# Tortoise-TTS 详解

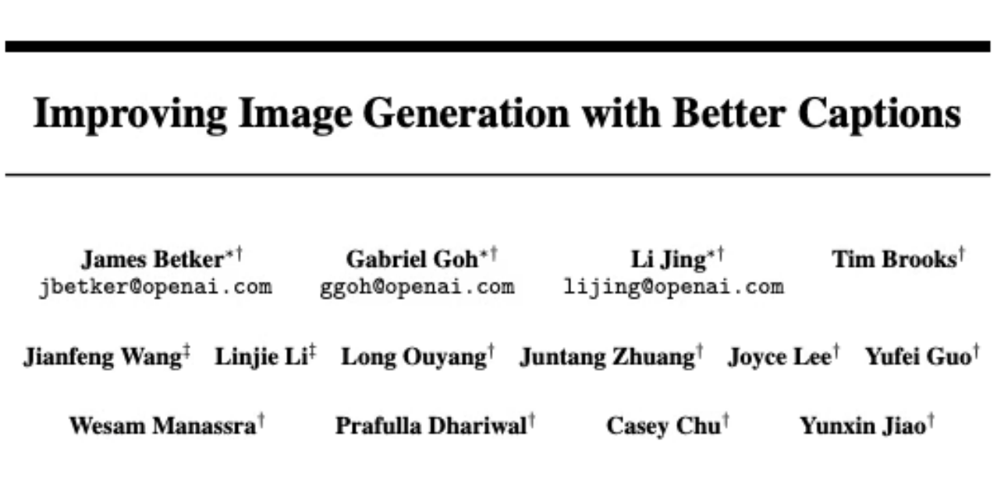
- 论文题目：**Better speech synthesis through scaling**
- 论文链接：https://arxiv.org/pdf/2305.07243
- 开源地址：https://github.com/neonbjb/tortoise-tts（⭐ 14k+）
  - 额外开源：https://github.com/152334H/tortoise-tts-fast (⭐ 800+)

## 作者简介

- **James Betker** 是 OpenAI 的研究工程师，主要研究图像和音频的生成式建模
  - 在加入 OpenAI 之前，James Betker 在 GARMIN 国际航电瑞士有限公司担任了 12 年的软件工程师，并曾在谷歌担任软件工程师。

**近期核心工作：**

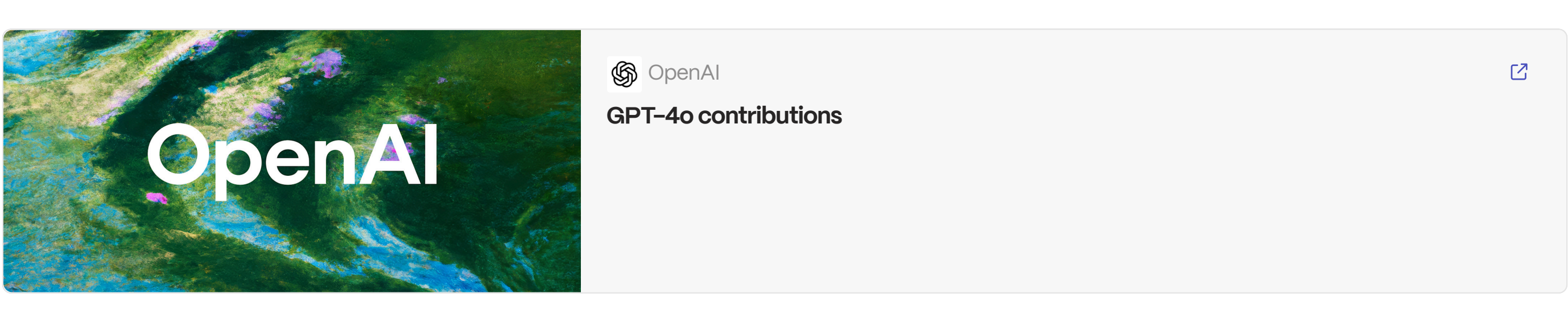
- DALL-E 3 一作：https://cdn.openai.com/papers/dall-e-3.pdf



**Improving Image Generation with Better Captions**

James Betker*† jbetker@openai.com
Gabriel Goh*† ggoh@openai.com
Li Jing*† lijing@openai.com
Tim Brooks†
Jianfeng Wang‡ Linjie Li‡ Long Ouyang† Juntang Zhuang† Joyce Lee† Yufei Guo†
Wesam Manassra† Prafulla Dhariwal† Casey Chu† Yunxin Jiao†

- GPT-4o 核心贡献者，图像/音频 Lead 之一



OpenAI
**GPT-4o contributions**

## 模型结构

### 灵感来源

- DALL-E: Zero-Shot Text-to-Image Generation (https://arxiv.org/pdf/2102.12092)
- zero-shot 图像生成：将描述文本与视觉 Token（dVAE）拼接作为一条训练样本
- DALL-E 使用 CLIP (Contrastive Model) 作为 Reranking 的分数来源

### Tortoise-TTS 结构图

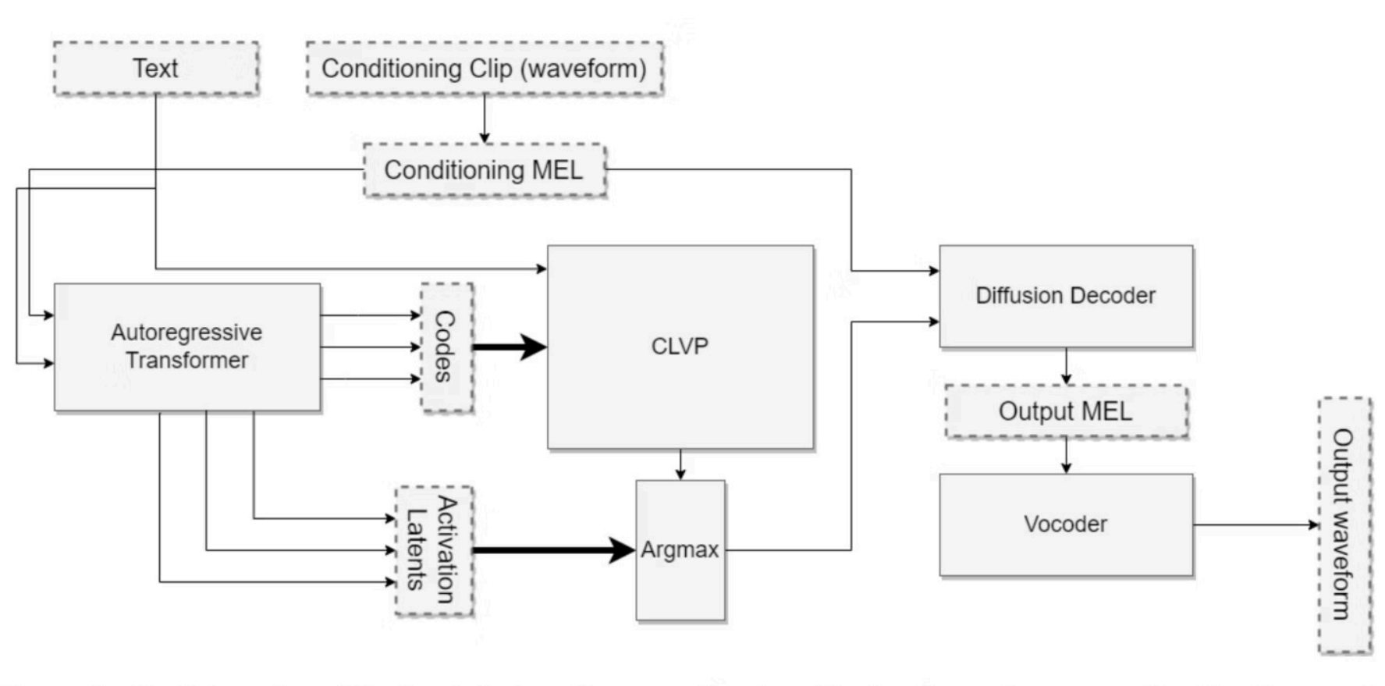
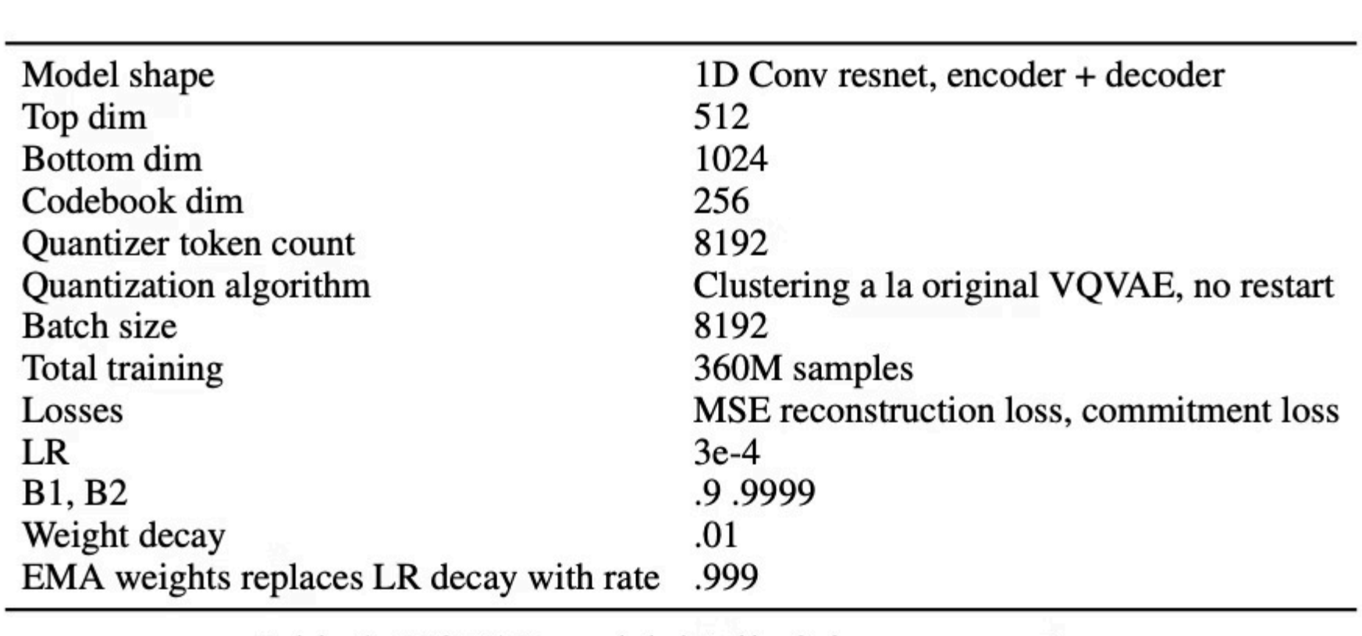


