



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

БПМИ-191:
Абрамов Арсений
Оганисян Ваге
Асланов Алишер

Глубинное Обучение для Работы со Звуком

Москва, 2021



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

БПМИ-191:
Абрамов Арсений
Оганнисян Ваге
Асланов Алишер

ПРЕДСТАВЛЕНИЕ ЗВУКА

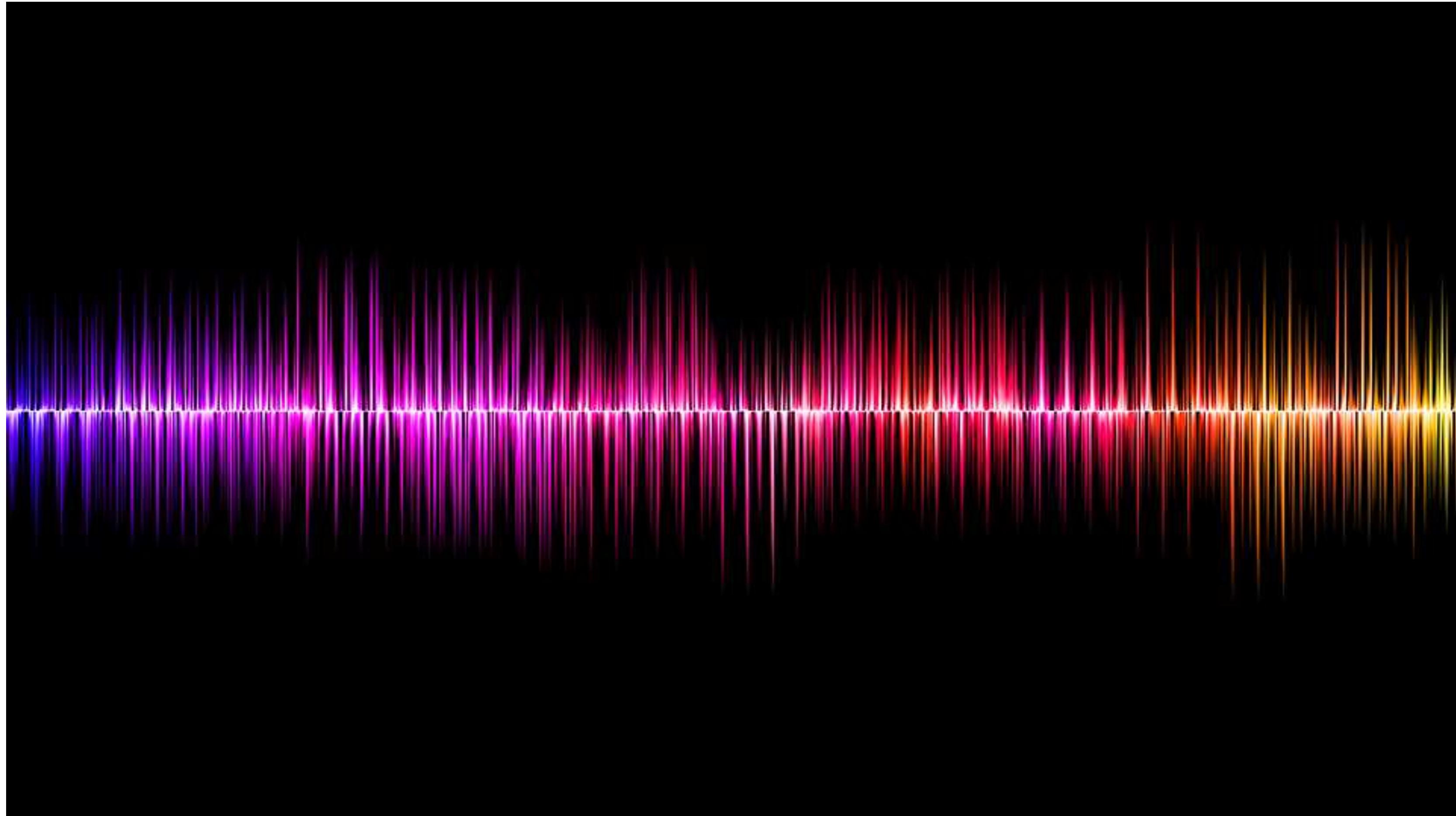
Москва, 2021



Что такое Звук?

БПМИ-191, Абрамов Арсений

Звук – волна. Характеризуется амплитудой и частотой.

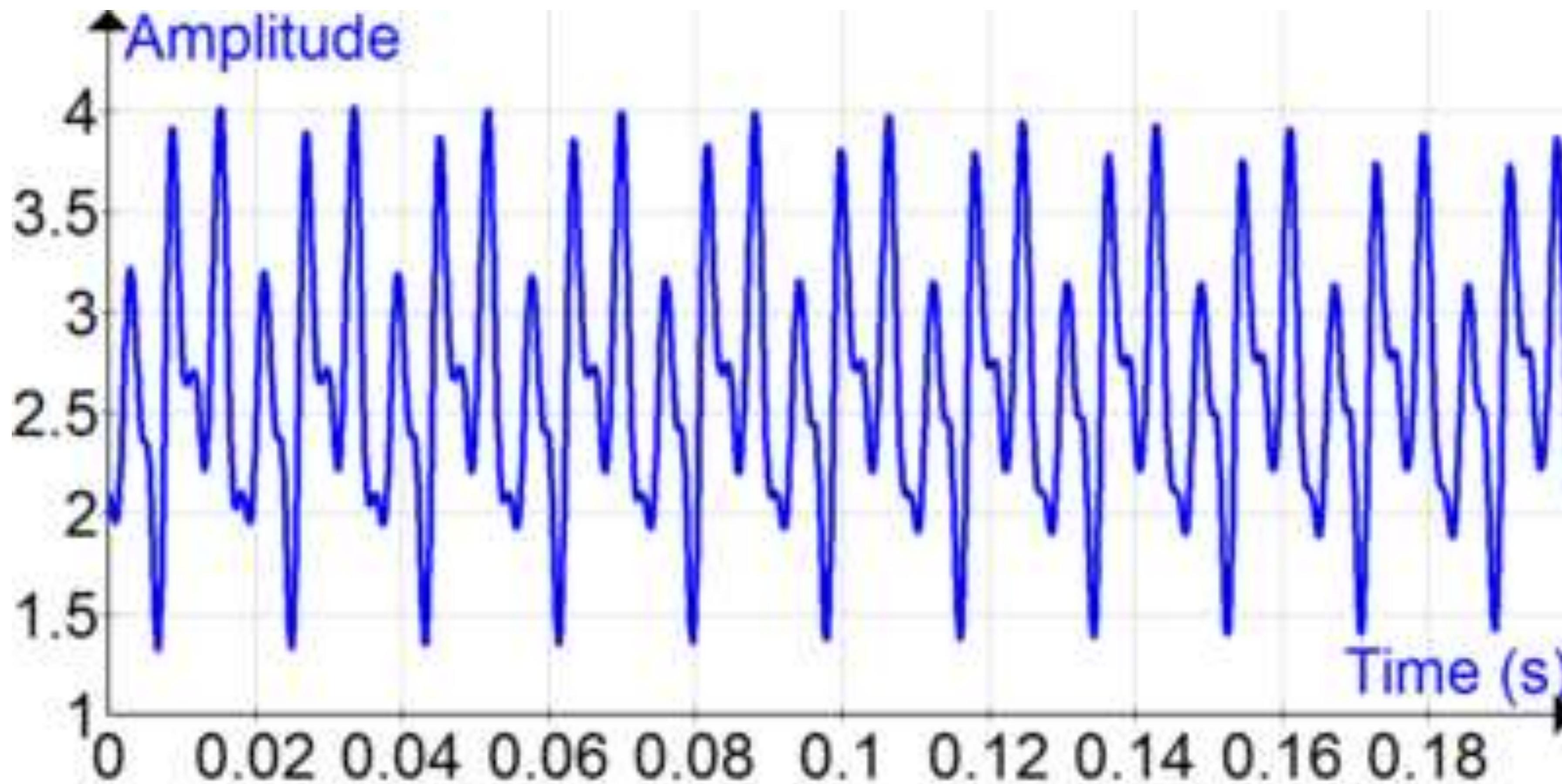




Что такое Звук?

БПМИ-191, Абрамов Арсений

Звук моделируется временными рядами.

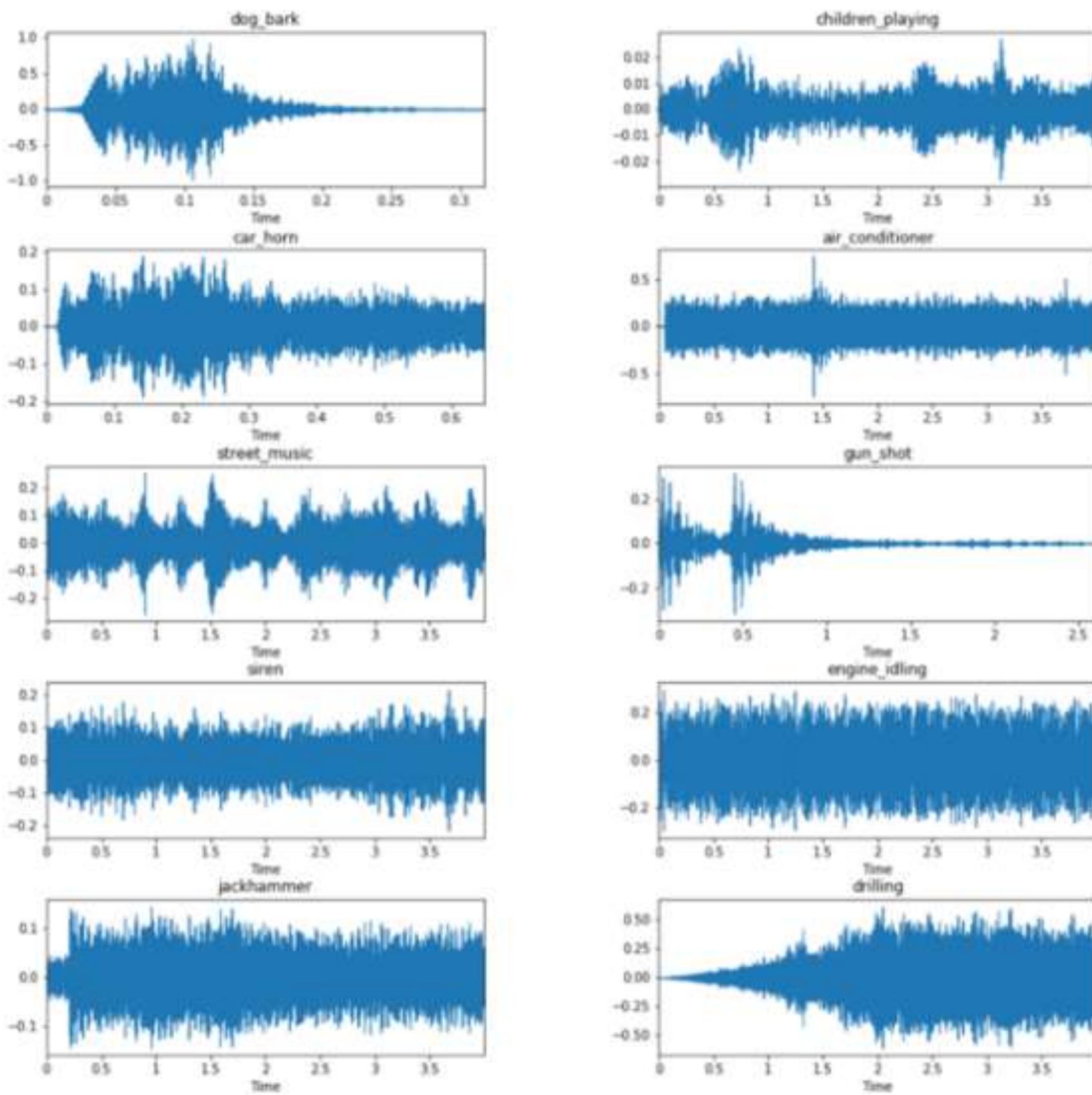




Что такое Звук?

БПМИ-191, Абрамов Арсений

Звук – это НЕструктурированные данные.





Что такое Звук?

БПМИ-191, Абрамов Арсений

Структурированные данные.

In [4]:	data3							
Timestamp	ID	Рейтинг	Группа (в формате 182)	МИ?	Осенний курс по выбору, приоритет 1	Осенний курс по выбору, приоритет 2	Осенний курс по выбору, приоритет 3	Весенний курс по выбору, приоритет 1
0 2020-05-15 01:12:50.543	93ff79a51cd602f1dd3028ba2c129503	704,0	181	NaN	Язык SQL	Высокопроизводительные вычисления	Матричные вычисления	Дискретная оптимизация
1 2020-05-15 02:46:48.066	26b01b1c4cd5656bab18d24c548834fb	646,0	181	NaN	Высокопроизводительные вычисления	Безопасность компьютерных систем	Язык SQL	Дискретная оптимизация
2 2020-05-15 03:12:41.480	30f3653fc176d54e89ac3179c455c6dd	624,0	185	NaN	Безопасность компьютерных систем	Матричные вычисления	Моделирование временных рядов	Дискретная оптимизация
3 2020-05-15 04:43:08.994	1528f0eaa027580820ccfd92a53ad68	579,0	182	NaN	Statistical Learning Theory	Высокопроизводительные вычисления	Матричные вычисления	Дискретная оптимизация
4 2020-05-15 07:47:17.197	496ea4f0d4abe264b1bb1b80eb3830c5	632,0	183	NaN	Высокопроизводительные вычисления	Безопасность компьютерных систем	Теория баз данных	Компьютерные сети
...
218 2020-05-20 11:49:38.801	bd416140ecdb32b6dbd7f40820bf63b1	517,0	185	NaN	Безопасность компьютерных систем	Теория баз данных	Язык SQL	Машинное обучение 2
219 2020-05-20 11:57:07.326	e6f5eb76b34e7ab7bac753e6cb0a2279	634,0	184	NaN	Теория баз данных	Безопасность компьютерных систем	Моделирование временных рядов	Компьютерные сети
220 2020-05-21 16:33:15.899	1341f488fae5f1ccf164960fd6506cd0	584,0	188	NaN	Язык SQL	Теория баз данных	Безопасность компьютерных систем	Промышленное программирование на языке Java
221 2020-05-24 01:21:31.946	04bcb5c9d23813ffa940e1febb27fad	646,0	186	NaN	Теория баз данных	Высокопроизводительные вычисления	Безопасность компьютерных систем	Компьютерные сети
222 2020-05-19 04:24:29.000	9d19b0d5f4fc8d7edc2258406f872c4e	694,0	188	NaN	Безопасность компьютерных систем	Язык SQL	Высокопроизводительные вычисления	Компьютерные сети

223 rows × 12 columns

Неструктурированные данные.

Студент с ID 93ff79a51cd602f1dd3028ba2c129503 учится в группе 181, не на специализации МИ. Выбрал такие-то курсы по выбору, имеет, кстати, рейтинг 704.

Что же касается студента с ID, у него рейтинг ниже – 632, но выбрал другие курсы – такие-то. А ещё он учится в другой группе – 183.

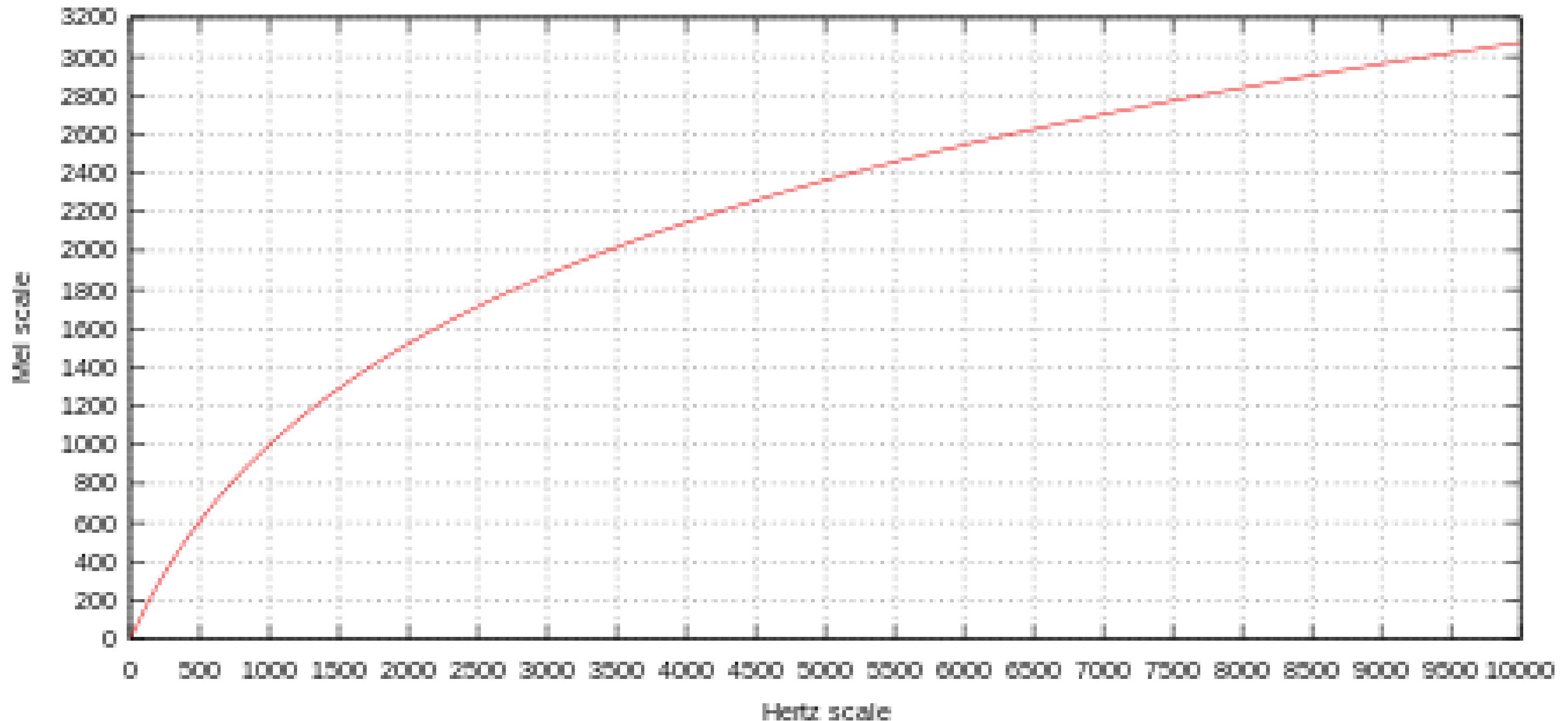


ПРЕДСТАВЛЕНИЕ ЗВУКА

БПМИ-191, Абрамов Арсений

Мел шкала.

