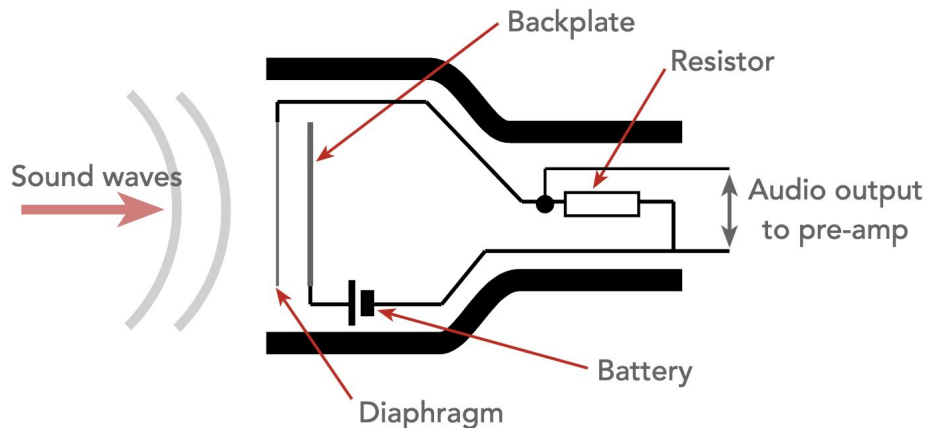
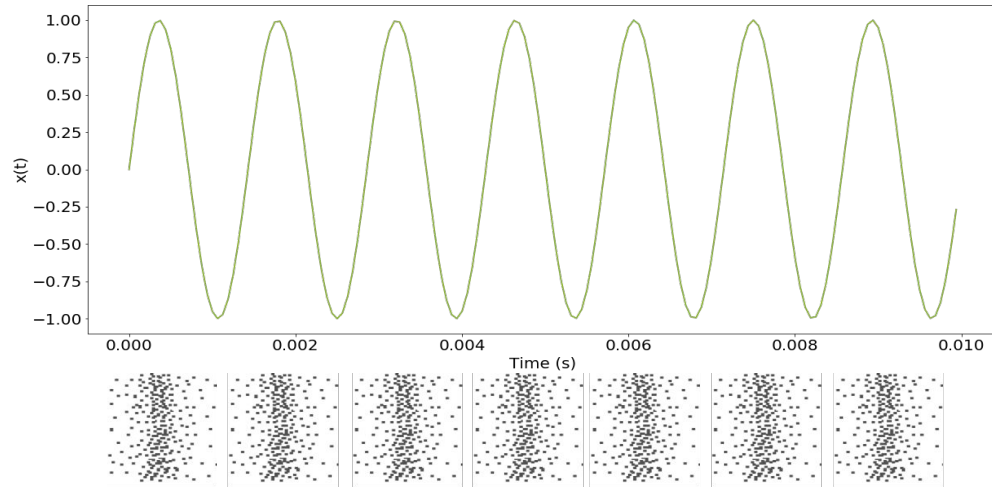


Распознавание речи

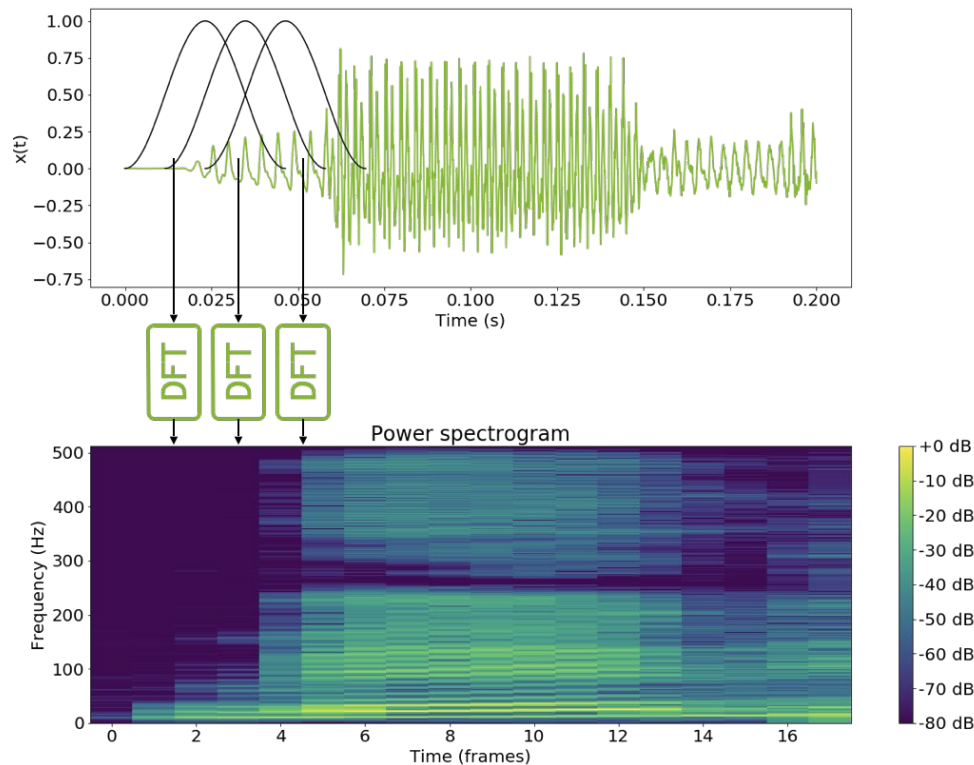
Что такое звук?

