DiffRO

- 论文题目：**Differentiable Reward Optimization for LLM based TTS system**
- 论文链接：https://arxiv.org/abs/2507.05911

相关信息

- 其他链接：
  - CosyVoice 2：https://arxiv.org/abs/2412.10117
  - CosyVoice 3：https://arxiv.org/abs/2505.17589
- 开源讨论：https://github.com/FunAudioLLM/CosyVoice/issues/1442
- 相关博客：
  - https://sankar1535.substack.com/p/revolutionizing-text-to-speech-how
- ReadPaper 链接：

pay.readpaper.com

ReadPaper

论文概览

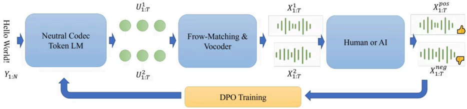
TTS + RL 现存问题

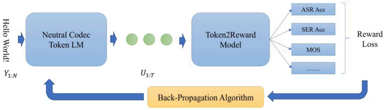
1. 针对基于 LLM + CFM + Vocoder 的模型，LLM 的 RLHF 不好直接获取 human feedback，一般只能将 LLM 输出 token 还原为音频，再通过 AI 或者人工反馈来提供偏好信息
2. TTS 合成音频的 diversity 不够，趋向于一致性，导致 reward 建模需要区分细粒度上正负例的差异
   a. 有可能差异不大很难定义/决定哪条是正例、哪条是负例
3. TTS 的评估，一般都是以人的主观听感来评判发音准确度、自然流畅度、情感表现力等维度；在 instruct-tts 的场景上，这些维度的控制可能是存在一些冲突的
   a. DPO 正例、负例的评价方式，可能会耦合发音、韵律、音色、情感化等维度，不容易设计不同维度的优化

论文关键改进

- RL 不是在生成完成音频之后做的，而是在 LLM 的 text-to-token 过程中直接做的
- gumbel_softmax 模拟 LLM 中 argmax 的操作，将 token2reward 的过程变为可以微分，具备训练的可能
- 提出 MTR（多任务 multi-task reward），提高对 instruction 的 follow 能力，比如 zero-shot 的情绪、音质等控制能力

论文思路图





T2R (Token2Reward)

ASR 任务：token2text

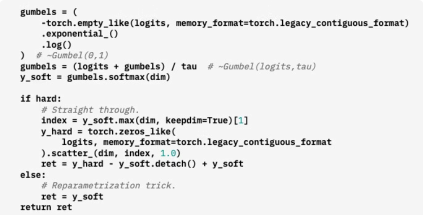
- 对于 LLM 模块的 token，额外增加一个 ASR 任务的约束，希望生成的 token 能够尽可能接近语义信息

$$\tilde{U}_t = \mathrm{argmax} P_{\pi_\theta}(U_t | U_{1:n-1}; Y)$$
$$Y_n^* = \mathrm{argmax} P_{ASR}(\tilde{Y}_n | Y_{1:n-1}; \tilde{U}_{1:T})$$

- 第一行公式：表示 LLM 预测 token 的过程，argmax 仅用来表征了 greedy 解码的过程（此处实际上用「采样」更贴切）
  - 疑问：在实际推理阶段，往往采用 top_k / top_p 采样，从这一角度训练和推理实际是 mismatch 的？
  - 是否考虑其他的方法?
    - 方案一：不考虑将 LLM 预测的 logits 进行采样，而是直接在 latents 上进行 latent2text 的 ASR 优化?
      - 类似 Qwen2.5-Omni 的思想，直接在 hidden states 上将 LLM 与后续模型组合起来，而不模拟采样过程
    - 方案二：即使需要模拟采样过程，是否应该适配 top_k 或者 top_p 采样方案，再用对应的可微分近似操作
      - top_k/top_k 本身也不可导，是否有对应的可微分方法？（gumbel top_k/soft top_k，但是感觉不是本质改善）
- 第二行公式：表示使用采样出来的 token 结果，再进行 ASR 预测文本的过程
  - 实际上是训练一个 token2text 的 ASR 模型，ASR 预测的后验概率可以当成 reward，直接反映在 ASR 任务的 loss 里
  - 采样过程本身不可导，本文采用的是 gumbel_softmax 的方式进行可微

