Cross-Modal Video Anomaly Detection: Visual-Audio Alignment via Optical Flow and Fourier Analysis

Anna Dai Lily Ru Matt Wang
Johns Hopkins University
Baltimore, MD, USA
{ndai3,yru2,hwang302}@jh.edu

Abstract

We propose a system for detecting audio-visual misalignment in video content by combining compact visual and audio representations with two complementary detection methods. Visual features are extracted via dense optical flow, while audio features are captured using Mel-Frequency Cepstral Coefficients (MFCCs). To detect temporal inconsistencies, we evaluate both a threshold-based method using normalized Euclidean distance and an enhanced Multi-Layer Perceptron (MLP) classifier with residual connections and Swish activations. Using a synthetically augmented version of the AVE dataset, we demonstrate that both methods effectively identify cross-modal anomalies, with the MLP achieving an F1 score of 0.7433 and the thresholding method offering competitive performance (F1 = 0.7266) with lower computational cost. Our results highlight the trade-off between accuracy and efficiency, and point toward practical applications in content moderation, quality control, and accessibility. This work offers a lightweight, extensible framework for cross-modal anomaly detection with potential for real-world deployment.

1. Introduction

With the rapid proliferation of multimedia content, ensuring the temporal consistency between audio and visual streams has become increasingly important. [19] While traditional video anomaly detection systems focus primarily on detecting unusual visual patterns, many real-world anomalies arise from cross-modal discrepancies—such as an actor's lips moving out of sync with their voice, or sound effects that fail to match visual events. These types of anomalies are critical in contexts like deepfake detection, content authentication, and automated quality assurance.

This paper presents a system for cross-modal video anomaly detection, specifically targeting synchronization mismatches between audio and visual modalities. We focus on temporal misalignment (e.g., delayed or advanced audio), as well as modality corruption (e.g., audio noise, mute, or video playback distortion) that may disrupt a viewer's perception of coherence.

To tackle this challenge, we extract compact representations of video and audio using optical flow and Mel-Frequency Cepstral Coefficients (MFCCs), respectively. We then evaluate two approaches for detecting misalignment: a simple thresholding method based on the normalized Euclidean distance between audio-visual features, and a Multi-Layer Perceptron (MLP) classifier trained on aligned and synthetically misaligned samples.

Our contributions are as follows:

- We build a feature extraction pipeline that converts raw videos into synchronized audio-visual feature vectors.
- We introduce a dataset of synthetically generated audio-visual misalignment based on the AVE dataset to address the lack of labeled anomalies.
- We compare a baseline thresholding method and a neural MLP classifier, analyzing trade-offs in performance, complexity, and robustness.

Overall, this work contributes a lightweight and extensible framework for studying audio-visual synchronization, with implications for both anomaly detection research and practical media analysis tools.

2. Dataset

We use the Audio-Visual Event (AVE) dataset [18] as the foundation for our anomaly detection experiments. AVE is a large-scale, diverse collection of audio-visual aligned video clips designed for studying sound-source localization and multi-modal event understanding. It contains 4,143 videos across 28 event categories, such as dog barking, playing violin, and fireworks.

2.1. Annotation Format and Conversion

The original AVE annotations are provided in a text format with complex delimiters:

Category&VideoID&Quality&StartTime&EndTime.

To facilitate downstream processing, we convert the annotations into a structured CSV format. This conversion simplifies data access in Python and enables batch preprocessing. Additional derived fields, such as segment duration, are included to support filtering and sampling operations.

2.2. Segment Trimming

Each video, based on the timestamp annotations, is split into shorter segments with random length. These trimmed segments serve as the base units for aligned (positive) and misaligned (negative) training samples. We use FFmpeg [1] to extract each segment with both audio and video streams preserved. Segments shorter than one second are excluded to maintain data quality.