Эмпирическое приближение зависимости между воспринимаемой человеком частотой звука и его высотой.



$$m = 2595 \log_{10} \left(1 + \frac{f}{700} \right)$$

Герц

Мел

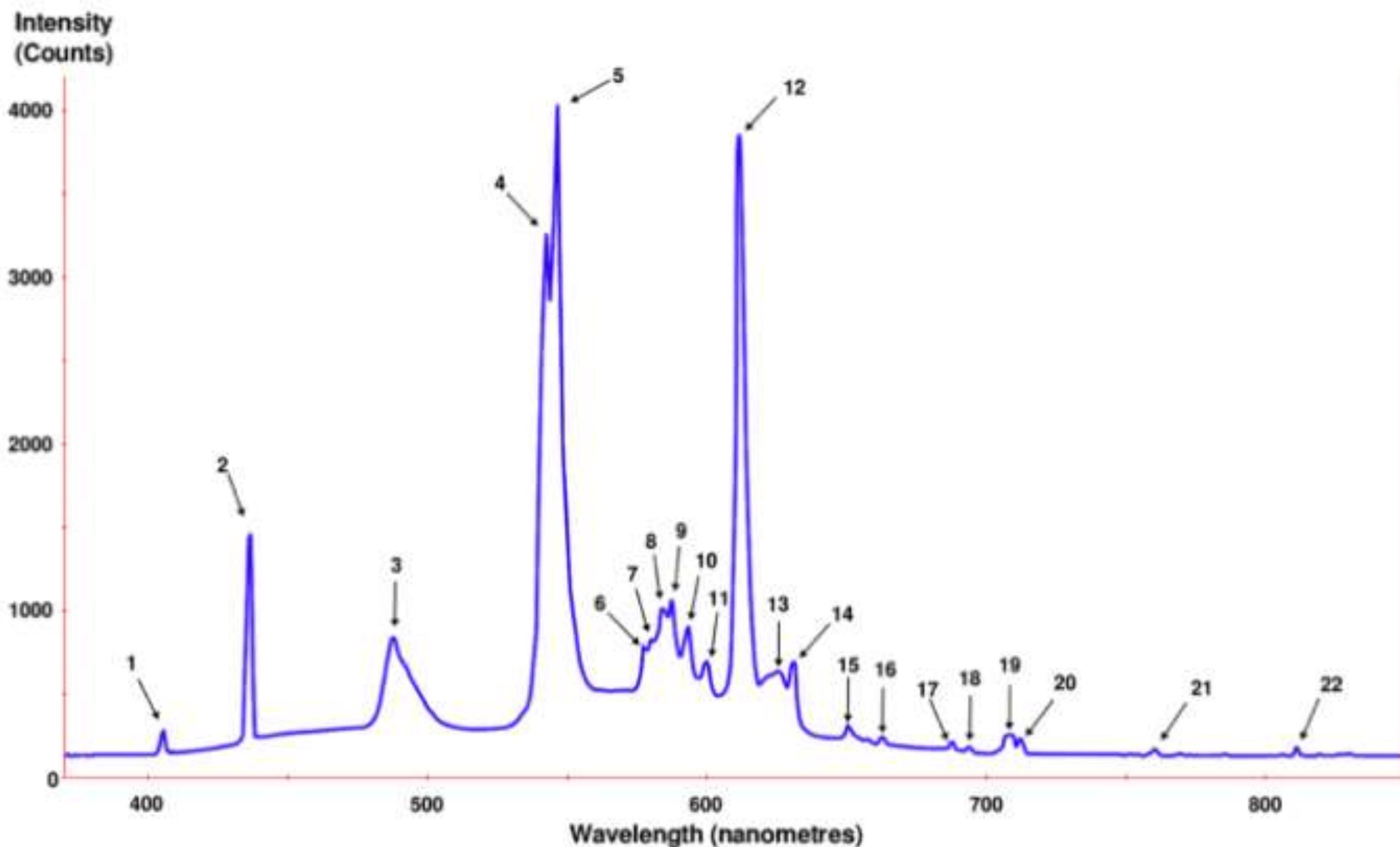


ПРЕДСТАВЛЕНИЕ ЗВУКА

БПМИ-191, Абрамов Арсений

Спектральная плотность

Спектральная плотность - функция, описывающая распределение мощности сигнала в зависимости от частоты.



Спектральная мощность – площадь под графиком спектральной плотности.

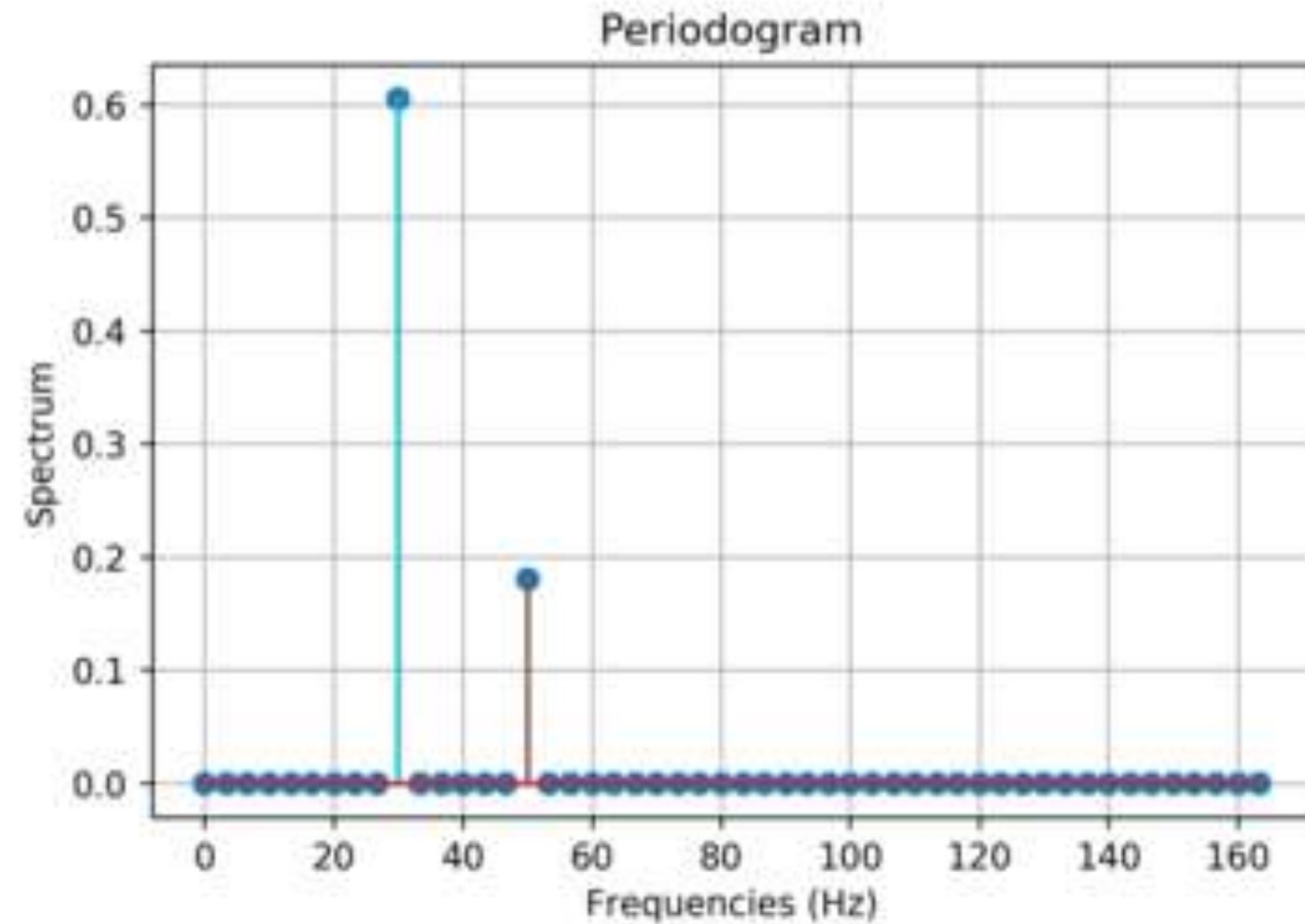


ПРЕДСТАВЛЕНИЕ Звука

БПМИ-191, Абрамов Арсений

Периодограмма

Периодограмма - оценка спектральной мощности сигнала.



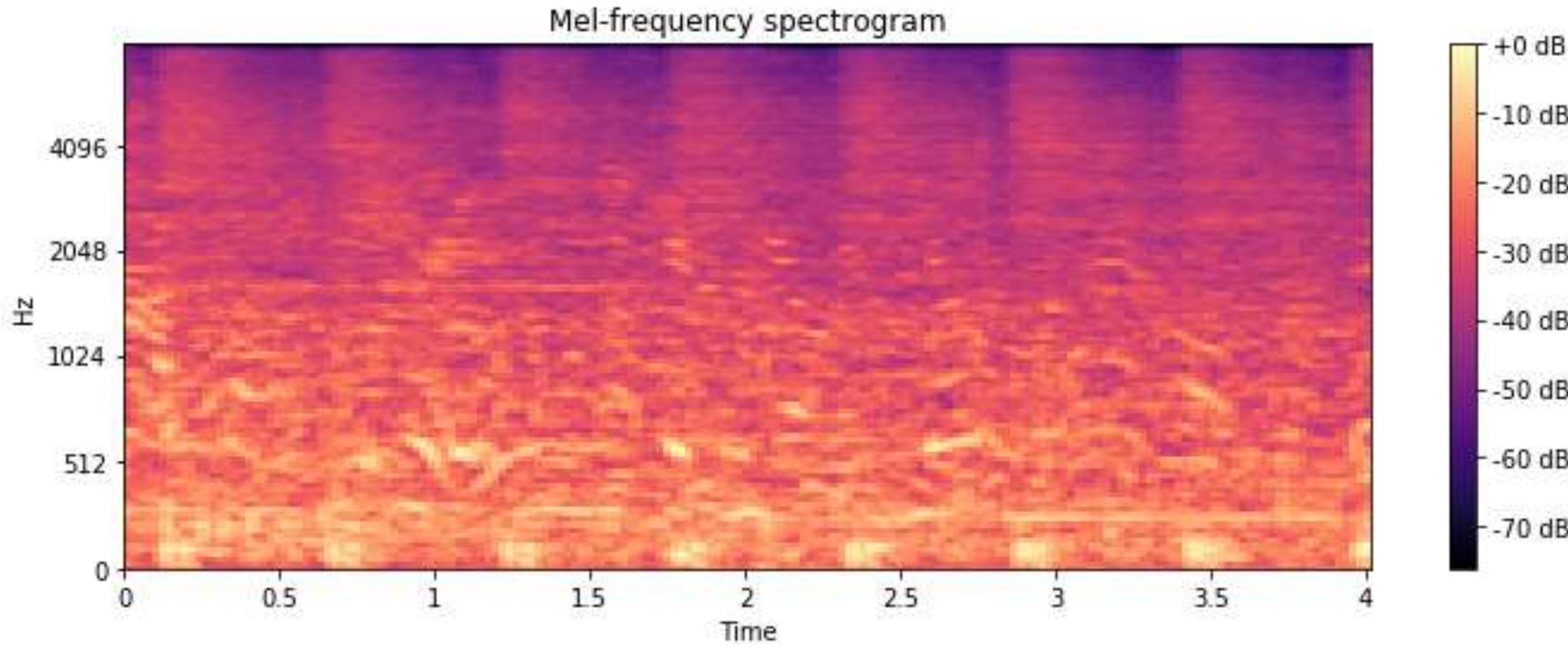


ПРЕДСТАВЛЕНИЕ Звука

БПМИ-191, Абрамов Арсений

Спектrogramma

Спектrogramma – визуальное представление зависимости спектральной плотности сигнала от времени.





ОБРАБОТКА ЗВУКА

БПМИ-191, Абрамов Арсений

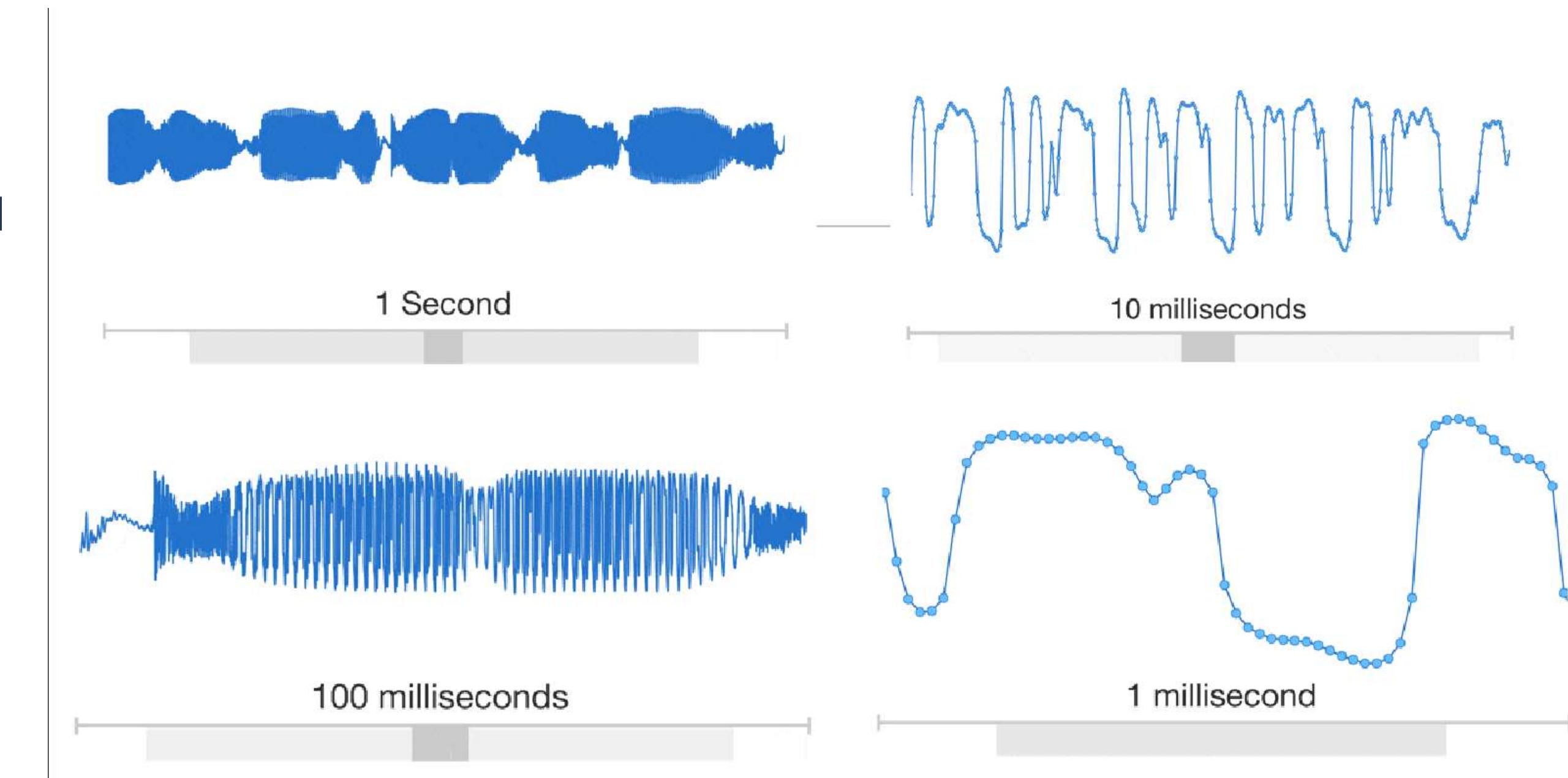
Первые шаги





Шаг 1: кадрирование (audio sampling)

