# Technical Report: Lightweight Swahili ASR for Edge Devices

## 1. Introduction

This report details a solution for the AI for Good Swahili ASR Challenge, a project aimed at developing a lightweight, real-time, and privacy-preserving Automatic Speech Recognition (ASR) system for Swahili-speaking communities. Our solution addresses the primary challenge of deploying AI in low-resource environments by leveraging a fine-tuned, state-of-the-art model optimized for efficiency. A key emphasis of our approach is the use of **real-world, conversational data**, robust **noise augmentation**, and **Parameter-Efficient Fine-Tuning (PEFT)** to ensure the model's performance and practicality.

## 2. The Power of Real-World Data

The core of our solution's success lies in our strategic choice of training data. Standard ASR benchmarks often consist of "read speech"—clean, studio-recorded audio—which does not reflect the complexities of natural conversation. Real-world conversational speech is messy; it includes disfluencies like pauses and interruptions, as well as background noise. To build a robust model, we deliberately chose datasets that capture these real-world conditions.

- **Dataset Selection**: We used the **Sunbird/salt** dataset, which is a collection of multispeaker and studio-recorded Swahili conversations. This dataset is crucial because it includes multiple speakers and reflects the natural rhythm and variations of spoken language. By training on data from diverse contexts, our model learns to generalize beyond ideal conditions, making it more effective in real-life scenarios like phone calls or face-to-face interactions.
- **Improved Generalization**: This focus on conversational data directly addresses the challenge's goal of building a solution that "works under real-world, conversational conditions." A model trained on such data is far less likely to fail when confronted with the natural quirks of human speech, which are common in low-resource environments.

## 3. Noise Augmentation Strategy: Building Robustness

To further enhance the model's resilience to noisy environments, we implemented a sophisticated **noise augmentation strategy**. This method applies noise directly to the audio waveform during training, making the model more robust to varying acoustic environments without requiring pre-processing of the entire dataset. We used the **Sunbird/urban-noise-uganda-61k** dataset, which contains a variety of real-world urban noises from East Africa. This choice is key because it introduces a high degree of

**ecological validity**, training the model to distinguish speech from sounds it will actually encounter in the target communities.

A custom DataCollator randomly selects a noise clip from the pool and superimposes it onto a clean speech segment. The amplitude of the added noise is also randomized, ensuring the model doesn't overfit to a single signal-to-noise ratio. This dynamic approach ensures every training sample is unique, significantly improving the model's generalization capabilities.

## 4. Audio Data Analysis and Inference Performance

A thorough analysis of the audio data was crucial for developing a robust model. We processed the Sartify_ITU_Zindi_Testdataset to understand its characteristics and identify potential issues. All files in the dataset were found to have a **16 kHz sample rate** and were **mono-channel**, simplifying the data pipeline by eliminating the need for complex resampling or channel conversion. The average audio duration was approximately **6.08 seconds**, indicating that the model would need to handle a range of conversational snippets.

## 5. Model Selection and Architecture

The choice of the base model was critical for balancing performance with the strict resource constraints of the challenge.

- **Model**: We selected the **Abdoul27/whisper-turbo-v3-model**.
- **Architecture**: This model is based on the state-of-the-art **Whisper architecture**, a sequence-to-sequence Transformer model. Crucially, it is a **distilled version**, meaning it has been intentionally made smaller and more efficient than the original Whisper models. This "turbo" designation signifies its optimization for speed, making it an ideal candidate for real-time inference on edge devices.
- **Justification**: This model was chosen for two primary reasons:
    1. **Pre-existing Knowledge**: It was pre-trained on the **Common Voice 12 Swahili dataset**. This gave it a strong foundational understanding of Swahili phonetics, grammar, and vocabulary, which significantly accelerated the fine-tuning process and led to better final performance.
    2. **Efficiency**: Its distilled nature directly addresses the challenge's requirements for a low memory footprint and a fast Real-Time Factor (RTFx). It is designed to deliver high accuracy without the computational overhead of larger models.

## 6. Training with PEFT and 8-bit Quantization

Training a large ASR model like Whisper-Turbo on limited hardware (NVIDIA T4 with ≤16 GB VRAM) is a major challenge. We tackled this by using **Parameter-Efficient Fine-Tuning (PEFT)** with **LoRA** and **8-bit quantization**.

- **PEFT/LoRA**: Instead of fine-tuning the entire model (which would require immense memory), LoRA freezes the original model weights and adds small, trainable "adapter" matrices to the attention layers. Only these tiny matrices are trained, drastically reducing memory usage and computational cost while preserving the model's core knowledge.
- **8-bit Quantization**: We loaded the model in 8-bit, which converts the model's weights to a lower-precision format. This compression further reduces the memory footprint and speeds up matrix multiplications, enabling us to train with a larger batch size on the resource-constrained GPU.

This combination of techniques made fine-tuning a state-of-the-art model feasible on consumer-grade hardware, directly addressing the core resource limitations of the challenge.

Our training strategy was designed to achieve the best performance possible on a single, resource-constrained NVIDIA T4 GPU.

### Memory and Resource Optimization

To fit the model within the 16 GB GPU memory limit, we used several key techniques. We enabled **gradient checkpointing** and **16-bit mixed-precision training (fp16)** to drastically reduce memory usage. We used the largest possible batch size that would fit (4) and then used **gradient accumulation** to simulate a more stable effective batch size of 8.

### Stable and Efficient Training

We used a conservative approach to fine-tuning to protect the model's valuable pre-trained knowledge. A low **learning rate** (1e-5) combined with a **warmup** period of 500 steps ensured stable convergence. Given the dataset size and the efficiency of our fine-tuning method, a single **epoch** was sufficient to reach strong performance without overfitting. To guarantee the final model was the best one, we evaluated it regularly during training. The key metric was **Word Error Rate (WER)**, the industry standard for this task. The trainer was set to automatically save only the model checkpoint that achieved the lowest WER on the validation set.

## 7. Inference with Beam Search Decoding

For inference, we used a Hugging Face pipeline with **beam search decoding**. This is a powerful technique for generating the most probable transcription from the model's output. Unlike greedy decoding, which simply selects the single most likely token at each step, beam search maintains a "beam" of the top-k most likely hypotheses at each time step. By exploring multiple promising paths simultaneously, it avoids premature errors and is more likely to find the globally optimal transcription, leading to a lower Word Error Rate (WER). We configured the pipeline with num_beams=3 and repetition_penalty=1.2 to balance decoding accuracy and speed.

## 8. Results and Conclusion

The final solution achieved a public leaderboard score of **18.22 WER** and a private score of **17.81 WER**, demonstrating its strong performance on the test data. The entire test dataset of 4089 files was transcribed in **1 hour, 24 minutes, and 29 seconds**, which translates to a high-speed inference of approximately **1.24 seconds per audio file**.

- **Final Metrics**:
    - **WER**: **17.81**
    - **Average Inference Time**: **~1.24 seconds per audio file**

Our solution successfully combines a strategically chosen dataset, a dynamic augmentation strategy, and advanced fine-tuning techniques to create a highly efficient and accurate Swahili ASR model. This approach proves that high-performance AI for low-resource languages can be developed and deployed on edge devices, unlocking new possibilities for digital inclusion and privacy.