Figure 1: TorToise-v2 architectural design diagram. Inputs of text and a reference audio clip (for speaker cloning) flow through a series of decoding and filtering networks to produce high-quality speech.

## 模块详解

### 基础准备一：Text/Speech Tokenizer

- Text Tokenizer 使用英文的 BPE
- Speech Tokenizer 使用 Mel Token：在梅尔特征上进行**单层 VQ-VAE** 训练，4 倍下采样，单层 Codebook 大小为 8192
  - 训练要点：batch size 越大，越容易训练重构出清晰的音频
  - 训练数据：音频采样率 22050Hz，单条样本为 40960 个采样点
  - 训练配置如下：



| | |
|---|---|
| Model shape | 1D Conv resnet, encoder + decoder |
| Top dim | 512 |
| Bottom dim | 1024 |
| Codebook dim | 256 |
| Quantizer token count | 8192 |
| Quantization algorithm | Clustering a la original VQVAE, no restart |
| Batch size | 8192 |
| Total training | 360M samples |
| Losses | MSE reconstruction loss, commitment loss |
| LR | 3e-4 |
| B1, B2 | .9 .9999 |
| Weight decay | .01 |
| EMA weights replaces LR decay with rate | .999 |

Table 1: VQVAE model details & hyperparameters

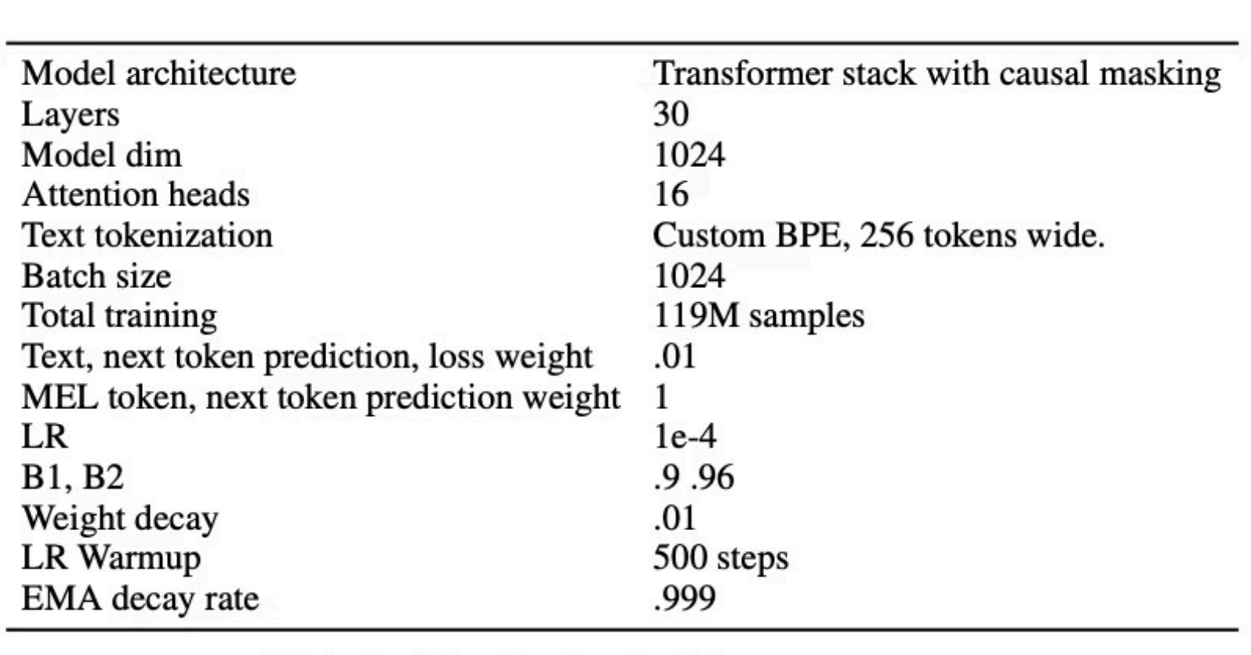
### 基础准备二：Speech Conditioning Input

1. The speech conditioning input starts as **one or more audio clips of the same speaker as the target**. These clips are converted to MEL spectrograms and fed through an encoder consisting of a stack of self-attention layers. The autoregressive generator and the DDPM **have their own conditioning encoders**, both of which are learned alongside their respective networks.
2. The output of these layers is **averaged to produce a single vector.** The vectors from **all of the encoded conditioning clips** are then averaged again before being fed as an input into the autoregressive or conditioning networks.
3. The intuition behind the conditioning input is that it **provides a way for the models to infer vocal characteristics like tone and prosody** such that the search space of possible speech outputs corresponding to a given textual input is greatly reduced.
4. Speech conditioning encodings are learned by a **separate encoder** that takes in the MEL spectrogram of a related clip **(another clip of the same person speaking)** and produces a single vector embedding that is placed at the front of the attention context. Two encodings were produced for each training sample, which are averaged together. The maximum input length to the conditioning encoder is 132,300 samples, or 6 seconds of audio.

