

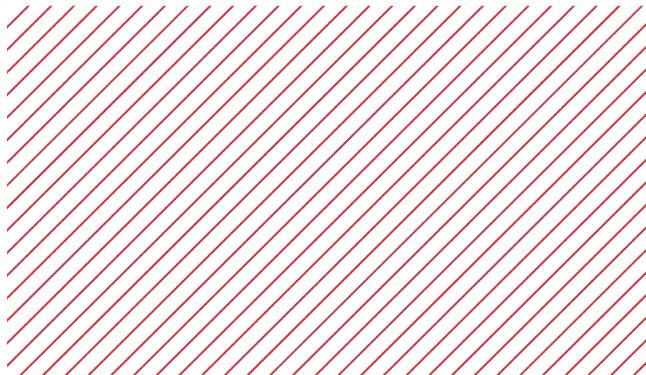
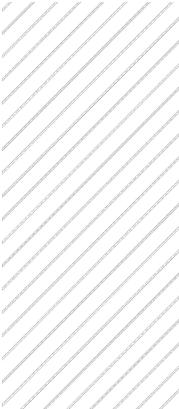
академия  
больших  
данных



mail.ru  
group

# Вокодеры в TTS

Свищев Алексей  
ЦРТ, ведущий научный сотрудник





# План занятия

---

1. Краткая история термина вокодер
2. Мотивация использования вокодеров в современных TTS
3. Последовательные архитектуры вокодеров
  - a. WaveNet
  - b. WaveRNN
  - c. LPCNet
  - d. Функции потерь для последовательных архитектур
4. Декомпозиция сигнала на субчастотные диапазоны (subbands подходы)
5. Об измерении качества и робастности
6. Тренды и перспективы

# Кратчайшая история термина

## Vocoder - voice encoder

1939 год Bell Labs VODER - Voice Operating Demonstrator

1961 год - HY-2 вокодер для защищенных систем связи



HY - 2 - VOCODER  
• First mobile digital device for converting speech into a form compatible for encryption by a general purpose key generator-QG-15 for transmission over narrowband radio and telephone.  
• Developed and improved to Donald Duck's voice  
• Vietnam War era

# Кратчайшая история термина

1959 год SIEMENS SYNTHESIZER - применение вокодеров в

музыке

Аппаратные и виртуальные

В системах связи (CELP, MELP)

В музыке

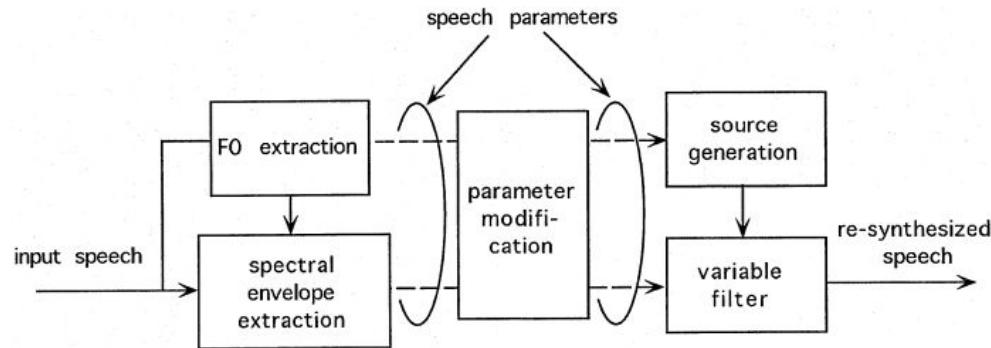


[https://upload.wikimedia.org/wikipedia/commons/thumb/5/5c/Siemens\\_Electronic\\_Music\\_Studio%2C\\_Deutsches\\_Museum\\_%28front1%29.jpg/1280px-Siemens\\_Electronic\\_Music\\_Studio%2C\\_Deutsches\\_Museum\\_%28front1%29.jpg](https://upload.wikimedia.org/wikipedia/commons/thumb/5/5c/Siemens_Electronic_Music_Studio%2C_Deutsches_Museum_%28front1%29.jpg/1280px-Siemens_Electronic_Music_Studio%2C_Deutsches_Museum_%28front1%29.jpg)  
<https://pop-music.ru/upload/medialibrary/6f5/6f52614a6173c425f20f445b40b226c6.jpg>

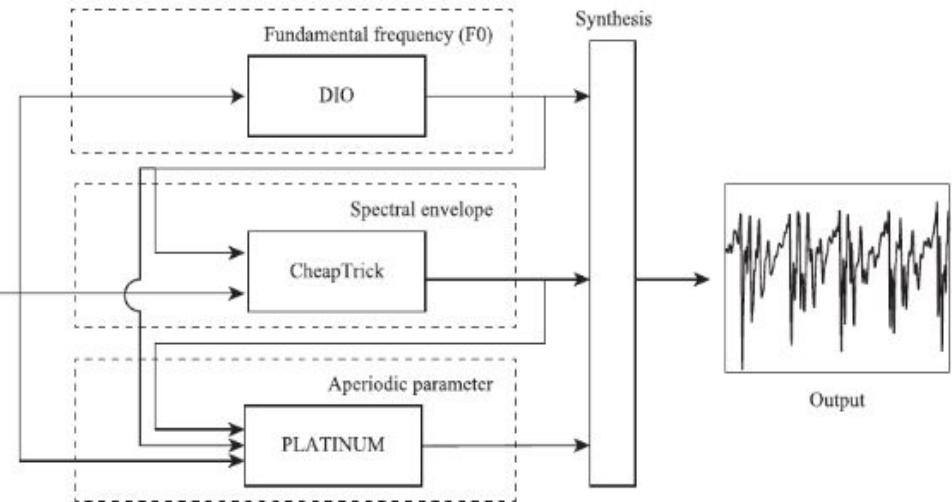
# Вокодеры в TTS до нейросетевого бума

1999 год

STRAIGHT (Speech Transformation and Representation using Adaptive Interpolation of weiGHTed spectrum)



2016 год  
WORLD



Morise M., Yokomori F., Ozawa K. WORLD: a vocoder-based high-quality speech synthesis system for real-time applications //IEICE TRANSACTIONS on Information and Systems. – 2016. – Т. 99. – №. 7. – С. 1877-1884.

Kawahara H., Masuda-Katsuse I., De Cheveigne A. Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds //Speech communication. – 1999. – Т. 27. – №. 3-4. – С. 187-207.

# Зачем вокодеры в современных TTS

Синтез речи — в широком смысле — восстановление формы речевого сигнала по его параметрам. В узком смысле — формирование речевого сигнала по печатному тексту.



