B站/公众号/YouTube：唐国梁Tommy

GLM-TTS 实战指南

智谱AI开源的**高质量文本转语音**合成系统
基于大语言模型的零样本语音克隆技术

零样本克隆 | 3-10秒音频 | 情感表达 | 中英混合

开始学习

01 第 01 章

项目概览

GLM-TTS 是由智谱AI开源的基于大语言模型的高质量文本转语音合成系统

1.1 什么是 GLM-TTS?

GLM-TTS 是由智谱AI开源的**基于大语言模型的高质量文本转语音(TTS)合成系统**，于2025年12月11日正式开源。该项目的核心亮点在于：

- **零样本语音克隆**
  仅需 3-10 秒的音频样本即可克隆任意说话人的声音
- **强化学习增强**
  通过多奖励RL框架（GRPO算法）实现更自然的情感表达
- **流式推理支持**
  适用于实时交互场景
- **音素级精细控制**
  解决多音字和生僻字的发音问题

1.2 解决传统TTS三大核心挑战

音色克隆成本高

需要特定说话人的大量训练数据(数小时级别)，无法快速适配新说话人

情感表达平淡

自回归生成的语音韵律单调，缺乏真实人类的情感起伏

发音精准度低

多音字(如"行")、生僻字自动推理错误率高，无法满足专业场景

1.3 核心价值

零样本克隆
3-10秒音频即可克隆声音
无需长时间录音或微调

RL情感优化
多奖励强化学习框架
CER从1.03降至0.89

流式推理
实时生成音频流
适用于对话、直播等场景

音素控制
Phoneme-in机制
精准控制多音字发音

多语言支持
主要中文，支持中英混合
灵活应对多语言场景

高质量输出
商业级音质
SIM相似度76.4

1.4 性能评估（seed-tts-eval 中文测试集）

模型	CER ↓	SIM ↑	开源
GLM-TTS (基础版)	1.03	76.1	✅
GLM-TTS_RL (强化学习版)	**0.89**	76.4	✅
VoxCPM	0.93	77.2	✅
IndexTTS2	1.03	76.5	✅
CosyVoice2	1.38	75.7	✅

注：CER越低越好，SIM越高越好

1.5 典型应用场景

有声读物制作
精准控制发音，处理生僻字

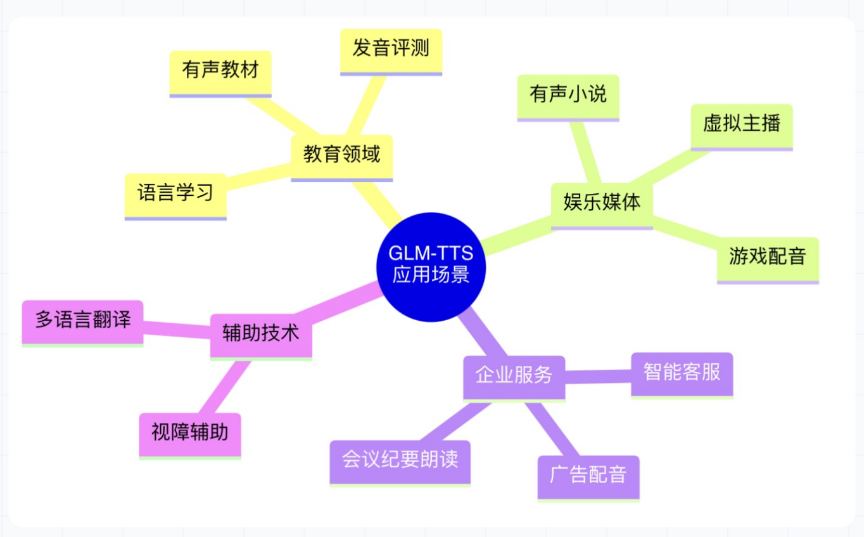
教育培训
多音字发音评测、口语教学

智能客服
实时对话，情感化表达

视频配音
快速克隆声音，批量生成

游戏NPC语音
动态生成角色对话

播客制作
高质量语音合成



02 第 02 章

环境准备与安装

配置开发环境，安装必要的依赖包

2.1 系统要求

组件	最低要求	推荐配置
操作系统	Linux / macOS / Windows	Linux (Ubuntu 22.04)
Python	3.10 - 3.12	Python 3.12
GPU	NVIDIA GPU (6GB+ VRAM)	NVIDIA RTX 5090 (32GB)
CUDA	CUDA 11.8+	CUDA 12.8
内存	16GB RAM	32GB+ RAM
存储	50GB 可用空间	100GB+ SSD

💡 推荐配置：Ubuntu 22.04 + CUDA 12.8 + RTX 5090 (32GB)