- **以上信息总结**
  - 额外的条件输入 clips 是**与目标音频相同 speaker 的不同音频**
  - Encoder 的输入是梅尔特征，采用多层 self-attention 结构，并且最后平均得到**单个向量**（多条 clip 时还要再次取平均）
    - 注意一：**得到的单个向量，被放在 attention context 的最前面**
    - 注意二：LLM 和 DDPM 使用的是不同的 Encoder，各自跟着 LLM/DDPM 一起学习
    - 注意三：Encoder 设置最长输入 6 秒音频
  - 基本出发点：从额外提供的条件输入，来预测声学特性比如音调、韵律等

### 第一部分：LLM

- 功能：Text + Conditioning Mel → LLM → 输出 Mel Codes
- **LLM 模型结构**：GPT-2 (Decoder-Only)，参数量大约 350-400M



| | |
|---|---|
| Model architecture | Transformer stack with causal masking |
| Layers | 30 |
| Model dim | 1024 |
| Attention heads | 16 |
| Text tokenization | Custom BPE, 256 tokens wide. |
| Batch size | 1024 |
| Total training | 119M samples |
| Text, next token prediction, loss weight | .01 |
| MEL token, next token prediction weight | 1 |
| LR | 1e-4 |
| B1, B2 | .9 .96 |
| Weight decay | .01 |
| LR Warmup | 500 steps |
| EMA decay rate | .999 |

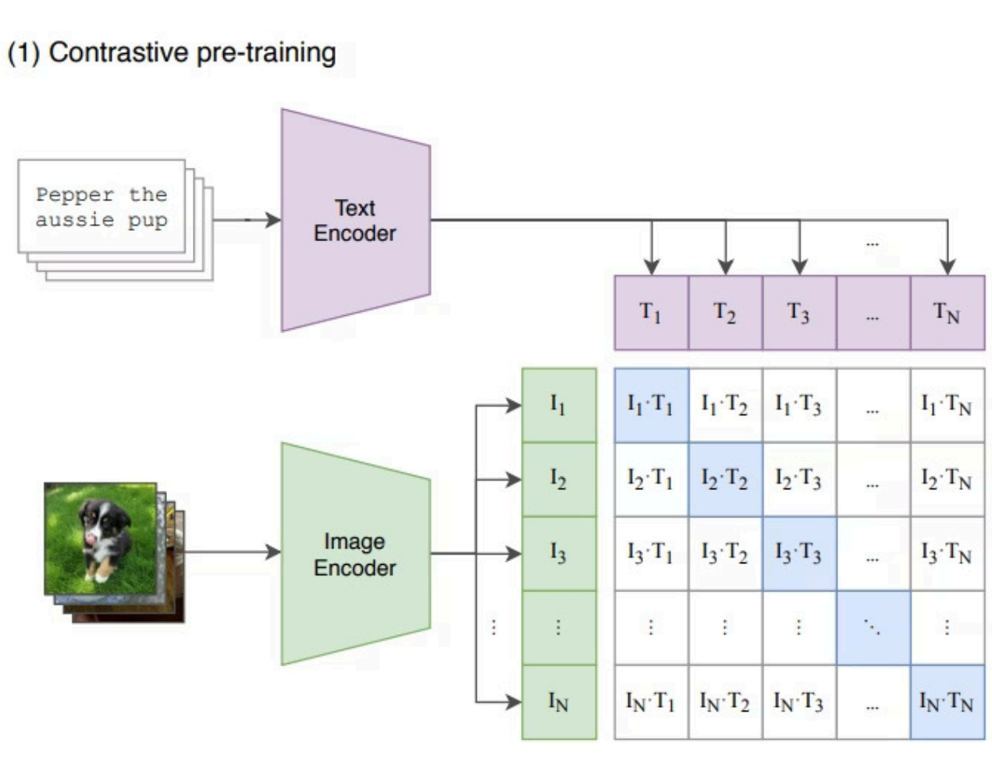
Table 2: AR prior details & hyperparameters

After training the autoregressive decoder to convergence, I fine-tuned it on the clean audio datasets from LibriTTS and HIFITTS.

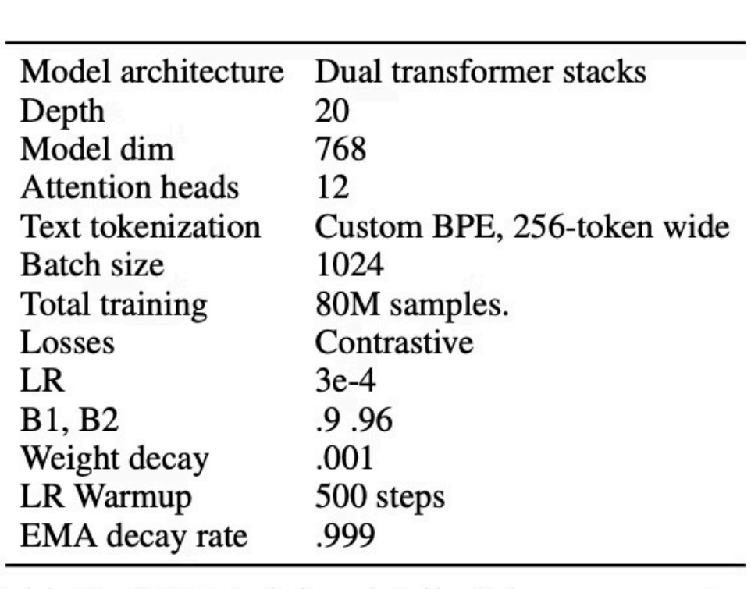
- 训练序列：<SC> <BT> <T> <T> <T> ... <T> <ET> <BM> <M> <M> <M> ... <M> <EM>
  - <SC> speech condition input embedding
  - <BT> <ET> 文本 token 的起始和结束符号
  - <BM> <EM> 语音梅尔 token 的起始和结束符号
  - <T> 文本 token
  - <M> 语音梅尔 token
- **LLM 位置编码**：<T> 和 <M> 各自的可学习位置编码
- LLM 输入音频：最长为 27s （604 个 Mel Token）

### 第二部分：CLVP Reranking

- 功能：Text + Codes → CLVP (Re-ranking)
  - 与 CLIP 是完全一致的思路，只不过对比学习的两路输入是离散 speech code 和相应文本
  - 作用：衡量合成的 Mel Token 与文本之间的距离/相关程度
    - 同一条文本，采样生成不同的 token 序列后，用 CLVP 打分进行排序



- **模型结构及实验配置**
  - 参数量大约 200M



| | |
|---|---|
| Model architecture | Dual transformer stacks |
| Depth | 20 |
| Model dim | 768 |
| Attention heads | 12 |
| Text tokenization | Custom BPE, 256-token wide |
| Batch size | 1024 |
| Total training | 80M samples. |
| Losses | Contrastive |
| LR | 3e-4 |
| B1, B2 | .9 .96 |
| Weight decay | .001 |
| LR Warmup | 500 steps |
| EMA decay rate | .999 |

Table 3: CLVP training details & hyperparameters

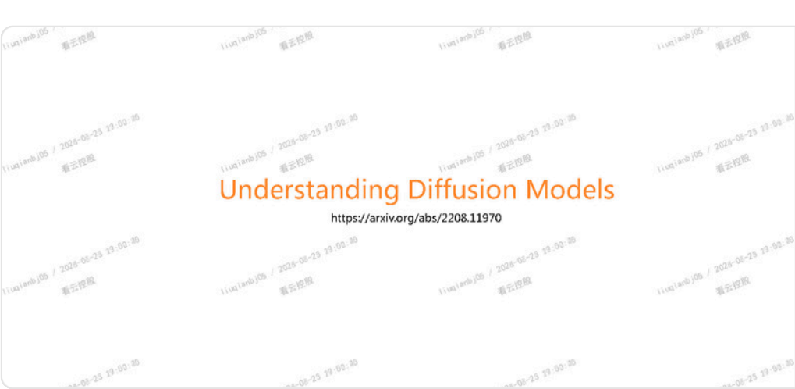
- 参考代码
  - https://github.com/neonbjb/tortoise-tts/blob/main/tortoise/models/clvp.py
  - https://github.com/neonbjb/tortoise-tts/blob/main/tortoise/api.py#L477

