Modeling and Scaling of Generative AI Systems
@ University of Neuchatel Fall 2025

Adaptive AI Podcast Clip Generator: Wizard of POdz

Group 08: Sebastian Käslin, Kaushik Raghupathruni

Contents

1. Project Description

Goal: Build an AI system that generates **podcast clips** using:

1. **LLM-based Script Generation (content creation)**
2. **TTS-based Audio Synthesis (audio generation)**

Models: No Fine-Tuning is performed. All models are used in inference only mode.

Request Characteristics:

1. **Podcast Topic:** Sampled from ~50 themes such as technology, politics, sports, fiction.
2. **Script Length**: short, medium, long.
3. **Voice style:** Male, Female, Neutral, Emotional, Excited.

Output:

1. A .wav audio file containing spoken podcast clip about the chosen topic. Guest personality is reflected in the speech.

2. System Modeling

1. **Queueing Model:**
   We treat the LLM and TTS stage as 2 stage service pipeline.
   We model the system as a Tandem Queue (M/M/1→M/M/1)
   where the service rate μ adapts based on the global queue length N
   a. Inflow: Poisson arrival process with rate λ (configurable: 1, 4, 8 req/min).
   b. Nodes: Sequential processing through LLM (Node 1) → TTS (Node 2).

2. **Markov Chain Model (Switching logic):**
   We implement a State-Dependent Service Rate policy.
   The system state is defined by the total Queue Length (N).
   State transitions:
   a. N→N+1: Rate λ (User Request)
   b. N→N−1: Rate μ effective (Service Completion)

   Adaptive Control Policy: The service rate μ bifurcates based on system load:
   The policy defines the service rate μ as a step function of the system state N, utilizing the HQ model for normal operations and the Fast model strictly for avoiding overload

2. System Modeling

3. **Predictive Regression Model:**
We train a regression model to directly predict end-to-end latency using runtime features. The switching logic looks at Predicted Latency (from the regression model) to decide when to switch.

Inputs:
- LLM Choice (HQ/Fast)
- TTS Choice (HQ/Fast)
- Token Count
- GPU utilisation
- Current Queue Length
- Batch size
- Request arrival rate

Outputs:
- Predicted end to end latency.

3. Performance of LLM models

Experiment Settings: (https://github.com/rkaushik97/MSGAI-AI-Podcast-Generator/blob/main/LLM_QPS.ipynb)

- **Hardware**: Nvidia RTX 4090
- **Max New Tokens**: 150
- **Warm up**: Yes
- **Number of Runs per setting**: 10
- **Models tested**:
  - Llama 3.1 8B Instruct (High Quality/Heavy Model)
  - Gemma 2 2B Instruct (Fast Model/Light Model)
- **Batch sizes tested**: 1, 5 [We do batching to take advantage of parallelism]

Model	Batch Size	Latency (seconds)	Queries per second	Tokens per second
Llama 3.1 8B Instruct	1	2.98	0.34	50.4
	5	3.51	1.43	213
Gemma 2 2B Instruct	1	1.75	0.57	85
	5	1.90	2.70	400

4.Performance of TTS models

Experiment Settings: (https://github.com/rkaushik97/MSGAI-AI-Podcast-Generator/blob/main/tts_testing.ipynb)

- **Hardware**: NVIDIA GeForce GTX 1650 (4GB)
- **Warm up**: Yes
- **Dataset:** The LJ Speech Dataset (short audio clips of a single speaker)
- **Models tested**:
  - SpeechT5 TTS (High Quality/Heavy Model)
  - Massively Multilingual Speech (MMS) (Fast Model/Light Model)
- **Batch sizes tested**: 1, 5, speechT5 TTS does not support batch mode

Model	Batch Size	Average Latency (seconds)	Average Clip duration (seconds)	Throughput (second of audio generated per sec)
SpeechT5 TTS (no batch mode available)	1	3.58	3.91	1.09
	5	-	-	-
Massively Multilingual Speech (MMS): en TTS	1	2.45	6.41	2.64
	5	15.9	8.94	2.81

5.Evaluation Metrics

1. **Latency Metrics**
   - P50 (median)
   - P99 latency (tail latency)
2. **Throughput Metrics**
   - Completed podcast clips per minute
3. **Quality Metrics**
   - LLM-as-a-judge score (1–10) for script coherence
   - Simple MOS-like (1–5) subjective audio rating
4. **GPU Utilisation**
   - Compute Load
   - VRAM utilisation

Thank you