2.2 环境安装步骤

1 克隆项目

```bash
# 克隆仓库
git clone https://github.com/zai-org/GLM-TTS.git
cd GLM-TTS
```

2 创建虚拟环境

```bash
# 使用 conda 创建虚拟环境
conda create -n glmtts python=3.12 -y
conda init bash && source /root/.bashrc
conda activate glmtts
```

3 安装核心依赖

```bash
# 安装基础依赖
pip install -r requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple

# 必须安装 FFmpeg
conda install -c conda-forge -y ffmpeg
```

03 第 03 章

模型下载与配置

下载预训练模型并配置运行环境

3.1 模型架构说明

模型组件	作用	大小
Tokenizer	文本编码（vq32k-phoneme-tokenizer）	~500MB
LLM	文本→语音Token生成	~7GB
Flow	Token→Mel频谱转换	~1GB
Vocoder	Mel频谱→音频波形	~200MB
Frontend	说话人嵌入提取（campplus.onnx）	~100MB

总计约 ~9GB 存储空间

3.2 下载方式

HuggingFace（科学上网） | ModelScope（国内）

方式一：从 HuggingFace 下载

```bash
# 创建模型目录
mkdir -p ckpt

# 安装 huggingface-cli
pip install -U huggingface_hub

# 下载完整模型
huggingface-cli download zai-org/GLM-TTS --local-dir /root/autodl-tmp/GLM-TTS/ckpt
```

3.3 配置文件说明

configs/ 目录结构

`spk_prompt_dict.yaml` 说话人提示音频字典	`lora_adapter_configV3.1.json` LoRA适配器配置
`G2P_able_1word.json` 单字音素转换表	`G2P_all_phonemes.json` 完整音素列表
`G2P_replace_dict.jsonl` 音素替换字典	`custom_replace.jsonl` 自定义替换规则

💡 无需修改配置即可使用基础功能，可根据需求调整

04 第 04 章

快速实践

三种使用方式：命令行、Shell脚本、Web界面

命令行推理 | Shell脚本 | Web界面

1 基础用法

```bash
python glmtts_inference.py \
    --data=example_zh \
    --exp_name=_test \
    --use_cache
```

参数说明：

`--data`：示例数据文件名（不含.jsonl后缀）
`--exp_name`：实验名称，输出目录前缀
`--use_cache`：启用缓存，加速推理

2 启用音素控制

```bash
python glmtts_inference.py \
    --data=example_zh \
    --exp_name=_test_phoneme \
    --use_cache \
    --use_phoneme
```

> *use_phoneme 控制是否启用音素（phoneme）转换，用于将文本转换为音素表示。*

处理流程：

- 文本规范化
- 如果启用，进行 G2P 转换（将汉字转为拼音音素，如 "你好" → "n|i-h|ao"）
- 使用转换后的音素文本进行后续的语音合成

主要作用：

音素控制（Phoneme-in）用于解决多音字和生僻字的发音歧义，提升发音准确性。

工作原理：

- 全局 G2P 转换
  将输入文本转换为完整音素序列
- 动态替换
  使用 G2P_replace_dict.jsonl（替换字典）指定多音字/生僻字的音素
  使用 G2P_able_1word.json（白名单）决定哪些字保留原文本，哪些替换为音素
  不在白名单中的字会被替换为音素表示
- 混合生成
  将替换后的音素与保留的文本组合，形成混合输入
  例如："你好" → "<|N|><|I|>好"（只替换"你"，保留"好"）

3 指定采样率

```bash
python glmtts_inference.py \
    --data=example_zh \
    --exp_name=_test_sample_rate \
    --sample_rate=24000 \
    --use_cache
```

`sample_rate`（采样率）指定音频的采样频率，单位为 Hz，表示每秒的采样点数。

主要作用：

- 影响音频特征提取：采样率影响梅尔频谱的计算，进而影响特征表示；
- 控制输出音频的采样率：`torchaudio.save(wav_save_path, tts_speech, sample_rate)`
- 影响声码器选择：24000 Hz：使用 HiFT 声码器，32000 Hz：使用 Vocos 声码器

使用建议：

- 24000 Hz：常用，文件更小，处理更快
- 32000 Hz：音质更好，文件更大，处理更慢

4.4 输入数据格式

JSONL 格式示例

```json
{"uttid": "0", "prompt_text": "他当时还跟线下其他的站姐吵架，然后，打架进局子了。", "prompt_speech": "examples/prompt/jiayan_zh.wav", "syn_text": "我看了
{"uttid": "1", "prompt_text": "他当时还跟线下其他的站姐吵架，然后，打架进局子了。", "prompt_speech": "examples/prompt/jiayan_zh.wav", "syn_text": "每次熬
```

