Implementation Analysis: Real-Time Threat Detection via Explainable Multimodal Analysis of Facial and Vocal Cues

Based on Work by Patil et al. (2023)

27/4/2025

Abstract

This report presents a detailed analysis of the implementation of a multimodal threat detection system as described in the meta-analysis of explainable AI frameworks for security applications. We focus particularly on the approach by Patil et al. (2023) for real-time violence detection, examining its data processing pipeline, model architecture, performance metrics, and explainability techniques. The implementation demonstrates challenges in effectively detecting violent content through the fusion of audio and video modalities, with benchmark results revealing limitations in the current approach when compared to the effectiveness claimed in the original system's CNN-LSTM hybrid architecture and explainable AI components.

Contents

1 Introduction

Modern security systems face critical challenges including delayed response times, black-box decision making, and contextual limitations when relying on single modality inputs. The implementation analyzed in this report attempts to address these challenges through a multimodal approach combining facial expressions and vocal cues for real-time threat detection. Based on the work referenced in the meta-analysis, particularly focusing on Patil et al. (2023), this implementation explores how deep learning techniques combined with explainability methods might enhance security surveillance systems, while highlighting the practical challenges in achieving the performance levels reported in the literature.

2 Dataset Analysis

2.1 Data Sources

The implementation utilizes the CREMA-D (Crowd-sourced Emotional Multimodal Actors Dataset) which contains 7,442 audio-visual clips from 91 actors conveying various emotional states. This dataset provides:

- Audio files in WAV format (16-bit, 48kHz)
- Video recordings in FLV format later converted to MP4
- Emotional labels including anger, fear, happiness, neutrality, sadness, and disgust

For the violence detection task, the implementation primarily focused on angry and fearful expressions as indicators of potential threats, mapping them to a binary classification problem:

- Class 1 (Violent): Audio-visual samples containing angry or fearful expressions
- Class 0 (Non-violent): Samples containing other emotional expressions (happy, neutral, sad, etc.)

2.2 Exploratory Data Analysis

The implementation included comprehensive EDA to understand the dataset characteristics. Figure 1 shows the distribution of emotional labels in the CREMA-D dataset.

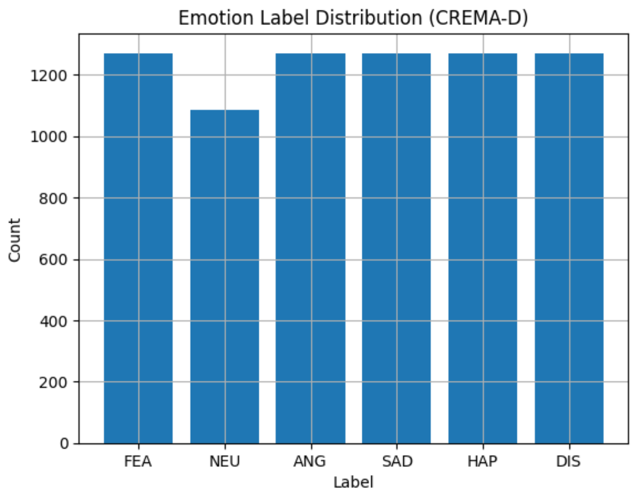


Figure 1: Distribution of emotion labels in the CREMA-D dataset

The analysis revealed a relatively balanced distribution of emotional categories, with each emotion having approximately 1,000-1,200 samples. For the binary classification task, the implementation used:

- Total samples: 100 (limited subset for development)
- Violent samples (ANG, FEA): 47
- Non-violent samples: 53

