

# Deep Learning School

бесплатно.



онлайн.



фундаментально.

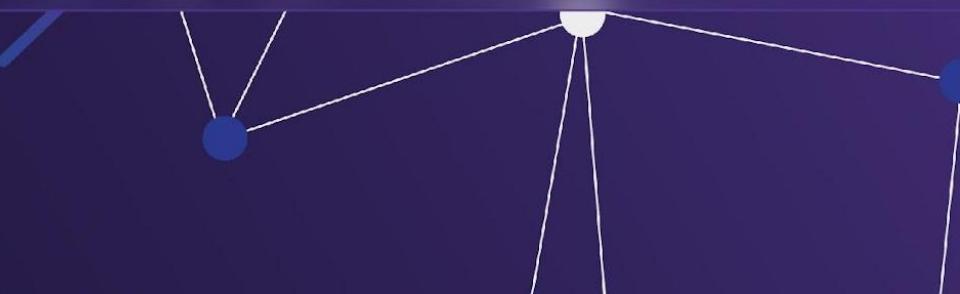
2024



# Генерация речи. Лекция 2

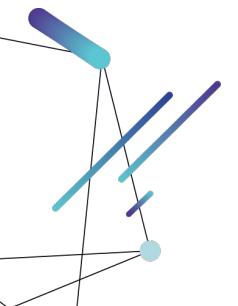
Садекова Таснима

Huawei

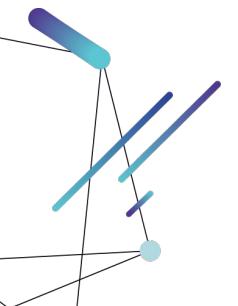


# Содержание

1. LM-based модели
2. Flow matching-based модели
3. Моделирование атрибутов речи (disentangled representations)
4. Смешанные модели
5. Основные тенденции и текущие задачи

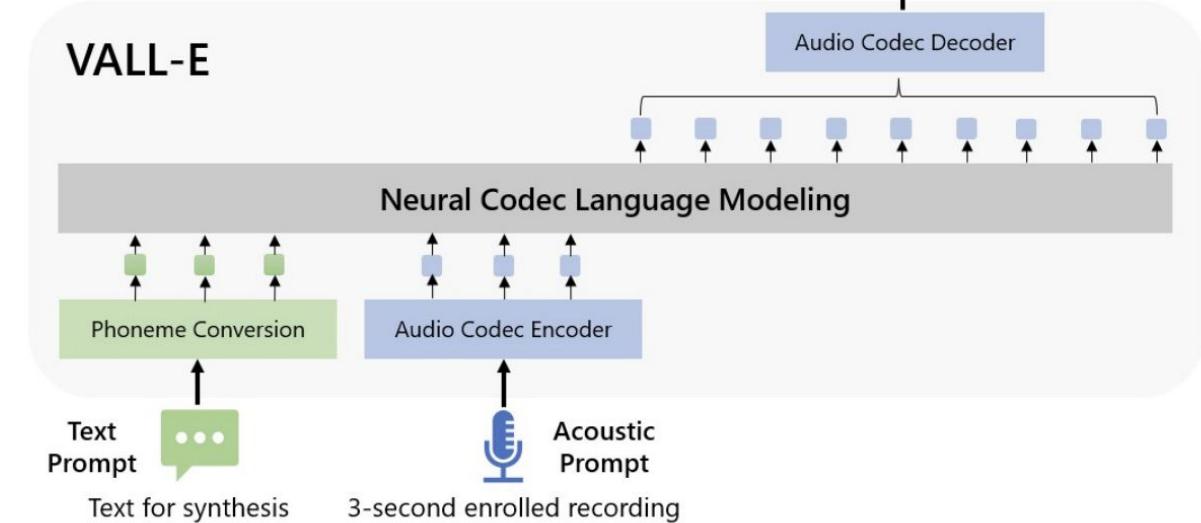


# LM-based модели



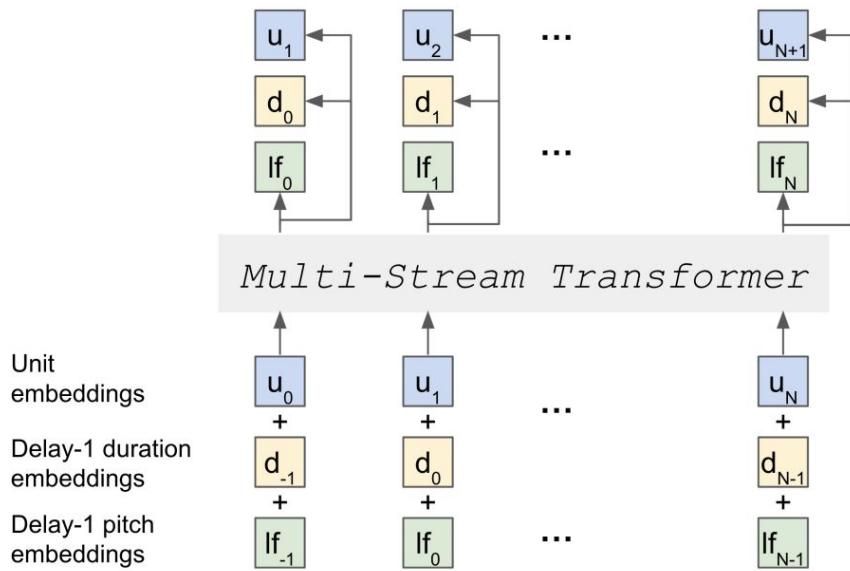
# VALL-E (2023)

- ❑ LM-based подход
- ❑ Авторегрессионная модель
- ❑ Дискретное представление аудио
- ❑ Идея из моделей генерации текста: вместо усложнения архитектуры увеличение модели и размера датасета
  - LibriLight: 60k часов
- ❑ In-context learning способности
  - Промпт – референсная запись голоса и текст



# VALL-E (2023)

*Дискретное моделирование в аудио*



## GSLM (generative spoken language model) (2022)

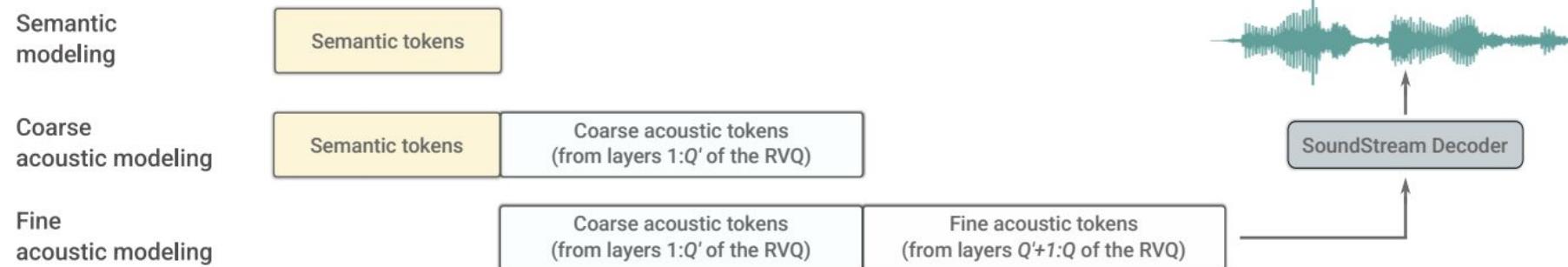
- аудио моделирование (без текста)
- аудио признаки:
  - семантика: HuBERT токены
  - просодия: длительности, квантизованные значения F0 (pitch)
- сценарии:
  - audio continuation
  - prosody continuation
- генерация аудио – специальная модель HiFi-GAN, обученная на HuBERT токенах, квантизованных значениях F0 (pitch) и speaker embeddings

# VALL-E (2023)

## Дискретное моделирование в аудио

[AudioLM](#) (2022) – аудио моделирование (без текста)

- аудио признаки:
  - семантика: w2v-BERT
  - акустика: токенизатор и детокенизатор SoundStream (50Hz, 12 кодбуков)
- иерархическое предсказание:
  - 1) semantic tokens – определяет текст
  - 2) coarse acoustic tokens – по семантическим токенам предсказывается 4 первых кода. Глобальная акустическая информация: голос, акустические условия записи, ... (flatten operation)
  - 3) fine acoustic tokens – по предыдущим coarse токенам предсказываются оставшиеся. Детальная акустическая информация
- Аудио восстанавливается декодером кодека
- сценарии использования
  - unconditional generation
  - audio continuation
  - acoustic modeling



# VALL-E (2023)

Датасет  $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}$ :

- $\mathbf{x} = \{x_0, x_1, \dots, x_L\}$  – последовательность **фонем**
- $\mathbf{y}$  – аудио, используемые для извлечения дискретных токенов  $\text{Encodec}(\mathbf{y}) = \mathbf{C}^{T \times 8}$

**Encodec:** 75Hz, 8 кодбуков, на основе SoundStream

Задача во время инференса:

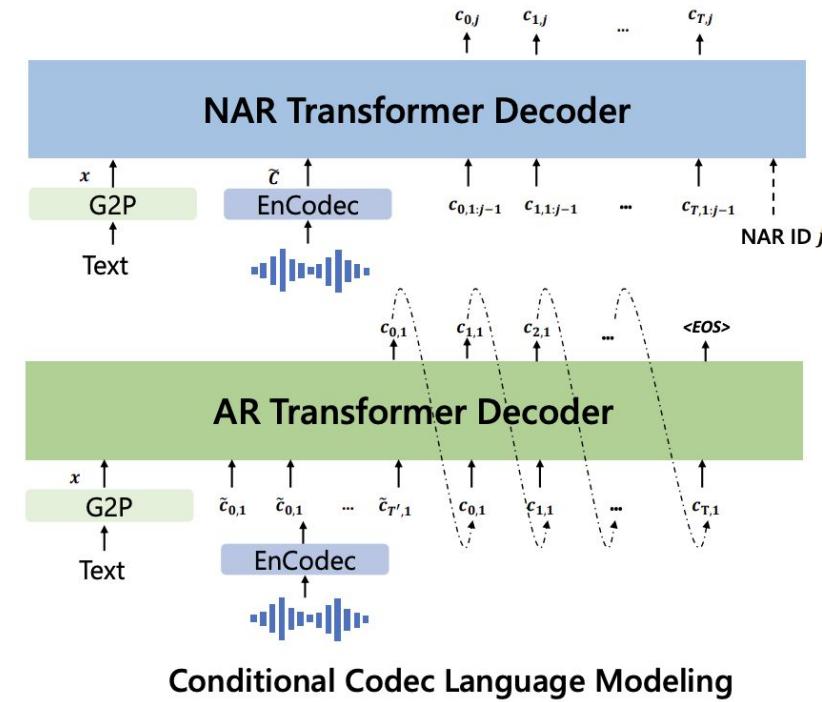
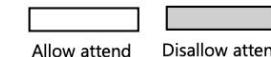
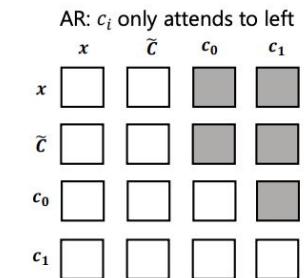
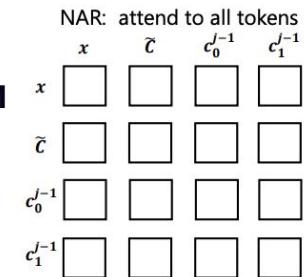
- вход: <text prompt> + <text target> + <codes prompt>
- выход: <codes target> (с копированием голоса, стиля)

Задача во время обучения:

- вход: <text> + <codes>
- выход: <codes>

Восстановление аудио с помощью декодера токенизатора

$$\text{Decodec}(\mathbf{C}) \approx \hat{\mathbf{y}}$$



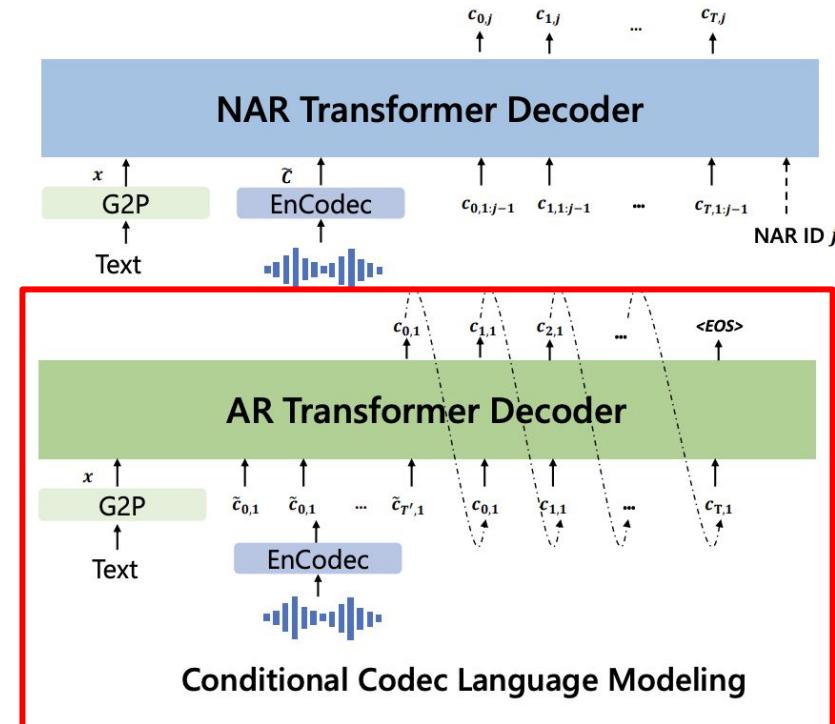
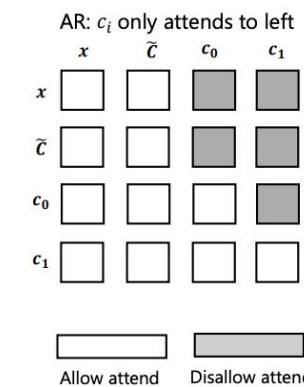
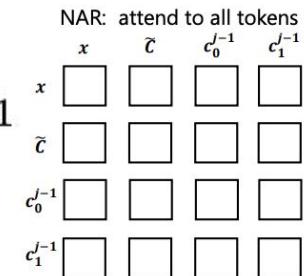
[Статья](#), [github](#)

# VALL-E (2023)

Задача моделирования делится на две (decoder-only LM):

## 1. AR: авторегрессионное предсказание первых кодов $\mathbf{c}_{:,1}$

- первые коды содержат больше всего информации
- отвечает за выравнивание
- $x$  преобразуется в вектора  $\bar{W}_x$
- с преобразуется в вектора  $W_a$
- $<EOS>$  токены после конца текста и кодов
- positional embedding отдельно для текста и кодов

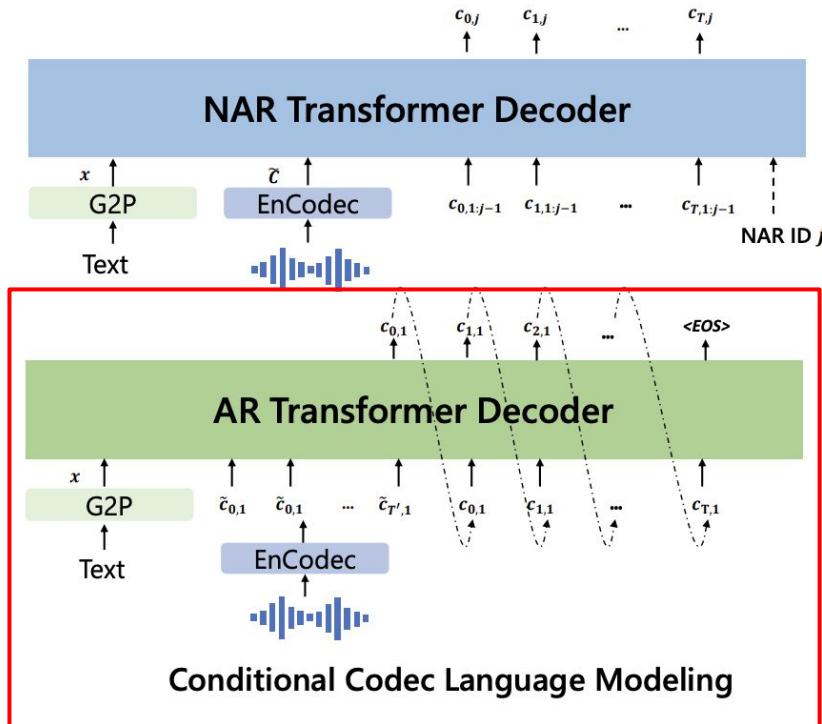
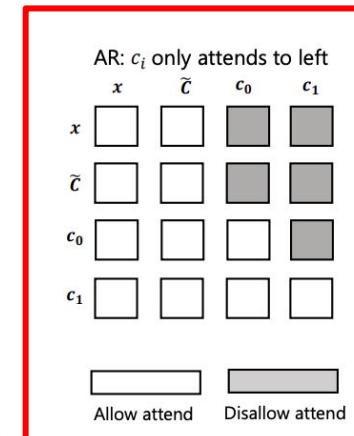
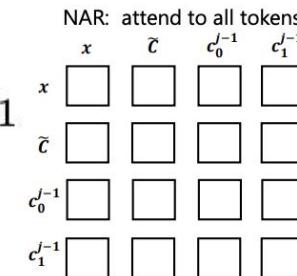


# VALL-E (2023)

Задача моделирования делится на две (decoder-only LM):

## 1. AR: авторегрессионное предсказание первых кодов $\mathbf{c}_{:,1}$

- первые коды содержат больше всего информации
- отвечает за выравнивание
- $x$  преобразуется в вектора  $\bar{W}_x$
- $c$  преобразуется в вектора  $\bar{W}_a$
- $<EOS>$  токены после конца текста и кодов
- positional embedding отдельно для текста и кодов
- causal attention mask
- задача – максимизация вероятности следующего токена (cross-entropy loss)



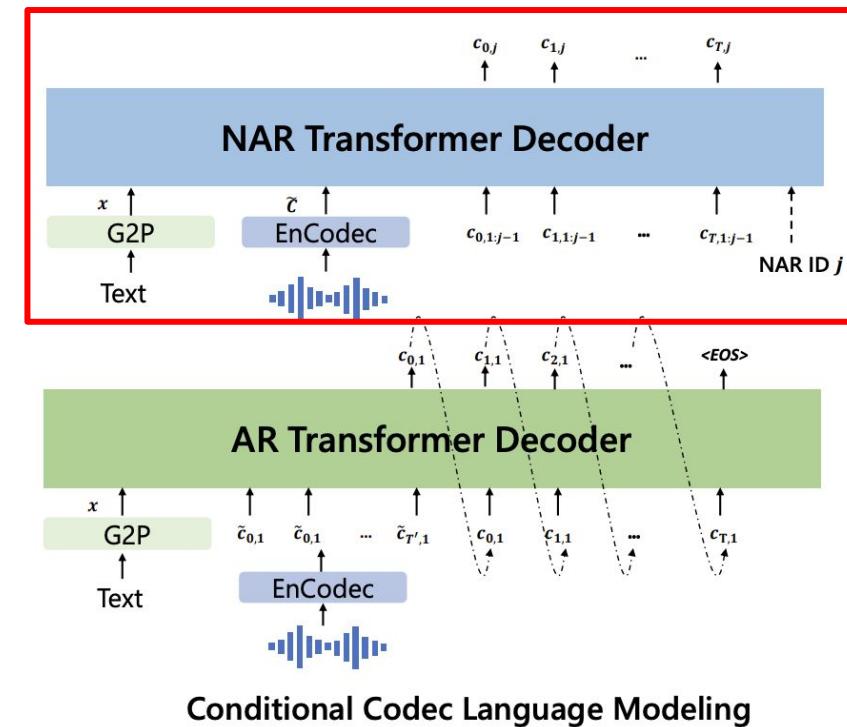
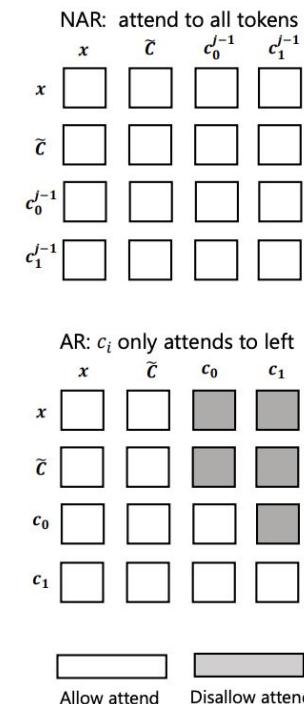
# VALL-E (2023)

Задача моделирования делится на две (decoder-only LM):

2. **NAR:** неавторегрессионное предсказание всех оставшихся токенов
  - 8 отдельных acoustic embedding layers
  - итеративное предсказание **7 раз** для кодов 2-8
  - вход  $i$ -ой итерации – **текст**, полное дискретное представление промпта и сумма эмбеддингов со всех предыдущих кодбуков для предсказываемого участка

$$(\mathbf{e}_x, \mathbf{e}_{\tilde{c}}, \mathbf{e}_{c_{:, <i}}) \quad e_{c_{t,j}} = W_a^j \odot c_{t,j}$$

$$\mathbf{e}_{c_t} = \sum_{j=1}^{i-1} e_{c_{t,j}}$$



# VALL-E (2023)

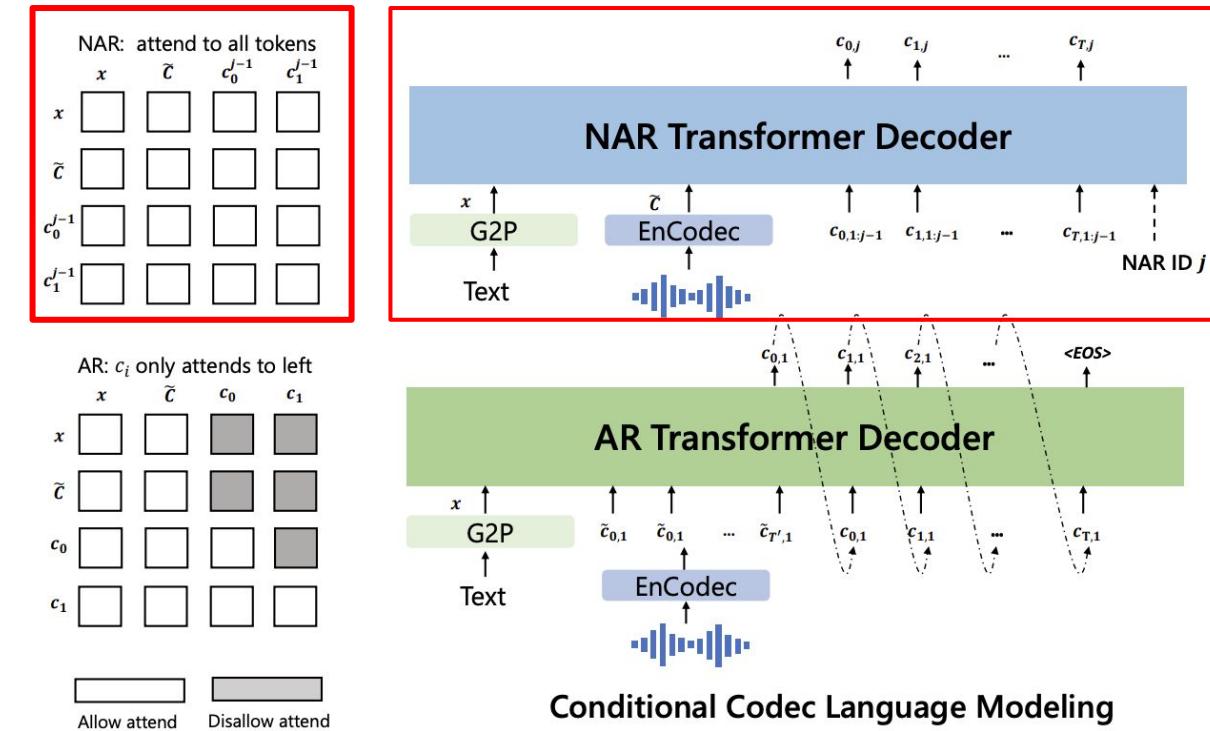
Задача моделирования делится на две (decoder-only LM):