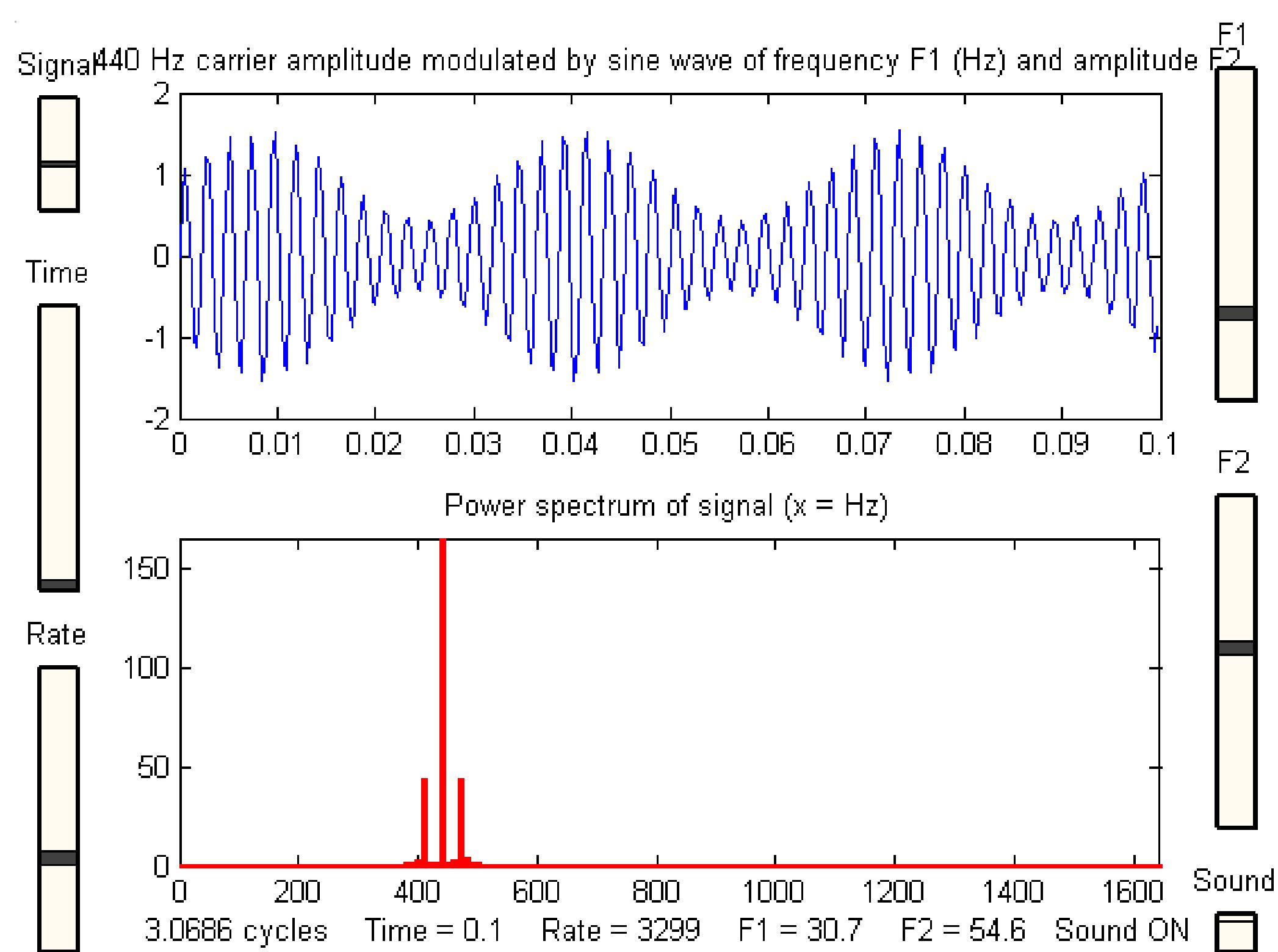
- Предлагается поделить сигнал на короткие промежутки (*кадры*), чтобы представить образец в виде совокупности дискретных временных промежутков.
- (Мы считаем, что на коротких отрезках сигнал не меняется).
Под «коротким» обычно подразумевается промежуток в 20-40 ms.
Обычно отрезки ещё и пересекаются.





Шаг 2: вычисление оценки спектральной мощности для каждого кадра

- Применим Оконное Преобразование Фурье (STFT) к каждому кадру.



$$\text{STFT}\{x[n]\}(m, \omega) \equiv X(m, \omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}$$

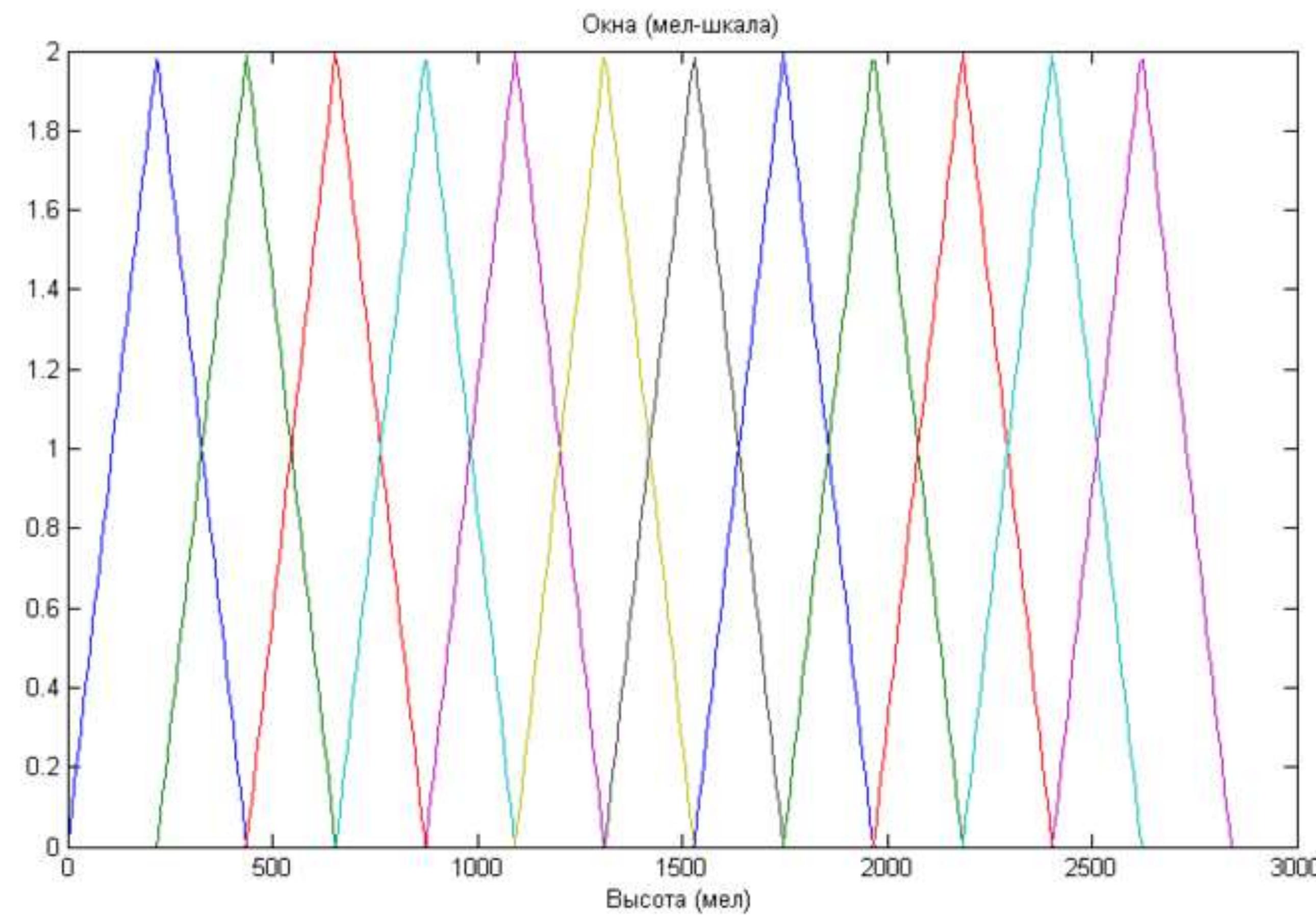
Сигнал
Размер кадра
Окно



MFCC

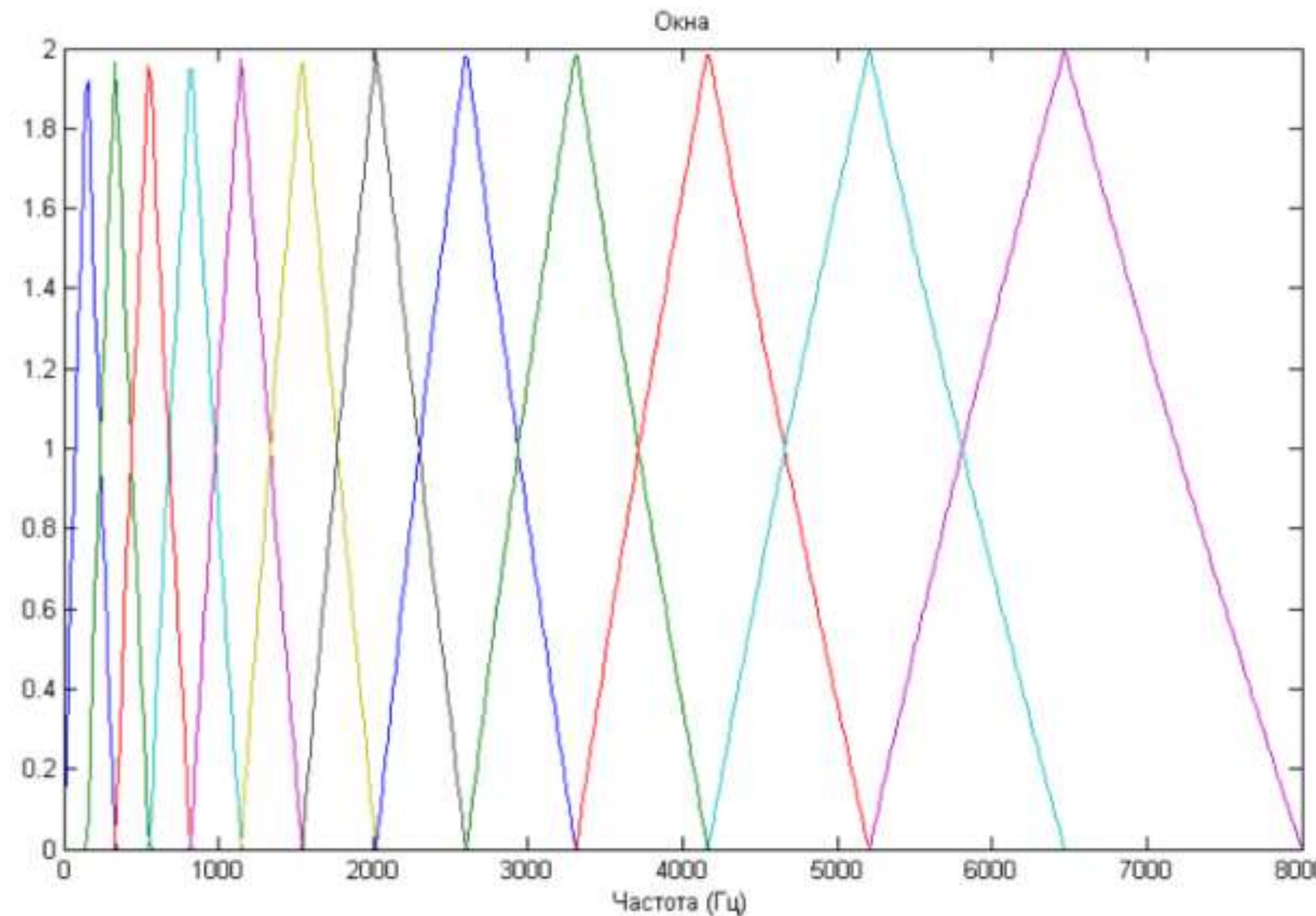
БПМИ-191, Абрамов Арсений

Шаг 3.1: переводим полученный спектр в мел-шкалу.





Шаг 3.2: переводим в частотную шкалу.





Шаг 3.3: наложить мел-фильтр.

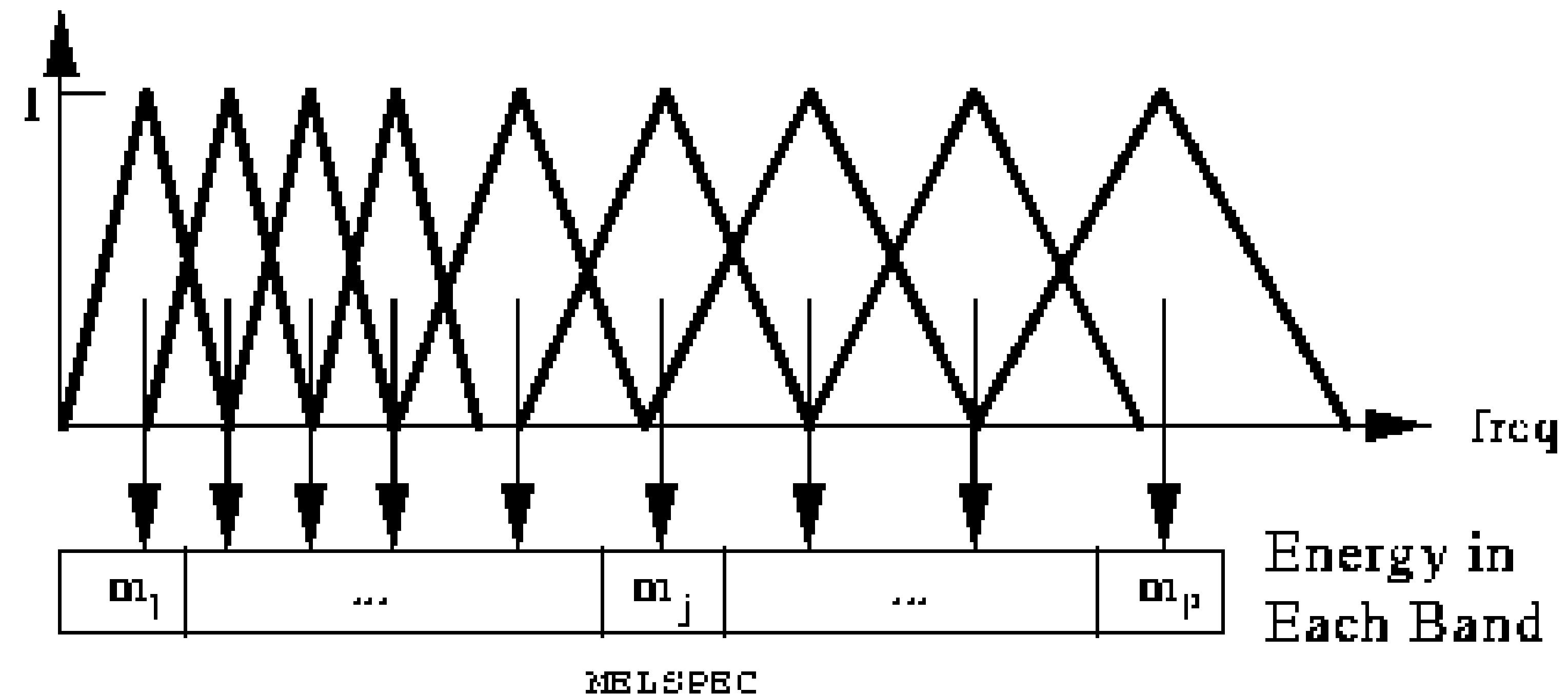
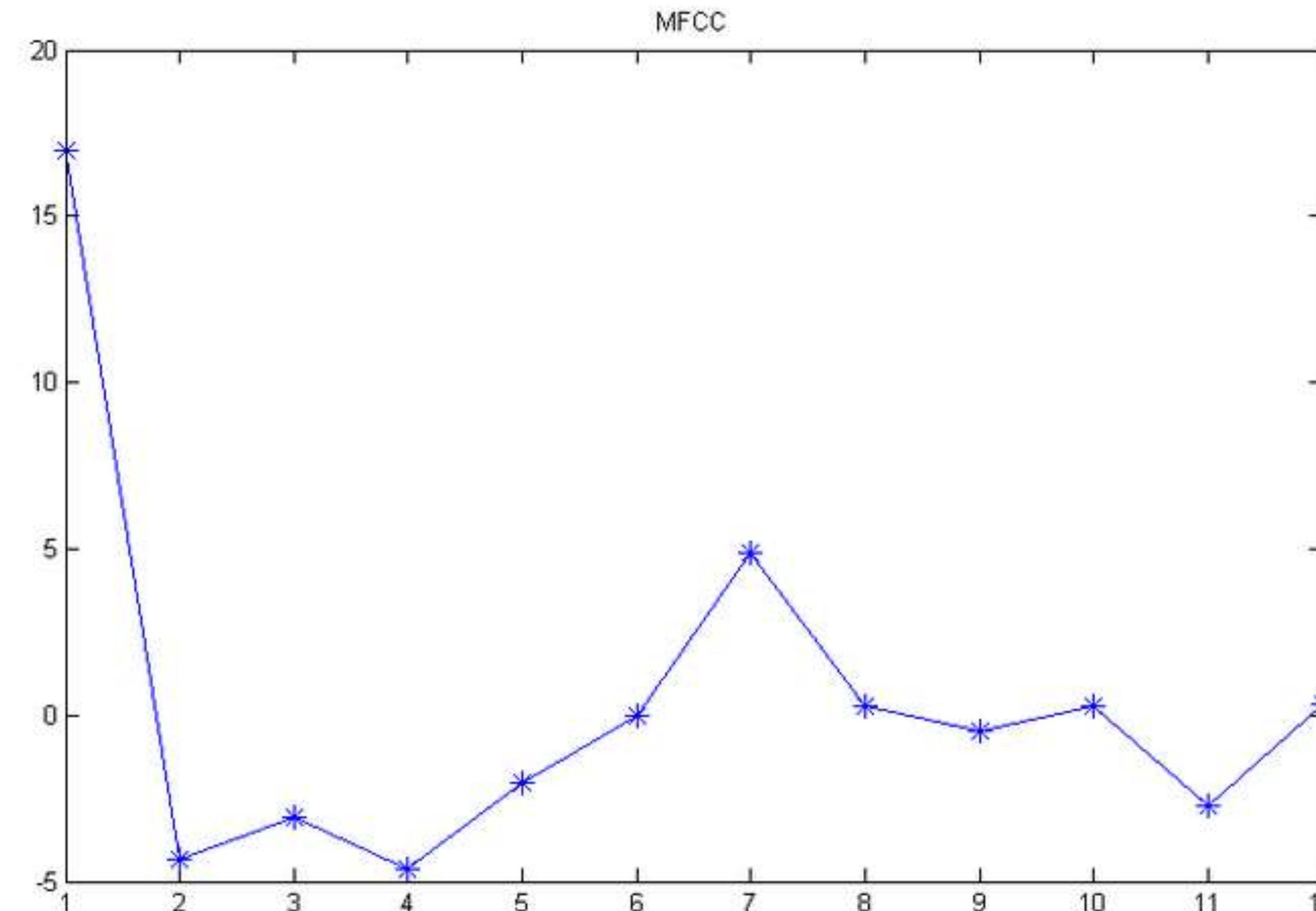