- **Звук** – это колебания воздуха, то есть серия возрастающих и падающих значений давления воздуха
- **Микрофон** улавливает эти колебания воздуха и преобразовывает их в электрические колебания
- Эти колебания преобразуются в **аналоговый** сигнал, а затем и в **цифровой** сигнал



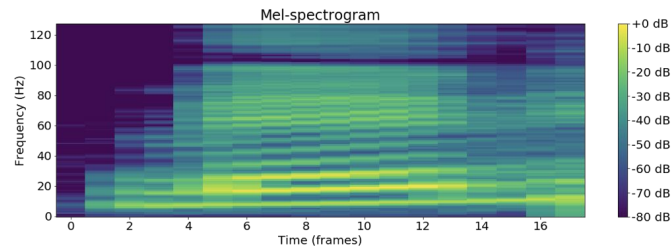
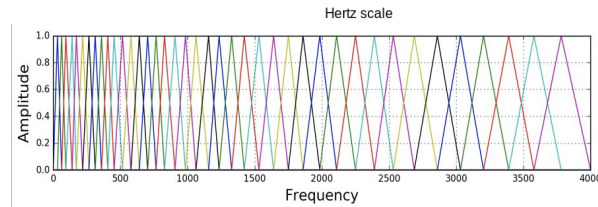
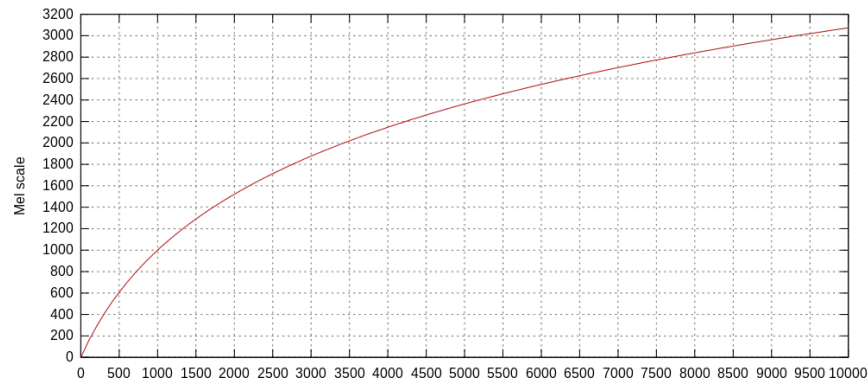
Спектрограмма (та самая)

- Нарезаем сигнал на **окна с пересечением**
- Применяем **оконную функцию** к вырезанному окну
- Применяем дискретное преобразование **Фурье**
- Считаем **квадрат комплексной нормы**
- Берем **половину вектора** в силу его симметричности

