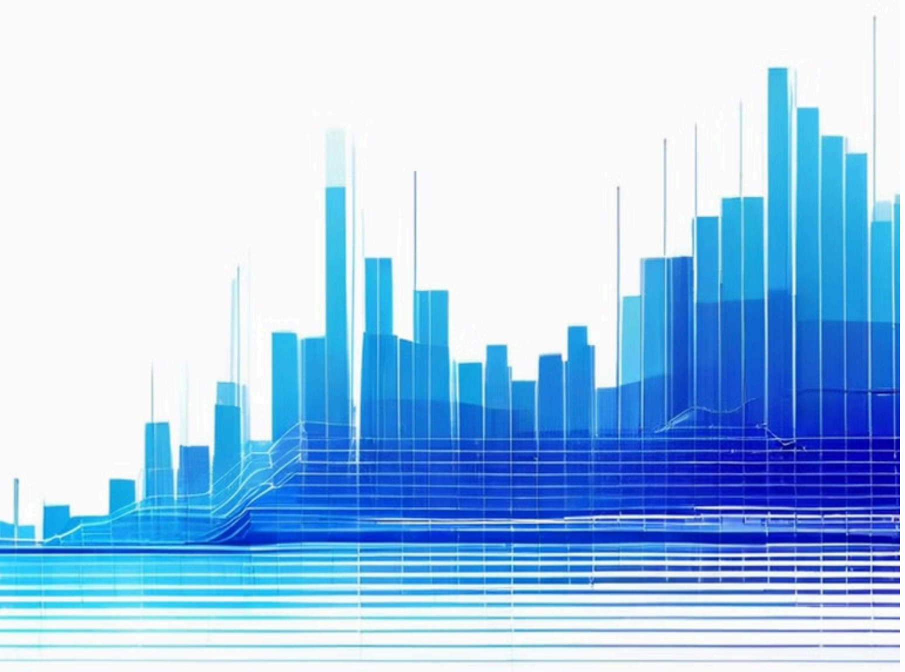
Stutter Detection using MFCC and Persody features at word ,syllable and utterence levels.



Dataset Overview: SEP-28k

Dataset Source

Released by Apple as part of the ml-stuttering-events-dataset project

Dataset Size

28,177 audio clips from podcasts, each 3 seconds long (16kHz sampling rate)

Binary Classification Task

Identifying stuttered vs. non-stuttered speech

Stutter Categories

Prolongation

Elongated sounds (e.g., "mmmmmy name")

Block

Speech stoppage/struggle with sounds

Sound Repetition

Repeating phonemes (e.g., "m-m-my")

Word Repetition

Repeating entire words (e.g., "I-I-I want")

Interjection

Filler words (e.g., "um", "uh")

Quality Control Labels (Labels we have removed)

Unsure

PoorAudioQuality

DifficultToUnderstand

NaturalPause

Music

NoSpeech

Preprocessing Overview

Download Statistics

- Total episodes attempted: 385
- Successfully downloaded: 263 (68.3%)
- Failed downloads: 122 (31.7%)

Source-specific success rates:

- High success (95-100%): HeStutters, HVSA, StutterTalk, WomenWhoStutter
- Low success (0%): StrongVoices, StutteringIsCool

Dataset Creation Process

- Filtered out clips with any quality issues
- Ensured consistent length (48,000 samples = 3 seconds)
- Created binary labels based on stutter presence

Final dataset composition:

- Total valid clips: 9,751
- Stuttered clips: 6,322 (64.8%)
- Non-stuttered clips: 3,429 (35.2%)

Feature Extraction Process and Model Building

MFCC Features Extraction for Stutter Detection

- **Definition**: Mel-Frequency Cepstral Coefficients represent the short-term power spectrum of sound
- **Importance**: Capture the vocal tract characteristics that are crucial for detecting speech abnormalities
- **Advantage**: MFCCs mimic human auditory perception by using mel scale (logarithmic perception of pitch)
- **Application**: Particularly effective for detecting stutter patterns due to their sensitivity to rapid spectral changes.