### 第三部分：DDPM

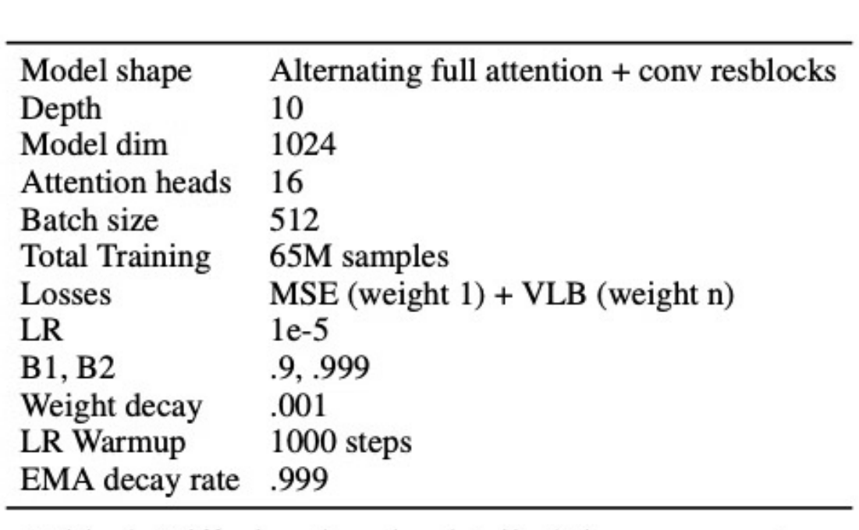
- **功能**：Latents + Conditioning Mel → DDPM → 输出梅尔特征
- **动机**：为什么需要额外的 DDPM 将 token 还原为梅尔?
  - VQ-VAE 解码器的学习效果有限，比如 DALL·E 生成直接用 VAE 解码器，图像经常模糊、不连贯（blurry incoherence）

**模型细节**

- 卷积 + self-attention 层 → 接近 Transformer-Unet 结构，但没有使用 UNet 的上下采样
  - 参数量大约 80M
- **输入信息**：speech conditioning input + time step t + LLM output latents
- **训练数据**：speech conditioning input 最长 5s，目标音频最长 10s
- **梅尔特征**：预测的特征是上采样到 24kHz 后提取出来的，为了适配后续的 UnivNet
- **训练 Loss**：MSE loss 是 VLB 变分下界 的一个近似项，用来实际优化
  - 原始 VLB loss 不如 MSE 简洁有效，故多数实现中优化目标是 MSE，VLB 作为辅助或理论分析用



File upload
**Understanding-Diffusion-Models.pdf**
10.7 MB



| | |
|---|---|
| Model shape | Alternating full attention + conv resblocks |
| Depth | 10 |
| Model dim | 1024 |
| Attention heads | 16 |
| Batch size | 512 |
| Total Training | 65M samples |
| Losses | MSE (weight 1) + VLB (weight n) |
| LR | 1e-5 |
| B1, B2 | .9, .999 |
| Weight decay | .001 |
| LR Warmup | 1000 steps |
| EMA decay rate | .999 |

Table 4: Diffusion decoder details & hyperparameters

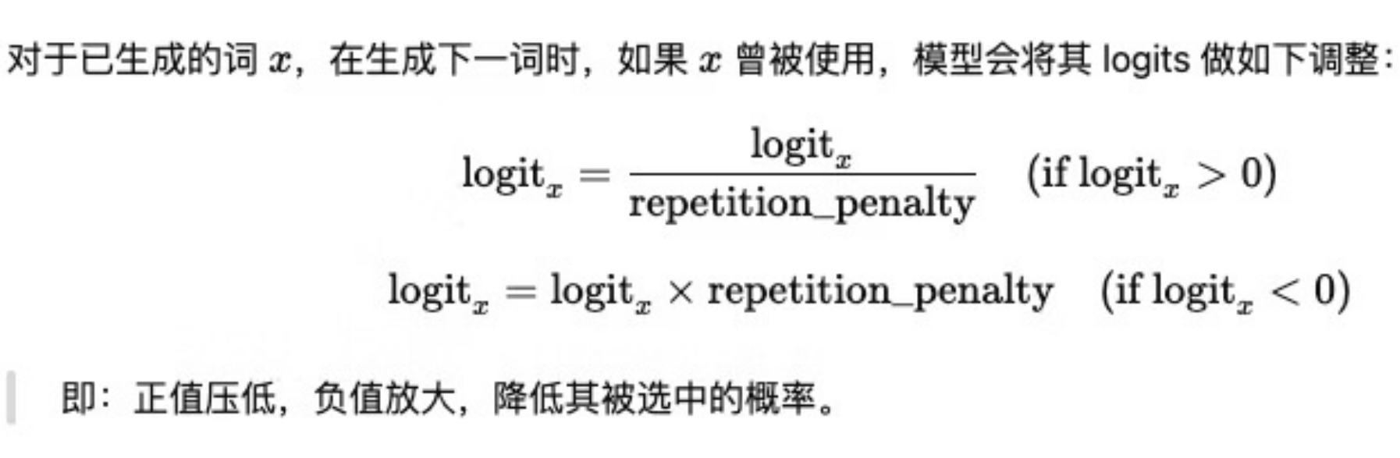
**实验尝试记录**

- DDPM 直接建模音频 PCM（不是梅尔特征）
  - 训练时间长，合成效果不如 mel + vocoder
- **使用 LLM output latents 而不是 mel token**
  - 效果不如 latents，使用 LLM latents 是效果提升最大的改进
- CFG classifier free guidance
  - 训练时，15%的概率丢弃 speech conditioning input 和 LLM output latents
- 未做的猜想：**将待合成文本作为 DDPM 的额外输入**，补充更多语义信息

- **DDPM 的缺点**
  - 输出维度与输入维度相同，输出的维度在开始采样前就必须确定
  - 采样过程时间长，要从 DDPM 中生成一个样本，需要进行数百到数千次的迭代采样
    - 即使模型本身效果好，但每次生成的计算成本和延迟都非常高，影响了实际应用，特别是实时对话、交互式系统

## 模型推理说明

- 将条件输入和文本输入到自回归模型中，并解码生成大量候选语音输出。
  - 自回归模型解码阶段，采用 nucleus sampling（核采样）策略，参数设置为：top P=0.8，重复惩罚=2，softmax 温度=0.8
  - repetition penalty 重复惩罚项的含义



对于已生成的词 $x$，在生成下一词时，如果 $x$ 曾被使用，模型会将其 logits 做如下调整：

$$\text{logit}_x = \frac{\text{logit}_x}{\text{repetition\_penalty}} \quad (\text{if logit}_x > 0)$$

$$\text{logit}_x = \text{logit}_x \times \text{repetition\_penalty} \quad (\text{if logit}_x < 0)$$

