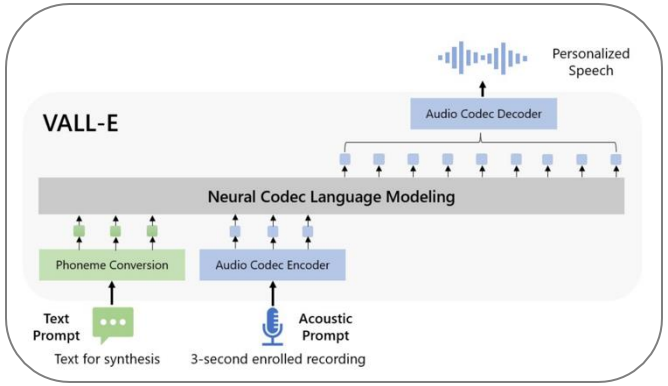
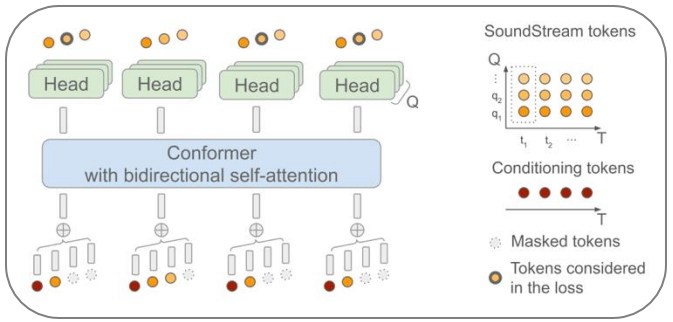
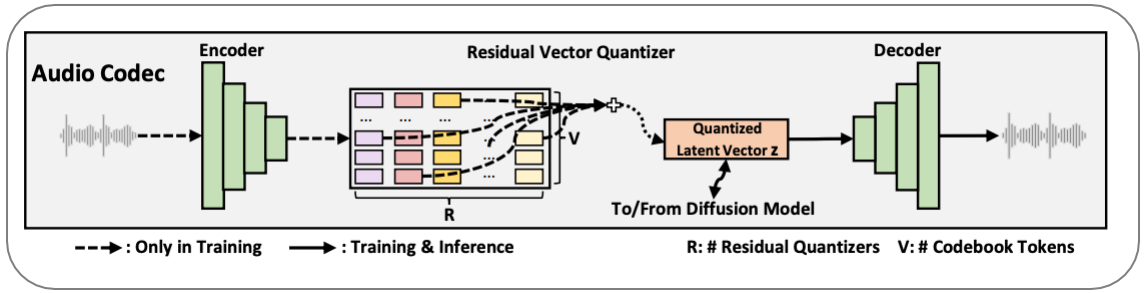
NaturalSpeech 3: Zero-Shot Speech Synthesis with Factorized Codec and Diffusion Models

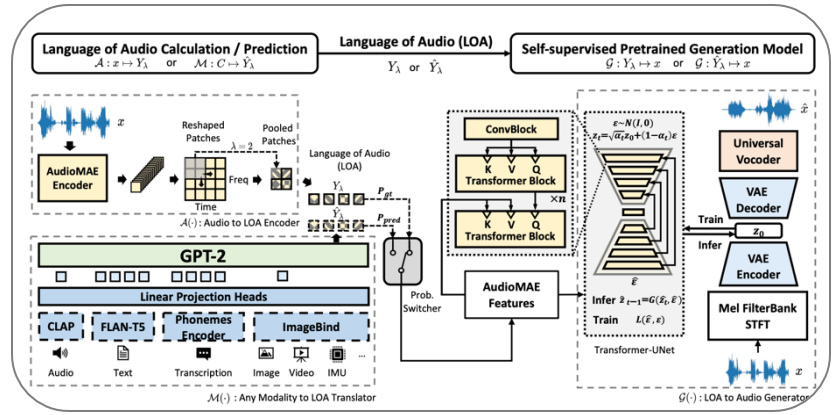
2024-04-01

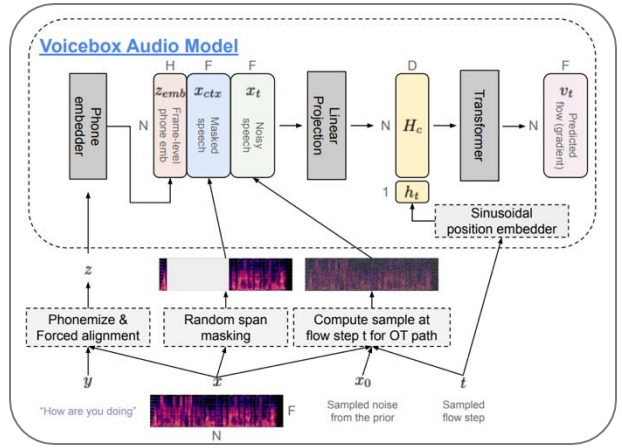


	离散表征	连续表征
自回归建模 (AR)	• VALL-E (s1) / Speech-X • Spear-TTS/AudioLM • Make-A-Voice	• AudioLDM2 (s1)
非自回归建模 (Diffusion/ MaskGIT/ Flow-Matching/ ...)	• SoundStorm • **NaturalSpeech3** • MobileSpeech • UniCATS	• NaturalSpeech2 • StyleTTS2 • VoiceBox





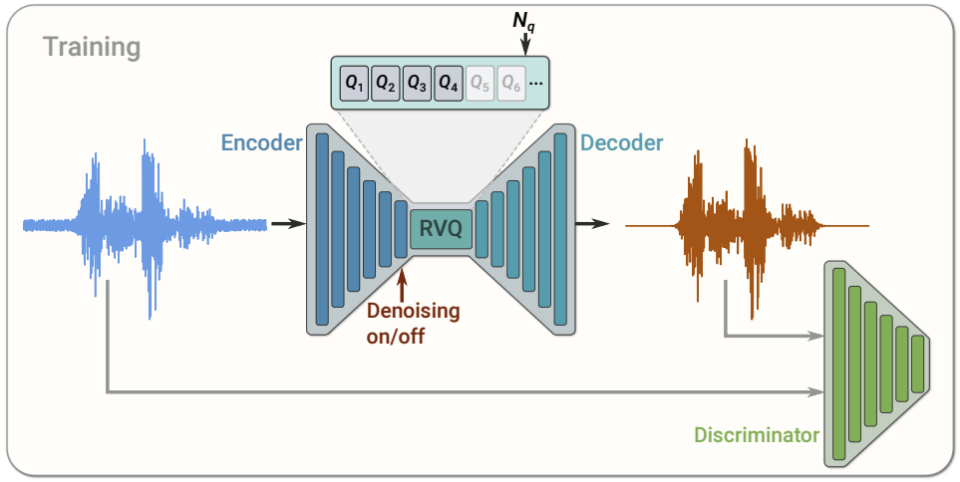


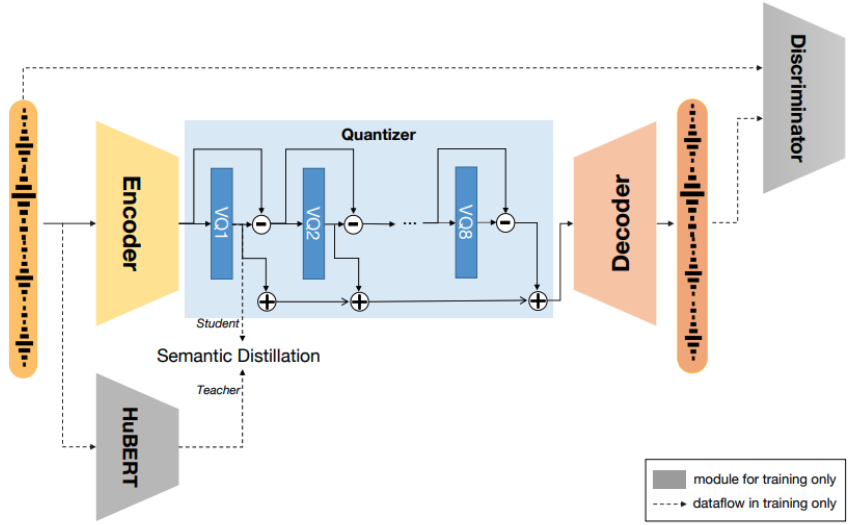


FACodec: Attribute **FA**ctorized Speech **Codec**

Neural Audio Codec

- **出发点**
  - 音频编解码：音频的低码率传输
  - 音频生成任务：离散化表征更适合一些模型
- **现有工作的不足**
  - RVQ 的残差建模，并不具有解耦的效果
  - 更新的工作：局限于语义和声学两方面分解
  - 表征的粒度仍然不够细化，可控性差
    - 声学的细节耦合在一起：音色/韵律/背景音...
- **问题：**如何得到更适合于语音生成任务、更可控的 Codec?

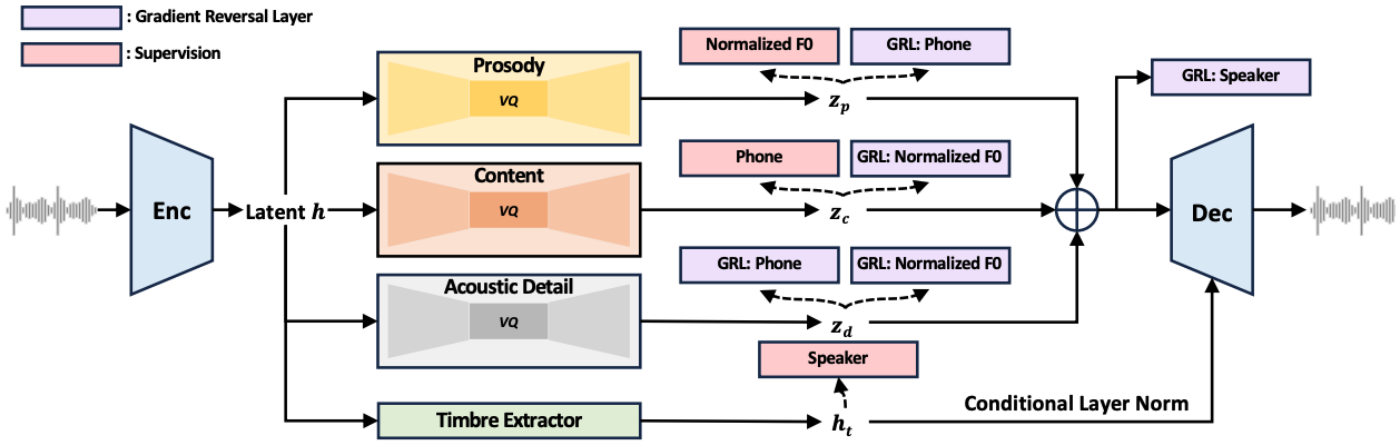




FACodec: Factorizing Speech Attributes

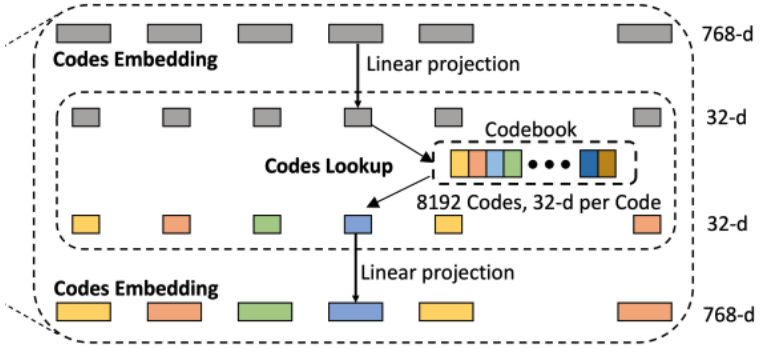
- **FACodec: 将语音信息分解为四大属性**
  - 韵律（p）、音色（t）、内容（c）、声学细节（d）
  - 优点：可控性更好、建模难度更低（分而治之）
  - 难点：如何保证各部分属性信息的解耦效果?