MFCC Features Extraction Process

- **Pre-emphasis**: Apply filter (coef=0.97) to amplify higher frequencies
- **Framing**: Segment audio into 25ms windows with 10ms hop length
- **Mel Filterbank**: Apply 13 filters on mel scale to mimic human hearing
- **DCT Transformation**: Convert to cepstral domain to separate vocal tract information
- **Normalization**: Scale coefficients for consistent feature ranges across recordings

Features Derived from MFCCs

- **Delta Coefficients**: Capture velocity (first derivative) of spectral change
- **Delta-Delta**: Measure acceleration (second derivative) of spectral trajectories
- **Temporal Fluctuation**: Standard deviation of frame-to-frame differences
- **Local Variability**: Measure of rapid changes in 50ms windows (stuttering indicator)
- **Transition Rate**: Rate of spectral transitions (repetition indicator)
- **MFCC Stability**: Overall stability of spectral envelope (fluency indicator)

Feature Vector Composition

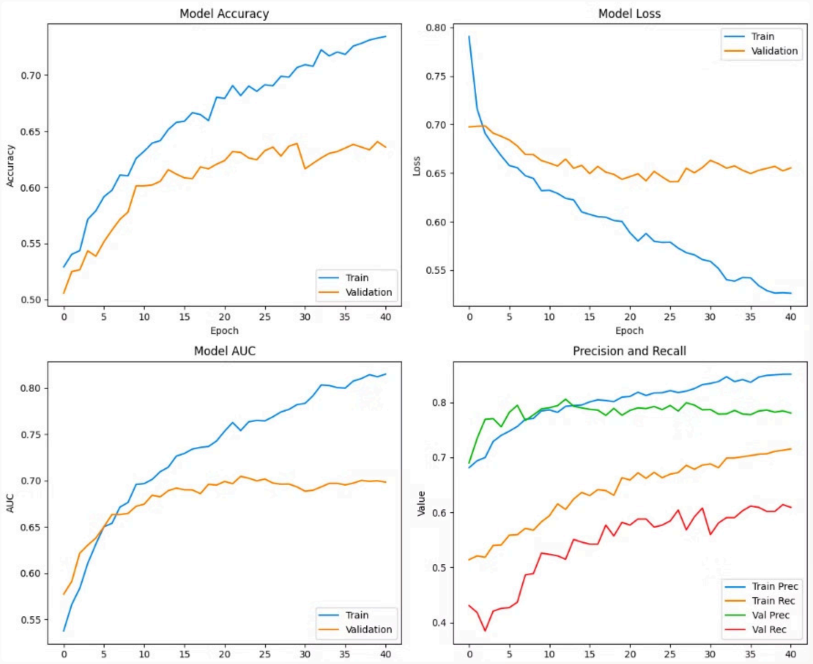
- **Basic MFCC Statistics**: Mean and standard deviation of 13 coefficients
- **Temporal Dynamics**: Delta and Delta-Delta mean values
- **Variability Metrics**: Temporal fluctuation and local variability patterns
- **Transition Features**: Overall transition rate and stability measures
- **Total Features**: 80 features per audio clip (extracted with optimized parameters)

DNN Model Architecture for Stutter Detection using mfcc features

- **Input Layer**: Dense layer with 128 neurons (**input shape = 80 features**)
- **Hidden Layers**: 64→32 neurons with ReLU activation
- **Regularization**: BatchNormalization + Dropout (0.4, 0.4, 0.3) to prevent overfitting
- **Output Layer**: Single neuron with sigmoid activation (binary classification)
- **Total Parameters**: 21,633 (84.50 KB) for efficient deployment

