Automatic Speech Recognition (ASR)

Tal Rosenwein

https://twitter.com/i/status/1265512829806927873

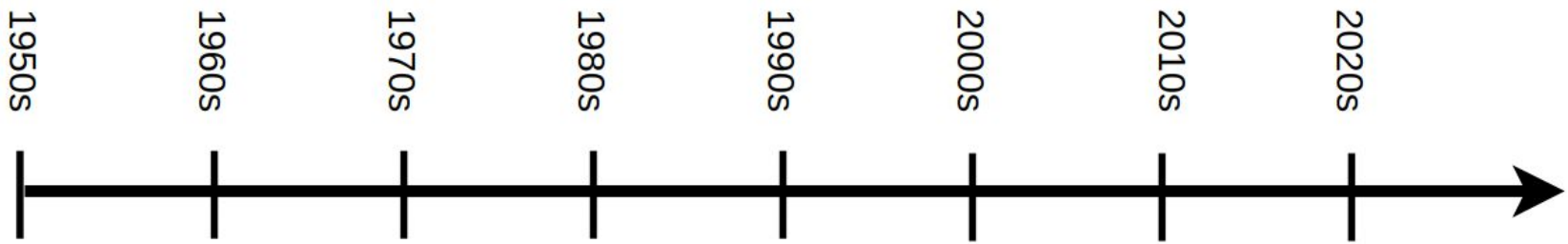
Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
    - Decoding
    - Language models
    - Pronunciation models
    - Feature extraction
    - Acoustic models
    - HMM

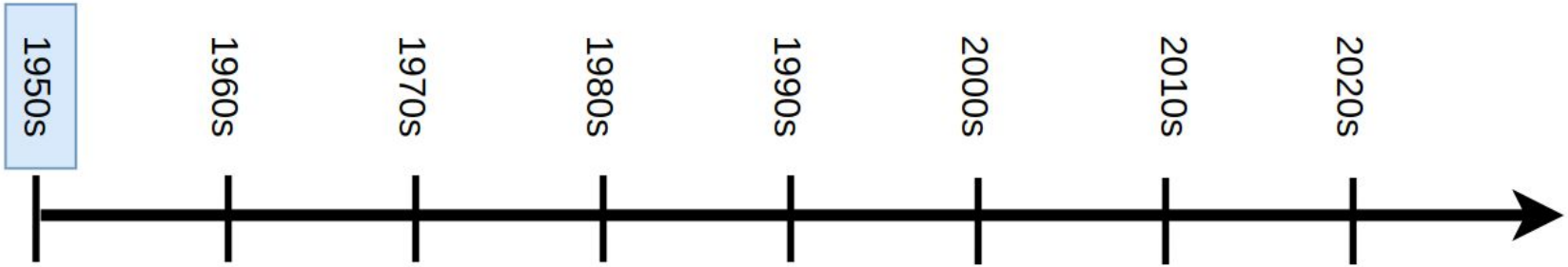
Outline

- **History**
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
    - Decoding
    - Language models
    - Pronunciation models
    - Feature extraction
    - Acoustic models
    - HMM

History



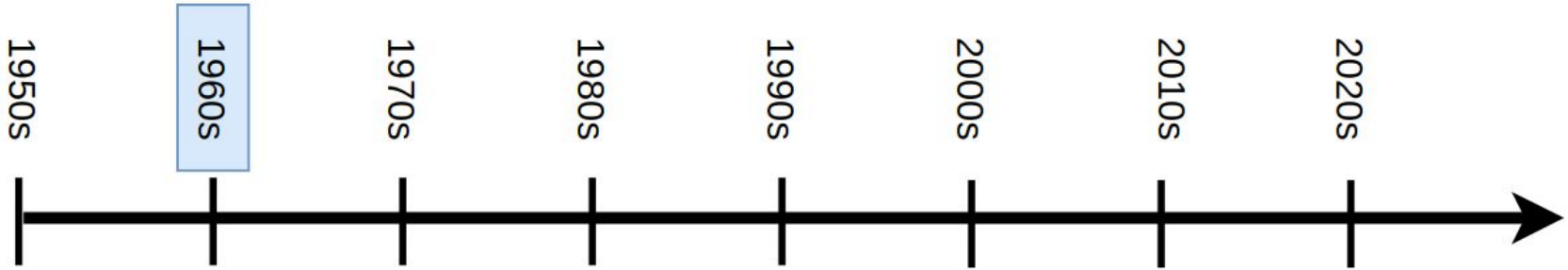
History



1952 Bell Labs Audrey. Not shown is six foot high rack of supporting electronics.

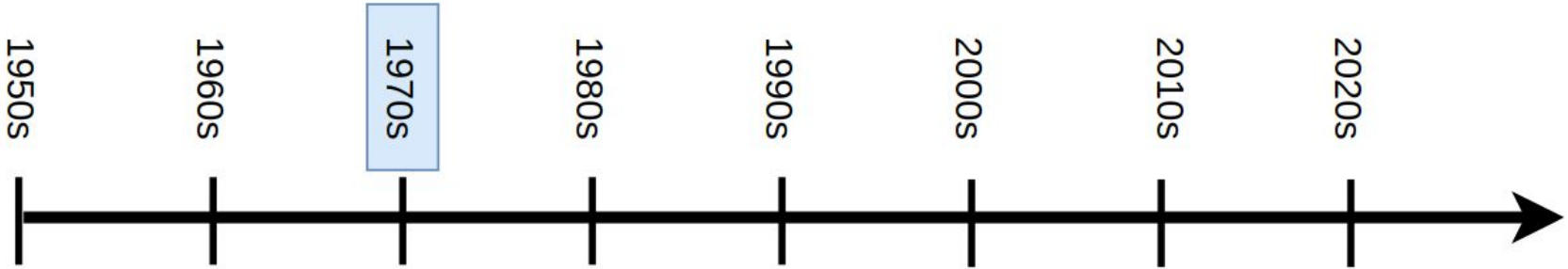
source

History



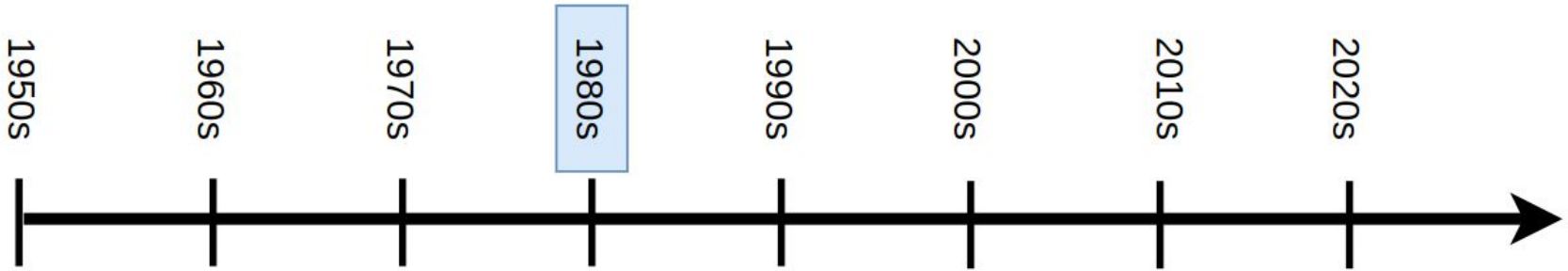
“Shoebox” by IBM

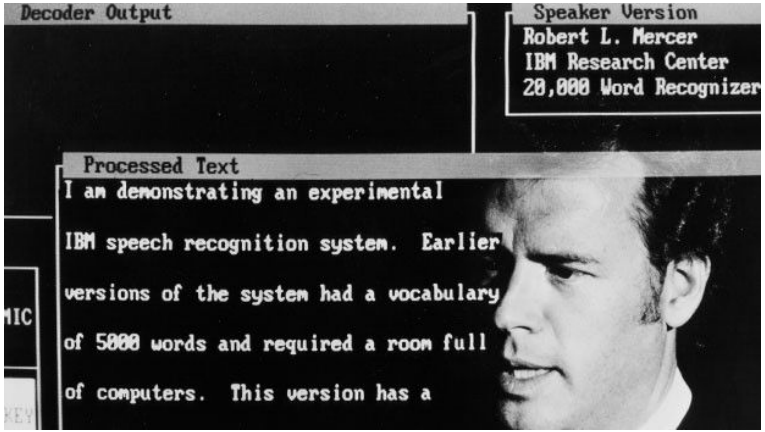
History



Growing vocabulary- DARPA funding

History

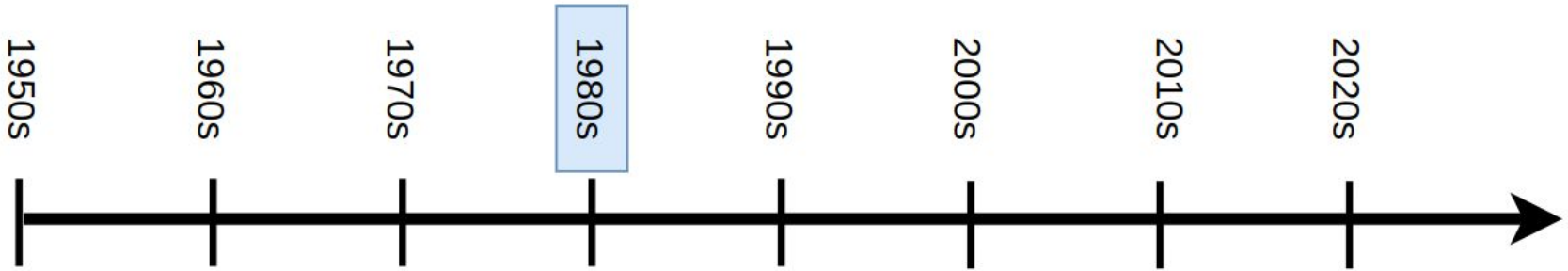


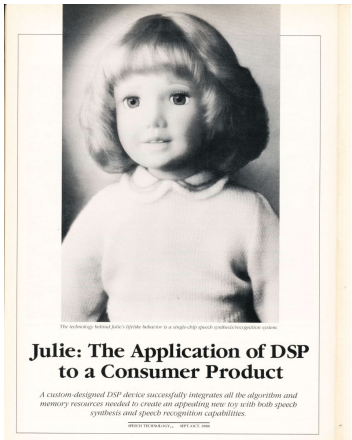


First speech recognition capability for a personal computer.

source

History

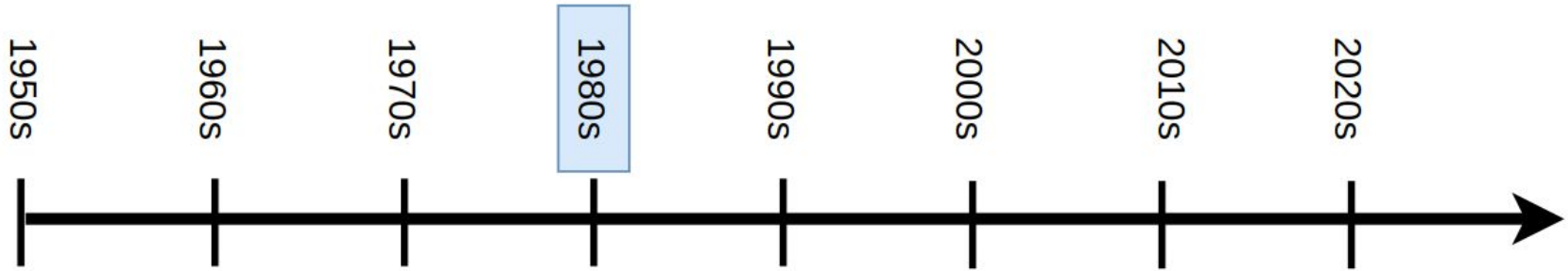




Julie- Voice first Doll.

source

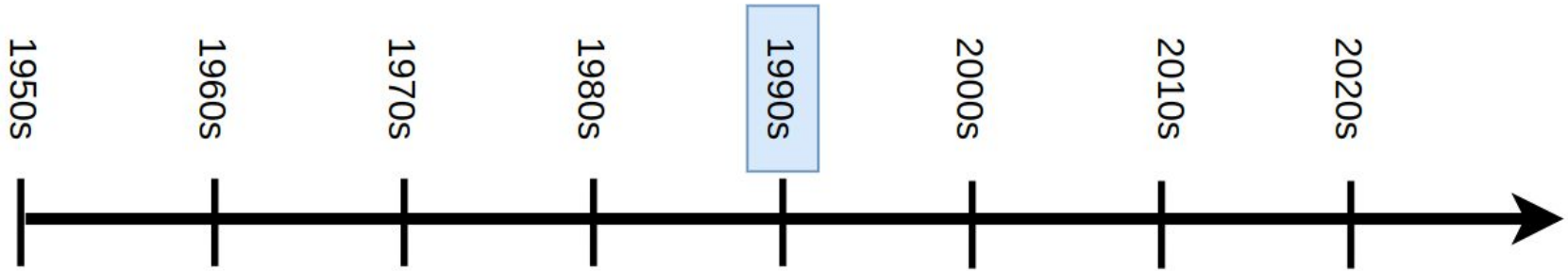
History



1989: Hidden Markov Model Toolkit (HTK) was introduced

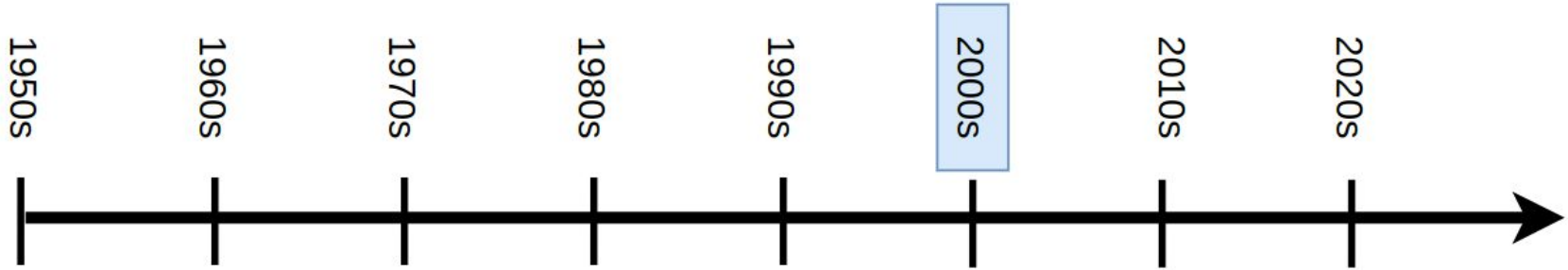
source

History



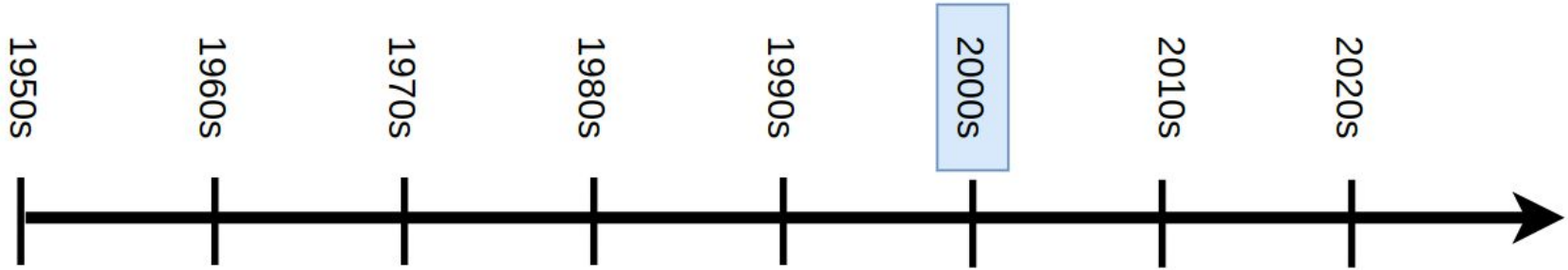
General purpose continuous speech product for consumer PCs

History



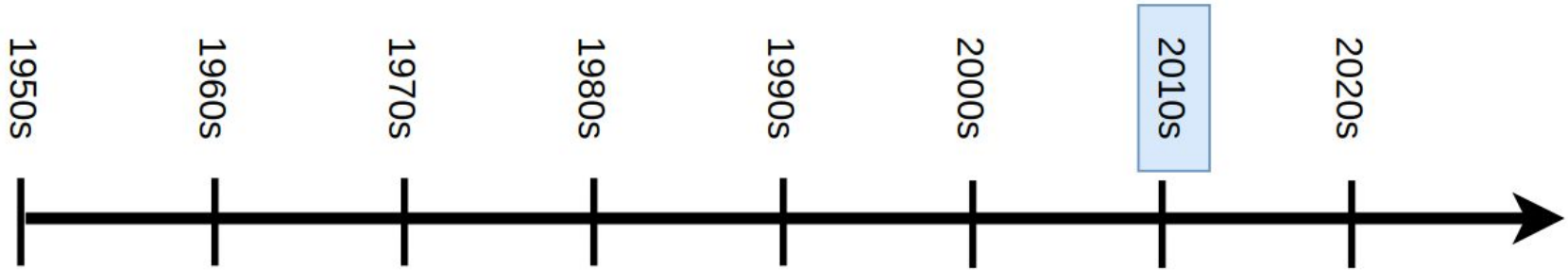
Voice search

History



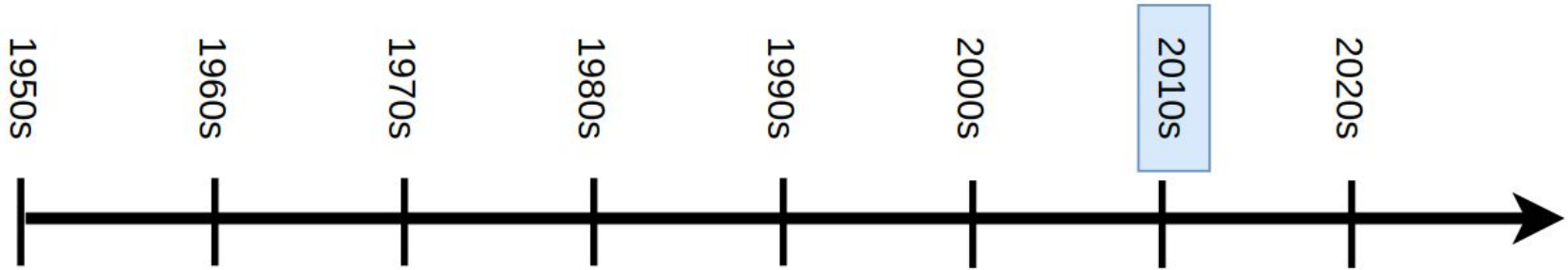
Kaldi (2009)

History



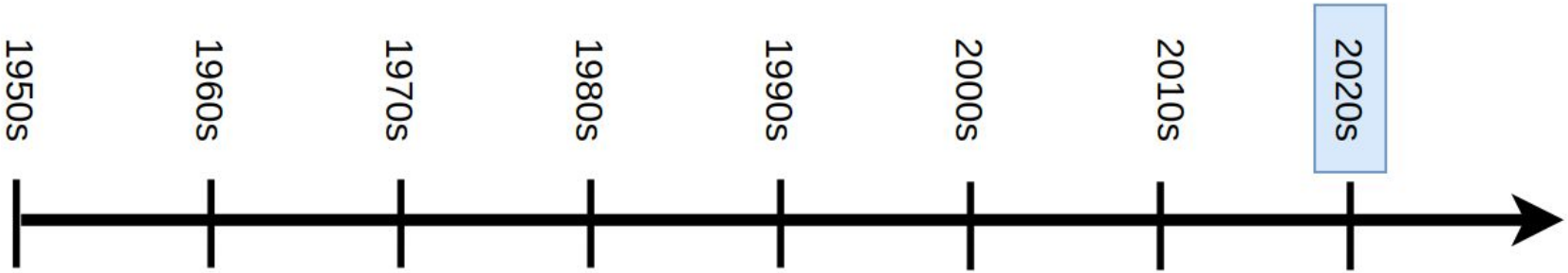
Voice enabled applications (PAs)

History



DNNs reached human level (2017)

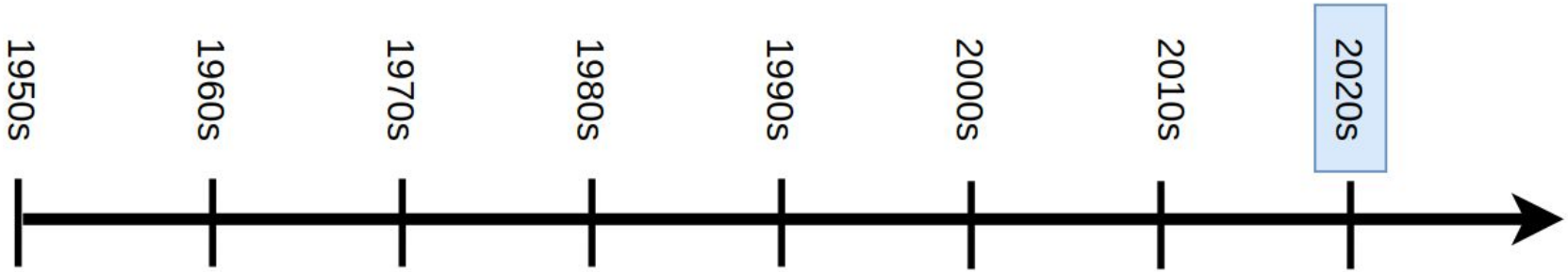
History



On device embedded ASR

source

History



Self Supervised, Multi-Modal models

Outline

- History
- **ASR as a system**
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
    - Decoding
    - Language models
    - Pronunciation models
    - Feature extraction
    - Acoustic models
    - HMM

ASR As a System

So we have the best ASR engine, but now what?

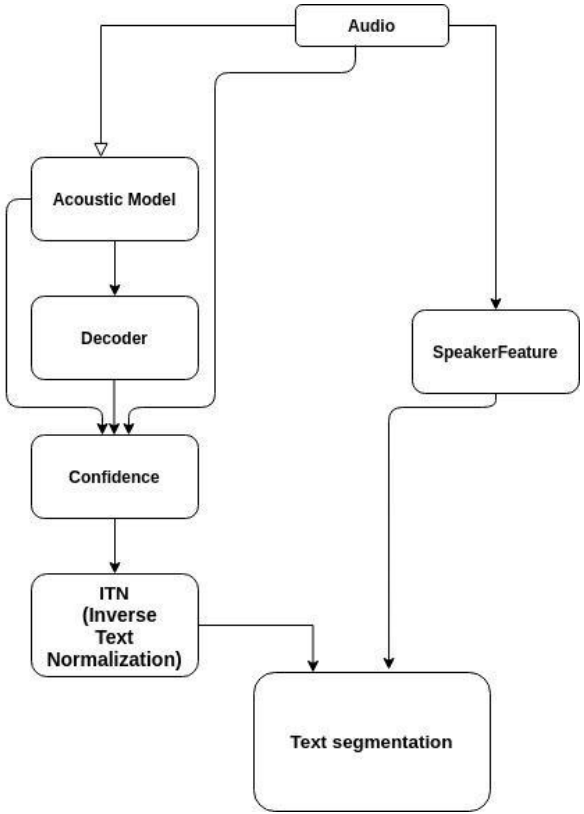
“you know there's something interesting about you i had to come meet you hmm huh what no there's something interesting about you i just wanted what to come meet you i don't know i just got that vibe that's weird is that okay no it's not oh oops”

ASR As a System

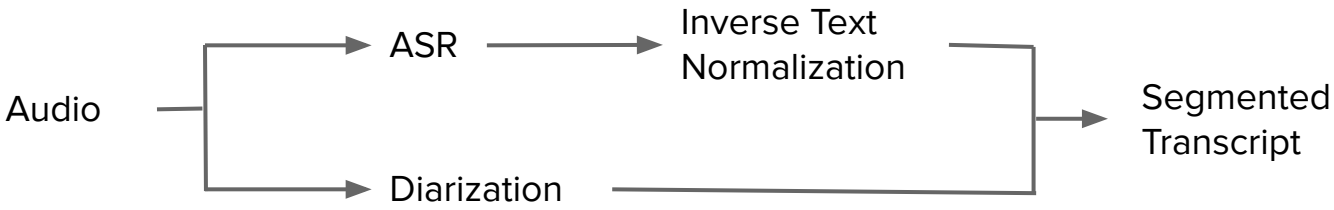
Making The Transcript More Intelligible

A: You now there's something interesting about, you I had to come meet you.
B: Hmm?
A: Huh?
B: What?
A: No, there's something interesting about you I just wanted
B: What?
A: to come meet you. I don't know, I just got that vibe
B: That's weird.
A: Is that okay?
B: No.
A: it's not. Oh, oops.

