“Hidden Markov Models for Speech Recognition”

by B.H. Juang and L.R. Rabiner

Papers We Love Bucharest

Stefan Alexandru Adam

9th of November 2015

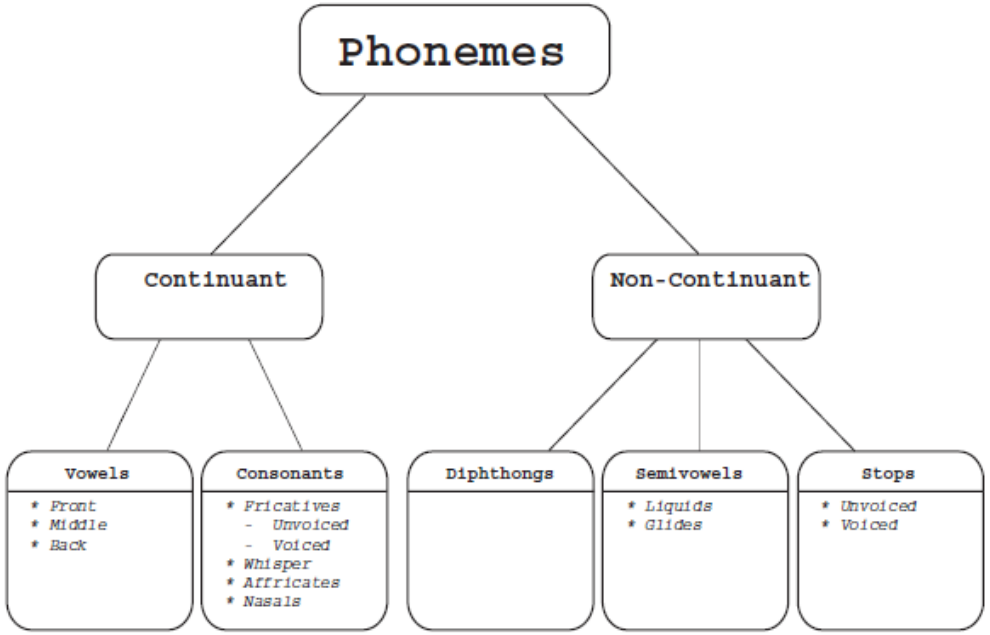
TechHub

The history of Automated Speech Recognition (ASR)

- 1950-1960 Baby Talk (only digits)
- 1970 Speech Recognition Takes Off (1011 words)
- 1980 **Big Revolution** - Prediction based approach
  - Discovery the use of HMM in Speech Recognition
  - See *Automatic Speech Recognition—A Brief History of the Technology Development by B.H. Juang and Lawrence R. Rabiner*
- 1990s: Automatic Speech Recognition Comes to the Masses
- 2000 computer toped out 80% accuracy
- Now Siri, Google Voice, Cortana

Speech signal

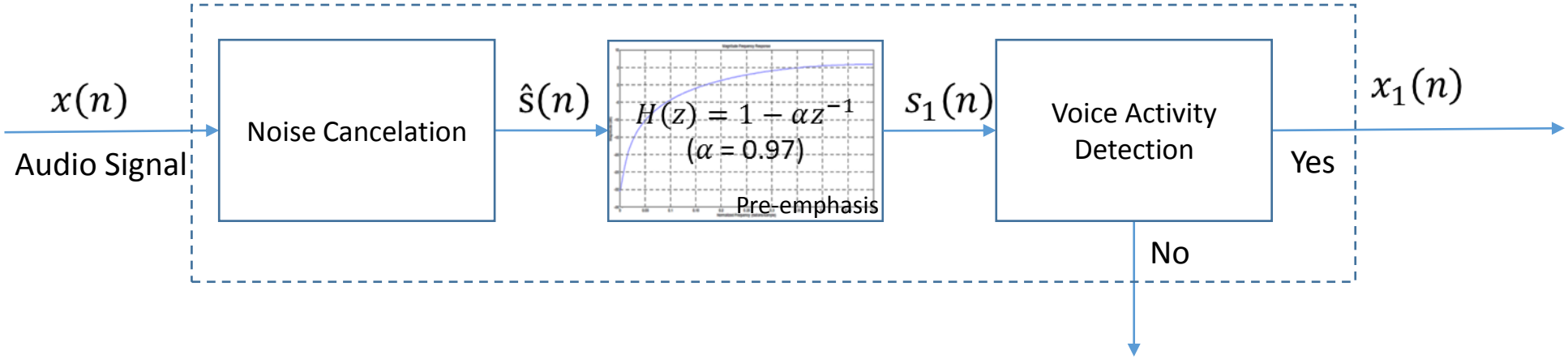
- The voice band is 0-4000 Hz
  - In telephony 300 – 3400 Hz
  - The sampling according to Nyquist –Shannon sampling theorem should be at least 8000 Hz (twice as signal frequency)
- Phoneme
  - The smallest unit of a phonetic language



ASR Model


Audio Signal
Preprocessing
Voice signal
Features Extraction
Features Vector
Detection Engine
Words

ASR Model – Preprocessing



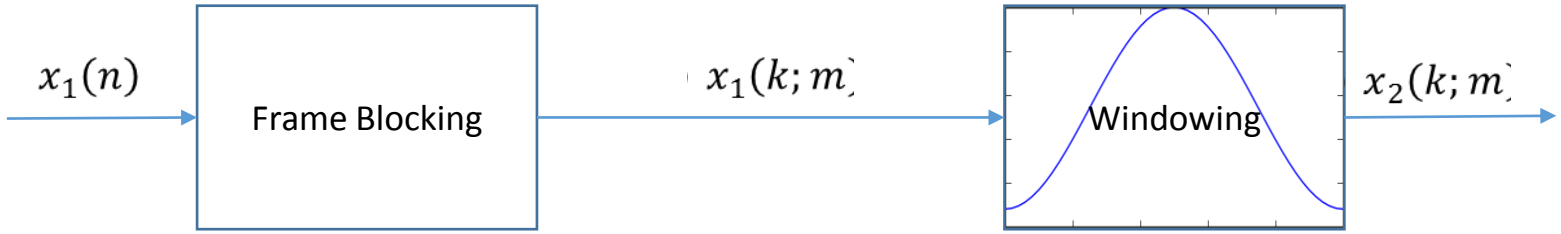
It is assumed that the initial part of the signal (usually 200 ms) is noise and silence.

$$VAD(m) = \begin{cases} 1, & W_{s_1}(m) \geq t_w \\ 0, & W_{s_1}(m) < t_w \end{cases}$$

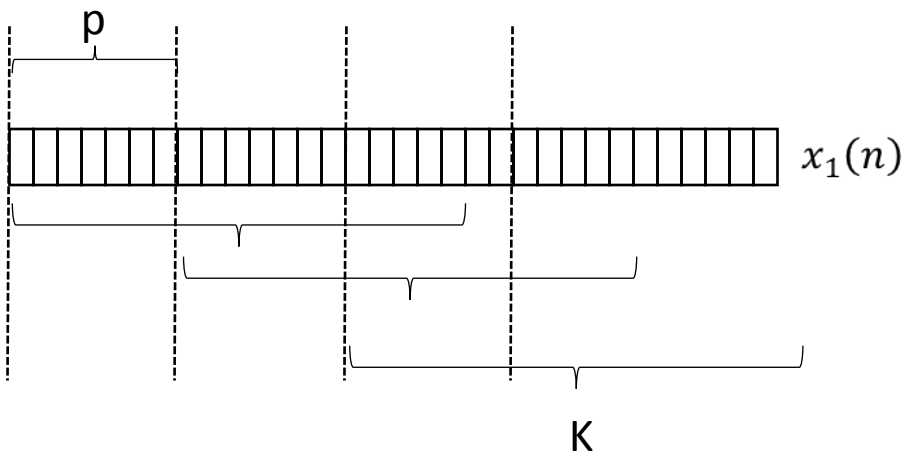
, where t_w is equal with W_{s_1} computed for the first 200 ms.

The W_{s_1} usually depends on signal energy, power, zero crossing rate

ASR Model – Features Extraction. Frame Blocking and Windowing



Each frame is K samples long and overlaps the previous one with P samples



In order to remove discontinuities a Hamming window is applied

$$w(k) = 0.54 - 0.46\cos\left(\frac{2\pi k}{K-1}\right)$$

$$x_2(k;m) = x_1(k;m) \cdot w(k)$$

ASR – Model – Features Extraction

Extract the sensitive information from the signal

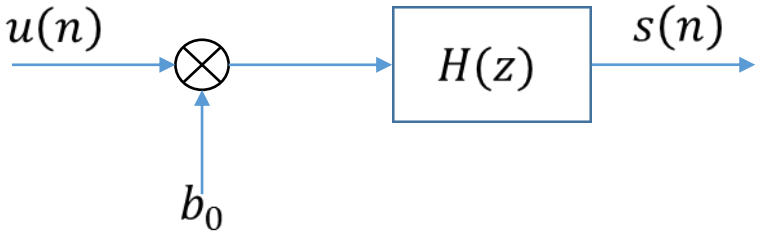
- This information is presented as a double vector

There are three main approaches

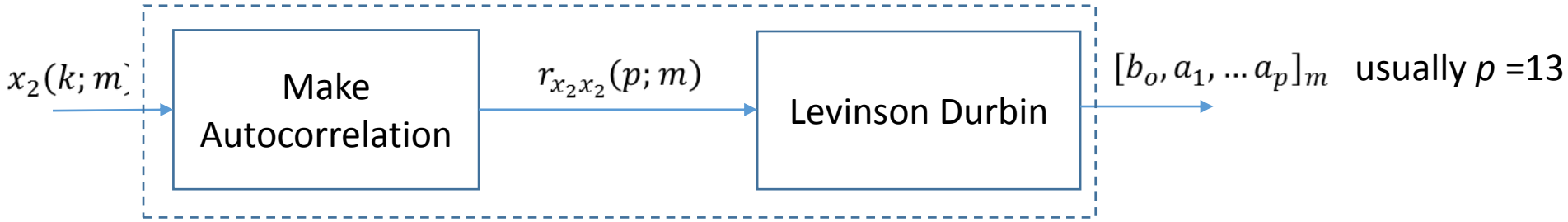
- Linear Prediction Coding (LPC)
- Mel Frequency Cepstral Coefficient (MFCC)
- Perceptual Linear Prediction (PLP)

ASR – Model – Features Extraction - LPC

The idea is to determine the transfer function $H(z)$ of the sound. Given a speech sample at time n, $s(n)$ can be modelled as a linear combination of the past p speech samples.



$$s(n) = b_0 u(n) + \sum_{1}^{\bar{p}} a_i s(n-i)$$



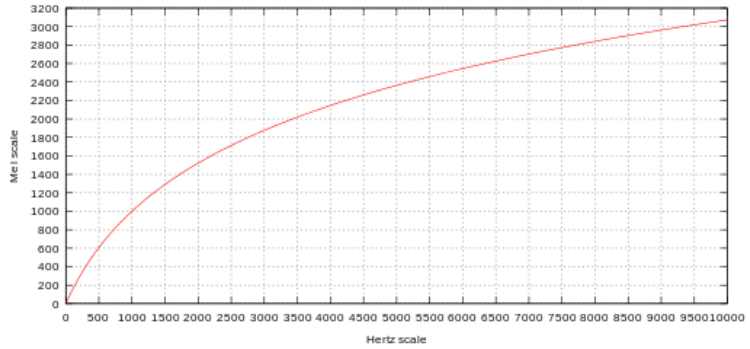
usually p =13

ASR – Model – Features Extraction - MFCC

- Mel scale
  - Is a perceptual scale of pitches judged by listener to be equal in distance from one another
  - Converts f hertz into m mel.

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

$$f = 700 \cdot (10^{\frac{m}{2595}} - 1)$$



Hz	20	160	394	670	1000	1420	1900	2450	3120	4000	5100	6600	9000	14000
mel	0	250	500	750	1000	1250	1500	1750	2000	2250	2500	2750	3000	3250

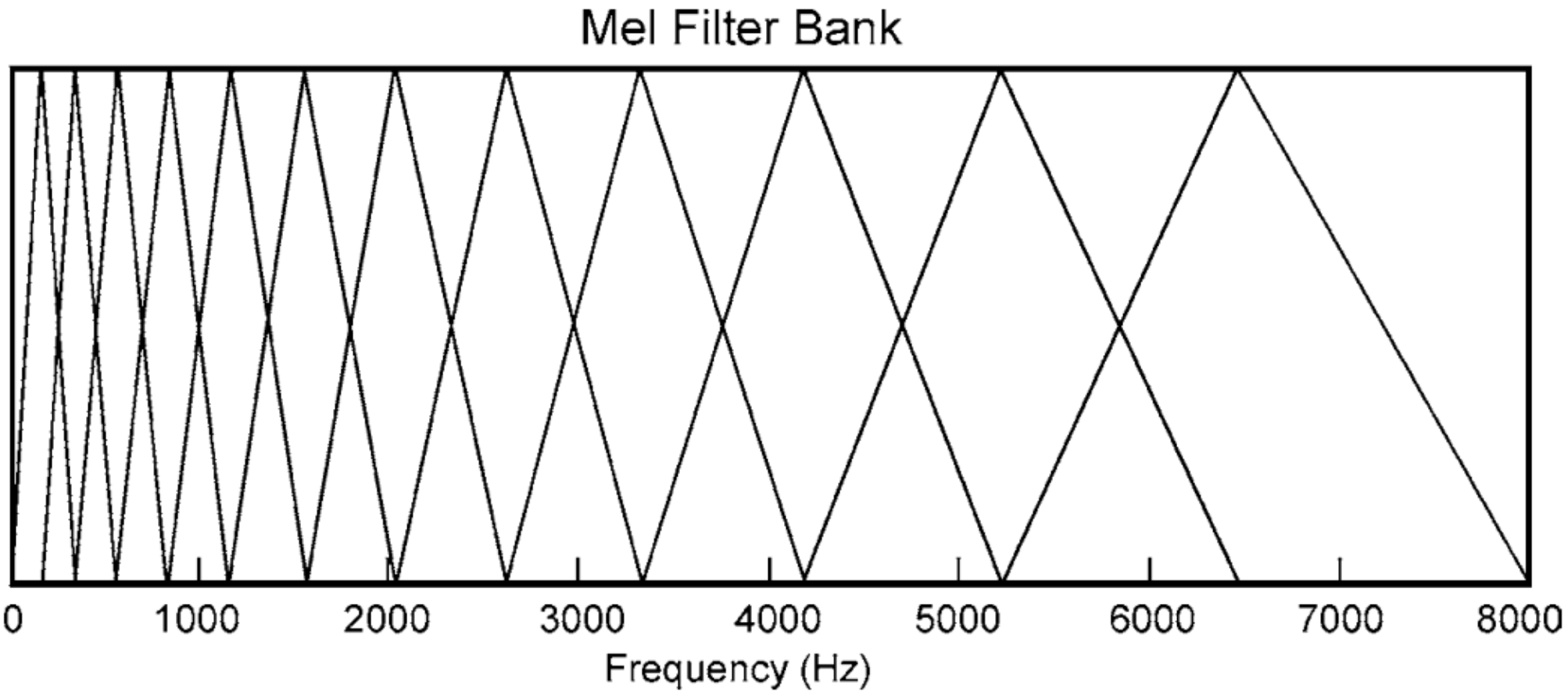
ASR – Model – Features Extraction – MFCC Mel Filter banks

Steps

1. Establish a lower and a higher frequency (typically 0 - 4000 Hz)
2. Convert this interval to a mel interval
3. Divide this interval in equal parts (typically 26 - 40 filter banks)
4. Convert back to hertz
5. Define the filter bank function

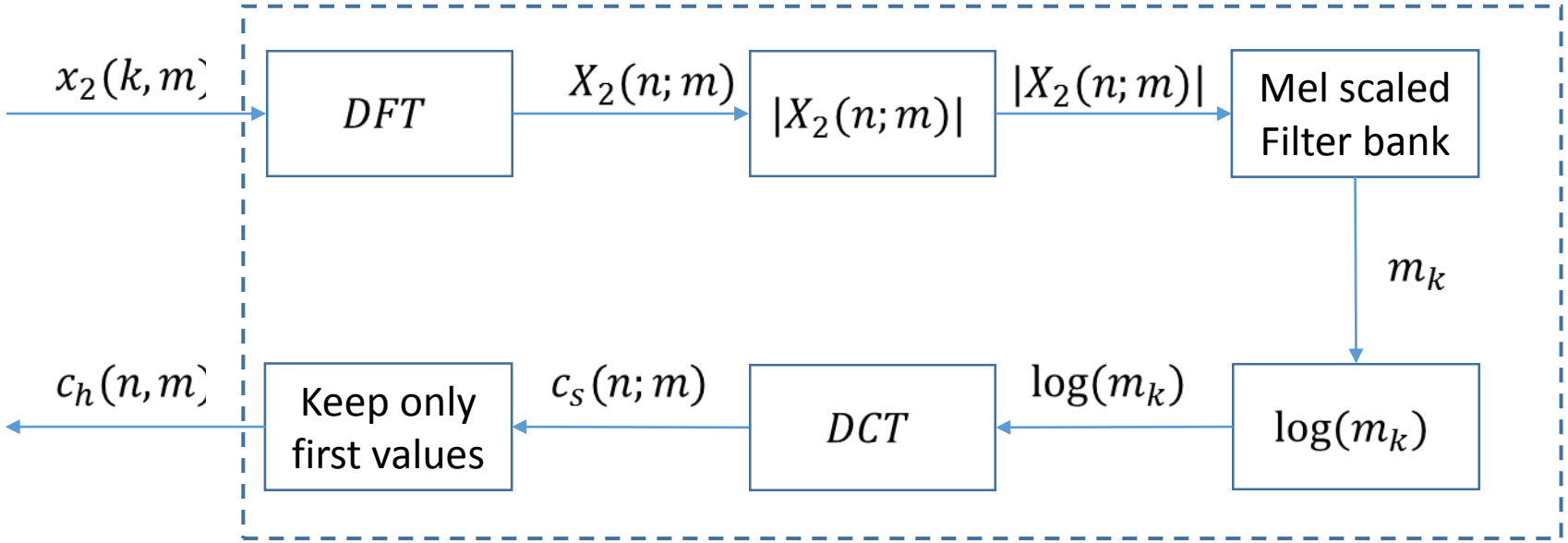
$$H_m(k) = \begin{cases} 0, & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)}, & f(m-1) \le k \le f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)}, & f(m) \le k \le f(m+1) \\ 0, & k > f(m+1) \end{cases}$$

ASR – Model – Features Extraction – MFCC Mel Filter banks



ASR – Model – Features Extraction - MFCC

Features Extraction Flow



- Signal Energy E, $\Delta c_h(n,m)$and $\Delta\Delta c_h(n,m)$coefficients are usually included in the output

ASR – Model – Detection Engine

- Pattern matching
- Hidden Markov Models
- Neural Networks

Markov Model

Markov Model is used to model randomly changing systems where future states depend only on the present state.

$$X = [X_1 \dots X_N] \; X_t \in S = \{s_1, s_2, \dots, s_N\}$$

X is called a *Markov Process*

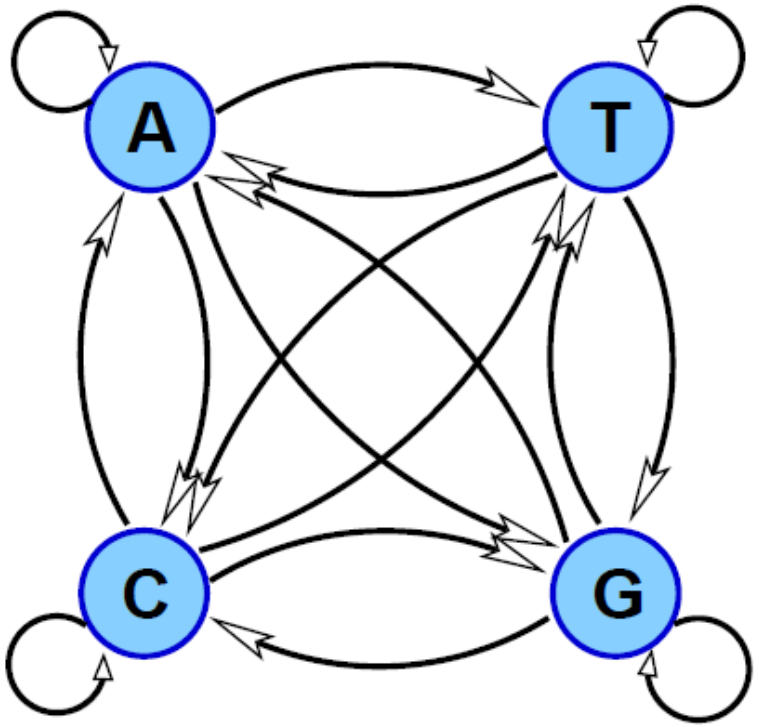
S = set of states

Π = initial state probabilities:

$\pi_i = P(X_1 = s_i); \sum_1^N \pi_i = 1$

A = transition probabilities:

$a_{ij} = P(X_{t+1} = s_j | X_t = s_i); \sum_1^N a_{ij} = 1 \; \forall i$



Hidden Markov Model

V = output alphabet = $\{v_1, v_2, \ldots, v_m\}$

B = output emission probabilities:

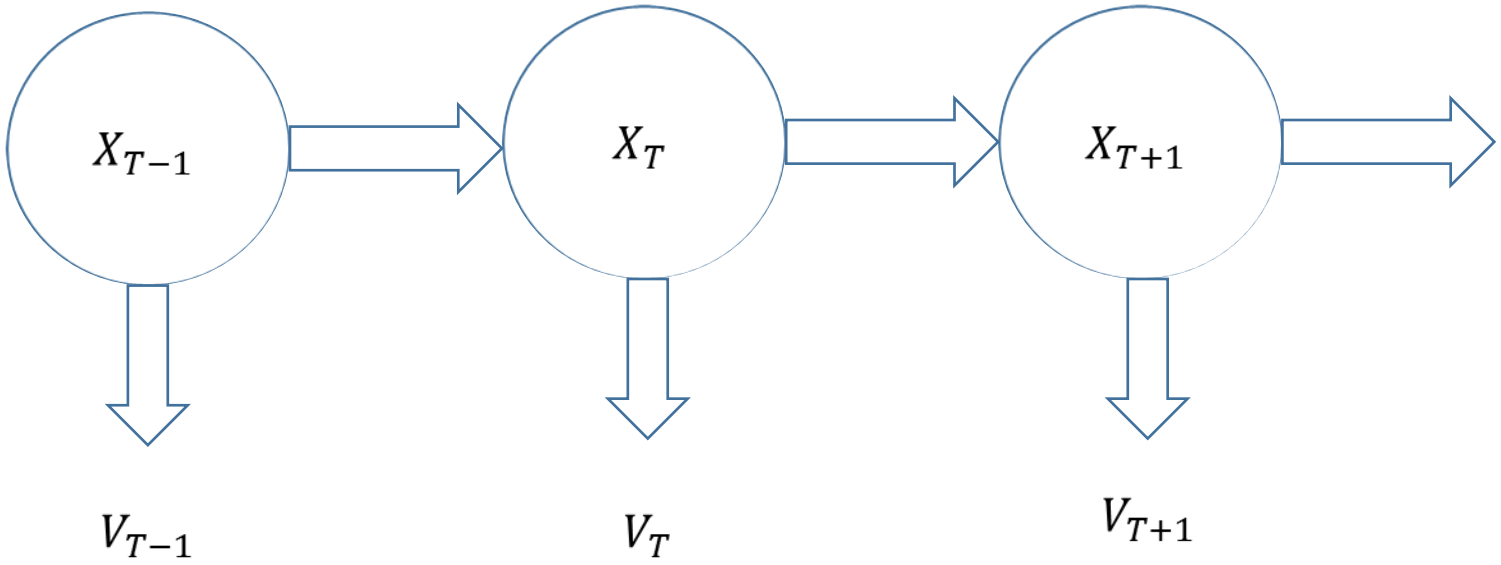
$$b_{ij} = P(O_t = v | X_t = s_i, X_{t+1} = s_j)$$

Notice that B doesn't depend on time.

The states of the system are hidden and we observe only the emissions.

Hidden Markov Model

t = 1;
start in state s_1 with probability π_i (i.e., $X_1 = i$);
forever do
move from state s_i to state s_j with prob. a_{ij} (i.e., $X_{t+1} = j$);
emit observation symbol O_t = v with probability b_{ij};
t = t + 1;



Hidden Markov Model

Questions

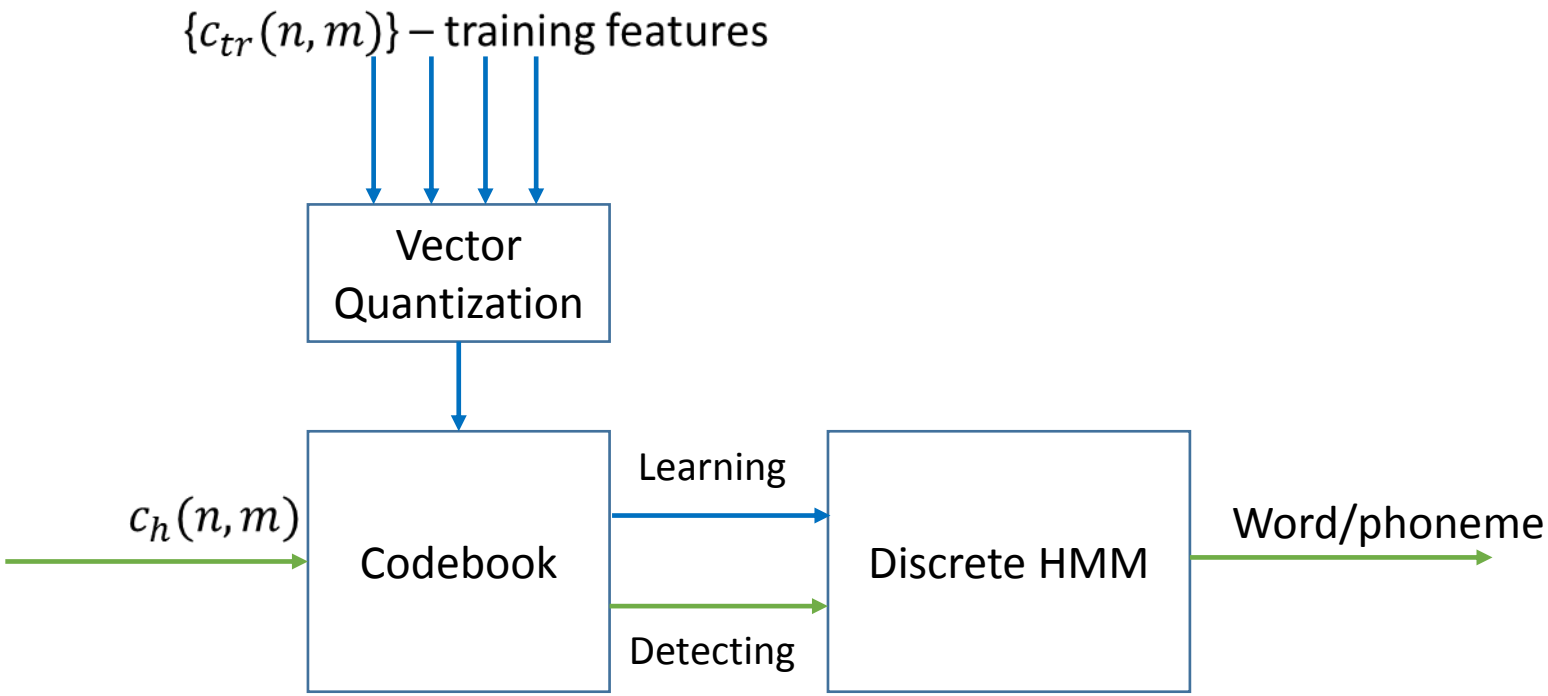
1. Given the current and last *t-i* emissions, what is the probability of the system beeing in state X_t?
2. Knowing the current state X_t what is the probability to observe the following *n* emissions $V_{t+1:n}$?
3. Knowing the emissions sequence what is the most likely states sequence?
4. Knowing the emissions sequence $V_{1:n}$ what is the probability of the system to be in state X_k where k<n?