> 即：正值压低，负值放大，降低其被选中的概率。

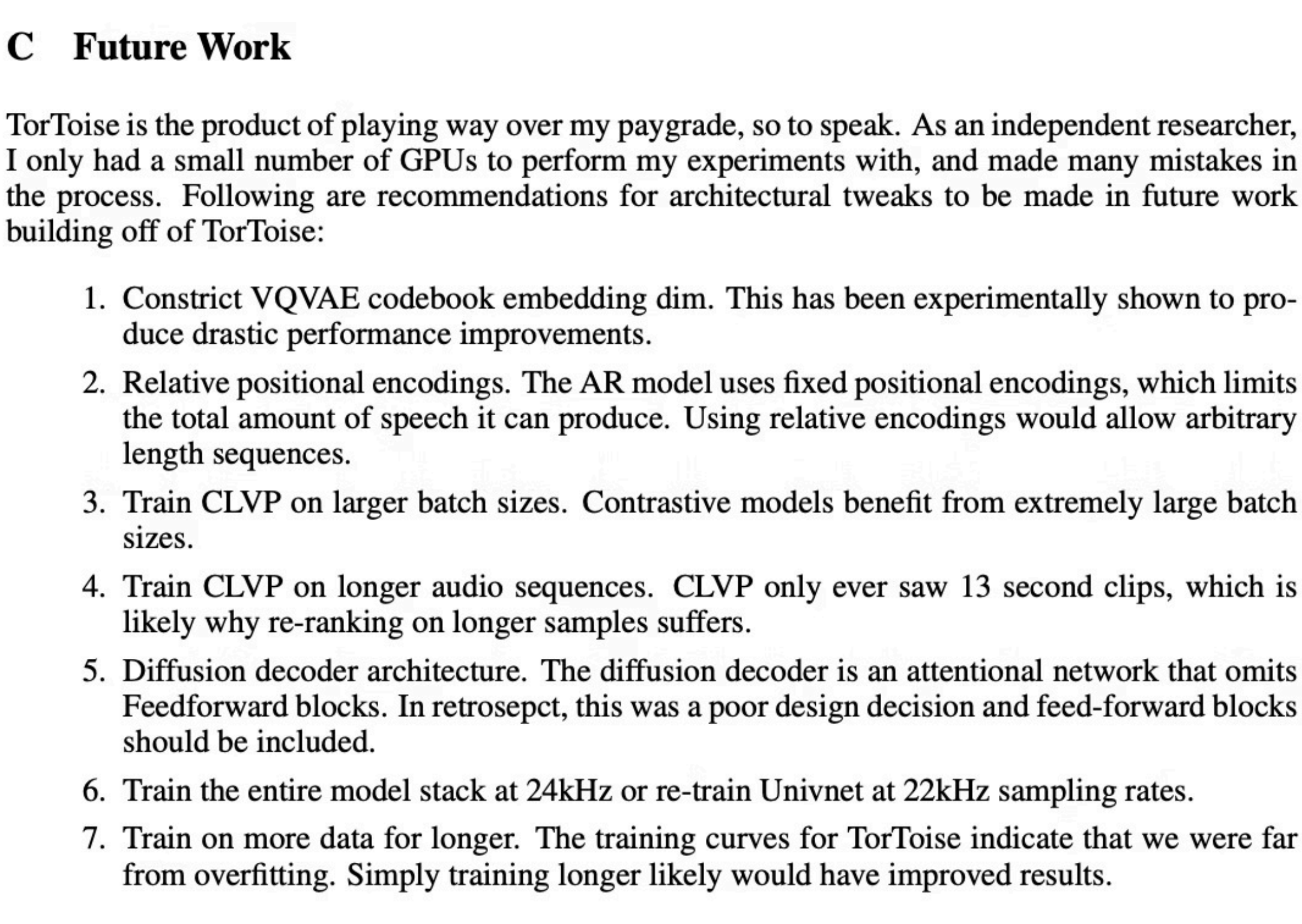
- 使用 CLVP（语音-文本相关性模型）对每个语音候选项与文本之间的相关性打分。
- 选出前 k 个语音候选项，并对每个候选项执行以下操作：
  - 使用 DDPM 解码为 MEL 频谱图
  - 使用传统的声码器 UnivNet 将 MEL 频谱图转换为波形音频
- 推理代码：https://github.com/neonbjb/tortoise-tts/blob/main/tortoise/api.py#L405

## 其他技术细节

- 训练数据量：49000 小时
- 训练数据处理
  - 数据来源：从网上抓取的有声书和播客内容
  - 音频切分：音频数据通过检测 500 毫秒的静音进行切分，并保留了时长在 5 到 20 秒之间的音频片段
  - 音频分类：用于筛除包含背景噪音、音乐、音质差（例如电话录音）、多人同时说话以及混响的音频
  - 语音转写：使用 wav2vec2-large 模型
    - 使用 LibriTTS 和 HiFiTTS 对 ASR 模型进行微调，能够一并识别预测出标点符号（引号、逗号和感叹号）

## 未来工作

受限于独立研究的资源/成本，很多后续潜在的优化点没有展开实验。



### C Future Work

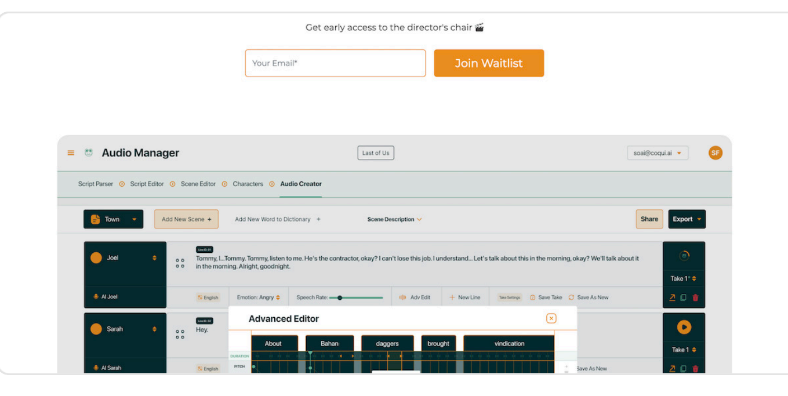
TorToise is the product of playing way over my paygrade, so to speak. As an independent researcher, I only had a small number of GPUs to perform my experiments with, and made many mistakes in the process. Following are recommendations for architectural tweaks to be made in future work building off of TorToise:

1. Constrict VQVAE codebook embedding dim. This has been experimentally shown to produce drastic performance improvements.
2. Relative positional encodings. The AR model uses fixed positional encodings, which limits the total amount of speech it can produce. Using relative encodings would allow arbitrary length sequences.
3. Train CLVP on larger batch sizes. Contrastive models benefit from extremely large batch sizes.
4. Train CLVP on longer audio sequences. CLVP only ever saw 13 second clips, which is likely why re-ranking on longer samples suffers.
5. Diffusion decoder architecture. The diffusion decoder is an attentional network that omits Feedforward blocks. In retrosepct, this was a poor design decision and feed-forward blocks should be included.
6. Train the entire model stack at 24kHz or re-train Univnet at 22kHz sampling rates.
7. Train on more data for longer. The training curves for TorToise indicate that we were far from overfitting. Simply training longer likely would have improved results.

# XTTS 概述

- 论文题目：**XTTS: a Massively Multilingual Zero-Shot Text-to-Speech Model**
- 论文链接：https://arxiv.org/abs/2406.04904
- 开源地址：https://github.com/coqui-ai/TTS（⭐ 41k+）

# 团队介绍



Coqui

**Coqui**

Coqui, Freeing Speech.