Fig. 5.3 Mel-Scale Filter Bank



Шаг 4: DCT





MFCC

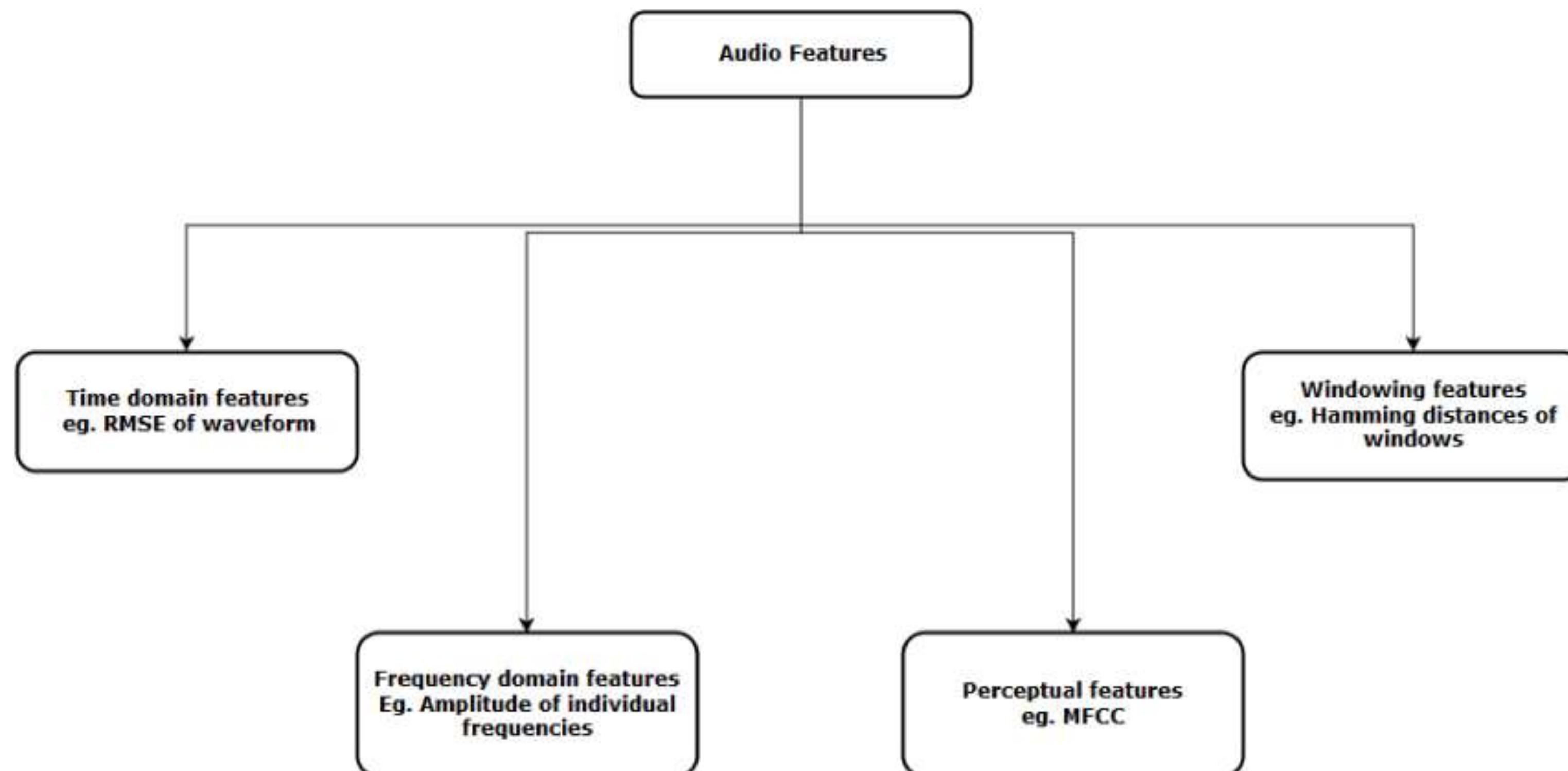
БПМИ-191, Абрамов Арсений

```
mfccs = librosa.feature.mfcc(y=librosa_audio, sr=librosa_sample_rate,  
n_mfcc = 40)
```



ПРИЗНАКИ ЗВУКА

БПМИ-191, Абрамов Арсений





Типы задач.

БПМИ-191, Абрамов Арсений

Распознавание звука.



Генерация звука.





ИСТОЧНИКИ

Источники

Источники

- **Niko Laskaris: How to apply machine learning and deep learning methods to audio analysis**
URL: <https://towardsdatascience.com/how-to-apply-machine-learning-and-deep-learning-methods-to-audio-analysis-615e286fcbbc>
- **Faizan Shaikh: Getting Started with Audio Data Analysis using Deep Learning (with case study)**
URL: <https://www.analyticsvidhya.com/blog/2017/08/audio-voice-processing-deep-learning/>
- **Yuchen Fan, Matt Potok, Christopher Shroba: Deep Learning for Audio**
URL: https://slazebni.cs.illinois.edu/spring17/lec26_audio.pdf

Что такое автоматическое распознавание речи?

Автоматическое распознавание речи (Automatic Speech Recognition, ASR) – это трансформация произнесённой речи в текстовое представление.



Почему ASR желателен для всех языков?

- Речь является основным средством человеческого общения;
- Естественный интерфейс как для людей, разбирающихся в технологиях, так и для других тоже;
- Способ сохранения языков, находящихся под угрозой исчезновения;

Что делает ASR сложной задачей?

Множество источников изменчивости речи!

- *Стиль*: разговорная речь или читаемый текст? Непрерывная речь или изолированные слова?
- Окружающая среда: фоновой шум, настройки канала, акустика помещения и так далее;
- *Характеристики спикера*: тон речи, акцент, возраст и так далее;
- *Особенности задачи*: количество слов в словаре, ограничения языка и так далее;

Отбор признаков

Что за признаки из звукового файла у нас извлекаются? Объект нашего распознавания – это **звуковой сигнал**.

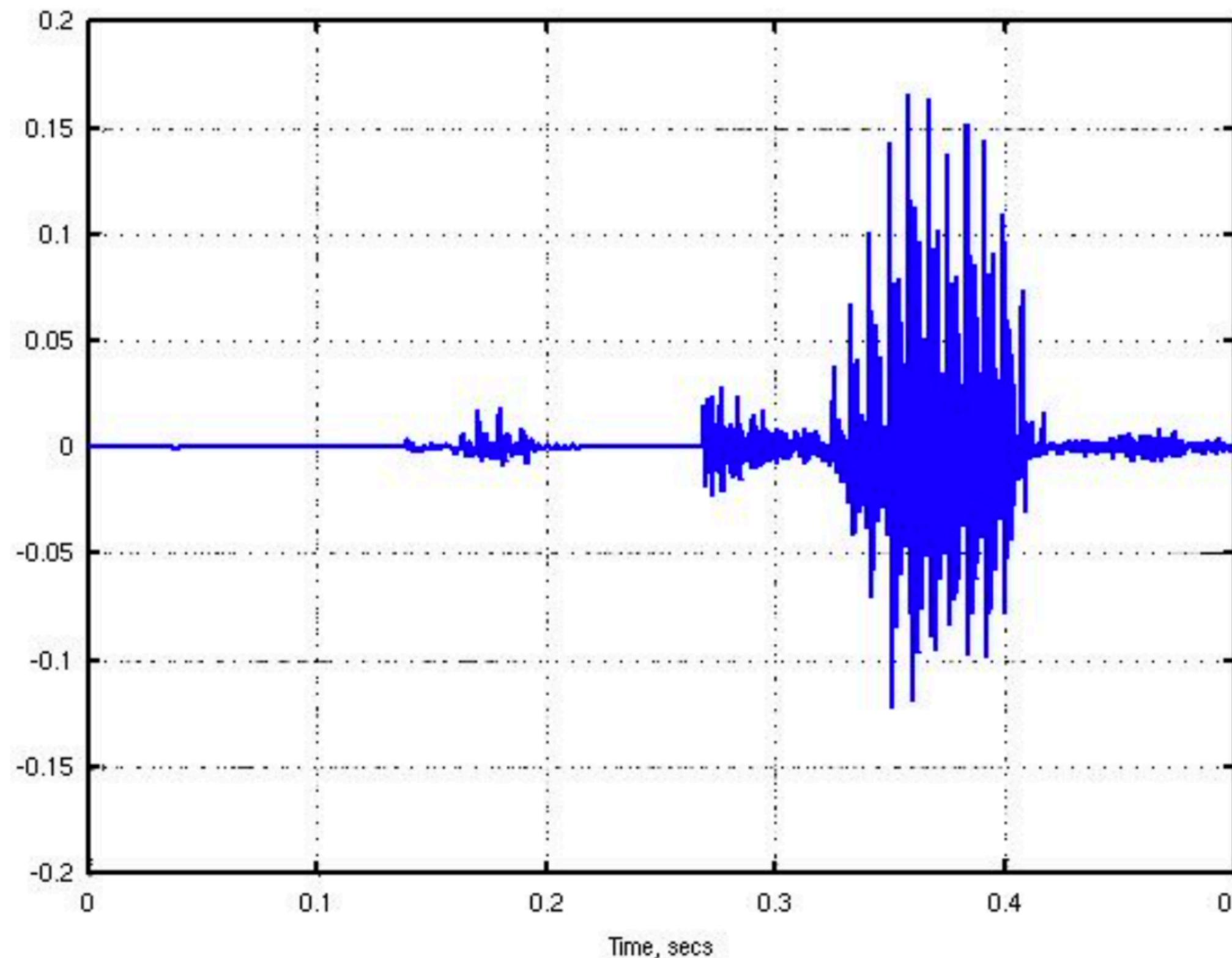
Математически, звуковой сигнал является совокупностью синусоид.



Отбор признаков

Определяющей особенностью сигнала является то, какие синусоиды в него вложены. А как определить, синусоиды с какой частотой дают наибольший вклад в сигнал? Провести спектральный анализ!

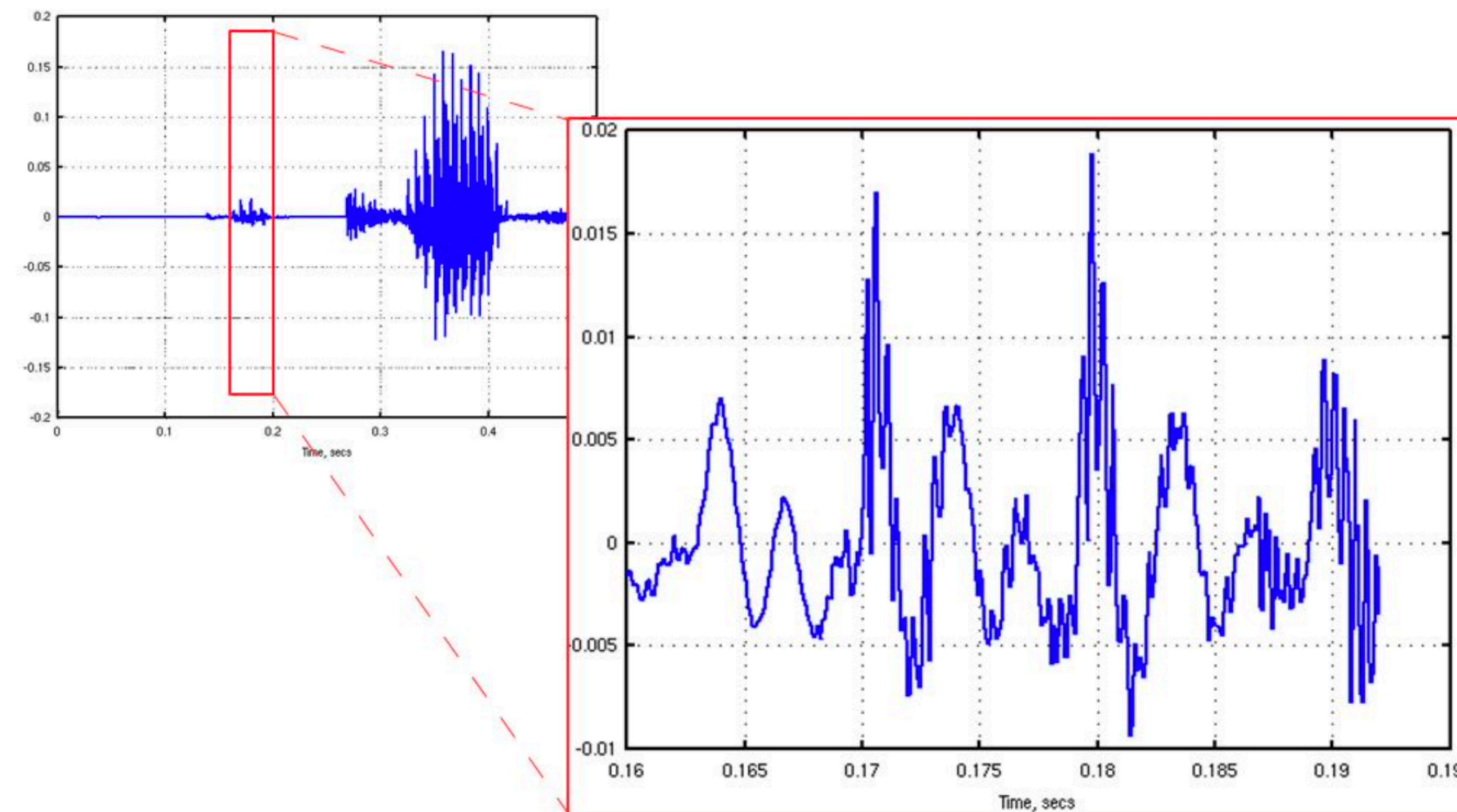
А можно ли провести спектральный анализ у вот такого сигнала?