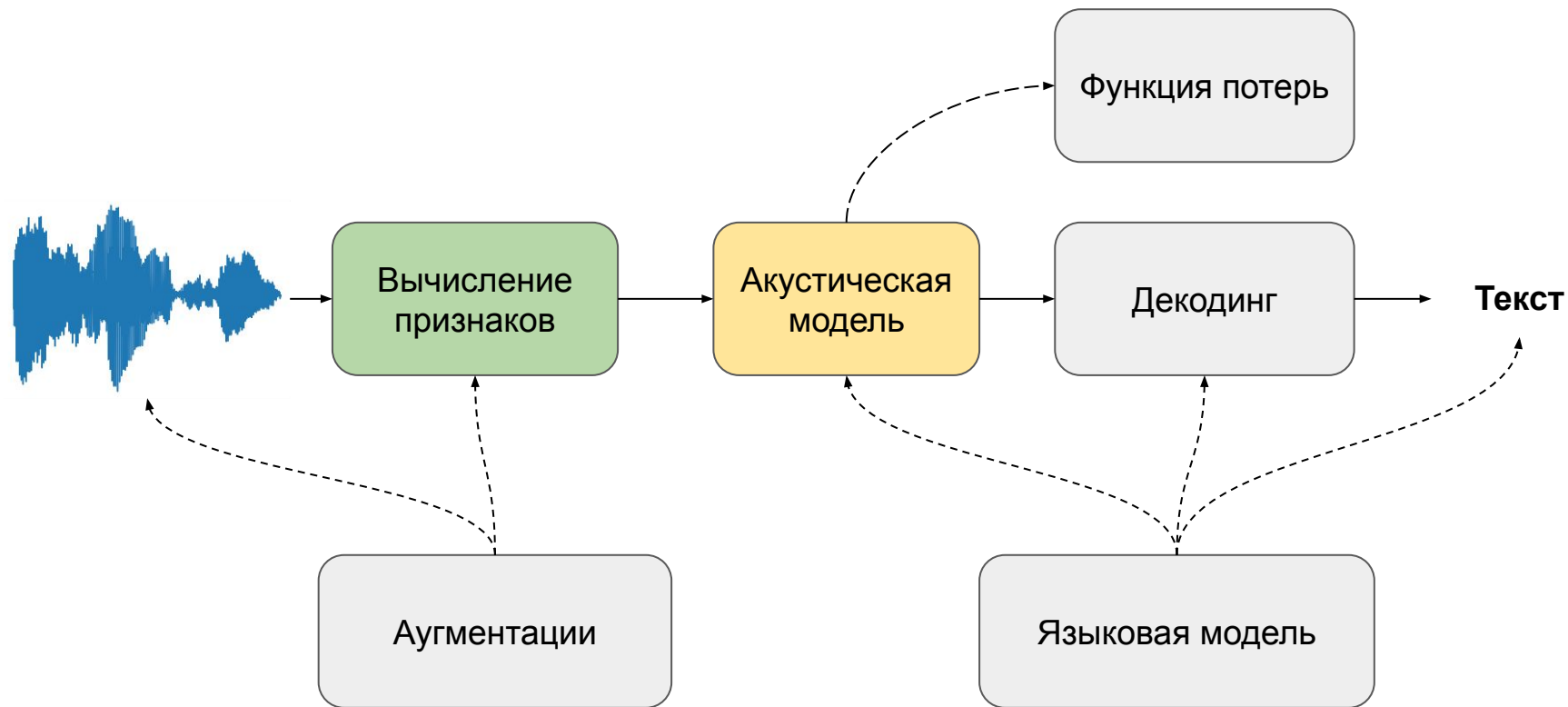


МелСпектрограмма

- Человеческое ухо хорошо слышит **низкие** частоты и плохо **высокие**
- Учитываем в спектрограмме **низкие** частоты больше, а **высокие** меньше
- $m = 2595 \log_{10} \left(1 + \frac{f}{700} \right) = 1127 \ln \left(1 + \frac{f}{700} \right)$
 $f = 700 \left(10^{\frac{m}{2595}} - 1 \right) = 700 \left(e^{\frac{m}{1127}} - 1 \right)$
- В конце берем **логарифм** от мелспектрограммы



Распознавание голоса в 2к21

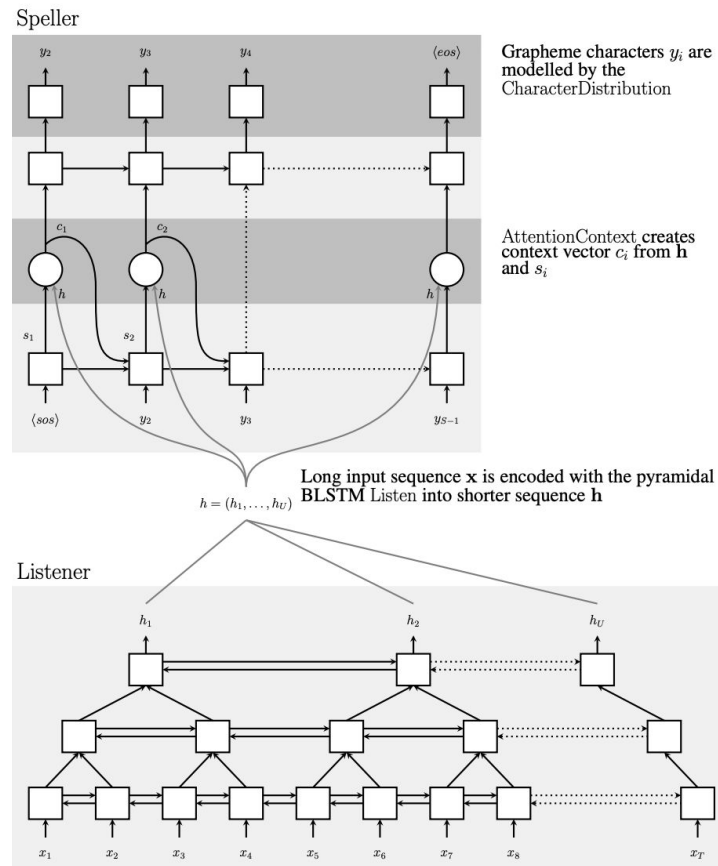


Первым делом метрики!

- Word Error Rate (WER) -- доля “неправильных” слов
- $$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}$$
- **S** – кол-во замен
- **D** – кол-во удалений
- **I** – кол-во вставок
- **C** – кол-во совпадений
- **N – S + D + C** – кол-во слов всего
- - True: quick **brown** fox jumped over **a** lazy dog
 - Pred: quick **brow** **an** fox jumped over lazy dog

Listen, Attend & Spell

- Базовая старая, но **хорошая** модель. seq2seq с механизмов внимания для выравнивания
- **Listener** - пирамидальный bi-LSTM энкодер
- **Speller** - обычный декодер :)
- **Attend** - внимание со скалярным произведением в качестве функции близости

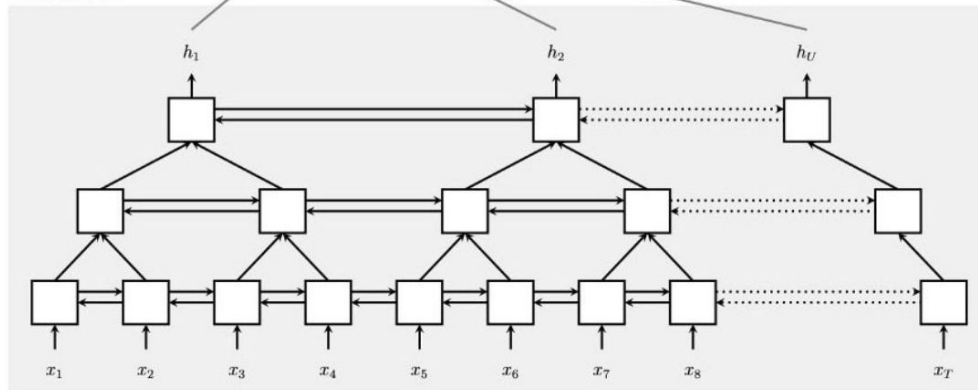


Listener

- На вход батч из **МелСпектрограм**
- Конкатенируем **соседним активации** в одну и передаем выше
- Уменьшаем временную размерность в **4-8 раз**

