Yandex

SCHOOL OF DATA ANALYSIS

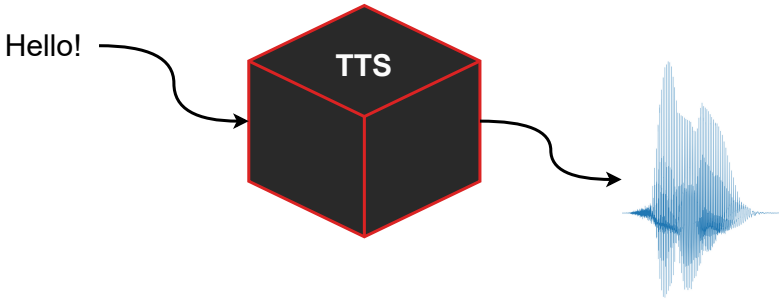
TTS: intro

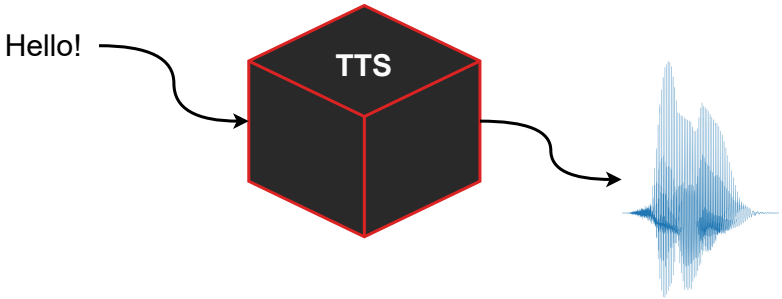
Vladimir Kirichenko

April 18th 2022

Overview

- Problem definition
- Metrics
- Text preprocessing
- Concatenative approach
- Parametric synthesis (teaser)


Hello!
TTS



Input:

- Text with abbreviations, numbers, strange symbols etc.
- Markup for TTS: stress, explicit transcriptions, intonation instructions

Output:

- waveform

- No "right" or "wrong" output.
- Subjective evaluation of quality.
- Different (incomparable) kinds of errors.

1. Remove "hard" errors (mispronounced words, severe intonation problems)
2. Ask assessors to rate each sample on a scale of 1 to 5
3. Average the results

Rating	Quality	Distortion
5	Excellent	Imperceptible
4	Good	Just perceptible, but not annoying
3	Fair	Perceptible and slightly annoying
2	Poor	Annoying, but not objectionable
1	Bad	Very annoying and objectionable

CrowdMOS[1]

- Scores determination and screening
- CI-s estimation
- MOS estimation with crowdsourcing platforms (e.g. Amazon Mechanical Turk)

[1] Link CrowdMOS: An Approach for Crowdsourcing Mean Opinion Score Studies

Pros:

- Absolute scales
- Good for sound quality estimation

Cons:

- Scores depends on the crowdsource platform (incomparable)
- ”Hard” errors are not considered
- Bad for pronunciation and intonation assessment

MUSHRA

MUltiple Stimuli with Hidden Reference and Anchor

1. For each estimated example provide “reference” with natural speech
2. Ask assessors how similar reference and example utterances are (on a 1-5 scale)
3. Average the results

Pros:

- Absolute scale
- More stable than MOS
- Good for intonation and speaker similarity (to the original) assessment

Cons:

- Depends a lot on a test set
- Expensive to change test set
- Pronunciation issues still are not considered

1. Ask assessors to mark sentences with any kind of "hard" (perceptible and annoying) error

2. Average the results

Pros:

- SER considers "hard" errors
- Cheap collection of the test set
- Good for searching bad utterances in the set

Cons:

- Intonation error rate is quite noisy
- Not sensitive to sound quality issues

1. Make pairs of sentences (with the same text) from two different sources (two different syntheses / original and synthesis)

2. Ask assessors to choose the most preferable and correct audio for each pair

3. Average the votes for each synthesis

Pros:

- Sensitive to all kinds of issues
- Easy to estimate confidence of the score (binomial test)
- Good to compare syntheses quality

Cons:

- Relative scale
- Noisy if the syntheses are almost equal

1. Ask assessors to determine which audio is original and which is generated

2. Compute the rate of "original" answer

Pros:

- Good to estimate general impression of TTS in voice dialogs
- Considers small details of the intonation

Cons:

- Ignores overall sound quality
- Bad channel can boost the metric

Link LJ Speech

- Single speaker
- ≈24 hours
- Audiobooks, neutral intonation
- Useful for single-speaker TTS with a stable intonation

Link VCTK

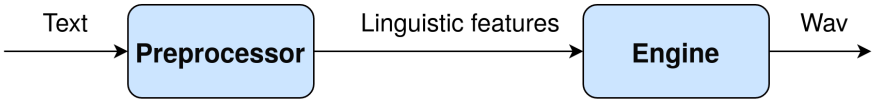
- 110 speakers
- ≈400 sentences per speaker (unique for the speaker)
- Different styles and intonation
- Useful for multi-speaker TTS, voice conversion models

Link M-AILABS (based on LibriVox)

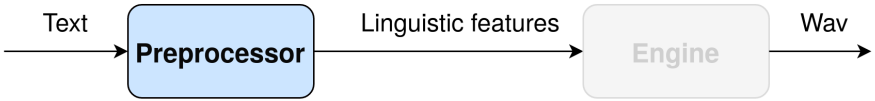
- ≈ 1000 hours
- 8 languages (incl. 47 hours of Russian)
- Different styles and intonation, mostly narrative
- Useful for multi-language and multi-speaker models

- Recorded specially for TTS construction
- Thorough selection of texts and speakers
- Clean acoustic environment
- Additional features (emotions, special accents) can be added
- Very expensive to create