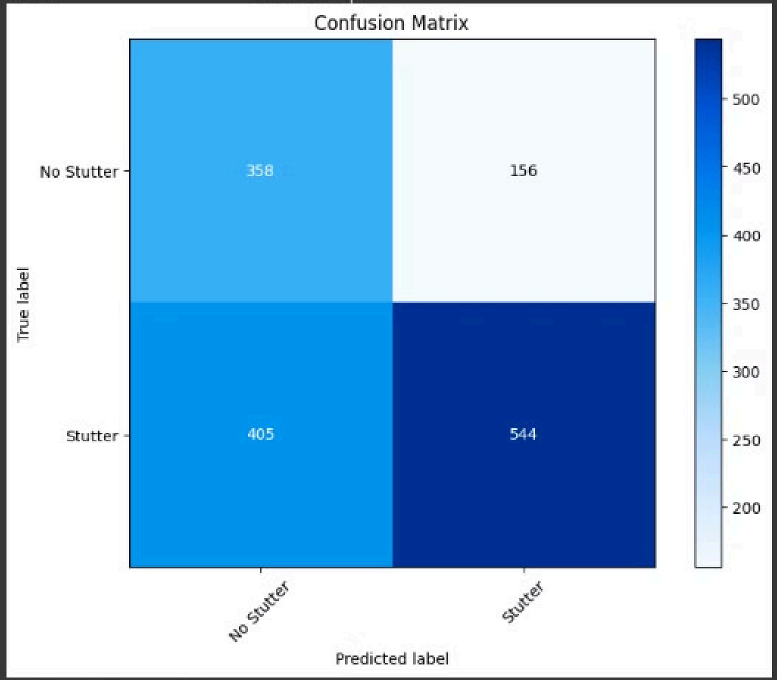
Training Process

- **Dataset Split**: 70% training, 15% validation, 15% testing
- **Class Balancing**: Weighted loss function for imbalanced stutter/non-stutter samples
- **Optimization**: Adam optimizer with binary cross-entropy loss
- **Early Stopping**: Patience=15 epochs with best weights restoration
- **Learning Rate Schedule**: ReduceLROnPlateau with factor=0.5, patience=5



Model Performance

- **Accuracy**: 61.65% on test set
- **AUC**: 0.6783 (reasonable discrimination ability)
- **Precision**: 0.7771 (high reliability of positive predictions)
- **Recall**: 0.5732 (moderate detection of actual stutters)
- **F1-Score**: 0.66 for stutter class vs 0.56 for non-stutter class



Classification Report:

	precision	recall	f1-score	support
No Stutter	0.47	0.70	0.56	514
Stutter	0.78	0.57	0.66	949
accuracy			0.62	1463
macro avg	0.62	0.63	0.61	1463
weighted avg	0.67	0.62	0.62	1463

Prosody Features (Utterance Level)

- We have extracted persody features at utterance level (full 3-second clips)

Feature extraction process

- **Pre-processing**: Pre-emphasis filtering (coef=0.97) to enhance higher frequencies
- **Pitch Analysis**: Using pYIN algorithm for robust pitch tracking (F0 extraction)
- **Energy Analysis**: Root Mean Square (RMS) analysis of amplitude envelope
- **Temporal Analysis**: Zero-crossing rate and speech rate calculation
- **Voice Quality**: Jitter, shimmer, and HNR measurements for voice stability

