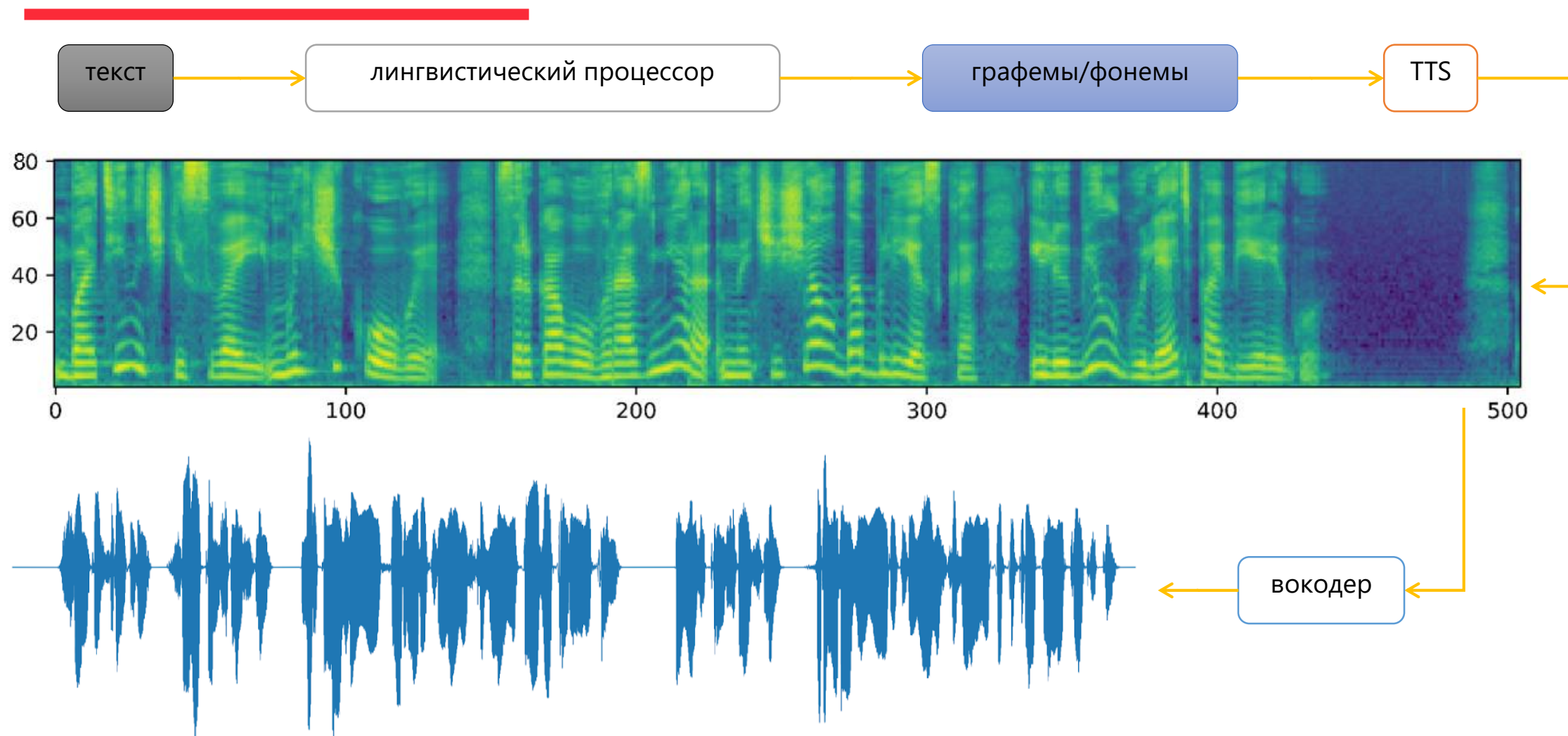




Text to Speech

Калиновский И.А.
Just AI, к.т.н.

Синтез речи на основе DNN



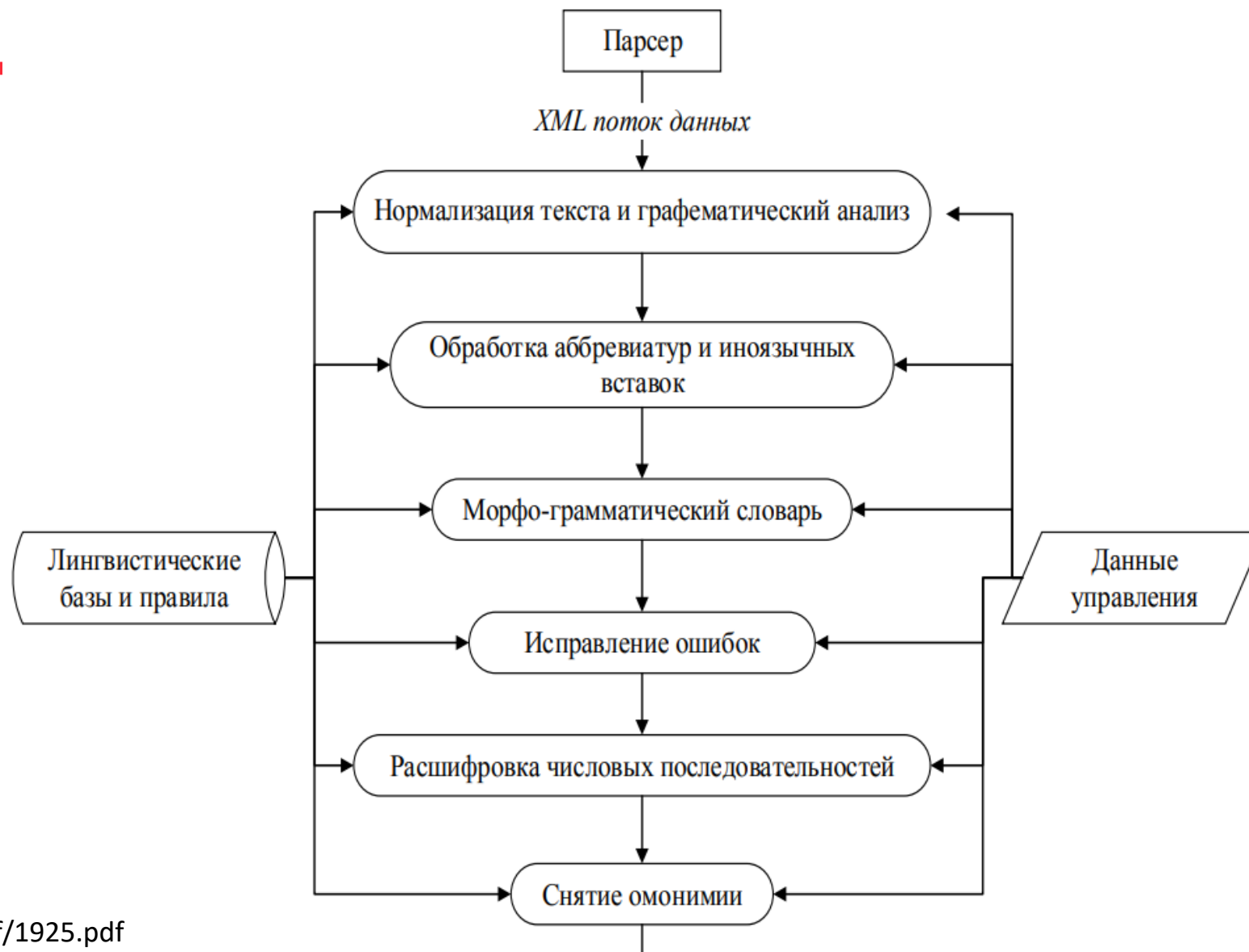
Фонетическое членение речи

Непрерывный поток звучащей речи является структурированным, он членится на разные по объему отрезки, фонетические единицы, которые образуют иерархическую систему.

Сегодня светит солнце, но дует ветер.

синтагмы	<i>сегодня светит солнце</i>	<i>но дует ветер</i>
слова	<i>сегодня светит солнце</i>	<i>но дует ветер</i>
слоги	<i>се-го-дня све-тит сол-нце</i>	<i>но ду-ет ве-тер</i>
фонемы	<i>s'/i₁-v/o₀-d'/n'/a₄ s/v'/e₀-t'/i₄/t s/o₀-n/c/y₄</i>	<i>n/o₁ d/u₀-j/i₄/t v'/e₀-t'/i₄/r</i>

Лингвистический процессор





Как собрать для русского языка?

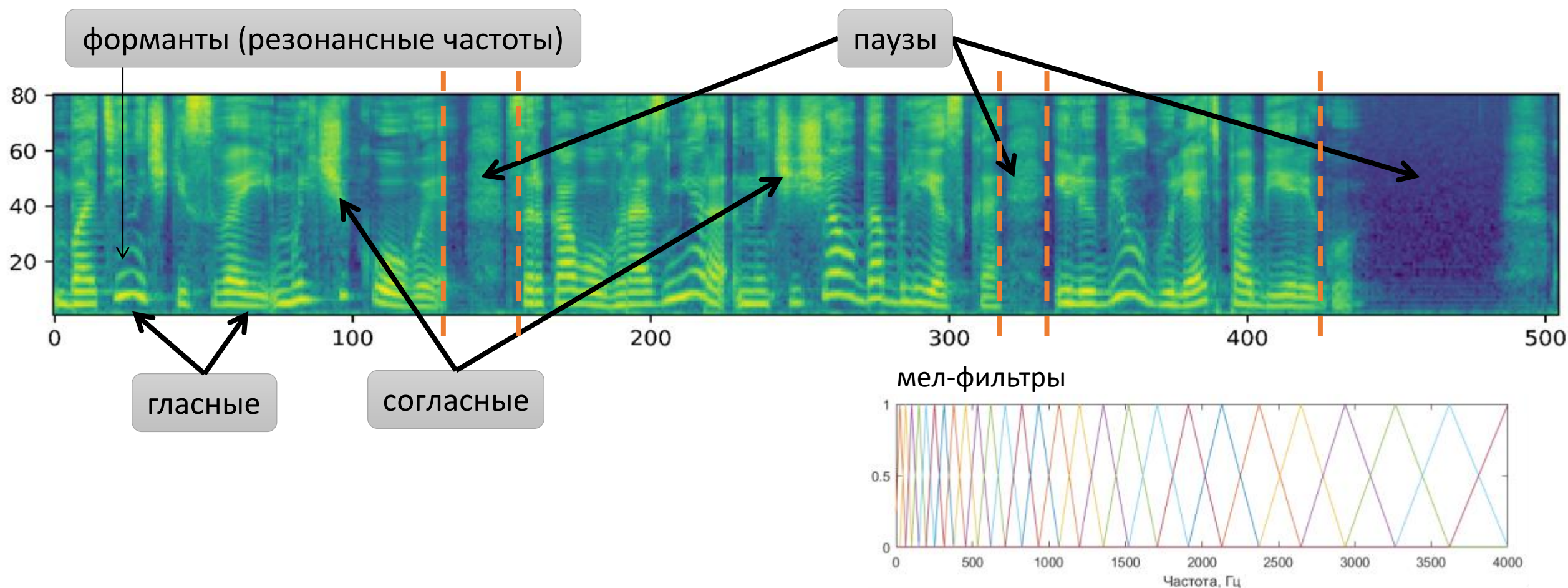
Нормализовать текст: https://github.com/snakers4/russian_stt_text_normalization