1 к 1500

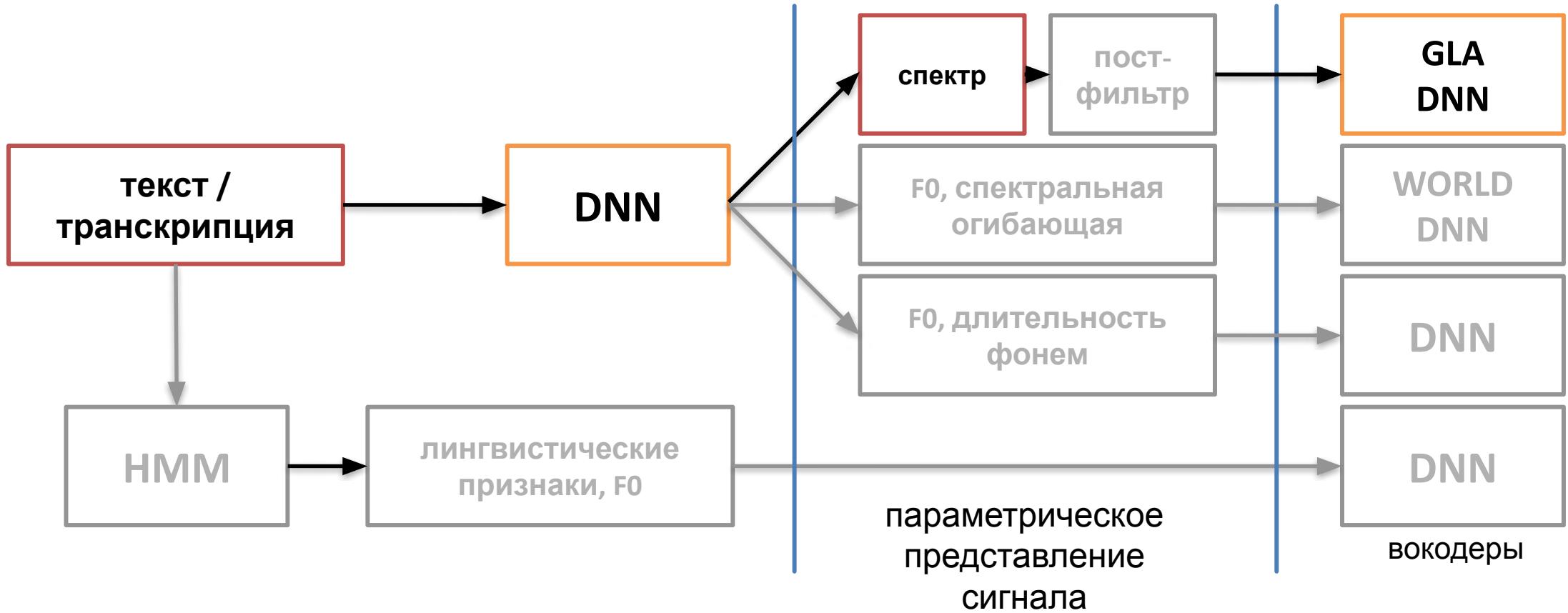
157 символов  
314 байт



PCM  
pulse code modulation

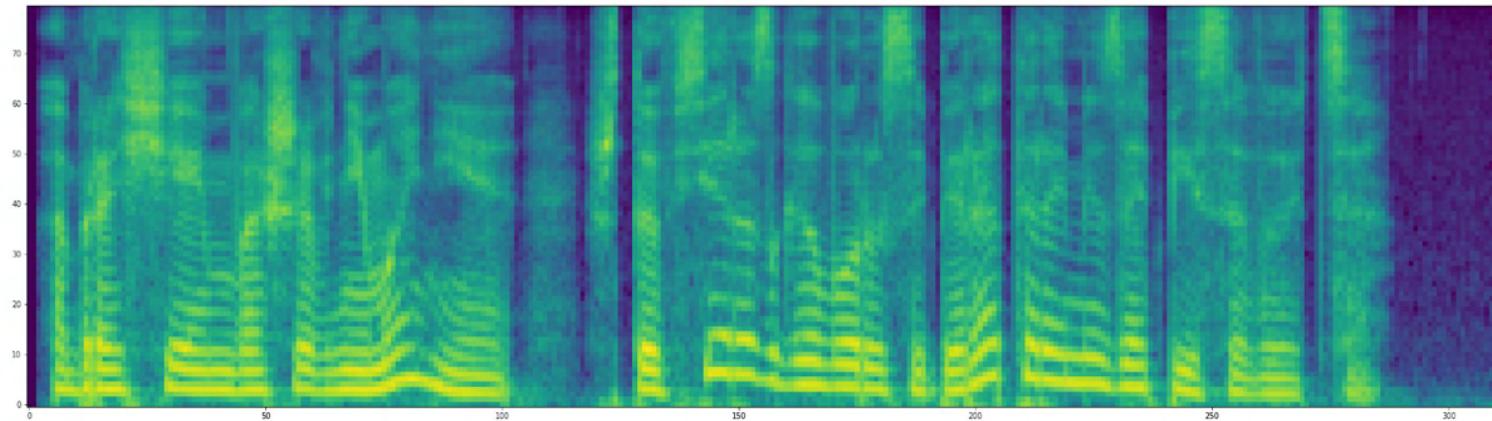
236 800 значений int16  
473 600 байт

# Зачем вокодеры в современных TTS

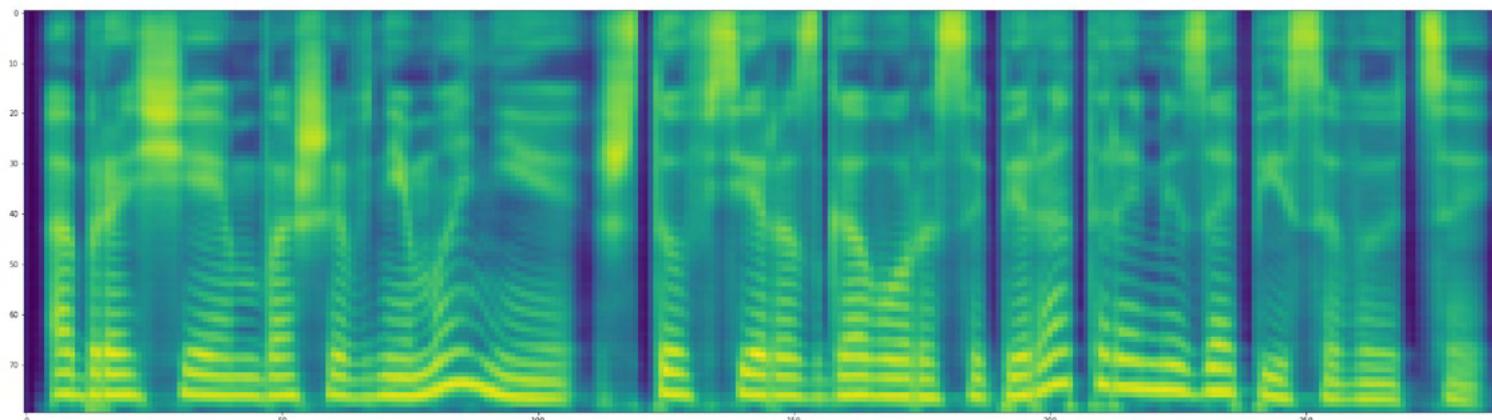


# Зачем вокодеры в современных TTS

Настоящая мел-спектрограмма



Синтезированная мел-спектрограмма



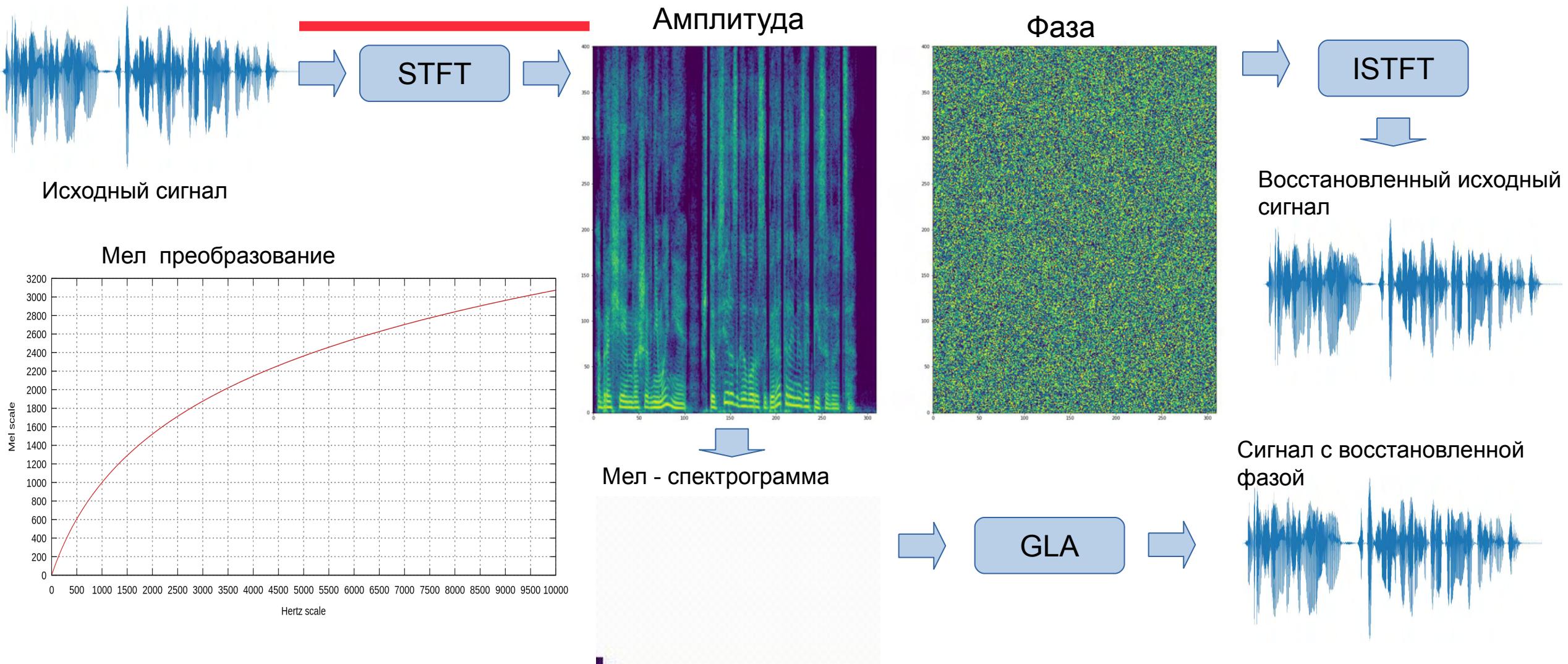


# Зачем вокодеры в современных TTS

---

- Декомпозиция сложной разнородной задачи на специфицированные
  - Разный масштаб коррелированности признаков
- Уменьшение степени неопределенности задачи для акустической модели
  - Неопределенность фазы
  - Шкалирование амплитуды
- Упрощение процесса обучения акустической модели
  - Разная скорость сходимости компонент
  - Разные наборы функций потерь
  - Может быть дополнительный препроцессинг признаков (аугментации)
- Коррекция локальных ошибок в акустических признаках
- Уточнение информации о высокочастотных составляющих акустических признаков

# Синтез без вокодера





# Нейросетевые вокодеры для TTS

---

- По количеству поддерживаемых дикторов:
  - singlespeaker
  - multispeaker
  - universal
- По типу оборудования и сложности вычислений:
  - высококачественные (TPU, GPU серверы),
  - малоресурсные (CPU серверы)
  - встроенные системы)
- По способу генерации:
  - последовательные (autoregressive)
  - параллельные (non-autoregressive)

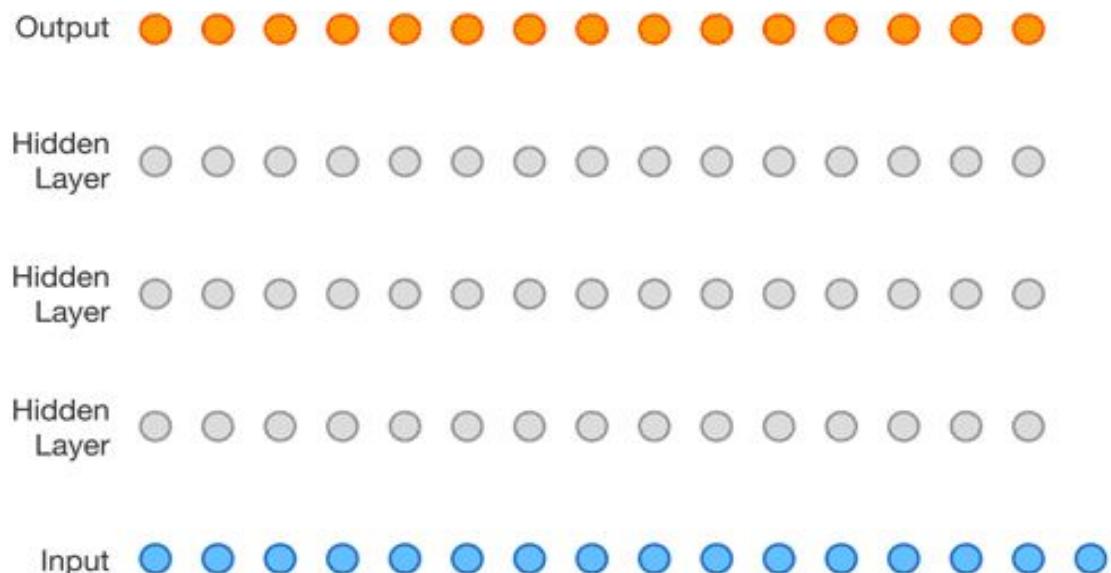
# Wavenet

$$p(\mathbf{x} \mid \mathbf{h}) = \prod_{t=1}^T p(x_t \mid x_1, \dots, x_{t-1}, \mathbf{h}).$$

Авторегрессия, последовательная генерация на основе предыдущих отчетов и акустических признаков.

Первая версия 16 кГц, 8 бит u-law, cat.  
Почти полноценная TTS обусловленная на лингвистические признаки и логарифм F0.