Gumbel-Softmax

- Gumbel Softmax 允许模型中有从离散的分布（比如类别分布 categorical distribution）中采样，将这个过程变得可微，从而允许反向传播时可以用梯度更新模型参数。
- 注意：gumbel-softmax 模拟的是从 softmax 分布中【采样】并选择一个类别的过程，而不是单纯模拟 argmax **选最大类别。**
- pytorch 官方代码：
  - https://docs.pytorch.org/docs/stable/_modules/torch/nn/functional.html#gumbel_softmax



```
gumbels = (
    -torch.empty_like(logits, memory_format=torch.legacy_contiguous_format)
    .exponential_()
    .log()
)  # ~Gumbel(0,1)
gumbels = (logits + gumbels) / tau  # ~Gumbel(logits,tau)
y_soft = gumbels.softmax(dim)

if hard:
    # Straight through.
    index = y_soft.max(dim, keepdim=True)[1]
    y_hard = torch.zeros_like(
        logits, memory_format=torch.legacy_contiguous_format
    ).scatter_(dim, index, 1.0)
    ret = y_hard - y_soft.detach() + y_soft
else:
    # Reparametrization trick.
    ret = y_soft
return ret
```

参考链接

参考文章：

- https://zhuanlan.zhihu.com/p/633431594
- https://sassafras13.github.io/GumbelSoftmax/

参考论文：

arXiv.org

Categorical Reparameterization with Gumbel-Softmax

Categorical variables are a natural choice for representing discrete structure in the world. However, stochastic neural networks rarely use categorical latent variables due to the...

arXiv.org

The Concrete Distribution: A Continuous Relaxation of Discrete...

The reparameterization trick enables optimizing large scale stochastic computation graphs via gradient descent. The essence of the trick is to refactor each stochastic nod...

参考视频：点击此处

1. 逆变换采样

- 累积概率分布（CDF）：将多类别的概率分布 p 从前往后累加起来
- 从均匀分布中采样出一个随机值，对于 CDF 里概率刚好超过随机值的类别就作为被采样的类别结果
- 上述【基于 CDF 和均匀分布】的采样方式，被称为「逆变换采样」
  - https://zhuanlan.zhihu.com/p/396618080
  - https://www.math.pku.edu.cn/teachers/lidf/docs/statcomp/html/_statcompbook/rng-nonuni.html
  - 附加知识：如何生成[0, 1] 之间均匀分布的数值?
    - 首先 生成伪随机数（很多伪随机数生成器算法，s），再放缩到 [0, 1] 数值范围内
- 【注意】逆变换采样的前提是，CDF 是可以直接获取或求解的，否则还需要使用【拒绝采样】等方法

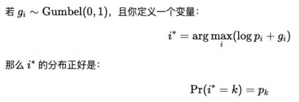
2. Gumbel Max

- 解读论文：https://arxiv.org/pdf/2110.01515
- Gumbel-Max：通过在一个概率分布上增加 gumbel 噪声带来随机性，使得每次采样的结果不一样
  - Gumbel-Max 算法提供了一种通过对数概率 + Gumbel 噪声再取 argmax 的方式，实现对离散概率分布的采样。
- Gumbel-Max Trick也是使用了重参数技巧，把采样过程分成了确定性的部分和随机性的部分

$$z = argmax_i(log(\pi_i) + g_i)$$

$$g_i = -log(-log(u_i)), u_i \sim U(0,1)$$

- 第二行公式：g_i 用到了采样思想，先用均匀分布采样出一个随机值，然后代入到 gumbel 分布的CDF函数的逆函数（inverse function，或者称为反函数），即得到采样值
    - Gumbel-Max trick 里使用的 gumbel 分布是标准 gumbel 分布，μ=0，β=1
    - 标准 gumbel 分布的 CDF 是 $y = e^{-e^{-x}}$，那它的逆函数就是 $x = -ln(-ln(x))$
- Gumbel-Max 和 torch.multinomial 实际都是模拟采样操作，仍然存在 argmax 不可导的问题
- 补充问题：**为什么 Gumbel-Max 中使用的是对数概率 $log(\pi_i)$?**
    - 核心点：公式推导出，Gumbel noise + 对数概率，再取 argmax，才能保证采样到的结果正好服从原来的离散分布



