Table of Contents

Portable PsyAgent User Manual

Table of Contents

Project Overview

Portable PsyAgent is a portable psychological assessment agent system that supports multiple large model evaluators and local Ollama models. The system can perform multi-dimensional personality assessments on AI agents, generate detailed analysis reports, and support targeted enhanced stress testing.

Features

- **Multi-dimensional Personality Assessment** - Supports Big Five personality trait analysis
- **Multi-evaluator Support** - Supports OpenAI, Claude, Gemini, DeepSeek, GLM, Qwen, and local Ollama
- **Configuration-driven** - Easily switch models and parameters through configuration files
- **Detailed Analysis Reports** - Generates comprehensive reports including motivation analysis, personality traits, and behavioral patterns
- **Local Assessment** - Supports fully localized Ollama model evaluation
- **Debug Logs** - Complete conversation logs and debugging information
- **Batch Analysis** - Automatically processes large numbers of assessment reports with intelligent batch processing and progress tracking
- **Stress Testing** - Supports emotional stress, cognitive traps, and context load stress testing

System Requirements

- Python 3.8 or higher
- Windows, Linux, or macOS operating system
- At least 4GB RAM (8GB or higher recommended)
- Disk space: At least 10GB available space

Installation Guide

1. Clone the Project

```
git clone https://github.com/ptreezh/AgentPsyAssessment
cd AgentPsyAssessment
```

2. Install Dependencies

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


# Optional: Install Google Gemini support
pip install google-generativeai
```

3. Configure Environment Variables

Create a .env file or set environment variables:

```
# OpenAI
OPENAI_API_KEY=your_openai_key


# Anthropic Claude
ANTHROPIC_API_KEY=your_claude_key


# Google Gemini
GOOGLE_API_KEY=your_gemini_key


# Alibaba Cloud Qwen
DASHSCOPE_API_KEY=your_qwen_key


# DeepSeek
DEEPSEEK_API_KEY=your_deepseek_key


# GLM
GLM_API_KEY=your_glm_key
```

Quick Start

Using Ollama Local Models (Recommended)

Install Ollama

```
# Windows
# Download from https://ollama.ai/download


# Linux
curl -fsSL https://ollama.ai/install.sh | sh


# macOS
brew install ollama
```

Download Models

```
# Start Ollama service
ollama serve


# Download recommended models
ollama pull llama3:latest
ollama pull qwen3:8b
ollama pull mistral-nemo:latest
```

Basic Assessment

```
# Use default evaluator
python shared_analysis/analyze_results.py data/your_data.json


# Use specific evaluators
python shared_analysis/analyze_results.py data/your_data.json --evaluat
ors gpt claude


# Use local Ollama evaluators
python shared_analysis/analyze_results.py data/your_data.json --evaluat
ors ollama_llama3 ollama_qwen3
```

Detailed Usage

Project Structure

```
portable_psyagent/
├── llm_assessment/          # Assessment module
│   ├── run_assessment_unified.py  # Unified assessment entry
```

```
│   ├── roles/               # Role definition files
│   ├── test_files/          # Test question banks
│   ├── results/             # Assessment results
│   └── services/            # Core services
├── shared_analysis/         # Analysis module
│   ├── analyze_results.py   # Results analysis main program
│   ├── analyze_big5_results.py  # Big Five analysis
│   ├── analyze_motivation.py    # Motivation analysis
│   └── ollama_evaluator.py  # Ollama evaluator
├── interference_materials/  # Interference materials (stress testing)
├── config/                  # Configuration files
├── batch_analysis_output/   # Batch analysis output
└── docs/                    # Documentation
```

Assessment Module

Running Assessments

```
# Basic assessment
python llm_assessment/run_assessment_unified.py --model_name gemma3:lat
est --test_file big5 --role_name a1


# Assessment with stress testing
python llm_assessment/run_assessment_unified.py --model_name gemma3:lat
est --test_file big5 --role_name a1 --emotional-stress-level 3 --cognit
ive-trap-type p


# Set temperature parameter
python llm_assessment/run_assessment_unified.py --model_name gemma3:lat
est --test_file big5 --role_name a1 --tmpr 0.7


# Set context length
python llm_assessment/run_assessment_unified.py --model_name gemma3:lat
est --test_file big5 --role_name a1 --context-length-mode static --cont
ext-length-static 4
```

Assessment Parameters

- `--model_name`: Model identifier (e.g., ollama/gemma3:latest)
- `--test_file`: Test file name or path
- `--role_name`: Role name
- `--emotional-stress-level`: Emotional stress level (0-4)
- `--cognitive-trap-type`: Cognitive trap type (p, c, s, r)

- `--tmpr`: Model temperature setting
- `--context-length-mode`: Context length mode (auto, static, dynamic, none)
- `--timeout`: Model response timeout (seconds)

Analysis Module

Motivation Analysis

```
# Run motivation analysis (no API required)
python shared_analysis/analyze_motivation.py data/your_data.json --debu
g
```

Big Five Personality Analysis

```
# Basic Big Five analysis
python shared_analysis/analyze_big5_results.py data/your_data.json
```

Comprehensive Analysis

```
# Use default evaluator
python shared_analysis/analyze_results.py data/your_data.json


