Audio Processing Pipeline Documentation

# Table of Contents

1. Architecture Overview  
2. Component Documentation  
3. API Reference  
4. Configuration Guide  
5. Performance Considerations

# 1. Architecture Overview

The Audio Processing Pipeline is a modular system designed for real-time audio processing, speech recognition, and response generation. It consists of multiple components that work together to process audio input and generate appropriate responses.

## Core Components

The pipeline consists of the following core components:

* • AudioCapture: Handles microphone input and audio file loading  
  • VAD (Voice Activity Detection): Detects presence of speech in audio  
  • AudioPreprocessing: Performs noise reduction and audio normalization  
  • SpeechToText: Converts audio to text using speech recognition  
  • TextProcessing: Analyzes text for intent and entities  
  • ResponseGenerator: Generates appropriate responses  
  • TextToSpeech: Converts text responses to speech

# 2. Component Documentation

## AudioCapture

Purpose: Captures audio input from microphone or loads audio files.  
  
Key Features:  
• Real-time audio streaming support  
• Multiple input device support  
• Configurable sample rate and chunk size  
• Buffer management for streaming audio  
  
Configuration Parameters:  
• sample\_rate: Audio sampling rate (default: 16000)  
• chunk\_size: Size of audio chunks (default: 1024)  
• channels: Number of audio channels (default: 1)  
• device\_index: Input device selection

## Voice Activity Detection (VAD)

Purpose: Detects presence of speech in audio input.  
  
Key Features:  
• Real-time speech detection  
• Configurable sensitivity  
• Noise filtering  
• Frame-level speech detection  
  
Configuration Parameters:  
• frame\_duration: Analysis frame duration (default: 30ms)  
• sensitivity: Detection sensitivity (default: 3)  
• trigger\_level: Speech activation threshold

# 3. API Reference

## REST API Endpoints

The pipeline exposes the following REST API endpoints:

### GET /status

Returns the current status of the pipeline.  
  
Request: None  
  
Response:  
{  
 "status": "running",  
 "components": ["AudioCapture", "VAD", ...],  
 "uptime": "2h 30m",  
 "processed\_requests": 150  
}

### POST /process/audio

Process audio data and return the response.  
  
Request:  
{  
 "audio\_base64": "base64\_encoded\_audio\_data",  
 "sample\_rate": 16000  
}  
  
Response:  
{  
 "success": true,  
 "transcription": "detected speech text",  
 "response": "generated response",  
 "audio\_response\_base64": "base64\_encoded\_audio\_response"  
}

# 4. Configuration Guide

The pipeline can be configured through a configuration file or environment variables.  
  
Key Configuration Areas:  
• Component-specific settings  
• Performance optimization parameters  
• API server configuration  
• Logging and monitoring settings

## Example Configuration

```python  
config = {  
 "audio\_capture": {  
 "sample\_rate": 16000,  
 "chunk\_size": 1024,  
 "channels": 1  
 },  
 "vad": {  
 "sensitivity": 3,  
 "frame\_duration": 30  
 },  
 "speech\_to\_text": {  
 "model": "whisper-small",  
 "language": "en"  
 }  
}  
```

# 5. Performance Considerations

Key performance factors to consider:  
  
• CPU Usage: The pipeline's performance depends on available CPU resources  
• Memory Usage: Components like speech recognition may require significant memory  
• Latency: Real-time processing requires optimized component configurations  
• Scalability: The pipeline can be scaled horizontally for multiple concurrent users

## Performance Optimization

Optimization strategies:  
  
• Use appropriate model sizes for speech recognition  
• Configure buffer sizes based on available memory  
• Enable component-level caching where appropriate  
• Implement proper error handling and recovery mechanisms