Advanced Predictive Analytics: Multimodal Sentiment Analysis Project

Project Overview

In this project, you will develop a comprehensive sentiment analysis system that processes both text and speech inputs. You will compare traditional neural network approaches with state-of-the-art large language models, providing valuable insights into the evolution and capabilities of different sentiment analysis techniques.

Learning Objectives

- Implement sentiment and emotion analysis using neural network architectures
- Deploy and utilize speech-to-text models locally
- Integrate with commercial LLMs via API
- Critically evaluate and compare different approaches to sentiment analysis
- Gain hands-on experience with the complete ML pipeline, from data preparation to model evaluation

Project Components

Part 1: Text-Based Sentiment Analysis

Develop a neural network model capable of:

- Binary sentiment classification (positive/negative)
- Multi-class emotion detection (joy, sadness, anger, fear, surprise, etc.)

Requirements:

- Use either PyTorch or TensorFlow (Keras) as your primary framework
- Experiment with at least two architectures (e.g., LSTM, CNN, Transformer)
- Train on a standard dataset (options include IMDb, and GoEmotions)
- Document model performance using appropriate metrics

Part 2: Speech-to-Text Integration

Extend your system to process spoken language:

- Deploy OpenAI's Whisper model locally to transcribe speech-to-text
- Feed the transcribed text into your sentiment analysis model from Part 1
- Evaluate the end-to-end pipeline performance

Requirements:

- Use the Whisper model for transcription (either base or small version)
- Collect a small test set of audio samples (can be self-recorded)
- Analyze how transcription errors impact sentiment analysis accuracy

Part 3: LLM-Based Sentiment Analysis

Implement sentiment analysis using commercial LLMs:

- Set up API access to OpenAI's GPT and/or Anthropic's Claude
- Develop prompts for sentiment and emotion classification
- Compare results with your neural network-based approach

Requirements:

- Use LangChain to manage API calls and prompt engineering
- Document prompt designs and their effectiveness
- Analyze cost-efficiency alongside performance metrics

Technical Requirements

Local Development Environment

- Python 3.9+ with appropriate ML libraries
- Virtual environment management (conda, venv, or similar)

Key Libraries and Frameworks

- **Deep Learning**: TensorFlow (Keras) or PyTorch
- **NLP Processing**: HuggingFace Transformers, NLTK, or spaCy
- **Speech-to-Text**: Whisper via OpenAI-Whisper package
- **LLM Integration**: LangChain and relevant provider packages
- **Visualization**: Matplotlib, Seaborn, or Plotly

Whisper Local Deployment

Install the OpenAI Whisper package:

bash
Copy
```
pip install openai-whisper
```

Basic usage example:

python
Copy
```
import whisper

# Load model (options: "tiny", "base", "small", "medium", "large")
model = whisper.load_model("base")

# Transcribe audio
result = model.transcribe("audio_sample.mp3")
text = result["text"]
```

LangChain Setup for LLM API Access

Install required packages:

bash
Copy
```
pip install langchain openai anthropic
```

Example configuration with Claude:

python
Copy
```
from langchain.llms import Anthropic
from langchain.prompts import PromptTemplate

# Initialize Claude client
claude = Anthropic(api_key="your_api_key")

# Create a prompt template
sentiment_prompt = PromptTemplate(
    input_variables=["text"],
    template="Analyze the sentiment of the following text and classify it as positive, negative, or neutral. Also identify the primary emotion expressed (joy, sadness, anger, fear, surprise, disgust). Text: {text}"
)
```

Note on API Access: Anthropic provides $5 of free credit when you create a new Claude account, which should be sufficient for this project. You can sign up at https://claude.ai to access this credit.

Deliverables

1. **Code Repository**:
    - Model implementation code
    - Data preprocessing scripts
    - Evaluation scripts
    - README with setup instructions
2. **Technical Report**:
    - 8-10 page report including:
    - Methodology and implementation details
    - Experimental results and analysis
    - Comparative evaluation of different approaches
    - Discussion of limitations and potential improvements
3. **Demo**: A simple web or command-line interface demonstrating your system's capabilities

Evaluation Criteria

- **Technical Implementation (45%)**: Quality, correctness, and efficiency of code
- **Analysis & Comparison (30%)**: Depth and insight of comparative analysis
- **Documentation (10%)**: Clarity and completeness of documentation
- **Innovation (15%)**: Novel approaches or extensions beyond basic requirements

Recommended Timeline

- **Weeks 1-2-3**: Text-based sentiment analysis implementation
- **Weeks 4**: Speech-to-text integration
- **Week 5**: LLM implementation and comparative analysis
- **Week 6**: Iterate through your components and improve them as much as you can and work on the final report and demo preparation

Resources

- HuggingFace Transformers Documentation
- OpenAI Whisper GitHub
- LangChain Documentation
- Anthropic Claude API Documentation