# Use specific evaluators
python shared_analysis/analyze_results.py data/your_data.json --evaluat
ors gpt claude


# Use local Ollama evaluators
python shared_analysis/analyze_results.py data/your_data.json --evaluat
ors ollama_llama3 ollama_qwen3
```

Stress Testing Module

Supported Stress Test Types

1. **Emotional Stress Testing** - Affects model performance through different levels of emotional stress
2. **Cognitive Trap Testing** - Introduces paradoxes, circularity, semantic fallacies, and procedural traps

3. **Context Load Testing** - Tests model processing capabilities by increasing context length

Stress Testing Parameters

- `-esL, --emotional-stress-level`: Emotional stress level (0-4)
- `-ct, --cognitive-trap-type`: Cognitive trap type
  - `p`: Paradox trap
  - `c`: Circularity trap
  - `s`: Semantic fallacy trap
  - `r`: Procedural trap
- `--context-length-mode`: Context length mode
  - `auto`: Automatic detection
  - `static`: Fixed length
  - `dynamic`: Dynamic ratio
  - `none`: Disable context injection

Configuration Files

Ollama Configuration (`config/ollama_config.json`)

```
{
  "ollama": {
    "base_url": "http://localhost:11434",
    "timeout": 120,
    "models": {
      "llama3": {
        "name": "llama3:latest",
        "temperature": 0.1,
        "max_tokens": 1024,
        "description": "Meta Llama 3 - General-purpose large model"
      },
      "qwen3": {
        "name": "qwen3:8b",
        "temperature": 0.1,
        "max_tokens": 1024,
        "description": "Alibaba Cloud Qwen3 - 8B parameter version"
      },
      "mistral": {
        "name": "mistral-nemo:latest",
```

```
        "temperature": 0.1,
        "max_tokens": 1024,
        "description": "Mistral NeMo - High-performance reasoning model
"
      }
    }
  },
  "evaluators": {
    "ollama_llama3": {
      "provider": "ollama",
      "model": "llama3",
      "description": "Llama3 local evaluator"
    },
    "ollama_qwen3": {
      "provider": "ollama",
      "model": "qwen3",
      "description": "Qwen3 local evaluator"
    },
    "ollama_mistral": {
      "provider": "ollama",
      "model": "mistral",
      "description": "Mistral NeMo local evaluator"
    }
  }
}
```

Batch Processing

Batch Analysis

```
# View file statistics
python ultimate_batch_analysis.py --stats


# Quick test (5 files)
python ultimate_batch_analysis.py --quick


# Analyze specific model (e.g., deepseek)
python ultimate_batch_analysis.py --filter deepseek


# Complete batch analysis (all 294 files)
python ultimate_batch_analysis.py


# One-click start for Windows users
start_batch_analysis.bat
```

Supported Assessment Data

The system supports automatic analysis of assessment reports in the `results/results` directory, containing test data from multiple model series.

Batch Analysis Features

- **Automatic Format Conversion** - Supports original assessment data formats
- **Intelligent Batch Processing** - Supports resume and error recovery
- **Progress Tracking** - Real-time display of analysis progress and estimated time
- **Detailed Reports** - Generates JSON and Markdown format summaries
- **Flexible Filtering** - Filters by model, sample size, and other conditions

Troubleshooting

Common Issues

1. **Ollama Connection Failure**

   ```
   # Check Ollama service
   ollama ps
   curl http://localhost:11434/api/tags
   ```

2. **Batch Analysis Interrupted**

   ```
   # Check output directory
   ls -la batch_analysis_results/

   # Re-run (will automatically skip completed files)
   python ultimate_batch_analysis.py --filter deepseek
   ```

3. **Insufficient Memory**

   ```
   # Reduce batch size
   python ultimate_batch_analysis.py --sample 10
   ```

4. **API Key Issues**

   ```
   # Check environment variables
   echo $OPENAI_API_KEY
   ```

5. **Missing Modules**

   ```
   # Install missing dependencies
   pip install google-generativeai
   ```

Debug Mode

```
# Enable detailed debug output
python shared_analysis/analyze_results.py data.json --evaluators ollama
_llama3
```

Check log files: - `logs/evaluator_conversation_log.txt` - Conversation logs - `logs/debug_info.json` - Debug information

API Reference

Assessment API

run_assessment_unified.py

Main parameters: - `--model_name` (required): Model identifier - `--test_file` (required): Test file - `--role_name`: Role name - `--debug`: Enable debug mode - `--test_connection`: Test model connectivity only

Stress testing parameters: - `--emotional-stress-level`: Emotional stress level (0-4) - `--cognitive-trap-type`: Cognitive trap type (p, c, s, r) - `--tmpr`: Model temperature setting - `--context-length-mode`: Context length mode - `--timeout`: Response timeout

Analysis API

analyze_results.py

Main functions: - Comprehensive analysis of assessment results - Generates Big Five and MBTI personality assessments - Supports multiple evaluators

analyze_motivation.py

Main functions: - Motivation analysis of assessment results - Generates motivation test reports - Supports Markdown format output

analyze_big5_results.py

Main functions: - Specialized analysis of Big Five personality traits - Generates detailed Big Five scoring reports

License