2.3. Synthetic Misalignment Generation

Since the AVE dataset does not contain labeled misaligned samples, we generate synthetic anomalies to enable evaluation and supervised training. For each trimmed segment, we randomly decide whether it will remain aligned (no change) or apply one of the following transformations:

- **Time Shift:** We delay the audio stream relative to the video using the `adelay` filter. Mathematically, this simulates a time-domain translation:

$$x'(t) = x(t - \Delta t)$$

  where Δt is randomly sampled between 100 ms and 80% of the segment duration. This shift results in temporal misalignment without altering audio content.

- **Mute:** The audio track is removed entirely using the `-an` flag. The resulting signal becomes:

$$x'(t) = 0 \quad \text{for all } t$$

  which effectively eliminates all temporal cues from the audio stream.

- **Noise:** We replace the original audio with additive white noise generated via `anoisesrc`. This can be viewed as:

$$x'(t) = x(t) + \epsilon(t), \quad \epsilon(t) \sim \mathcal{N}(0, \sigma^2)$$

  where $\epsilon(t)$ is Gaussian-distributed noise.

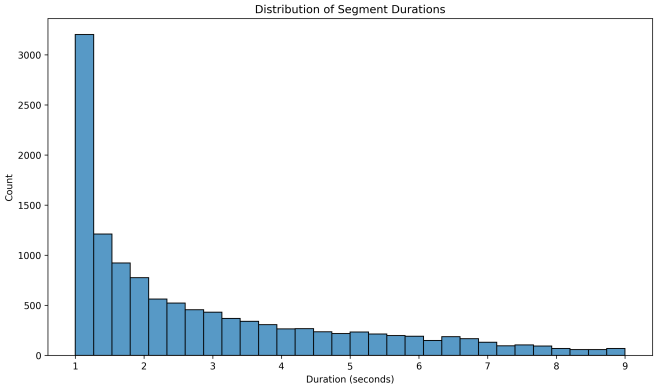

Figure 1. Distribution of segment durations after trimming. Most segments are short, with a right-skewed distribution extending to 9 seconds.

- **Distort:** We modify the temporal pacing of the video using the `setpts` filter:

$$\texttt{PTS}' = 0.75 \cdot \texttt{PTS} + \sin(N \cdot 0.05) \cdot \frac{0.05}{\texttt{TB}}$$

  where N is the frame index and `TB` is the timebase. This creates nonlinear frame intervals, resulting in perceived warp in the visual timeline while preserving the original audio.

The final dataset includes 12,114 segments: 6,049 aligned (50.1%) and 6,065 misaligned (49.9%), with misalignment types approximately evenly distributed. Metadata for each sample, including its label and transformation type, is saved in a CSV file to support model training and evaluation.

2.4. Dataset Statistics

Figure 1 shows the distribution of segment durations after trimming based on the AVE annotations. Most segments are between 1 and 2 seconds long, with a steep drop-off as duration increases. This long-tailed distribution reflects natural variability in event boundaries and ensures a diversity of temporal contexts during training.

To maintain class balance, we generated a roughly equal number of aligned and misaligned samples. Figure 2 illustrates the breakdown of the four synthetic misalignment types: `noise`, `distort`, `mute`, and `time_shift`. The first three categories are nearly equal in frequency, while `time_shift` is slightly underrepresented due to stricter constraints on segment length.

These visualizations confirm that our dataset includes a wide range of durations and misalignment types, helping to improve the robustness and generalization of our models.

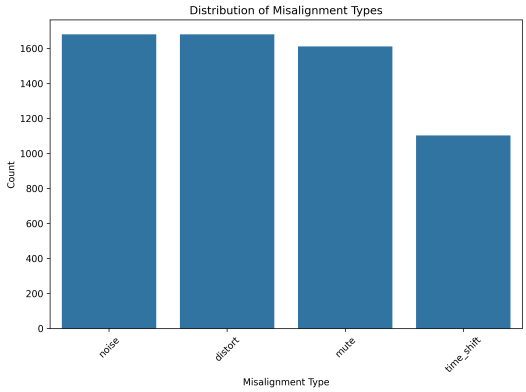

Figure 2. Distribution of misalignment types across synthetic negative samples. The roughly balanced types support robust training.