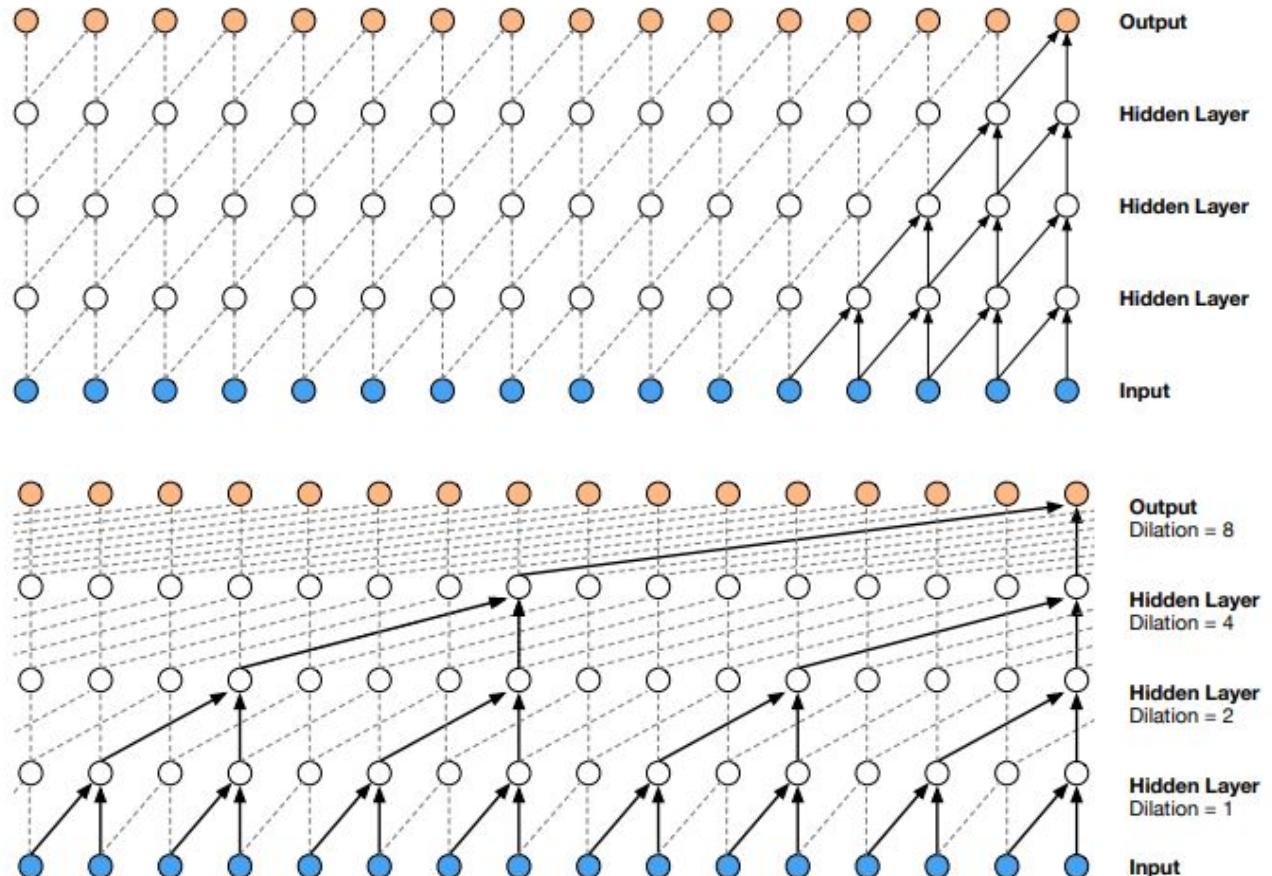
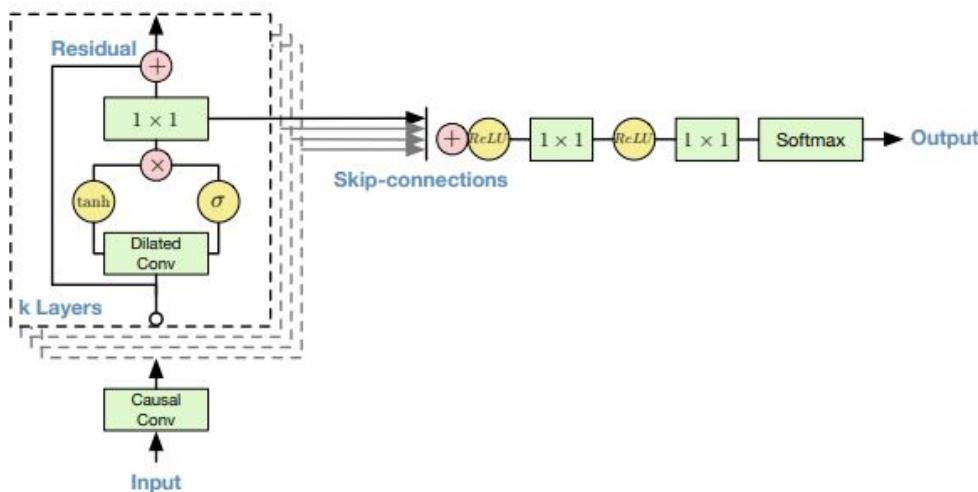
Вторая версия в связке с Tacotron 2.  
24 кГц, 16 бит, MoL  
Локально обусловлена на мел-спектrogramму.



# Wavenet

## Основные идеи

- Параллельное обучение, но последовательный синтез
- Dilated convolution (расширяющиеся, расширенные свертки) вместо causal convolutions для расширения рецептивного поля
- Residual connections для увеличения глубины сети
- Skip-connections для формирования прогноза
- Gated activation unit (из PixelCNN) обусловленный на локальные признаки
- Стэк блоков с циклическим dilation

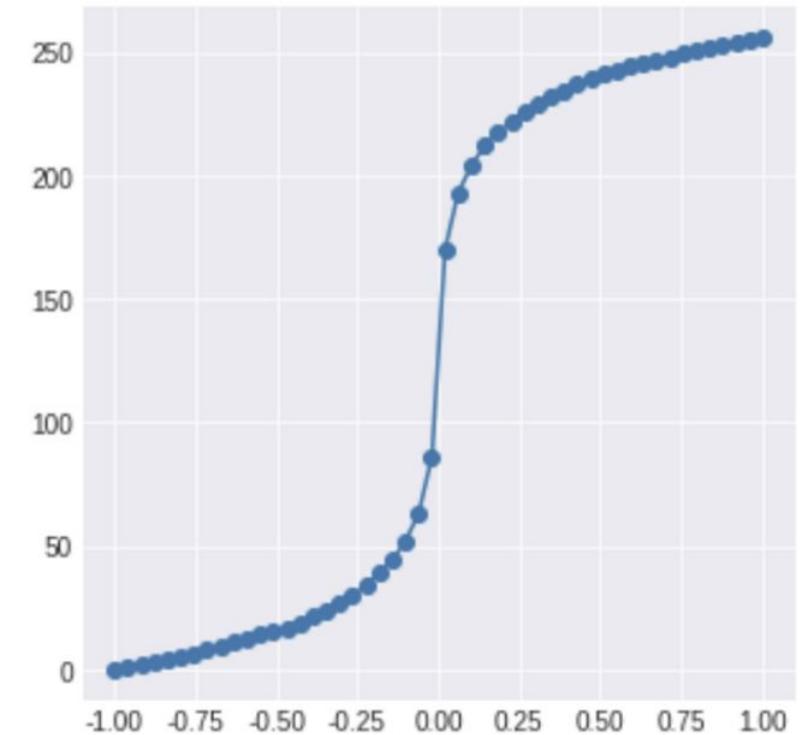
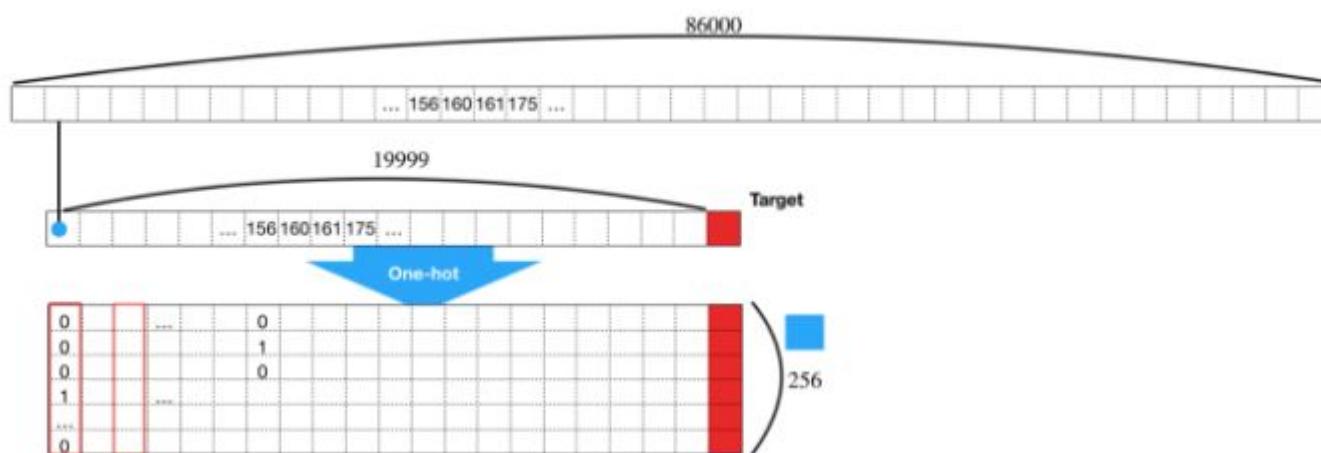


# Wavenet

## Как на самом деле это работает?

Как формируется минибатч

1. кодирование по мю-закону (лучше кодировать не сам сигнал, а разницы соседних отчетов, например, делать pre-emphasis)
2. сэмплирование участка фиксированной длины из примера (длительность определяется рецептивным полем сети)
3. one-hot кодирование
4. target - сдвинутая на один отчет вправо последовательность



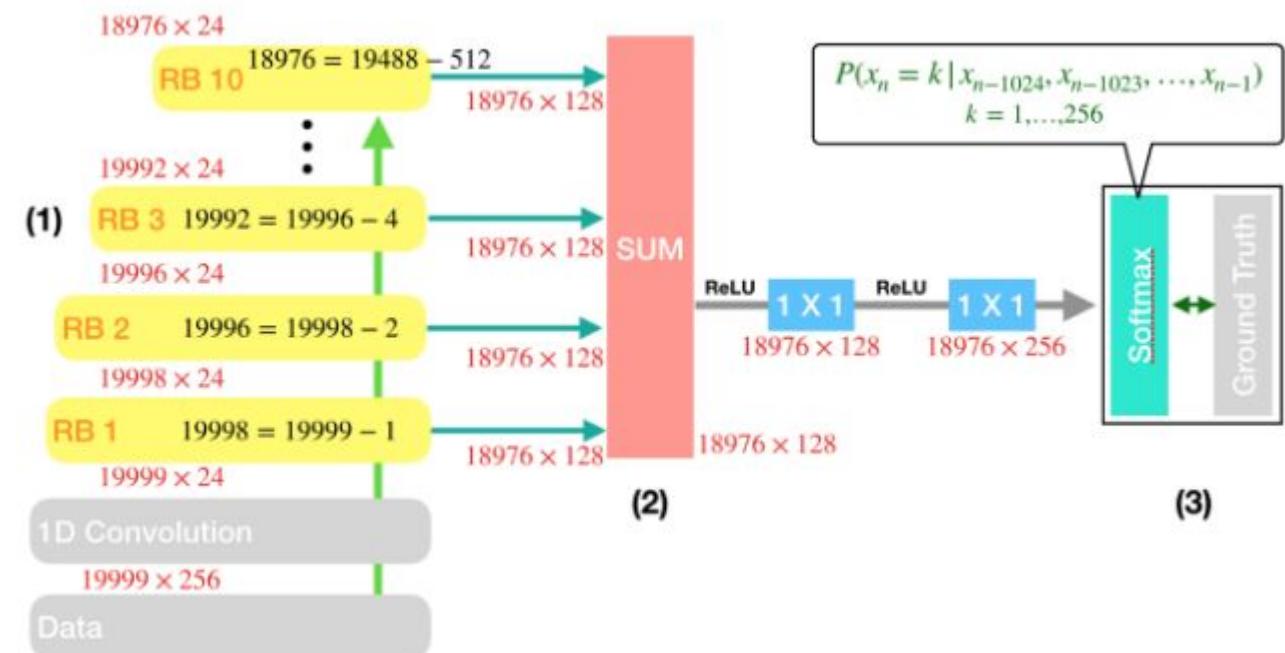
$$f(x_t) = \text{sign}(x_t) \frac{\ln(1 + \mu|X_t|)}{\ln(1 + \mu)}, -1 < x_t < 1, \mu = 256$$

# Wavenet

## Как на самом деле это работает?

Учится параллельно (сразу много прогнозов)  
Синтезирует последовательно (проходит все блоки для прогноза одного значения и так в цикле для каждого нового значения)