TTS Pipeline

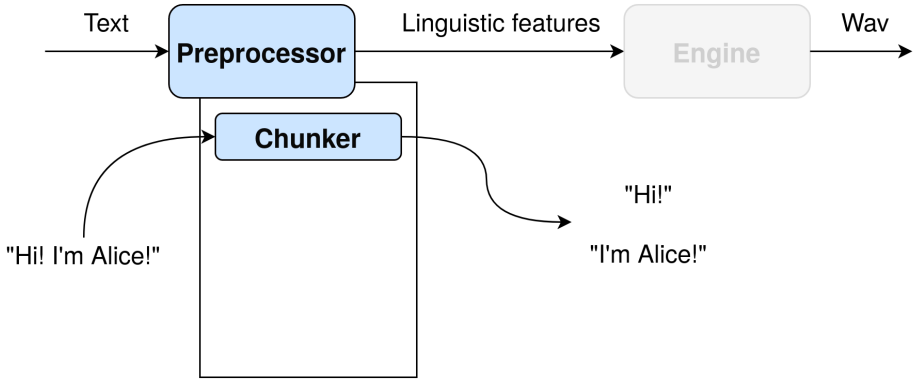


Preprocessor

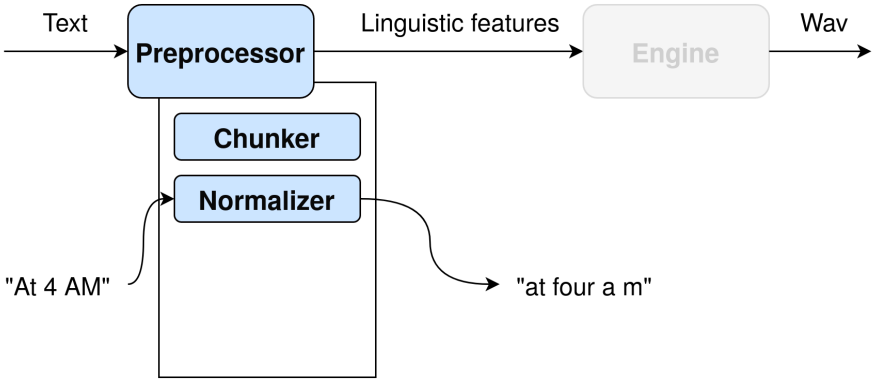


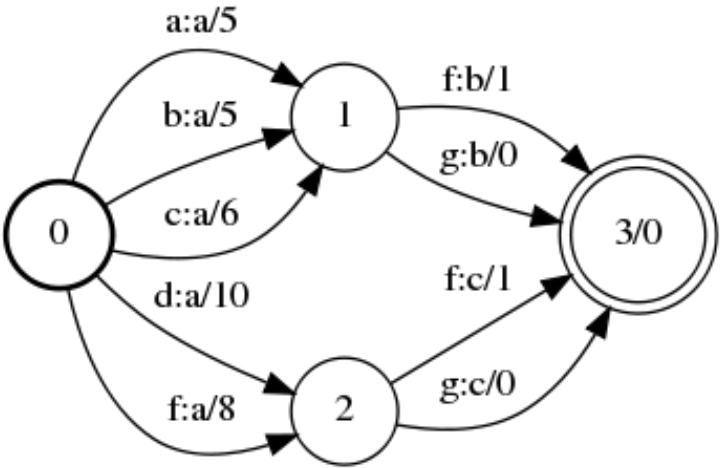
- Sentences splitting: *"Hi! My name is Alice."* $\rightarrow$ *"Hi!"*, *"My name is Alice."*
- Number and abbreviations expansion: *"42 m"* $\rightarrow$ *"forty two meters"*
- Homographs disambiguation: *"fast increase"* $\rightarrow$ *"fast ́increase"*
- Transcribation: *"a great idea"* $\rightarrow$ /ə/ /greɪt/ /aɪˈdiə/

Chunker


Text
Preprocessor
Linguistic features
Engine
Wav
Chunker
"Hi! I'm Alice!"
"Hi!"
"I'm Alice!"

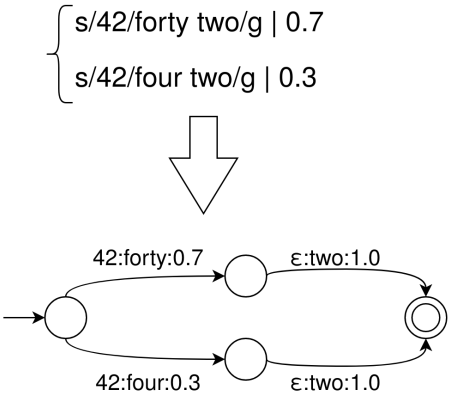
Normalizer


Text
Preprocessor
Linguistic features
Engine
Wav
Chunker
Normalizer
"At 4 AM"
"at four a m"


a:a/5
b:a/5
c:a/6
d:a/10
f:a/8
f:b/1
g:b/0
f:c/1
g:c/0
0
1
2
3/0

WFST

- WFST are equal to weighted regex substitutions
- Union and concatenation operations corresponds the same in regex-s
- With the composition operation superposition of substitutions can be implemented
- With WFST complex and domain-specific normalization rules can be applied

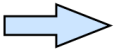


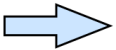
Pros:

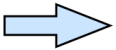
- Fast
- Easy to create new rules and handle corner cases
- Easy to combine basic substitutions using

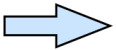
Cons:

- Local, not considering context
- Rules could interfere producing unpredictable results
- Manual rules creation is complex

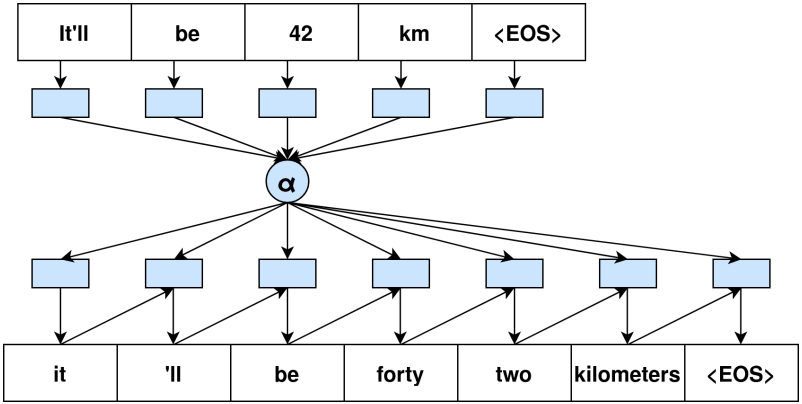
д. 2 к. 3 

1961 г. 

д. 2 к. 3  дом две кубические копейки

1961 г.  тысяча девятьсот шестьдесят один грамм

Seq2Seq Neural Network



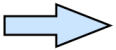
Pros:

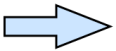
- Context-dependent
- With the proper traing data can handle different with manual control
- Can utilize synthesized data

Cons:

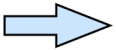
- Very complex to control corner cases
- Hard to collect enough dara
- Unstable and unpredictable results
- More computationally heavy

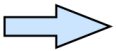
Seq2Seq Normalization

habr.com 

@? 

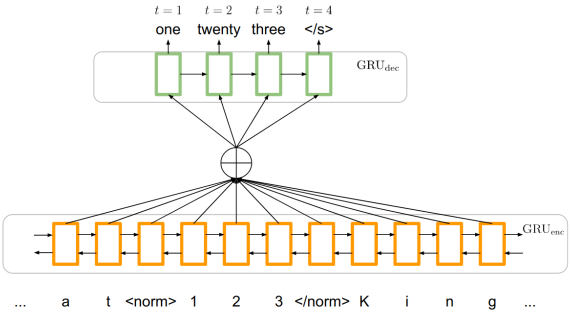
Seq2Seq Normalization

habr.com  habr точка командир

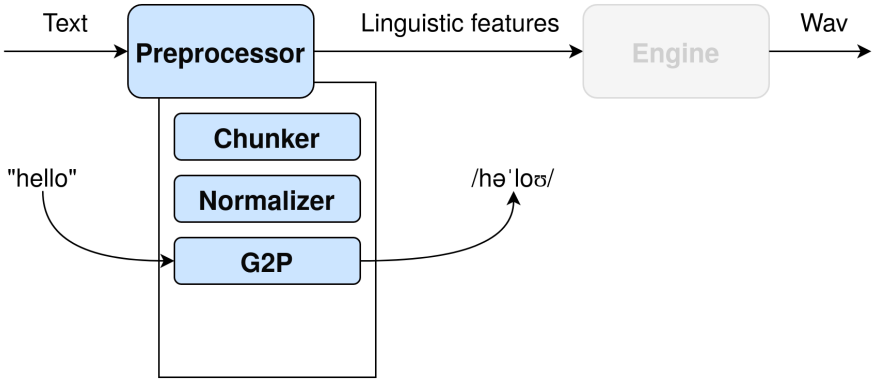
@?  собака адрес?

Hybrid Approach

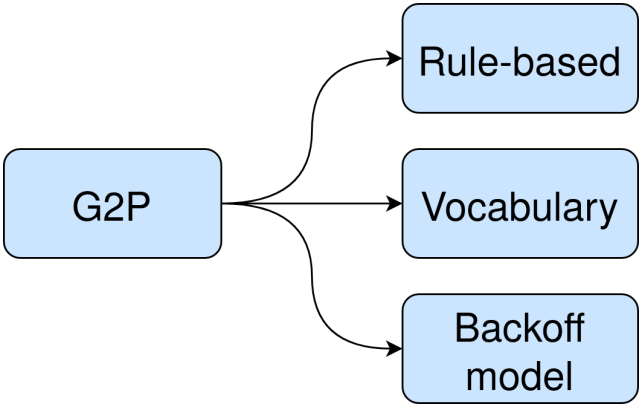
1. Mark the text need to be normalized with separate model.
2. Generate hypotheses for the marked text.
3. Filter the hypotheses with rule set or FST grammar.



[1] Link Neural Models of Text Normalization for Speech Applications


Text
Preprocessor
Linguistic features
Engine
Wav
Chunker
Normalizer
G2P
"hello"
/həˈloʊ/

- Homographs: "import" — /ˈɪmpɔrt/ or /ɪmˈpɔrt/
- Foreigh words: "A short poem à la Ogden Nash" — /ɑ lɑ/
- Abbreviations: "CI" — /si/ /aɪ/


G2P
Rule-based
Vocabulary
Backoff model

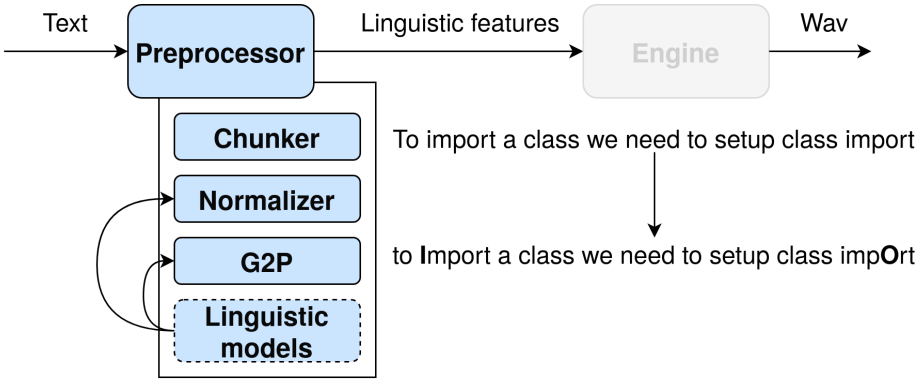
G2P Pros:

- Separate G2P engine can serve for several TTS backends
- Easy to control and fix certain words pronunciation
- Easy to control language in mixed queries

G2P Cons:

- More expensive markup
- Additional components — additional bugs
- Textual information (capitalization, punctuation, spelling) is lost
- Not all languages require G2P

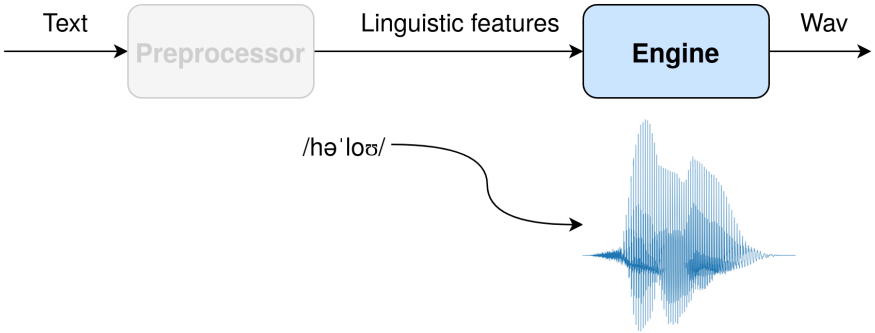
Language model information



Break!