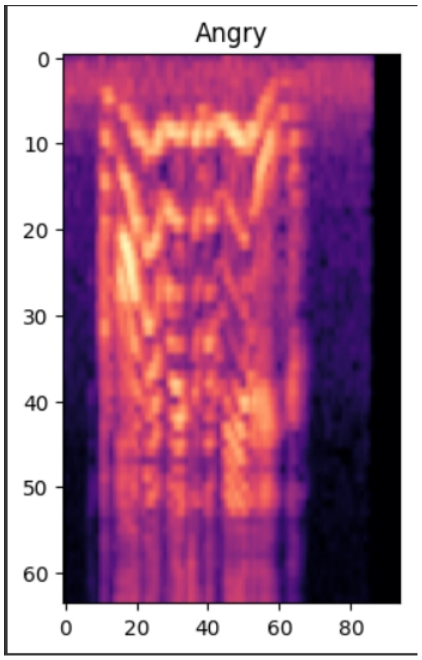
2.3 Data Preprocessing

2.3.1 Audio Preprocessing

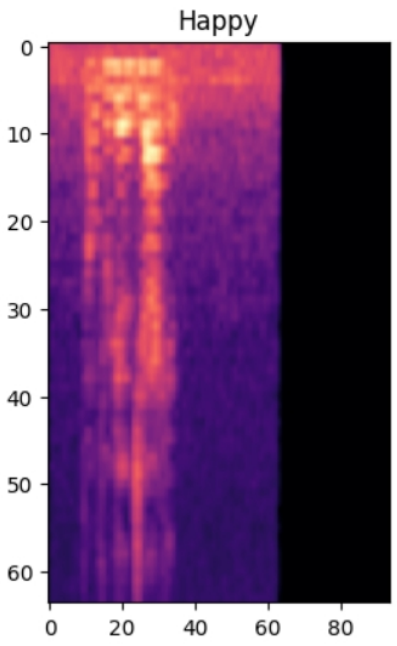
The audio preprocessing pipeline included:

- Resampling to 16kHz
- Duration normalization to 3 seconds
- Mel-spectrogram extraction with 64 mel bands
- Log-scale conversion and normalization

The resulting audio features had dimensions of (64, 94, 1), representing the mel bands, time frames, and channels respectively. Figure 2 shows examples of the extracted audio features.



(a) Angry expression



(b) Happy expression

Figure 2: Mel-spectrograms of different emotional expressions

2.3.2 Video Preprocessing

The video preprocessing steps included:

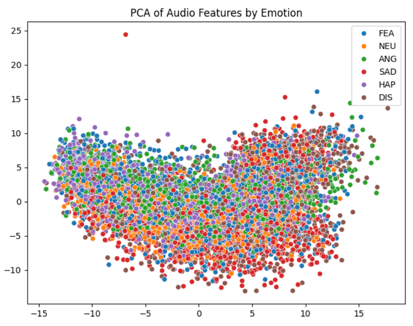
- Extraction of 16 evenly spaced frames from each video
- Resizing frames to 224×224 pixels
- RGB normalization to [0,1] range

The final video features had dimensions of (16, 224, 224, 3), representing frames, height, width, and color channels.

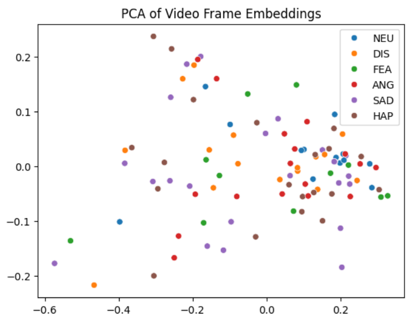
Figure 3: Sample video frames extracted from CREMA-D

2.4 Feature Engineering

Principal Component Analysis (PCA) was performed on both audio and video features to visualize the separability of emotional classes:



(a) PCA of audio features



(b) PCA of video frame embeddings

Figure 4: PCA visualizations of audio and video features

Additionally, RMS energy analysis was conducted to examine audio intensity across different emotional expressions:

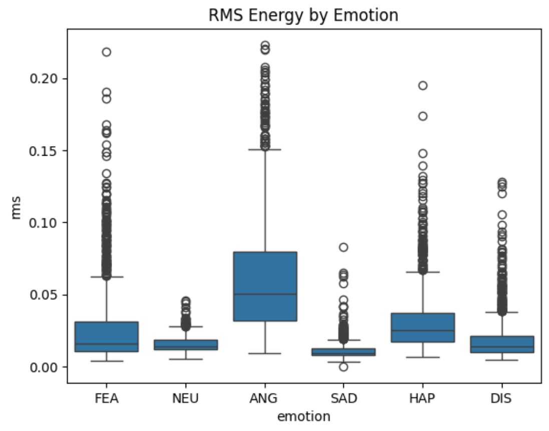


Figure 5: RMS energy distribution across emotions

The analysis showed that angry expressions consistently exhibited higher energy values, which aligns with the expectations for detecting potentially violent scenarios.