docs.coqui.ai

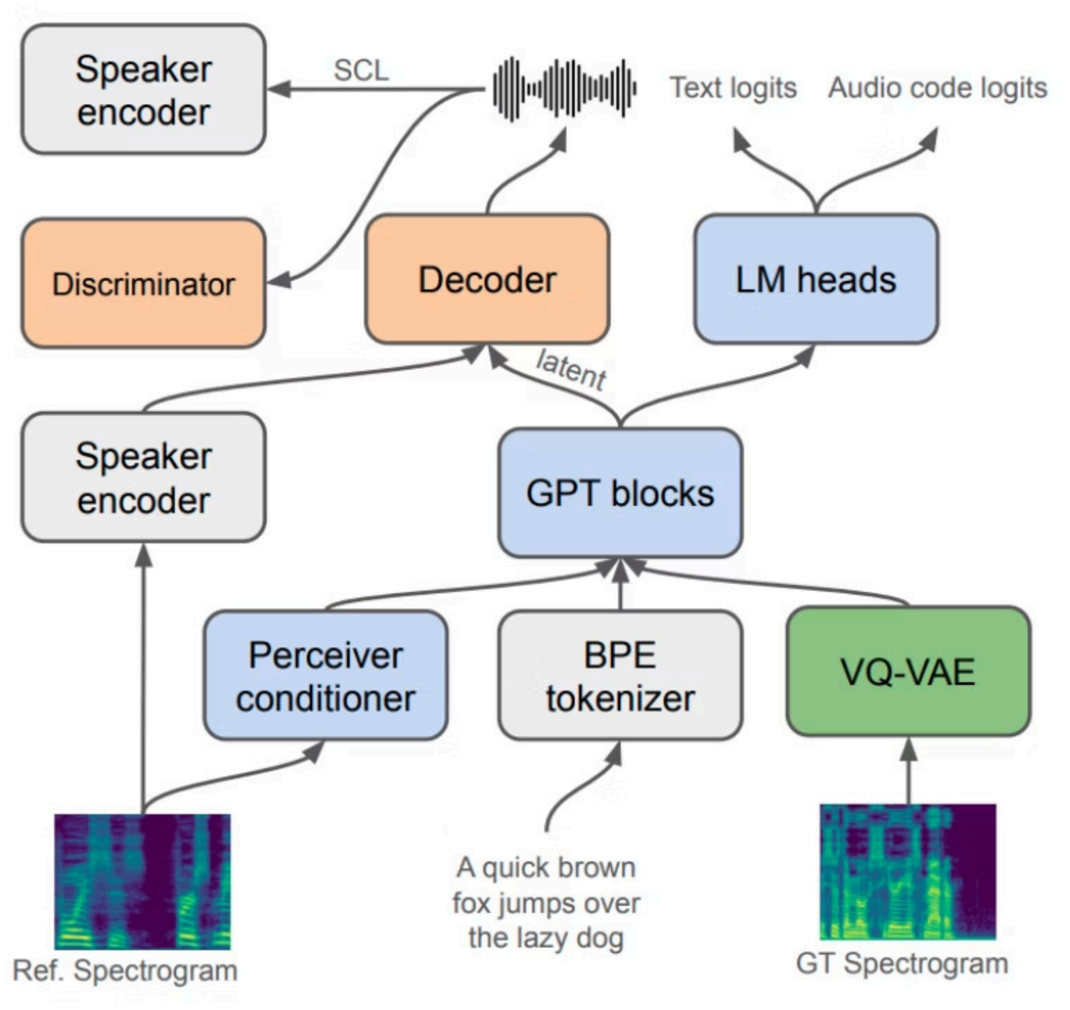
**ⓍTTS – TTS 0.22.0 documentation**

ⓍTTS is a super cool Text-to-Speech model that lets you clone voices in different languages by using just a quick 3-second audio clip. Built on the 🐢Tortoise, ⓍTTS has important model changes that make cross-language voice cloning and multi-lingual speech generation super easy....

| 时期 | 关键事件 |
|---|---|
| 2018–2020 | Mozilla TTS 项目起步，后面临裁员与战略调整 |
| 2021 年 3 月 | Coqui.ai 成立，接盘开源语音项目 |
| 2021–2023 | 快速发展，模型迭代，商业探索与社区扩张 |
| 2023 年 12 月 | 宣布关闭商业 SaaS，转回开源项目 |
| 2024–2025 | 项目持续开源、社区维护，GitHub 库不断更新 |

# 模型结构

- 基本与 Tortoise-TTS 一致



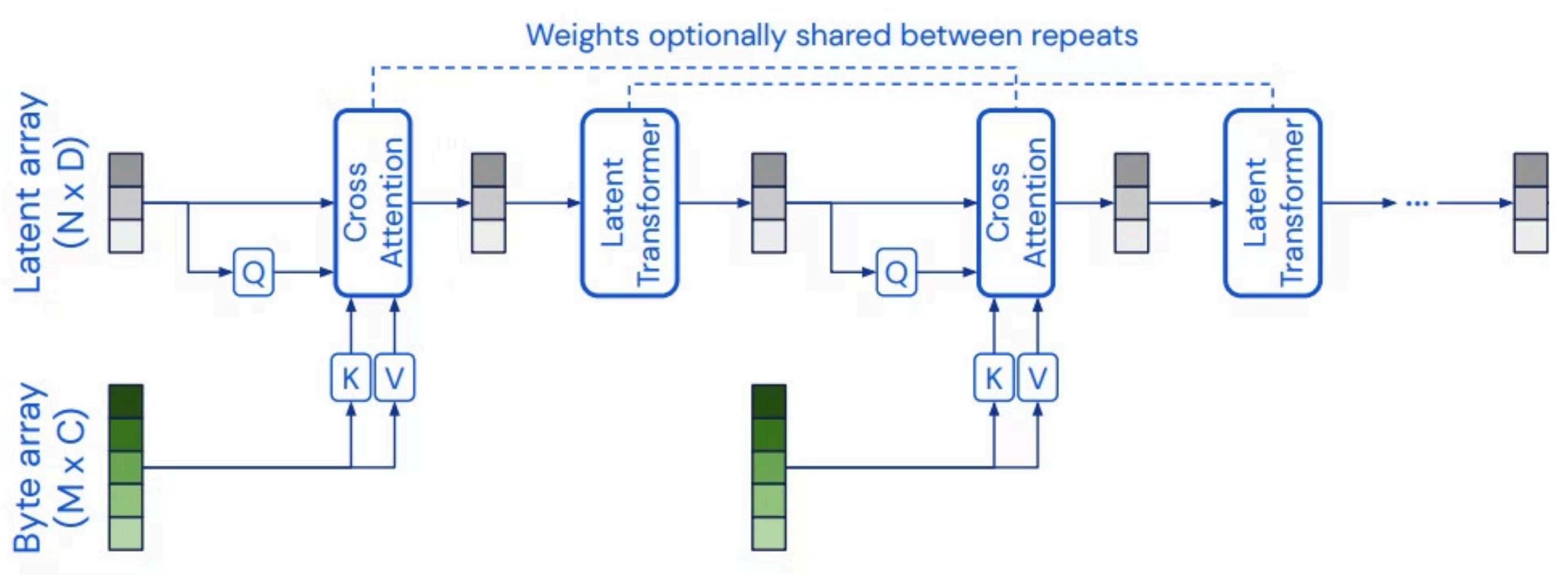
# 模型详解

# VQ-VAE

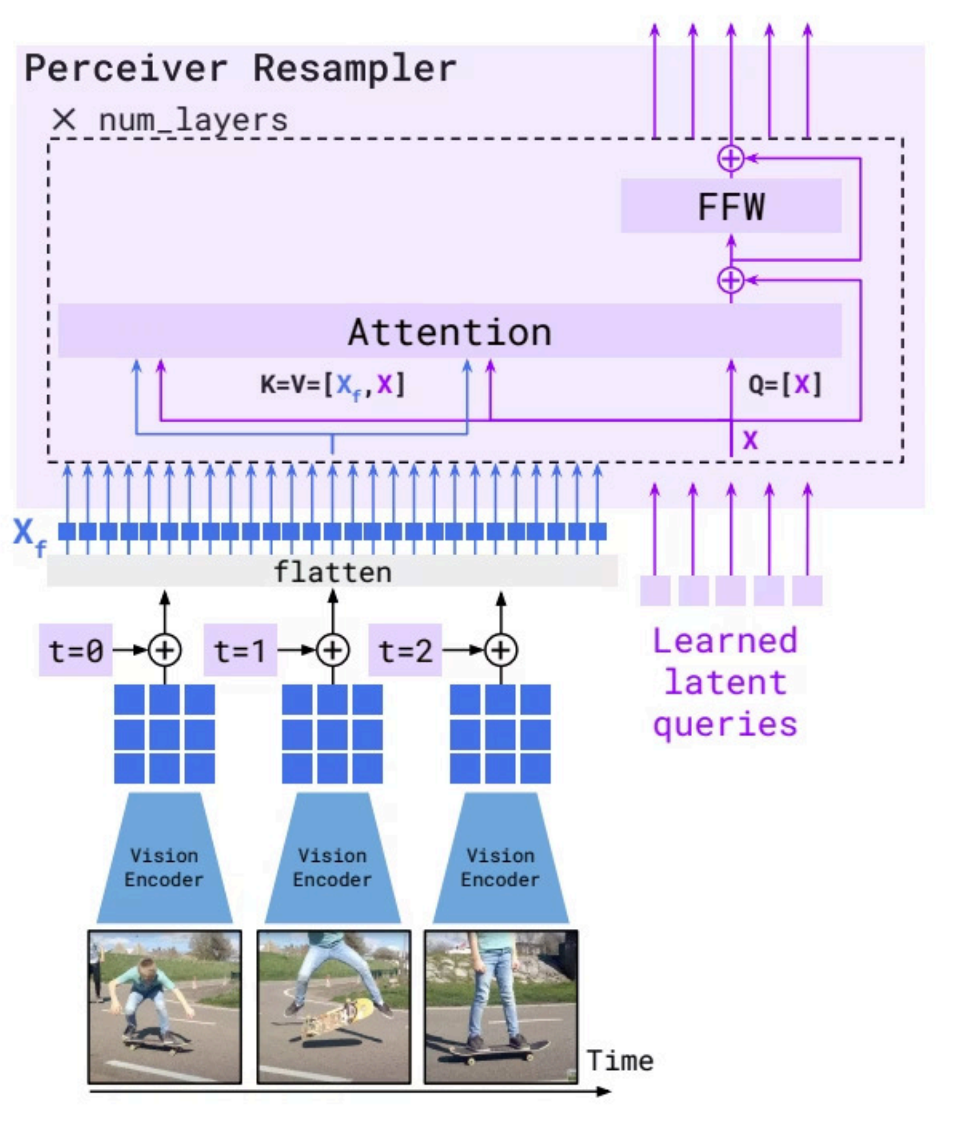
- 方案：语音采样率为 22050 Hz，提取梅尔特征的 hop_size 为 256；VQ-VAE 进行 4 倍下采样：8192 codes，21.53 Hz 帧率。
- 改进点：VQ-VAE 训练完成后，挑选 top 1024 高频的 code，其他 code 过滤不使用（能够达到更好的表现力）
- 说明：直接使用 VQ-VAE 的 Decoder 还原梅尔特征，合成的发音/音质都比较差