2. **NAR:** неавторегрессионное предсказание всех оставшихся токенов
  - 8 отдельных acoustic embedding layers
  - итеративное предсказание **7 раз** для кодов 2-8
  - вход  $i$ -ой итерации – **текст**, полное дискретное представление промпта и сумма эмбеддингов со всех предыдущих кодбуков для предсказываемого участка

$$(\mathbf{e}_x, \mathbf{e}_{\tilde{c}}, \mathbf{e}_{c_{:, <i}}) \quad e_{c_{t,j}} = W_a^j \odot c_{t,j}$$

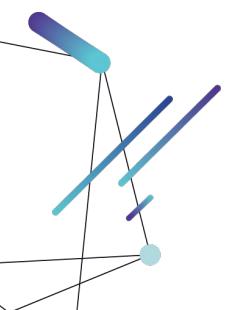
$$\mathbf{e}_{c_t} = \sum_{j=1}^{i-1} e_{c_{t,j}}$$

- нет каузальной маски
- positional embedding отдельно для текста и кодов
- итоговое представление – сумма представлений всех уровней кодов  $\mathbf{e}_{\tilde{c}_t} = \sum_{j=1}^8 e_{\tilde{c}_{t,j}}$



# VALL-E (2023)

- Архитектура: 12 Transformer blocks, 16 attention heads
- Размеры: ~120-130M каждая из моделей
- Датасет: LibriLight 60k часов английской речи, ~7000 спикеров
  - используются аудио длины 10-20 секунд
  - промпт для NAR модели 3 секунды
  - 16kHz
- Ресурсы: 16 NVIDIA TESLA V100 32GB GPUs
- RTE: ~1.3



# VALL-E (2023)

- Архитектура: 12 Transformer blocks, 16 attention heads
- Размеры: ~120-130M каждая из моделей
- Датасет: LibriLight 60k часов английской речи, ~7000 спикеров
  - используются аудио длины 10-20 секунд
  - промпт для NAR модели 3 секунды
  - 16kHz
- Ресурсы: 16 NVIDIA TESLA V100 32GB GPUs
- RTE: ~1.3

- Zero-shot TTS: промпт unseen голоса, 3 секунды
- 3 запуска, усредненные метрики

## □ Метрики качества

### **объективные метрики:**

- speaker similarity (WavLM-TDNN)
- robustness WER (e HuBERT-Large fine-tuned for ASR)

### **субъективные метрики:**

- naturalness - comparative mean opinion score (CMOS). От -3 до 3.
- similarity - similarity mean opinion score(SMOS)

# VALL-E (2023)

## *Результаты*

### Промпт из LibriSpeech

По **объективным** метрикам:

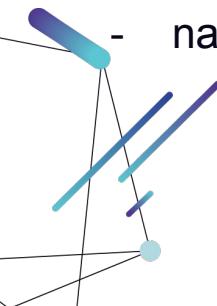
- лучше модели Your TTS по обеим метрикам
- лучше других похожих моделей. WER лучше из-за использования фонем, а не ssl юнитов
- у модели GSLM нет контроля над голосом

По **субъективным** метрикам:

- намного лучший показатель по похожести голоса, чем у

YourTTS

- naturalness лучше YourTTS, есть отрыв с GT



model	WER	SPK
GroundTruth	2.2	0.754
<b>Speech-to-Speech Systems</b>		
GSLM	12.4	0.126
AudioLM*	6.0	-
<b>TTS Systems</b>		
YourTTS	7.7	0.337
VALL-E	5.9	<b>0.580</b>
VALL-E-continual	<b>3.8</b>	0.508

	SMOS	CMOS (v.s. VALL-E)
YourTTS	$3.45 \pm 0.09$	-0.12
VALL-E	$4.38 \pm 0.10$	0.00
GroundTruth	$4.5 \pm 0.10$	+0.17

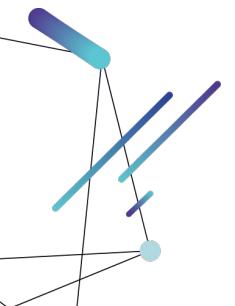
# VALL-E (2023)

## *Результаты*

### Промпт из VCTK

- Метрики хуже, чем в предыдущем эксперименте
- Однако показатели VALL-E лучше бейзлайна
- Более длинные промпты дают лучшие результаты

### Demo



speaker similarity

	3s prompt	5s prompt	10s prompt
<b>108 full speakers</b>			
YourTTS*	0.357	0.377	0.394
VALL-E	0.382	0.423	<b>0.484</b>
GroundTruth	0.546	0.591	0.620
<b>11 unseen speakers</b>			
YourTTS	0.331	0.337	0.344
VALL-E	0.389	0.380	<b>0.414</b>
GroundTruth	0.528	0.556	0.586

	SMOS	CMOS (v.s. VALL-E)
YourTTS*	$3.70 \pm 0.09$	-0.23
VALL-E	$3.81 \pm 0.09$	0.00
GroundTruth	$4.29 \pm 0.09$	-0.04

[Статья](#), [github](#)

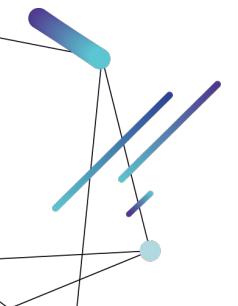
# VALL-E (2023)



- качество
- хорошее копирование голоса
- хорошее копирование окружения и эмоций
- разнообразие (sampling-based метод)

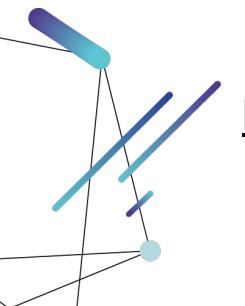


- надежность
- неполное покрытие характеристик голосов
- скорость генерации



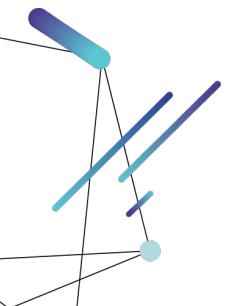
# VALL-E (2023). Улучшения

- [VALL-E X](#) (2023) – расширение на несколько языков, дополнительный language id, копирование с записей на другом языке
  - [VioLA](#) (2023) – расширение на другие задачи speech-to-text, text-to-text, and speech-to-speech
  - Улучшение стабильности:
    - [ELLA-V](#) (2024)
    - [VALL-E R](#) (2024) – ограничения для monotonic alignment
    - [RALL-E](#) (2024) – дополнительное моделирование длительностей и высот
    - [VALL-E T](#) (2024)
  - [VALL-E 2](#) (2024)
    - repetition aware sampling – борьба с повторяющимися токенами
    - grouped code modeling – ускорение, предсказание нескольких токенов за раз
  - [MELLE](#) (2025) – предсказание мел-спектрограмм
- [Визуализация](#) всех версий от Microsoft



Появились параллельно: Spear-TTS, Bark

# Flow matching-based модели



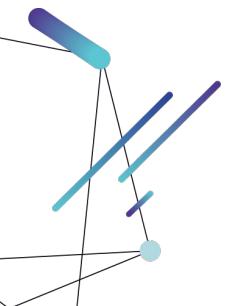
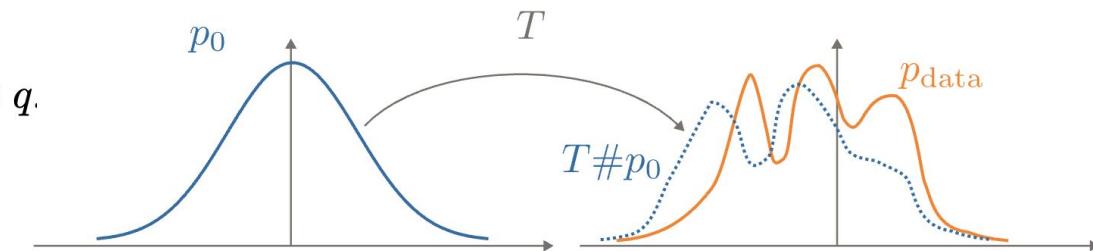
# Flow matching

- Цель – преобразовать некоторое простое априорное распределение  $p_0$  в более сложное распределение данных  $p_1 \approx q$ .
- Идея основана на терминах, описывающих continuous normalizing flows (CNF):
  - **векторное поле**  $v_t : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ , которое моделирует
  - **flow** (поток) – отображение, которое сдвигает точки из prior распределения в target  $\phi_t : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$
  - связь между ними определяется ОДУ

$$\frac{d}{dt} \phi_t(x) = v_t(\phi_t(x)); \quad \phi_0(x) = x. \quad (1)$$

- **вероятностный путь** (probability path – time-dependent probability density function)  $p : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}_{>0}$  определяется уравнением

$$p_t(x) = p_0(\phi_t^{-1}(x)) \det \left[ \frac{\partial \phi_t^{-1}}{\partial x}(x) \right] \quad (2)$$



# Flow matching

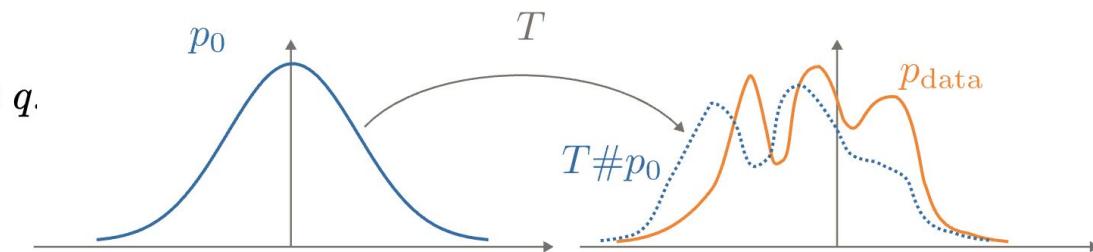
- Цель – преобразовать некоторое простое априорное распределение  $p_0$  в более сложное распределение данных  $p_1 \approx q$ .
- Идея основана на терминах, описывающих continuous normalizing flows (CNF):

- **векторное поле**  $v_t : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ , которое моделирует
- **flow** (поток) – отображение, которое сдвигает точки из prior распределения в target  $\phi_t : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$
- связь между ними определяется ОДУ

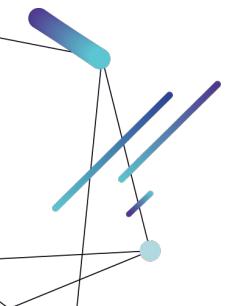
$$\frac{d}{dt} \phi_t(x) = v_t(\phi_t(x)); \quad \phi_0(x) = x. \quad (1)$$

- **вероятностный путь** (probability path – time-dependent probability density function)  $p : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}_{>0}$  определяется уравнением

$$p_t(x) = p_0(\phi_t^{-1}(x)) \det \left[ \frac{\partial \phi_t^{-1}}{\partial x}(x) \right] \quad (2)$$



- Для семплирования из  $p_t(x)$ 
  - семплируем  $x_0$  из  $p_0$
  - находим решение для  $\phi_t(x_0)$  из (1)
  - подставляем в (2)



# Flow matching

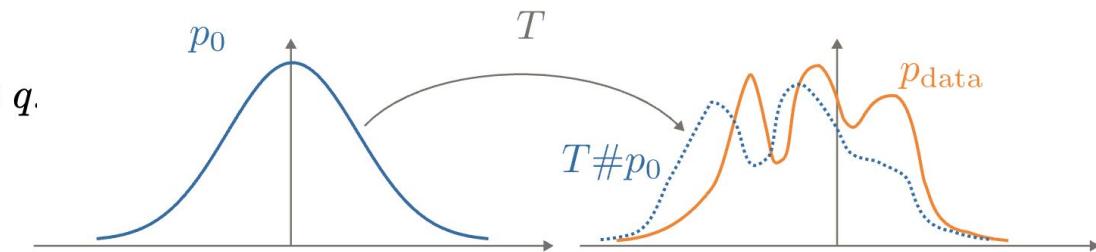
- Цель – преобразовать некоторое простое априорное распределение  $p_0$  в более сложное распределение данных  $p_1 \approx q$ .
- Идея основана на терминах, описывающих continuous normalizing flows (CNF):

- **векторное поле**  $v_t : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$ , которое моделирует
- **flow** (поток) – отображение, которое сдвигает точки из prior распределения в target  $\phi_t : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}^d$
- связь между ними определяется ОДУ

$$\frac{d}{dt} \phi_t(x) = v_t(\phi_t(x)); \quad \phi_0(x) = x. \quad (1)$$

- **вероятностный путь** (probability path – time-dependent probability density function)  $p : [0, 1] \times \mathbb{R}^d \rightarrow \mathbb{R}_{>0}$  определяется уравнением

$$p_t(x) = p_0(\phi_t^{-1}(x)) \det \left[ \frac{\partial \phi_t^{-1}}{\partial x}(x) \right] \quad (2)$$

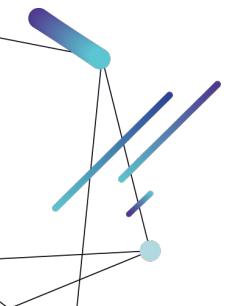


- Для семплирования из  $p_t(x)$ 
  - семплируем  $x_0$  из  $p_0$
  - находим решение для  $\phi_t(x_0)$  из (1)
  - подставляем в (2)
- $v_t$  параметризуется **нейронной сетью**
  - FM:  $\mathcal{L}_{FM}(\theta) = \mathbb{E}_{t,p_t(x)} \|u_t(x) - v_t(x; \theta)\|^2$
  - CFM:  $\mathcal{L}_{CFM}(\theta) = \mathbb{E}_{t,q(x_1),p_t(x|x_1)} \|u_t(x | x_1) - v_t(x; \theta)\|^2$

Optimal transport path:

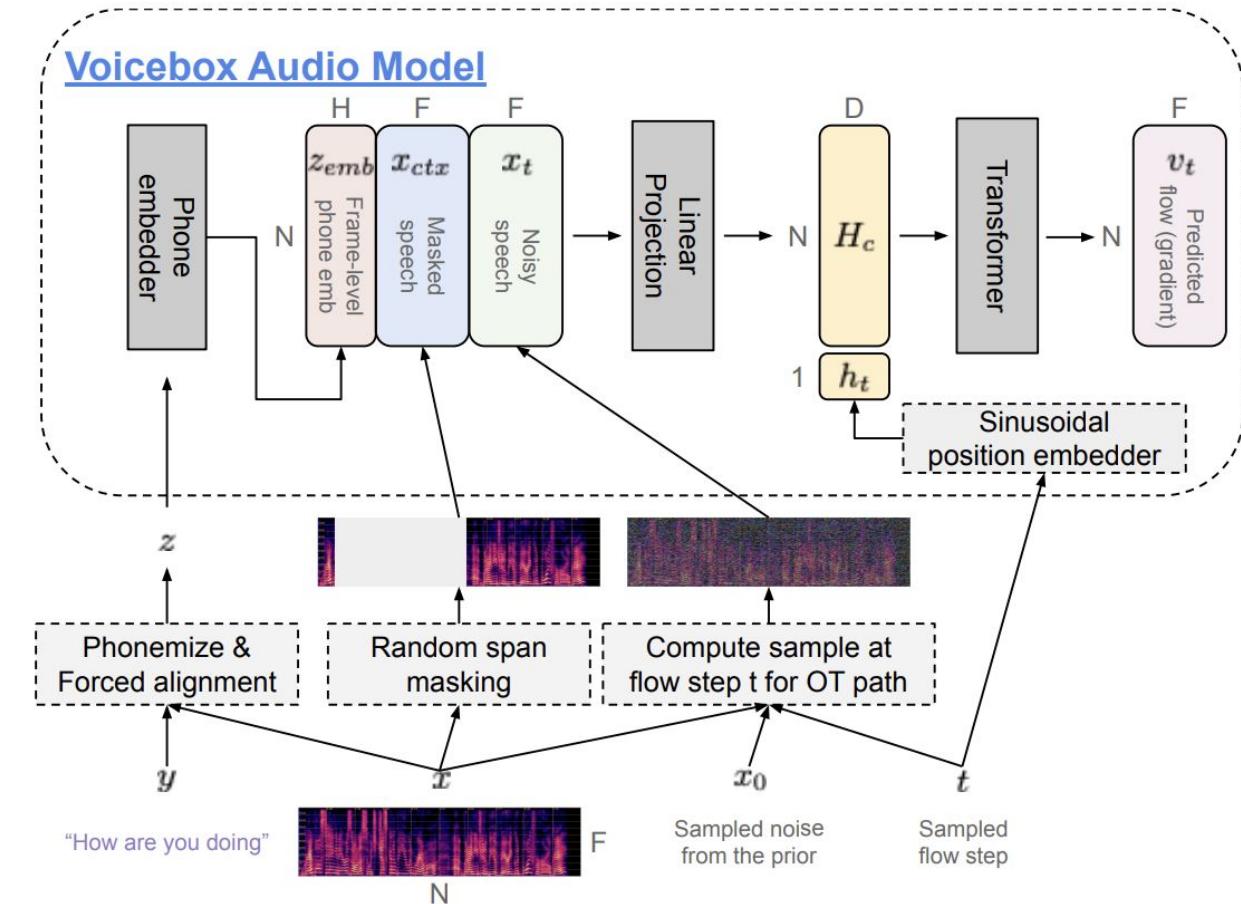
$$p_t(x | x_1) = \mathcal{N}(x | tx_1, (1 - (1 - \sigma_{min})t)^2 I)$$

$$u_t(x | x_1) = (x_1 - (1 - \sigma_{min})x) / (1 - (1 - \sigma_{min})t)$$



# VoiceBox (2023)

- Flow-matching based модель
- Неавторегрессионная
- Задача: text-guided speech infilling (in-context learning)
- Multi-task: zero-shot TTS, speech editing, denoising
- Признаки: мел-спектrogramмы

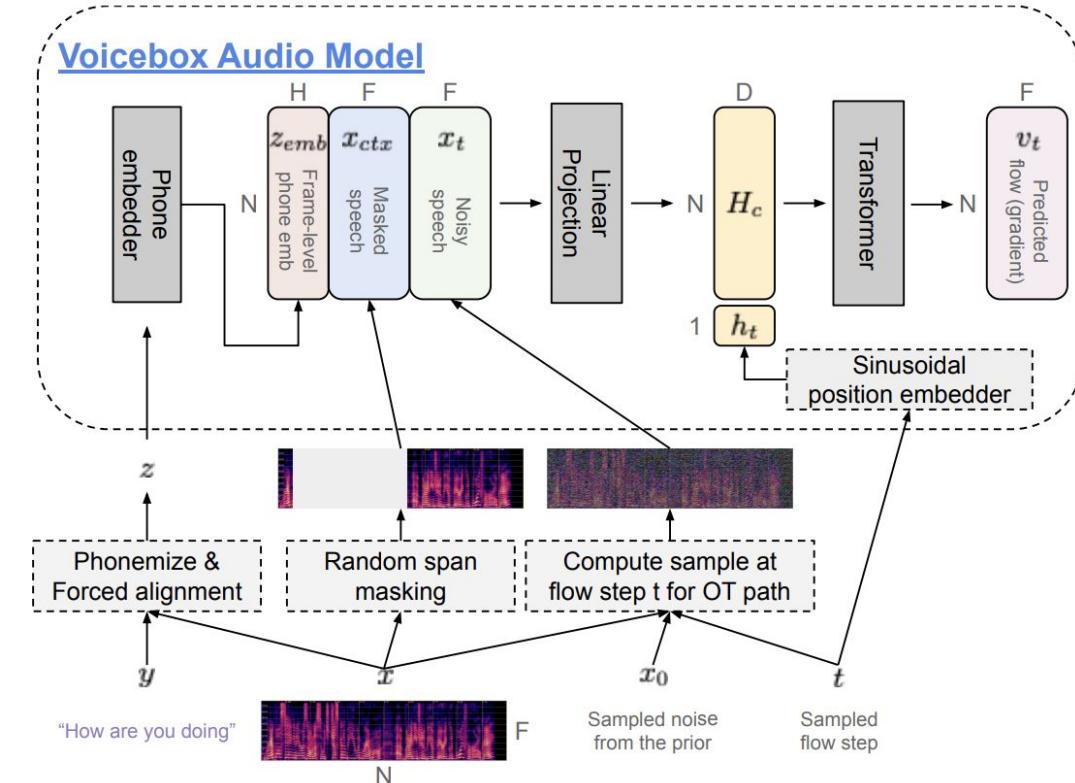


[Статья](#), [github](#) (unofficial)

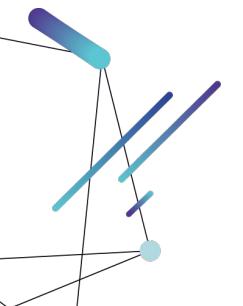
# VoiceBox (2023)