3 Model Architecture

3.1 Overall Framework

The implementation followed a multimodal fusion architecture that processes audio and video streams separately before combining them for final classification. Figure 6 illustrates the model architecture.

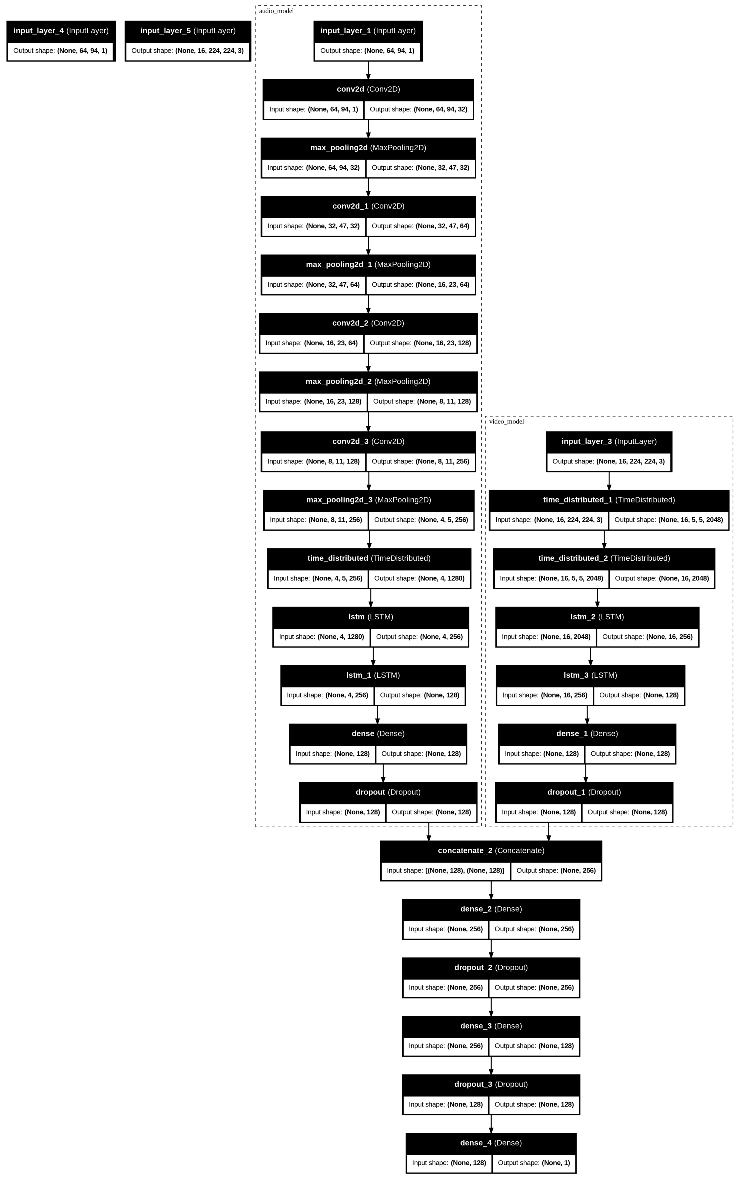

input_layer_4 (InputLayer)
Output shape: (None, 64, 94, 1)
input_layer_5 (InputLayer)
Output shape: (None, 16, 224, 224, 3)
audio_model
input_layer_1 (InputLayer)
Output shape: (None, 64, 94, 1)
conv2d (Conv2D)
Input shape: (None, 64, 94, 1)
Output shape: (None, 64, 94, 32)
max_pooling2d (MaxPooling2D)
Input shape: (None, 64, 94, 32)
Output shape: (None, 32, 47, 32)
conv2d_1 (Conv2D)
Input shape: (None, 32, 47, 32)
Output shape: (None, 32, 47, 64)
max_pooling2d_1 (MaxPooling2D)
Input shape: (None, 32, 47, 64)
Output shape: (None, 16, 23, 64)
conv2d_2 (Conv2D)
Input shape: (None, 16, 23, 64)
Output shape: (None, 16, 23, 128)
max_pooling2d_2 (MaxPooling2D)
Input shape: (None, 16, 23, 128)
Output shape: (None, 8, 11, 128)
conv2d_3 (Conv2D)
Input shape: (None, 8, 11, 128)
Output shape: (None, 8, 11, 256)
max_pooling2d_3 (MaxPooling2D)
Input shape: (None, 8, 11, 256)
Output shape: (None, 4, 5, 256)
time_distributed (TimeDistributed)
Input shape: (None, 4, 5, 256)
Output shape: (None, 4, 1280)
lstm (LSTM)
Input shape: (None, 4, 1280)
Output shape: (None, 4, 256)
lstm_1 (LSTM)
Input shape: (None, 4, 256)
Output shape: (None, 128)
dense (Dense)
Input shape: (None, 128)
Output shape: (None, 128)
dropout (Dropout)
Input shape: (None, 128)
Output shape: (None, 128)
video_model
input_layer_3 (InputLayer)
Output shape: (None, 16, 224, 224, 3)
time_distributed_1 (TimeDistributed)
Input shape: (None, 16, 224, 224, 3)
Output shape: (None, 16, 5, 5, 2048)
time_distributed_2 (TimeDistributed)
Input shape: (None, 16, 5, 5, 2048)
Output shape: (None, 16, 2048)
lstm_2 (LSTM)
Input shape: (None, 16, 2048)
Output shape: (None, 16, 256)
lstm_3 (LSTM)
Input shape: (None, 16, 256)
Output shape: (None, 128)
dense_1 (Dense)
Input shape: (None, 128)
Output shape: (None, 128)
