Automatic Speech Recognition
Research Master 2020-21

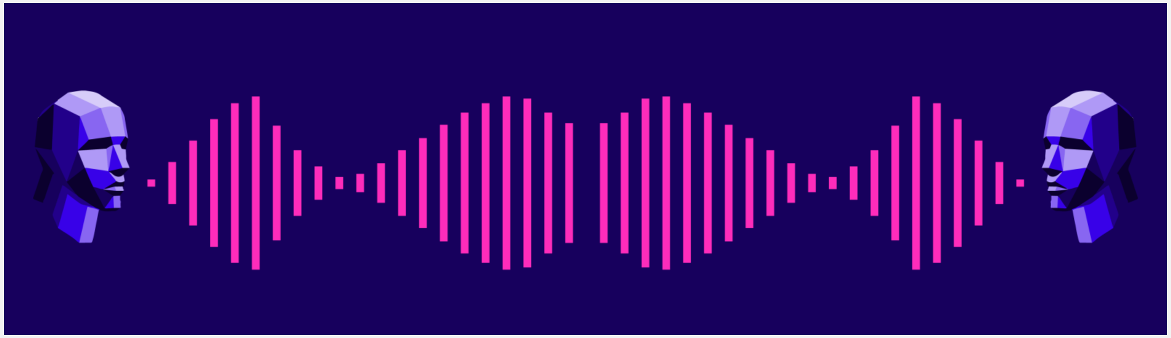
Radboud University

Nijmegen

Louis ten Bosch, E8.16

ASR is a dynamic field.
Research is changing rapidly.

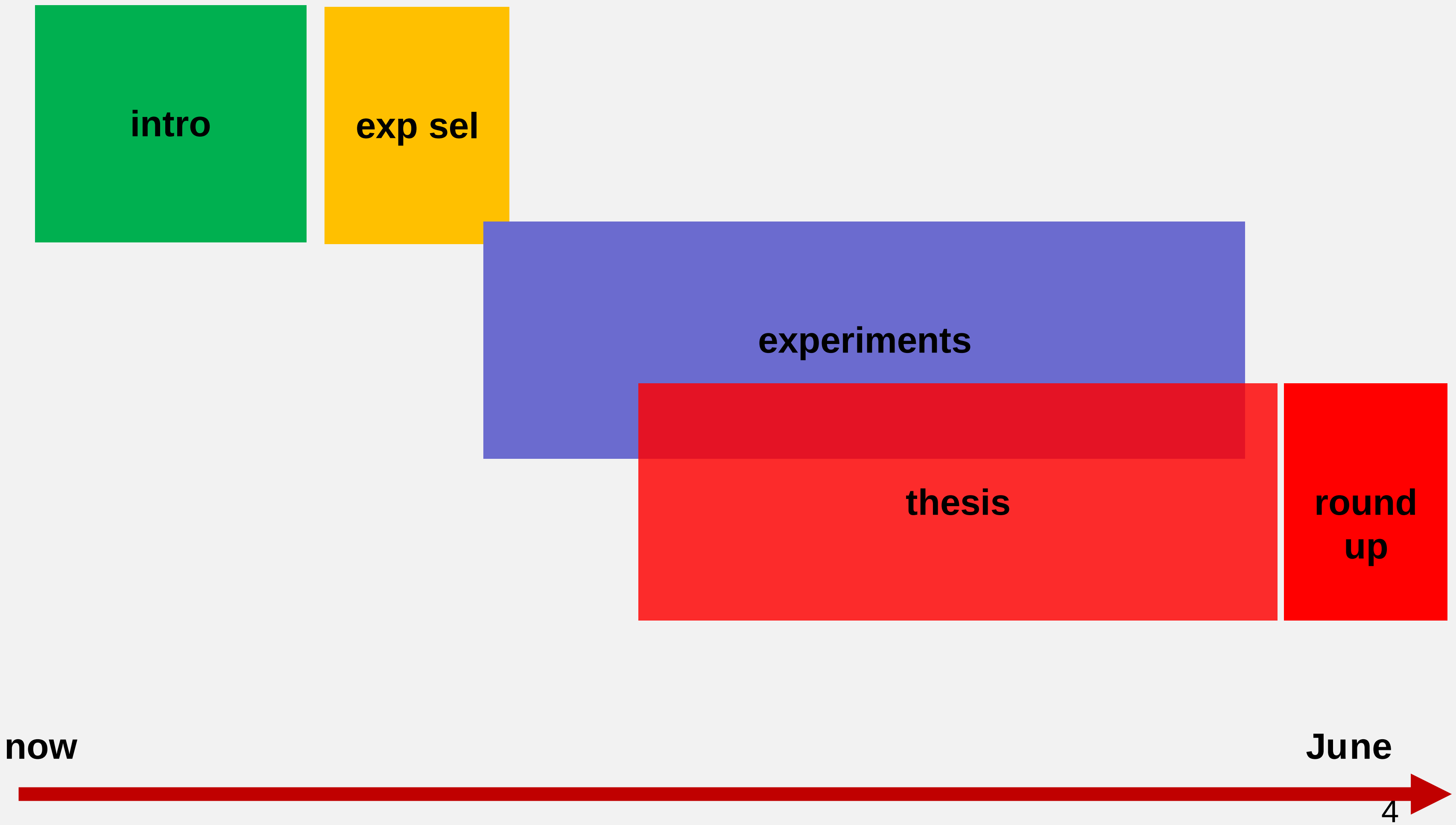
ASR is a broad field.
We only discuss the main issues.



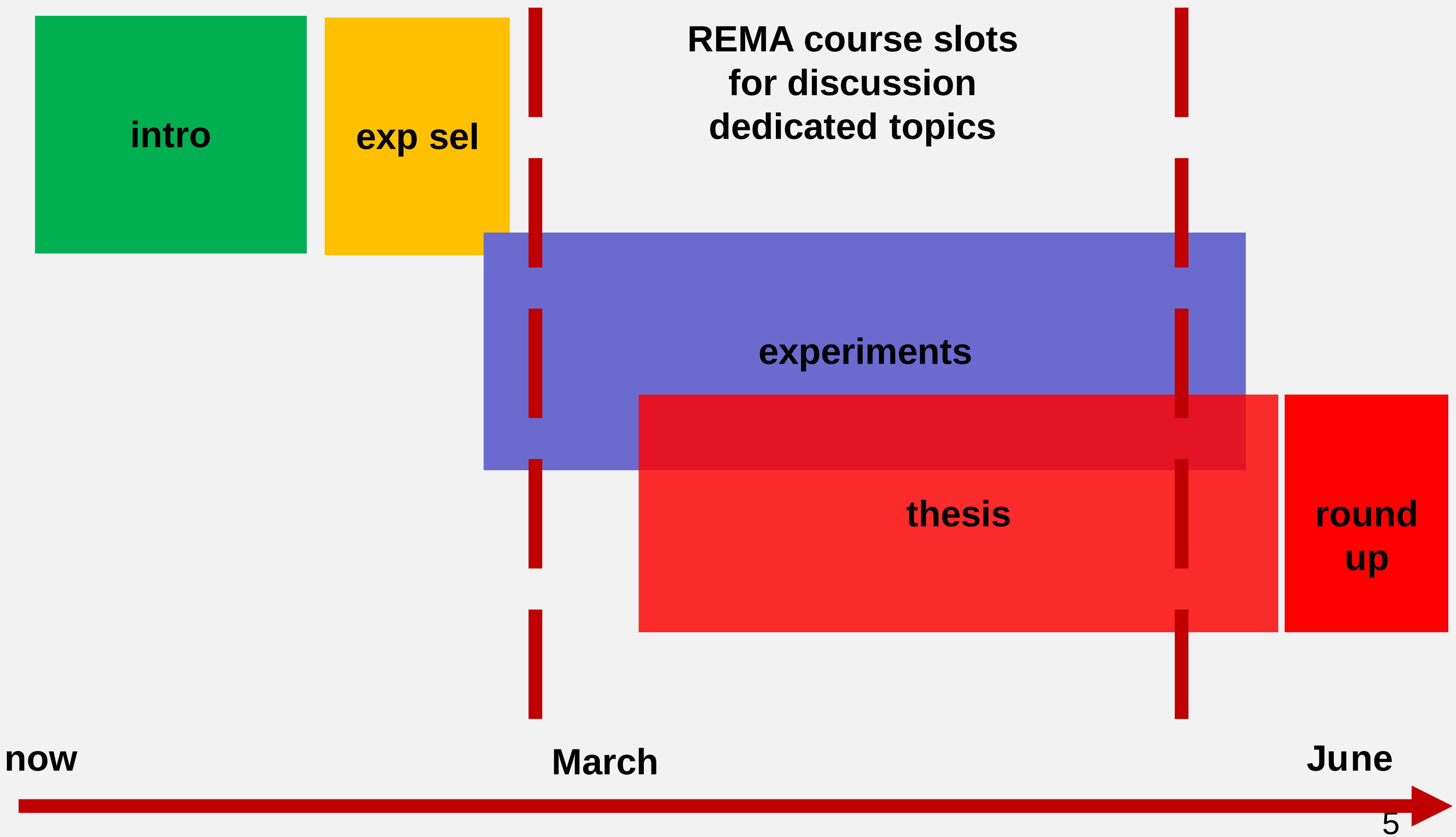
Schedule

- Wednesdays, start 15h30
- every week until June 2 (incl.)
  - with exceptions – see course schedule

Agenda



Agenda


intro
exp sel
REMA course slots
for discussion
dedicated topics
experiments
thesis
round
up
now
March
June

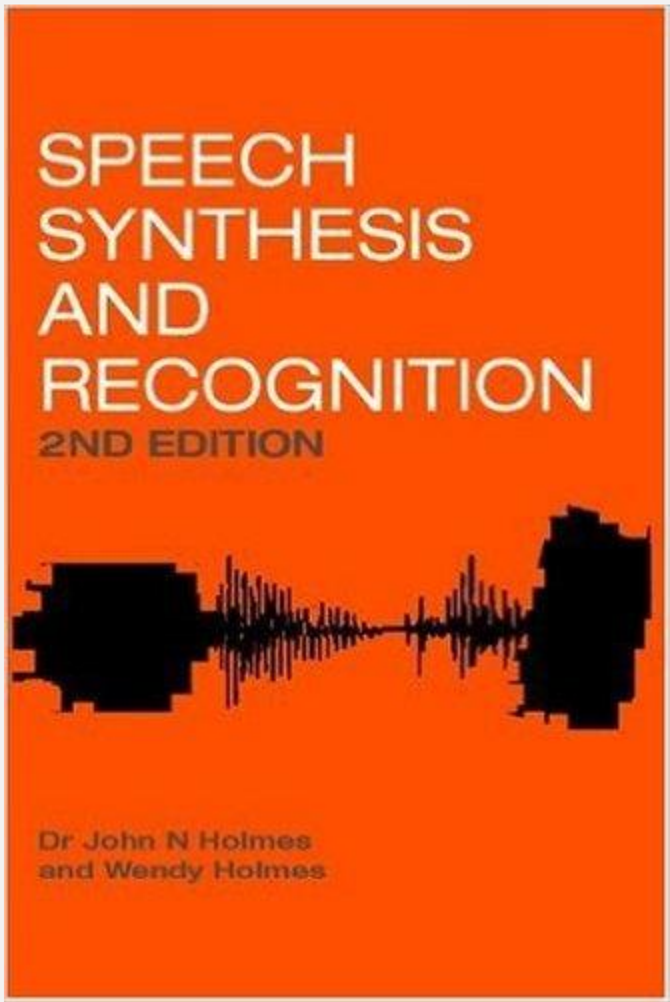
Global agenda this course

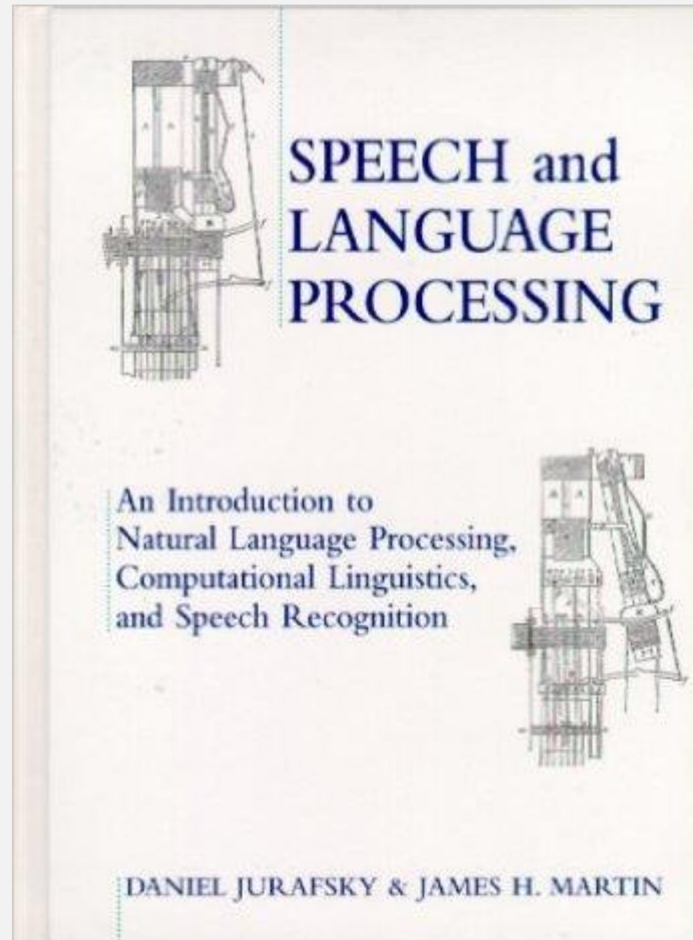
- Introductory part (approx. three weeks)
  – phonetics, speech, global ASR architecture
  – see also reading material (BrightSpace)
- Computational/experimental part (twelve weeks)
  – exploration, preparation (weeks 4 to 5)
  – experiments (individual or in groups; weeks 5 to 12)
  – reporting by individuals
- Goal: thesis (individual)
  – deadline: final version: June 15

Reading

- Books: e.g.
  - Classical ASR: Holmes & Holmes (pdf on BrightSpace; see ASR chapters)
  - Jurafsky & Martin (more up-to-date)
- Papers (much more up to date)
  - there is a very large body of recent papers
    - about many highly specialized topics
  - Examples on BrightSpace
- These slides
  - will be updated on BrightSpace on a regular basis

Books


SPEECH
SYNTHESIS
AND
RECOGNITION
2ND EDITION
Dr John N Holmes
and Wendy Holmes

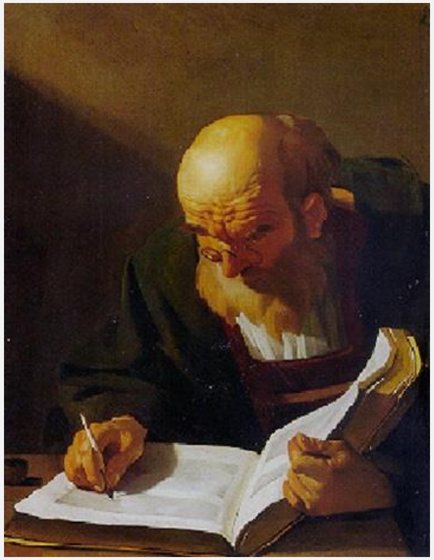

SPEECH and
LANGUAGE
PROCESSING
An Introduction to
Natural Language Processing,
Computational Linguistics,
and Speech Recognition
DANIEL JURAFSKY & JAMES H. MARTIN

Aim of this course

- Provide background, insight
- Research oriented
- Indicate issues in ASR
- Invite future outlook

Thesis

- Topic:
  - ASR is way too broad to be dealt with in 16 weeks
  - choose your topic narrow enough
  - usually based on an experiment
  - but may also be theoretical
- Starting point: a research question
- Experiment: individual or in a group
- Writing: individual
- Assessment: individual (see doc on BrightSpace)

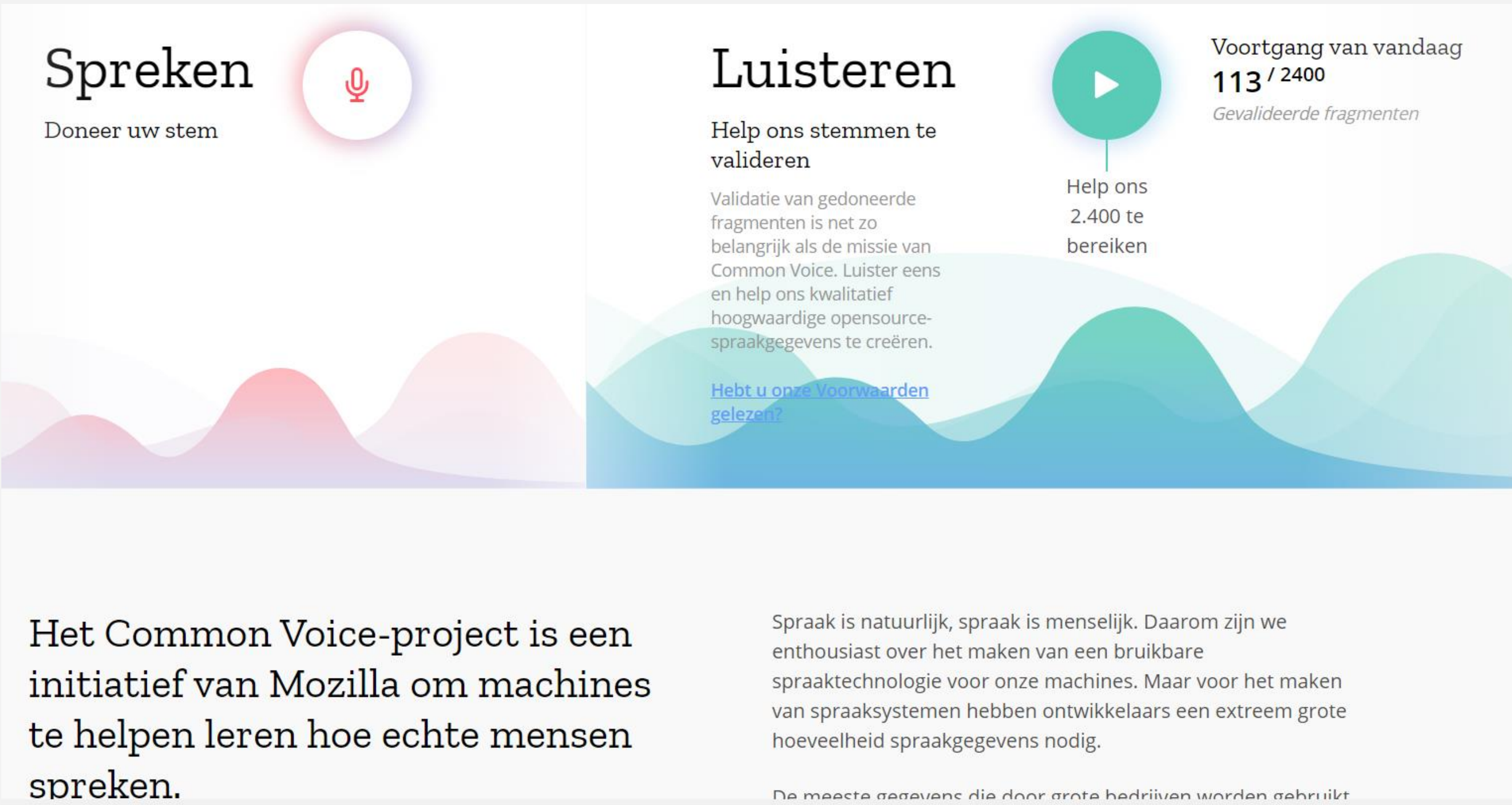


Experiments

- You may choose your own platform
- For experiments on Ponyland (science.ru.nl)
  – login
  – LINUX, perl/python, bash
- Proposals for experiments after the introductory stage (week 4 and later)

Crowd sourcing speech

https://commonvoice.mozilla.org/nl

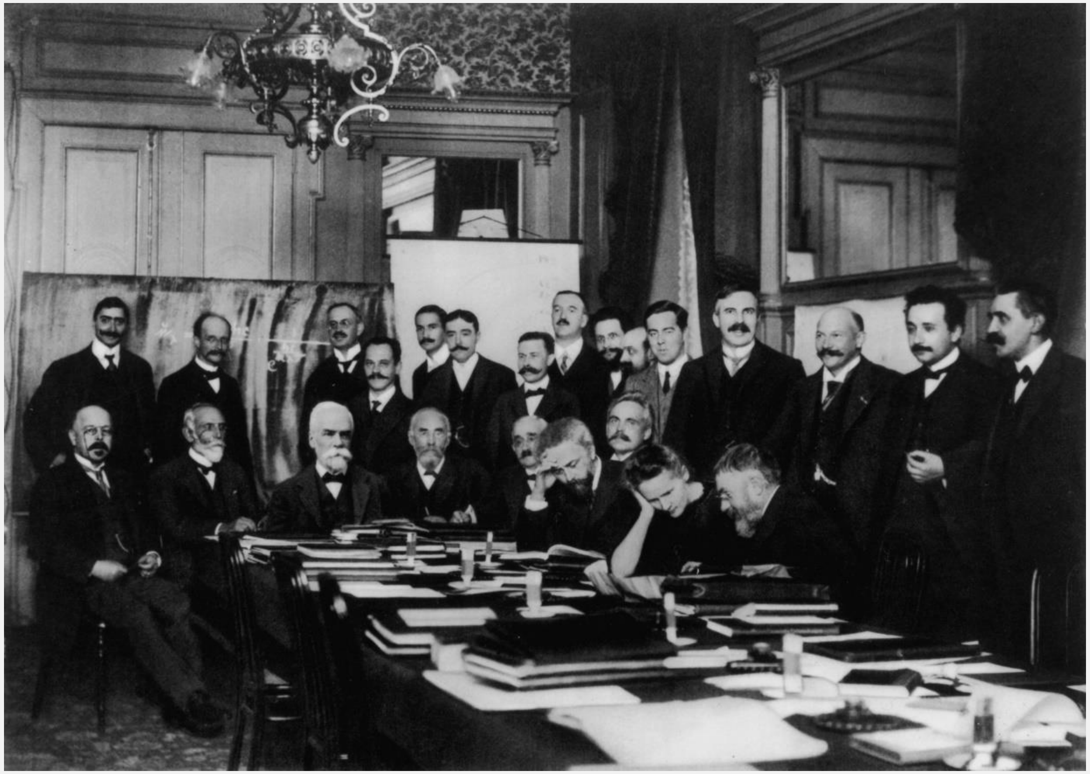


Options for experiments

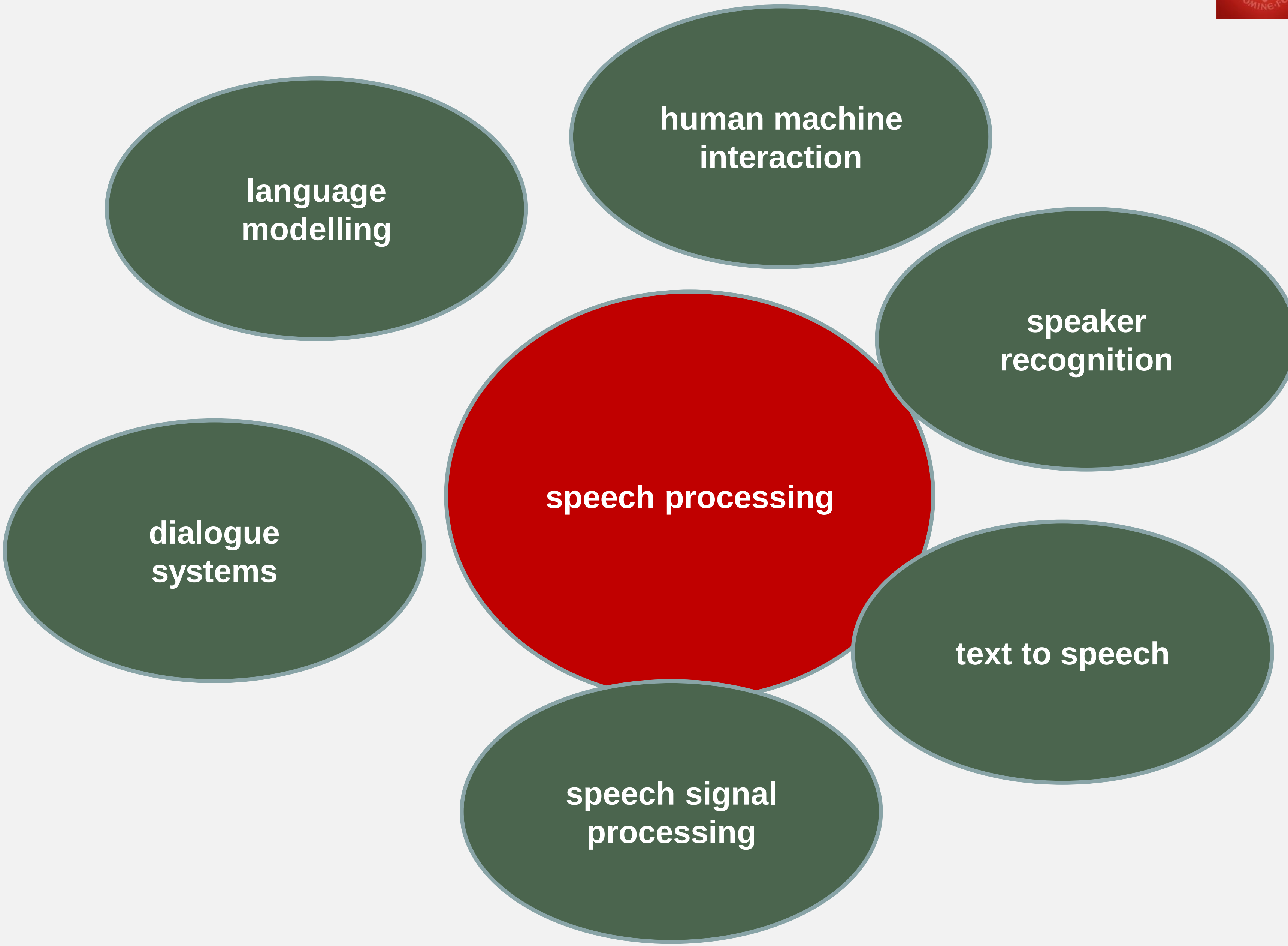
just examples

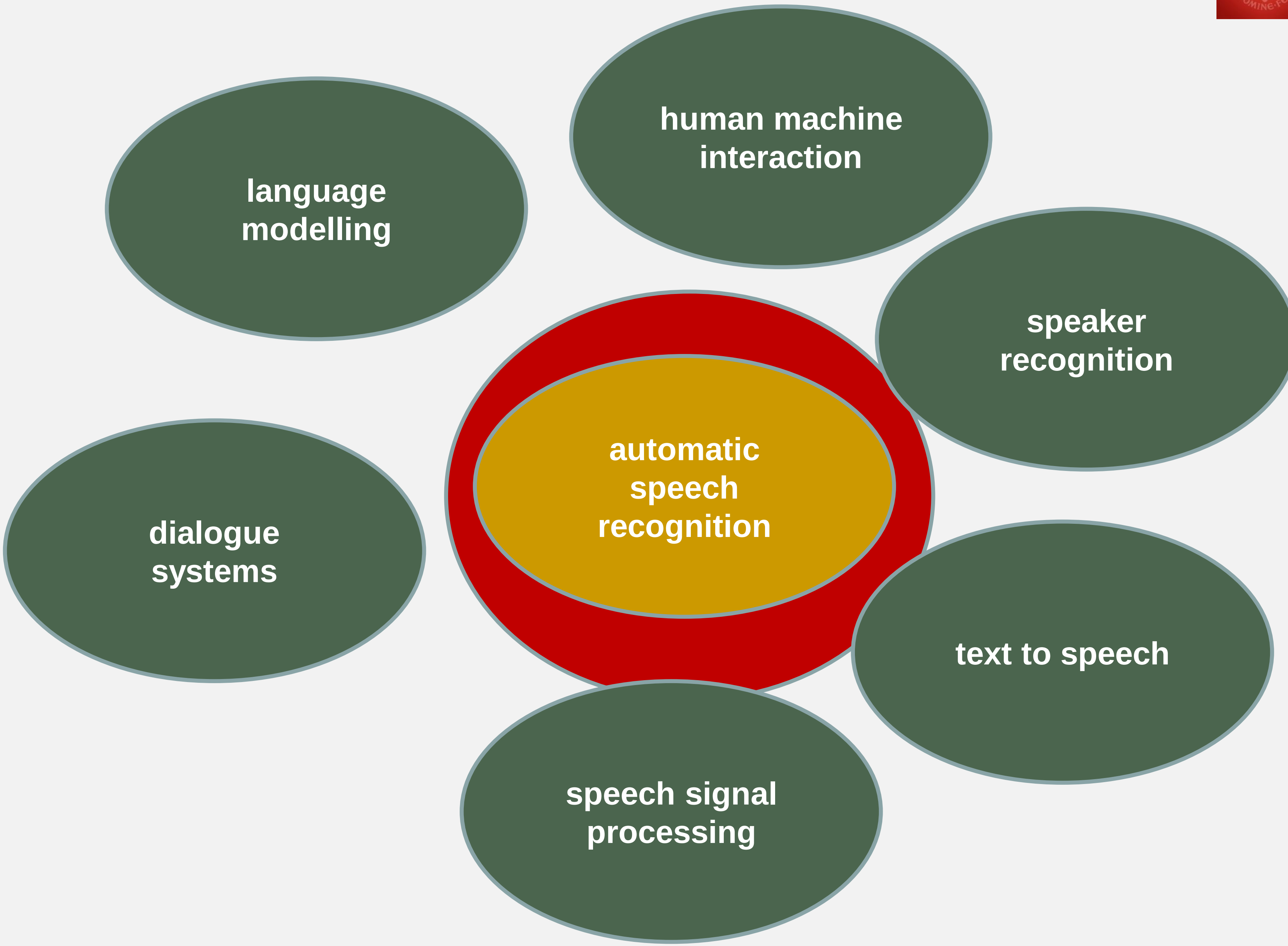
1. ASR (audio → text) for command and control (Dutch) (github Jurriaan)
2. Keyword spotting for Dutch (Tara Sainath papers)
3. Personalized ASR
4. EARSHOT (Magnuson et al., 2020) (github Pepijn)
5. Estimation of #talkers in multi-talker audio input (audio → {0,1,2,3,…})
6. Acoustic/phonological feature detectors (audio → vector)
7. Isomorphisms between latent structures in DNNs (X-H-Y, X-H2-Y)
8. Relation between *acoustic* and *symbolic* distance (log prob score and levenshtein distance, e.g. bull – bill – ball, morse – horse – force)
9. Automatic improvement readability of ASR output (+ punctuation)
10. Voice morphing and unmorphing (teams Alice, Bob, Marc)
11. Audio unshredding (continuity constraints)
12. Detection of fake audio (how)

Audience?



- For now:
  - rondje
  - background, your own interest in ASR
  - AI/Data Science/Linguistics/…

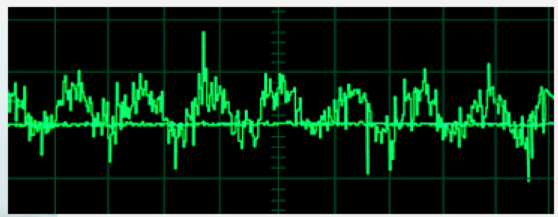

human machine interaction
language modelling
speaker recognition
speech processing
dialogue systems
text to speech
speech signal processing


language modelling
human machine interaction
speaker recognition
automatic speech recognition
dialogue systems
text to speech
speech signal processing

ASR = Speech to Text

Automatic Speech Recognition is the process in which a computer converts speech into text

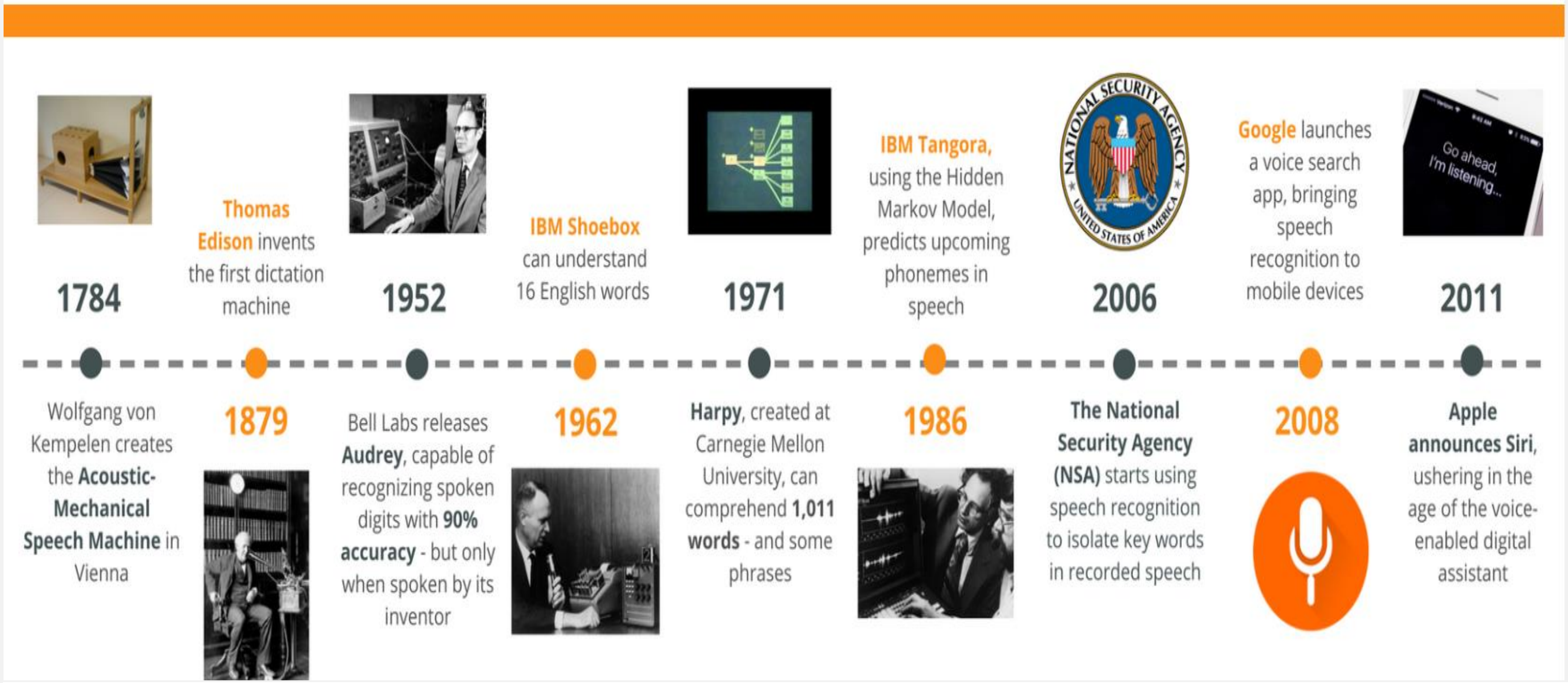
No understanding. No semantics!



"…. these cells indicate ..."

ASR
long history since 1750, breakthrough since DNNs (2010)

Jan 10 · 18 min read



ASR is now common business

- Human machine interaction
  - – speech is the most natural means of communication ~ 100000 y
  - – ease of communication
  - – safety (pilots, drivers, etc)
  - – communication with robots, care bots
  - – hands-free
  - – teacher, instructor, caregiver
- Linguistics (e.g.), research
  - – automatic determination of dialect, speaking style, mood, …
  - – large-scale processing of speech corpora

ASR application areas

- Commercial speech recognition applications
  - voice dialing, …
  - dictation programs (law, medical, office, …; avoidance of RSI)
  - speech interface for games
  - embedded (robots, car, home, ambient intelligence…)

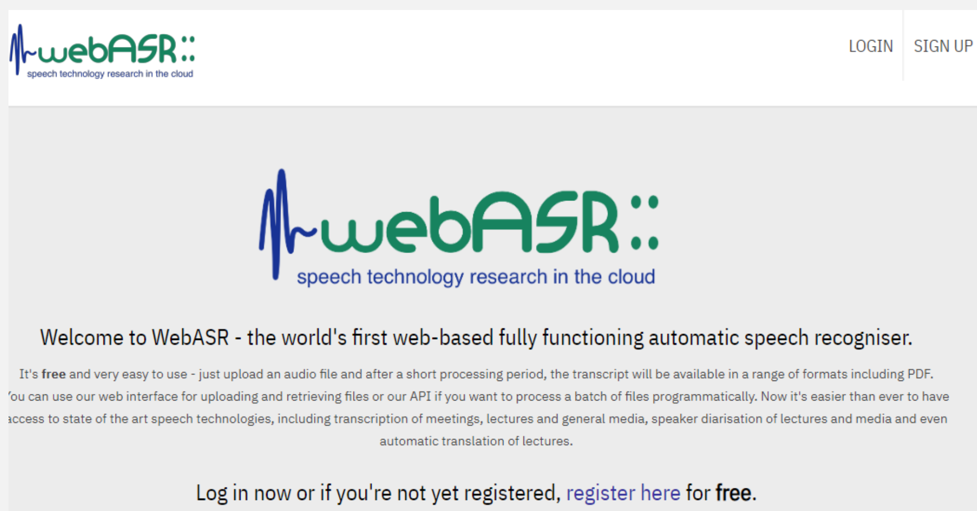
- Education/pathology (e.g.)
  - CAPT, CALL
  - assessment of pathological speech

ASR application areas

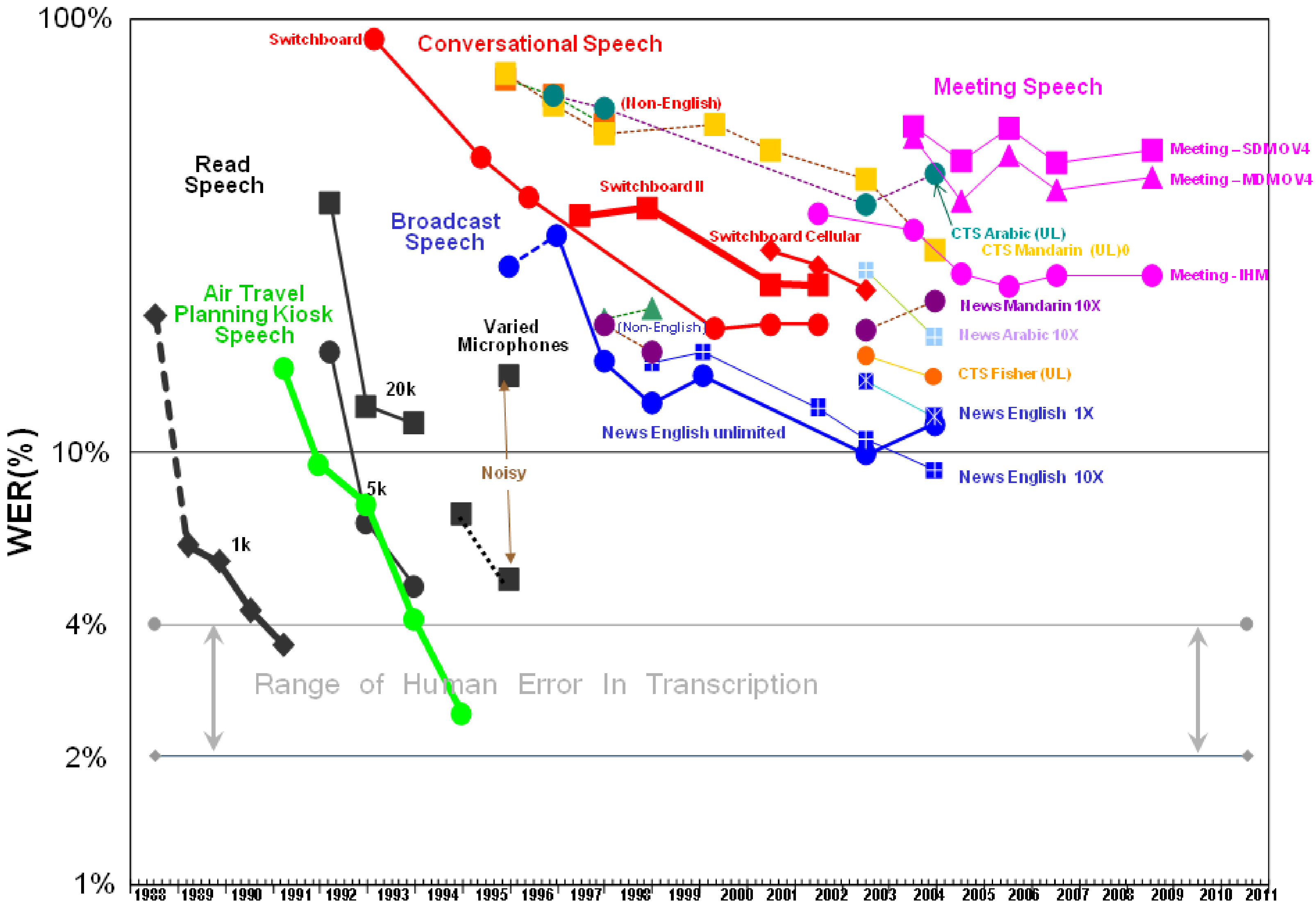
- ASR =?= a model of Human Speech Processing (HSP)
  - By **modelling** human speech comprehension we may learn more about the comprehension process itself
- Issues:
  - plausibility of the processes
  - plausibility of the representations
  - how to get from subsymbolic information to symbolic information?

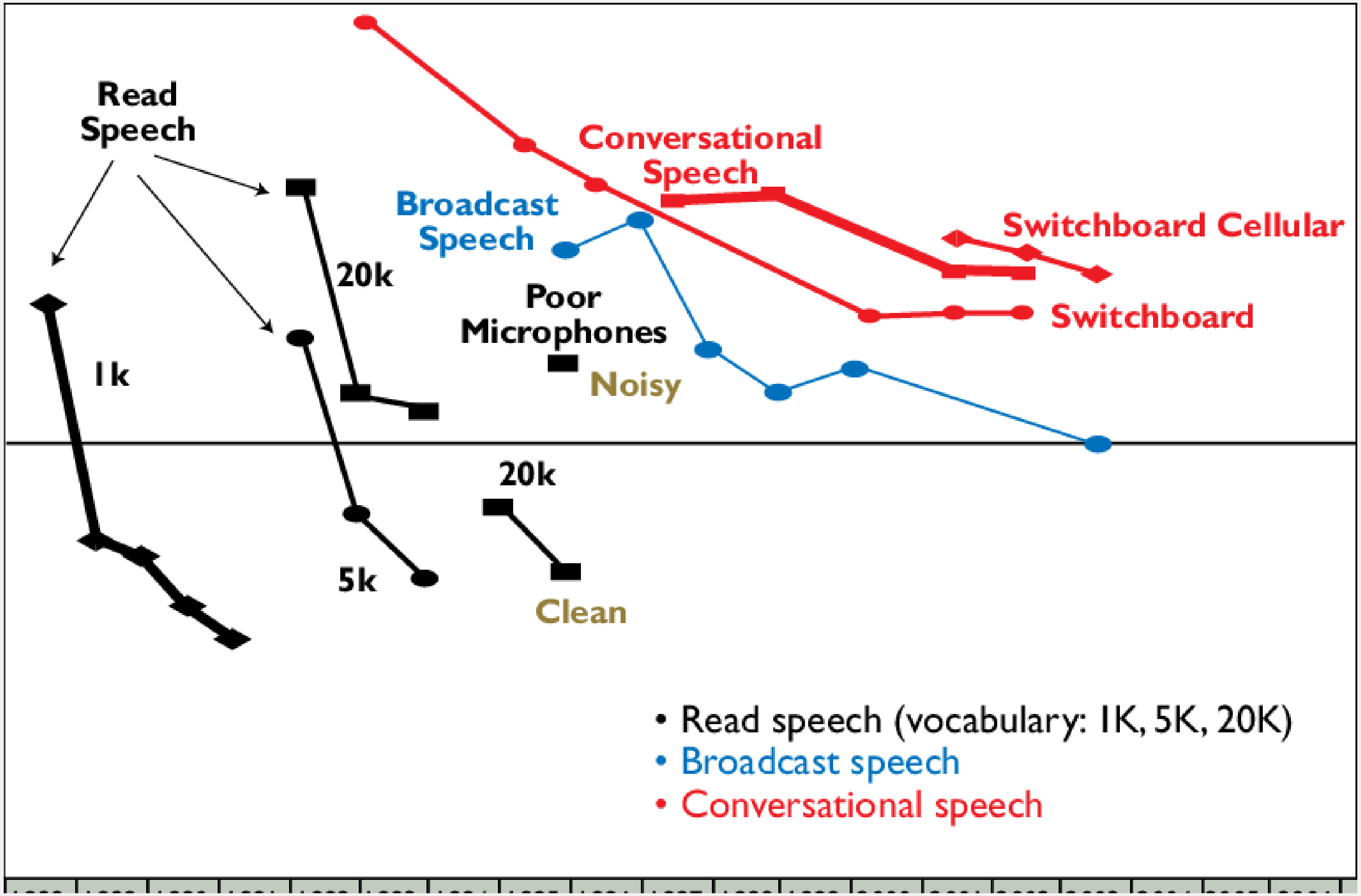
ASR as webservice

- webasr.org
- https://nlspraak.ewi.utwente.nl/ (ENG/NL)
- https://cls.ru.nl/online-asr/
- https://alphacephei.com/vosk/

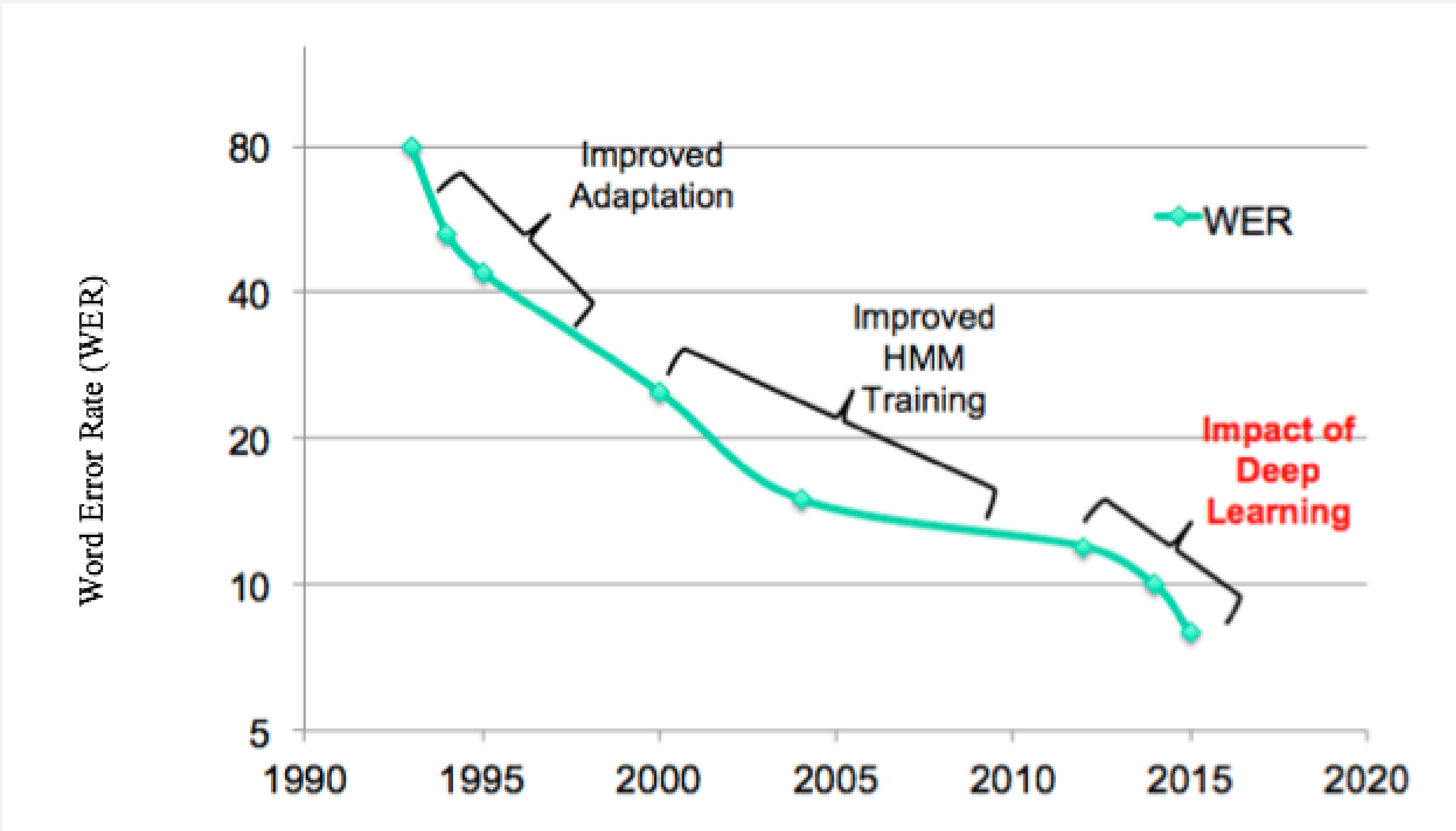


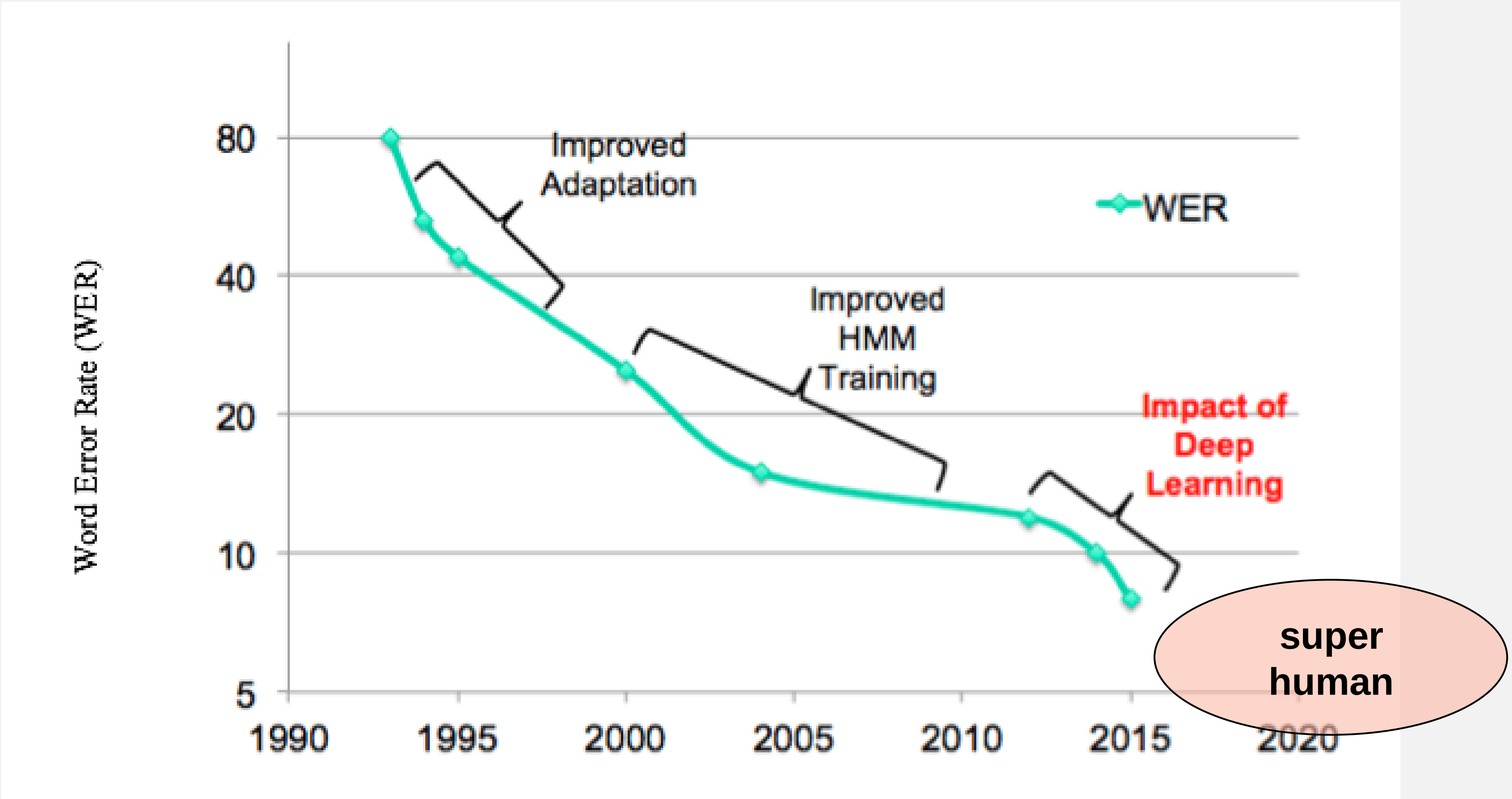
NIST STT Benchmark Test History – May. '09





(By Xuedong Huang, 2016)


Word Error Rate (WER)
80
40
20
10
5
1990
1995
2000
2005
2010
2015
2020
Improved Adaptation
Improved HMM Training
Impact of Deep Learning
WER


Word Error Rate (WER)
80
40
20
10
5
1990
1995
2000
2005
2010
2015
2020
Improved Adaptation
Improved HMM Training
Impact of Deep Learning
WER
super human

Four ways to look at ASR

- ASR as tool (black box) in larger system
  - In a dialogue system
  - As a tool for the hearing-impaired
  - In cases where typing is dangerous/impossible
- ASR as a computational model for human speech processing
  - Not straightforward, plausibility issues
- ASR as a tool to unravel the structure in the speech signal
  - Automatic segmentation into phones
- ASR as a research tool itself
  - WER minimization, noise robustness etc.