Датасет ( $x, y$ ),

- $y = (y^1, y^2, \dots, y^M)$  – последовательность фонем длины  $M$
- $l = (l^1, l^2, \dots, l^M)$  – длительности фонем во фреймах
- $x = (x^1, x^2, \dots, x^N)$  – мел-спектrogramma длины  $N$   
представляется в виде двух частей  
 $x_{mis} = m \odot x$  – замаскированный участок  
 $x_{ctx} = (1 - m) \odot x$  – контекст



[Статья](#), [github](#) (unofficial)



# VoiceBox (2023)

Датасет ( $x, y$ ),

- $y = (y^1, y^2, \dots, y^M)$  – последовательность фонем длины  $M$
- $l = (l^1, l^2, \dots, l^M)$  – длительности фонем во фреймах
- $x = (x^1, x^2, \dots, x^N)$  – мел-спектrogramma длины  $N$   
представляется в виде двух частей  
 $x_{mis} = m \odot x$  – замаскированный участок  
 $x_{ctx} = (1 - m) \odot x$  – контекст

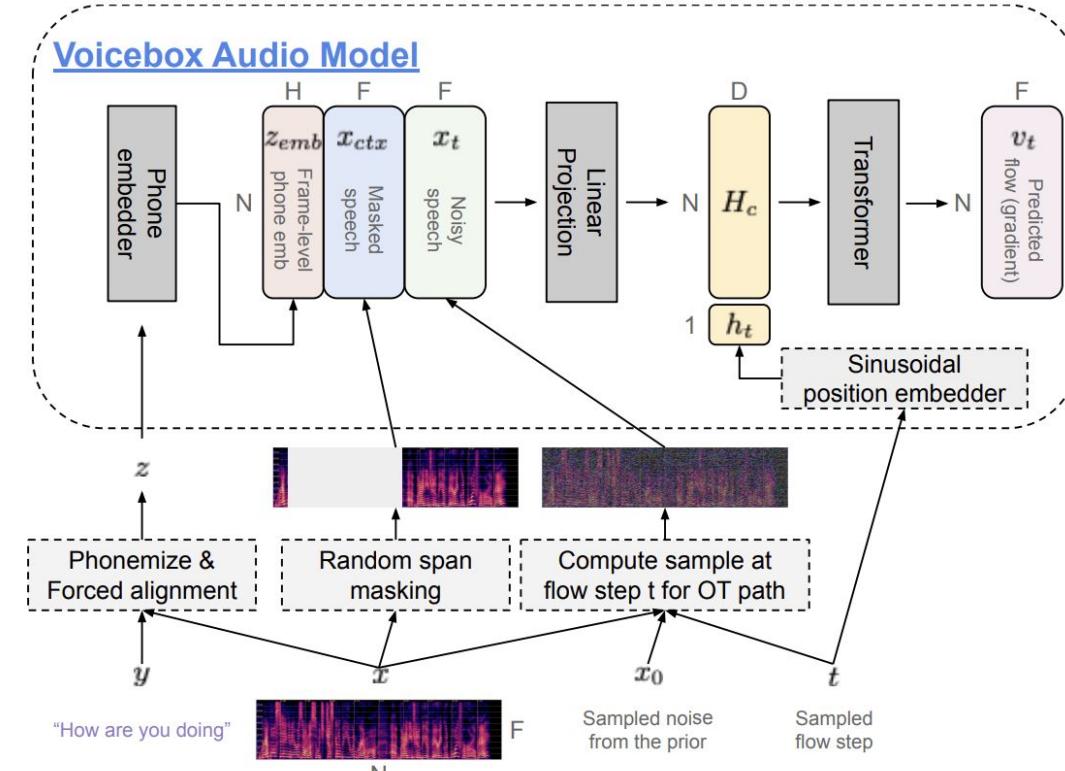
Задача:  $p(x_{mis} | y, x_{ctx})$  разбивается на две

- 1) предсказание длительностей  $l$  и преобразование  $y$  в  $z$

$$z = \text{rep}(y, l) = (z^1, z^2, \dots, z^N)$$

- 1) предсказание замаскированной части:

- признаки ( $x_t, x_{ctx}$ , and  $z_{emb}$ ) конкатенируют по оси признаков
- применяют проекционный слой, получают признаки  $H_c \in \mathbb{R}^{N \times D}$ , где  $N$  – длина спектrogramмы,  $D$  – размерность Transformer
- обучаю модель  $v_t(x_t, x_{mis}, z; \theta) \in \mathbb{R}^{N \times F}$ , где  $F$  – размерность мел-спектrogramмы по частоте



Статья, [github](#) (unofficial)

# VoiceBox (2023)

## Acoustic model

### Loss:

$$\mathcal{L}_{\text{audio-CFM}}(\theta) = \mathbb{E}_{t, m, q(x, z), p_0(x_0)} \|u_t(x_t | x) - v_t(x_t, x_{ctx}, z; \theta)\|^2$$

На обучении,  $x$  – мел-спектрограмма текущего семпла  
 $x_0$  – семпл из prior распределения (шум)

Тогда можем вычислить

$$x_t = (1 - (1 - \sigma_{\min})t)x_0 + tx$$

$$u_t(x_t | x) = x - (1 - \sigma_{\min})x_0$$

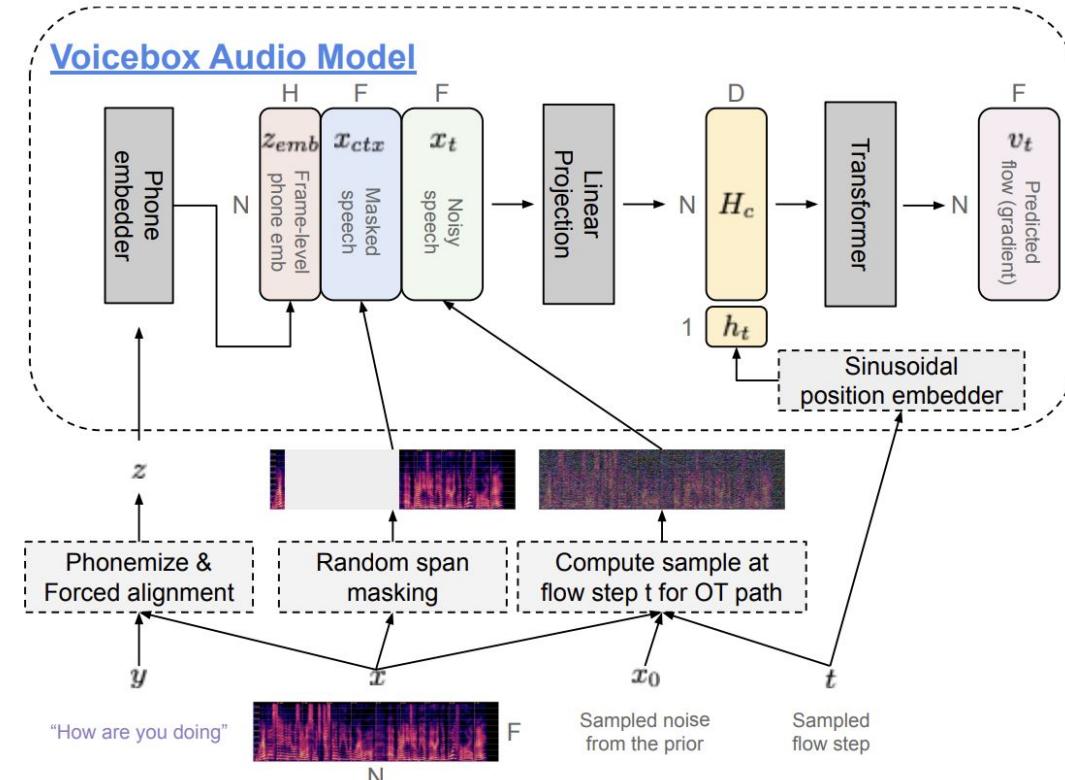
Лосс с учетом только замаскированных фреймов:

$$\mathcal{L}_{\text{audio-CFM-m}}(\theta) = \mathbb{E}_{t, m, q(x, z), p_0(x_0)} \|m \odot (u_t(x_t | x) - v_t(x_t, x_{ctx}, z; \theta))\|^2$$

## Duration predictor

2 варианта:

- 1) также основанный на flow matching
- 2) как в FastSpeech2



[Статья](#), [github](#) (unofficial)

# VoiceBox (2023)

## Inference

- семплируем  $\phi_0(x_0) = x_0$
- итеративно решаем с помощью ODE solver уравнение

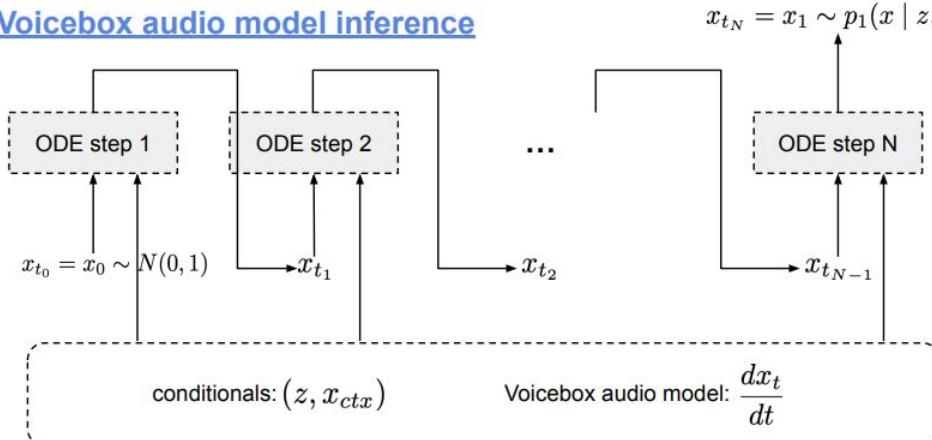
$$\frac{d\phi_t(x_0)}{dt} = v_t(\phi_t(x_0), x_{ctx}, z; \theta)$$

чтобы приблизить  $\phi_1(x_0)$

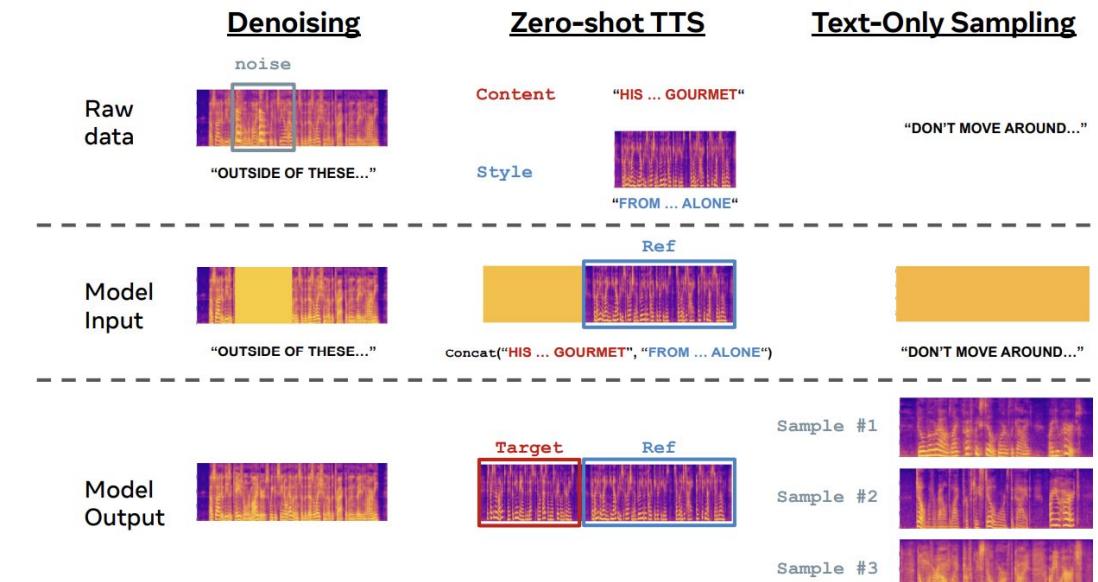
- Техника **classifier-free guidance** (CFG): одновременное обучение в conditional и unconditional режиме (без  $z$  и  $x_{ctx}$ )

$$\tilde{v}_t(w, x_{mis}, z; \theta) = (1 + \alpha) \cdot v_t(w, x_{ctx}, z; \theta) - \alpha \cdot v_t(w; \theta)$$

## Voicebox audio model inference



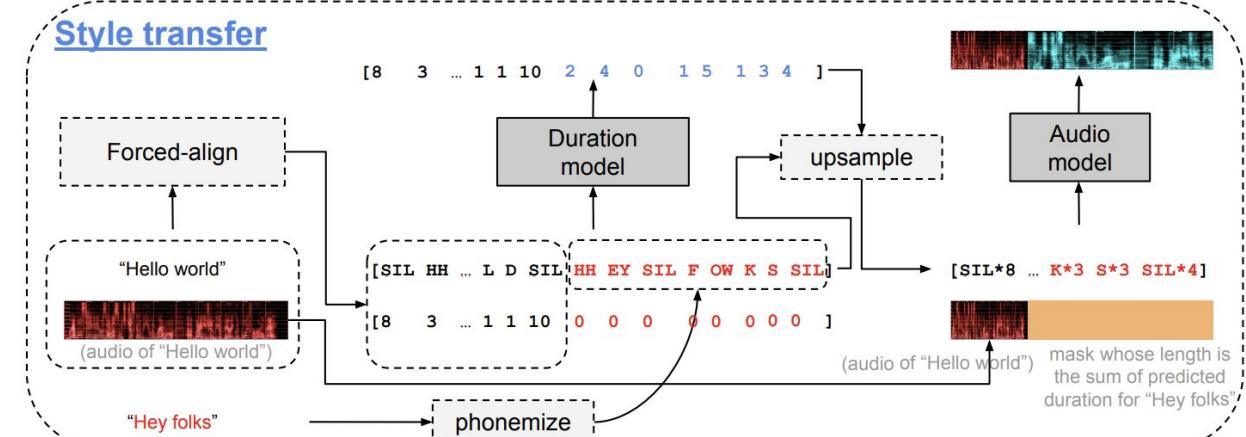
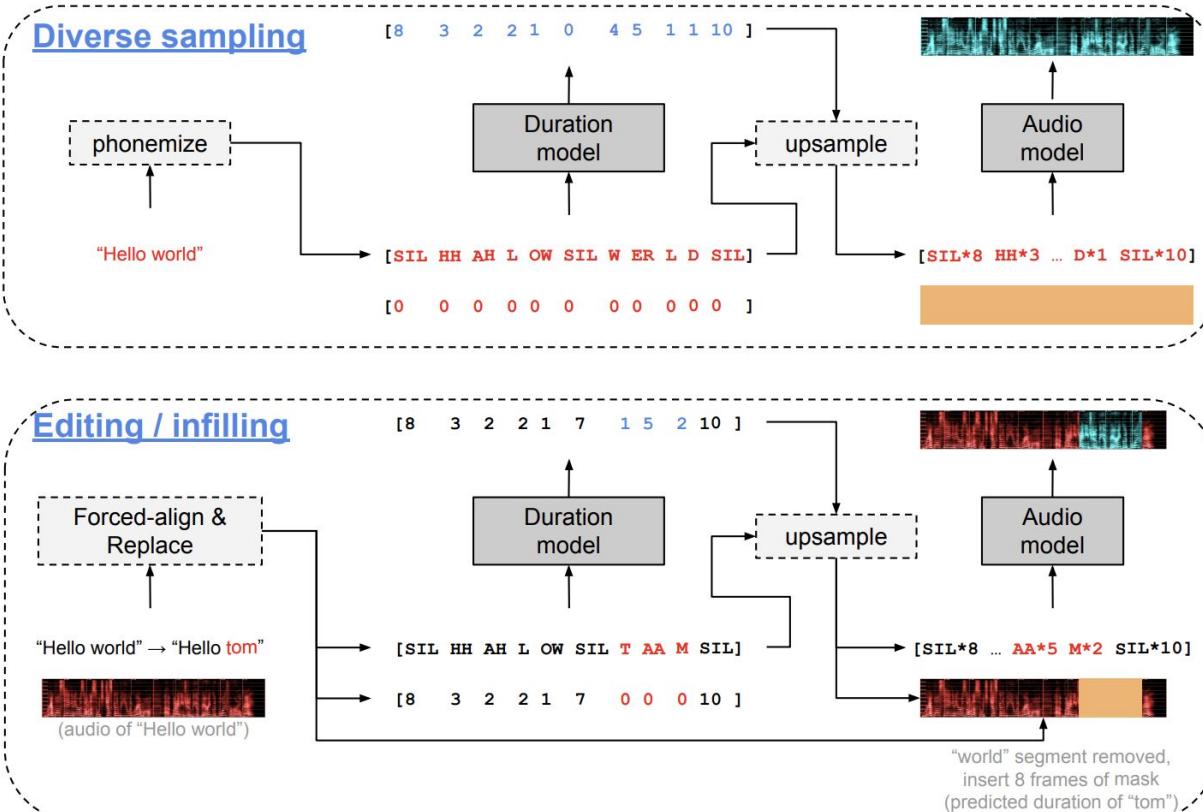
## Сценарии использования



[Статья](#), [github](#) (unofficial)

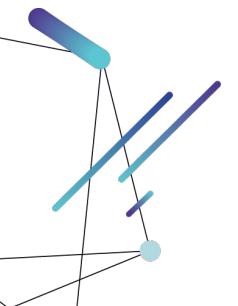
# VoiceBox (2023)

## Сценарии использования



# VoiceBox (2023)

- Архитектура: 24 layers, 16 attention heads, U-net style skip connections
- Размер: 330M + 28/34M duration predictor
- Датасет:
  - 60K часов английских аудиокниг
  - 50K часов мультиязычных данных на 6 языков (En, Fr, De, Es, Pt, Pt)
- Montreal Forced Aligner (MFA)
- Ресурсы: 32 GPU
- Вокодер HiFi-GAN, обучен на их английском датасете
- 32 итерации на инференсе



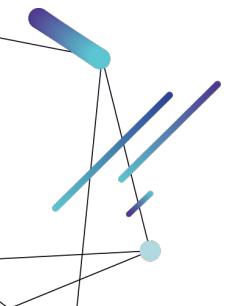
# VoiceBox (2023)

## Zero-shot TTS

Первый эксперимент:

- Английский язык
- Промпт – 3 секунды, LibriSpeech
- SIM-o – с GT записью
- SIM-r – с ресинтезированной вокодером GT записью

Model	WER	SIM-o	SIM-r	QMOS	SMOS
Ground truth	2.2	0.754	n/a	$3.98 \pm 0.14$	$4.01 \pm 0.09$
<i>cross-sentence</i>					
A3T	63.3	0.046	0.146	-	-
YourTTS	7.7	0.337	n/a	$3.27 \pm 0.13$	$3.19 \pm 0.14$
VALL-E	5.9	-	0.580	-	-
VB-En	1.9	0.662	0.681	$3.78 \pm 0.10$	$3.71 \pm 0.11$
<i>continuation</i>					
A3T	18.7	0.058	0.144	-	-
VALL-E	3.8	0.452*	0.508	-	-
VB-En ( $\alpha = 0.7$ )	2.0	0.593	0.616	-	-



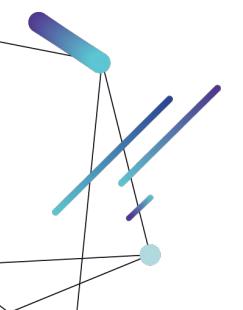
# VoiceBox (2023)

## Zero-shot TTS

Второй эксперимент:

- На всех языках
- baseline: YourTTS
- Промпт на том же языке – лучшие метрики

Ref	De		En		Es		Fr		Pl		Pt		
	WER	SIM-o	WER	SIM-o	WER	SIM-o	WER	SIM-o	WER	SIM-o	WER	SIM-o	
GT	-	5.9	0.725	5.0	0.636	4.1	0.729	5.2	0.714	4.9	0.743	5.8	0.725
YT	De	n/a	n/a	7.3	0.373	n/a	n/a	11.3	0.361	n/a	n/a	13.7	0.263
	En	n/a	n/a	7.0	0.403	n/a	n/a	11.4	0.298	n/a	n/a	14.1	0.234
	Es	n/a	n/a	7.6	0.327	n/a	n/a	11.6	0.316	n/a	n/a	13.5	0.256
	Fr	n/a	n/a	7.6	0.363	n/a	n/a	10.7	0.459	n/a	n/a	13.1	0.299
	Pl	n/a	n/a	7.8	0.349	n/a	n/a	11.8	0.370	n/a	n/a	15.1	0.308
	Pt	n/a	n/a	7.6	0.322	n/a	n/a	11.8	0.297	n/a	n/a	13.6	0.436
	AVG	n/a	n/a	7.5	0.356	n/a	n/a	11.4	0.350	n/a	n/a	13.9	0.299
VB-Multi ( $\alpha = 1.0$ )	De	4.8	0.632	4.8	0.522	3.6	0.442	5.3	0.489	5.5	0.449	5.4	0.420
	En	5.9	0.435	4.2	0.535	4.1	0.423	6.8	0.423	8.3	0.402	7.6	0.385
	Es	4.9	0.460	4.3	0.479	3.6	0.613	5.3	0.473	5.2	0.436	5.4	0.435
	Fr	4.9	0.476	4.3	0.485	3.7	0.479	5.1	0.602	4.8	0.408	5.4	0.418
	Pl	4.7	0.491	3.8	0.503	3.5	0.528	5.1	0.503	4.0	0.641	4.9	0.476
	Pt	4.9	0.422	4.6	0.426	3.7	0.476	5.5	0.453	4.8	0.406	5.2	0.620
	AVG	5.0	0.486	4.4	0.492	3.7	0.494	5.5	0.491	5.5	0.457	5.7	0.459



# VoiceBox (2023)

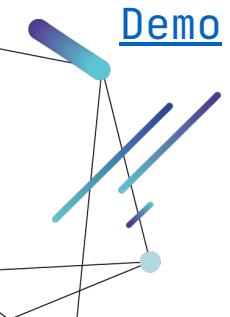
## Zero-shot TTS

Второй эксперимент:

- На всех языках
- Генерация на английском с промптами на разных языках

	Ref=De	Ref=En	Ref=Es	Ref=Fr	Ref=Pl	Ref=Pt
<b>SMOS (target text = En)</b>						
YT	$3.26 \pm 0.11$	$3.24 \pm 0.11$	$3.22 \pm 0.12$	$3.48 \pm 0.10$	$3.26 \pm 0.09$	$3.38 \pm 0.11$
VB-Multi ( $\alpha = 1.0$ )	$3.89 \pm 0.10$	$3.93 \pm 0.08$	$3.84 \pm 0.10$	$3.92 \pm 0.09$	$3.81 \pm 0.08$	$3.96 \pm 0.09$
<b>QMOS (target text = En)</b>						
YT	$3.29 \pm 0.12$	$3.17 \pm 0.13$	$3.29 \pm 0.12$	$3.08 \pm 0.12$	$3.35 \pm 0.12$	$3.21 \pm 0.12$
VB-Multi ( $\alpha = 1.0$ )	$3.67 \pm 0.09$	$3.48 \pm 0.09$	$3.45 \pm 0.11$	$3.31 \pm 0.12$	$3.75 \pm 0.11$	$3.35 \pm 0.13$

Demo



# VoiceBox (2023)

Другие эксперименты

- **Denoising**
  - искусственное зашумление
  - требует текст и указания позиции шумной части

Table 5: Transient noise removal where noise overlaps with 50% of the speech at a -10dB SNR.

