CLASSIFICATION OF PHONATION TYPES IN THE SPEAKING VOICE

Presented by - Manvendra Singh, Pavan Kumar
Professor ~M Kiran Reddy

INTRODUCTION

- **Phonation** is producing voiced sounds via vocal fold vibrations and sub-glottal pressure.
- Key types—**breathy**, **neutral, and pressed**—are vital in speech and singing.
- Classifying these helps in **emotion analysis**, **Speech Pathology**, .

OBJECTIVE

Classify phonation types from acoustic voice signals into

- Breathy
- Neutral
- Pressed

using Self Supervised models.

RELATED WORK

- The original work proposed the use of **Tunable Q-factor Wavelet Transform (TQWT)** to decompose acoustic signals into sub-bands.
- **Shannon entropy** was then calculated from each sub-band to quantify the information content of the signal.
- These entropy features were used as input to a **Feed Forward Neural Network (FFNN)** classifier trained separately for singing and speaking voice.
- The system achieved 81.67% for speaking voice, outperforming traditional features like MFCCs and SFFCCs.
- This method effectively captured the oscillatory nature of different phonation types but involved manual feature extraction and domain-specific tuning.

MODELS USED

Feature Extraction

- Facebook/Hubert-large-ls960-ft
- Facebook/Wav2vec2-base-960h
- Facebook/Wav2vec2-large-960h-lv60-self

Classifiers

- SVM
- MLP
- Adaboost
- Random Forest

Best

Hubert-large-ls960-ft (Layer 5)

+

SVM

BACKGROUND

Wav2vec 2.0 Base , wav2vec 2.0 Large , Hubert are Self Supervised models by Meta AI, they use transformer-based architecture and is pre-trained on large-scale unlabeled audio data

Hubert

Hubert uses a novel approach where it predicts clustered units derived from masked audio segments, enabling it to learn robust contextual representations of speech