Отбор признаков

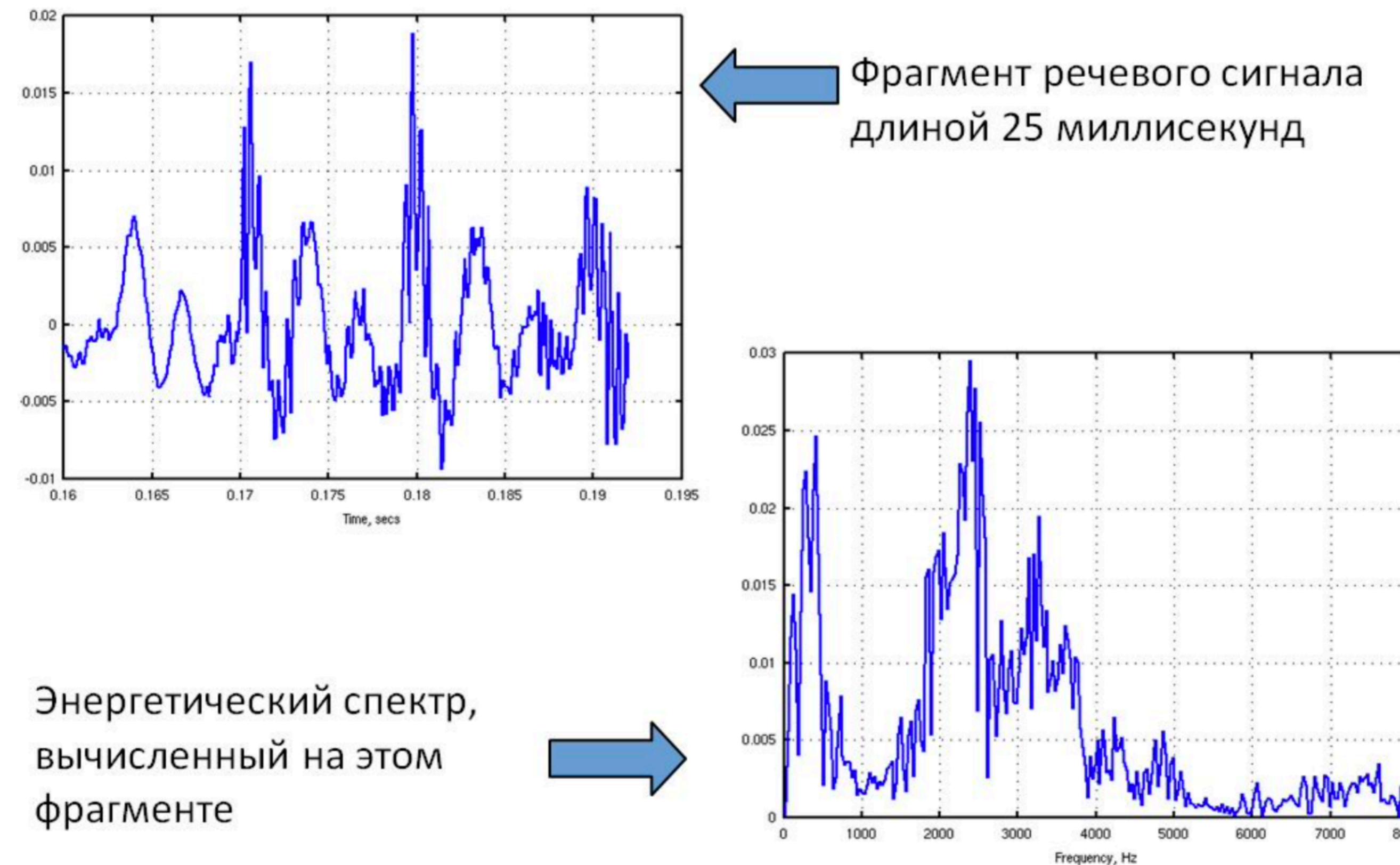
Ответ: нет, нельзя, поскольку этот сигнал не периодический и спектральный анализ (к примеру, быстрое преобразование Фурье) провести нельзя.

Для этого мы порежем наш отрезок на меленькие отрезочки по 25 миллисекунд, тогда мы уже можем наблюдать что-то похожее на период.



Отбор признаков

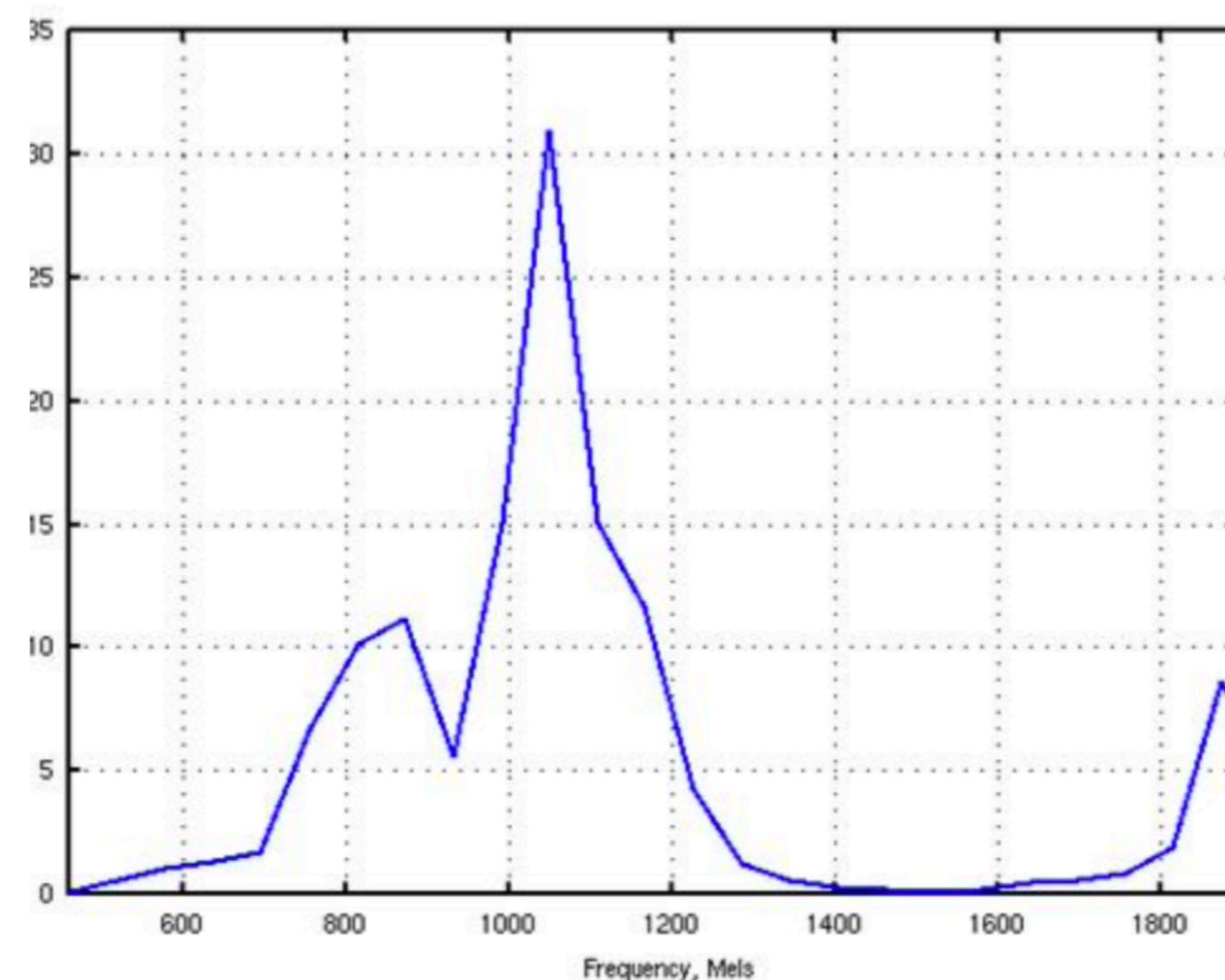
То есть берём это окошко и проводим спектральный анализ. И вот как этот спектр меняется по времени, является определяющей особенностью сигнала.



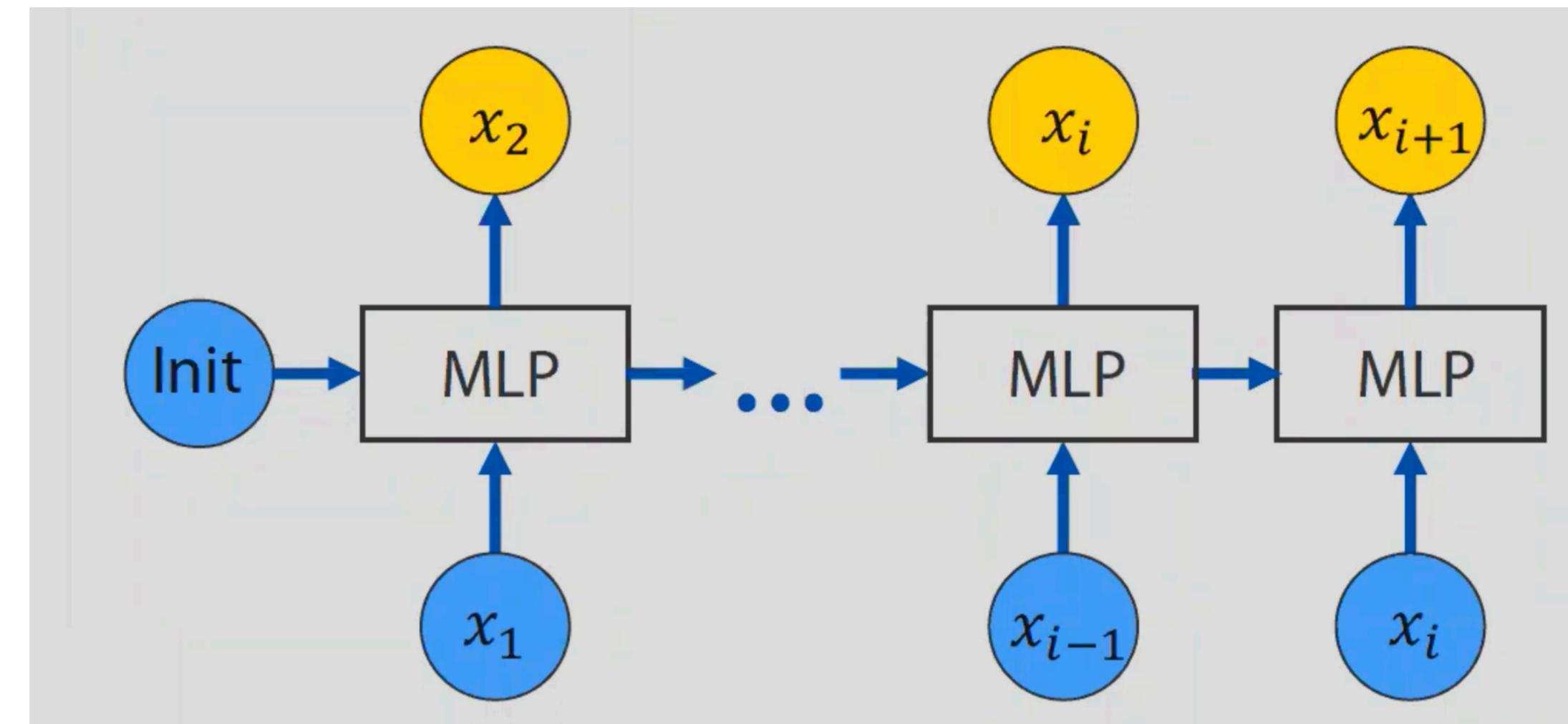
Отбор признаков

Но если посмотрим на график, то спектр очень шумный. На это влияют всякие параметры, связанные с акустическими условиями, с особенностями говорящего и так далее.

Чтобы избавиться от этого применяется специальная мел-шкала, которая преобразовывает наш спектр. И в результате мы получаем красивый сглаженный спектр, с которым ASR-у работать намного проще.



Рекуррентные нейронные сети (RNN)



На каждом шаге:

- Вход фиксированного размера – один новый токен и некий выход с предыдущего шага;
- Используется одна и та же сеть, следовательно меньше параметров;

Connectionist Temporal Classification (CTC)

Как в целом работает распознавание речи? У нас есть набор аудиофайлов и их транскрипции (множество символов). К сожалению, мы не знаем, как именно символы из транскрипции соответствуют аудиофайлам. Это делает обучение распознаванию речи сложнее, чем может показаться на первый взгляд.

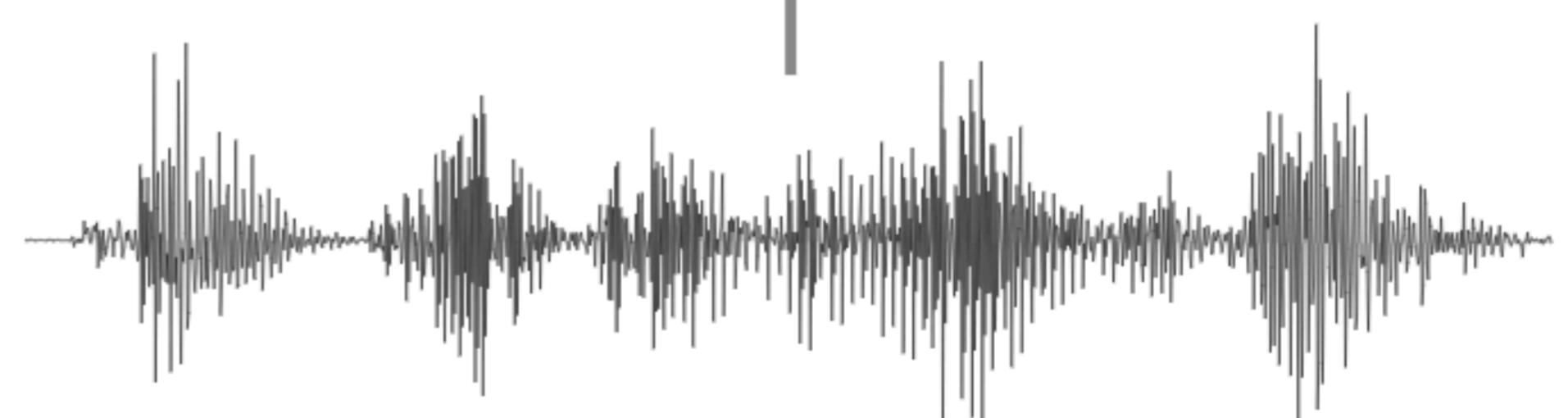
the quick brown fox



The quick brown fox

Handwriting recognition: The input can be (x, y) coordinates of a pen stroke or pixels in an image.

jumps over the lazy dog



Speech recognition: The input can be a spectrogram or some other frequency based feature extractor.

Connectionist Temporal Classification (CTC)

Connectionist Temporal Classification (CTC) – это способ обойти незнание соответствия между входом и выходом. Как мы увидим, он особенно хорошо подходит для таких приложений, как распознавание речи и рукописного ввода.

Connectionist Temporal Classification (CTC)

Чтобы быть более формальным, давайте рассмотрим отображение входной последовательности (к примеру, аудио)

$$X = [x_1, \dots, x_T]$$

к соответствующим выходным последовательностям (транскрипциям)

$$Y = [y_1, \dots, y_U]$$

Мы хотим найти точное соответствие между X и Y .

Connectionist Temporal Classification (CTC)

Существуют проблемы, которые мешают нам использовать более простые алгоритмы. В частности:

- X и Y могут отличаться в размерах;
- Соотношение длин X и Y может варьироваться;
- У нас нет точного сопоставления элементов X и Y ;

Connectionist Temporal Classification (CTC)

Алгоритм СТС решает эти проблемы. Для данного X это дает нам распределение выходных данных по всем возможным Y .

Мы можем использовать это распределение либо для вывода вероятного результата, либо для оценки вероятности данного результата.

СТС. Алгоритм

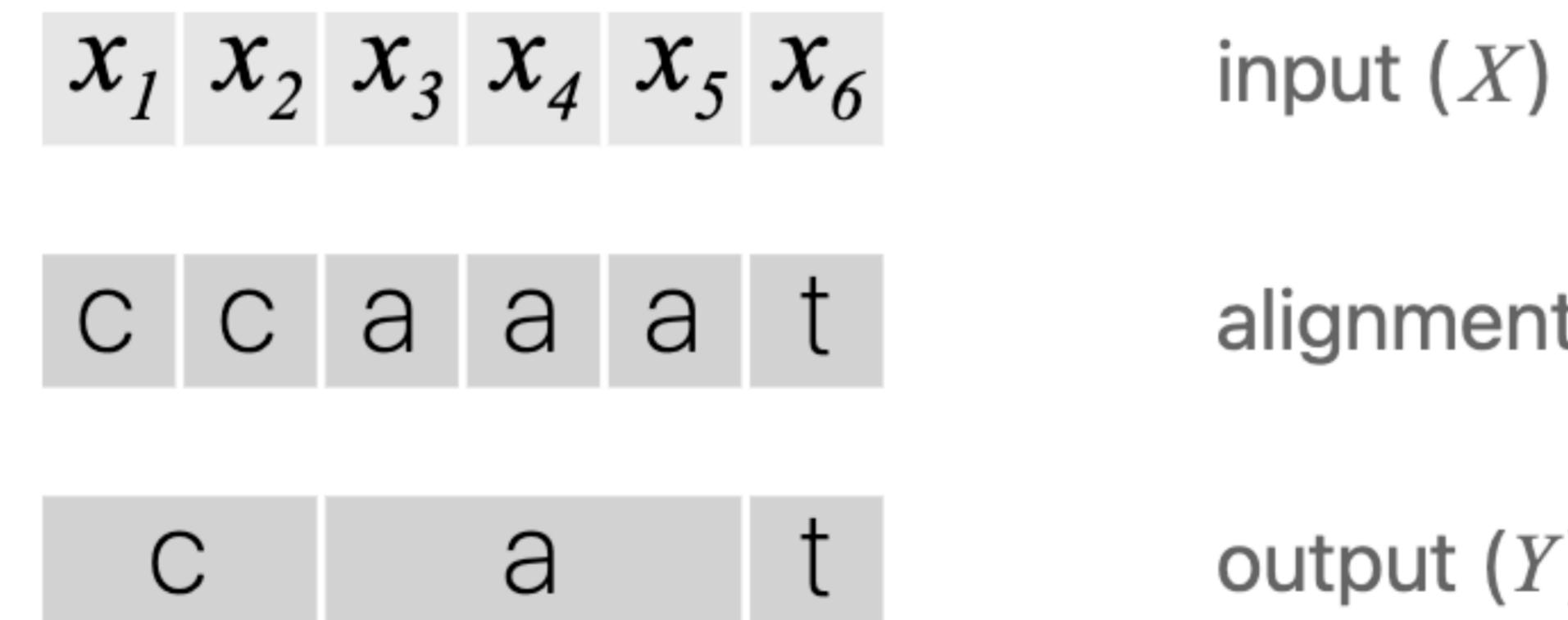
Алгоритм СТС может назначить вероятность для всякого Y , исходя из заданного X . Ключом к вычислению этой вероятности является то, как СТС думает о выравнивании между входами и выходами.