ASR As a System

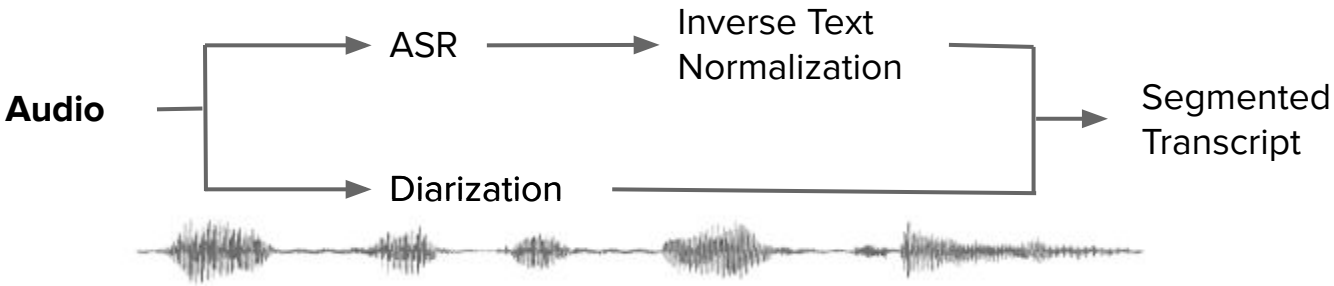


ASR As A System



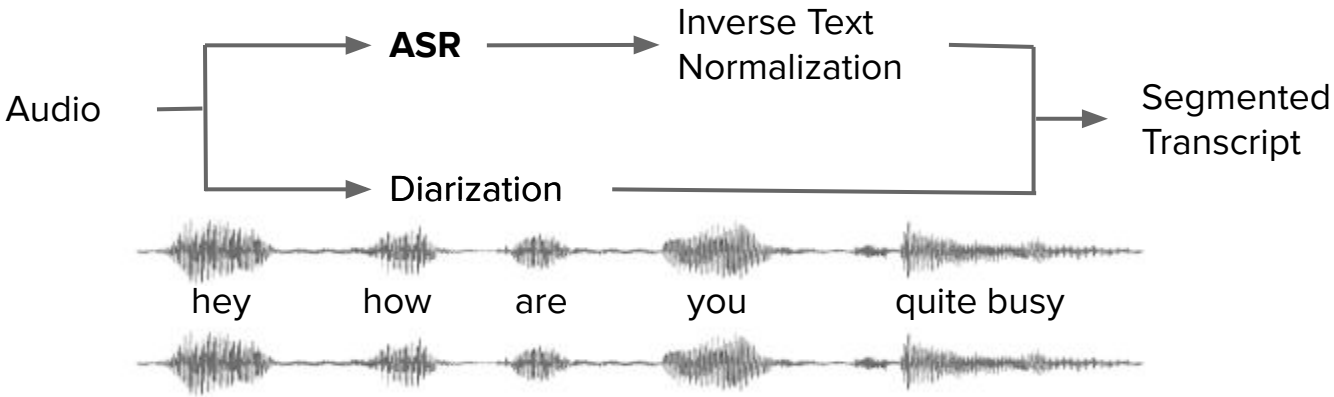
source

ASR As A System



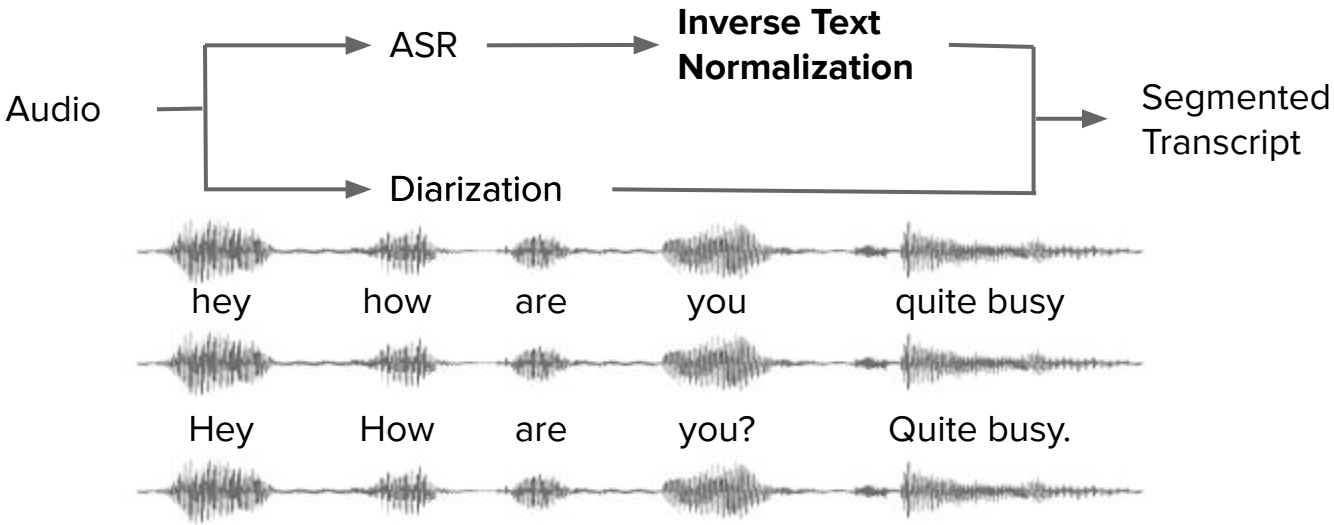
source

ASR As A System - Transcript



source

ASR As A System - ITN



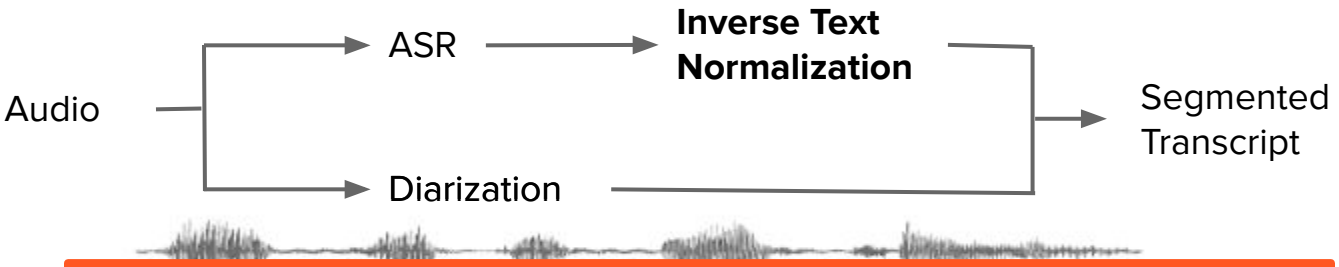
source

ASR As A System - ITN

Spoken Form	Written Form
one forty one Dorchester Avenue Salem Massachusetts	141 Dorchester Ave., Salem, MA
set an alarm for five thirty p.m.	Set an alarm for 5:30 PM
add an appointment on September sixteenth twenty seventeen	Add an appointment on September 16, 2017
twenty percent of fifteen dollars seventy three	20% of $15.73
what is two hundred seven point three plus six	What is 207.3+6

source

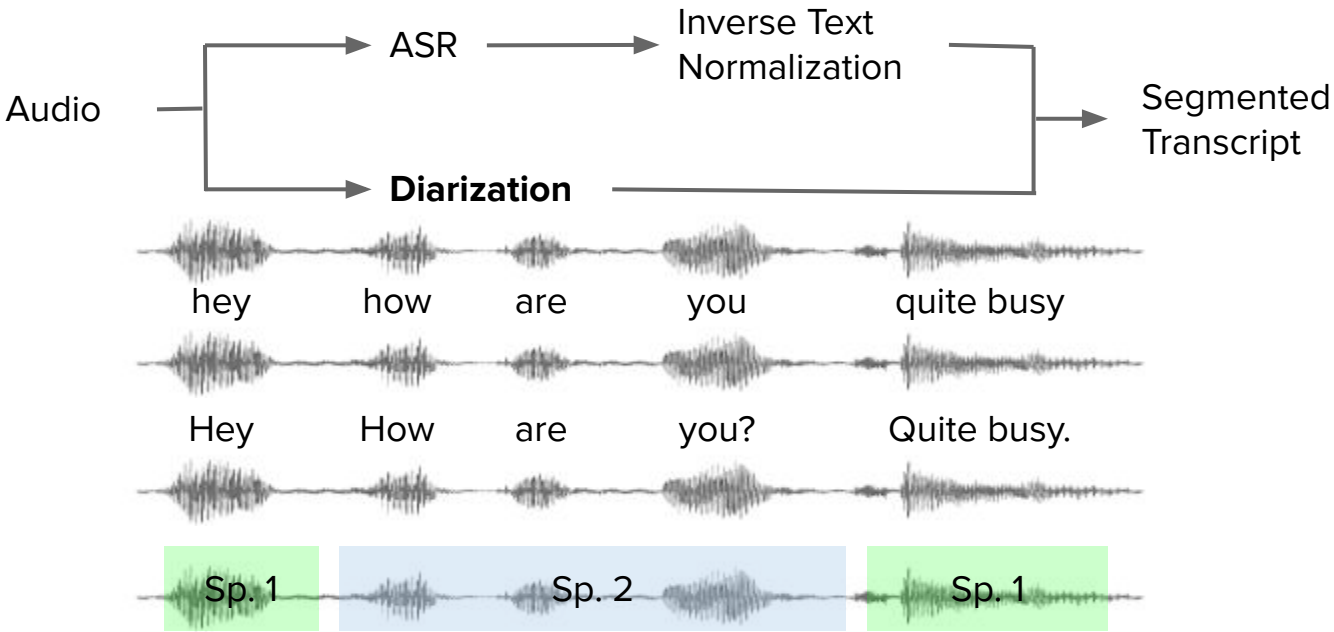
ASR As A System - ITN



Why **Inverse** text normalization?
Because usually during the acoustic model training we **normalize** the spoken text into written form.

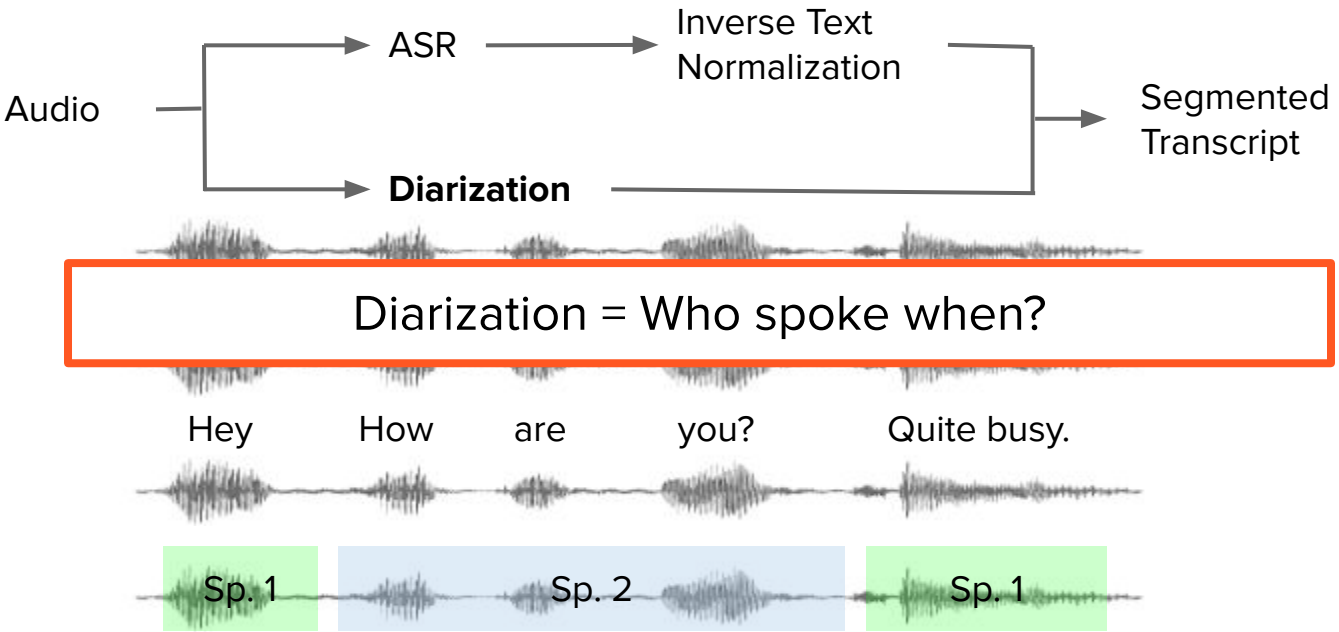
source

ASR As A System - Diarization



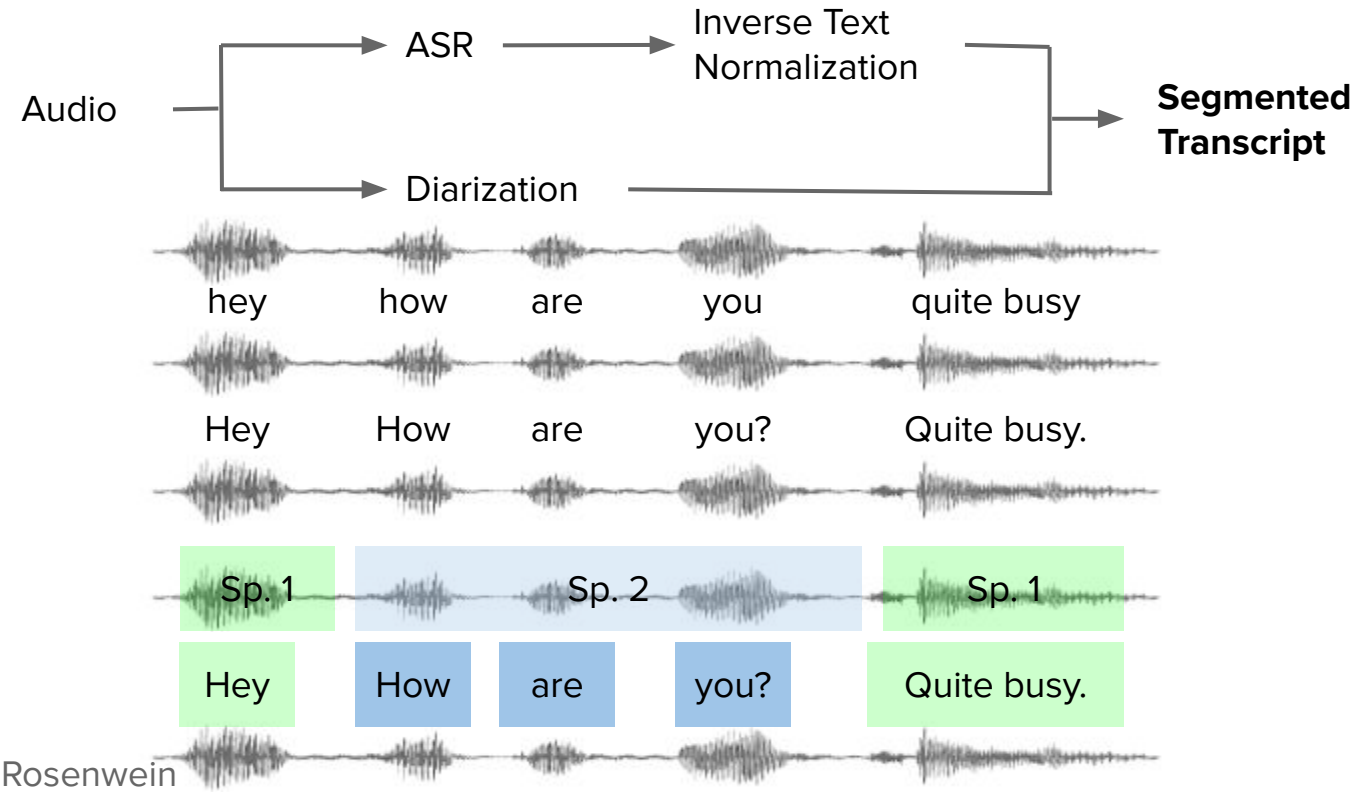
source

ASR As A System - Diarization



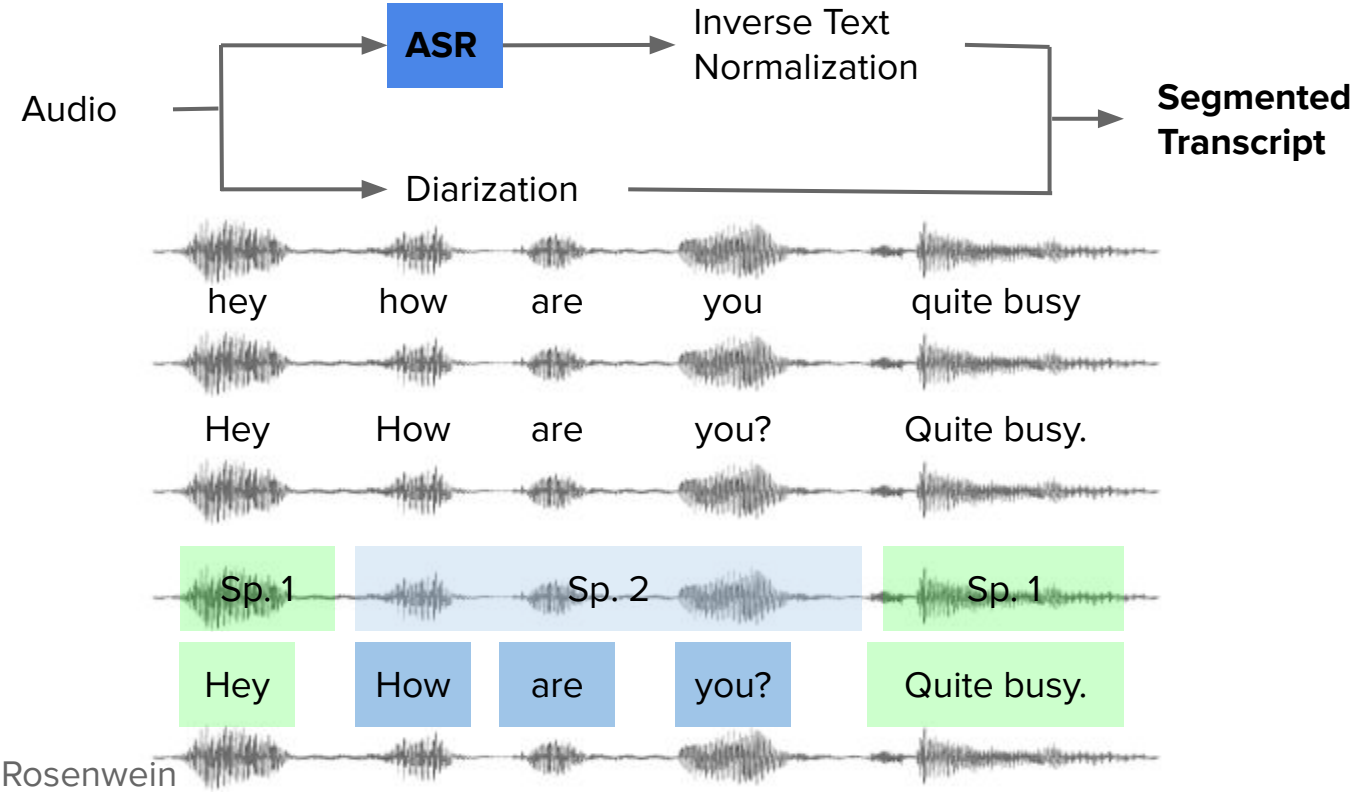
source

ASR As A System - Final Output



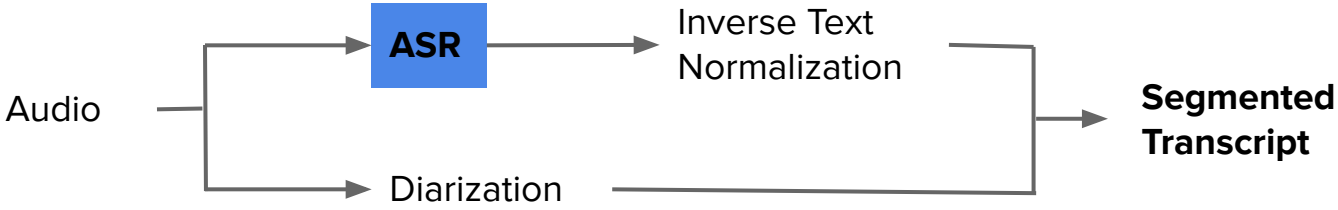
source

ASR As A System - Our Main Focus



source

ASR As A System - E2E Training of System



Joint Speech Recognition and Speaker Diarization via Sequence Transduction

Laurent El Shafey, Hagen Soltau, Izhak Shafran

Google

shafey@google.com, soltau@google.com, izhak@google.com

Abstract

Speech applications dealing with conversations require not only recognizing the spoken words, but also determining who spoke when. The task of assigning words to speakers is typically addressed by merging the outputs of two separate systems, namely, an automatic speech recognition (ASR) system and a speaker diarization (SD) system. The two systems are trained independently with different objective functions. Often the SD systems operate directly on the acoustics and are not constrained to respect word boundaries and this deficiency is overcome in an *ad hoc* manner. Motivated by recent advances in sequence to sequence learning, we propose a novel approach to tackle the two tasks by a joint ASR and SD system using a recurrent neural network transducer. Our approach utilizes both

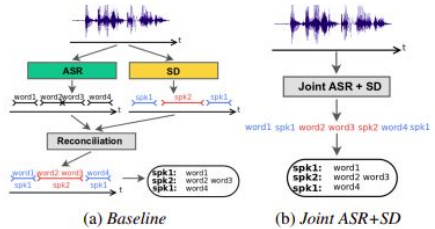


Figure 1: *Comparison of the conventional speech recognition and speaker diarization system (Figure 1a) with the proposed approach (Figure 1b), where the task consists of generating a speaker-decorated transcript from raw audio.*

hello dr jekyll <spk:pt> hello mr hyde what brings you here today <spk:dr> I am struggling again with my bipolar disorder <spk:pt>

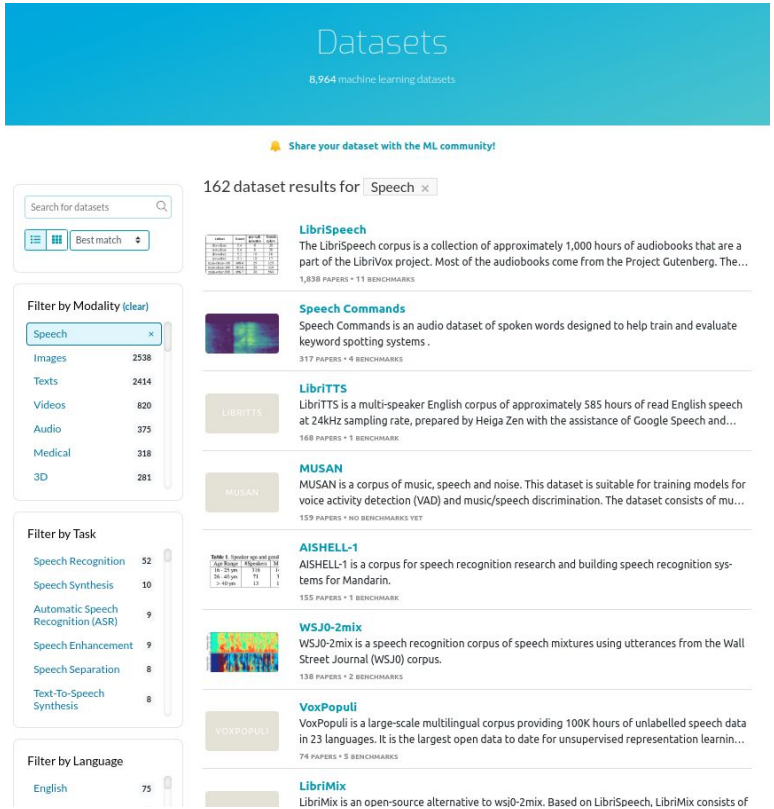
Figure 2: *Example of an output sequence for our joint ASR and SD RNN-T system. The corresponding input would be the raw audio signal. Speaker turns are displayed in different colors.*

source

Outline

- History
- ASR as a system
- **Datasets & Evaluation metrics**
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
    - Decoding
    - Language models
    - Pronunciation models
    - Feature extraction
    - Acoustic models
    - HMM

Datasets



source

Leaderboard

📐 The 🤗 Open ASR Leaderboard ranks and evaluates speech recognition models on the Hugging Face Hub.
We report the Average WER (⬇️) and RTF (⬇️) - lower the better. Models are ranked based on their Average WER, from lowest to highest. Check the 📈 Metrics tab to understand how the models are evaluated.
If you want results for a model that is not listed here, you can submit a request for it to be included 🤗✨.
The leaderboard currently focuses on English speech recognition, and will be expanded to multilingual evaluation in later versions.

