National College of Ireland

Configuration Manual

MSc Research Project

PsycoPipe-Multimodal Emotion Analysis Framework Using Fine-tuned SER, ASR with LLM models For Psychiatric Conversations.

Soumya Madhav

Student ID: x23332018

School of Computing

National College of Ireland

Supervisor: Sheresh Zahoor

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Soumya Madhav
Student ID:	X23332018
Programme:	MSc in Artificial Intelligence **Year:** 2024-2025
Module:	MSc Research Project
Lecturer:	Sheresh Zahoor
Submission Due Date:	01-09-2025
Project Title:	PsycoPipe- Multimodal Emotion Analysis Framework Using Fine-tuned SER, ASR models with LLM For Psychiatric Conversations
Word Count:	1655 **Page Count:** 11

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Soumya Madhav

Date: 31-08-2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

PsycoPipe Configuration Manual

Comprehensive Guide for Psychiatric Conversation Analysis Pipeline

Project Overview

PsycoPipe is a comprehensive pipeline for generating, analyzing, and evaluating psychiatric conversations using advanced AI models. The system combines:

- **Text Generation**: Psychiatric conversations using LLM (Gemma3)
- **Audio Generation**: Text-to-speech using IndexTTS with EARS voice dataset
- **Speech Recognition**: Fine-tuned Whisper ASR model
- **Emotion Recognition**: Fine-tuned Wav2Vec2 SER model
- **Ensemble Analysis**: Combined SER and ASR+LLM predictions
- **Evaluation**: Comprehensive performance analysis

Key Components

- **ASR Model**: Fine-tuned Whisper Tiny for psychiatric terminology
- **SER Model**: Fine-tuned Wav2Vec2 for 17 emotion categories
- **TTS System**: IndexTTS 1.5 with EARS voice dataset
- **LLM Integration**: Ollama with Gemma3 for analysis
- **Ensemble System**: Advanced emotion detection combining multiple modalities

---

System Requirements

Hardware Requirements

Component	Current/Recommended System	Minimum Requirements
CPU	AMD Ryzen 9 7900X 12-Core Processor (24 threads)	8+ cores, 3.0+ GHz
RAM	61GB total (43GB available)	32GB minimum, 64GB recommended
GPU	NVIDIA GeForce RTX 4090 (24GB VRAM)	NVIDIA GPU with 12GB+ VRAM (RTX 3080 or better)
Storage	SSD recommended for fast I/O operations	100GB+ free space
OS	Linux 6.8.0-78-generic	Linux (Ubuntu 20.04+ recommended)

Software Requirements

- **Python**: 3.10+ (required for virtual environments)
- **CUDA**: 12.7+ (for GPU acceleration)
- **Ollama**: Latest version for LLM integration
- **FFmpeg**: For audio processing

File Structure

```
PsycoPipe_FinalCode/
├── config.json                          # Main configuration file
├── config_loader.py                     # Configuration management
├── psyco_requirements.txt               # Main environment
dependencies
├── indextts_requirements.txt            # IndexTTS specific
dependencies
│
├── # Core Pipeline Scripts
├── generate_psy_text_csv.py             # Phase 1: Text generation
├── generate_inference_audio_indexTTS.py # Phase 2: Audio generation
├── remap_speech_emotion_recognition.py  # Phase 3: SER inference
├── whisper_asr_analysis_with_LLM.py     # Phase 4: ASR + LLM analysis
├── advance_ensemble_emotion_detection.py # Phase 5: Ensemble analysis
├── comprehensive_analysis_advanceEnsemble.py # Phase 6: Evaluation
│
├── # Training Data and Scripts
├── training_data/
│   └── scripts/
│       ├── FinetuneASR_Fast/           # ASR model training
│       │   ├── generate_psychiatric_audio_indextts15.py
│       │   ├── generate_psychiatric_text_samples.py
│       │   ├── whisper_finetune_Fast.py
│       │   └── whisper-ft-tiny/         # Trained ASR model
│       └── Finetuning_SER/             # SER model training
│           ├── generate_emotion_audio_for_SER_tuning.py
│           ├── generate_emotion_text_samples.py
│           ├── fast_ser_train.py
│           └── ser_w2v2_fast/           # Trained SER model
│
├── # Data Directories
├── conversations/                       # Generated conversation files
├── audio_outputs/                       # Generated audio files
├── Ears_voices/                         # EARS voice dataset
│   ├── p001/                           # Speaker 1 voice files
│   ├── p002/                           # Speaker 2 voice files
│   └── p003/                           # Speaker 3 voice files
│
├── # IndexTTS System
├── index-tts/                          # IndexTTS installation
│   ├── checkpoints/                     # Model checkpoints
│   └── indextts/                       # Source code
│
└── # Analysis Outputs
└── analysis_outputs_*/                  # Evaluation results
```