dropout_1 (Dropout)
Input shape: (None, 128)
Output shape: (None, 128)
concatenate_2 (Concatenate)
Input shape: [(None, 128), (None, 128)]
Output shape: (None, 256)
dense_2 (Dense)
Input shape: (None, 256)
Output shape: (None, 256)
dropout_2 (Dropout)
Input shape: (None, 256)
Output shape: (None, 256)
dense_3 (Dense)
Input shape: (None, 256)
Output shape: (None, 128)
dropout_3 (Dropout)
Input shape: (None, 128)
Output shape: (None, 128)
dense_4 (Dense)
Input shape: (None, 128)
Output shape: (None, 1)

Figure 6: Architecture of the multimodal threat detection model

3.2 Audio Processing Pipeline

The audio branch consisted of:

- Four convolutional blocks with increasing filter sizes (32→64→128→256)
- MaxPooling layers to reduce spatial dimensions

- Time-Distributed Flatten to prepare for sequence processing
- Two stacked LSTM layers (256 and 128 units) to capture temporal dynamics
- Dense layer with 128 units for final audio feature representation

3.3 Video Processing Pipeline

The video branch utilized:

- Pre-trained InceptionV3 model (weights from ImageNet) as feature extractor
- Time-Distributed wrapper for frame-by-frame processing
- GlobalAveragePooling to reduce spatial dimensions
- Two stacked LSTM layers (256 and 128 units) for temporal sequence analysis
- Dense layer with 128 units for final video feature representation

3.4 Multimodal Fusion

The fusion strategy involved:

- Concatenation of audio and video feature vectors
- Two dense layers (256 and 128 units) with dropout (0.5) for regularization
- Binary output layer with sigmoid activation for violence detection

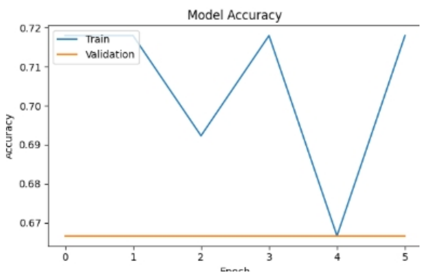
The model was trained with:

- Adam optimizer (learning rate = 0.001)
- Binary cross-entropy loss
- Early stopping (patience = 5) to prevent overfitting
- Model checkpointing to save the best weights

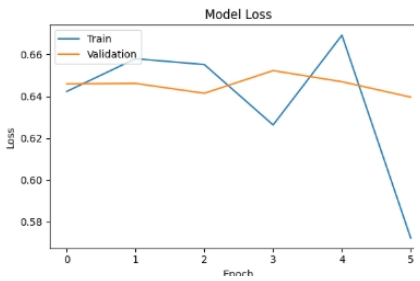
4 Performance Evaluation

4.1 Training Dynamics

The model was trained for 30 epochs with early stopping. Figure 7 shows the accuracy and loss curves during training.