Prosody features that we have extracted

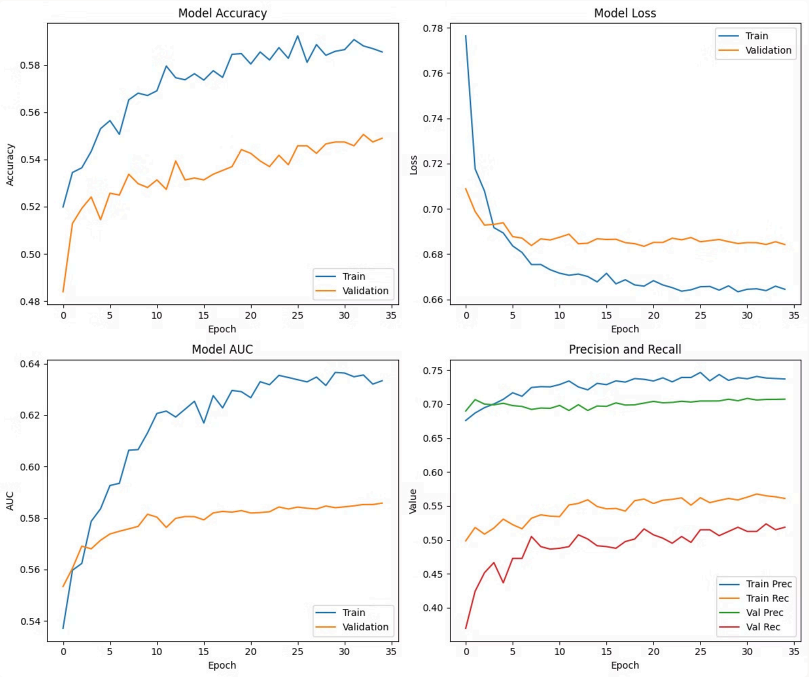
- **Pitch Features**:
  - F0_mean: Average pitch height
  - F0_std: Pitch variability
  - F0_range: Span between minimum and maximum pitch
- **Energy Features**:
  - RMS_mean: Overall loudness
  - RMS_std: Loudness variability
- **Temporal Features**:
  - ZCR: Rate of sign-changes in waveform (consonant detection)
  - Speech_rate: Density of onsets per second

Voice Quality Features

- **Jitter**: Cycle-to-cycle pitch variation (>0.015 indicates stutter)
  - Formula: Average absolute difference between consecutive F0 periods
- **Shimmer**: Cycle-to-cycle amplitude variation (>0.035 indicates stutter)
  - Formula: Average absolute difference between consecutive amplitude peaks
- **HNR Estimate**: Harmonics-to-Noise Ratio (lower values indicate roughness)
  - Normal speech >18dB, stuttered speech often <12dB

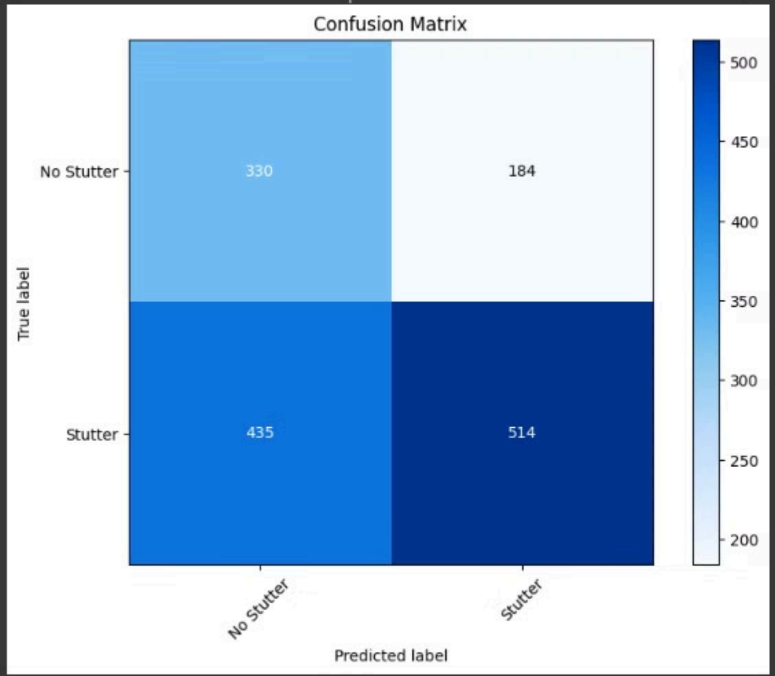
Model Architecture for Stutter Detection using prosody features

