Синтез речи

Лекция №2

Гриша Стерлинг, SberDevices

Вокодеры ч.2:
1. Parallel WaveNet
2. WaveGlow
3. WaveRNN
4. LPCNet
5. GAN

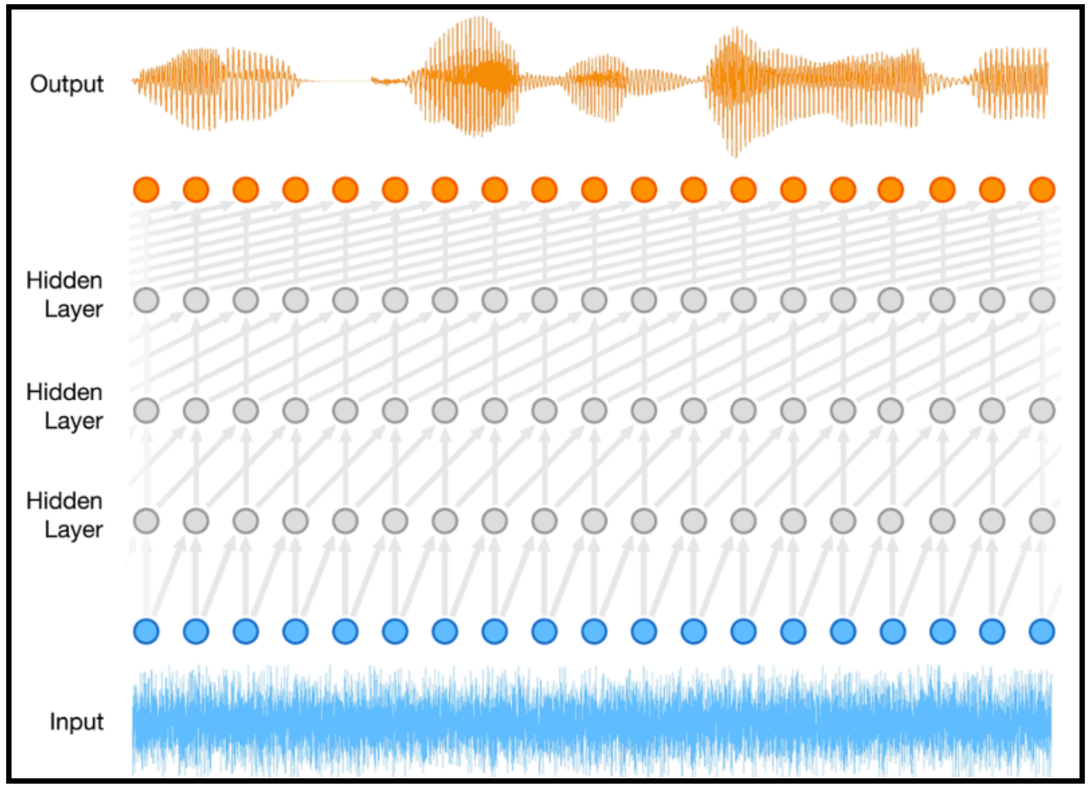
Акустика ч.1:
6. Tacotron

Неавторегрессионные вокодеры

Parallel WaveNet

WaveNet:
+ крутое качество
- в 20 раз медленнее realtime :)