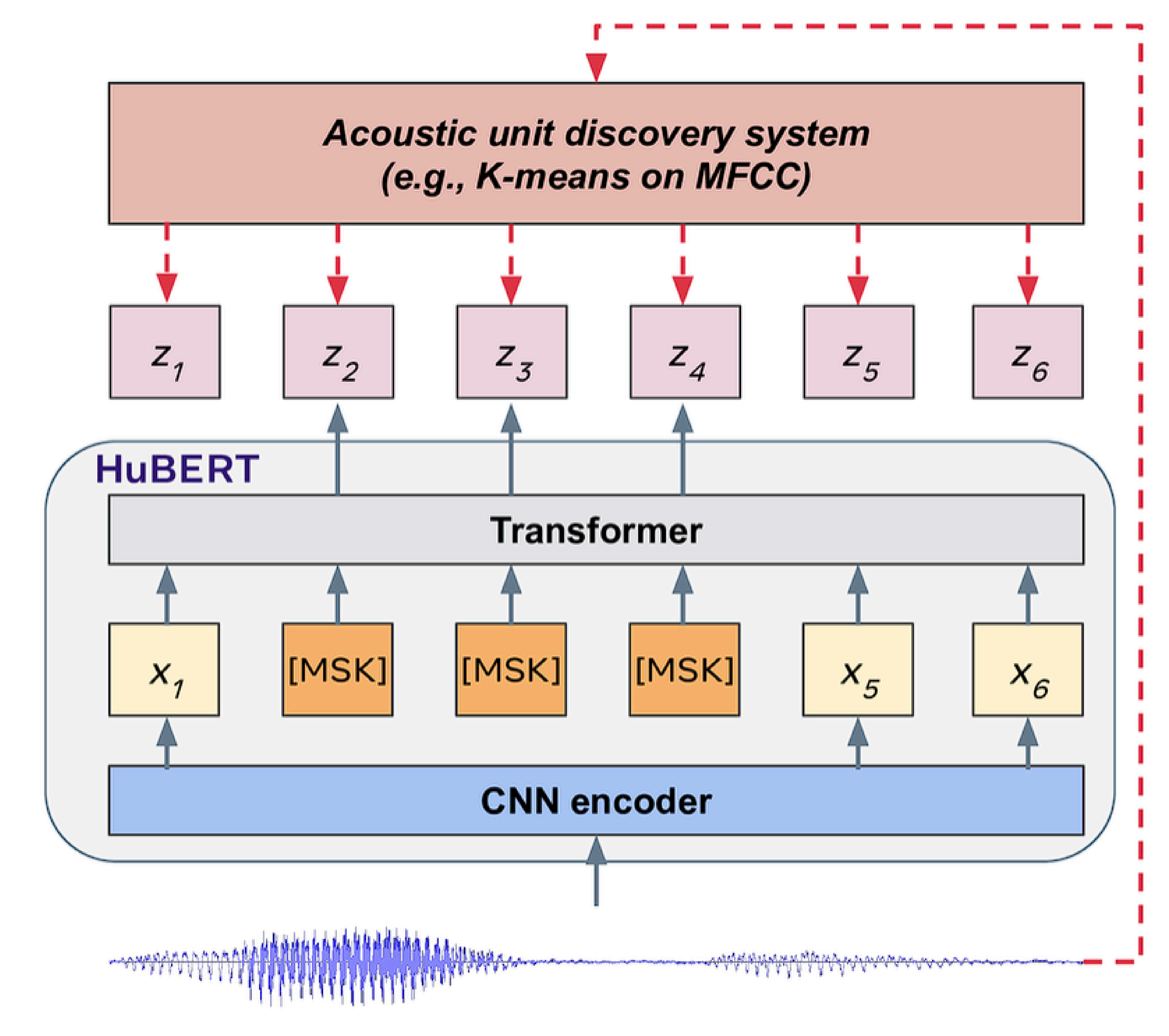
wav2vec 2.0 B

It employs a convolutional neural network (CNN) to process raw audio, followed by a transformer to capture contextual dependencies

wav2vec 2.0 L

Wav2Vec 2.0 Large (Wav2Vec2L) is a more powerful variant of Wav2Vec 2.0 B, with 24 transformer layers and a larger parameter count.

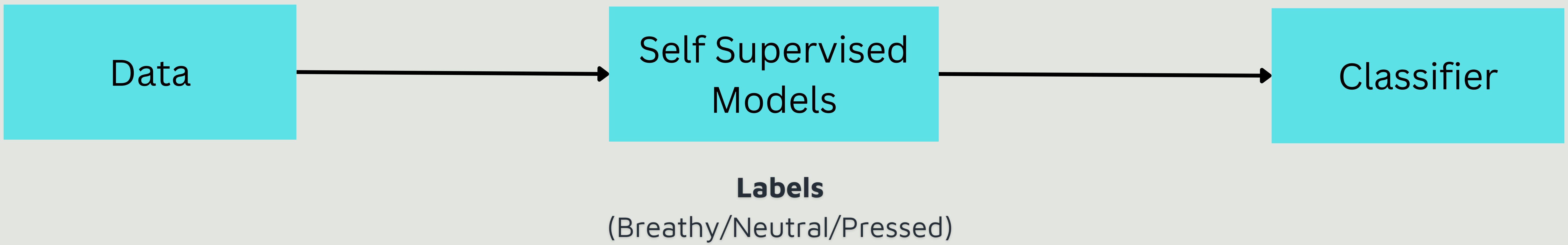
HUBERT ARCHITECTURE



- **Raw Audio Input:** Takes 16 kHz waveform as input to avoid manual feature extraction.
- **Convolutional Feature Encoder:** CNN extracts low-level features and down samples the audio.
- **Masking Module:** Randomly masks audio segments to enforce context learning.
- **Transformer Encoder:** Uses self-attention to model short and long-term dependencies.

- **Projection Layer:** Projects encoder output to a space matching cluster embeddings.
- **Codebook (Cluster Embeddings):** Holds k-means cluster targets for self-supervised prediction.

PROPOSED SYSTEM



(a) Training Phase

(b) Testing Phase

DATA

- **Dataset Composition:** The study uses recordings of the 8 Finnish vowels, each spoken in three phonation types — breathy, neutral, and pressed — by 11 speakers (6 female, 5 male) aged 18 to 48.
- **Data Volume:** Each vowel was repeated three times, totaling 792 isolated vowel samples (3 repetitions × 3 phonation types × 8 vowels × 11 speakers).
- **Recording Conditions:** The data was captured in an anechoic chamber at a sampling rate of 44.1 kHz, ensuring high-quality, noise-free recordings.