# LLM

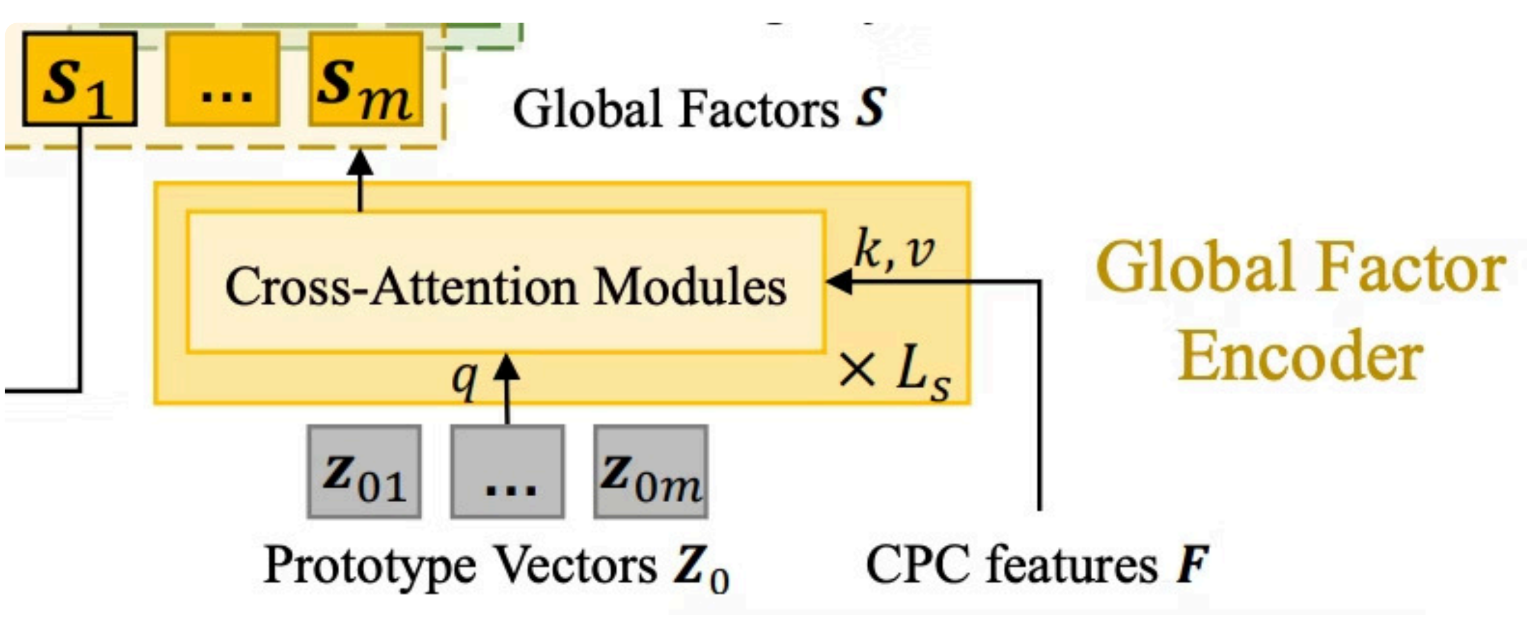
- 模型参数量 443M
- 文本 Tokenizer：BPE token 量级为 6681
- 改进点：Conditioning Encoder，采用 **Perceiver Resampler 结构**
    - 输入是参考音频的梅尔特征，输出是 32 个 1024 维的向量
    - 效果验证：比 Tortoise-TTS 单个 speaker embedding 的声音复刻效果更好
- Perceiver Resampler
    - **Perceiver:** https://arxiv.org/pdf/2103.03206



    - **Perceiver Resampler:** https://arxiv.org/pdf/2204.14198



    - **Retriever-TTS:** https://arxiv.org/pdf/2206.13865



# Decoder

- 输入特征：GPT-2 输出的 latents 表征，**输出直接一步到位到波形**
- 模型结构：使用 HiFiGAN 26M 参数模型
    - 每个上采样模块，都增加额外的 speaker embedding（**预训练好的说话人模型 H/ASP**）
- 损失函数增加 SCL 说话人一致性 loss
    - 合成音频/真实音频的梅尔特征，分别经过 H/ASP 模型得到的 spk emb，计算余弦相似度（**最大化余弦相似度**）

# 实验相关

- 训练数据：英文 14k 小时，其他语种 13k 小时
- 推理参数：温度系数 0.75，repetition penalty 10，top_k=50，top_p=0.85

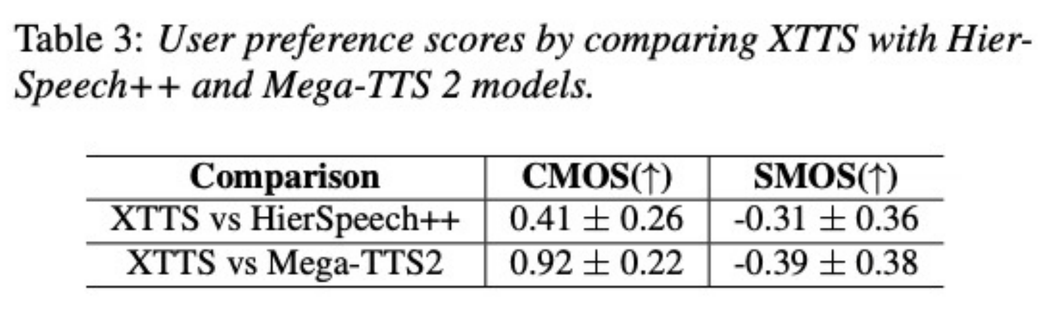


Table 3: *User preference scores by comparing XTTS with Hier-Speech++ and Mega-TTS 2 models.*

| Comparison | CMOS(↑) | SMOS(↑) |
|---|---|---|
| XTTS vs HierSpeech++ | 0.41 ± 0.26 | -0.31 ± 0.36 |
| XTTS vs Mega-TTS2 | 0.92 ± 0.22 | -0.39 ± 0.38 |

