Turkmen Whisper Training App – Console Application

This tool allows you to fine-tune OpenAI's Whisper model on your own validated Turkmen audio dataset. It is built as a Python console application using Hugging Face's `transformers` library and is optimized to utilize GPU resources such as RTX 4090.

📁 Folder Structure in delivery

Whisper_Training_App/

├── dataset/

│ ├── metadata.csv

│ └── wavs/

│ ├── file1.wav

│ ├── file2.wav

│ └── ...

├── train_whisper.py → Console app for Whisper training

├── requirements.txt → Required Python packages

├── documentation.pdf → This document

The metadata.csv must have two columns: file and text. The file paths should be relative to the dataset/ folder. Metadata.csv can be generated by using the first console application sent. **The transcription work of 1972 wav files have already been been completed and the metadata being shared in this delivery is a validated data which can directly be used for training. However, if there are any observations with this transcription, first application can be run again to generate a new metadata file.**

🔧 System Requirements

- Python 3.8 to 3.10 (recommended)
- Linux or Windows OS
- NVIDIA GPU (CUDA-enabled) — e.g., RTX 4070 or 3070
- CUDA-compatible PyTorch installed
- FFmpeg and libsndfile installed
- `metadata.csv` file and `.wav` dataset folder

⚙️ Environment Setup (Linux)

1. Install system packages

```
sudo apt update
sudo apt install ffmpeg libsndfile1
```

2. Install PyTorch with GPU support (CUDA 11.8)

```
pip install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu118
```

3. Install additional Python packages

```
pip install -r requirements.txt
```

▶️ How to Run the Application

From the root folder, run:

```
python train_whisper.py --model openai/whisper-small
```

Optional arguments:

- `--csv_path` – path to metadata file (default: `dataset/metadata.csv`)
- `--audio_folder` – folder with `.wav` files (default: `dataset`)
- `--output_dir` – where trained model will be saved (default: `whisper-output/`)

You can choose any pre-trained Whisper model:

- `openai/whisper-tiny`
- `openai/whisper-base`
- `openai/whisper-small`
- `openai/whisper-medium`
- `openai/whisper-large`

📤 Output

After training completes, a model directory will be created at the specified `--output_dir`. This will include:

```
config.json
```

```
pytorch_model.bin
tokenizer.json
training_args.bin
...
```

These files represent your fine-tuned Whisper model and can be used for transcription or further development.

🧠 How It Works – Logic Behind the Script

1. **Loads metadata and audio:**
   It reads your `metadata.csv`, loads the corresponding `.wav` files at 16 kHz using the Hugging Face `datasets` library.
2. **Processes inputs for Whisper:**
   It uses `WhisperProcessor` to tokenize both the audio and the transcript text and prepares training input tensors.
3. **Loads the base model:**
   The selected Whisper model is downloaded and initialized for training.
4. **Fine-tunes the model:**
   Using Hugging Face's `Trainer`, it fine-tunes the model on your dataset with configurable parameters like learning rate, batch size, and steps.
5. **Uses GPU if available:**
   The script automatically detects GPU and enables mixed precision (`fp16=True`) for faster and memory-efficient training. Works seamlessly on RTX 4070 or 3070.
6. **Saves trained model:**
   After training, it saves all model weights and configuration in a folder for future use.

🆘 Need Help?

If you encounter any issues during setup or training, you can contact the developer for assistance. The code is modular and extensible for further use cases like evaluation, inference, or continuing training from a checkpoint.