Environment Setup

1. Base Environment Setup

```
# Create base environment
python3.10 -m venv psyco_base
source psyco_base/bin/activate

# Install base requirements
pip install -r psyco_requirements.txt
```

2. IndexTTS Environment Setup

```
# Create separate environment for IndexTTS
python3.10 -m venv indextts_env
source indextts_env/bin/activate

# Install IndexTTS requirements
pip install -r indextts_requirements.txt

# Install IndexTTS from source
cd index-tts
pip install -e .
```

3. Ollama Setup

```
# Install Ollama
curl -fsSL https://ollama.ai/install.sh | sh

# Pull Gemma3 model
ollama pull gemma3

# Start Ollama service
ollama serve
```

4. Verify Installation

```
# Test GPU availability
python -c "import torch; print(f'CUDA available:
{torch.cuda.is_available()}')"

# Test IndexTTS
python -c "from indextts.infer import IndexTTS; print('IndexTTS
imported successfully')"
```

```
# Test Ollama
python -c "import ollama; print('Ollama available')"
```

5. Environment Management

The indextts_env is essential for running index_tts based audio generation, all other code runs in the psyco_base env

Activate Environments:

```
# Activate base environment
source psyco_base/bin/activate

# Activate IndexTTS environment
source indextts_env/bin/activate
```

Deactivate Environment:

```
deactivate
```

Pipeline Phases

The PsycoPipe project operates in two main phases, Note that each Step in a phase, If we download the entire bundle with the current outputs/dependencies and models packaged, you can run any step of each phase to see the relevant output for that script.

Phase 1: Fine-Tuning System

Part 1: Synthetic Training Data Generation

Scripts: -

- `training_data/scripts/FinetuneASR_Fast/generate_psychiatric_text_samples.py` - Generate ASR training text
- `training_data/scripts/FinetuneASR_Fast/generate_psychiatric_audio_indextts15.py` - Convert ASR text to audio
- `training_data/scripts/Finetuning_SER/generate_emotion_text_samples.py` - Generate SER training text
- `training_data/scripts/Finetuning_SER/generate_emotion_audio_for_SER_tuning.py` - Convert SER text to audio

Execution:

Script 1: Generate ASR training text

```
# Activate base environment
source psyco_base/bin/activate
```

```
# Navigate to ASR training directory
cd training_data/scripts/FinetuneASR_Fast/

# Generate ASR training text samples
python generate_psychiatric_text_samples.py
```

Output: - `training_data/scripts/FinetuneASR_Fast/psychiatric_text_samples.json` - ASR training text samples

```
(base) soumya-ai:~/soumya/PsycoPipe/PsycoPipe/training_data/scripts/Finetuning_ASR$ python generate_psychiatric_text_samples.py
 Generating Psychiatric/Medical Text Samples for ASR Fine-tuning
============================================================
Ollama available with models: ['gemma3:latest']

 Generating samples for category: patient_symptoms
   Subcategories: 15
 Generating samples for: depression symptoms
```

Script 2: Convert ASR text to audio

```
# Activate IndexTTS environment
source indextts_env/bin/activate

# Generate ASR training audio from text
python generate_psychiatric_audio_indextts15.py
```