ASR can be used in different ways

- For word recognition
  - open domain (any topic, most complex)
  - narrow domain, e.g., restaurants in Nijmegen (less complex)
  - Dictation (medical, court, police, broadcasting, subtitling, etc.)
- For keyword spotting
  - Simpler than full-blown speech recognition
  - For topic monitoring of thousands of telephone conversations in parallel (wakeup calls e.g. “hello siri”, security, eavesdropping)
- For forced alignment
  - For research purposes (onsets and offsets of words, syllables)
  - Medical assessments of voice quality, etc.
- **The main difference is in the WORD SEARCH SPACE**

Complexity of an ASR task

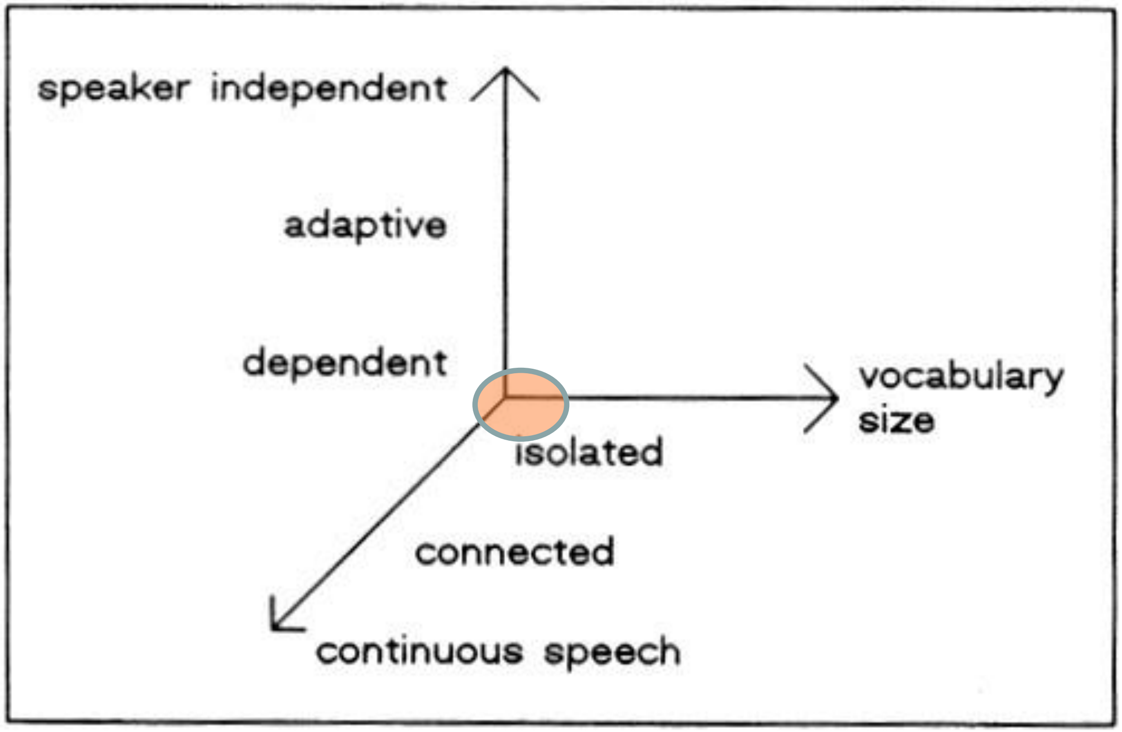


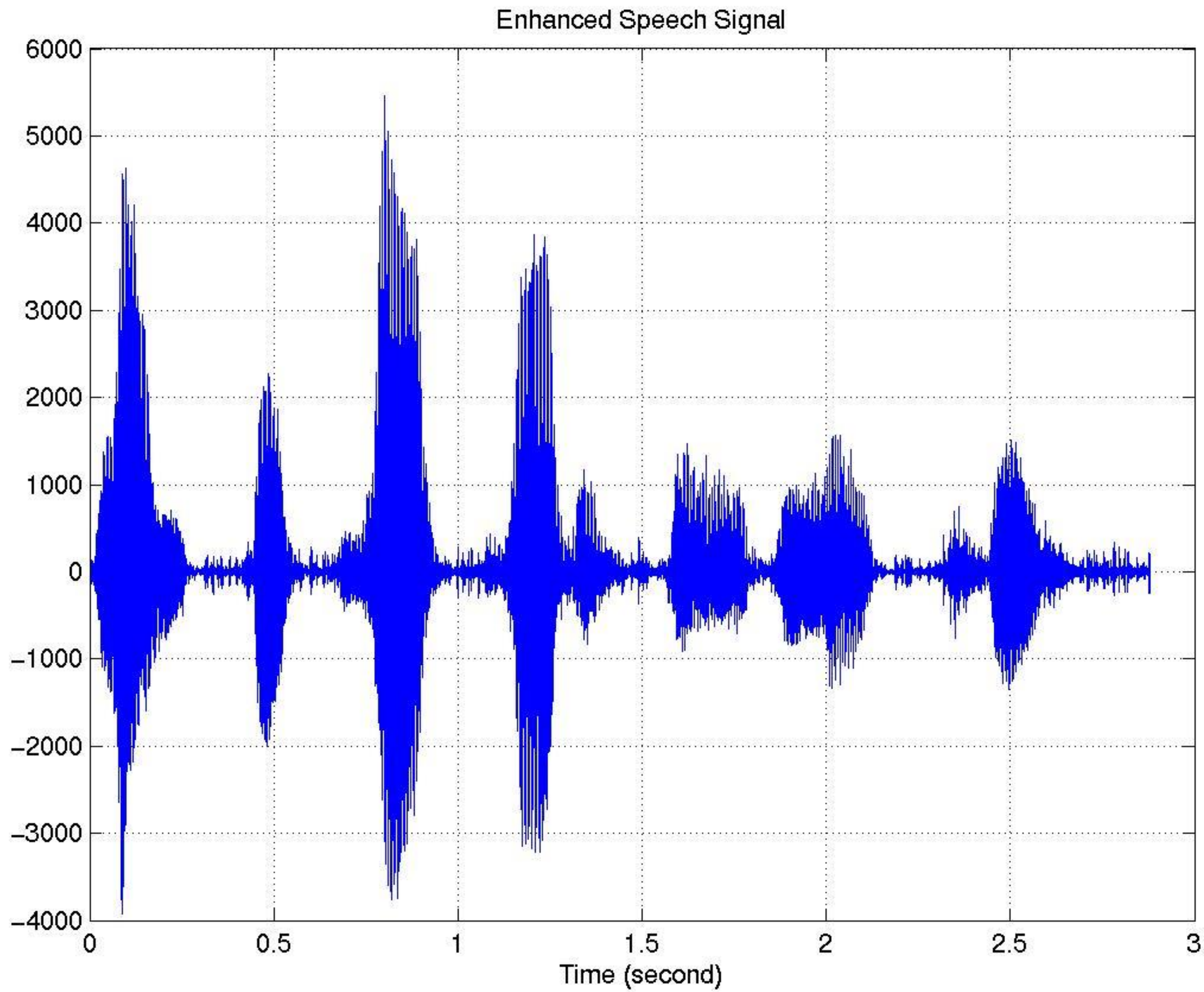
Figure 7.5. A three-dimensional space defined by the different functionalities provided by a recognizer.

RADBOUD UNIVERSITEIT NIJMEGEN

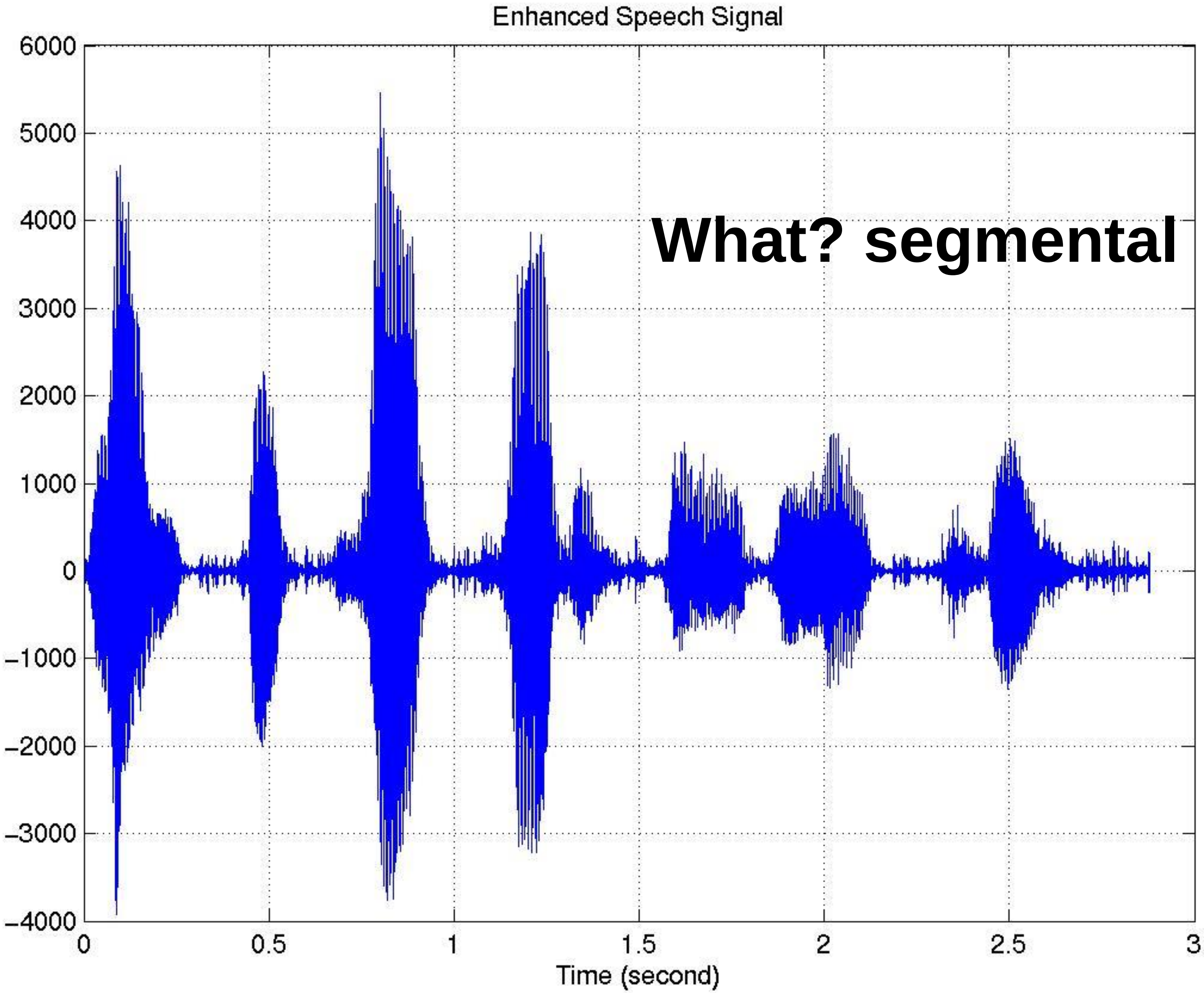
Characteristics speech signal

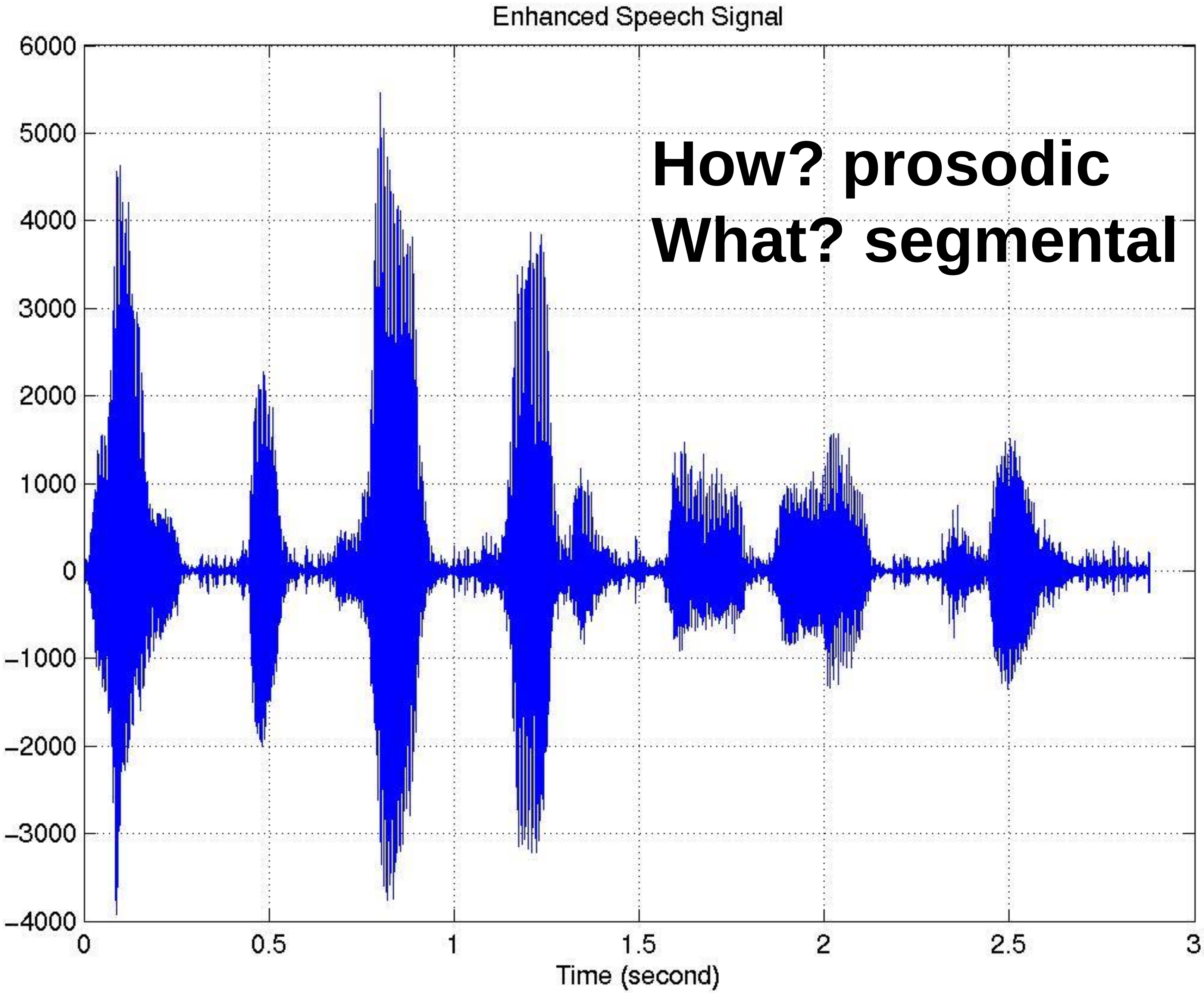
- No symbols
- No segmentation / boundaries / word boundaries
- Enormous variation (speakers, accents, mood, …)
- Background noise

- *Totally different from text!*
- *Speech is crucial for language evolution and language acquisition*

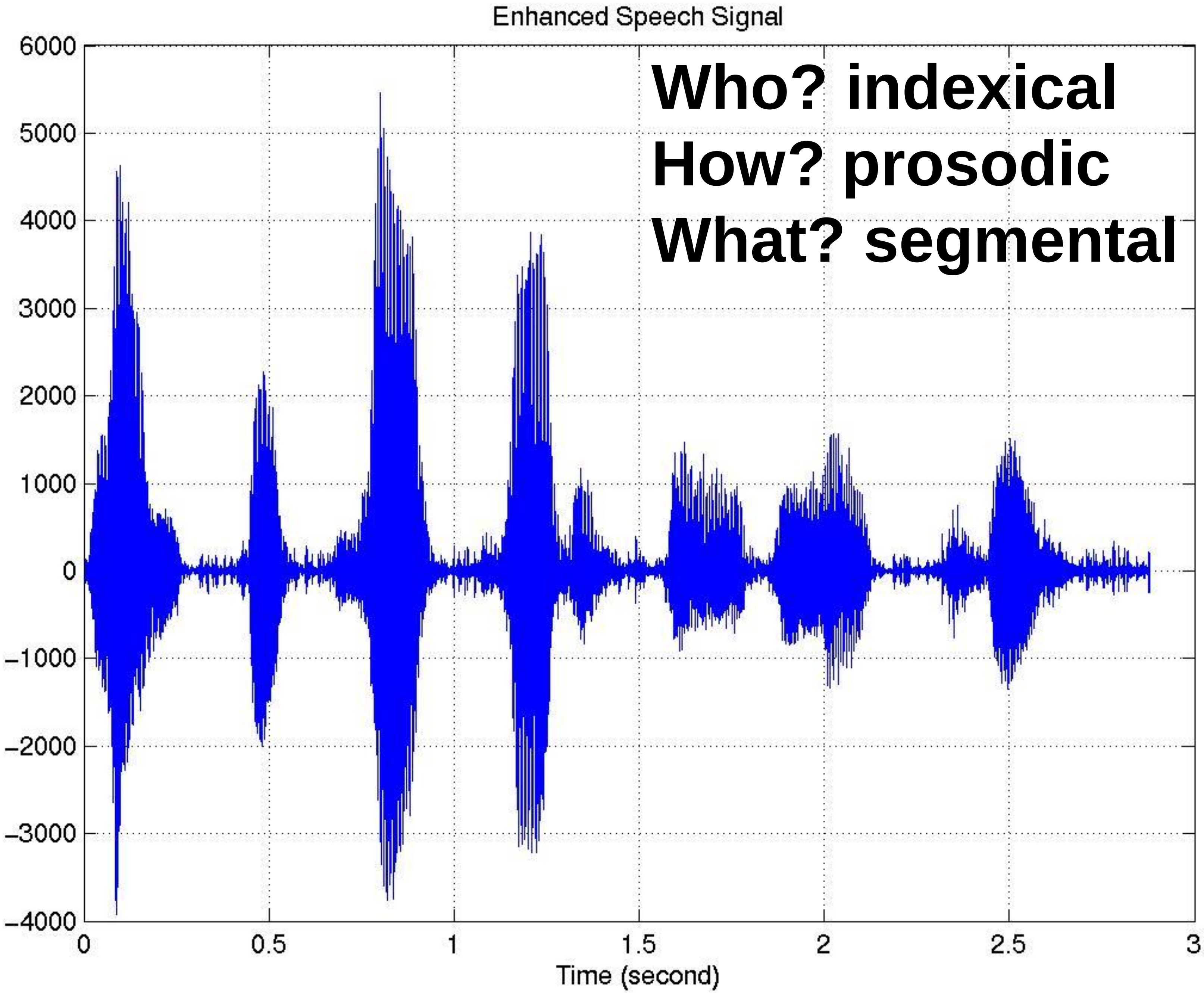


Enhanced Speech Signal
6000
5000
4000
3000
2000
1000
0
−1000
−2000
−3000
−4000
0
0.5
1
1.5
2
2.5
3
Time (second)

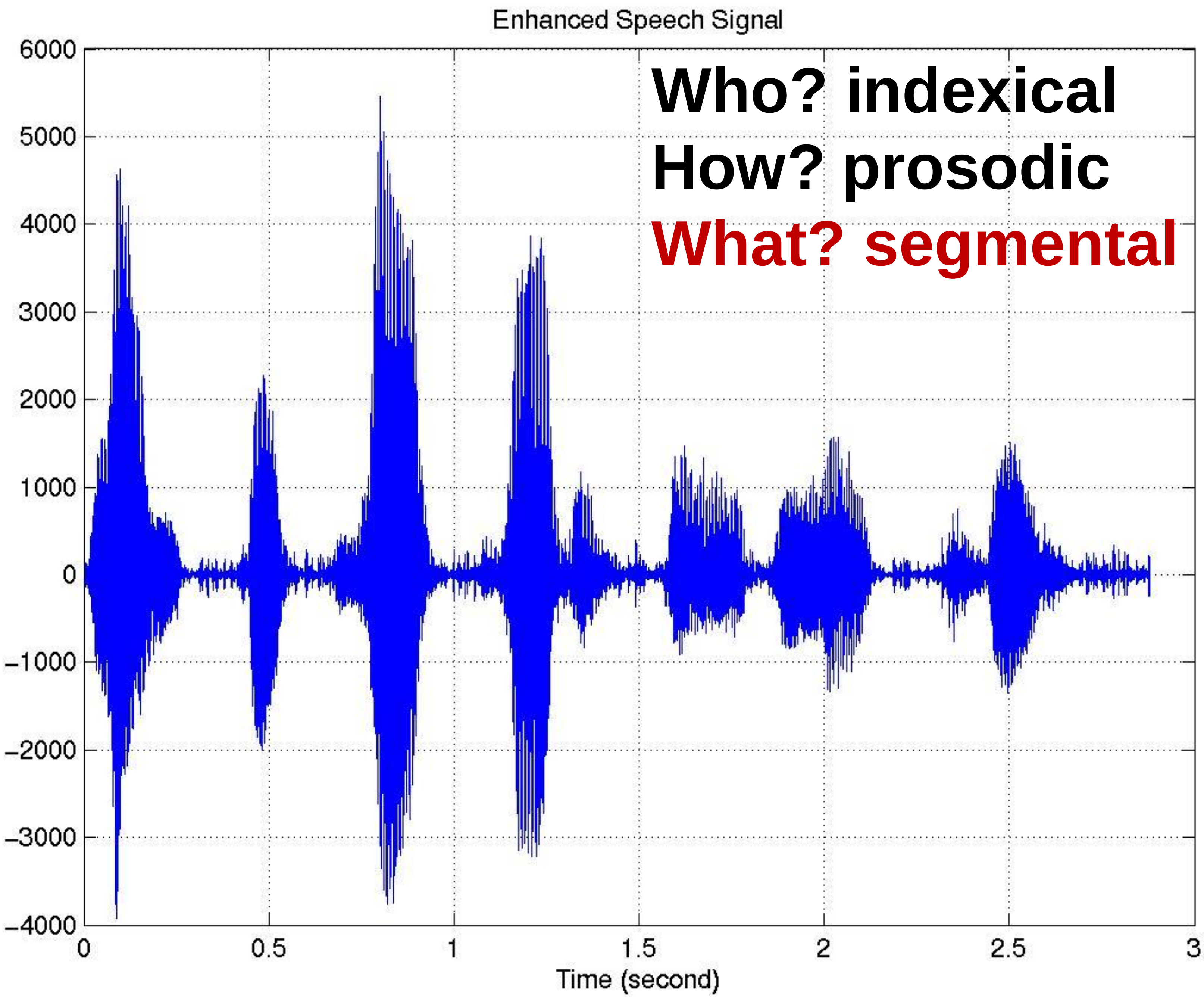

Enhanced Speech Signal
6000
5000
4000
3000
2000
1000
0
−1000
−2000
−3000
−4000
0
0.5
1
1.5
2
2.5
3
Time (second)
What? segmental



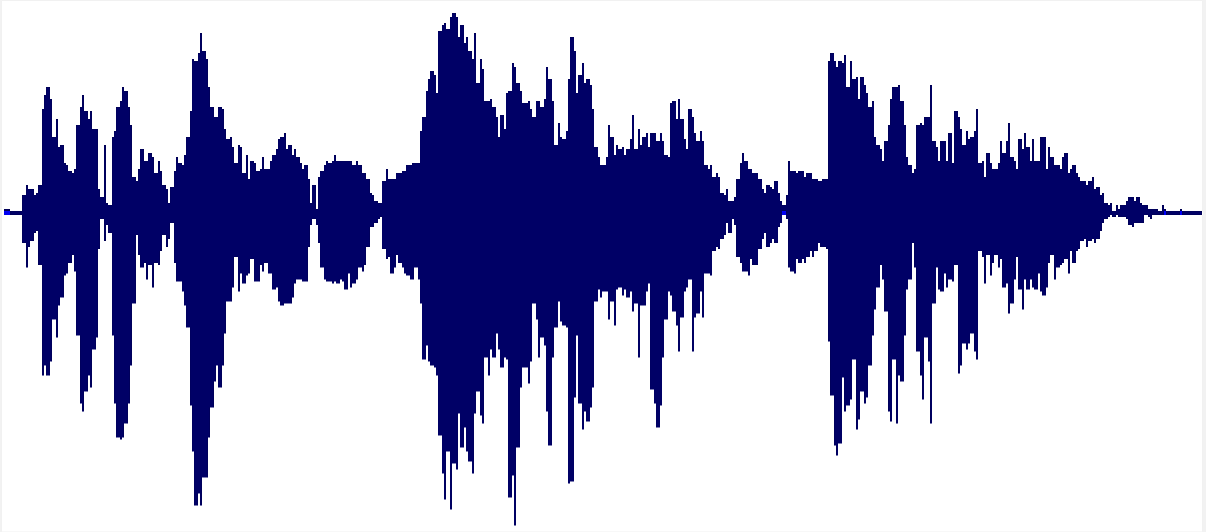
How? prosodic
What? segmental



Who? indexical
How? prosodic
What? segmental

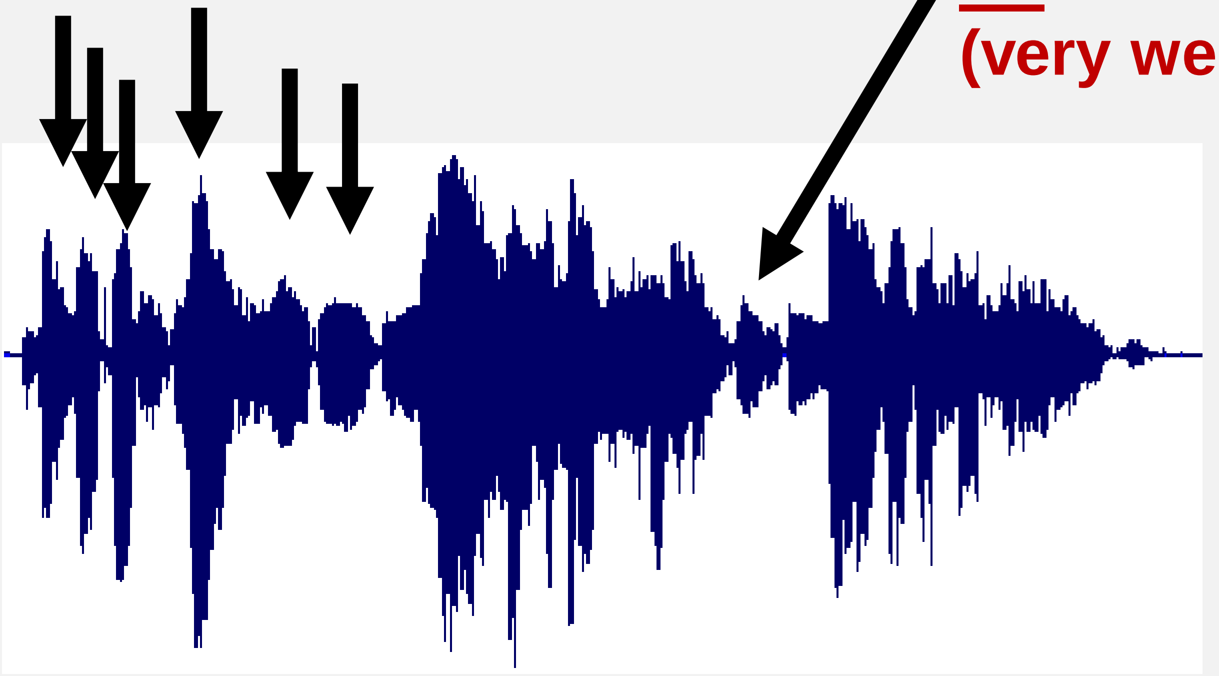

Enhanced Speech Signal
Who? indexical
How? prosodic
What? segmental
6000
5000
4000
3000
2000
1000
0
−1000
−2000
−3000
−4000
0
0.5
1
1.5
2
2.5
3
Time (second)

from signal to words



from signal to words

Syllable structure weakly visible – thanks to the energy contour
No word boundaries (very weak cues)



from signal to words

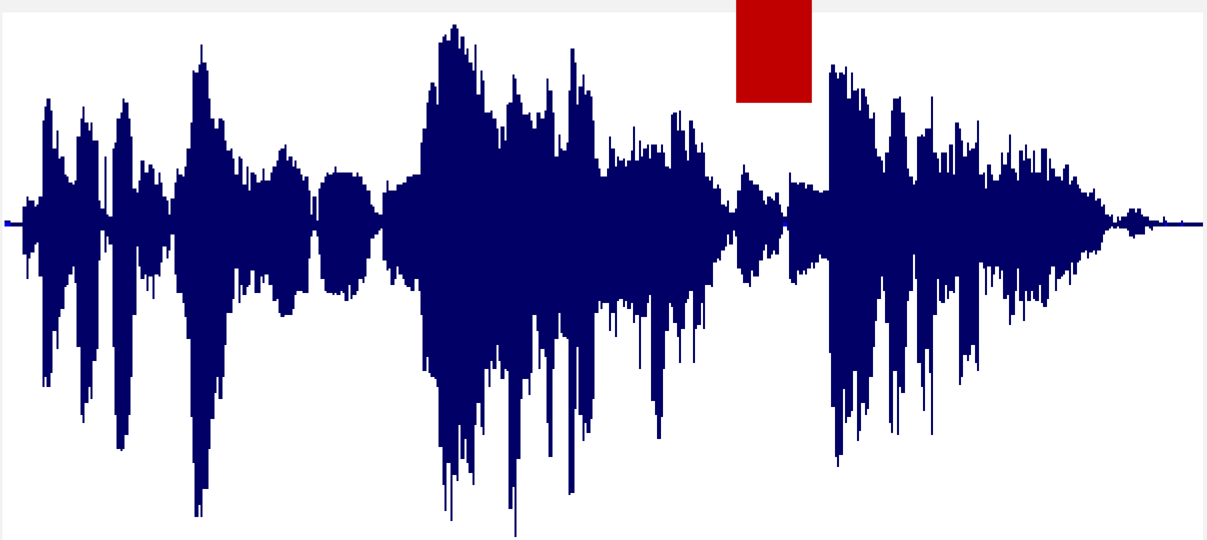
High level description

There is no X: say(“I”, X) & speculative(X) …

NLP/NLU

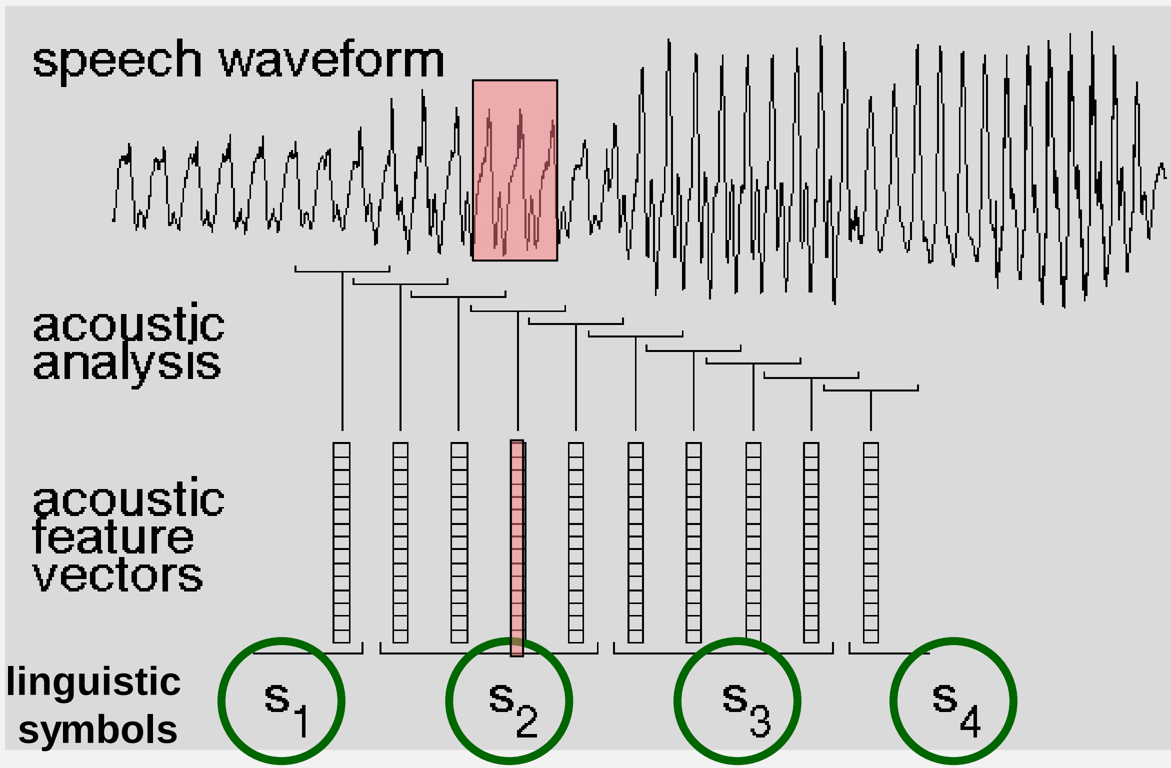
... I did not say anything speculative, but interestingly ...

ASR

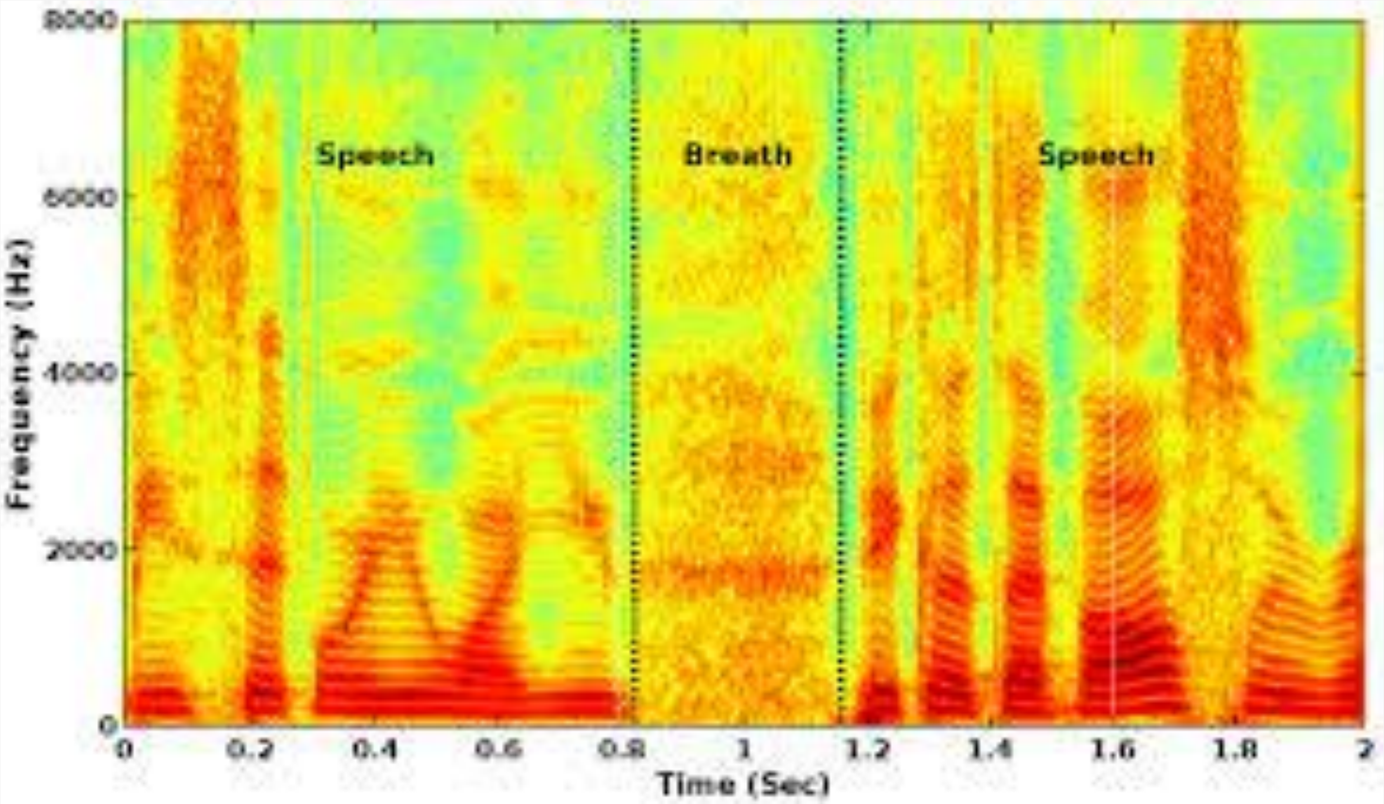


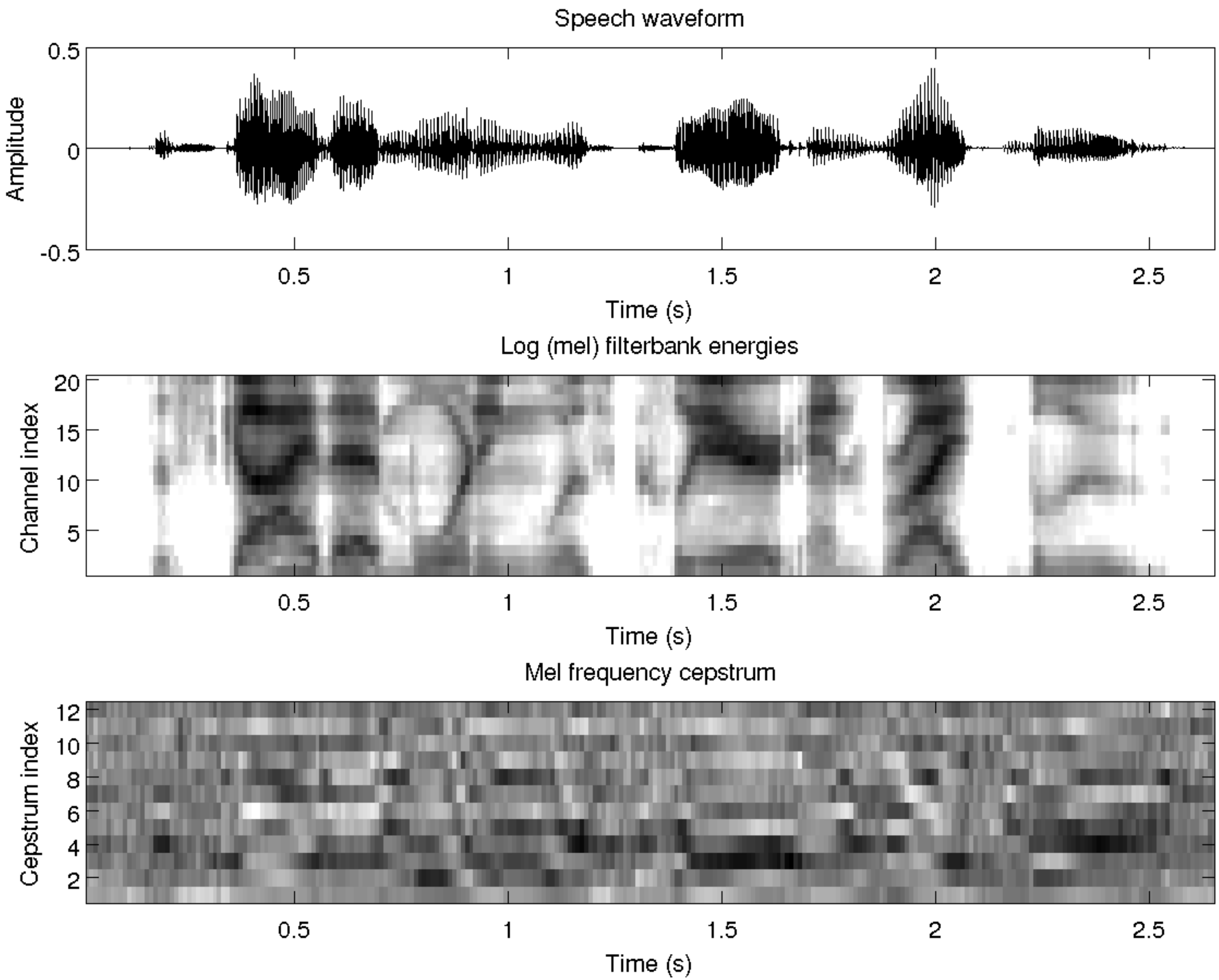
Low level description

ASR: features ("front end")

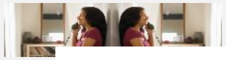


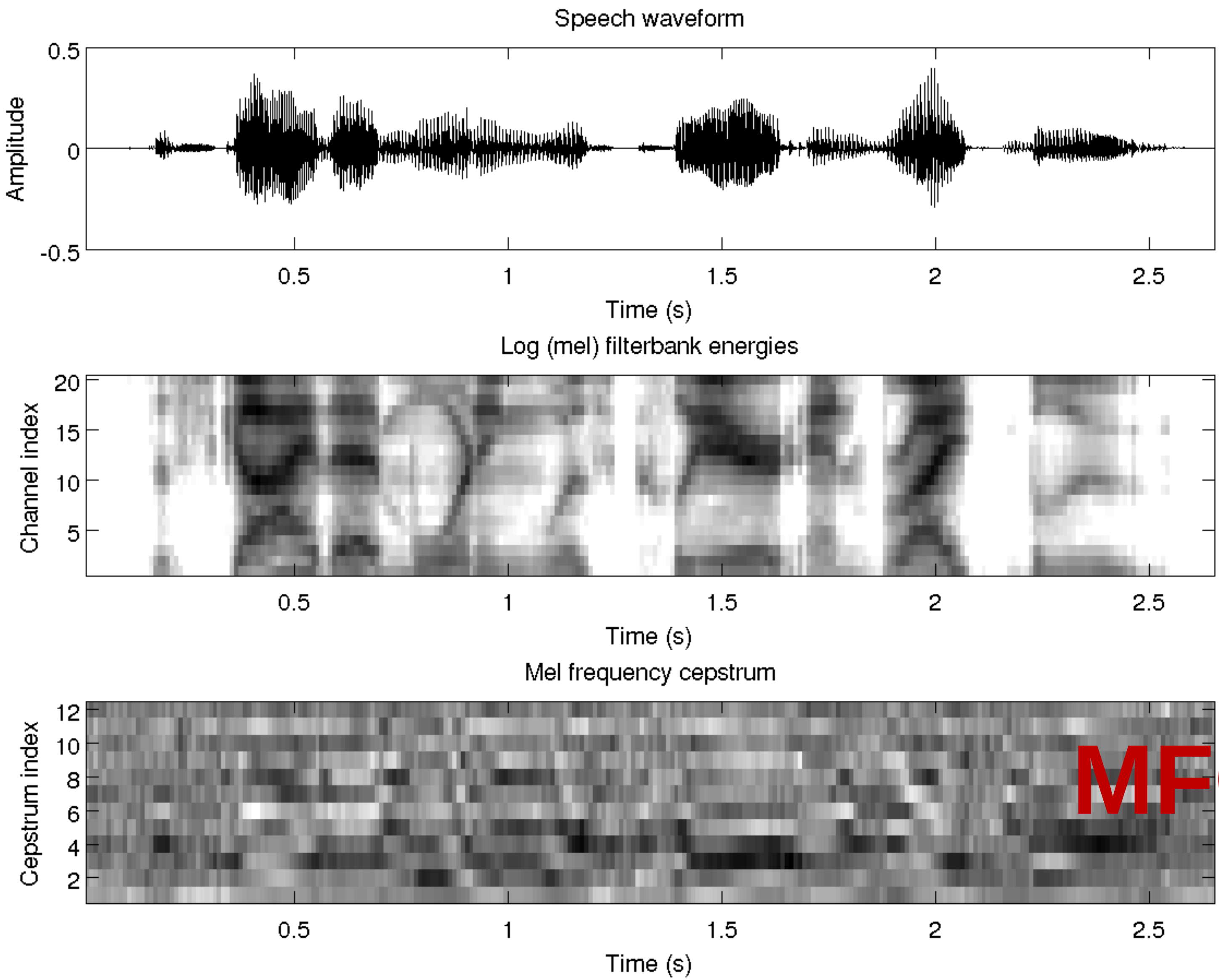
spectral analysis





Speech waveform
Amplitude
0.5
0
-0.5
0.5
1
1.5
2
2.5
Time (s)
Log (mel) filterbank energies
Channel index
20
15
10
5
0.5
1
1.5
2
2.5
Time (s)
Mel frequency cepstrum
Cepstrum index
12
10
8
6
4
2
0.5
1
1.5
2
2.5
Time (s)





