The AI in Music

Daze Lu
Feb 8 & 22, 2025

Task for Lecture 2

Lecture 2 Fine tuning the music model

- Fine tuning introduction
- Fine tuning environment preparation
- prepare the music data
- Fine tuned model deployment

Fine Tuning

Fine-tuning is the process of **taking a general LLM and training it further** using a smaller dataset that is relevant to a **specific task or industry**.

Think of it like:

- **A general doctor (LLM) vs. a specialist (Fine-Tuned LLM)**
  - A general doctor knows medicine in a broad sense.
  - A cardiologist (heart specialist) is trained further in heart diseases.
  - Fine-tuning is like giving the LLM "specialist training."

Why Do We Fine-Tune an LLM?

Fine-tuning is useful when:

- You need the model to **generate highly accurate answers** in a **specific domain** (e.g., legal, medical, financial).
- You want the model to adopt a specific **style or tone** (e.g., customer service chatbots).
- The base model does not **understand specialized terminology** (e.g., scientific research papers).

💡 **Example: Fine-Tuning for Medical Diagnosis**

- A general LLM might not be great at diagnosing diseases.
- Fine-tuning with **thousands of patient records and medical papers** will help the model understand medical terminology and provide better responses.

Fine-Tuning vs. Prompt Engineering

Feature	Fine-Tuning 🛠️	Prompt Engineering 💡
Needs new training?	Yes ✅	No ❌
Custom data?	Yes ✅	No ❌
Flexibility	Best for domain-specific tasks	Works well for general tasks
Example	Training an AI on medical records	Writing a better prompt for ChatGPT

How is Fine-Tuning Done?

Step 1: Collecting Domain-Specific Data

- Get a dataset that represents the task (e.g., customer support chats, legal documents, medical records).
- Ensure data is **clean and structured**.

💡 **Example:** If fine-tuning for legal text analysis, you need thousands of **court rulings, contracts, and case studies**.

Step 2: Preprocessing the Data

- Convert text into a format the model understands (e.g., JSON, CSV).
- Remove **irrelevant or misleading data**.
- Format **inputs and outputs** properly.

How is Fine Tuning done?(2)

Step 3: Training the Model

- The pre-trained LLM is further trained on the new dataset.
- It adjusts its parameters (weights) to **learn the new information**.

Step 4: Evaluating and Testing

- Test the fine-tuned model on **real-world examples**.
- Compare its performance with the base model.
- Adjust and retrain if needed.

Types of Fine-Tuning

(A) Supervised Fine-Tuning

- The model learns from **labeled data** (input-output pairs).
- Works well for structured tasks like:
  - **Summarization**
  - **Translation**
  - **Question Answering**

Types of Fine-Tuning(2)

(B) Reinforcement Learning from Human Feedback (RLHF)

- The model is trained using **human preference feedback**.
- Used in **chatbots and assistants** to make responses more helpful.

💡 **Example:**

- If an AI gives **bad responses**, humans rate them.
- The model adjusts based on what **humans prefer**.

Hands-on practice

https://github.com/learner-crapy/audiocraft-test/blob/main/README.md

https://colab.research.google.com/drive/1vRwQC8GjpS3yT9XMkEhAatD6V3PTZmpo?usp=sharing

https://7be9c1fa2211d86cfc.gradio.live

https://4bb0a5d62b8fc10486.gradio.live

```
conda create -n audiocraft python==3.10 -y
conda activate audiocraft
git clone https://github.com/facebookresearch/audiocraft.git
cd audiocraft
conda install -c conda-forge pesq
pip install -e .
conda install "ffmpeg<5" -c conda-forge
python -m demos.musicgen_app
```

lud3@unlv.nevada.edu

```
python -m demos.musicgen_app --share
```

The errors and solutions

AttributeError: module 'gradio' has no attribute 'make_waveform'

import gradio

gradio.__version__

'5.13.1'