Мотивация:
хочется синтезировать всю вавку за один инференс сетки

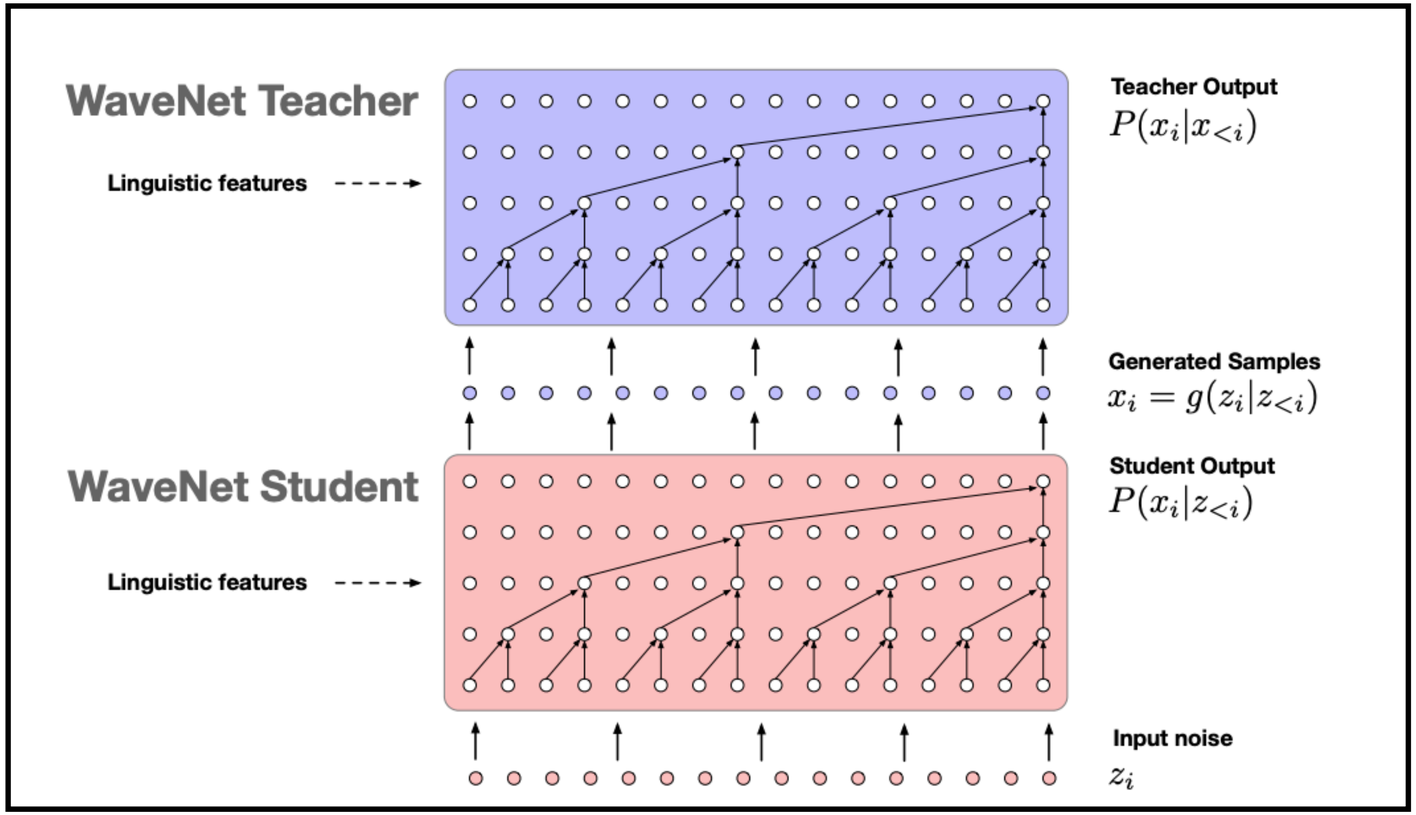


Parallel WaveNet

Knowledge distillation:

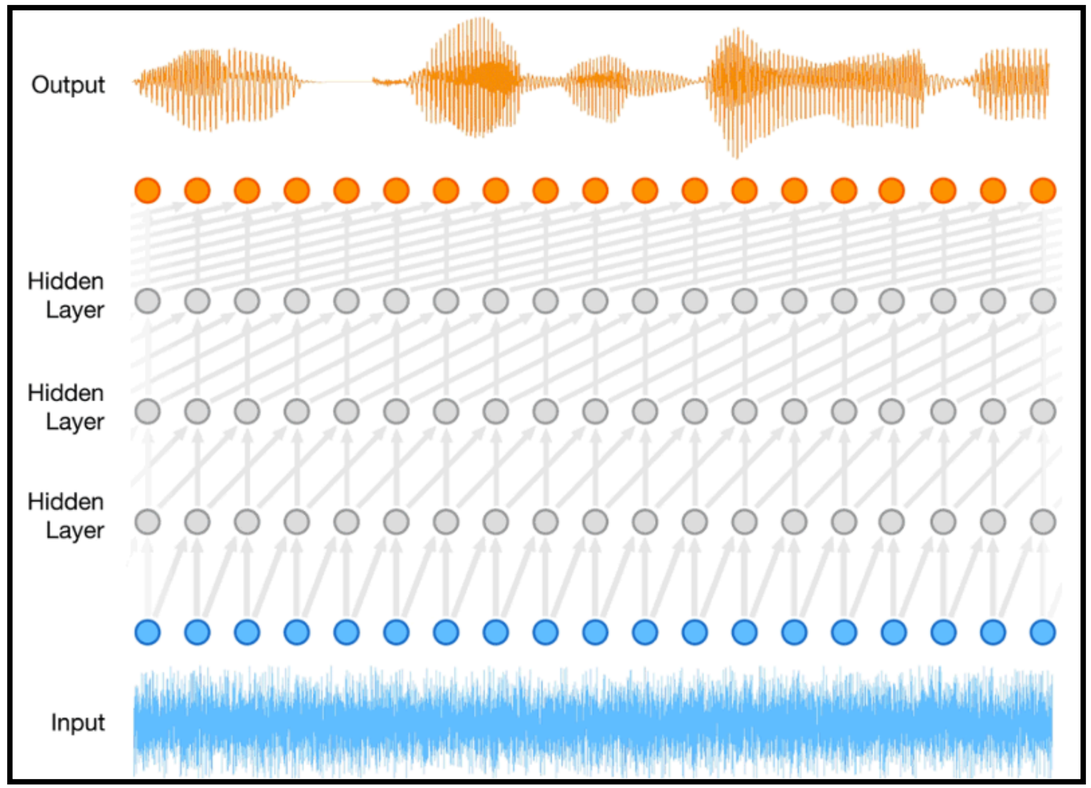
$$D_{\mathrm{KL}}\left(P_S||P_T\right) = H(P_S, P_T) - H(P_S)$$

Чтобы все лучше училось:
- stft loss
- perceptual loss
- contrastive loss



WaveGlow

$$z \sim \mathcal{N}(z; 0, \boldsymbol{I})$$
$$\boldsymbol{x} = \boldsymbol{f}_0 \circ \boldsymbol{f}_1 \circ \dots \boldsymbol{f}_k(\boldsymbol{z})$$
$$\boldsymbol{z} = \boldsymbol{f}_k^{-1} \circ \boldsymbol{f}_{k-1}^{-1} \circ \dots \boldsymbol{f}_0^{-1}(\boldsymbol{x})$$



