Adaptive AI Podcast Clip Generator - Wizard of POdz

Modeling and Scaling of Generative AI Systems – Project Proposal

Sebastian Käslin Kaushik Raghupathruni

November 16, 2025

Contents

1 Problem Description

The problem we want to work on is a common one when designing a generative AI system, i.e., the trade-off between output quality and inference latency. In particular, the **tail latency (P99)**—the 99th percentile of response times—often becomes a critical factor, when considering high-load conditions and aiming at ensuring a positive overall user experience. Our aim is to make the user experience the best possible, however, resources costs need to be also taken into consideration.

Our proposition is an **Adaptive AI Podcast Clip Generator - Wizard of POdz**, that as the name suggests, let you enter in a fantasy world where you can generate podcast audio clips on different topics and with special guests. The system generates the audio clips through two main stages:

1. **Script Generation**: via a large language models (LLMs)
2. **Text-to-Speech (TTS)**: via neural speech synthesis models

To reduce latency spikes, we will implement **adaptive model switching** based on real-time system load. The idea is that during light traffic, a high-quality combination of models will be used; under heavy load, a faster, lower-latency models combination will take over. The main goal is to minimize P99 latency while maintaining a good output quality in the generated content.

2 Experiment Setup and Pipeline Design

The experimental system consists of a two-stage pipeline deployed on a GPU-enabled platform (PyTorch + HuggingFace Transformers).

Stage	**High-Quality Model (Baseline)**	**Low-Latency Model (Adaptive)**
Script Generation (LLM)	Llama 3.1 8B Instruct (4-bit) [3]	Gemma 2 2B Instruct [2]
Text-to-Speech (TTS)	Coqui XTTS v2 (emotional) [1]	Microsoft SpeechT5 (fast) [4]

Table 1: Models used in the two-stage inference pipeline.

The models in Table 1 represent examples of possible models that will be used, experiments will take place in order to empirically compare models' quality and latency.

Adaptive Model Switching: Requests arrive into a central queue shared by the two stages. The system dynamically chooses between the high-quality and low-latency models for *both* LLM and TTS components based on real-time load measurements. The goal is to maintain service responsiveness (low tail latency) while preserving quality in the generated content.

Per-Stage Thresholds. Two independent queue thresholds are defined:

- N_{LLM}: Threshold controlling the switch between Llama 3.1 (8B) and Gemma 2 (2B).
- N_{TTS}: Threshold controlling the switch between Coqui XTTS and SpeechT5.

When the queue length or GPU utilization of a stage exceeds its threshold, the corresponding stage switches to the fast model; once the queue decreases below a lower bound (hysteresis), it reverts to the high-quality model. This prevents oscillations and ensures stability under variable load.

Joint Model Pairing. Because the two stages are sequential, the output rate of the LLM directly determines the arrival rate for the TTS stage. To capture this dependency, a joint decision policy is also explored. The system can select one of four possible model combinations:

$$\{(HQ_{\text{LLM}}, HQ_{\text{TTS}}), (HQ_{\text{LLM}}, Fast_{\text{TTS}}), (Fast_{\text{LLM}}, HQ_{\text{TTS}}), (Fast_{\text{LLM}}, Fast_{\text{TTS}})\}.$$

and each combination has different latency, cost, and quality characteristics.

3 Exploration of Experiments

We will vary several input parameters and collect performance metrics for each configuration. The experiments are designed to evaluate how adaptive switching across both stages (LLM and TTS) impacts overall latency, throughput, and quality under different load conditions. We will consider a dataset of different unique podcast topics to evaluate the system.

Input Parameters

- **LLM settings:** Model type (*Llama 3 8B* vs. *Gemma 2 2B*), temperature, and maximum tokens.
- **TTS settings:** Model type (*Coqui XTTS* vs. *SpeechT5*), voice style, and inference steps.
- **Batch size:** 1 vs. 4 concurrent requests.
- **Load:** Low (1 req/min), medium (4 req/min), high (8 req/min).
- **Per-stage thresholds:**
  - $N_{\text{LLM}} \in \{2, 5, 10\}$
  - $N_{\text{TTS}} \in \{2, 5, 10\}$
- **Switching policy:**
  - Static (always high-quality)
  - Per-stage adaptive (independent thresholds)
  - Joint adaptive (global decision on model pair)

Performance Metrics

- **End-to-End Latency:** P50, P95, and P99 latency measured across both stages.
- **Per-Stage Latency:** Average and tail latency for LLM and TTS individually.
- **Throughput:** Number of requests successfully processed per minute.
- **GPU Utilization:** Real-time memory and compute usage for each stage.
- **Switching Behavior:** Frequency of model switches and hysteresis stability.
- **Quality Metrics:**
  - **LLM-as-a-Judge Score:** 1–10 coherence and creativity rating by an independent LLM evaluator.
  - **Audio Quality (MOS):** Human-rated mean opinion score (1–5 scale).
- **Composite Score:** A weighted objective function combining latency, quality, and cost:

$$C = \alpha \cdot \text{P99}_{\text{lat}} + \beta \cdot (1 - Q_{\text{score}}) + \gamma \cdot \text{Cost},$$

  where α, β, γ are tunable parameters balancing responsiveness, perceptual quality, and resource efficiency.

4 Predictive Analysis and Modeling

This part of the project focuses on developing a predictive model to estimate end-to-end latency and throughput under varying system loads and model configurations. The objective is to bridge the gap between analytical predictions and empirical measurements, allowing the definition of adaptation policies.

Expected Outcome

The outcome of this analysis is a calibrated latency model that can accurately predict tail latency under varying load conditions and guide dynamic switching decisions in real time.

References

[1] Coqui. Coqui xtts v2 model card. https://huggingface.co/coqui/XTTS-v2, 2024. Accessed: 2025-10-21.

[2] Google. Gemma 2 2b instruct model card. https://huggingface.co/google/gemma-2-2b-it, 2024. Accessed: 2025-10-21.

[3] Meta AI. Llama 3.1 8b instruct model card. https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct, 2024. Accessed: 2025-10-21.

[4] Microsoft. Speecht5 text-to-speech model card. https://huggingface.co/microsoft/speecht5_tts, 2025. Accessed: 2025-10-22.