- **保证各属性解耦效果的四大方法**
  1. VQ（Bottleneck）：信息瓶颈，控制保留的信息量
  2. Supervision: 不同模块各司其职
  3. GRL（梯度反转）：进一步去除各属性模块的混淆
  4. Detail Dropout：对 detail 模块的随机 dropout

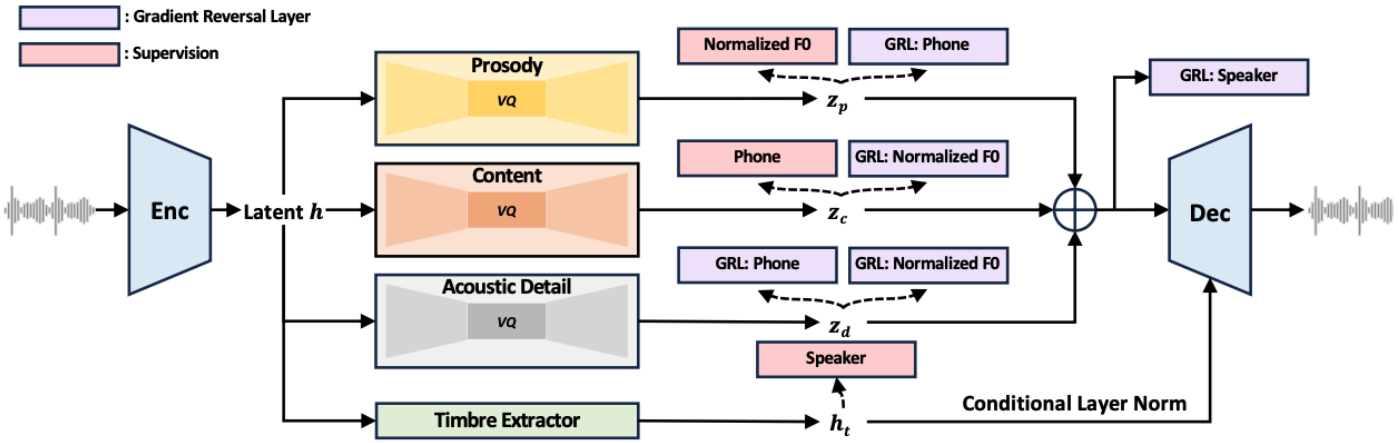


FACodec: Factorizing Speech Attributes

- **方法一：Information Bottleneck （ VQ)**
  - Improved VQ（DAC）：
    - VQ 在很低的维度（8 维）表征空间进行
    - 实践效果：提高 Codebook 的利用率
  - 不同特性使用了不同层数的 RVQ
    - prosody: 1 层
    - content: 2 层
    - detail: 3 层
    - 每层 VQ 的 codebook size 是 1024
  - 简单对应成：具有 6 层 VQ 的 Codec

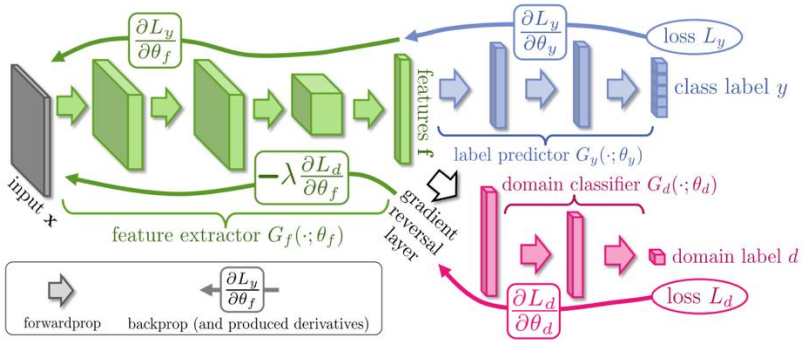


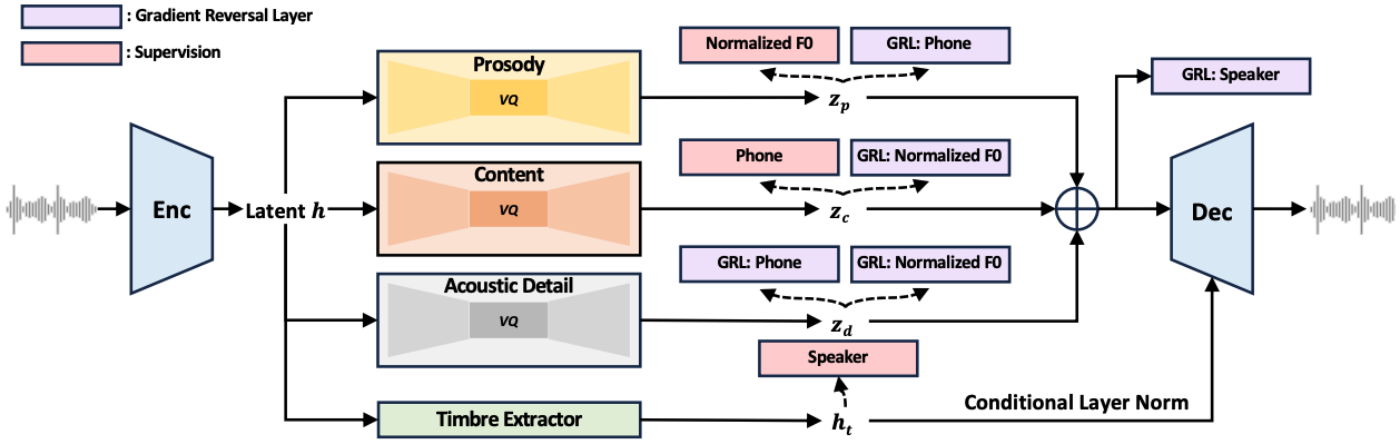
- **方法二：Supervision 监督学习**
  - prosody:
    - 帧级别的 pitch (norm) 回归预测
    - 代码中实际还预测了 V/UV 的二分类任务
  - content: 强制对齐后，帧级别的 phone 分类
  - timbre: speaker embedding 预测 speaker-id（代码实际没有使用）



FACodec: Factorizing Speech Attributes

- **方法三：GRL 梯度反转**
  - 问题：只靠 supervision 无法保证各模块内信息的绝对解耦
    - 比如：预测 phone 的模块，可能不只建模了 content，也建模了音色或者韵律
  - speaker-GRL: GRL 增加分类层预测 spk 标签，但梯度反转

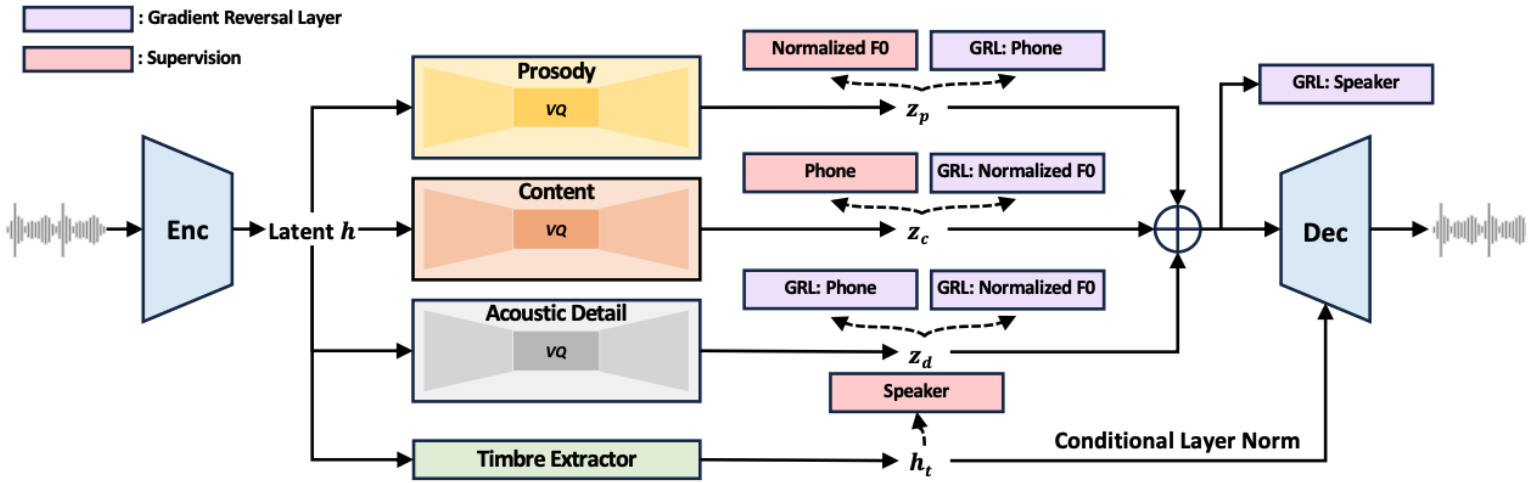




- **方法四：Detail Dropout**
  - 出发点：prosody + content + timbre 原则上足够恢复基本语音信息，只不过音质细节上不佳
  - 训练过程中，以一定概率随机 mask detail 的输出，强制从其他模块的输出也能恢复基本的波形

FACodec: Factorizing Speech Attributes

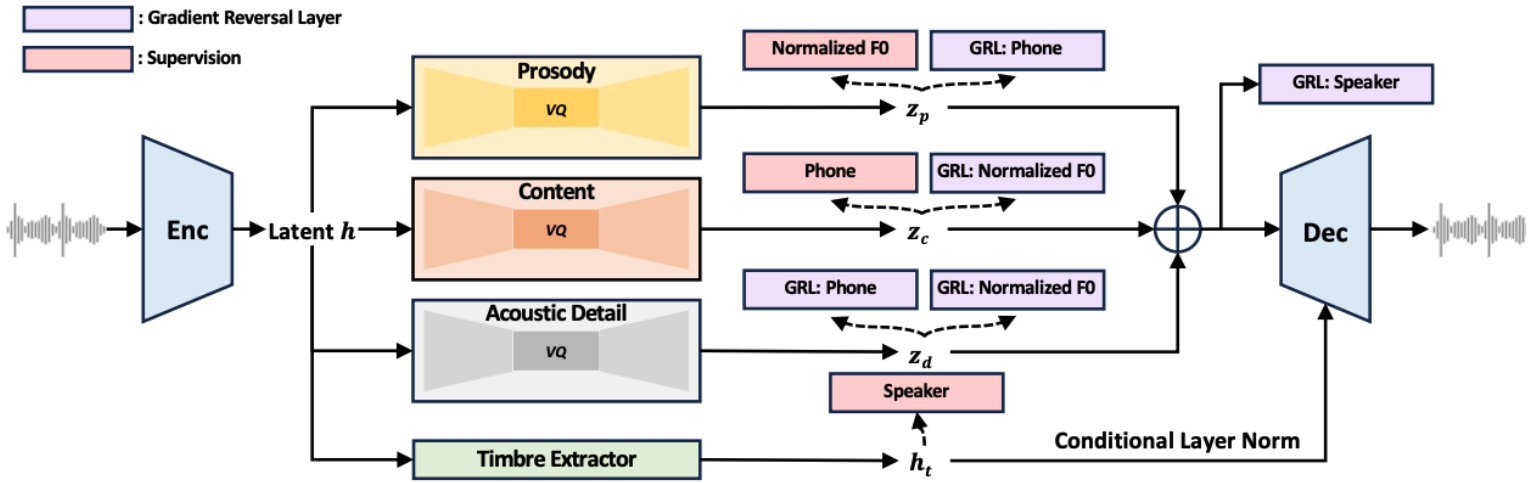
- **模型结构细节**
  - 音频采样率 16kHz, 降采样倍数 200，每秒 80 帧
  - Encoder / Decoder: 与 DAC (descript audio codec) 相同（加入了 Snake 激活函数）
  - Timbre Extractor: 4 层 Transformer, 输出结果 average pooling 得到 speaker embedding
    - Speaker embedding 以 Conditional Layer Norm 的方式加入到 Decoder 中
  - 每个任务，Supervision 和 GRL 的 Predictor: 多层卷积结构，GRL 梯度反转系数为 1.0
  - GAN 的判别器：MPD （多周期）、MSD（多尺度 STFT）