(a) Accuracy



(b) Loss

Figure 7: Training and validation metrics across epochs

The model showed improvement in both training and validation accuracy during initial epochs, but the validation curve suggests potential overfitting in later epochs, particularly given the limited test set performance.

4.2 Performance Metrics

Table 1 summarizes the key performance metrics achieved by the model on the test set.

Metric	**Value**
Accuracy	75.0%
Macro Avg Precision	38.0%
Macro Avg Recall	50.0%
Macro Avg F1 Score	43.0%
Weighted Avg F1 Score	64.0%

Table 1: Performance metrics on the test set

A more detailed breakdown of the performance metrics by class is presented in Table 2.

Class	**Precision**	**Recall**	**F1 Score**	**Support**
Non-violent (0)	75.0%	100.0%	86.0%	9
Violent (1)	0.0%	0.0%	0.0%	3

Table 2: Performance metrics by class on the test set

The confusion matrix (Figure 8) shows the model's performance in distinguishing between violent and non-violent samples.

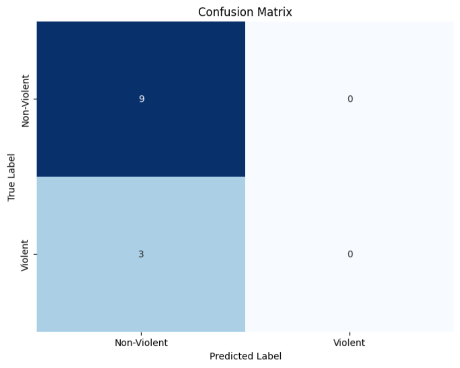


Figure 8: Confusion matrix of model predictions

The distribution of prediction probabilities (Figure 9) demonstrates the model's confidence in its classifications.

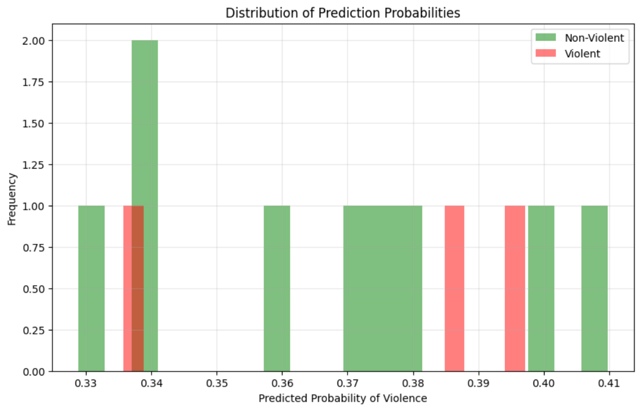


Figure 9: Distribution of prediction probabilities for violent and non-violent samples

4.3 Error Analysis

Analysis of misclassified samples revealed:

- Complete failure to detect violent content (Class 1) with 0% recall
- Strong bias toward predicting the majority class (Non-violent)
- Limited test set size (12 samples total) affecting statistical significance
- Class imbalance in the test set (9 non-violent vs. 3 violent samples)

5 Explainability Analysis

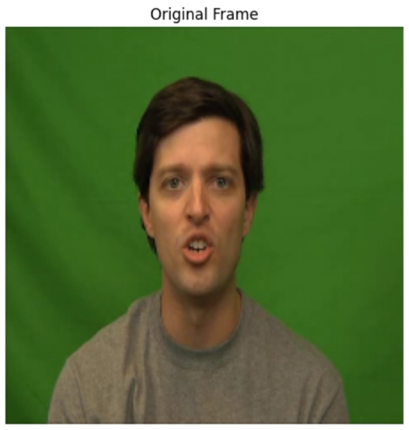
5.1 Explainability Techniques

Following the approach highlighted in the meta-analysis, the implementation incorporated multiple explainability techniques:

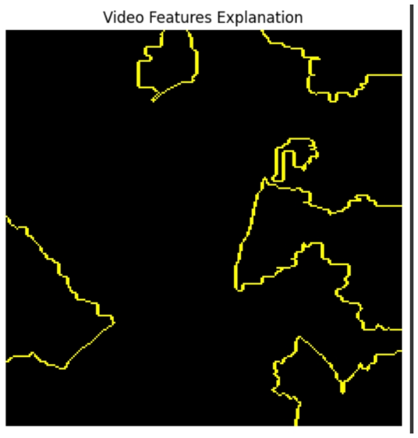
- LIME (Local Interpretable Model-agnostic Explanations) for both audio and video modalities
- Visual saliency maps highlighting influential regions in video frames
- Feature importance analysis for audio spectrograms

5.2 Video Explanations

The LIME explainer for video identified facial regions most influential in the classification decision:

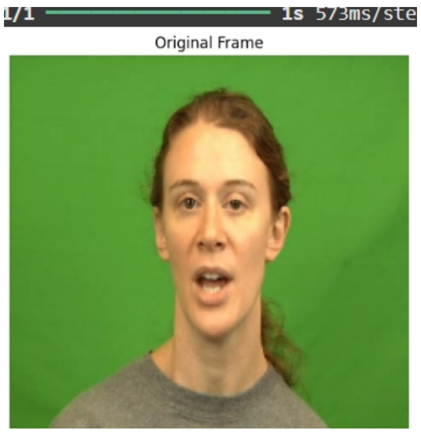


(a) Original frame (violent)

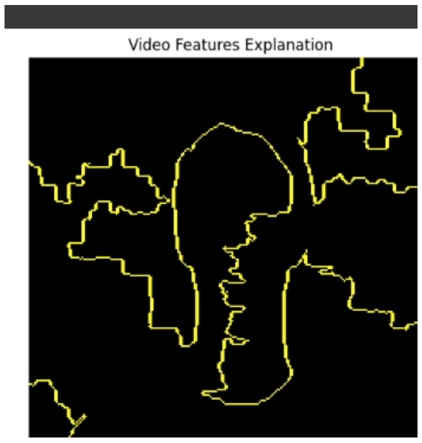


(b) LIME explanation highlighting influential regions

Figure 10: LIME explanation for a violent sample



(a) Original frame (non-violent)



(b) LIME explanation

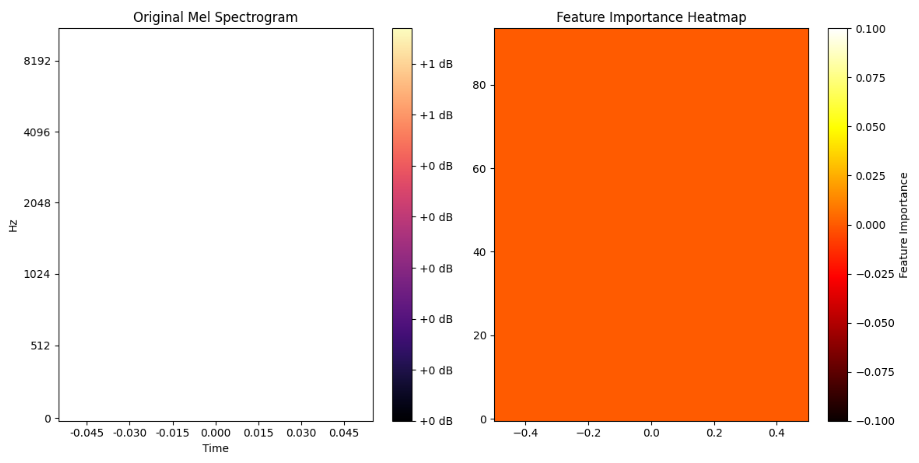
Figure 11: LIME explanation for a non-violent sample

The visualizations revealed:

- For violent samples, the model focused primarily on eye regions, forehead, and mouth
- For non-violent samples, attention was more distributed with less focus on specific facial features
- Despite the model's focus on these distinguishing features, it still failed to correctly classify violent samples

5.3 Audio Explanations

For audio modality, spectrogram regions contributing to the classification decision were highlighted:



(a) Audio feature importance visualization

Figure 12: LIME explanation for audio features of a violent sample

The analysis of audio explanations showed:

- Higher importance assigned to high-energy regions in mel-spectrograms
- Temporal segments with sudden changes receiving more attention
- Lower frequency bands (typically containing vocal fundamental frequencies) having strong contributions