FEATURE EXTRACTION

Audio Processing

- For each audio directory (Normal, Breathy, Pressed)
- Iterate through audio files using tqdm for progress tracking
- Load audio with librosa at 16kHz sampling rate
- Process audio using model processor

Feature Extraction

- Extract hidden states from specified layer
- Compute mean features across time dimension
- Validate features:
- Ensure 1D array
- Check for zeros or NaNs
- Verify feature dimension matches expected size

Label Assignment

- Assign labels (0: Normal, 1: Breathy, 2: Pressed)
- Stack features and labels into arrays

Output

- Combine features (X) and labels (Y) from all directories
- Save to a pickle file in the format: features/task_type/model_name_l

CLASSIFICATION

Data Loading

- Load feature data from pickle file
- Initialize stratified k-fold cross-validation

Cross-Validation

- Perform k-fold cross-validation:
- Split data into training and test sets
- Standardize features
- Train classifier on training set
- Predict on test set
- Compute accuracy

Result Aggregation

- Aggregate results (mean and standard deviation of accuracy)

Output

- Save results (model details, parameters, performance metrics) to a JSON file

RESULT

- **Fine-tuned HuBERT model at Layer 5 for best feature extraction.**
- **Classifier used: Support Vector Machine (SVM) with RBF kernel.**

- **Optimal hyperparameters:**
1. **C = 100**
2. **gamma = 'auto'**
3. **probability = True**

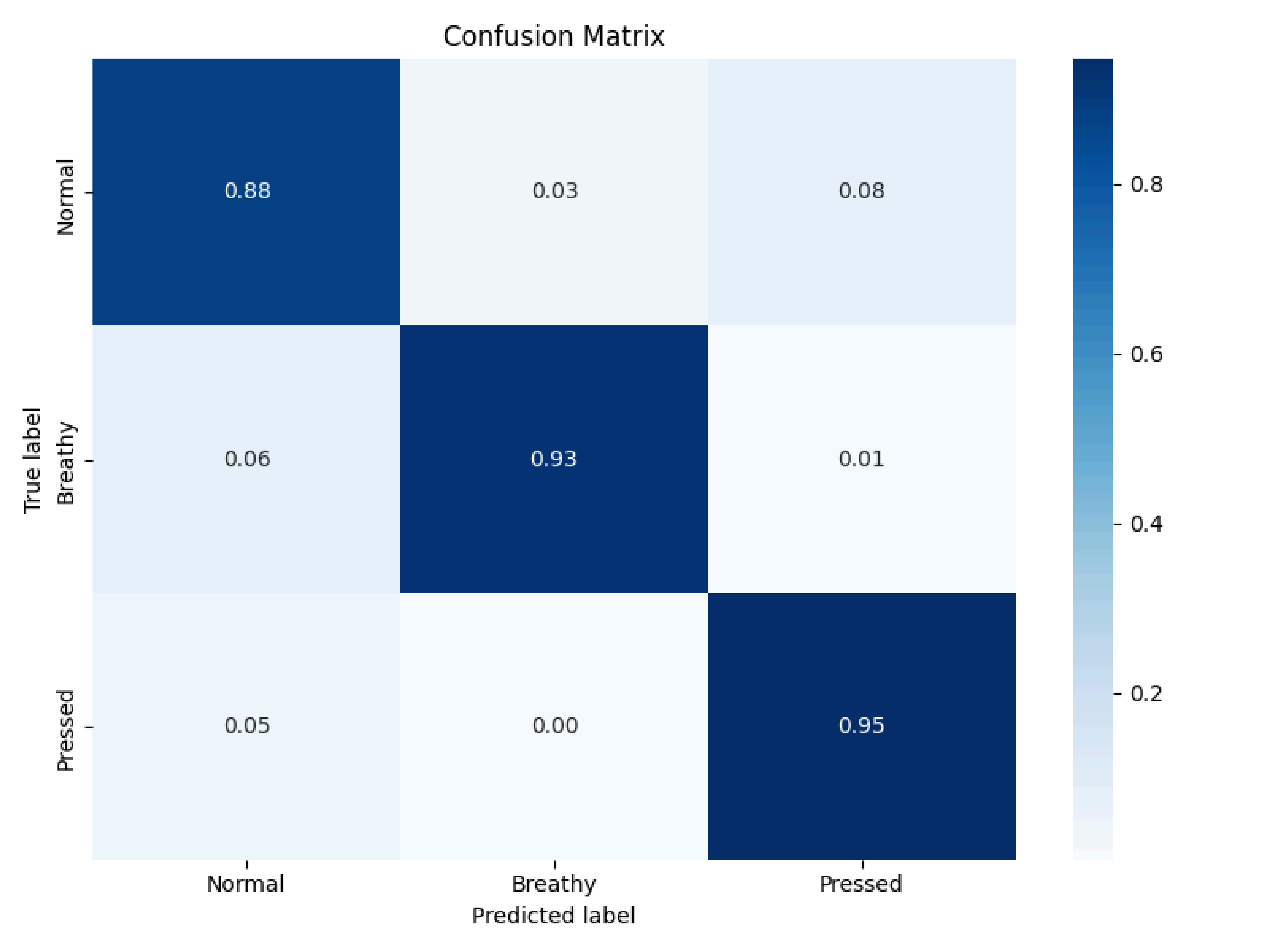
- **Achieved accuracy: 91.916% ± 2.84%.**