Speech waveform
Amplitude
0.5
0
-0.5
0.5
1
1.5
2
2.5
Time (s)
Log (mel) filterbank energies
Channel index
20
15
10
5
Time (s)
Mel frequency cepstrum
Cepstrum index
12
10
8
6
4
2
Time (s)

MFCC

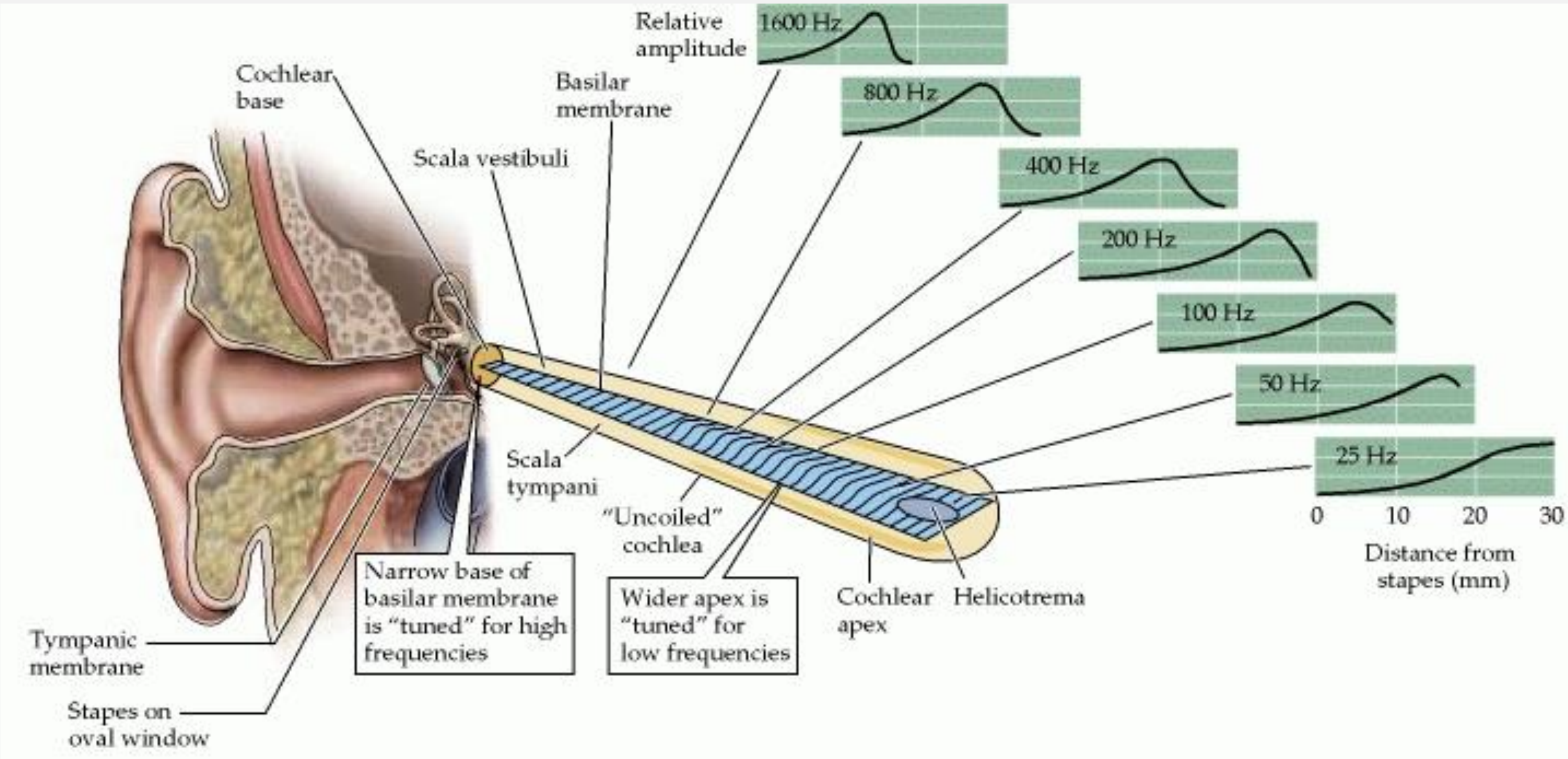
MFCC vectors as audio representation

Very many sites show information about MFCCs, often with useful Python function calls

See e.g.
https://www.kaggle.com/ilyamich/mfcc-implementation-and-tutorial

https://pypi.org/project/python_speech_features/

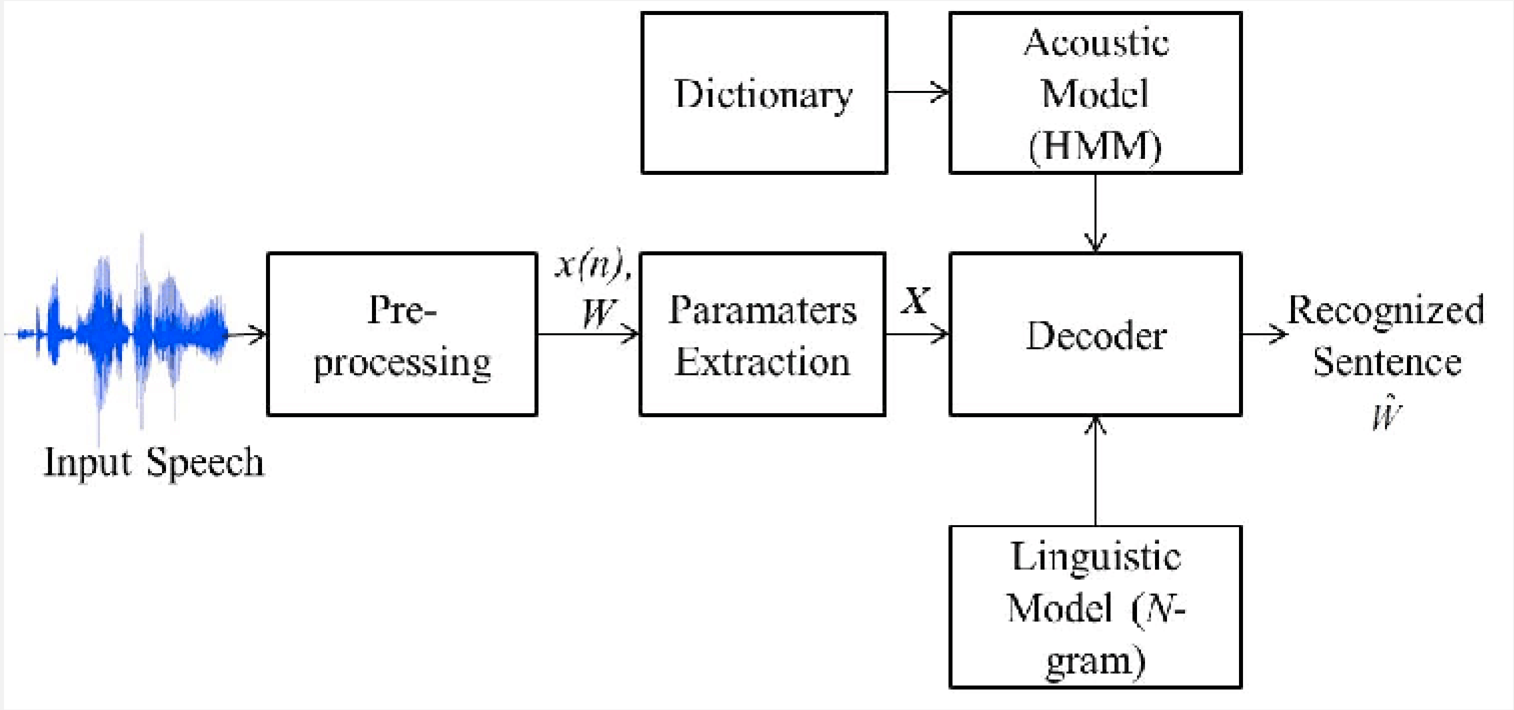
Librosa python library
https://librosa.org/doc/latest/index.html



Weber's law https://www.youtube.com/watch?v=hHG8io5qIU8

Physics	Perception
Energy as function of frequency	log(E) as function of log(f)

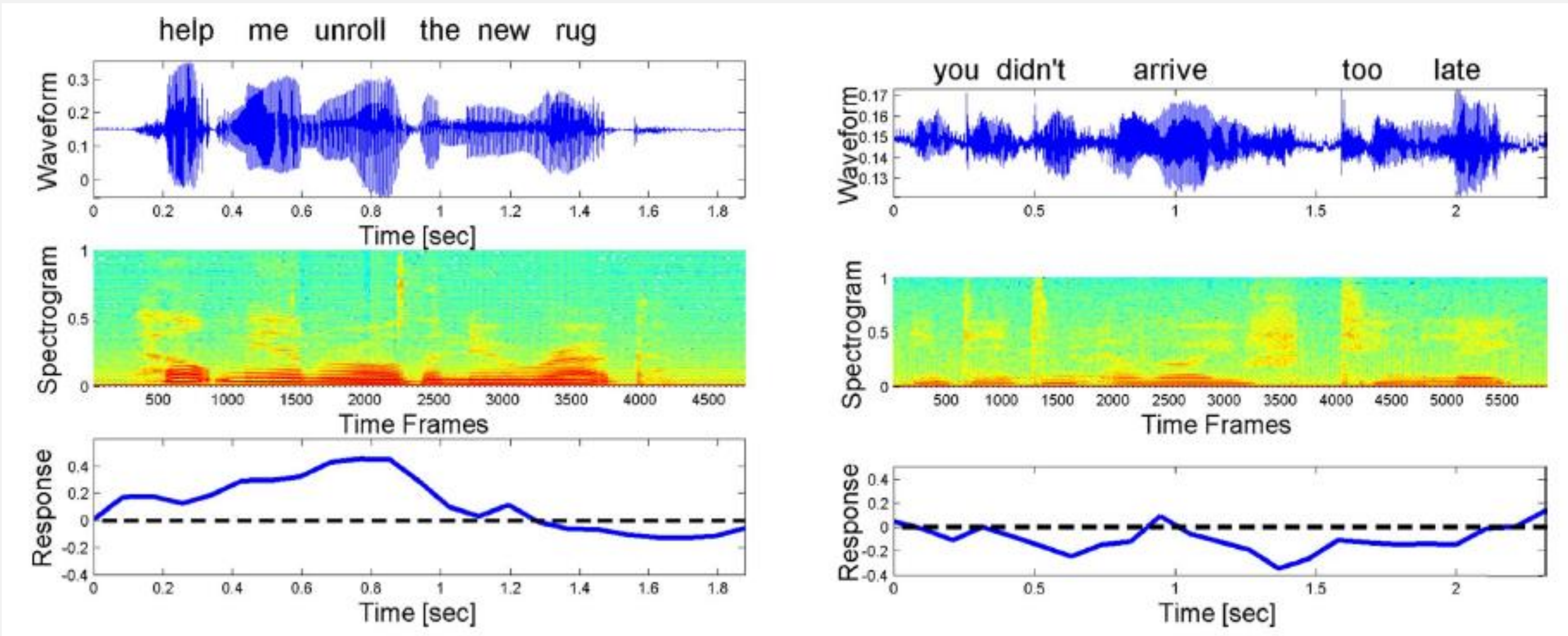
ASR: classical architecture

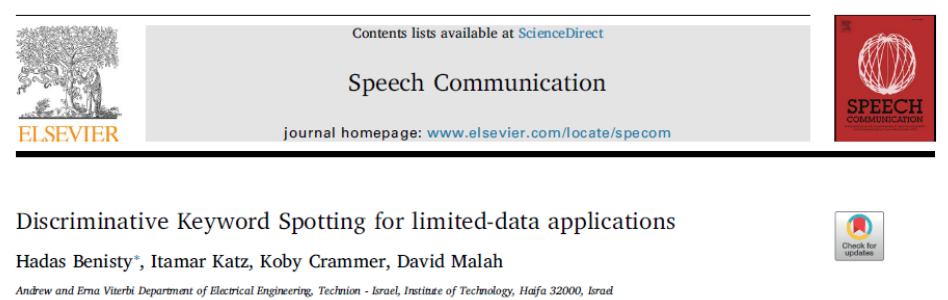


“Free” recognition

```
audio  0.18 0.15 ik 1.00
audio  0.33 0.09 heb 1.00
audio  0.42 0.09 wel 0.95
audio  0.51 0.09 eens 0.88
audio  0.6  0.12 met 1.00
audio  0.72 0.39 serie 0.97        Siri
audio  1.11 0.63 gesproken 1.00
audio  7.35 0.21 dit 0.87
audio  7.56 0.15 gaat 0.94
audio  7.71 0.12 wel 0.94
audio  7.83 0.30 enkel 0.56
audio  8.14 0.44 soepel 1.00
audio 24.99 0.09 dan 0.91
audio 25.08 0.12 zou 0.95
```

Keyword Spotting





Contents lists available at ScienceDirect

Speech Communication

journal homepage: www.elsevier.com/locate/specom

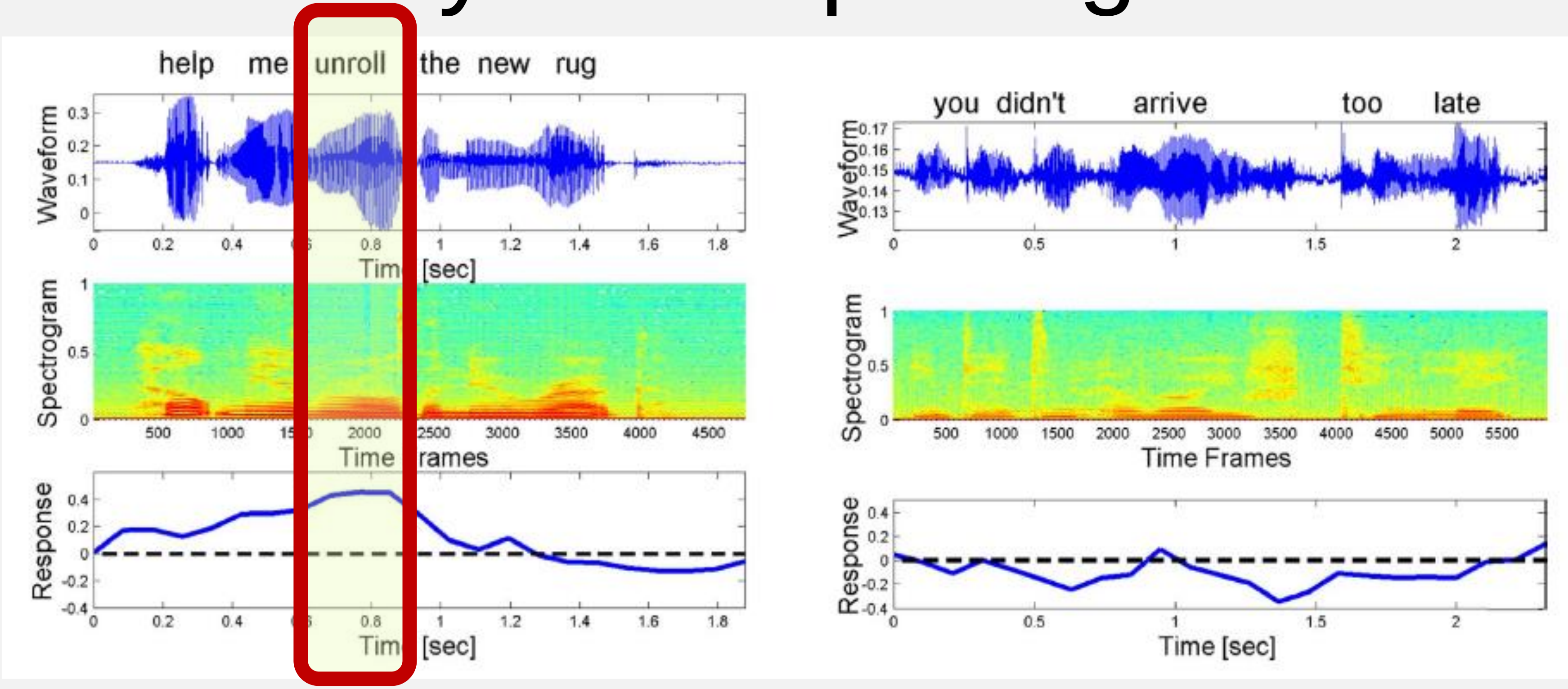
ELSEVIER

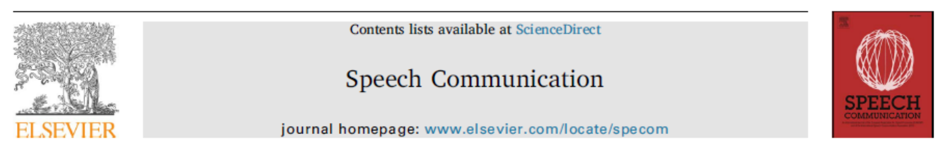
Discriminative Keyword Spotting for limited-data applications

Hadas Benisty*, Itamar Katz, Koby Crammer, David Malah

Andrew and Erna Viterbi Department of Electrical Engineering, Technion - Israel, Institute of Technology, Haifa 32000, Israel

Keyword Spotting





Contents lists available at ScienceDirect

Speech Communication

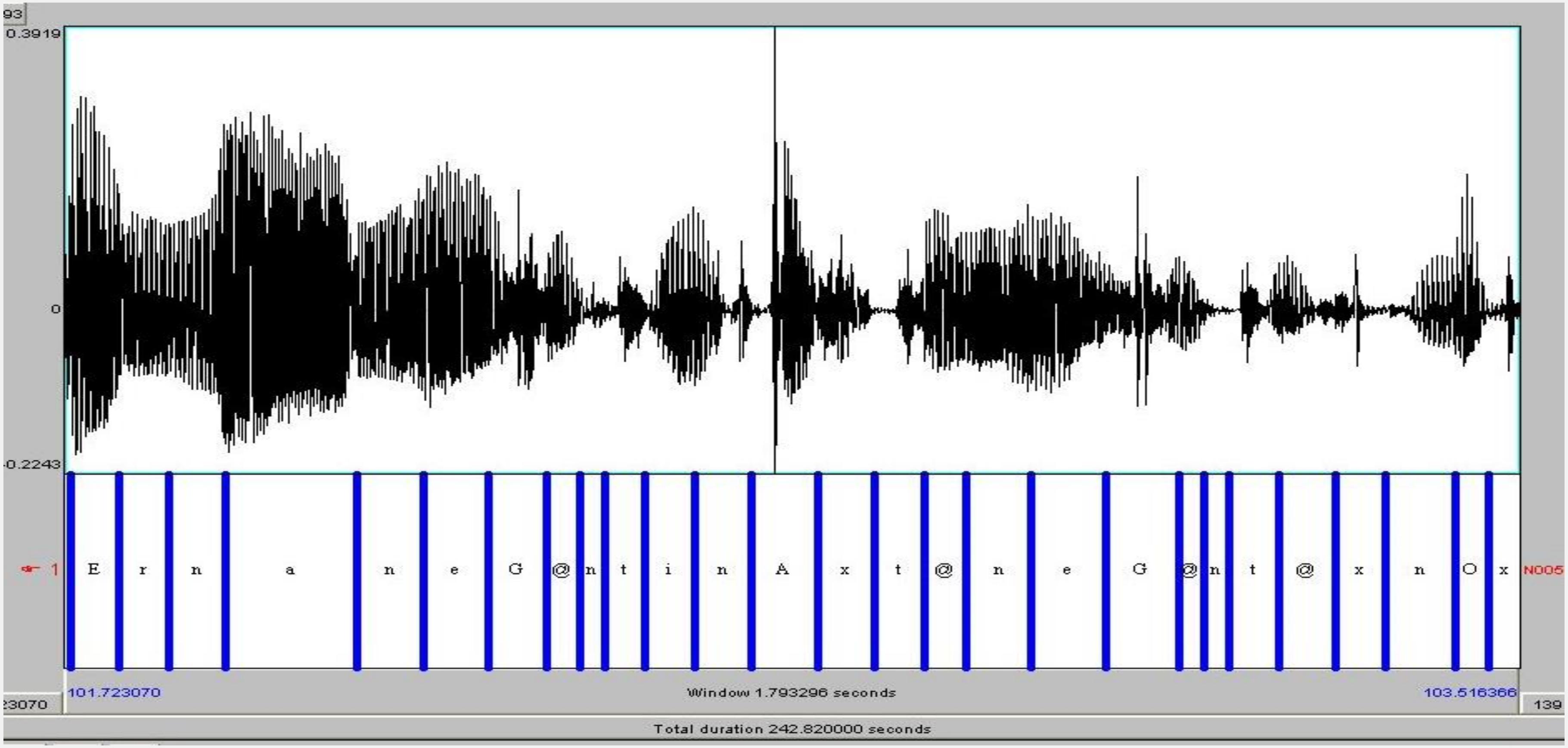
journal homepage: www.elsevier.com/locate/specom

Discriminative Keyword Spotting for limited-data applications

Hadas Benisty*, Itamar Katz, Koby Crammer, David Malah

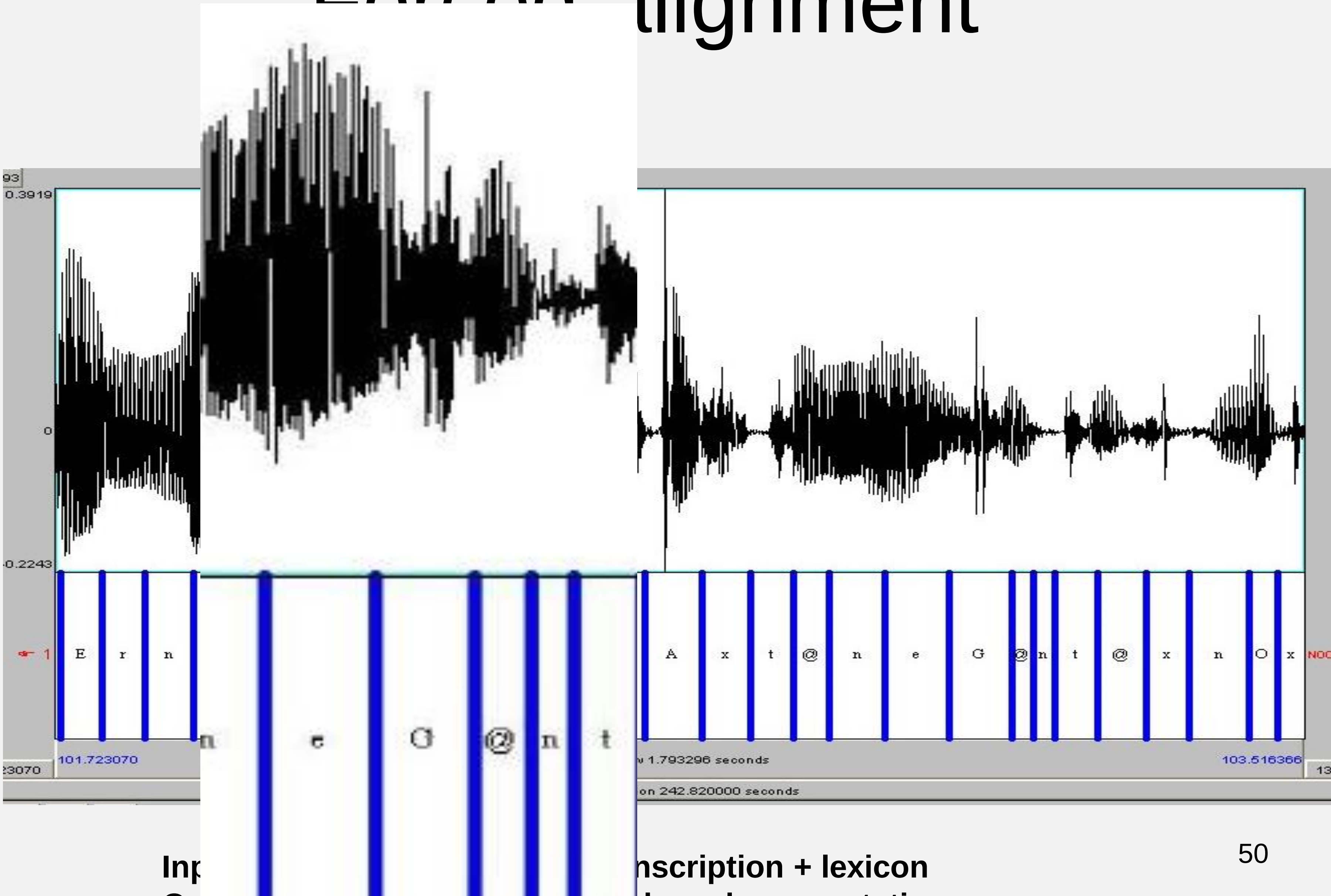
Andrew and Erna Viterbi Department of Electrical Engineering, Technion - Israel, Institute of Technology, Haifa 32000, Israel

Forced alignment



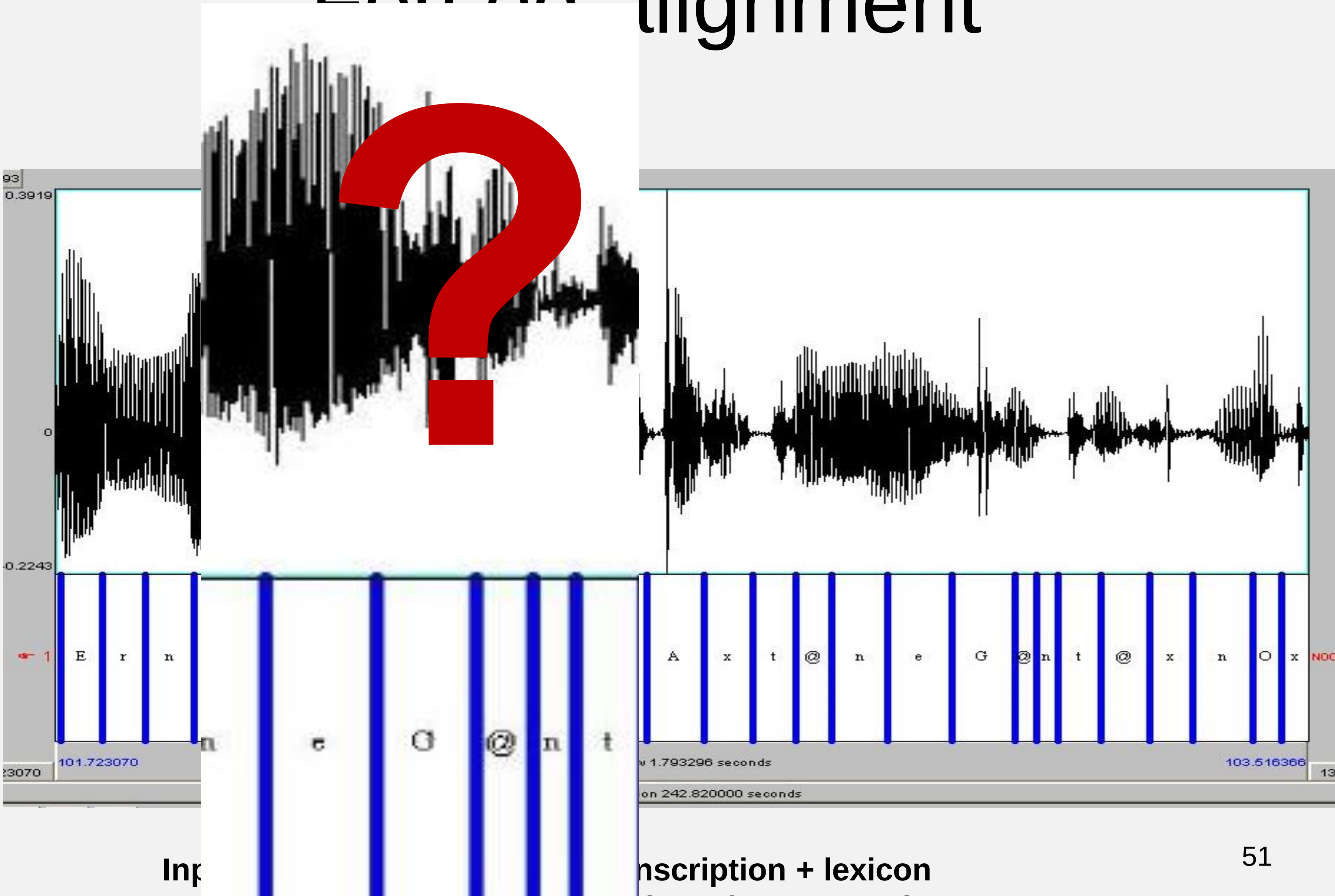
Input: wave file + word-level transcription + lexicon
Output: a phone-based or word-based segmentation

Forced alignment

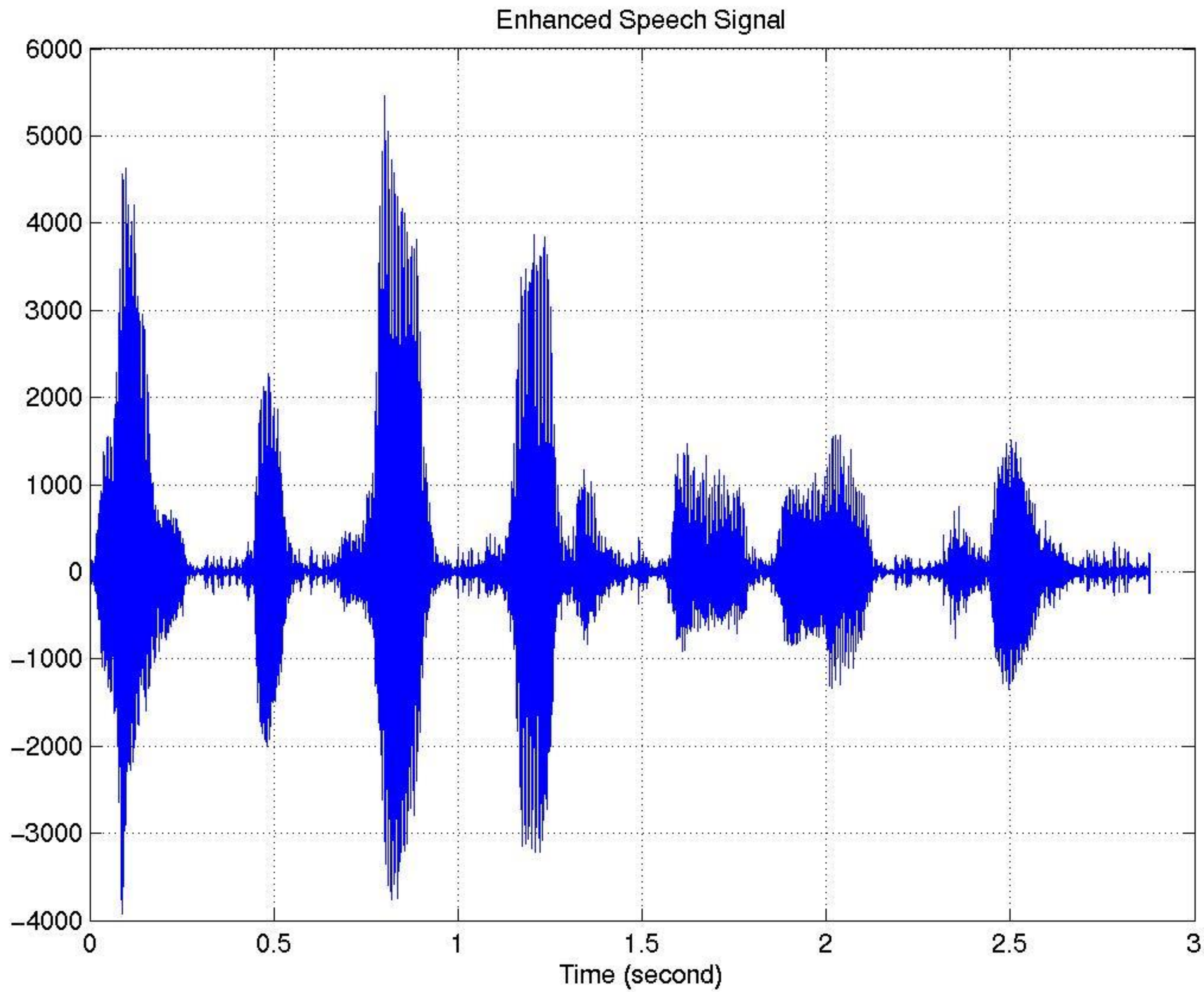


Inp[illegible]nscription + lexicon
Output: a phone-based or word-based segmentation

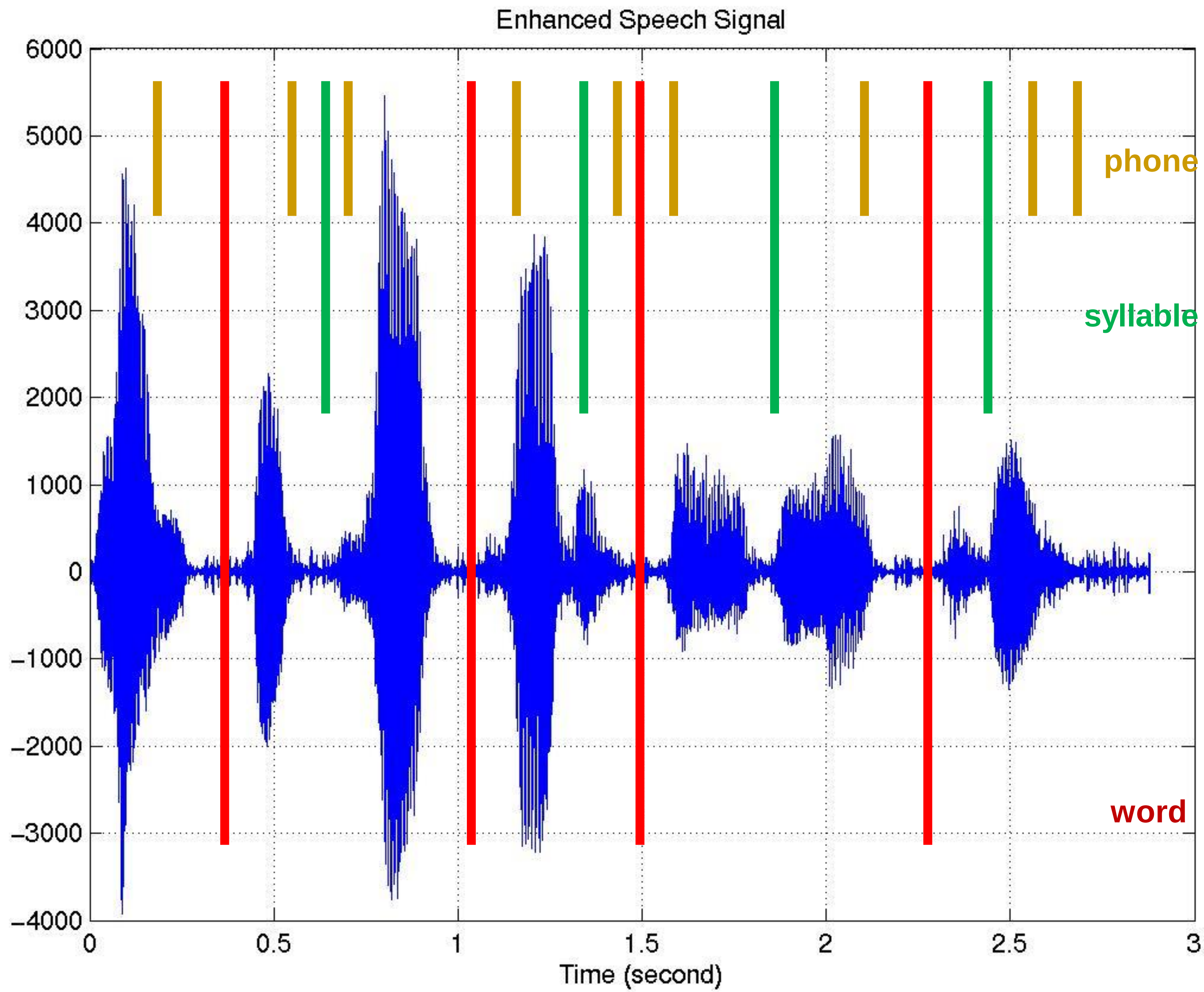
Forced alignment



Inp… nscription + lexicon
Output: a phone-based or word-based segmentation



Enhanced Speech Signal
6000
5000
4000
3000
2000
1000
0
−1000
−2000
−3000
−4000
0
0.5
1
1.5
2
2.5
3
Time (second)

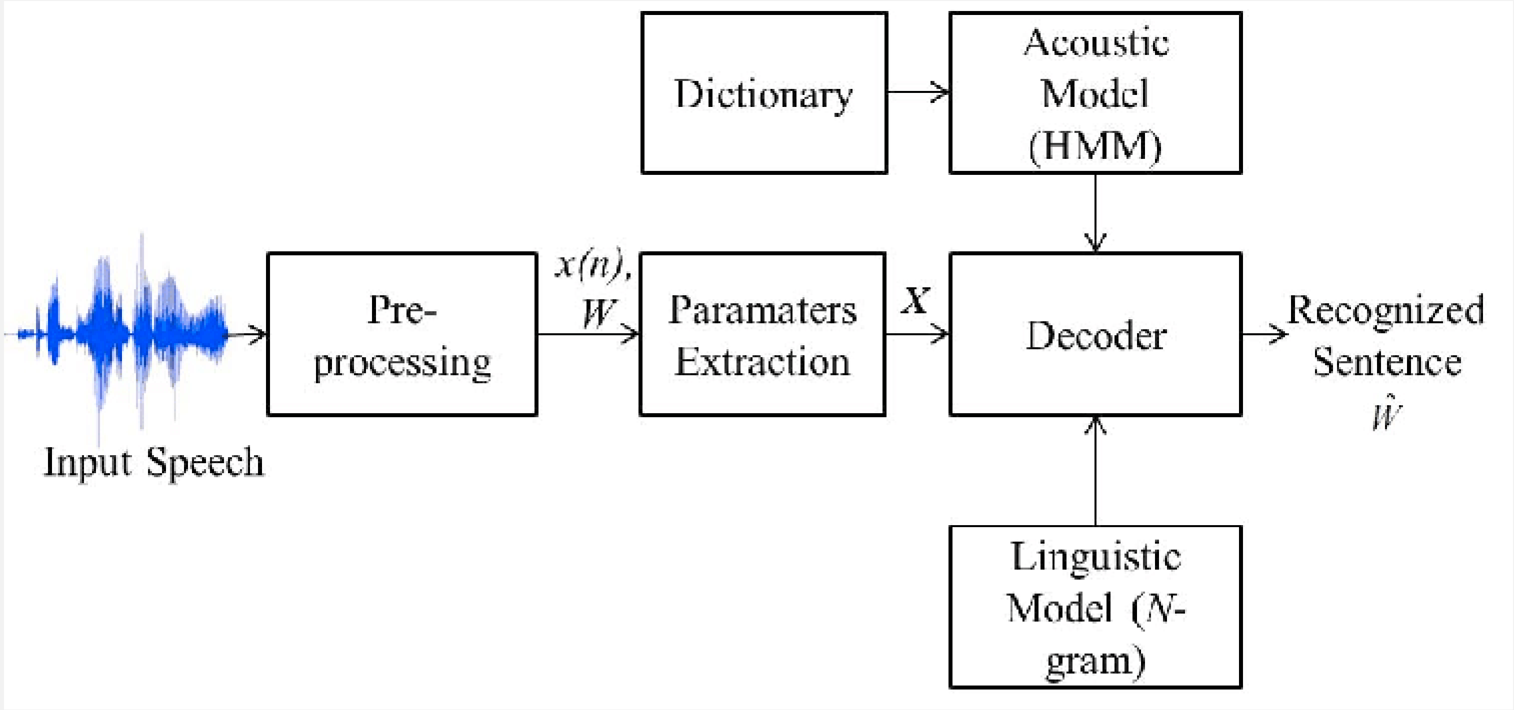


Enhanced Speech Signal
6000
5000
4000
3000
2000
1000
0
−1000
−2000
−3000
−4000
0
0.5
1
1.5
2
2.5
3
Time (second)
phone
syllable
word

Small history: ASR

- 1975: recognition using templates
- 1990: Gaussian Mixture Models (GMMs) + Hidden Markov Models (HMMs)
- in 2009 Brno: **proto-KALDI** system based on FSTs (FST = finite state transducer)
  - https://web.cs.ucdavis.edu/~rogaway/classes/120/spring13/eric-transducers.pdf
- since 2012-13: advent of Artificial Neural Networks (ANNs, DNNs, RNNs)

ASR: architecture since 1980

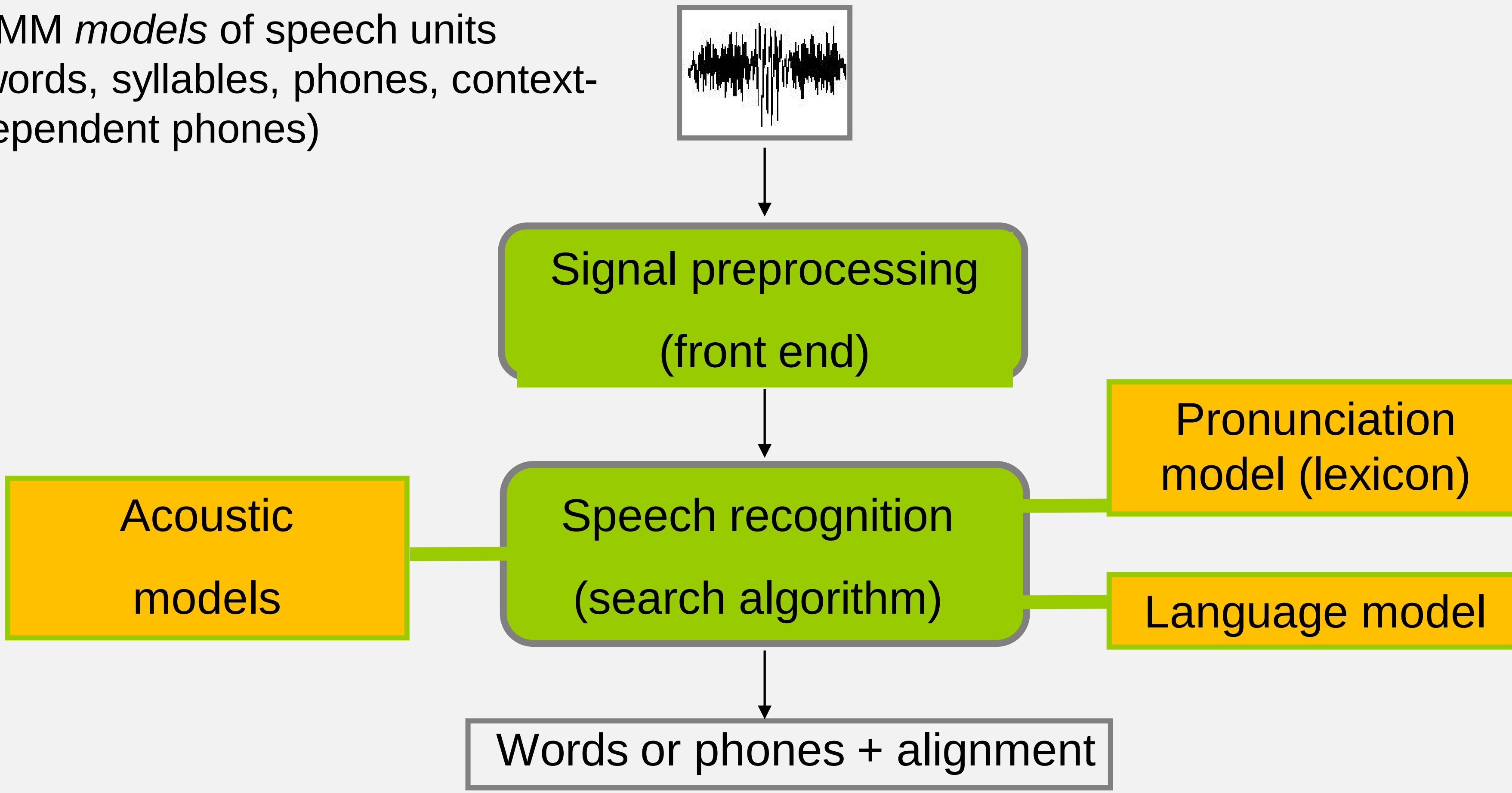


Since 2013: DNNs replace or modify one or more components

ASR

Basic principle

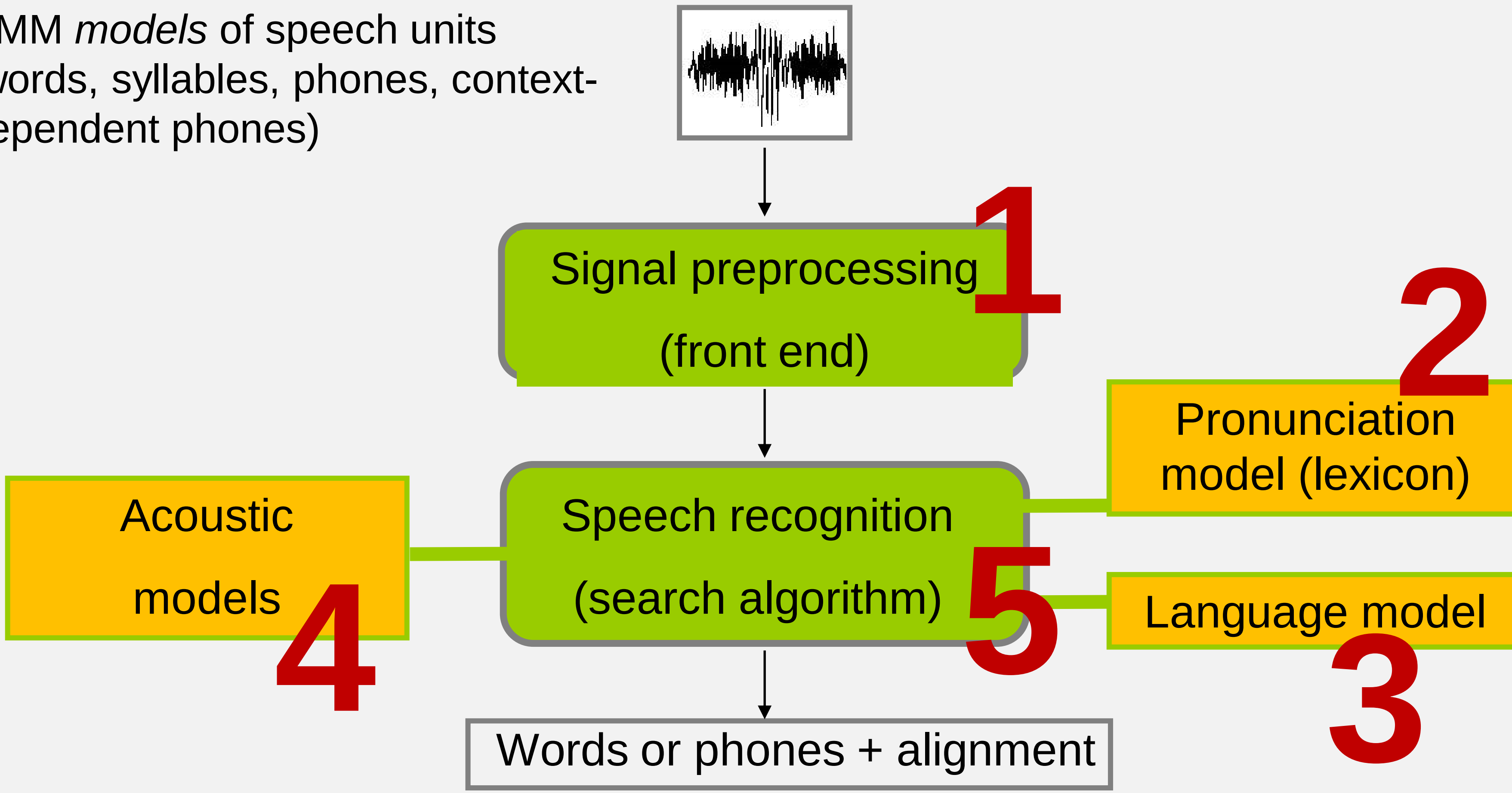
HMM *models* of speech units (words, syllables, phones, context-dependent phones)



ASR

Basic principle

HMM *models* of speech units (words, syllables, phones, context-dependent phones)



N-Best, etc.

