid search was performed for all model families:

#### Logistic Regression (40 combinations):

- $C \in \{0.01, 0.1, 1.0, 10.0\}$
- Penalty  $\in \{\text{L1}, \text{L2}, \text{ElasticNet}\}$
- Solver  $\in \{\text{liblinear}, \text{saga}\}$

#### SVM (36 combinations):

- $C \in \{0.1, 1.0, 10.0\}$
- $\gamma \in \{\text{scale}, \text{auto}, 0.01, 0.1\}$
- Kernel  $\in \{\text{RBF}, \text{linear}, \text{poly}\}$

#### XGBoost (64 combinations):

- n\_estimators  $\in \{100, 200\}$

- `max_depth`  $\in \{3, 5\}$
- `learning_rate`  $\in \{0.01, 0.1\}$
- `subsample`  $\in \{0.8, 1.0\}$
- `colsample_bytree`  $\in \{0.8, 1.0\}$

Best hyperparameters were selected based on cross-validation F1 scores.

### C.3 Regularization Strategies

Models were trained with comprehensive regularization:

- **Linear models:** L1/L2 regularization with cross-validated penalty strength
- **Neural networks:** Weight decay ( $10^{-5}$  to  $10^{-4}$ ), dropout (0.1–0.3)
- **All models:** Early stopping with patience of 5–10 epochs monitoring validation loss
- **XGBoost:** Early stopping with 20% validation split and maximum 200 estimators

### C.4 Optimization Details

PyTorch models used the AdamW optimizer with:

- Learning rates:  $10^{-4}$  to  $10^{-3}$
- Cosine annealing learning rate scheduling
- Gradient clipping (max norm 1.0)
- Mixed precision training (FP16) for memory efficiency

Batch sizes varied from 1 (with gradient accumulation for effective batch size 8–16) for memory-intensive models to 32 for feature-based models.

## C.5 Data Augmentation During Training

During training, videos underwent stochastic augmentation:

- Compression: JPEG quality 60–100, H.264 bitrate 1–10 Mbps
- Temporal: Frame dropping 0–10%, frame rate conversion
- Spatial: Resolution jitter  $\pm 10\%$ , random crops, color adjustments
- Noise: Gaussian noise injection, compression artifact simulation

This  $11\times$  augmentation effectively increased dataset diversity and improved generalization to platform transformations.

## D Memory Optimizations

### D.1 Frame-by-Frame Video Decoding

Our implementation uses PyAV to decode only required frames instead of loading entire videos into memory. This reduces per-video memory from  $\sim 1.87$  GB (300 frames at  $1920 \times 1080$ ) to  $\sim 37$  MB (6 frames), achieving  $50\times$  memory reduction. The implementation seeks to specific frame indices and decodes only those frames, with fallback to full video loading if frame-by-frame decoding fails.

### D.2 Adaptive Chunked Frame Loading

Memory-intensive models (X3D, SlowFast, I3D, R(2+1)D) use adaptive chunked loading. The system starts with initial chunk sizes (4 for X3D/SlowFast, 10 for other memory-intensive models, 30 for moderately intensive models) and adapts using an AIMD (Additive Increase Multiplicative Decrease) algorithm: on OOM, chunk size is halved; on successful processing, chunk size increases incrementally. This prevents OOM errors while maximizing throughput.

### D.3 OOM Error Handling

The training pipeline implements comprehensive OOM handling:

1. Detection of CUDA OOM errors from multiple error message patterns

2. Automatic retry with reduced batch size (up to 4 attempts)
3. Aggressive garbage collection (10 passes of cache clearing and synchronization)
4. Model-specific batch size constraints (X3D requires batch size  $\leq 1$ , automatically enforced with gradient accumulation)

## D.4 Resource Allocation

Training jobs were configured with 80 GB RAM, 24-hour time limits, and single GPU allocation. Environment variables were configured to reduce memory fragmentation and limit memory block sizes.

## D.5 Training Time Measurements

Actual training times from experimental logs:

- ViT-GRU: 72,883 seconds ( $\sim 20.25$  hours) for 5-fold cross-validation, with feature extraction consuming  $\sim 18\text{--}20$  hours due to processing 400 frames per video
- Gradient boosting models: 1–2 hours
- XGBoost on pretrained features: 3–6 hours per model, with feature extraction as the primary bottleneck

Inference times:

- Baselines: < 10 ms per video
- XGBoost: 50–200 ms including feature extraction
- Deep learning: 100–500 ms on GPU

## D.6 Memory Usage by Model Type

- Feature-based models (baseline, XGBoost): GPU memory: 16 GB, RAM: 80 GB
- XGBoost on pretrained features: GPU memory: 16 GB, RAM: 80 GB

- Deep learning models: GPU memory: 16 GB, RAM: 80 GB
- X3D: Consumed full GPU memory (16 GB) before failing with OOM errors, RAM: 80 GB

## E Caching Mechanisms

To accelerate training and reduce redundant video decoding operations, we implemented comprehensive caching mechanisms.

### E.1 Frame Caching

The system caches processed frames (after transforms) to disk in compressed numpy format, eliminating repeated video decoding across epochs and folds.

**Cache Key Generation.** Cache keys are generated using MD5 hashes of video path, modification time, number of frames, and optional seed. This ensures cache invalidation when videos are modified and supports deterministic frame sampling.

**Storage Format.** Frames are stored as compressed numpy arrays (uint8 format), achieving 2–3× disk space reduction compared to uncompressed storage. Cache files are organized in subdirectories based on hash prefixes to avoid filesystem limitations.

**Memory Efficiency.** The caching system minimizes memory footprint: frames are processed one at a time, cloned to CPU before caching, and memory is immediately freed with aggressive garbage collection. Cache loading is lazy—frames are loaded only when needed.

**Performance Impact.** Frame caching reduces training time by 30–50% for subsequent epochs. For models processing 400–1000 frames per video (e.g., ViT-GRU), caching eliminates ∼18–20 hours of redundant video decoding across 5-fold cross-validation. Cache hit rates typically exceed 90% after the first epoch.

### E.2 Video Metadata Caching

Video metadata caching avoids expensive video container opening operations by caching frame counts, FPS, dimensions, and duration.

**Two-Level Caching.** The system uses both in-memory caching (process lifetime) for fast access during training runs and persistent caching (JSON file on disk) for persistence across runs.

**Cache Invalidation.** Cache keys are based on video path and modification time, ensuring automatic invalidation when video files are modified.

**Performance Impact.** Video metadata caching reduces initialization time by 60–80% for datasets with thousands of videos. For our dataset of 3277 videos, metadata extraction without caching requires ~30–60 minutes; with caching, subsequent runs complete in seconds.

### E.3 Cache Management

**Cache Size.** Frame cache size depends on dataset size and frames per video. For 3277 videos with 400 frames each at  $256 \times 256$  resolution, cache size is approximately 400–600 GB with compression. Video metadata cache is minimal ( $\sim 1\text{--}2$  MB).

**Cache Operations.** The system provides utilities for selective or complete cache clearing and cache size monitoring. Cache hit/miss rates are logged during training for performance analysis.