Output: - `training_data/scripts/FinetuneASR_Fast/psychiatric_audio_outputs_indextts15/` - ASR training audio files

```
==================================================
FULL GENERATION COMPLETE!
Generated 2893/2893 audio files
Output directory: psychiatric_audio_outputs_indextts15

Audio Generation by Category:
   clinical_observations: 750 audio files
   medical_assessments: 495 audio files
   patient_symptoms: 602 audio files
   psychiatric_terminology: 400 audio files
   treatment_discussions: 646 audio files

Speaker Voice Distribution:
   p001: 625 audio files
   p002: 607 audio files
   p003: 550 audio files
   p004: 554 audio files
   p005: 557 audio files

Voice File Distribution:
   emo_neutral_freeform.wav: 1459 audio files
   emo_neutral_sentences.wav: 1434 audio files
(base) soumya-ai:~/soumya/PsycoPipe/PsycoPipe/training data/scripts/Finetuning ASR$
```

Script 3: Generate SER training text

```
# Activate base environment
source psyco_base/bin/activate
```

```
# Navigate to SER training directory
cd training_data/scripts/Finetuning_SER/

# Generate SER training text samples
python generate_emotion_text_samples.py
```

Output: - `training_data/scripts/Finetuning_SER/emotion_text_samples.json` - SER training text samples

```
(base) soumya-ai:~/soumya/PsycoPipe/PsycoPipe/training_data/scripts/Finetuning_SER$ python generate_emotion_text_samples.py
Emotion Text Sample Generator
==================================================
Model: gemma3
Samples per emotion: 150
Total emotions: 17
Connected to Ollama. Available models: ['gemma3:latest']

Processing emotion: amazement
Generating 150 samples for emotion: amazement
 Generated 150 samples for amazement

Processing emotion: amusement
```

Script 4: Convert SER text to audio

```
# Activate base environment
source psyco_base/bin/activate

# Navigate to SER training directory
cd training_data/scripts/Finetuning_SER/

# Generate SER training text samples
python generate_emotion_text_samples.py
```

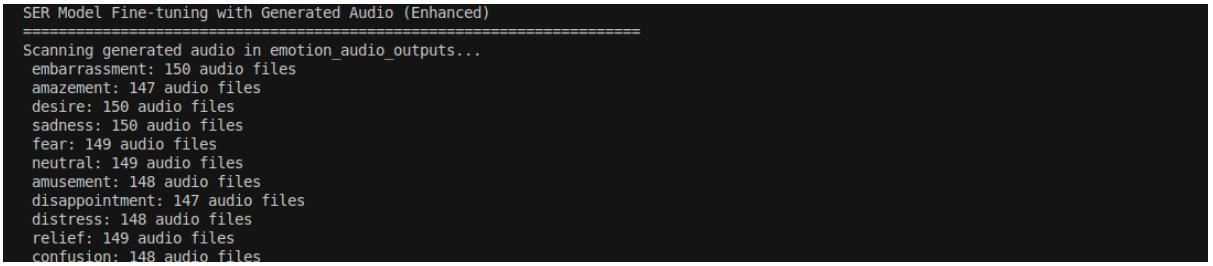
Output: - `training_data/scripts/Finetuning_SER/emotion_audio_outputs/` - SER training audio files

```
(base) soumya-ai:~/soumya/PsycoPipe/PsycoPipe/training_data/scripts/Finetuning_SER$ python generate_emotion_audio.py
Using device: cuda
🎮 GPU: NVIDIA GeForce RTX 4090
🎮 GPU Memory: 23.5 GB
Loading TTS model...
Model loaded in 6.28 seconds
🎵 Emotion Audio Generation
==================================================
Loaded 17 emotions
Total samples: 2523

Processing emotion: amazement (147 samples)
Processing sample 1/147: Wow, look at that incredible sunset!...
Generated: amazement_001.wav (2.88s, 7.31s) [male voice]
Processing sample 2/147: I can't believe how fluffy that puppy is....
Generated: amazement_002.wav (4.64s, 2.55s) [male voice]
Processing sample 3/147: That's absolutely stunning, you have to look at it...
Generated: amazement_003.wav (3.60s, 2.28s) [male voice]
```