3. Methodology

3.1. Visual Feature Extraction

For the visual modality, we extract motion-sensitive features using **dense optical flow**, which captures pixel-level displacement between consecutive grayscale frames. This technique is effective for identifying local motion dynamics and temporal misalignment.

Videos are sampled uniformly at 5 frames per second. Each frame is resized to 96×96 and converted to grayscale. Let $I_t(x, y)$ and $I_{t+1}(x, y)$ denote two consecutive grayscale frames at time steps t and $t + 1$. The goal of optical flow is to estimate a displacement field $\mathbf{d}_t(x, y) = (u_t(x, y), v_t(x, y))$ such that the brightness constancy assumption holds:

$$I_t(x, y) \approx I_{t+1}(x + u_t(x, y),\ y + v_t(x, y))$$

This assumes that the brightness of each point remains constant as it moves across frames. We compute $\mathbf{d}_t$ using the Farneback method [5, 7], which fits a quadratic polynomial to local image neighborhoods and solves for the displacement analytically.

From the estimated flow field $\mathbf{d}_t(x, y)$, we compute the magnitude and angle of motion at each pixel:

$$M_t(x, y) = \sqrt{u_t(x, y)^2 + v_t(x, y)^2}$$

$$\Theta_t(x, y) = \arctan 2\,(v_t(x, y), u_t(x, y))$$

Here, M_t measures how much motion occurs at each location, while Θ_t encodes the direction of that motion.

To represent the entire frame as a single feature vector, we flatten the magnitude and angle matrices and concatenate them:

$$\mathbf{v}_t = \text{vec}(M_t) \,\|\, \text{vec}(\Theta_t)$$

where $\text{vec}(\cdot)$ denotes flattening into a column vector and $\|$ denotes concatenation. This gives a fixed-length feature vector for each frame, capturing both the strength and direction of motion over time.

3.2. Audio Feature Extraction

To capture perceptually meaningful structure in the audio stream, we extract **Mel-Frequency Cepstral Coefficients (MFCCs)**, a compact representation that approximates the human auditory response [2].

The audio is first resampled to 16 kHz to ensure uniform resolution across samples. We apply the Short-Time Fourier Transform (STFT) to convert the 1D waveform into a time-frequency representation [4]:

$$X(t, \omega) = \sum_{n=-\infty}^{\infty} x[n] \cdot w[n - t] \cdot e^{-i\omega n}$$

where $x[n]$ is the audio signal, $w[n]$ is a Hann window, and $X(t, \omega)$ is the complex-valued spectrogram.

To account for human pitch perception, we apply a Mel-scale filter bank to the magnitude spectrum [17]:

$$M_k(t) = \sum_{\omega} |X(t, \omega)|^2 \cdot H_k(\omega), \quad k = 1, \ldots, K$$

where $H_k(\omega)$ is the k-th triangular Mel filter. The filter bank emphasizes lower frequencies, which are perceived more distinctly by humans and often carry critical information such as rhythm, timbre, and pitch.

Next, we apply a logarithmic transform to compress dynamic range, followed by the Discrete Cosine Transform (DCT) to decorrelate the coefficients [3, 6]:

$$\text{MFCC}_n(t) = \sum_{k=1}^{K} \log M_k(t) \cdot \cos\left(\frac{\pi n(k - 0.5)}{K}\right)$$

We retain the first $N = 13$ coefficients, capturing the fundamental spectral envelope while discarding insignificant variation. This yields a feature matrix of shape $(T, 13)$, where T is the number of audio frames.

3.3. Feature Processing and Storage

Once the visual and audio features are extracted, we perform temporal alignment and label association before storing the data for downstream tasks.