- **Input Layer**: 64 neurons receiving 13 prosodic features
- **Hidden Layers**: 32→16 neurons with ReLU activation
- **Regularization**: BatchNormalization + Dropout (0.3, 0.3, 0.2)
- **Output Layer**: Single neuron with sigmoid activation
- **Optimization**: Adam optimizer with binary cross-entropy loss
- **Training Strategy**: Early stopping (patience=15) and LR reduction



Model Performance

- **Accuracy**: 57.69% on test set
- **AUC**: 0.6322 (moderate discrimination ability)
- **Precision**: 0.7364 (74% of predicted stutters are correct)
- **Recall**: 0.5416 (54% of actual stutters detected)
- **Class-specific Performance**:
  - Stutter class: F1-score = 0.62
  - No Stutter class: F1-score = 0.52




```
Classification Report:
              precision    recall  f1-score   support

  No Stutter       0.43      0.64      0.52       514
     Stutter       0.74      0.54      0.62       949

    accuracy                           0.58      1463
   macro avg       0.58      0.59      0.57      1463
weighted avg       0.63      0.58      0.59      1463
```

Word-Level Features

Boundary Detection

- **Energy-Based Segmentation Method**:
  - Calculate RMS energy contour (hop length = 512)
  - Apply adaptive threshold: mean(energy) + 0.5 × std(energy)
  - Identify silent regions (energy < threshold)
  - Filter for minimum silence duration (≥ 150ms)
  - Convert to speech segments between silences
  - Keep only segments > 100ms

Word-Level Analysis

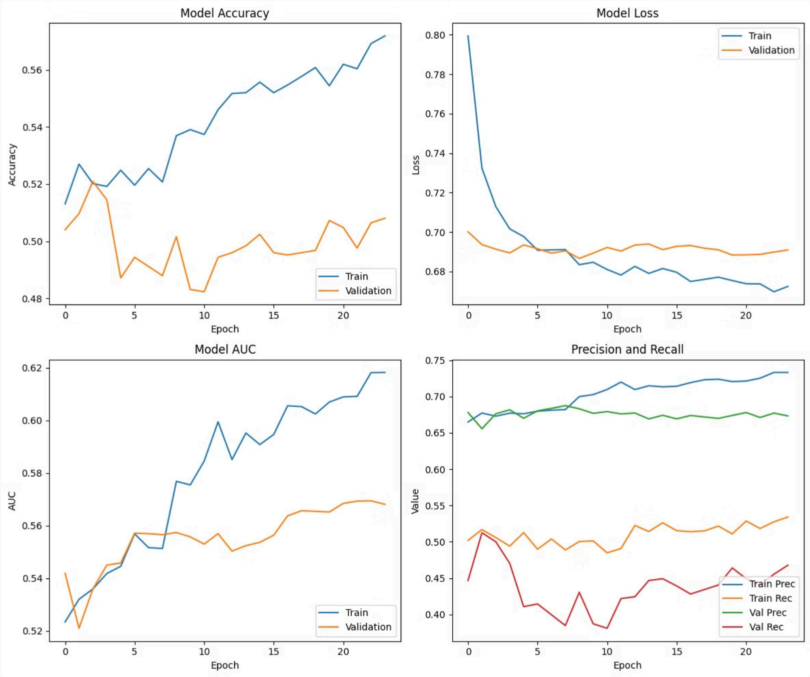
- **Acoustic Features**:
  - MFCC coefficients (mean, std)
  - Delta & Delta-Delta MFCCs
  - Transition rate & MFCC stability
  - Temporal fluctuation & local variability
- **Prosodic Features**:
  - F0 (pitch) statistics
  - Jitter & Shimmer (voice stability)
  - Harmonics-to-Noise Ratio
  - Zero-crossing rate