若 $g_i \sim \mathrm{Gumbel}(0,1)$，且你定义一个变量：

$$i^* = \arg\max_i(\log p_i + g_i)$$

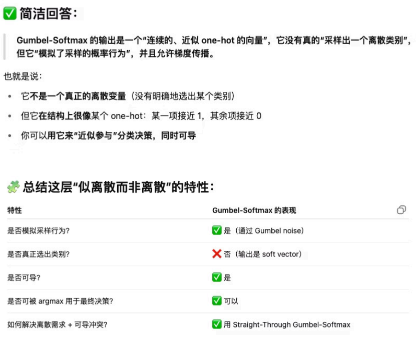
那么 i^* 的分布正好是：

$$\Pr(i^* = k) = p_k$$

3. Gumbel-Softmax 解决不可导的问题

思想纠正：Gumbel-Softmax 不是对 argmax 的近似，而是对采样过程的近似。

- Gumbel-Softmax 模拟的是从 softmax 分布中**采样并选择一**个类别的过程，而不是单纯选最大值。
- 在 Gumbel-Max 的基础上，通过采用 softmax 操作，去除不可导的 argmax
- 当 softmax 温度趋近 0 时，Gumbel-Softmax 输出接近 one-hot（注意是接近 one-shot，但实际仍然是 soft vector）
    - For τ>0 , the Gumbel Softmax is a continuous relaxation of the discrete sampling and therefore can be seen of soft vector.
- 疑问：虽然 gumbel-softmax 是模拟的离散采样的过程，但 gumbel-softmax 实际上输出还是连续的，没有真正的再通过比如 softmax 来采样出离散类别序列。这一点应该如何理解？



✅ **简洁回答：**

> **Gumbel-Softmax 的输出是一个“连续的、近似 one-hot 的向量”，它没有真的“采样出一个离散类别”，但它“模拟了采样的概率行为”，并且允许梯度传播。**

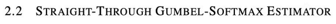
也就是说：

- **它不是一个真正的离散变量**（没有明确地选出某个类别）
- **但它在结构上很像**某个 one-hot：某一项接近 1，其余项接近 0
- **你可以用它来“近似参与”分类决策，同时可导**

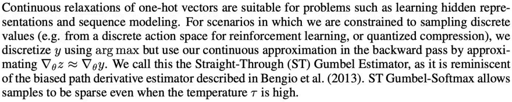
总结这层“似离散而非离散”的特性：

特性	Gumbel-Softmax 的表现
是否模拟采样行为?	✅ 是（通过 Gumbel noise）
是否真正选出类别?	❌ 否（输出是 soft vector）
是否可导?	✅ 是
是否可被 argmax 用于最终决策?	✅ 可以
如何解决离散需求 + 可导冲突?	✅ 用 Straight-Through Gumbel-Softmax

- 如果想在 Gumbel-Softmax 的基础上，真正实现离散采样出 token 的需求，可以使用 straight-through gumbel-softmax（ST-GS），在 gumbel-softmax 原论文中也被提到
    - https://fabianfuchsml.github.io/gumbel/
    - https://arxiv.org/pdf/1611.01144



2.2 STRAIGHT-THROUGH GUMBEL-SOFTMAX ESTIMATOR



Continuous relaxations of one-hot vectors are suitable for problems such as learning hidden representations and sequence modeling. For scenarios in which we are constrained to sampling discrete values (e.g. from a discrete action space for reinforcement learning, or quantized compression), we discretize y using arg max but use our continuous approximation in the backward pass by approximating $\nabla_\theta z \approx \nabla_\theta y$. We call this the Straight-Through (ST) Gumbel Estimator, as it is reminiscent of the biased path derivative estimator described in Bengio et al. (2013). ST Gumbel-Softmax allows samples to be sparse even when the temperature τ is high.