Temporal Alignment. Since audio and visual streams may differ slightly in length due to rounding or frame rate discrepancies, we truncate the longer stream to match the length of the shorter one. This ensures synchronized feature vectors across both modalities at each time step.

Label Association. Labels indicating whether a segment is aligned or misaligned are extracted from file naming patterns or metadata CSV files generated during preprocessing.

Storage Format. The final aligned audio and visual feature matrices, along with their corresponding labels, are saved in NumPy's compressed `.npz` format. This format allows efficient loading during training and evaluation while keeping disk usage low. Each `.npz` file contains:

- `audio`: MFCC feature matrix
- `visual`: Optical flow feature matrix
- `label`: Binary value (1 = aligned, 0 = misaligned)

4. Models

To detect audio-visual misalignment, we propose two complementary approaches: a lightweight statistical method based on thresholding, and a data-driven neural model using a enhanced multilayer perceptron (MLP) classifier. The thresholding method computes alignment scores from low-level audio and visual features and flags anomalies based on a learned decision boundary. In contrast, the MLP model learns to classify aligned and misaligned pairs directly from feature vectors, leveraging non-linear interactions between modalities.

4.1. Thresholding Method

4.1.1 Motivation and Key Idea

After extracting frame–level feature vectors from the audio and visual streams of each video clip, we measure their synchrony with a continuous alignment score. A clip is deemed normal if its score is below a data-driven threshold and anomalous otherwise. The threshold is selected on the training set through a Receiver Operating Characteristic (ROC) analysis [8] so that the resulting detector offers the best compromise between sensitivity (true–positive rate) and specificity (false–positive rate).

4.1.2 Data Preparation

We begin by loading the precomputed feature matrices:

$$\mathbf{A} = \big[a_1,\, a_2,\, \ldots,\, a_N\big] \in \mathbb{R}^{d\times N}$$
$$\mathbf{V} = \big[v_1,\, v_2,\, \ldots,\, v_M\big] \in \mathbb{R}^{d\times M}$$

where d is the feature dimensionality, and N and M are the numbers of audio and visual frames, respectively. Because the two modalities can differ in length, we keep the first $L = \min(N, M)$ frames of each modality so that both streams align in time. The dataset is then randomly split into a stratified $70\%/30\%$ training/test partition that preserves the class distribution.

4.1.3 Alignment Score Computation

For each clip, let $\mathbf{a} = [a_1, \ldots, a_L]$ and $\mathbf{v} = [v_1, \ldots, v_L]$ denote the truncated audio and visual feature sequences of length L. The alignment score is computed as the normalized Euclidean distance between these sequences:

$$s(\mathbf{a}, \mathbf{v}) = \frac{\|\mathbf{a} - \mathbf{v}\|_2}{\sqrt{L}}$$

$$\text{where} \quad \|\mathbf{a} - \mathbf{v}\|_2 = \sqrt{\sum_{t=1}^{L} \|a_t - v_t\|_2^2}.$$

The normalization by $\sqrt{L}$ serves two purposes. First, it ensures that the alignment score is consistent and comparable across clips of different lengths or overall feature magnitudes. More importantly, this normalization reflects the average per-frame discrepancy between modalities, rather than the total accumulated difference.

This is critical for anomaly detection: we are interested in identifying relative misalignments that signal a divergence from normal temporal synchrony. By normalizing, we suppress scale differences and highlight meaningful variations in synchrony. By assumptionx, lower scores would indicate strong temporal synchrony, whereas higher scores suggest misalignment and therefore a potential anomaly.

4.1.4 Optimal Threshold Selection via ROC Analysis

Since the polarity of alignment scores can vary across datasets—where, in some cases, lower scores indicate stronger misalignment—we evaluate both the original scores $s(\mathbf{a}, \mathbf{v})$ and their negation $-s(\mathbf{a}, \mathbf{v})$ on the training data. We compute the area under the ROC curve (AUC) for both and retain the version that achieves the higher AUC. This flipping step ensures a consistent interpretation in which higher (effective) scores always reflect a greater likelihood of anomaly. Consequently, threshold selection and downstream evaluation remain robust and comparable across diverse input distributions and modalities.