- (1) Одни цикл прохода по res-блокам (циклов в оригинале 3 по 10 блоков). Последовательно увеличивается receptive field. Каждый блок сверток немножко “отъедает” от последовательности слева (во время обучения) На инференсе блоки используют всю длину входящей последовательности.
- (1) Кроп и сумма skip-выходов из блоков (на инференсе будет  $1 \times 128$ )
- (2) Softmax и loss на обучении (softmax на инференсе будет только для 1го значения)



# Wavenet

## Скорость

Недостатки:

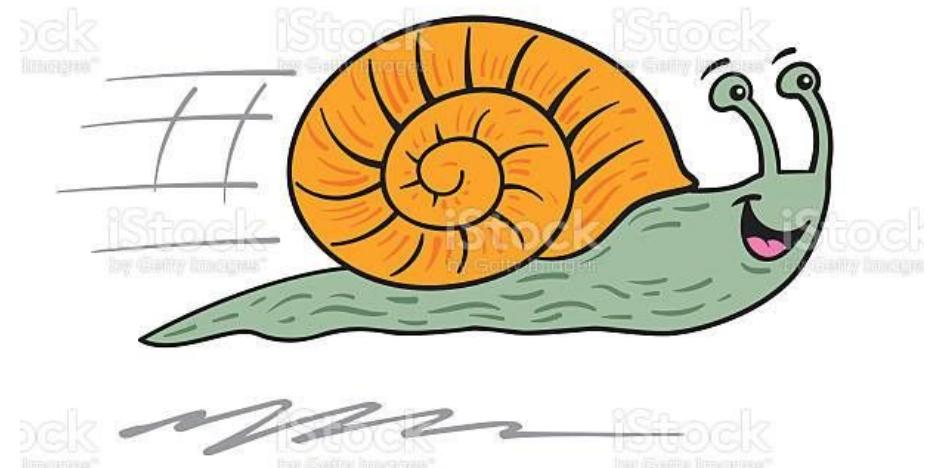
- много слоев -> много атомарных операций (и издержек на них)
- много дублирующих вычислений (для GPU)
- последовательный вывод по одному значению

Очень медленный инференс - 100-200 значений в секунду (нужно 16-24 тыс в секунду для синтеза в реальном времени)

Решения:

- монолитное CUDA - ядро для атомарного запуска шага цикла (для GPU)
- дистилляция в сеть с параллельным инференсом (Parallel WaveNet)
- кэширование повторяющихся вычислений (для CPU, Fast WaveNet)
- реализация оптимального С-кода для инференса
  - аппроксимация нелинейных функций (многочленами)
  - переход к fixed point вычислениям (квантование)
  - распараллеливание матричных вычислений
  - применением AVX инструкций
  - оптимизация работы с кэшем (минимизация кэш-промахов)

До 30-40 тыс сэмплов на ядро в секунду (на CPU)



618064312

# Wavenet

## Качество

MOS (Mean Opinion Score) оценку проводили на 100 примерах, от 8 независимых оценок на пример для каждой модели.

Все оценки независимые (не парные сравнения)

Оценка от 1 до 5 с шагом 0.5

Все модели обучены на одних и тех же данных (один диктор female 24.6 часов профессионального английского)

(внутренний датасет, не доступен)

- Гибрид с Tacotron 2 лучше чистого WaveNet на лингвистике (и робастнее)
- Учить лучше на синтезированных мел-спектrogramмах
- Для синтеза из мел-спектра не нужно большое рецептивное поле

System	MOS
Parametric	$3.492 \pm 0.096$
Tacotron (Griffin-Lim)	$4.001 \pm 0.087$
Concatenative	$4.166 \pm 0.091$
WaveNet (Linguistic)	$4.341 \pm 0.051$
Ground truth	$4.582 \pm 0.053$
Tacotron 2 (this paper)	<b><math>4.526 \pm 0.066</math></b>

System	MOS
Tacotron 2 (Linear + G-L)	$3.944 \pm 0.091$
Tacotron 2 (Linear + WaveNet)	$4.510 \pm 0.054$
Tacotron 2 (Mel + WaveNet)	<b><math>4.526 \pm 0.066</math></b>

Training	Synthesis	
	Predicted	Ground truth
Predicted	$4.526 \pm 0.066$	$4.449 \pm 0.060$
Ground truth	$4.362 \pm 0.066$	$4.522 \pm 0.055$

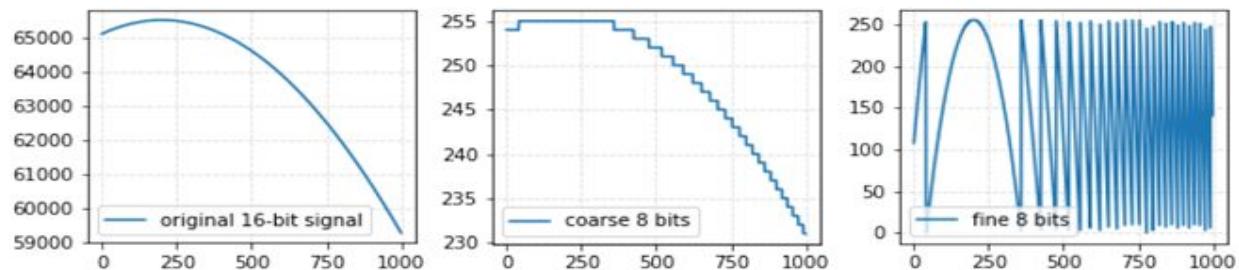
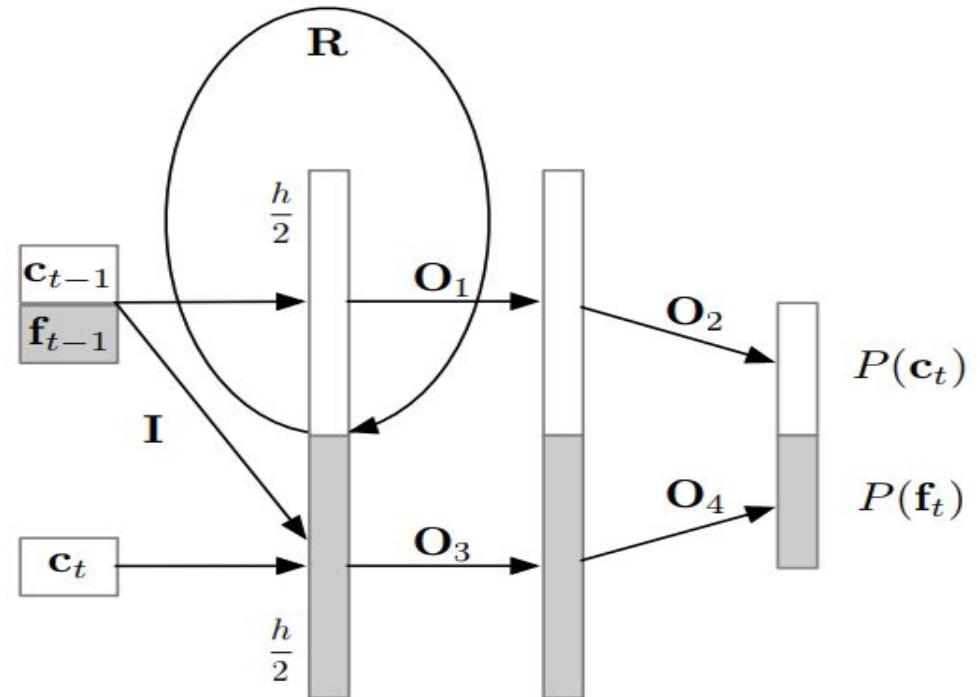
Total layers	Num cycles	Dilation cycle size	Receptive field (samples / ms)	MOS
30	3	10	6,139 / 255.8	$4.526 \pm 0.066$
24	4	6	505 / 21.0	$4.547 \pm 0.056$
12	2	6	253 / 10.5	$4.481 \pm 0.059$
30	30	1	61 / 2.5	$3.930 \pm 0.076$

# WaveRNN

## ОСНОВНЫЕ ИДЕИ

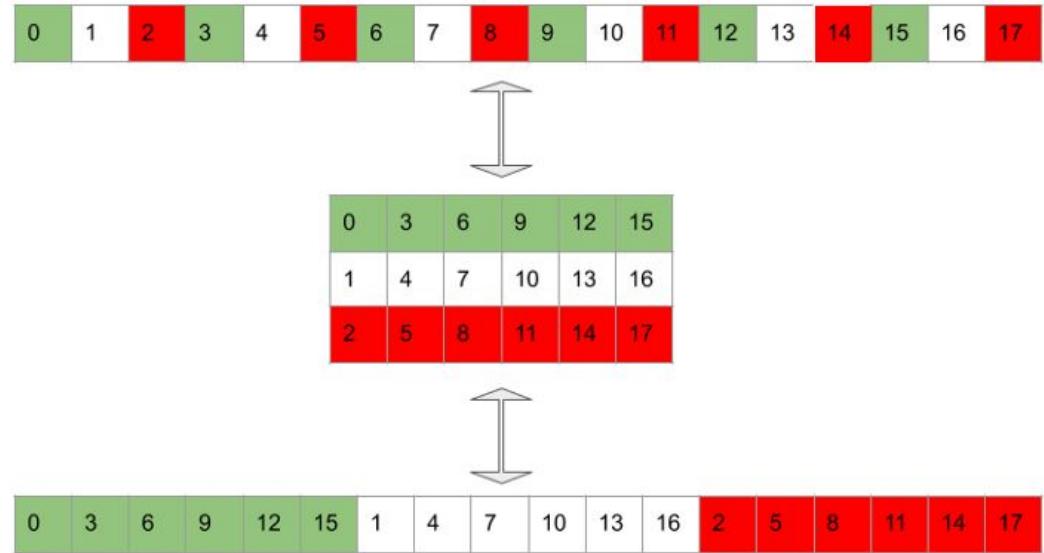
Стремление уменьшить размер сети до одной GRU ячейки  
Уменьшение количества слоев и соответственно издержек  
Декомпозиция прогнозирования значений (нижние и верхние 8 бит 16-битного значения сигнала)  $256 + 256$  вместо  $256 \times 256$  значений  
Маскирование gate матриц в GRU  
Глубокая спарсификация (разрежение весов) слоев  
Subscale или батч вывод  
Эффективная реализация CUDA - ядра

$$\begin{aligned}\mathbf{x}_t &= [\mathbf{c}_{t-1}, \mathbf{f}_{t-1}, \mathbf{c}_t] \\ \mathbf{u}_t &= \sigma(\mathbf{R}_u \mathbf{h}_{t-1} + \mathbf{I}_u^* \mathbf{x}_t) \\ \mathbf{r}_t &= \sigma(\mathbf{R}_r \mathbf{h}_{t-1} + \mathbf{I}_r^* \mathbf{x}_t) \\ \mathbf{e}_t &= \tau(\mathbf{r}_t \circ (\mathbf{R}_e \mathbf{h}_{t-1}) + \mathbf{I}_e^* \mathbf{x}_t) \\ \mathbf{h}_t &= \mathbf{u}_t \circ \mathbf{h}_{t-1} + (1 - \mathbf{u}_t) \circ \mathbf{e}_t \\ \mathbf{y}_c, \mathbf{y}_f &= \text{split}(\mathbf{h}_t) \\ P(\mathbf{c}_t) &= \text{softmax}(\mathbf{O}_2 \text{relu}(\mathbf{O}_1 \mathbf{y}_c)) \\ P(\mathbf{f}_t) &= \text{softmax}(\mathbf{O}_4 \text{relu}(\mathbf{O}_3 \mathbf{y}_f))\end{aligned}$$