Model	WER	SIM-o	QMOS
Clean speech	2.2	0.687	$4.07 \pm 0.15$
Noisy speech	41.2	0.287	$2.50 \pm 0.15$
Demucs	32.5	0.368	$2.86 \pm 0.17$
A3T	11.5	0.148	$3.10 \pm 0.15$
VB-En ( $\alpha = 0.7$ )	2.0	0.612	$3.87 \pm 0.17$

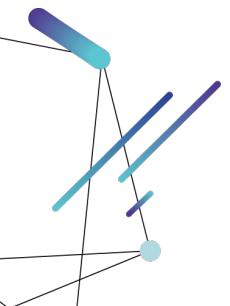
# VoiceBox (2023)

Другие эксперименты

- Denoising
- Diversity sampling
  - Fréchet Speech Distance (FSD)
  - сравнивают DP

**Table 6: Diverse speech generation from LS test-other text.**

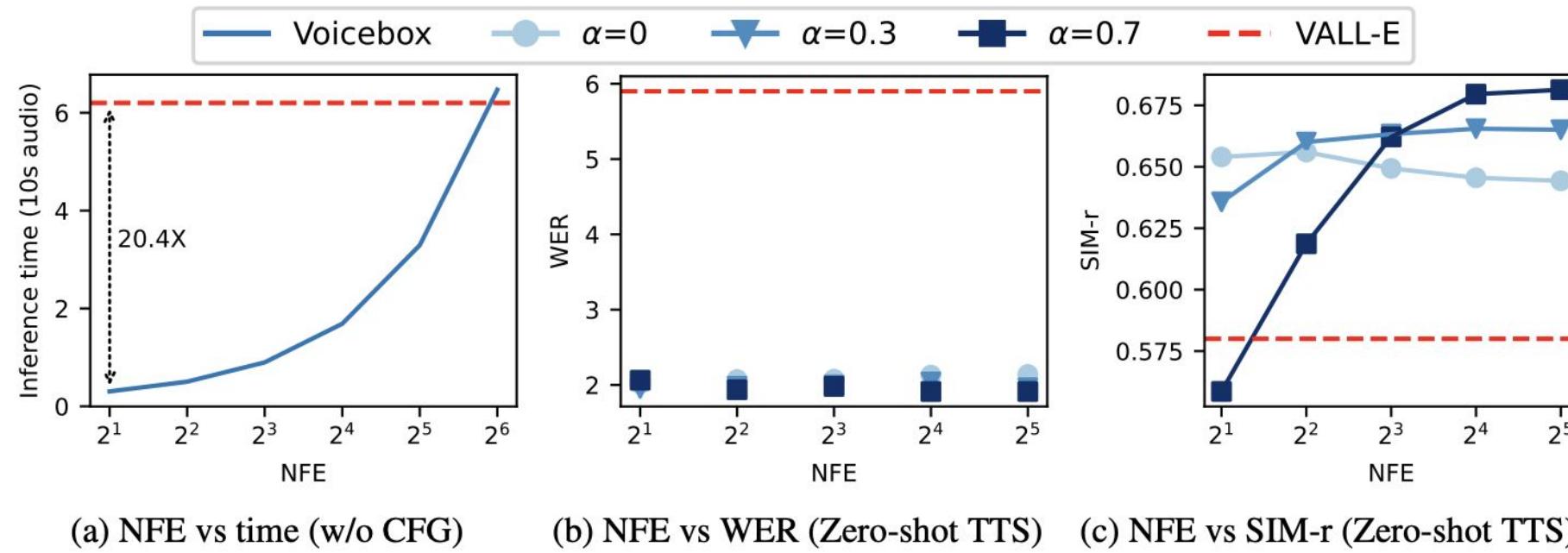
Model	WER	FSD
Ground truth	4.3	171.1
<i>require additional input</i>		
VITS-VCTK	10.6	306.6
YourTTS (ref=LS train)	9.0	277.9
<i>text-only</i>		
A3T	37.9	373.0
VITS-LJ	5.6	344.2
VB-En ( $\alpha = 0$ , dur=regr)	3.1	155.7
VB-En ( $\alpha = 0$ , dur=FM, $\alpha_{dur} = 0$ )	5.6	159.8



# VoiceBox (2023)

Другие эксперименты

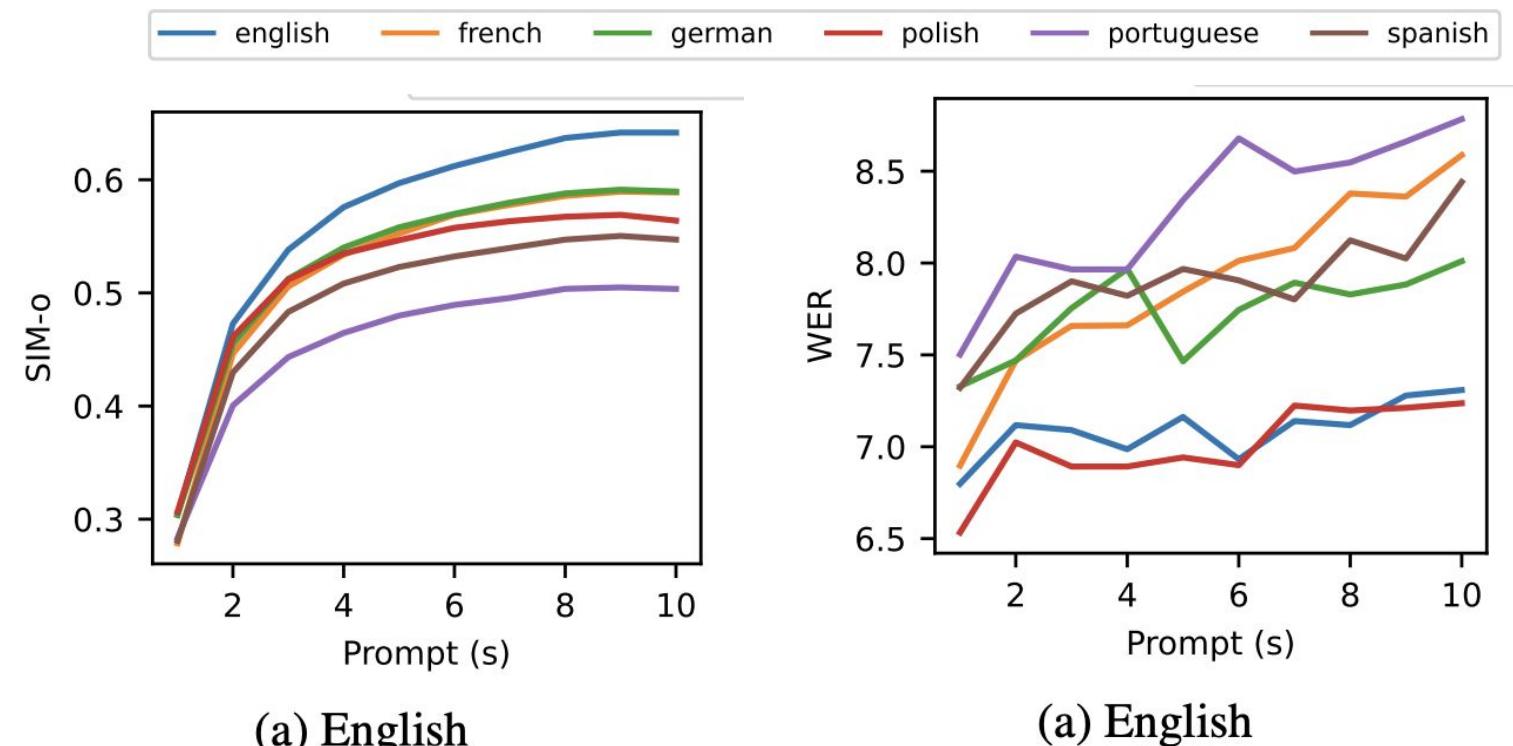
- Denoising
- Diversity sampling
- Эффективность



# VoiceBox (2023)

## Другие эксперименты

- Denoising
- Diversity sampling
- Эффективность
- Зависимость от длины промпта



(a) English

(a) English

Demo

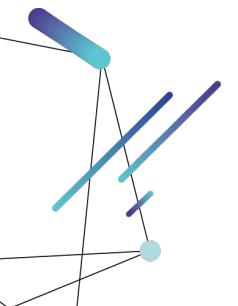
# VoiceBox (2023)



- хорошее копирование голоса, стиля
- разнообразие
- несколько задач
- более быстрая, чем VALL-E



- нужен alignment для обучающего датасета



# E2-TTS (2024)

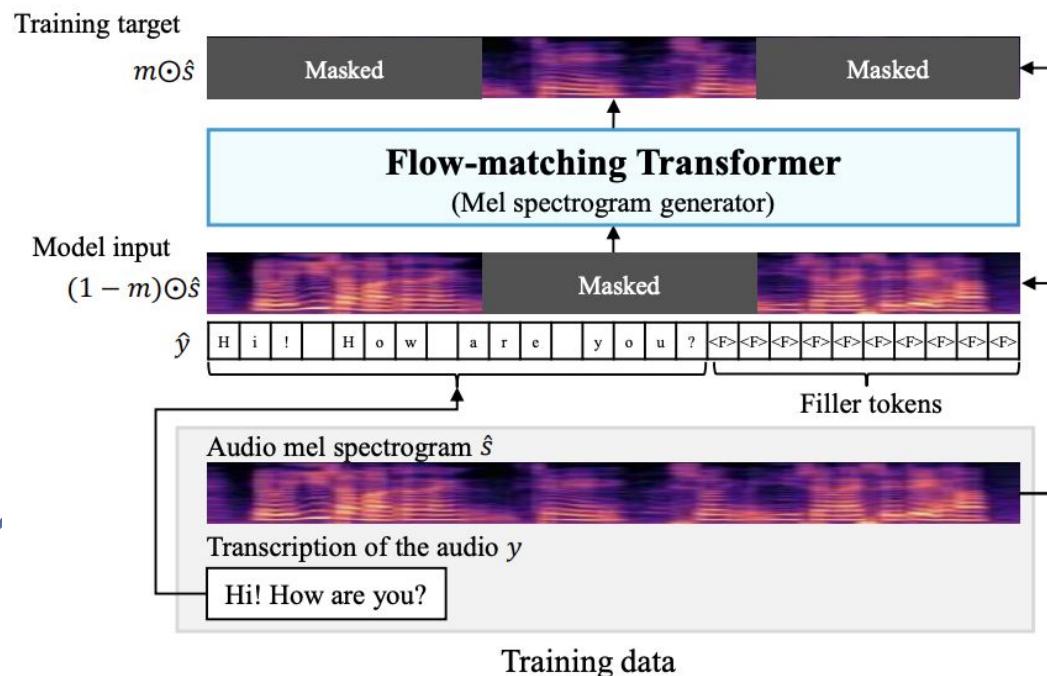
Авторегрессионное выравнивание – нестабильное  
Выравнивание с помощью DP ограничивает разнообразие



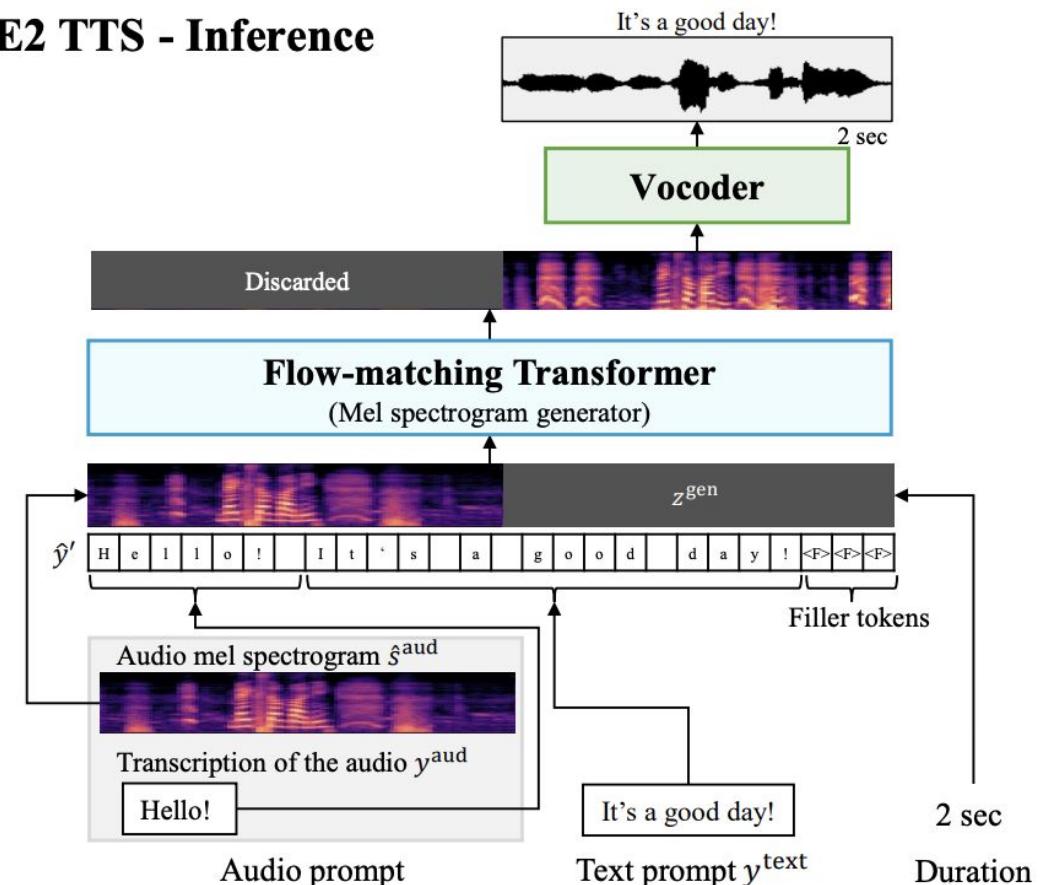
- 1) Задаем итоговую длительность аудио (эвристика)
- 2) Добавляем к фонемам pad токены до этой длины
- 3) FM модель восстанавливает аудио (спектрограмму)

[Статья, demo](#)

## E2 TTS - Training



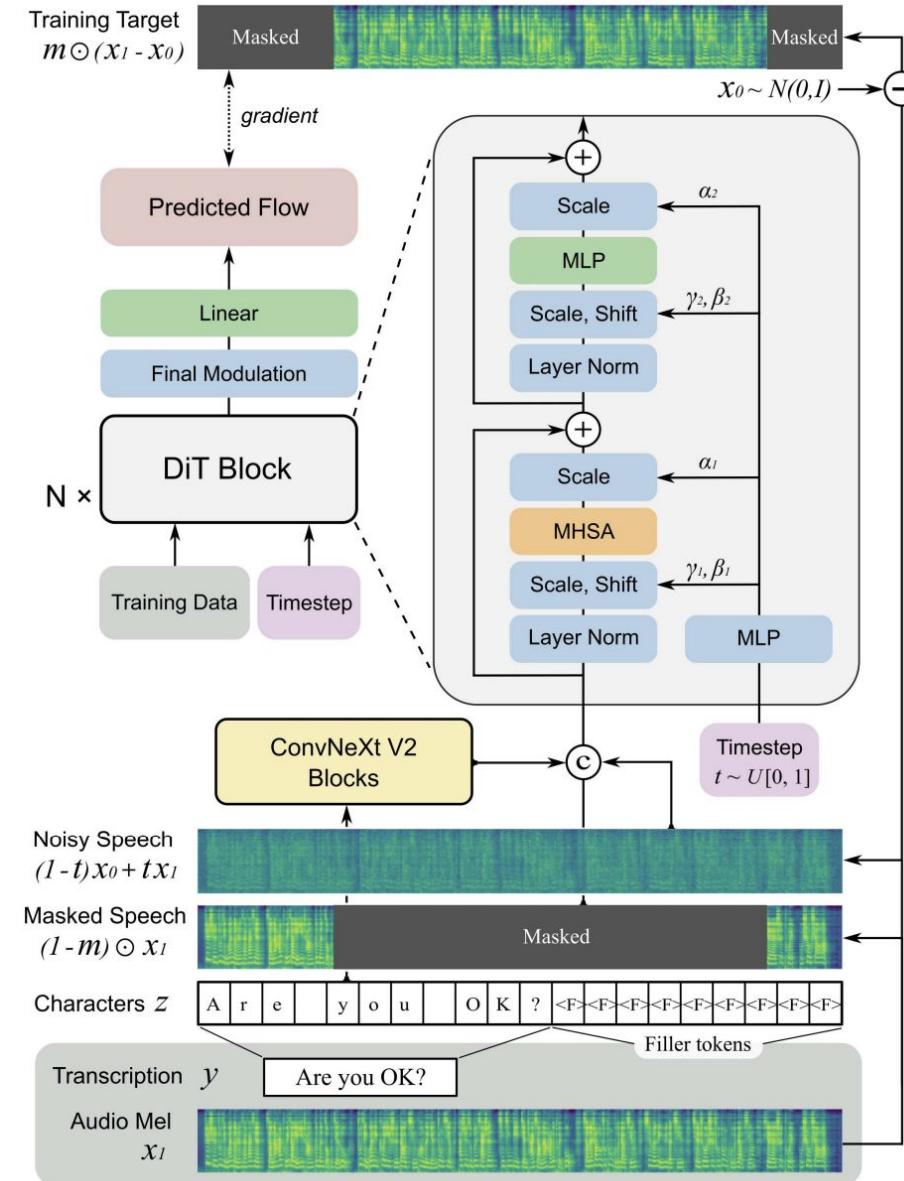
## E2 TTS - Inference



# F5-TTS (2025)

- ❑ a Fairytaler that Fakes Fluent and Faithful speech with Flow matching
- ❑ без выравнивания по фонемам, duration predictor, text encoder
- ❑ infilling task - предсказание кусочка аудио по аудио контексту и тексту
- ❑ Обучение пары  $(x_1, y)$ , где  $x_1$ - мел спектrogramma, у - текст
- ❑ Вход модели:
  - текст – последовательность букв  $c_i$ / символов pinyin для китайского – с паддинг токенами до количества аудио фреймов N
$$z = (c_1, c_2, \dots, c_M, \underbrace{\langle F \rangle, \dots, \langle F \rangle}_{N-M})$$
  - замаскированная мел-спектrogramma  $(1 - m) \odot x_1, m \in [0, 1]^{F \times N}$
  - мел-спектrogramma , зашумленная до шага t  $x_t = (1 - t)x_0 + tx_1$
  - конкатенация **по размерности признаков** (так отдельное пространство признаков)

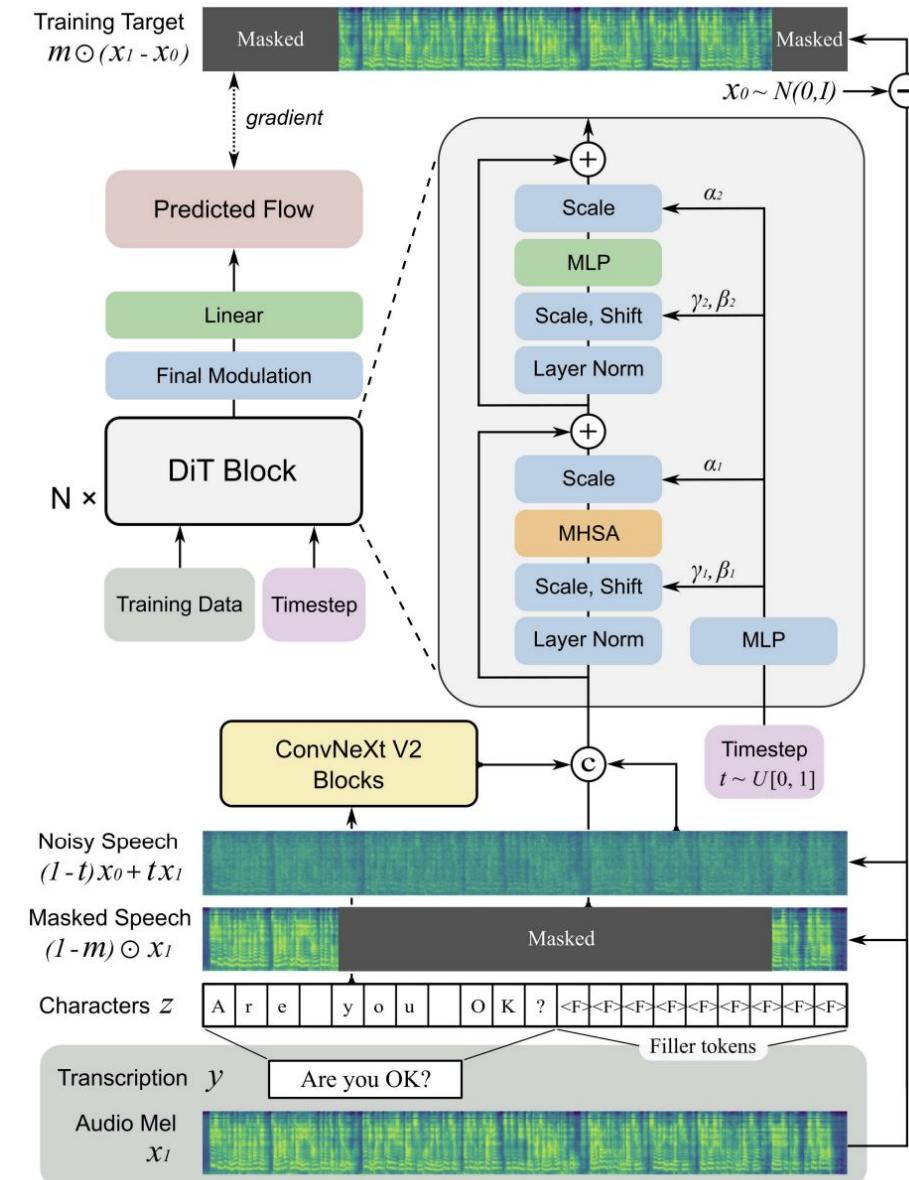
Выход –  $m \odot x_1$



[Статья](#), [github](#)