- **C:** Controls the trade-off between achieving a low training error and a low testing error (regularization strength).
- **Kernel:** Defines the function used to project data into higher dimensions; RBF captures non-linear patterns.
- **Gamma:** Determines how far the influence of a single training point reaches; small gamma means far, large gamma means close.
- **Probability:** Enables the model to output probability estimates instead of just class labels.

RESULTS

Model	Accuracy	Std-Accuracy
Support Vector Machine	91.916	2.84
MLP Classifier	89.506	4.47
AdaBoost	71.960	4.00
Random Forest	87.107	3.97

CONFUSION MATRIX



WHY ?

- **Feature Representation:** Traditional techniques (MFCCs, FBANKs, LPC) produce low-dimensional, features capturing spectral properties, while Hubert/Wav2Vec 2.0 extract high-dimensional, learned embeddings encoding both acoustic and linguistic information with contextual awareness.
- **Learning and Context:** Traditional methods use deterministic algorithms without training, processing short audio frames with limited context, whereas Hubert/Wav2Vec 2.0 employ self-supervised learning and transformers to capture long-range dependencies and adapt to diverse speech patterns.

CONCLUSION

- HuBERT embeddings outperforms Speech Processing Techniques used

- Hubert + SVM achieves 91.916 % accuracy, surpassing Previous Standard of 81.67%.

FUTURE WORK

While the current system using HuBERT + SVM achieved a high accuracy of 91.91%, there is still room for improvement through the following directions:

- Incorporating More Data
- Data Augmentation (Time-stretching, etc)
- Explore newer or larger self-supervised models.

REFERENCE

Research Paper Link

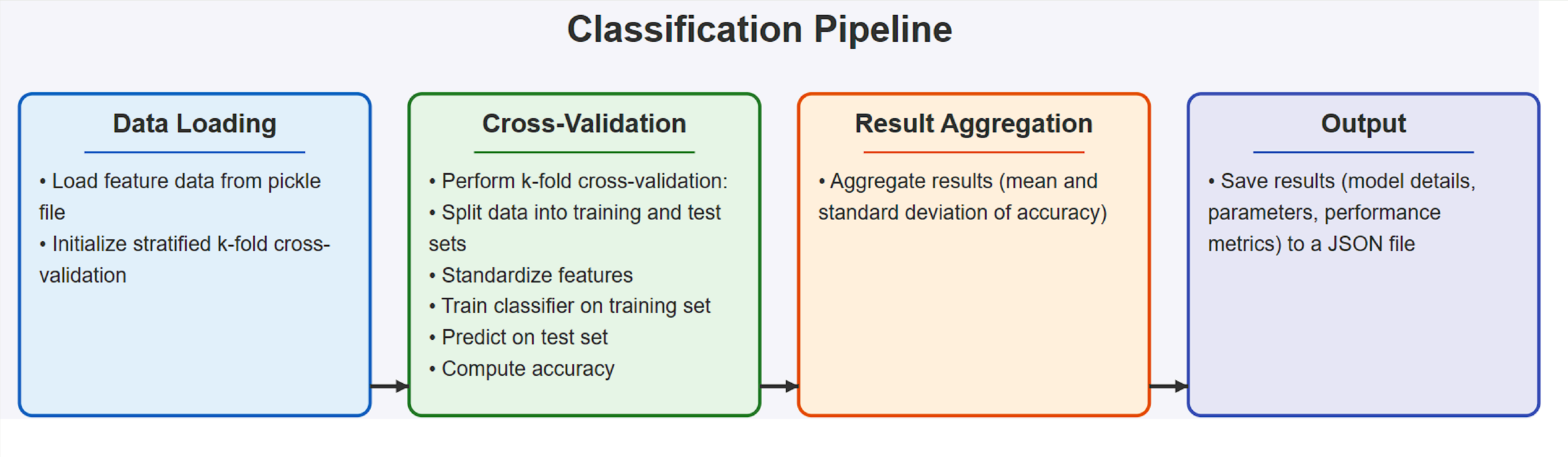
Thank You

For your attention

OVERVIEW

- Introduction
- Objective
- Related Works
- Models Used
- Architecture
- Proposed System
- Feature extraction
- Classification
- Result
- Conclusion