Мы начнем с рассмотрения этих выравниваний, а затем покажем, как использовать их для вычисления функции потерь и получения итогового результата.

СТС. Алгоритм

Сначала посмотрим, что это за выравнивания, чтобы понять, как в конечном счете считается функция потерь.

Давайте рассмотрим наивный подход: пусть у нас есть вход с размером 6 и $Y = [c, a, t]$. Один из способов выравнивания X и Y – это назначить каждому входу какой-либо выход, потом объединить совпадения.



СТС. Алгоритм

У такого подхода есть две проблемы:

- Обычно, не имеет смысла сопоставлять каждому входу какой-либо выход. Например, в речи могут быть периоды тишины без соответствующего выхода;
- У нас нет возможности создавать выходные строки с подряд идущими символами. Рассмотрим выравнивание [h, h, e, l, l, l, o]. Сворачивание повторов приведет к появлению “helo” вместо “hello”.

СТС. Алгоритм

Чтобы обойти эти проблемы, СТС вводит новый символ в набор разрешаемых выходов. Этот новый символ иногда называют пустым символом. Мы будем называть его ϵ . Он ничему не соответствует и в конце просто удаляется.

СТС. Алгоритм

Выравнивания, разрешённые СТС, имеют ту же длину, что и входные данные. Мы разрешаем любое выравнивание, которое сопоставляется с Σ после объединения повторов и удаления символов ϵ .

Diagram illustrating the initial state of the algorithm. The input string "hello" is shown as tokens: h, h, e, ϵ , ϵ , |, |, |, |, o. Each character is enclosed in a separate box.

First, merge repeat characters.

Diagram illustrating the state after merging repeat characters. The tokens are: h, e, ϵ , |, ϵ , |, o. The merged repeat characters (ϵ) are shown in grey boxes.

Then, remove any ϵ tokens.

Diagram illustrating the state after removing ϵ tokens. The tokens are: h, e, |, |, |, o. The removed ϵ tokens are shown in grey boxes.

The remaining characters are the output.

Diagram illustrating the final output of the algorithm. The tokens are: h, e, |, |, o. The output is shown in grey boxes.

СТС. Алгоритм

Если Σ содержит два одинаковых символа подряд, то допустимое выравнивание должно содержать символ ϵ между ними. С помощью этого правила мы можем различать выравнивания, которые сворачиваются в “hello”, и те, которые сворачиваются в “helo”.

h h e ϵ ϵ | | | ϵ | | o

First, merge repeat characters.

h e ϵ | ϵ | o

Then, remove any ϵ tokens.

h e | | | o

The remaining characters are the output.

h e | | o

СТС. Алгоритм

Давайте вернемся к выходу $[c, a, t]$ с входом длины 6. Посмотрим на несколько примеров допустимых и недопустимых выравниваний.

Valid Alignments



Invalid Alignments



corresponds to
 $Y = [c, c, a, t]$



has length 5

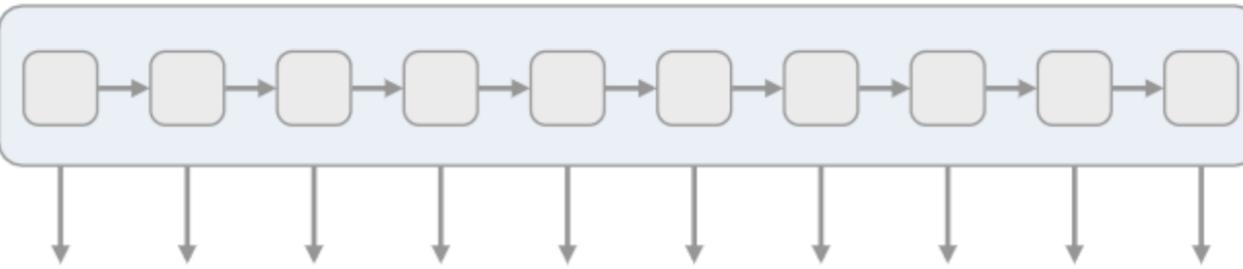
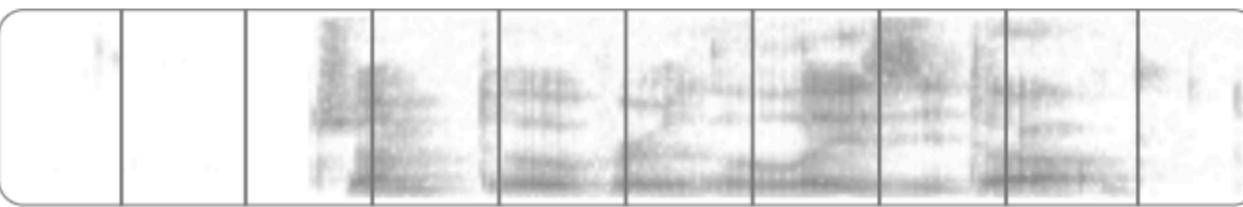


missing the 'a'

СТС. Функция потерь

Выравнивания СТС дают нам естественный способ перехода от вероятностей на каждом временном шаге к вероятности выходной последовательности.

СТС. ФУНКЦИЯ ПОТЕРЬ



h	h	h	h	h	h	h	h	h	h	h
e	e	e	e	e	e	e	e	e	e	e
o	o	o	o	o	o	o	o	o	o	o
ϵ										

h	e	ϵ			ϵ			o	o
h	h	e			ϵ	ϵ		ϵ	o
ϵ	e	ϵ			ϵ	ϵ		o	o

h	e			o
e			o	
h	e		o	

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
 $\{h, e, |, o, \epsilon\}$ for each input step.

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

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

СТС. Функция потерь

Чтобы быть точным, итоговая вероятность считается так:

$$P(Y | X) \text{ условная вероятность} = \sum_{A \in \mathcal{A}_{(X,Y)}} \prod_{t=1}^T p_t(a_t | X) \text{ каждый символ}$$

СТС. Функция потерь

Модели, обученные с СТС, обычно используют рекуррентную нейронную сеть (RNN), чтобы оценивать вероятности каждого символа $p_t(a_t | X)$. Такая сеть хорошо работает, поскольку она учитывает предыдущий контекст.

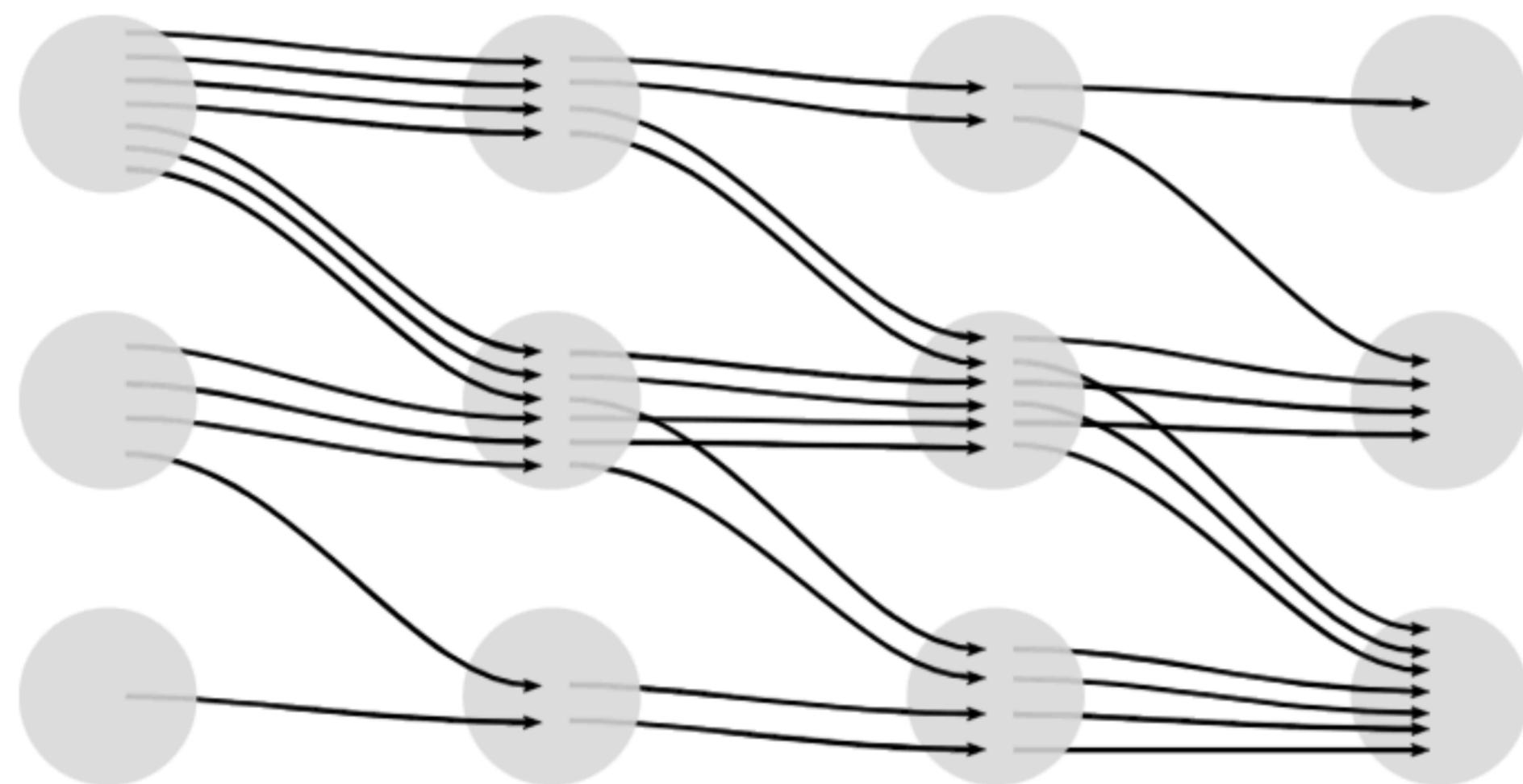
СТС. Функция потерь

Если мы не будем осторожны, подсчёт СТС-loss может дорого нам обойтись. Мы могли бы попробовать простой подход и вычислить оценку для каждого выравнивания, суммируя их все по ходу дела.

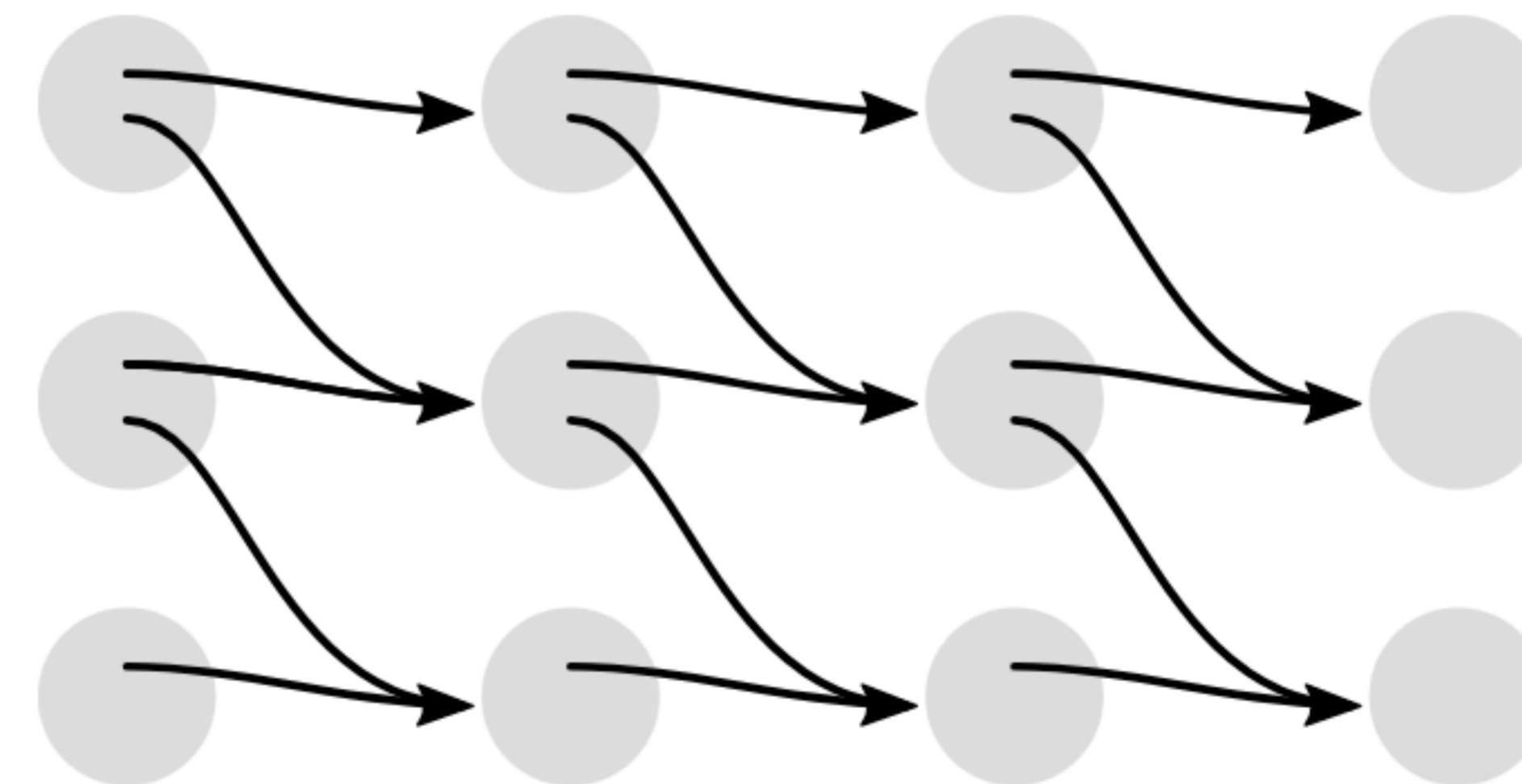
Проблема в том, что может быть огромное количество выравниваний. Для большинства задач это было бы слишком медленно.

СТС. Функция потерь

К счастью, мы можем вычислить потери намного быстрее с помощью динамического программирования. Ключевой момент заключается в том, что если два выравнивания достигли одного и того же результата на одном и том же шаге, то мы можем просто объединить их.



Summing over all alignments can be very expensive.



Dynamic programming merges alignments, so it's much faster.

СТС. Функция потерь

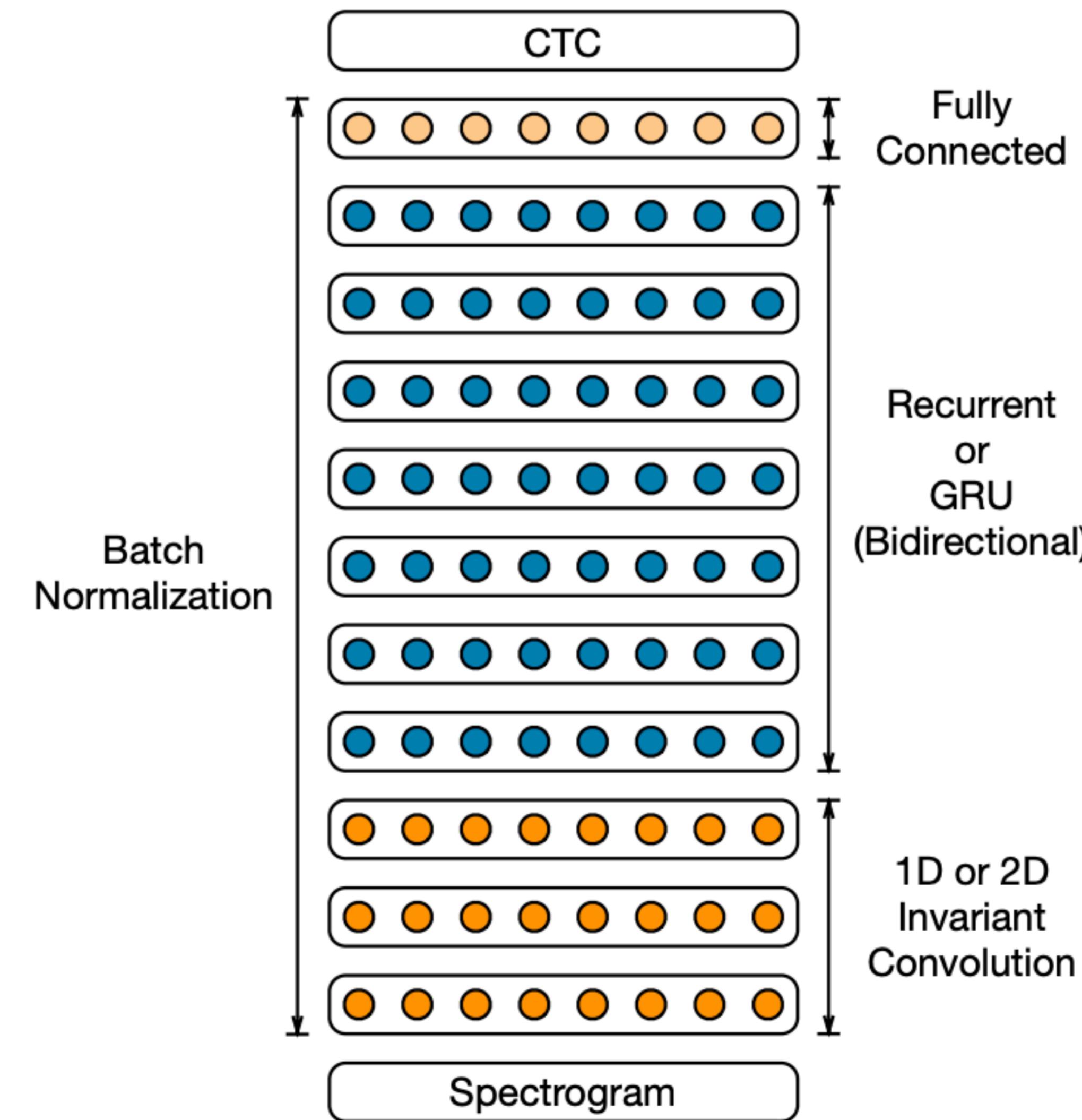
Для обучающего набора \mathcal{D} параметры модели настраиваются так, чтобы минимизировать

$$\sum_{(X,Y) \in \mathcal{D}} -\log p(Y | X)$$

вместо максимизации самой вероятности.