Part 2: Fine-tuning SER and ASR on Synthetic Data

Scripts: - `training_data/scripts/FinetuneASR_Fast/whisper_finetune_Fast.py` - Fine-tune Whisper ASR model - `training_data/scripts/Finetuning_SER/fast_ser_train.py` - Fine-tune Wav2Vec2 SER model



Execution:

Script 5: Fine-tune Whisper ASR model

```
# Activate base environment
source psyco_base/bin/activate

# Navigate to ASR training directory
cd training_data/scripts/FinetuneASR_Fast/

# Fine-tune Whisper ASR model
python whisper_finetune_Fast.py
```

Output: - `training_data/scripts/FinetuneASR_Fast/whisper-ft-tiny/` - Fine-tuned ASR model - Training metrics and validation results - WER comparison reports

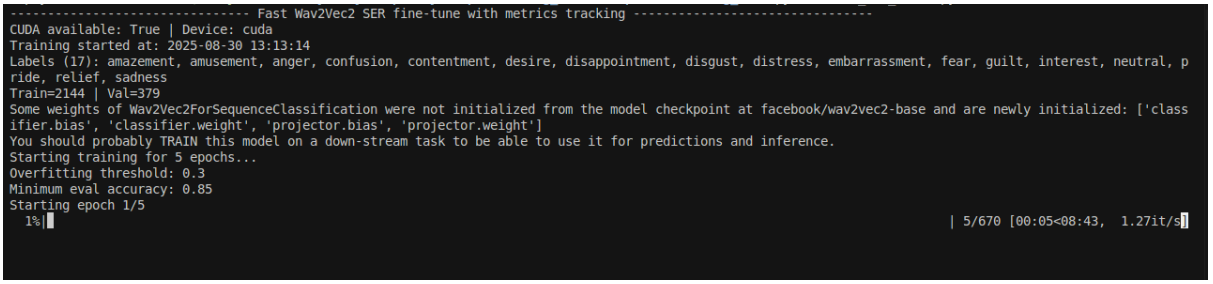
Script 6: Fine-tune Wav2Vec2 SER model

```
# Activate base environment
source psyco_base/bin/activate

# Navigate to SER training directory
cd training_data/scripts/Finetuning_SER/

# Fine-tune Wav2Vec2 SER model
python fast_ser_train.py
```

Output: - `training_data/scripts/Finetuning_SER/ser_w2v2_fast/` - Fine-tuned SER model - Training metrics and validation results - Emotion classification reports



Phase 2: Inference System

Part 1: Synthetic Data Generation For System Evaluation

Scripts: - `generate_psy_text_csv.py` - Generate evaluation conversations - `generate_inference_audio_indexTTS.py` - Convert conversations to audio

Execution:

Script 7: Generate evaluation conversations

```
# Activate base environment
source psyco_base/bin/activate

# Ensure Ollama is running
ollama serve

# Generate evaluation conversations
python generate_psy_text_csv.py
```

Output: - `conversations/conversation_*.txt` - Raw conversation text files - `conversations/conversation_*.csv` - Parsed conversation data files - Generation statistics and metadata

Script 8: Convert conversations to audio

```
# Activate IndexTTS environment
source indextts_env/bin/activate

# Generate evaluation audio from conversations
python generate_inference_audio_indexTTS.py
```

Output: - `audio_outputs/conversation_*/line*.wav` - Individual audio files - Voice mapping statistics and coverage report - Audio generation metadata

Part 2: Fine-tuned ASR + LLM, SER and Ensemble Analysis

Scripts: - `remap_speech_emotion_recognition.py` - SER inference using fine-tuned model - `whisper_asr_analysis_with_LLM.py` - ASR + LLM analysis using fine-tuned model - advance_ensemble_emotion_detection.py - Ensemble analysis combining SER and ASR+LLM

Execution:

Script 9: SER inference using fine-tuned model

```
# Activate base environment
source psyco_base/bin/activate

# Run SER analysis using fine-tuned model
python remap_speech_emotion_recognition.py
```