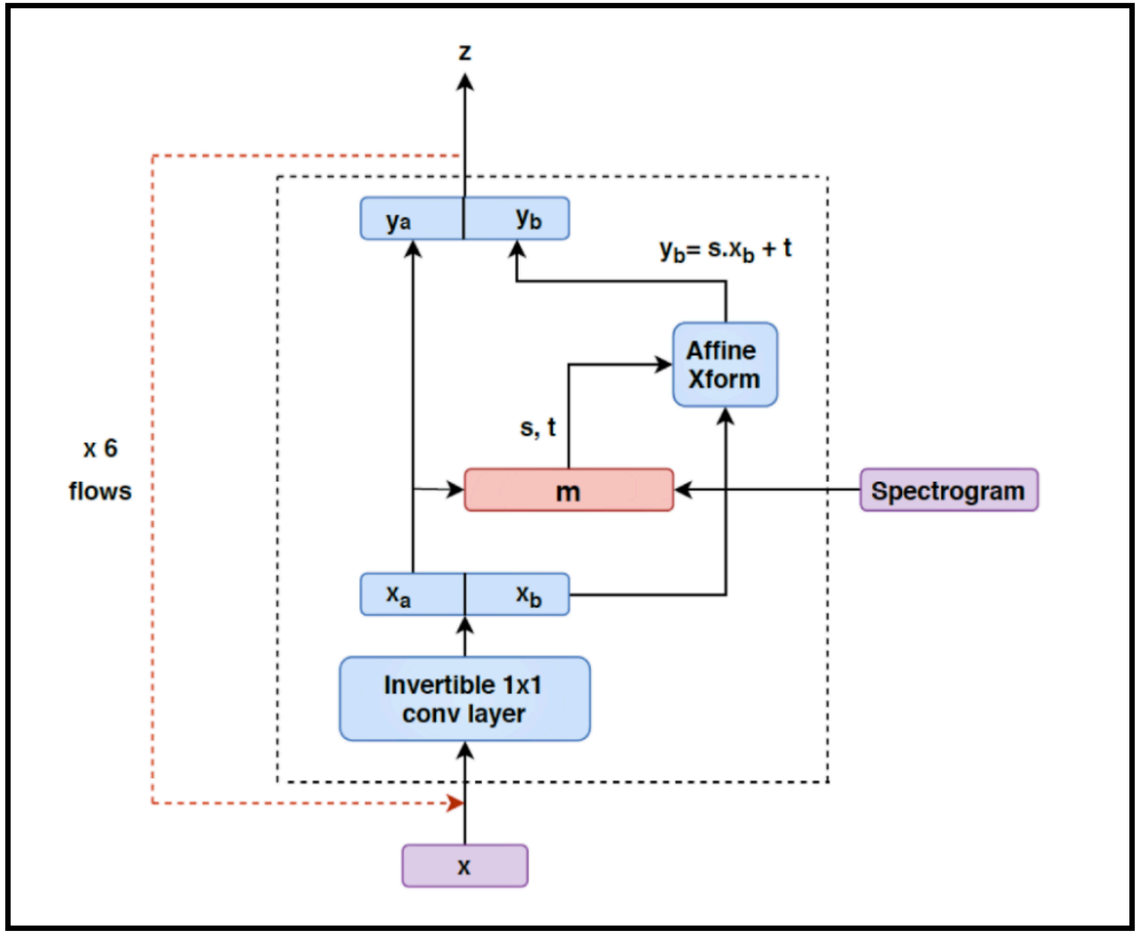
Обычно:
- для какого-то входа считаем выход
- считаем лосс и градиенты

WaveGlow:
- подставляем вейвформу в конец сетки
- считаем с помощью обратных преобразований, из какого входа оно было получено
- считаем лосс между z~N(0, 1) и z’

WaveGlow

$$\boldsymbol{x}_a, \boldsymbol{x}_b = split(\boldsymbol{x})$$
$$(\log \boldsymbol{s}, \boldsymbol{t}) = WN(\boldsymbol{x}_a, \textit{mel-spectrogram})$$
$$\boldsymbol{x}_b\prime = \boldsymbol{s} \odot \boldsymbol{x}_b + \boldsymbol{t}$$
$$\boldsymbol{f}^{-1}_{coupling}(\boldsymbol{x}) = concat(\boldsymbol{x}_a, \boldsymbol{x}_b\prime)$$

$$\log p_\theta(\boldsymbol{x}) = -\frac{\boldsymbol{z}(\boldsymbol{x})^T \boldsymbol{z}(\boldsymbol{x})}{2\sigma^2} + \sum_{j=0}^{\#coupling} \log \boldsymbol{s}_j(\boldsymbol{x}, \textit{mel-spectrogram}) + \sum_{k=0}^{\#conv} \log\det|\boldsymbol{W}_k|$$



Авторегрессионные вокодеры

WaveRNN

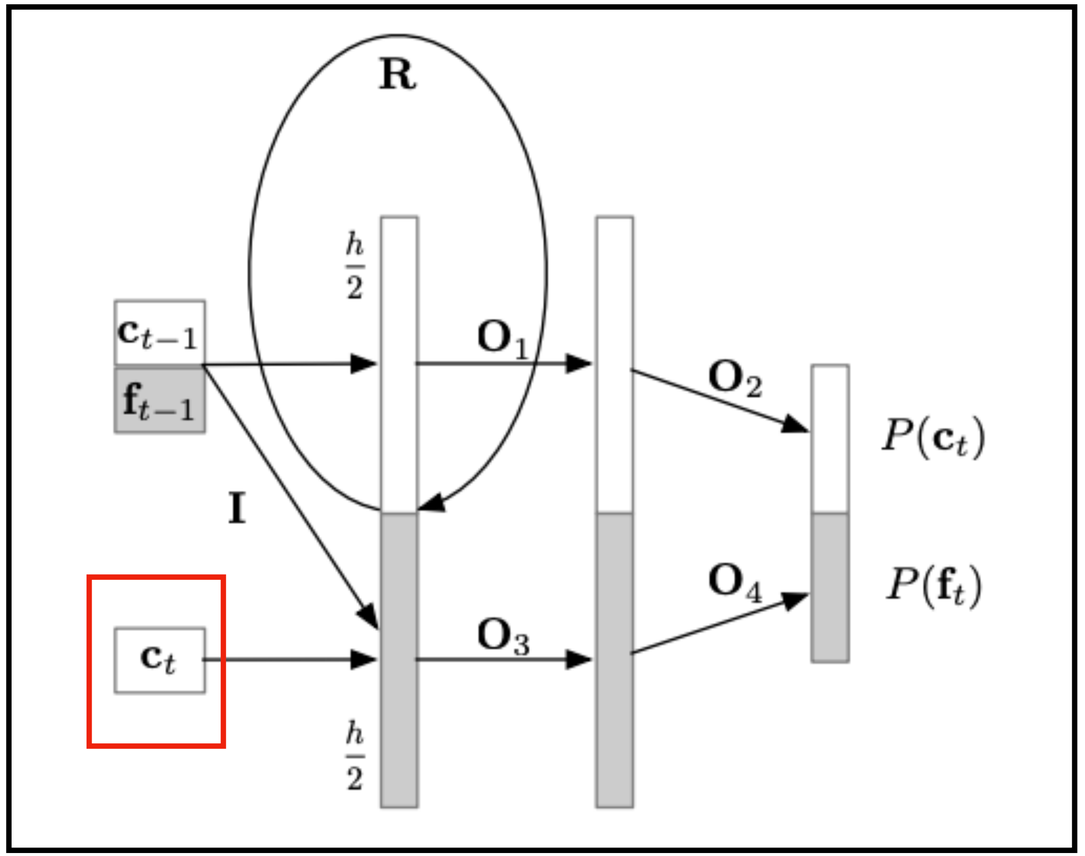
$$p(\mathbf{x} \mid \mathbf{h}) = \prod_{t=1}^{T} p(x_t \mid x_1, \ldots, x_{t-1}, \mathbf{h}) .$$