CLASSIFICATION

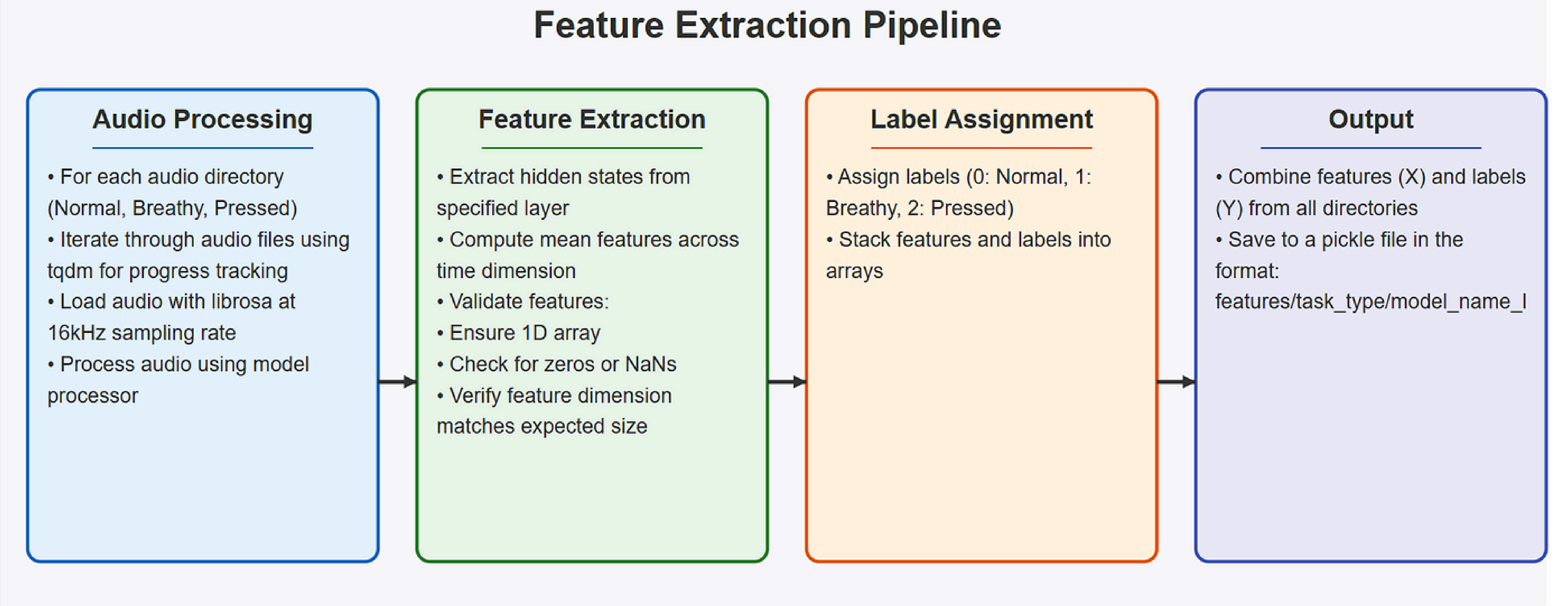


- Load feature data from pickle file
- Initialize stratified k-fold cross-validation
- Perform k-fold cross-validation:
  - Split data into training and test sets
  - Standardize features
  - Train classifier on training set
  - Predict on test set
  - Compute accuracy
- Aggregate results (mean and standard deviation of accuracy)

Output

- Save results (model details, parameters, performance metrics) to a JSON file.

FEATURE EXTRACTION



Feature Extraction

- For each audio directory (Normal, Breathy, Pressed):
  - Iterate through audio files using tqdm for progress tracking
  - Load audio with librosa at 16kHz sampling rate
  - Process audio using model processor
  - Extract hidden states from specified layer
  - Compute mean features across time dimension
  - Validate features:
    - Ensure 1D array
    - Check for zeros or NaNs
    - Verify feature dimension matches expected size
  - Assign labels (0: Normal, 1: Breathy, 2: Pressed)
- Stack features and labels into arrays.

Output

- Combine features (X) and labels (Y) from all directories
- Save to a pickle file in the format: features/task_type/model_name_layer.pkl