Deep Learning Data Processing Notes

ITAI 2377 Lab 04: Deep Learning Data Preprocessing

Colab Notebook Activities

- Uploaded `.wav` and `.mp3` audio files
- Loaded and trimmed the first 5 seconds using `librosa.load()`
- Extracted 20 MFCCs with `librosa.feature.mfcc()`
- Visualized MFCCs using `matplotlib` and `librosa.display`
- Standardized (normalized) MFCC values across coefficients
- Printed the shape of the raw and standardized MFCC matrices

Output and Results

- The MFCC matrix had a shape of **(20, 216)**, showing 20 coefficients across 216 time frames
- The MFCC plot clearly showed the spectral features of the uploaded audio (like a Meshuggah track 🤘)
- Standardization centered the data around **0** with a standard deviation of **1**, making it more suitable for model input

Concept Reflections

What are MFCCs?

MFCCs (Mel-Frequency Cepstral Coefficients) capture the tonal and timbral essence of audio, mimicking how human ears interpret sound. They're commonly used in speech recognition, music classification, and audio tagging.

Why is preprocessing still important with deep learning?

Even though deep learning models can uncover complex patterns, they still need clean, consistent input. Preprocessing ensures data is structured and noise is reduced so the model isn't distracted by irrelevant patterns or poor-quality signals.

Normalization vs. Standardization

- **Normalization** scales values to a specific range like [0, 1]
- **Standardization** transforms data to have a mean of 0 and standard deviation of 1

For this lab, I used standardization to center and scale the MFCCs for better learning behavior.

Use Case: Data Augmentation

- **Common**: Adding noise or pitch shifts to voice commands to boost model robustness
- **Uncommon**: Augmenting audio from underrepresented groups (e.g., people with speech impairments) to improve fairness
- **Inclusive**: Enhancing gesture videos in sign language recognition by applying transformations like rotation and lighting changes to simulate signing variability

Preprocessing Challenges & Mitigation Strategies

Challenge	Mitigation Strategy
Data Leakage	Always split train/test before scaling; use only training stats
Corrupt or Missing Files	Add validation checks, try/except blocks, and logging
Inconsistent Processing	Use reusable, well-tested preprocessing functions
Over-cleaning	Visually inspect effects; balance cleaning with data richness
Hardcoded Assumptions	Extract metadata programmatically and keep logic parameterized

Real-World Parallel: Medical Imaging

In medical applications like lung CT scan analysis, preprocessing steps—such as contrast adjustment, cropping regions of interest, and rescaling—help models focus on the relevant anatomy. These steps are critical to improving accuracy and trust in life-dependent predictions.

What I Took Away from This Lab

- Applied audio-specific preprocessing techniques like MFCC extraction and standardization
- Strengthened understanding of how normalization and augmentation apply to real-world AI tasks
- Built more resilient, reusable data pipelines
- Reflected on fairness and inclusion in AI through underrepresented use case