Идеи:

- использовать hidden state у GRU слоя вместо кучи сверток

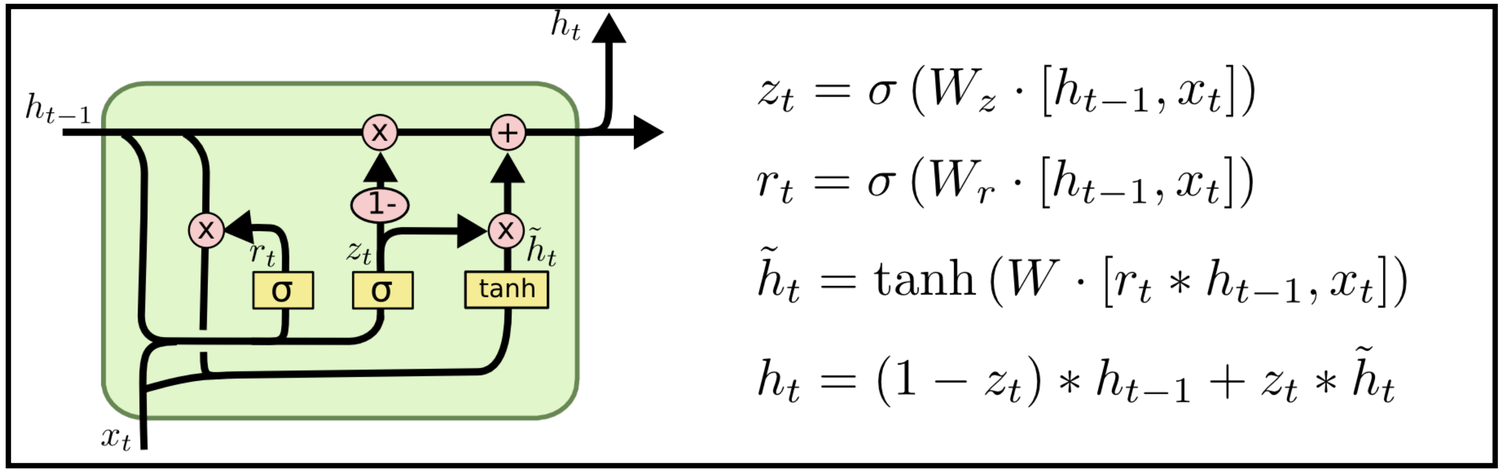
- coarse и fine представление вместо mu-law

- оптимизации (спарсификация, subscale)



WaveRNN

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



$$
\begin{aligned}
z_t &= \sigma\left(W_z \cdot [h_{t-1}, x_t]\right) \\
r_t &= \sigma\left(W_r \cdot [h_{t-1}, x_t]\right) \\
\tilde{h}_t &= \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right) \\
h_t &= (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t
\end{aligned}
$$

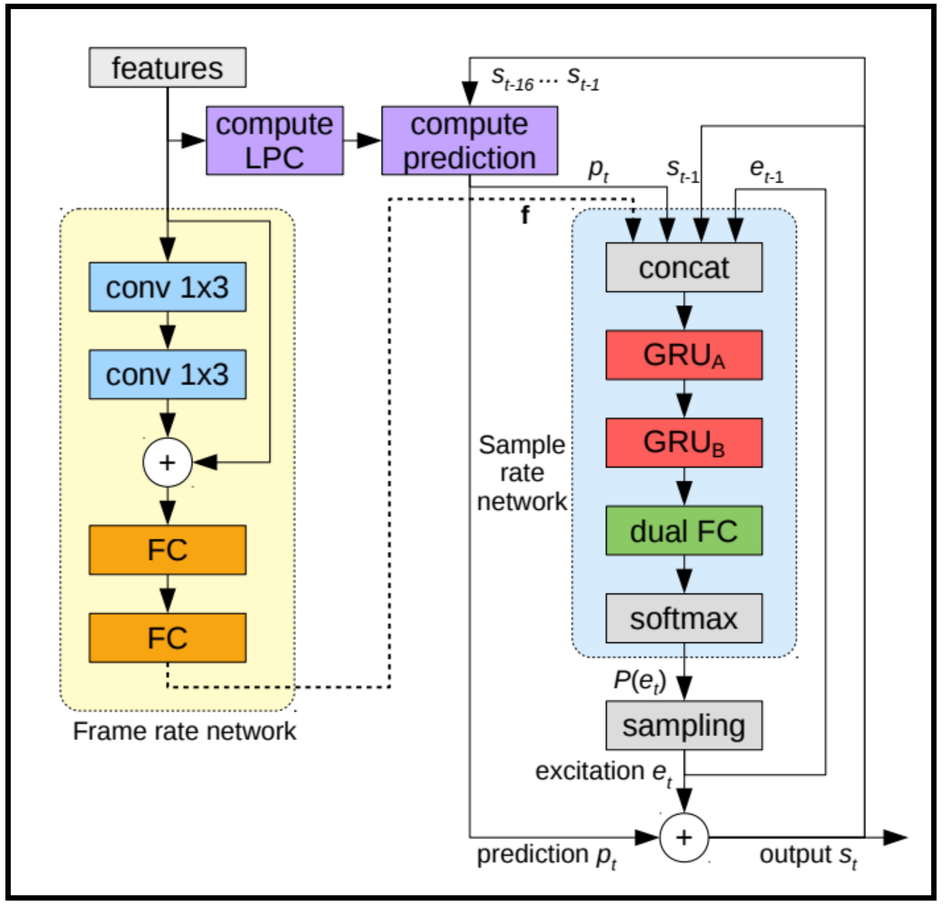
+ conditioning

LPCNet

Идея:
помочь WaveRNN с помощью linear predictive coding

$$x_t = \sum_{p=1}^{P} a_p x_{t-p} + \epsilon_t$$

a_p считаются по формулам
e_t предсказываются сеткой



LPCNet

Как считаются LP коэффициенты:

1) Есть теорема, что автокорреляционная функция - это обратное ффт от квадрата спектральной функции

$$\Psi(\tau) = \int_{-\infty}^{\infty} f(t) f^*(t-\tau) \mathrm{d}t$$

$$\Psi(\tau) \sim \mathrm{Re}\,\mathrm{fft}^{-1}\left(|\mathrm{fft}(\vec{x})|^2\right)$$

2) LP коэффициенты получаются как решения системы линейных уравнений

$$\begin{bmatrix} R_{\hat{n}}(0) & R_{\hat{n}}(1) & . & . & R_{\hat{n}}(p-1) \\ R_{\hat{n}}(1) & R_{\hat{n}}(0) & . & . & R_{\hat{n}}(p-2) \\ . & . & . & . & . \\ . & . & . & . & . \\ R_{\hat{n}}(p-1) & R_{\hat{n}}(p-2) & . & . & R_{\hat{n}}(0) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ . \\ . \\ \alpha_p \end{bmatrix} = \begin{bmatrix} R_{\hat{n}}(1) \\ R_{\hat{n}}(2) \\ . \\ . \\ R_{\hat{n}}(p) \end{bmatrix}$$

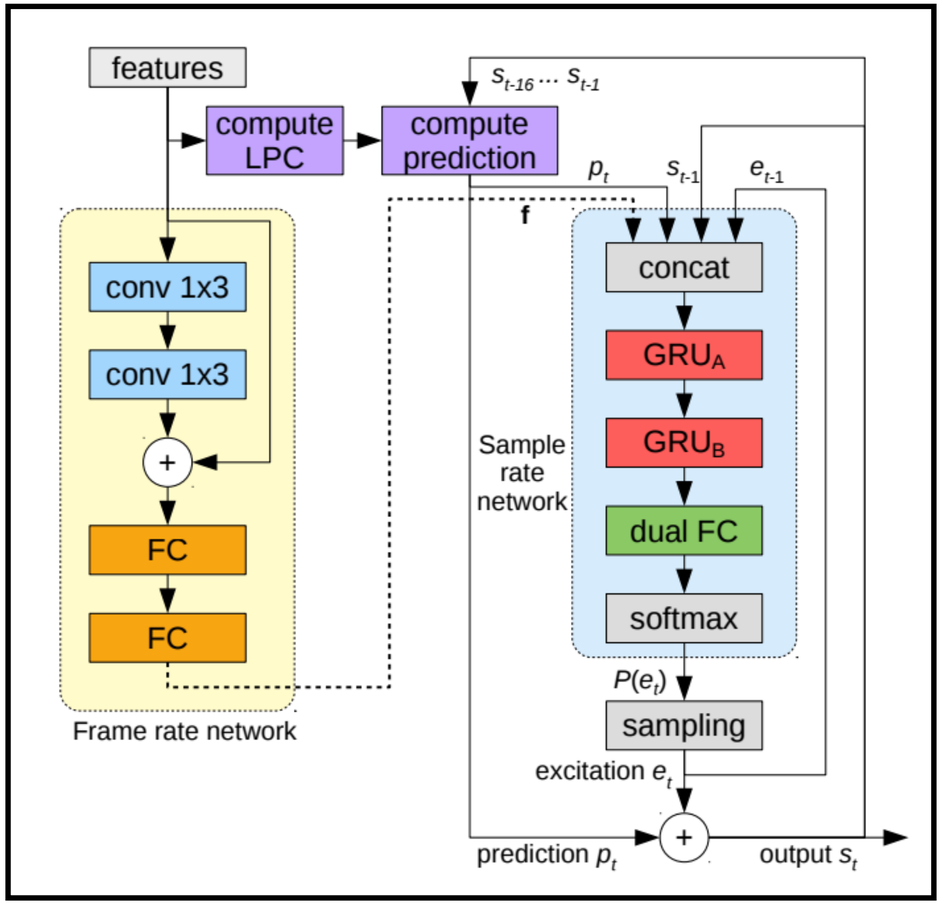
<- это Теплицева матрица - симметричная, с одинаковыми элементами на диагонали, поэтому решение ищется за O(n^2)

LPCNet

RNN input:
- LP предсказание
  (по P значениям сигнала)
- значение сигнала на прошлом шаге
- excitation на прошлом шаге

RNN output:
- excitation на текущем шаге

LPCNet output:
- s_t = LPC(s<t) + E_t



LPCNet

Архитектура nn унаследована от WaveRNN:

$$
\begin{aligned}
\mathbf{u}_t =& \sigma\left(\mathbf{W}_u \mathbf{h}_t + \mathbf{v}_{s_{t-1}}^{(u,s)} + \mathbf{v}_{p_{t-1}}^{(u,p)} + \mathbf{v}_{e_{t-1}}^{(u,e)} + \mathbf{g}^{(u)}\right) \\
\mathbf{r}_t =& \sigma\left(\mathbf{W}_r \mathbf{h}_t + \mathbf{v}_{s_{t-1}}^{(r,s)} + \mathbf{v}_{p_{t-1}}^{(r,p)} + \mathbf{v}_{e_{t-1}}^{(r,e)} + \mathbf{g}^{(r)}\right) \\
\widetilde{\mathbf{h}}_t =& \tanh\left(\mathbf{r}_t \circ \left(\mathbf{W}_h \mathbf{h}_t\right) + \mathbf{v}_{s_{t-1}}^{(h,s)} + \mathbf{v}_{p_{t-1}}^{(h,p)} + \mathbf{v}_{e_{t-1}}^{(h,e)} + \mathbf{g}^{(h)}\right) \\
\mathbf{h}_t =& \mathbf{u}_t \circ \mathbf{h}_{t-1} + \left(1 - \mathbf{u}_t\right) \circ \widetilde{\mathbf{h}}_t \\
P\left(e_t\right) =& \operatorname{softmax}\left(\text{dual\_fc}\left(\operatorname{GRU}_{\mathrm{B}}\left(\mathbf{h}_t\right)\right)\right) ,
\end{aligned}
\tag{5}
$$