To convert alignment scores into binary anomaly labels, we sweep a dense grid of candidate thresholds $\tau \in \{\tau_1, \ldots, \tau_K\}$ on the training set. For each threshold τ_k, we assign an anomalous predicted label if

$$\begin{cases} s(\mathbf{a}, \mathbf{v}) > \tau_k, & \text{if } \texttt{flipped} = \texttt{False}, \\ s(\mathbf{a}, \mathbf{v}) < \tau_k, & \text{if } \texttt{flipped} = \texttt{True}. \end{cases}$$

This procedure yields an operating point in ROC space defined by:

$$\text{TPR}(\tau_k) = \frac{\text{TP}(\tau_k)}{\text{TP}(\tau_k) + \text{FN}(\tau_k)}$$

$$\text{FPR}(\tau_k) = \frac{\text{FP}(\tau_k)}{\text{FP}(\tau_k) + \text{TN}(\tau_k)}.$$

After finding the optimal ROC curve, we choose the threshold $\tau^\star$ that maximizes Youden's J statistic [16], which is defined as the difference between true positive rate (TPR) and false positive rate (FPR):

$$J(\tau) = \text{TPR}(\tau) \; - \; \text{FPR}(\tau)), \quad \tau^\star = \arg\max_\tau J(\tau)$$

Then we set the point on the ROC curve farthest from random guessing as the optimal threshold $\tau^\star$, which we then apply to the alignment scores of the test clips for performance evaluation.

4.2. MLP Classifier

We propose an enhanced Multi-Layer Perceptron (MLP) model, `SyncDetectorMLP`, designed to classify whether a given audio-visual pair is temporally aligned. This model builds upon standard feedforward architectures [10, 14] by incorporating two key innovations: (1) the Swish activation function, which improves expressiveness through smoother nonlinearity, and (2) residual blocks that stabilize training and support deeper structures.

4.2.1 Input Representation

Each sample is constructed by concatenating visual and audio feature sequences along the feature dimension, followed by temporal average pooling. This yields a fixed-dimensional vector representing multimodal content. The motivation for this design is to preserve temporal alignment while ensuring dimensional consistency across samples.

4.2.2 Swish Activation Function

We use the **Swish** activation function [15], defined as:

$$\text{Swish}(x) = x \cdot \sigma(x),$$

where $\sigma(x)$ is the sigmoid function. Unlike ReLU, Swish is smooth and non-monotonic, enabling better gradient flow and preserving negative activations—particularly beneficial in deep or residual architectures.

4.2.3 Residual Block Structure

To improve gradient propagation and model depth, we employ residual blocks [9]. Each residual block takes the form:

$$R(x) = \text{Swish}(x + F(x)),$$

where $F(x)$ is a two-layer non-linear transformation:

$$F(x) = \text{BN}_2\left(W_2 \cdot \text{Swish}\left(\text{BN}_1(W_1 x + b_1)\right) + b_2\right).$$

Here, $W_1, W_2 \in \mathbb{R}^{d\times d}$ are linear projection weights, $b_1, b_2 \in \mathbb{R}^d$ are biases, BN_1, BN_2 are batch normalization layers [11], and Dropout adds stochastic regularization. The skip connection improves information flow and alleviates vanishing gradient issues.

4.2.4 Model Architecture

The overall network maps an input feature vector x to a binary prediction $\hat{y}$ via two residual stages and a final sigmoid output:

$$z = W_3 \cdot R_2\left(W_2 \cdot R_1\left(W_1 x + b_1\right) + b_2\right) + b_3, \quad \hat{y} = \sigma(z),$$

where W_1, W_2, W_3 and b_1, b_2, b_3 are the weights and biases of linear layers, and R_1, R_2 are residual blocks as defined above. The output $\hat{y}$ represents the probability that the input segment is synchronized.