ASR components

The front-end: extracting features from audio

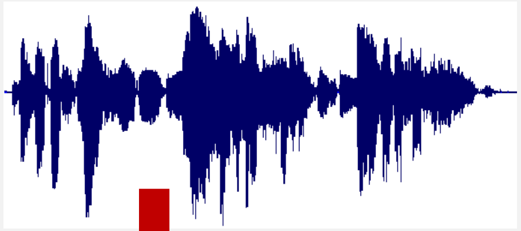
The lexicon specifies per word the corresponding sequence of speech units. There may be pronunciation variants (more variants per word) (word list, FST)

The language model specifies the probability of all possible word sequences. It is derived from large text corpora. The tokens in the LM are the words figuring in de lexicon (N-grams, NN-LM)

The acoustic models describe for each speech unit its internal organisarion (HMM topology) and the probability density function of the corresponding feature vectors (GMM, NN)

The search algorithm decides for the observed frames (under the constraints of acoustic model, lexicon and language model) which word sequence is the most likely one (often FST based)

Examples: HTK Hidden Markov Modelling ToolKit (Cambridge University, UK), KALDI



Front-end

Feature vectors v1, v2, v3, …

Acoustic model

Speech sounds E p @ l …

Lexicon

Words apple

Language model P(W| context)

Sentence This apple is …

Some terminology

• Sentence	linguistic concept
• Utterance	a stretch of spoken words
• Backchannel	uhm, ehm, ja
• Word	everything between spaces?
• Segment	linguistic: “segmental tier”
• Phoneme	smallest unit carrying semantic diff
• Phone	spoken version of a phoneme
• Allophone	context dependent phone
• Coarticulation	influence of neighboring phones

Automatic Speech Recognition

- (Usually) based on statistical models
- Given “training data” from the target language, we train a statistical models of speech
  - AM: audio, LM: text
- Given a waveform, we can work out the most likely word sequence via Bayes

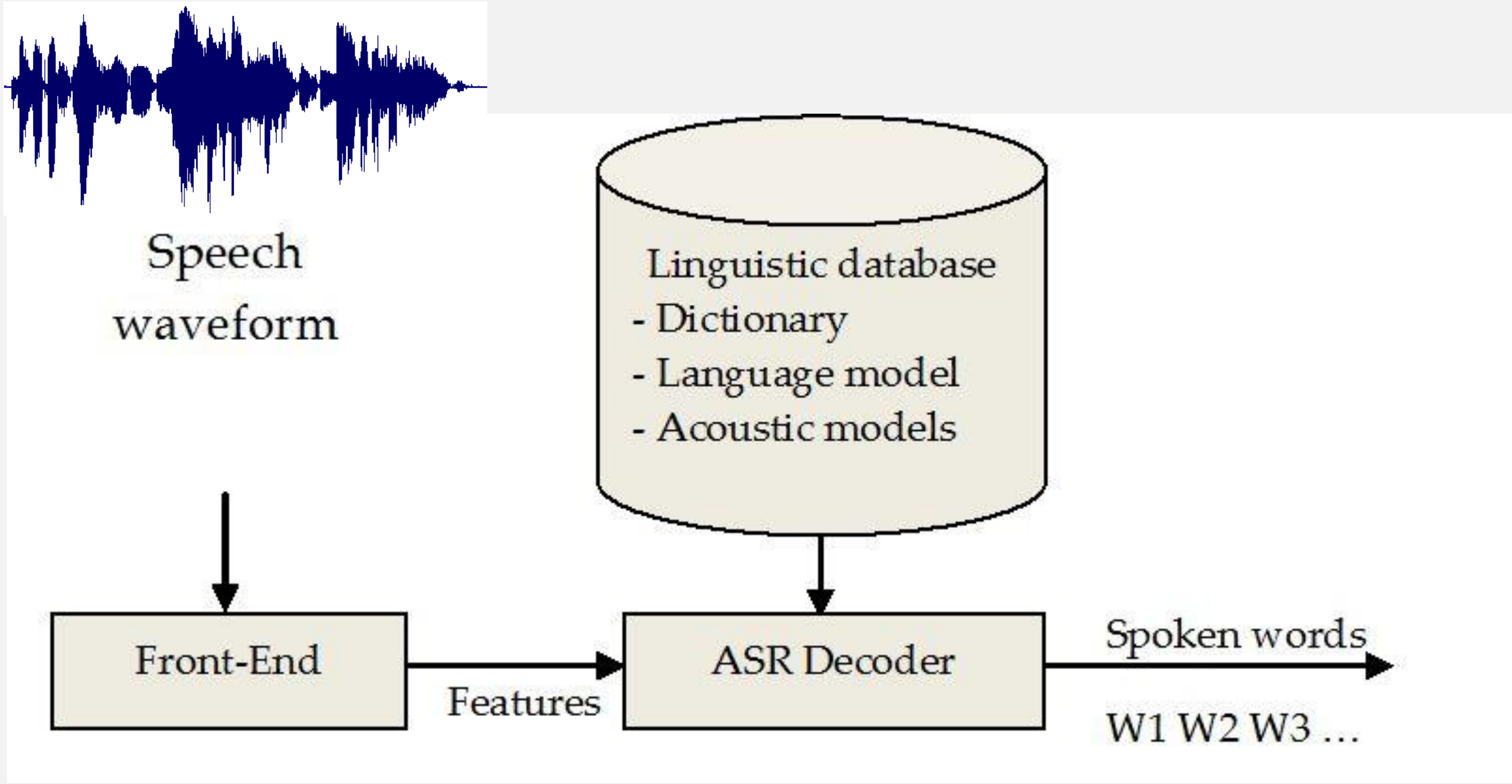
P(words | audio) ~ P(audio | words) P(words)

Automatic Speech Recognition

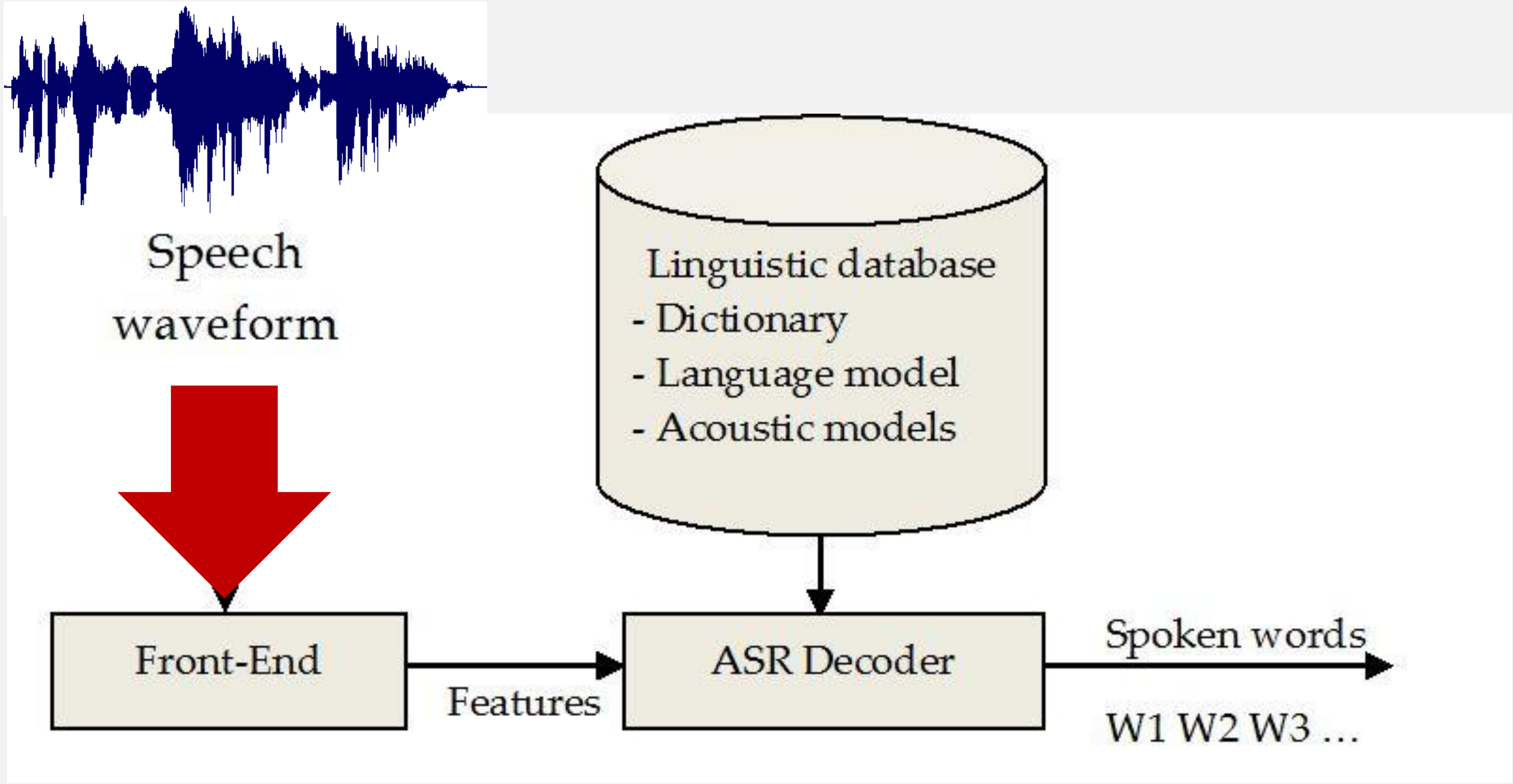
- (Usually) based on statistical models
- Given "training data" from the target language, we train a statistical models of speech
  - AM: audio, LM: text
- Given a waveform, we can work out the most likely word sequence via Bayes

$$P(\text{words} \mid \text{audio}) \sim \underbrace{P(\text{audio} \mid \text{words})}_{\textbf{AM}} \; \underbrace{P(\text{words})}_{\textbf{LM}}$$

ASR architecture

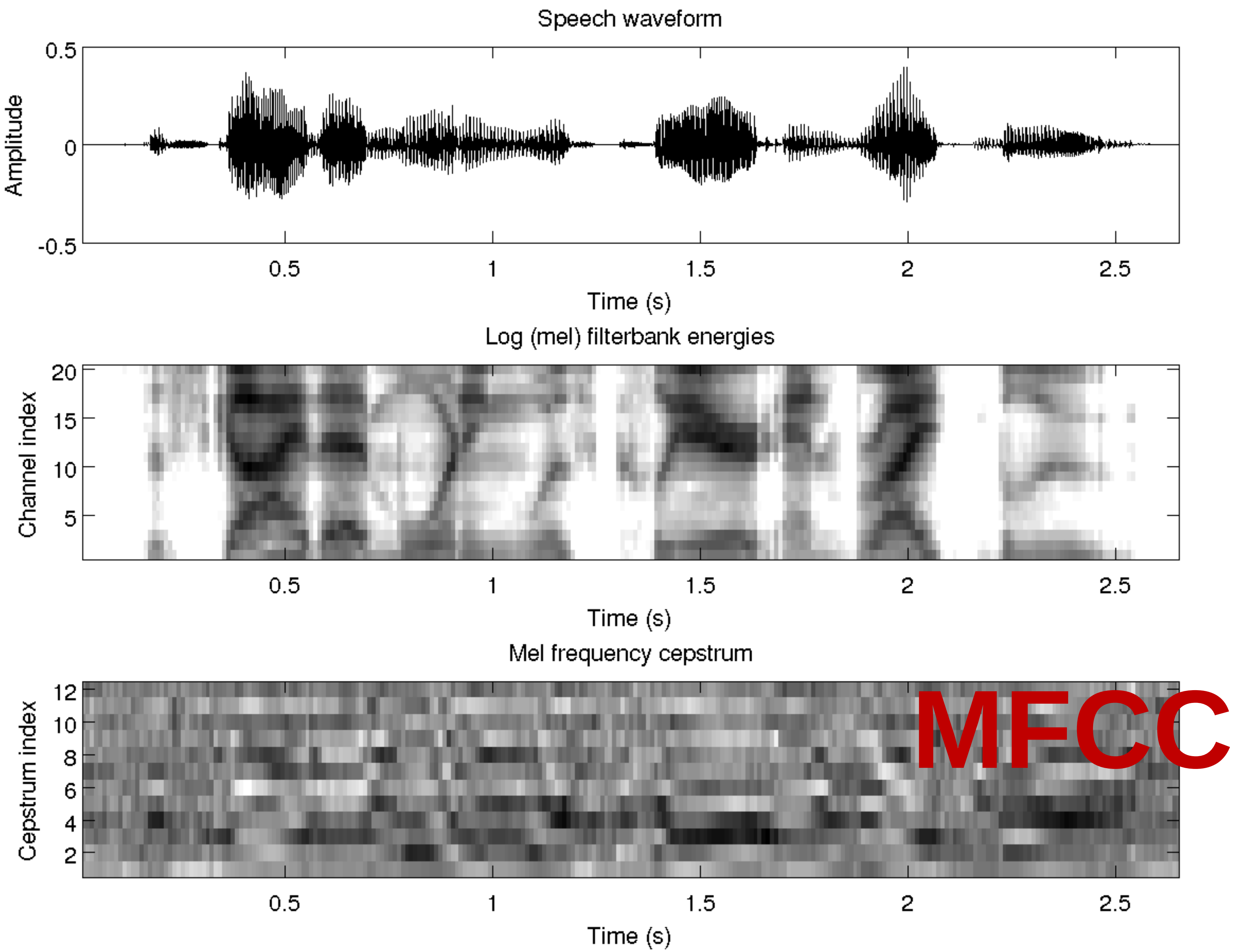


ASR architecture



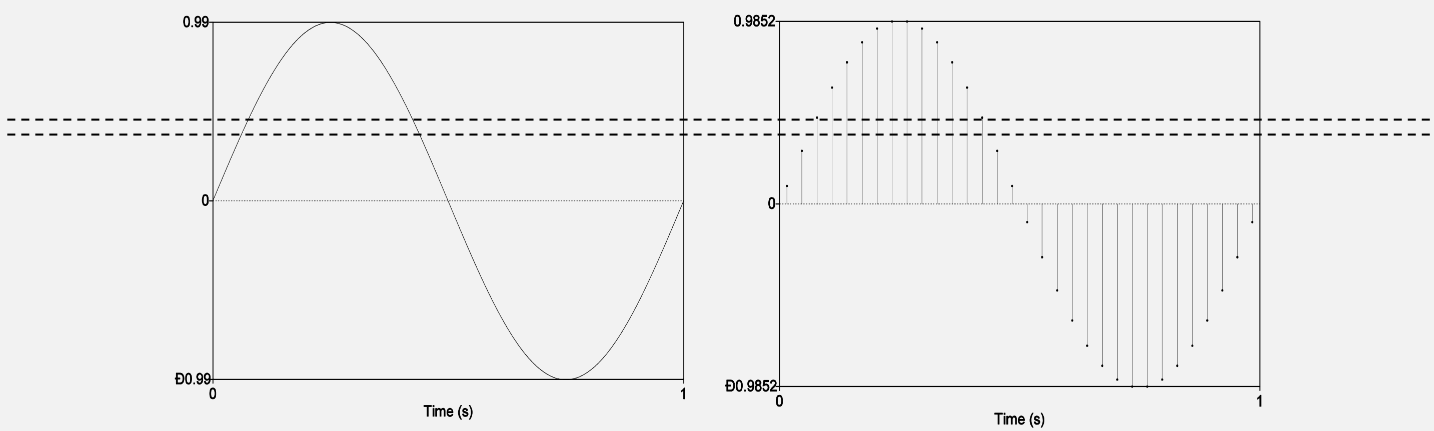
From audio to features

- Analog signal to digital signal
- Digital signal to feature vectors
  - 5 steps
- E.g. to MFCC
  - Standard but not unique
  - See “python mfcc”


Speech waveform
Amplitude
0.5
0
-0.5
0.5
1
1.5
2
2.5
Time (s)
Log (mel) filterbank energies
Channel index
20
15
10
5
Time (s)
Mel frequency cepstrum
Cepstrum index
12
10
8
6
4
2
Time (s)
MFCC

Analogue-to-digital (AD) conversion

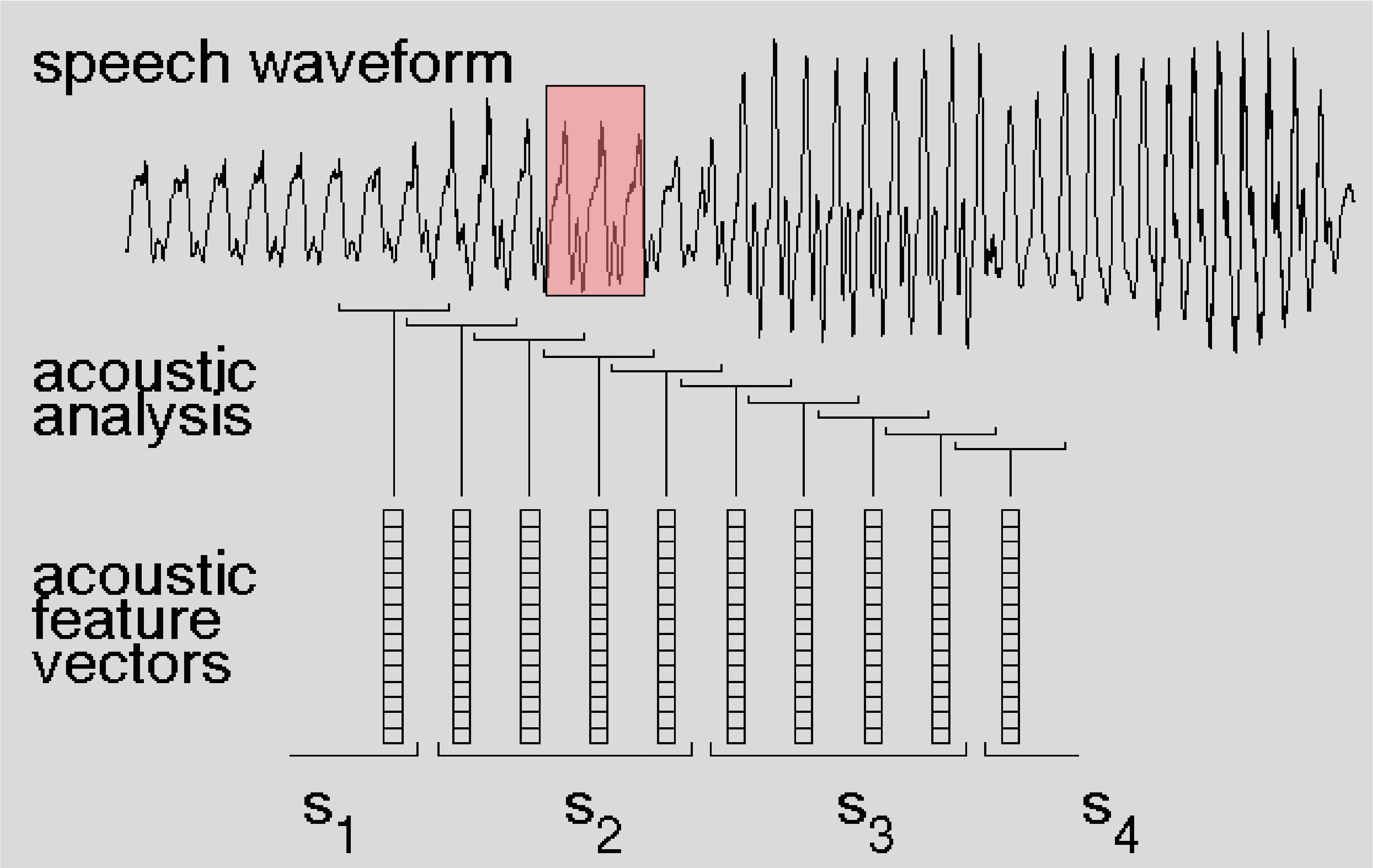
- Discretisation in time
  - sampling frequency or sampling rate (samples/sec, Hz) determines the highest frequency that can be represented. **Nyquist**.
- Discretisation in amplitude
  - Number of possible amplitude values is determined by available storage space per sample, e.g.
    - 8 bits (1 byte): 2^8 (256) possible values
    - 16 bits (2 bytes): 2^{16} (65536) possible values
  - Amplitude values are rounded to nearest discrete value



Signal to features

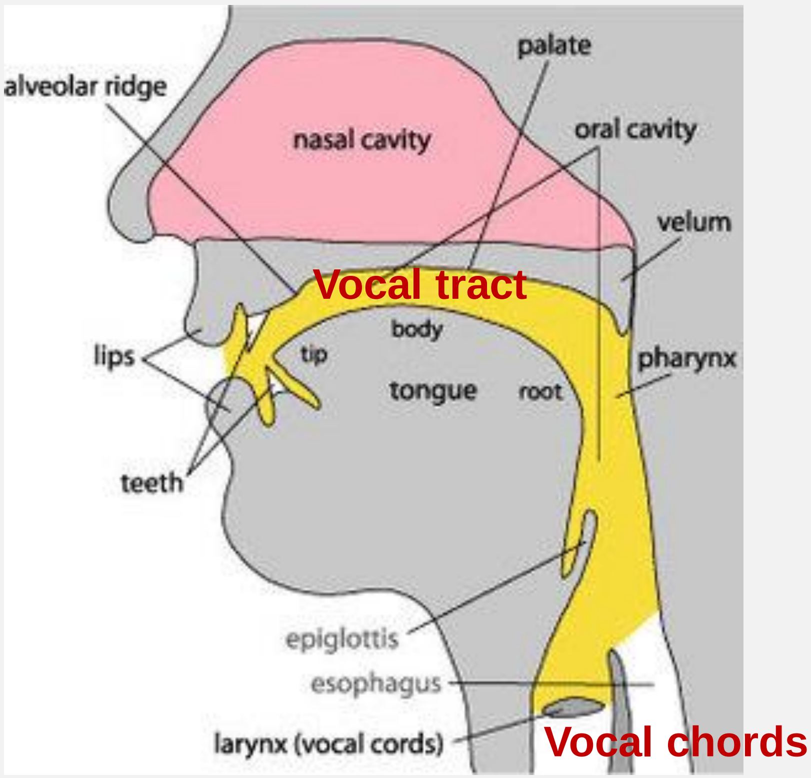
5 steps to convert a digital speech signal into *acoustic feature vectors*:

1. Segmentation: Division of the speech signal (oscillogram) into segments with a sufficiently short duration to assume that the vocal tract does not change shape (*frame-based analysis)*
2. Apply *windowing*
3. Convert the windowed signal from the time domain to the frequency domain using the *Fast Fourier Transform (FFT)*
4. Apply perceptual weighting on the spectral energy distribution *(e.g. Mel* or *Bark scale)* so that the energy in a certain frequency band becomes approximately equally important for the computer as for a human listener.
5. Decorrelate the output of the filters so that the histograms of the resulting features can be modeled adequately by mutually independent Gaussian probability density functions.



speech waveform
acoustic analysis
acoustic feature vectors
s_1
s_2
s_3
s_4

Segmentation: vocal tract



Articulation is relatively slow

About 12-14 speech sounds per second, i.e. 70 ms. per phone, on average

Articulations move synchronously/in parallel → assimilation of properties of neighboring sounds

Segmentation: analysis window

A long analysis window (e.g. 50 ms)

→ very precise description of how energy is distributed over frequency (good spectral resolution)

→ but: poor temporal resolution

→ → good but only useful if speech sounds last long enough: e.g. vowels, fricatives

A short analysis window (e.g. 10 ms) allows a detailed tracking of fast changes → good time resolution (useful for an accurate description of plosives) but poor spectral resolution

In practice, window length is often 25 ms (2.5 glottal periods for males; about 5 glottal periods for females)

Segmentation: duration

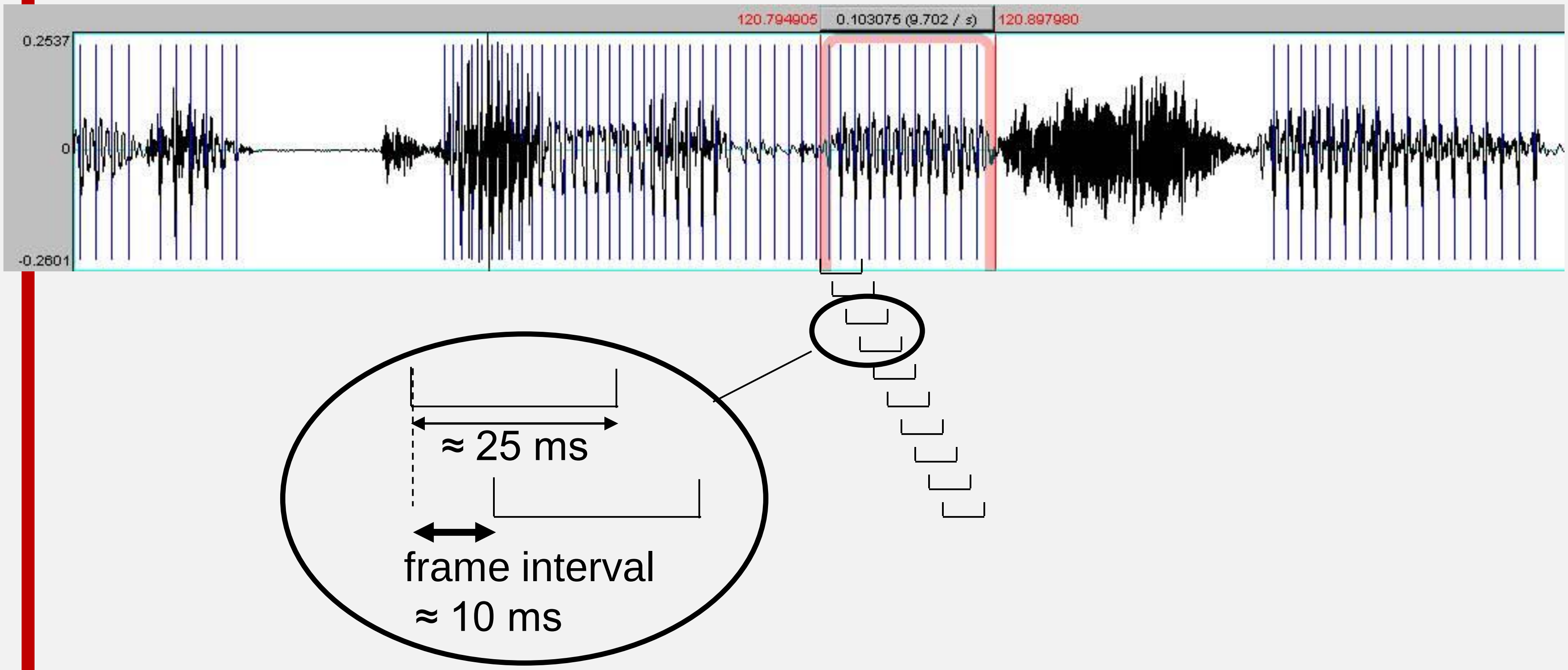
Segment the speech signal in small intervals (**frames**)

What is being said is mainly determined by the shape of the vocal tract. The shape of the vocal tract determines the energy envelope spectrum.
https://www.youtube.com/watch?v=cVovOKLlSb0

What is a sensible analysis duration for a frame? The time interval in which the vocal tract does not change substantially is determined by the number of speech sounds per second that is produced (≈13-15). (70ms) A defendable analysis frame duration is 25 ms.

How many frames per second? To accurately describe the changes in vocal tract shape over time, the number of frames per second must be at least twice as high as the highest frequency with which the vocal tract changes (Nyquist criterion) → 100 times/second (i.e. every 10 ms).

Feature extraction



25ms analysis window length
10ms frame shift
If sample freq = 16kHz, 25ms ~ 0.025*16000 = 400 samples.

Feature extraction

5 steps to convert a digital speech signal into *acoustic feature vectors*:

1. Divide the speech signal (oscillogram) in segments with a sufficiently short duration to assume that the vocal tract does not change shape (*frame-based analysis)*
2. Apply *windowing*
3. Convert the windowed signal from the time domain to the frequency domain using the *Fast Fourier Transform (FFT)*
4. Apply perceptual weighting on the spectral energy distribution *(e.g*. *Mel* or *Bark scale)* so that the energy in a certain frequency band becomes approximately equally important for the computer as for a human listener.
5. Decorrelate the output of the filters so that the histograms of the resulting features can be modeled adequately by mutually independent Gaussian probability density functions.

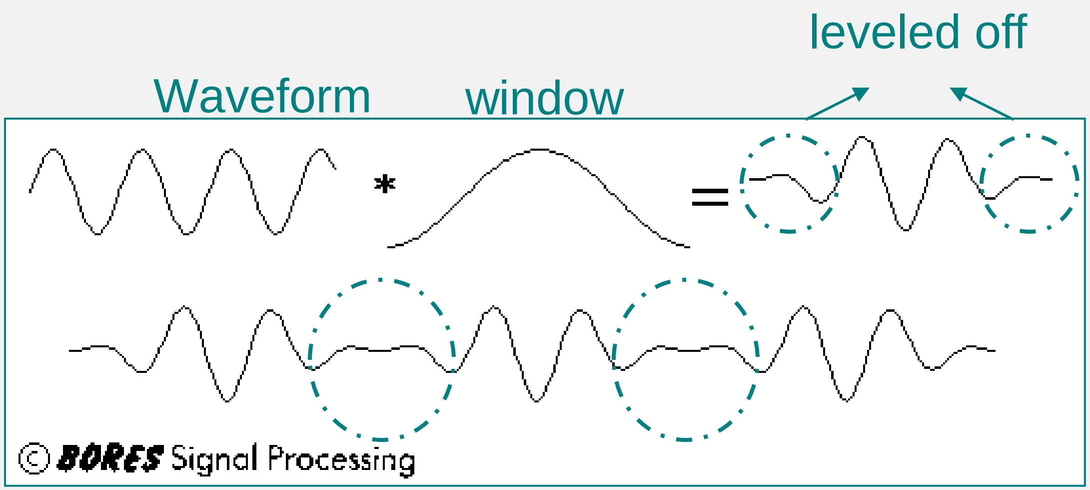
Feature extraction: windowing

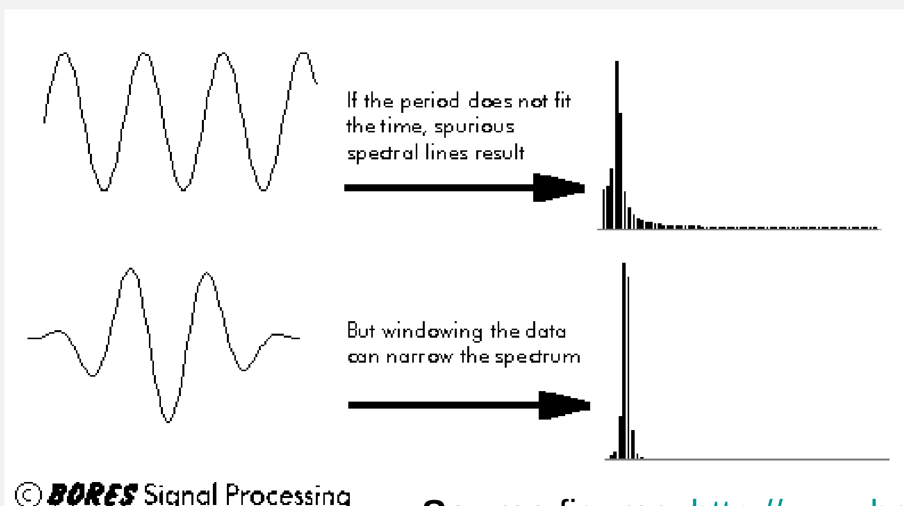
Taking a “rectangular” unweighted portion out of a wave file leads to audible artefacts. These artefacts can be avoided by proper windowing: taper off the beginning and end of the signal.

For a well-chosen window, the spectrum is nearly identical to a signal of which the core part is repeated indefinitely.

Often used windows are Hamming and Hanning windows. See e.g. https://www.youtube.com/watch?v=YsqGQzJ_2V0

Windowing





Source figures: http://www.bores.com/courses/intro/freq/3_window.htm

Feature extraction

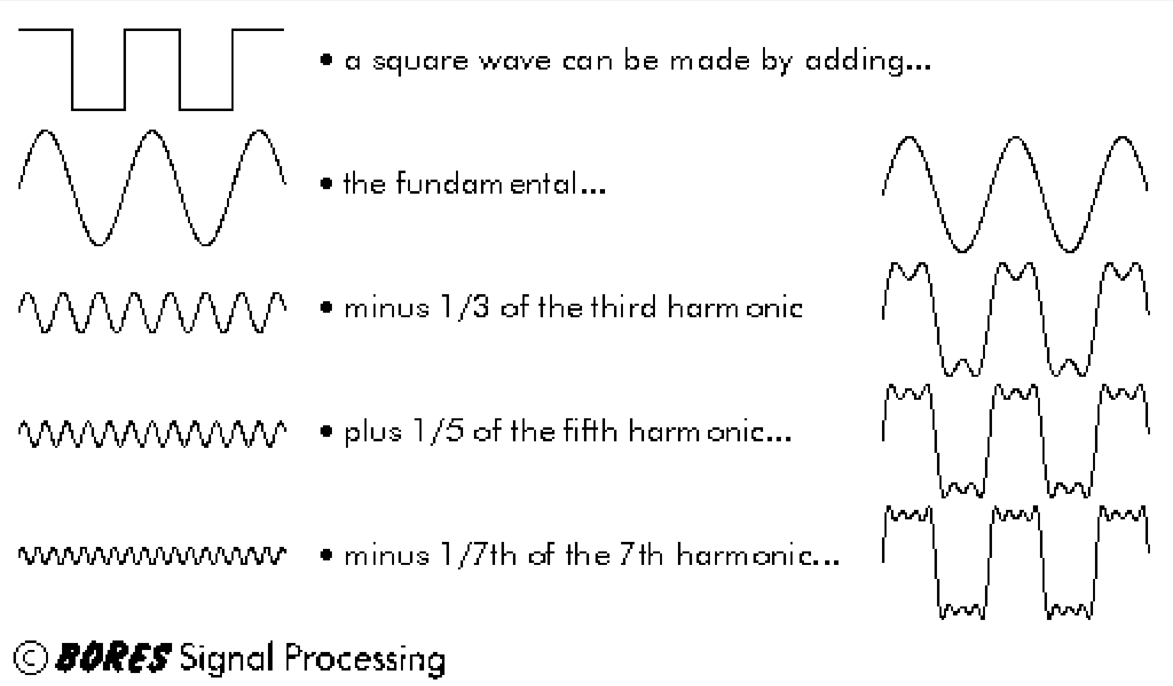
5 steps to convert a digital speech signal into *acoustic feature vectors*:

1. Divide the speech signal (oscillogram) in segments with a sufficiently short duration to assume that the vocal tract does not change shape (*frame-based analysis)*
2. Apply *windowing*
3. Convert the windowed signal from the time domain to the frequency domain using the *Fast Fourier Transform (FFT)*
4. Apply perceptual weighting on the spectral energy distribution *(e.g.* *Mel* or *Bark scale)* so that the energy in a certain frequency band becomes approximately equally important for the computer as for a human listener.
5. Decorrelate the output of the filters so that the histograms of the resulting features can be modeled adequately by mutually independent Gaussian probability density functions.

Feature extraction: FFT

Fast Fourier transform (FFT): from time domain to frequency domain

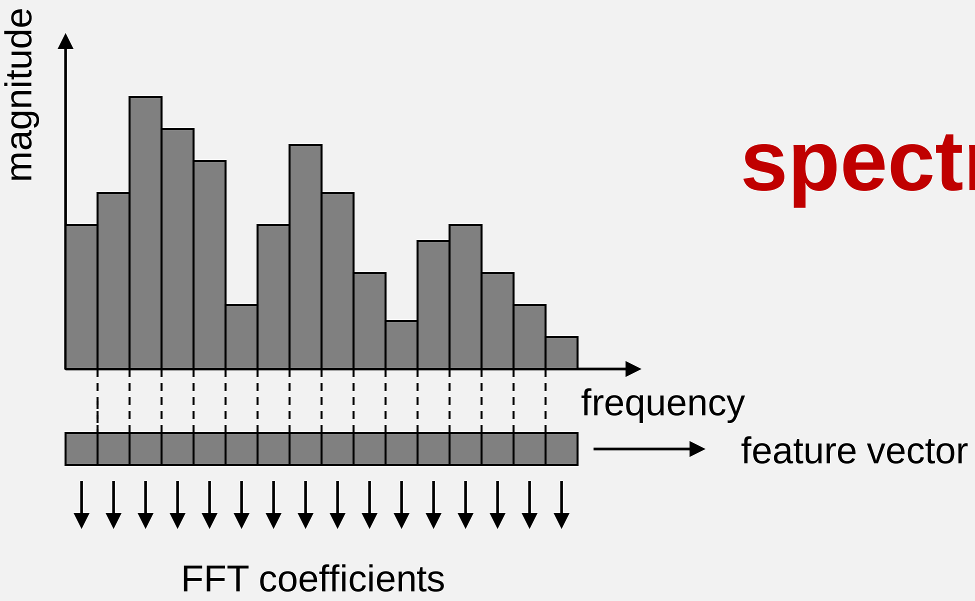
– Jean-Baptiste Fourier: Every waveform is the sum of sine waves with a certain magnitude and phase



Taken from: http://www.bores.com/courses/intro/freq/3_ft.htm

FFT

A *Fast Fourier Transform (FFT)* decomposes the speech signal within each window as a weighted sum of complex exponentials



spectrum

FFT

The resulting FFT coefficients are written in one *vector* (one per frame).

The more coefficients, the more accurate the description, but higher order coefficients may be noisy.

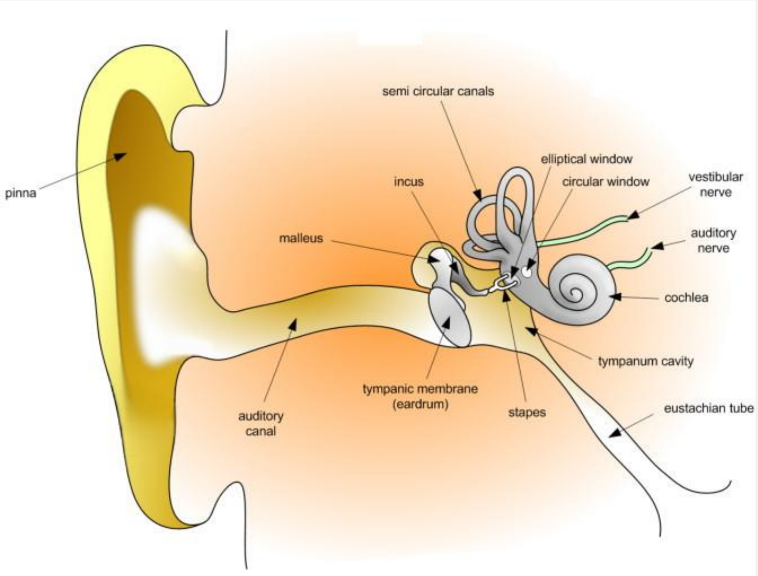
Feature extraction

5 steps to convert a digital speech signal into *acoustic feature vectors*:

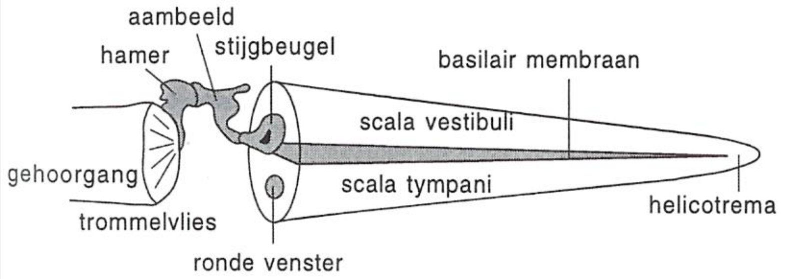
1. Divide the speech signal (oscillogram) in segments with a sufficiently short duration to assume that the vocal tract does not change shape (*frame-based analysis)*
2. Apply *windowing*
3. Convert the windowed signal from the time domain to the frequency domain using the *Fast Fourier Transform (FFT)*
4. Apply perceptual weighting on the spectral energy distribution *(e.g. Mel* or *Bark scale)* so that the energy in a certain frequency band becomes approximately equally important for the computer as for a human listener.
5. Decorrelate the output of the filters so that the histograms of the resulting features can be modeled adequately by mutually independent Gaussian probability density functions.

Feature extraction: Perceptual weighting

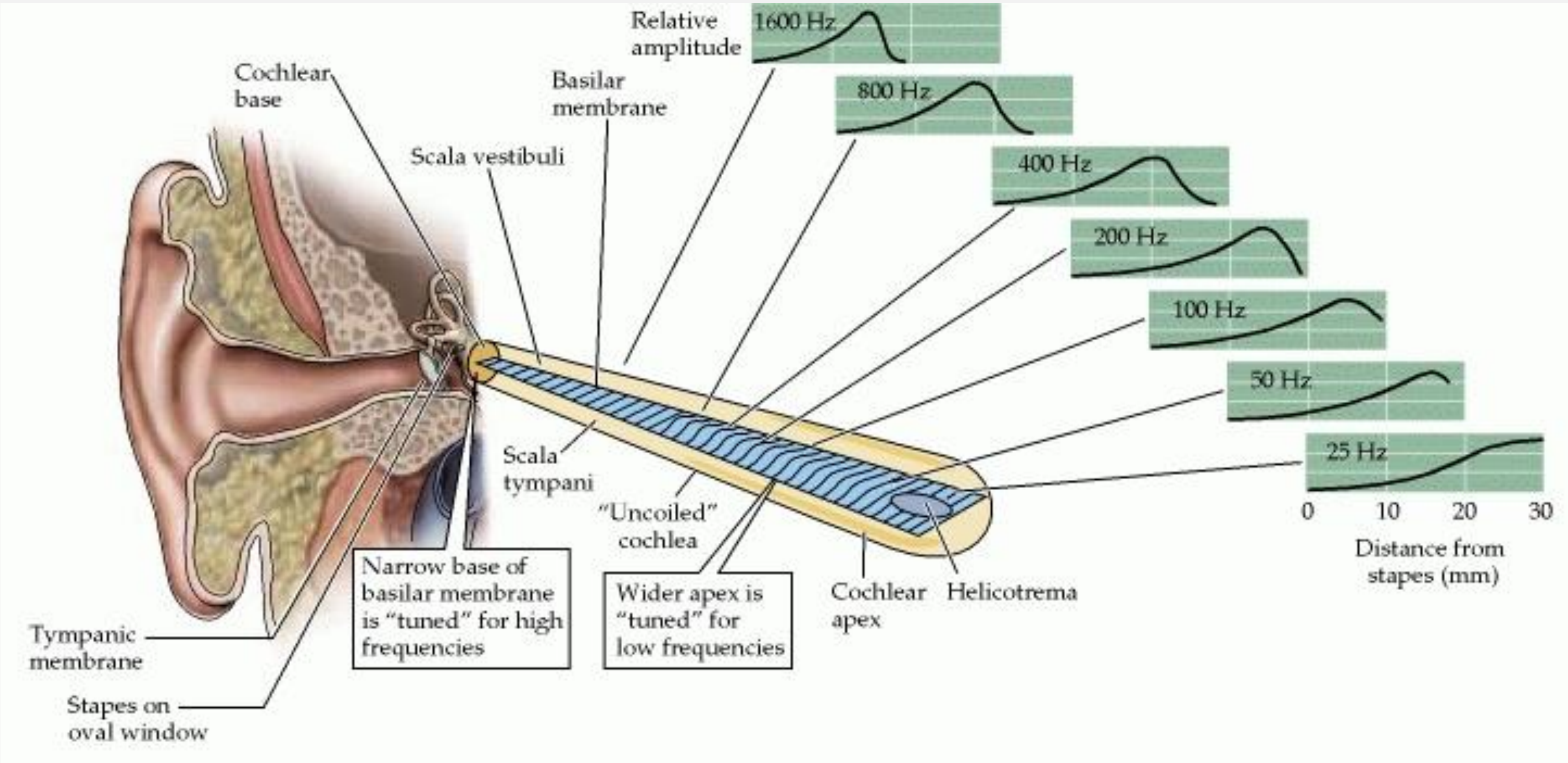
Convert FFT coefficients into a set of features using a perceptual weighting inspired by the processing characteristics of the human auditory system.



malleus, incus, stapes



Cochlea in normal (left, wikipedia) en unrolled form



Relative amplitude
1600 Hz
800 Hz
400 Hz
200 Hz
100 Hz
50 Hz
25 Hz
0
10
20
30
Distance from stapes (mm)
Cochlear base
Basilar membrane
Scala vestibuli
Scala tympani
"Uncoiled" cochlea
Cochlear apex
Helicotrema
Tympanic membrane
Stapes on oval window
Narrow base of basilar membrane is "tuned" for high frequencies
Wider apex is "tuned" for low frequencies

Cochlea

- The cochlea: tube filled with fluid of which the diameter gradually tapers off
  - It is divided into 2 compartiments by the basilar membrane which decreases in thickness
- Approx. 30,000 hair cells attached to the basilar membrane
- A sound makes the basilar membrane vibrate. For a pure tone (a sinusoide) the oscillations are largest at one specific location of the basilar membrane.
- This peak location is determined by the frequency of the tone. At the end: low freqs. At the beginning: high freqs.