`uttid` 唯一标识符	`prompt_text` 提示音频的文本内容
`prompt_speech` 提示音频文件路径	`syn_text` 要合成的目标文本

05 第 05 章

核心功能

深入了解 GLM-TTS 的核心技术能力

5.1 零样本语音克隆

工作原理

1. **说话人嵌入提取**
   使用 CAM++ 模型从提示音频中提取 192 维说话人特征向量
2. **语音Token提取**
   Speech Tokenizer 将音频量化为离散Token序列
3. **条件生成**
   LLM 基于提示音频的Token和说话人特征生成目标文本的语音Token
4. **音频重建**
   Flow 模型和 Vocoder 将Token转换为音频波形

5.2 音素级精细控制（Phoneme-in）

应用场景

多音字 行（xíng / háng）、重（zhòng / chóng）、参（cān / shēn / cēn）

生僻字 龘（dá）、靐（bìng）、㗊（kā）

专有名词 人名、地名的特殊读音

配置替换字典

编辑 `configs/custom_replace.jsonl`：

```json
{"word": "行", "pinyin": "háng"}
{"word": "参差", "pinyin": "cēn1 cī1"}
{"word": "单于", "pinyin": "chán2 yú2"}
```

5.3 情感与韵律控制

基础模型的情感表达

GLM-TTS 基础模型已具备一定情感表达能力，主要通过提示音频的情感迁移和文本的情感暗示实现。

```json
# 欢快情绪
{"uttid": "0", "prompt_text": "哇！太棒了！", "prompt_speech": "happy_voice.wav", "syn_text": "今天是个好日子！"}

# 平静叙述
{"uttid": "1", "prompt_text": "今天天气不错。", "prompt_speech": "calm_voice.wav", "syn_text": "公司宣布了新的政策。"}

# 疑问语气
{"uttid": "2", "prompt_text": "这是真的吗？", "prompt_speech": "question_voice.wav", "syn_text": "你确定要这样做吗？"}
```

强化学习增强模型（GLM-TTS_RL）

通过 GRPO 算法优化的模型提供更强的情感控制：

笑声奖励
自动在合适位置添加笑声

情感奖励
增强情感表达强度

韵律奖励
优化语调和停顿

06 第 06 章

架构解析

深入了解 GLM-TTS 的技术架构和实现原理

6.1 技术栈

技术分类	具体技术	版本/规格	选型理由
深度学习框架	PyTorch	2.3.1	业界标准，动态图灵活性高
大语言模型	Llama	Causal LM	开放架构，社区生态丰富
位置编码	RoPE	-	相对位置编码，支持长序列
参数高效微调	LoRA	PEFT 0.7.0	低资源微调，训练成本降低95%
Flow模型	DiT	自研	生成质量优于GAN，训练稳定
声码器	Vocos / HiFiGAN	32kHz / 24kHz	Vocos速度快，HiFiGAN质量高
强化学习	GRPO	自研	无需Critic网络，稳定性强
分布式训练	DeepSpeed	0.14.5	ZeRO优化，节省显存80%

6.2 架构范式

两阶段级联生成架构

GLM-TTS 采用 **两阶段级联生成架构（Two-Stage Cascade Generation）**，结合自回归语言模型与流匹配扩散模型。

第一阶段：语言模型
- 文本分词（Tokenizer）
- 语音Token提取（SpeechTokenizer）
- 说话人嵌入提取（CamPlus ONNX）
- LLM自回归生成

第二阶段：声学模型
- Token Embedding
- DiT推理（10 timesteps ODE）
- Flow Matching生成Mel频谱
- Vocoder音频重建

07 第 07 章

总结

GLM-TTS 的技术价值与竞品对比

7.1 技术价值总结

1 架构创新
双阶段生成 + 强化学习对齐，兼顾音质与情感

2 算法创新
GRPO多奖励优化，无需Critic网络，训练稳定性高

3 工程创新
流式推理 + LoRA微调 + 混合音素控制，易用性强

CER	零样本克隆	LoRA微调
0.89	**3-10秒**	**5%**
开源最佳	音频即可	参数训练

7.2 与竞品对比

维度	GLM-TTS	Seed-TTS	CosyVoice	F5-TTS	MiniMax
开源	✓	✗	✓	✓	✗
CER	**0.89**	1.12	1.12	1.53	**0.83**
情感控制	RL优化	⚠ 有限	⚠ 有限	✗	✓
流式推理	✓	✗	⚠ 部分	✗	✓
音素控制	混合输入	✗	G2P替换	✗	✓

B站/公众号/YouTube：唐国梁Tommy

https://tgltommy.com