4.2.5 Layer Specification

The layer-wise architecture is:

- **Input Projection:** Linear(9229 $\rightarrow$ 512) + BatchNorm + Swish + Dropout(0.3)
- **Residual Stage 1:** Linear(512 $\rightarrow$ 256) + Residual Block(256)
- **Residual Stage 2:** Linear(256 $\rightarrow$ 64) + Residual Block(64)
- **Output Layer:** Linear(64 $\rightarrow$ 1)

4.2.6 Training Configuration

We train the model using binary cross-entropy with logits:

$$\mathcal{L}(y, \hat{y}) = -\left[y \log \sigma(\hat{y}) + (1-y)\log(1-\sigma(\hat{y}))\right],$$

where $y \in \{0, 1\}$ is the ground-truth label.

Optimization is performed using the **Adam** algorithm [12, 13], an adaptive gradient method that maintains per-parameter learning rates. At each step t, Adam updates parameters θ_t according to:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta \mathcal{L}_t$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\left(\nabla_\theta \mathcal{L}_t\right)^2,$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t},$$

$$\theta_{t+1} = \theta_t - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

where α is the learning rate (8×10^{-7} in our case), and the default parameters of $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$.

4.2.7 Discussion

We explored a wide range of architectural and hyperparameter configurations to arrive at the final setup presented. These included experimenting with simpler MLP structures, larger learning rates, shallower networks, and smaller projection dimensions. Through extensive empirical evaluation, we selected the configuration that exhibited the most consistent and pronounced downward trends in both training and development loss, while also achieving high accuracy and F1 scores. This final model demonstrated the best balance between stability and performance.

5. Evaluation and Results

5.1. Evaluation Metrics

We report standard classification metrics for both methods:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$F_1 \text{ Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

5.2. Thresholding Results

Running our model on the preprocessed test data, we generated the ROC curve as shown in Figure 3. We can observe that AUC [20] is 0.57 (area below the ROC curve), and the optimal threshold is -0.513. With the selected threshold, we generated performance metrics.

As shown in Table 1, the simple thresholding baseline achieves a moderate overall accuracy of 0.6225. Its high recall of 0.7704 demonstrates good sensitivity to genuine misalignments, but the comparatively lower precision (0.6875) reveals a palpable number of false alarms. The resulting F_1 score (0.7266) represents a reasonable balance.

5.3. MLP Results

Figure 4 presents the learning curves of the enhanced MLP model trained for 100 epochs, including loss, accuracy, F1 score, precision, and recall for both training and validation sets.

Loss. The training and validation losses both decrease smoothly, suggesting stable optimization. The small gap between them indicates that the model generalizes well without overfitting.

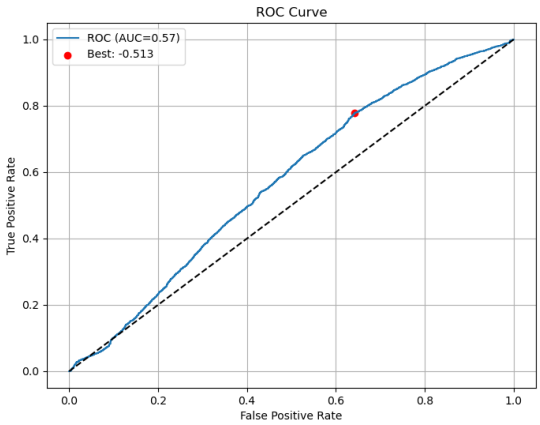

Figure 3. ROC curve of the thresholding detector where AUC = 0.57. The red dot marks the optimal threshold $\tau^\star = -0.513$.