Hidden Markov Model

For each question there is a separated dynamic programming algorithm.

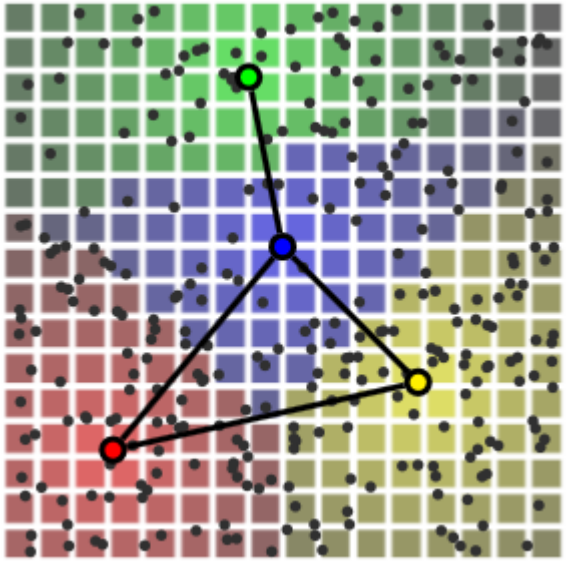
1. Forward Algorithm (recursive computation)
2. Backward Algorithm (recursive computation)
3. Viterbi algorithm
4. Forward – Backward Algorithm based on 1 and 2
5. Baum-Welch Algorithm based on 4

ASR – Model – HMM Detection Engine



ASR – Model – HMM Detection Engine Vector Quantization

- Apply vector quantization and define the Codebook
  - Usually the length of the Codebook is several hundreds
  - Apply K-Means algorithm



ASR – Model – HMM Detection Engine Training

Given a HMM $\lambda = (N, A, B)$

Record samples for a specific word or phoneme.

Using the quantized features extracted train the model using Viterbi or Baum-Welch.

- quantized features play the role of an Observations Sequence

- usually the number of training samples is 10. The samples can be grouped in two categories (man and woman).

Label (tag) the model with the corresponding word or phoneme.

ASR – Model – HMM Detection Engine Detection

Given a quantized features sequence iterates throw each existing HMM and find out which has the maximum probability.

Detection properties

- Detection is very fast (also easy to parallelize)
- Accuracy 80 % in the base implementation
- Limitations – doesn’t take in consideration observation duration

ASR – Model – HMM Detection Engine Demo

- https://github.com/AdamStefan/Speech-Recognition