Deep Speech 2

Модель DeepSpeech2 выглядит примерно вот так



Deep Speech 2

Входными данными для этой нейронной сети являются последовательности логарифмических спектrogramм, посчитанные на временных окнах 20-25 миллисекунд.

Выходом этой нейронной сети является словарь (транскрипт) входного аудиофайла. Потом с помощью СТС алгоритма получаем нужный нам результат.

Источники

- <https://www.youtube.com/watch?v=eke2h9fGtu0>
- <https://www.youtube.com/watch?v=gNe3eQ1FyQk>
- https://www.youtube.com/watch?v=HyUtT_z-cms
- <https://www.youtube.com/watch?v=JpS0LzEWr-4>
- <https://distill.pub/2017/ctc/?undefined>

Спасибо за внимание!



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

Факультет Компьютерных Наук

Глубокое обучение для работы со звуком: генеративные модели

Асланов Алишер, БПМИ191

Москва, 2021



Постановка задач

- Сгенерировать последовательность (звук) без каких-либо начальных условий
- Условная генерация (например, подгон под какой-либо стиль)
- Text-to-Speech (TTS)
- Speech enhancement, voice conversion, source separation*, etc.

* Например, <https://vocalremover.org/>



Более формально

Имеем последовательность $\mathbf{x} = \{x_1, \dots, x_T\}$ и вероятностную модель

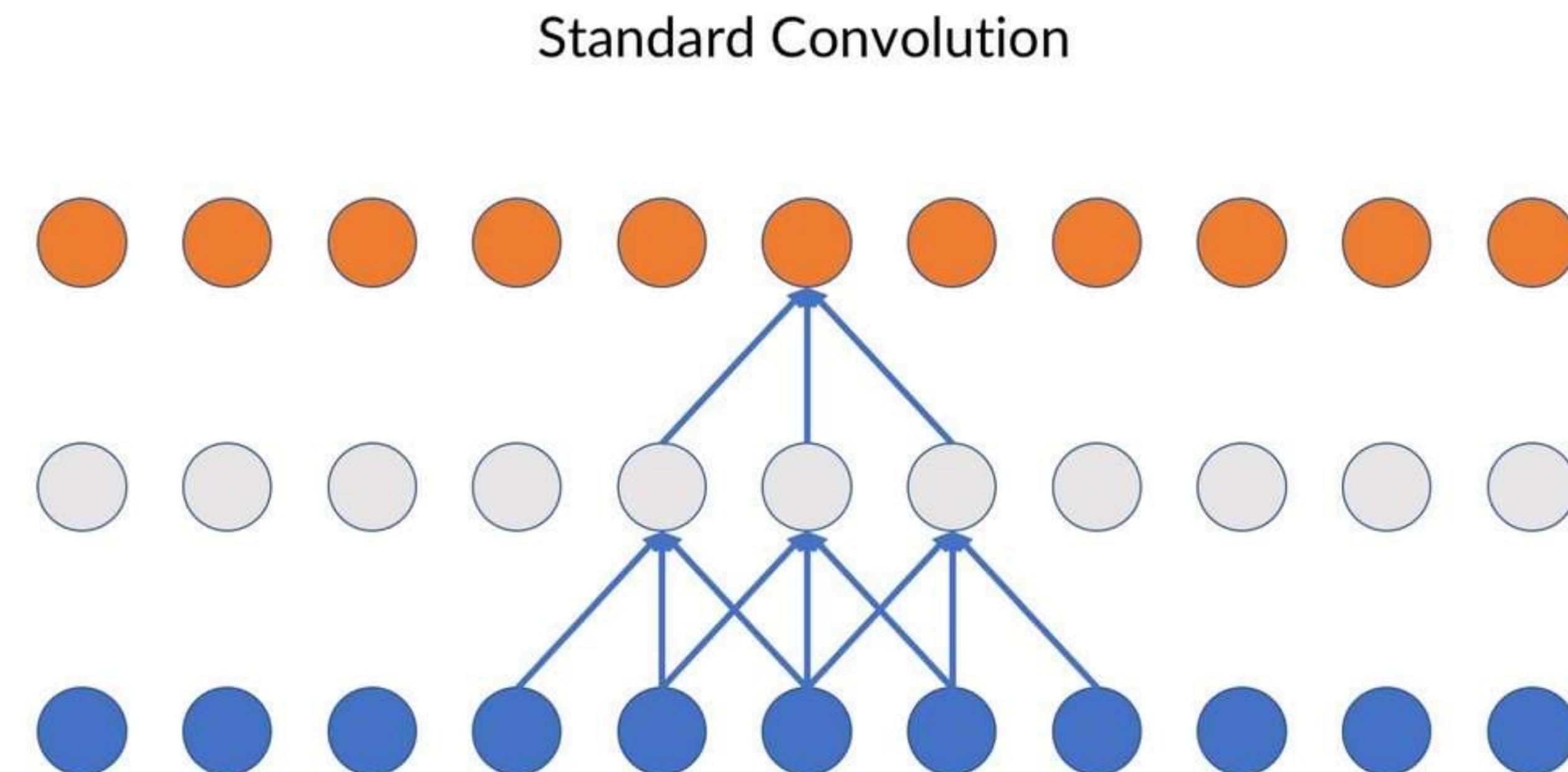
$$p(\mathbf{x}) = \prod_{t=1}^T p(x_t \mid x_1, \dots, x_{t-1})$$

Хотим предсказывать следующий элемент последовательности, зная все предыдущие



Попробуем использовать свёртки

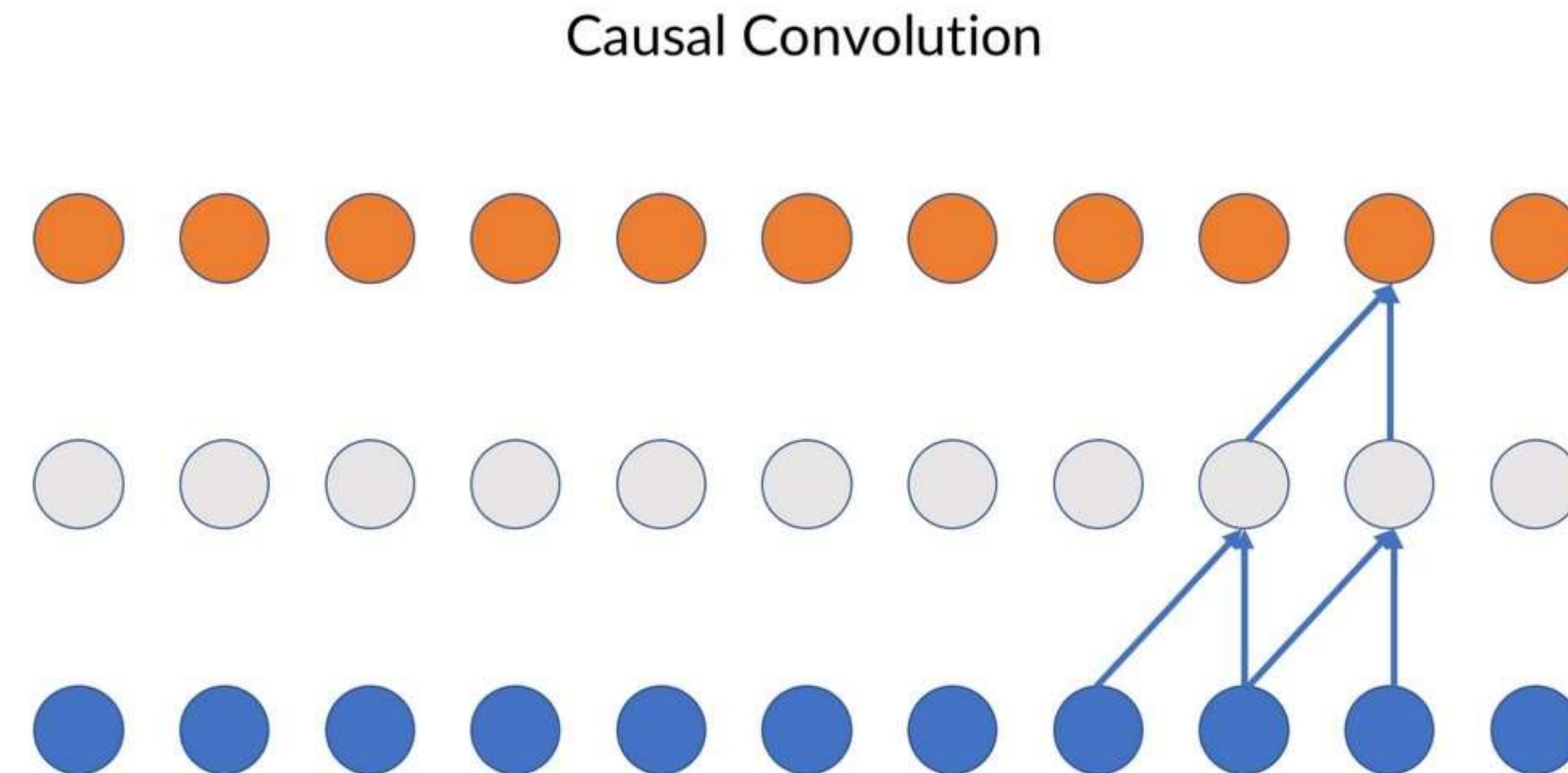
Проблема: много данных на вход + стандартная архитектура не подойдет:





Causal convolutions

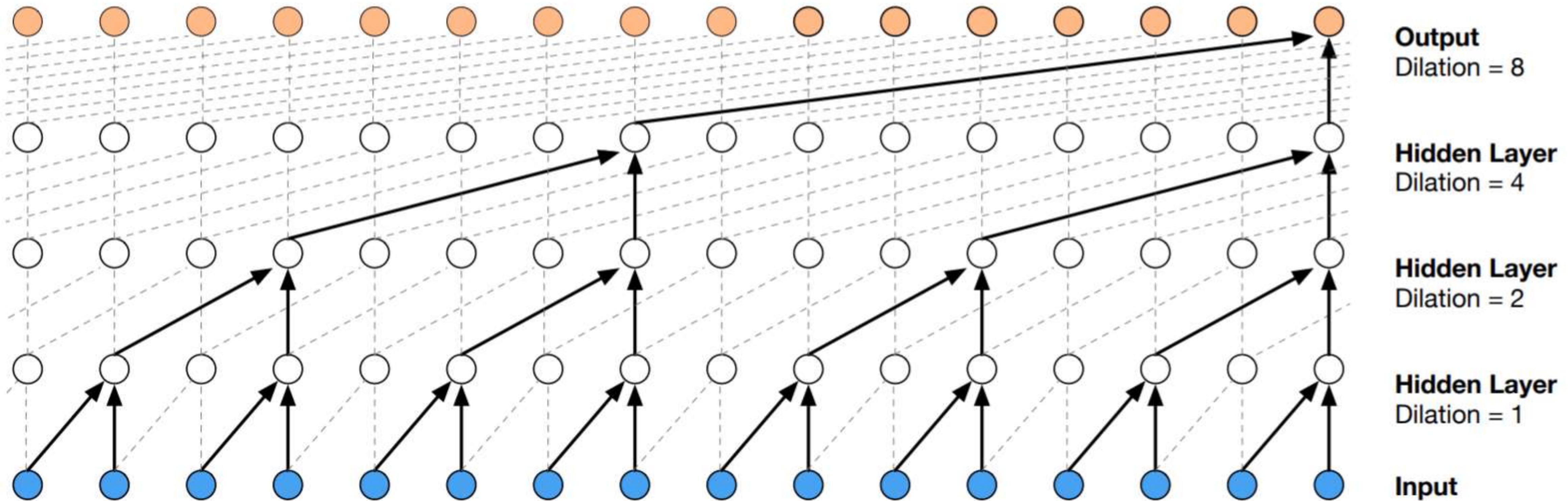
Подстроим архитектуру под наши нужды — не будем «смотреть вперёд»:





Dilated causal convolutions

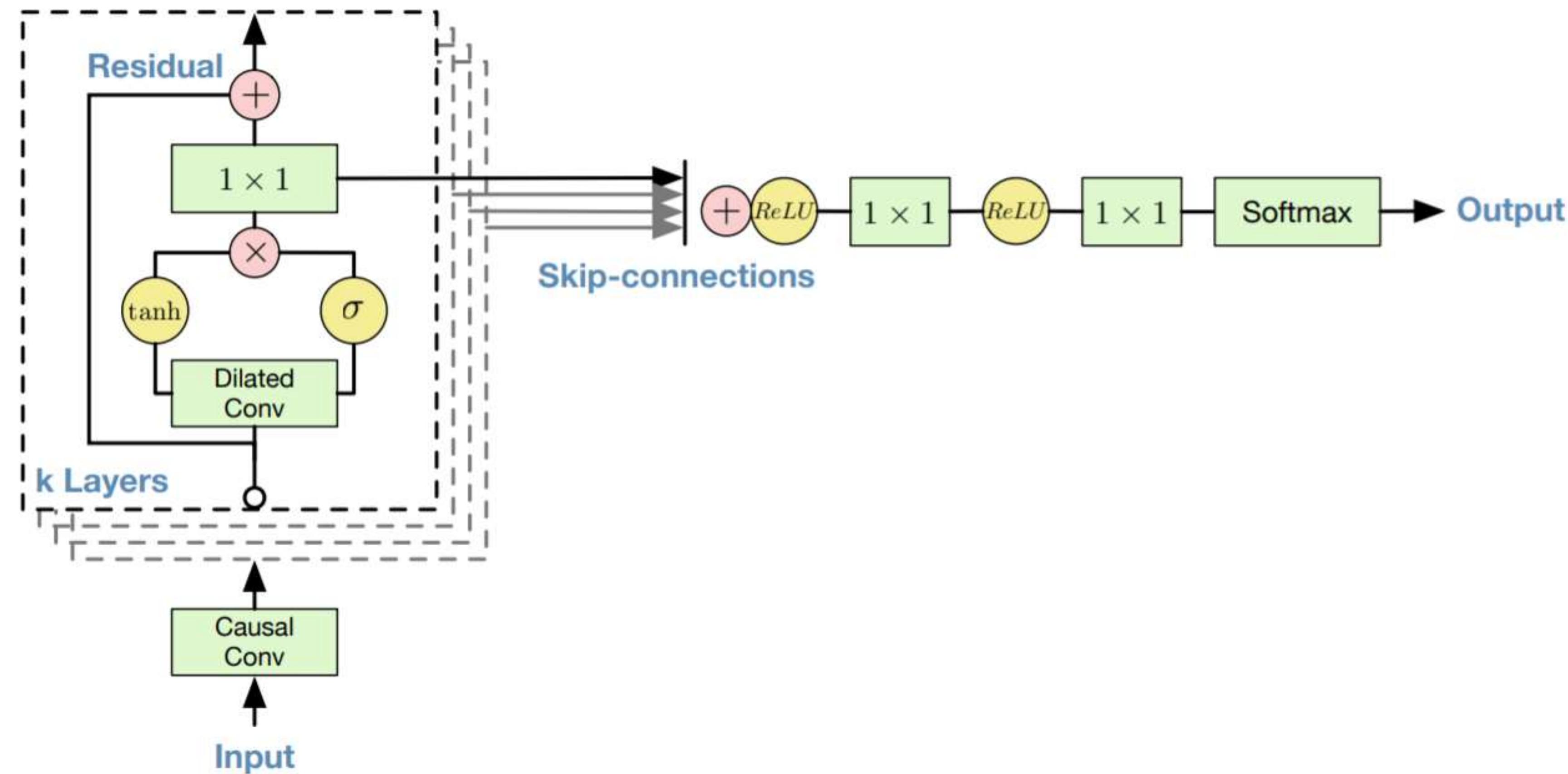
Учтём также «далёкие» зависимости:





Модель WaveNet (2016)

Устроена следующим образом:





Модель WaveNet: технические детали и обучение

- Функция активации $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$
- Residual block и skip-connections — от проблемы затухающих/взрывающихся градиентов
- Обучение — максимизация логарифмической функции правдоподобия по параметрам