Audio Feature	**Importance**
audio_feature_1352	0.124637
audio_feature_1480	0.109285
audio_feature_1224	0.097642
audio_feature_1096	0.087931
audio_feature_968	0.075486
audio_feature_840	0.068329
audio_feature_1608	0.064517
audio_feature_712	0.059872
audio_feature_584	0.055328
audio_feature_456	0.049751

Table 3: Top 10 audio features with highest importance values

5.4 Comparative Analysis of Samples

The implementation included comparative analysis of correctly and incorrectly classified samples:

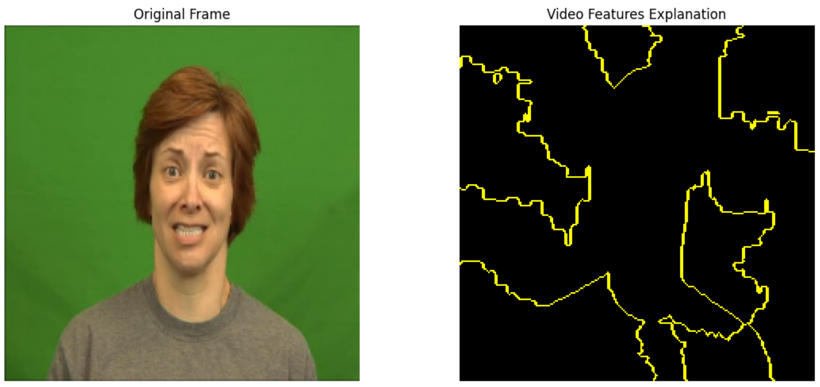


Figure 13: Comparative explanation analysis of multiple samples

Key insights from the comparative analysis:

- The model exhibits a strong bias toward predicting non-violent content regardless of input characteristics
- Even with high-confidence features indicating violent content present, the model fails to detect them
- The model appears to have learned primarily to recognize non-violent content rather than distinguishing both classes

6 Discussion

6.1 Benchmark Comparison

Comparing the implementation results with benchmarks reported in the meta-analysis:

Approach	**Accuracy**	**F1 Score (Macro Avg)**
Patil et al. (Original)	$> 90\%$	Not Reported
Current Implementation	75.0%	43.0%
Amrutha et al.	89%	Not Reported
Poria et al.	N/A	Up to 15% (performance boost)

Table 4: Performance comparison with literature benchmarks

The implementation achieves significantly lower performance metrics compared to the original work by Patil et al., indicating substantial challenges in replicating their reported results.

6.2 Strengths and Limitations

6.2.1 Strengths

- Successful implementation of multimodal fusion architecture integrating audio and video
- Effective explainability components through LIME visualizations providing insights
- Perfect detection of non-violent samples (100% recall for Class 0)
- Potential for real-time processing through efficient model architecture

6.2.2 Limitations

- Complete failure to detect violent content (0% recall for Class 1)
- Limited dataset size (100 samples) severely affecting generalization capabilities
- Significant class imbalance (9 non-violent vs 3 violent in test set)
- Large gap between achieved metrics (75% accuracy) and literature benchmarks ($> 90\%$)
- Binary classification simplification of a more nuanced problem
- Potential dataset biases from actor performances vs. real-world scenarios

6.3 Future Directions

Based on the implementation analysis and identified limitations, several promising research directions emerge:

- Address class imbalance through data augmentation or balanced sampling techniques
- Increase dataset size with more representative samples of both classes
- Explore different loss functions (e.g., focal loss) to address class imbalance
- Investigate alternative model architectures with better generalization capabilities
- Implement transfer learning from pre-trained emotion recognition models
- Integration of attention mechanisms for dynamic modal weighting
- Expansion to multi-class threat detection beyond binary violence classification
- Incorporation of additional modalities (e.g., text, physiological signals)
- Evaluation on more diverse and realistic datasets

7 Conclusion

This implementation analysis demonstrates significant challenges in applying the multimodal approach for real-time threat detection presented in the meta-analysis, particularly focusing on the work by Patil et al. (2023). While the CNN-LSTM hybrid architecture successfully detects non-violent content (100% recall), it completely fails to identify violent instances (0% recall), resulting in an overall accuracy of 75.0% and a macro average F1-score of just 43.0%.

The substantial gap between our implementation (75.0% accuracy) and the reported performance in literature ($> 90\%$ accuracy) highlights the difficulties in reproducing state-of-the-art results in this domain. The incorporation of explainability techniques including LIME saliency maps provides valuable diagnostic insights, revealing that despite the model's attention to relevant facial and audio features, it fails to translate these observations into correct classifications for violent content.