FACodec: Factorizing Speech Attributes

- **训练细节**
  - 损失函数
    - VQ: 重构 loss (多尺度 mel loss) + codebook loss + commitment loss
    - Supervision: phone loss, f0 loss
    - GRL: GRL-phone loss, GRL-f0 lss, GRL-spk loss
    - GAN: GAN loss + feature matching
  - 训练数据：LibriLight 数据集；训练配置：8 卡 V100，800k steps

$$\lambda_{\text{rec}}\mathcal{L}_{\text{rec}} + \lambda_{\text{codebook}}\mathcal{L}_{\text{codebook}} + \lambda_{\text{commit}}\mathcal{L}_{\text{commit}}$$
$$\lambda_{\text{ph}}\mathcal{L}_{\text{ph}} + \lambda_{\text{f0}}\mathcal{L}_{\text{f0}}$$
$$\lambda_{\text{gr-ph}}\mathcal{L}_{\text{gr-ph}} + \lambda_{\text{gr-f0}}\mathcal{L}_{\text{gr-f0}} + \lambda_{\text{gr-spk}}\mathcal{L}_{\text{gr-spk}}$$
$$\lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{feat}}\mathcal{L}_{\text{feat}}$$



FACodec: Factorizing Speech Attributes

- **FACodec 效果评测**
  - 波形重建

Table 13: The reconstruction quality evaluation of codecs. ♣ means results are infered from offical checkpoints. ★ means the reproduced checkpoint. ♦ means the reproduced model following the original paper's implementation and experimental setup. All models use a codebook size of 1024. We use **bold** to indicate the best result and underline to indicate the second-best result. Abbreviation: H (Hop Size), N (Codebook Number).

Models	Sampling Rate	H	N	Bandwidth	PESQ ↑	STOI ↑	MSTFT ↓	MCD ↓
EnCodec♣	24kHz	320	8	6.0 kbps	3.28	0.94	0.99	2.70
EnCodec★	16kHz	320	10	5.0 kbps	3.10	0.92	0.97	3.10
HiFi-Codec♣	16kHz	320	4	2.0 kbps	3.17	0.93	0.98	3.05
DAC♣	16kHz	320	9	4.5 kbps	**3.52**	**0.95**	0.97	2.65
SoundStream♦	16kHz	200	6	4.8 kbps	3.03	0.90	1.07	3.38
SoundStream♦	16kHz	200	12	9.6 kbps	3.45	0.94	**0.92**	2.76
FACodec	16kHz	200	6	4.8 kbps	3.47	**0.95**	0.93	**2.59**

- Zero-Shot VC
  - source 音频：待转换的原始音色的音频，使用 FACodec 抽取 prosody/content/detail $z_c^{source}, z_p^{source}, z_d^{source}$
  - prompt 音频：目标音色的参考音频，使用 FACodec 的 timbre extractor 抽取 speaker embedding h_t^{prompt}
  - 使用 FACodec 的 Decoder 解码出 target 音频 $\mathcal{D}(z_c^{source}, z_p^{source}, z_d^{source}, h_t^{prompt})$

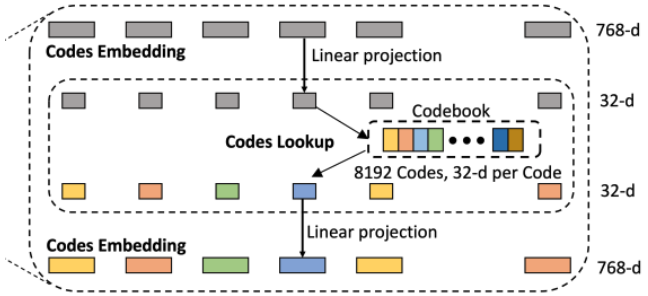
Models	Sim-O ↑	WER ↓
Ground Truth	-	3.25
YourTTS	0.72	10.1
Make-A-Voice (VC)	0.68	6.20
LM-VC	0.82	4.91
UniAudio	**0.87**	4.80
FACodec	0.86	**3.46**

Sim-O: 与 Original Prompt 之间的音色相似度

- demo 音频：https://speechresearch.github.io/naturalspeech3
- 试用页面：https://huggingface.co/spaces/amphion/naturalspeech3_facodec

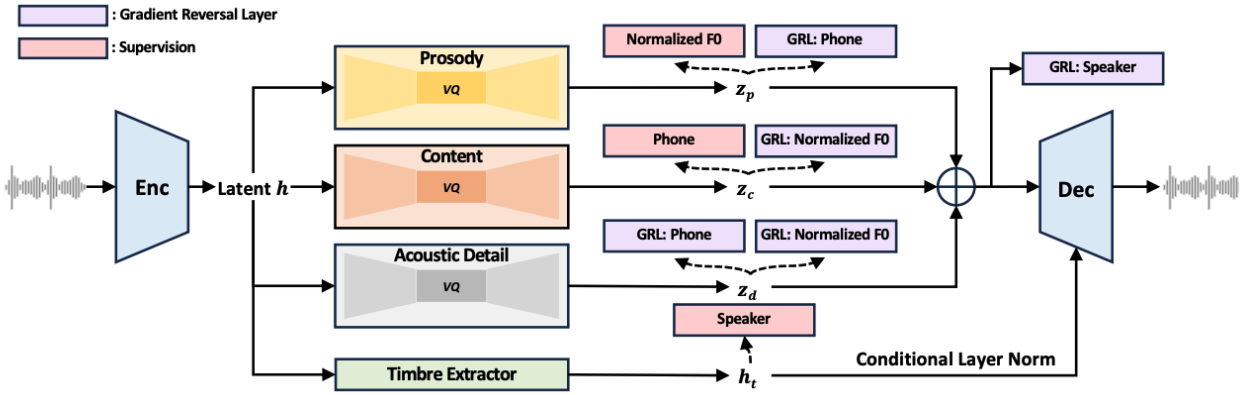
FACodec: Factorizing Speech Attributes

- **核心方法消融实验**
  - 所有 VQ 中是否加入 bottleneck
    - Zero-shot Voice Conversion 实验



	Sim-O ↑
w. information bottleneck	**0.86**
w.o. information bottleneck	0.73

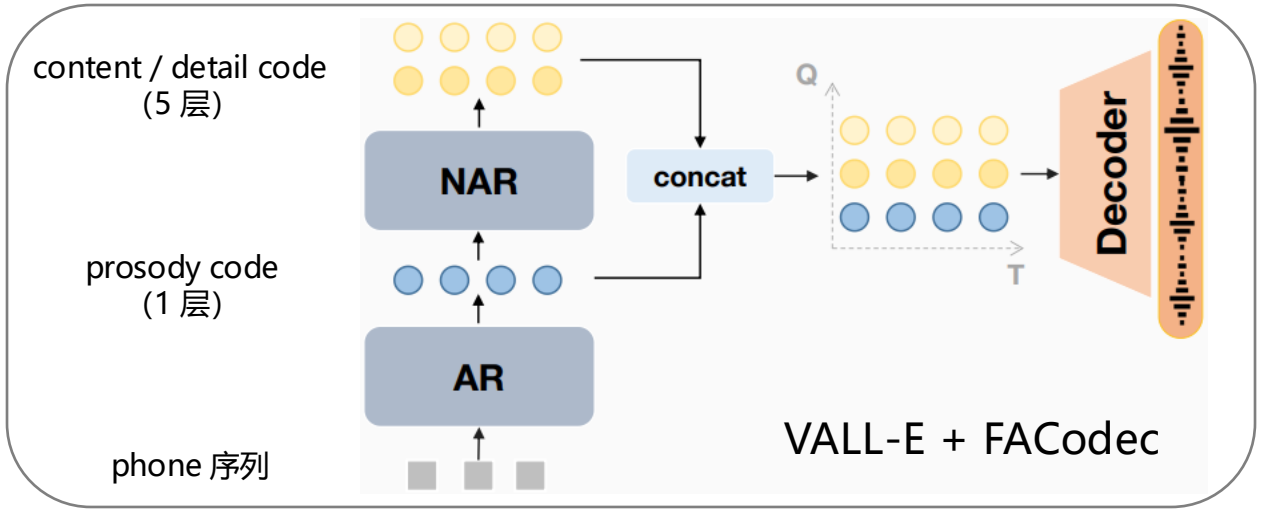
  - Detail 模块的重要性（提高合成音质）



	Codebook Number	PESQ ↑	STOI ↑	MSTFT ↓	MCD ↓
FACodec	6	**3.47**	**0.95**	**0.93**	**2.59**
- acoustic details quantizers	3	3.09	0.92	1.08	3.12
SoundStream	6	3.03	0.90	1.07	3.38

FACodec: Factorizing Speech Attributes

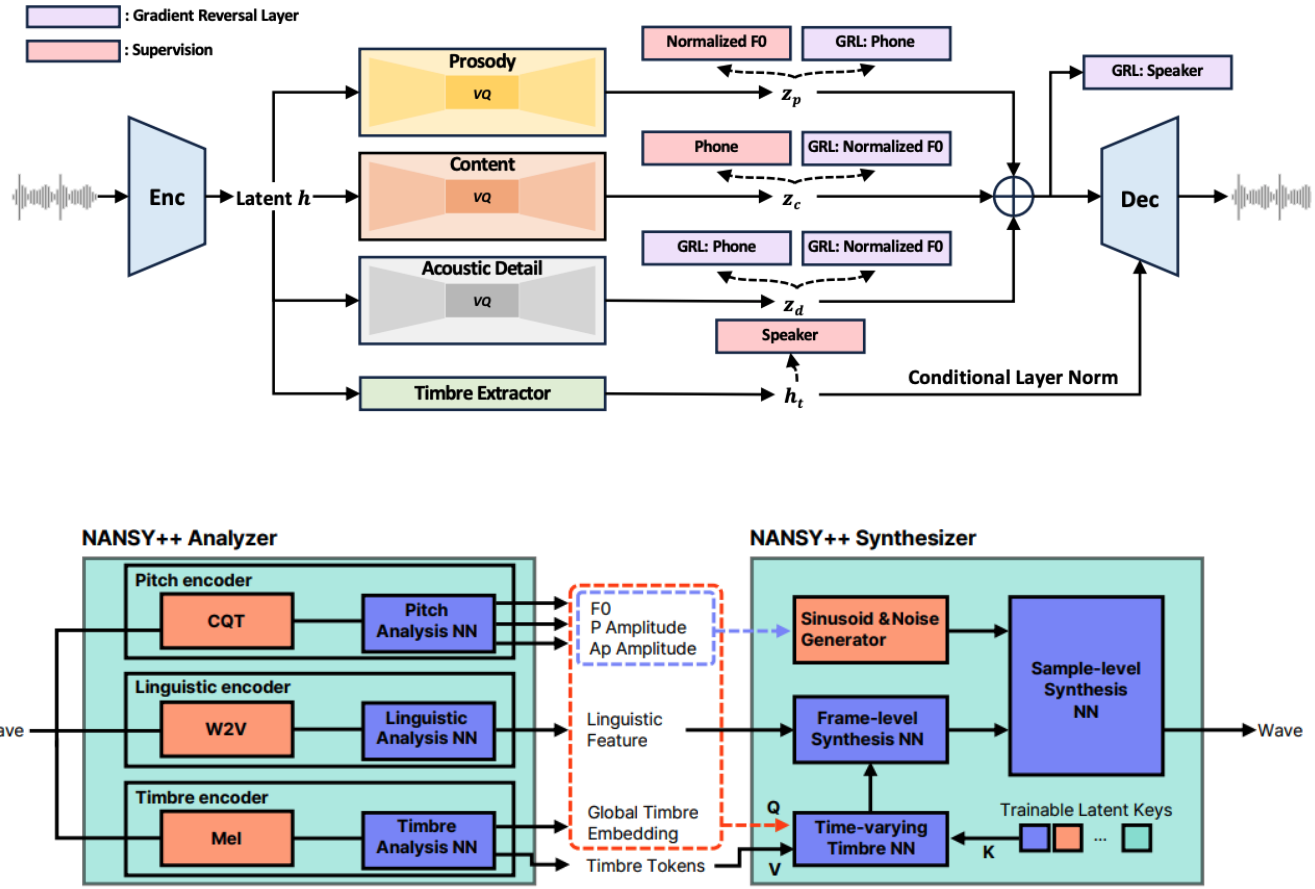
- **FACodec 实战效果：VALL-E**



	Sim-O / R ↑	WER↓	CMOS↑	SMOS↑
VALL-E + FACodec	**0.57 / 0.65**	**5.60**	**+0.24**	**3.61**
VALL-E♦	0.47 / 0.51	6.11	0.00	3.46

- Sim-O: 与 original prompt 之间的音色相似度
- Sim-R: 与 reconstructed prompt 之间的音色相似度

Comparative Analysis: NANSY++



对比项	FACodec	NANSY++
分解属性	• Prosody • Content • Timbre • Detail	• Pitch • Linguistic • Timbre
解耦方案	• Information-Bottleneck • Supervision • GRL • Structured Dropout	• Information-Perturbation • Contrastive Learning
可借鉴点	• 解耦更多属性 • 各属性的表征离散化	• **解耦不需任何要监督信息**

注：Speech-Resynthesis (2021) 等论文同样采用解耦的思想，
只不过对各属性解耦方法有所不同，详见龙哥 VC 的 PPT

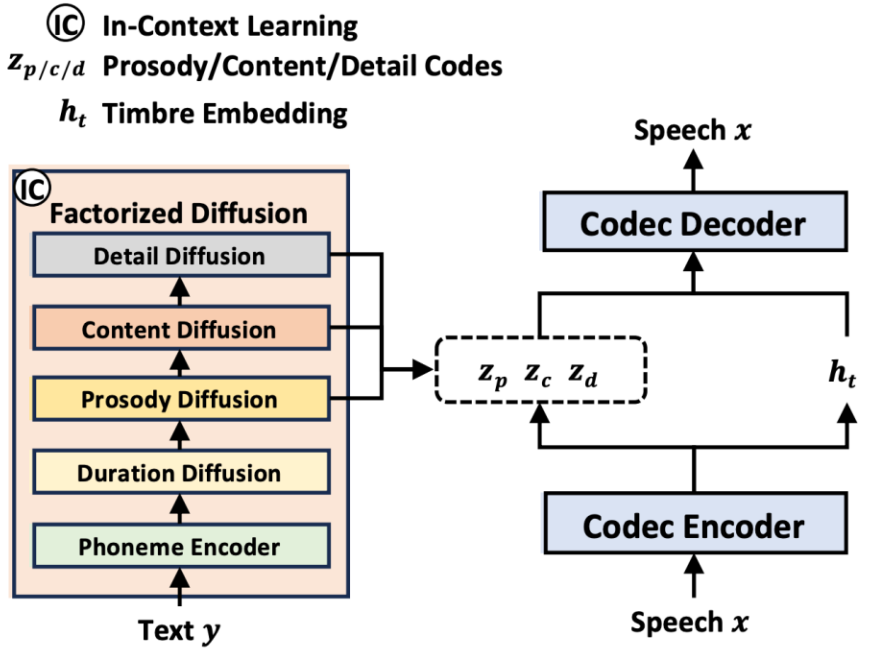
Zero-Shot TTS with **Factorized Diffusion Models**

NaturalSpeech3: Factorized Diffusion Models

- **Zero-Shot TTS 问题简化：**
  - 基于 FACodec，生成一条音频只需获得 prosody/content/detail/timbre 四个属性即可
    - timbre 可以直接从目标说话人的音频抽取得到
  - Zero-Shot TTS 转换为对三个属性 codec 的预测（分而治之）
  - 优势：三个属性解耦之后，支持使用不同的 prompt

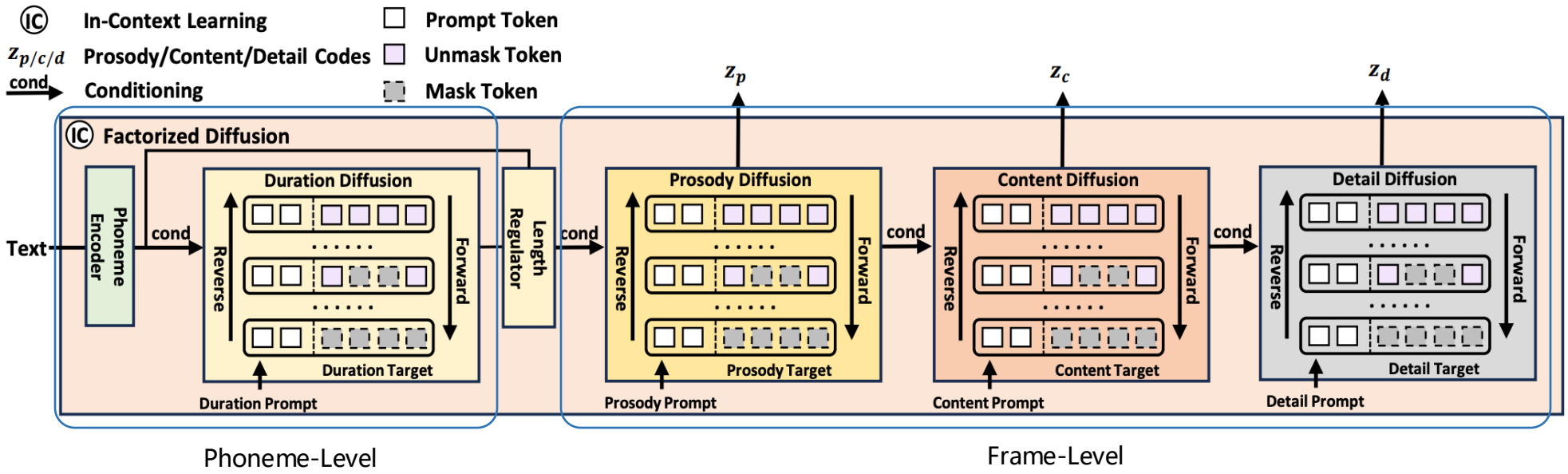
- **问题定义：**
  - 基于「音素序列」生成「多层 codec 」的预测任务
  - 候选模型：VALL-E, Spear-TTS/SoundStorm…（具备 prompt 能力即可）

- NaturalSpeech3 采用一种支持 prompt 的『新』模型
  - **Discrete Diffusion**



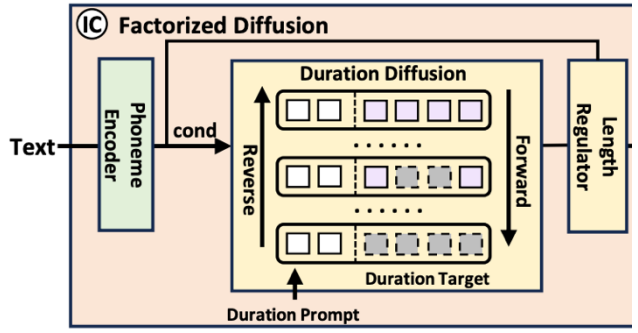
NaturalSpeech3: Factorized Diffusion Models

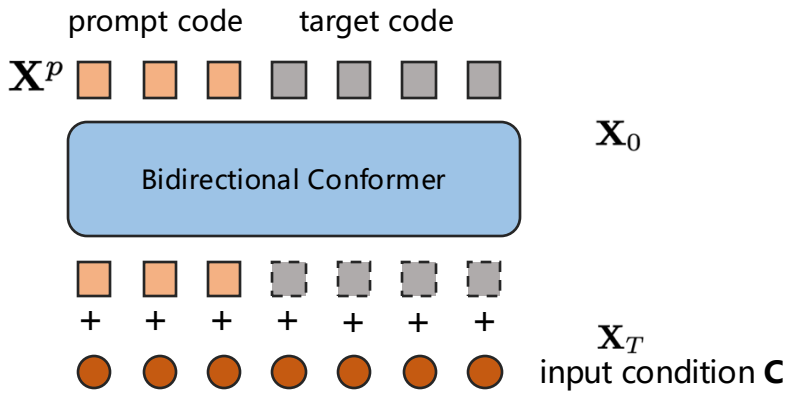
- **模型结构初探**



NaturalSpeech3: Factorized Diffusion Models

- **Discrete Diffusion （离散空间的 Diffusion）**





前向过程：mask 一定比例的 token（类比加噪）

反向过程：还原一部分被 mask 的 token（类比去噪）

0 时刻状态：$\mathbf{X}_0 = \mathbf{X}$ 真实的 token 序列

T 时刻状态：$\mathbf{X}_T$ 完全被 mask 的序列（T 是预先设置的 step 数）

t 时刻状态：$\mathbf{X}_t = \mathbf{X} \odot \mathbf{M}_t$

前向过程：加噪策略

$$\mathbf{M}_t = [m_{t,i}]_{i=1}^N \quad m_{t,i} \overset{iid}{\sim} \text{Bernoulli}(\sigma(t)) \quad \sigma(t) \in (0,1]$$

$m_{t,i} = 1$ 的位置表示被替换为 mask token

$$\sigma(t) = \sin(\frac{\pi t}{2T}), t \in (0, T]$$

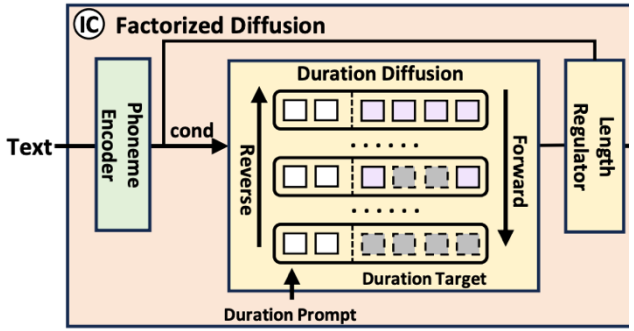
随着时间 t 增大，被 mask 的 token 比例增大，直到时间 T，σ(T)=1

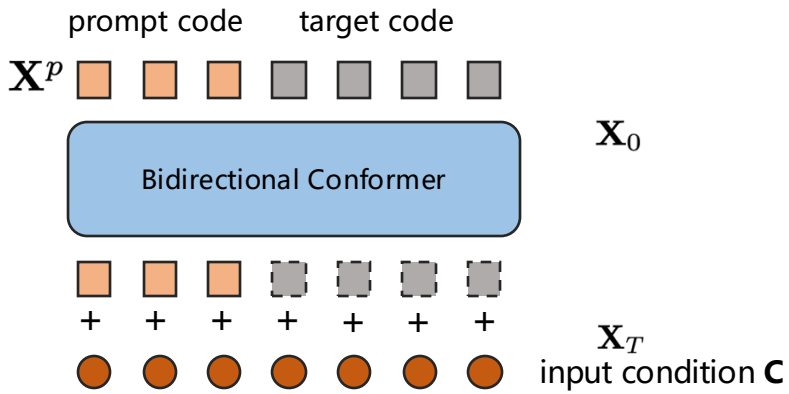
反向过程：（模型实际建模）

- 理论上从 $q(\mathbf{X}_{t-\Delta t}|\mathbf{X}_0, \mathbf{X}_t)$ 分布中采样得到 $\mathbf{X}_{t-\Delta t}$，但 $\mathbf{X}_0$ 实际未知
- 使用模型对 $p_\theta(\mathbf{X}_{t-\Delta t}|\mathbf{X}_t)$ 进行建模
- 增加额外的 prompt 和输入条件 C，实际建模的是 $p_\theta(\mathbf{X}_0|\mathbf{X}_t, \mathbf{X}^p, \mathbf{C})$

NaturalSpeech3: Factorized Diffusion Models

- **Discrete Diffusion （离散空间的 Diffusion）**





上一页结论：考虑 prompt 和输入条件 C，实际建模的是 $p_\theta(\mathbf{X}_0|\mathbf{X}_t, \mathbf{X}^p, \mathbf{C})$

训练目标

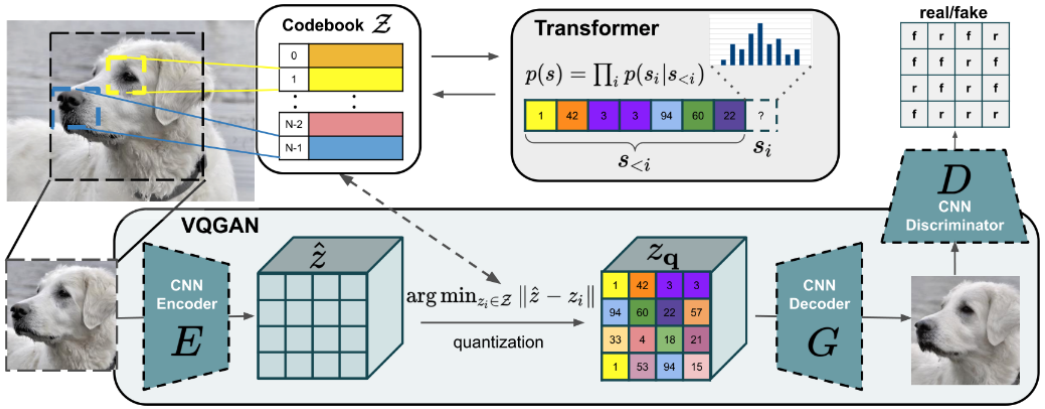
$$\mathcal{L}_{\text{mask}} = \mathop{\mathbb{E}}_{\mathbf{X}\in\mathcal{D}, t\in[0,T]} -\sum_{i=1}^{N} m_{t,i} \cdot \log(p_\theta(x_i|\mathbf{X}_t, \mathbf{X}^p, \mathbf{C}))$$

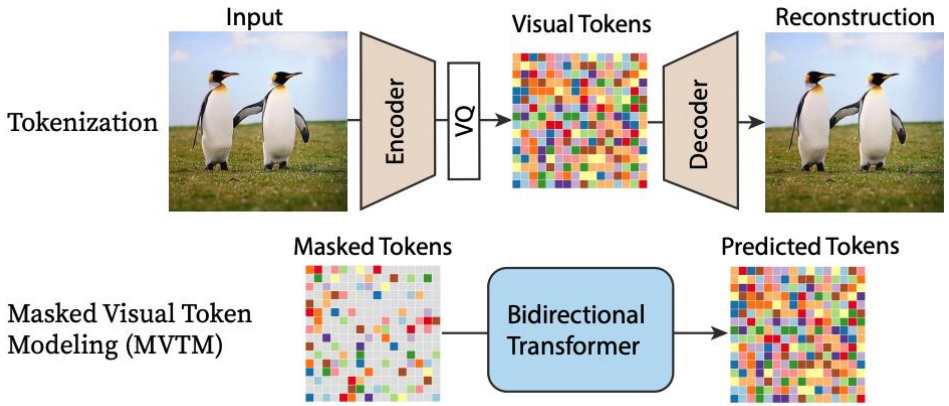
推理过程（从 $\mathbf{X}_t$ 生成 $\mathbf{X}_{t-\Delta t}$ ）

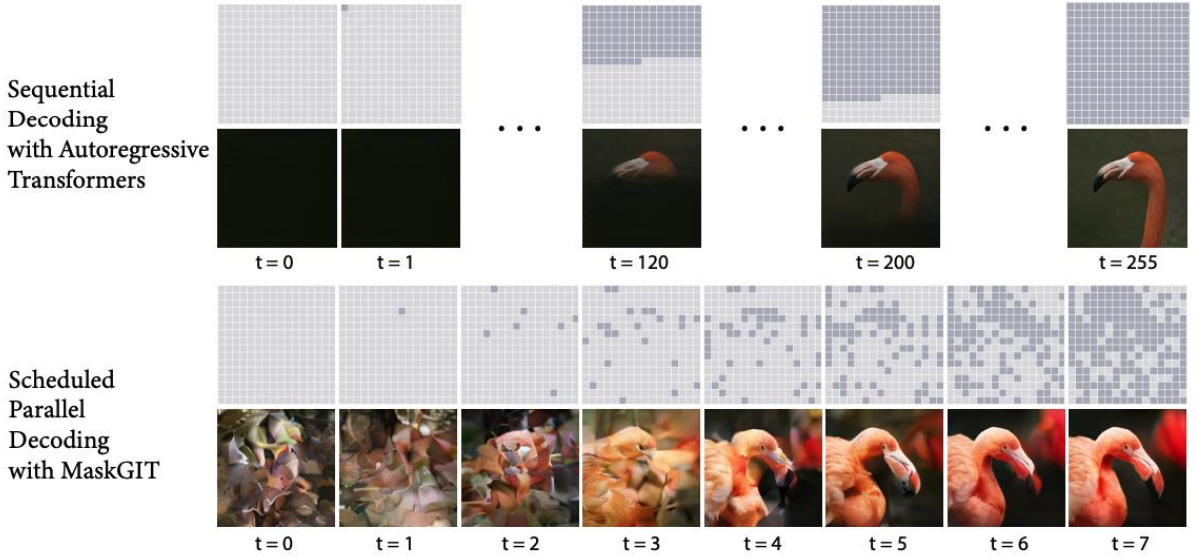
- 已知 $\mathbf{X}_t$，先从 $p_\theta(\mathbf{X}_0|\mathbf{X}_t, \mathbf{X}^p, \mathbf{C})$ 中采样得到 $\hat{\mathbf{X}}_0$
- 将 $\hat{\mathbf{X}}_0$ 中置信分数最低的 $\lfloor N \cdot \sigma(t-\Delta t) \rfloor$ 个 token 重新 mask
- 注意：$\hat{\mathbf{X}}_0$ 中没有被重新 mask的位置，置信分数为 1（避免已经生成好的 token 被重新 mask）

Discrete Diffusion? MaskGIT/SoundStorm!

From MaskGIT to Discrete Diffusion







From MaskGIT to Discrete Diffusion

- **第一阶段：**VQ-GAN 训练 Image Tokenizer
- **第二阶段：**Masked Visual Token Modeling（MVTM）
  - 训练过程：
    - 先从 0 ~ 1 之间选择一个概率值（mask schedule） $\gamma(r) \in (0, 1]$
    - 从 Y 中随机选择 个 token，进行 mask $\mathbf{Y} = [y_i]_{i=1}^N \quad \mathbf{M} = [m_i]_{i=1}^N$
    - 训练目标是预测被 mask 的 token

$$\mathcal{L}_{\text{mask}} = -\mathop{\mathbb{E}}_{\mathbf{Y}\in\mathcal{D}}\Big[\sum_{\forall i\in[1,N], m_i=1} \log p(y_i|Y_{\overline{\mathbf{M}}})\Big]$$

  - 推理过程：迭代式并行解码
    - 目标：所有 token 均被 mask → 预测出每一个被 mask 的 token
    - 一步解码：与训练过程不一致，效果不很好
    - 迭代式解码
      - 对被 mask 的 token，预测概率分布
      - 每个 token 从概率分布中采样（采样到的结果对应的概率为置信分数）
      - 根据时间 t 和 mask schedule, 计算下一步被 mask 的 token 个数 n $n = \lceil \gamma(\frac{t}{T})N \rceil$
      - 根据置信分数排序，选择分数最低的 n 个 token 进行 mask
      - 对未被 mask 的 token，置信分数为 1（之后不再改变）
      - 重复以上步骤

From MaskGIT to Discrete Diffusion

- **关于 masking schedule 的讨论**
  - mask schedule 在训练和推理阶段都会用到
  - 训练阶段：随机选择 r，模拟不同 mask 比例下的情况（不用考虑推理时有多少步）
  - 推理阶段：
    - 采样总步数为 T：0 时刻表示完全 mask，T 时刻生成完毕
    - mask 的比例随着采样步数 t 的增加，逐渐变小（直至最后一步不再 mask）
    - Mask schedule function 需要是从 0 到 1 的单调递减函数
- **几种不同的 mask schedule function**
  - Linear function
  - Concave function: cosine, square, cubic, exponential（初期下降慢，后期下降快）
  - Convex function: square root, logarithmic （初期下降快，后期下降慢）

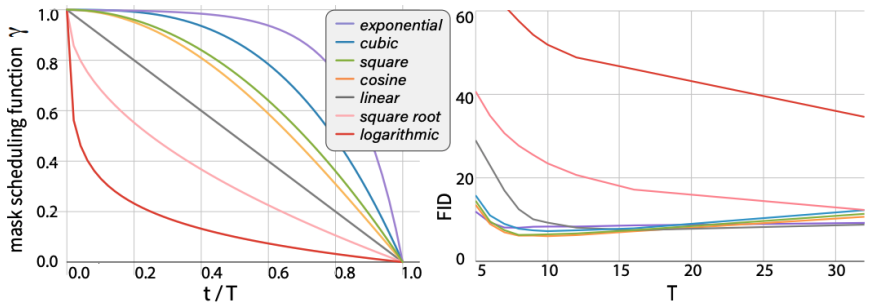


Figure 8. **Choices of Mask Scheduling Functions** $\gamma(\frac{t}{T})$, and **number of iterations** T. On the left, we visualize seven functions

cosine schedule is the best

From MaskGIT to Discrete Diffusion

- **MaskGIT is better and faster than VQ-GAN**
  - Better: 使用了双向的 context 信息
  - Faster: 从 AR 变为了 NAR

Model	FID ↓	IS ↑	Prec ↑	Rec ↑	# params	# steps	CAS ×100 ↑	
ImageNet 256×256							Top-1 (76.6)	Top-5 (93.1)
DCTransformer [32] ▫	36.51	n/a	0.36	**0.67**	738M	>1024		
BigGAN-deep [4]	6.95	**198.2**	**0.87**	0.28	160M	1	43.99	67.89
Improved DDPM [33]▫	12.26	n/a	0.70	0.62	280M	250		
ADM [12]▫	10.94	101.0	0.69	0.63	554M	250		
VQVAE-2 [37]▫	31.11	∼45	0.36	0.57	13.5B†	5120	54.83	77.59
VQGAN [15]▫	15.78	78.3	n/a	n/a	1.4B	256		
VQGAN*	18.65	80.4	0.78	0.26	227M	256	53.10	76.18
MaskGIT (Ours)	**6.18**	182.1	0.80	0.51	227M	*8*	**63.14**	**84.45**
ImageNet 512×512								
BigGAN-deep [4]	8.43	**232.5**	**0.88**	0.29	160M	1	44.02	68.22
ADM [12]▫	23.24	58.06	0.73	**0.60**	559M	250		
VQGAN*	26.52	66.8	0.73	0.31	227M	1024	51.29	74.24
MaskGIT (Ours)	**7.32**	156.0	0.78	0.50	227M	*12*	**63.43**	**84.79**

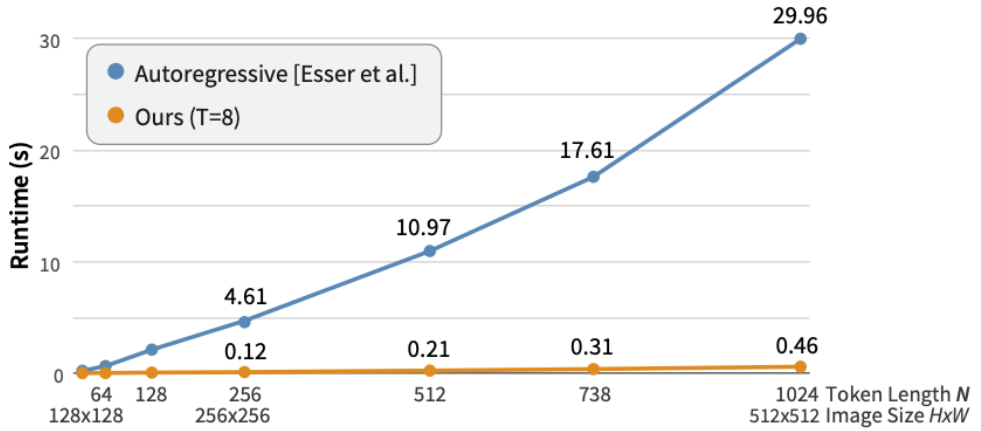
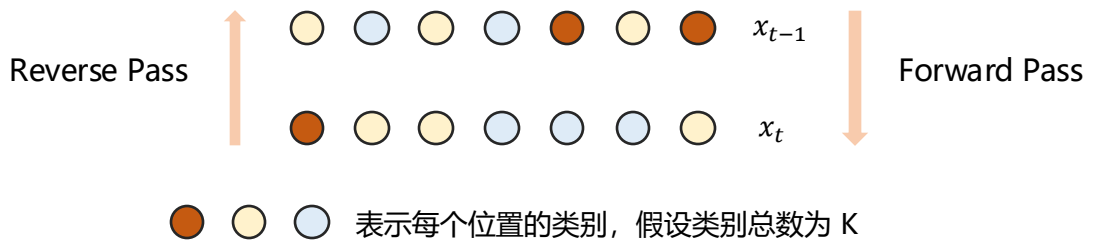


Figure 4. **Transformer wall-clock runtime comparison between VQGAN [15] and ours.** All results are run on a single GPU.

From MaskGIT to Discrete Diffusion

- **Discrete Diffusion (General Case)**



用转移在矩阵 $\boldsymbol{Q}_t$ 表示 forward pass 中，上一时刻 $t-1$ 到下一时刻 t 在 K 个不同类别之间的转移概率矩阵

- 转移矩阵为 K × K

$$[\boldsymbol{Q}_t]_{ij} = q(x_t = j | x_{t-1} = i)$$

- 转移概率计算

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathrm{Cat}(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_{t-1}\boldsymbol{Q}_t)$$

- 在时间 t 角度上，增加马尔可夫性假设

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$$

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathrm{Cat}\left(\boldsymbol{x}_t; \boldsymbol{p} = \boldsymbol{x}_0\overline{\boldsymbol{Q}}_t\right), \quad \text{with} \quad \overline{\boldsymbol{Q}}_t = \boldsymbol{Q}_1\boldsymbol{Q}_2 \ldots \boldsymbol{Q}_t$$

从 0 时刻到当前 t 时刻的转移矩阵

From MaskGIT to Discrete Diffusion

- **不同的转移概率概率矩阵**
- **1. Uniform Discrete Diffusion**

$$[\boldsymbol{Q}_t]_{ij} = \begin{cases} 1 - \frac{K-1}{K}\beta_t & \text{if} \quad i = j \\ \frac{1}{K}\beta_t & \text{if} \quad i \neq j \end{cases} \qquad \boldsymbol{Q}_t = \begin{bmatrix} \alpha_t + \beta_t & \beta_t & \cdots & \beta_t \\ \beta_t & \alpha_t + \beta_t & \cdots & \beta_t \\ \vdots & \vdots & \ddots & \vdots \\ \beta_t & \beta_t & \cdots & \alpha_t + \beta_t \end{bmatrix}$$

- **2. Uniform Diffusion with absorbing state**
  - 除了 K 个类别之外，增加 [MASK] 作为吸收状态
  - **BERT:** 80% mask, 10% 替换为其他 Token, 10% 保持不变
  - 结论：BERT(MLM) 是一种特殊的离散 diffusion

$$[\boldsymbol{Q}_t]_{ij} = \begin{cases} 1 & \text{if} \quad i = j = m \\ 1 - \beta_t & \text{if} \quad i = j \neq m \\ \beta_t & \text{if} \quad j = m, i \neq m \end{cases} \qquad \boldsymbol{Q}_t = \begin{bmatrix} \alpha_t + \beta_t & \beta_t & \beta_t & \cdots & 0 \\ \beta_t & \alpha_t + \beta_t & \beta_t & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix}$$

- **3. Mask Transition Diffusion**
  - 第 2 种情况的特例，只允许 mask 和保持不变
  - 结论：MaskGIT 是一种特殊的离散 diffusion

$$\boldsymbol{Q}_t = \begin{bmatrix} \beta_t & 0 & 0 & \cdots & 0 \\ 0 & \beta_t & 0 & \cdots & 0 \\ 0 & 0 & \beta_t & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix}$$

- **4. Auto Regressive Models are Discrete Diffusion Models...**

From MaskGIT to Discrete Diffusion

- **不同的转移概率概率矩阵**
- **1. Uniform Discrete Diffusion（U)**

- **2. Uniform Diffusion with absorbing state （MU)**
  - 除了 K 个类别之外，增加 [MASK] 作为吸收状态
  - **BERT:** 80% mask, 10% 替换为其他 Token, 10% 保持不变
  - 结论：BERT(MLM) 是一种特殊的离散 diffusion

- **3. Mask Transition Diffusion（M)**
  - 第 2 种情况的特例，只允许 mask 和保持不变
  - 结论：MaskGIT 是一种特殊的离散 diffusion

TABLE IV
ABLATION STUDY FOR THREE DIFFERENT TRANSITION MATRICES. FURTHERMORE, WE ALSO DISCUSS THE EFFECT OF THE FINAL MASK RATE $\overline{\gamma}_T$ ON MASK AND UNIFORM MATRIX. U DENOTES THE UNIFORM MATRIX, M DENOTES THE MASK MATRIX.

ID	Matrix	Mask rate	FID↓	KL↓	SPICE↑	CIDEr↑
1	U	0	10.14	4.31	0.101	0.35
2	MU	0.1	10.63	4.47	0.093	0.29
3		0.3	10.64	4.35	0.099	0.34
4		0.5	10.75	4.31	0.096	0.32
5		0.7	9.84	4.37	0.102	0.34
6		0.9	**9.76**	**4.21**	**0.103**	**0.36**
7	M	1	11.5	4.46	0.103	0.34

增加 Uniform Diffusion

- 建模目标更难，帮助更好的捕捉 context 信息，而不仅仅是关注 mask token
- 采样时如果某一步局部出现错误的结果，也能减少错误累积

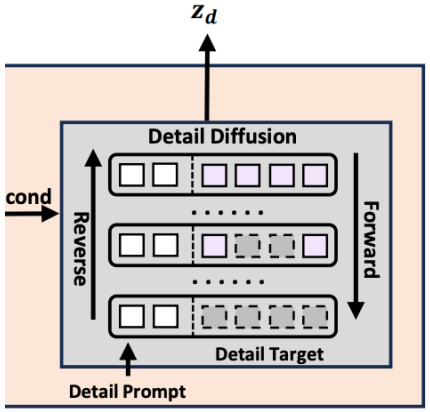
$$\boldsymbol{Q}_t = \begin{bmatrix} \alpha_t + \beta_t & \beta_t & \beta_t & \cdots & 0 \\ \beta_t & \alpha_t + \beta_t & \beta_t & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix}$$

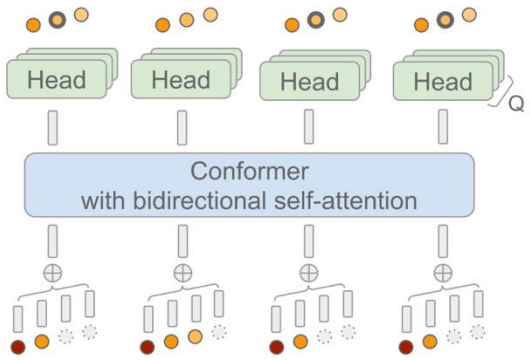
For the mask and uniform transition matrix, we linearly increase $\overline{\gamma}_t$ and $\overline{\beta}_t$ from 0 to 0.9 and 0.1, and decrease $\overline{\alpha}_t$ from 1 to 0.

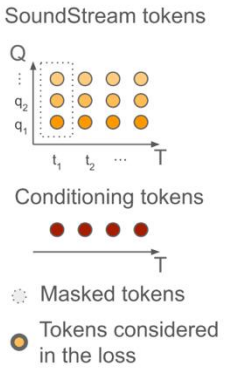
NaturalSpeech3: Factorized Diffusion Models

- **多层 RVQ Discrete Diffusion**

  类比 SoundStorm







训练流程（coarse-to-fine）

- 1. 对每一条音频，随机采样一个 $t \in \{1, 2, ..., T\}$
  - T 表示输入序列最大长度
  - 0~ t 作为 prompt； t ~ T 作为训练的 target
- 2. 随机采样一个 q-level ~ $\{1, Q\}$ (content $Q = 2$; detail $Q = 3$)
- 3. 针对 q-level 的 token 构造 0-1 mask 序列