→ **TONOTOPIC mapping**

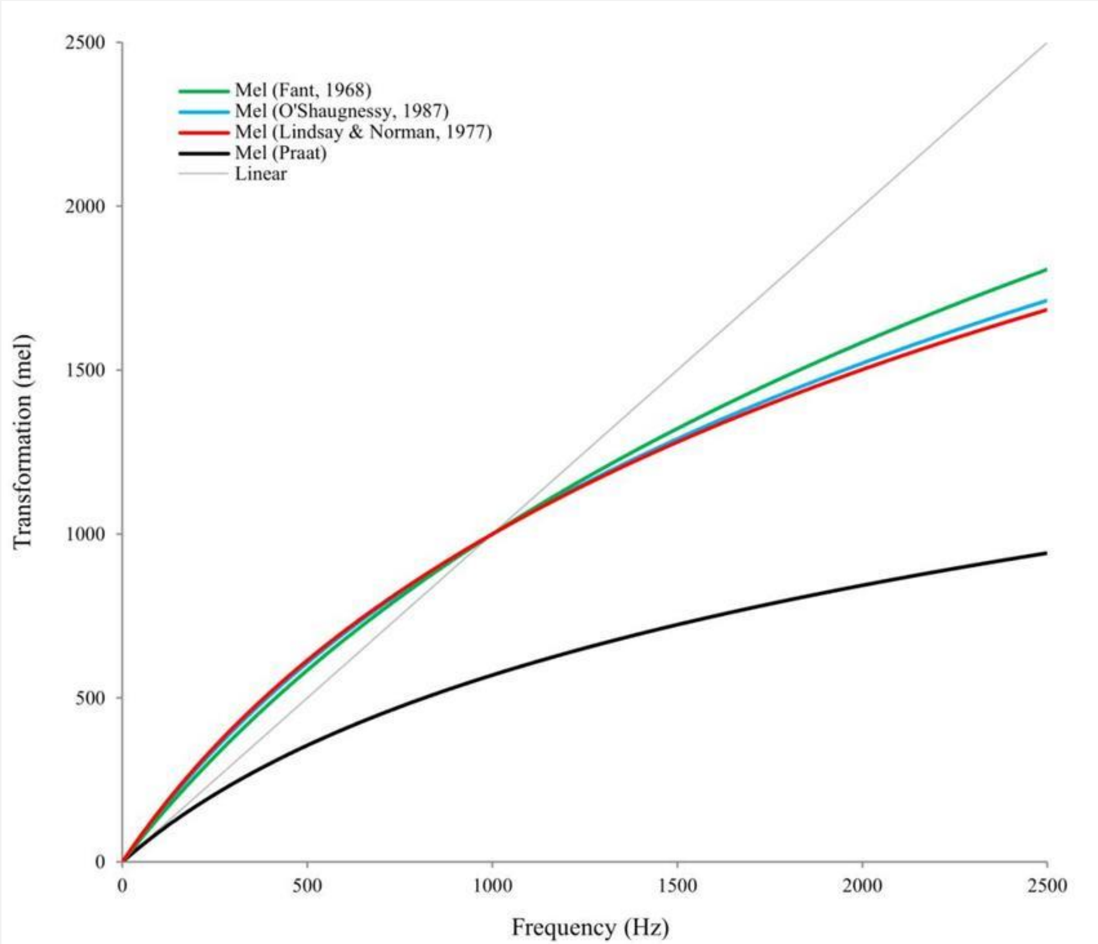
Perceptual weightings

- The human ear is not sensitive to frequency along a linear scale
- Psycho-acoustical frequency scales often used to approximate the human sensitivity are the ***Mel scale*** and the ***Bark scale***

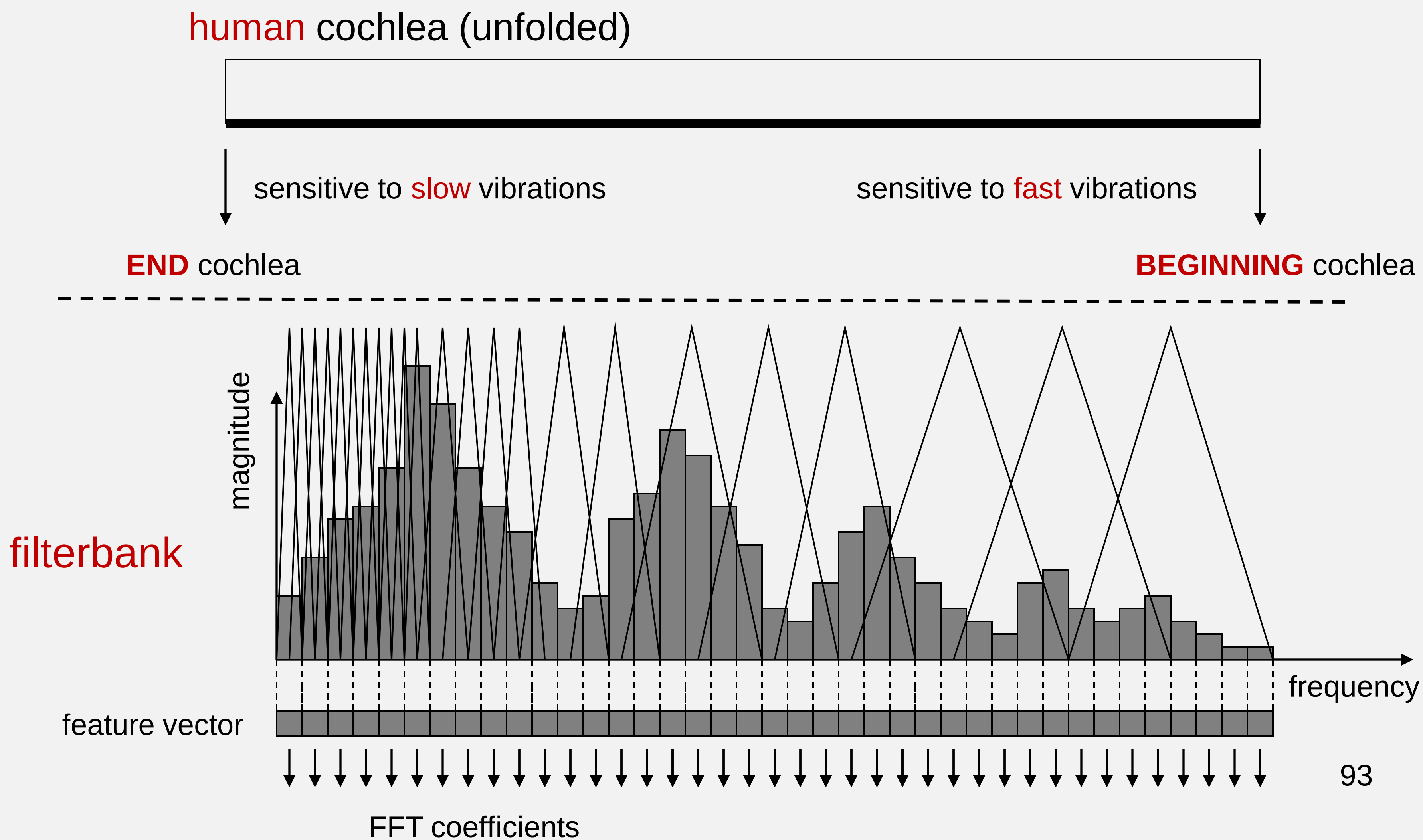
frequency to mel

mel(f) =
1125 log(1+f/700)

There are several other transformations, all log-like.



Mel-filterbank



Apply log() to all energy values in each filter

- E → log(E)

$$\Delta Percept = \frac{\Delta PhysicalQuantity}{PhysicalQuantity}$$

- **Weber's law**
- **https://www.youtube.com/watch?v=hHG8io5qIU8**
  - Energy → loudness
  - Fundamental frequency → pitch (piano)
  - Perception of physical phenomena
  - Estimation of physical quantities
  - Duration, length, pressure, …

Feature extraction

5 steps to convert a digital speech signal into *acoustic feature vectors*:

1. Divide the speech signal (oscillogram) in segments with a sufficiently short duration to assume that the vocal tract does not change shape (*frame-based analysis)*
2. Apply *windowing*
3. Convert the windowed signal from the time domain to the frequency domain using the *Fast Fourier Transform (FFT)*
4. Apply perceptual weighting on the spectral energy distribution *(e.g. Mel* or *Bark scale)* so that the energy in a certain frequency band becomes approximately equally important for the computer as for a human listener.
5. Decorrelate the output of the filters so that the statitics of the resulting features can be modeled by mutually independent Gaussian probability density functions.

Decorrelation

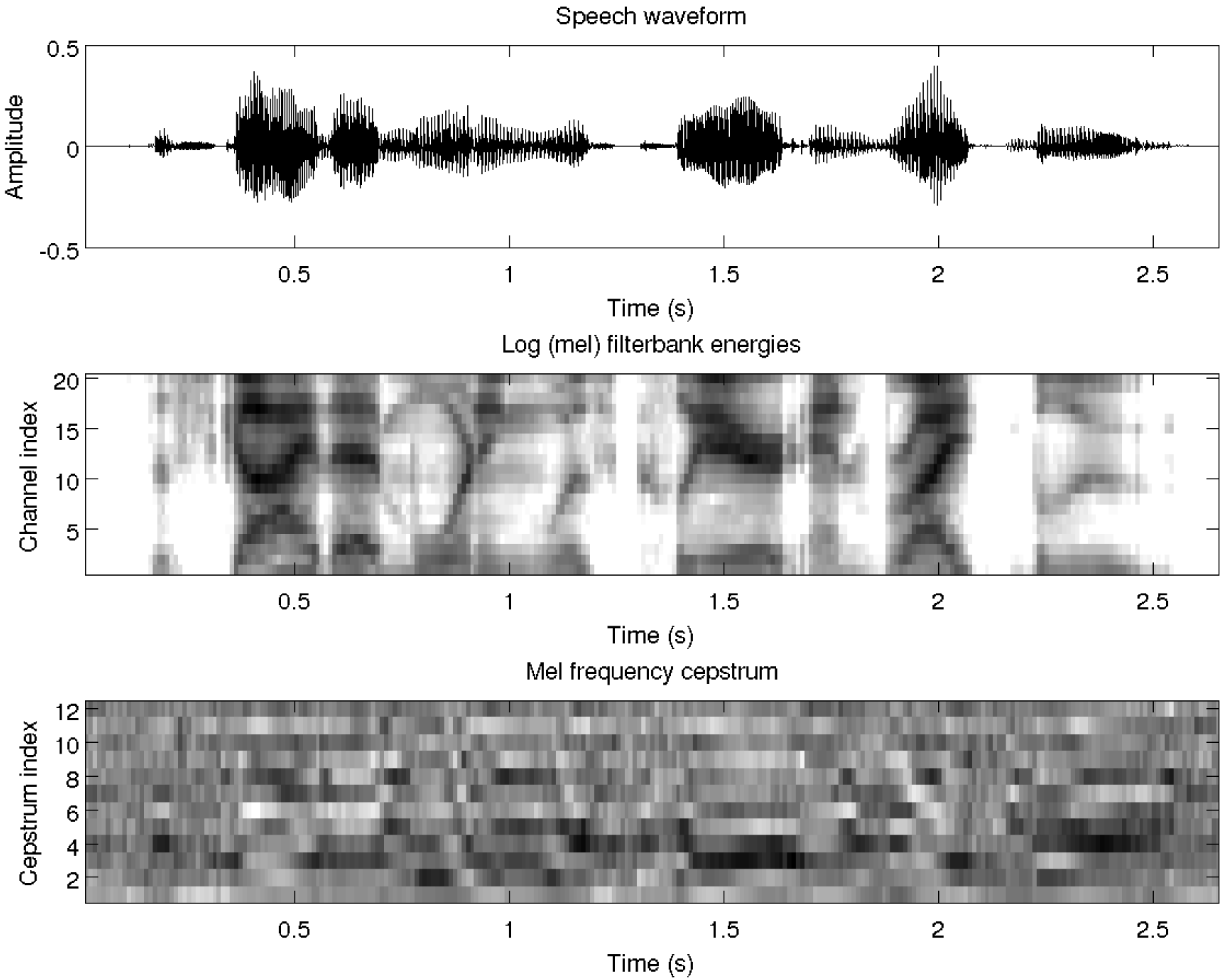
The amount of energy in neighbouring filters is strongly correlated. In order to reduce this correlation, a Discrete Cosine Transform (DCT) is performed on the 21 filter bank coefficients.

As a result we obtain the **Mel Frequency Cepstral Coefficients (MFCCs).**

These MFCCs are approximately statistically independent. The first 12 coefficients $c_1..c_{12}$ suffice to describe the relevant details of the spectrum within each frame.

See e.g.

https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html for comments on this DCT step


Speech waveform
Amplitude
0.5
0
-0.5
0.5
1
1.5
2
2.5
Time (s)
Log (mel) filterbank energies
Channel index
20
15
10
5
0.5
1
1.5
2
2.5
Time (s)
Mel frequency cepstrum
Cepstrum index
12
10
8
6
4
2
0.5
1
1.5
2
2.5
Time (s)

Signal preprocessing

Summary

		output	typical size
0.	A/D conversion	digital signal	-
1.	sliding portion	analysis stretch	400 samples
2.	windowing	windowed signal	400 samples
3.	FFT	spectrum	400 magnitudes
4.	filterbank	feature vector	21 filterbank energies
5.	MFCC	smaller feature vector	12 features

assuming 16kHz, 25 ms analysis frame

MFCC vectors as audio representation

Very many sites show information about MFCCs, often with useful Python function calls

See e.g.
https://www.kaggle.com/ilyamich/mfcc-implementation-and-tutorial

https://pypi.org/project/python_speech_features/

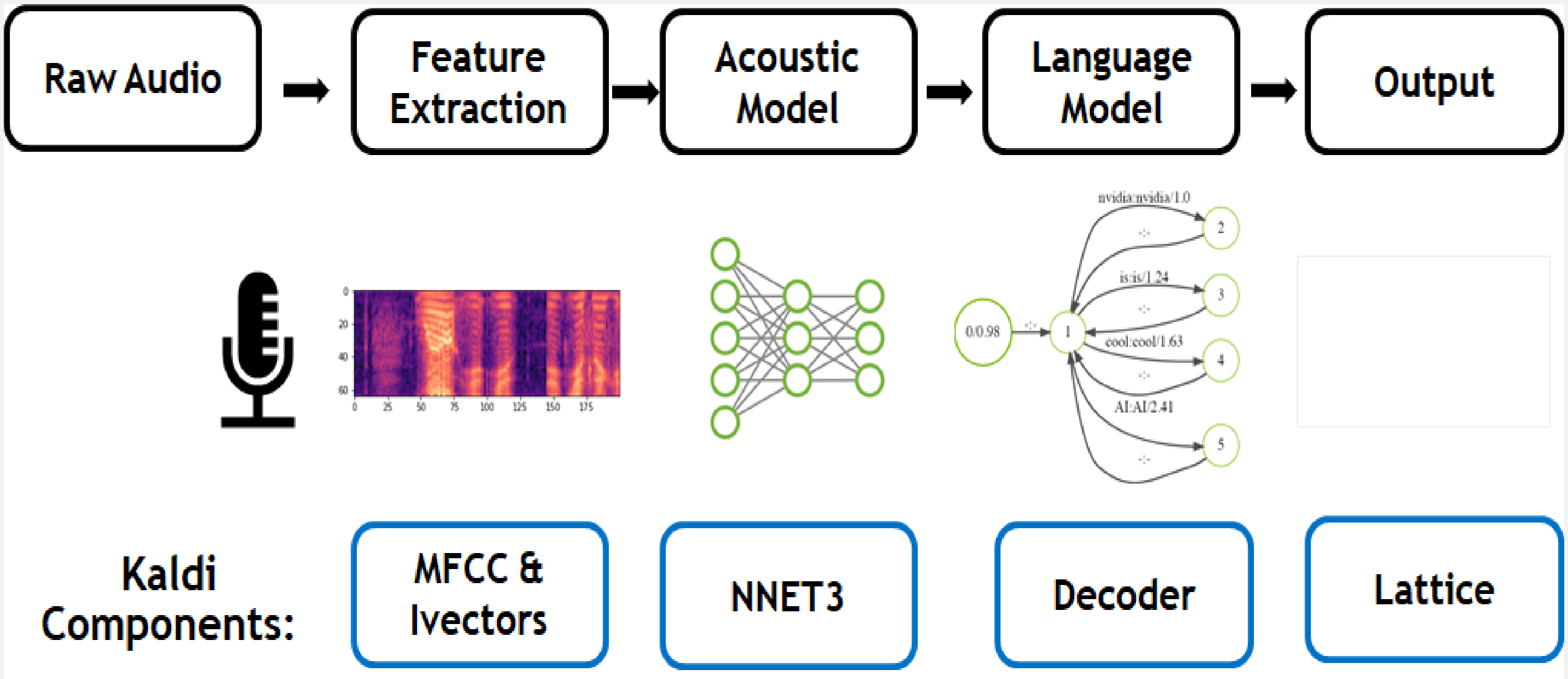
Librosa python library
https://librosa.org/doc/latest/index.html

Difference between image and speech recognition/classification

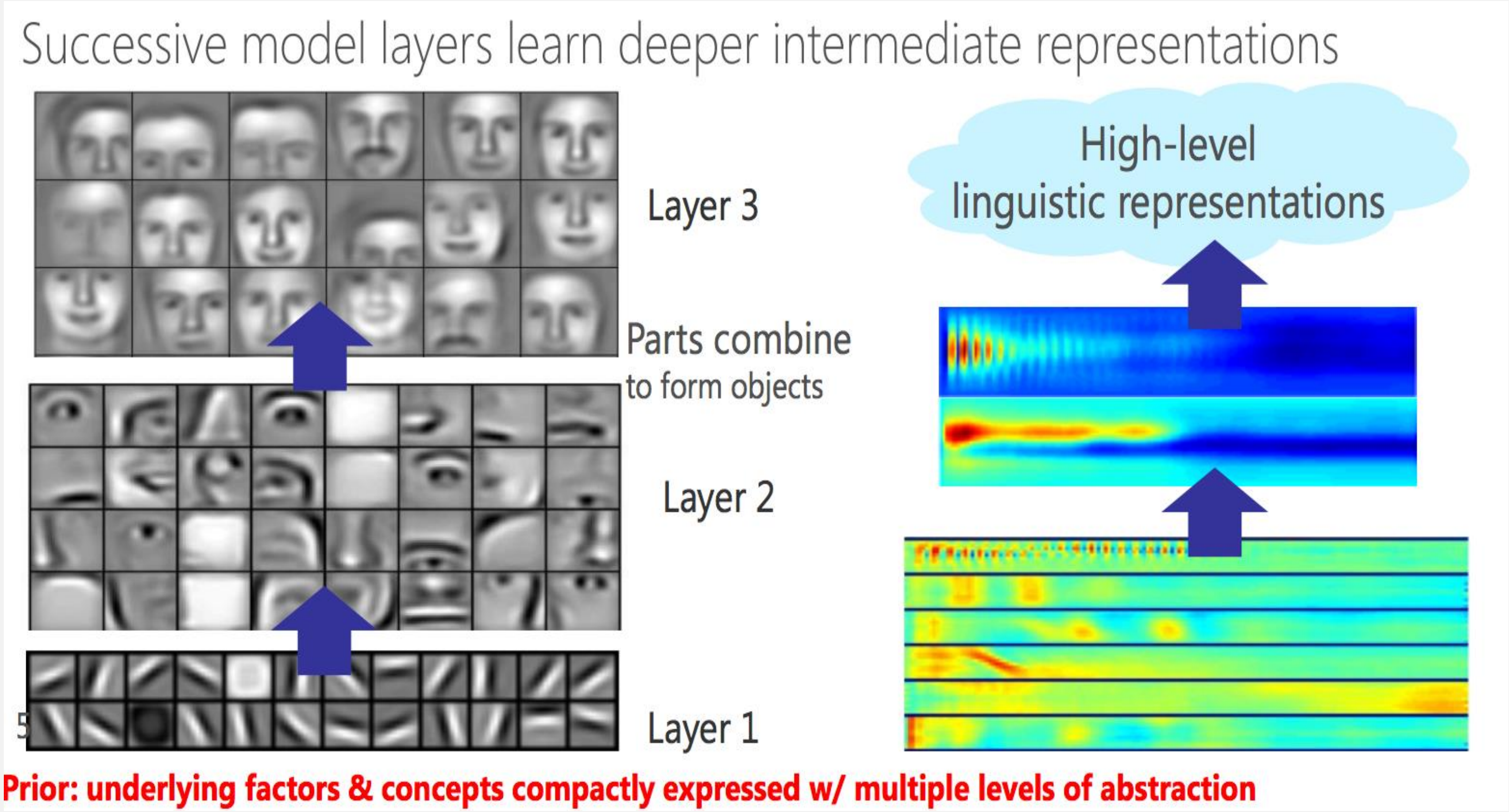
- Images
  - Static (often)
  - Size 400 x 400 (or so)
  - Foreground hides background
  - Focus

- Speech
  - No clear segmentation
  - Unfolding over time (100 spectral frames/s)
  - Words are not immediately available
  - All phones are context/speaker/mood/ … dependent

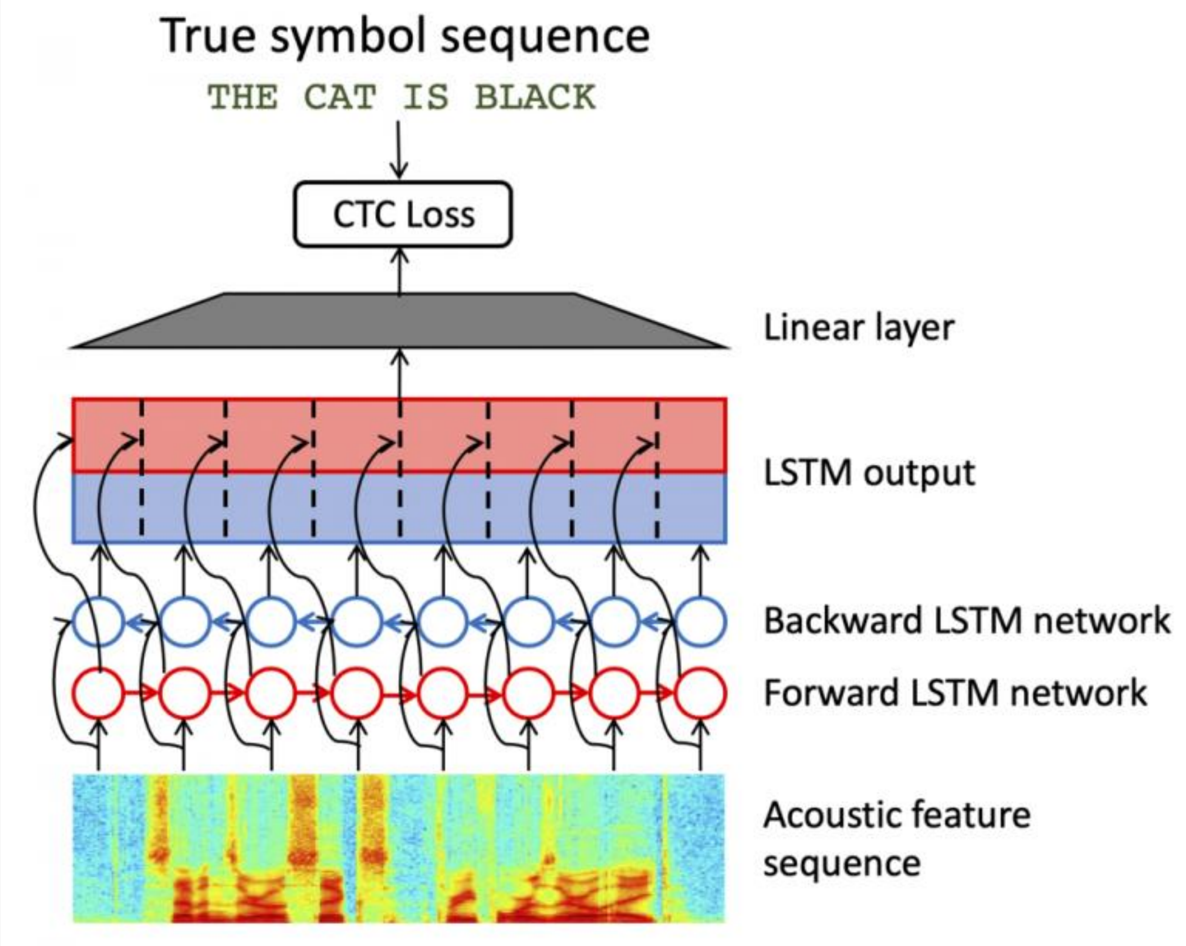
Classical and new (deep) learning



https://devblogs.nvidia.com/gpu-accelerated-speech-to-text-with-kaldi-a-tutorial-on-getting-started/



Lee, Largman, Pham & Ng, NIPS 2009



Nowadays, ASR is an example of deep learning

Figure from paper by IBM team, Kurata et al. (2019)

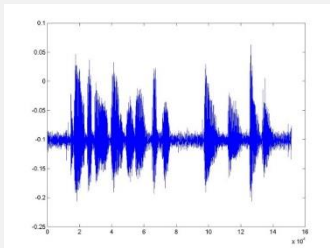
	pro	**con**
Classical approach	Insight-based	Lower performance
Recent deep learning approaches	Higher performance (relative 30-60% reduction error rates)	What do ***we*** learn? Explainable AI Open AI Responsible AI

Deep learning in ASR

- It is very useful to know about the conventional approach before diving into DNN-based approaches

- Nowadays hundreds of (highly specialized) papers on DNNs in ASR

conventional ASR



useful features

basic speech units

words

sentences

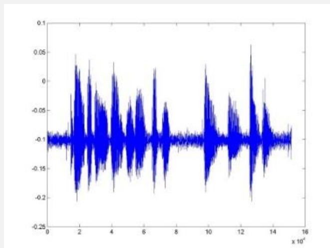
hidden markov models

Gaussian mixtures

dictionary
appel ɑ p ə l

language model

modern ASR



useful features

basic speech units

words

sentences

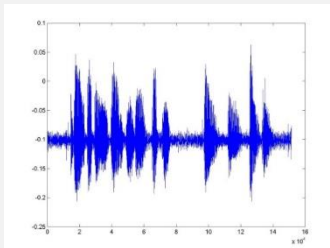
hidden markov models

DNN

FST

FST

modern ASR



useful features

basic speech units

words

sentences

hidden markov models

DNN

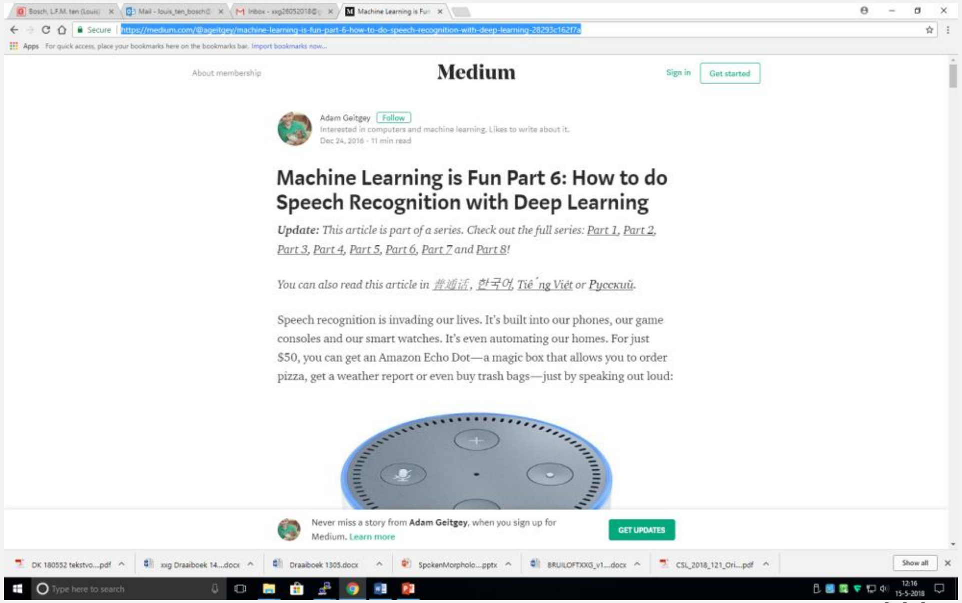
FST

FST

DNN
end-to-end

ASR and deep learning

- https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a



ASR techniques

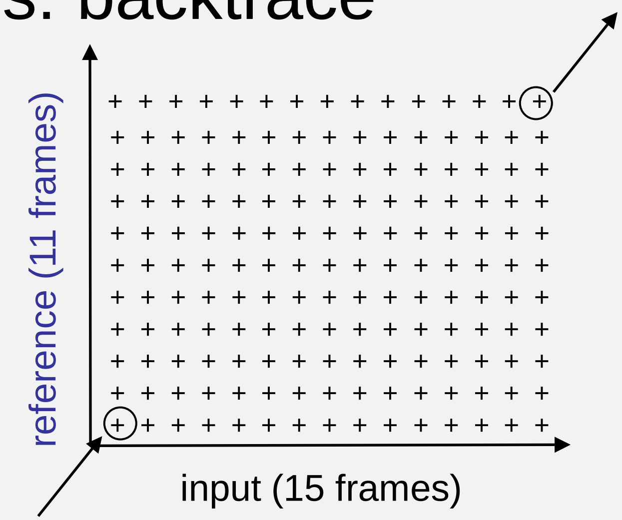
- Template matching ('70)
  - Representative speech patterns (e.g. words, syllables) are stored
  - And labeled
  - New unknown data identified by comparison
- Probabilistic matching ('80-now)
  - Often by Hidden Markov Modeling (HMM)
  - An HMM is characterized by hidden states graph + state-state transition probabilities
  - Each state is characterized by a (statistical) distribution in the acoustic space
- ANN-based (2012-now)
  - Many directions, most of them based on earlier approaches

Template matching

- Used 1960-70 for *isolated* words
  - Record a database of words from same speaker (or different speakers)
  - In the test, compare a test item to all of the stored speech tokens
- Distance: Dynamic Time Warping
  - DTW: a way to define a “distance” between sequences of unequal length (see later)

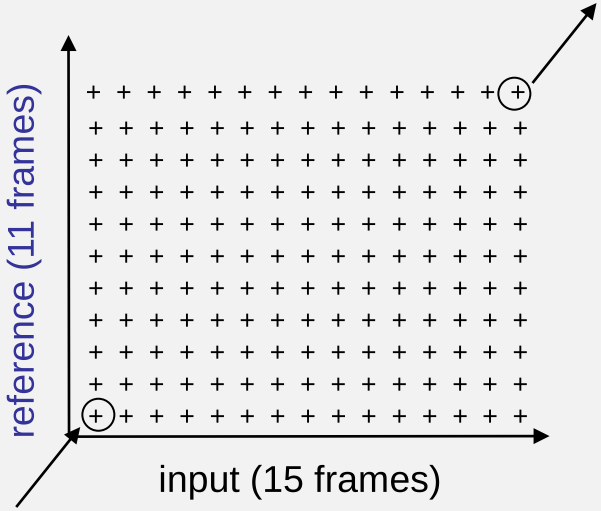
Template based recognition

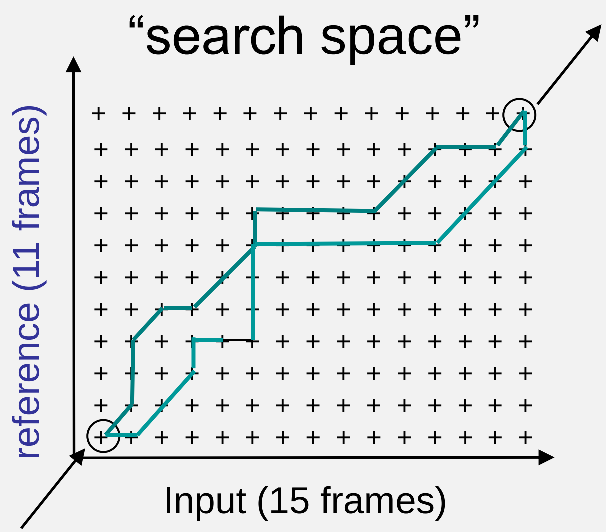
- By dynamic programming
- Define a trellis matrix (or lattice)
- First pass: in each node (cell), compute all local matching scores, compute forward global scores and keep pointers
- Second pass: backtrace



dynamic programming, Viterbi algorithm

This algorithm provides the best alignment between input and reference sequences





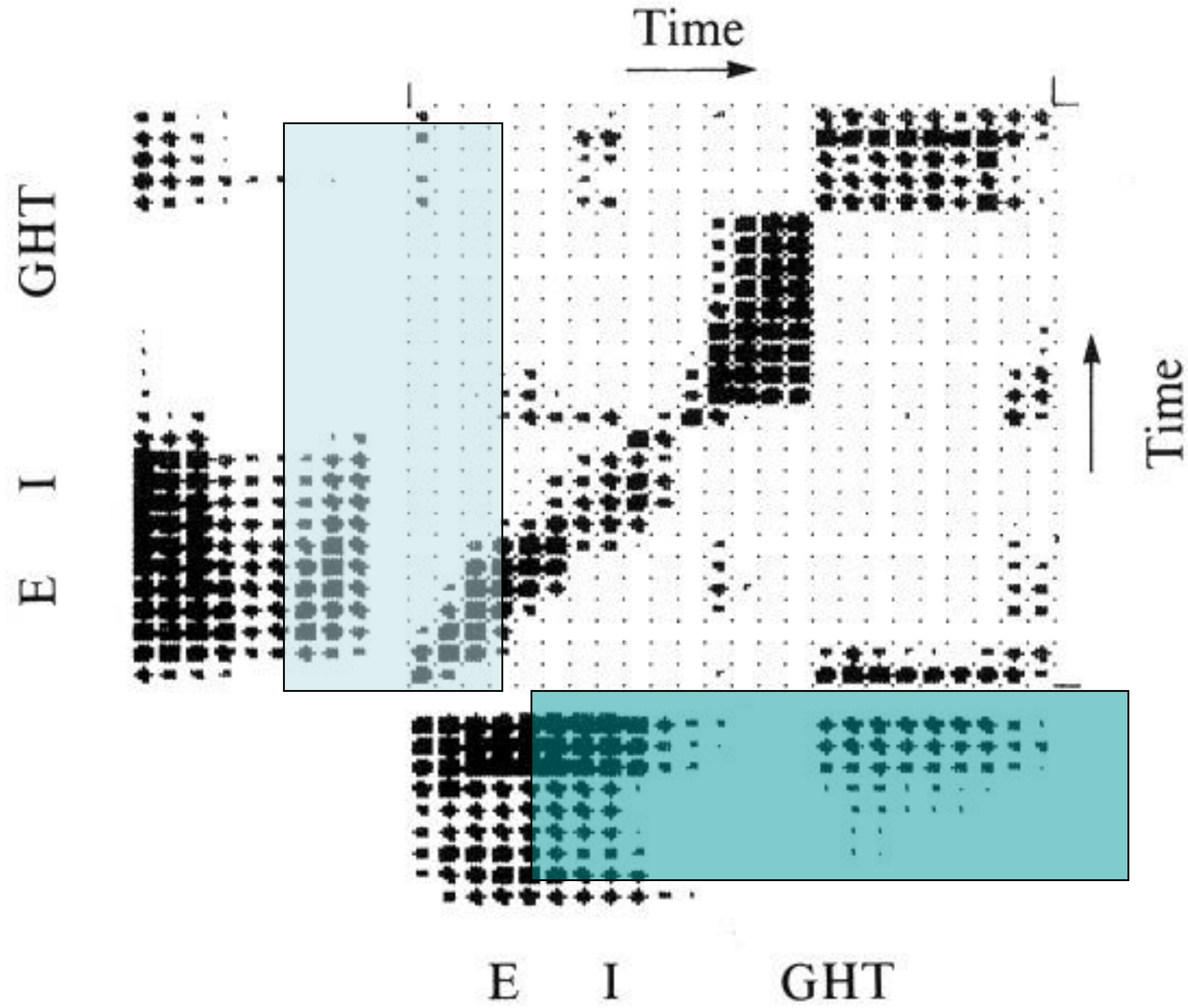
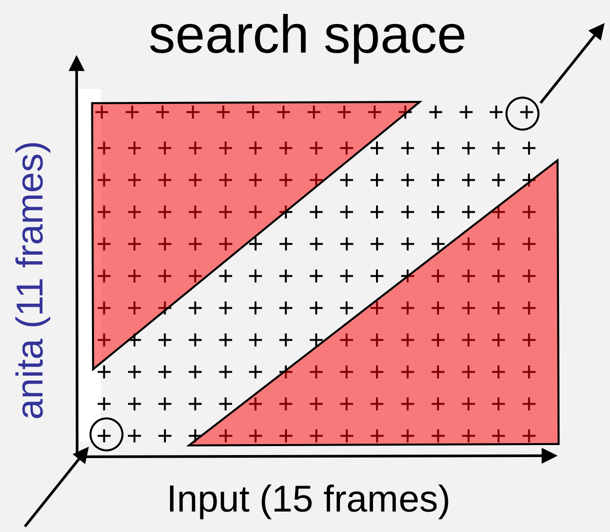


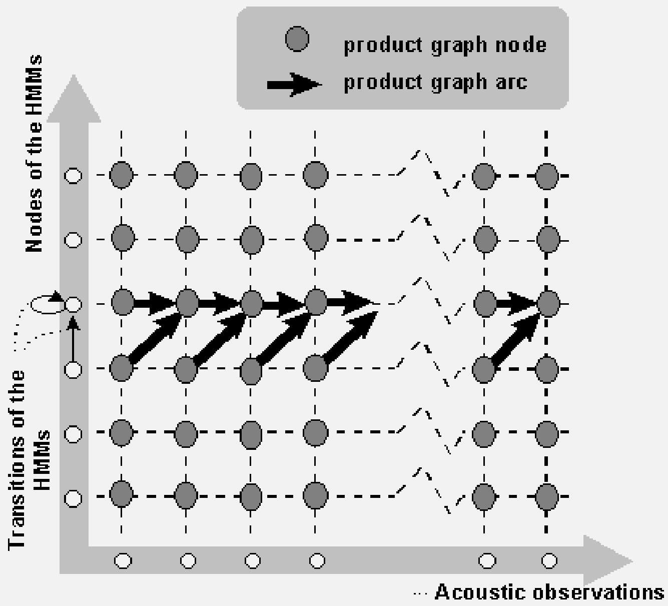
Fig. 7.2 Graphical representation of Euclidean distance between frames of the spectrograms shown in Fig. 7.1. The larger the blob the smaller the distance. It can be seen that there is a path of fairly small distances between the bottom left and top right when two examples of 'eight' are compared,

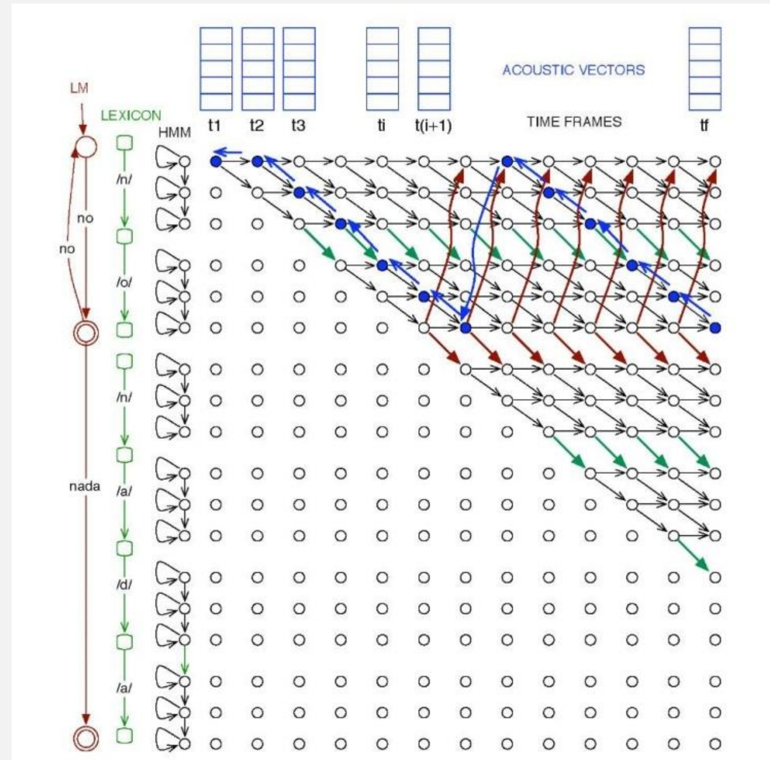
pruning

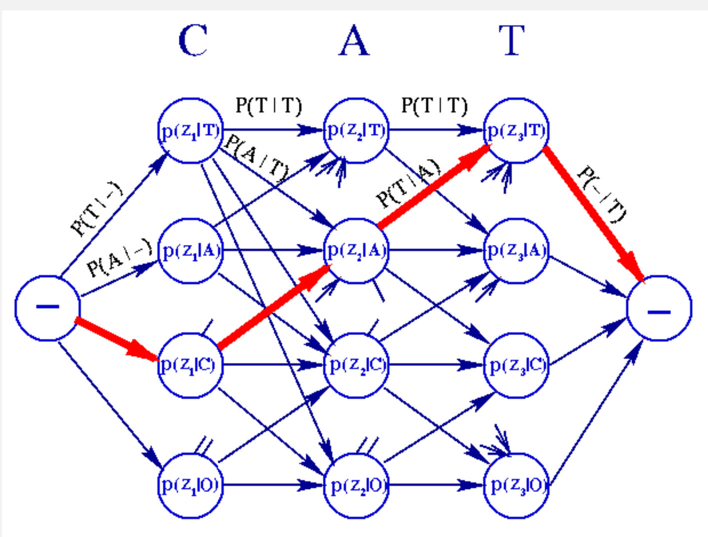
Computational costs can be further reduced by pruning: prohibit too unlikely alignments by applying local scores that prohibit/penalize implausible search areas



Other examples of dynamic search







1980-1990: towards probabilistic models

Three main factors played a role in ASR advancement

- more speech data & more disk space
- More powerful CPU/GPU
- better algorithms for estimating Hidden Markov Models parameters (Baum-Welch, 1972)

Exactly these factors also play a role in the current network approaches in ASR.

Bayes plays key role in almost all approaches

$$P(A|B)=\frac{P(A\cap B)}{P(B)}$$

Definition of P(A|B)

$$P(A\cap B)=P(A|B)P(B)$$

$$P(A|B)P(B)=P(B|A)P(A)$$

$$P(A|B)=\frac{P(B|A)P(A)}{P(B)}$$

Bayes plays key role in almost all approaches

$$P(A\mid B)=\frac{P(A\cap B)}{P(B)}$$

Definition of P(A|B)

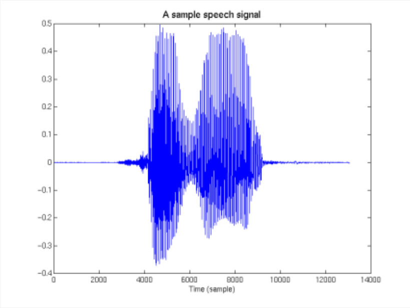
$$P(A\cap B)=P(A\mid B)P(B)$$

$$P(A\mid B)P(B)=P(B\mid A)P(A)$$

$$\boxed{P(A\mid B)=\frac{P(B\mid A)P(A)}{P(B)}}$$

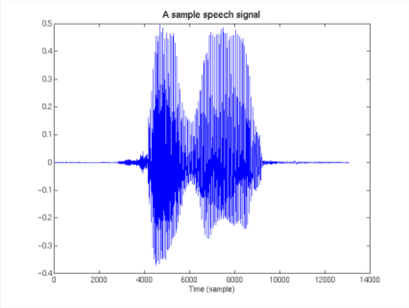
Bayes in ASR

P(I said|



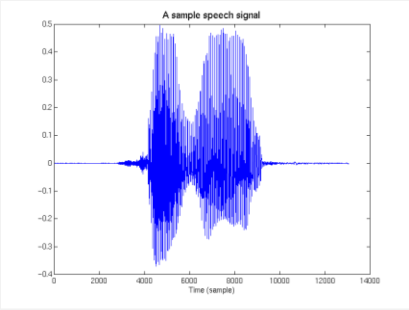
)

P(I sad|



)

P(I sat|



)

Bayes in ASR

$$P(\textit{I said} \mid \textit{utterance})$$

$$P(\textit{I sad} \mid \textit{utterance})$$

$$\ldots$$

$$P(\textit{word(s)} \mid \textit{utterance})$$

Bayes in ASR

P(I said | utterance)

P(I sad | utterance)

…

P(word(s) | utterance)

hypothese

Bayes in ASR

$$\boldsymbol{P(word(s) \mid utterance\ )}$$

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

$$P(word \mid utterance) = \frac{P(utterance \mid word)P(word)}{P(utterance)}$$

Bayes

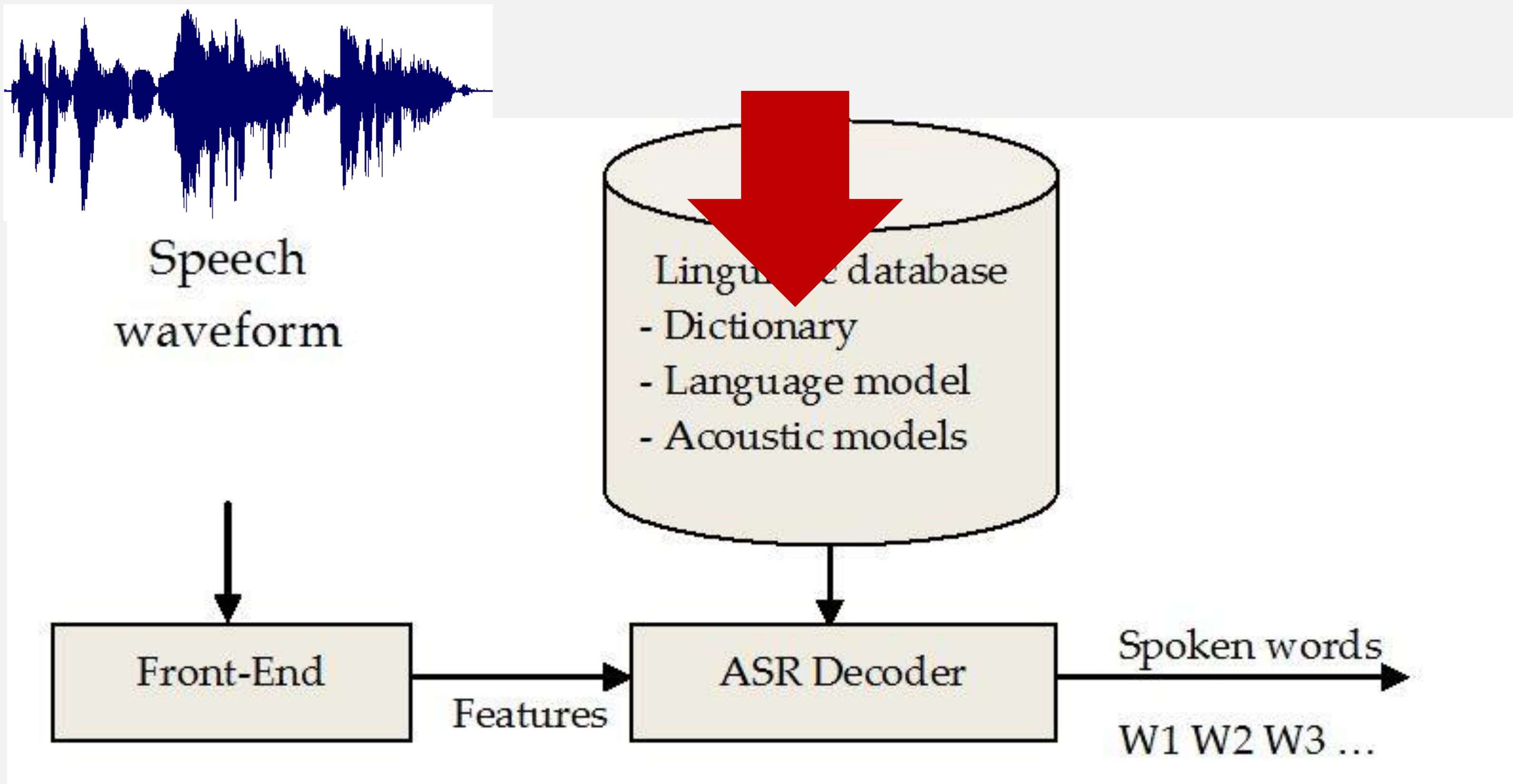
$$P(word \mid utterance) = \frac{P(utterance \mid word)P(word)}{P(utterance)}$$

$$\arg\max P(word_i \mid utterance) =$$

$$\arg\max P(utterance \mid word_i)P(word_i)$$

AM **LM**

ASR architecture



Lexicon/dictionary

word form	phonemic transcription
bang	b A N
bankbiljet	b A N b I l j E t
barbaren	b A r b a r @
barbecue	b A r b @ k j u w
onmiddellijk	O m I d @ l @ k
yesterday	j E s t @ r d e
goedemorgen	x u @ m O r x

Lexicon/dictionary (Dutch)

word form	phonemic transcription
bang	**b A N**
bankbiljet	**b A N b I l j E t**
barbaren	**b A r b a r @**
barbecue	**b A r b @ k j u w**
onmiddellijk	**O m I d @ l @ k**
yesterday	**j E s t @ r d e**

For a specific language, lexicons are often available, but not always a clean source of information.
Multiples used?
Phonetic alphabet used?
Compounding? (*Rechtsschutzversicherungsgesellschaften*)

Lexicons…

A pronunciation lexicon lists for every word the sequence of phonetic symbols that describes its pronunciation

- Pronunciation mostly in terms of broad phonetic transcription
- Usually no lexical stress
- Usually no additional info, such as POS, morphological properties

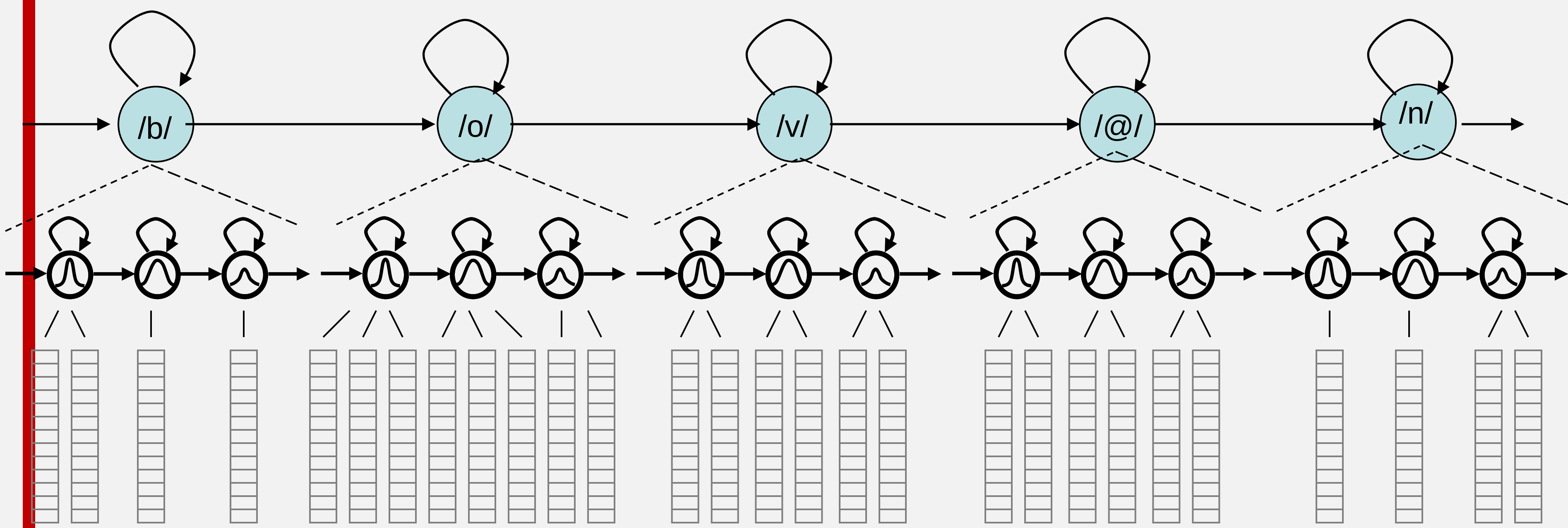
Two types:

- *Canonical lexicon*: one pronunciation/word (norm pronunciation)
- *Multiple pronunciation lexicon*: different pronunciations/word to account for possible pronunciation variation

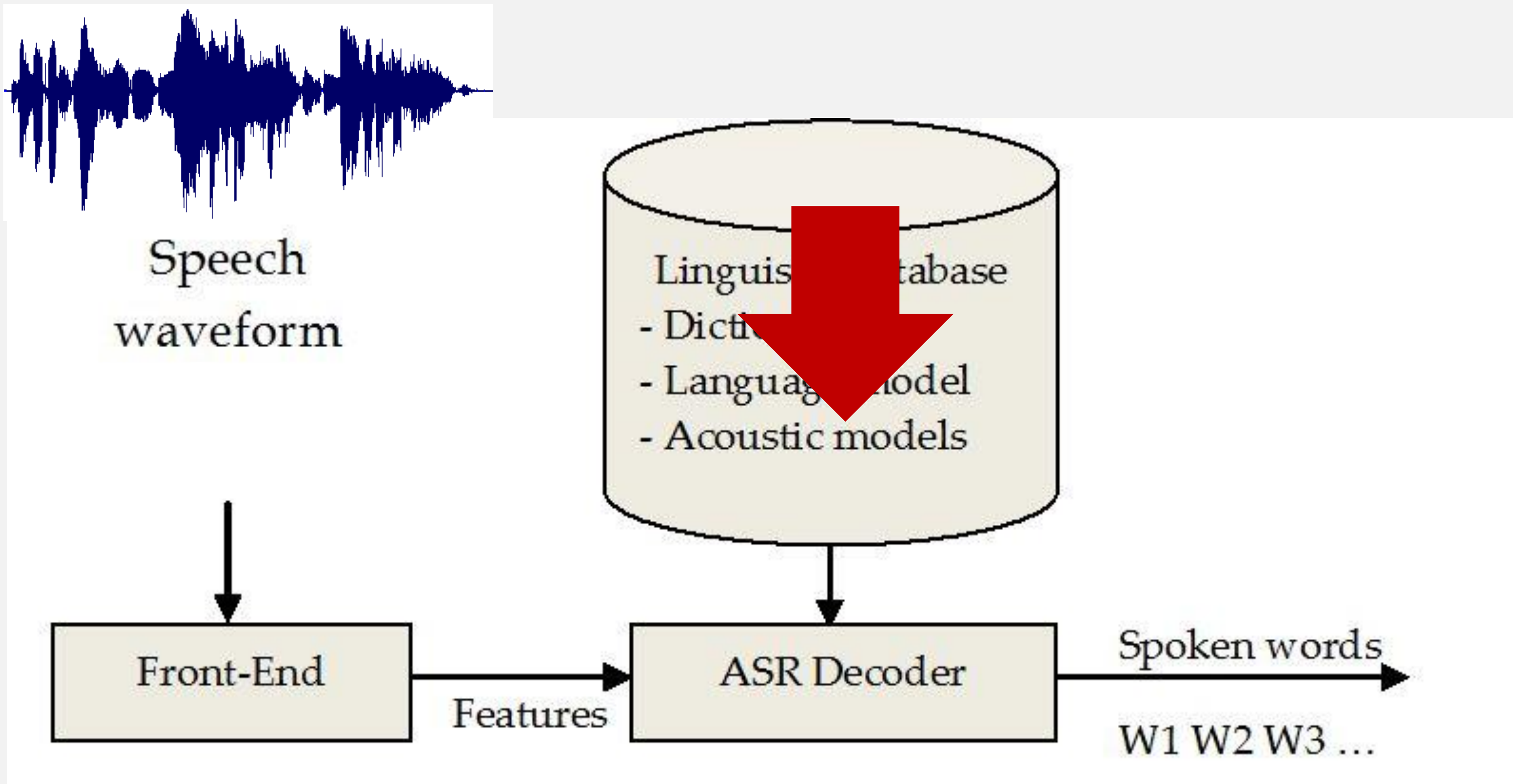
	Pro	Con
Canonical	simple	actual pronunciation may mismatch (reduction phenomena)
Multiple pronunciation	better description of actual realization of the speech signal	Increased confusion across words (homophones)

Modeling words from phones

A word can be represented as a concatenation of phone models



ASR architecture

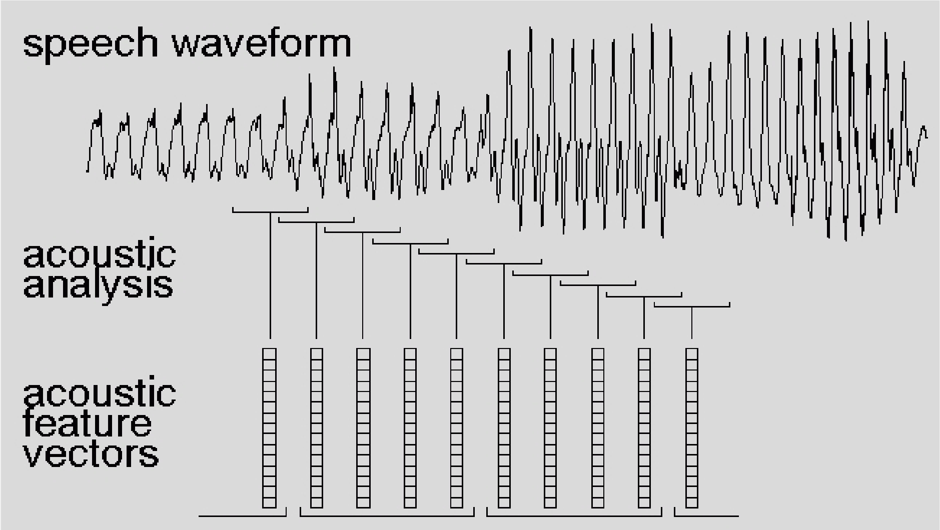


Acoustic model

- The AM describes the statistics of speech sounds, for each speech unit

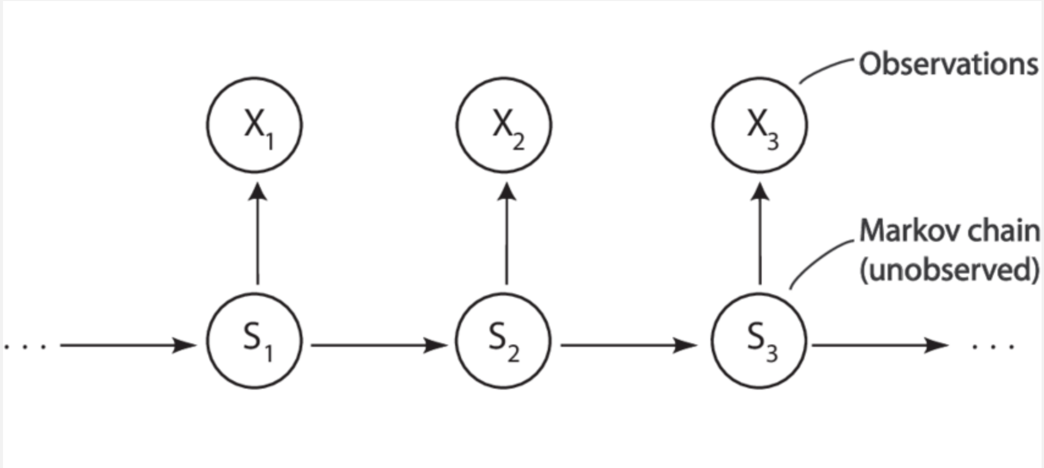
- Different speech units can be used, e.g.:
  - Phones (monophones): p, t, k, f, v, a, i, u,…
  - Triphones, quintphones, …
  - Syllables
  - Words
  - Sentences
  - Etc.

Hidden Markov Models



Speech signal

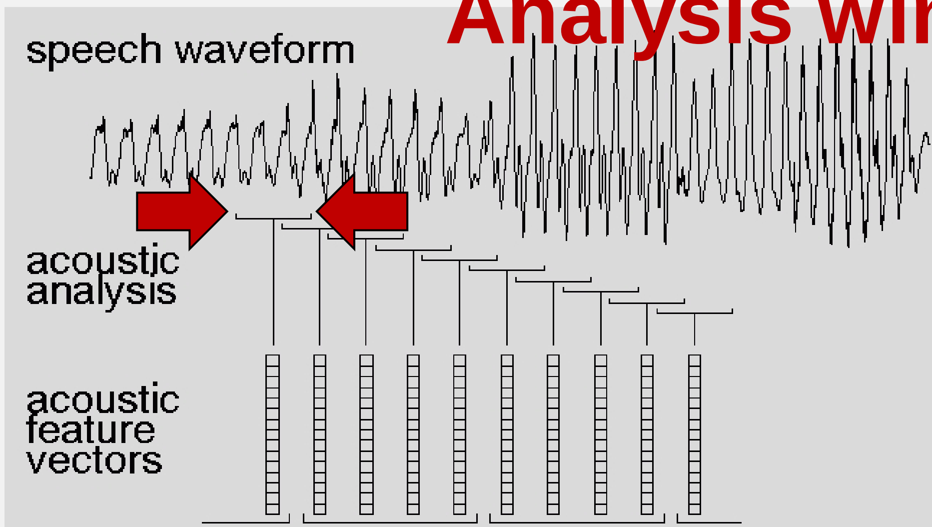
Acoustic vectors [x]



States of high level representations (phones, syllables, words)

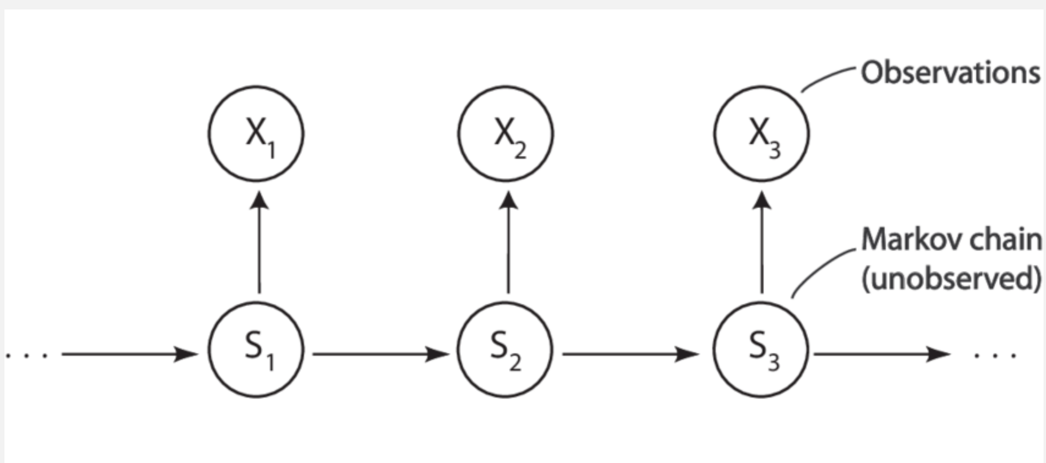
Hidden Markov Models

Analysis window duration



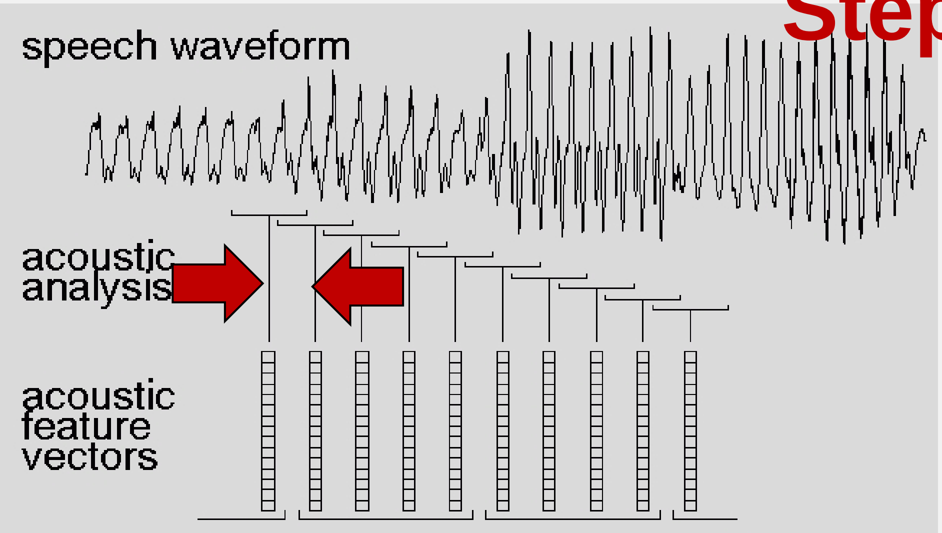
Speech signal

Acoustic vectors [x]



States of high level representations (phones, syllables, words)

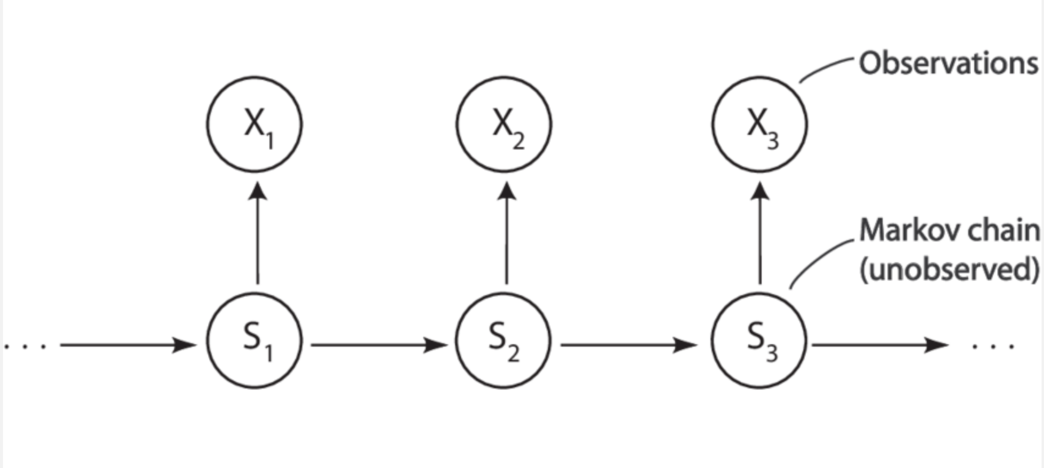
Hidden Markov Models



Step size

Speech signal

Acoustic vectors [x]



States of high level representations (phones, syllables, words)

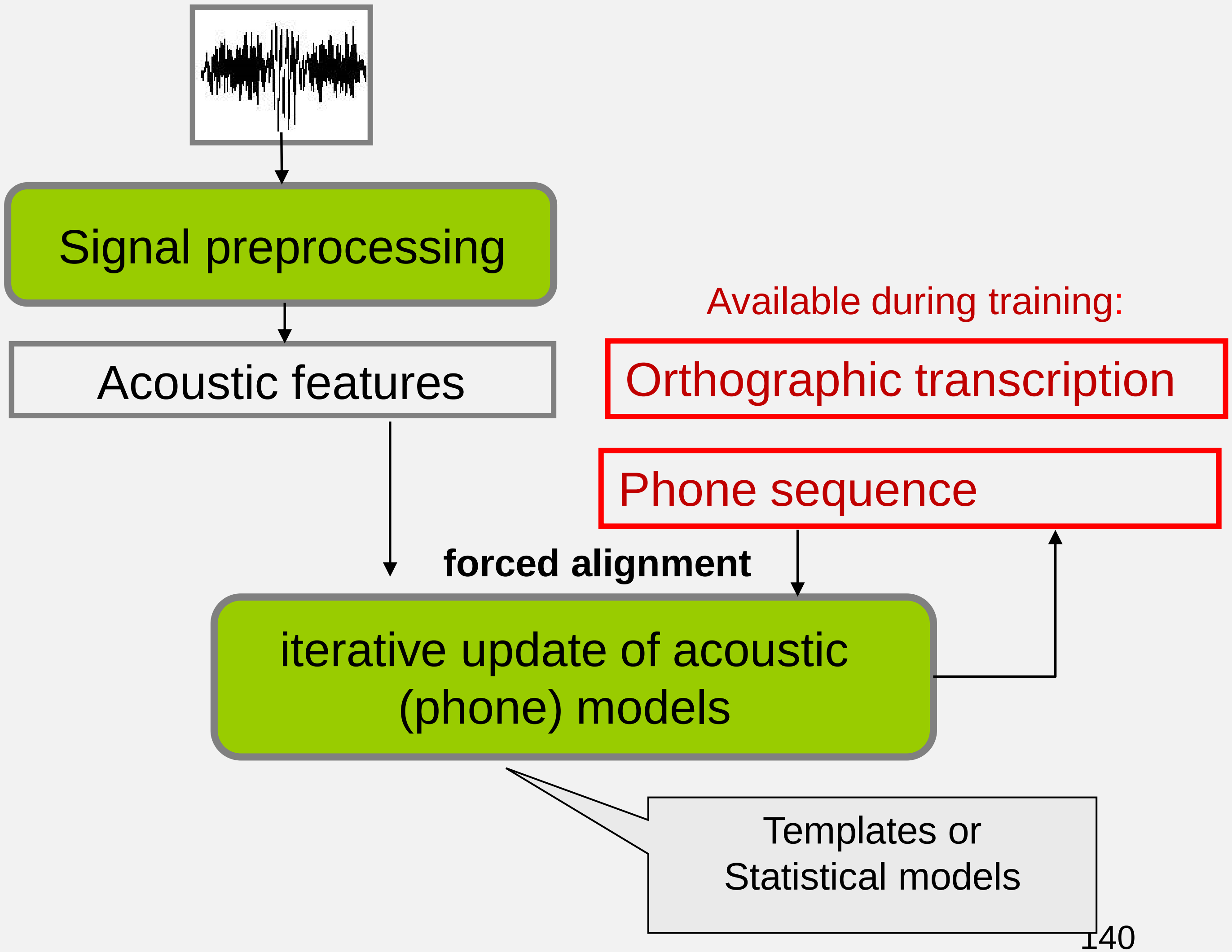
Acoustic models

- Describe the statistics of the corresponding acoustic feature vectors per speech unit
- Different speech units can be used:
  - Sentences
  - Words
  - Syllables (e.g., Mandarin)
  - Phones
- Mostly implemented as HMMs (3-state)
- The most frequently used units are phones (because very generic), triphones, quintphones
  - Triphone = 1-left-1-right context-dependent phone
  - Quintphone = 2-left-2-right context-dependent phon

Some numbers

- Languages use about 30 – 50 phones
- An AM contains 30 – 50 context independent phones
  - ‘apple’ E p @ l
- May contain 2000 – 3000 triphones
  - ‘apple’ ${}_{\#}E_{p}$ ${}_{E}p_{@}$ ${}_{p}@_{l}$ ${}_{@}l_{\#}$
- Several other options

ASR in a diagram: training

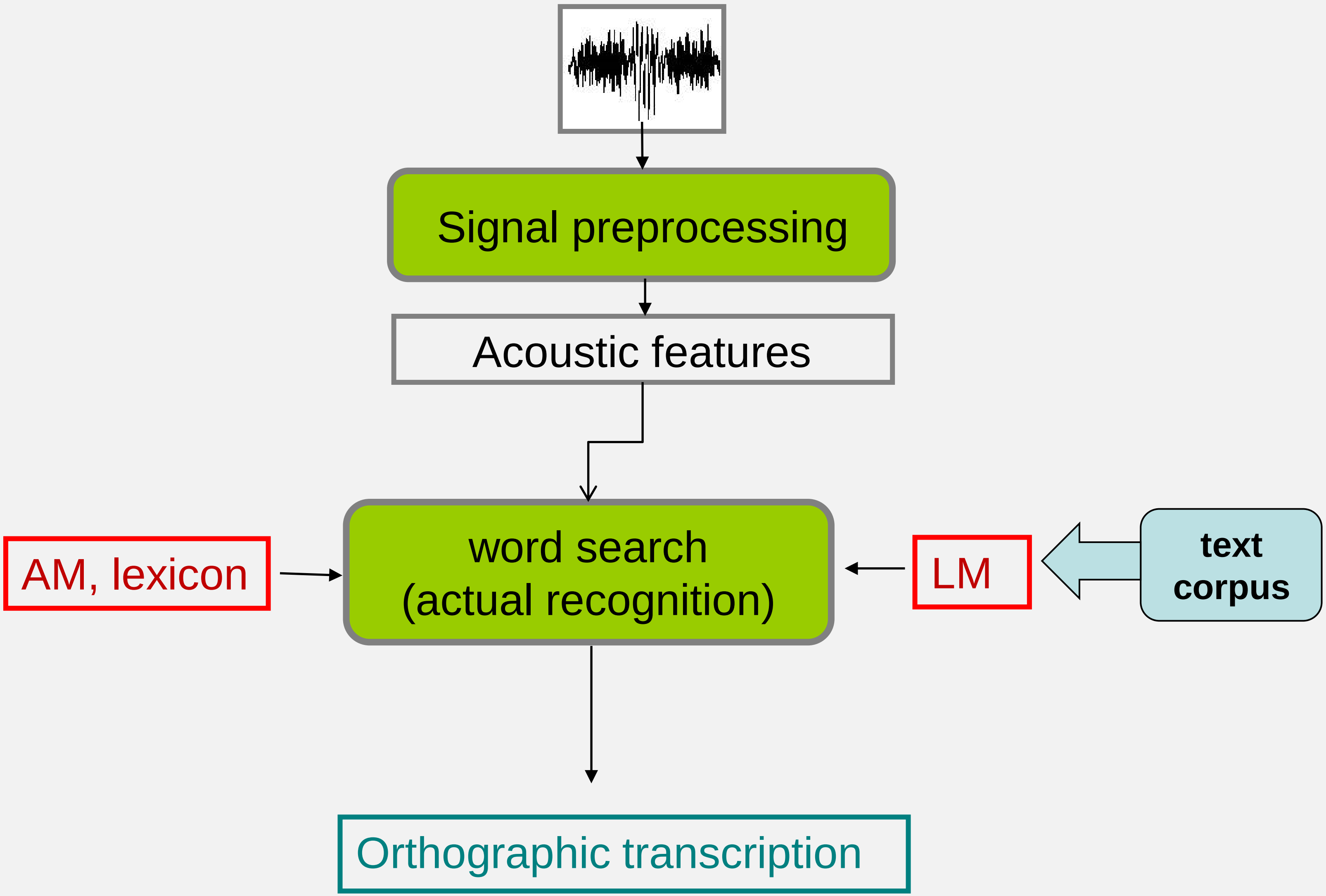


Model based ASR: training

(simplified)

1. Starting material: A *training corpus* with:
   - *Speech*
   - *Orthographic transcriptions*
   - *Pronunciation dictionary (lexicon)*
2. Create lousy acoustic HMM (phone) models
3. The transcriptions are “forced aligned” with the speech signal (no LM)
4. Corresponding HMM models are updated
5. Back to 3, until convergence

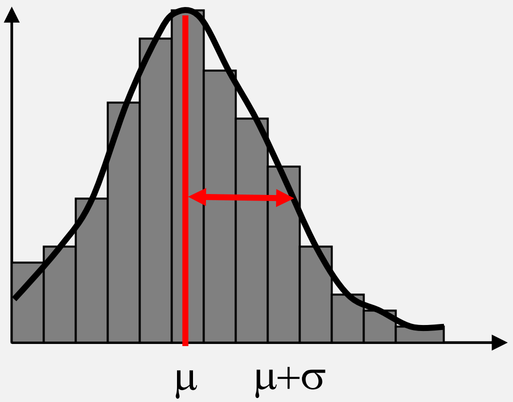
ASR in a diagram: test



Statistics per HMM state

Via DNN, or via a Gaussian mixture

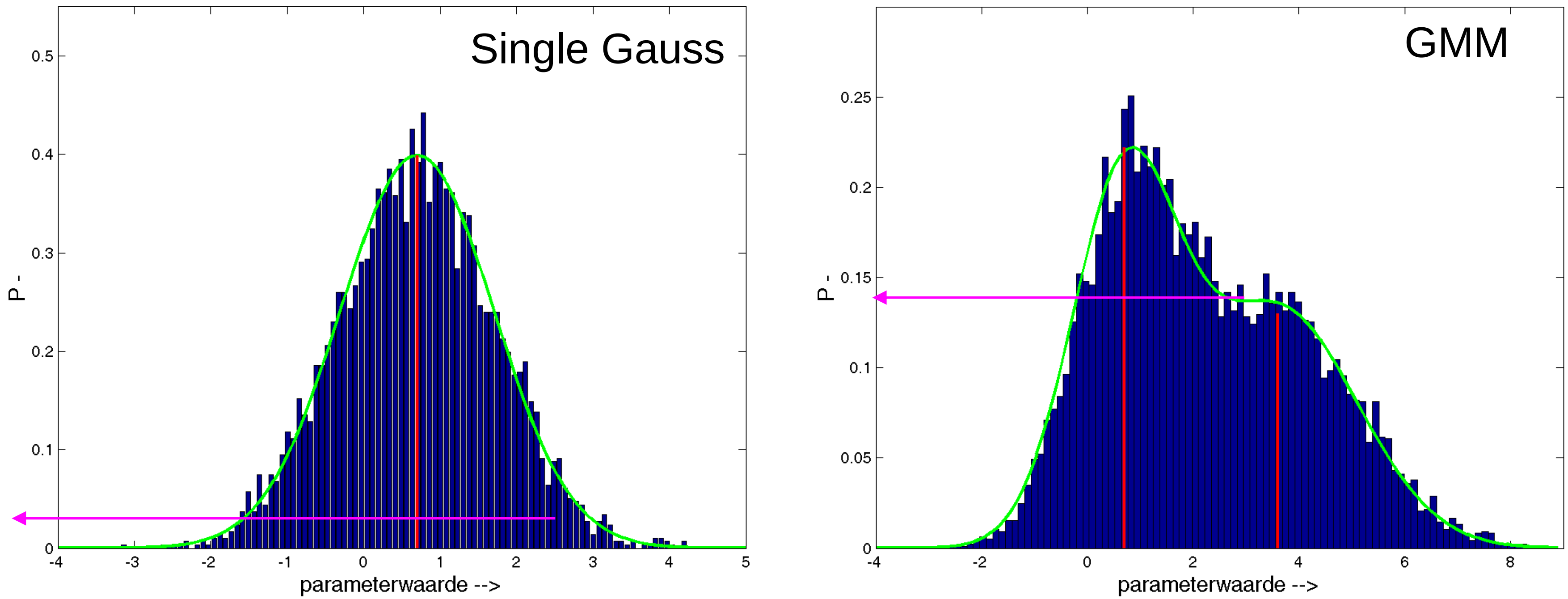
For every state of the HMM, the distribution of the corresponding feature vectors is obtained by using the frame-to-state relations from a forced alignment using previous HMMs



Assuming a normal distribution

Mean (μ) and stdev (σ)

Model based ASR



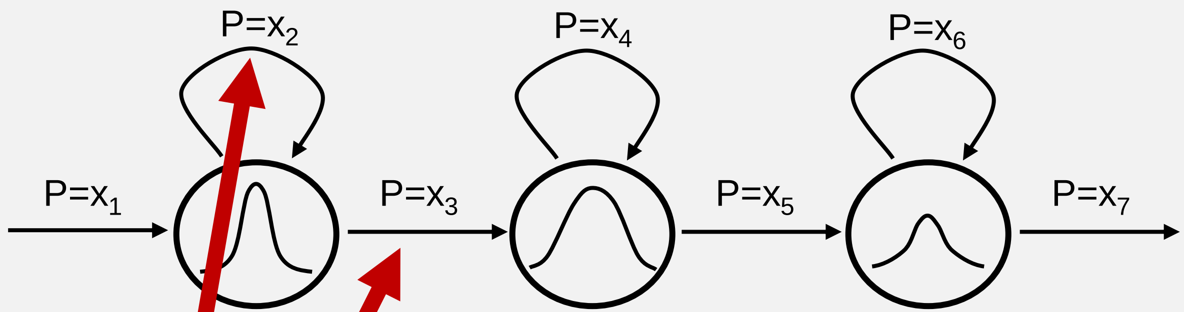
Using a single Gaussian distribution is usually not adequate for modeling the frame distribution.
Instead one may use a weighted sum of Gaussians (a Gaussian mixture, “GMM”), or a DNN.

Hidden Markov Models

- Hidden: The sequence of states which caused the observed sequence of feature vectors is not an observable.
- Markov property: the state of a system at time *t* only depends on the state the system was in at time *t*-1.
- Model: The acoustic properties of a stationary speech segment are described with an acoustic model. Its parameters are emission and transition probabilities.

Hidden Markov Models

A typical phone model consists of three states

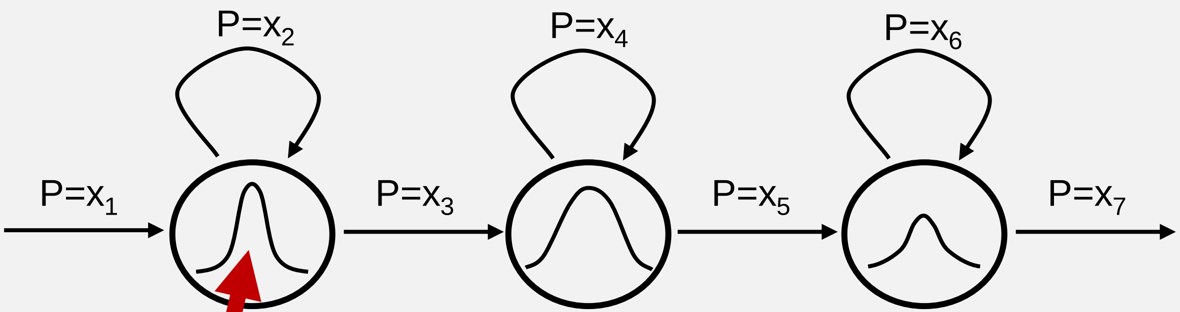


– *States* Each state models a quasi-stationary part of the spectrogram. It is assumed to generate a feature vector every 10 ms which is drawn randomly according to a state-dependent distribution

– *Transitions* from one state to the same or the next (or another future state) Transition probabilities are learned during training

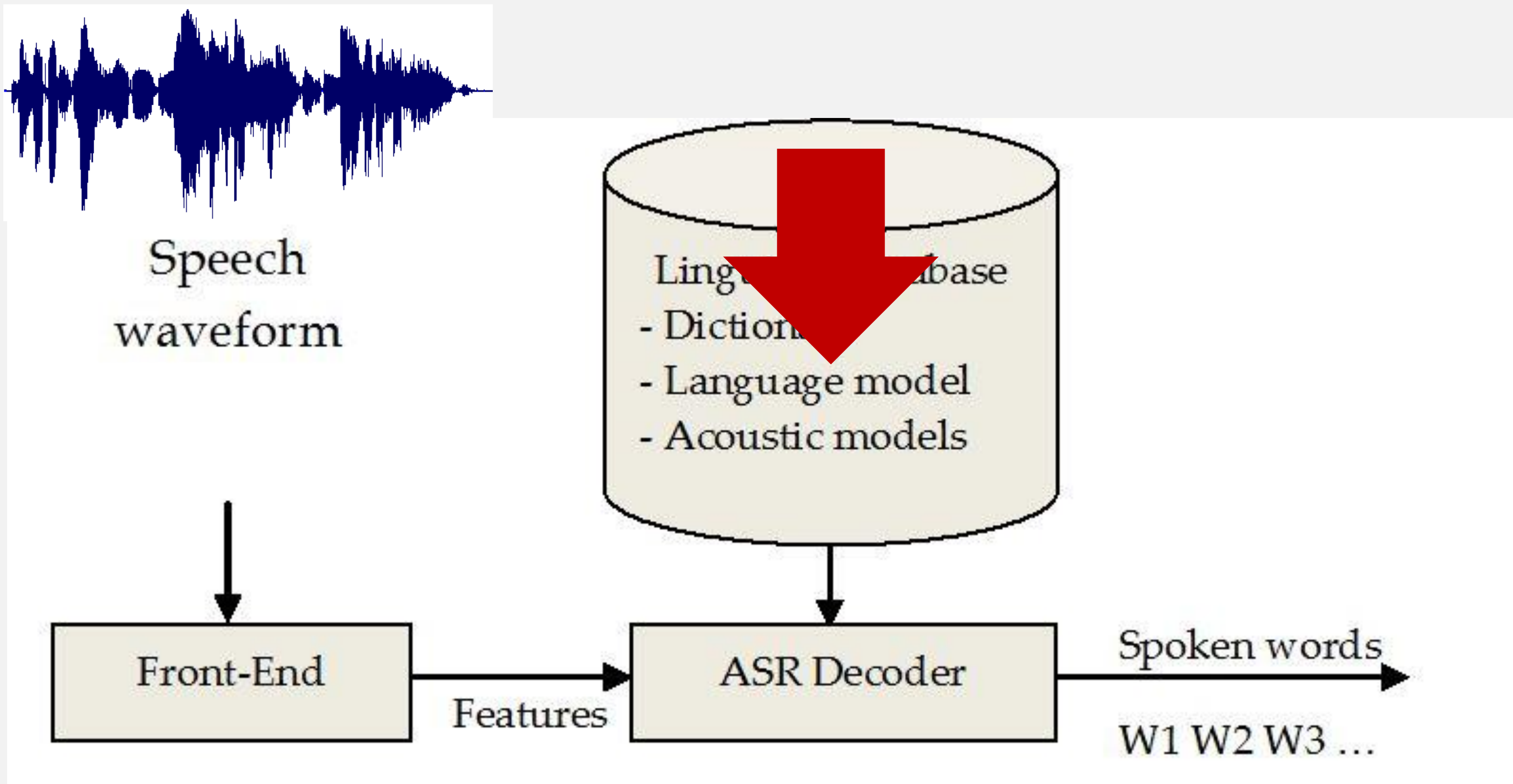
Hidden Markov Models

A typical phone model consists of three states



- *States* Each state models a quasi-stationary part of the spectrogram. It is assumed to generate a feature vector every 10 ms which is drawn randomly according to a state-dependent distribution
- *Transitions* from one state to the same or the next (or another future state) Transition probabilities are learned during training
- *Emission* probabilities

ASR architecture



Language model (LM)

LM = A statistical description of “all” sentences

P(“let’s go to the party”) =

P(“<BOS> let’s go to the party <EOS>”) =

P(“let’s | “<BOS>”) *

P(“go | “<BOS> let’s”) *

P(“to” | ““<BOS> let’s go”) *

P(“the” | “<BOS> let’s go to”) *

P(“party” | “<BOS> let’s go to the”) *

P(“<EOS>” | “<BOS> let’s go to the party”)

Thomas Bayes

Statistical LM in ARPA format

$-{}^{10}\log(p)$ n-gram [back-off]

ARPA = Advanced Research Agency Projects

```
\data\
ngram 1=37445
ngram 2=138797
ngram 3=51837
ngram 4=53201

\1-grams:
-1.727079 A    -1.184703
-5.57354  AACHEN    -0.30103
-5.57354  AAMI -0.30103
-4.833177 AARON     -0.3245111
-5.27251  AARONS    -0.39794
-5.57354  AARRON    -0.30103
…
-0.4887864     INCLUDE A -0.8016323
-1.698039      INCLUDE ART
-1.698428      INCLUDE BODY
…
-0.1107919     ALSO INCLUDE A -0.4771213
-0.1641603     TO INCLUDE A   -1.079181
-0.611348      TO INCLUDE EVERY    -0.69897
…
-0.153755      ALSO INCLUDE A TEN
-0.01548983    TO INCLUDE A NEW
-0.102658      WILL INCLUDE A SHIFT
-0.01771067    IT INCLUDES A STOPOVER
…
\end\
```

Language models

- **Unweighted language models**: a rule based formalism to describe which sequences of words form legal sentences:

  we [want to] go home [tonight]

- **Probabilistic language models**: a formalism that estimates probabilities of word sequences from text:

  P(word | precontext)

  P("tonight" | "to go home")

  Can be a very long list of N-grams

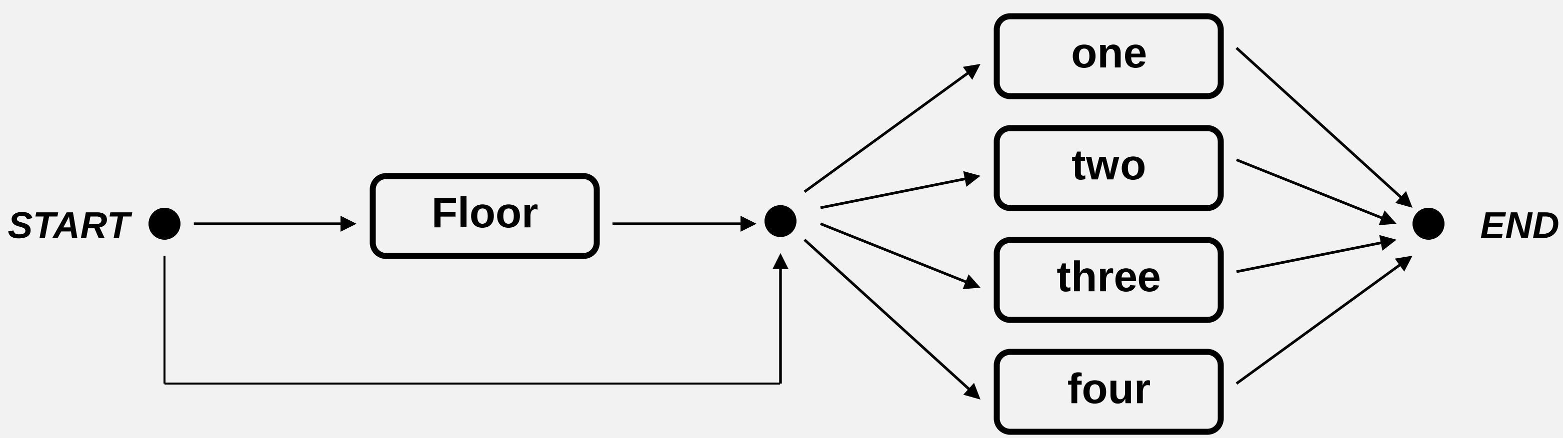
- **Vectorized language models**: e.g., word2vec, in combination with ANNs

  each word is represented as a vector (dim 200-300)

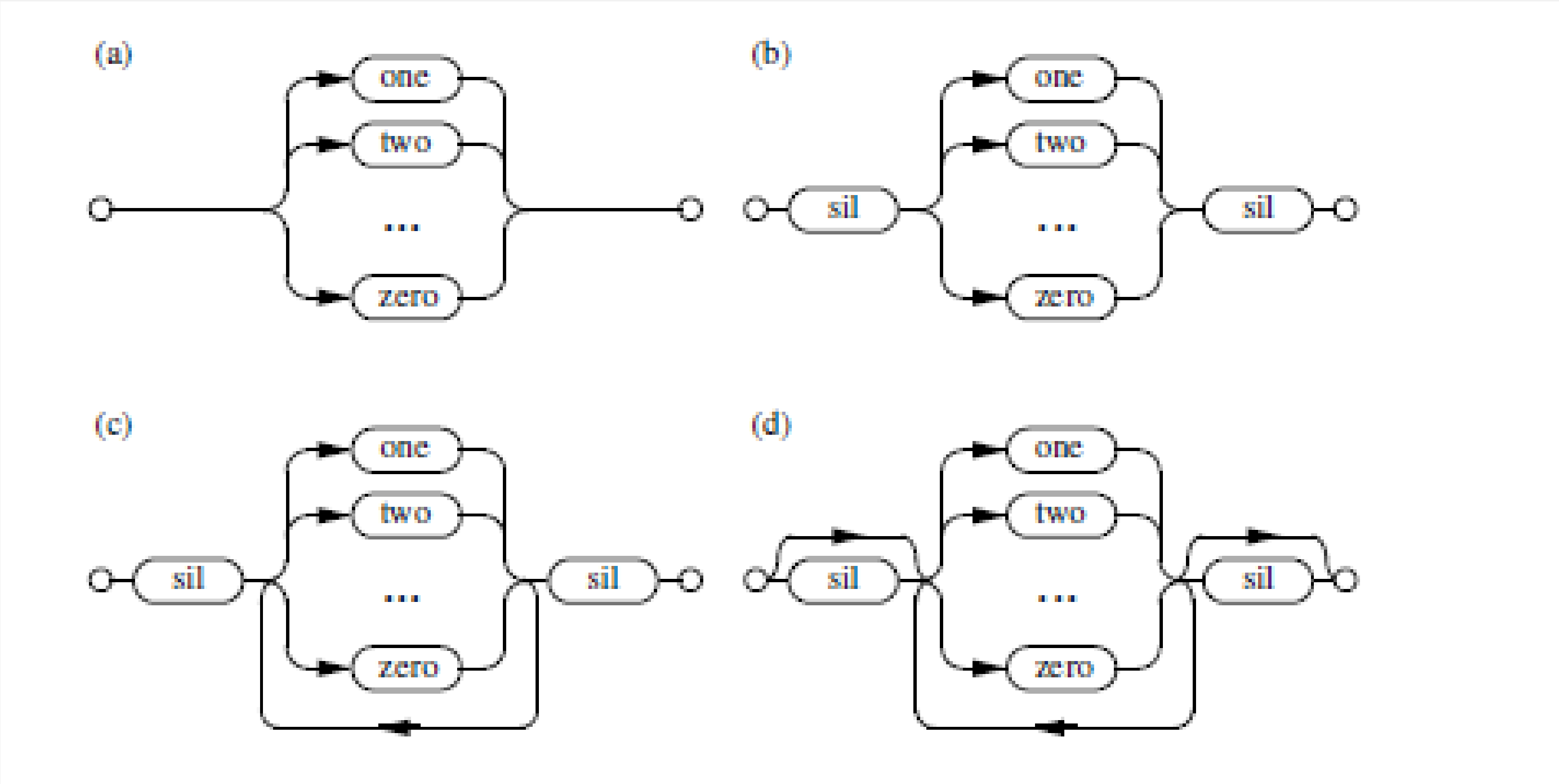
Language models

- Unweighted language models:
  - used for applications using low-end memory
  - often a regular (finite state) grammar (→ regexp)
  - can be constructed by hand for a limited domain

  - Example:



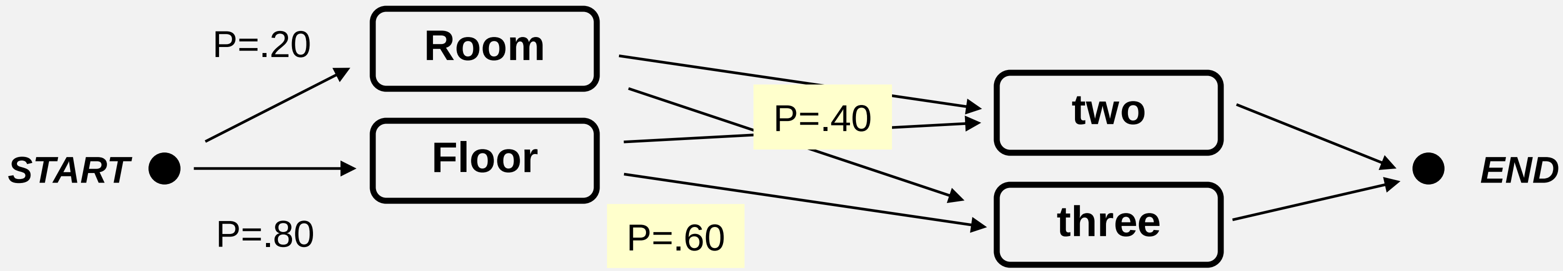
Other regular grammars




```
$digit = one | two | three | four | five |
         six | seven | eight | nine | zero;
(
  sil < $digit > sil
)
```

Probabilistic language models

- Mostly used in tasks where more memory is available.
- Estimated by means of orthografic transcriptions from a text corpus



Probabilistic language model

- models P(current word | precontext)
- in practice often
  - an N-gram $P(w \mid w_1\ w_2\ w_3\ \ldots.\ w_{k-1})$
  - a skipgram $P(w \mid w_1 * w_3)$

Probabilistic language model

P(“let’s go to the party”) =

P(“let’s | “<BOS>”) *

P(“go | “<BOS> let’s”) *

P(“to” | “““<BOS> let’s go”) *

P(“the” | “<BOS> let’s go to”) *

P(“party” | “<BOS> let’s go to the”) *

P(“<EOS>” | “<BOS> let’s go to the party”)

Probabilistic language model

P(“let’s go to the party”) =

P(“let’s | “<BOS>”) *

P(“go | “<BOS> let’s”) *

P(“to” | ““‘<BOS> let’s go”) *

P(“the” | [redacted] let’s go to”) *

P(“party” | “[redacted] go to the”) *

P(“<EOS>” | [redacted] to the party”)

Probabilistic language model

approximation

P(“let’s go to the party”) =

P(“let’s | “<BOS>”) *

P(“go | “<BOS> let’s”) *

P(“to” | ““‘“<BOS> let’s go”) *

P(“the” | [redacted] let’s go to”) *

P(“party” | “[redacted] go to the”) *

P(“<EOS>” | [redacted] to the party”)

by cutting down to 4-grams

Unobserved word sequences

- Not all possible sequences of words may have occurred in the training material.
- Assigning unobserved sequences a probability equal to 0.0 will prohibit recognition of these sequences in a test.
- Therefore: assign these unobserved sequences a small probability by a clever estimate based on the text data in the training corpus

Unobserved word sequences

Ways to assign probabilities to non-observed word sequences:

- Smoothing: assigning the non-observed sequences (events) a small probability.
- Discounting: Estimate frequency of unobserved trigram by using the frequency of observed bigram of last two words. (Unobserved 4grams → 3grams, unobserved 3grams → bigrams, etc.)