Output: - SER predictions for each audio file - Top 3 emotion predictions with confidence scores - Emotion mapping statistics - conversations/conversation_*_with_ser.csv - CSV with SER results

Script 10: ASR + LLM analysis using fine-tuned model

```
# Activate base environment
source psyco_base/bin/activate

# Ensure Ollama is running
ollama serve

# Run ASR + LLM analysis using fine-tuned model
python whisper_asr_analysis_with_LLM.py
```

Output: - Transcribed text from audio files - LLM analysis of psychiatric content - Emotion predictions from text analysis - WER (Word Error Rate) calculations - conversations/conversation_*_with_asr_llm.csv - CSV with ASR+LLM results

Script 11: Ensemble analysis combining SER and ASR+LLM

```
# Activate base environment
source psyco_base/bin/activate

# Ensure Ollama is running
ollama serve

# Run ensemble analysis combining SER and ASR+LLM
python advance_ensemble_emotion_detection.py
```

Output : Combined emotion predictions from both modalities Confidence scores and reasoning - Ensemble decision analysis - Performance metrics - conversations/conversation_*_with_ensemble.csv - CSV with ensemble results

Part 3: Results Evaluation

Scripts : `comprehensive_analysis_advanceEnsemble.py` - Final evaluation and reporting .

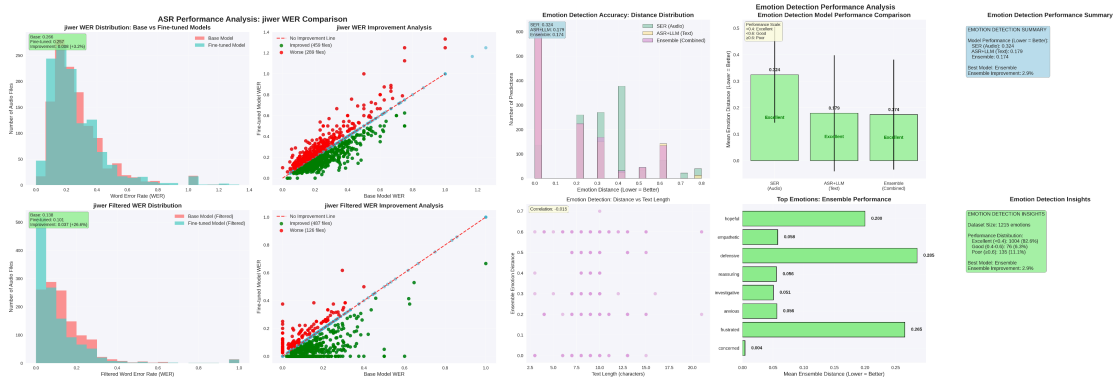
Execution:

Script 12: Final evaluation and reporting

```
# Activate base environment
source psyco_base/bin/activate

# Run comprehensive evaluation and reporting
python comprehensive_analysis_advanceEnsemble.py
```

Output : Performance dashboards and metrics - Statistical analysis reports - Visualization plots and charts - Detailed evaluation summaries - `analysis_outputs_*/` - Complete evaluation results directory - WER analysis by category - Emotion accuracy comparisons - Ensemble performance metrics



Configuration Reference

Main Configuration File (`config.json`)