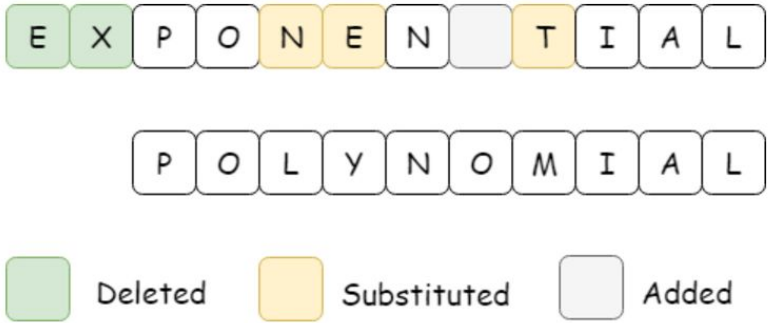
🏅 Leaderboard | 📈 Metrics | 🤗✨ Request a model here!

model	Average WER ⬇	RTF (1e-3) ⬇	AMI	Earnings22	Gigaspeech	LS Clean	LS Other	SPGISpeech	Tedli
openai/whisper-large-v3	7.7	10.3	16.01	11.3	10.02	2.03	3.91	2.95	3.9
nvidia/stt_en_fastconformer_transducer_xlarge	8.06	12.3	18.28	16.37	11.58	1.5	2.88	4.4	4.49
openai/whisper-large-v2	8.06	10.5	16.82	12.02	10.57	2.56	5.16	3.77	4.01
nvidia/stt_en_fastconformer_transducer_xxlarge	8.07	14.4	18.81	16.66	11.95	1.38	2.52	4.98	4.74
distil-whisper/distil-large-v2	8.31	4.93	14.65	12.12	10.31	2.95	6.39	3.28	4.3
nvidia/stt_en_fastconformer_ctc_xxlarge	8.34	5	17.62	16.44	11.61	1.69	3.4	4.91	4.64
nvidia/stt_en_conformer_ctc_large	8.39	7.5	15.97	15.83	11.59	2.06	4.16	5.6	4.41
openai/whisper-medium.en	8.5	10.7	16.43	12.59	11.13	3.02	5.84	3.41	4.15
nvidia/stt_en_fastconformer_ctc_xlarge	8.52	2.9	18.41	17.89	11.84	1.73	3.47	5.04	4.71
nvidia/stt_en_fastconformer_ctc_large	8.9	1.8	18.59	18.67	12.15	1.95	4.04	5.03	4.74
nvidia/stt_en_fastconformer_transducer_large	8.94	10.4	20.2	19.36	12.16	1.67	3.64	4.43	4.44
openai/whisper-large	9.2	10.5	17.9	15.77	11.84	3.04	6.01	3.99	4.69
nvidia/stt_en_conformer_transducer_large	9.27	21.8	22.27	19.91	12.5	1.64	3.51	4.96	5.15
distil-whisper/distil-medium.en	9.32	3.95	16.02	12.89	11.26	3.6	7.67	3.77	4.8

source

Evaluation Metrics - Edit Distance

Edit Distance (ED): sums the Insertion, Deletion & Substitution errors



$$ED = \frac{Ins. + Subs. + Del.}{|GT|}$$

Character Error Rate (CER): ED Between Characters
Word Error Rate (WER): ED Between Words

CER < WER

source

Evaluation Metrics - Edit Distance

WER sometimes correlates poorly with user perception of transcription quality:

- Does not consider semantic correctness
- Weights all errors equally (stop words, Nouns...)- The words that are rare count more.
- Ignores multi-hypothesis (don’t vs do not, going to vs gonna).

Evaluation Metrics- Semantic Distance

Table 2: *Examples of Ref/Hyp that has the biggest gap between WER and SemDist.*

Gap	WER	SemDist	Ref/Hyp
12011	16.67	0.00	**Ref:** hey portal play **mister** blue sky **Hyp:** hey portal play **mr** blue sky.
8742	50.00	0.00	**Ref:** I smell **hot dogs** **Hyp:** I smell **hotdogs.**
6390	6.67	0.01	**Ref:** keep away it is a nightmare thank God we are separated about four thousand **kilometres** **Hyp:** keep away, it is a nightmare. thank god we are separated about four thousand **kilometers.**
3807	20.00	0.02	**Ref:** keep **time zones** in mind for the next zoom call **Hyp:** keep **timezones** in mind for the next Zoom call.
2725	66.67	3.14	**Ref:** **I'm** eagerly waiting **Hyp:** **I am** eagerly waiting.

(a) top-5 (Rank_{WER} - $\text{Rank}_{\text{SemDist}}$) gap

Gap	WER	SemDist	Ref/Hyp
2486	6.25	302.73	**Ref:** of course in the first time **but** one by one I able to handle those complaints **Hyp:** of course, in the first time, **birth** one by one, I able to handle those complaints.
2388	10.00	473.47	**Ref:** okay then it's kind of a great **weekly** for him **Hyp:** okay, then it's kind of a great **victory** for him.
2345	10.00	426.67	**Ref:** do you know the most **whom** he used to admire **Hyp:** do you know the most **home** he used to admire?
2314	7.69	286.03	**Ref:** sure but uh I think you will have to be patient because **usually it's** something like once a year or a little bit more but not much **Hyp:** sure but uh I think you will have to be patient because **you read** something like once a year or a little bit more but not much.
2300	10.00	375.39	**Ref:** yeah I think I've seen that in the **news** before **Hyp:** yeah, I think I've seen that in the **morning** before

source

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- **ASR levels of difficulty**
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
    - Decoding
    - Language models
    - Pronunciation models
    - Feature extraction
    - Acoustic models
    - HMM

ASR Levels Of Difficulty

Keywords spotting

Closed Set Voice Commands

Open Domain Questions

'Sterile' LVCSR

Real Life LVCSR

Hey Siri

Say "Hey Siri, send a message."

Image source 1 2 3 4 5

ASR Levels Of Difficulty

Keywords spotting

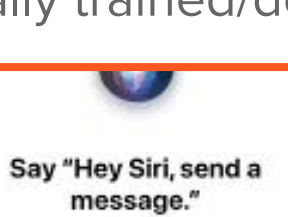
Closed Set Voice

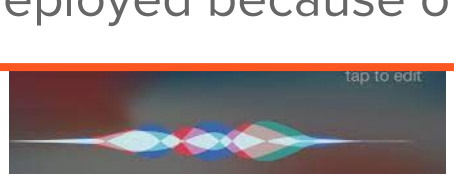
Open Domain

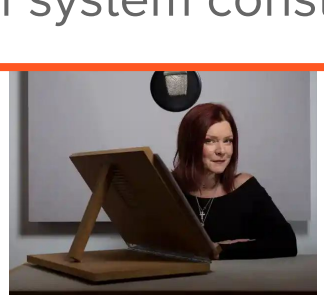
'Sterile' LVCSR

Real Life LVCSR

Models that solve downstream tasks other than transcription, are usually trained/deployed because of system constraints







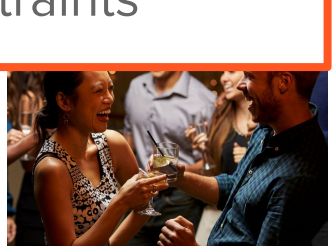


Image source 1 2 3 4 5

Levels Of Difficulty: Why ASR Is Difficult?

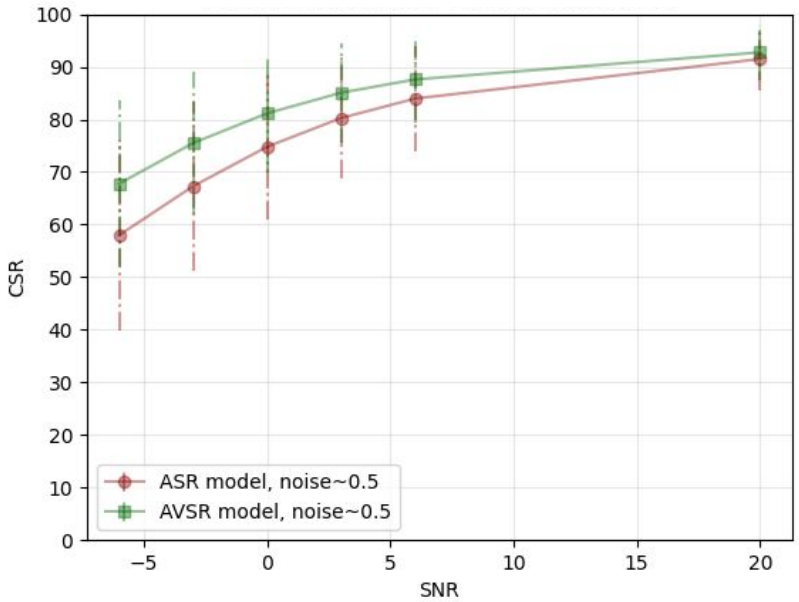
- **Variety of Speech Attributes:**
  - Accent
  - Rate
  - Gain
  - Gender

Levels Of Difficulty: Why ASR Is Difficult?

- **Variety of Speech Attributes:**
  - Accent
  - Rate
  - Gain
  - Gender
- **Real-Life Scenarios:**
  - Background Noise / Cocktail Party
  - Structured Vs Unstructured Speech

Levels Of Difficulty: Why ASR Is Difficult?

- **Variety of Speech Attributes:**
  - Accent
  - Rate
  - Gain
  - Gender
- **Real-Life Scenarios:**
  - Background Noise / Cocktail Party
  - Structured Vs Unstructured Speech



Levels Of Difficulty: Why ASR Is Difficult?

- **Variety of Speech Attributes:**
  - Accent
  - Rate
  - Gain
  - Gender
- **Real-Life Scenarios:**
  - Background Noise / Cocktail Party
  - Structured Vs Unstructured Speech
- **Audio Is Not Sufficient To Determine The Transcription:**
  - Large Context is Needed

Levels Of Difficulty: Why ASR Is Difficult?

- **Variety of Speech Attributes:**
  - Accent
  - Rate
  - Gain
  - Gender
- **Real-Life Scenarios:**
  - Background Noise / Cocktail Party
  - Structured Vs Unstructured Speech
- **Audio Is Not Sufficient To Determine The Transcription:**
  - Large Context is Needed
- **Non-Lexical Conversational Sounds (NLCS) are hard to recognize for ASR**

Levels Of Difficulty: Why ASR Is Difficult?

- Varie
  -
  -
  -
  -
- Real
  -
  -
- Audi
  -
- Non

JOURNAL ARTICLE

"Mm-hm," "Uh-uh": are non-lexical conversational sounds deal breakers for the ambient clinical documentation technology?

Get access >

Brian D Tran, Kareem Latif, Tera L Reynolds, Jihyun Park, Jennifer Elston Lafata, Ming Tai-Seale, Kai Zheng ✉

Journal of the American Medical Informatics Association, ocad001, https://doi.org/10.1093/jamia/ocad001

Published: 23 January 2023 **Article history** ▾

for ASR

source

Levels Of t?

- Variety of
  - Accer
  - Rate
  - Gain
  - Gend
- Real-Life S
  - Backg
  - Struct
- Audio Is N
  - Large
- Non-Lexic ize for ASR

Materials and Methods

In this study, we evaluated 2 contemporary ASR engines, Google Speech-to-Text Clinical Conversation ("Google ASR"), and Amazon Transcribe Medical ("Amazon ASR"), both of which have their language models specifically tailored to clinical conversations. The empirical data used were from 36 primary care encounters. We conducted a series of quantitative and qualitative analyses to examine the word error rate (WER) and the potential impact of misrecognized NLCS on the quality of clinical documentation.

Results

Out of a total of 135 647 spoken words contained in the evaluation data, 3284 (2.4%) were NLCS. Among these NLCS, 76 (0.06% of total words, 2.3% of all NLCS) were used to convey clinically relevant information. The overall WER, of all spoken words, was 11.8% for Google ASR and 12.8% for Amazon ASR. However, both ASR engines demonstrated poor performance in recognizing NLCS: the WERs across frequently used NLCS were 40.8% (Google) and 57.2% (Amazon), respectively; and among the NLCS that conveyed clinically relevant information, 94.7% and 98.7%, respectively.

source

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- **Dynamic time warping (DTW)**
- ASR problem formulation
- Inference pipeline
    - Decoding
    - Language models
    - Pronunciation models
    - Feature extraction
    - Acoustic models
    - HMM

Dynamic Time Warping (DTW)

- Used for aligning two sequences
  - Two instances of the same / different speaker are not exactly the same.

Dynamic Time Warping (DTW)

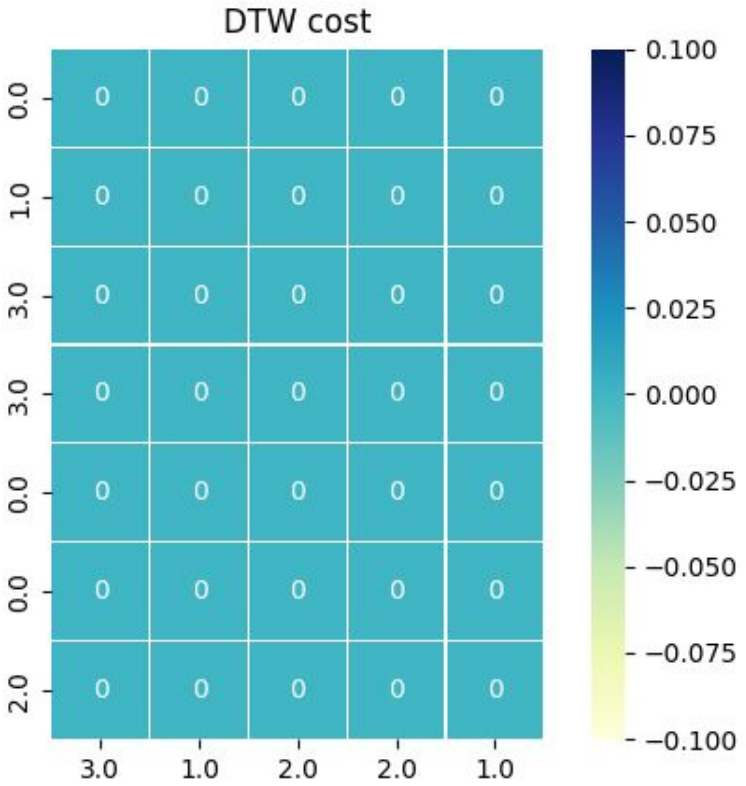
- Used for aligning two sequences
  - Two instances of the same / different speaker are not exactly the same.
- How do we compare 2 files/sequences with different lengths
  - Align (warp) the sequences
  - Calculate the distance

Dynamic Time Warping (DTW)

Given two sequences:

x = [3, 1, 2, 2, 1]

y = [2, 0, 0, 3, 3, 1, 0]

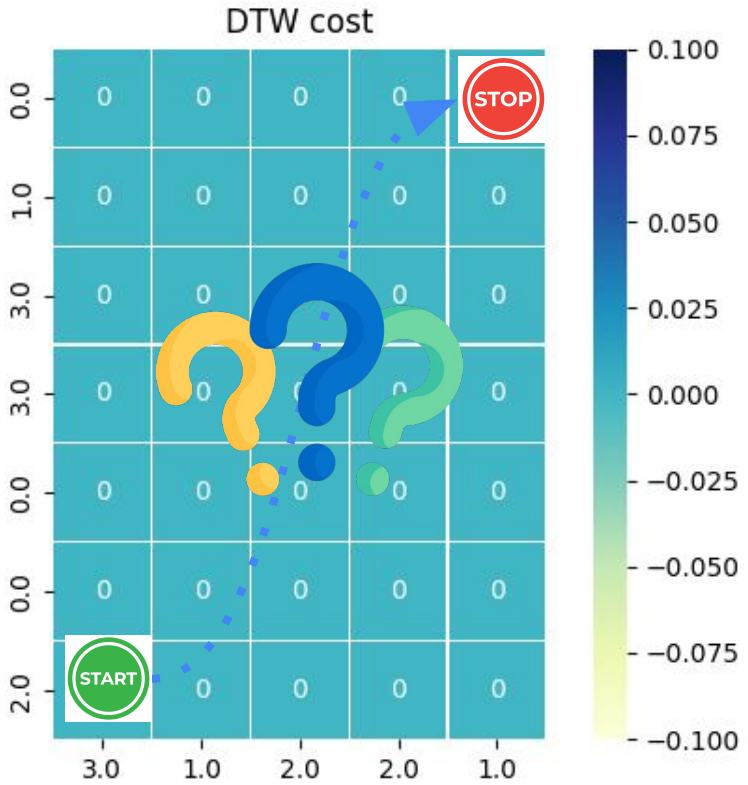


Dynamic Time Warping (DTW)

Given two sequences:

x = [3, 1, 2, 2, 1]

y = [2, 0, 0, 3, 3, 1, 0]



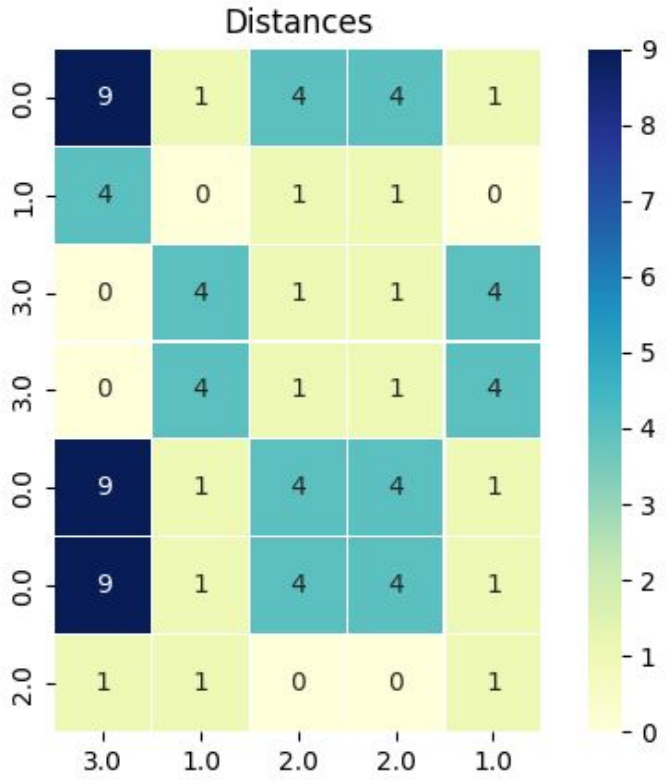
Source 1, 2

Dynamic Time Warping (DTW)

Given two sequences:

x = [3, 1, 2, 2, 1]

y = [2, 0, 0, 3, 3, 1, 0]



Dynamic Time Warping (DTW)

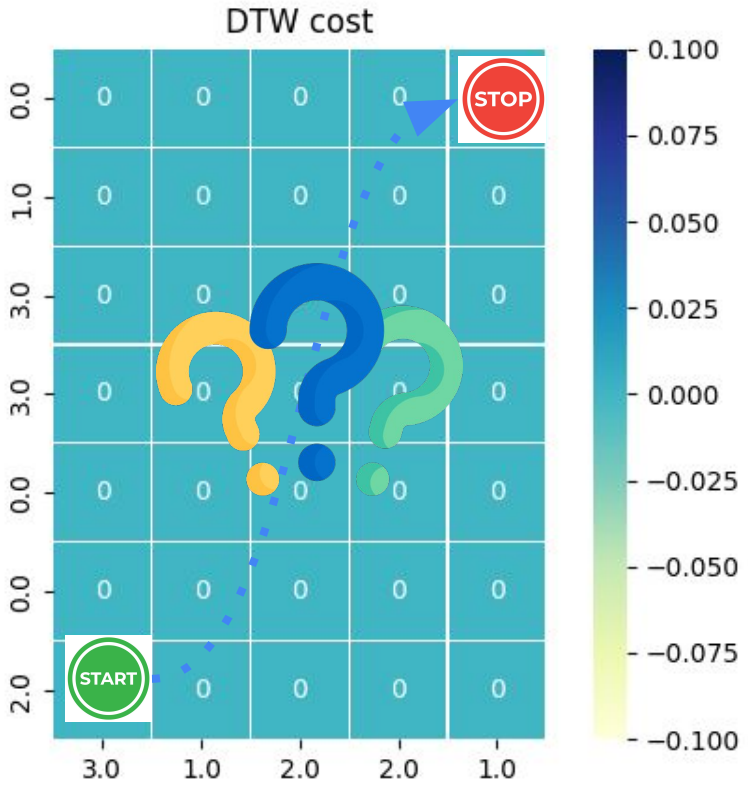
Given two sequences:

x = [3, 1, 2, 2, 1]

y = [2, 0, 0, 3, 3, 1, 0]

Constraints: Monotonic alignment

Using DP for efficiency



Source 1, 2

Dynamic Time Warping (DTW)

Given two sequences:

x = [3, 1, 2, 2, 1]

y = [2, 0, 0, 3, 3, 1, 0]

$$DTW[0,0] = d(0,0)$$

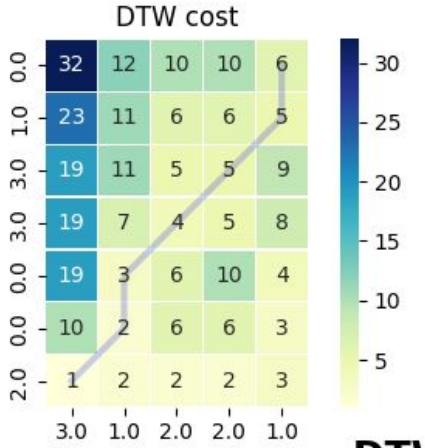
$$DTW[i,j] = d(i,j) + min \begin{bmatrix} DTW[i-1,j-1] \\ DTW[i,j-1] \\ DTW[i-1,j] \end{bmatrix}$$

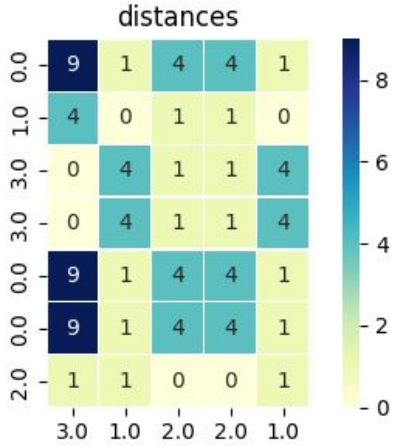
Dy

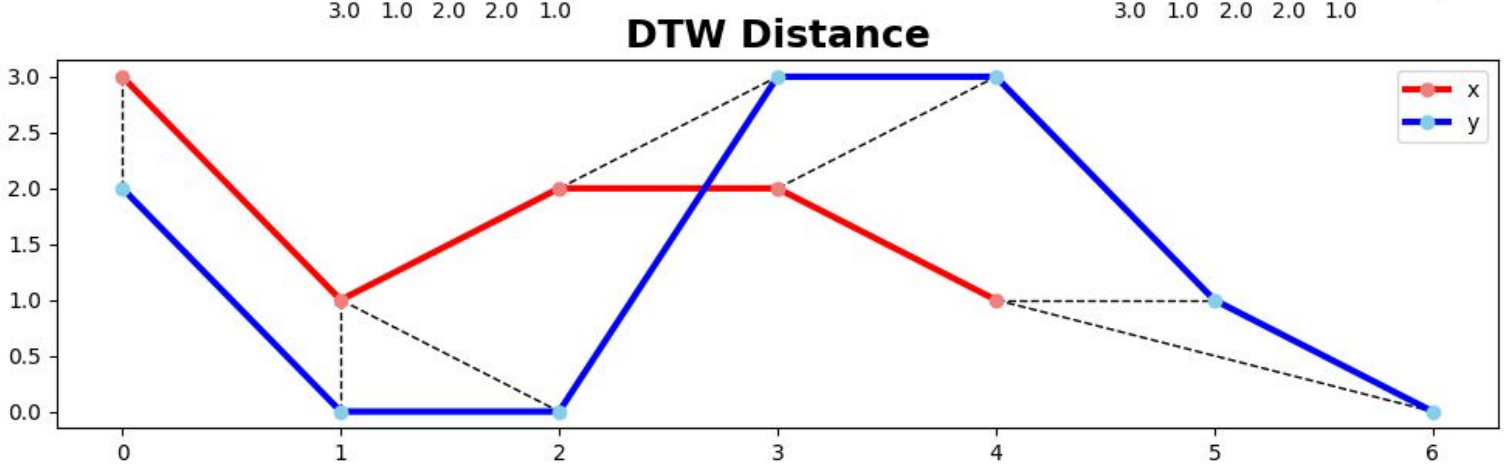
Gi

x =

y =


DTW cost
0.0 32 12 10 10 6
1.0 23 11 6 6 5
3.0 19 11 5 5 9
3.0 19 7 4 5 8
0.0 19 3 6 10 4
0.0 10 2 6 6 3
2.0 1 2 2 2 3
3.0 1.0 2.0 2.0 1.0
30 25 20 15 10 5


distances
0.0 9 1 4 4 1
1.0 4 0 1 1 0
3.0 0 4 1 1 4
3.0 0 4 1 1 4
0.0 9 1 4 4 1
0.0 9 1 4 4 1
2.0 1 1 0 0 1
3.0 1.0 2.0 2.0 1.0
8 6 4 2 0