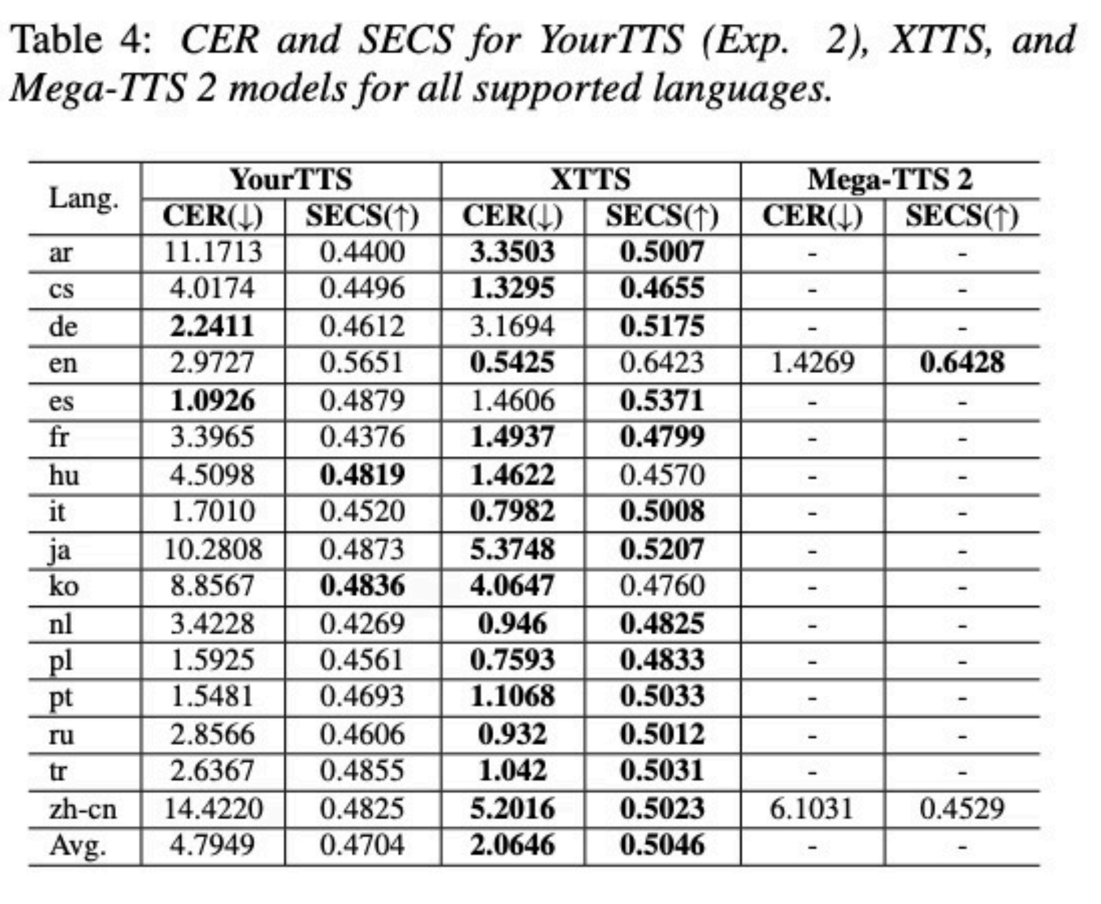
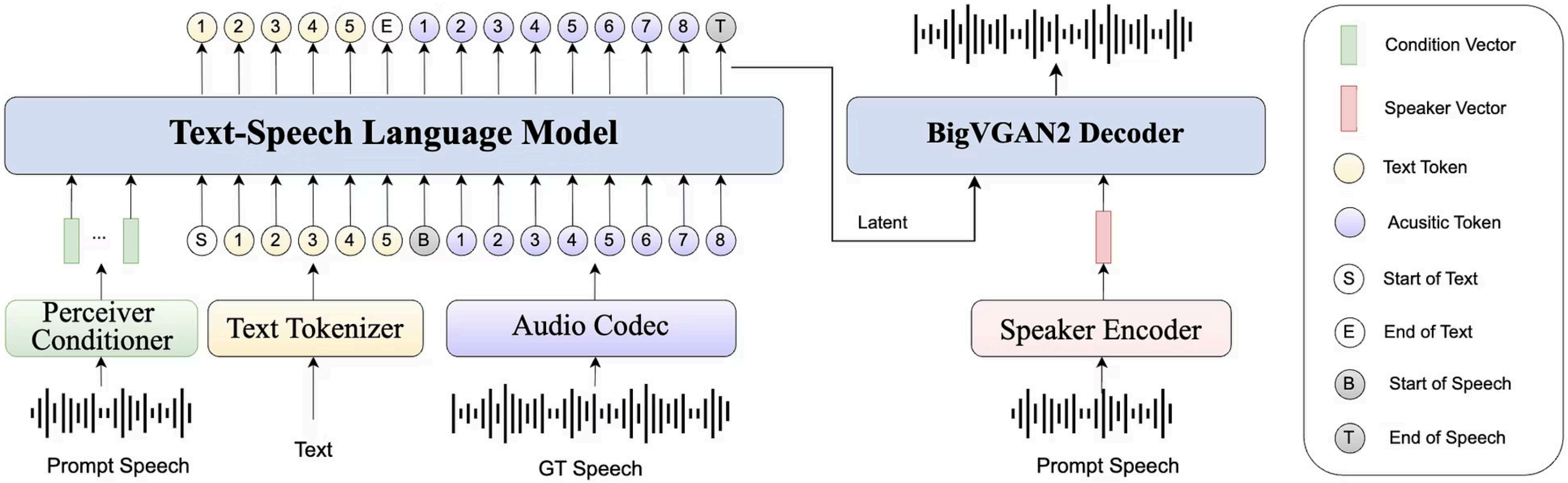


Table 4: *CER and SECS for YourTTS (Exp. 2), XTTS, and Mega-TTS 2 models for all supported languages.*

| Lang. | YourTTS | | XTTS | | Mega-TTS 2 | |
|---|---|---|---|---|---|---|
| | CER(↓) | SECS(↑) | CER(↓) | SECS(↑) | CER(↓) | SECS(↑) |
| ar | 11.1713 | 0.4400 | **3.3503** | **0.5007** | - | - |
| cs | 4.0174 | 0.4496 | **1.3295** | **0.4655** | - | - |
| de | **2.2411** | 0.4612 | 3.1694 | **0.5175** | - | - |
| en | 2.9727 | 0.5651 | **0.5425** | 0.6423 | 1.4269 | **0.6428** |
| es | **1.0926** | 0.4879 | 1.4606 | **0.5371** | - | - |
| fr | 3.3965 | 0.4376 | **1.4937** | **0.4799** | - | - |
| hu | 4.5098 | **0.4819** | **1.4622** | 0.4570 | - | - |
| it | 1.7010 | 0.4520 | **0.7982** | **0.5008** | - | - |
| ja | 10.2808 | 0.4873 | **5.3748** | **0.5207** | - | - |
| ko | 8.8567 | **0.4836** | **4.0647** | 0.4760 | - | - |
| nl | 3.4228 | 0.4269 | **0.946** | **0.4825** | - | - |
| pl | 1.5925 | 0.4561 | **0.7593** | **0.4833** | - | - |
| pt | 1.5481 | 0.4693 | **1.1068** | **0.5033** | - | - |
| ru | 2.8566 | 0.4606 | **0.932** | **0.5012** | - | - |
| tr | 2.6367 | 0.4855 | **1.042** | **0.5031** | - | - |
| zh-cn | 14.4220 | 0.4825 | **5.2016** | **0.5023** | 6.1031 | 0.4529 |
| Avg. | 4.7949 | 0.4704 | **2.0646** | **0.5046** | - | - |

# Index-TTS 概述

- 论文题目：**IndexTTS: An Industrial-Level Controllable and Efficient Zero-Shot Text-To-Speech System**
  - 作者：B 站 - 人工智能平台部
- 论文链接：hhttps://arxiv.org/pdf/2502.05512
- 开源地址：https://github.com/index-tts/index-tts（⭐ 3k+）



# 总结

1. Mel VQ 作为建模的 Token?
    a. 前提：梅尔特征基本能够重建波形
    b. 优势：训练非常简单
    c. 缺点：没有解耦音色/内容/韵律
2. Flow Matching 使用 Speaker Encoder 的输出 embedding 作为条件?
3. Perceiver sampler 相比于单 embedding speaker encoder 的优势?
4. 几篇比较接近的论文对比

| 名称 | 论文 | 时间 | spk-emb 拼接策略 | Tokenizer | Decoder | Vocoder | Vocoder 输入 |
|---|---|---|---|---|---|---|---|
| Tortoise-TTS | 2305.07243 | 2023.05 | 拼接单个 spk_emb | Mel-VQ | DDPM/DDIM | Univnet | mel |
| Chat-TTS | - | 2024.05 | 不拼接 | Mel-VQ | 非自回归 Decoder | Vocos | mel |
| Fish-speech 1.0 | github | 2024.05 | 不拼接 | Mel VQ | VITS | HiFiGAN | embedding |
| XTTS-v2 | 2406.04904 | 2024.06 | 拼接：perceiver | Mel-VQ | 整体 HiFiGAN | | latents |
| Index-TTS | 2502.05512 | 2025.02 | 拼接：perceiver | Mel-VQ | 整体 BigVGAN | | latents |