MTR (Multi-Task Reward)

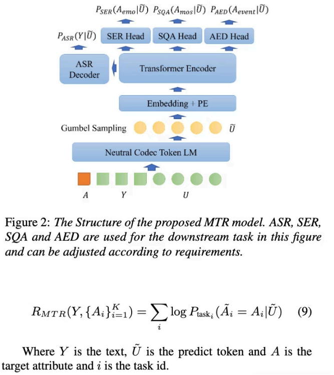


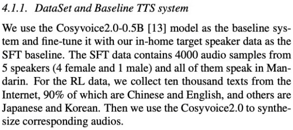
Figure 2: *The Structure of the proposed MTR model. ASR, SER, SQA and AED are used for the downstream task in this figure and can be adjusted according to requirements.*

$$R_{MTR}(Y, \{A_i\}_{i=1}^{K}) = \sum_i \log P_{\text{task}_i}(\tilde{A}_i = A_i|\tilde{U}) \quad (9)$$

Where Y is the text, $\tilde{U}$ is the predict token and A is the target attribute and i is the task id.

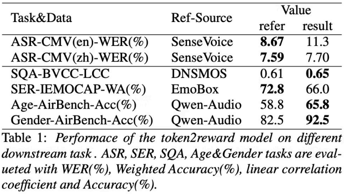
- 除了 ASR，其他具体任务包括：情感识别、音质 MOS 估计、音频事件检测（按论文还有年龄和性别）

实验细节

数据集及基线	 *4.1.1. DataSet and Baseline TTS system* We use the Cosyvoice2.0-0.5B [13] model as the baseline system and fine-tune it with our in-home target speaker data as the SFT baseline. The SFT data contains 4000 audio samples from 5 speakers (4 female and 1 male) and all of them speak in Mandarin. For the RL data, we collect ten thousand texts from the Internet, 90% of which are Chinese and English, and others are Japanese and Korean. Then we use the Cosyvoice2.0 to synthesize corresponding audios.
Reward 模型训练	• 将 SenseVoice 原本接收语音表征的 CNN 替换为接收语音 token 的 embedding 层 • 对于下游任务，每个任务增加额外的 attention pooling 层（因为是整条语音级别的分类任务） • 训练数据：使用各任务的模型打出伪标签（文本、情绪、MOS、年龄性别、声音事件） 　◦ 声音事件因为一条中可能存在多个时间，所以采用的是 binary CE loss

    - MOS 打为 5 分档
    - gender 分别男、女、儿童三类
    - 年龄范围 0 - 100（理论范围）

Reward 模型实验结果



Task&Data	Ref-Source	Value	
		refer	result
ASR-CMV(en)-WER(%)	SenseVoice	**8.67**	11.3
ASR-CMV(zh)-WER(%)	SenseVoice	**7.59**	7.70
SQA-BVCC-LCC	DNSMOS	0.61	**0.65**
SER-IEMOCAP-WA(%)	EmoBox	**72.8**	66.0
Age-AirBench-Acc(%)	Qwen-Audio	58.8	**65.8**
Gender-AirBench-Acc(%)	Qwen-Audio	82.5	**92.5**

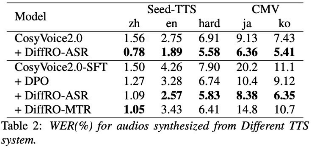
Table 1: *Performace of the token2reward model on different downstream task . ASR, SER, SQA, Age&Gender tasks are evalueted with WER(%), Weighted Accuracy(%), linear correlation coefficient and Accuracy(%).*

- ASR 结果比原始 SenseVoice 稍差
- 情感分类也变差（评价标准：按类别加权的准确率）
- 其他 Reward 模型均比用来打伪标签的模型更好
- DNSMOS 评价标准：线性相关系数
- 以上结果说明：token 包含了足够的情感、年龄、性别、音质的信息

DPO v.s. DiffRO

- 10000条文本，90%中英文，10%日语韩语
- DPO 实验中 beta=0.1, 学习率 1e-5
    - 每条文本合成 5 次，选出正负例
-

DiffRO-ASR 实验结论分析



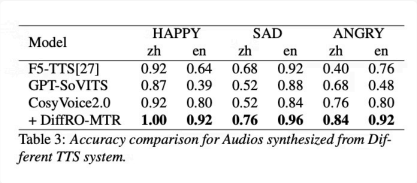
Model	Seed-TTS			CMV	
	zh	en	hard	ja	ko
CosyVoice2.0	1.56	2.75	6.91	9.13	7.43
+ DiffRO-ASR	**0.78**	**1.89**	**5.58**	**6.36**	**5.41**
CosyVoice2.0-SFT	1.50	4.26	7.90	20.2	11.1
+ DPO	1.27	3.28	6.74	10.4	9.12
+ DiffRO-ASR	1.09	**2.57**	**5.83**	**8.38**	**6.35**
+ DiffRO-MTR	**1.05**	3.43	6.41	14.8	10.7