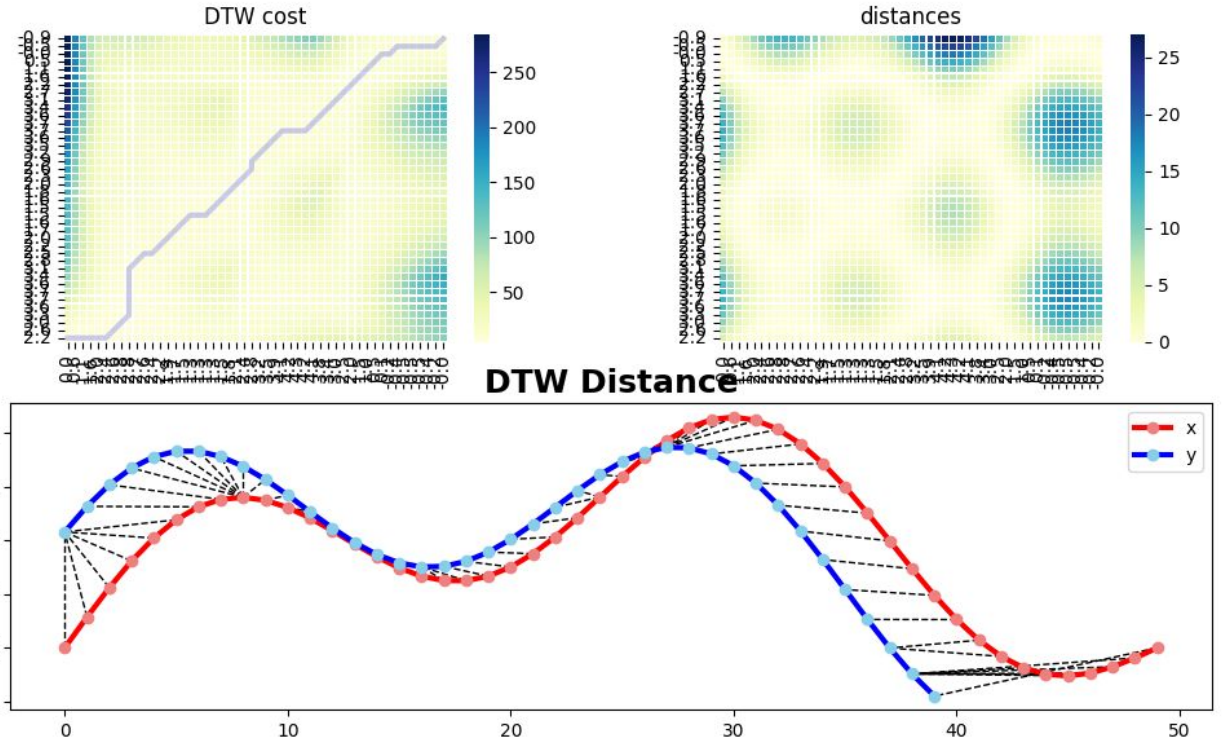

DTW Distance
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0 1 2 3 4 5 6
x
y

Dynamic Time Warping (DTW)

Given tw

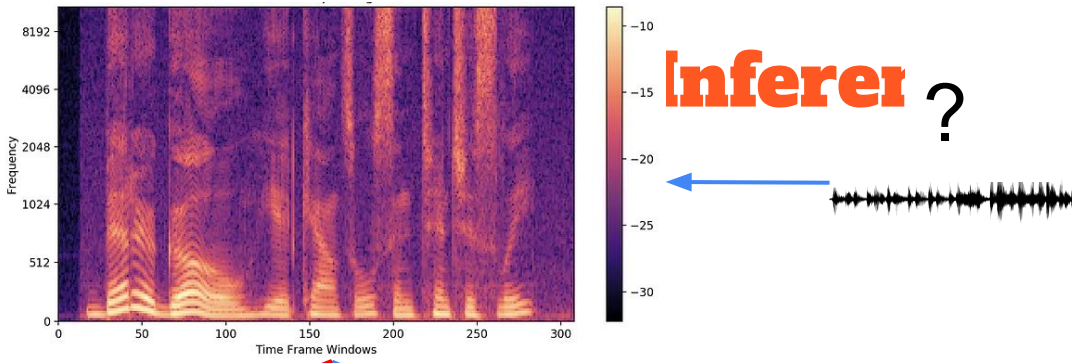
x = [3, 1, 2

y = [2, 0, (

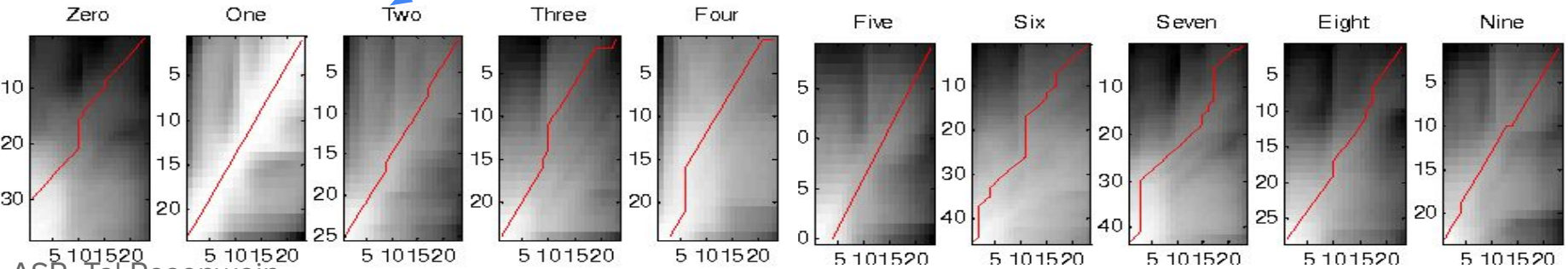


Dynamic Time Warping (DTW) - Inference

Dynamic Time W Inferer ?



$$class = arg \min_{c \in Lexicon} DTW(Query, L_C)$$



Dynamic Time Warping (DTW) - Limitations

DTW has several limitations:

- How to represent a class?
- Classifying a whole word ➔ Limited dictionary size
- Pre processing is required
- Exhaustive search
- Non parametric / non-probabilistic

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- **ASR problem formulation**
- Inference pipeline
  - Decoding
  - Language models
  - Pronunciation models
  - Feature extraction
  - Acoustic models
  - HMM

ASR Pipeline

DTW's limitations:

- Large multi-class
- Non probabilistic / learnable approach.

ASR pipeline consists of 3 main modules:

- **Acoustic model:** maps from audio signal to phonemes.
- **Language model:** computes to probability of a given word sequence.
- **Pronunciation model:** different ways of pronunciation a given word.

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W}{argmax} P(W|O)$$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W}{argmax} P(W|O)$$

Define:

1. Acoustic input O- a sequence of individual observations: $O = o_1, o_2, ..., o_t$
2. A sentence as a sequence of words: $W = w_1, w_2, ..., w_n$
3. Phoneme sequence $P = p_1, p_2, ..., p_n$, for example:
   $W: w_1 = banana,\ P: p_1 = buh, p_2 = na, p_3 = nuh,$
4. Aligned phoneme sequence $Q = q_1, q_2, ..., q_T$, for example:
   $W: w_1 = banana,\ P: p_1 = buh, p_2 = na, p_3 = nuh,\ Q: q_1 = buh,\ q_2 = buh,\ q_3 = na,\ q_4 = na, q_5 = nah$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W}{argmax} P(W|O)$$

$$\widehat{W} = \underset{W \in L}{argmax} P(W|O) = \underset{W \in L}{argmax} \frac{P(O|W)P(W)}{P(O)}$$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W}{argmax} P(W|O)$$

$$\widehat{W} = \underset{W \in L}{argmax} P(W|O) = \underset{W \in L}{argmax} \frac{P(O|W)P(W)}{P(O)}$$

$$\widehat{W} = \underset{W \in L}{argmax} P(O|W)P(W)$$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W \in L}{argmax} P(O|W)P(W)$$

$$P(O|W) = \sum_{P} P(O|P,W)P(P|W)$$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W \in L}{argmax} P(O|W)P(W)$$

$$P(O|W) = \sum_{P} P(O|P,W)P(P|W)$$

For example, The word ‘tomato can be pronounced in several ways (Google search):

- Tuh May tow (American English)
- Tuh Maa tow (British English)

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W \in L}{argmax} P(O|W)P(W)$$

$$P(O|W) = \sum_{P} P(O|P,W)P(P|W)$$

$$\approx \max_{P} P(O|P,W)P(P|W)$$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W \in L}{argmax} P(O|W)P(W)$$

$$P(O|W) = \sum_{P} P(O|P,W)P(P|W)$$

$$\approx \max_{P} P(O|P,W)P(P|W)$$

$$\approx \max_{P} P(O|P)P(P|W)$$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

$$\widehat{W} = \underset{W}{argmax} P(W|O)$$

$$\widehat{W} = \underset{W \in L}{argmax} P(O|W)P(W)$$

$$\widehat{W} = arg \max_{W} \max_{P} P(O|P)P(P|W)P(W)$$

ASR Pipeline - Problem Formulation

What is the most likely word sequence W_hat out of all sentences in the language L given some acoustic input O?

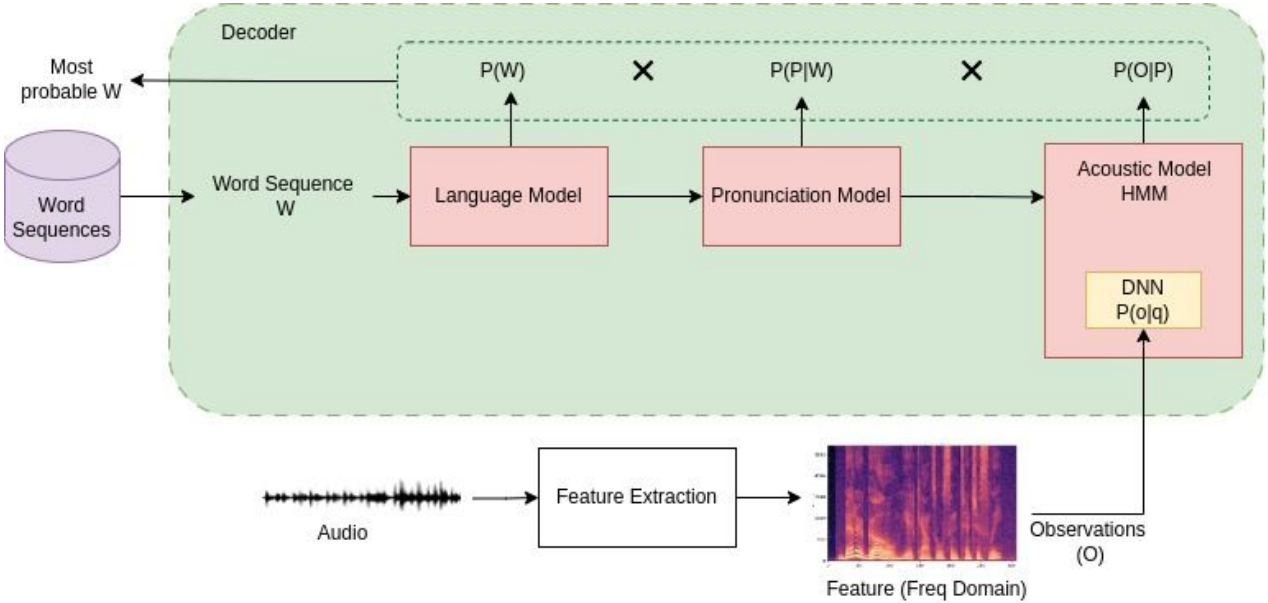
- **P(O|P)**: Acoustic model- the probability that this audio was generated by this phoneme
- **P(P|W)**: Pronunciation model- the probability that this phoneme sequence generates the word sequence.
- **P(W)**: Language model- the probability that this word sequence to be generated.

$$\hat{W} = arg \max_{W} \max_{P} P(O|P)P(P|W)P(W)$$

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- **Inference pipeline**
  - Decoding
  - Language models
  - Pronunciation models
  - Feature extraction
  - Acoustic models
  - HMM

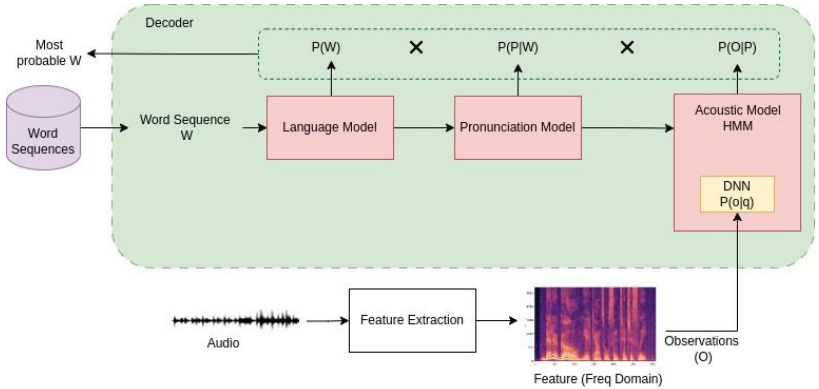
ASR Pipeline - Inference Pipeline



$$\hat{W} = arg \max_{W} \max_{P} P(O|P)P(P|W)P(W)$$

Source 1, 2

ASR Pipeline - Inference Pipeline



HMM-GMM has 5 components:

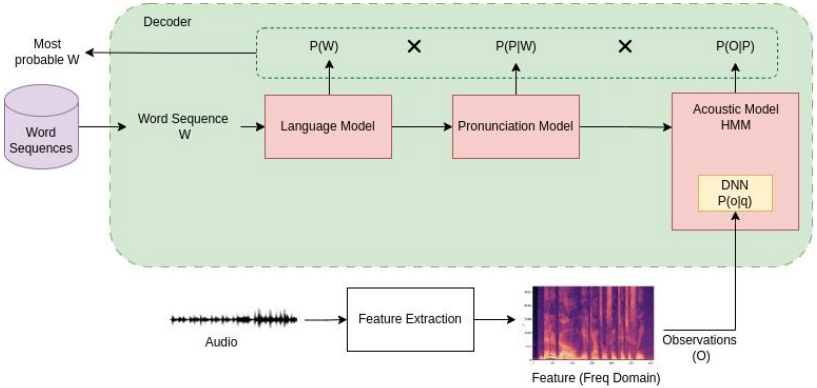
1. **Feature extraction**: from audio to spectral features
2. **Acoustic models**:
   a. GMM for computing p(o|q), where q is the HMM hidden state.
   b. HMM that wraps the GMM and outputs the p(o|p)
3. **Lexicon/Pronunciation Model**: prob. for phoneme seq given the word seq p(p|w).
4. **Language Model**: captures semantics. Usually using N-grams
5. **Decoder:** Find the most probable word sequence using DP / search algorithm

Source 1, 2

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
  - **Decoding**
  - Language models
  - Pronunciation models
  - Feature extraction
  - Acoustic models
  - HMM

ASR Pipeline - Decoding



Searching over all possible word sequences is too exhaustive

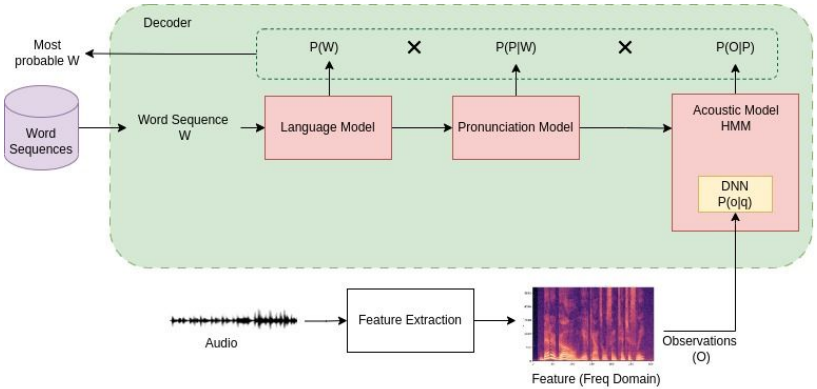
Solution: Search over paths with signal from the acoustics

Source 1, 2

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
  - Decoding
  - **Language models**
  - Pronunciation models
  - Feature extraction
  - Acoustic models
  - HMM

ASR Pipeline - Language Model

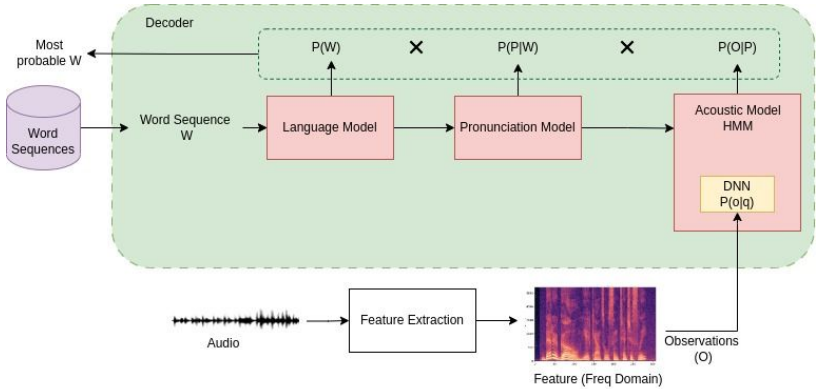


Language models are extremely important during decoding as they:

- **Enlarge the acoustic model's context and add semantics into decoding.**
- **Enforce transcript:** Speech Commands- "Hey Siri Call X"
- **Recover from acoustic model errors.**

Source 1, 2

ASR Pipeline - Language Model



Language Model **injects A-Prior information** into the decoding process, where it **enforces** (**by some extent**) the **output** sequence to **make sense**

$$P(w_1, ..., w_n) = \prod_{i=1}^{m} P(w_i | w_1, ..., w_{i-1})$$

Causal

Source 1, 2

ASR Pipeline - Language Model

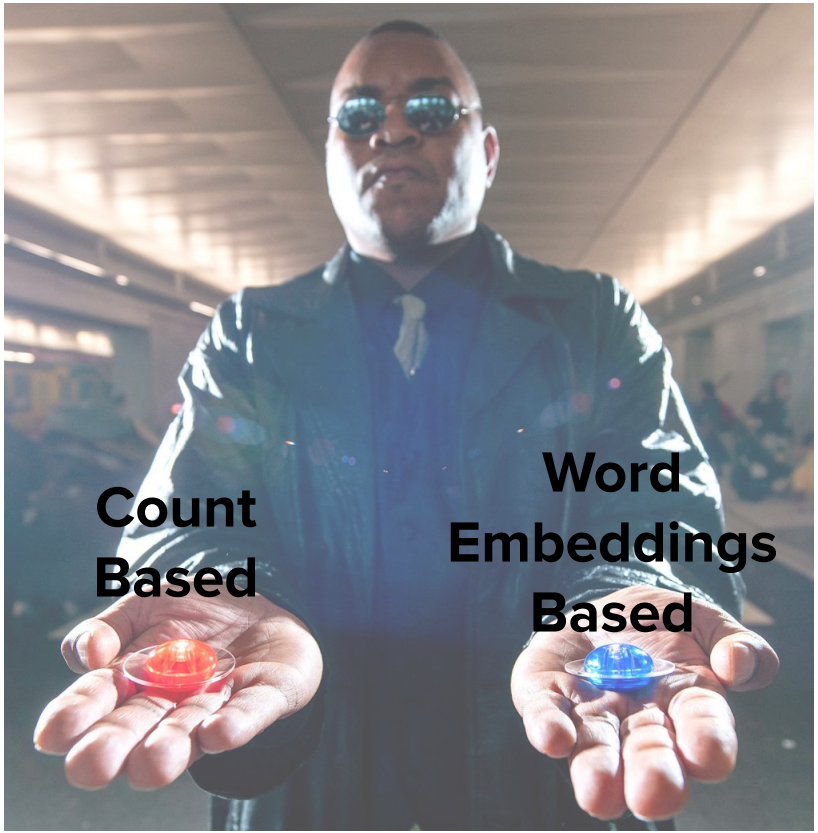


Image source

ASR Pipeline - Language Model - Count Based

“What You See Is What You Get”

All Out Of Vocabulary (OOV) Have The Same Prob.

***P*(“bla bla fds”) = *P*(“gotten into his”) = *P*(“got iphone X”)**

Lookup Tables (GBs)

Main tools:

- KenLM: https://github.com/kpu/kenlm
- SRILM: http://www.speech.sri.com/projects/srilm/

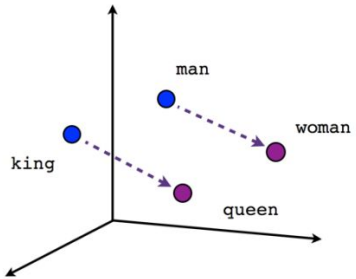
-1.5299898	looking into his
-1.6576592	get into his
-1.5772834	way into his
-1.5478122	them into his
-1.4195554	got into his
-0.5696159	year into his
-0.49294835	reached into his
-1.7273724	went into his
-1.0968437	well into his
-1.4932973	back into his

ASR Pipeline - Language Model - Embed. Based

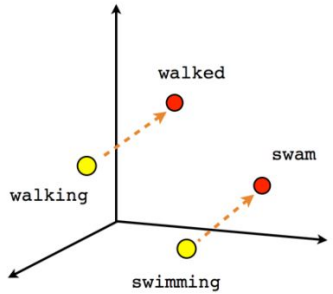
ASR Pipeline - Language Model - Embed. Based

Word2Vec: Discrete words into continuous (embedding) space.

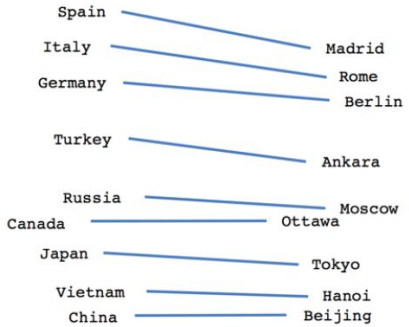
Can improves performances on OOV words (generalization).



Male-Female



Verb tense



Country-Capital

king − man + woman ~= queen

Image source 1 2

ASR Pipeline - Language Model - Embed. Based

Contextualized Word Embedding: Words in different contexts / Homonyms (words that have multiple meanings) should have different representations based on their context:

- I eat a **date** every day
- Let's go on a **date** sometime

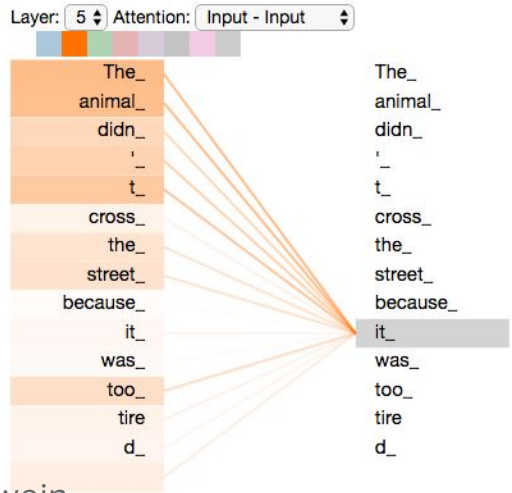
Pioneering works:

- **ELMo:** 2018 Deep contextualized word representations.
- **BERT:** 2019 BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

ASR Pipeline - Language Model - Embed. Based

Attention is the mechanism that takes into account the context of a given word, but doesn't learn a specific weight for the exact sequence.

It is affected by 'the animal'.