# F5TTS (2025)

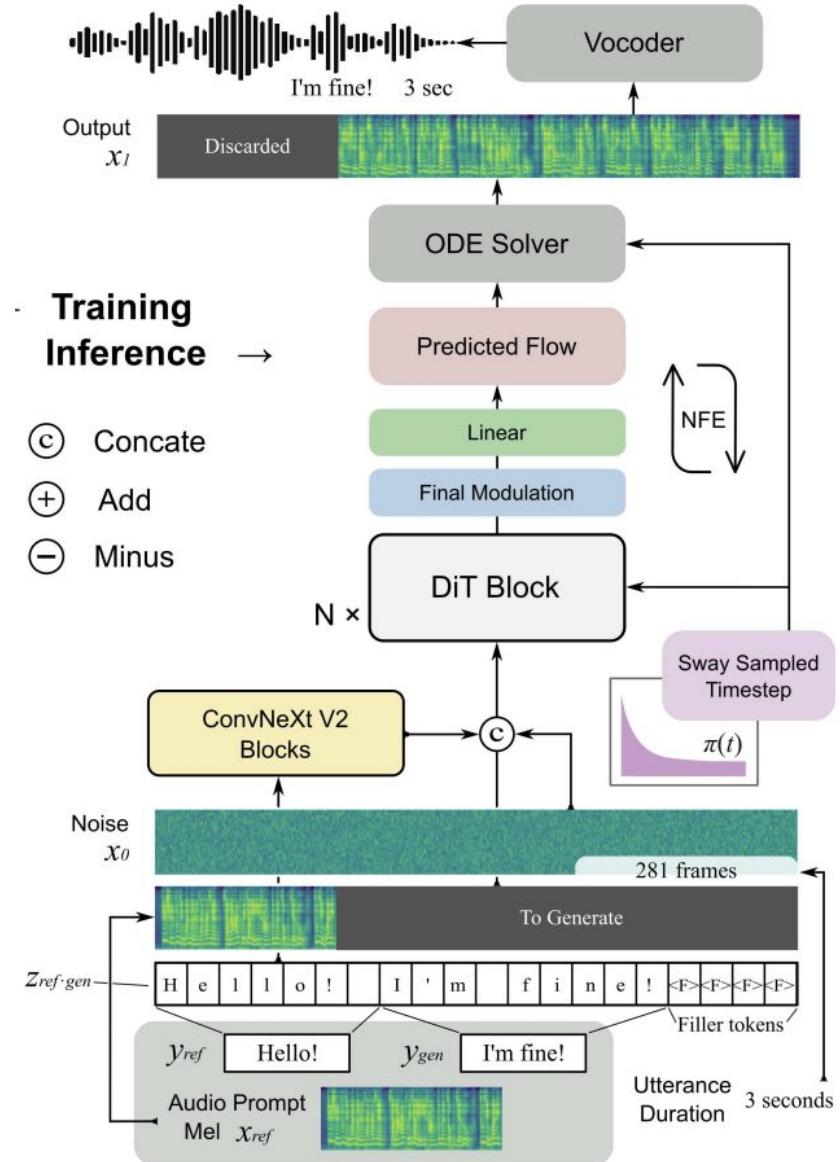
- ❑ Обработка текста – ConvNeXt V2 blocks (моделирует внутренние представления текста)
- ❑ Основной backbone – latent Diffusion Transformer (DiT)
- ❑ Positional embedding: flow step, текста
- ❑ CFG
- ❑ Предлагают более быстрый алгоритм семплирования Sway Sampling



[Статья](#), [github](#)

# F5TTS (2025)

- ❑ Датасет: in-the-wild multilingual speech dataset Emilia, 95K часов английского и китайского
- ❑ Ресурсы: 8 NVIDIA A100 80G GPUs
- ❑ Архитектура:
  - DiT: 22 layers, 16 attention heads, 1024/2048 embedding/FFN
  - ConvNeXt: 4 layers, 512/1024 embedding/FFN dimension
- ❑ Размер: 335.8M параметров
- ❑ Euler ODE solver, 32 итерации
- ❑ Вокодер: pre-trained Vocos



[Статья](#), [github](#)

# F5TTS (2025)

Model	#Param.	#Data	WER(%)↓	SIM-o↑	RTF↓
<b>LibriSpeech <i>test-clean</i></b>					
Ground Truth ( <i>2.2 hours subset</i> )			2.2	0.754	-
VALL-E 2	-	50K EN	2.44	0.643	0.732
MELLE	-	50K EN	2.10	0.625	0.549
MELLE-R2	-	50K EN	2.14	0.608	0.276
Voicebox	330M	60K EN	<b>1.9</b>	<b>0.662</b>	0.64
DiTTo-TTS	740M	55K EN	2.56	0.627	<b>0.162</b>
Ground Truth ( <i>40 samples subset</i> )			1.94	0.68	-
Voicebox	330M	60K EN	2.03	0.64	0.64
NaturalSpeech 3	500M	60K EN	<b>1.94</b>	0.67	0.296
MaskGCT	1048M	100K Multi.	2.634	<b>0.687</b>	-
<b>LibriSpeech-PC <i>test-clean</i></b>					
Ground Truth ( <i>1127 samples 2 hrs</i> )			2.23	0.69	-
Vocoder Resynthesized			2.32	0.66	-
CosyVoice	~300M	170K Multi.	3.59	0.66	0.92
FireRedTTS	~580M	248K Multi.	2.69	0.47	0.84
E2 TTS (32 NFE)	333M	100K Multi.	2.95	<b>0.69</b>	0.68
F5-TTS (16 NFE)	336M	100K Multi.	2.53	0.66	<b>0.15</b>
F5-TTS (32 NFE)	336M	100K Multi.	<b>2.42</b>	0.66	0.31

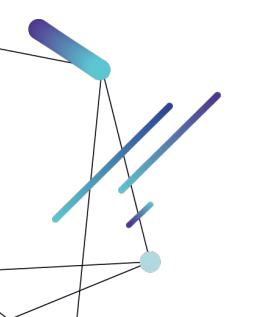
Model	WER(%)↓	SIM-o↑	CMOS↑	SMOS↑
<b>Seed-TTS <i>test-en</i></b>				
Ground Truth	2.06	0.73	0.00	3.91
Vocoder Resynthesized	2.09	0.70	-	-
CosyVoice	3.39	0.64	0.02	3.64
FireRedTTS	3.82	0.46	-1.46	2.94
MaskGCT	2.623*	<u>0.717*</u>	-	-
Seed-TTS <sub>DiT</sub>	<b>1.733*</b>	<b>0.790*</b>	-	-
E2 TTS (32 NFE)	2.19	0.71	0.06	<u>3.81</u>
F5-TTS (16 NFE)	1.89	0.67	<u>0.16</u>	3.79
F5-TTS (32 NFE)	1.83	0.67	<b>0.31</b>	<b>3.89</b>
<b>Seed-TTS <i>test-zh</i></b>				
Ground Truth	1.26	0.76	0.00	3.72
Vocoder Resynthesized	1.27	0.72	-	-
CosyVoice	3.10	0.75	-0.06	3.54
FireRedTTS	<u>1.51</u>	0.63	-0.49	3.28
MaskGCT	2.273*	<u>0.774*</u>	-	-
Seed-TTS <sub>DiT</sub>	<b>1.178*</b>	<b>0.809*</b>	-	-
E2 TTS (32 NFE)	1.97	0.73	-0.04	3.44
F5-TTS (16 NFE)	1.74	0.75	<u>0.02</u>	<u>3.72</u>
F5-TTS (32 NFE)	1.56	0.76	<b>0.21</b>	<b>3.83</b>

[Demo](#)

# Особенности flow-matching TTS моделей

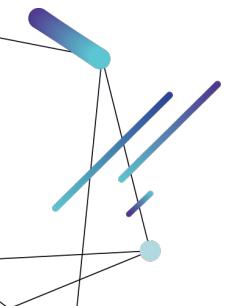


- хорошие моделирующие способности
- разнообразие
- скорость генерации

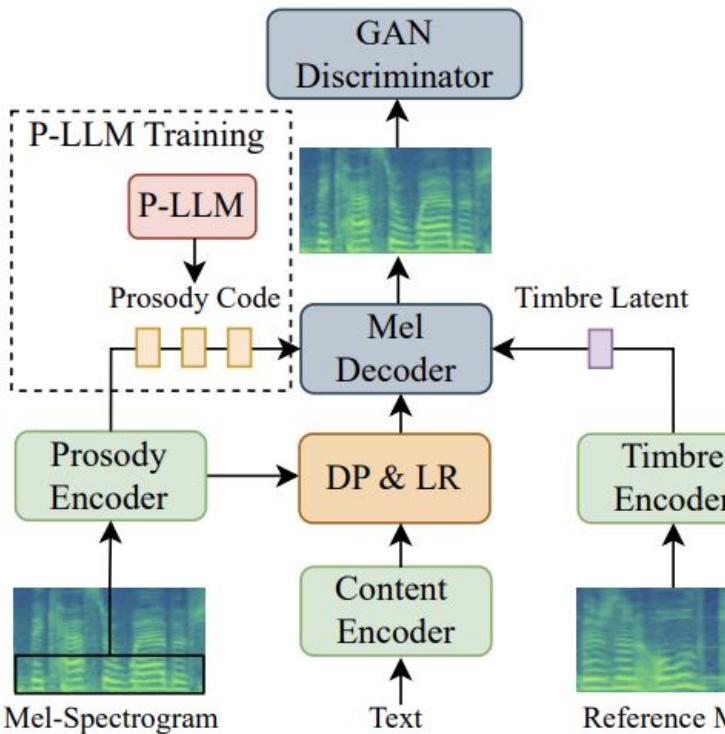


Другие модели: [DiTTo-TTS](#)

# Моделирование атрибутов речи



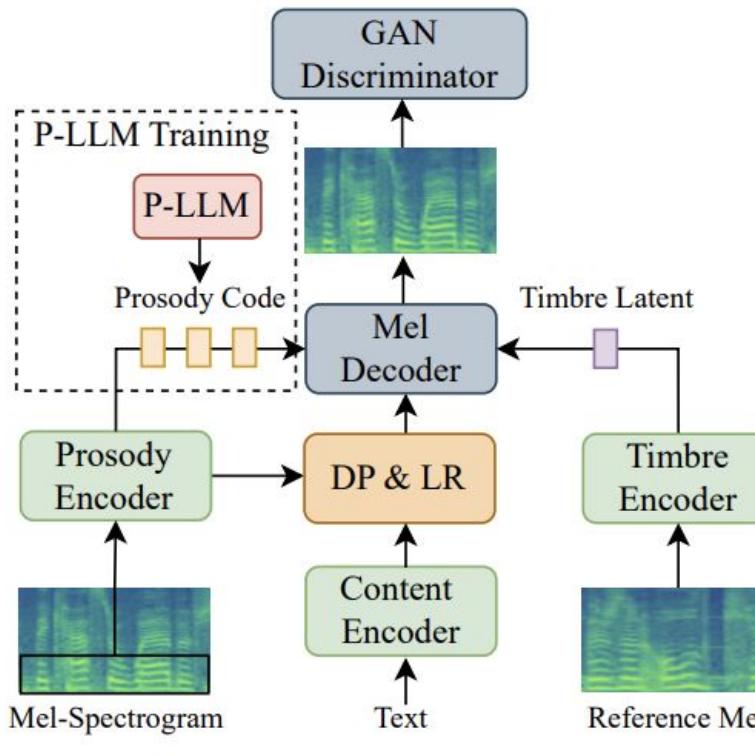
# MegaTTS (2023)



- Разделение речи на атрибуты, которые моделируются разными модулями
- Более понятный вариант, чем латентные признаки или дискретные токены
- Фокусируются на следующих атрибутах
  - используют мел-спектрограммы для отделения **фазы**
  - **голос** моделируется одним глобальным вектором на аудио
  - **просодия** кодируется с помощью латентных признаков из VQGAN-based обучаемого блока
  - **лингвистическая информация** кодируется текстовым энкодером

[Статья](#)

# MegaTTS (2023)



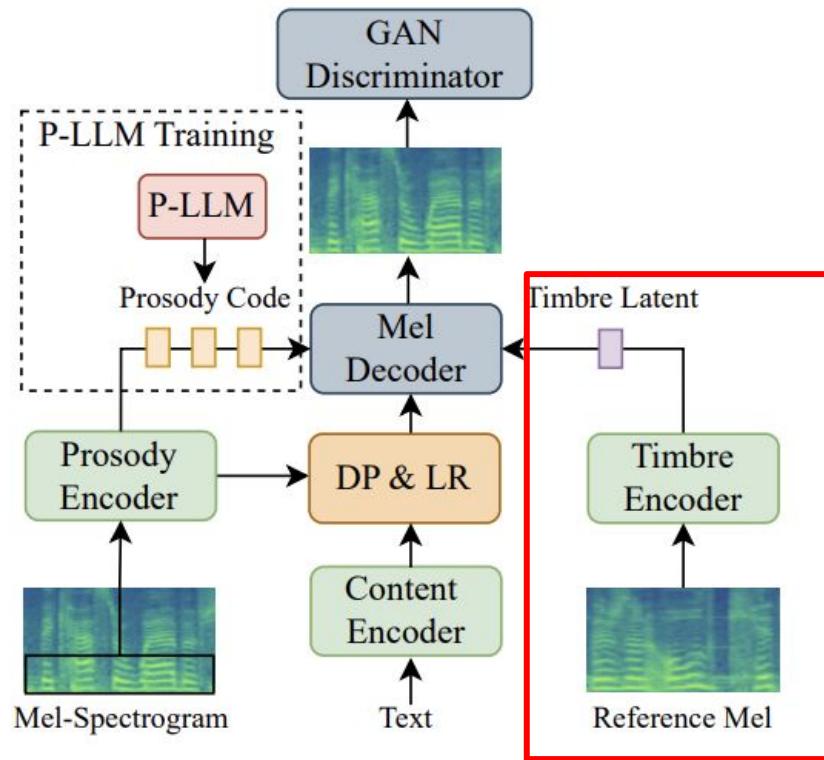
[Статья](#)

- Разделение речи на атрибуты, которые моделируются разными модулями
- Более понятный вариант, чем латентные признаки или дискретные токены
- Фокусируются на следующих атрибуатах
  - используют мел-спектрограммы для отделения **фазы**
  - **голос** моделируется одним глобальным вектором на аудио
  - **просодия** кодируется с помощью латентных признаков из VQGAN-based обучаемого блока
  - **лингвистическая информация** кодируется текстовым энкодером
- Обученные токены просодии моделируются LM  
Идея:
  - просодия моделирует и локальные и глобальные зависимости
  - не требуется жесткая монотонность как в при моделировании фонем

↓

LM хорошо подходит для просодических токенов

# MegaTTS (2023)



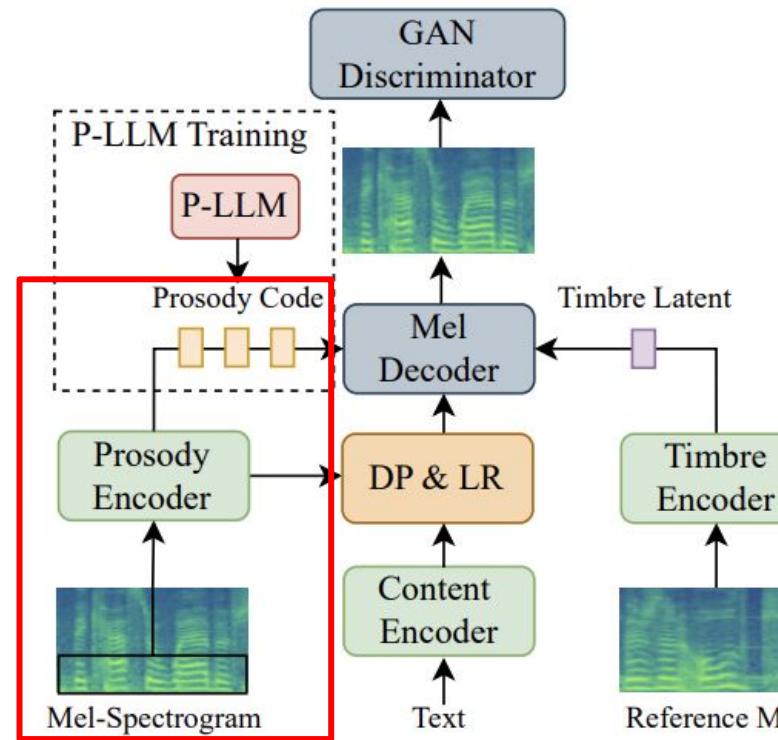
4 основных блока

## 1. Timbre encoder

- стек из сверток
- average pooling по оси времени
- вход: референсная мел-спектрограмма
- выход: эмбеддинг голоса  $H_{timbre}$

[Статья](#)

# MegaTTS (2023)

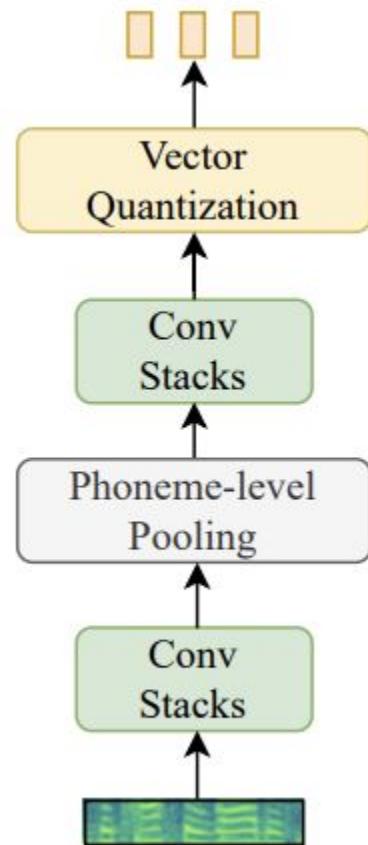


[Статья](#)

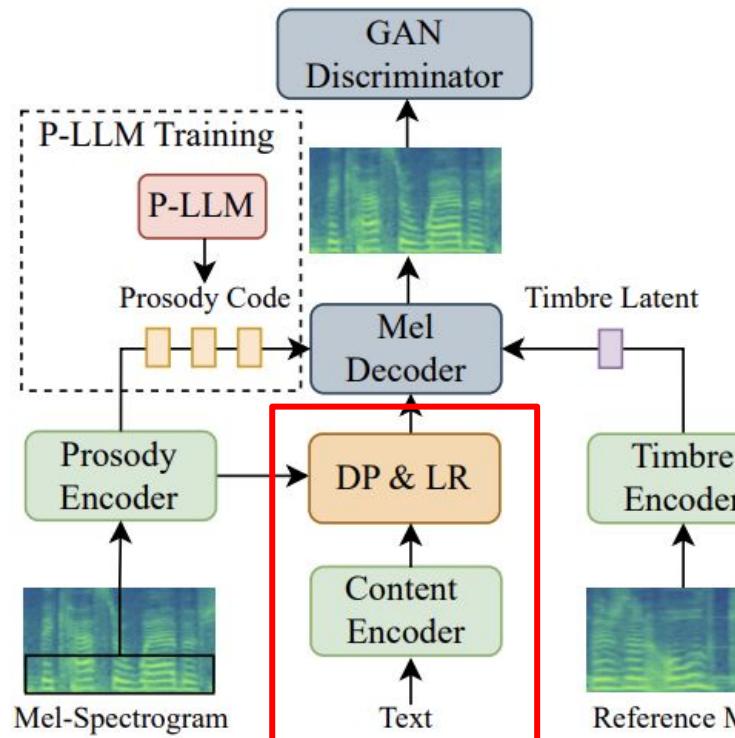
## 5 основных блоков

1. Timbre encoder
2. **Prosody encoder**

- архитектура
  - первый стек сверток + phoneme-level pooling сжимают длину спектrogramмы до уровня фонем
  - второй стек сверток учитывает phoneme-level зависимости
  - VQ слой превращает полученные представления в дискретные токены
- используют только первые 20 бинов мел-спектrogramмы – содержат всю информацию о просодии и меньше информации о контенте и тембре
- нужны gt длительности



# MegaTTS (2023)

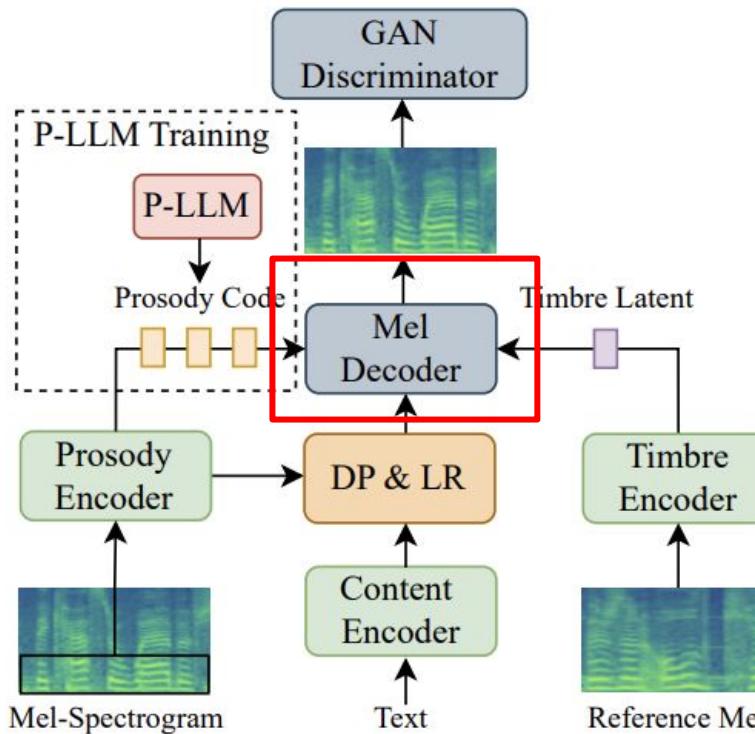


## 5 основных блоков

1. Timbre encoder
2. Prosody encoder
3. **Content encoder**
  - архитектура
    - 4-layer Transformer, 2 attention heads
    - 1D convolutions
  - Выравнивание с помощью duration predictor и upsampling
  - Duration predictor дополнительно обуславливается на информацию о просодии
  - DP состоит из 3-layer 1D convolution

[Статья](#)