- Clever methods available: e.g. “Kneser-Ney”, “Good-Turing”.

bigram, ARPA file

```
\data\
ngram 1=<num 1-grams>
ngram 2=<num 2-ngrams>

\1-grams:
P(!ENTER)       !ENTER  B(!ENTER)
P(W1)            W1     B(W1)
P(W2)            W2     B(W2)
...
P(!EXIT)        !EXIT   B(!EXIT)

\2-grams:
P(W1 | !ENTER)   !ENTER W1
P(W2 | !ENTER)   !ENTER W2
P(W1 | W1)       W1     W1
P(W2 | W1)       W1     W2
P(W1 | W2)       W2     W1
....
P(!EXIT | W1)    W1     !EXIT
P(!EXIT | W2)    W2     !EXIT
\end\
```

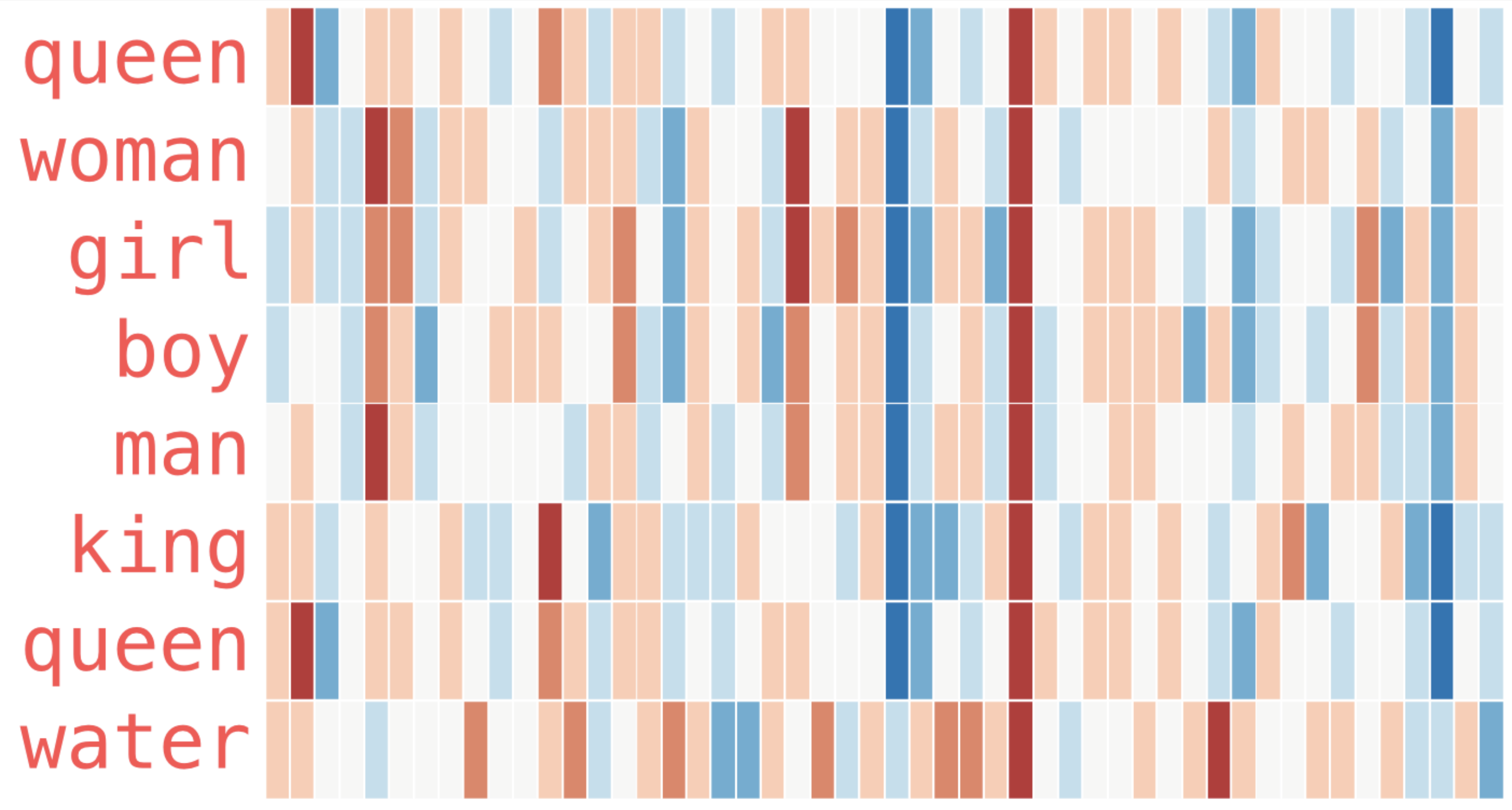
How to create an LM for ASR

See for a detailed discussion
https://www.sciencedirect.com/topics/computer-science/language-modeling

There are many python tools to create LMs - e.g.
https://medium.com/analytics-vidhya/a-comprehensive-guide-to-build-your-own-language-model-in-python-5141b3917d6d

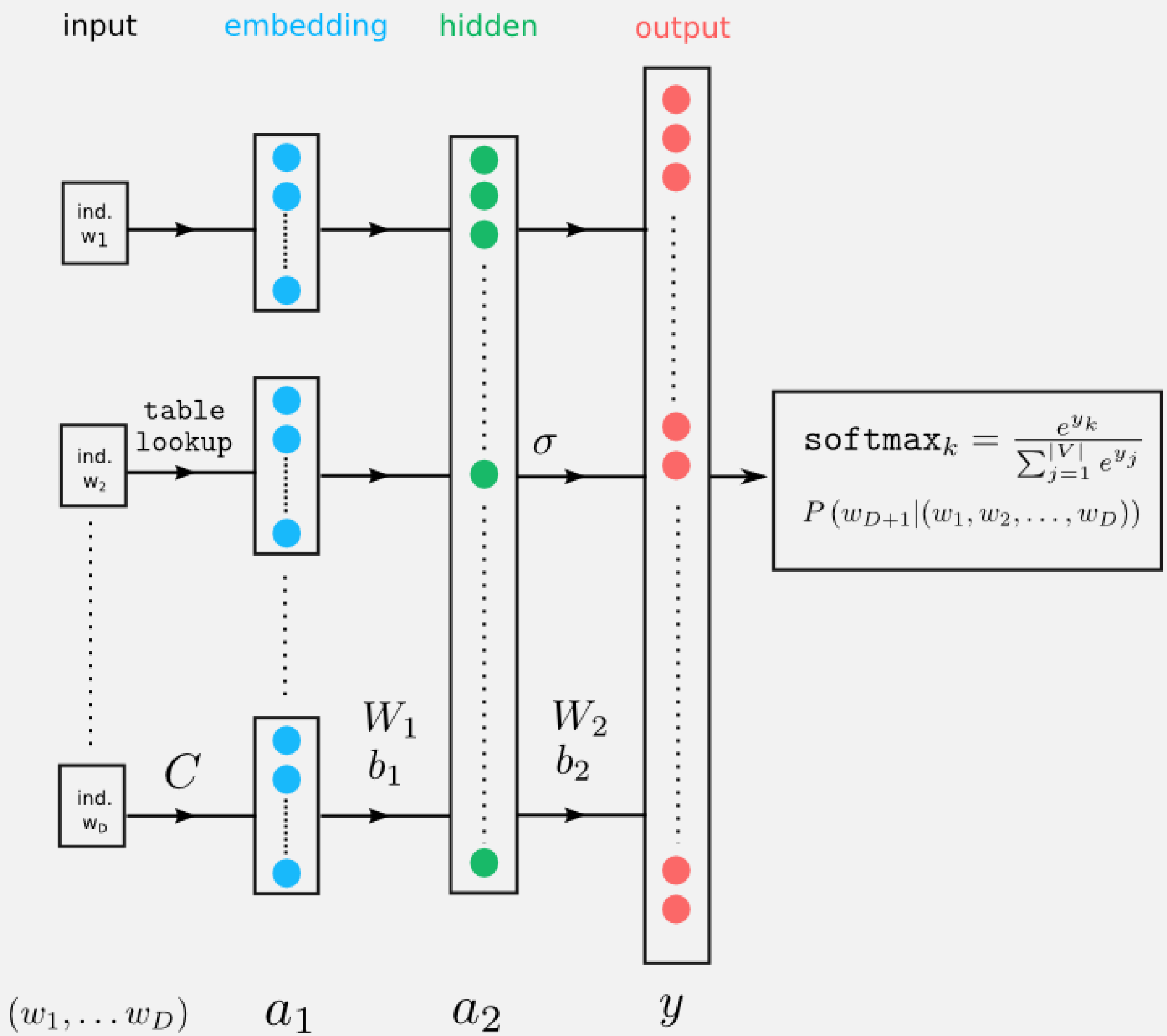
Steps	Examples
Raw	<html ccdes> Between A, B and C, there is a 3-way interaction, Mr. Bilmes reported.</>
Clean up	Between A, B and C, there is a 3-way interaction, Mr. Bilmes reported.
Tokenisation	between A B and C here is a 3 way interaction Mr. Bilmes reported
Rewrites	between A B and C there is a three way interaction mister Bilmes reported
LM-building	output: arpa file, NN-LM

Word embeddings, word2vec

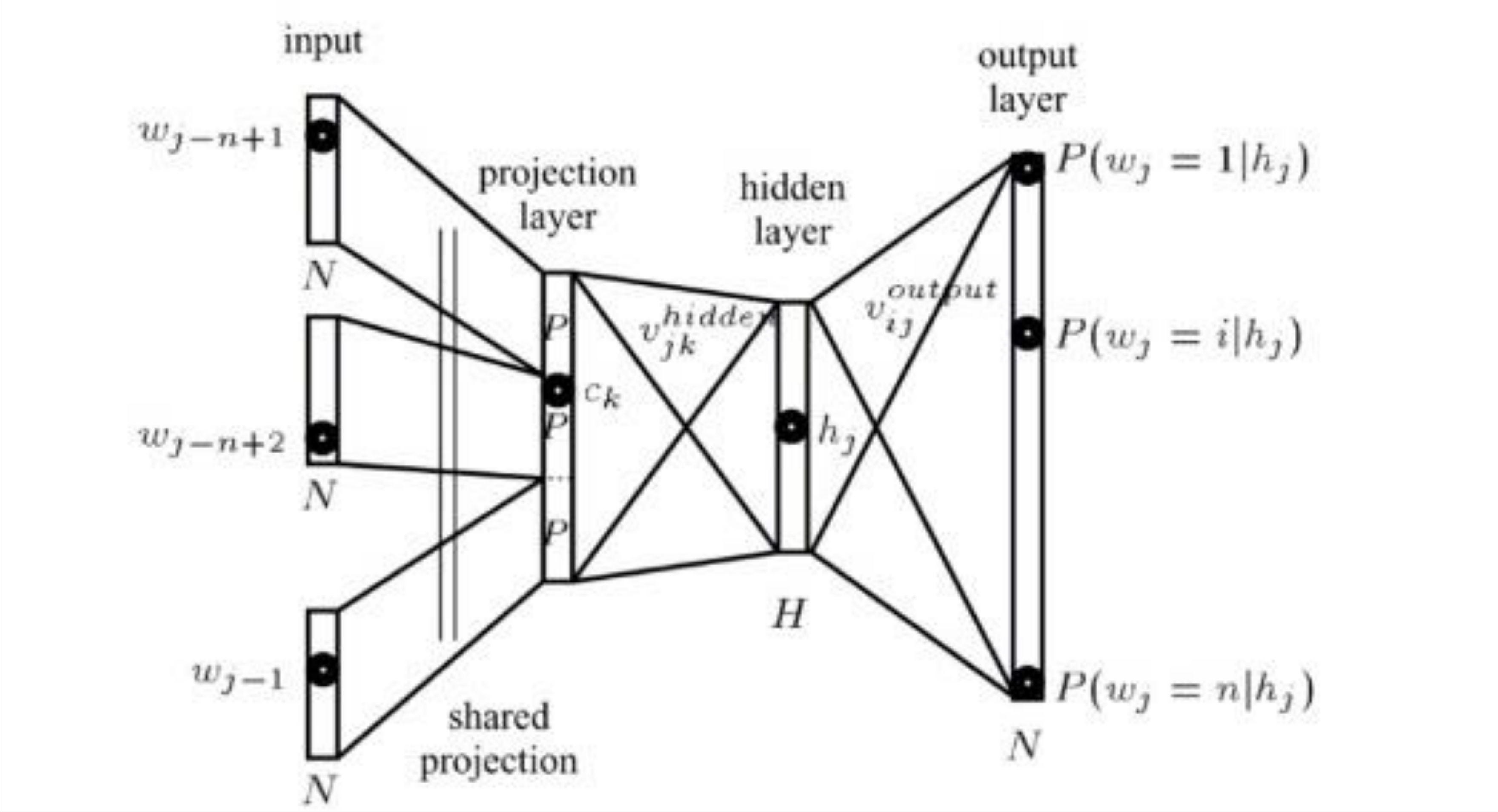


From: lajammar.github.io (retrieved 2020)

NN language model



(R)NN-LM



input
w_{j-n+1}
N
w_{j-n+2}
N
w_{j-1}
N
shared
projection
projection
layer
P
c_k
P
P
hidden
layer
v_{jk}^{hidden}
h_j
H
v_{ij}^{output}
output
layer
$P(w_j = 1|h_j)$
$P(w_j = i|h_j)$
$P(w_j = n|h_j)$
N

Use of data

- Test data = training data
  - Weakness: very poor generalization power of model
  - Sometimes useful
- Separate training and test data (held out data)
  - Already better. Test on new unseen data.

Better use of data

- training of the recognizer using a training set
- "tuning" of the recognizer using a development set
- testing of the recognizer using a held-out test set

no overlap
in data sets

- more advanced schemes:
  - Rotating schemes, N-fold cross validation schemes
  - Adversarial training and test methods

Underfitting, overfitting, regularization

- An ASR system contains many parameters (usually millions)
- Models with lots of parameters can easily overfit to training data
- This overfitting might hamper generalization (the quality of a model on new, unseen data)
- Remedy: Regularization, by e.g. simplicity, sparsity, dropout, early stopping, Akaike Information Criterion (AIC), manifolds, etc.

Evaluation of ASR quality

Usually based on a reference transcription

Expressed in terms of insertions, deletions, and substitutions

Example:

```
ASR       :       glove read    in         books
reference:     he loves reading       good books
-----------------------------------------------
               del subs subs    ins  del  corr
```

$$WER = \frac{\#del + \#sub + \#ins}{N} \times 100\%$$

$$Wacc = \frac{\#corr - \#del - \#sub}{N} \times 100\%$$

N = number of words in reference

Automatic phonetic transcription?

Three types of errors (at the phoneme level):

phonetic transcription: A t b o v @ n d @ n

reference transcription*: z A t l o v @ n d @

1 2 3

1. Deletion
2. Substitution
3. Insertion

Phone Error Rate (PER)

$$PER = \frac{\text{dels} + \text{subs} + \text{ins}}{N} \times 100\%$$

N = total number of phones in reference transcription

Many variants possible

phonetic transcription: A t b o v @ n d @ n

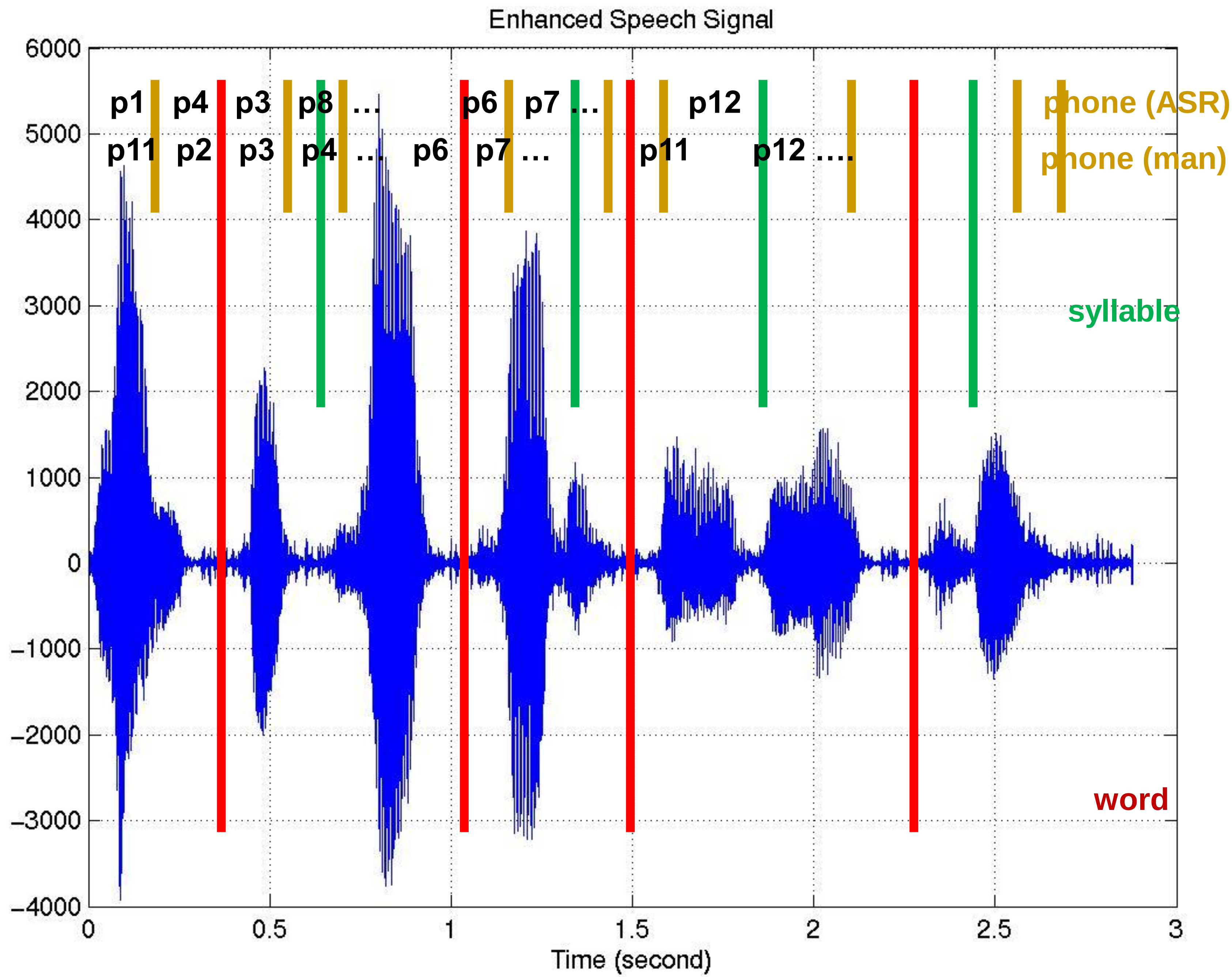
reference transcription: z A t l o v @ n d @

- Option 1: consider hypothesis and reference sequences without taking into account the segmentation over time → Levenshtein
- Option 2: take time duration into account and construct a phone-phone confusion matrix frame by frame

Is WER a good measure?

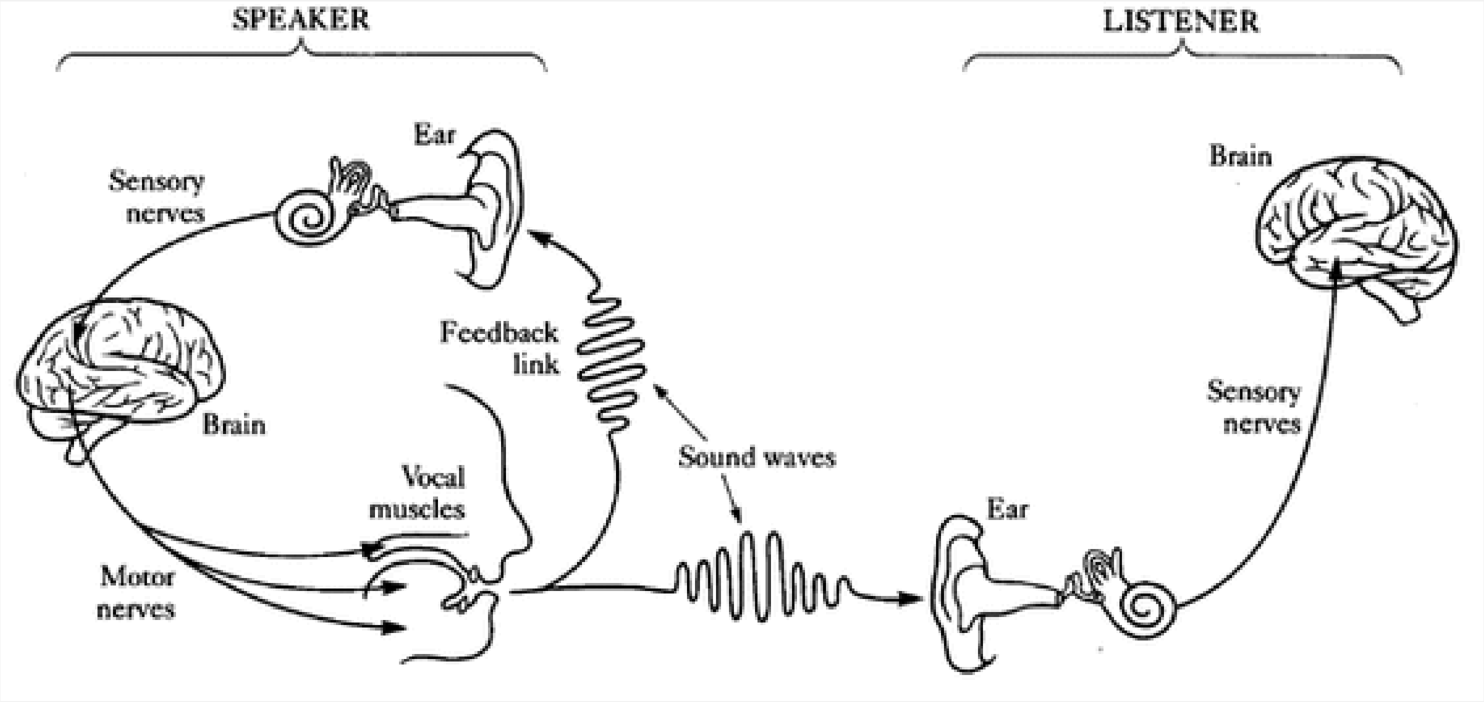
- Usability of WER depends on the task and the performance
- For some applications, 70% word accuracy is OK; for other ones 98% is just not enough

- WER still decreases (on average)
- ASR might soon become supra human

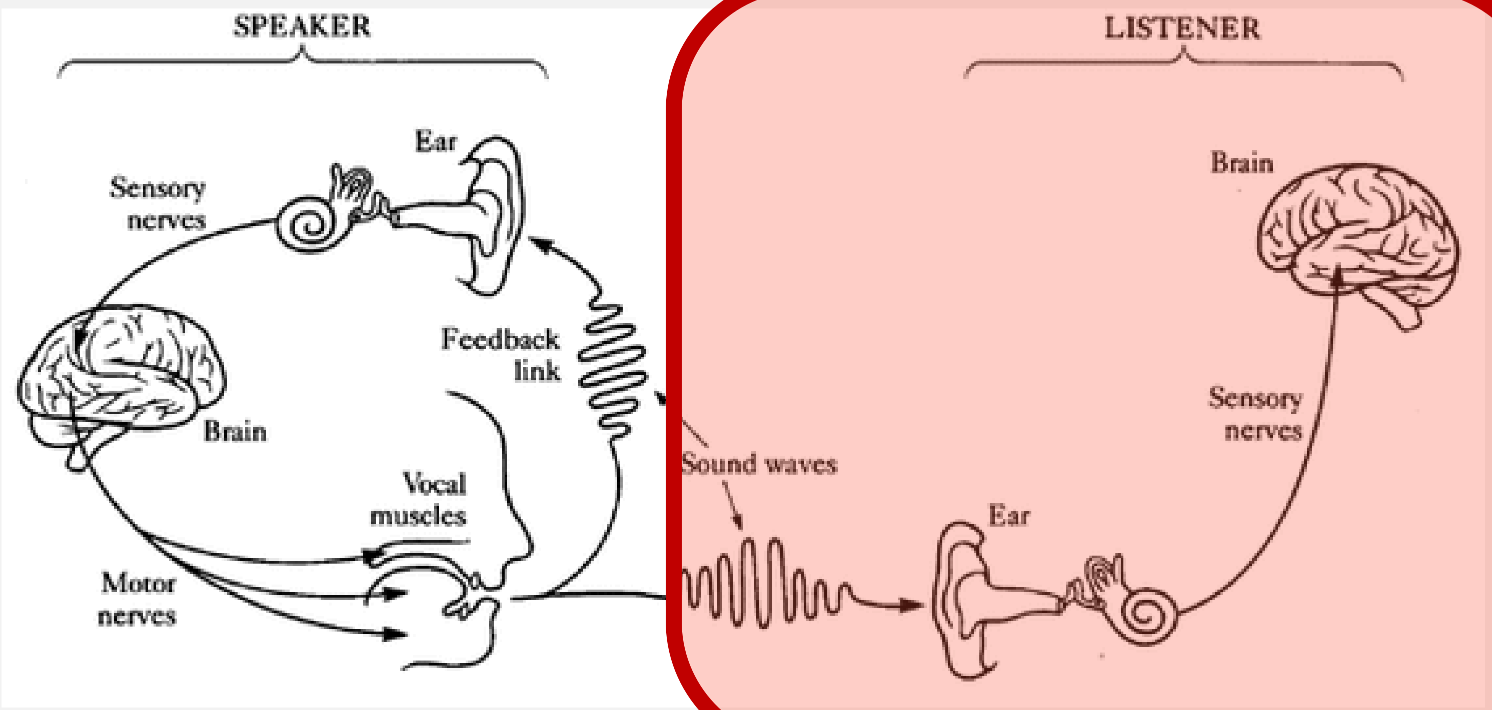


Enhanced Speech Signal
p1
p4
p3
p8
…
p6
p7
…
p12
p11
p2
p3
p4
…
p6
p7
…
p11
p12
….
phone (ASR)
phone (man)
syllable
word
6000
5000
4000
3000
2000
1000
0
−1000
−2000
−3000
−4000
0
0.5
1
1.5
2
2.5
3
Time (second)

What about humans?



Speaker-listener loop

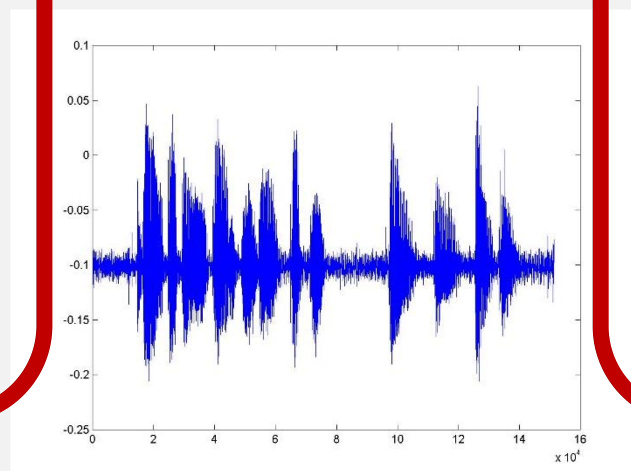

SPEAKER
LISTENER
Ear
Sensory nerves
Brain
Feedback link
Vocal muscles
Motor nerves
Sound waves
Brain
Sensory nerves
Ear

speaker

Concept

Grammar
Mental lexicon

Muscle activation, articulation



listener

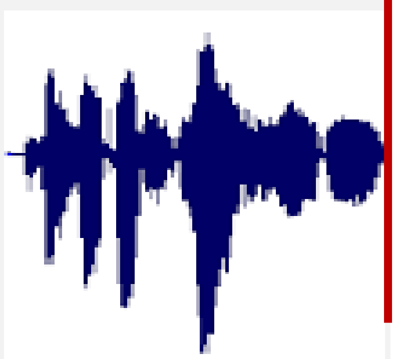
Concept

Grammar
Mental lexicon

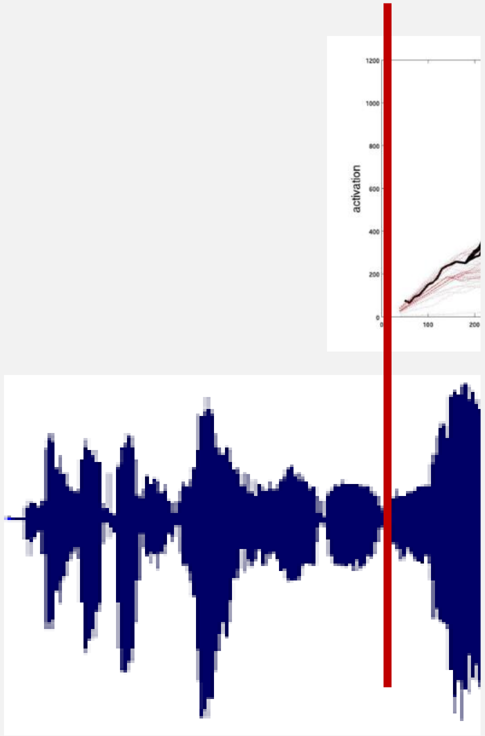
Auditory pathway

Comprehension of continuous speech

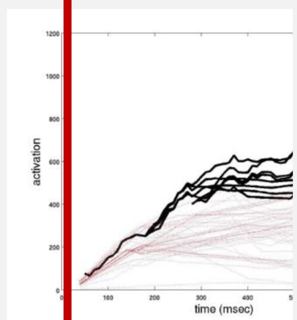
← **past** **future** →

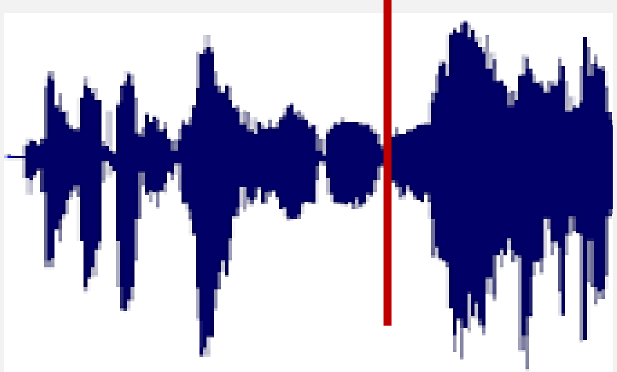


Comprehension of continuous speech

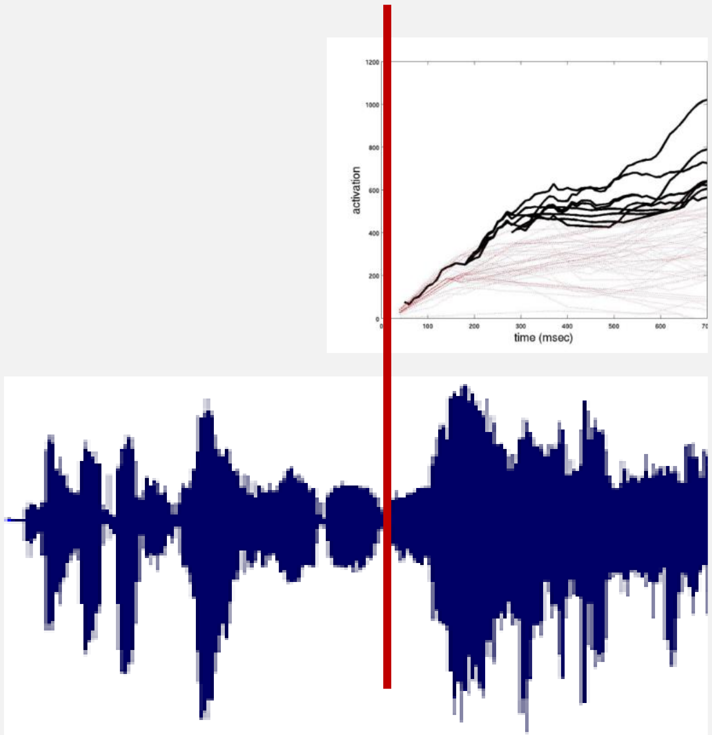


Comprehension of continuous speech

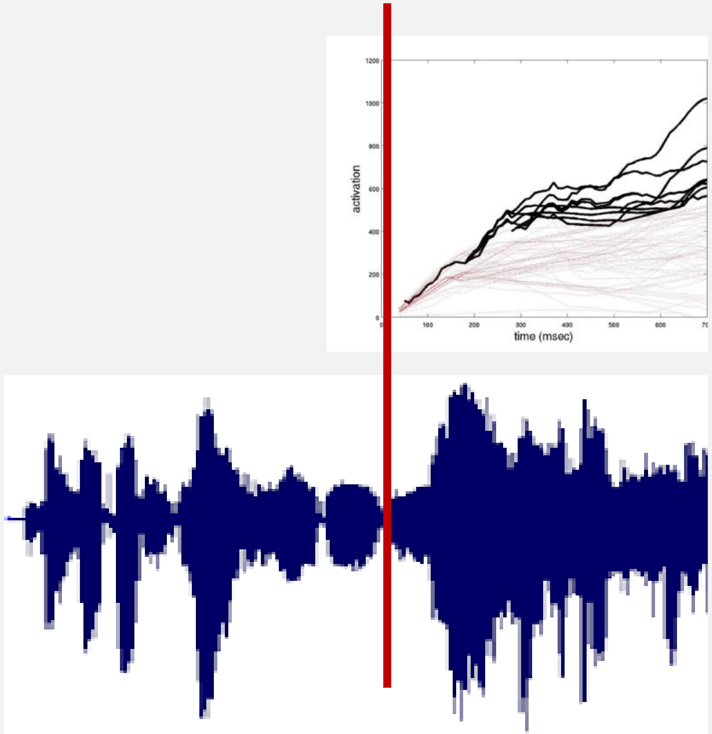




Comprehension of continuous speech

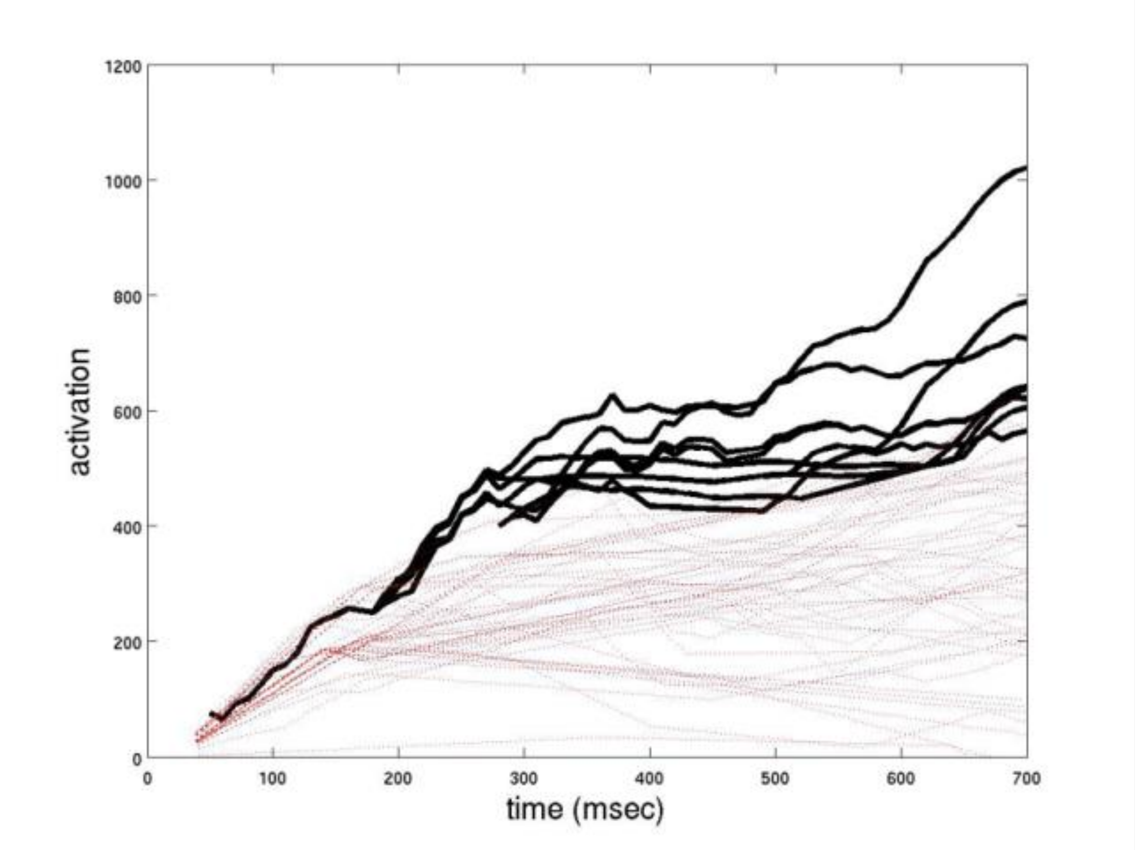


Comprehension of continuous speech



!

Activation

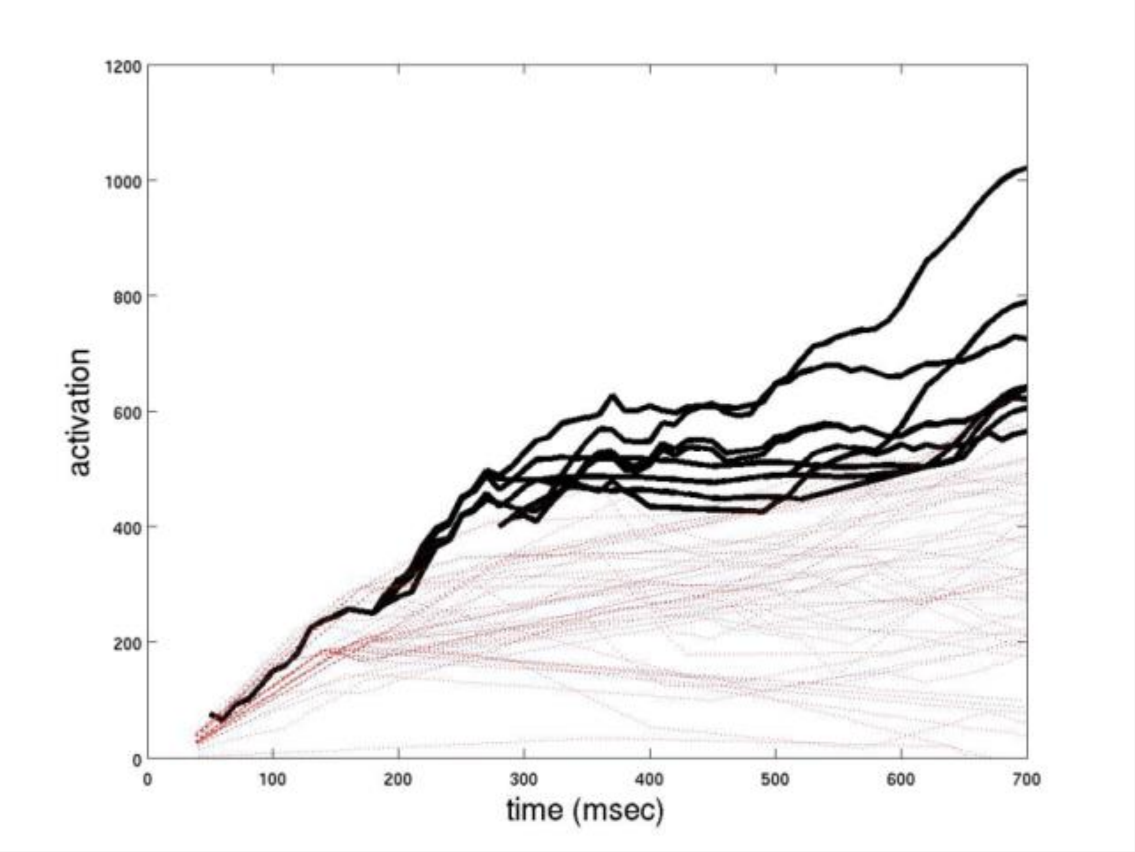


Words compete in the listener's head

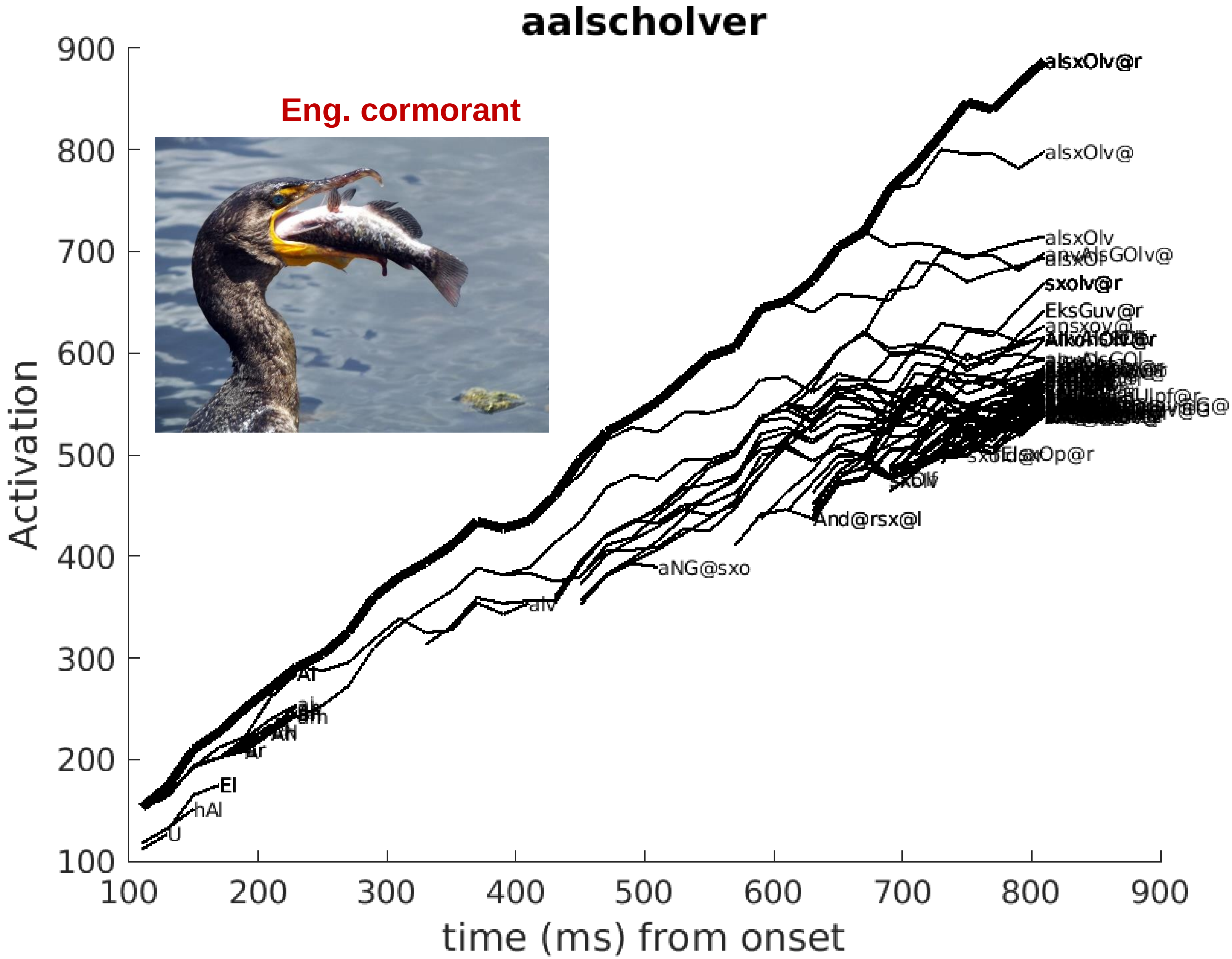
You have to select a winner 3 times a second

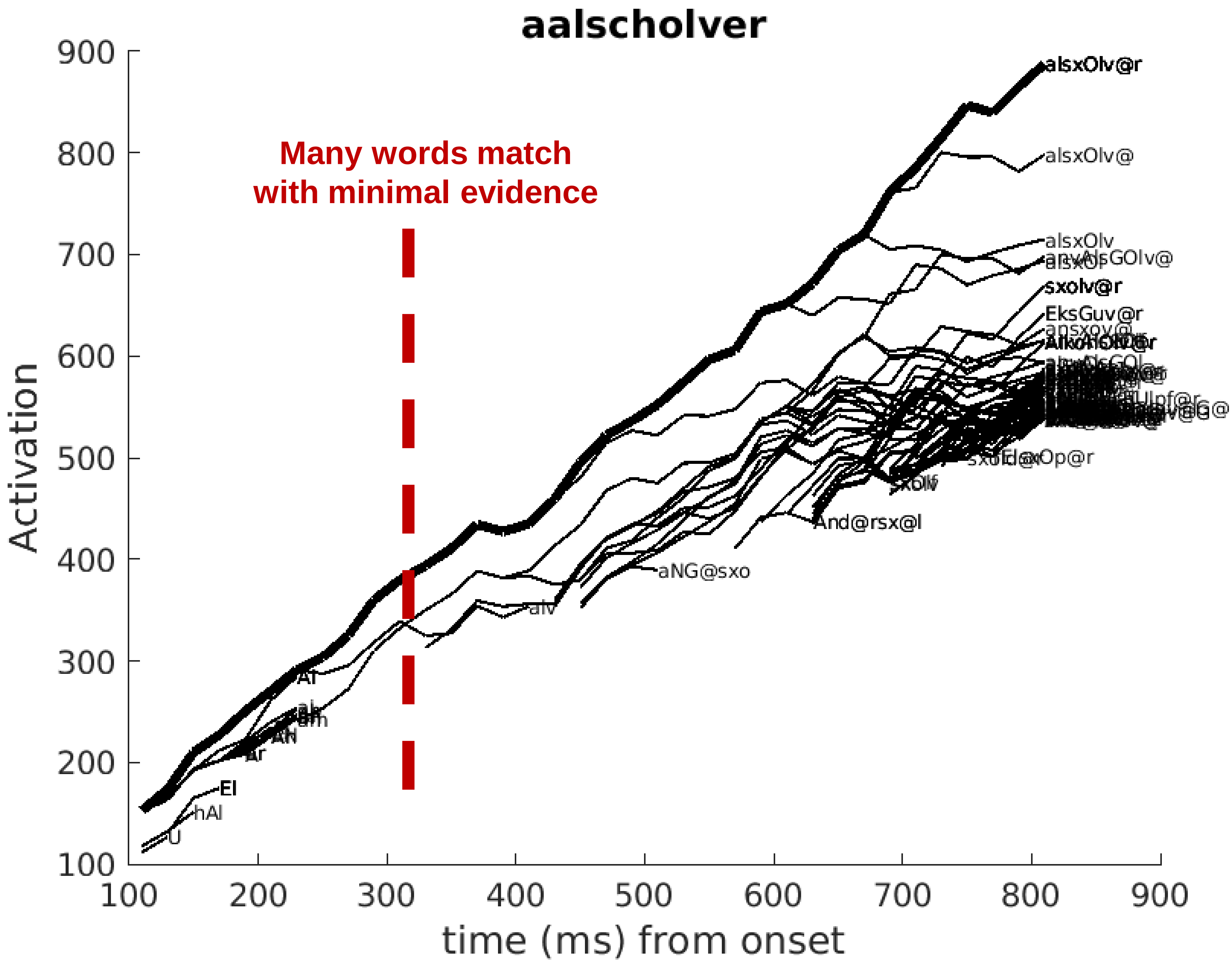
±60000 (±100) options per word

Activation

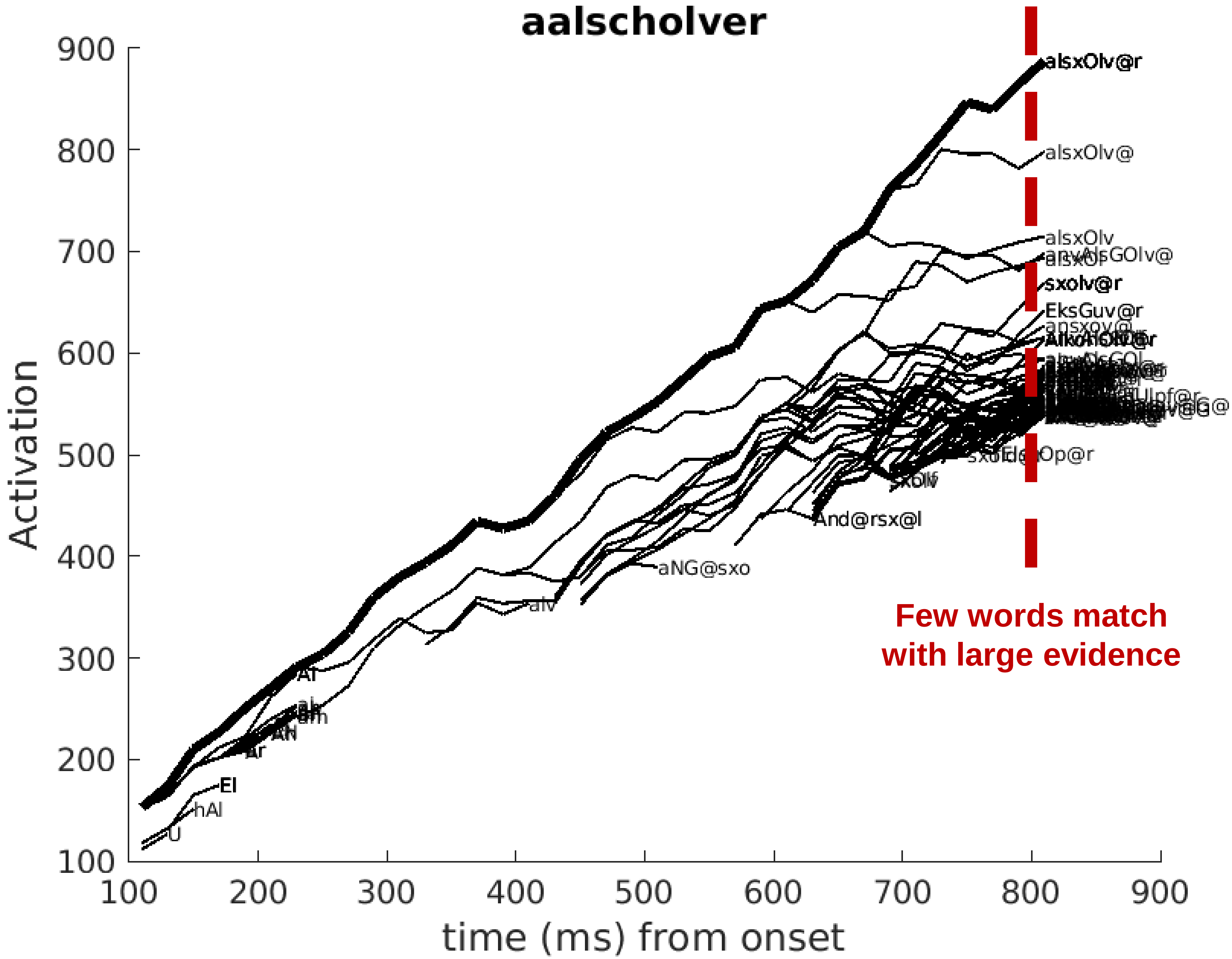


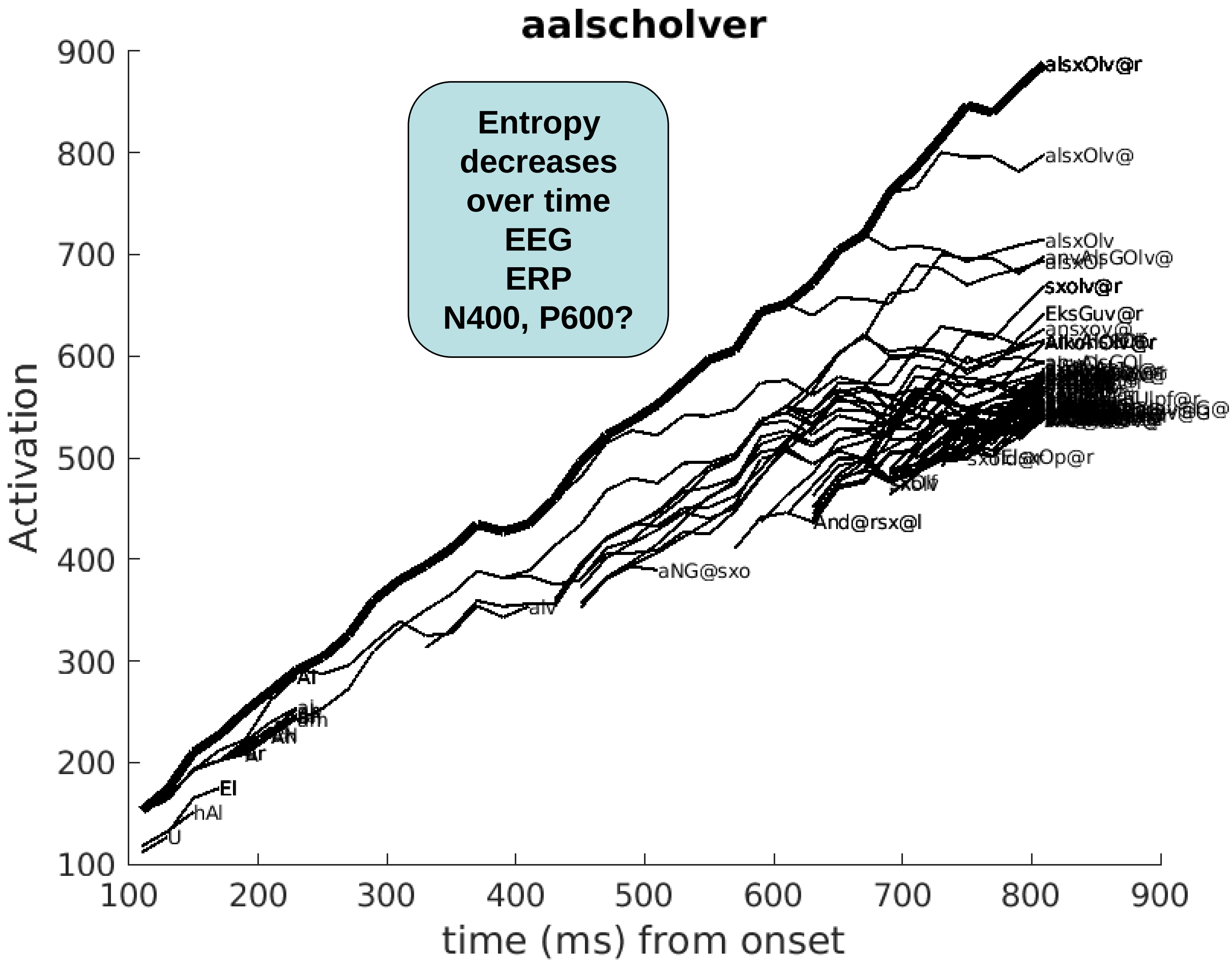
Activation = log(P(signal|word)) + λ log P(word)
Example here: duration 0.68 sec.
Shown: top 200 candidates, recomputed each 10 ms


aalscholver
Eng. cormorant
Activation
time (ms) from onset
alsxOlv@r
alsxOlv@
alsxOlv
sxolv@r
EksGuv@r
And@rsx@l
aNG@sxo
alv
AI
EI
hAl
U
Ar
100
200
300
400
500
600
700
800
900