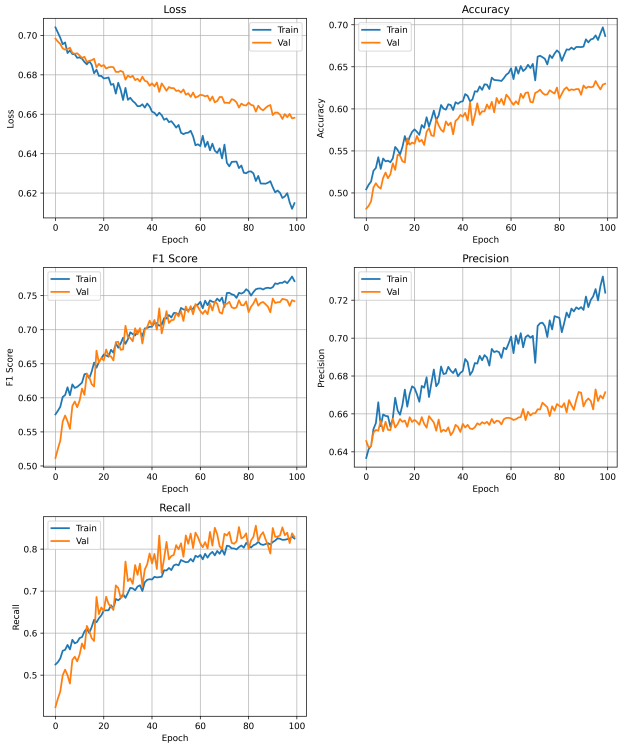

Figure 4. Training and validation metrics over 100 epochs.

Accuracy. Accuracy steadily improves, reaching 0.69 on the training set and 0.63 on validation. The close tracking of both curves demonstrates consistent learning across seen and unseen data.

Precision. Training precision increases steadily from 0.66 to 0.73, while validation precision improves more slowly and plateaus near 0.67. This suggests that the model

Metric	Thresholding	MLP Classifier
Accuracy	0.6225	0.6299
Precision	0.6875	0.6714
Recall	0.7704	0.8286
F_1 Score	0.7266	0.7433

Table 1. Comparison of performance between threshold-based alignment scoring and MLP classifier.

maintains high specificity, though with some overconfidence on training data.

Recall. Recall reaches over 0.80 on both splits, indicating strong sensitivity to positive (aligned) examples. The early gain in validation recall is particularly notable, pointing to rapid improvements in the model's ability to capture true positives.

F1 Score. Both curves converge near 0.75, indicating balanced performance between precision and recall. The strong F1 scores reflect the model's robustness to potential class imbalance.

Overall, the final results summarized in Table 1 demonstrate that the enhanced MLP architecture accurately detects audio-visual misalignment and generalizes well across validation data.

5.4. Method Comparison

While the thresholding approach offers a simple and interpretable decision rule based on Euclidean distance between audio-visual features, it may struggle in complex or borderline cases. As shown in Table 1, thresholding achieves strong recall (0.7704), indicating high sensitivity to misalignments. However, its lower precision (0.6875) suggests a higher false-positive rate, likely caused by overflagging subtly misaligned or borderline-aligned clips.

In contrast, the MLP classifier yields slightly better overall performance, with a higher F_1 score (0.7433 vs. 0.7266) and improved recall (0.8286), reflecting better detection of diverse misalignment patterns. Although its precision (0.6714) is marginally lower than that of thresholding, the overall balance and adaptability of the MLP offer a net gain in discriminative power.

These results demonstrate that learning-based methods can capture more complex feature relationships. However, this comes at the cost of increased training and inference complexity. For applications with strict latency or resource constraints, the thresholding method remains a viable and computationally efficient alternative.

6. Conclusion

Our work demonstrates that cross-modal anomaly detection through audio-visual alignment is both feasible and practically valuable. The proposed methods—threshold-based detection and neural classification—offer complementary advantages for different deployment scenarios. The thresholding approach provides a lightweight solution suitable for real-time applications, while the MLP classifier delivers higher accuracy at the cost of greater computational requirements. Both methods achieve promising results on synthetic misalignments, with F1 scores exceeding 0.72.