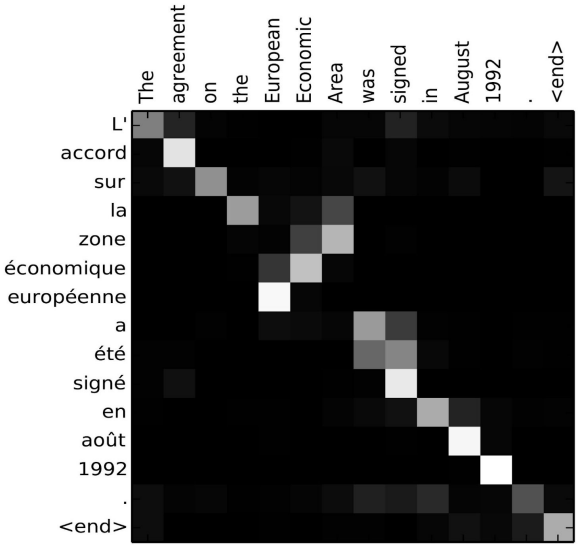
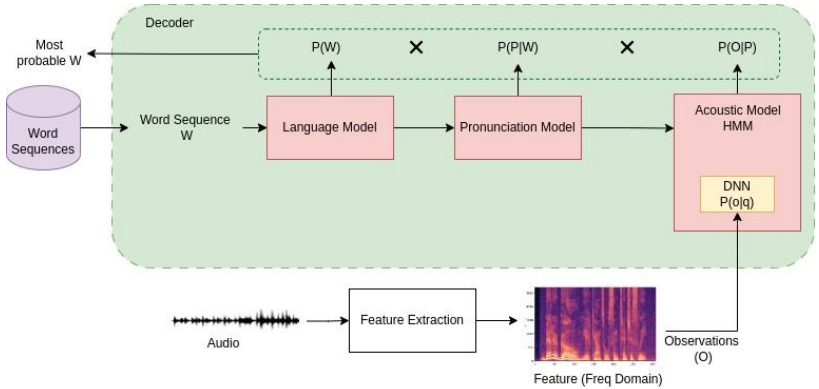


Image source

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
  - Decoding
  - Language models
  - **Pronunciation models**
  - Feature extraction
  - Acoustic models
  - HMM

ASR Pipeline - Lexicon/Pronunciation Model



P(p|w)

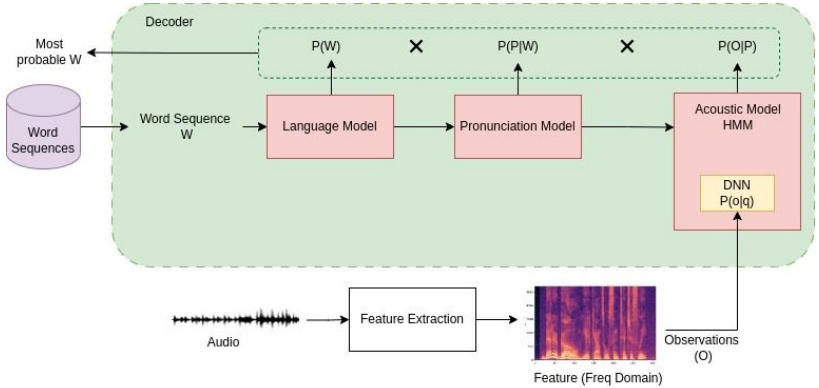
```
ADVANCING       AH D V AE N S IH NG
ADVANI  AE D V AA N IY
ADVANTA AE D V AE N T AH
ADVANTA'S       AE D V AE N T AH Z
ADVANTA'S(2)    AH D V AE N T AH Z
ADVANTA(2)      AH D V AE N T AH
ADVANTAGE       AE D V AE N T IH JH
ADVANTAGE(2)    AH D V AE N T IH JH
ADVANTAGE(3)    AE D V AE N IH JH
ADVANTAGE(4)    AH D V AE N AH JH
ADVANTAGED      AE D V AE N T IH JH D
ADVANTAGED(2)   AH D V AE N T IH JH D
ADVANTAGED(3)   AE D V AE N IH JH D
ADVANTAGED(4)   AH D V AE N IH JH D
ADVANTAGEOUS    AE D V AH N T EY JH AH S
ADVANTAGES      AE D V AE N T IH JH IH Z
ADVANTAGES(2)   AH D V AE N T IH JH IH Z
ADVANTAGES(3)   AE D V AE N IH JH IH Z
ADVANTAGES(4)   AH D V AE N IH JH IH Z
ADVANTEST       AE D V AE N T AH S T
ADVANTEST(2)    AH D V AE N T AH S T
ADVECTION       AE D V EH K SH AH N
```

Source 1, 2, 3

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
  - Decoding
  - Language models
  - Pronunciation models
  - **Feature extraction**
  - Acoustic models
  - HMM

ASR Pipeline - Feature Extraction



MFCCs / Mel Filter Banks, Raw audio

MFCCs - Mel Frequency Cepstral Coefficients

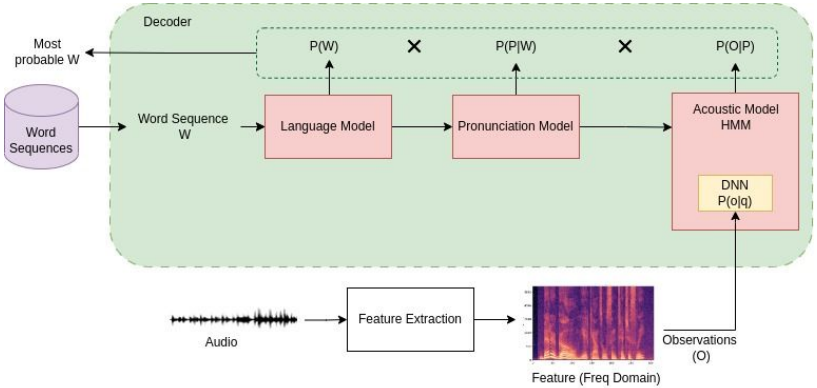
DC Removal → Pre-Emphasis Filter (HPF) → Power Spectrum Calculation (STFT) → Apply Mel Filter Banks → Log → Discrete Cosine Transform → Liftering

Source 1, 2

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
  - Decoding
  - Language models
  - Pronunciation models
  - Feature extraction
  - **Acoustic models**
  - HMM

ASR Pipeline - Acoustic Model

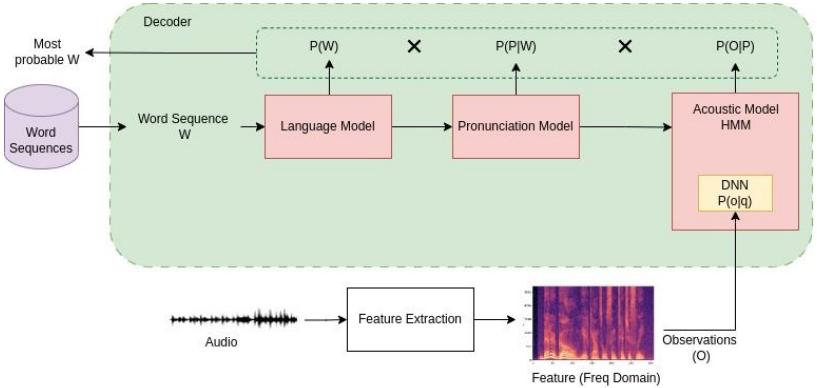


$$P(O|P)$$

The probability of observing the feature vector *O* given the phoneme sequence *P*

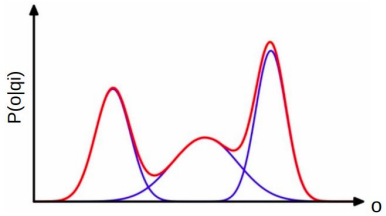
Source 1, 2

ASR Pipeline - Acoustic Model - Inference



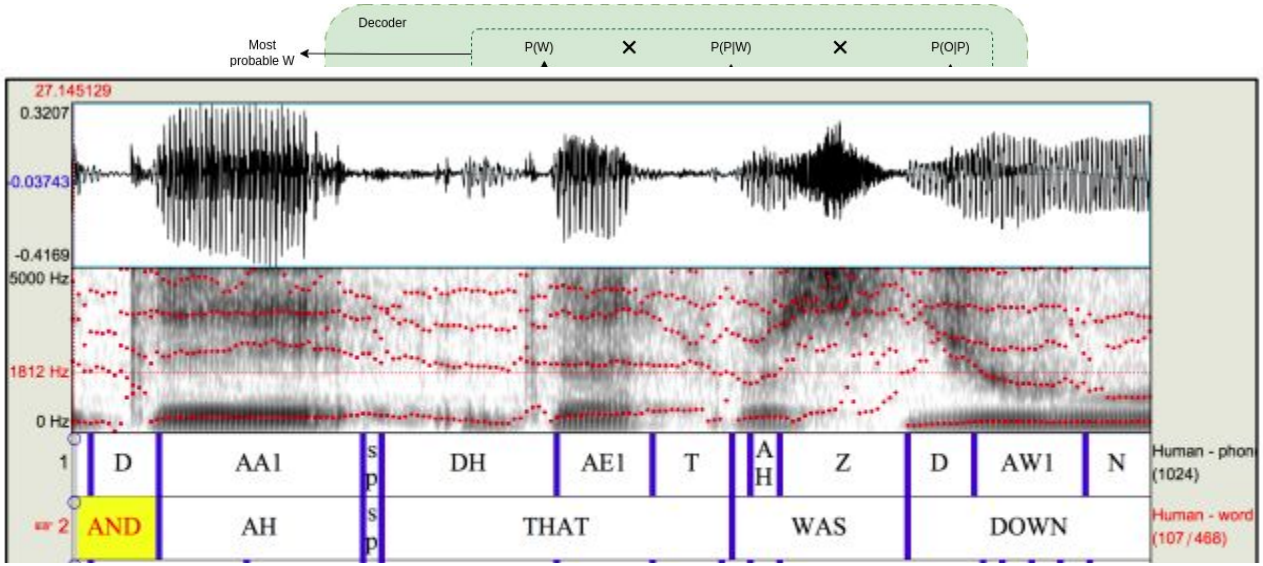
$$P(O|P)$$

- A single model for all states (multiclass over all states Q)- p(q_i|o).
- A model per each state- p(o|q_i).



Source 1, 2

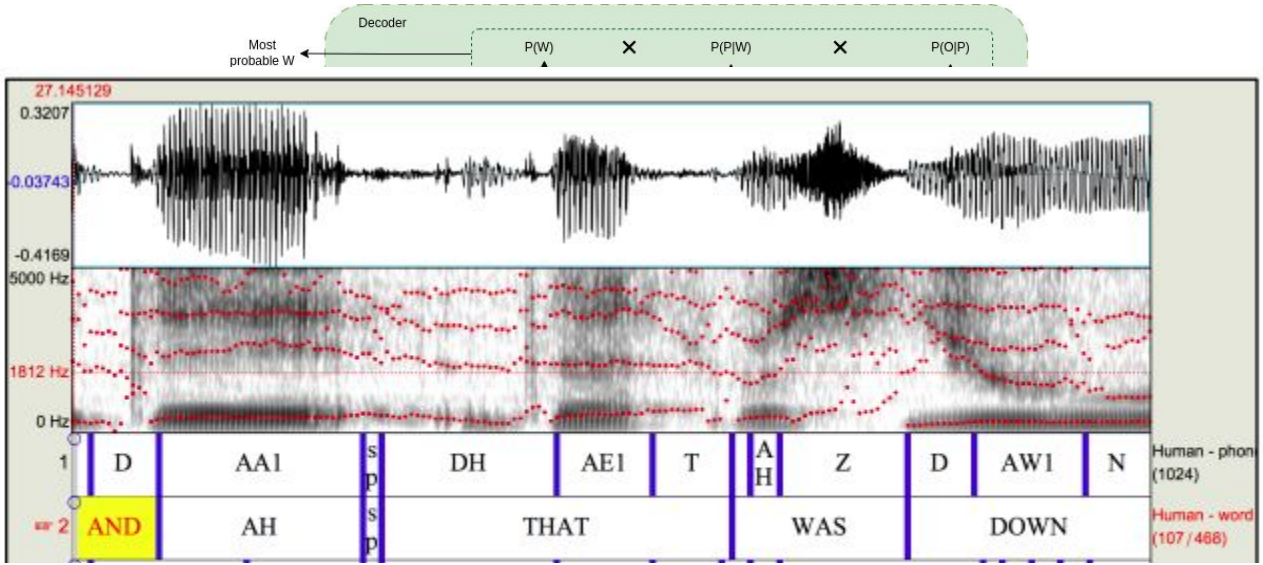
ASR Pipeline - Acoustic Model - Inference



Let’s define an alignment Q

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Inference



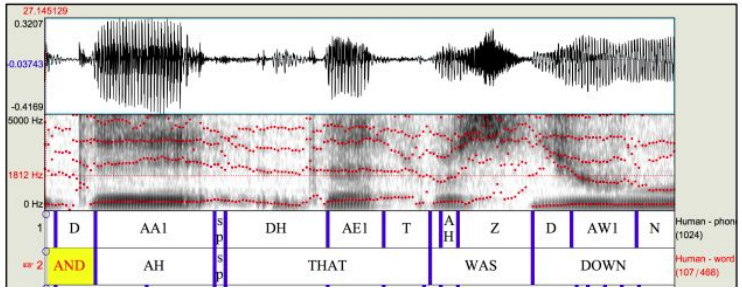
Let's define an alignment Q

P is the phoneme sequence and Q is the **aligned** phoneme sequence.

- For example: p=[cat]. len(p) = 3.
- q=[cccaaaaaa.....ttttt]. len(q) = T.

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Inference



$$P(O|P)$$

P is the phoneme sequence and Q is the **aligned** phoneme sequence.

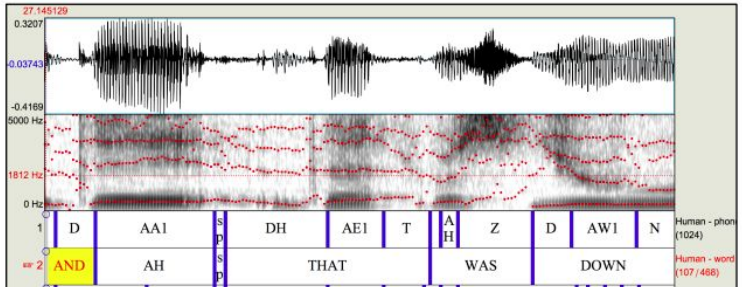
- For example: p=[cat]. len(p) = 3.
- q=[cccaaaaaa.....ttttt]. len(q) = T.

Had we known the alignment Q, the probability of the observation given the alignment can be calculated as follows:

$$P(O|Q) = \prod_{t=1}^{T} P(o_t|q_t)$$

Source 1

ASR Pipeline - Acoustic Model - Inference



$$P(O|P)$$

P is the phoneme sequence and Q is the **aligned** phoneme sequence.

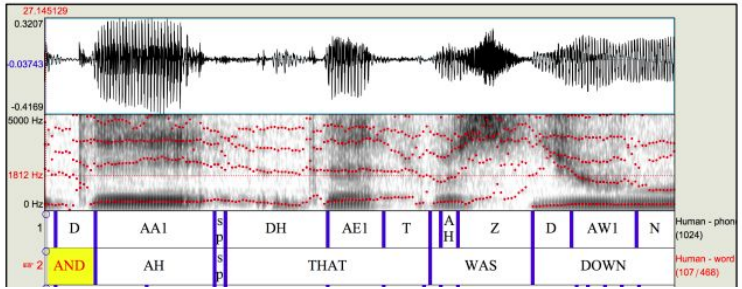
- For example: p=[cat]. len(p) = 3.
- q=[cccaaaaaa.....ttttt]. len(q) = T.

In reality we do not know the alignment, hence:

$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P)$$

Source 1

ASR Pipeline - Acoustic Model - Inference



$$P(O|P)$$

P is the phoneme sequence and Q is the **aligned** phoneme sequence.

- For example: p=[cat]. len(p) = 3.
- q=[cccaaaaaa.....ttttt]. len(q) = T.

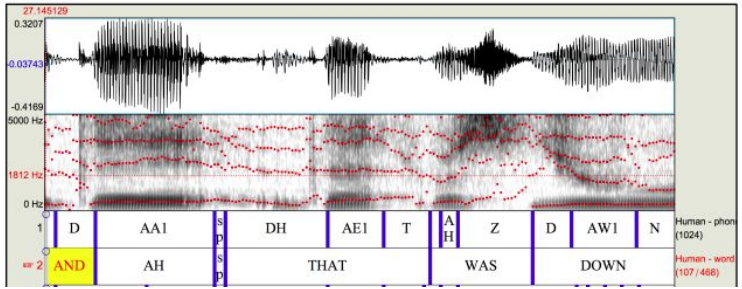
In reality we do not know the alignment, hence: $$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P)$$

Applying First order markov assumption:

$$P(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ..., q_1 = S_l) = P(q_t = S_j | q_{t-1} = S_i)$$

Source 1

ASR Pipeline - Acoustic Model - Inference



$$P(O|P)$$

P is the phoneme sequence and Q is the **aligned** phoneme sequence.

- For example: p=[cat]. len(p) = 3.
- q=[cccaaaaaa.....ttttt]. len(q) = T.

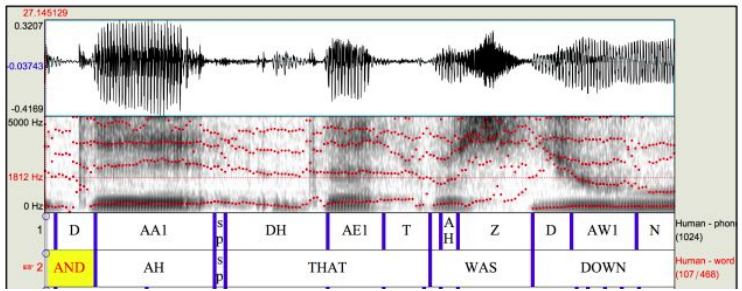
In reality we do not know the alignment, hence:

$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P)$$

Applying first order markov assumption: $P(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ..., q_1 = S_l) = P(q_t = S_j | q_{t-1} = S_i)$

Source 1

ASR Pipeline - Acoustic Model - Inference



$$P(O|P)$$

P is the phoneme sequence and Q is the **aligned** phoneme sequence.

- For example: p=[cat]. len(p) = 3.
- q=[cccaaaaaa.....ttttt]. len(q) = T.

In reality we do not know the alignment, hence:

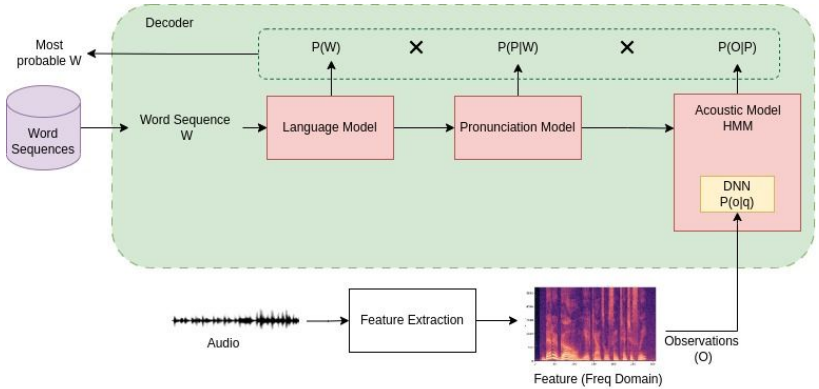
$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P)$$

Applying first order markov assumption: $P(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ..., q_1 = S_l) = P(q_t = S_j | q_{t-1} = S_i)$

Results in:

$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P) = \sum_{Q \in A(P)} \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1}, P)$$

ASR Pipeline - Acoustic Model - Training



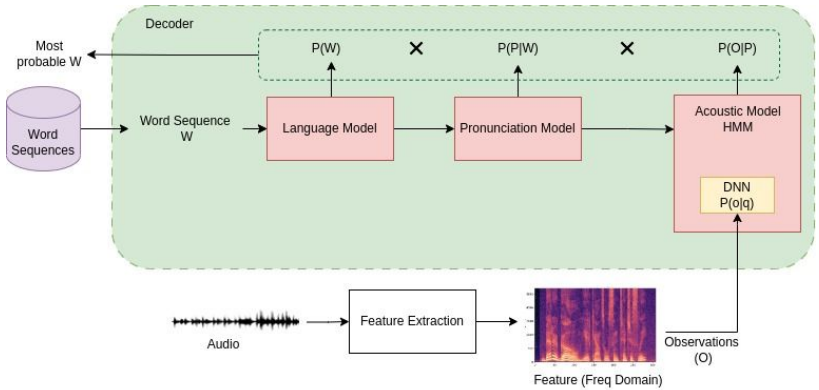
$$P(O|P)$$

Components to be trained:

- Acoustic model
- Transition probability (HMM)

$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P) = \sum_{Q \in A(P)} \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1}, P)$$

ASR Pipeline - Acoustic Model - Training



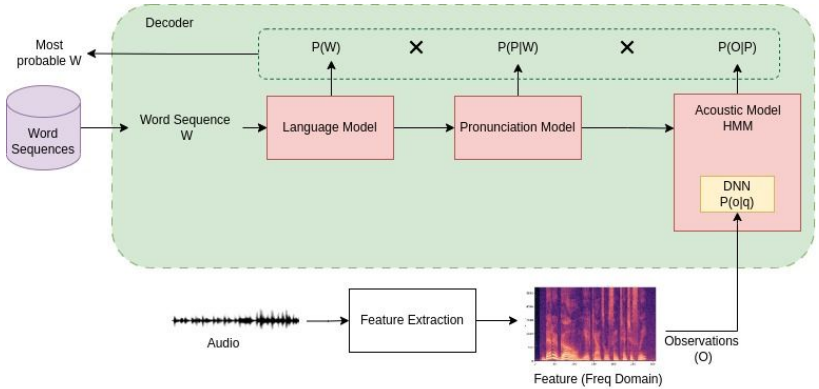
$$P(O|P)$$

Components to be trained:

- **Acoustic model**
- Transition probability (HMM)

$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P) = \sum_{Q \in A(P)} \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1}, P)$$

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

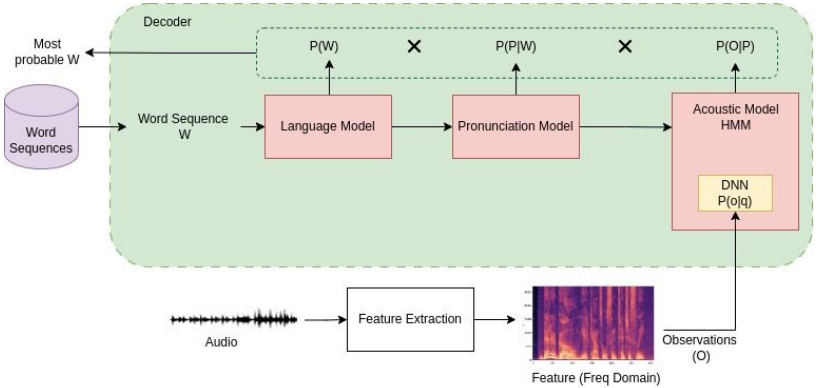
Acoustic model

P(o|q) is trained in a supervised manner

- o and q are known/labeled for each frame.
- Collected manually and/or using Force alignment

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Acoustic model

P(o|q) is trained in a supervised manner

- o and q are known/labeled for each frame.
- Collected **manually** and/or using Force alignment

“She had your dark suit in greasy wash water all year”

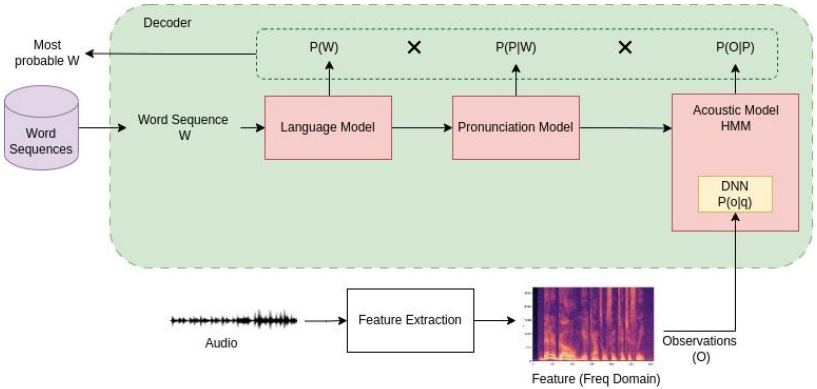
```
0 3050 h#
3050 4559 sh
4559 5723 ix
5723 6642 hv
6642 8772 eh
8772 9190 dcl
9190 10337 jh
10337 11517 ih
11517 12500 dcl
12500 12640 d
12640 14714 ah
14714 15870 kcl
15870 16334 k
16334 18088 s
18088 20417 ux
20417 21199 q
21199 22560 en
22560 22920 gcl
```

```
22920 23271 g
23271 24229 r
24229 25566 ix
25566 27156 s
27156 28064 ix
28064 29660 w
29660 31719 ao
31719 33360 sh
33360 33754 epi
33754 34715 w
34715 36080 ao
36080 36326 dx
36326 37556 axr
37556 39561 ao
39561 40313 l
40313 42059 y
42059 43479 ih
43479 44586 axr
44586 46720 h#
```