acoustic model encoder $h = (h_1, \dots, h_U)$ — dense representation of x

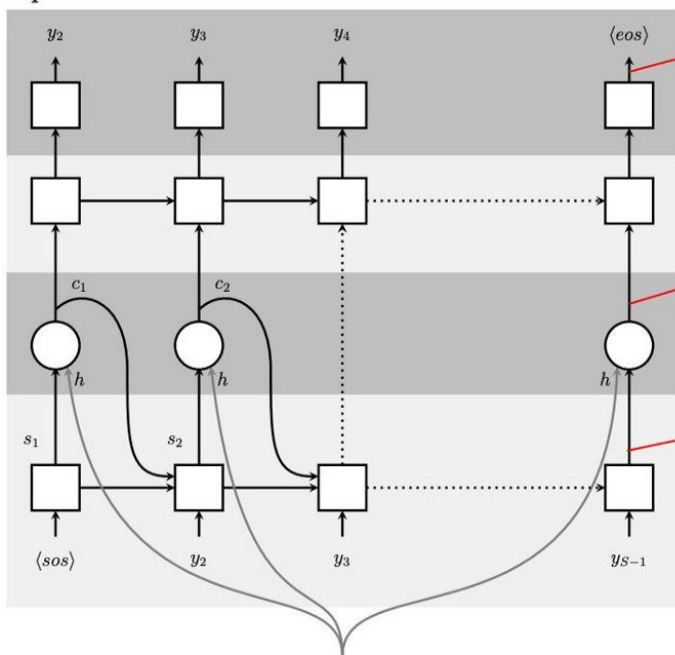
Listener



input sequence of filter bank spectra features

Attend & Spell

Speller



$$\mathbf{h} = (h_1, \dots, h_U)$$

dot product

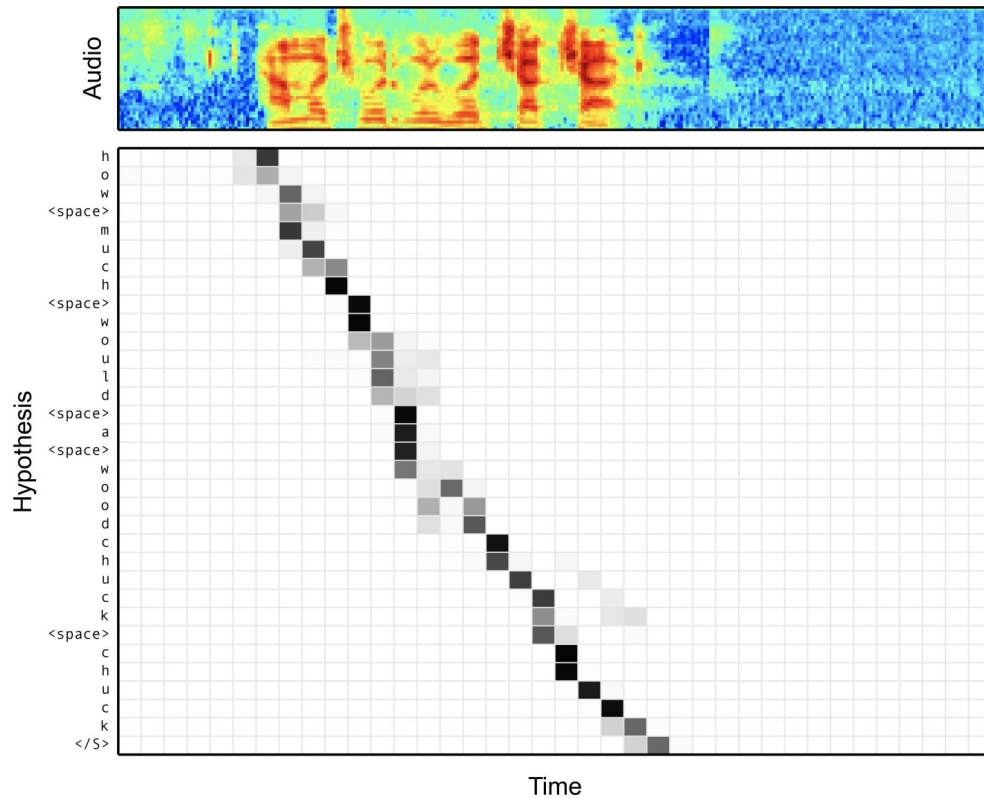
$$e_{i,u} = \langle \phi(s_i), \psi(h_u) \rangle$$