The `config.json` file is the central configuration hub for the entire PsycoPipe system. It contains all parameters, paths, and settings organized into logical sections:

```
{
  "paths": {
    "base_dir": ".",
    "training_data_dir": "training_data",
    "scripts_dir": "training_data/scripts",
    "finetuning_asr_dir": "training_data/scripts/Finetuning_ASR",
    "finetuning_ser_dir": "training_data/scripts/Finetuning_SER",
    "conversations_dir": "conversations",
    "audio_outputs_dir": "audio_outputs",
    "psychiatric_audio_dir":
"training_data/scripts/Finetuning_ASR/psychiatric_audio_outputs",
    "voice_cache_dir": "ears_voices/ears"
```

```
  },
  "models": {
    "finetuned_whisper_path":
"training_data/scripts/Finetuning_ASR/finetuned_whisper_asr_modera
te",
    "finetuned_ser_path":
"training_data/scripts/Finetuning_SER/finetuned_ser_model_generate
d_audio_full",
    "finetuned_ser_checkpoint":
"training_data/scripts/Finetuning_SER/finetuned_ser_model_generate
d_audio_full/checkpoint-889"
  },
  "text_generation": {
    "num_conversations": 100,
    "exchanges_per_convo": 10,
    "model_name": "gemma3",
    "max_retries": 3,
    "delay_between_requests": 1.0,
    "genders": ["male", "female"]
  },
  "audio_generation": {
    "gpu_memory_gb": 24,
    "default_batch_size": 4,
    "gpu_memory_threshold": 0.85,
    "max_workers": 4,
    "audio_sample_rate": 22050,
    "audio_format": "wav",
    "output_dir": "audio_outputs",
    "max_audio_duration": 30.0,
    "min_audio_duration": 1.0
  },
  "voice_selection": {
    "use_ears_voices": true,
    "use_kyutai_voices": false,
    "voice_diversity": true,
    "gender_balance": true,
    "max_voices_per_conversation": 2,
    "voice_quality_threshold": 0.8
  },
  "evaluation": {
    "sample_size": 10,
    "max_samples": 100,
```

```
    "random_seed": 42,
    "save_results": true,
    "generate_plots": true,
    "output_format": "json",
    "conversation_limits": {
      "asr_evaluation": 100,
      "ser_evaluation": 100,
      "ensemble_evaluation": 100
    }
  },
  "wer_analysis": {
    "categories": [
      "psychiatric_terminology",
      "treatment_discussions",
      "medical_assessments",
      "symptoms_diagnosis"
    ],
    "min_samples_per_category": 5,
    "confidence_threshold": 0.7,
    "language_constraint": true,
    "language": "en"
  },
  "ser_evaluation": {
    "emotion_categories": 17,
    "confidence_threshold": 0.5,
    "batch_size": 8,
    "audio_preprocessing": {
      "sample_rate": 16000,
      "normalize": true,
      "trim_silence": true,
      "max_duration": 30.0
    }
  },
  "whisper_config": {
    "model_size": "base",
    "language": "en",
    "task": "transcribe",
    "beam_size": 5,
    "temperature": 0.0,
    "no_speech_threshold": 0.6,
    "logprob_threshold": -1.0,
    "compression_ratio_threshold": 2.4
```

```
  },
  "ser_config": {
    "model_type": "Wav2Vec2ForSequenceClassification",
    "feature_extractor": "Wav2Vec2FeatureExtractor",
    "num_labels": 17,
    "attention_probs_dropout_prob": 0.1,
    "hidden_dropout_prob": 0.1,
    "layer_norm_eps": 1e-12
  },
  "performance": {
    "track_gpu_usage": true,
    "track_cpu_usage": true,
    "track_generation_times": true,
    "track_audio_durations": true,
    "save_performance_logs": true,
    "log_interval": 10
  },
  "prompt_templates": {
    "psychiatric_consultation": "Generate a realistic psychiatric
consultation between a doctor and a patient. The doctor
conversation can include psychiatric medical terminology.\nThe
conversation should have approximately {exchanges_per_convo}
back-and-forth exchanges.\nEach line should include an emotion cue
in square brackets (e.g., [anxious], [calm], [angry]).\n\nThe
doctor is {doctor_gender} and the patient is
{patient_gender}.\n\nFormat it like:\nDoctor [emotion]:
text\nPatient [emotion]: text\n\nKeep the language sensitive,
professional, and suitable for mental health analysis.\n the
emotion cue should be only in the begining and should not be in
between the text \n do not add unnecessary quotes to the text
\nOnly return the conversation text.",
    "psychiatric_terminology": "Generate a realistic psychiatric
conversation using medical terminology.\nInclude terms related to:
{category}\nThe conversation should be between a psychiatrist and
patient discussing {topic}.\nUse professional medical language
while keeping it accessible.\nFormat: Speaker [emotion]: text"
  }
}
```