Source 1, 2, 3

Timit

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Acoustic model

P(o|q) is trained in a supervised manner

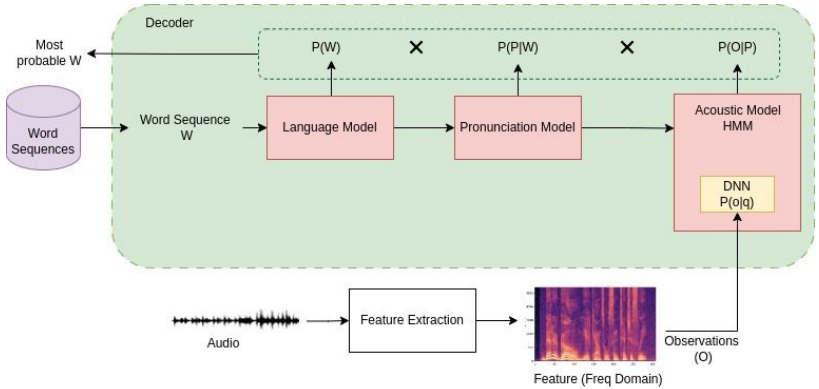
- o and q are known/labeled for each frame.
- Collected **manually** and/or using Force alignment

Link

"MARGUERITE TO BE UNABLE TO LIVE APART FROM ME IT WAS THE DAY AFTER THE EVENING WHEN SHE CAME TO SEE ME THAT I SENT HER MANON LESCAUT FROM THAT TIME SEEING THAT I COULD NOT CHANGE MY MISTRESS'S LIFE I CHANGED MY OWN" - Librispeech

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

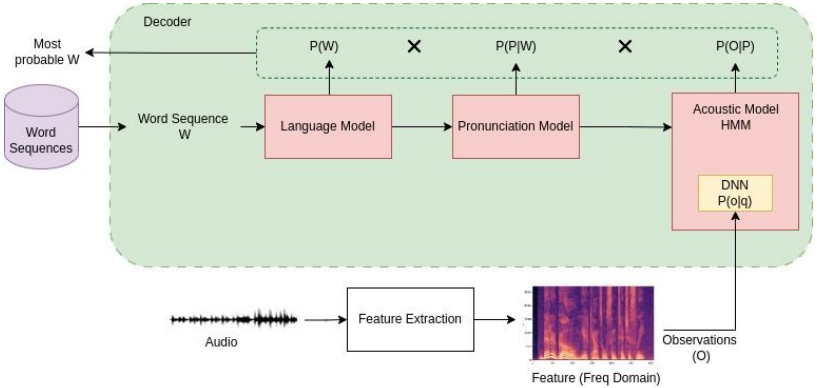
Acoustic model

P(o|q) is trained in a supervised manner

- o and q are known/labeled for each frame.
- Collected manually and/or using **Force alignment**

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

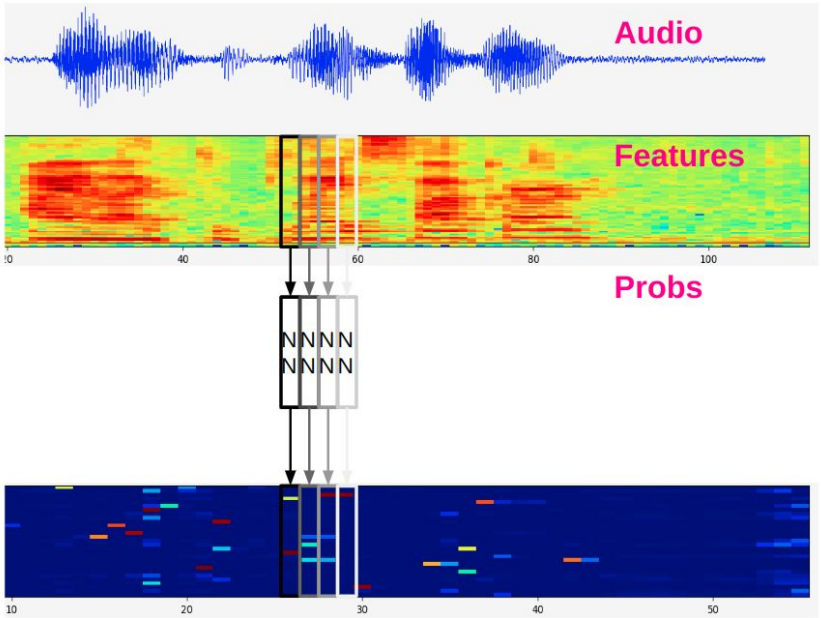
Acoustic model

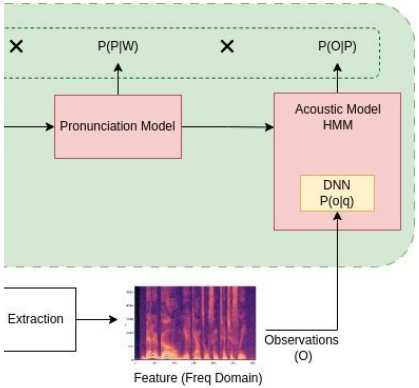
P(o|q) is trained in a supervised manner

Is one frame enough for classifying a phoneme?

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training





$(O|P)$

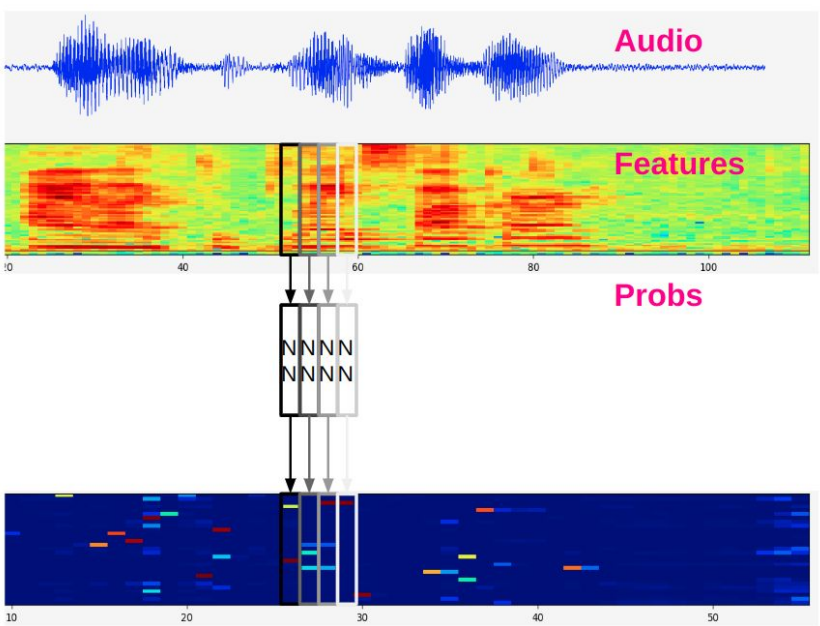
Ac

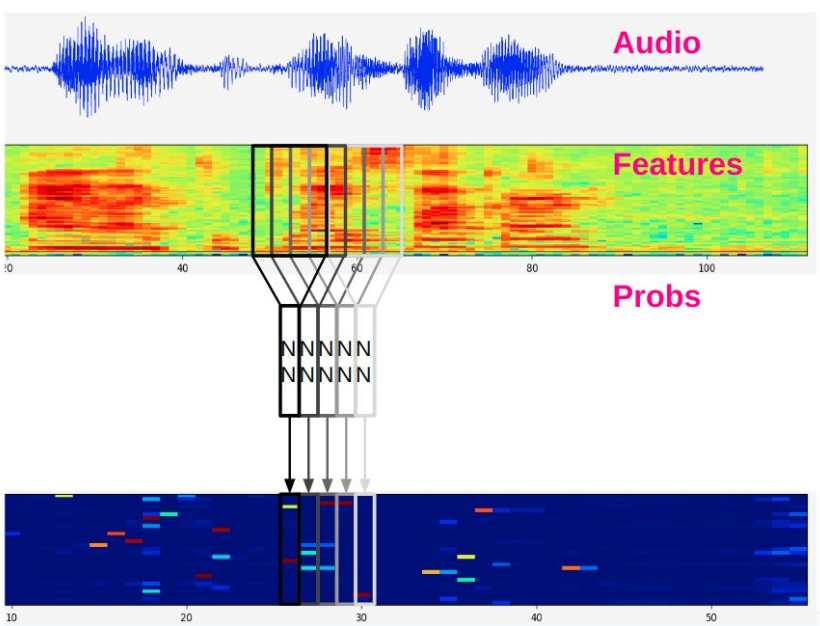
P(

Is ne?

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training





$(O|F$

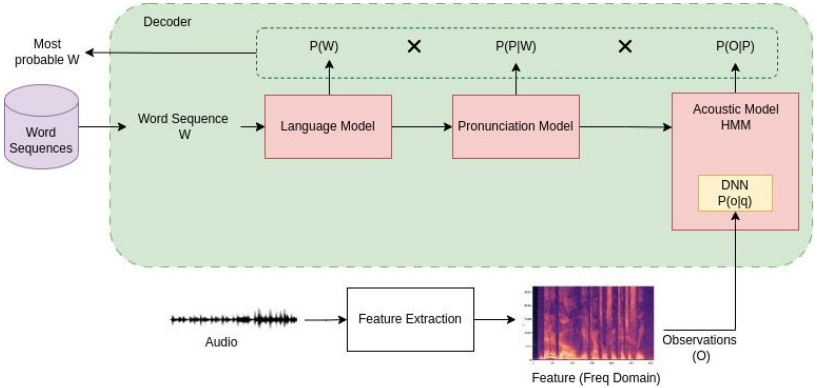
Ac

P(

Is ne?

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Acoustic model

P(o|q) is trained in a supervised manner

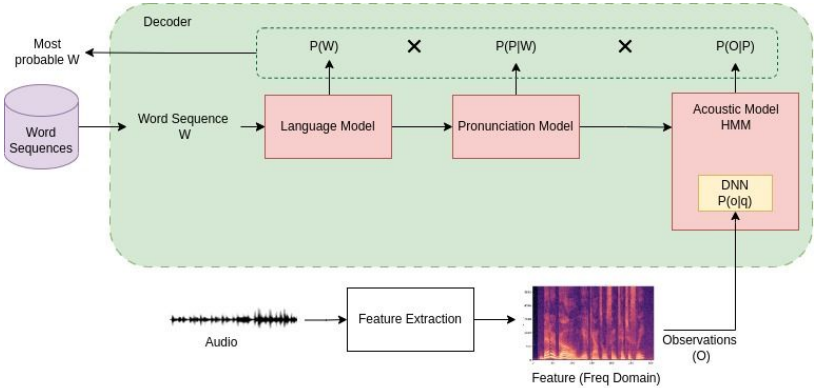
Is one frame enough for classifying a phoneme?

A single classifier for all phonemes (states)- discriminative model

$$L_\Theta(Q|O) = \sum_t -log(P(q = q_t|O_t))$$

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Acoustic model

P(o|q) is trained in a supervised manner

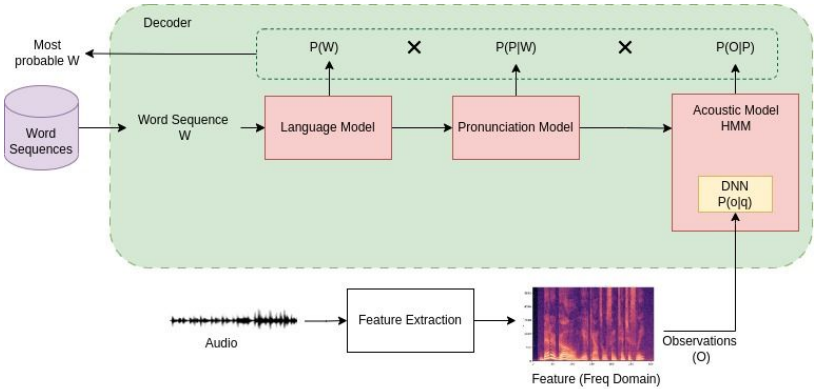
Is one frame enough for classifying a phoneme?

A single classifier for all phonemes (states)- discriminative model

$$L_\Theta(Q|O) = \sum_t -log(P(q = q_t|O_t)) \qquad P(O|Q) = \frac{P(Q|O)P(O)}{P(Q))}$$

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Components to be trained:

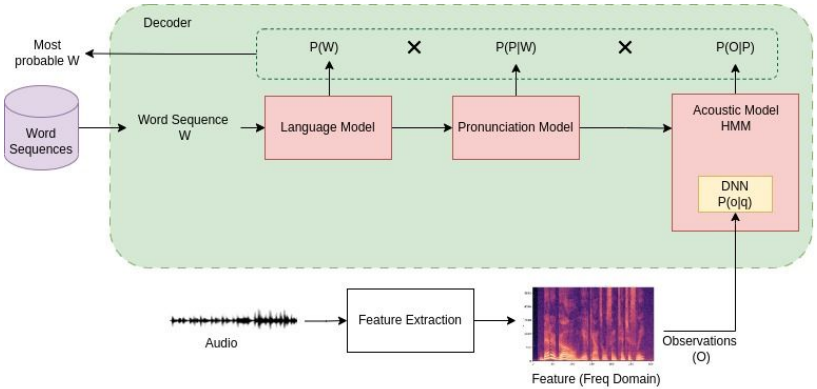
- **Acoustic model**
- Transition probability (HMM)

$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P) = \sum_{Q \in A(P)} \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1}, P)$$

Outline

- History
- ASR as a system
- Datasets & Evaluation metrics
- ASR levels of difficulty
- Dynamic time warping (DTW)
- ASR problem formulation
- Inference pipeline
  - Decoding
  - Language models
  - Pronunciation models
  - Feature extraction
  - Acoustic models
  - **HMM**

ASR Pipeline - Acoustic Model - Training



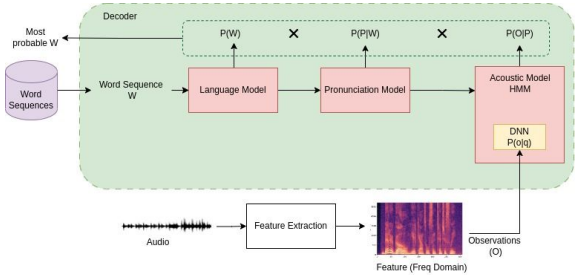
$$P(O|P)$$

Components to be trained:

- Acoustic model
- **Transition probability (HMM)**

$$P(O|P) = \sum_{Q \in A(P)} P(O|Q)P(Q|P) = \sum_{Q \in A(P)} \prod_{t=1}^{T} P(o_t|q_t)P(q_t|q_{t-1}, P)$$

ASR Pipeline - Acoustic Model - Training



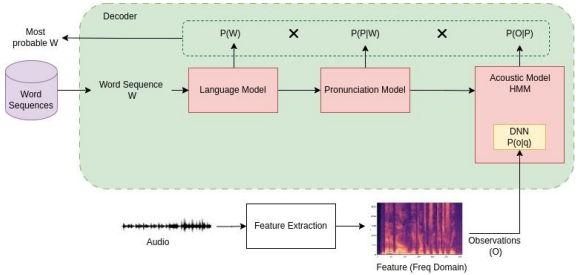
$$P(O|P)$$

First-Order **Observable** Markov Chain

State weather: $Q = q_1, q_2, \ldots, q_N$; the state at time t is q_t

Source 1, 2

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

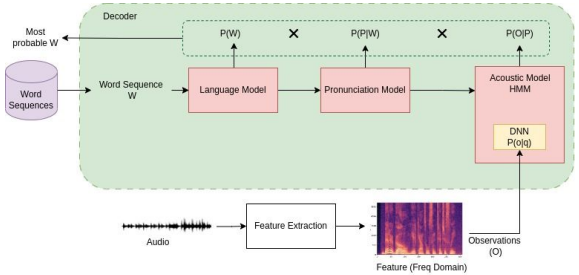
First-Order **Observable** Markov Chain

Current state depends on **previous state only** (not on the entire state history).

$$P(q_t|q_{t-1}, q_{t-2}.., q_1) = P(q_t|q_{t-1})$$

Source 1, 2

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

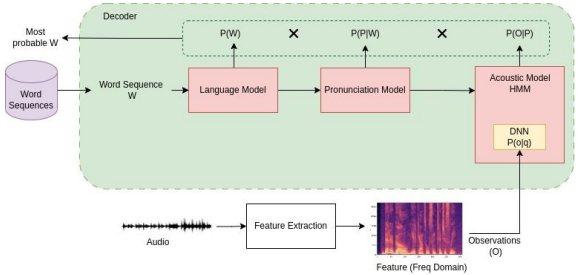
First-Order **Observable** Markov Chain

Current state depends on **previous state only** (not on the entire state history).

Add state transitions

Source 1, 2

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

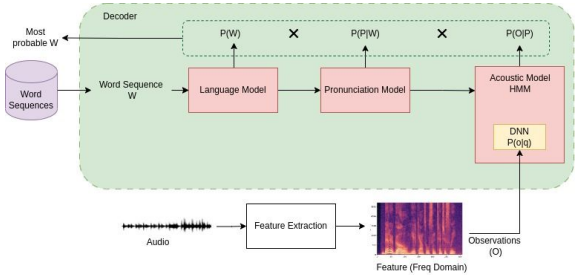
First-Order **Observable** Markov Chain

Current state depends on **previous state only** (not on the entire state history).

Add state transitions & transitions probabilities

Source 1, 2

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

First-Order **Observable** Markov Chain

Current state depends on **previous state only** (not on the entire state history).

Add state transitions & transitions probabilities

The set of transition probabilities is annotated as $A = a_{01}, a_{02}, ..., a_{n1} ..., a_{nn}$, where a_{ij} is defined as the probability of transitioning from state i to state j:

$$a_{ij} = P(q_t = j | q_{t-1} = i)$$

Source 1, 2

ASR Pipeline - Acoustic Model - Training

t

t-1

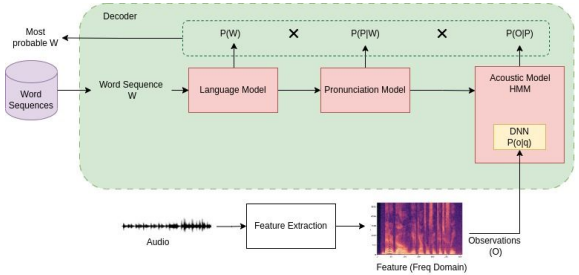
	1	2	3
1	0.5	0.3	0.2
2	0.4	0.2	0.4
3	0.0	0.3	0.7

The set of transition probabilities is annotated as $A = a_{01}, a_{02}, \ldots, a_{n1} \ldots, a_{nn}$, where a_{ij} is defined as the probability of transitioning from state i to state j:

$$a_{ij} = P(q_t = j | q_{t-1} = i) \qquad \sum_{j=1}^{N} a_{ij} = 1; \quad 1 \le i \le N$$

Source 1, 2

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

First-Order **Observable** Markov Chain

Current state depends on **previous state only** (not on the entire state history).

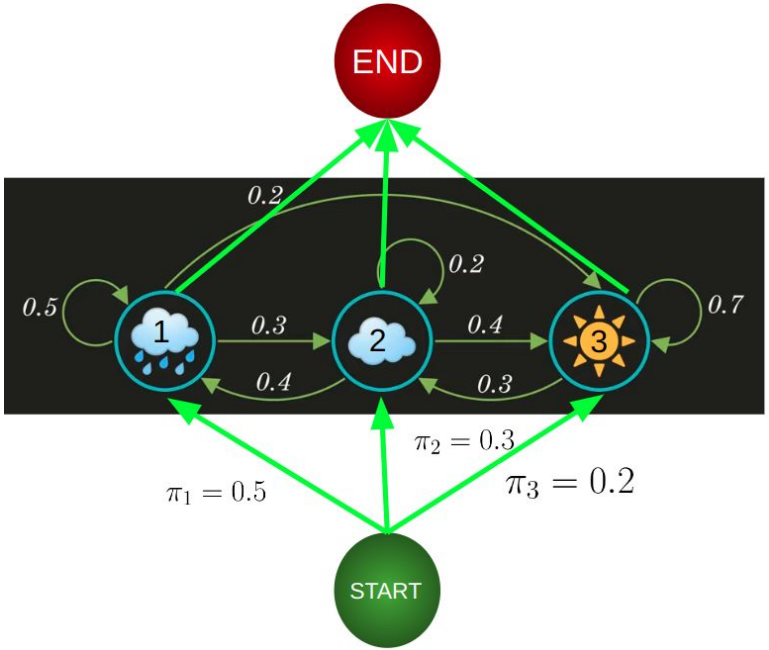
Add state transitions & transitions probabilities

First-Order Observable Markov Chain has distinguished **start** and **end** states

$$\pi_i = P(q_1 = i) \quad 1 \le i \le N \qquad \sum_{j=1}^{N} \pi_j = 1$$

Source 1, 2

ASR Pipeline - Acoustic Model - Training



t

t-1

	1	2	3
1	0.5	0.3	0.2
2	0.4	0.2	0.4
3	0.0	0.3	0.7

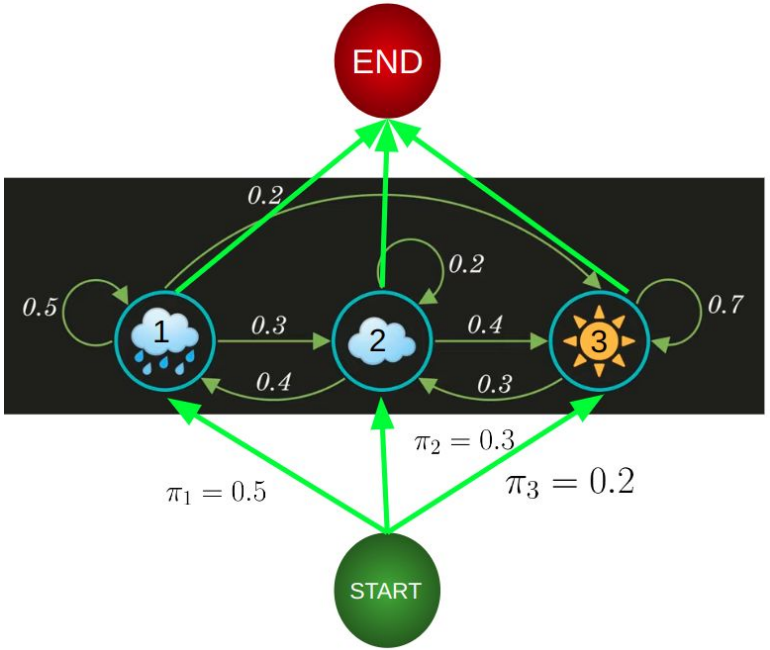
π

$$\begin{bmatrix} 0.5 \\ 0.3 \\ 0.2 \end{bmatrix}$$

$$\pi_i = P(q_1 = i) \quad 1 \le i \le N \qquad \sum_{j=1}^{N} \pi_j = 1$$

Source 1, 2

ASR Pipeline - Acoustic Model - Training



t

t-1

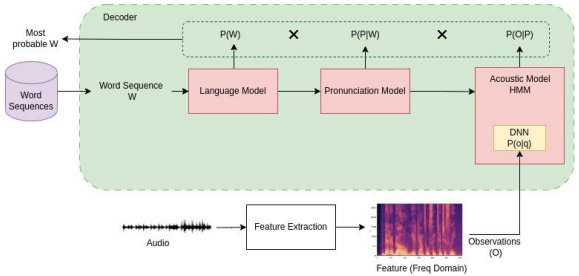
	1	2	3
1	0.5	0.3	0.2
2	0.4	0.2	0.4
3	0.0	0.3	0.7

$$\pi = \begin{bmatrix} 0.5 \\ 0.3 \\ 0.2 \end{bmatrix}$$

The probability of the trajectory 1, 2, 3 ,1

$$P(1,2,3,1) = \pi_1 * a_{21} * a_{32} * a_{13} = 0.5 * 0.3 * 0.4 * 0.0 = 0.0$$

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

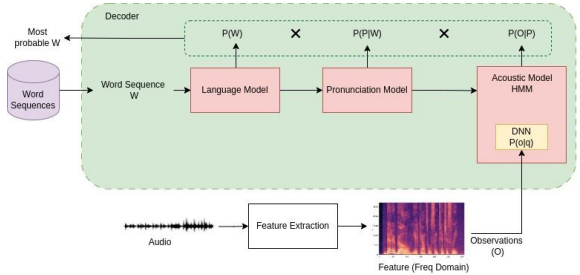
Problem:

- States are observable in Markov chains ➔ output symbols == state symbols.
- This is not the case in speech recognition. We can observe (hear) the audio, but the transcript (phoneme states) is ‘hidden’ from us.