$$\alpha_{i,u} = \frac{\exp(e_{i,u})}{\sum_u \exp(e_{i,u})}$$

$$c_i = \sum_u \alpha_{i,u} h_u$$

decoder state

Attention Visualization

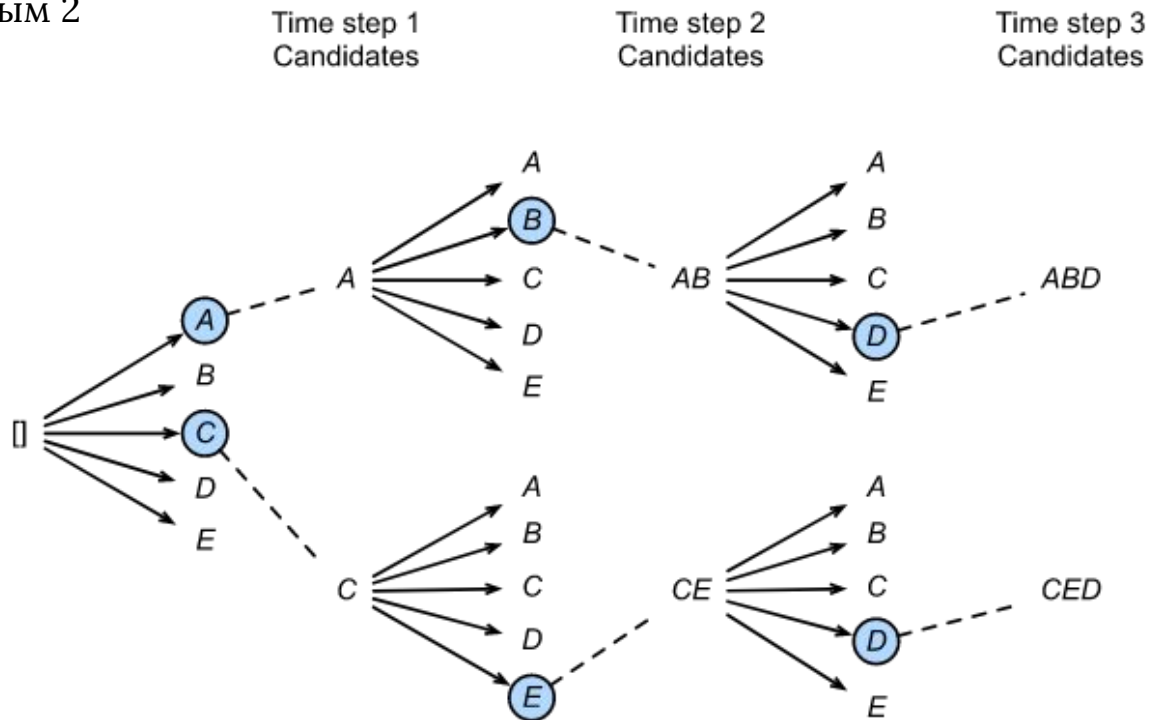


Несколько деталей о LAS

- Минимизируем кросс-энтропию между предсказаниями (**категориальное распределение**) и правильными буквами (**one-hot**)
- Учись с помощью **teacher-forcing**-а
- Декодируем с помощью **beam-searching**-а
- Добавляем **LM** для улучшения качества

Что такое beam-searching?

Для простоты фиксируем размер словаря равным 5 и размер beam-а равным 2

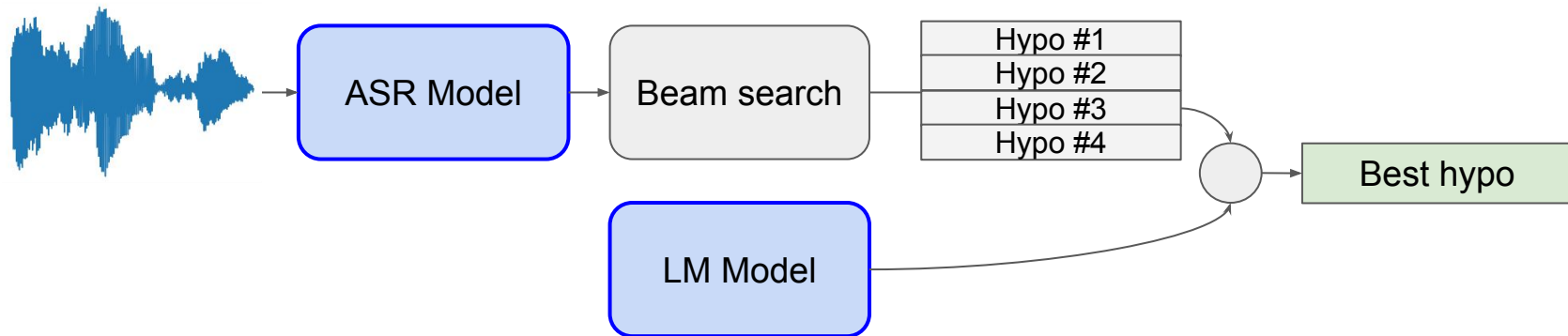


Что такое LM fusing?

Second-pass rescoring

Обучаем ASR и LM отдельно. Пересчитываем вероятности гипотез с помощью LM

$$P_{\text{final}} = P_{\text{ASR}}(Y \mid X) + \alpha \cdot P_{\text{LM}}(Y) + \beta \cdot \text{len}(Y)$$