# MegaTTS (2023)

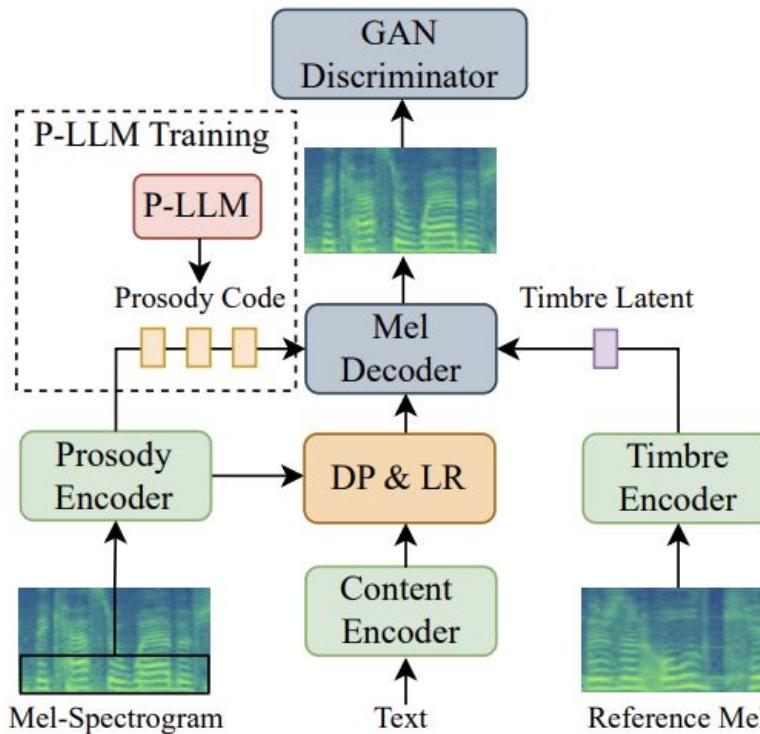


## 5 основных блоков

1. Timbre encoder
2. Prosody encoder
3. Content encoder
4. **Mel decoder – GAN-based**
  - стек сверток
  - multi-length discriminator (на окнах разной длины)
  - принимает на вход все признаки

[Статья](#)

# MegaTTS (2023)



[Статья](#)

5 основных блоков

1. Timbre encoder
2. Prosody encoder
3. Content encoder
4. Mel decoder

**Обучение**

- все блоки учатся вместе

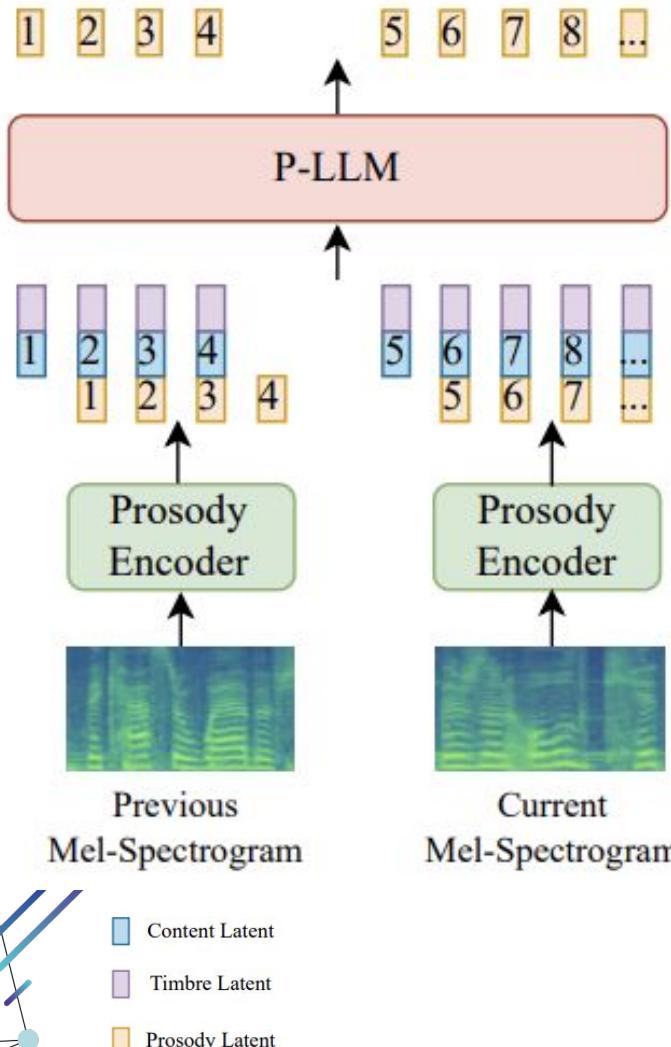
$$\mathcal{L}_{VQ} = \|y_t - \hat{y}_t\|^2 + \|\text{sg}[E(y_t)] - z_q\|_2^2 + \|\text{sg}[z_q] - E(y_t)\|_2^2$$

$$\mathcal{L} = \mathbb{E} [\mathcal{L}_{VQ} + \mathcal{L}_{Adv}] ,$$

- timbre encoder принимает на вход произвольную запись того же спикера для лучшего разделения от контента

При генерации нет референсной записи точно совпадающей с текущим текстом. Нужно как-то моделировать просодию по референсной записи → P-LLM

# MegaTTS (2023)



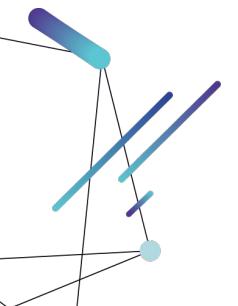
5 основных блоков

1. Timbre encoder
2. Prosody encoder
3. Content encoder
4. Mel decoder
5. PLLM

- учится отдельно
- конкатенируем текущий content vector, timbre vector и предыдущий prosody vector
- 8 Transformer layers with 8 attention heads + convolutions

# MegaTTS (2023)

- ❑ Датасет: GigaSpeech (en), WenetSpeech (zh) 20k часов
- ❑ Размер: 222.5M
- ❑ Ресурсы: 8 NVIDIA A100 GPUs
- ❑ Вокодер: HiFi-GAN V1
- ❑ RTF=0.3



# MegaTTS (2023)

- ❑ Датасет: GigaSpeech (en), WenetSpeech (zh) 20k часов
- ❑ Размер: 222.5M
- ❑ Ресурсы: 8 NVIDIA A100 GPUs
- ❑ Вокодер: HiFi-GAN V1
- ❑ Дополнительные метрики
  - Objective: pitch distance. Dynamic time warping (DTW) алгоритм для сравнения питчей.
  - Subjective: MOS-P – естественность просодии (высоты, энергии, длительностей)

## Zero-shot TTS

Dataset	Method	Subjective		Objective	
		MOS-Q ( $\uparrow$ )	MOS-P ( $\uparrow$ )	MOS-S ( $\uparrow$ )	Pitch ( $\downarrow$ )
VCTK	Ground Truth	4.35 $\pm$ 0.11	4.48 $\pm$ 0.10	4.33 $\pm$ 0.13	-
	YourTTS [8]	4.04 $\pm$ 0.10	4.18 $\pm$ 0.09	3.76 $\pm$ 0.12	32.43
	Mega-TTS	<b>4.27 <math>\pm</math> 0.09</b>	<b>4.32 <math>\pm</math> 0.11</b>	<b>4.27 <math>\pm</math> 0.10</b>	<b>17.45</b>
LibriSpeech	Ground Truth	4.23 $\pm$ 0.13	4.49 $\pm$ 0.11	4.29 $\pm$ 0.16	-
	YourTTS [8]	3.83 $\pm$ 0.12	4.06 $\pm$ 0.13	3.22 $\pm$ 0.21	44.05
	Mega-TTS	<b>4.08 <math>\pm</math> 0.17</b>	<b>4.21 <math>\pm</math> 0.17</b>	<b>3.90 <math>\pm</math> 0.18</b>	<b>35.46</b>

Method	CMOS-Q	CMOS-P	MOS-S ( $\uparrow$ )
VALL-E [58]	-0.23	-0.27	4.06 $\pm$ 0.22
Mega-TTS	<b>0.00</b>	<b>0.00</b>	<b>4.11 <math>\pm</math> 0.21</b>

[Demo](#)

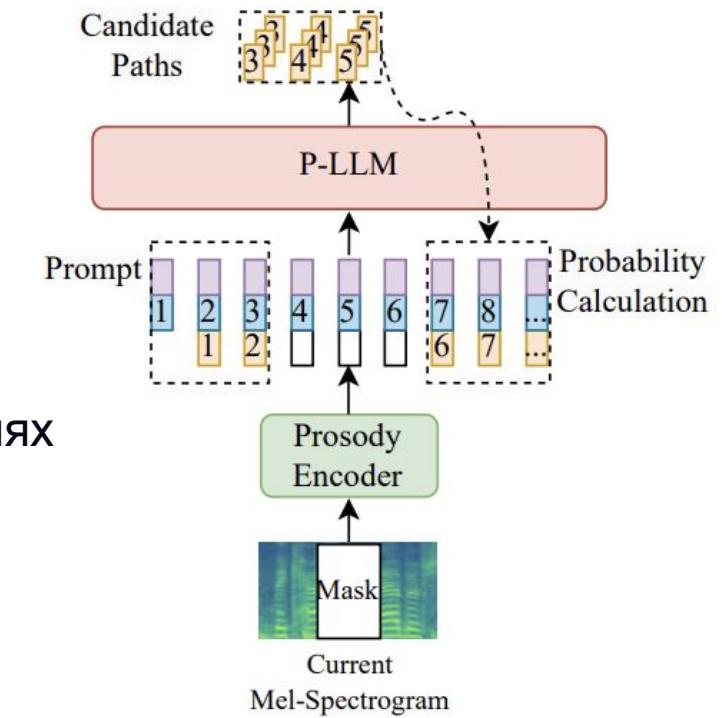
# MegaTTS (2023)

**Speech editing** – замена фонем, восстановление просодии

Method	MOS-Q ( $\uparrow$ )	MOS-P ( $\uparrow$ )	MOS-S ( $\uparrow$ )
EditSpeech [52]	$3.57 \pm 0.12$	$3.87 \pm 0.14$	$3.93 \pm 0.14$
A <sup>3</sup> T [3]	$3.73 \pm 0.13$	$3.96 \pm 0.14$	$3.97 \pm 0.12$
Mega-TTS	<b><math>3.81 \pm 0.14</math></b>	<b><math>4.11 \pm 0.14</math></b>	<b><math>4.36 \pm 0.16</math></b>

**Robustness evaluation** – количество ошибок на 50 сложных предложениях

Method	Repeats	Skips	Error Sentences	Error Rate
Tacotron [59]	10	16	22	44%
VALL-E [58]	8	11	14	28%
FastSpeech [48]	0	0	0	0%
Mega-TTS	0	0	0	0%



[Demo](#)

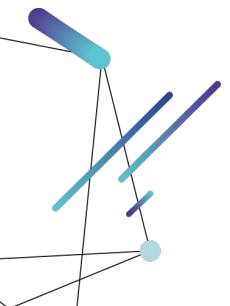
# MegaTTS series

## MegaTTS2 (2024)

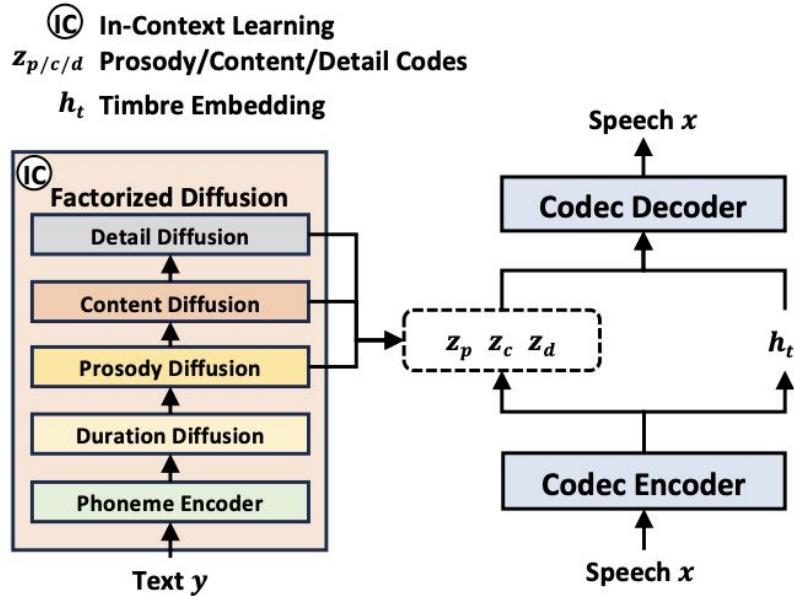
- multi-reference timbre encoder (аналог few-shot)
- PLLM не обуславливается на текст
- prosody interpolation method – перенос стиля от другого спикера

## MegaTTS3 (2025)

- innovative sparse alignment algorithm
- диффузионная архитектура DiT
- может генерировать аудио длины 1 минута
- не сохранилась исходная идея с разделение аудио на атрибуты

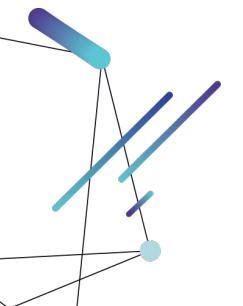


# NaturalSpeech3 (2024)



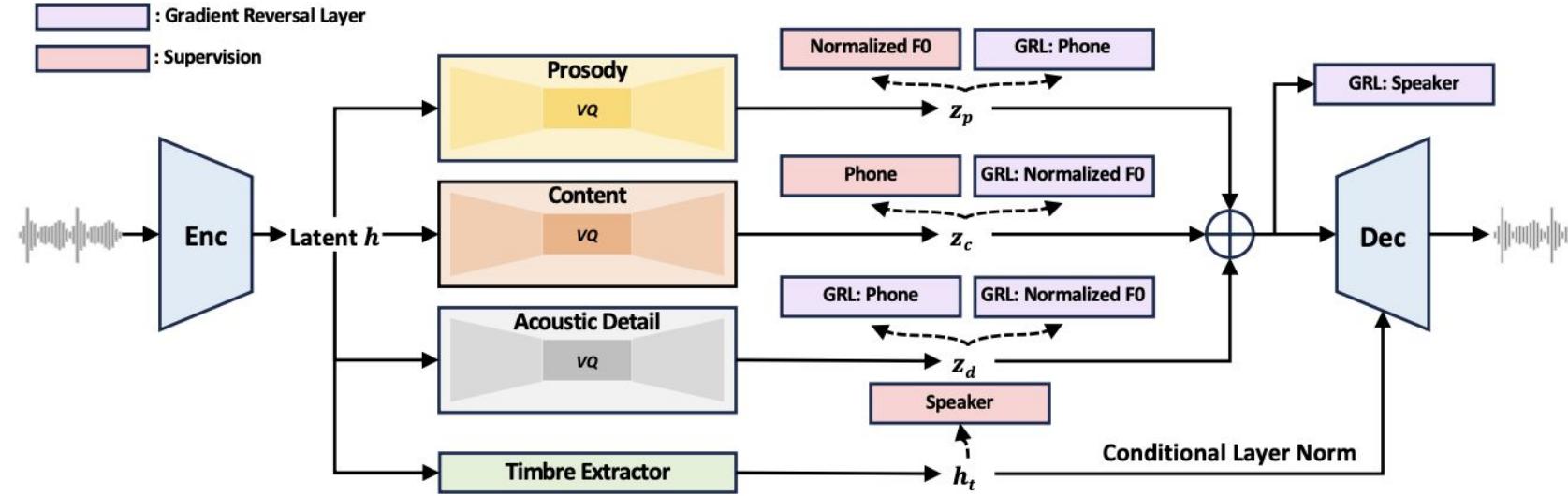
- ❑ Схожая идея с разделением аудио на атрибуты
- ❑ Здесь выполняется с помощью кодека FaCodec (factorized codec)
- ❑ Каждый из атрибутов далее иерархически моделируется с помощью диффузионных моделей (factorized diffusion model)
- ❑ Неавторегрессионная

[Статья](#), [github](#) for codec



# NaturalSpeech3 (2024)

FaCodec



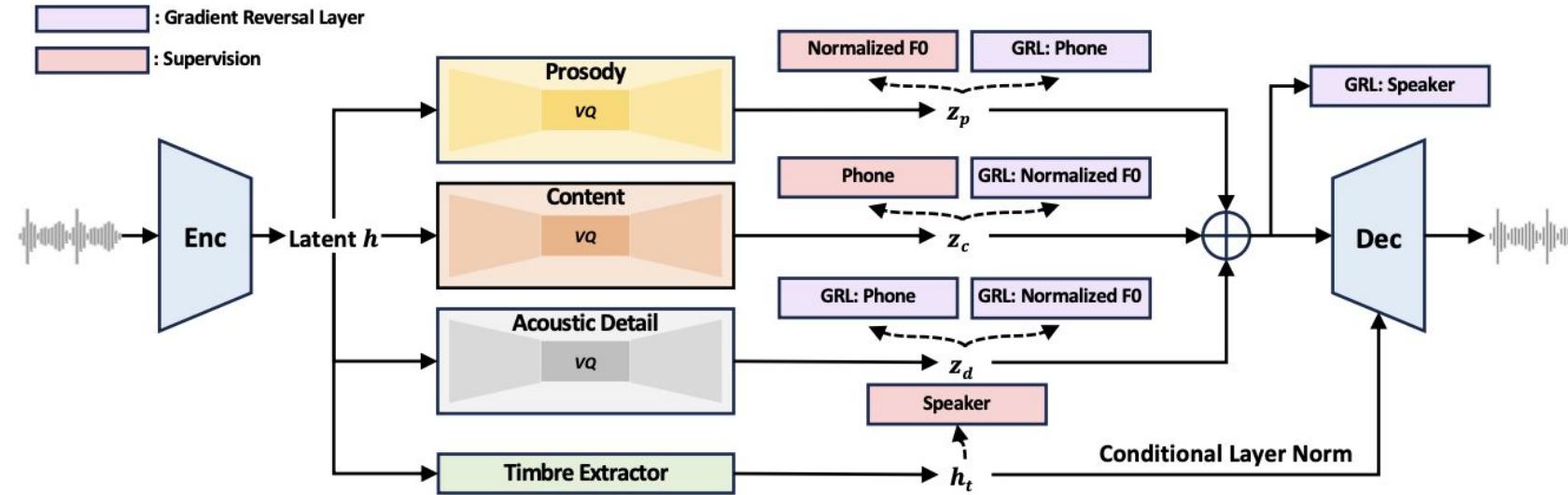
Вместо RVQ factorized vector quantizers (FVQ)

Состоит из следующих блоков:

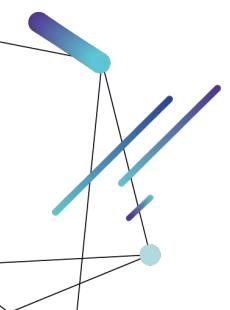
- **Encoder** – последовательность сверток, которая делает downsampling из 16k семплов в 200 ( $h$ )
- **Timbre extractor** – Transformer encoder, преобразует  $h$  в глобальный вектор голоса  $h_t$
- **3 FVQ** для моделирования просодии, контента и акустических деталей – fine-grained атрибуты, представляемые в виде дискретных токенов
- **Decoder** – зеркальный энкодер, но с большим количеством параметров

# NaturalSpeech3 (2024)

FaCodec

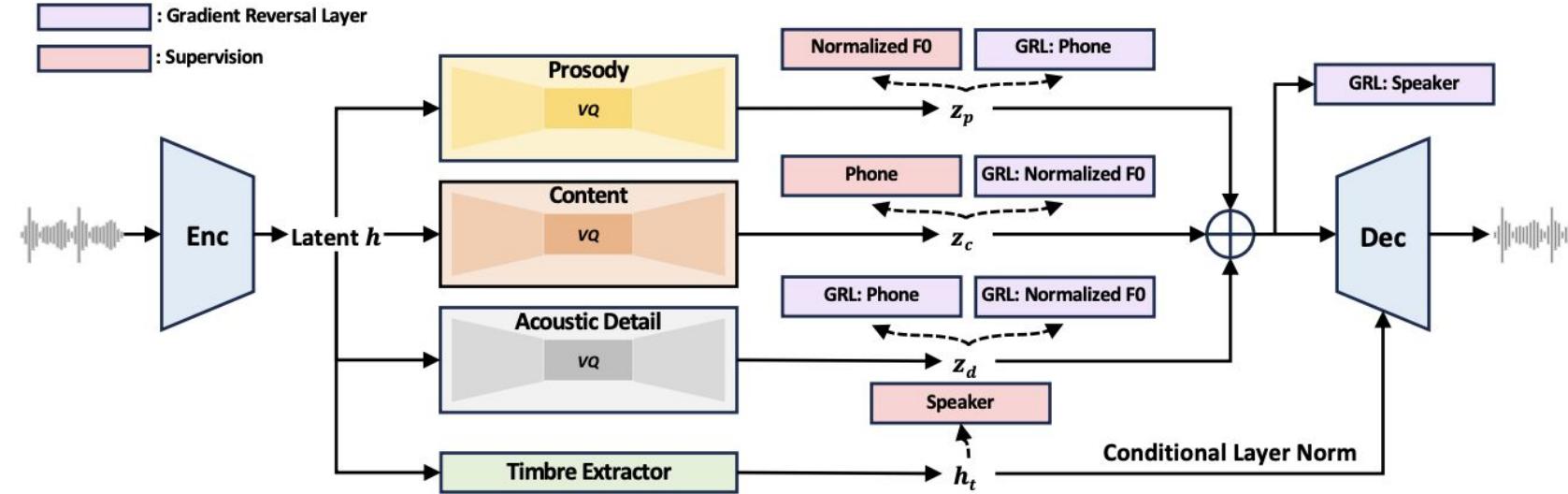


Простая факторизация не гарантирует disentangled пространства



# NaturalSpeech3 (2024)

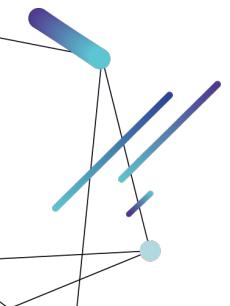
FaCodec



Простая факторизация не гарантирует disentangled пространства

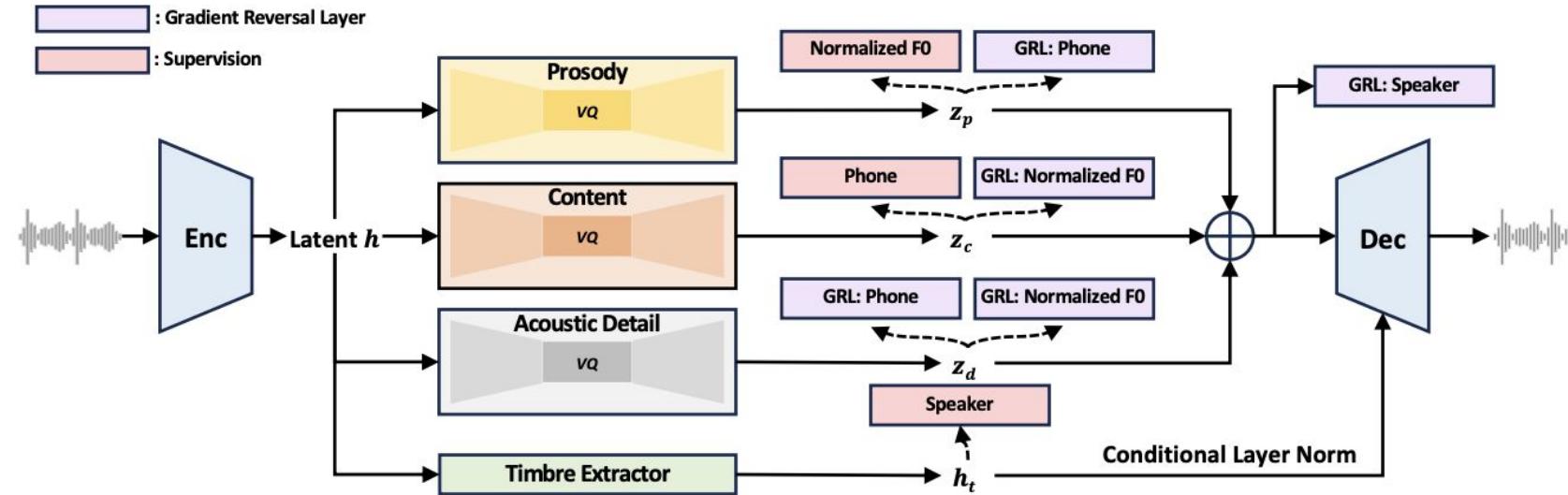
Специальные техники:

- **Information bottleneck**: проецирование в низкоразмерное пространство, в котором и делается квантизация. Гарантирует, что такие эмбеддинги дискретных кодов содержат меньше информации



# NaturalSpeech3 (2024)

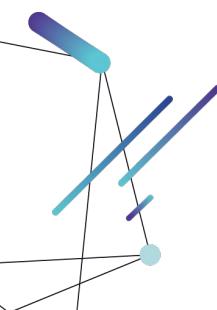
FaCodec



Простая факторизация не гарантирует disentangled пространства

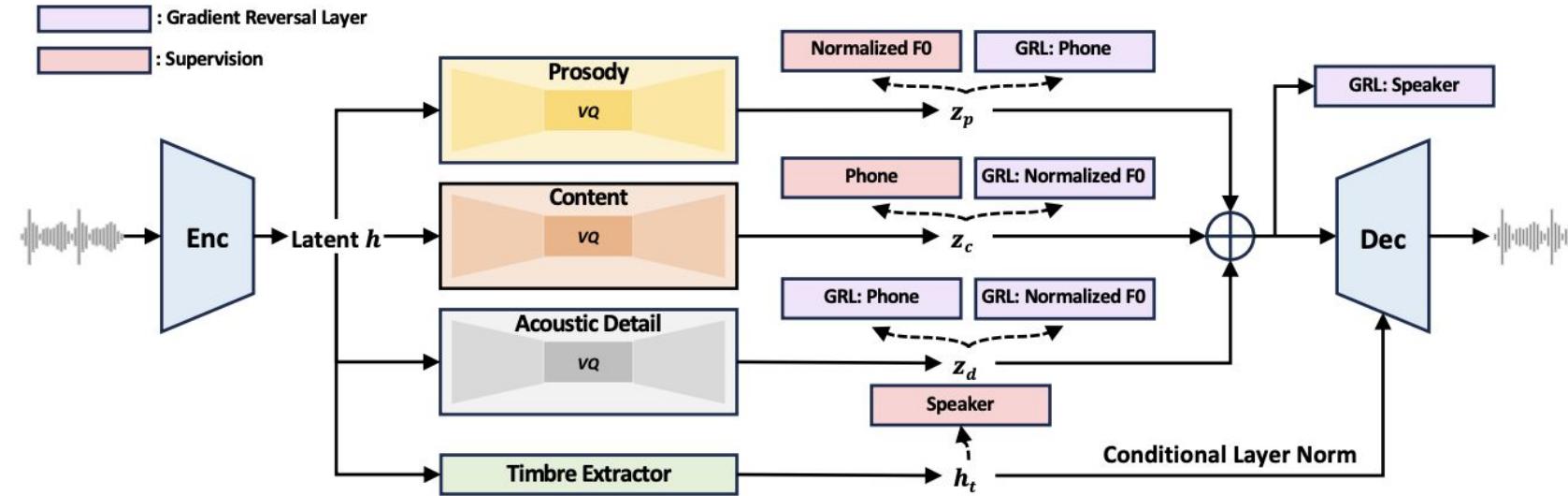
Специальные техники:

- **Information bottleneck**: проецирование в низкоразмерное пространство, в котором и делается квантизация. Гарантирует, что такие эмбеддинги дискретных кодов содержат меньше информации
- **Supervision**: дополнительная downstream задача на предсказание каких-либо GT меток



# NaturalSpeech3 (2024)

FaCodec



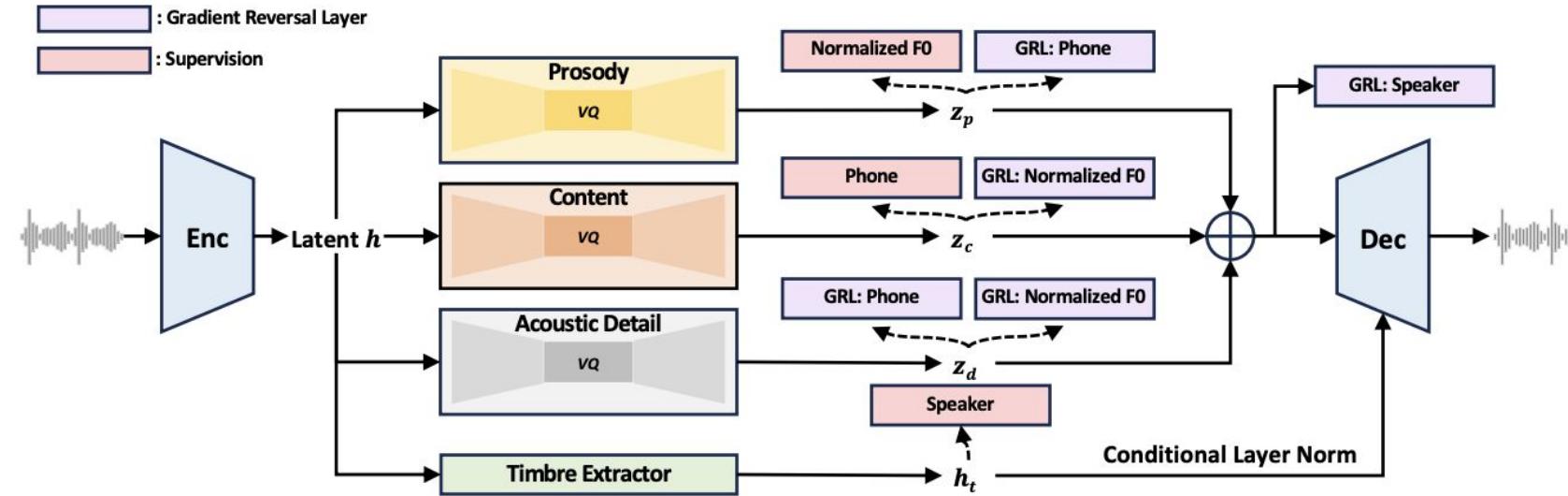
Простая факторизация не гарантирует disentangled пространства

Специальные техники:

- **Information bottleneck**: проецирование в низкоразмерное пространство, в котором и делается квантизация. Гарантирует, что такие эмбеддинги дискретных кодов содержат меньше информации
- **Supervision**: дополнительная downstream задача на предсказание каких-либо GT меток
- **Gradient Reversal**: avoiding the information leak, eliminate undesired information in latent space. На backpropagation градиент на предсказание ненужной информации берут со знаком “-”

# NaturalSpeech3 (2024)

FaCodec



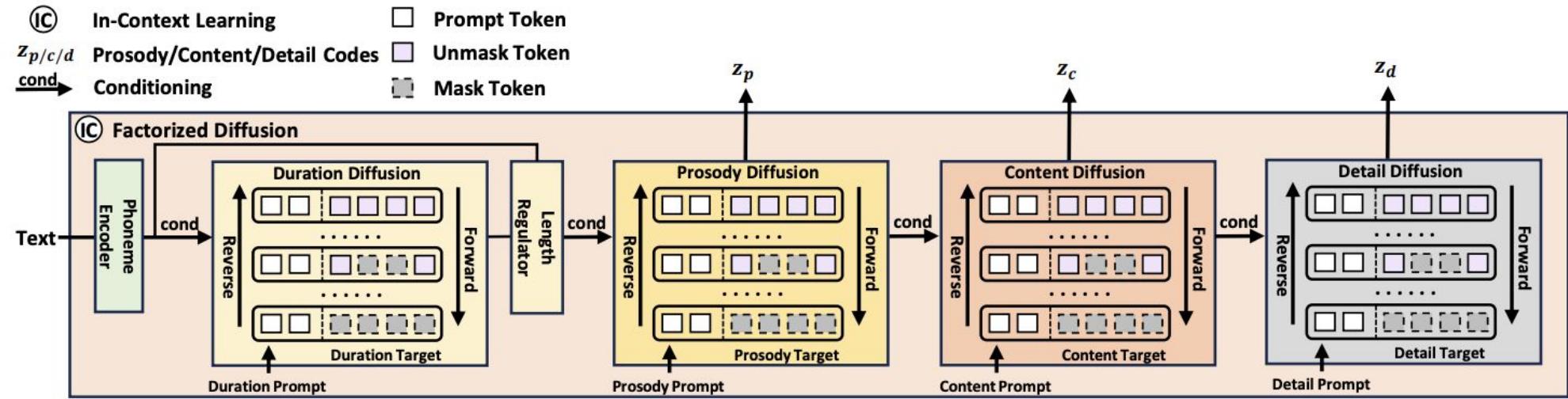
Простая факторизация не гарантирует disentangled пространства

Специальные техники:

- **Information bottleneck**: проецирование в низкоразмерное пространство, в котором и делается квантизация. Гарантирует, что такие эмбеддинги дискретных кодов содержат меньше информации
- **Supervision**: дополнительная downstream задача на предсказание каких-либо GT меток
- **Gradient Reversal**: avoiding the information leak, eliminate undesired information in latent space. На backpropagation градиент на предсказание ненужной информации берут со знаком “-”
- **Detail Dropout**: маскирование акустических признаков с некоторой вероятностью  $p$

# NaturalSpeech3 (2024)

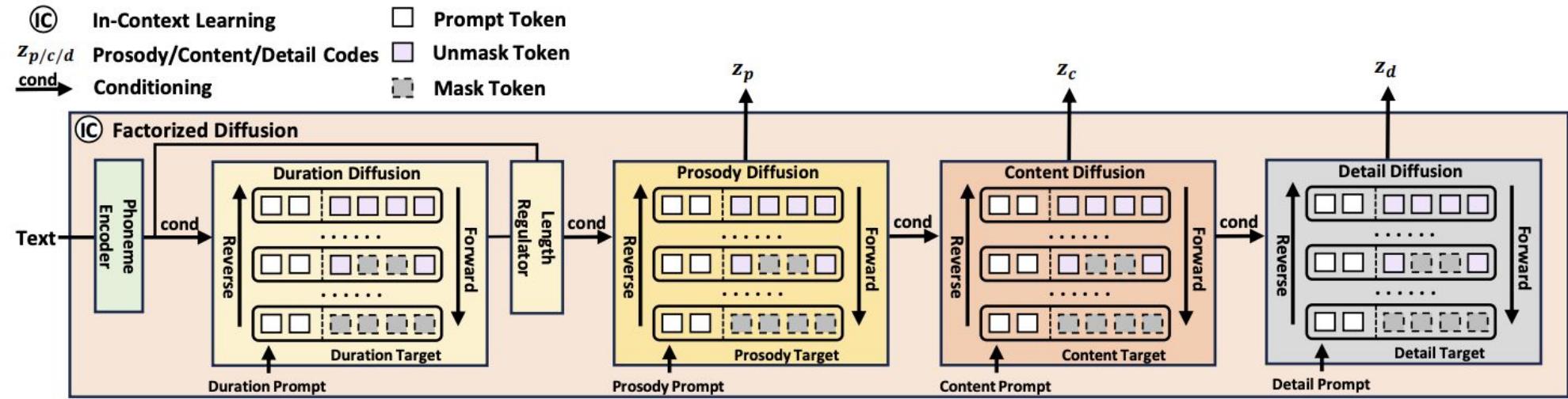
## Factorized Diffusion Model



- Восстанавливают вектора  $z$
- Поэтапная реконструкция. Все части обуславливаются на промпт и признаки из предыдущего блока
- На первом этапе важно восстановить длительности для выравнивания текста и аудио и моделирования других признаков на уровне аудио
- На последнем этапе восстанавливаем акустические детали

# NaturalSpeech3 (2024)

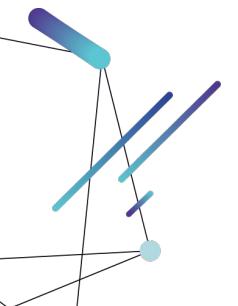
## Factorized Diffusion Model



- Восстанавливают вектора  $z$
- Поэтапная реконструкция. Все части обуславливаются на промпт и признаки из предыдущего блока
- На первом этапе важно восстановить длительности для выравнивания текста и аудио и моделирования других признаков на уровне аудио
- На последнем этапе восстанавливаем акустические детали
- Для in-context learning при использовании промпта берется незашумленная версия
- Вектор голоса здесь не моделируется. Он напрямую извлекается из аудио и подается на вход декодеру

# NaturalSpeech3 (2024)

- ❑ Датасет: LibriLight, 60k часов, ~7000 голосов, 16KHz
- ❑ Ресурсы:
  - FaCodec: 8 NVIDIA TESLA V100 32GB GPUs
  - Factorized Diffusion Model: 8 A100 80GB GPUs
- ❑ Размер diffusion model: 500M
- ❑ по 4 итерации для каждого диффузионного блока, всего 60 итераций с учетом CFG
- ❑ Сценарии использования:
  - Zero-shot TTS
  - копирование эмоций
  - модификация атрибутов



# NaturalSpeech3 (2024)

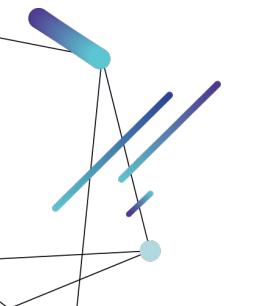
**LibriSpeech test-clean**

	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	<b>0.67</b>	<b>0.76</b>	<b>1.81</b>	<b>0.00</b>	<b>4.01</b>

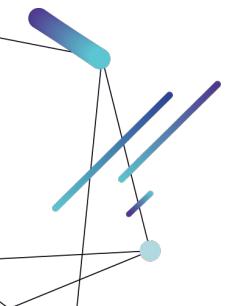
**RAVDESS**

	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	<b>4.28</b>	<b>0.52</b>	<b>0.00</b>	<b>4.72</b>

[Demo](#)

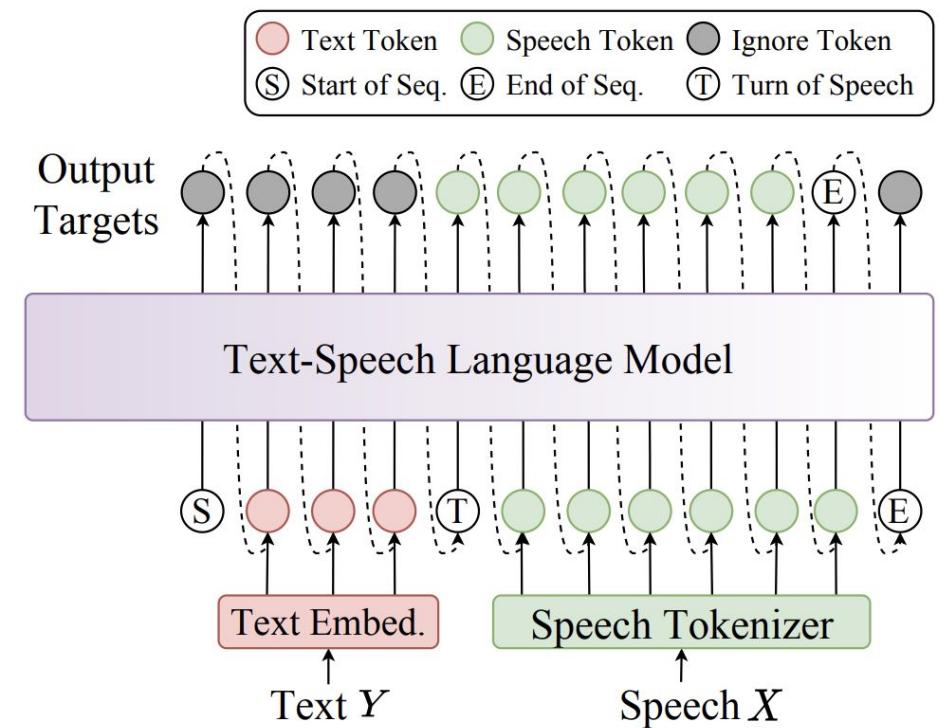


# Смешанные модели



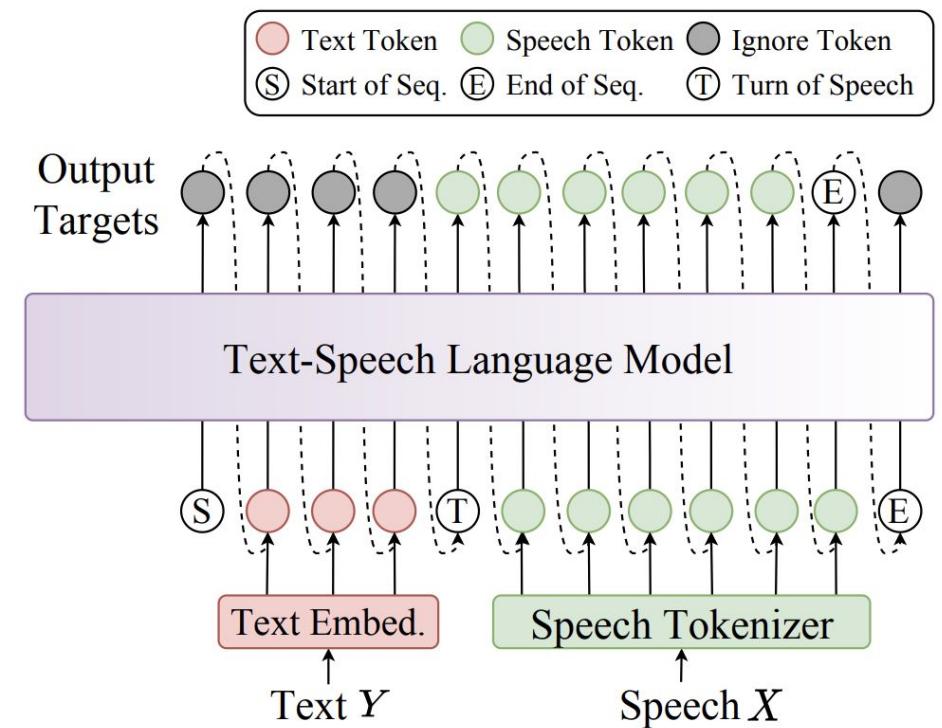
# CosyVoice2 (2025)

- ❑ Каскадная модель из трех блоков:
  - LM-based предсказание токенов
  - Flow matching блок для восстановления спектрограмм
  - Вокодер
- ❑ Supervise токенайзер с частотой 25Hz и одним кодбуком
- ❑ Поддержка стриминга



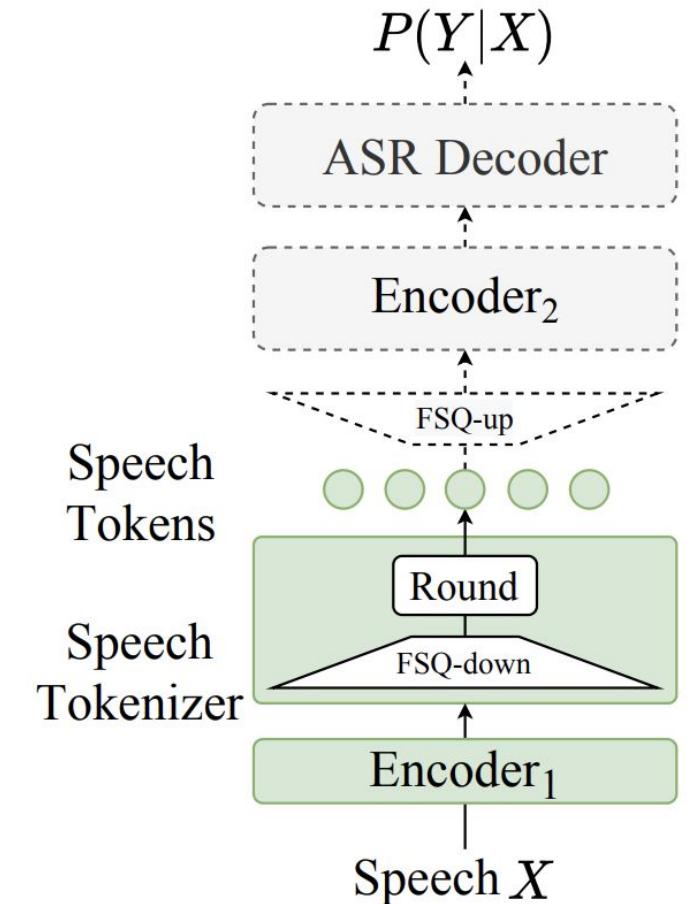
# CosyVoice2 (2025)