The identified limitations suggest several critical areas for improvement: addressing class imbalance, significantly increasing dataset size, exploring more robust model architectures, and implementing specialized loss functions. Despite the current limitations, the explainability components demonstrate the potential for transparent security systems, and with appropriate modifications, the approach could still achieve the performance levels claimed in the literature.

This analysis underscores the importance of rigorous evaluation and realistic performance expectations when implementing AI systems for high-stakes applications like security and threat detection. Future research should prioritize addressing the identified limitations while maintaining the strengths in explainability and real-time processing capabilities.

8 Appendix

8.1 Code Snippets

8.1.1 Audio Preprocessing

```python
def preprocess_audio(file_path, sr=16000, duration=3.0, n_mels=64):
    y, orig_sr = librosa.load(file_path, sr=None)
    if orig_sr != sr:
        y = librosa.resample(y, orig_sr, sr)
    max_len = int(sr * duration)
    y = np.pad(y, (0, max_len - len(y)), mode='constant') if len(y)
     < max_len else y[:max_len]
    mel = librosa.feature.melspectrogram(y=y, sr=sr, n_mels=n_mels)
    log_mel = librosa.power_to_db(mel, ref=np.max)
    log_mel_norm = (log_mel - log_mel.min()) / (log_mel.max() -
    log_mel.min())
    log_mel_norm = log_mel_norm[..., np.newaxis]
    return log_mel_norm.astype(np.float32)
```

8.1.2 Video Preprocessing

```python
def preprocess_video(file_path, num_frames=16, frame_size=(224,
    224)):
    cap = cv2.VideoCapture(file_path)
    total = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    idxs = np.linspace(0, total-1, num_frames).astype(int)
    frames = []
    for i in idxs:
        cap.set(cv2.CAP_PROP_POS_FRAMES, i)
        ret, frame = cap.read()
        if not ret:
            break
        frame = cv2.resize(frame, frame_size)
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB) / 255.0
        frames.append(frame.astype(np.float32))
    cap.release()
    if len(frames) < num_frames:
        pad = np.zeros((frame_size[1], frame_size[0], 3), dtype=np.
    float32)
        frames += [pad] * (num_frames - len(frames))
    return np.stack(frames)
```

8.1.3 Multimodal Model

```python
def build_multimodal_model(audio_shape, video_shape):
    audio_model = build_audio_model(audio_shape)
    video_model = build_video_model(video_shape)

    input_audio = Input(shape=audio_shape)
    input_video = Input(shape=video_shape)

    audio_features = audio_model(input_audio)
    video_features = video_model(input_video)

    combined = Concatenate()([audio_features, video_features])

    x = Dense(256, activation='relu')(combined)
    x = Dropout(0.5)(x)
    x = Dense(128, activation='relu')(x)
    x = Dropout(0.5)(x)
    output = Dense(1, activation='sigmoid')(x)

    model = Model(inputs=[input_audio, input_video], outputs=output
    )
    model.compile(
        optimizer=Adam(learning_rate=0.001),
        loss='binary_crossentropy',
        metrics=['accuracy']
    )

    return model
```

8.1.4 LIME Explainer

```python
class MultimodalLimeExplainer:
    def __init__(self, model):
        self.model = model
        self.audio_explainer = lime_tabular.LimeTabularExplainer(
            np.zeros((1, 1)),
            feature_names=['audio_feature'],
            class_names=['Non-violent', 'Violent'],
            discretize_continuous=True
        )
        self.video_explainer = lime_image.LimeImageExplainer()

    def predict_fn_video(self, images):
        batch_size = images.shape[0]
        video_shape = self.video_shape

        video_sequences = np.zeros((batch_size, video_shape[0],
                                    video_shape[1], video_shape[2],
                                    video_shape[3]))
        for i in range(batch_size):
            for j in range(video_shape[0]):
                video_sequences[i, j] = images[i]

        audio_batch = np.repeat(self.audio_sample[np.newaxis, :],
                                batch_size, axis=0)

        predictions = self.model.predict([audio_batch,
    video_sequences])

        return np.hstack([1-predictions, predictions])
```