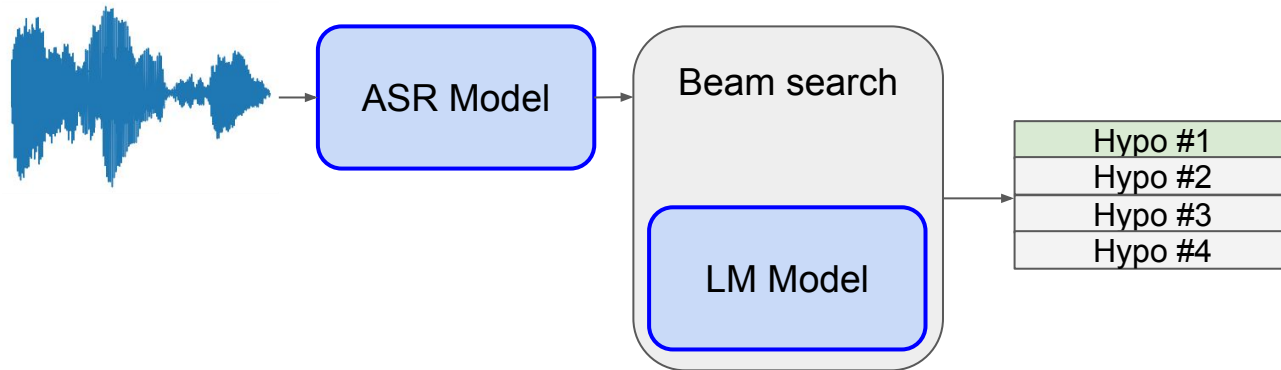


Что такое LM fusing?

Shallow fusing

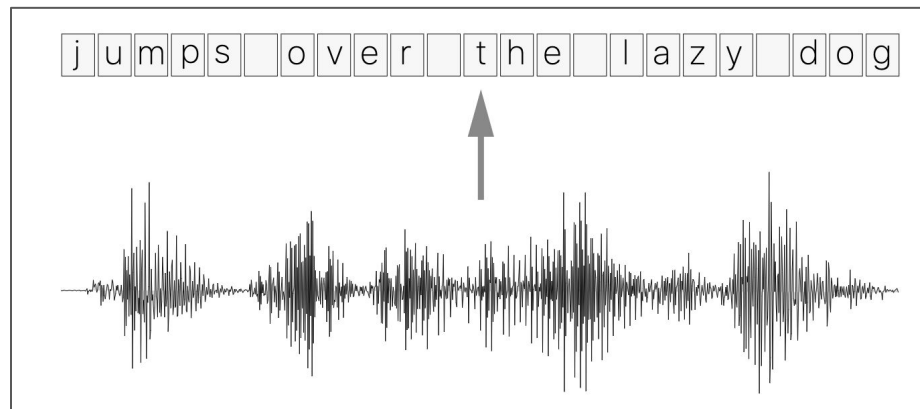
Обучаем ASR и LM отдельно. Добавляем LM на каждый шаг BM

$$P_{\text{final}} = P_{\text{ASR}}(Y \mid X) + \alpha \cdot P_{\text{LM}}(Y) + \beta \cdot \text{len}(Y)$$



Connectionist Temporal Classification

- Вход и выход разной длины и они не выровнены
- Хочется уметь считать $P(Y | X)$ для любой длины $|X|$ и $|Y|$
- $\operatorname{argmax} P(Y | X)$



Connectionist Temporal Classification

- Для каждого фрейма делаем предсказание над словарем (например, букв)
- Склеиваем соседние одинаковые предсказания
- Возникают проблемы с тишиной между словами и повторяющимися символами

x_1 x_2 x_3 x_4 x_5 x_6

input (X)

c c a a a t

alignment

c a t

output (Y)

Connectionist Temporal Classification

h h e ϵ ϵ l l l ϵ l l o

h e ϵ l ϵ l o

h e l l o

h e l l o

First, merge repeat characters.

Then, remove any ϵ tokens.

The remaining characters are the output.

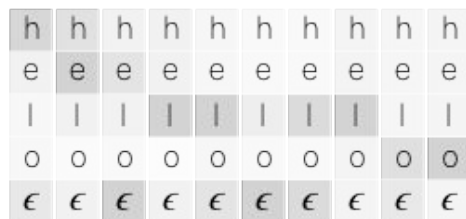
Connectionist Temporal Classification



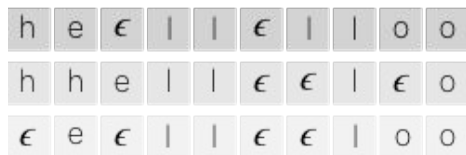
We start with an input sequence, like a spectrogram of audio.



The input is fed into an RNN, for example.



The network gives $p_t(a | X)$, a distribution over the outputs {h, e, l, o, c} for each input step.



With the per time-step output distribution, we compute the probability of different sequence:



By marginalizing over alignments, we get a distribution over outputs:

Connectionist Temporal Classification

$$p(Y \mid X) = \sum_{A \in \mathcal{A}_{X,Y}} \prod_{t=1}^T p_t(a_t \mid X)$$

The CTC conditional
probability

marginalizes over the
set of valid alignments

computing the **probability** for a
single alignment step-by-step.

Но как считать и пробрасывать градиенты через СТС?



