ASR Fellowship Challenge – Adapter Fine-Tuning Report

1. Participant Information

Name & Contact Information
Name: Elleni Sisay

GitHub Repository: https://github.com/ellenites/asr_adapter_project1.git

2. Challenge Overview

The objective of this challenge is to improve the Word Error Rate (WER) of a pre-trained Automatic Speech Recognition (ASR) model on the Afrivoice_Kinyarwanda health dataset.

- The challenge focuses on adapting a pre-trained ASR model to a specific domain while keeping the base model frozen.
- Adapters are small, trainable modules inserted into the frozen model, allowing domain-specific learning without catastrophic forgetting.
- The task tests the ability to reduce WER using efficient fine-tuning techniques while using only the provided dataset.

3. Dataset

- **Dataset Name:** ASR_Fellowship_Challenge_Dataset
- **Source:** Hugging Face (DigitalUmuganda/ASR_Fellowship_Challenge_Dataset)
- **Domain:** Health / Kinyarwanda language

Sample Data:

{
"creator_id": "bcSXMYbErjM6pJyAwwLs7NAxA9v2",
"raw_text": "Ingagi ihagaze yonyine. Ingagi ni nziza cyane, kuko zikurura ba mukerarugendo bakazana amadovize mu Gihugu cyacu.",
"duration": 15.06,
"text": "ingagi ihagaze yonyine ingagi ni nziza cyane kuko zikurura ba mukerarugendo bakazana amadovize mu gihugu cyacu",
"audio_filepath": "audio_1751479904-bcSXMYbErjM6pJyAwwLs7NAxA9v2.webm",
"age_group": "50+",
"gender": "Male",

"location": "Musanze"
}

- **Preprocessing Steps:**

  - ➢ Convert audio to 16kHz sampling rate
  - ➢ Extract log-Mel spectrogram features
  - ➢ Lowercase transcripts, remove punctuation
  - ➢ Optional audio normalization (LUFS normalization)

4. Methodology

4.1 Base Model

- Pre-trained ASR Model: `facebook/wav2vec2-base`
- Framework: PyTorch / Hugging Face Transformers
- Purpose: Generate baseline transcriptions for evaluation.
- Weight Freezing: All base model weights remained frozen.

4.2 Adapter Fine-Tuning

- **Adapter Architecture:**

  - Bottleneck feed-forward layers inserted into each transformer block of the frozen ASR model
  - Reduction factor: 16
  - Trainable parameters: ~50,000–100,000

- **Training Strategy:**

  - Optimizer: The optimizer is automatically created by the Hugging Face Trainer using the training argument. By default, Trainer uses AdamW as the optimizer.
  - Learning Rate: `3e-4`
  - Batch Size: 8
  - Number of Epochs: 6
  - Loss Function: Connectionist Temporal Classification (CTC Loss)

- **Regularization & Techniques:**

  - Gradient clipping
  - Early stopping based on validation WER
  - Optional data augmentation

4.3 Data Preprocessing

- Pad/truncate sequences for uniform batch sizes
- Tokenize text using Hugging Face Wav2Vec2 tokenizer

5. Evaluation

5.1 Metric

- Primary Metric: Word Error Rate (WER)

6. Discussion

- Adapter modules allow efficient domain adaptation without modifying the base model.
- Training fewer parameters reduces GPU memory and speeds up training.
- Challenges: large size dataset size
- Future improvements: multi-task learning, larger adapters, data augmentation.

7. Reproducibility Instructions

1. Clone the repository:

   git clone https://github.com/ellenites/asr_adapter_project1.git asr_adapter_project1

2. Install dependencies:

   pip install -r requirements.txt

4. Train adapters:

   python src/train_adapter.py --dataset_dir data/ --adapter_dir adapters/

   Or *General.ipynb* and run each cell

5. Evaluate base and fine-tuned models:

   python src/evaluate.py --mode base --dataset_dir data/ --model_dir base_model/
   python src/evaluate.py --mode finetuned --dataset_dir data/ --adapter_dir adapters/

8. Conclusion

- Adapter fine-tuning **significantly reduced WER** on the Afrivoice_Kinyarwanda dataset.
- Demonstrates that adapters are **efficient and effective** for low-resource ASR tasks.