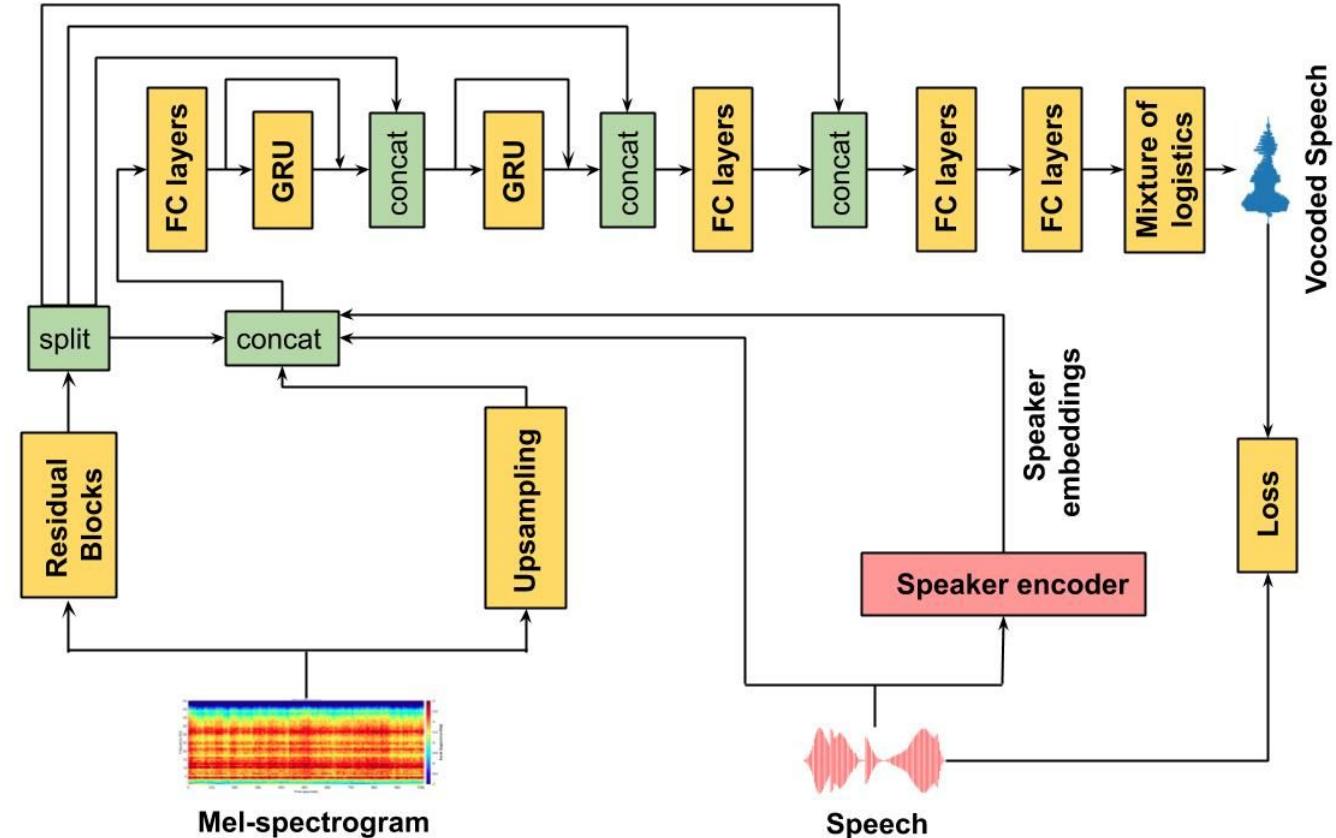
# WaveRNN subscale подход

- Схема параллельного сэмплирования
- Актуальна для большей утилизации GPU или использования нескольких ядер на CPU
- Немного снижает качество синтеза
- Требует иной схемы обучения и изменений в архитектуре сети



# WaveRNN в открытом доступе

- Оригинальная WaveRNN не была выложена
- Оригинальная WaveRNN опирается на лингвистические признаки и F0 (почти полноценная TTS)
- Энтузиасты не смогли повторить подход
- WaveRNN в открытом доступе опирается на мел-спектрограмму
- Аналог использует 2 GRU ячейки
- Не используется маскирование слоев GRU
- Вместо subscale предложен batched синтез (совместимо с subscale)
- Длина обучающей последовательность 1000-2000 значений



<https://github.com/dipjyoti92/SC-WaveRNN>

<https://github.com/fatchord/WaveRNN>

# WaveRNN

## Производительность и качество

Качество WaveRNN немного уступает WaveNet

Очень высокий уровень спарсификации на большом размере ячейки (2048 не ухудшает качества, даже улучшает)

Subscale подход позволяет добиться высокой скорости синтеза

Скорость синтеза оригинальной модели до 96 тыс сэмплов в секунду на GPU, до 160 тыс в секунду в subscale.

Открытые модели (Tacotron 2 + WaveRNN) не уступают качеством синтеза оригинальной.

Скорость синтеза на GPU около 2 тыс сэмплов в секунду

(большие издержки фреймворков).

В батч режиме до 100 тыс сэмплов в секунду (может ухудшать качество).

MODEL	NLL	MOS
WAVENET	5.29	$4.51 \pm 0.08$
WAVERNN 224	5.67	$3.73 \pm 0.09$
WAVERNN 384	5.56	$4.23 \pm 0.09$
WAVERNN 896	5.42	$4.37 \pm 0.07$
WAVERNN 2048	5.33	$4.46 \pm 0.07$
SPARSE WR MOBILE	5.52	$4.33 \pm 0.08$
SPARSE WR 224 / 1536@97.8%	5.48	$4.39 \pm 0.07$
SPARSE WR 384 / 2048@96.4%	5.42	$4.48 \pm 0.07$
SUBSCALE WR 1024 (16×)	5.52	$4.30 \pm 0.08$
SUBSCALE WR 1024 (8×)	5.46	$4.39 \pm 0.06$
FUSED SUBSCALE WR 896 (2×)	5.45	$4.31 \pm 0.08$

SIZE	SPARSITY %	TYPE	PLATFORM	SAMPLES/SEC
512	95%	4×4	SD 835	29,100
512	95%	4×4	SD 808	19,800
512	95%	16×1	SD 835	31,400
512	95%	16×1	SD 808	21,600

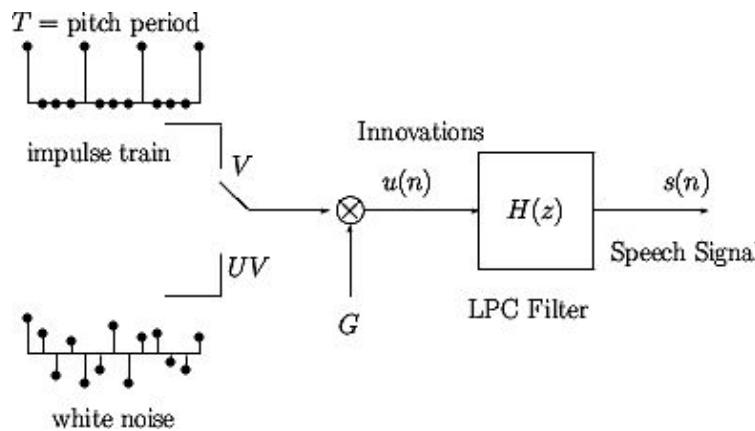
# LPCNet

## основные идеи

Линейное предсказание  
сигнала

$$s[n] = G \cdot e[n] + \sum_{k=1}^p h_k \cdot s[n-k]$$

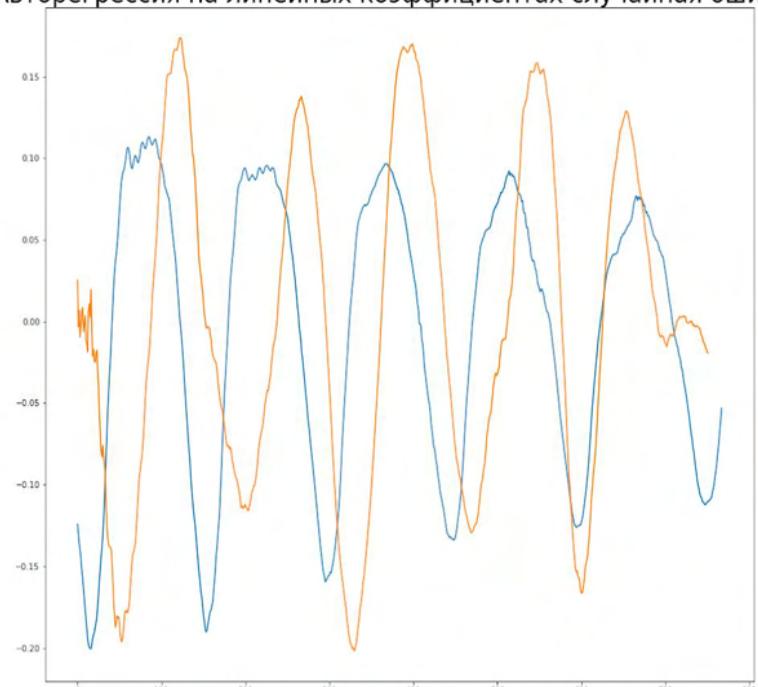
Модель источник - фильтр



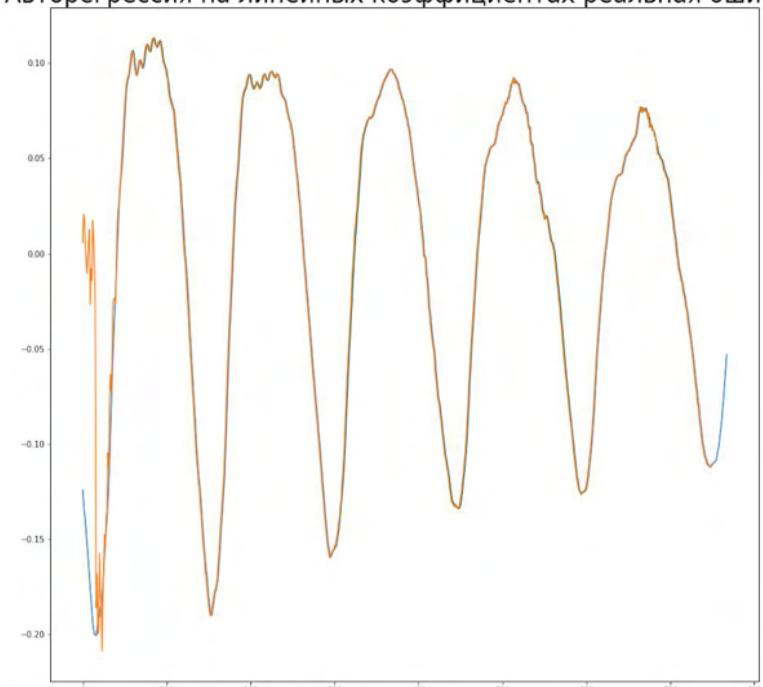
Более естественная  
декомпозиция задачи