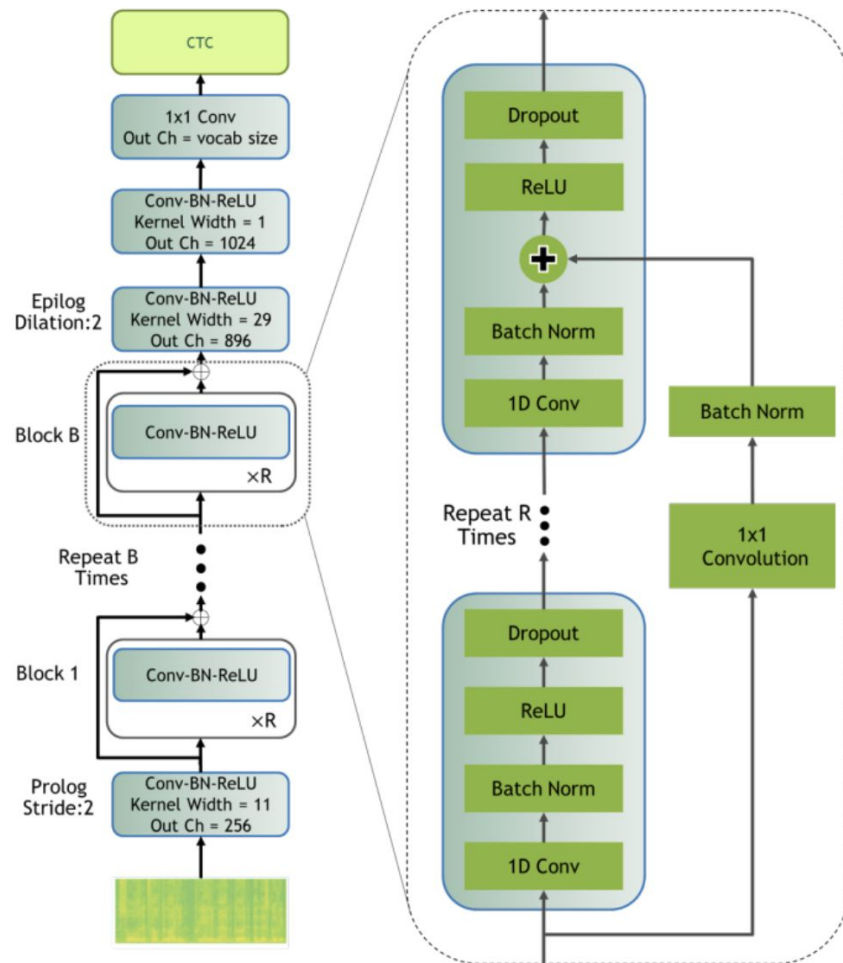
Connectionist Temporal Classification

Динамическое
программирование



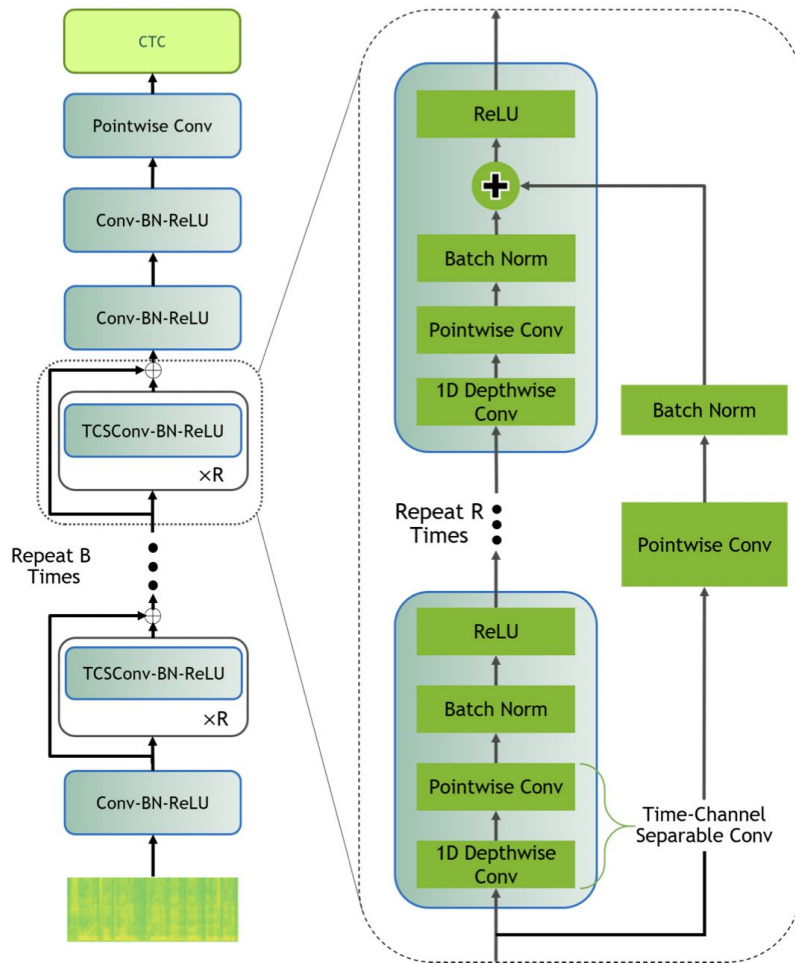
Jasper

- 1D convolution, BatchNorm, ReLU, and Dropout
- Residual connections
- Функция потерь CTC
- Уменьшаем временную размерность в 4 раза
- Очень быстрый и дешевый инференс



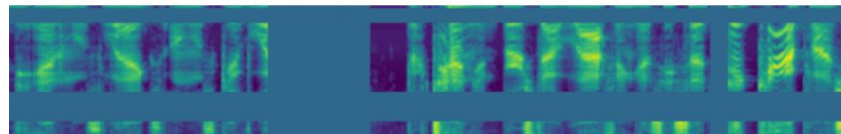
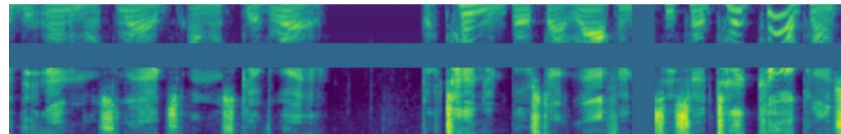
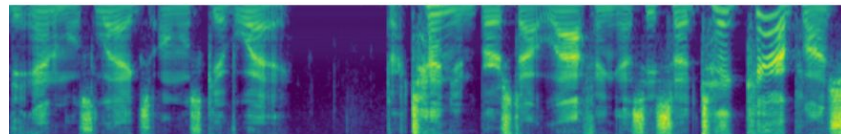
QuarzNet