`pip install gradio==4.44.1`

EROR: Failed building wheel for pesq

windows

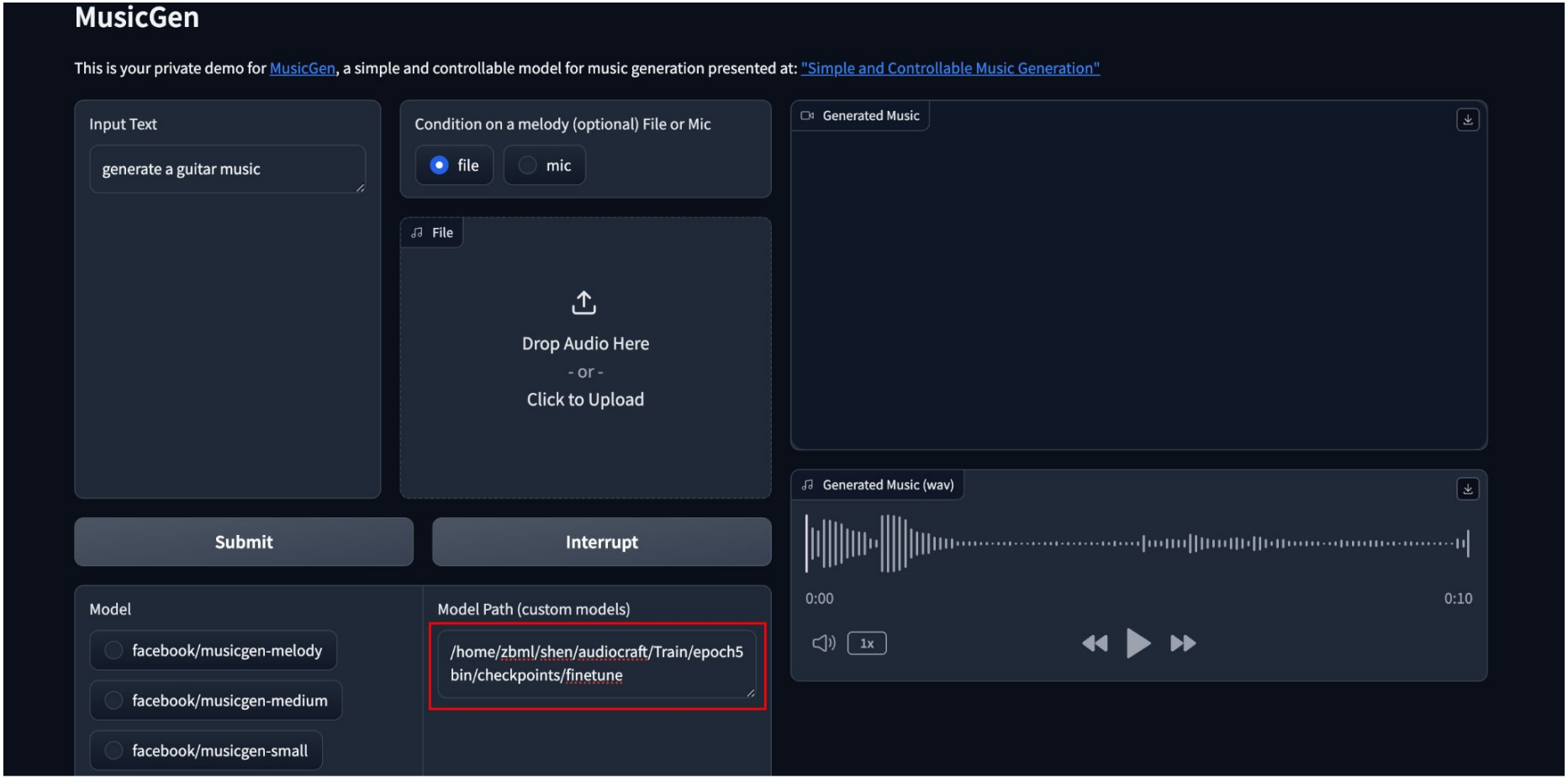
conda install -c conda-forge pesq

OSError: Can't load tokenizer for 't5-base'.

server

models--t5-base

models--facebook--encodec_32khz



The folder should have state_dict.bin compression_state_dict.bin