Model Architecture

- **Input Layer**:
  - Dense (128 neurons, ReLU)
  - BatchNormalization + Dropout (0.4)
- **Hidden Layers**:
  - Dense (64) → BatchNorm → Dropout (0.4)
  - Dense (32) → BatchNorm → Dropout (0.3)
  - Dense (16) → BatchNorm → Dropout (0.2)
- **Output Layer**:
  - Dense (1, sigmoid activation)
  - Binary classification (stutter/no-stutter)

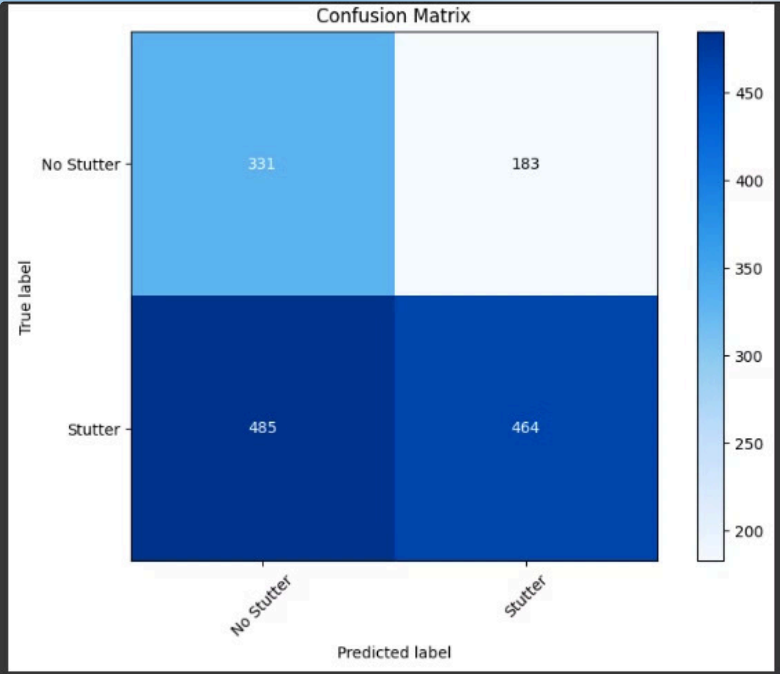
Training Process

- **Optimization**:
  - Adam optimizer with binary cross-entropy loss
  - Class weights for imbalanced dataset
  - Batch size: 32
- **Training Strategy**:
  - Early stopping (patience=15)
  - Learning rate reduction (factor=0.5)
  - 70/15/15 train-validation-test split



Model Performance

- **Classification Metrics**:
  - Accuracy: 54.34%
  - AUC: 0.6073
  - Precision: 0.7172 (good positive prediction reliability)
  - Recall: 0.4889 (moderate stutter detection rate)
- **Class Performance**:
  - Stutter class: P=0.72, R=0.49, F1=0.58
  - No Stutter class: P=0.41, R=0.64, F1=0.50



Classification Report:

	precision	recall	f1-score	support
No Stutter	0.41	0.64	0.50	514
Stutter	0.72	0.49	0.58	949
accuracy			0.54	1463
macro avg	0.56	0.57	0.54	1463
weighted avg	0.61	0.54	0.55	1463

Syllable-Level Features

Syllable Segmentation Approach

- **Group Delay Function (GDF)**: A signal processing technique for accurate syllable boundary detection.
- **Multi-band Processing**: Analyzes three sub-band signals to improve detection accuracy
- **Advantages**: More precise than energy-based methods for detecting stutter patterns

Syllable Boundary Detection Process

- **Step 1**: Create three filtered versions of the speech signal
    - Original signal (full spectrum)
    - Low-pass filtered signal (removes fricatives)
    - Band-pass filtered signal (attenuates semivowels)
- **Step 2**: Compute energy contours for each sub-band
- **Step 3**: Apply Group Delay Function transformation
- **Step 4**: Detect peaks in GDF (syllable boundaries)
- **Step 5**: Combine evidence from all sub-bands

Model Architecture

Input Layer:

64 neurons receiving syllable features

Hidden Layers:

32 neurons with ReLU + BatchNorm + Dropout (0.3)

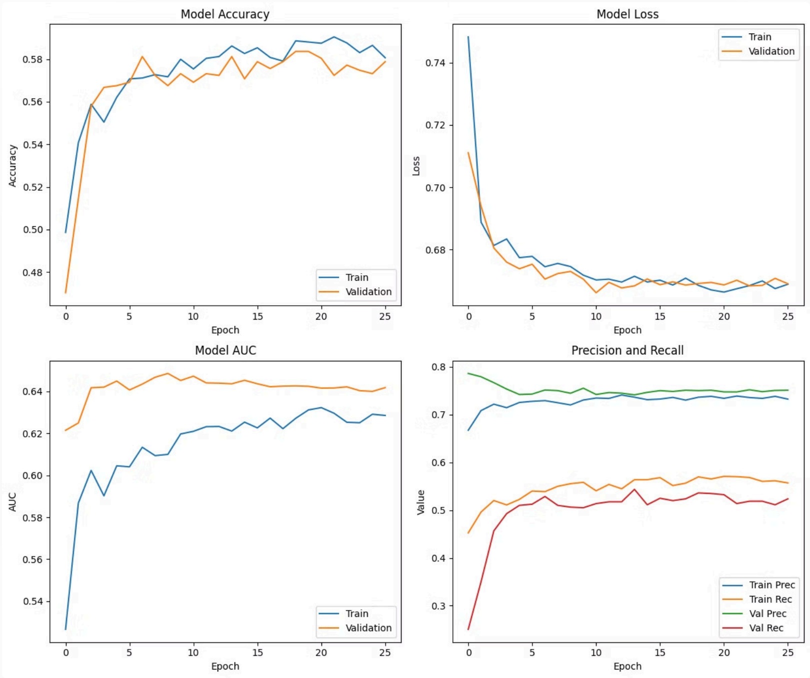
16 neurons with ReLU + BatchNorm + Dropout (0.2)

Output Layer:

Single neuron with sigmoid activation

Total Parameters: 4,417 (17.25 KB)

Trainable Parameters: 4,193 (16.38 KB)