10 min

Engine



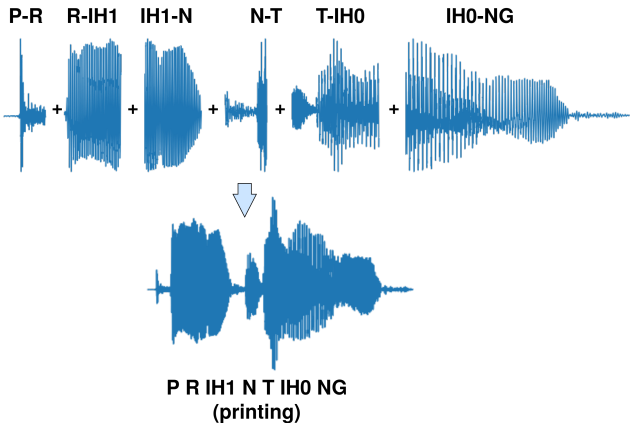
Duration Problems

- Dimensionality gap:
  - Average duration of phoneme in Russian $\approx$ 20-150 ms
  - Duration of 22.05 kHz PCM sample $\approx$ 0.045 ms

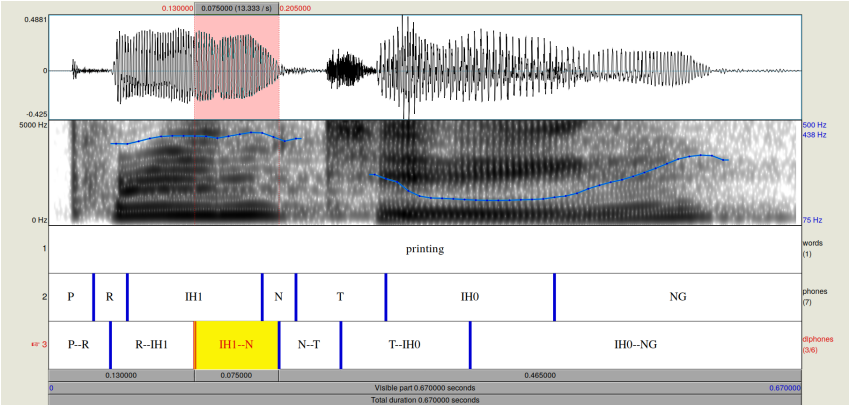
- The same phoneme can have different durations depending on context

Concatenative Synthesis

- Let's just copy human voice!
- Split recording at some phoneme-labeled chunks
- Construct phrase from these chunks at the inference



Diphones



- Midpoint — the most "stable" point in phoneme
- There is $(\#Phoneset)^2$ different diphones
- Diphone concatenative TTS is ok for non-expressive speech

Pros:

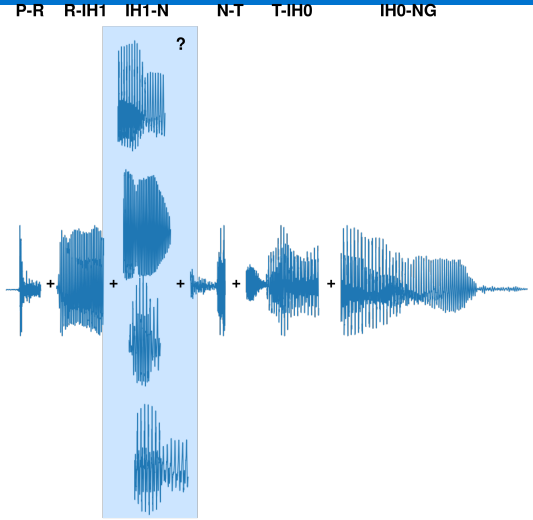
- Very fast
- Requires little RAM

Cons:

- Bad quality, annoying joins
- Unable to generative expressive and well-intonated speech

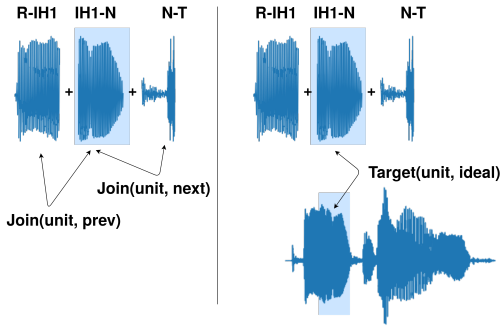
Unit Selection

- Let's store multiple samples (units) for each diphone
- And select the most suitable one at the inference phase
- Let's use join and target cost functions to determine which unit is better