Разбить на предложения: <https://natasha.github.io/razdel>

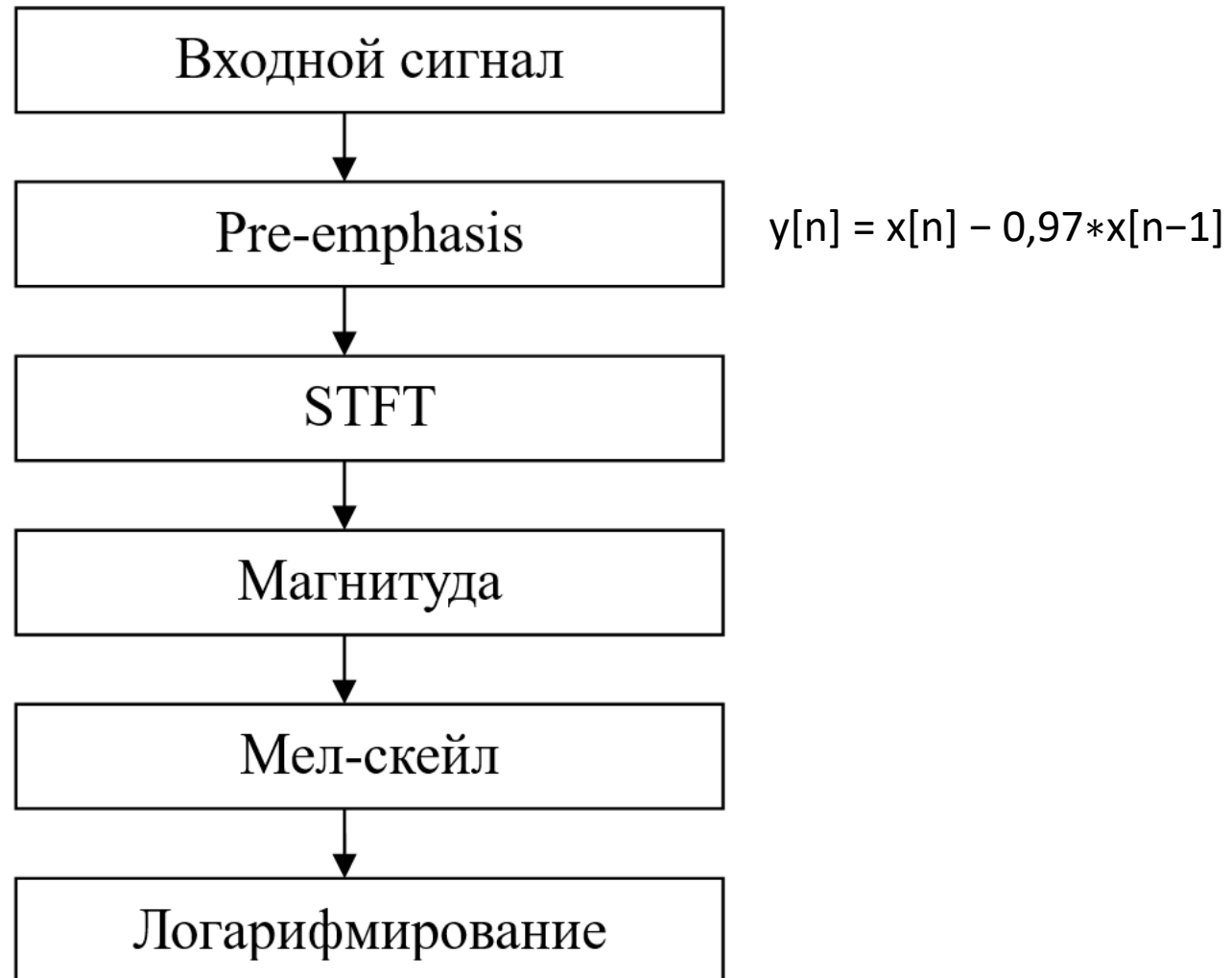
Построить транскрипцию: https://github.com/nsu-ai/russian_g2p

Спектр речевого сигнала

«Ботиночки свои мальчиковые, сорокового размера, начищал до блеска и любил гулять по берегу.»



Как подготовить аудиофичи?





Где взять датасет?

RUSLAN (22000 семплов, 44КГц, PCM16): <https://ruslan-corpus.github.io>

OPEN STT: https://github.com/snakers4/open_stt

Другие: <https://github.com/coqui-ai/open-speech-corpora>

Извлечь аудио с субтитрами из YouTube

Разметить аудиокниги

Записать в студии



Как оценивать качество?

С привлечением слушателей:

- MOS: https://en.wikipedia.org/wiki/Mean_opinion_score
- MUSHRA: <https://en.wikipedia.org/wiki/MUSHRA>

Автоматические методы:

- ASR
- PESQ: https://en.wikipedia.org/wiki/Perceptual_Evaluation_of_Speech_Quality
- Neural MOS prediction: <https://arxiv.org/pdf/2007.08267.pdf>



Классификация моделей

По архитектуре:

Авторегрессионные

- обучение и инференс
- только инференс

Параллельные

По типу блоков:

Рекуррентные

Сверточные

Трансформеры

По типу входных фичей:

Текст

Лингвистические признаки

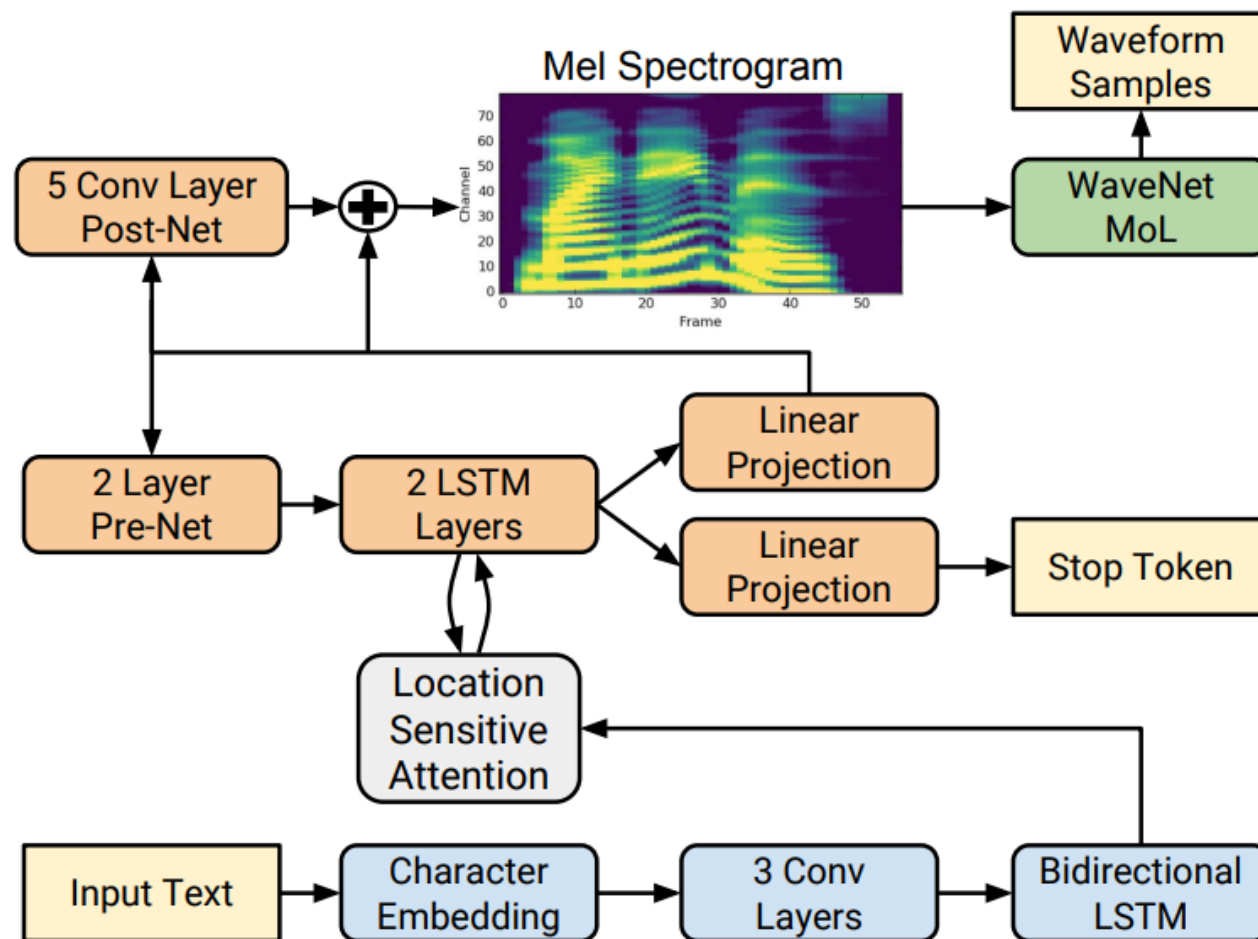
Мел-спектр

Длительности фонем

Пич (F_0)

Энергия

Tacotron2, 2018



Location Sensitive Attention

$$e_{i,j} = w^T \tanh(W s_{i-1} + V h_j + b)$$

h – выходы текстового энкодера, s – состояние декодера

$$f_i = F * \alpha_{i-1}$$

α – распределение внимания
на предыдущем шаге

$$e_{i,j} = w^T \tanh(W s_{i-1} + V h_j + U f_{i,j} + b)$$



Losses

L_1 или L_2 на спектр

BCE на Stop Token

Как быть, если нет вокодера под рукой?

Algorithm 1 Griffin–Lim phase reconstruction with $\text{STFT}_{\alpha,\beta}$

Input: $|\mathbf{X}^{[0]}| \in \mathbb{R}^{\alpha N \times T}$, where T is total frame number

Output: $\hat{x}(t)$

initial random phase $\Phi \in \mathbb{R}^{\alpha N \times T}$

$\mathbf{X}^{[1]} = |\mathbf{X}^{[0]}| \odot \Phi$

$x^{[1]}(t) = \text{iSTFT}_{\alpha,\beta} [\mathbf{X}^{[1]}]$

for $i = 1 : I - 1$ **do**

$\mathbf{Y}^{[i]} = \text{STFT}_{\alpha,\beta} [x^{[i]}(t)]$

$\mathbf{X}^{[i+1]} = |\mathbf{X}^{[0]}| \odot \exp \left\{ j \angle \mathbf{Y}^{[i]} \right\}$

$x^{[i+1]}(t) = \text{iSTFT}_{\alpha,\beta} [\mathbf{X}^{[i+1]}]$

end for

return $\hat{x}(t) = x^{[I]}(t)$



Tacotron2, 2018

Достоинства:

- Высокое качество речи
- Проверена работоспособность для множества языков
- Возможен синтез в реальном времени на CPU, в т.ч. в режиме стриминга

Недостатки:

- Нестабильный синтез (пропуск слов, безостановочная генерация)
- Деградация качества при увеличении длины высказывания
- Слабые возможности по управлению речью
- Медленная сходимость
- Медленное обучение и инференс (на GPU)



Как улучшить стабильность?