Conditional WaveNet

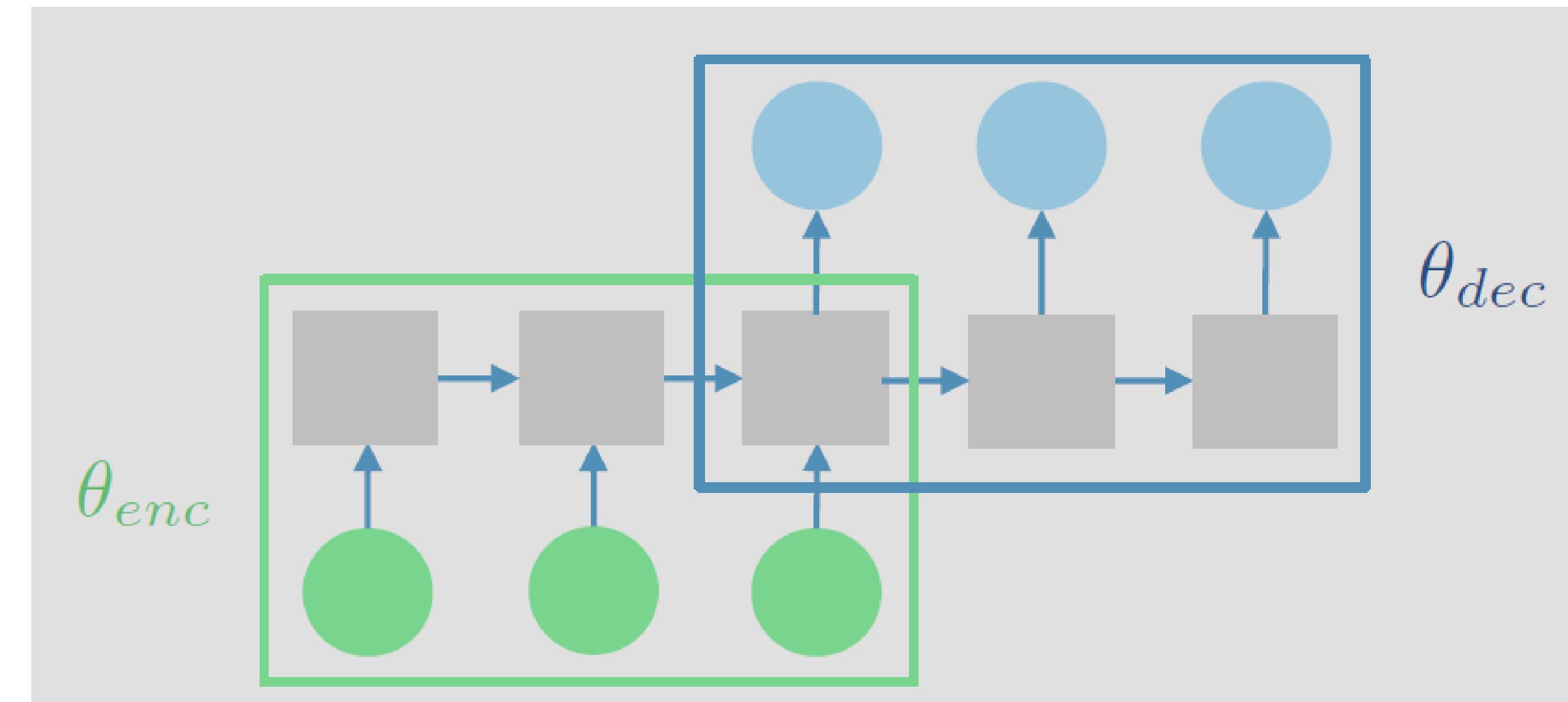
$$p(\mathbf{x} \mid \mathbf{h}) = \prod_{t=1}^T p(x_t \mid x_1, \dots, x_{t-1}, \mathbf{h})$$

- Функция активации меняется на $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T \mathbf{h}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k}^T \mathbf{h})$
- Максимизируемое правдоподобие тоже меняется



Применение WaveNet к задаче TTS

- Архитектура encoder-decoder
- Encoder выделяет лингвистические признаки из текста
- Decoder определяется как conditional WaveNet при условии выделенной encoder'ом информации





Применение WaveNet к задаче TTS: качество модели

- Используется *mean opinion score* — средняя субъективная оценка качества, оценивают люди
- Шкала от 1 до 5, результат вычисляется как $MOS = \frac{\sum_{n=1}^N R_n}{N}$

Speech samples	Subjective 5-scale MOS in naturalness	
	North American English	Mandarin Chinese
LSTM-RNN parametric	3.67 ± 0.098	3.79 ± 0.084
HMM-driven concatenative	3.86 ± 0.137	3.47 ± 0.108
WaveNet (L+F)	4.21 ± 0.081	4.08 ± 0.085
Natural (8-bit μ -law)	4.46 ± 0.067	4.25 ± 0.082
Natural (16-bit linear PCM)	4.55 ± 0.075	4.21 ± 0.071



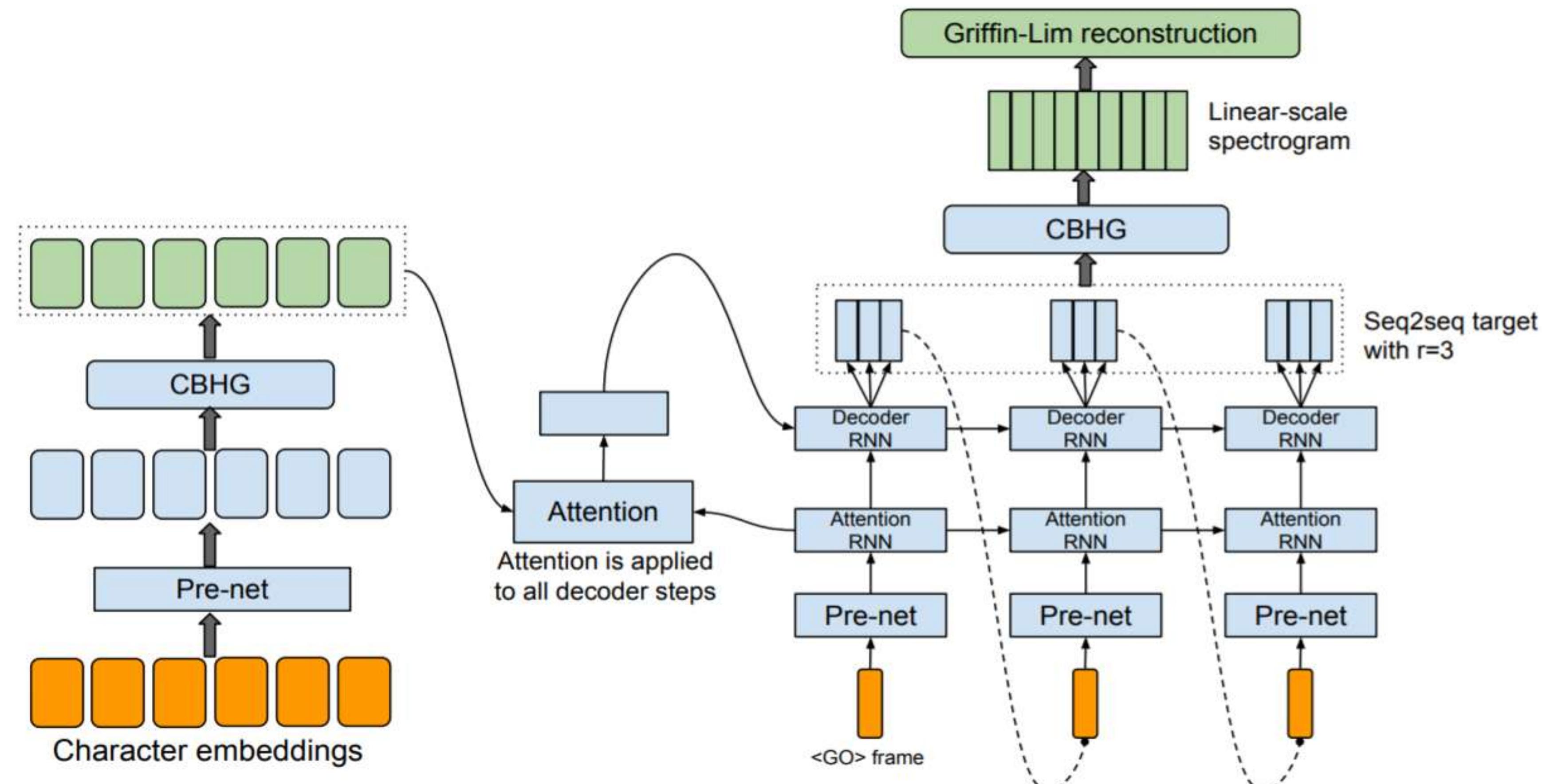
Продолжаем говорить о TTS. Модель Tacotron (2017)

- Принципиальное отличие от WaveNet — *end-to-end* подход к задаче TTS
- Принимает на вход текст и генерирует его озвучку без посредников, как это было при применении WaveNet
- Архитектура использует *рекуррентные нейронные сети*



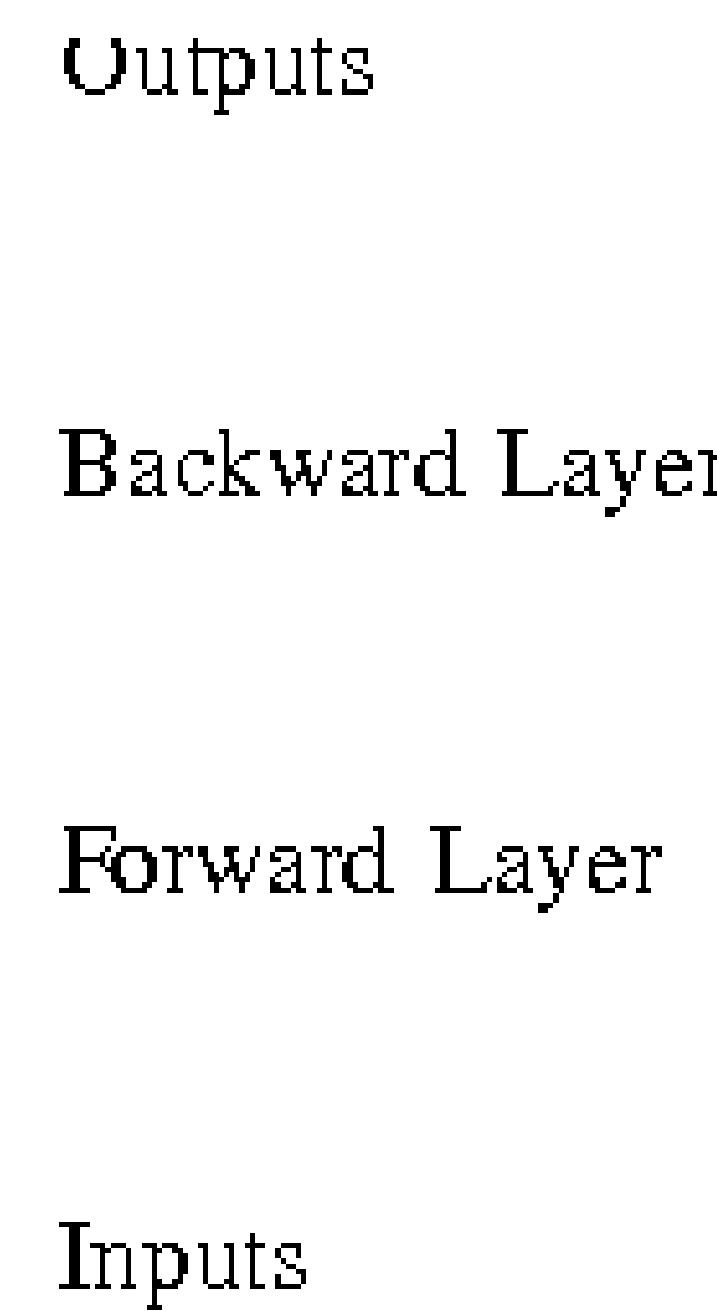
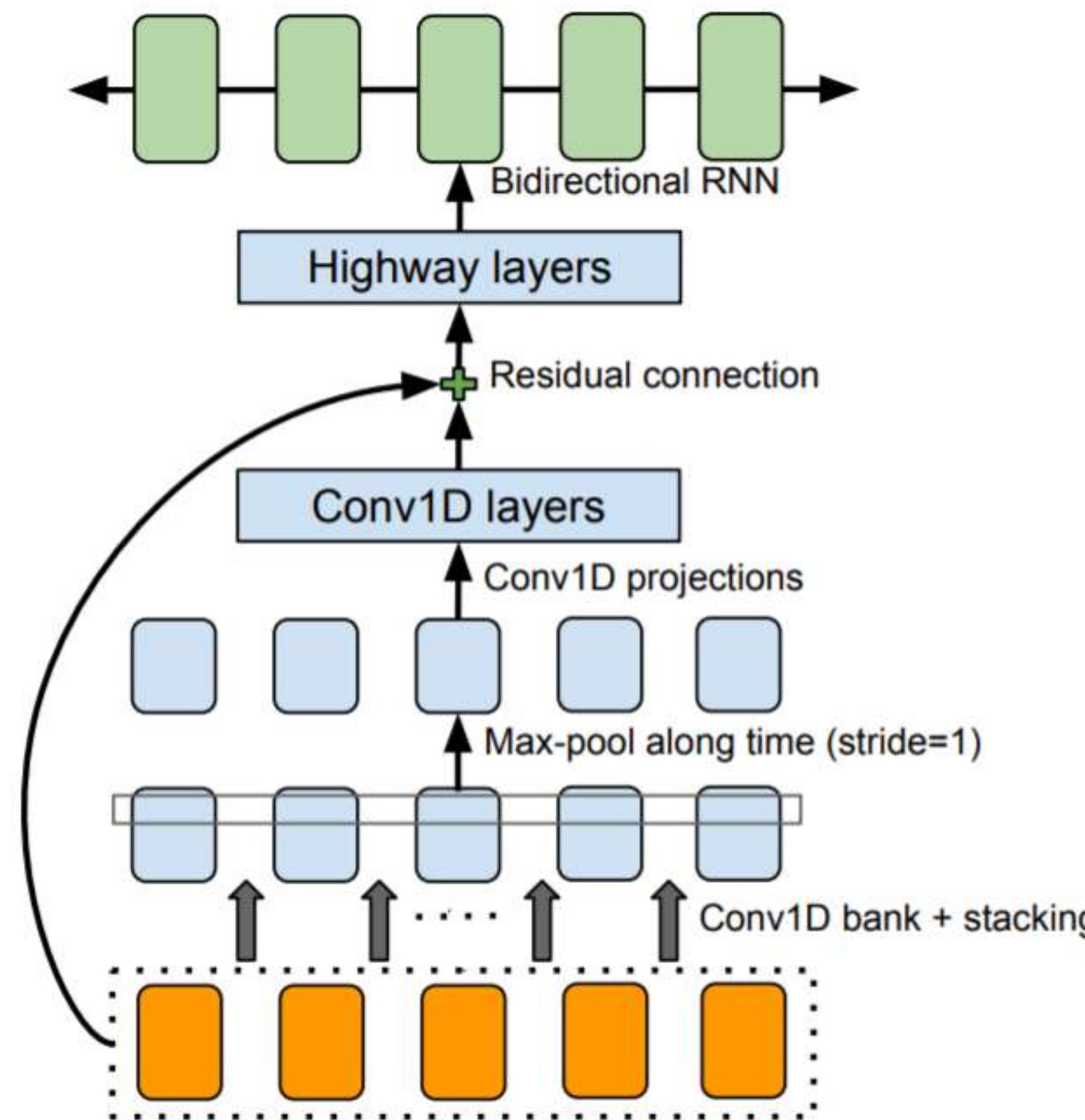
Модель Tacotron

Устроена следующим образом:





Модель Tacotron: модуль CHBG





Модель Tacotron: детали архитектуры и обучение

- *Attention unit* — полносвязная сетка, посредник между encoder'ом и decoder'ом
- *Attention RNN* (GRU, Gated Recurrent Unit) — рекуррентная сетка, отвечающая за «отбор» важных факторов
- Griffin-Lim algorithm — преобразование спектrogramмы непосредственно в звук (не обучается)
- Для обучения используется MAE (l1-loss)



Tacotron: качество модели

- MOS поуже, чем у WaveNet, но работает быстрее
- Tacotron 2 — улучшенная модель, показала достойные результаты*

Table 2: 5-scale mean opinion score evaluation.

	mean opinion score
Tacotron	3.82 ± 0.085
Parametric	3.69 ± 0.109
Concatenative	4.09 ± 0.119

* <https://google.github.io/tacotron/publications/tacotron2/index.html>



Список источников

- <https://arxiv.org/pdf/1609.03499v2.pdf>
- <https://habr.com/ru/company/Voximplant/blog/309648/>
- <https://theaisummer.com/skip-connections/>
- <https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec>
- https://en.wikipedia.org/wiki/Mean_opinion_score
- <https://arxiv.org/pdf/1703.10135.pdf>
- <https://medium.com/analytics-vidhya/bi-directional-rnn-basics-of-lstm-and-gru-e114aa4779bb>
- <https://habr.com/ru/post/409257/>



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

e-mail: amaslanov@edu.hse.ru