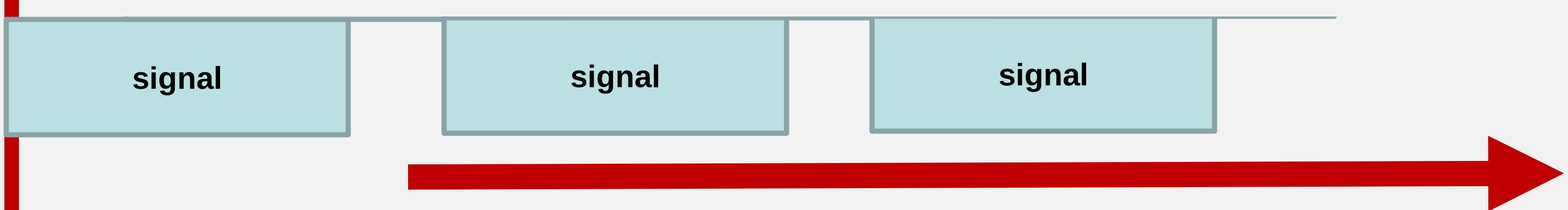
aalscholver
Many words match
with minimal evidence
Activation
time (ms) from onset
900
800
700
600
500
400
300
200
100
100
200
300
400
500
600
700
800
900
alsxOlv@r
alsxOlv@
alsxOlv
sxolv@r
EksGuv@r
And@rsx@l
aNG@sxo
alv
Al
Ar
El
hAl
U


aalscholver
Activation
time (ms) from onset
alsxOlv@r
alsxOlv@
alsxOlv
sxolv@r
EksGuv@r
And@rsx@l
aNG@sxo
alv
Al
El
hAl
U
Few words match
with large evidence

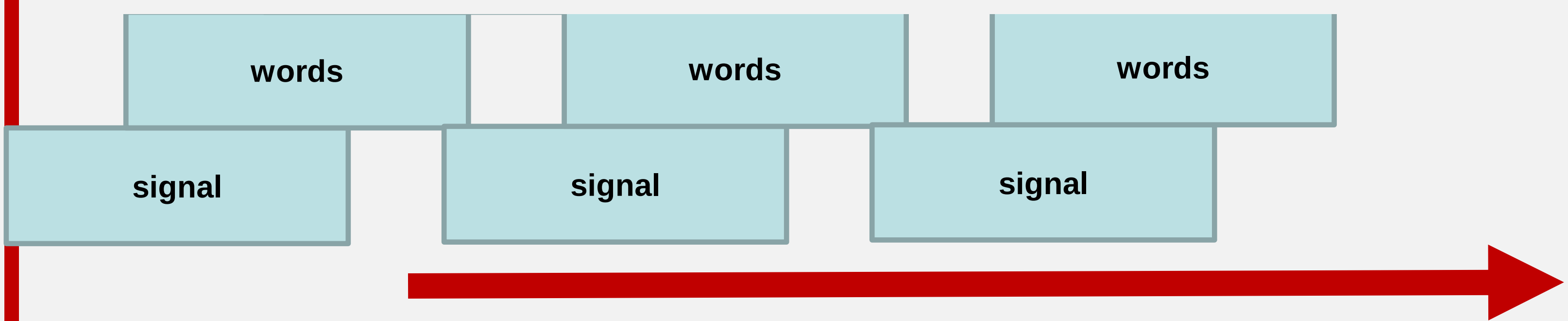


aalscholver
Entropy
decreases
over time
EEG
ERP
N400, P600?
Activation
time (ms) from onset
alsxOlv@r
alsxOlv@
alsxOlv
sxolv@r
EksGuv@r
And@rsx@l
aNG@sxo
alv
Al
EI
hAl
U

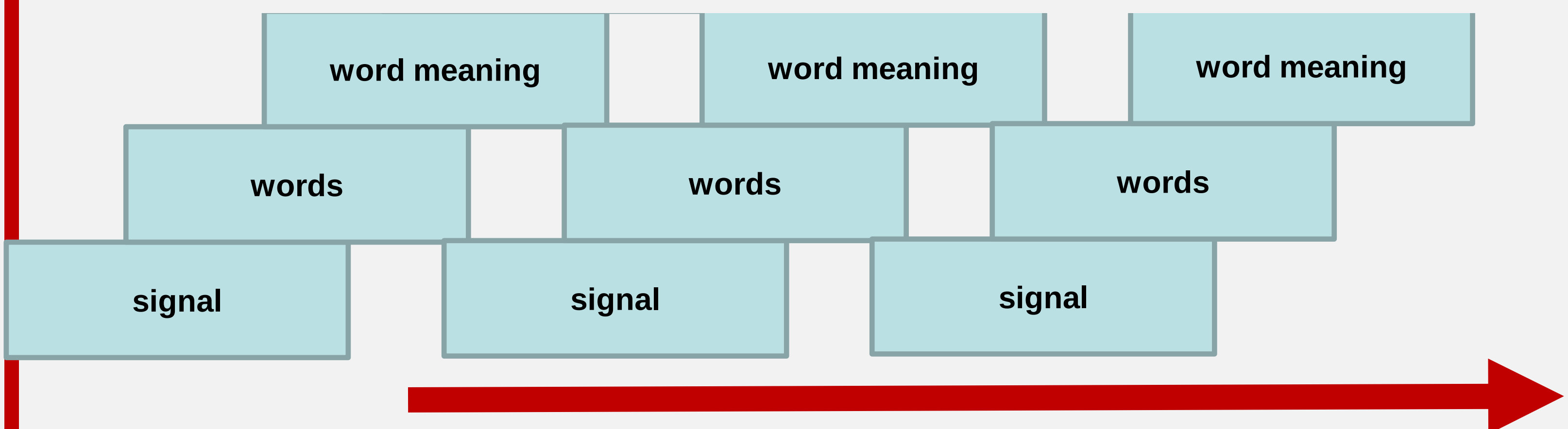
Speech signal, words, meaning



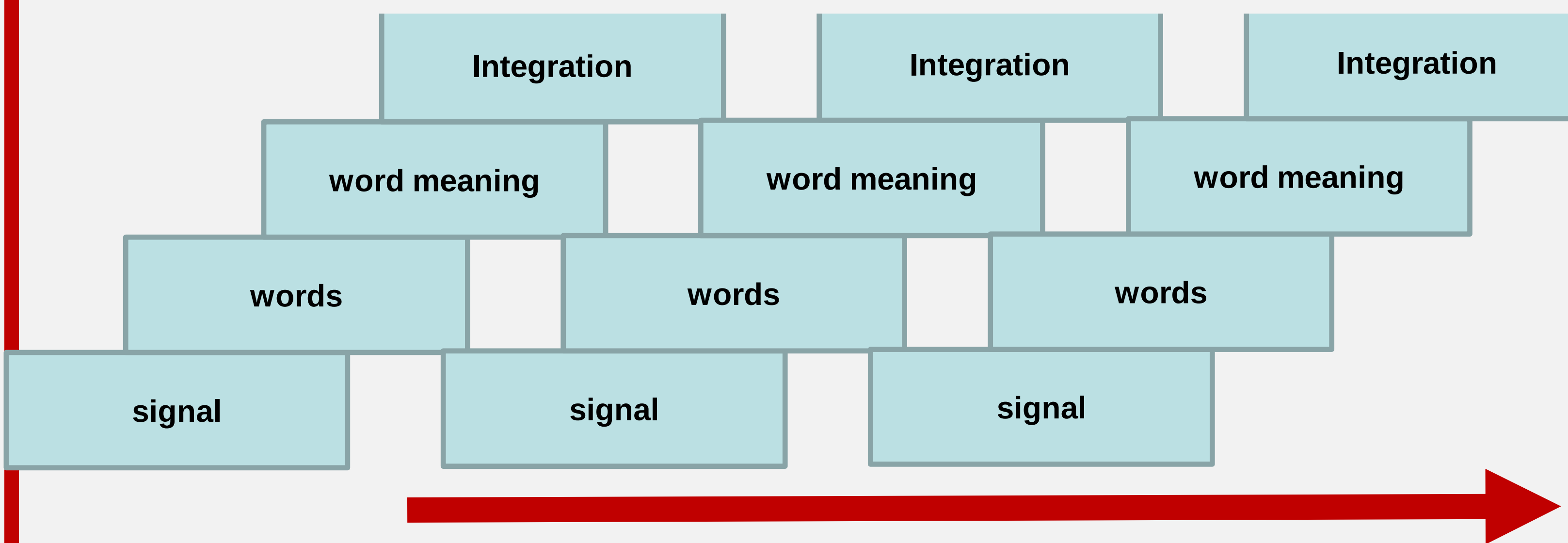
Speech signal, words, meaning



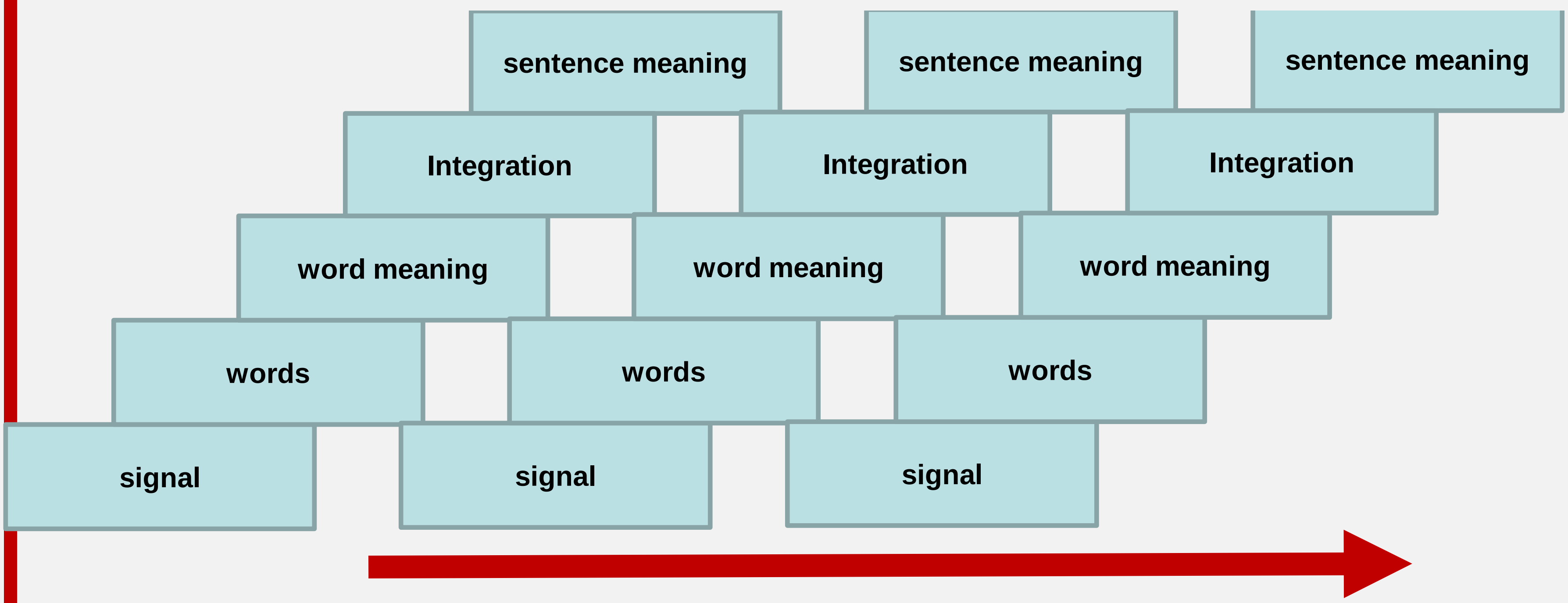
Speech signal, words, meaning



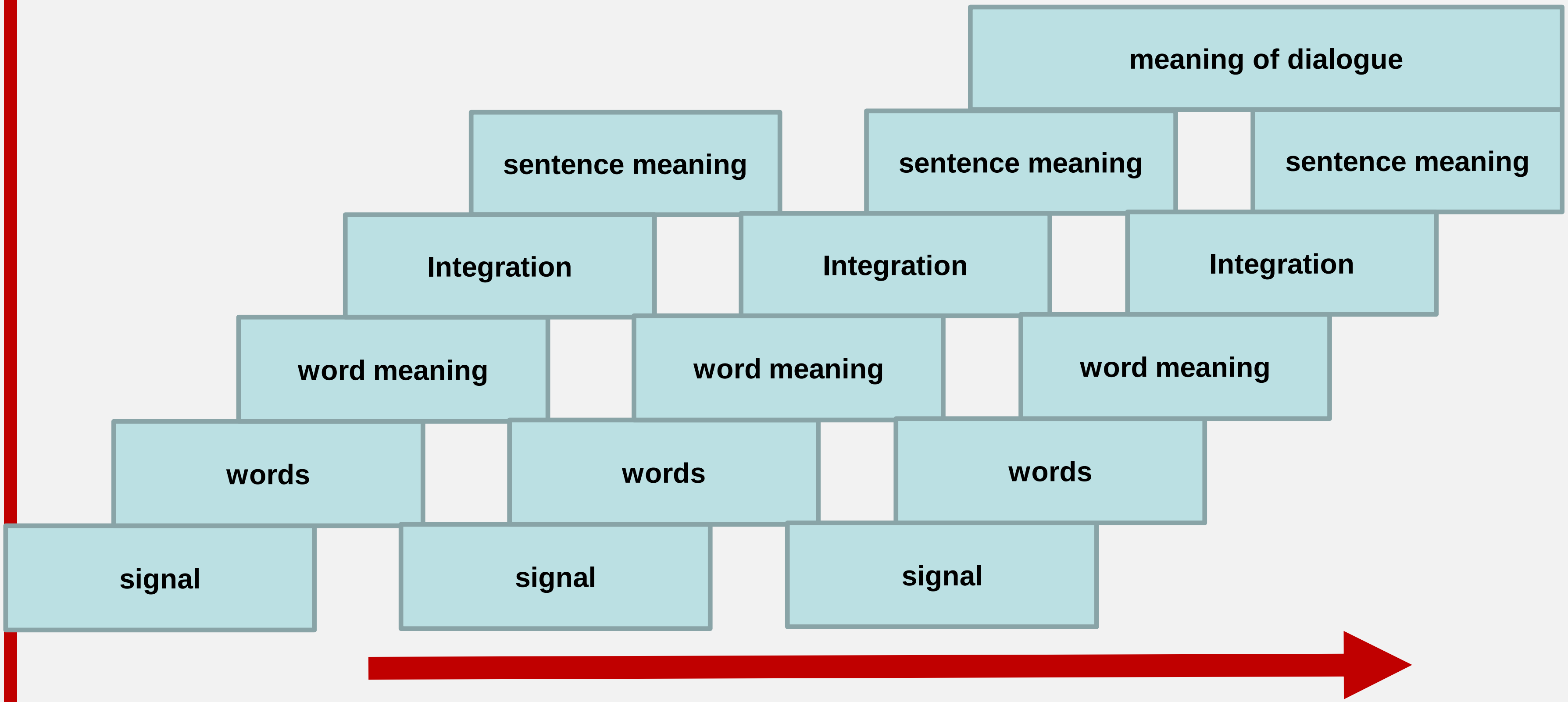
Speech signal, words, meaning



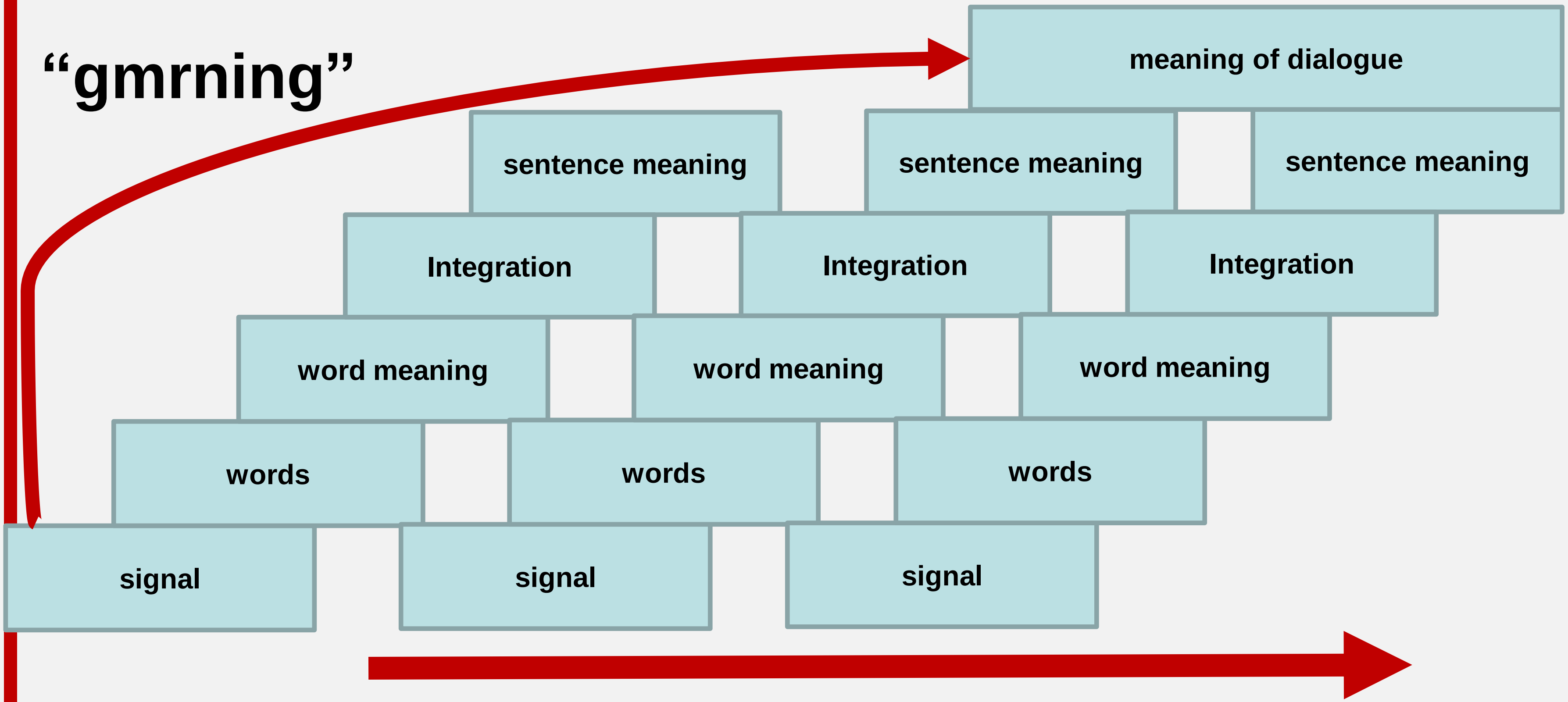
Speech signal, words, meaning



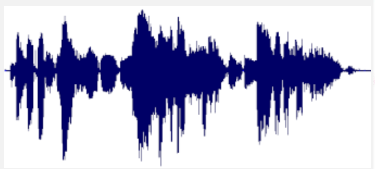
Speech signal, words, meaning

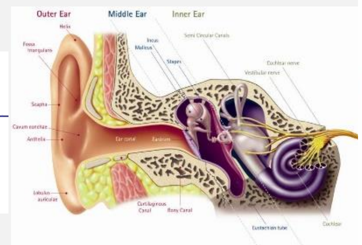


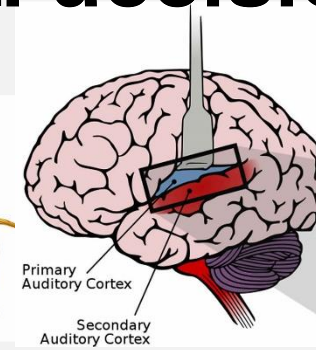
Speech signal, words, meaning



Psycholinguistic experiment (e.g., lexical decision)



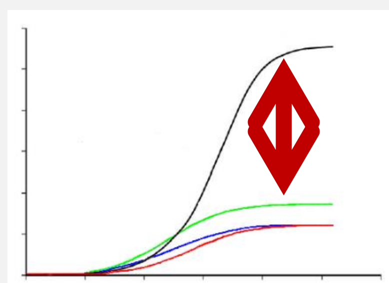




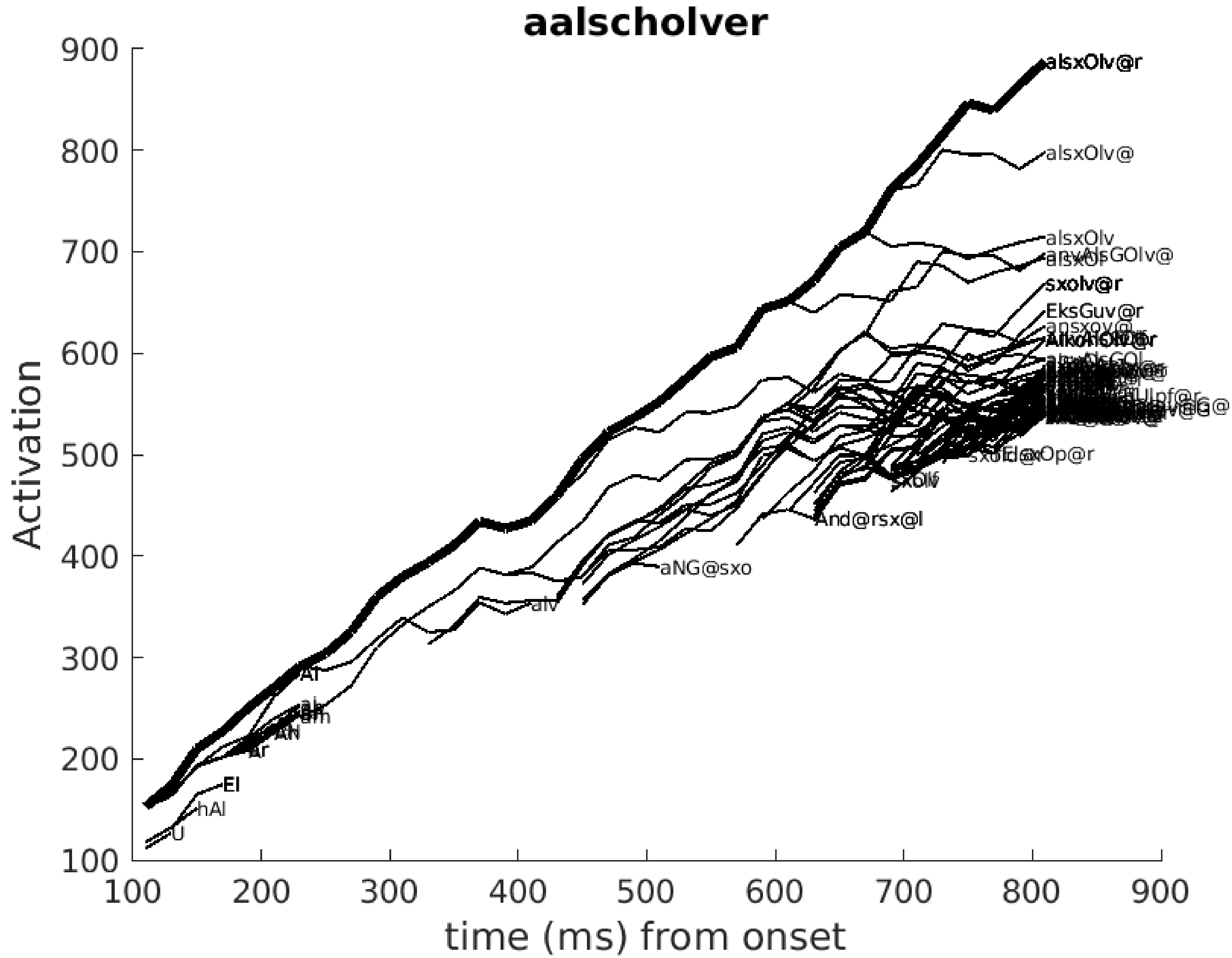
word activation, word competition

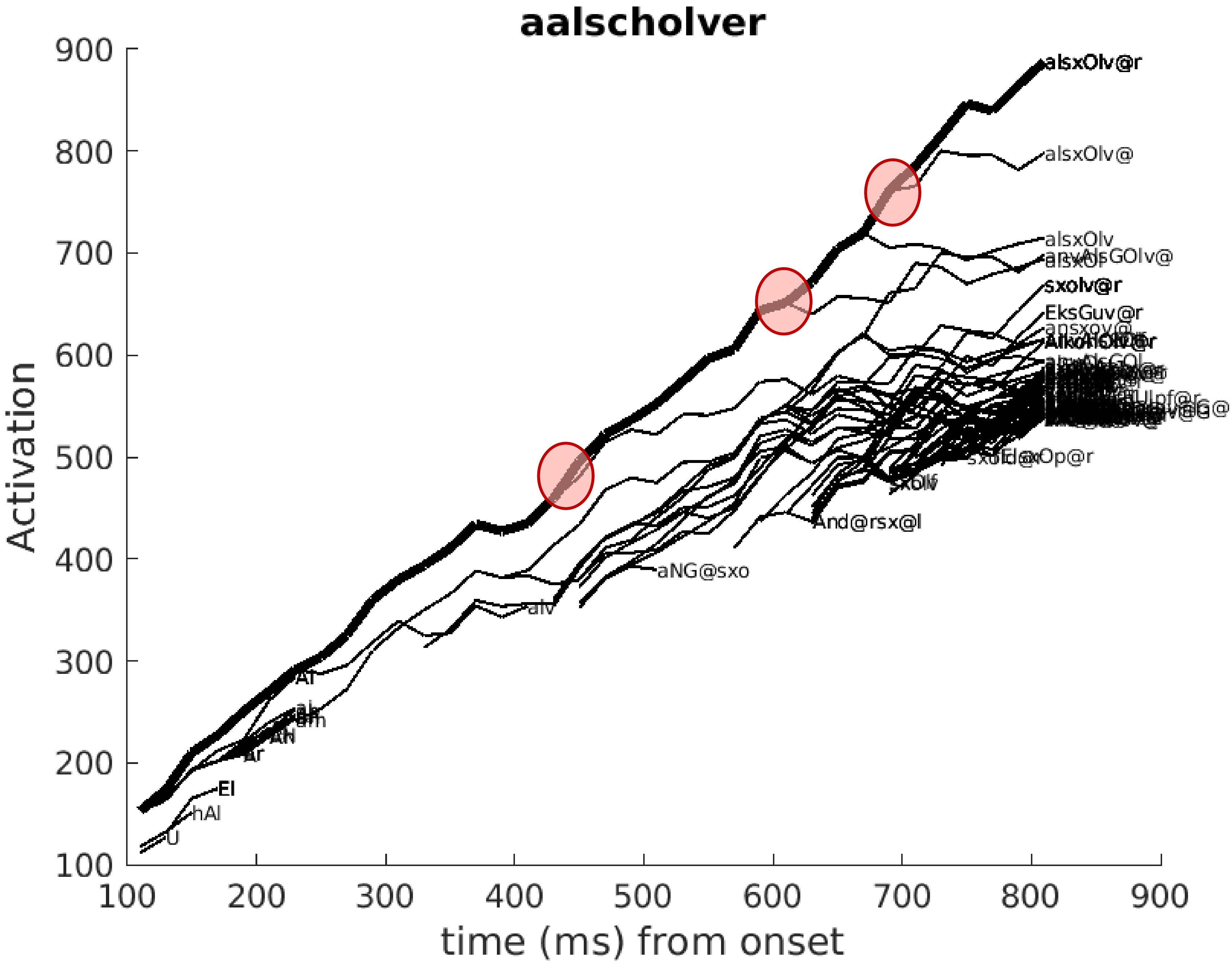
decision

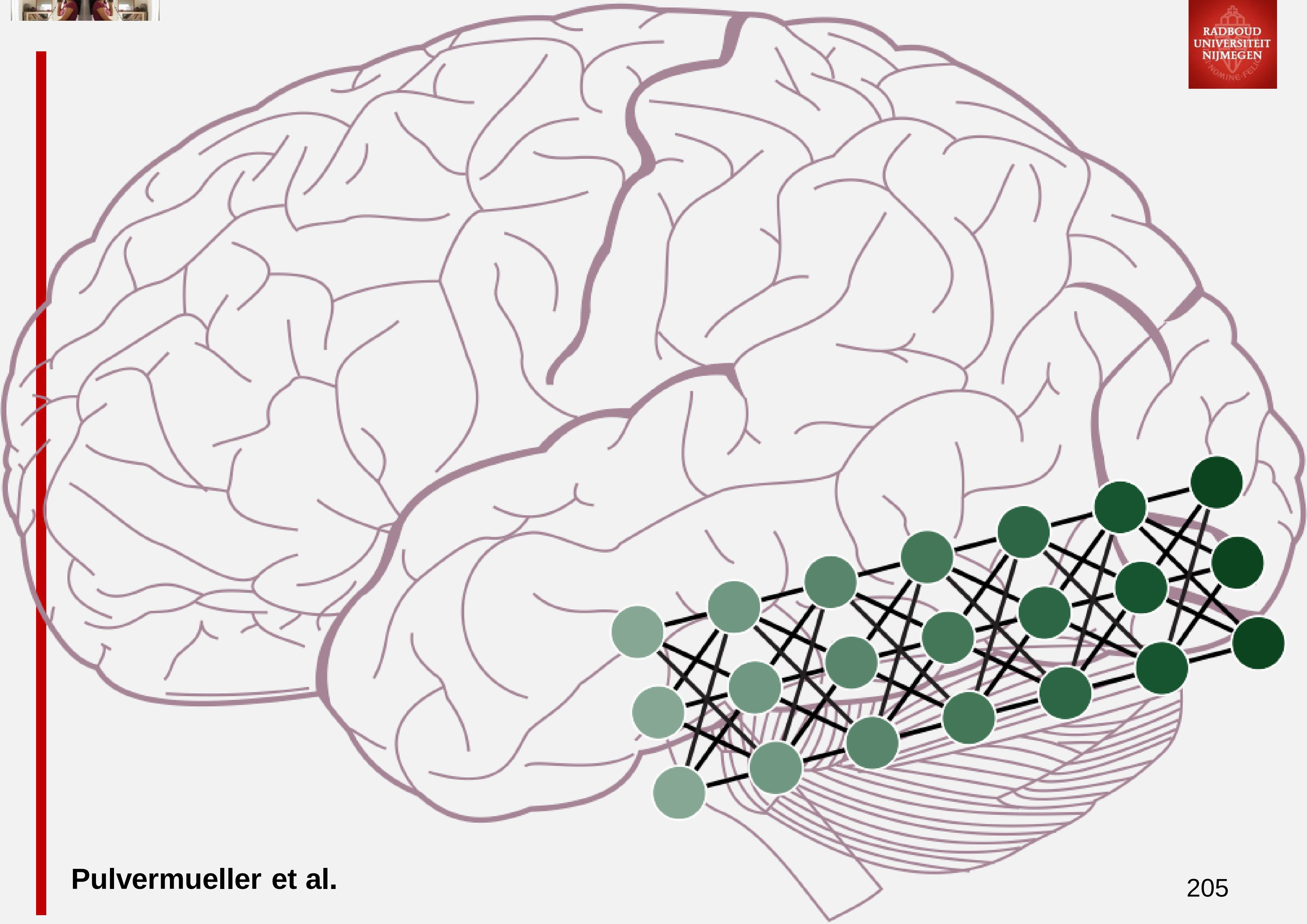
execution



reaction time (may be 1 sec!)


aalscholver
Activation
time (ms) from onset
900
800
700
600
500
400
300
200
100
100
200
300
400
500
600
700
800
900
alsxOlv@r
alsxOlv@
alsxOlv
sxolv@r
EksGuv@r
And@rsx@l
aNG@sxo
alv
Al
Ar
El
hAl
U


aalscholver
Activation
time (ms) from onset
900
800
700
600
500
400
300
200
100
100
200
300
400
500
600
700
800
900
alsxOlv@r
alsxOlv@
alsxOlv
sxolv@r
EksGuv@r
And@rsx@l
aNG@sxo
alv
Al
Ar
El
hAl
U



Pulvermueller et al.

Different ways to use ASR

- Free recognition
  - Wide search space
  - Broad topic
- Constrained recognition
  - Topic-based
  - Closed questions
- Forced alignment (wav + text)
  - Constrained search space
- Key word spotting
- Deep speech analysis

Why DNN are useful: view 1

- DNNs may discover structure in data sets because subsequent layers ignore more and more details that are irrelevant for correctly predicting output labels
- Increasingly abstract representations emerge by cascading multiple (nonlinear) transformations

  – most convincing in image classification tasks.

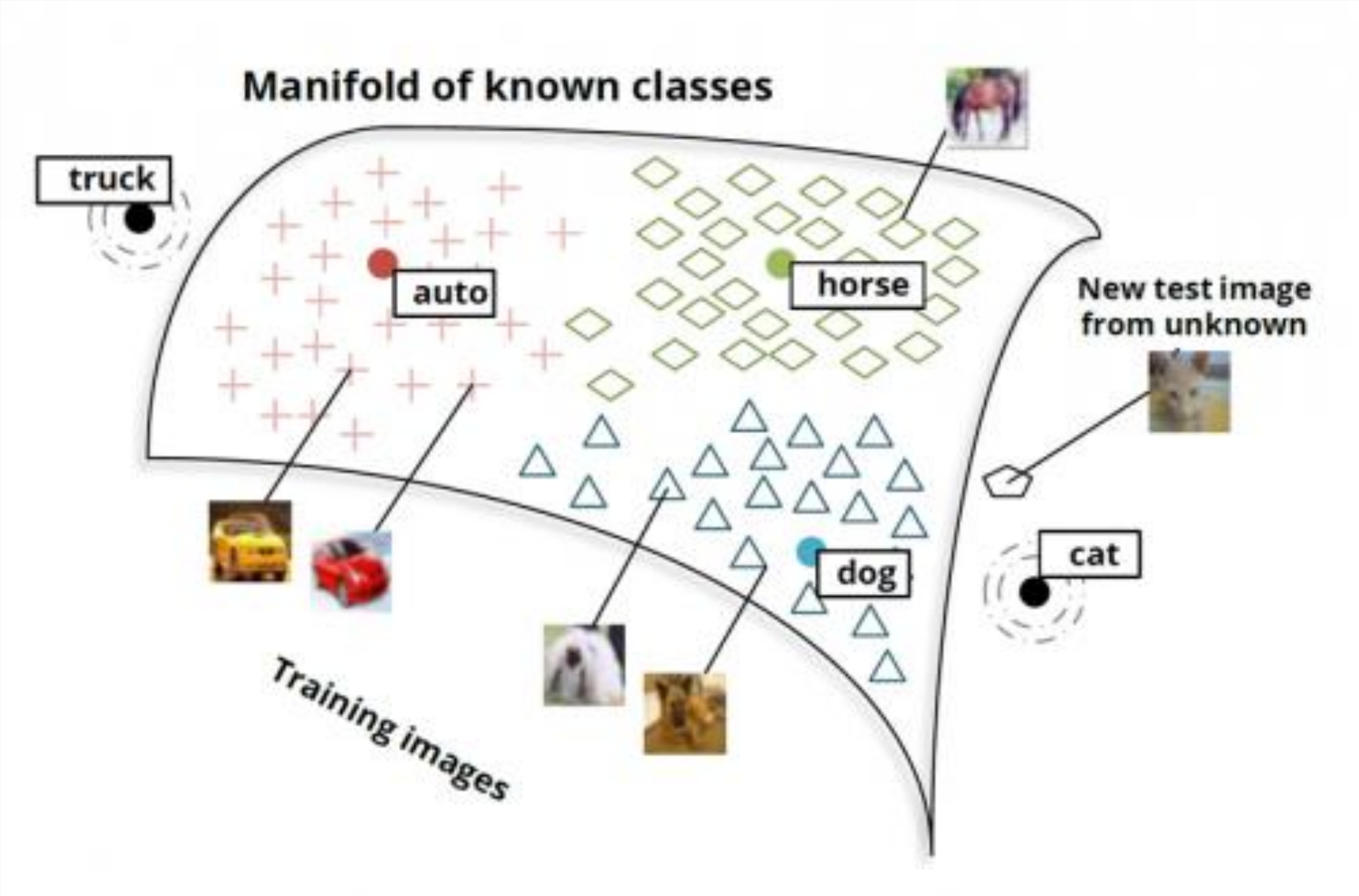
View 2

- focus on the role of DNNs to find optimal representations, in particular in the sense of features.
- learning of optimal representations can be achieved if the network is able to disentangle the underlying explanatory factors hidden in the observed data.

View 3

- more geometrically inspired interpretation of a DNN is based on the manifold assumption
  - the application of manifold learning methods on speech signals is (also) based on the relatively slow ballistic movements of articulators
- directions tangent to the manifold are well preserved while directions orthogonal to the manifolds aren’t

A manifold



Manifold of known classes
truck
auto
horse
New test image
from unknown
cat
dog
Training images

View 4

- A fourth approach is more theoretical and analyzes DNNs on an ’information plane’ using ’information bottleneck’
- Any DNN can be characterized by the mutual information between a hidden layer and the input and output variables, as a function of hidden layer depth
- bifurcation points of the information bottleneck trade-off

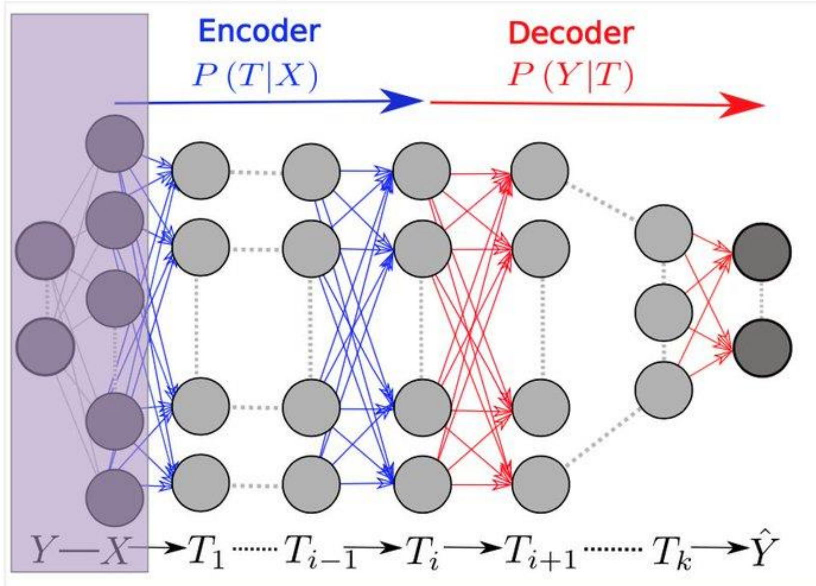


Figure 1: The DNN layers form a Markov chain of successive internal representations of the input layer X. Any representation of the input, T, is defined through an encoder, $P(T|X)$, and a decoder $P(\hat{Y}|T)$, and can be quantified by its *information plane* coordinates: $I_X = I(X;T)$ and $I_Y = I(T;Y)$. The Information Bottleneck bound characterizes the optimal representations, which maximally compress the input X, for a given mutual information on the desired output Y. After training, the network receives an input X, and successively processes it through the layers, which form a Markov chain, to the predicted output $\hat{Y}$. $I(Y;\hat{Y})/I(X;Y)$ quantifies how much of the relevant information is captured by the network.

Naftali Tishby 2017, 2018

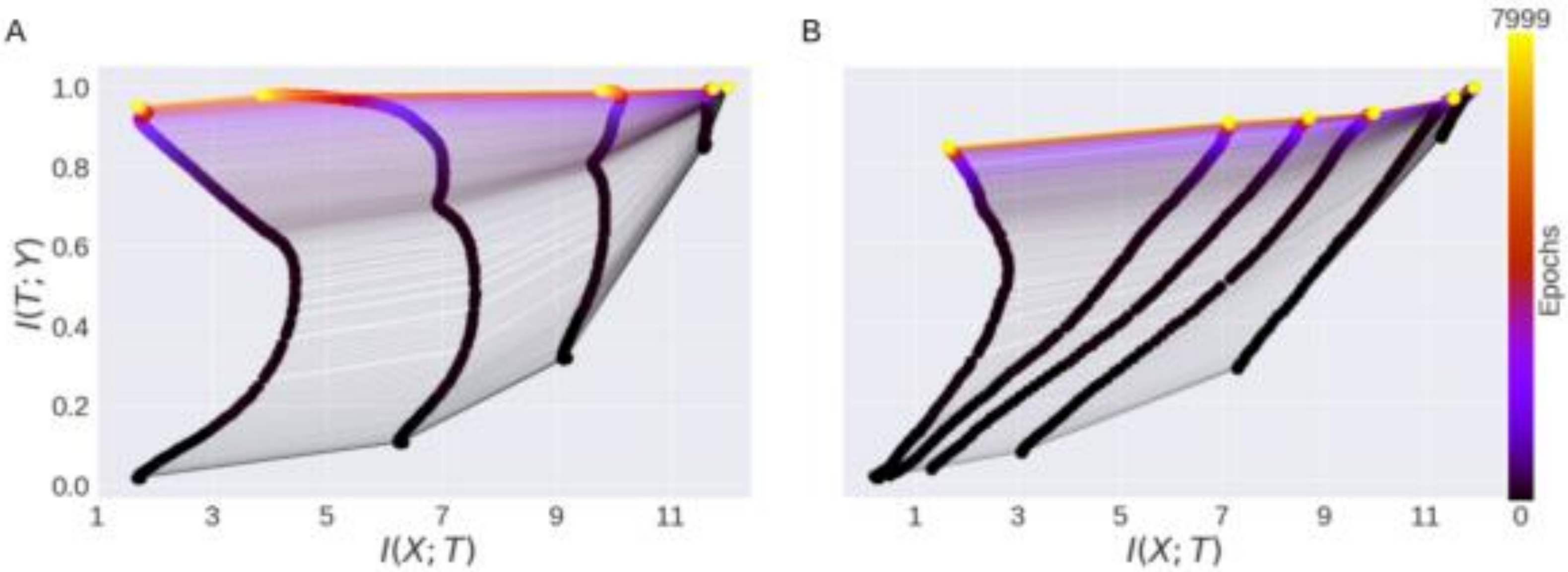
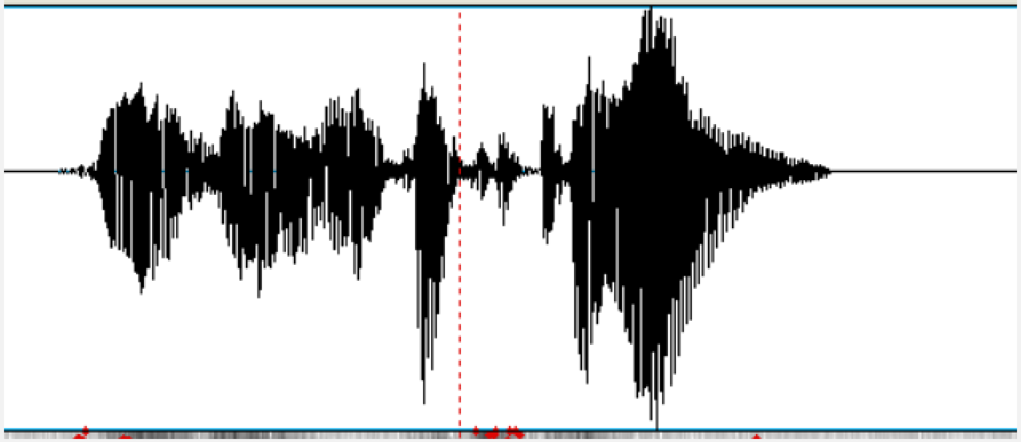


Figure 1: Information plane dynamics and neural nonlinearities. (A) Replication of Shwartz-Ziv & Tishby (2017) for a network with tanh nonlinearities (except for the final layer which contains sigmoidal neurons). The x-axis plots information between each layer and the input, while the y-axis plots information between each layer and the output. The color scale indicates training time in epochs. Each of the six layers produces a curve in the information plane with the input layer at far right, output layer at the far left. Different layers at the same epoch are connected by fine lines. (B) Information plane dynamics with ReLU nonlinearities (except for the final layer of 2 sigmoidal neurons). Here no compression phase is visible in the ReLU layers. For learning curves of both networks, see Appendix A

Brand new approaches in ASR

- Focus on learning
  - Start with empty lexicon, and a large speech corpus along with images

‘hey there look at the co-o-ow’



Dictionary learning

Given:

wav$_1$ + image$_1$

wav$_2$ + image$_2$

..

wav$_K$ + image$_K$

Textual supervision for visually grounded spoken language understanding

Bertrand Higy*, Desmond Elliott, Grzegorz Chrupala

*Corresponding author for this work

Cognitive Science & AI

Research output: Chapter in Book/Report/Conference proceeding › Conference contribution › Scientific › peer-review

How to dynamically build a lexicon?

- Lexicon contains links audio stretches (no spelling, no symbols yet) → image
- (Also an option for experiment)