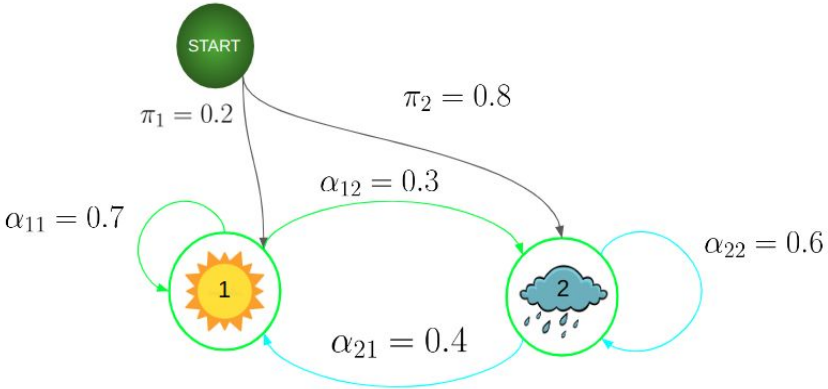
Solution: extend the markov chain to the **Hidden Markov Model** (HMM) where the states are hidden, meaning we can observe something related to the state, but we cannot access the state itself.

Source 1, 2

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$



$$a_{ij} = P(q_t = j | q_{t-1} = i)$$

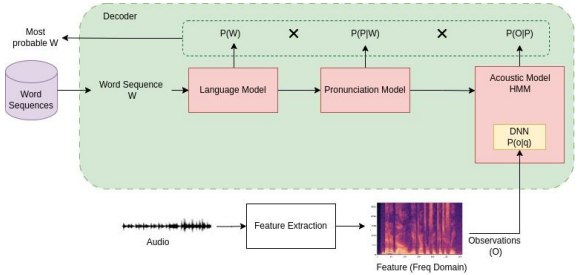
t π

t-1

$$\begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix} \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix}$$

Source 1, 2

ASR Pipeline - Acoustic Model - Training



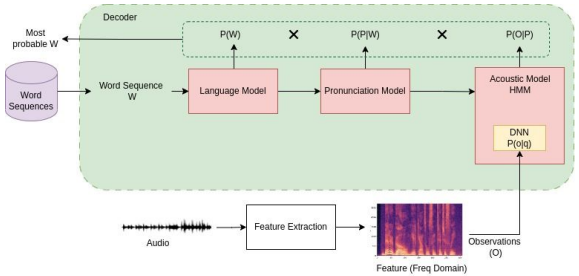
$$P(O|P)$$

We are **not exposed** to the **weather,** but we **do know how many ice-creams** Jack ate every day

- Hidden States: weather condition: hot / cold
- Observations: number of ice-creams he ate.

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

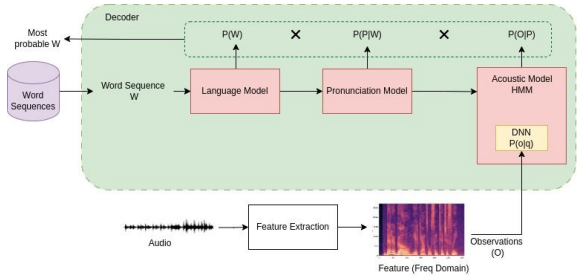
We are **not exposed** to the **weather,** but we **do know how many ice-creams** Jack ate every day

- Hidden States: weather condition: hot / cold
- Observations: number of ice-creams he ate.

Our job: figure out how is the weather (hidden states)- cold / hot, based on the number of ice-creams (observations) that he ate.

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

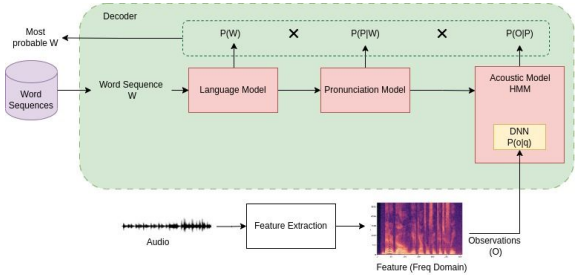
Define emission probability:

$$b_i(t) = P(o_t|q_t = i)$$

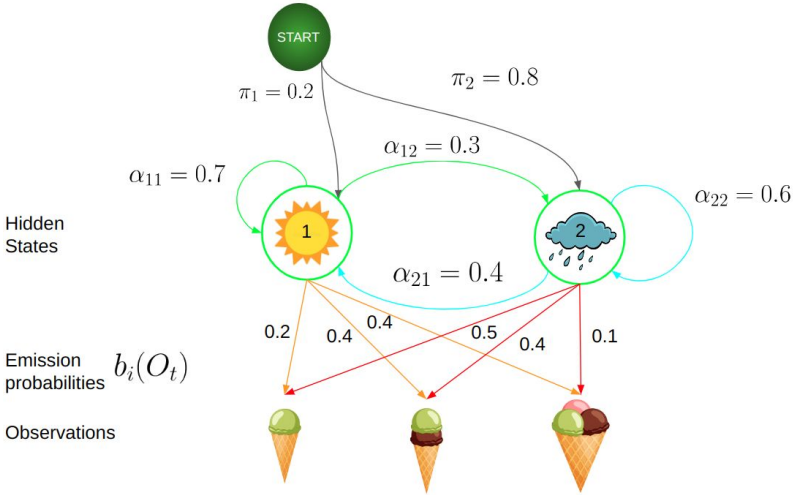
Emission probabilities are **stochastic** and **NOT deterministic**

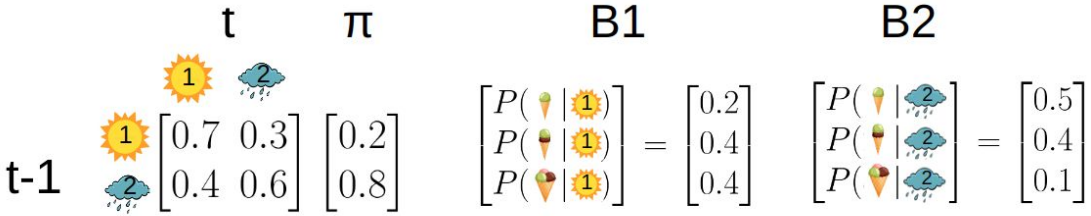
Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

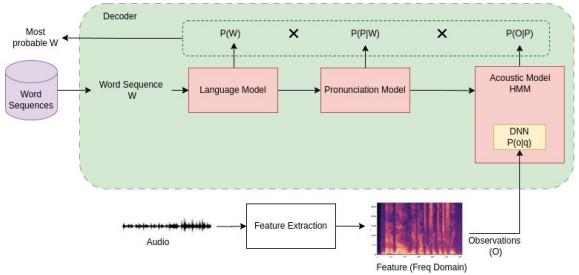




$$a_{ij}(t) = P(q_t = j | q_{t-1} = i)$$
$$b_i(t) = P(o_t | q_t = i)$$

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Output-independence assumption

$$P(o_t \mid O_1^{t-1}, q_1^t) = P(o_t \mid q_t)$$

The number of ice-creams that he ate (observation) depends only on the weather of that particular day (hidden state).

Source 1, 2, 3

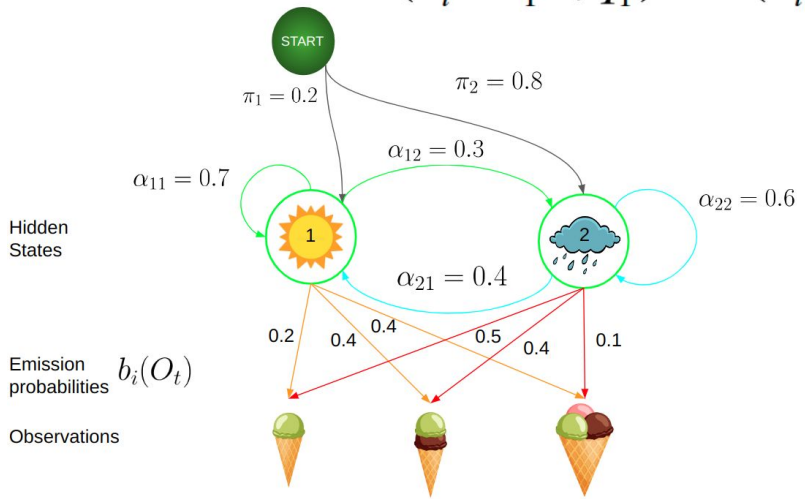
ASR Pipeline - Acoustic Model - Training

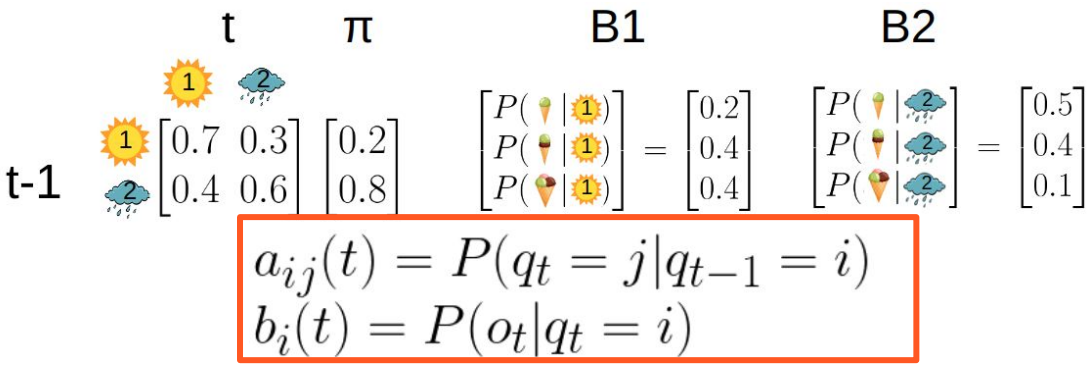
First order Markov assumption:

$$P(q_t|q_{t-1}, q_{t-2}.., q_1) = P(q_t|q_{t-1})$$

Output-independence assumption

$$P(o_t \mid O_1^{t-1}, q_1^t) = P(o_t \mid q_t)$$

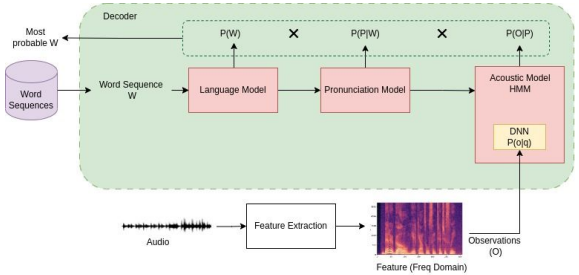




$$a_{ij}(t) = P(q_t = j|q_{t-1} = i)$$
$$b_i(t) = P(o_t|q_t = i)$$

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



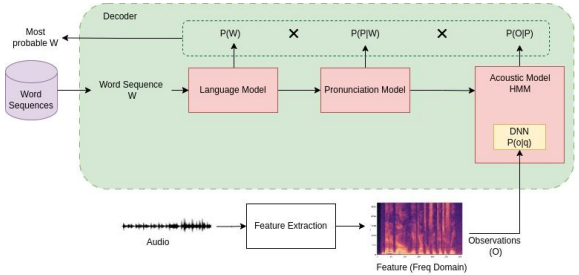
$$P(O|P)$$

HMM solves the following tasks:

- **Evaluation (likelihood):** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?
- **Decoding:** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?
- **Learning:** How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



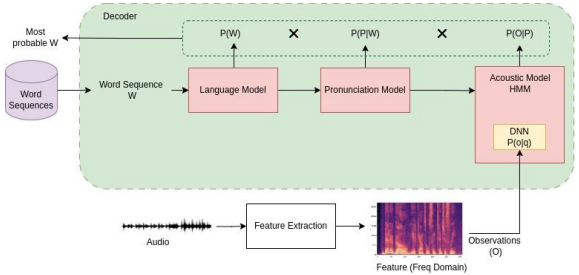
$$P(O|P)$$

HMM solves the following tasks:

- **Evaluation (likelihood):** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?
- **Decoding:** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?
- **Learning:** How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Decoding

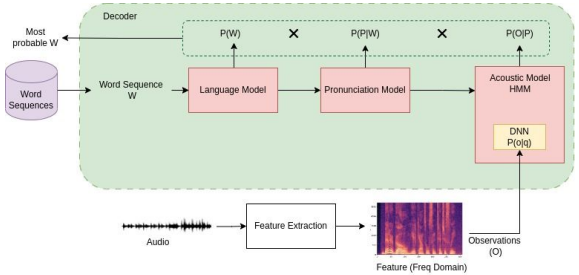
Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

$$\widehat{Q} = arg \max_{Q=\{q_1,..,q_N\}} P(O, Q|\lambda)$$

Source 1, 2, 3

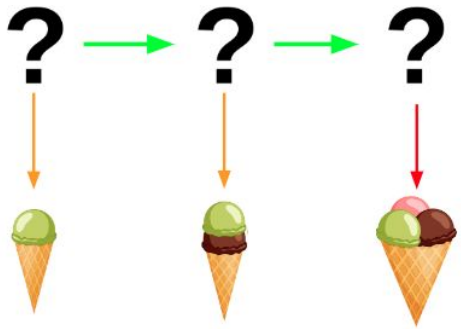
ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Decoding

Decoding solves the followings:

Given the observation sequei  **an HMM model $\lambda = \{A, B\}$, how do**

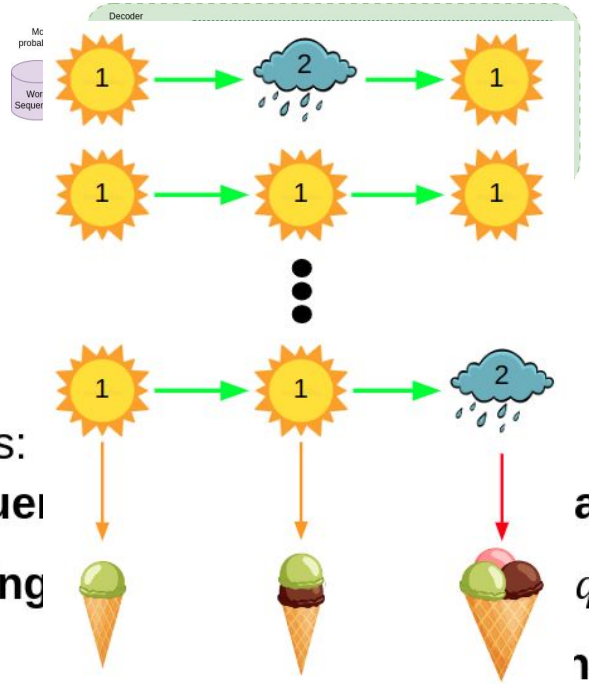
we choose a corresponding $q_2, \ldots, q_T\}$ **that is optimal, i.e., best**

ıs?

$$\widehat{Q} = arg \max_{Q=\{q_1,..,q_N\}} P(O, Q|\lambda)$$

Source 1, 2, 3

ASR Pipeline - Acoustic Model - Training



Decoding

Decoding solves the followings:

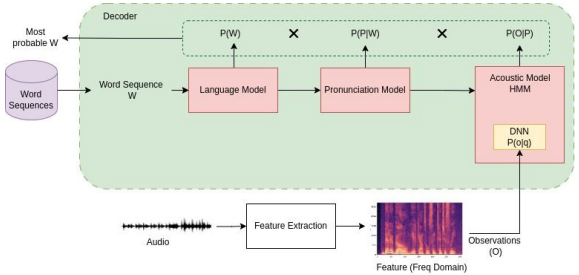
Given the observation sequel **an HMM model $\lambda = \{A, B\}$, how do**

we choose a corresponding $q_2,\ldots, q_T\}$ **that is optimal, i.e., best**

ıs?

$$\widehat{Q} = arg \max_{Q=\{q_1,..,q_N\}} P(O, Q|\lambda)$$

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

Decoding

$$P(2,2,1,H,C,C|A,B) =?$$

t π B1 B2

t-1

$$\begin{matrix} & 1 & 2 \\ 1 & 0.7 & 0.3 \\ 2 & 0.4 & 0.6 \end{matrix} \quad \begin{matrix} 0.2 \\ 0.8 \end{matrix}$$

$$\begin{bmatrix} P(\text{ice cream}|1) \\ P(\text{umbrella}|1) \\ P(\text{sunglasses}|1) \end{bmatrix} = \begin{bmatrix} 0.2 \\ 0.4 \\ 0.4 \end{bmatrix} \quad \begin{bmatrix} P(\text{ice cream}|2) \\ P(\text{umbrella}|2) \\ P(\text{sunglasses}|2) \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.4 \\ 0.1 \end{bmatrix}$$

$$a_{ij}(t) = P(q_t = j|q_{t-1} = i)$$
$$b_i(t) = P(o_t|q_t = i)$$

$$P(o_1 = 2, o_2 = 2, o_3 = 1, q_1 = H, q_2 = C, q_3 = C) =$$
$$\pi_H * p(2|H) * p(C|H) * p(2|C) * p(C|C) * p(1|C) =$$
$$0.2 * 0.4 * 0.3 * 0.4 * 0.6 * 0.5 = 0.00288$$

ASR Pipeline - Acoustic Model - Training

Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

General Solution

$$P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{n} P(o_i|q_i) \times \prod_{i=1}^{n} P(q_i|q_{i-1})$$

Finding the state sequence that maximizes the probability of the observed sequence:

- Using the naive approach: Iterate over all state sequences Q:
  - HHH, HHC, CCC, CHC, etc.
  - For each state sequence calculate P(O,Q)
  - Take the argmax
- This exhaustive search has **high computational complexity → infeasible**

ASR Pipeline - Acoustic Model - Training

Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

General Solution

$$P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{n} P(o_i|q_i) \times \prod_{i=1}^{n} P(q_i|q_{i-1})$$

Solution: Viterbi algorithm

Apply **dynamic programming** to **efficiently** calculate the max probability.

Viterbi finds the state sequence that maximizes the probability of observed sequence.

ASR Pipeline - Acoustic Model - Training

Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

General Solution

$$P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{n} P(o_i|q_i) \times \prod_{i=1}^{n} P(q_i|q_{i-1})$$

Solution: Viterbi algorithm

$$v_t(j) = \max_{q_0, q_1, \ldots, q_{t-1}} P(q_0, q_1 \ldots q_{t-1}, o_1, o_2 \ldots o_t, q_t = j|\lambda)$$

ASR Pipeline - Acoustic Model - Training

Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

General Solution

$$P(O,Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{n} P(o_i|q_i) \times \prod_{i=1}^{n} P(q_i|q_{i-1})$$

Solution: Viterbi algorithm

$$v_t(j) = \max_{q_0,q_1,\ldots,q_{t-1}} P(q_0, q_1 \ldots q_{t-1}, o_1, o_2 \ldots o_t, q_t = j|\lambda)$$

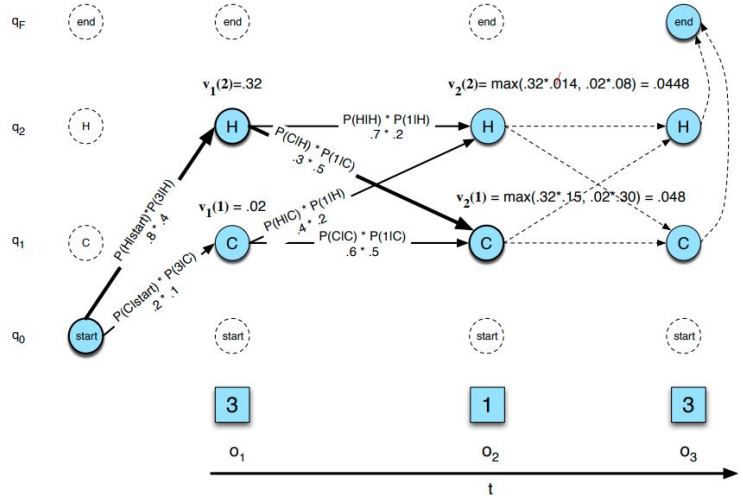
$$v_t(q_j) = \max_{i=1\ldots N} v_{t-1}(q = i)\alpha_{ij} b_j(o_t)$$

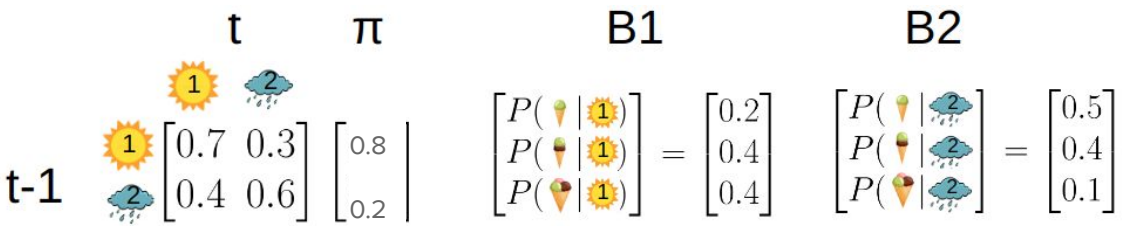
ASR Pipeline - Acoustic Model - Training

Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

Viterbi trellis





$$a_{ij}(t) = P(q_t = j | q_{t-1} = i)$$

$$b_i(t) = P(o_t | q_t = i)$$

Source 1

ASR Pipeline - Acoustic Model - Training

Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

Solution: Viterbi algorithm

Apply **dynamic programming** to efficiently calculate the max probability

1. **Initialization:**

$$v_1(j) = a_{0j} b_j(o_1) \quad 1 \le j \le N$$
$$bt_1(j) = 0$$

2. **Recursion** (recall that states 0 and q_F are non-emitting):

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t); \quad 1 \le j \le N, 1 < t \le T$$
$$bt_t(j) = \operatorname{argmax}_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t); \quad 1 \le j \le N, 1 < t \le T$$

3. **Termination:**

$$\text{The best score:} \quad P* = v_t(q_F) = \max_{i=1}^{N} v_T(i) * a_{i,F}$$
$$\text{The start of backtrace:} \quad q_T* = bt_T(q_F) = \operatorname{argmax}_{i=1}^{N} v_T(i) * a_{i,F}$$

Source 1

ASR Pipeline - Acoustic Model - Training

Decoding solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

Solution: Viterbi algorithm

Apply **dynamic programming** to efficiently calculate the max probability

1. **Initialization:**

$$v_1(j) = a_{0j} b_j(o_1) \quad 1 \le j \le N$$
$$bt_1(j) = 0$$

Viterbi finds the state sequence that maximizes the probability of observed sequence.

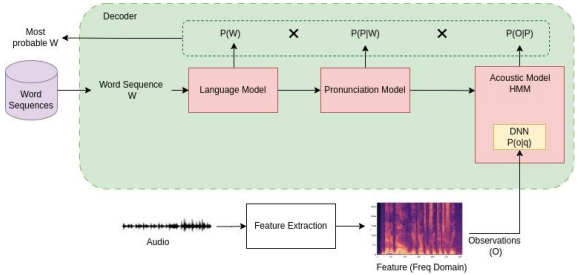
Main Usage: Force alignment

3. **Termination:**

$$\text{The best score:} \quad P* = v_t(q_F) = \max_{i=1}^{N} v_T(i) * a_{i,F}$$

$$\text{The start of backtrace:} \quad q_T* = bt_T(q_F) = \operatorname{argmax}_{i=1}^{N} v_T(i) * a_{i,F}$$

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

HMM solves the following tasks:

- **Evaluation (likelihood):** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?
- **Decoding:** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?
- **Learning:** How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

ASR Pipeline - Acoustic Model - Training

Evaluation solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?

The main difference from decoding is that we sum instead of using max.