6.1. Limitations

Reliance on synthetically generated misalignments. Although our transformations (mute, noise, distort, time shift) provide a controlled and diverse testbed, they may not fully reflect the complexity and variability of real-world anomalies. For example, real synchronization errors can involve multiple overlapping distortions, gradual drifts, or device-specific encoding artifacts not represented in our synthetic pipeline. Moreover, training solely on synthetic data risks overfitting to the artifact patterns of our own generation pipeline, potentially limiting generalization to unseen sources.

Skewed segment duration distribution. Our synthetic segment generation (via random sub-segmentation with a 1-second minimum) resulted in a strong bias toward short clips, particularly in the 1–2 second range. This imbalance may introduce unintended biases during training, such as length-sensitive thresholding behavior or MLP overfitting to short-term patterns. More careful sampling or duration-aware filtering during preprocessing could help alleviate this issue.

6.2. Directions for Future Research/Improvement

Misalignment-Specific Performance Analysis. Further analysis of how different misalignment types (e.g., mute, noise, distort, time shift) affect model behavior could inform targeted improvements to both feature extraction and classification techniques.

Balanced segment durations. Our current dataset skews heavily toward short clips due to preprocessing choices. Constructing a more balanced duration distribution may improve generalization and reduce biases in both thresholding and neural methods.

Architecture and hyperparameter tuning. Incorporating temporal modeling techniques such as recurrent architectures or attention mechanisms to MLP model in order to better capture long-range dependencies in audio-visual synchronization.

Multimodal Fusion Enhancements. Beyond simple concatenation, future work could explore advanced fusion techniques—such as cross-modal transformers—to better model interactions between audio and visual modalities.

References

[1] Ffmpeg. https://ffmpeg.org/. Accessed: 2025-05-13. 2

[2] Zrar Kh. Abdul and Abdulbasit K. Al-Talabani. Mel frequency cepstral coefficient and its applications: A review. *IEEE Access*, 10:122136–122158, 2022. 3

[3] N. Ahmed, T. Natarajan, and K.R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974. 3

[4] J. B. Allen and L. R. Rabiner. Short time spectral analysis, synthesis, and modification by discrete fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(3):235–238, 1977. 3

[5] Gary Bradski. Opencv library. https://opencv.org, 2000. Version 3.4. See also https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html. 3

[6] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980. 3

[7] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In Josef Bigun and Tomas Gustavsson, editors, *Image Analysis*, pages 363–370, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. 3

[8] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. ROC Analysis in Pattern Recognition. 4

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 5

[10] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. 5

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. 5

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 5

[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. 5

[14] Marius-Constantin Popescu, Valentina Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, 8, 07 2009. 5

[15] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017. 5

[16] Enrique F. Schisterman, Neil J. Perkins, Aiyi Liu, and Howard Bondell. Optimal cut-point and its corresponding youden index to discriminate individuals using pooled blood samples. *Epidemiology*, 16(1):73–81, 2005. 5

[17] S. S. Stevens, J. Volkmann, and E. B. Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 01 1937. 3

[18] Yapeng Tian, Jing Shi, Bochen Li, Zhiyao Duan, and Chenliang Xu. Audio-visual event localization in unconstrained videos. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*. Springer, 2018. 1

[19] Hao Zhu, Man-Di Luo, Rui Wang, Ai-Hua Zheng, and Ran He. Deep audio-visual learning: A survey. *International Journal of Automation and Computing*, 18(3):351–376, 2021. 1

[20] Şeref Kerem Çorbacıoğlu and Gökhan Aksel. Receiver operating characteristic curve analysis in diagnostic accuracy studies: A guide to interpreting the area under the curve value. *Turkish Journal of Emergency Medicine*, 23(4):195–198, October 2023. 6