Удалить длинные паузы

Добавить пунктуацию как дополнительный вход в модель

Использовать монотонное внимание:

- Forward Attention: <https://arxiv.org/pdf/1807.06736.pdf>
- Monotonic Chunkwise Attention: <https://arxiv.org/pdf/1712.05382.pdf>
- Location-Relative Attention: <https://arxiv.org/pdf/1910.10288.pdf>

CTC-loss

Forward Attention

Algorithm 1 Forward Attention

Initialize:

$$\hat{\alpha}_0(1) \leftarrow 1$$

$$\hat{\alpha}_0(n) \leftarrow 0, n = 2, \dots, N$$

for $t = 1$ to T **do**

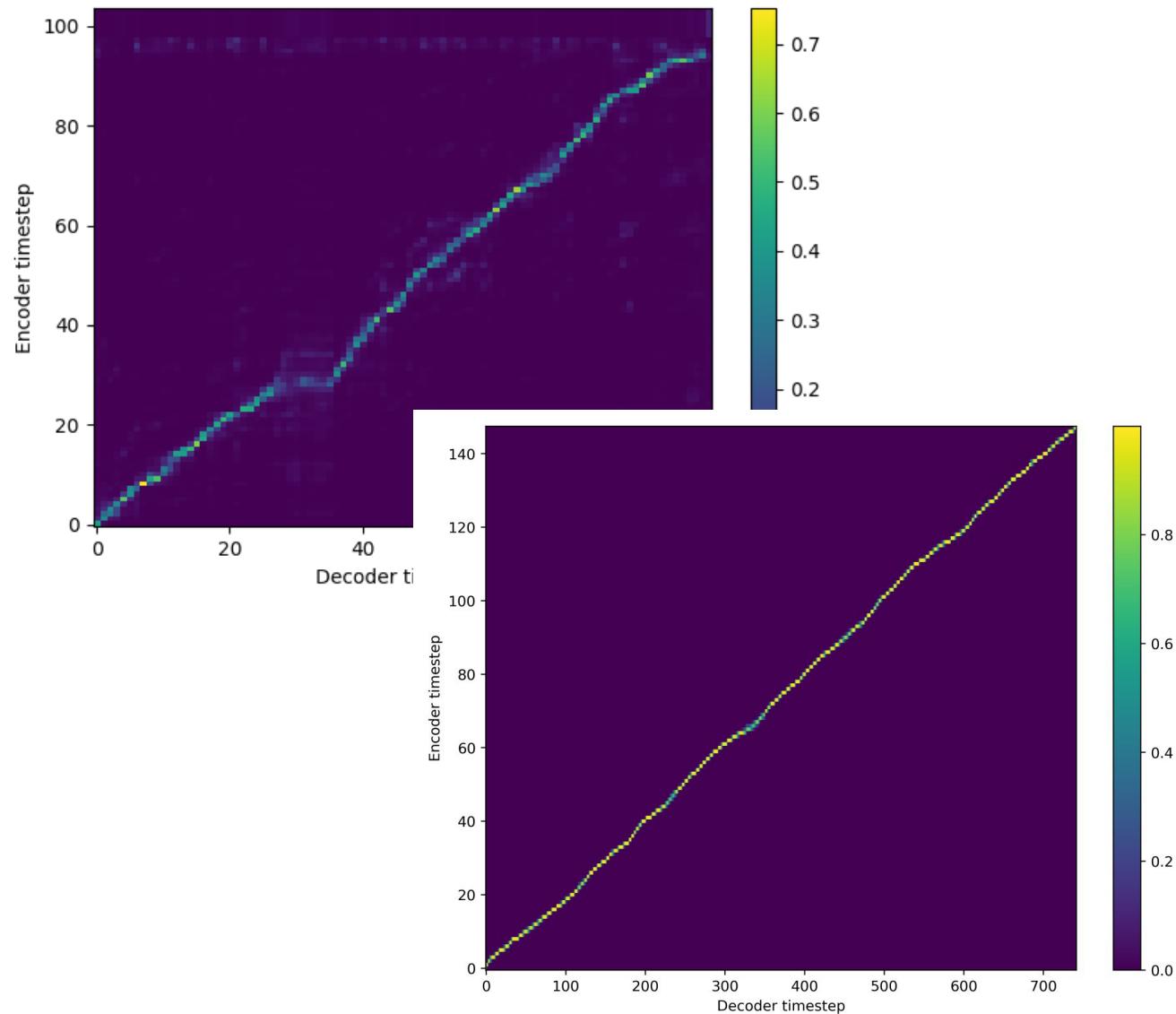
$$y_t(n) \leftarrow \text{Attend}(\mathbf{x}, \mathbf{q}_t)$$

$$\hat{\alpha}'_t(n) \leftarrow (\hat{\alpha}_{t-1}(n) + \hat{\alpha}_{t-1}(n-1))y_t(n)$$

$$\hat{\alpha}_t(n) \leftarrow \hat{\alpha}'_t(n) / \sum_{m=1}^N \hat{\alpha}'_t(m)$$

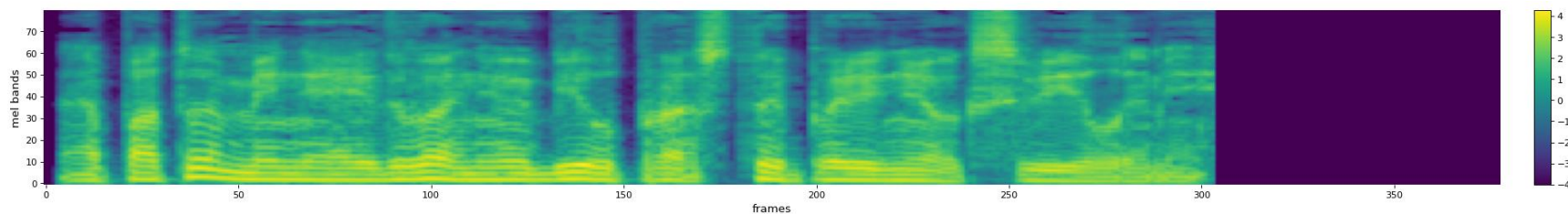
$$\mathbf{c}_t \leftarrow \sum_{n=1}^N \hat{\alpha}_t(n) \mathbf{x}_n$$

end for

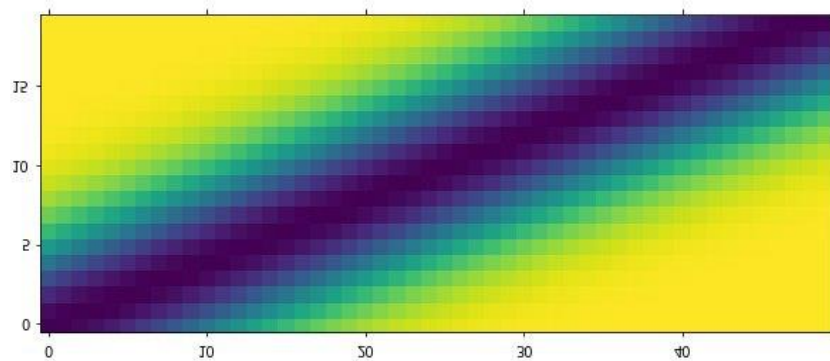


Как ускорить сходимость?

Семплировать в батч спектры равной длины



Применять Guided Attention Loss (<https://arxiv.org/pdf/1710.08969.pdf>)



Заполнять паузы несколькими символами

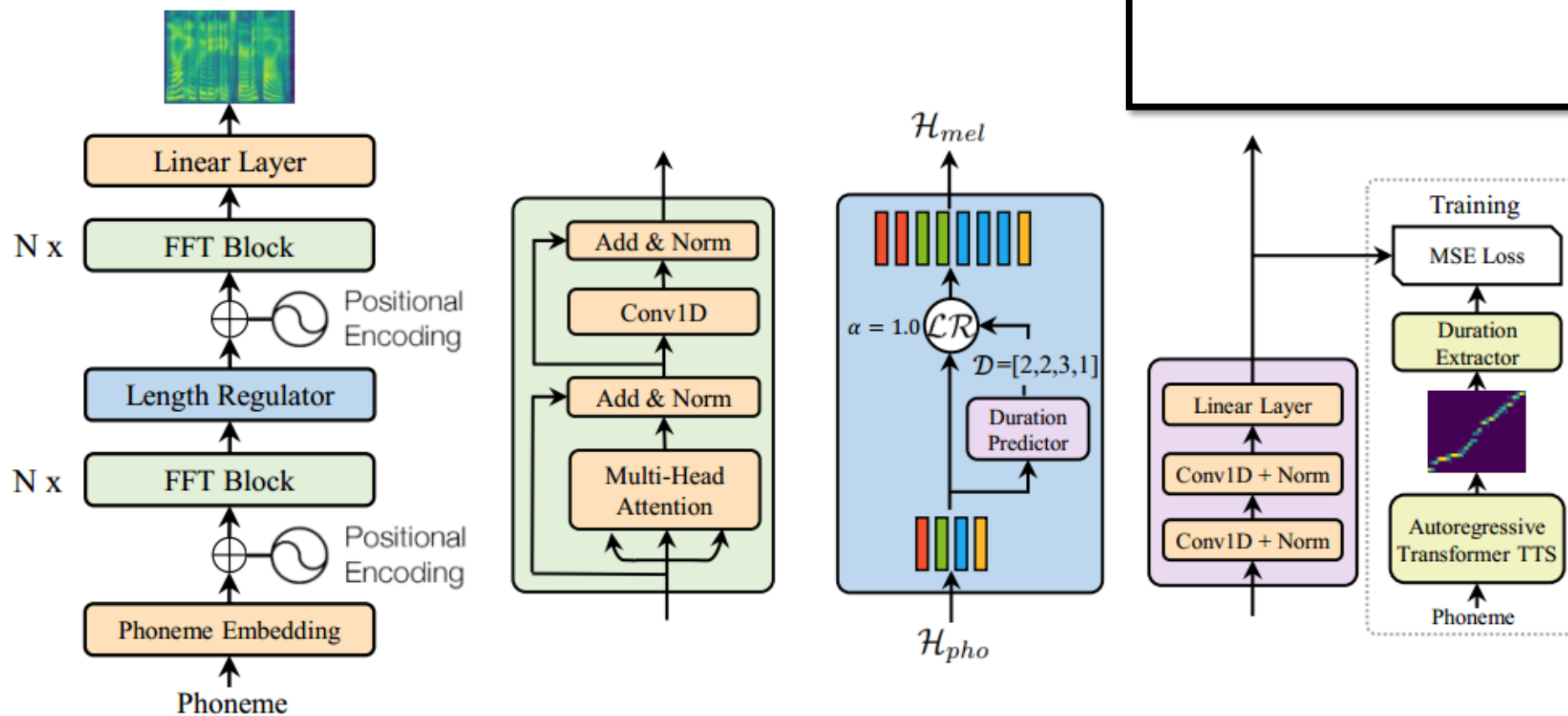


Как ускорить обучение?