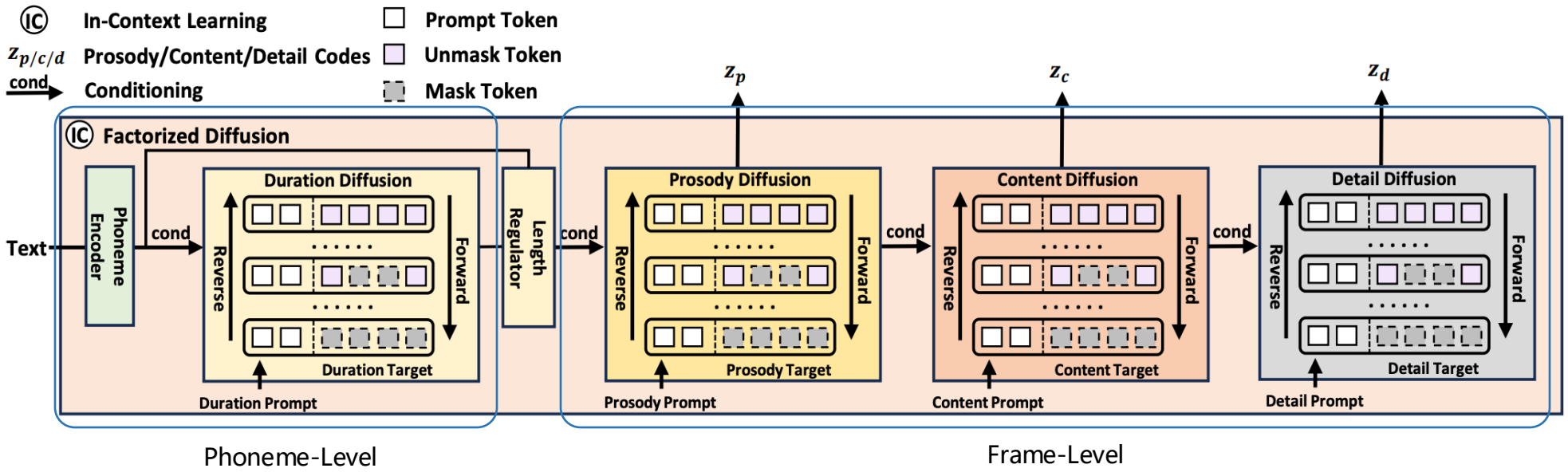
  $\mathbf{M}_t = [m_{t,i}]_{i=1}^{N} \quad \mathbf{X}_t = \mathbf{X} \odot \mathbf{M}_t$

SoundStorm	NaturalSpeech3
$M_i \sim \text{Bernoulli}(p) \quad p = \cos(u)$ $u \sim \mathcal{U}[0, \pi/2]$	$m_{t,i} \overset{iid}{\sim} \text{Bernoulli}(\sigma(t)) \quad \sigma(t) \in (0, 1]$ $\sigma(t) = \sin(\frac{\pi t}{2T}), t \in (0, T]$

- 4. mask 操作
  - prompt 部分：所有 Q 层 token 都不 mask；
  - target 部分：
    - q-level 当前层：根据生成的 $\mathbf{M}_t$ ， 对 token 进行 mask
    - q-level 以下：所有 token 全不 mask
    - q-level 以上：所有 token 全部 mask
- 5. 训练 Loss 使用交叉熵，只针对 q-level 被 mask 的 token 进行计算

NaturalSpeech3: Factorized Diffusion Models

- **模型结构细节**
  - Phoneme Encoder: 6 层 Transformer, 8 head
  - Phone-Level Diffusion (duration/phone-level prosody): 共享的 6 层 Transformer, 8 head
  - Frame-Level Diffusion (prosody/content/detail): 共享的 12 层 Transformer, 8 head



NaturalSpeech3: Factorized Diffusion Models

- **Factorized Diffusion 相比于 SoundStorm 的改进点**

- **改进一：Time Embedding**
  - Factorized Diffusion Transformer 的每个 LayerNorm 替换为 AdaCLN
    - Time embedding 作为 CLN 的 condition

- **改进二：Classifier-Free Guidance (CFG) with Rescale**
  - 训练时
    - 15% 的概率不使用 prompt 对 masked token 进行预测
  - 推理时
    - 不使用 prompt， unconditional 生成
    - 使用 prompt， conditional 生成
    - 两次生成在 logits 上进行插值
  - CFG with Rescale

$$g_{\text{uncond}} = g(\mathbf{X})$$

$$g_{\text{cond}} = g(\mathbf{X}|\mathbf{X}^p)$$

$$g_{\text{cfg}} = g_{\text{cond}} + \alpha \cdot (g_{\text{cond}} - g_{\text{uncond}})$$

$$g_{\text{final}} = \text{std}(g_{\text{cond}}) \times g_{\text{cfg}} / \text{std}(g_{\text{cfg}})$$

$$x_{cfg} = x_{neg} + w(x_{pos} - x_{neg})$$

$$\sigma_{pos} = \text{std}(x_{pos}), \quad \sigma_{cfg} = \text{std}(x_{cfg})$$

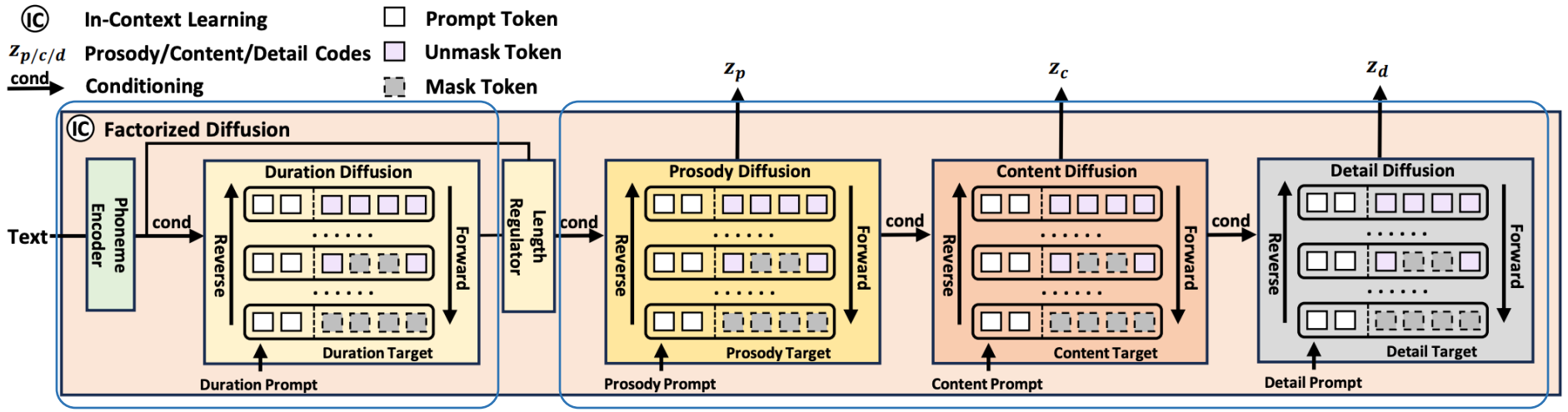
$$x_{rescaled} = x_{cfg} \cdot \frac{\sigma_{pos}}{\sigma_{cfg}}$$

$$x_{final} = \phi \cdot x_{rescaled} + (1 - \phi) \cdot x_{cfg}$$

NaturalSpeech3: Factorized Diffusion Models

- **模型推理**
  - phoneme-level prosody diffusion: 4 × 2 = 8 次采样
  - duration diffusion: 4 次采样（不加 CFG）
  - frame-level prosody/content/detail 共 6 层
    - 每层 4 × 2 = 8 次采样（增加 CFG）
  - 总采样次数：8 + 4 + 6 × 8 = 60 次
  - 生成音频：6 层 Codec + 目标说话人的参考音频提取的 timbre embedding，通过 FACodec 重建音频

- **NaturalSpeech3 v.s. Spear-TTS s1 + SoundStorm**
  - Phone-Encoder + Duration Diffusion + Length Regulator ≈ SpearTTS s1
  - Prosody+Content+Detail Diffusion ≈ SoundStorm



NaturalSpeech3: Factorized Diffusion Models

- **训练数据**
  - LibriLight: 6 万小时 16kHz 的英文有声书数据
  - Speaker 数目：7000+
  - Phone 序列获取：内部 ASR + g2p
  - Duration 获取：内部强制对齐工具

- **评测数据**
  - LibriSpeech test-clean：40 个 speaker，每人一条测试音频
  - RAVDESS 英文不同**情感**语音数据集：24 个 speaker
    - 8 种情感，每种情感两个强度（normal/strong），论文选择的 strong 强度下的 8 个情感
  - 指标一：音色相似度
    - SIM-O / SIM-R：与原始 prompt (O) / FACodec 重建 prompt (R)之间的 WavLM-TDNN 音色相似度
  - 指标二：UTMOS： 音质评价指标
  - 指标三：WER：开源英文 ASR 模型
  - 指标四：韵律相似度
    - MCD（梅尔倒谱系数之间的衡量）
    - MCD-Acc（对合成音频进行情感分类，与 prompt 音频的情感类别之间的 top-1 准确率）

NaturalSpeech3: Factorized Diffusion Models

- **NaturalSpeech3 实验结果**

- 音色相似度与可懂度/稳健性

Table 1: The evaluation results for NaturalSpeech 3 and the baseline methods on LibriSpeech test-clean. ♠ means the results are obtained from the authors. ♥ means the results directly obtained from the paper. ♣ means the results are infered from offical checkpoints. ♦ means the reproduced results. Abbreviation: LT (LibriTTS), V (VCTK), LJ (LJSpeech), LL* (Librilight Small, Medium), EX (Expresso), MS (MSSS Kor), NI (NIKL Kor). Please refer to Appendix A.4 for more results on 1) WER inferred by an advanced ASR system, and 2) UTMOS, an automatic metric for MOS.

	Training Data	Sim-O ↑	Sim-R ↑	WER↓	CMOS↑	SMOS↑
Ground Truth	-	0.68	-	1.94	+0.08	3.85
VALL-E ♥	Librilight	-	0.58	5.90	-	-
VALL-E ♦	Librilight	0.47	0.51	6.11	-0.60	3.46
NaturalSpeech 2♠	Librilight	0.55	0.62	1.94	-0.18	3.65
Voicebox♠	Self-Collected (60kh)	0.64	0.67	2.03	-0.23	3.69
Voicebox♦	Librilight	0.48	0.50	2.14	-0.32	3.52
Mega-TTS 2♠	Librilight	0.53	-	2.32	-0.20	3.63
UniAudio♠	Mixed (165kh)	0.57	0.68	2.49	-0.25	3.71
StyleTTS 2♣	LT + V + LJ	0.38	-	2.49	-0.21	3.07
HierSpeech++♣	LT + LL* + EX + MS + NI	0.51	-	6.33	-0.41	3.50
NaturalSpeech 3	Librilight	**0.67**	**0.76**	**1.81**	**0.00**	**4.01**

- 韵律/情感相似度

Table 2: The evaluation results for NaturalSpeech 3 and the baseline methods on RAVDESS. ♠ means the results are obtained from the authors. ♣ means the results are inferred from official checkpoints. ♦ means the reproduced results. Abbreviation: Avg (average MCD), Acc (MCD-Acc).

	Avg↓	Acc↑	CMOS↑	SMOS↑
Ground Truth	0.00	1.00	+0.17	4.42
VALL-E ♦	5.03	0.34	-0.55	3.80
NaturalSpeech 2♠	4.56	0.25	-0.22	4.04
Voicebox♦	4.88	0.34	-0.34	3.92
Mega-TTS 2♠	4.44	0.39	-0.20	4.51
StyleTTS 2♣	4.50	0.40	-0.25	3.98
HierSpeech++♣	6.08	0.30	-0.37	3.87
NaturalSpeech 3	**4.28**	**0.52**	**0.00**	**4.72**

	MCD↓							
	neutral	calm	happy	sad	angry	fearful	disgust	surprised
Ground Truth	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
VALL-E ♦	3.97	4.75	4.83	5.51	5.19	5.29	5.45	5.29
Voicebox♦	3.93	4.90	4.96	4.93	5.01	5.03	5.34	4.89
NaturalSpeech 2♠	**2.77**	**3.51**	4.85	4.88	5.42	5.23	5.31	4.52
Mega-TTS 2♠	3.28	4.39	4.44	4.67	**4.21**	5.00	5.42	**4.14**
StyleTTS 2♣	3.41	4.38	4.40	4.64	4.80	4.69	5.10	4.57
HierSpeech++♣	5.54	6.55	5.78	5.84	6.37	6.17	6.74	5.62
NaturalSpeech 3	3.23	4.32	**4.26**	**4.41**	4.64	**4.25**	**4.80**	4.45

NaturalSpeech3: Factorized Diffusion Models

- **消融实验**
  - - factorization: 使用普通的 SoundStream Codec + duration diffusion + soundstream discrete diffusion
  - - cfg: 不增加 CFG，但为了可比性，每层采样的次数翻倍，仍然保持总采样次数为 60 次

Table 3: The ablation study of factorization and classifier-free guidance (cfg) on LibriSpeech test-clean.