- Точно такая же архитектура
- Наличие Time-Channel Separable сверток



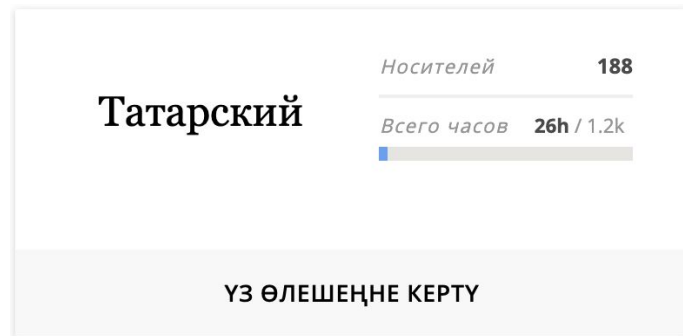
Какие бывают аугментации?

- Аугментации наше все
- Помогают с нехваткой данных и улучшением робастности
- Можно применять либо к звуковому сигналу, либо с уже МелСпектрограмме
- Добавляем шум, меняем громкость, скорость, частоту, акустику...
- Маскируем частоты или время в МелСпектрограмме



Что на счет открытых данных?

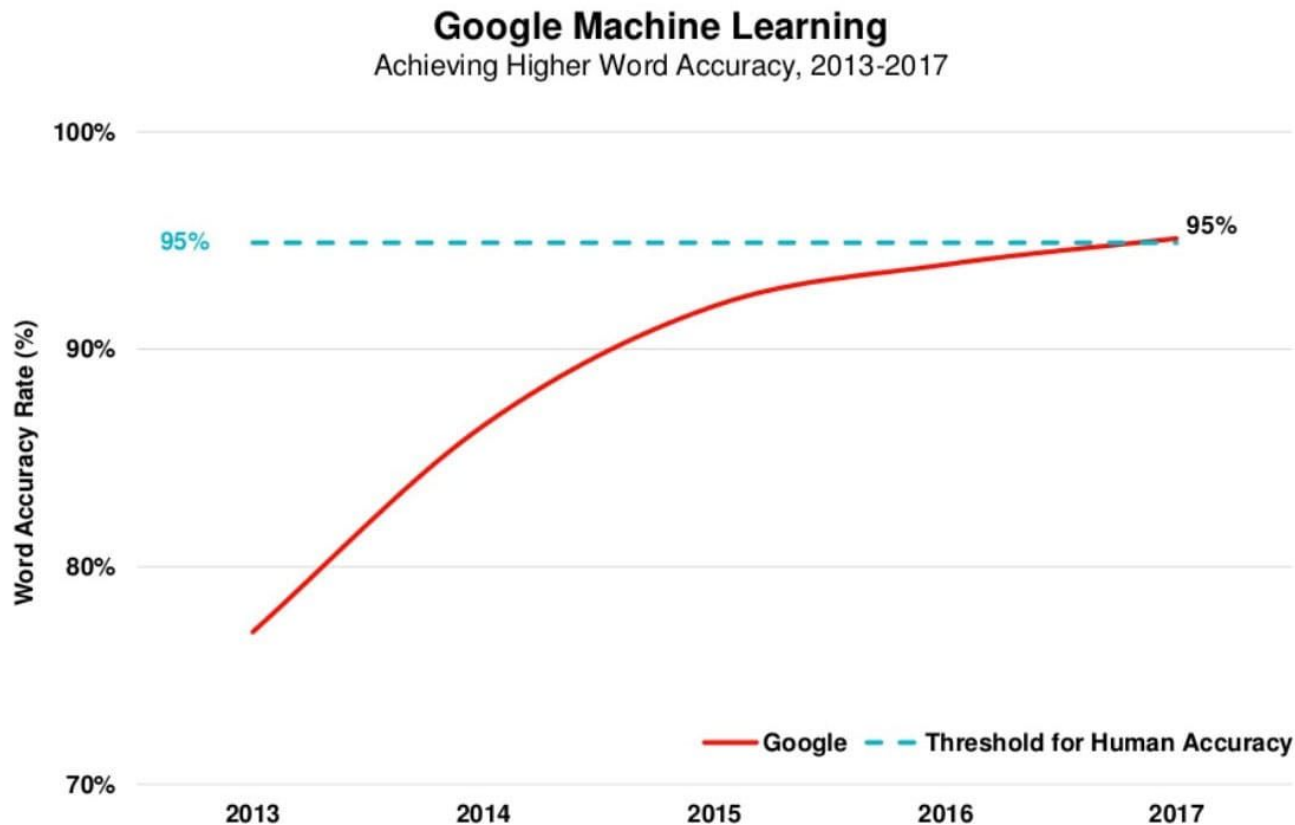
- **LibriSpeech.**
~1000 часов английской речи
- **CommonVoice**
Десятки языков и тысячи часов
- **VoxCeleb**
Тысячи голосов
- **И еще много чего**
<https://tinyurl.com/swker6w6>



А как разметать данные?



В целом это решенная задача :)



Вопросы?