Dataset preprocessing summary

1. Objective

The goal of the preprocessing is to prepare audio datasets (LJSpeech and WaveFake) for training deep learning models for audio deepfake detection later testing on adversarial examples. We modify he dataset to better fit our resource constraints and better model training, optimizing for CNNs.

2. Datasets

1. **LJSpeech (Real)**
   - Original size: 3.54 GB
   - Preprocessed size: 6.61 GB [increase due to PNGs]
   - Structure: Split into **`train`, `val`, `test`**, each containing subfolders **`audio`, `mel`**, and **`images`** (PNG spectrograms).

2. **LJspeechV2 (Real)**
   - Original size: 3.54 GB
   - Preprocessed size: 18.0 GB [increase due to higher PNG quality and 3 channel mel-spectogram]
   - Structure: Split into **`train`, `val`, `test`**, each containing subfolders **`audio`, `mel`**, and **`images`** (PNG spectrograms).
3. **WaveFake (Fake)**
   - Original size: ~30 GB
   - Preprocessed size: 22 GB without images; reduced size by skipping PNGs and Japanese data.
   - Structure: Split into **`train`, `val`, `test`,** each containing subfolders **`audio`** and **`mel`**.
   - Only English vocoders retained; Japanese data is excluded.

4. **Combined Dataset (Real+Fake)**
   - Size: 8.40 GB
   - Structure: Split into **`train`, `val`, `test`,** each containing subfolders.

3. Preprocessing Steps

3.1 Audio Preprocessing

Step	Description	Purpose	Notes / Improvements
Resampling	Convert all audio to fixed rate (22.05 kHz)	Ensure consistent temporal resolution	Applied to both LJSpeech and WaveFake
Silence Trimming	Remove leading/trailing silence using `librosa.effects.split`	Reduces noise bias	Top dB threshold = 40
LUFS Normalization	Loudness normalization to target -23 LUFS using `pyloudnorm`	Standardizes perceived volume across files	Previously done without `pyloudnorm`; now more accurate
Clipping Prevention	`np.clip(audio, -1.0, 1.0)`	Avoids distortion in audio playback and model input	Some warnings may still appear due to extreme peaks
Standardization of Mel-Spectrogram	`(mel - mean) / std`	Ensures zero mean and unit variance for CNN stability	Added in updated version
Mel-Spectrogram Computation	`librosa.feature.melspectrogram` + log scaling	Captures spectral features suitable for CNNs	`n_mels=80`, `n_fft=2048`, `hop_length=512`

Step	Description	Purpose	Notes / Improvements
3-Channel Stack (Optional)	Duplicate grayscale mel into 3 channels	Makes compatible with pre-trained CNNs expecting 3-channel input	Not used in final version to reduce storage
Resize Images (Optional)	Convert mel-spectrogram to 299x299 PNG for visualization	Human readability / visualization	Skipped for WaveFake due to storage concerns

3.2 Dataset Splitting

Split	Method	Purpose
`train` / `val` / `test`	`sklearn.model_selection.train_test_split` with `random_seed=42`	Ensure reproducible splits; maintain similar distributions across splits

- LJSpeech: Split into train/val/test (~80/10/10%)
- WaveFake: same split, but preprocessed separately.

3.3 Data Storage

Folder	Contents	Notes
`audio`	Normalized WAV files	Input for models if using raw audio
`mel`	Standardized Mel-	Input for CNNs

Folder	Contents	Notes
	spectrogram `.npy` files	
`images`	PNG spectrograms (optional)	For visualization only, skipped in final WaveFake preprocessing

- Logs folder created to store warnings and CSV metadata.

4. Dataset Merging (Real vs Fake)

- For training, a combined dataset is created:

  1. Real samples from LJSpeech.
  2. Fake samples randomly selected from WaveFake per split.
  3. **Balanced splits**: Each split (**`train`**, **`val`**, **`test`**) has equal number of real and fake samples.
  4. Folder structure per split:

     ```
     train/
           audio/
              real/
              fake/
          mel/
              real/
              fake/
     val/
          audio/
          mel/
     test/
          audio/
          mel/
     logs/
         train_labels.csv
         val_labels.csv
         test_labels.csv
     ```

- CSV files store file paths and labels (`0=real`, `1=fake`) for training.

5. Improvements Over Time

Version	Change / Improvement	Reason / Benefit
Original	Resampling, silence trimming, LUFS normalization, mel scaling, standardization	Basic preprocessing for LJSpeech
Updated	Added `pyloudnorm` for accurate LUFS	Better loudness consistency
Updated	Clipping prevention with `np.clip`	Reduces distortion in extreme peaks
Updated	Standardization explicitly applied to mel spectrogram	Ensures CNN stability, better transfer learning
Updated	Optionally stacked 3-channel spectrograms [skipped later on]	Compatible with pre-trained CNNs (later skipped to save storage)
Updated	Skipped PNG generation for WaveFake	Reduced dataset size from 31GB to 22 GB
Updated	Logs folder and CSV files for metadata	Track issues, facilitate model training
Updated	Balanced dataset merge with real/fake subfolders	Avoid class imbalance during training

6. Potential Issues Without Steps

Step Skipped	Consequence
Resampling	Models may receive inconsistent temporal resolution; poor training
Silence trimming	Noise and silent sections may bias the model
LUFS normalization	Perceived volume differences may cause model to focus on amplitude rather than content
Standardization	CNN activations may explode or vanish; slow or unstable training
Clipping prevention	Distorted audio could mislead the model or cause errors

Step Skipped	Consequence
Split balancing	Class imbalance leads to biased model toward over-represented class

7. Transfer Learning Considerations

- Mel-spectrogram `.npy` files can be fed into CNNs.
- Pre-trained models (ResNet-18, EfficientNet, MobileNet) can accept **grayscale single-channel inputs**.
- 3-channel stacking is optional; skipped here to reduce storage requirements.
- Augmentation (on-the-fly in Colab) can increase data diversity for both real and fake classes.