При хорошей оценке ошибки можно точно  
восстановить исходный сигнал

Авторегрессия на линейных коэффициентах случайная ошибка

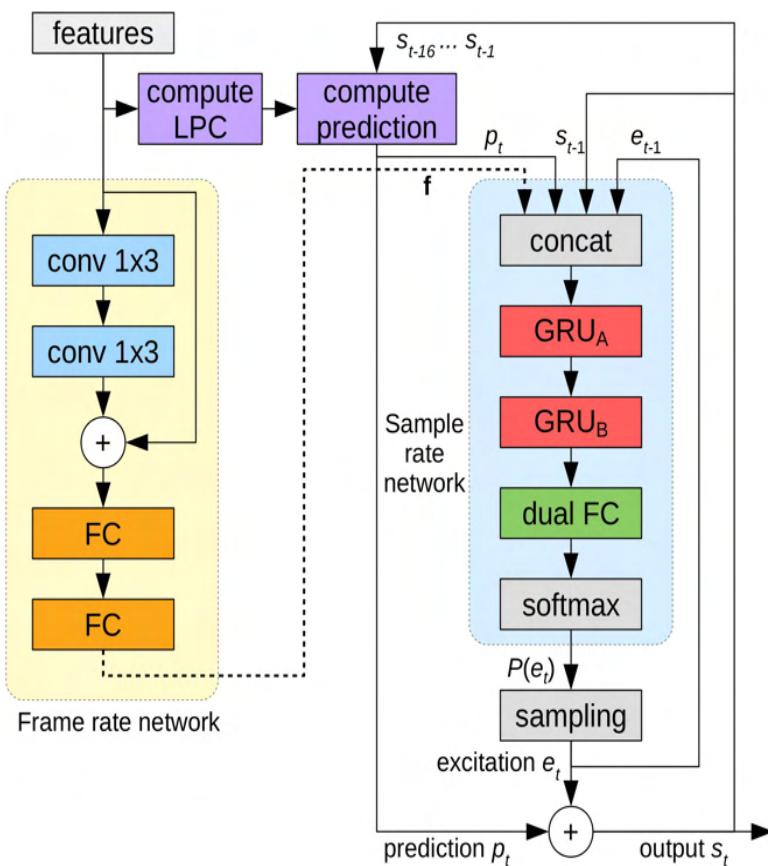


Авторегрессия на линейных коэффициентах реальная ошибка



# LPCNet основные идеи

Задача оценки распределения значения отсчета сигнала сводится к задаче оценки распределения ошибки линейного предсказания.



GRU\_A — концентрирует в себе до 90 % времени вычислений.

Размер ячейки в базовой версии модели 384.

Размер ячейки в базовой версии WaveRNN с аналогичным качеством 868.

3 умножения вектора на матрицу + несколько поэлементных операций для синтеза каждого 8 битного отсчета (мю кодирование)

$$z_t = \sigma(v_t^{z,s} + v_t^{z,p} + v_t^{z,e} + g_z + U_z h_{t-1} + b_z)$$

$$r_t = \sigma(v_t^{r,s} + v_t^{r,p} + v_t^{r,e} + g_r + U_r h_{t-1} + b_r)$$

$$h_t = z_t * h_{t-1} + (1 - z_t) * \sigma(v_t^{h,s} + v_t^{h,p} + v_t^{h,e} + g_h + U_h (r_t * h_{t-1}) + b_h)$$

$$P(e_t) = softmax(dual\_fc(GRU_b(h_t)))$$

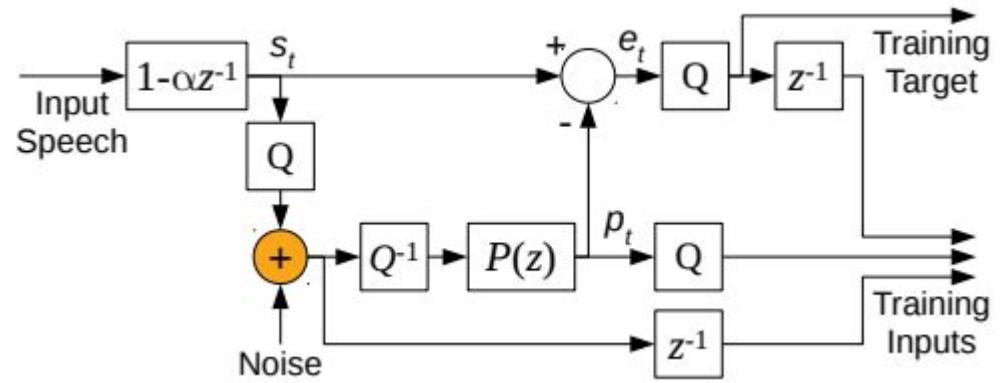
# LPCNet пайплайн

Обучение:

1. pre-emphasis - уменьшает шум квантования
2. мю-кодирование - 8 бит, 256 значений для прогноза
3. построение признаков
4. получение LPC-коэффициентов для фреймов из линейной магнитуды
5. LPC разложение сигнала с инъекцией шума
6. прогнозирование признаков кадра
7. прогнозирование сигнала ошибки
8. loss

Инференс:

1. получение LPC-коэффициентов для фреймов из линейной магнитуды
2. прогнозирование признаков кадра
3. прогнозирование сигнала ошибки
4. сэмплирование из распределения
5. полное восстановление сигнала
6. обратный мю - закон
7. inverse preemphasis



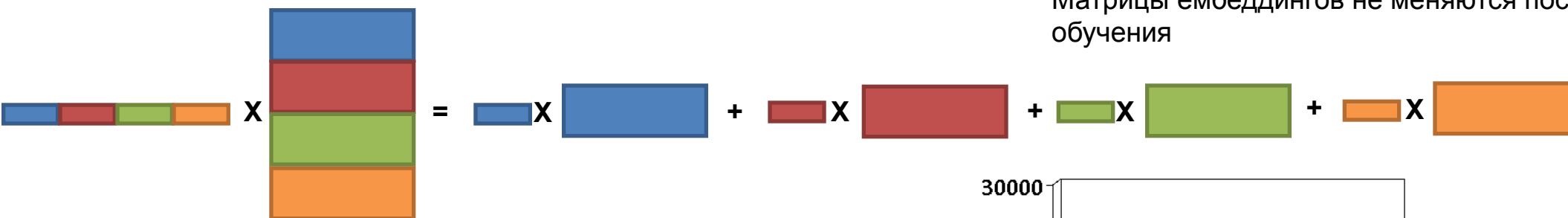
# LPCNet фолдирование

## Формула GRU

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

$$h_t = z_t * h_{t-1} + (1 - z_t) * \sigma(W_h x_t + U_h (r_t * h_{t-1}) + b_h)$$



$$W_z x_t = W_z^s x_t^s + W_z^p x_t^p + W_z^e x_t^e + W_z^f x_f = v_t^{z,s} + v_t^{z,p} + v_t^{z,e} + g_z$$

Аналогично другие

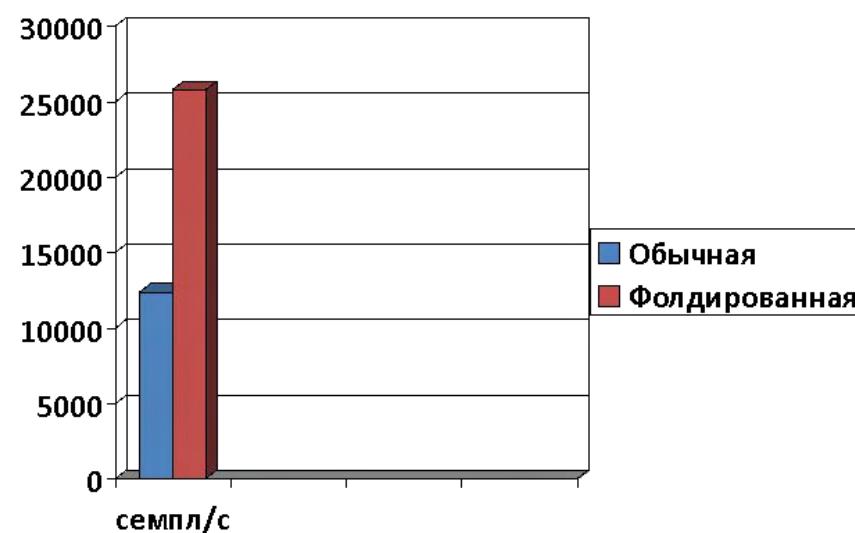
**Одна большая операция матричного умножения заменяется на 4 суммы векторов**

Вектор входных значений в LPCNet

эмбеддинг предыдущего значения сигнала	эмбеддинг LPC прогноза текущего значения сигнала	эмбеддинг предыдущего значения ошибки сигнала	вектор признаков фрейма
--	--	---	-------------------------