	Sim-O / Sim-R ↑	WER↓	CMOS↑	SMOS↑
NaturalSpeech 3	**0.67 / 0.76**	**1.81**	**0.00**	**4.01**
- factorization	0.55 / 0.61	2.49	-0.25	3.59
- cfg	0.64 / 0.72	**1.81**	-0.06	3.80

- **韵律表征的有效性**
  - 使用前 20 维梅尔特征作为 prosody VQ 的输入

Table 4: The ablation study of prosody representation on RAVDESS. Denote "Mel 20 Bins" using the first 20 bins in the mel-spectrogram as the prosody representation.

	MCD Avg↓	MCD-Acc↑
NaturalSpeech 3	**4.28**	**0.52**
Mel 20 Bins	4.34	0.46

NaturalSpeech3: Factorized Diffusion Models

- **扩充数据量和参数量的实验**

Table 7: The performance of NaturalSpeech 3 on an internal test set, with 500M model size and different hours of training data.

	Sim-O↑	WER↓
1K	0.69	3.39
60K	0.72	3.03
200K	**0.73**	**2.83**

Table 8: The performance of NaturalSpeech 3 on an internal test set, with 200K hours of training data and different model sizes.

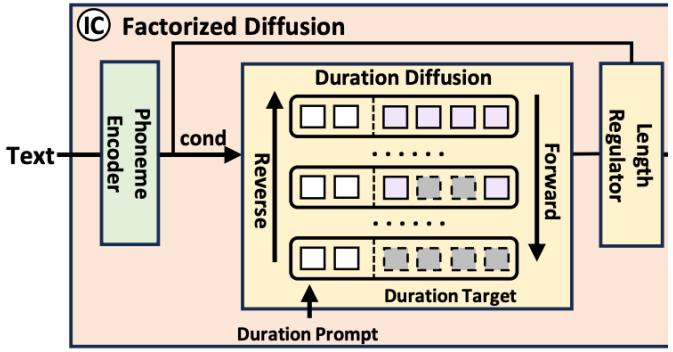
	Sim-O↑	WER↓
500M	0.73	2.83
1B	**0.75**	**2.62**

- **模型推理时间对比**

Models	NFE	RTF ↓	Sim-O ↑	Sim-R ↑	UTMOS ↑
NaturalSpeech 2	150	0.366	0.55	0.62	3.87
VALL-E	-	4.520	0.47	0.51	3.67
NaturalSpeech 3	60	0.296	**0.67**	**0.76**	**4.30**
NaturalSpeech 3 one-step	15	**0.067**	0.66	0.75	4.01

NaturalSpeech3: Factorized Diffusion Models

- **补充细节：Duration Diffusion with Phoneme-Level Prosody**



NaturalSpeech3 额外训练了一个 phone-level 的 prosody diffusion

- 训练时，phone-level prosody code 的获取
  - 从目标音频抽取 prosody VQ 前的 embedding
  - 根据对齐结果进行 phone-level pooling，从 codebook 中抽取 prosody code

- **模型一：phone-level prosody diffusion**
  - 输入：phoneme-encoder 的输出 + prompt 的 phone-level prosody code
  - 输出：target 区域的 phone-level prosody code

- **模型二：duration diffusion**
  - 输入：
    - phone-encoder 的输出
    - prompt 的 duration code
    - 预测得到的 target 区域的 phone-level 的 prosody code
  - 输出：target 的 duration code

NaturalSpeech3: Factorized Diffusion Models

- **Duration diffusion 模块的消融实验**

Table 11: The ablation results of the design of the duration predictor on LibriSpeech test-clean.

	Sim-O ↑	Sim-R ↑	WER↓	UTMOS↑
NaturalSpeech 3	**0.67**	**0.76**	**1.94**	**4.30**
Generation ablation	0.62	0.73	**1.94**	4.18
Objective ablation	0.62	0.72	2.38	4.13
Conditioning ablation	0.62	0.72	2.49	4.11
Prompting ablation	0.61	0.71	2.83	4.08

- Generation ablation: 4 步生成 → 1 步生成
- Objective ablation: CE 分类 loss → L2 回归 loss
- Conditioning ablation: 额外输入 phoneme-level prosody → 不加入 PL-prosody
- Prompting ablation: 有 prompt → 不加入 prompt

Takeaways

- **从建模的角度**
  - 理论的统一：Discrete Diffusion 包含了 MaskGIT/SoundStorm
  - Discrete Diffusion 作为 AR 的替换方案（VQ-Diffusion > MaskGIT > VQ-GAN）
    - Mega-TTS2: duration/prosody LLM 升级为 discrete diffusion
  - SoundStorm 替换为更一般情况的 Discrete Diffusion: https://github.com/yangdongchao/SoundStorm

- **SoundStorm 可以参考的优化点**
  - Codec 无缝替换为 FACodec
  - 转移矩阵将 uniform 与 mask 相结合：lucidrains 开源代码已支持（BERT Mask）
  - Classifier-Free Guidance on MaskGIT models （AR Models?）

- **从解耦的角度**
  - Information Bottleneck for Prosody VQ（Mega-TTS2）
  - 增加 supervision 和 GRL 增强解耦的效果

- 其他：RAVDESS 情感数据集

Comparative Analysis

- **MobileSpeech**

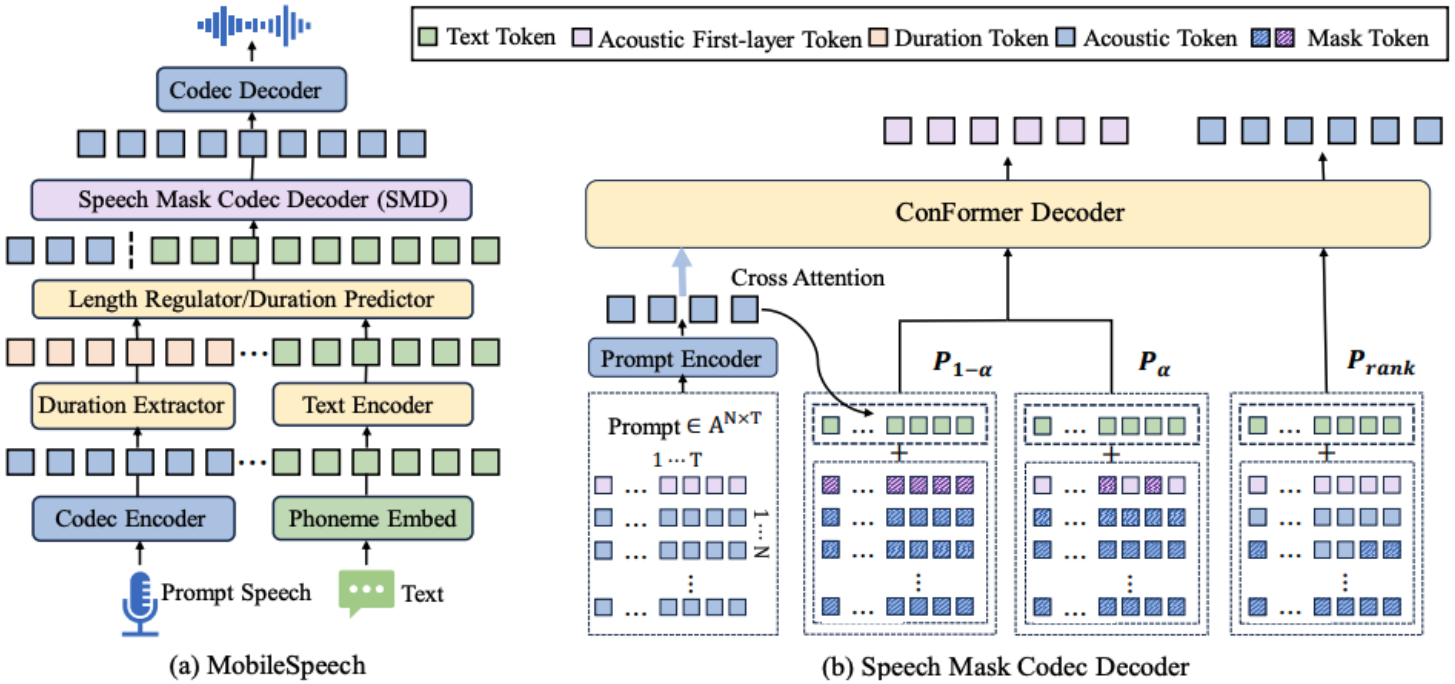


Figure 1: The overall architecture for MobileSpeech. In Figure (a) the Duration Extractor module is responsible for extracting prompt duration tokens from prompt acoustic tokens. SMD represents the generative module for target acoustic tokens. In Figure (b), we provide a detailed depiction of the multi-channel training process employed by the SMD module.

Comparative Analysis

- **UniCATS**
  - Semantic token: vq-wav2vec + k-means

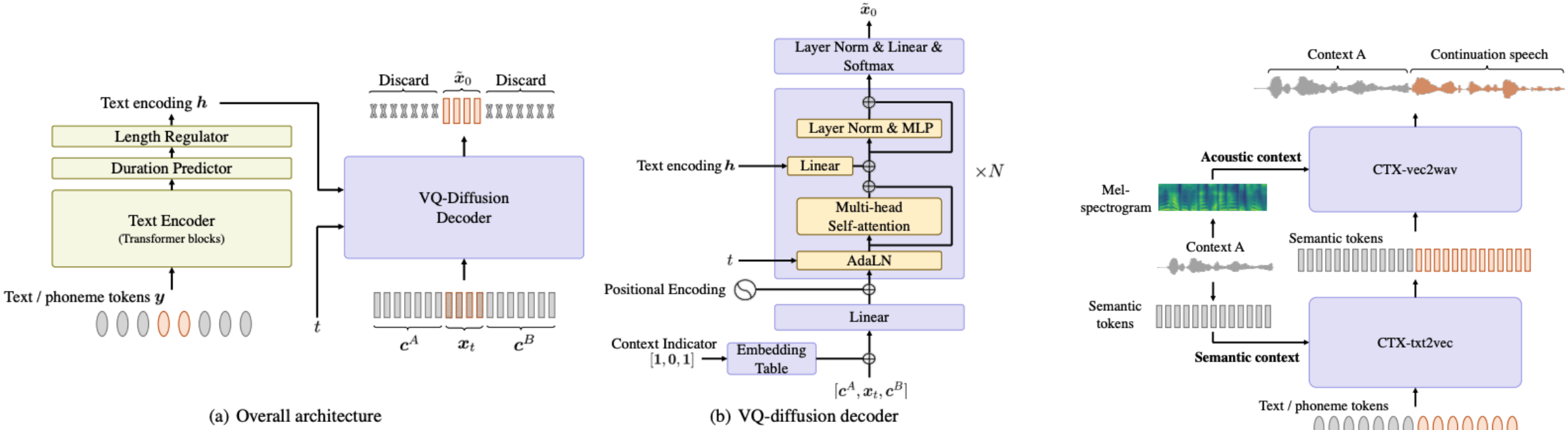


Figure 3: The model architecture of CTX-txt2vec with contextual VQ-diffusion.