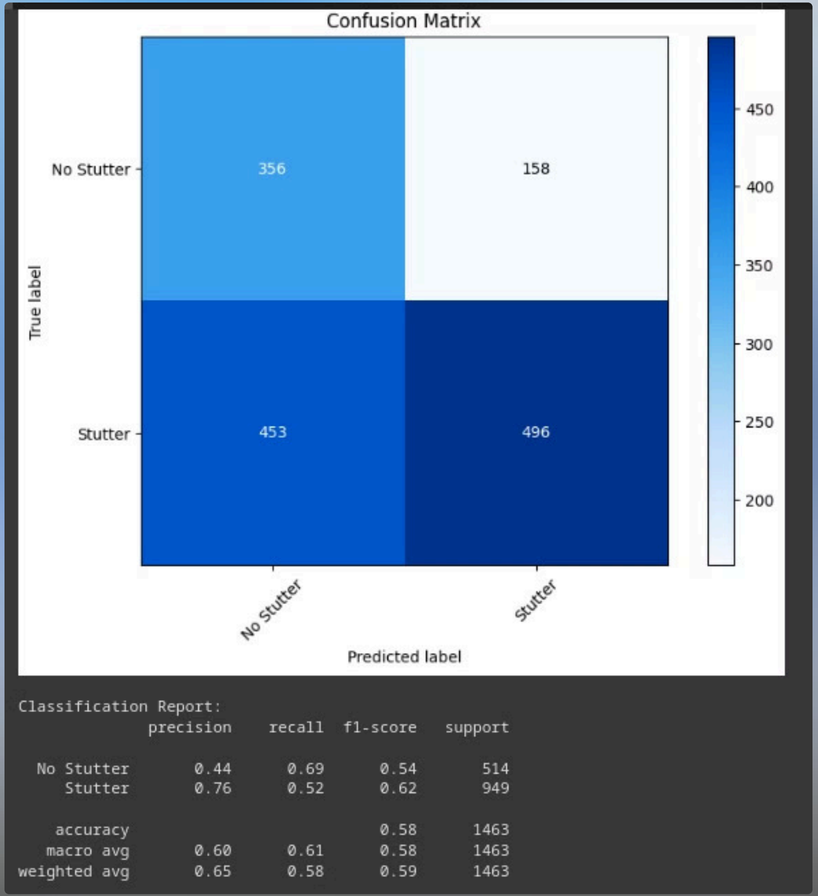


Training Process

- **Dataset Split**: 70% training, 15% validation, 15% testing
- **Optimization**: Adam optimizer with binary cross-entropy loss
- **Learning Rate**: Adaptive reduction (0.001 → 0.0005 → 0.00025)
- **Early Stopping**: Patience=15 with best weights restoration

Model Performance

- **Accuracy**: 58.24% on test set
- **AUC**: 0.6531 (moderate discrimination ability)
- **Precision**: 0.7584 (76% of predicted stutters are correct)
- **Recall**: 0.5227 (52% of actual stutters detected)
- **Class-specific Performance**:
  - Stutter class: P=0.76, R=0.52, F1=0.62
  - No Stutter class: P=0.44, R=0.69, F1=0.54



Model Performance Comparison

Metric/Feature	MFCC Model	Prosodic Model	Word-Level Model	Syllable-Level Model
Performance Metrics				
Accuracy	61.65%	57.69%	54.34%	58.24%
AUC	0.6783	0.6322	0.6073	0.65331
Precision	0.7771	0.7364	0.7172	0.7584
Recall	0.5732	0.5416	0.4889	0.5227
F1 (Stutter)	0.66	0.62	0.58	0.62
F1 (No Stutter)	0.56	0.52	0.50	0.54
Model Architecture				
Hidden Layers	128-64-32	64-32-16	128-64-32-16	64-32-16
Model Size	84.50 KB	51.62 KB	97.12 KB	17.25 KB

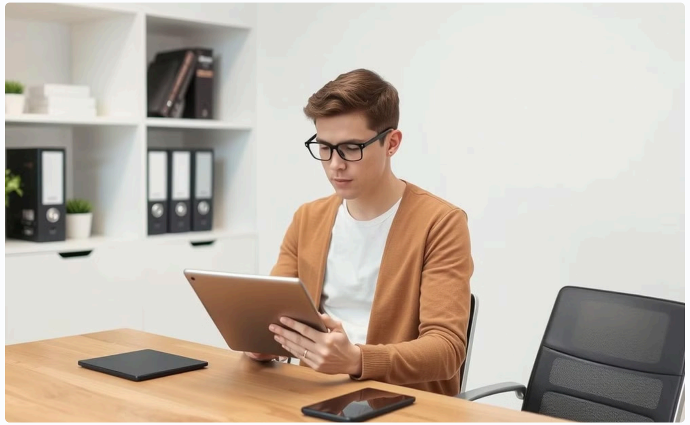
Key Observations:

- Prosodic features offer best balance of performance and interpretability
- Word-level model struggles with recall despite complex architecture
- All models show precision-recall trade-off, favoring precision

Future Directions

Multi-level Feature Integration

- Training and testing the models with permutations of the features discussed.
- We will also try to work different architectures like SVM,GMM, CNN with LSTM.



Multi-label Classification

- Extend stutter detection to classify specific stutter types such as blocks, prolongations, and repetitions using specialized and multi-task learning models.

Cross-Dataset Validation and K fold Validation

- Expand experiments to datasets like **FluencyBank**, testing model generalization to combine strengths from various training sources.
- We will try to evaluate the best model with **K-fold validation.**

Thank You