Матрицы ембеддингов не меняются после обучения

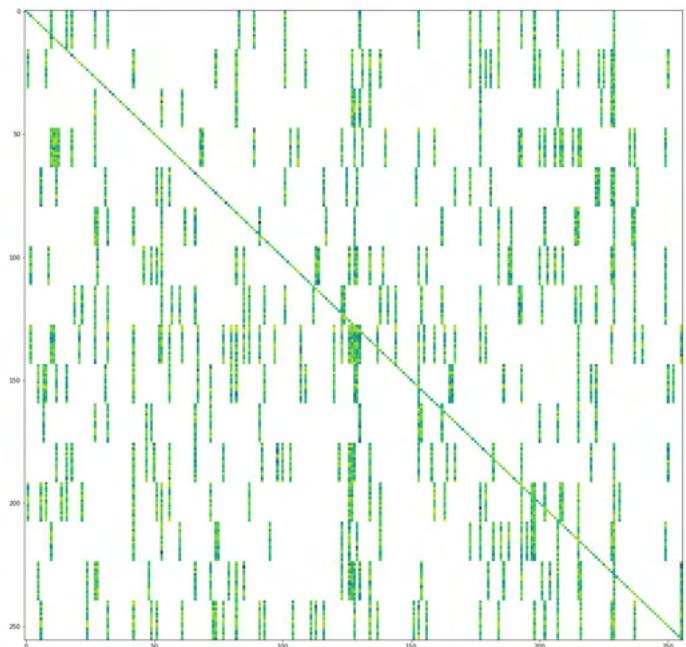
Вычисляется раз на фрейм



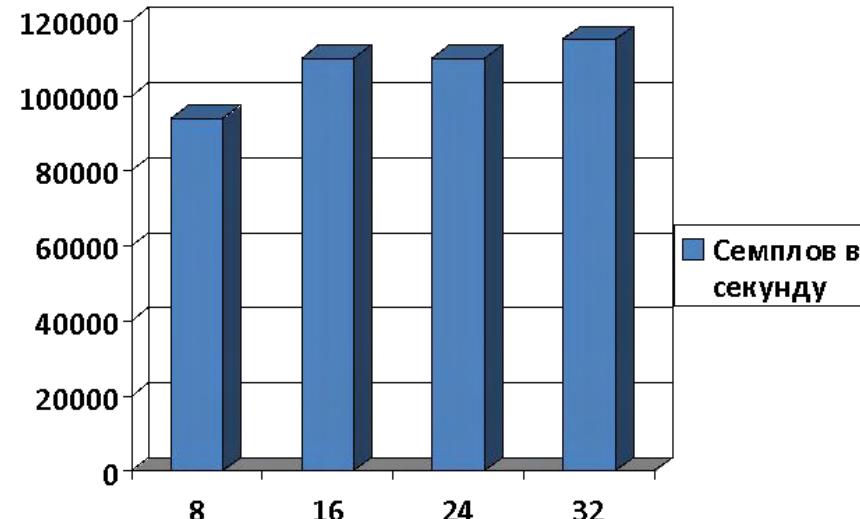
# LPCNet

## блочная спарсификация

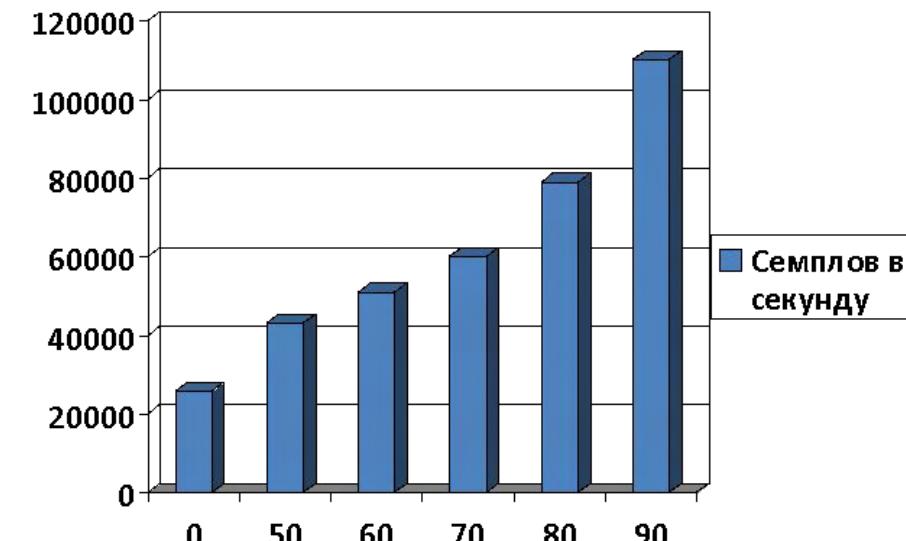
- В LPCNet спарсифицируется матрица U (рекуррентные веса)
- Спарсификация идет во время обучения
- Блоки размером  $1 \times 8, 1 \times 16, 1 \times 24, 1 \times 32$
- Применяется специальный алгоритм умножения



Спарсификация 90 %  
Блоки разного размера



Блок размером 16  
Разный уровень спарсификации



# LPCNet производительность и качество

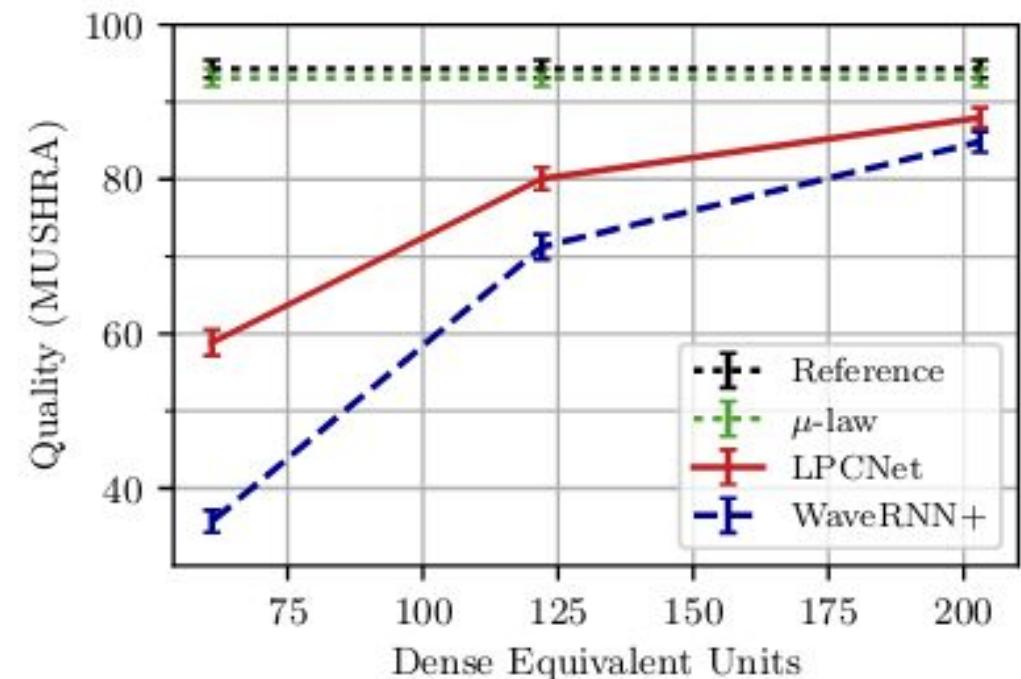
LPCNet превосходит WaveRNN аналогичной производительностью по качеству

Пример хорошего дизайна архитектуры, ориентированного на inference

Изначально задумывалась как кодек для систем связи (для Opus)

Есть реализация в качестве кодека для систем связи 1.6 кбит в секунду

Пиковая производительность на одном ядре 100 тыс сэмплов в секунду



# ФУНКЦИИ ПОТЕРЬ ДЛЯ ПОСЛЕДОВАТЕЛЬНЫХ архитектур

---

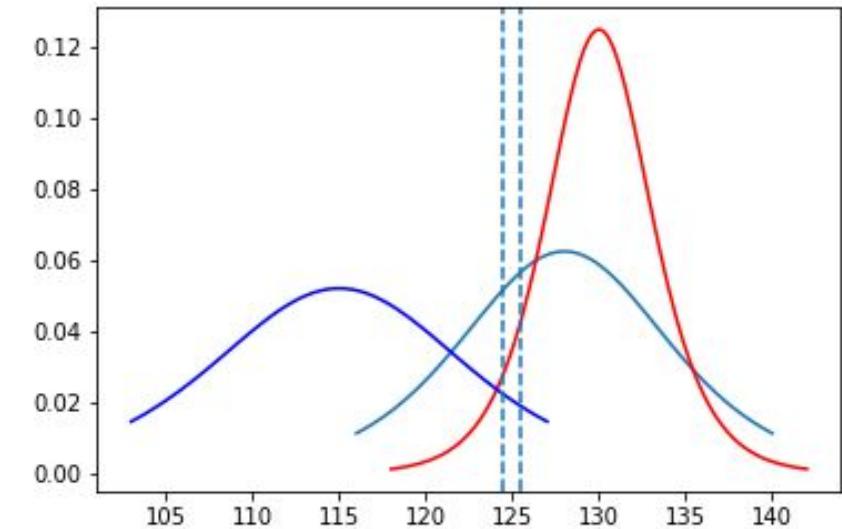
Negative log likelihood (NLL) - когда прогнозируем категориальное распределение

Gaussian - когда значение в целом унимодально и нормально распределено

Mixture of Logistics (MoL) - когда значение может быть мультимодально

Для MoL и Gaussian моделью прогнозируются параметры распределения

Для MoL есть аналитическое выражение функции распределения



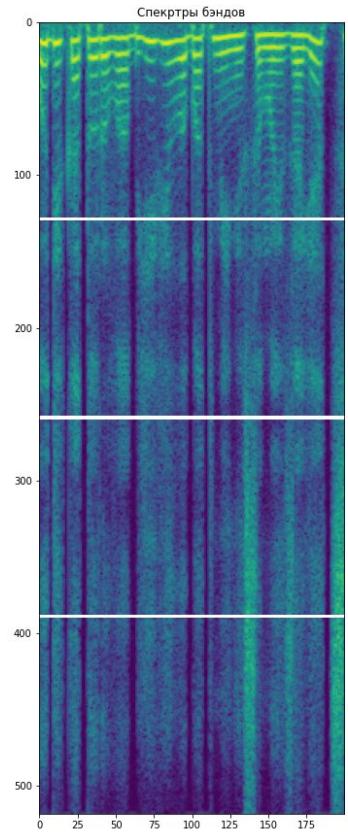
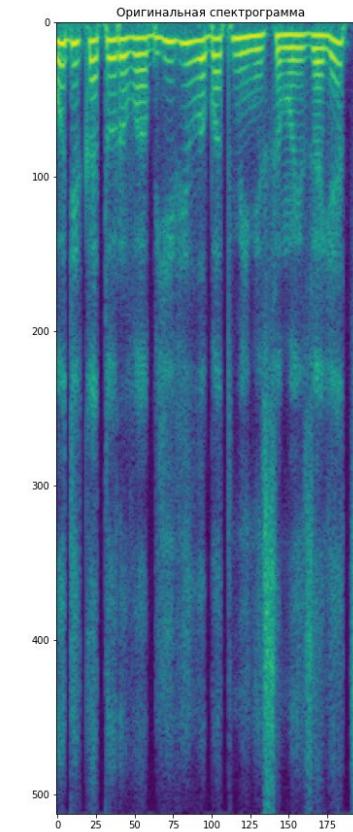
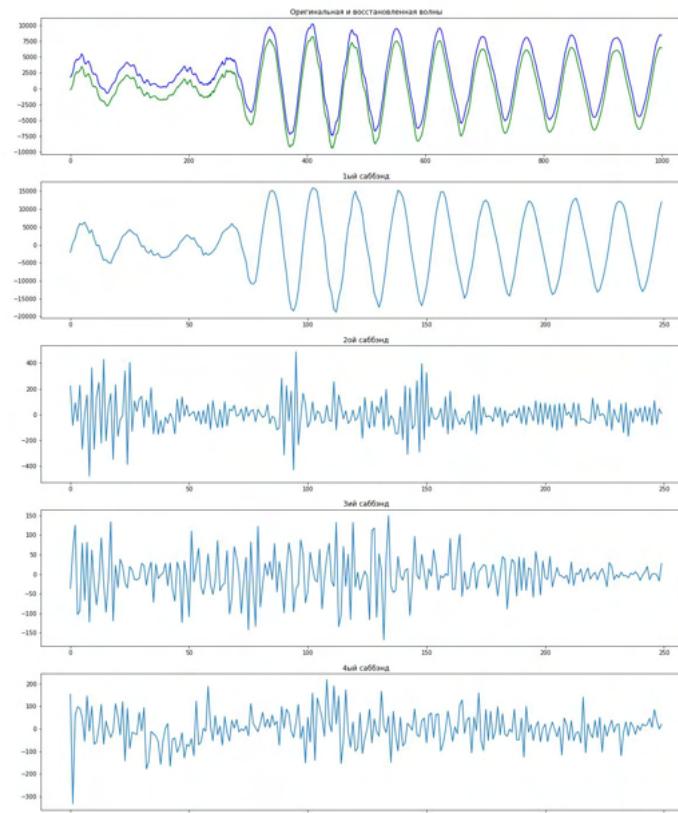
# Декомпозиция сигнала

Сигнал можно разбить на последовательности так, что каждой подпоследовательности будет соответствовать своя частотная область.