Unit Selection

- Join – how compatible the unit is to the nearest ones
- Target – how good the unit suites to some ideal general intonation contour
- General cost for an utterance: $C = \sum_i Join(u_i, u_{i-1}) + Target(u_i)$
- We can use beam search for inference
- Average complexity: $O(L \times N)$
  - L - length of the utterance
  - N - average number of units per diphone in database



We can use ML methods to train target and join:

- For target: predict ideal contour with neural net and compute distance to it
- For join: use similarity networks

Neural net-based unit selection was the SotA for TTS quality for 2017.
This technogy was used for previous generations of voice assistants:

- Amazon (Alexa)
- Apple (Siri)[1]
- Yandex (Alice)

[1] Link Deep Learning for Siri's Voice

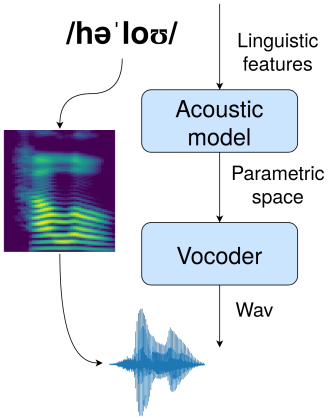
Pros:

- Fast inference
- In case of good concatenations output speech quality is equal to an original one
- We can handle different speech modes by adding new data to database

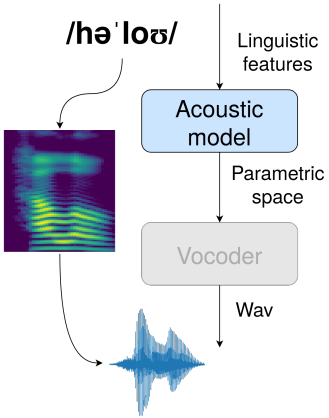
Cons:

- Need several tens of hours in database to achieve ok quality (tens of Gb of RAM)
- In case of bad joins generates annoying artifacts
- Speech quality drastically decreases outside pre-recorded domain

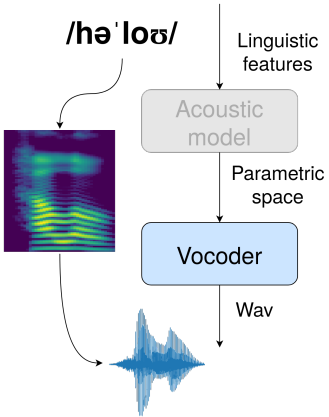
Parametric TTS



- To overcome dimensionality gap let's generate sound in two steps
- Firstly, generate a compact *parametrical representation* of the sound - e.g. mel-spectrogram
- Secondly, convert this compact representation into a full-scale waveform



- Acoustic model (AM) is a component responsible for creating intermediate sound representation
- Different generative models were used in AM
- Initially designed with decision tree and HMMs
- Today contemporary AM-s are attention-based neural networks
- AM-s are responsible for high-level quality of sound (intonation, articulation, duration and temp)



- Vocoder transforms sound into an output waveform
- Initially vocoders were deterministic DSP (digital signal processing) algorithms
- With the improvement of the hardware and DL development they were replaced with neural nets[1]
- Today neural vocoder is the heaviest part of para-sythesis
- Vocoder is responsible for low-level sound quality (absence/presence of artifacts, loudness, sampling rate)
- Neural vocoder-based TTS is a SotA in quality for the moment

[1] Link Wavenet: A Generative Model for Raw Audio

Pros:

- Very flexible and extendable (emotion, expressive synthesis etc.)
- Can train on a non-perfect data
- Produces SotA in quality and naturalness for today
- Attention-based AM-s can make do without GPU

Cons:

- Heavier computations (neural para-TTS requires a CPU core (or even GPU) to run just realtime)
- Neural para-TTS requires large amount of data (comparing to diphone TTS)
- Long time to train

Questions?