Увеличить `hor`

Предсказывать несколько фреймов за шаг

FastSpeech, 2019



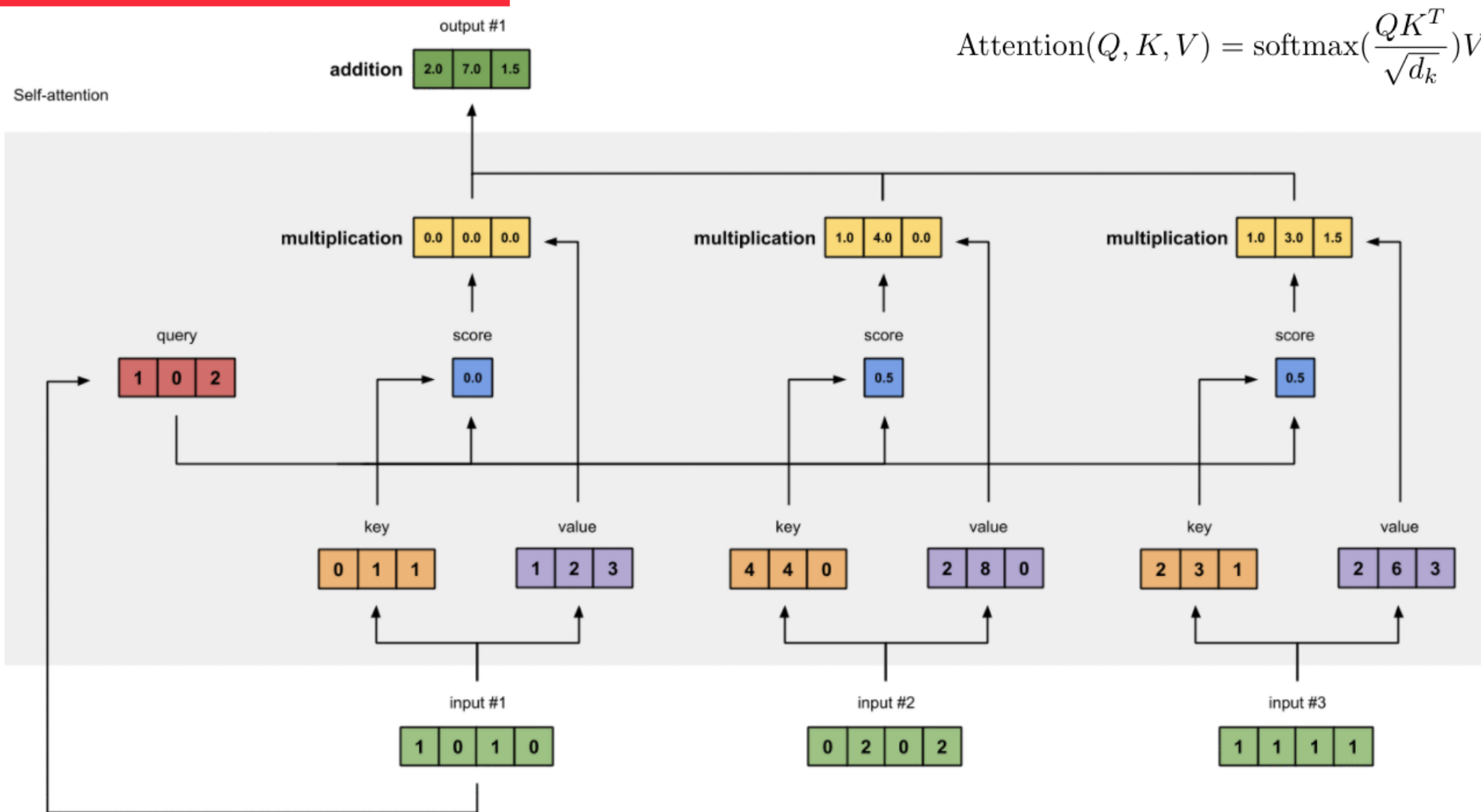
(a) Feed-Forward Transformer

(b) FFT Block

(c) Length Regulator

(d) Duration Predictor

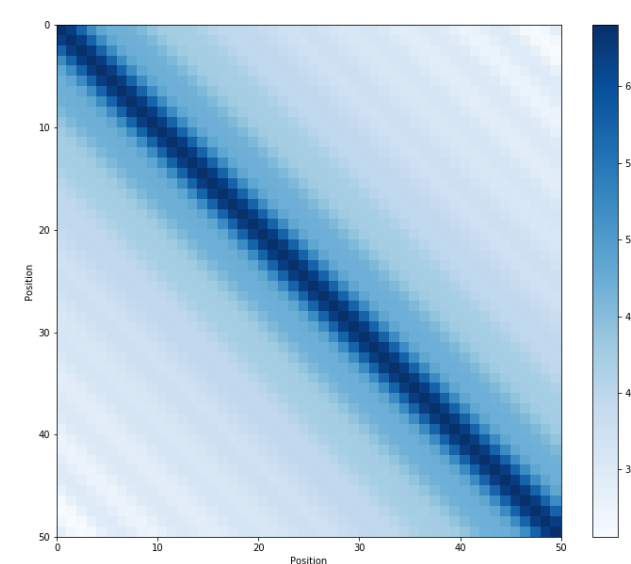
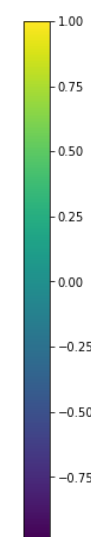
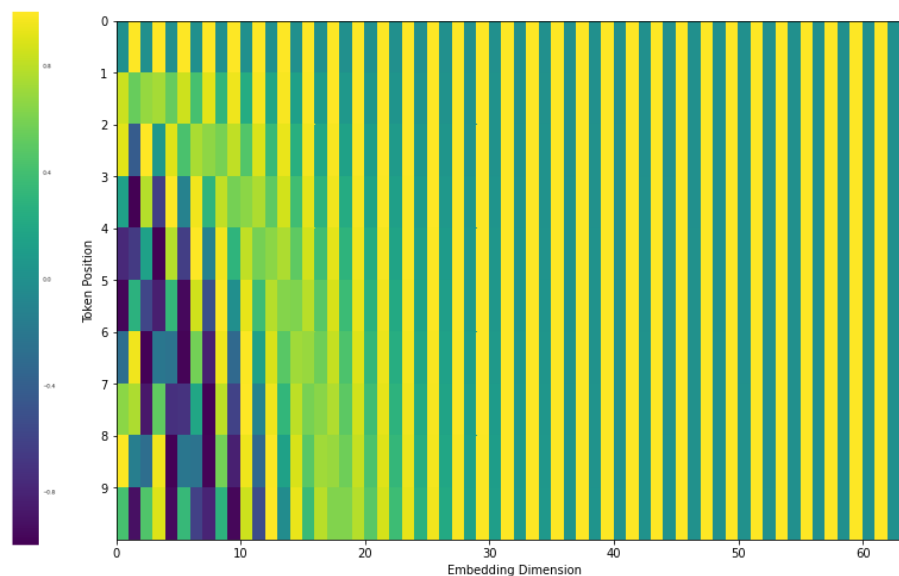
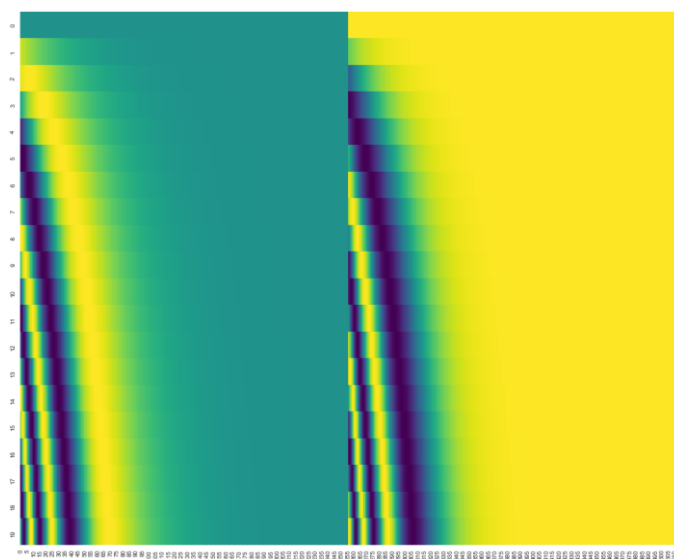
Scaled Dot-Product Attention, Self Attention



Positional Encoding

$$PE_{(pos, 2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos, 2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$



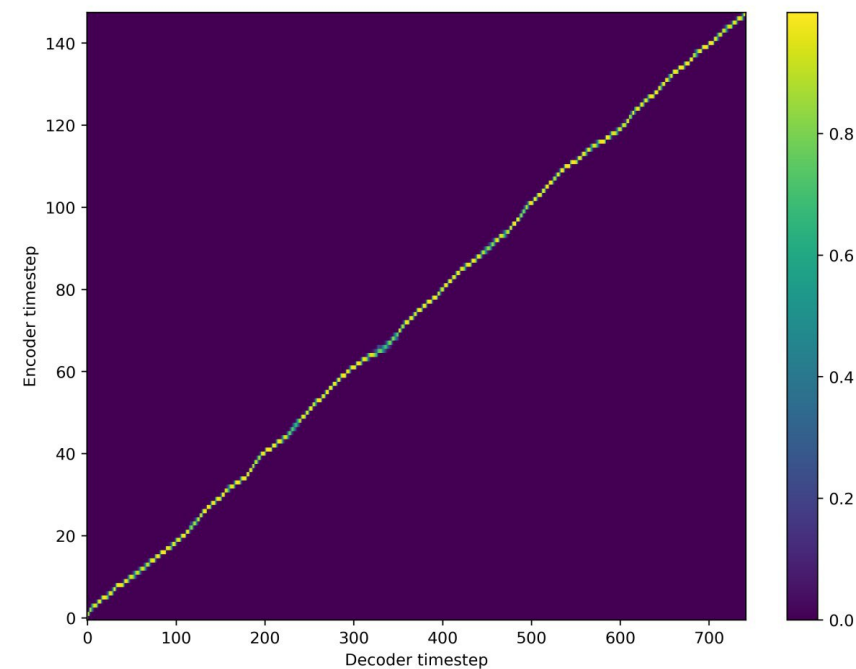
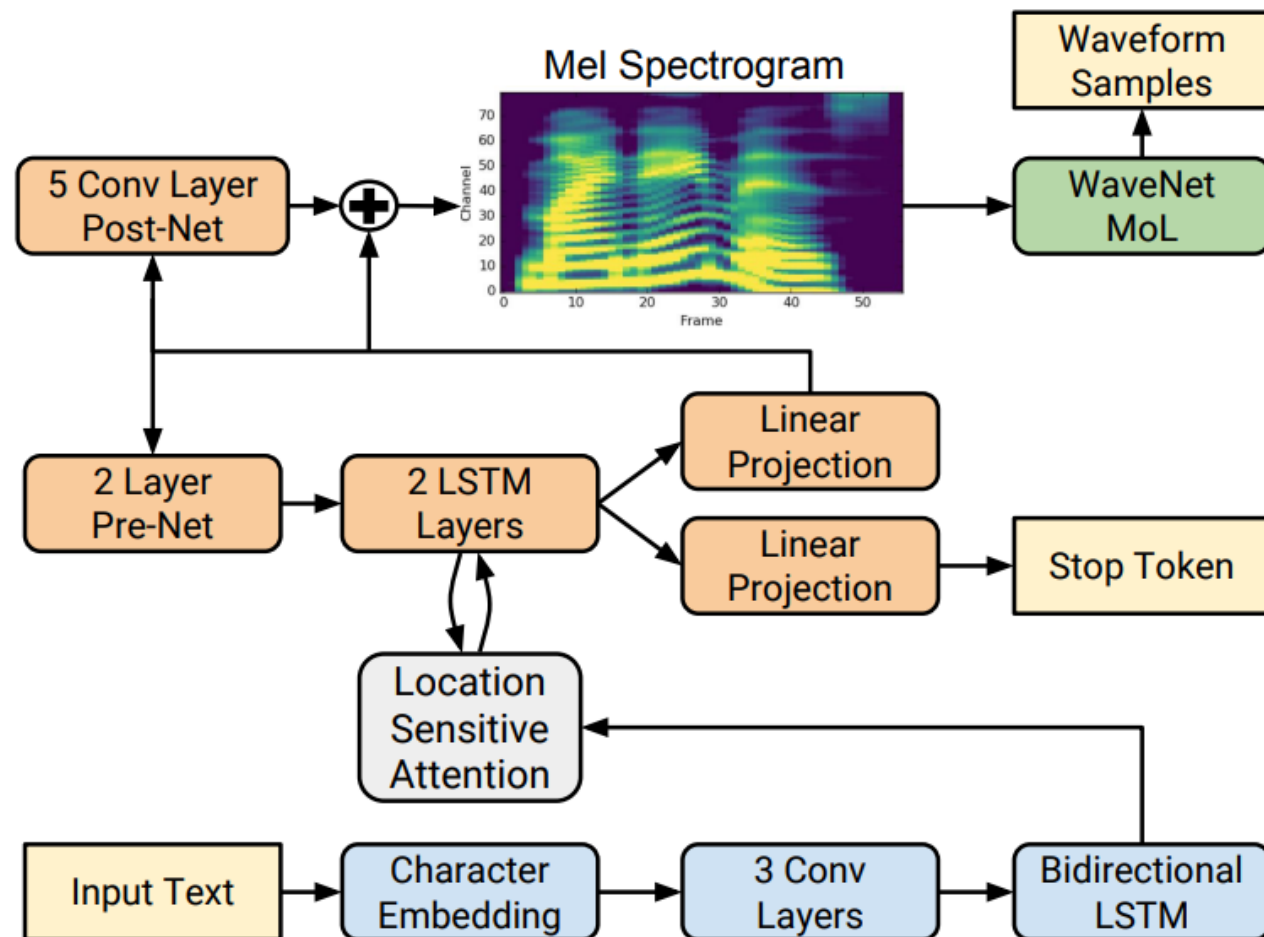