Table 2: *WER(%) for audios synthesized from Different TTS system.*

- SFT 之后，除了中文 seed-zh，其他音色均出现不同程度变差
    - 原因：SFT 截断的灾难性遗忘
- 基座或 SFT 上进行 DiffRO，效果一致性变好
    - 针对发音准确性而言，DiffRO-ASR 效果好于 DPO
    - 但是仍然和基座上的 DiffRO 有较明显差距
- DiffRO-MTR 会弱化发音准确性维度带来的提升
    - 因为关注了其他非发音的因素，优化目标存在一定的竞争

DiffRO-MTR

- 情感、音质、年龄、性别都在 DiffRO 允许的范畴里
    - 但是，音质、年龄和性别都和 CFM 模块也有很强的耦合关系
- 情感方面的控制能力

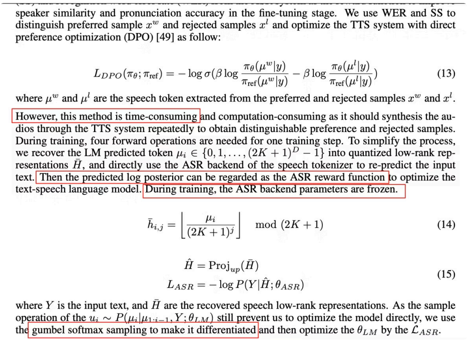


Model	HAPPY		SAD		ANGRY	
	zh	en	zh	en	zh	en
F5-TTS[27]	0.92	0.64	0.68	0.92	0.40	0.76
GPT-SoVITS	0.87	0.39	0.52	0.88	0.68	0.48
CosyVoice2.0	0.92	0.80	0.52	0.84	0.76	0.80
+ DiffRO-MTR	**1.00**	**0.92**	**0.76**	**0.96**	**0.84**	**0.92**

Table 3: *Accuracy comparison for Audios synthesized from Different TTS system.*

CosyVoice2/3 实践

CosyVoice2: https://arxiv.org/pdf/2412.10117



speaker similarity and pronunciation accuracy in the fine-tuning stage. We use WER and SS to distinguish preferred sample x^w and rejected samples x^l and optimize the TTS system with direct preference optimization (DPO) [49] as follow:

$$L_{DPO}(\pi_\theta;\pi_{\text{ref}}) = -\log\sigma\left(\beta\log\frac{\pi_\theta(\mu^w|y)}{\pi_{\text{ref}}(\mu^w|y)} - \beta\log\frac{\pi_\theta(\mu^l|y)}{\pi_{\text{ref}}(\mu^l|y)}\right) \tag{13}$$

where μ^w and μ^l are the speech token extracted from the preferred and rejected samples x^w and x^l.

However, this method is time-consuming and computation-consuming as it should synthesis the audios through the TTS system repeatedly to obtain distinguishable preference and rejected samples. During training, four forward operations are needed for one training step. To simplify the process, we recover the LM predicted token $\mu_i \in \{0, 1, \ldots, (2K+1)^D - 1\}$ into quantized low-rank representations $\bar{H}$, and directly use the ASR backend of the speech tokenizer to re-predict the input text. Then the predicted log posterior can be regarded as the ASR reward function to optimize the text-speech language model. During training, the ASR backend parameters are frozen.

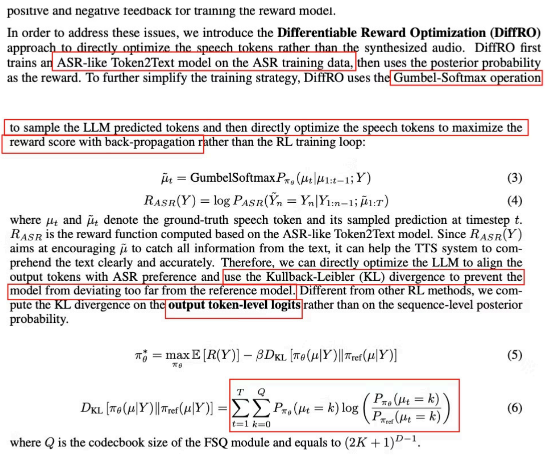
$$\bar{h}_{i,j} = \left\lfloor \frac{\mu_i}{(2K+1)^j} \right\rfloor \mod (2K+1) \tag{14}$$