- Текст токенизируется BPE



# CosyVoice2 (2025)

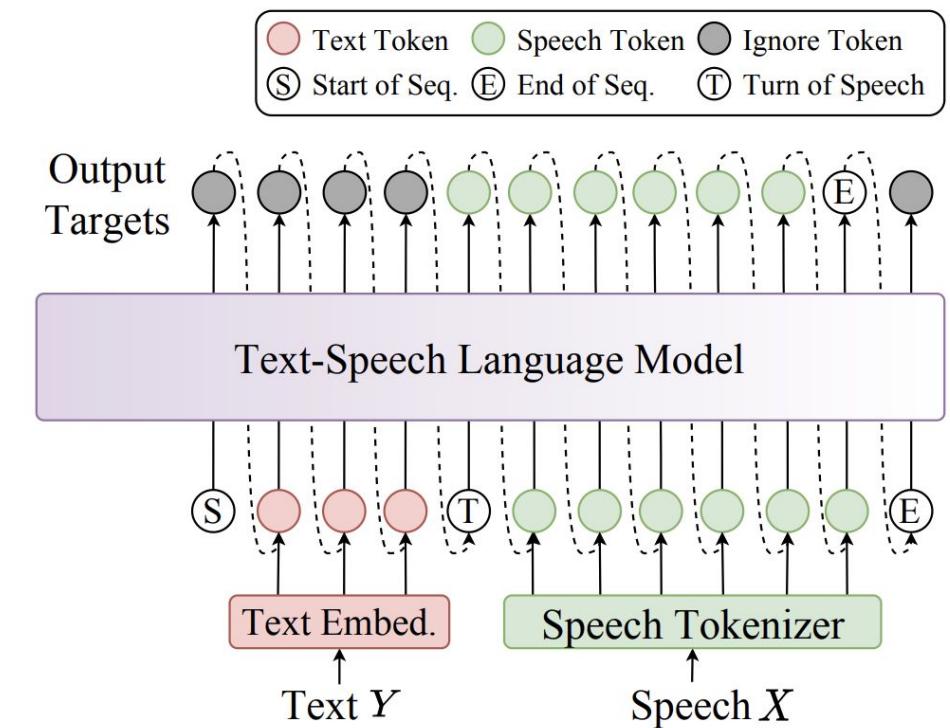
- Текст токенизируется BPE
- **Аудио токенизатор:**
  - используется обученная ASR модель SenseVoice
  - добавляется FVQ слой
  - дообучение на задачу ASR
  - первая часть (6 Transformer блоков) используется как токенайзер
  - частота 25Hz



[Статья](#), [github](#)

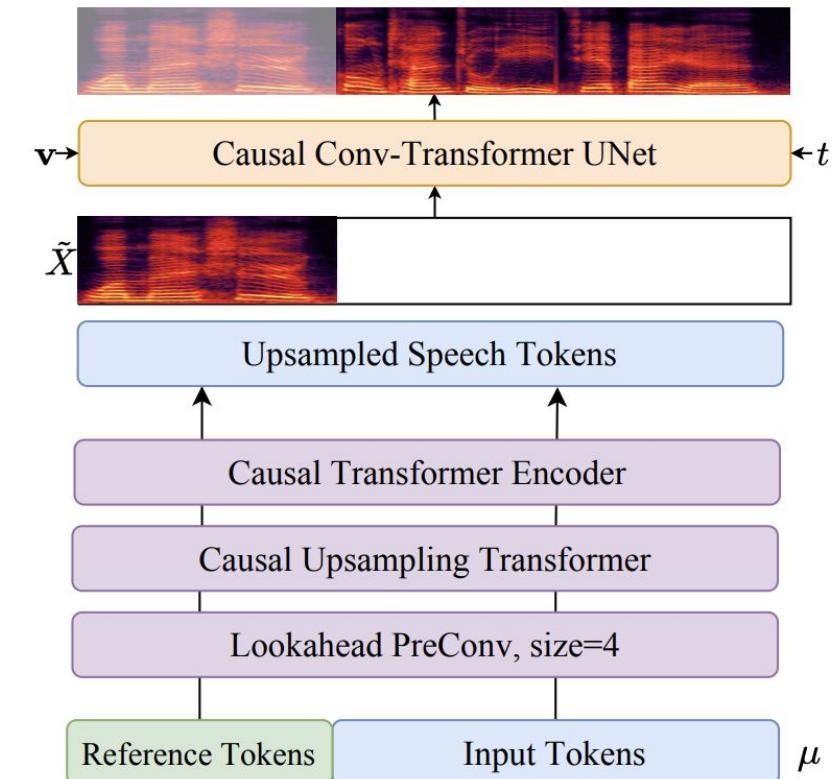
# CosyVoice2 (2025)

- Текст токенизируется BPE
- Аудио токенизатор
- LM:
  - инициализируется Qwen2.5-0.5B
  - на вход получает такую последовательность ->
  - не принимает эмбеддинг голоса по сравнению с CosyVoice



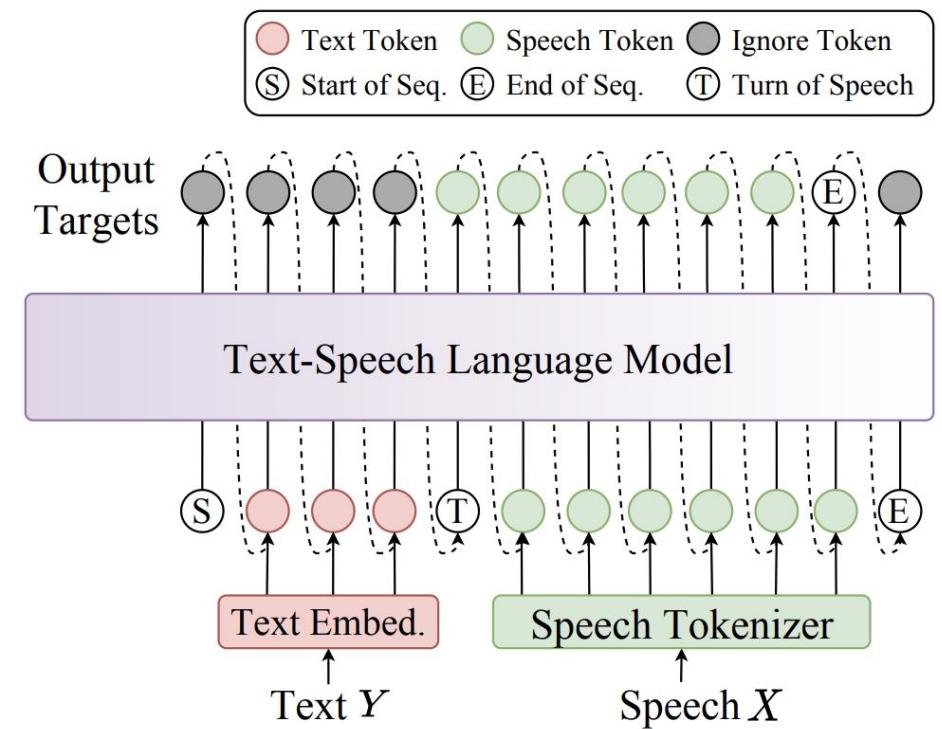
# CosyVoice2 (2025)

- Текст токенизируется BPE
- Аудио токенизатор
- LM
- **Flow matching блок:**
  - восстанавливает из дискретных токенов мел спектrogramму
  - частота 50Hz, поэтому дискретные токены нужно апсемплить
  - состоит из conv + causal transformer (upsampling) + conv Unet
  - во время обучения маскируется 70-100% последних фреймов
  - обуславливается на вектор голоса



# CosyVoice2 (2025)

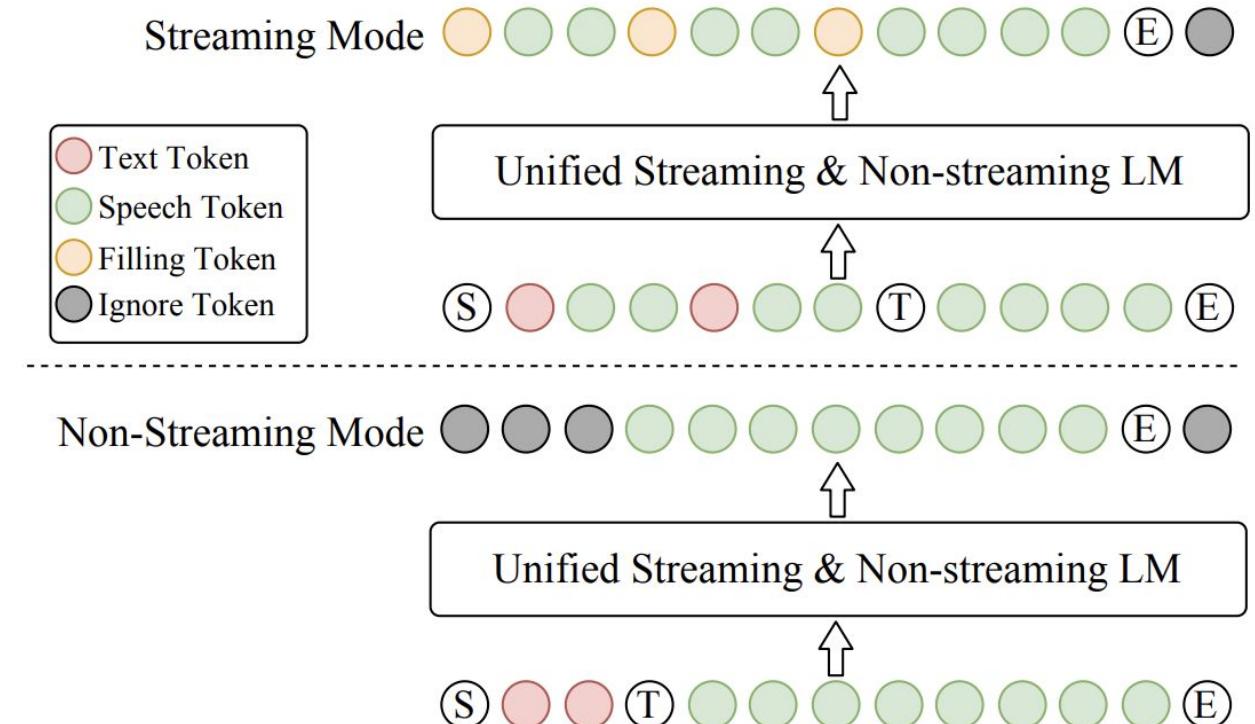
- Текст токенизируется BPE
- Аудио токенизатор
- LM
- Flow matching блок
- Вокодер HiFi-GAN**



# CosyVoice2 (2025)

## Поддержка стриминга

- LM:
  - чередование текстовых и акустических токенов в пропорции 5:15
  - filler tokens перед следующей частью текстовых токенов



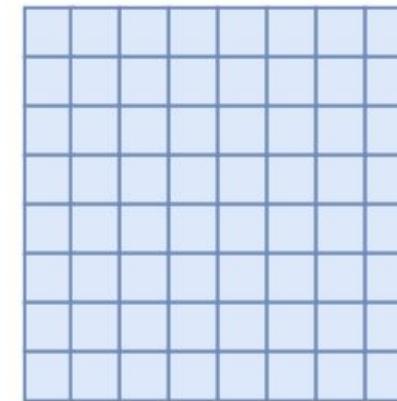
[Статья, github](#)

# CosyVoice2 (2025)

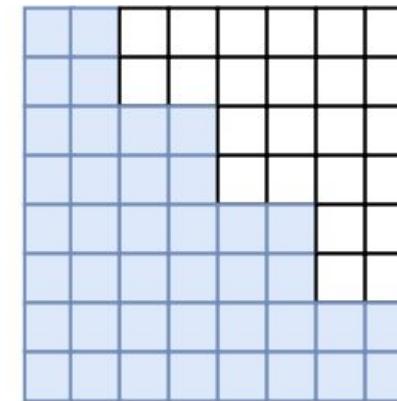
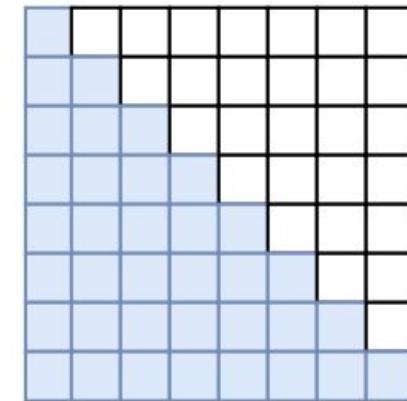
## Поддержка стриминга

- LM:
  - чередование текстовых и акустических токенов в пропорции 5:15
  - filler tokens перед следующей частью текстовых токенов
- Flow matching
  - пробовали разные типы каузальных масок
  - лучший вариант – блочная

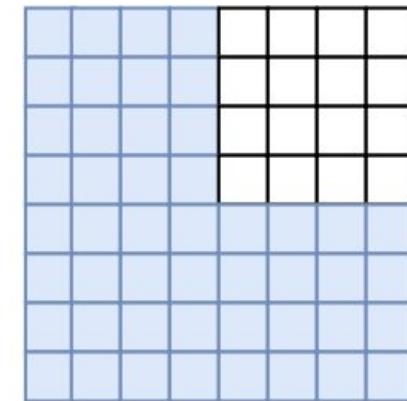
Non-causal Mask



Full-causal Mask



Chunk-M Mask



Chunk-2M Mask

[Статья](#), [github](#)

# CosyVoice2 (2025)

## Дообучение

- Instructed Generation
  - [laughter], [breath]
  - теги <strong>, <laughter>, ...
  - 1500 часов

### *Natural Language Instruction*

**Emotion:** 高兴(Happy), 悲伤(Sad), 惊讶(Surprised), 愤怒(Angry), 恐惧(Fearful), 厌恶(Disgusted), 冷静(Calm), 严肃(Serious)

**Speaking Rate:** 快速(Fast), 非常快速(Very Fast), 慢速(Slow), 非常慢速(Very Slow)

**Dialect:** 粤语, 四川话, 上海话, 郑州话, 长沙话, 天津话

**Role-playing:** 神秘(Mysterious), 凶猛(Fierce), 好奇(Curious), 优雅(Elegant), 孤独(Lonely), 机器人(Robot), 小猪佩奇(Peppa), etc.

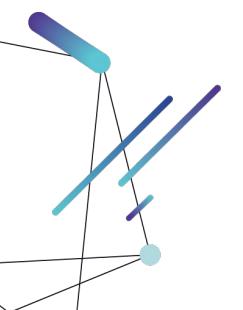
### *Fine-grained Instruction*

**Vocal Bursts:** [laughter], [breath], etc.

**Vocal Features:** <laughter></laughter>, <strong></strong>

### *Examples*

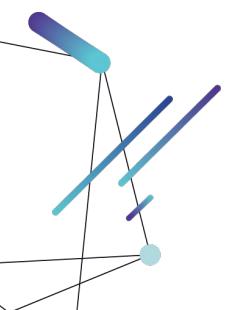
- 你能用高兴的情感说吗? < |endofprompt| >今天真是太开心了, 马上要放假了! I'm so happy, Spring Festival is coming!
- Please speaking very fast.< |endofprompt| >Today is a happy day, full of laughter and joy.
- 请问你能模仿粤语的口音吗? < |endofprompt| >多保重, 早休息。
- 尝试一下以机器人的角色和我交流。< |endofprompt| >接收知识光波!
- [laughter]有时候, 看着小孩子们的天真行为[laughter], 我们总会会心一笑。
- She pursued her dreams with <strong>enthusiasm</strong> and <strong>grit</strong>.



# CosyVoice2 (2025)

## Дообучение

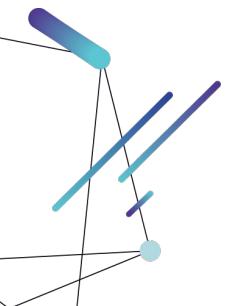
- Instructed Generation
- Multi-speaker SFT
- Reinforcement learning для улучшения similarity



# CosyVoice2 (2025)

- ❑ Датасет:
  - Speech tokenizer: 200kh английского и китайского
  - 168kh английского, китайского, японского и корейского
- ❑ Ресурсы: 64 V100 32Mb GPUs
- ❑ -

Language	Duration (hr)
ZH	130,000
EN	30,000
Yue	5,000
JP	4,600
KO	2,200



# CosyVoice2 (2025)

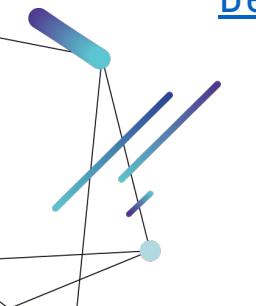
LibriSpeech

Model	WER (%)	NMOS	SS
<b>Human</b>	2.66	3.84	0.697
<b>ChatTTS [56]</b>	6.84	3.89	-
<b>GPT-SoVITs [57]</b>	5.13	3.93	0.405
<b>OpenVoice [58]</b>	3.47	3.87	0.299
<b>ParlerTTS [59]</b>	3.16	3.86	-
<b>EmotiVoice [60]</b>	3.14	3.93	-
<b>CosyVoice [34]</b>	2.89	3.93	0.743
<b>CosyVoice 2</b>	2.47	<b>3.96</b>	0.745
<b>CosyVoice 2-S</b>	<b>2.45</b>	3.90	<b>0.751</b>

Seed-TTS

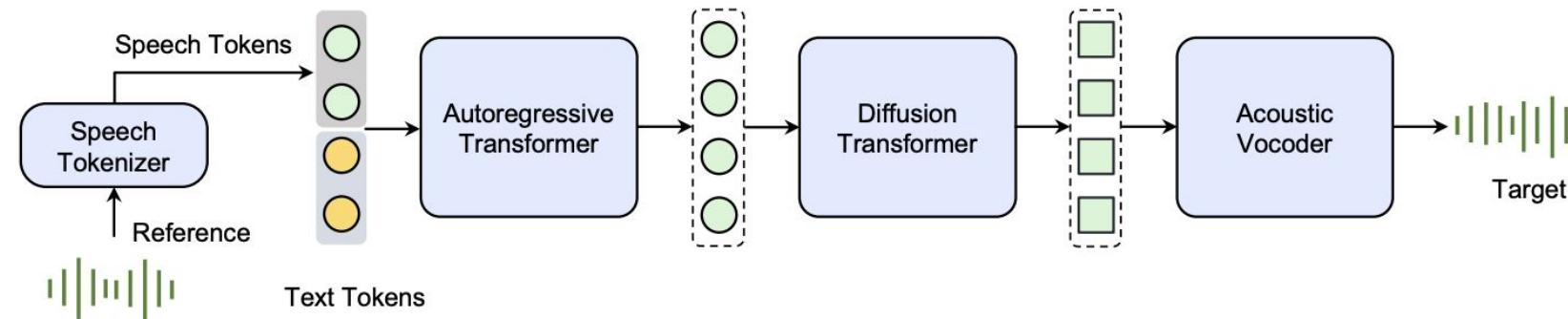
Model	<i>test-zh</i>		<i>test-en</i>		<i>test-hard</i>	
	CER (%) ↓	SS ↑	WER (%) ↓	SS ↑	WER (%) ↓	SS ↑
<b>Human</b>	1.26	0.755 (0.775)	2.14	0.734 (0.742)	-	-
<b>Vocoder Resyn.</b>	1.27	0.720	2.17	0.700	-	-
<b>Seed-TTS<sup>†</sup> [33]</b>	1.12	0.796	2.25	0.762	7.59	0.776
<b>FireRedTTS [35]</b>	1.51	0.635 (0.653)	3.82	0.460 (0.526)	17.45	0.621 (0.639)
<b>MaskGCT [18]</b>	2.27	0.774 (0.752)	2.62	0.714 (0.730)	10.27	0.748 (0.720)
<b>E2 TTS (32 NFE)<sup>†</sup> [31]</b>	1.97	0.730	2.19	0.710	-	-
<b>F5-TTS (32 NFE) [32]</b>	1.56	0.741 (0.794)	1.83	0.647 (0.742)	8.67	0.713 (0.762)
<b>CosyVoice [34]</b>	3.63	0.723 (0.775)	4.29	0.609 (0.699)	11.75	0.709 (0.755)
<b>CosyVoice 2</b>	1.45	0.748 (0.806)	2.57	0.652 (0.736)	6.83	0.724 (0.776)
<b>CosyVoice 2-S</b>	1.45	0.753 (0.812)	2.38	0.654 (0.743)	8.08	0.732 (0.785)

[Demo](#)

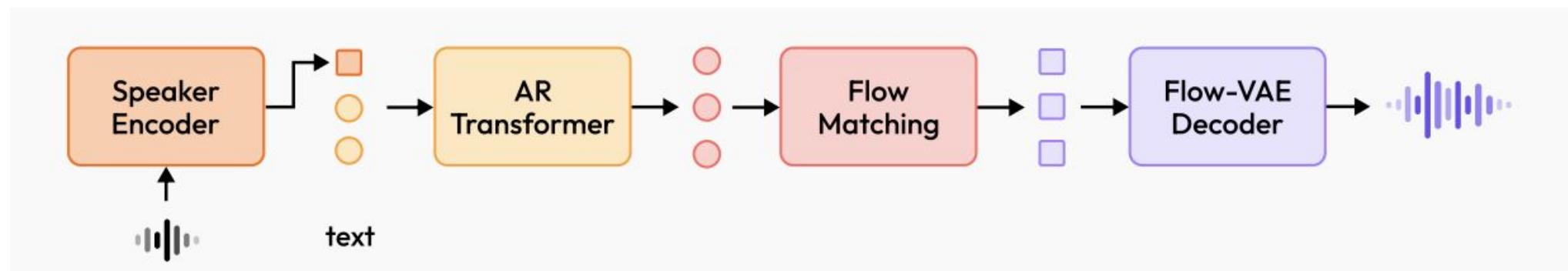


# Похожие модели

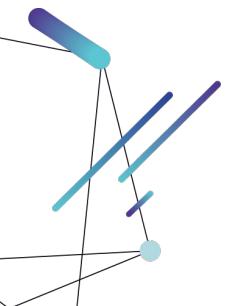
Seed-TTS



MinMax-Speech



# Основные тенденции и текущие задачи



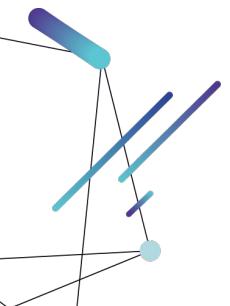
# Тенденции и актуальные задачи

## Тенденции

- Увеличение размера моделей и количества данных
- Multi-speaker, zero-shot TTS
- Multi-lingual
- Ускорение и поддержка streaming
- Дообучение на специфические задачи

## Актуальные задачи

- TTS для low-resourced языков
- Генерация длинных аудио, диалогов
- Добавление паралингвистических признаков для более естественной речи





**Спасибо  
за внимание!**

sadekova.t.r@gmail.com

**Садекова Таснима**

Huawei