Где взять фонемное выравнивание?

Обучить такотрон

Montreal Forced Aligner (<https://github.com/MontrealCorpusTools/Montreal-Forced-Aligner>)
или (<https://github.com/pettarin/forced-alignment-tools>)

Tacotron2





FastSpeech

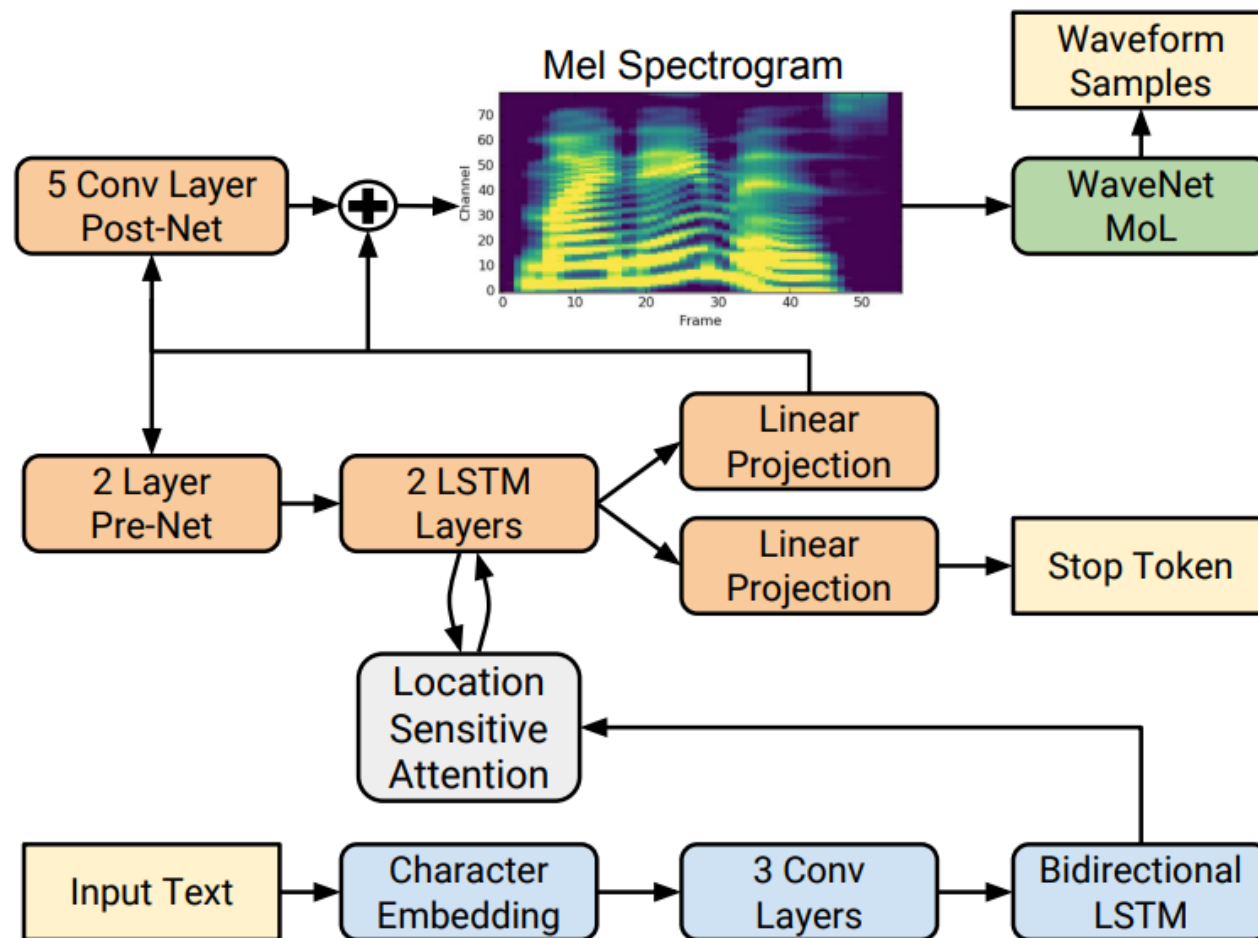
Достоинства:

- Высокое качество речи, но требуется тонкая настройка
- Возможен синтез в реальном времени на GPU
- Управление темпом речи и коробки

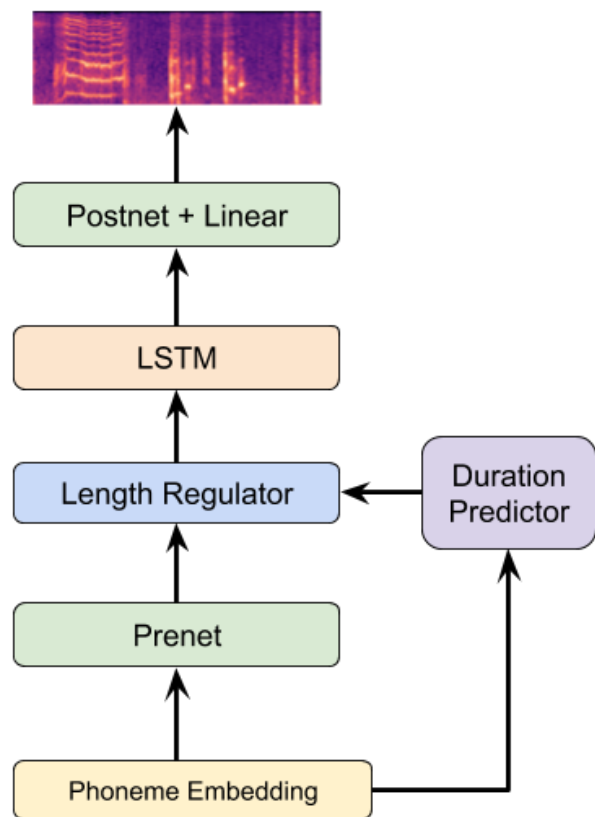
Недостатки:

- Зависимость от длины предложения
- Квадратичная сложность вычислений и потребления памяти

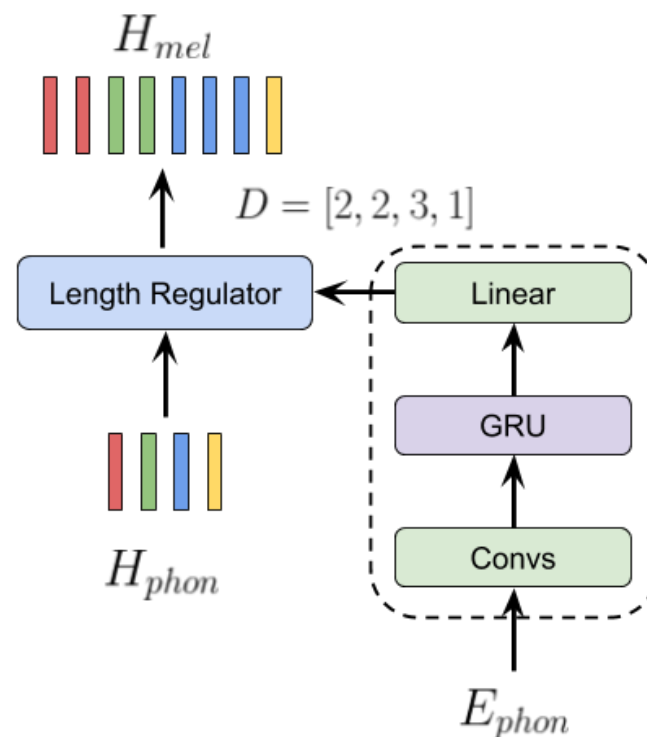
Как использовать аналогичную идею в Tacotron2?



Forward Tacotron, 2019



Forward Tacotron



Length Regulator with Duration Predictor



ForwardTacotron

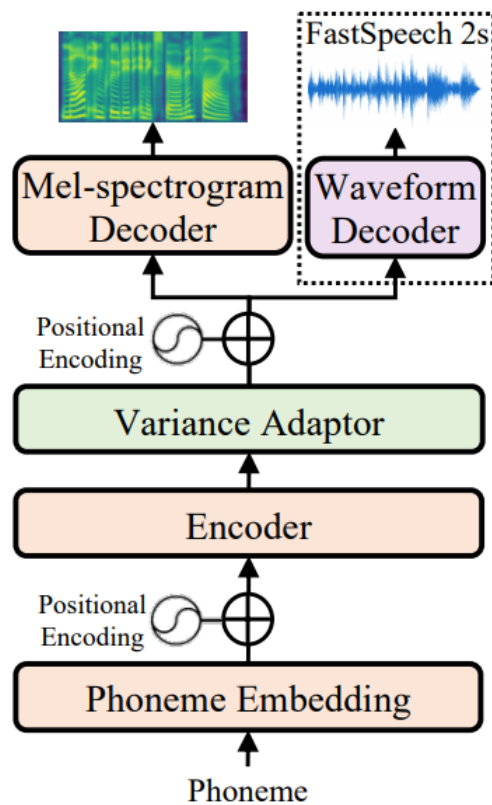
Достоинства:

- Высокое качество речи, сопоставимое с такотроном
- Быстрый инференс на GPU
- Управление темпом речи из коробки
- Работает с текстом любой длины

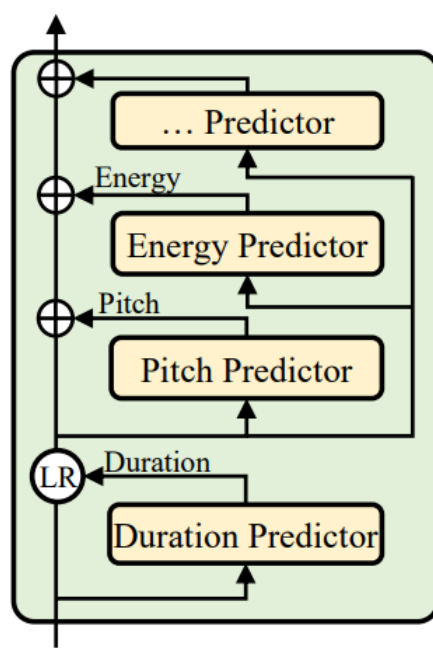
Недостатки:

- Медленное обучение
- Может зажевывать короткие фонемы

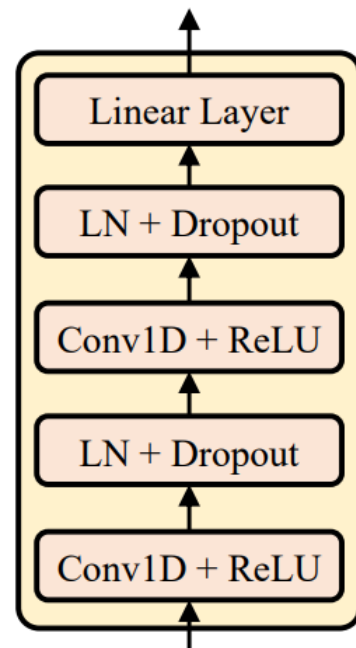
FastSpeech2, 2020



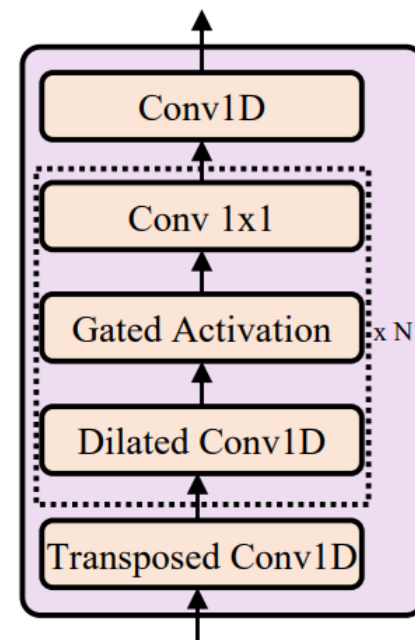
(a) FastSpeech 2



(b) Variance adaptor



(c)
Duration/pitch/energy
predictor



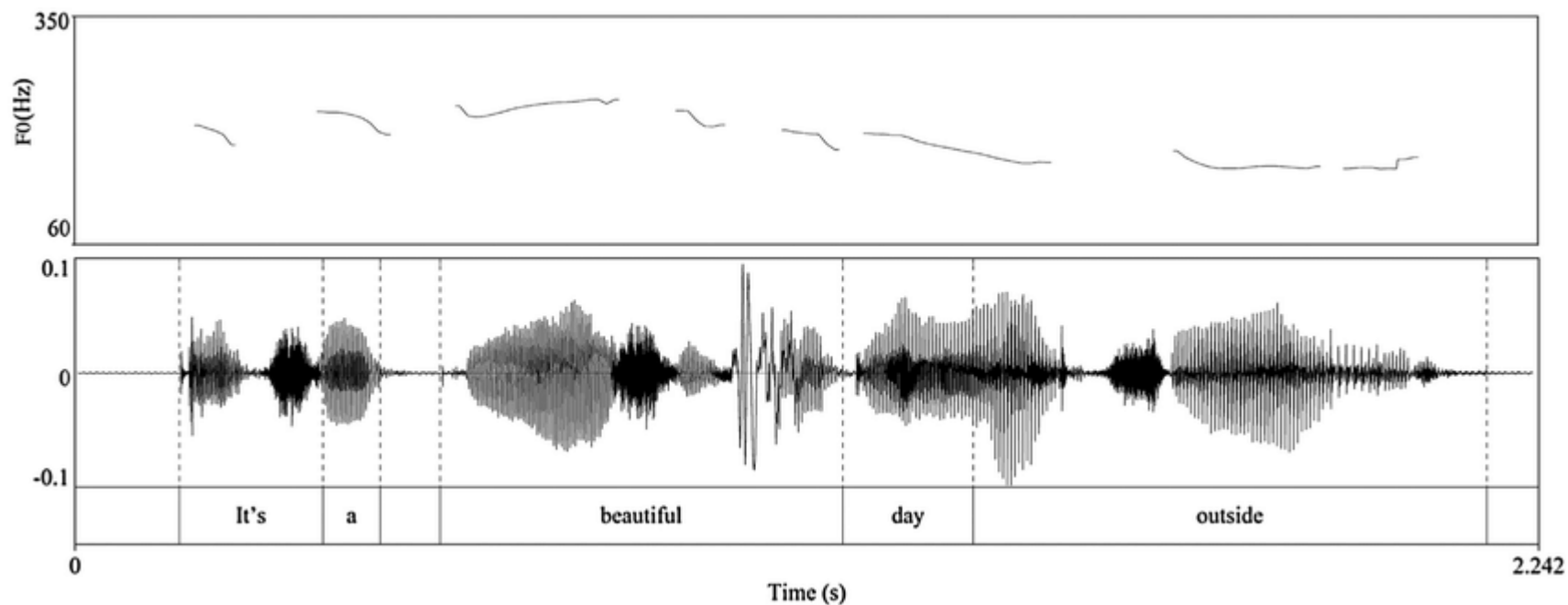
(d) Waveform decoder

Как подсчитать pitch?

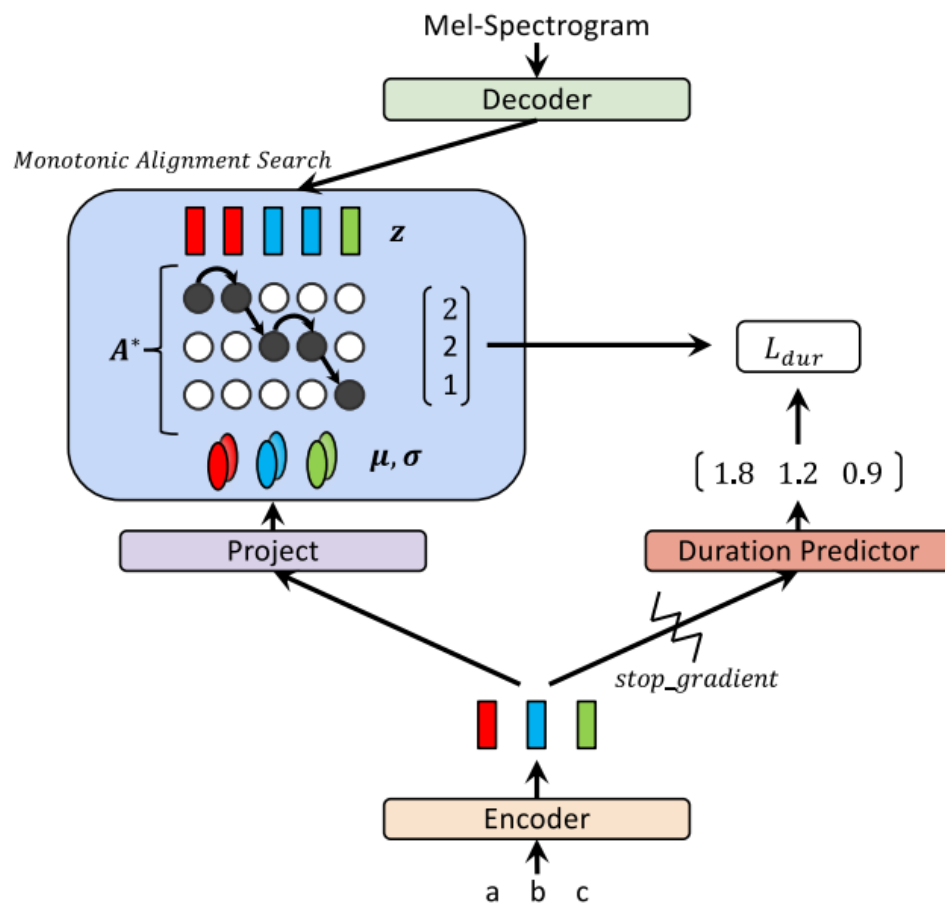
PyWORLD: <https://github.com/JeremyCCHsu/Python-Wrapper-for-World-Vocoder>

REAPER: <https://github.com/google/REAPER>

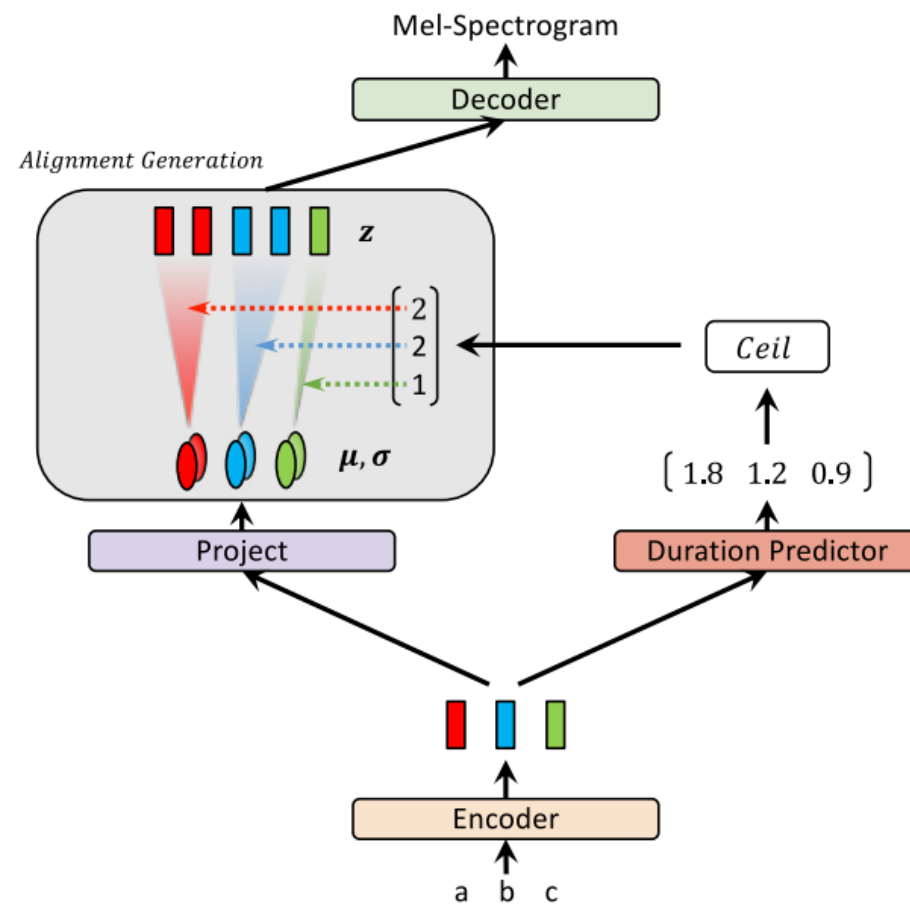
CREPE: <https://github.com/maxrmorrison/torchcrepe>



Glow-TTS, 2020

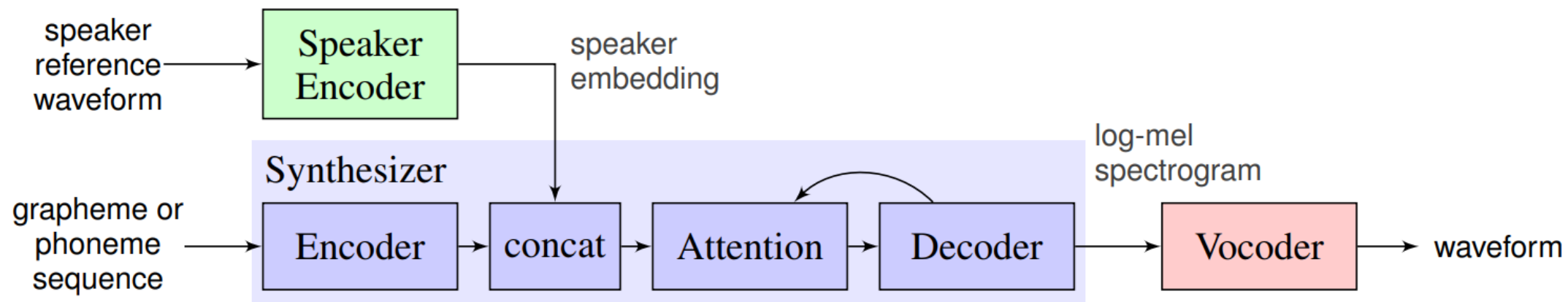


(a) An abstract diagram of the training procedure.



(b) An abstract diagram of the inference procedure.

Multispeaker TTS





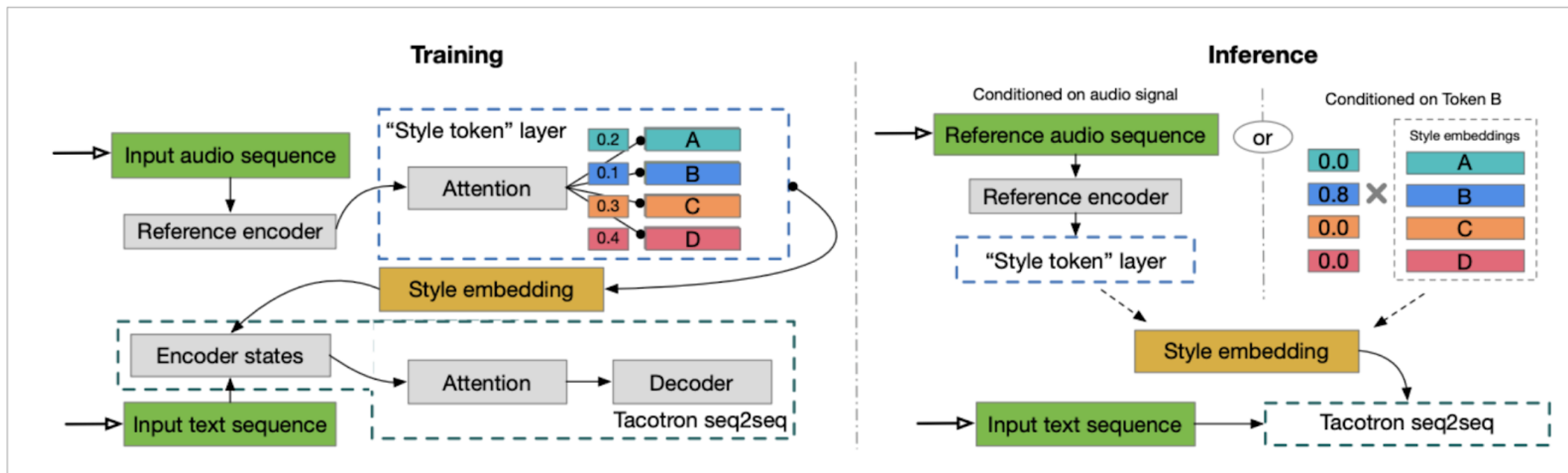
Как кодировать спикеров?

Индексы

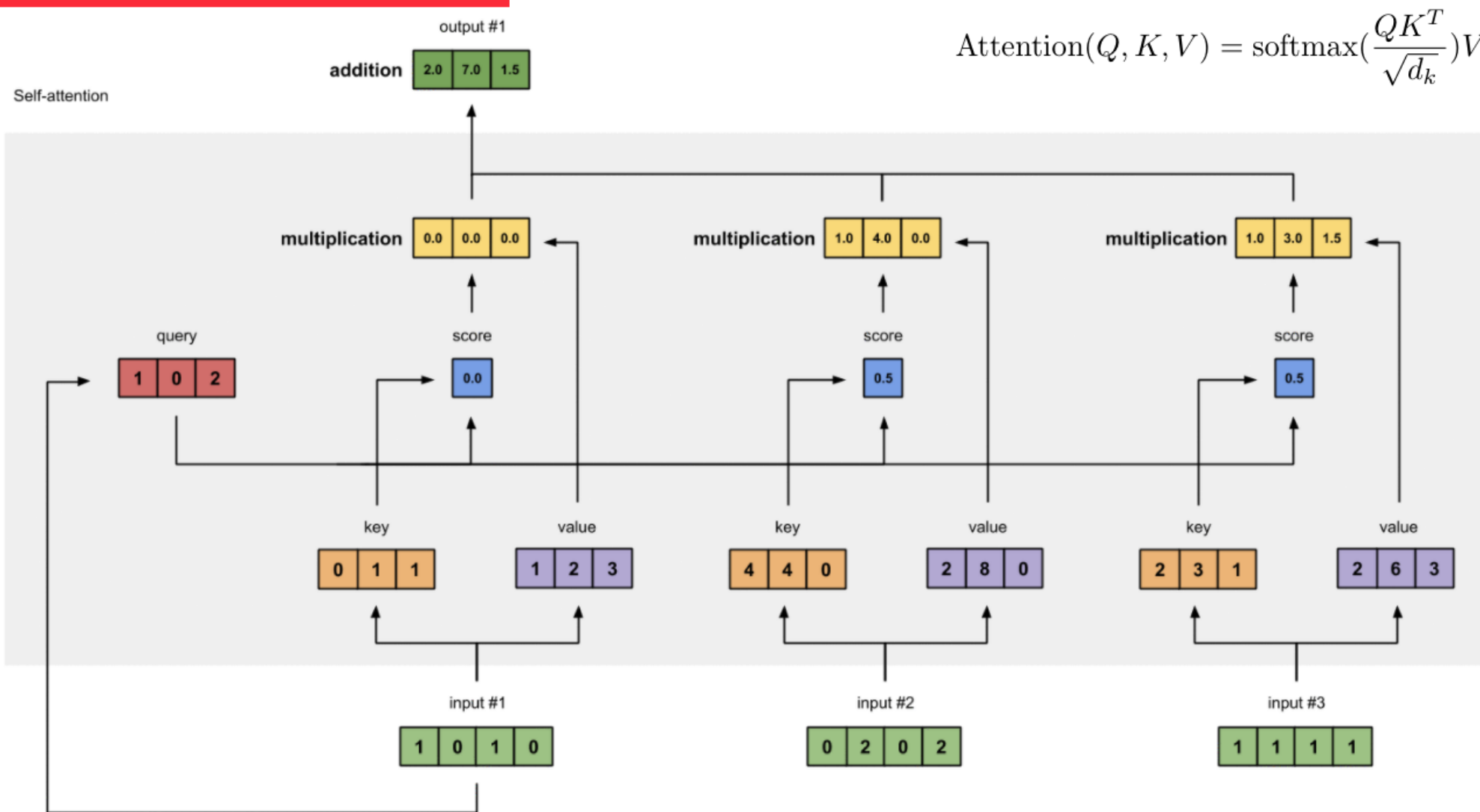
Обучаемые эмбединги

Био-эмбединги (<https://github.com/resemble-ai/Resemblyzer>)

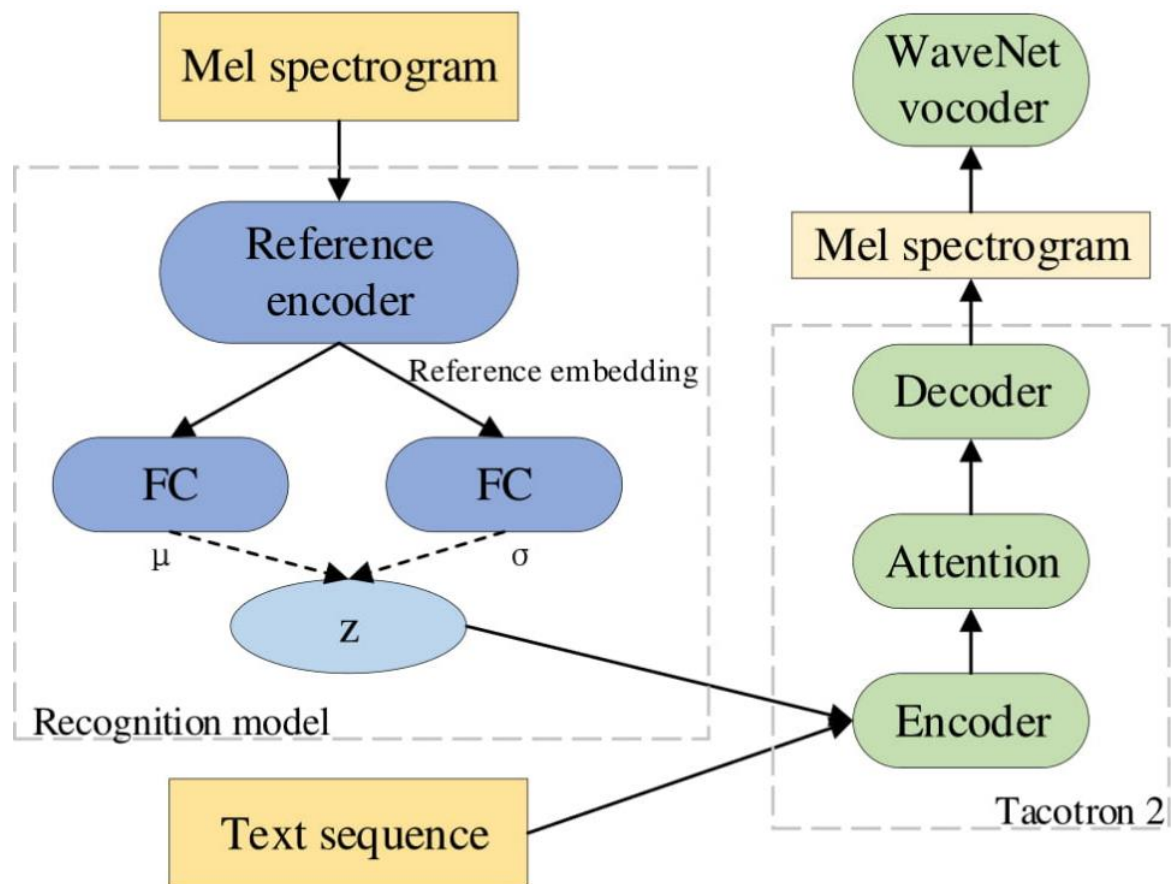
Global Style Tokens



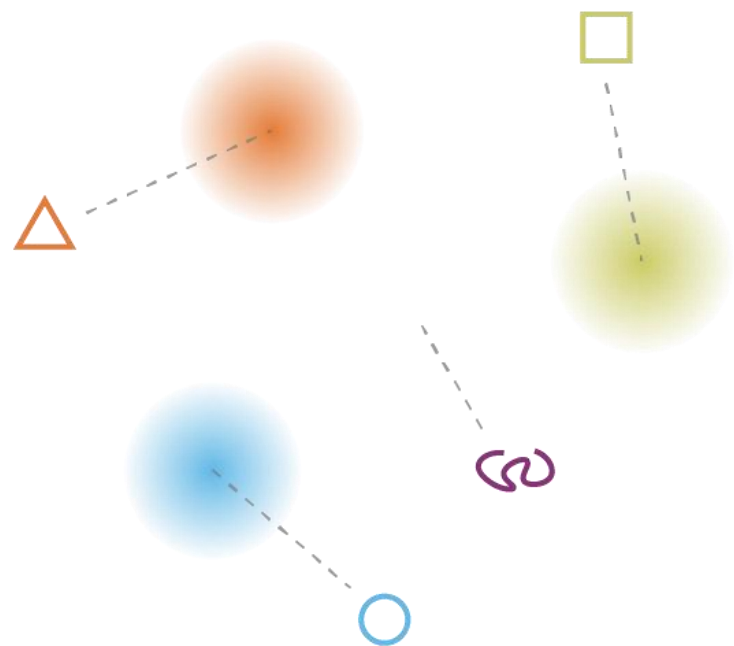
Scaled Dot-Product Attention, Self Attention



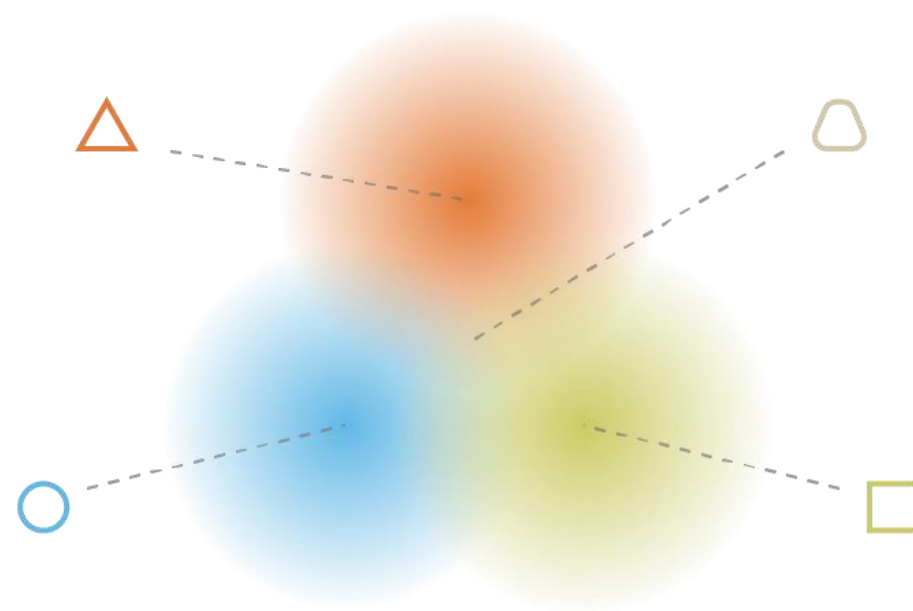
Variational Autoencoders



Variational Autoencoders

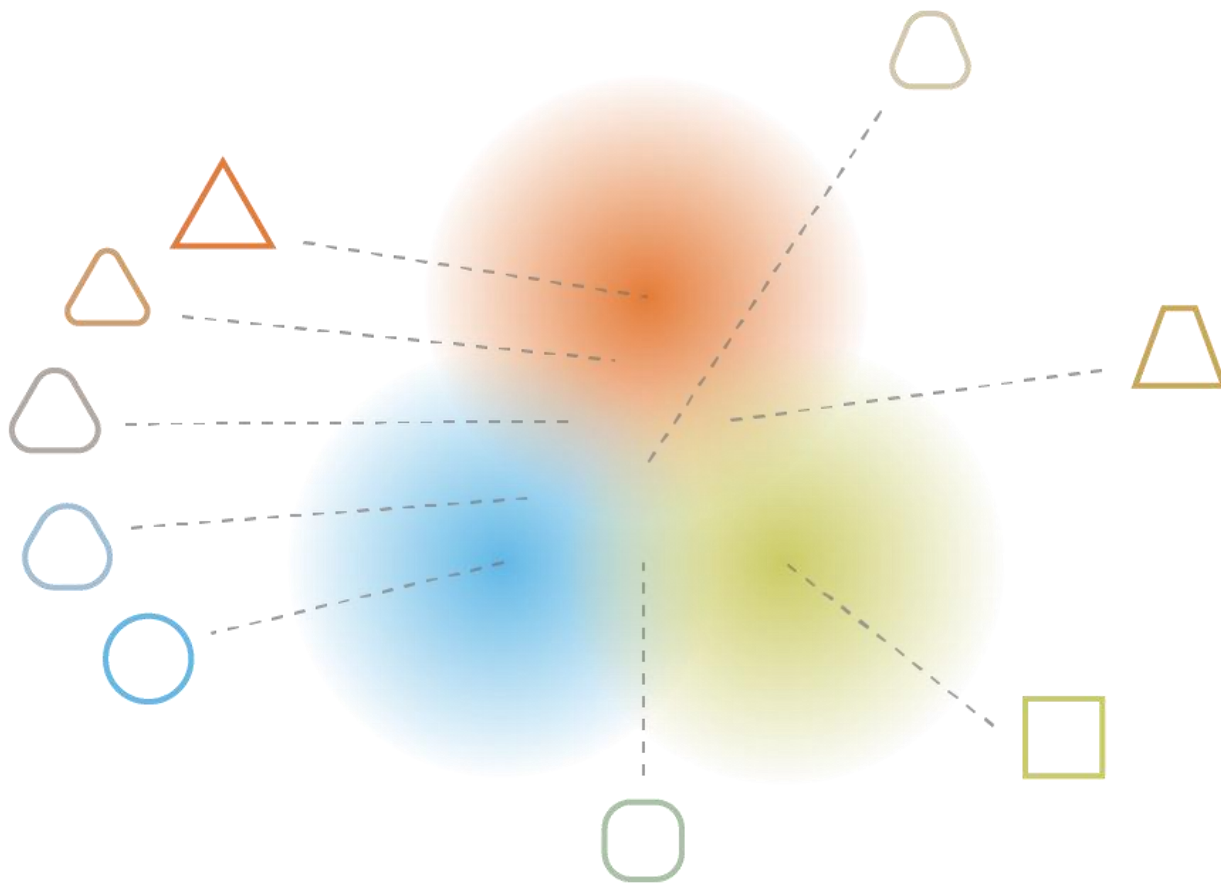


what can happen without regularisation

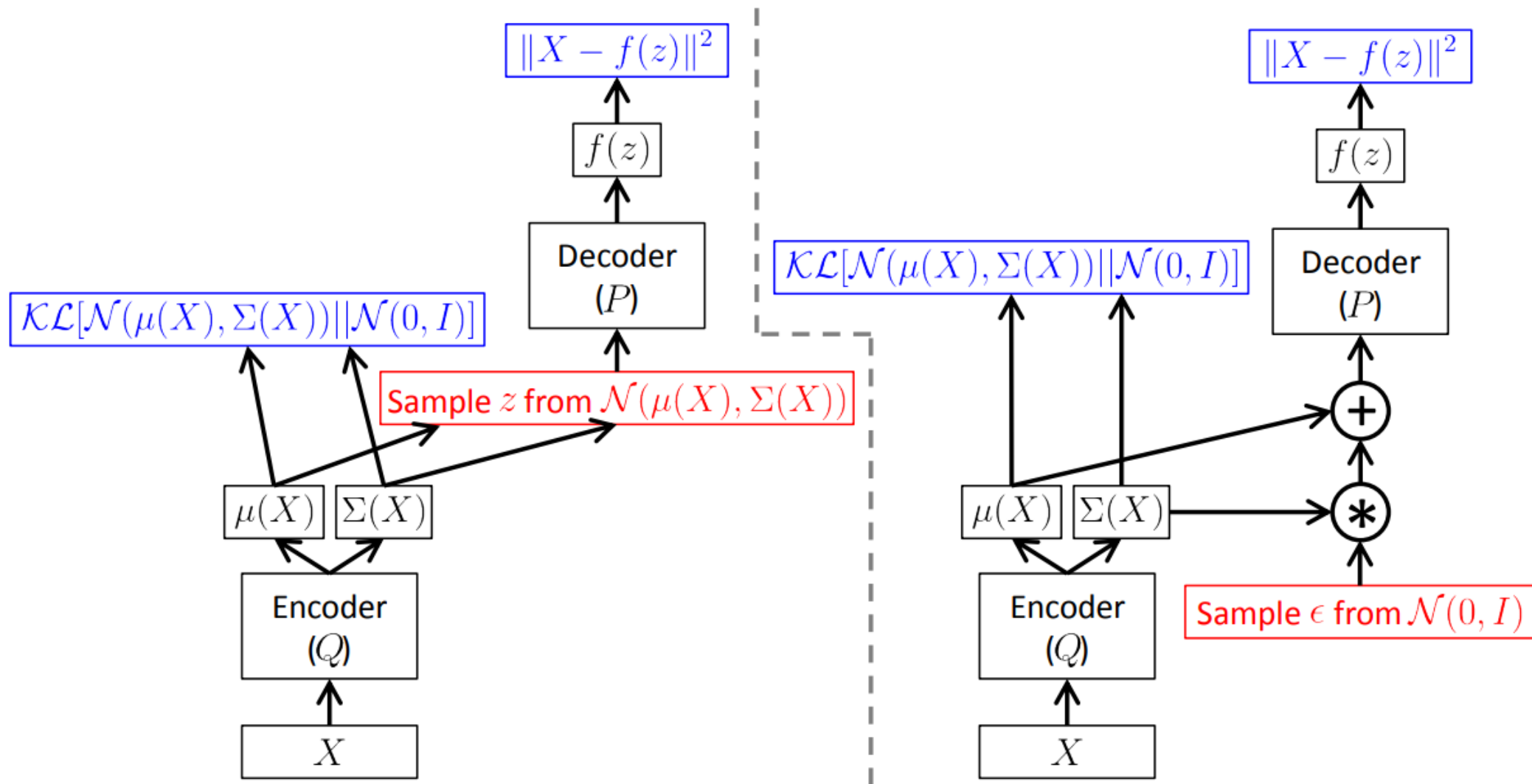


what we want to obtain with regularisation

Variational Autoencoders



Variational Autoencoders



SSML-разметка

Speech Synthesis Markup Language (<https://www.w3.org/TR/speech-synthesis11>)

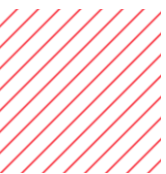
Я отойду на минутку. `<break time="5000"/>` Вы еще здесь?

`<say-as stress="2">` Перепрофилирование `</say-as>`

`<say-as interpret-as="telephone">` 2222230 `</say-as>`

`<prosody rate="0.5">` Надо есть, чтобы быстро бегать. `</prosody>`

Это `<emphasis level="strong">` огромный `</emphasis>` банковский счёт.



Спасибо за внимание!