$$\begin{aligned}\hat{H} &= \text{Proj}_{up}(\bar{H}) \\ L_{ASR} &= -\log P(Y|\hat{H};\theta_{ASR})\end{aligned} \tag{15}$$

where Y is the input text, and $\bar{H}$ are the recovered speech low-rank representations. As the sample operation of the $u_i \sim P(\mu_i|\mu_{1:i-1}, Y; \theta_{LM})$ still prevent us to optimize the model directly, we use the gumbel softmax sampling to make it differentiated and then optimize the θ_{LM} by the $\mathcal{L}_{ASR}$.

- 关键点：token2text 的 ASR 预测模块是完全固定的；CosyVoice2 只考虑了 ASR loss；同样使用了 gumbel softmax 采样来使得模型是可导的

CosyVoice3: https://arxiv.org/pdf/2505.17589



positive and negative feedback for training the reward model.

In order to address these issues, we introduce the **Differentiable Reward Optimization (DiffRO)** approach to directly optimize the speech tokens rather than the synthesized audio. DiffRO first trains an ASR-like Token2Text model on the ASR training data, then uses the posterior probability as the reward. To further simplify the training strategy, DiffRO uses the Gumbel-Softmax operation

to sample the LLM predicted tokens and then directly optimize the speech tokens to maximize the reward score with back-propagation rather than the RL training loop:

$$\tilde{\mu}_t = \text{GumbelSoftmax} P_{\pi_\theta}(\mu_t|\mu_{1:t-1}; Y) \tag{3}$$

$$R_{ASR}(Y) = \log P_{ASR}(\tilde{Y}_n = Y_n|Y_{1:n-1}; \tilde{\mu}_{1:T}) \tag{4}$$

where μ_t and $\tilde{\mu}_t$ denote the ground-truth speech token and its sampled prediction at timestep t. R_{ASR} is the reward function computed based on the ASR-like Token2Text model. Since $R_{ASR}(Y)$ aims at encouraging $\tilde{\mu}$ to catch all information from the text, it can help the TTS system to comprehend the text clearly and accurately. Therefore, we can directly optimize the LLM to align the output tokens with ASR preference and use the Kullback-Leibler (KL) divergence to prevent the model from deviating too far from the reference model. Different from other RL methods, we compute the KL divergence on the **output token-level logits** rather than on the sequence-level posterior probability.

$$\pi_\theta^* = \max_{\pi_\theta} \mathbb{E}\left[R(Y)\right] - \beta D_{\text{KL}}\left[\pi_\theta(\mu|Y) \| \pi_{\text{ref}}(\mu|Y)\right] \tag{5}$$

$$D_{\text{KL}}\left[\pi_\theta(\mu|Y) \| \pi_{\text{ref}}(\mu|Y)\right] = \sum_{t=1}^{T}\sum_{k=0}^{Q} P_{\pi_\theta}(\mu_t = k)\log\left(\frac{P_{\pi_\theta}(\mu_t = k)}{P_{\pi_{\text{ref}}}(\mu_t = k)}\right) \tag{6}$$

where Q is the codebook size of the FSQ module and equals to $(2K+1)^{D-1}$.

按照 CosyVoice3 的论文，反而训练细节更多一些

- TTS 先单独训练一个基础模型（不管是基座还是 SFT 都可以）
- 训练 Reward 模型（token2text 或者 token2emotion 这些模型都是单独训练的）
- 在基础模型作为参考模型的基础上，增加适配特定 Reward 的数据
  - token2text Reward 的数据：可以是任意带有文本标注的 TTS 训练数据
  - token2emo Reward 的数据：必须带有情感标注的数据
    - gender/age 这些也都是一样，需要相应的标注信息来计算 reward loss 才能反向传播优化模型参数
- 增加特定 Reward 的数据之后，固定住 reward 模型的参数，更新 TTS LLM 的模型参数即可
  - 注意：训练时还需要增加与参考模型的 KL 散度 Loss，保证模型参数变化与之前差距不大，类似 PPO

Made with GAMMA