ЧД сигнала будет понижаться кратно числу разбиений

Существуют способы разбиение с близкой к идеальной реконструкции

Разбитый сигнал можно прогнозировать параллельно или одновременно по несколько отсчетов за шаг



# Декомпозиция сигнала

СМФ — сопряженные зеркальные фильтры.

Сигнал подается параллельно на два фильтра  $G_0(z)$  и  $G_1(z)$  (предлагается использовать вейвлет Дебеши 40);

сигнал может быть децимирован вдвое;

для восстановления сигнала необходимо подать сигнал на интерполятор, который добавит нули между отсчетами;

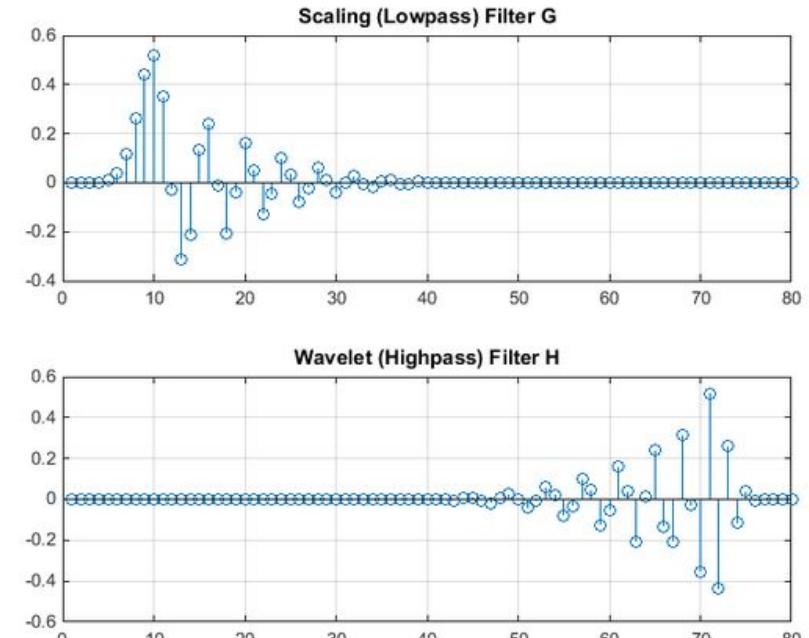
после этого сигнал подается на фильтры  $H_0(z)$  и  $H_1(z)$ ;

после чего сигналы можно поточечно складывать.

Условие идеальной реконструкции:

$$H_0(z) = G_1(z)$$

$$H_1(z) = -G_0(z)$$



# Измерение качества и робастность

---

Оценки с участием людей:

- MOS (Mean opinion score)
- Side-by-side evaluation
- MUSHRA (MULTiple Stimuli with Hidden Reference and Anchor)

Side-by-side evaluation для сравнения двух систем между собой (или нескольких попарно).

MOS независимые оценки разных моделей

MUSHRA аналогична MOS, но требует синтеза одинаковых образцов разными системами, что дает возможность делать меньше оценок для статистически значимого результата

Для вокодеров измерение может происходить на задаче ресинтеза, когда признаки получены из оригинальных примеров, и на задаче синтеза, когда признаки синтезированы какой-либо системой

Качество ресинтеза всегда выше.

MOS оценки в разных статьях часто не сравнимы.

Робастность - способность вокодера работать в новом домене, с новыми дикторами, на новых языках, с новыми TTS системами.

# Измерение качества и робастность

Table 4: *MOS for Speaker and language in/out domain experiments*

Vocoder Training Set					
	En_F	En_M	Ma_F	En_L	Lrg
Seen Speakers and Seen Language					
WN	<b>4.78±0.10</b>	<b>4.71±0.11</b>	4.63±0.12	<b>4.72±0.10</b>	<b>4.70±0.13</b>
WR	4.48±0.13	4.61±0.13	<b>4.66±0.11</b>	4.64±0.11	4.61±0.13
FF	3.87±0.17	4.29±0.15	4.45±0.10	3.28±0.19	3.58±0.17
PW	4.59±0.12	4.29±0.17	4.41±0.12	4.29±0.15	4.11±0.16
Unseen Speakers and Seen Language					
WN	2.27±0.14	2.86±0.17	3.27±0.16	<b>4.25±0.17</b>	<b>4.35±0.15</b>
WR	<b>2.60±0.14</b>	<b>2.89±0.15</b>	<b>3.54±0.14</b>	3.98±0.15	3.92±0.16
FF	1.76±0.15	2.21±0.14	2.94±0.13	2.99±0.18	3.13±0.21
PW	2.35±0.15	2.85±0.16	2.88±0.14	3.80±0.21	3.85±0.17
Unseen Speakers and Unseen Language					
WN	1.90±0.12	2.53±0.12	<b>3.85±0.15</b>	<b>4.33±0.15</b>	<b>4.33±0.17</b>
WR	<b>2.53±0.13</b>	<b>2.62±0.12</b>	3.30±0.15	4.30±0.16	4.16±0.17
FF	1.56±0.09	1.75±0.12	2.64±0.16	2.67±0.17	3.37±0.17
PW	2.17±0.11	2.54±0.12	2.49±0.13	3.79±0.20	3.97±0.19

Table 7: *MOS for text-to-speech (TTS) synthesis experiment*

Vocoder Training Set					
	LJ	En_F	En_L	Lrg	Cond
WN	4.10±0.19	2.59±0.24	3.54±0.20	3.66±0.21	<b>4.21±0.16</b>
WR	<b>4.16±0.18</b>	3.05±0.24	3.32±0.20	3.73±0.19	3.79±0.19
FF	2.75±0.27	2.16±0.29	2.50±0.27	2.28±0.28	2.86±0.30
PW	3.81±0.20	3.17±0.21	3.60±0.20	3.19±0.20	3.38±0.20
GT				4.54±0.16	

Table 1: *Overview of the training datasets.*

label	consist datasets	speakers	utterances	consist language
En_M	cmu_ma	1	1091	English
En_F	cmu_fe	1	1092	English
Ma_F	man_fe	1	8904	Mandarin
En_L	cmu_ma cmu_fe libri	560	35419	English
Lrg	cmu_ma cmu_fe libri bible	>600	38139	English French Japanese Korean Spanish Thai

Table 6: *MOS for Gender and language in/out domain experiments*

Model	Vocoder Training Set		
	En_M	En_F	Ma_F
Seen Gender and Seen Language			
WaveNet	2.41±0.23	3.47±0.24	3.57±0.20
WaveRNN	2.85±0.21	3.49±0.21	4.08±0.20
FFTNet	2.01±0.24	2.45±0.21	3.56±0.14
Parallel WaveGAN	2.68±0.22	3.47±0.20	3.34±0.17
Unseen Gender and Seen Language			
WaveNet	2.13±0.16	2.25±0.16	2.98±0.21
WaveRNN	2.36±0.20	2.29±0.15	3.01±0.20
FFTNet	1.52±0.15	1.97±0.20	2.34±0.15
Parallel WaveGAN	2.03±0.17	2.23±0.18	2.41±0.17
Seen Gender and Unseen Language			
WaveNet	1.92±0.16	3.05±0.23	4.10±0.22
WaveRNN	2.78±0.18	3.12±0.21	3.77±0.18
FFTNet	1.74±0.17	2.00±0.17	3.40±0.17
Parallel WaveGAN	2.29±0.19	2.92±0.22	2.92±0.21
Unseen Gender and Unseen Language			
WaveNet	1.88±0.16	2.01±0.16	3.59±0.20
WaveRNN	2.29±0.17	2.12±0.19	2.84±0.21
FFTNet	1.38±0.11	1.51±0.11	1.91±0.16
Parallel WaveGAN	2.06±0.16	2.17±0.15	2.05±0.17



# Тренды

---

- стремление построить универсальный вокодер
- стремление облегчить процесс адаптации к новым спикерам
- улучшение адаптивности вокодеров к новым данным
- параллелизм в вычислениях
- уменьшение вычислительной сложности
- гибридные подходы к генерации, применение аналитических методов обработки сигнала для упрощения задачи
- внедрение дифференцируемых подходов в обработке сигналов

# Список нейросетевых вокодеров

---

WaveNet	Oord A. et al. Wavenet: A generative model for raw audio //arXiv preprint arXiv:1609.03499. – 2016.
Parallel WaveNet	Oord A. et al. Parallel wavenet: Fast high-fidelity speech synthesis //International conference on machine learning. – PMLR, 2018. – С. 3918-3926.
FloWaveNet	Kim S. et al. FloWaveNet: A generative flow for raw audio //arXiv preprint arXiv:1811.02155. – 2018.
WaveRNN	Kalchbrenner N. et al. Efficient neural audio synthesis //International Conference on Machine Learning. – PMLR, 2018. – С. 2410-2419.
Multi-Band WaveRNN	Yu C. et al. Durian: Duration informed attention network for multimodal synthesis //arXiv preprint arXiv:1909.01700. – 2019.
LPCNet	Valin J. M., Skoglund J. LPCNet: Improving neural speech synthesis through linear prediction //ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). – IEEE, 2019. – С. 5891-5895.
Fast WaveNet	Paine T. L. et al. Fast wavenet generation algorithm //arXiv preprint arXiv:1611.09482. – 2016.

# Список нейросетевых вокодеров

---

FftNet	Jin Z. et al. FFTNet: A real-time speaker-dependent neural vocoder //2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). – IEEE, 2018. – С. 2251-2255.
SampleRNN	Mehri S. et al. SampleRNN: An unconditional end-to-end neural audio generation model //arXiv preprint arXiv:1612.07837. – 2016.
ClariNet	Ping W., Peng K., Chen J. Clarinet: Parallel wave generation in end-to-end text-to-speech //arXiv preprint arXiv:1807.07281. – 2018.
WaveGlow	Prenger R., Valle R., Catanzaro B. Waveglow: A flow-based generative network for speech synthesis //ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). – IEEE, 2019. – С. 3617-3621.
MelGAN	Kumar K. et al. Melgan: Generative adversarial networks for conditional waveform synthesis //arXiv preprint arXiv:1910.06711. – 2019.
GAN-TTS	Bińkowski M. et al. High fidelity speech synthesis with adversarial networks //arXiv preprint arXiv:1909.11646. – 2019.

# Список нейросетевых вокодеров

---

Parallel WaveGAN	Yamamoto R., Song E., Kim J. M. Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram //ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). – IEEE, 2020. – С. 6199-6203.
WaveVAE	Peng K. et al. Non-autoregressive neural text-to-speech //International Conference on Machine Learning. – PMLR, 2020. – С. 7586-7598.
WaveFlow	Ping W. et al. Waveflow: A compact flow-based model for raw audio //International Conference on Machine Learning. – PMLR, 2020. – С. 7706-7716.
WaveGrad	Chen N. et al. WaveGrad: Estimating gradients for waveform generation //arXiv preprint arXiv:2009.00713. – 2020.
DiffWave	Kong Z. et al. Diffwave: A versatile diffusion model for audio synthesis //arXiv preprint arXiv:2009.09761. – 2020.
Multi-Band MelGAN	Yang G. et al. Multi-band MelGAN: Faster waveform generation for high-quality text-to-speech //2021 IEEE Spoken Language Technology Workshop (SLT). – IEEE, 2021. – С. 492-498.
HifiGAN	Kong J., Kim J., Bae J. HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis //arXiv preprint arXiv:2010.05646. – 2020.