- Никаких coarse и fine
- Dual FC одинаковы по виду и предсказывают одно и то же
- GRU_a и GRU_b размером 384 и 16 юнитов вместо 896
- Сложный инференс (из-за voiced и unvoiced звука)
- Спарсификация

GAN-based вокодеры

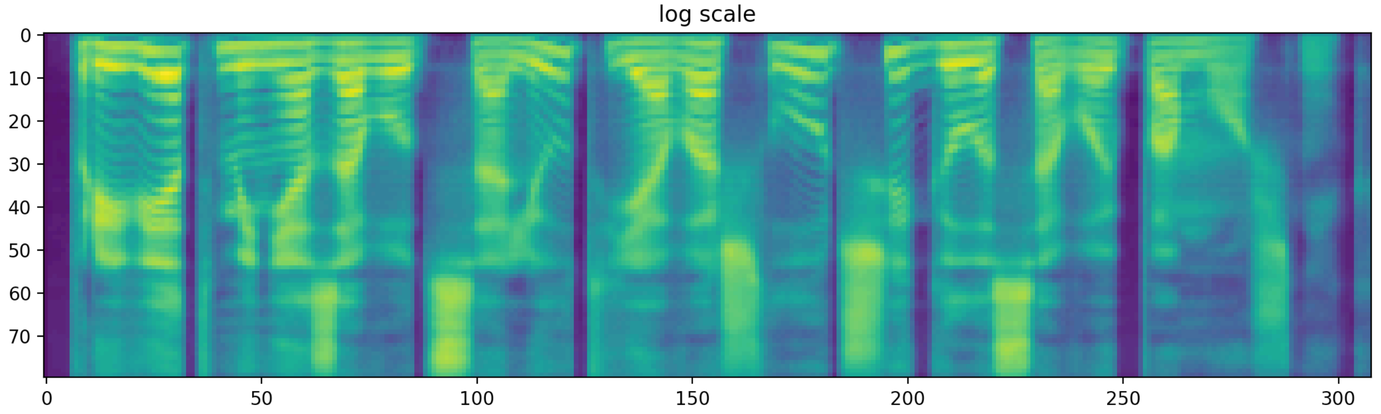
Adversarial vocoder

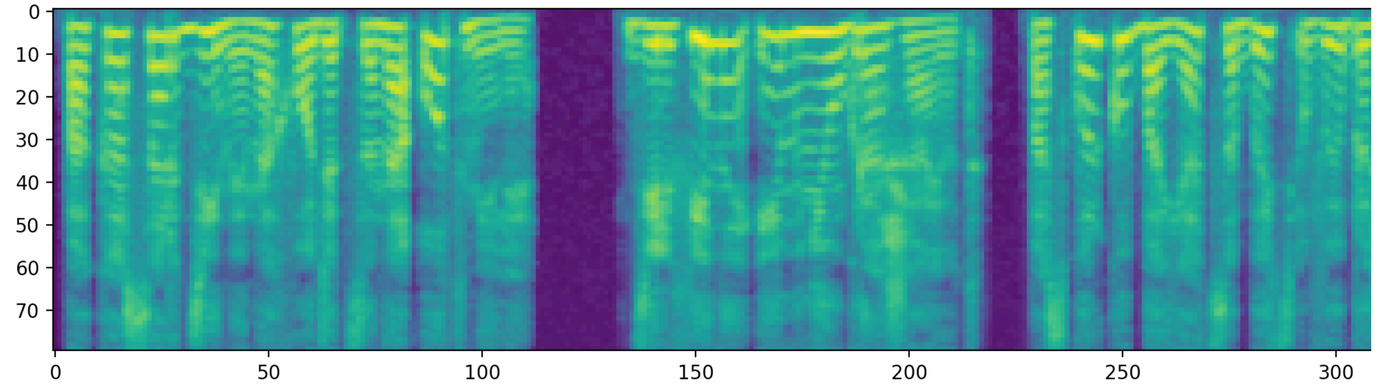
Синтезированные спектрограммы всегда слишком сглажены (из-за MSE лосса)

Схема adversarial vocoding:

1) x = synthesized mel-spec
2) nn(x) - обычная спектрограмма
3) GL-алгоритм

Здесь nn - это генератор в GAN





Остальные GAN-вокодеры

Идея:
Генератор синтезирует вейвформу
Дискриминатор пытается определить, настоящая ли это вейформа

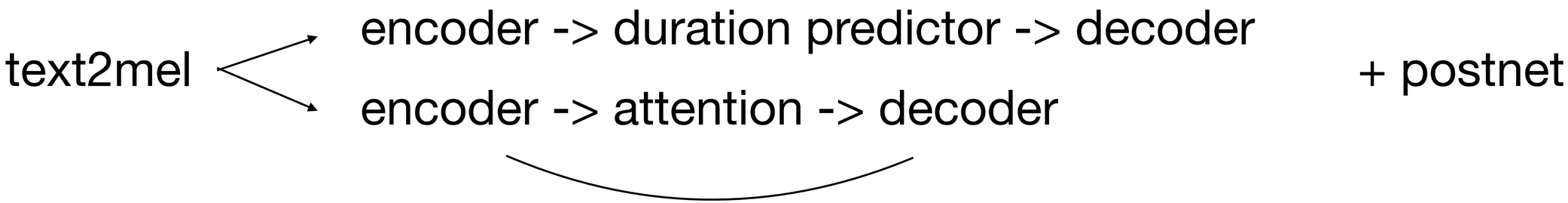
Примеры:

- WaveGAN: применили DCGAN к звуку
- MelGAN: добавили multiscale D и feature map loss
- StyleMelGan: G и D посложнее
- HiFiGAN: G и multiscale D + MSE от спектрограмм
- Multiband MelGAN: multiscale MSE от спектрограмм
- VocGAN: Сложный multiscale G и D + MSE от спектрограмм
- Parallel WaveGAN: G - кусок WaveNet

Акустические модели синтеза (авторегрессионные)