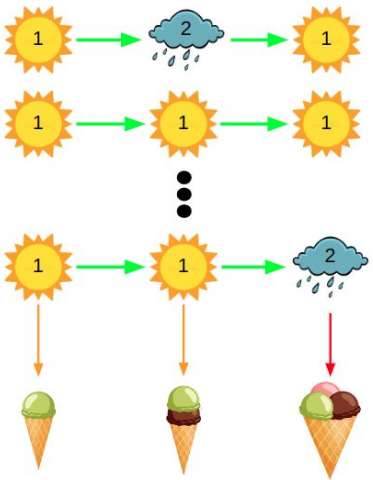
ASR Pipeline - Acoustic Model - Training

Evaluation solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?

Computing the likelihood of observing 1, 2, 3

$$P(O) = \sum_{Q} P(O, Q) = \sum_{Q} P(O|Q)P(Q) \qquad P(O, Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{n} P(o_i|q_i) \times \prod_{i=1}^{n} P(q_i|q_{i-1})$$



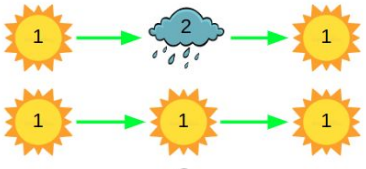
ASR Pipeline - Acoustic Model - Training

Evaluation solves the followings:

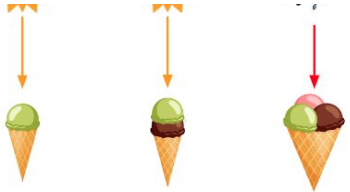
Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?

Computing the likelihood of observing 1, 2, 3

$$P(O) = \sum_{Q} P(O, Q) = \sum_{Q} P(O|Q)P(Q) \qquad P(O, Q) = P(O|Q) \times P(Q) = \prod_{i=1}^{n} P(o_i|q_i) \times \prod_{i=1}^{n} P(q_i|q_{i-1})$$



Using the naive approach is exhaustive



ASR Pipeline - Acoustic Model - Training

Evaluation solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?

Solution: Forward algorithm

Apply **dynamic programming** to compute **efficiently** the likelihood of the observation sequence by **summing** over all possible hidden state sequences

ASR Pipeline - Acoustic Model - Training

Evaluation solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?

Forward algorithm

$$\alpha_t(j) = P(o_1, o_2 \dots o_t, q_t = j|\lambda)$$

Each cell of the forward algorithm trellis, $\alpha_t(j)$, **Represents the probability of being in state j After seeing the first t observations** Given the automaton (**regardless** the path that it took- assuming all paths)

ASR Pipeline - Acoustic Model - Training

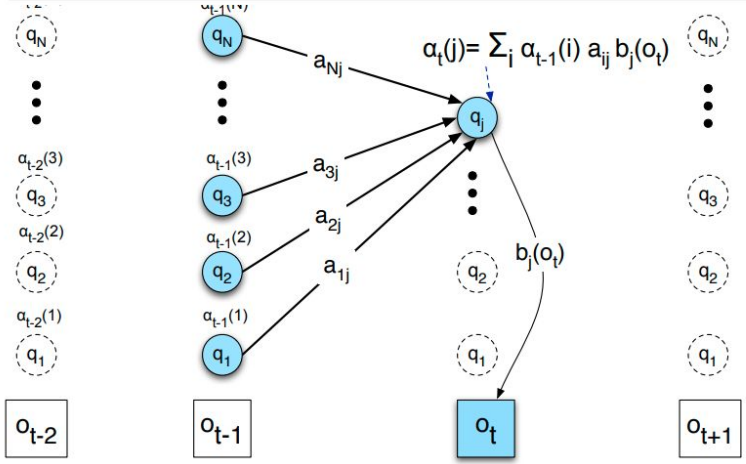
Evaluation solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do

we efficiently com **l sequence, given the**

$\alpha_{t-1}(i)$	the **previous forward path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j

Forward algorithm



Each cell of th **the probability of**

being in state /en the automaton

(**regardless** the

Source 1

ASR Pipeline - Acoustic Model - Training

Evaluation solves the followings:

Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?

The Forward Algorithm

1. Initialization

$$\alpha_1(j) = a_{0j} b_j(o_1) \quad 1 \le j \le N$$

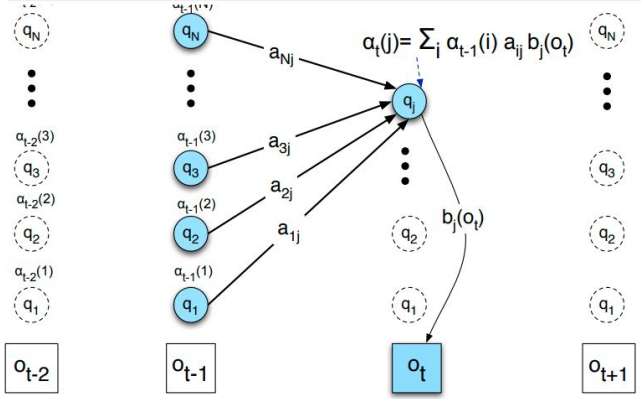
2. Recursion (since state 0 and F are non-emitting):

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t); \quad 1 \le j \le N, 1 < t \le T$$

3. Termination:

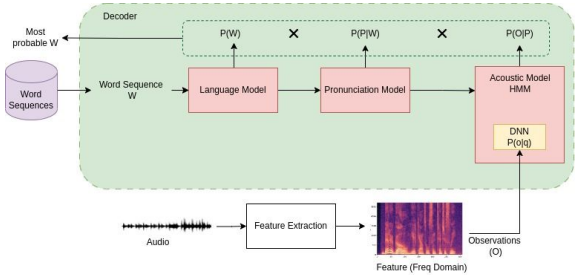
$$P(O|\lambda) = \alpha_T(q_F) = \sum_{i=1}^{N} \alpha_T(i) a_{iF}$$

$\alpha_{t-1}(i)$	the **previous forward path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j



Source 1

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

HMM solves the following tasks:

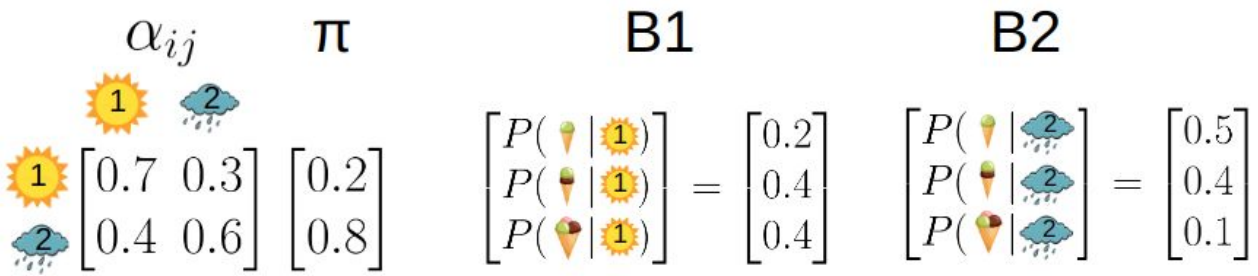
- **Evaluation (likelihood):** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability of the observed sequence, given the model?
- **Decoding:** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?
- **Learning:** How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

ASR Pipeline - Acoustic Model - Training

Learning

How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

$$O = \{o_1, ..., o_N\}; \lambda = \{A, B, \pi\}?$$



ASR Pipeline - Acoustic Model - Training

Learning

How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

$$O = \{o_1, ..., o_N\}; \lambda = \{A, B, \pi\}?$$

Define forward and backward variables:

- Forward variable: the probability of seeing the observations $o_1, o_2, ..., o_t$ and being in state i at time t

$$\alpha_t(i) = P(o_1, o_2, ..., o_t, q_t = S_i|\lambda)$$

- Backward variable: the probability of the ending partial sequence $q_{t+1}, q_{t+2}, ..., q_T$ starting state i at time t.

$$\beta_t(i) = P(q_{t+1}, q_{t+2}, ..., q_T|q_t = S_i, \lambda)$$

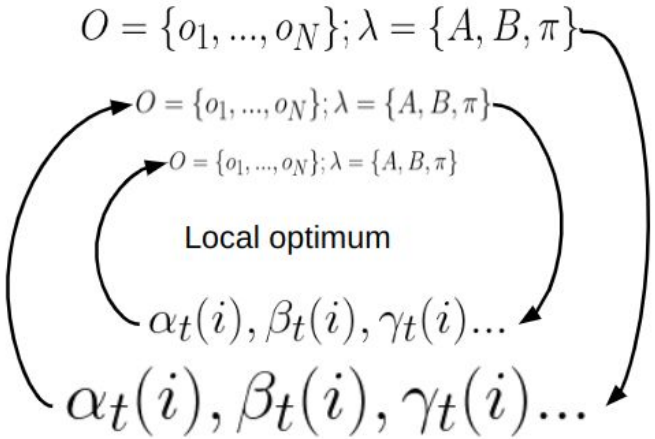
ASR Pipeline - Acoustic Model - Training

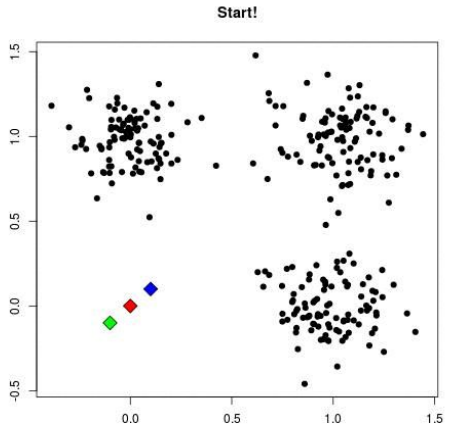
Learning

How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

$$O = \{o_1, ..., o_N\}; \lambda = \{A, B, \pi\}?$$

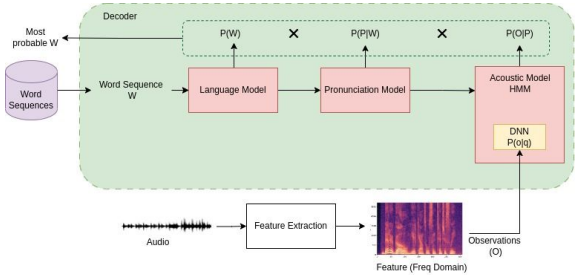
Baum Welch Algorithm - EM





Source 1

ASR Pipeline - Acoustic Model - Training



$$P(O|P)$$

HMM solves the following tasks:

- **Evaluation (likelihood):** Given the observation sequence $O = \{o_1, o_2, ..., o_T\}$, and an HMM model $\lambda = \{A, B\}$, how do we efficiently compute $P(O|\lambda)$, the probability

HMM for speech recognition

model $\lambda = \{A, B\}$, how do we choose a corresponding state sequence $Q = \{q_1, q_2, ..., q_T\}$ that is optimal, i.e., best explains the observations?

- **Learning:** How do we learn the parameters $\lambda = \{A, B\}$ to maximize $P(O|\lambda)$

ASR Pipeline - Acoustic Model - Notes

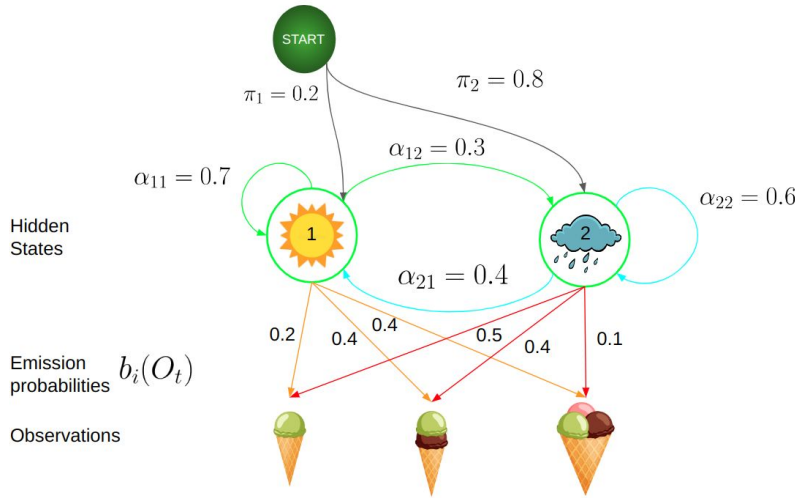
HMM for speech recognition

For a given phone/word string W, we will evaluate P(O|W):

- Observation sequence O- a series of MFCC vectors
- Hidden states W- phones and words

A and B defines:

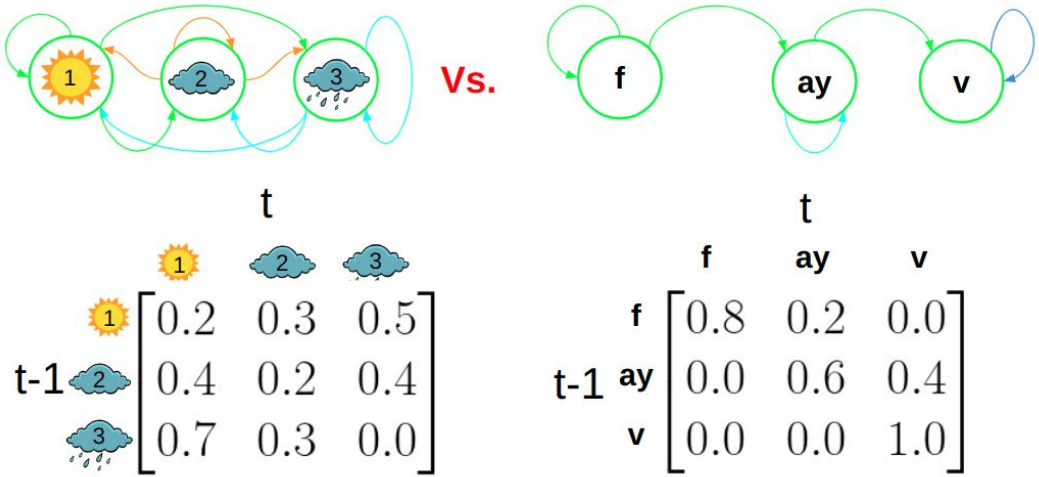
- A- **what** will be **said** (defines the phoneme sequence (hidden states) and the transition probabilities).
- B- **How** it will be **pronounced**



ASR Pipeline - Acoustic Model - Notes

HMM for speech recognition

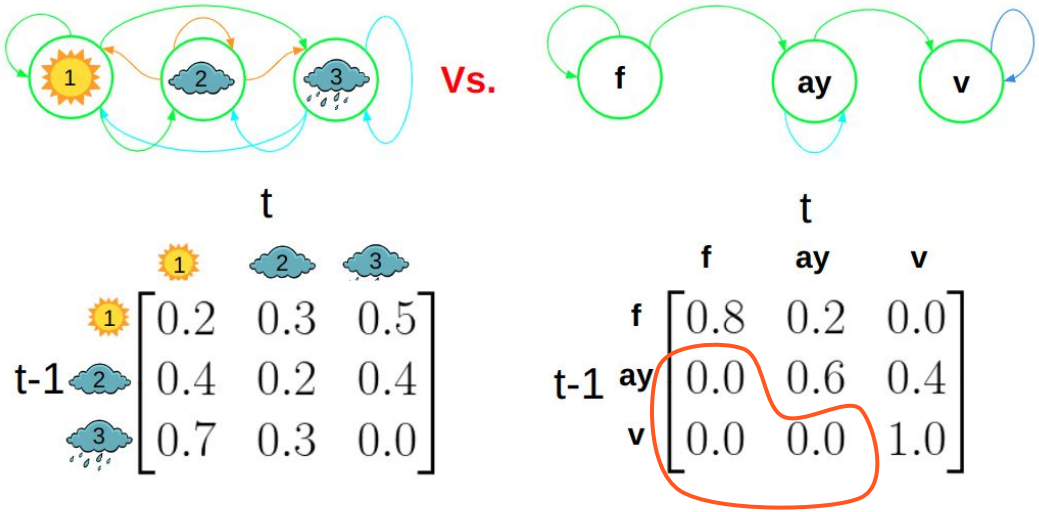
For a given phone/word string W, we will evaluate P(O|W)



ASR Pipeline - Acoustic Model - Notes

HMM for speech recognition

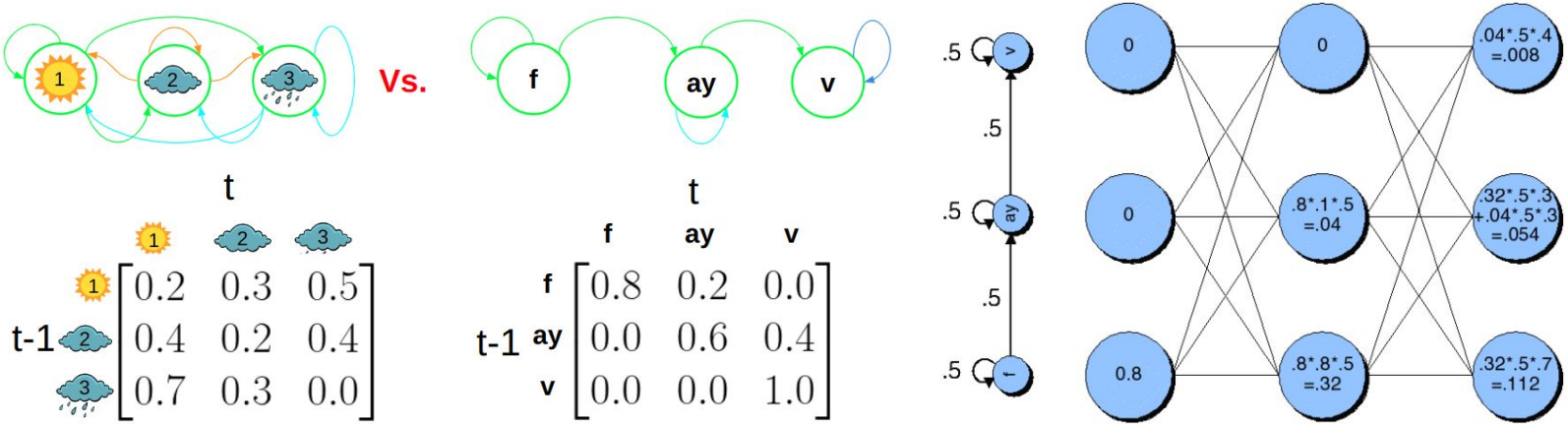
For a given phone/word string W, we will evaluate P(O|W - **evaluation/likelihood.**



ASR Pipeline - Acoustic Model - Notes

HMM for speech recognition

For a given phone/word string W, we will evaluate P(O|W):



Source 1

ASR Pipeline - Acoustic Model - Data

Alignment

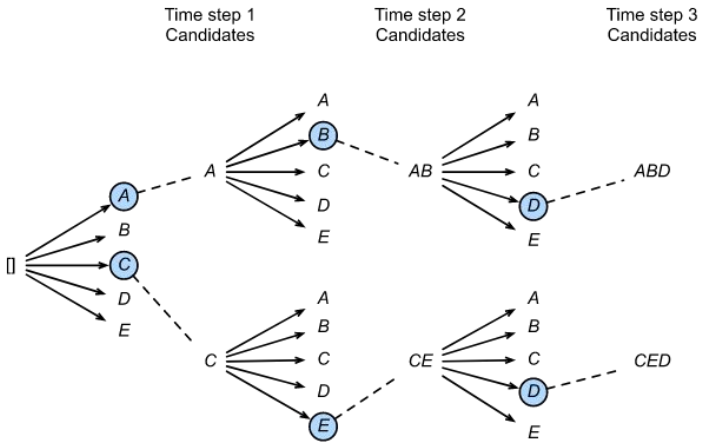
Because aligned data is expensive and hard to achieve (need phoneme level annotations), how can we obtain more training data?

- Training an acoustic model + HMM on a phone-level annotated data
- Take data with utterance level transcription
- Apply Viterbi (force-align) to find the best alignment while enforcing the ground truth text.
- Train an improved acoustic model + HMM using this pseudo labels.

ASR Pipeline - Acoustic Model - Inference

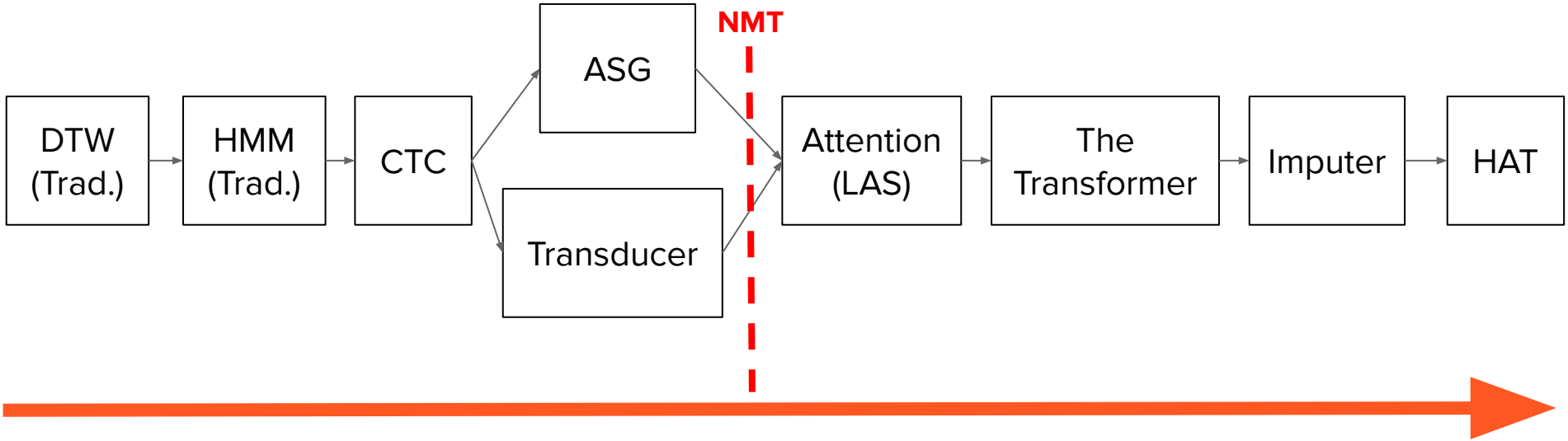
A word on the search

- In practice, it's not practical to search over all word sequences. The search space is too large and exhaustive.
- Solution: beam search



Source 1

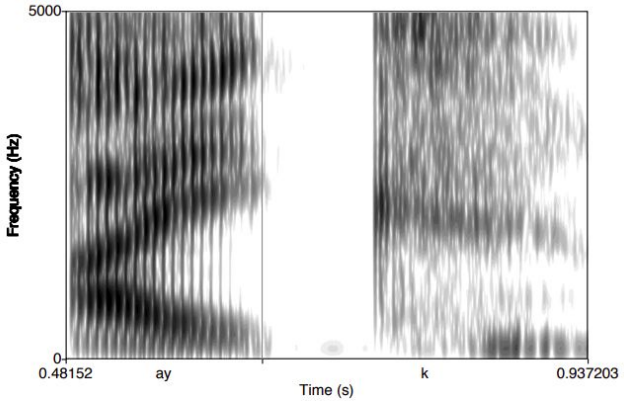
ASR Models - Overview



ASR Models - GMM- HMM

GMM-HMM Drawbacks

- Phonemes are not homogeneous
  - Each phone is divided into 3 substates (begin, middle, final)



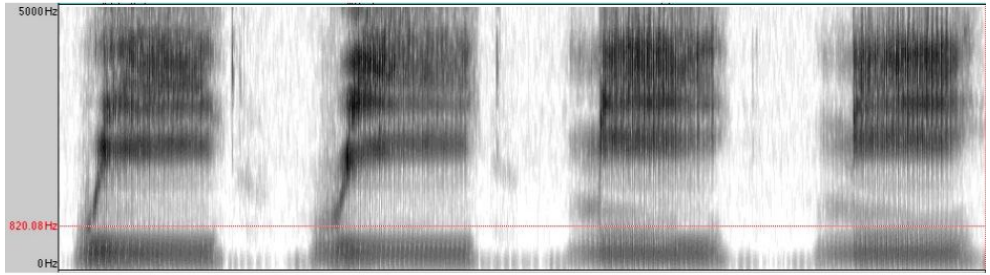
Source 1

ASR Models - GMM- HMM

GMM-HMM Drawbacks

- Phonemes are not homogeneous
  - Each phone is divided into 3 substates (begin, middle, final)
- The strongest factor affecting phonetic variability is the neighboring phone
  - Context-Independent (CI) Vs Context-Dependent (CD) phones: Triphones

w iy r iy m iy n iy



Source 1

ASR Models - DNN-HMM

- Acoustic Model Is A **Single** DNN For All States
- Improves Performances:
  - From 27% WER to 18% WER on SwitchBoard. Deep Neural Networks for Acoustic Modeling in Speech Recognition, Hinton el al (2012).
  - For comparison: current SOTA using End2End method is ~5% WER.