Параметрический синтез

Текст -> мел-спектрограмма -> вокодер



Tacotron

Возможно первая seq2seq + attention модель для синтеза

TACOTRON: TOWARDS END-TO-END SPEECH SYNTHESIS

Yuxuan Wang*, RJ Skerry-Ryan*, Daisy Stanton, Yonghui Wu, Ron J. Weiss†, Navdeep Jaitly,

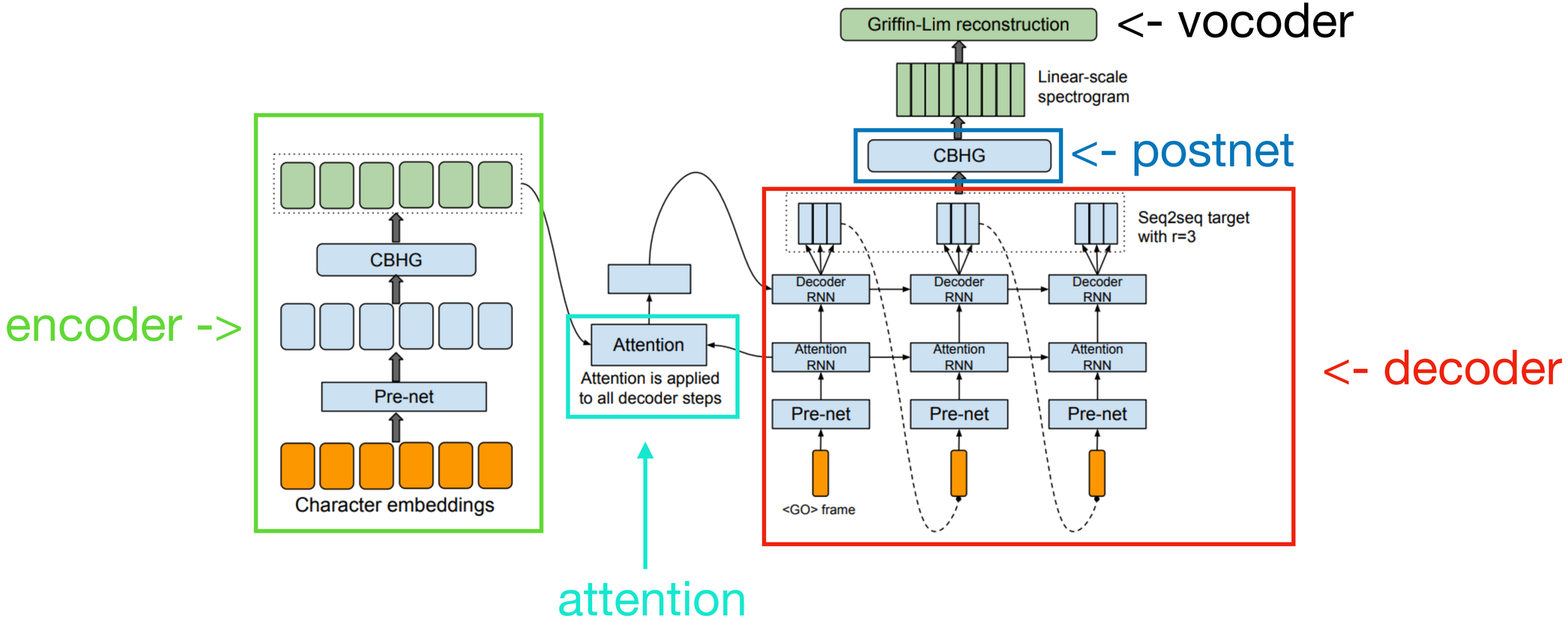
Zongheng Yang, Ying Xiao*, Zhifeng Chen, Samy Bengio†, Quoc Le, Yannis Agiomyrgiannakis,

Rob Clark, Rif A. Saurous*

Google, Inc.
{yxwang,rjryan,rif}@google.com

*These authors really like tacos.
†These authors would prefer sushi.

Tacotron

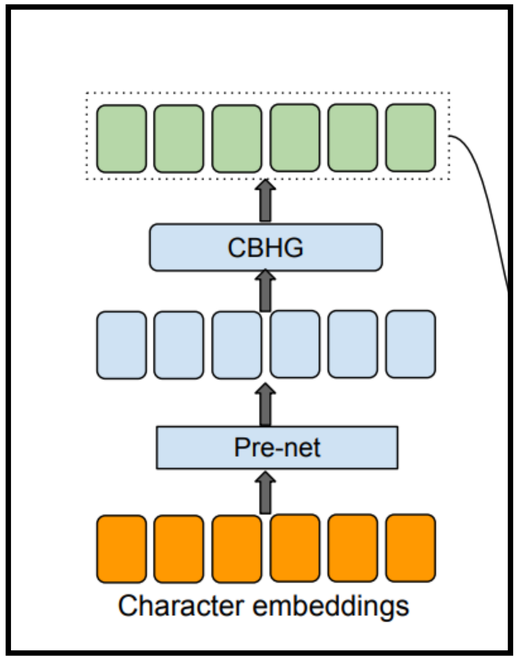

Griffin-Lim reconstruction
<- vocoder
Linear-scale spectrogram
CBHG
<- postnet
Seq2seq target with r=3
Decoder RNN
Attention RNN
Pre-net
<GO> frame
Attention
Attention is applied to all decoder steps
<- decoder
encoder ->
CBHG
Pre-net
Character embeddings
attention

Tacotron

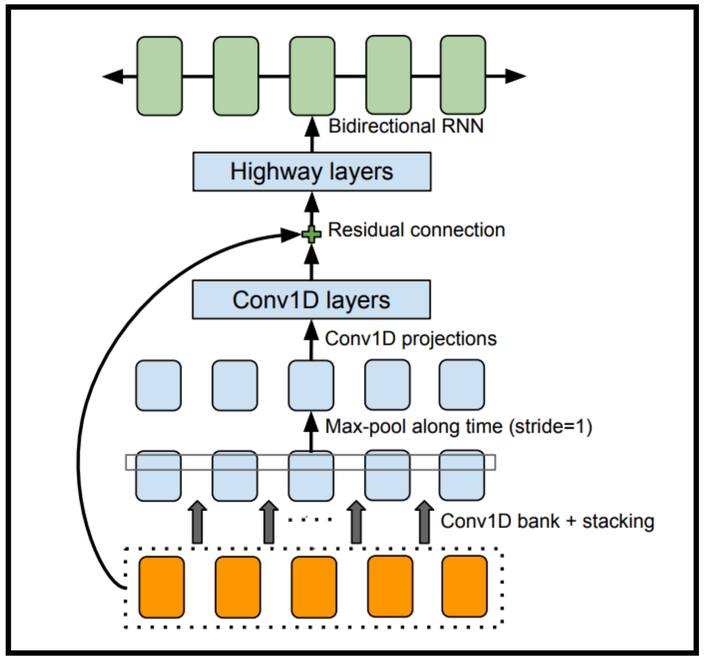
Задачи:

- получить многомерное представление букв (фонем)
- обогатить буквы контекстом (соседние слова и буквы)
- протащить градиенты до букв

Encoder:



CBHG module



batch_size x num_letters x encoder_shape

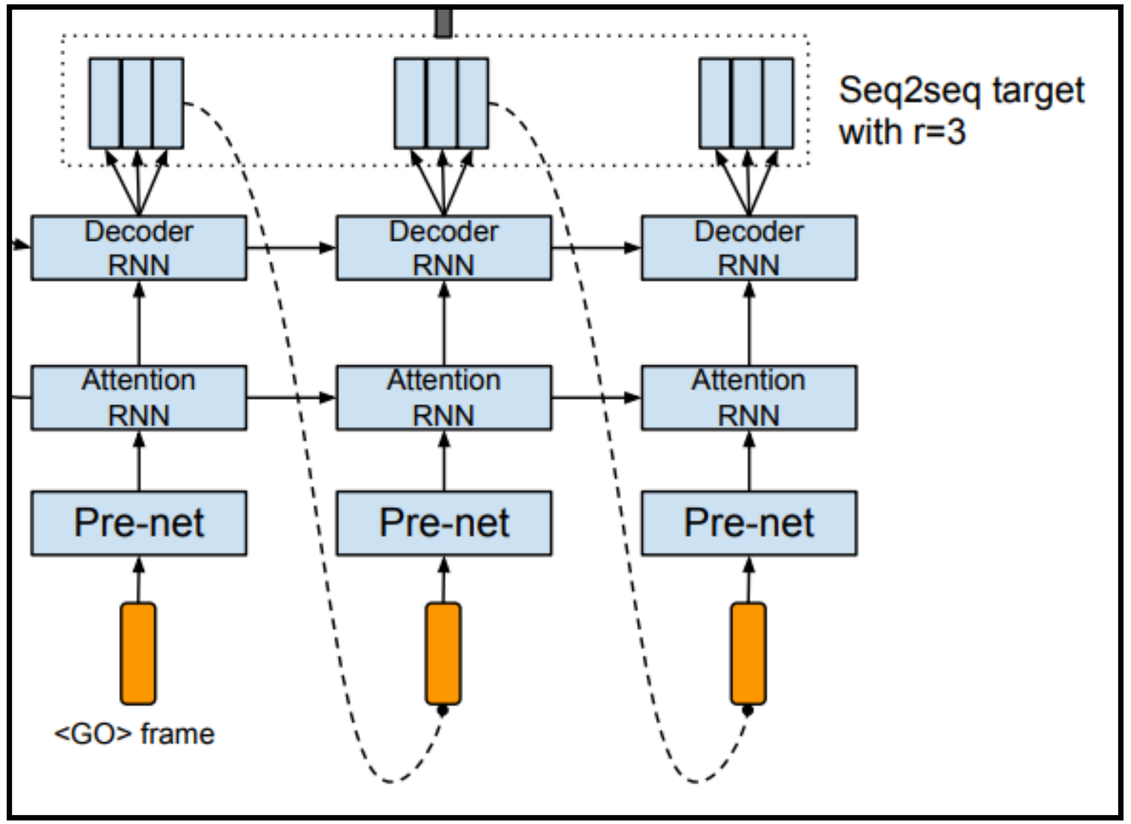
Tacotron

Процесс декодирования:

На шаге t:

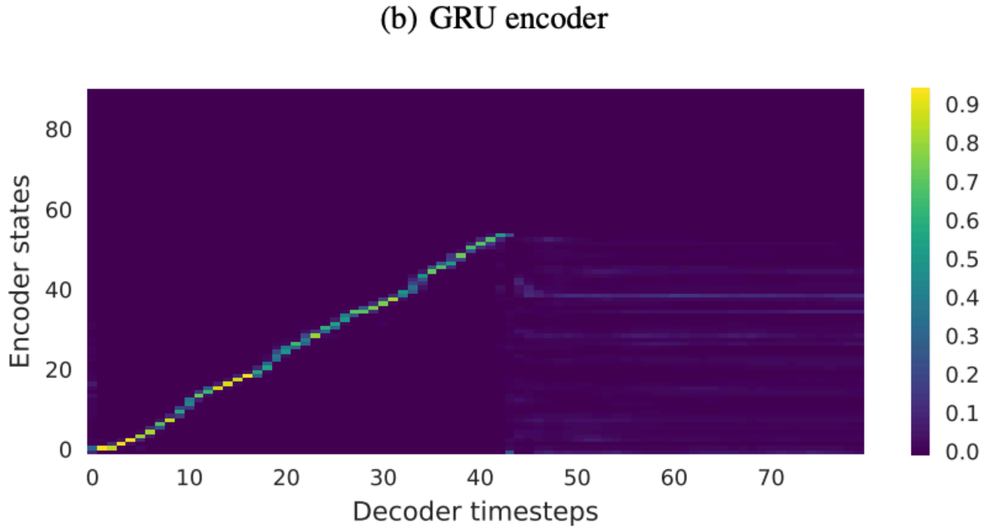
1. Attention rnn взвешивает выход энкодера (какую букву сейчас синтезируем)
2. Фрейм gt спектрограммы с шага t-1 прогоняем через bottleneck (prenet)
3. Конкатим контекст от attention и выход prenet, обновляем DecoderRNN
4. Предсказываем r фреймов (target)

Decoder



Tacotron

Что делает attention:



Зачем нужен postnet: