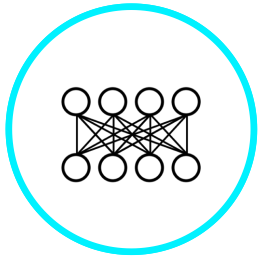
Algorithms for Speech and Natural Language Processing

Advanced acoustic modeling and language modeling for ASR

Emmanuel Dupoux, Robin Algayres (and slides by Neil Zeghidour)

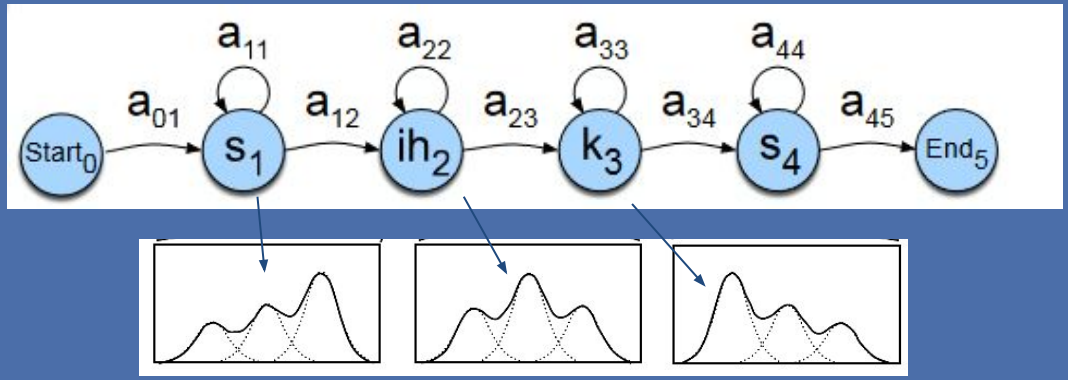


Advanced Acoustic modelling

- Large vocabulary ASR
- HMM-GMMs
- WFSTs

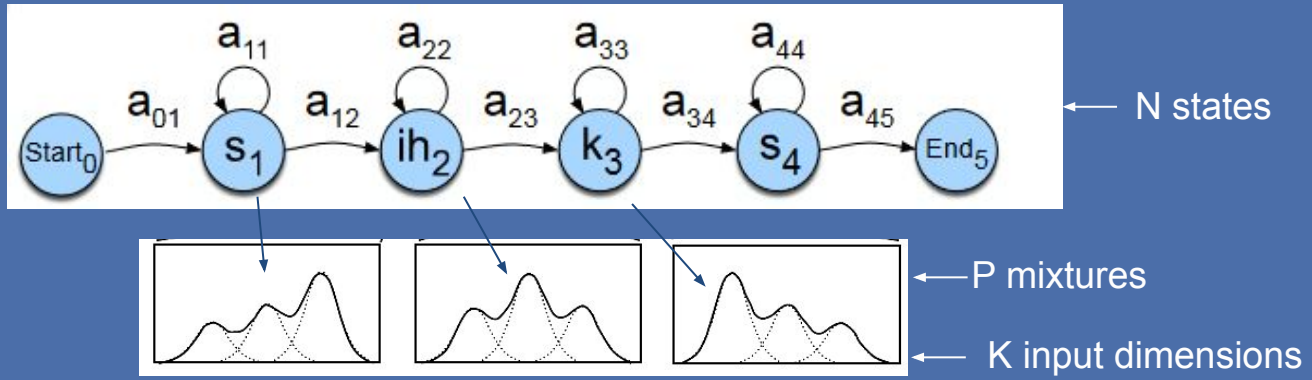
Using HMM for ASR

Let us consider the case of single-word HMM for the word ‘six’
Database: speech recordings of the word ‘six’
Goal: does a speech utterance says ‘six’ ?



Using HMM for ASR

Let us consider the case of single-word HMM for the word ‘six’
Database: speech recordings of the word ‘six’
Goal: does a speech utterance says ‘six’ ?

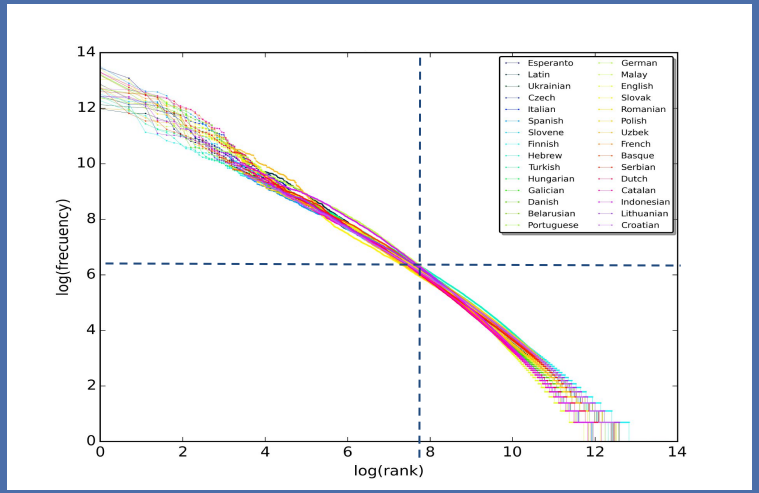


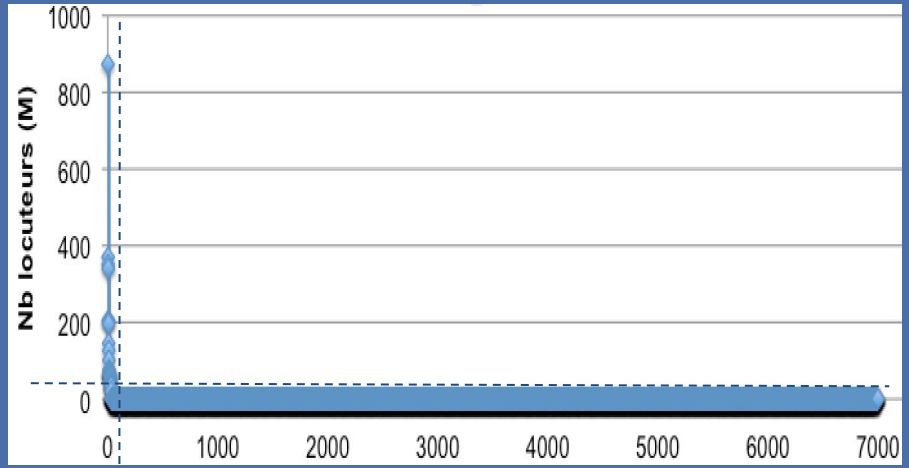
Each word is represented by NxPxK+N parameters (eg, 20x10x40+20=8020)
Problem?

Reminder from course 1: power laws

log(f)~-log(r)

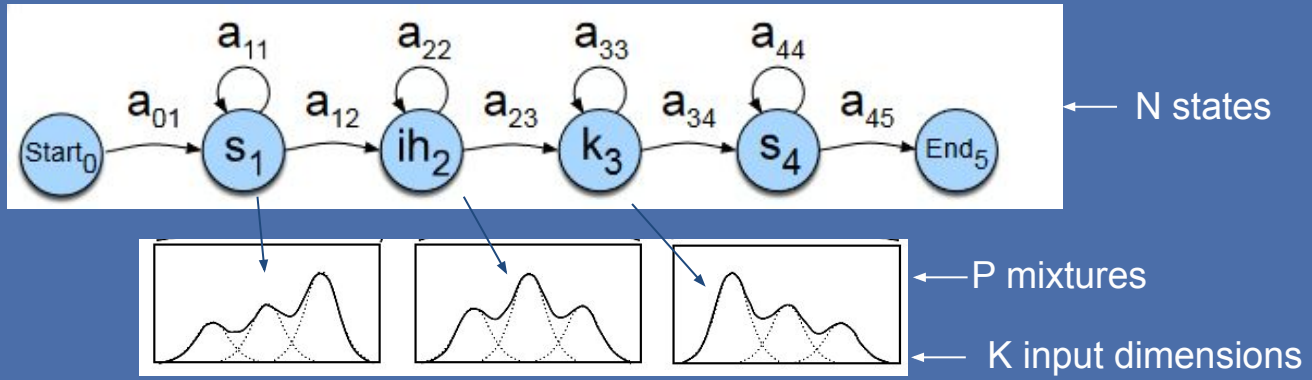
f~1/r





Using HMM for ASR

Let us consider the case of single-word HMM for the word ‘six’
Database: speech recordings of the word ‘six’
Goal: does a speech utterance says ‘six’ ?



Each word is represented by NxPxK parameters (eg, 20x10x40=8000)
→ ***How to learn the word parameters with large voc, where most words in a corpus are seen only once or twice?***

Reminder from course 1:

In all known languages, words are made of a combination of a small number of basic elements: **phonemes**

Phonological level

International Phonetic Alphabet

[aɪ pʰiː eɪ]

Graphemic level

enough, cough, draught, although, brought, through, thorough, hiccough

Phoneme inventories: between 20 and 200

Central Rotokas	Bilabial	Alveolar	Velar
Voiceless	p	t	k
Voiced	b ~ β	d ~ ɾ	g ~ ɣ

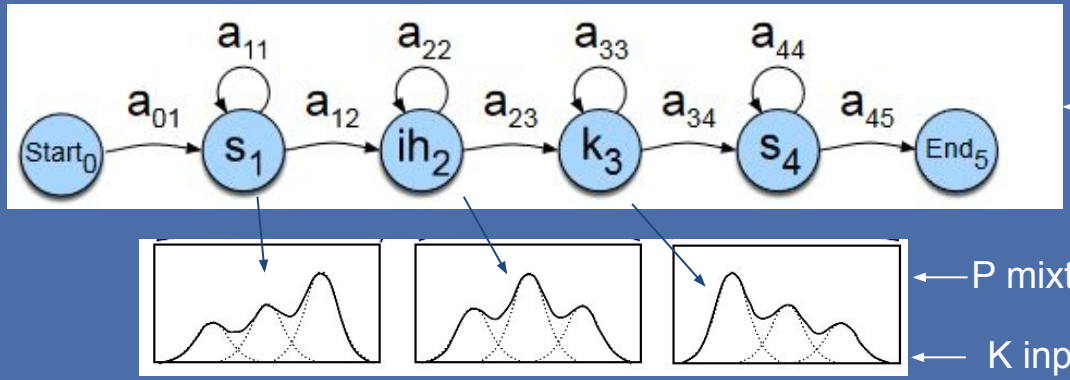
Ubykh		Labial		Alveolar			Postalveolar					Velar				Uvular					Glottal
							laminal closed		laminal		apical										
		plain	phar.	plain	lab.	lat.	plain	lab.	plain	lab.		pal.	plain	lab.	phar.	pal.	plain	lab.	phar.	phar. & lab.	
Plosive	voiceless	p	pˤ	t	tʷ							kʲ	k	kʷ		qʲ	q	qʷ	qˤ	qˤʷ	
	voiced	b	bˤ	d	dʷ							gʲ	g	gʷ							
	ejective	pʼ	pˤʼ	tʼ	tʷʼ							kʲʼ	kʼ	kʷʼ		qʲʼ	qʼ	qʷʼ	qˤʼ	qˤʷʼ	
Affricate	voiceless			t͡s			t͡ʃ		t͡ɕ	t͡ɕʷ	t͡ʂ										
	voiced			d͡z			d͡ʒ		d͡ʑ	d͡ʑʷ	d͡ʐ										
	ejective			t͡sʼ			t͡ʃʼ		t͡ɕʼ	t͡ɕʷʼ	t͡ʂʼ										
Fricative	voiceless	f		s		ɬ	ʃ	ʃʷ	ɕ	ɕʷ	ʂ		x			χʲ	χ	χʷ	χˤ	χˤʷ	h
	voiced	v	vˤ	z			ʒ	ʒʷ	ʑ	ʑʷ	ʐ		ɣ			ʁʲ	ʁ	ʁʷ	ʁˤ	ʁˤʷ	
	ejective					ɬʼ															
Nasal		m	mˤ	n																	
Approximant						l						j		w	wˤ						
Trill				r																	

Wikipedia

Phone-based ASR

For each phoneme in a language, build an HMM for it

/p/

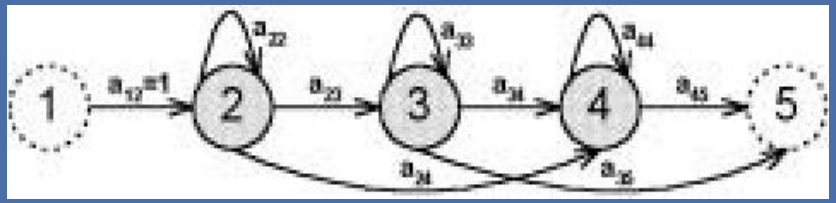


How many states for phonemes?

Phoneme-based ASR

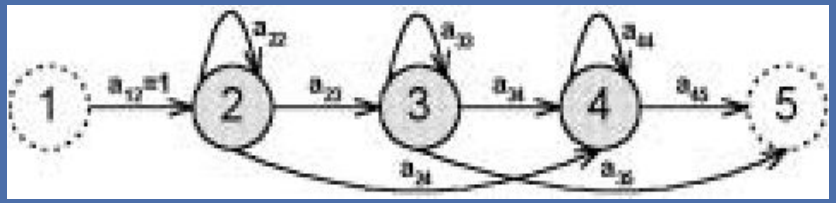
tristate phonemes

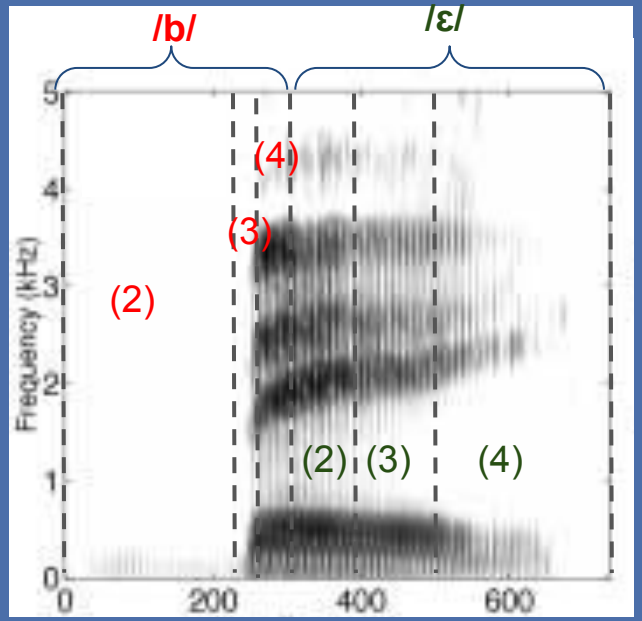
/b/



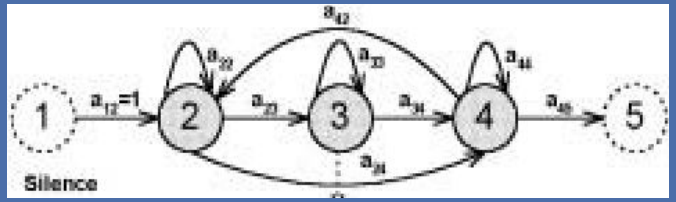
...

/ɛ/





/bɛ/

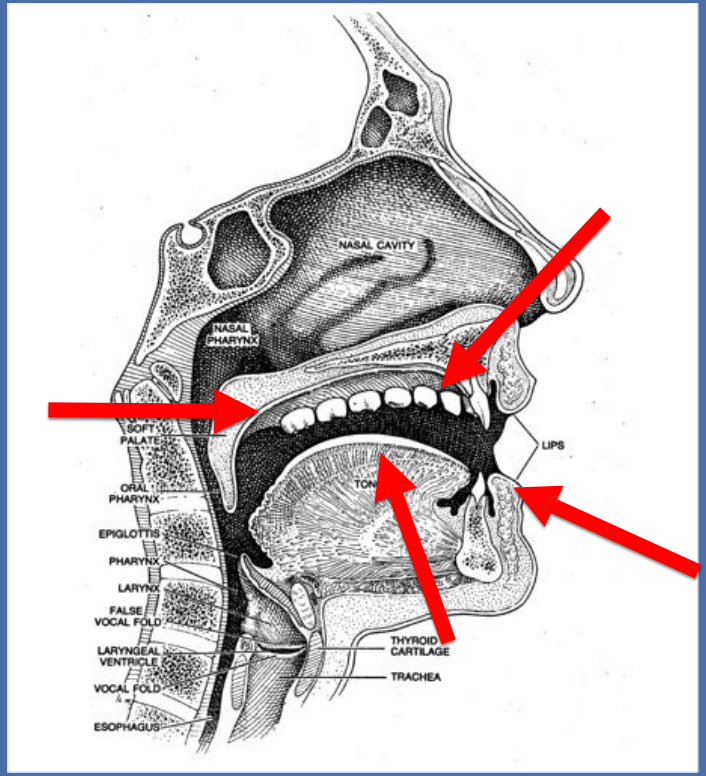


Reminder from last course: speech sounds are produced by a biomechanical system

Speech production

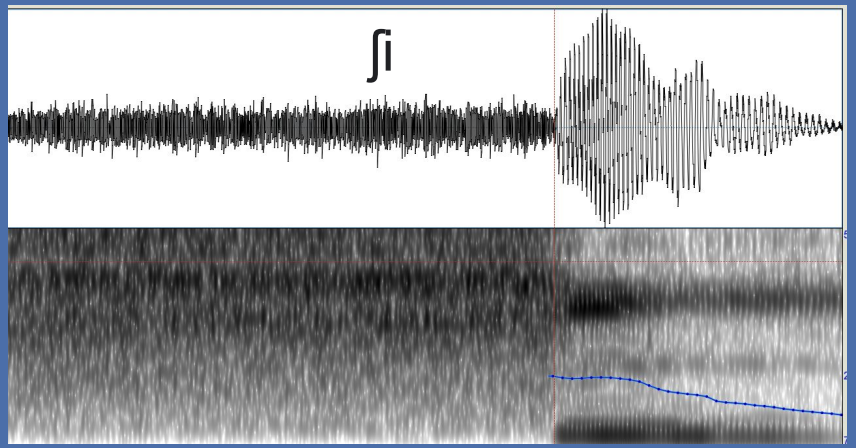
- Air is expelled from the lungs through the trachea
- Passes through vocal cords and resonate in the larynx
- Then through the nose and/or the mouth
- Vocal tract: the « articulators »: lips, tongue, teeth, palate, etc. The way we control the articulators defines the sound we produce.

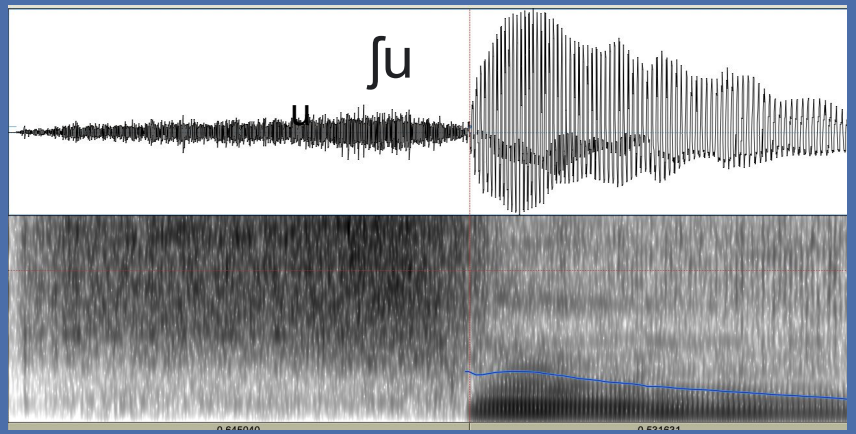
→ **Biomechanical motion: minimization of energy**

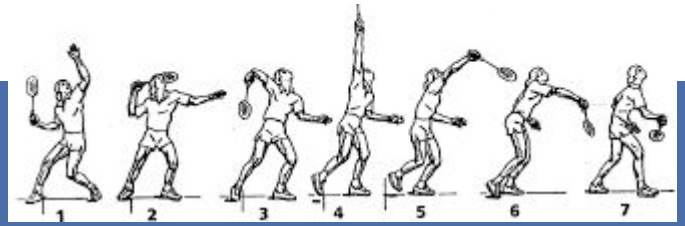


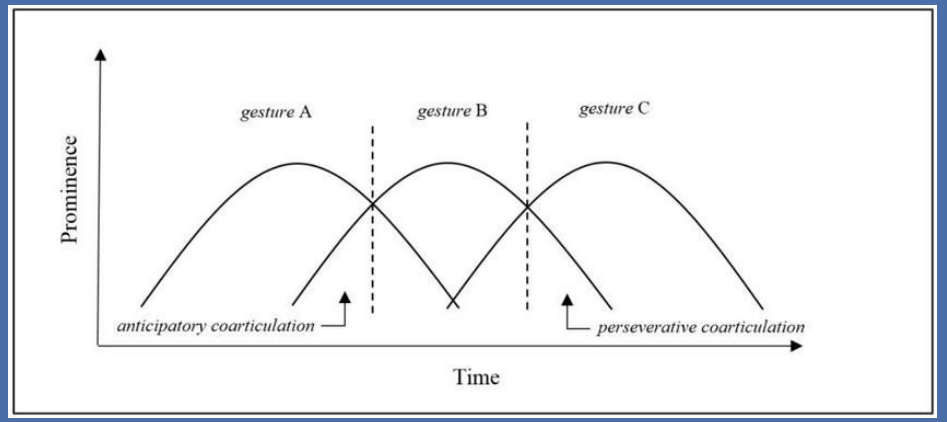
From Sundberg (1977)

Co-articulation: a consequence of biomechanical control

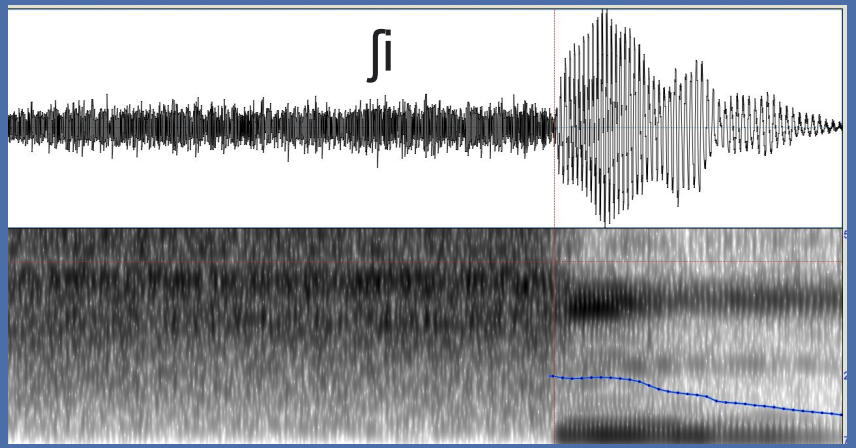


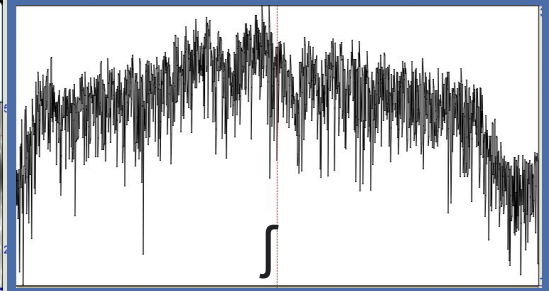


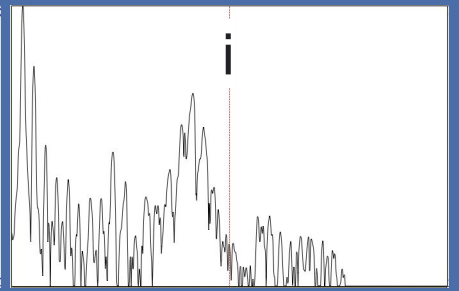


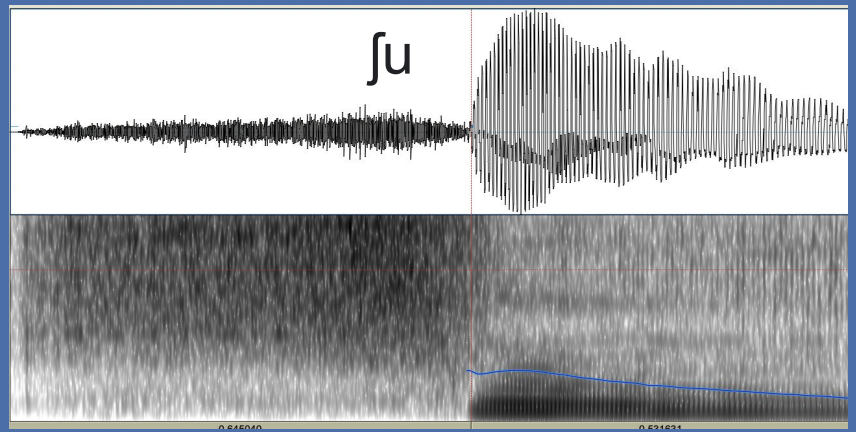


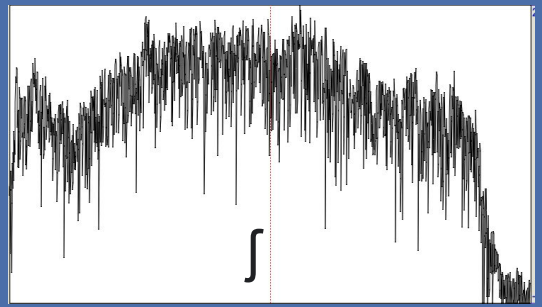
Co-articulation: a consequence of biomechanical control

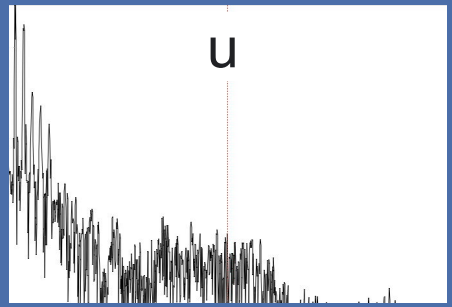


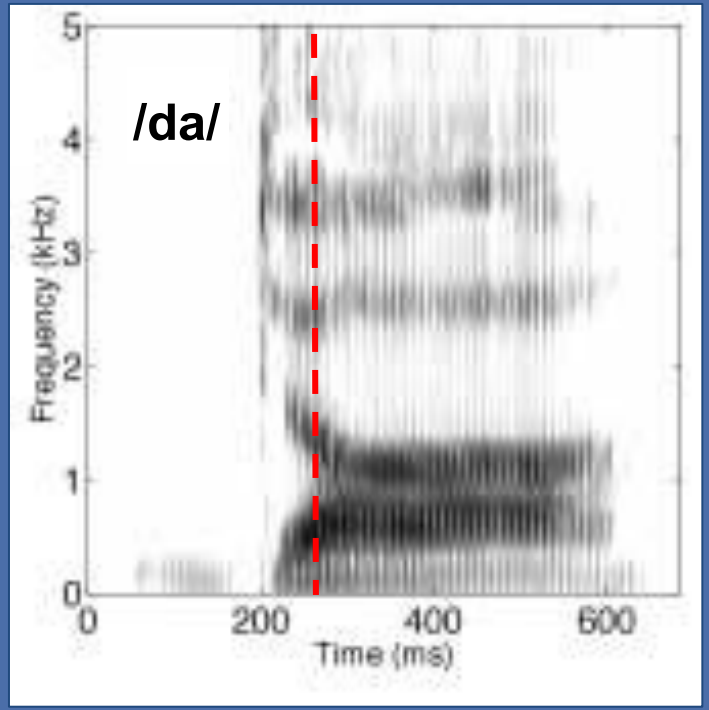


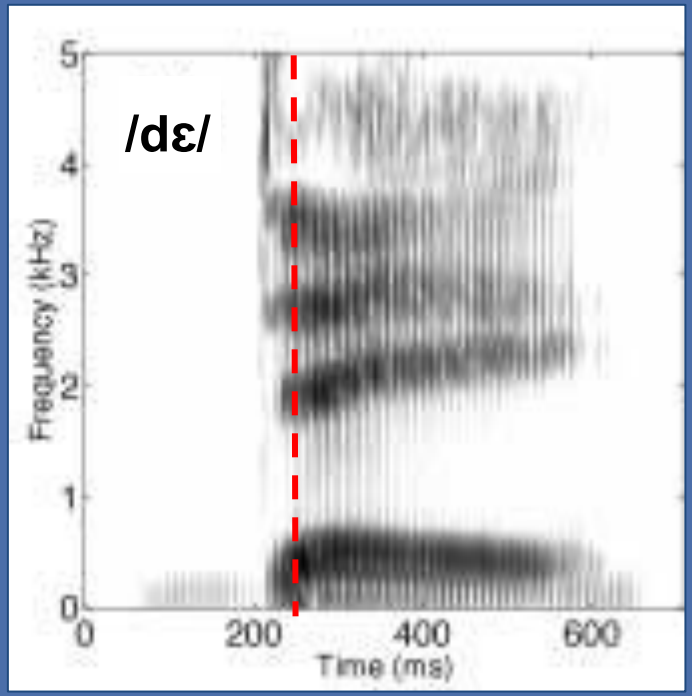






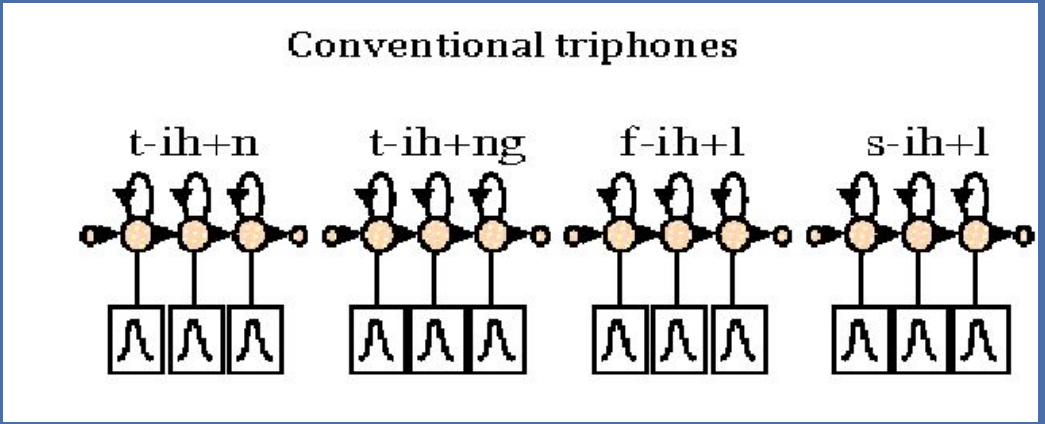



/da/
Frequency (kHz)
0
1
2
3
4
5
0
200
400
600
Time (ms)


/dɛ/
Frequency (kHz)
0
1
2
3
4
5
0
200
400
600
Time (ms)

Contextual-phone-based ASR

For each triphone in a language, build an HMM for it

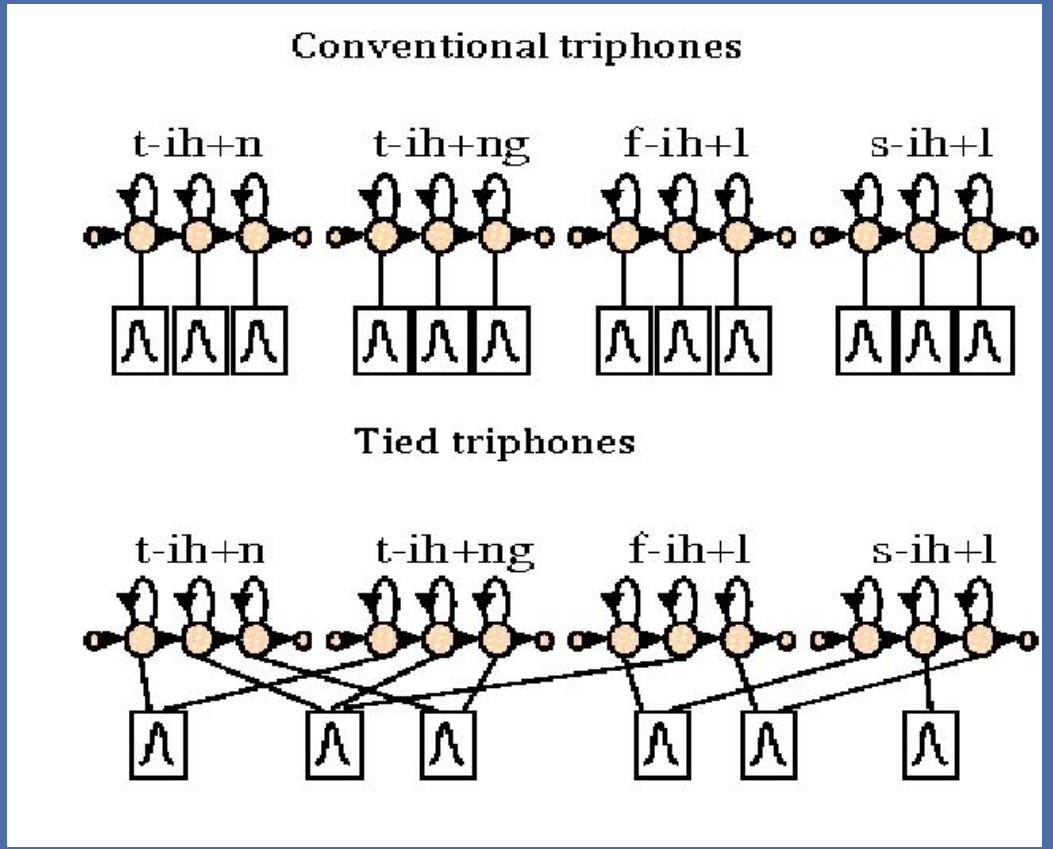


Problem: for English (40 phonemes), 64000 triphones
→ ***Power law, again***. *How do we deal with rare triphones?*

Contextual-phone-based ASR

For each triphone in a language, build an HMM for it, and merge them

More accurately, merge similar phone states



Contextual-phone-based ASR

For each triphone in a language, build an HMM for it, and merge them

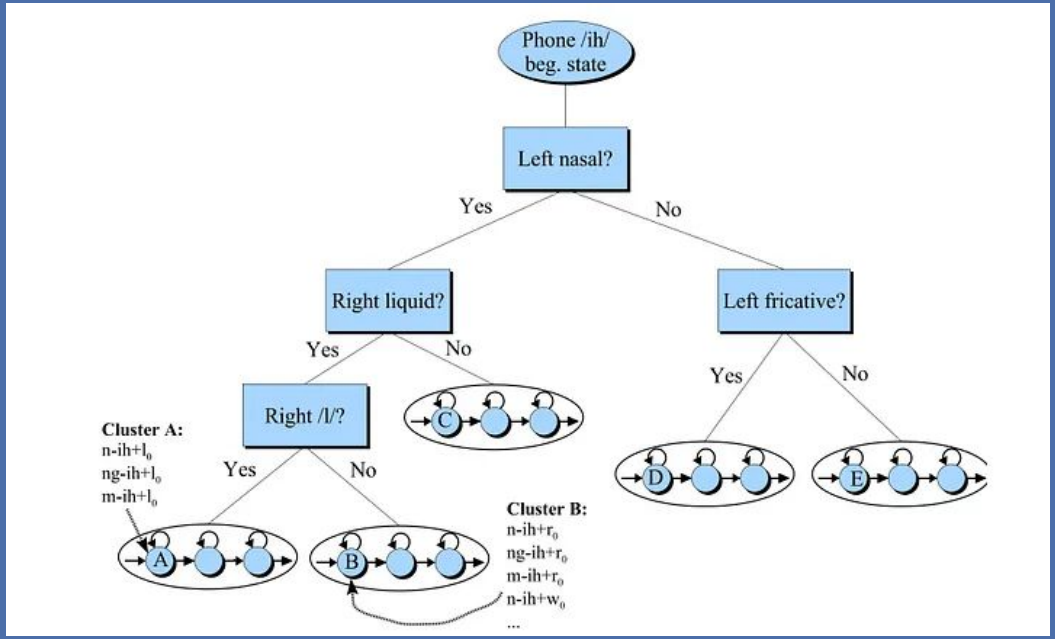
More accurately, merge similar phone states

Decision trees

- represent phonemes in terms of phonetic features (eg, voiced vs unvoiced, labial vs velar, etc)
- Split phonemes into contextual phones by asking questions

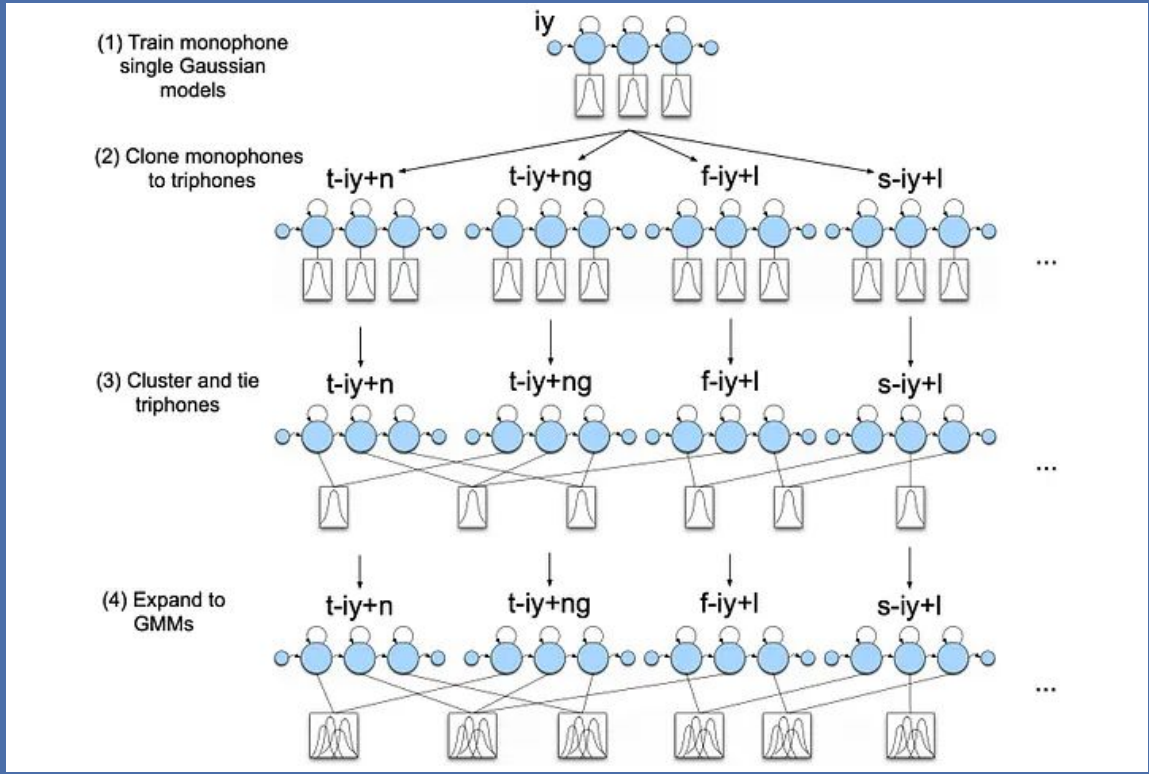
Data driven

merge triphones states



Contextual-phone-based ASR

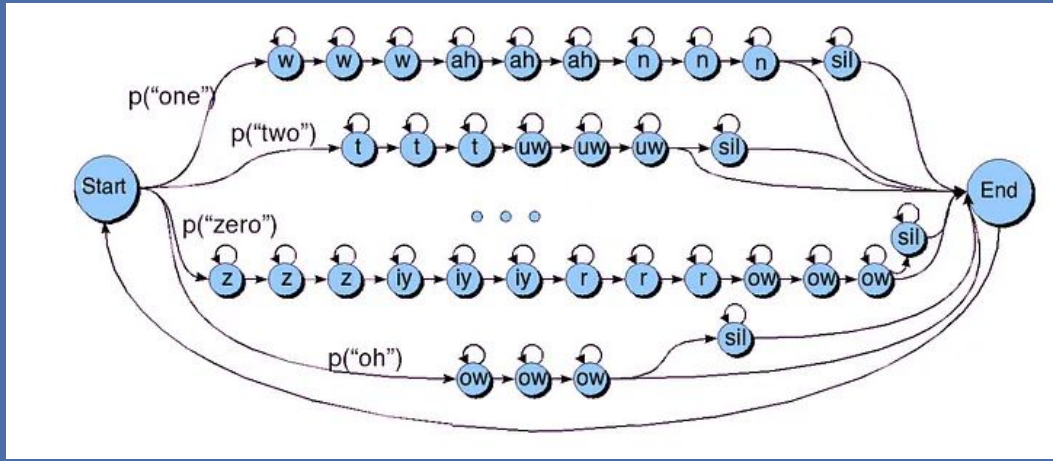
Full recipe



Contextual-phone-based ASR

For each contextual phone build an HMM for it

Make a list of words and build a large HMM out of it by concatenating the relevant contextual phones

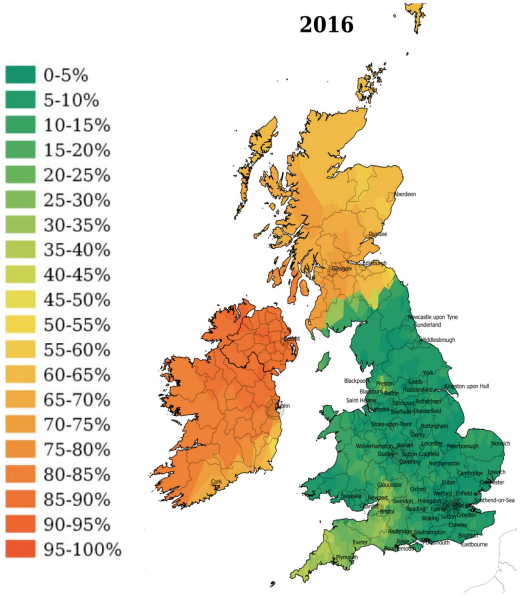


Are we done?

Reminder from first course: linguistic “variation”

anagement maagement maanagement
maangement magagement magement
mamagement mamangement manaagement manaement
managaement manageement manageemnt managegment
managemaent managemant managememt managemen managemenet
managementt managemet managemetn managemnent managemnet
managemnt managemrnt managemt managenent managenment managent
managerment managhement managmeent managmement managment managnment
manament manamgement mananement manangment manasgement
manegement manegment mangaement mangagement mangagment
mangament mangement manggement mangment
mangmt menagement mgmt mgnt
mnagement mngmnt mngmt

Do you pronounce the “r” in “arm” ?



Phonetic variation

Prononciations of the word ‘eleven’ in the state of Pennsylvanie, USA

ɬˌlɛˀvə̆n
ɬˌlɛvə̆n
ɬˌɬɛvə̆n
ə'lɛˀvə̆n
ɬˌlɛvə̆n
əlɛvə̆n
ə'ɬɛvn
əɬɛbm
ɬˌ'lɛvn
lɛvə̆n
əɬɛˌˀᵊvə̆n
ɬɛvɱ
əɬɛᵊvə̆n
ɬ'lɛvɱ
…
(total= 220 variants)

https://www.youtube.com/watch?v=HbDnxzrbxn4

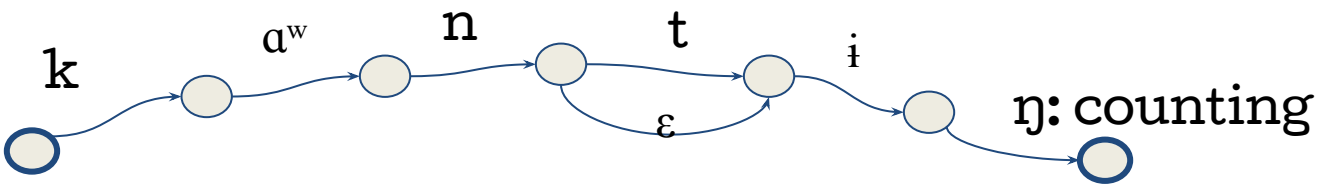
Phonological/Phonetic transformations

What are you doing?	'wʌtʃə'duɪn
I can inquire.	'aɪkŋ̩'kwaɪɚ
Did you eat yet?	'dʒitjɛʔ
I don’t believe him.	aɪ'doʊ̃bə'livɱ
We ought to have come.	wi'ɔɾ̪ɣ̩'kʌm

Pronunciation models

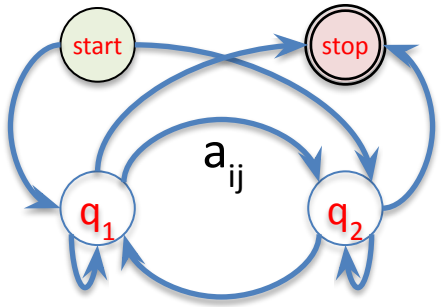
Using a pronunciation dictionary

orthography	SAMPA	IPA
counting	kA_wntIN	kɑʷntɨŋ
counting(1)	kA_wnIN	kɑʷnɨŋ
amortization	@mOrt@zeS@n	əmɔrtəzeʃən
amortization(1)	{mOrt@zeS@n	æmɔrtəzeʃən
amortization(2)	@mOrtA_jzeS@n	əmɔrtɑʲzeʃən
..etc		



Reminder from last course: Finite state Automata

Markov Chains

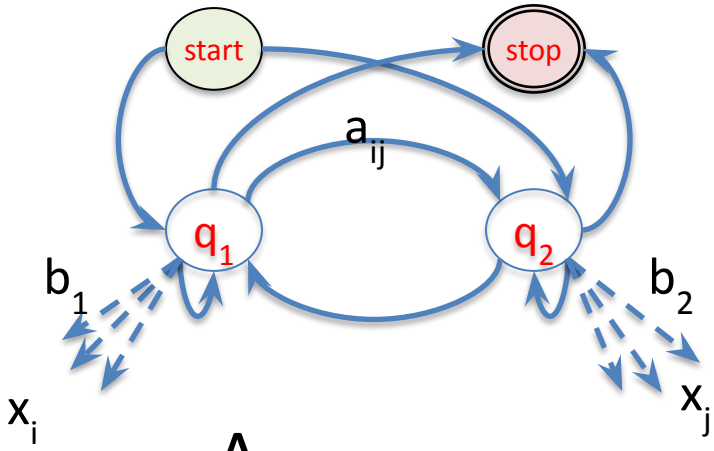


A

	q_0 (start)	q_1	q_2	q_F (stop)
q_0 (start)	0	a_{01}	a_{02}	0
q_1	0	a_{11}	a_{12}	a_{1F}
q_2	0	a_{21}	a_{22}	a_{2F}
q_F (stop)	0	0	0	0

Reminder from last course: Finite state Automata

HMMs



B

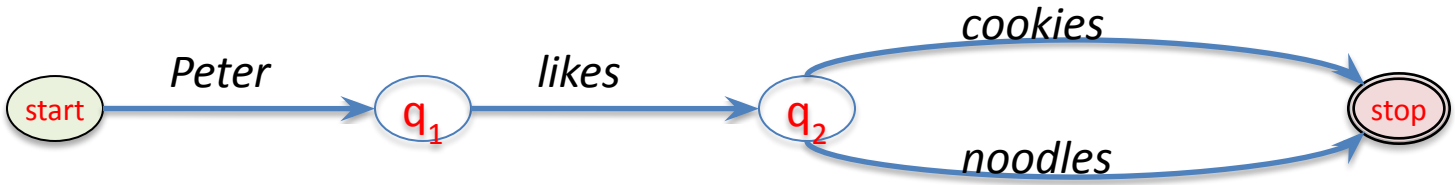
	q_1	q_2
x_1	b_{11}	b_{21}
x_2	b_{12}	a_{22}
x_3	b_{13}	a_{23}

Δ

	q_0 (start)	q_1	q_2	q_F (stop)
q_0 (start)	0	a_{01}	a_{02}	0
q_1	0	a_{11}	a_{12}	a_{1F}
q_2	0	a_{21}	a_{22}	a_{2F}
q_F (stop)	0	0	0	0

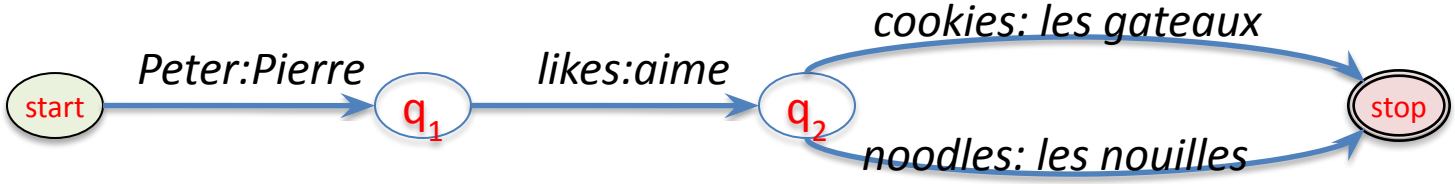
(weighted) finite state transducers

Finite State Acceptor (FSA)

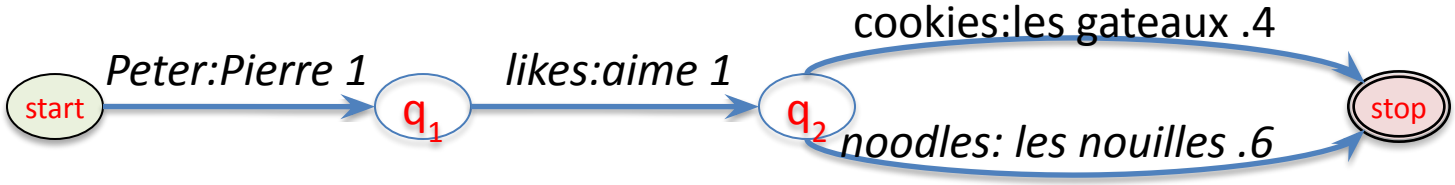


Weighted Finite State Acceptor (WFSA)

Finite State Transducer (FST)



Weighted Finite State Transducer (WFST)

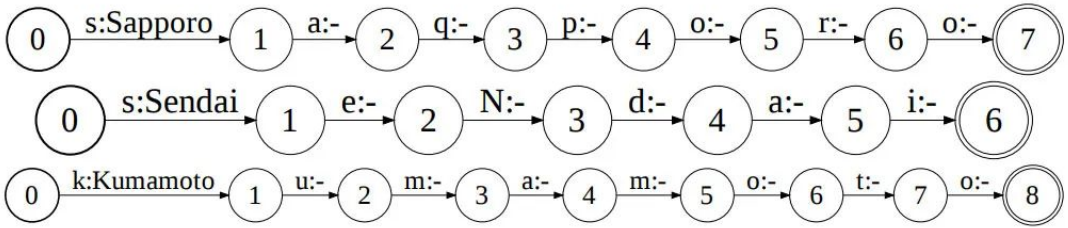


https://jonathan-hui.medium.com/speech-recognition-weighted-finite-state-transducers-wfst-a4ece08a89b7

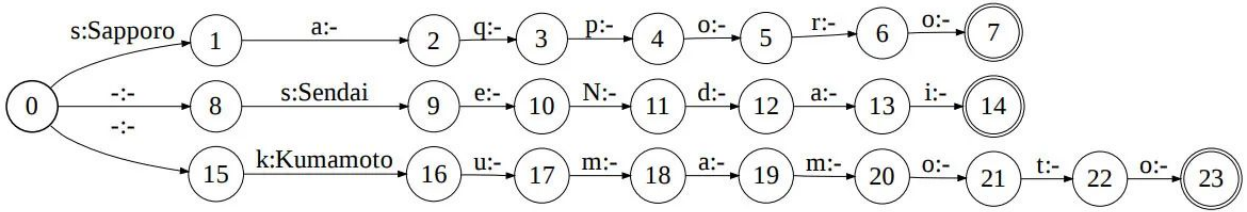
operations over wFSTs

sum of the union

$$[\![T1 \oplus T2]\!](x, y) = [\![T1]\!](x_1, y_1) \oplus [\![T2]\!](x_2, y_2)$$



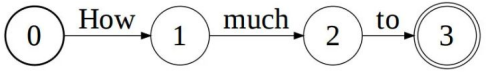
→

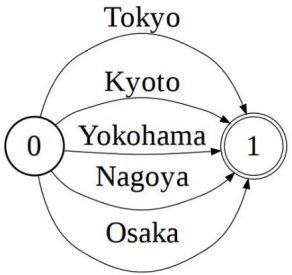


-:- Null Transitions that do not consume or generate symbols (allow deterministic transition if ambiguity occurs).

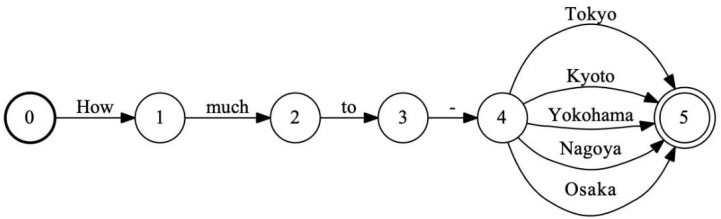
operations over wFSTs

$$[\![T1 \otimes T2]\!](x, y) = \bigoplus_{\substack{x = x_1 x_2 \\ y = y_1 y_2}} [\![T1]\!](x_1, y_1) \otimes [\![T2]\!](x_2, y_2)$$





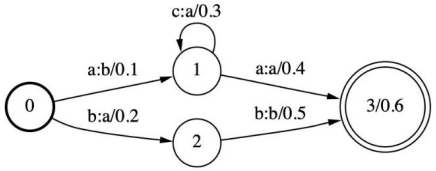
→



operations over wFSTs

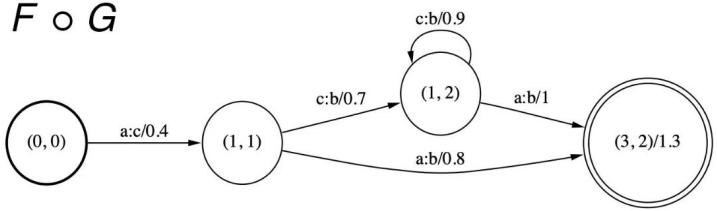
$$[\![T_1 \circ T_2]\!]((x,y) = \bigoplus_z [\![T_1]\!](x,z) \otimes [\![T_2]\!](z,y)$$

F



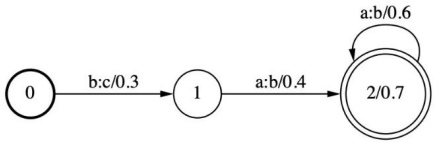
a a → b a
a c a → b a a

F ∘ *G*



a a → c b
a c a → c b b

G

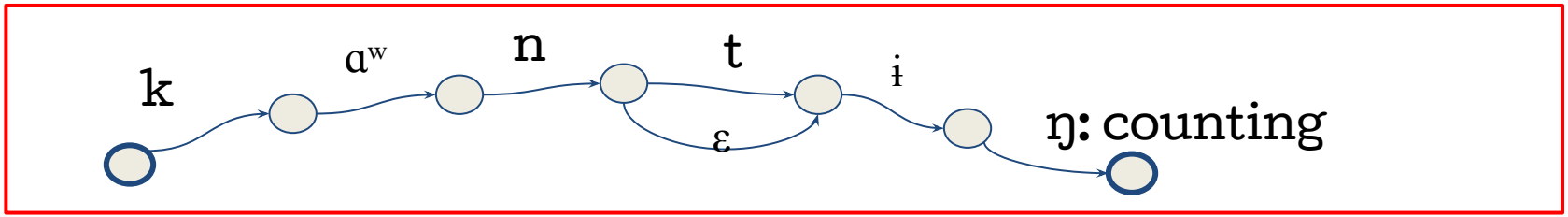


b a → c b
b a a → c b b

Pronunciation models

Using a pronunciation dictionary

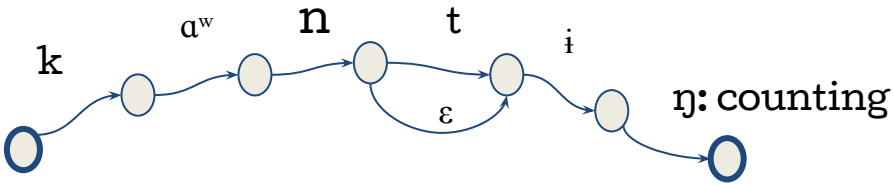
orthography	SAMPA	IPA
counting	kA_wntIN	kɑʷntɨŋ
counting(1)	kA_wnIN	kɑʷnɨŋ
amortization	@mOrt@zeS@n	əmɔrtəzeʃən
amortization(1)	{mOrt@zeS@n	æmɔrtəzeʃən
amortization(2)	@mOrtA_jzeS@n	əmɔrtɑʲzeʃən
..etc		



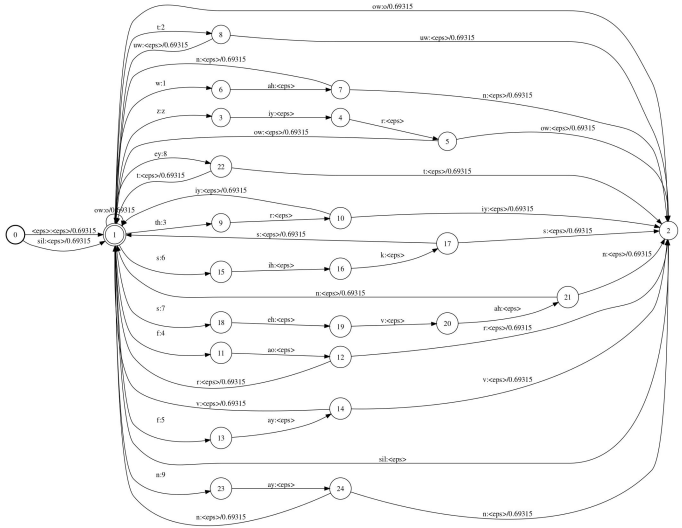
→**Pronounciations for a word can be modelled as a wFST**

→ **how to combine this with an HMM**

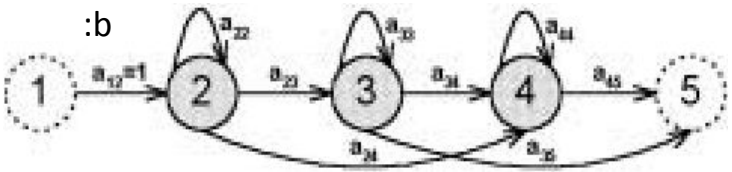
1. Build wFST model for all words



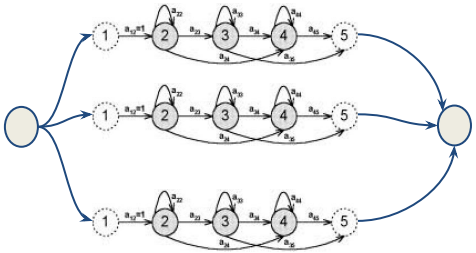
2. Sum all word wFST into a single wFST (P)



3. Turn each phone HMMs into an wFST



4. Sum them into a single wFST (A)



5. Compose A and P into a single (large) wFST

A°P

Et voilà!

Pronunciation models

Using a pronunciation dictionary

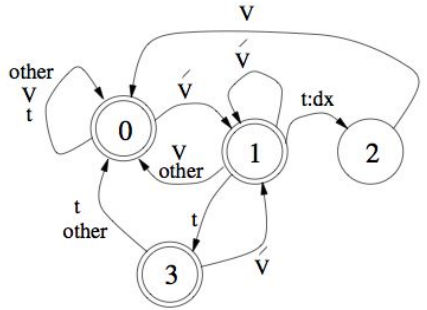
orthography	SAMPA	IPA
counting	kA_wntlN	kɑʷntɨŋ
counting(1)	kA_wnlN	kɑʷnɨŋ
amortization	@mOrt@zeS@n	əmɔrtəzeʃən
amortization(1)	{mOrt@zeS@n	æmɔrtəzeʃən
amortization(2)	@mOrtA_jzeS@n	əmɔrtɑʲzeʃən
..etc		

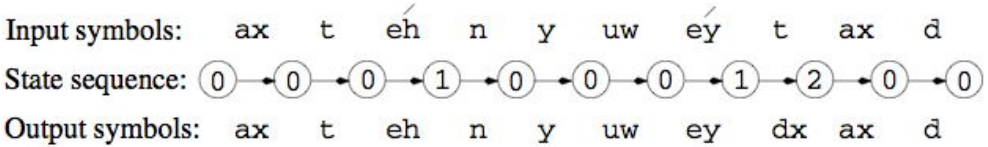
→ **But what is you dont have pronunciations for all of your words? Or you want to add words?**

1. phonological rules (hand crafted)

- ex: english flapping

$$/t/ \rightarrow [\text{dx}] \quad / \quad [+\text{vowel}] \_\_ \begin{bmatrix} +\text{vowel} \\ -\text{stress} \end{bmatrix} .$$





represent rules as a wFST!

Name	Rule	Example	Prob
Syllabic Rules*			
Syllabic n	[ax ix] n → en	butt**on**	.35
Syllabic m	[ax ix] m → em	bott**om**	.32
Syllabic l	[ax ix] l → el	bott**le**	.72
Syllabic r	[ax ix] r → axr	butt**er**	.77
Flapping	[tcl dcl] [t d]→ dx /V ___ [ax ix axr]	bu**tt**on	.87
Flapping-r	[tcl dcl] [t d]→ dx /V r ___ [ax ix axr]	bar**t**er	.92
H-voicing	hh → hv / [+voice] ___ [+voice]	**a**head	.92
L-deletion	l → Ø/ ___ y [ax ix axr]	mi**ll**ion	*n/a*
Gliding	iy → y / ___ [ax ix axr]	colon**i**al	*n/a*
Nasal-deletion	[n m ng] → Ø/ ___ [-voice -consonant]	ra**n**t	*n/a*
Function words			
h-deletion	h → Ø/ # ___	**h**e, **h**im	*n/a*
w-deletion	w → Ø/ # ___	**w**ill, **w**ould	*n/a*
dh-deletion	dh → Ø/ # ___	**th**is, **th**ose	*n/a*
Dental-deletion	[tcl dcl] [t d] → Ø/ [+vowel] ___ [th dh]	brea**d**th	*n/a*
Final dental-deletion	([tcl dcl]) [t d] → Ø/ [+cons +continuant] ___ #	sof**t** (as)	*n/a*
Slur	ax → Ø/ [+consonant] ___ [r l n] [+vowel]	cam**e**ra	*n/a*
Stressed slur	[+vowel +stress] r → er	w**ar**ts	*n/a*
Pre-stress contraction	ax → Ø/ [+cons] ___ [+cons] [+vowel +stress]	**s**enility	*n/a*
Ruh-reduction	r ax → er / [-word bdry] ___ [-word bdry]	sep**ar**able	*n/a*
Transitional stops			
t-introduction	Ø→ tcl / [+dental +nasal] ___ [+fricative]	prin[**t**]ce	*n/a*
t-deletion	[tcl] → Ø/ [+dental +nasal] ___ [+fricative]	prin**ts**	*n/a*

2. phonological rules (data driven)

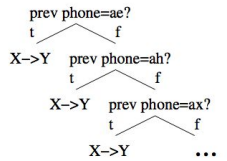
- using a phone recognizer to learn a pronunciation dictionary

a	baseball						game		
ax	b	ey	s	b	ao	l	g	ey	m
ax	b	eh ey	s	b	el		g	eh	m
ax	b	eh ey	s	b	el		g	eh	m

word sequence
canonical phone sequence
aligned surface phones
proposed new pronunciations

- inducing pronunciation rules
  - trigram models
  - decision trees

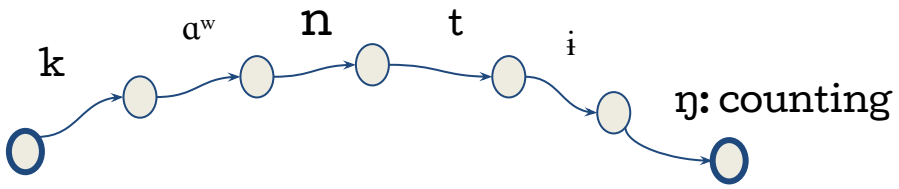
$$P(S^i) = \prod_{j=1}^{n} (P(s_j^i | c_{j-1}, c_j, c_{j+1}) \quad .$$



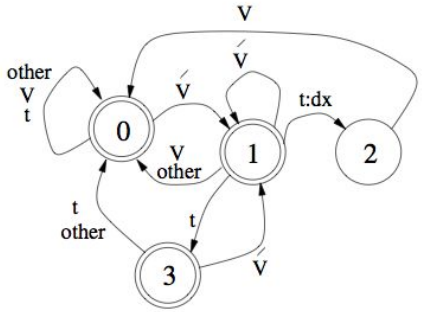
→ learning a prononciation wFST

Improved recipe:

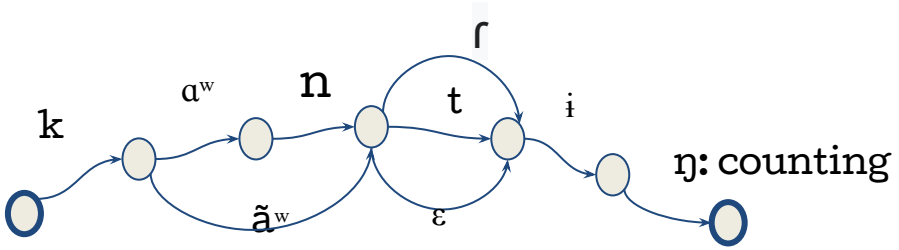
1. **Build a lexicon wFST (L)**



2. **Build or learn a prononciation wFST (F)**

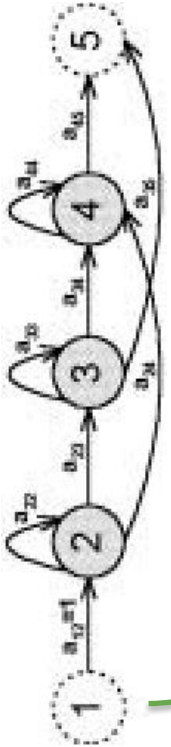


3. **Compose: P=F°L**



Decoding: the Viterbi algorithm!

$Q^* = \text{argmax}_Q P(X|Q)$



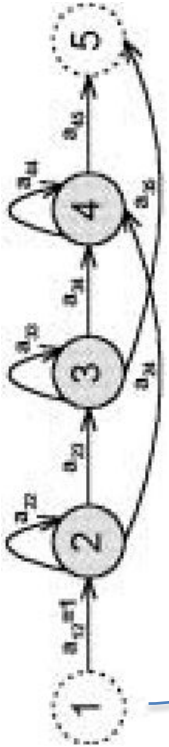
$a_{21}=p(q_2|q_1)$

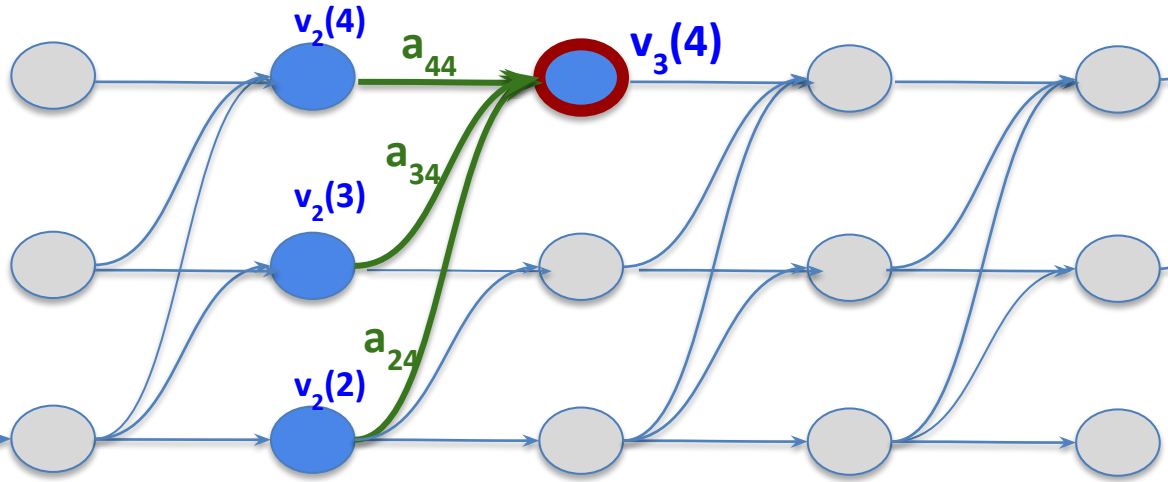
$v_1(2)=a_{21}.b_2(x_1)$

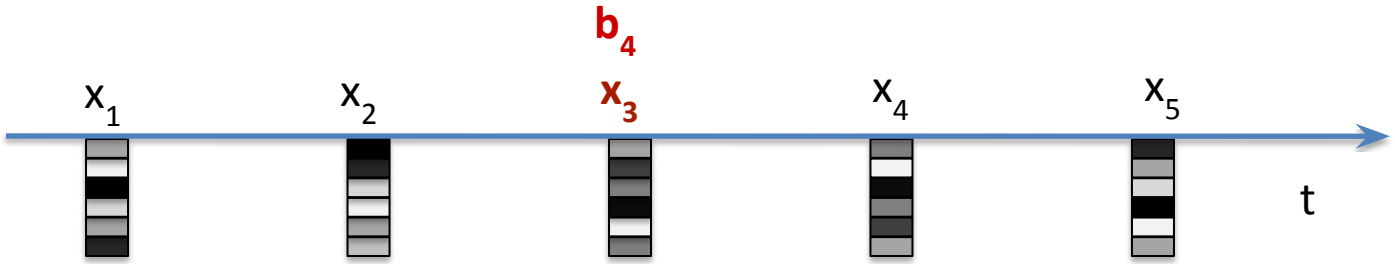
$b_2(x_1)=p(x_1|q_2)$

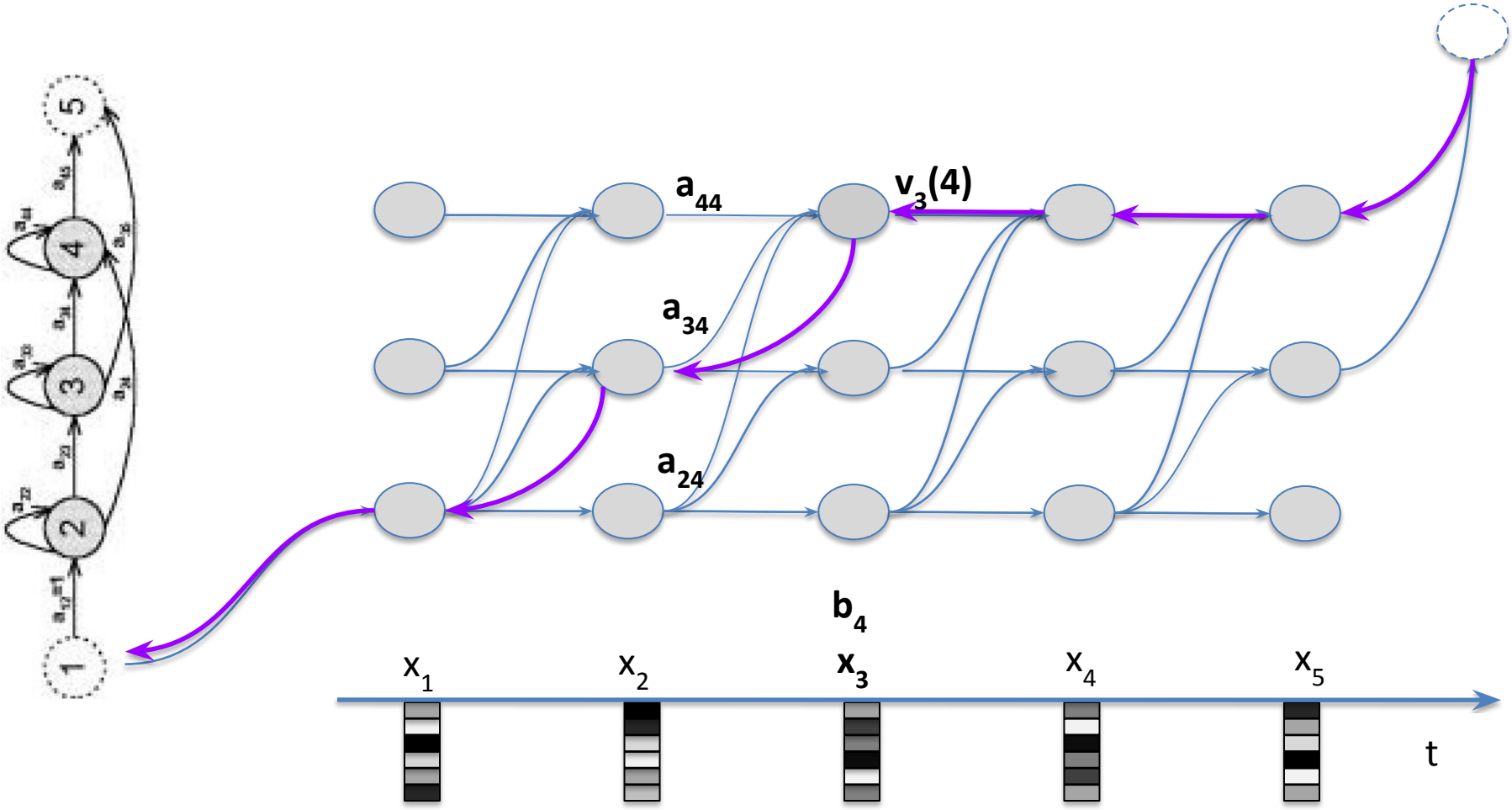
x_1 x_2 x_3 x_4 x_5 t

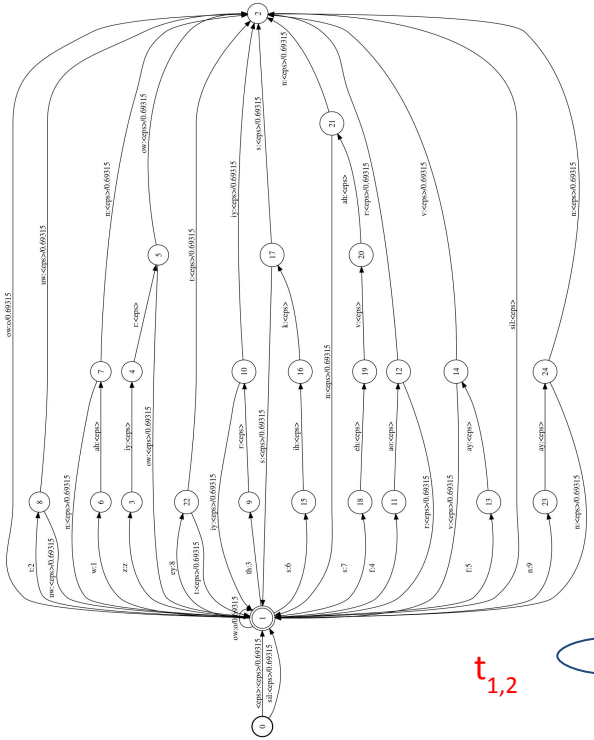
$$v_t(j) = max_{i=1:N}\, v_{t-1}(i)\, a_{ij}\, b_j(x_t)$$








a44
v3(4)
a34
a24
b4
x1
x2
x3
x4
x5
t



K states

N-1

1

0

$t_{1,2}$

d_{ij}

x_1 x_2 x_3 ….

x_N

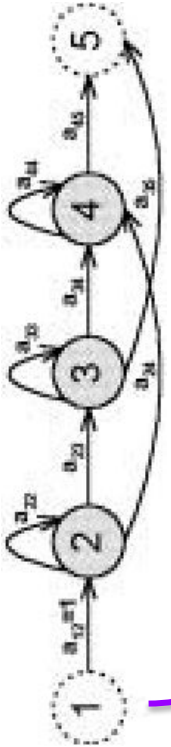
N frames

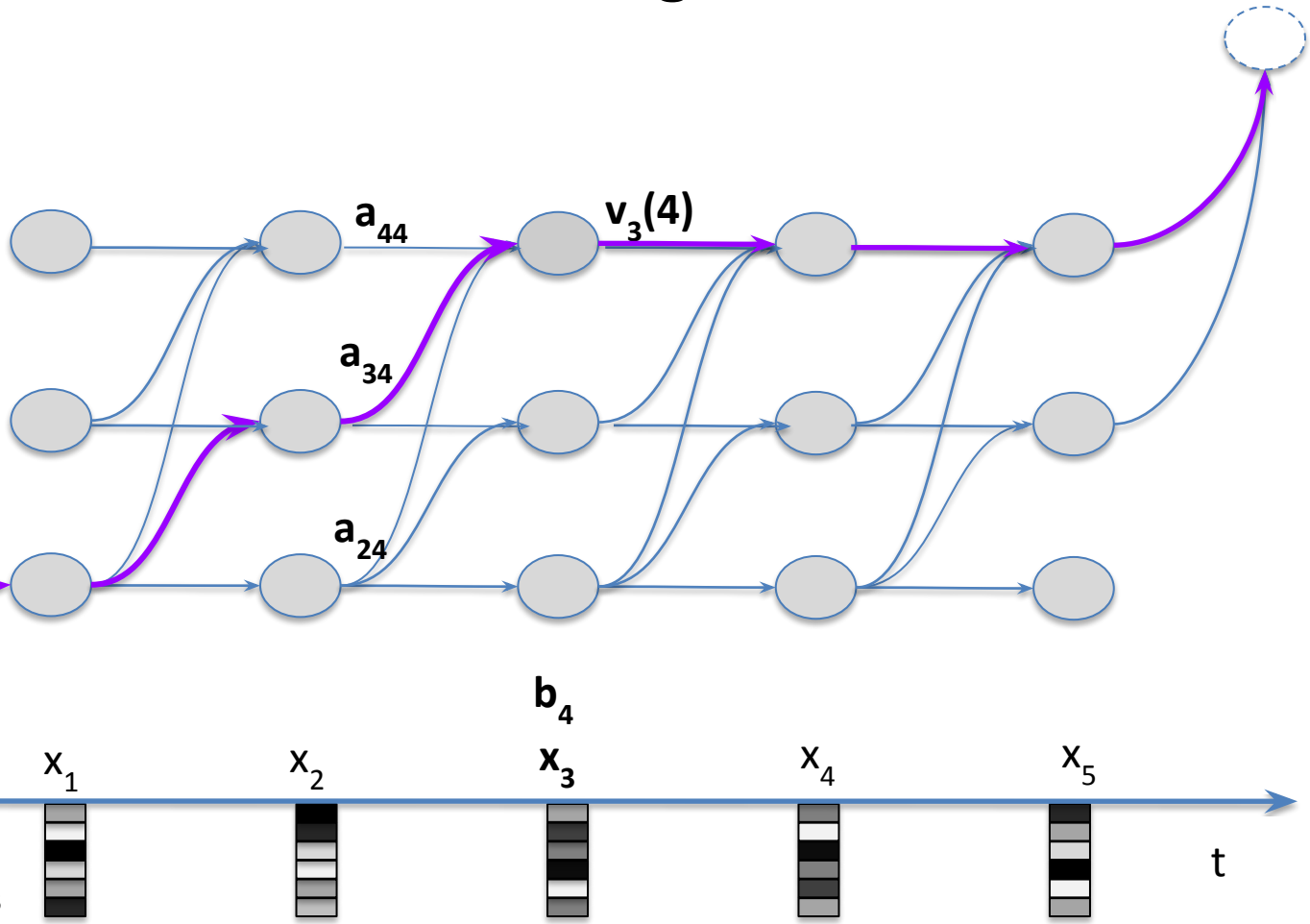
2

1

0

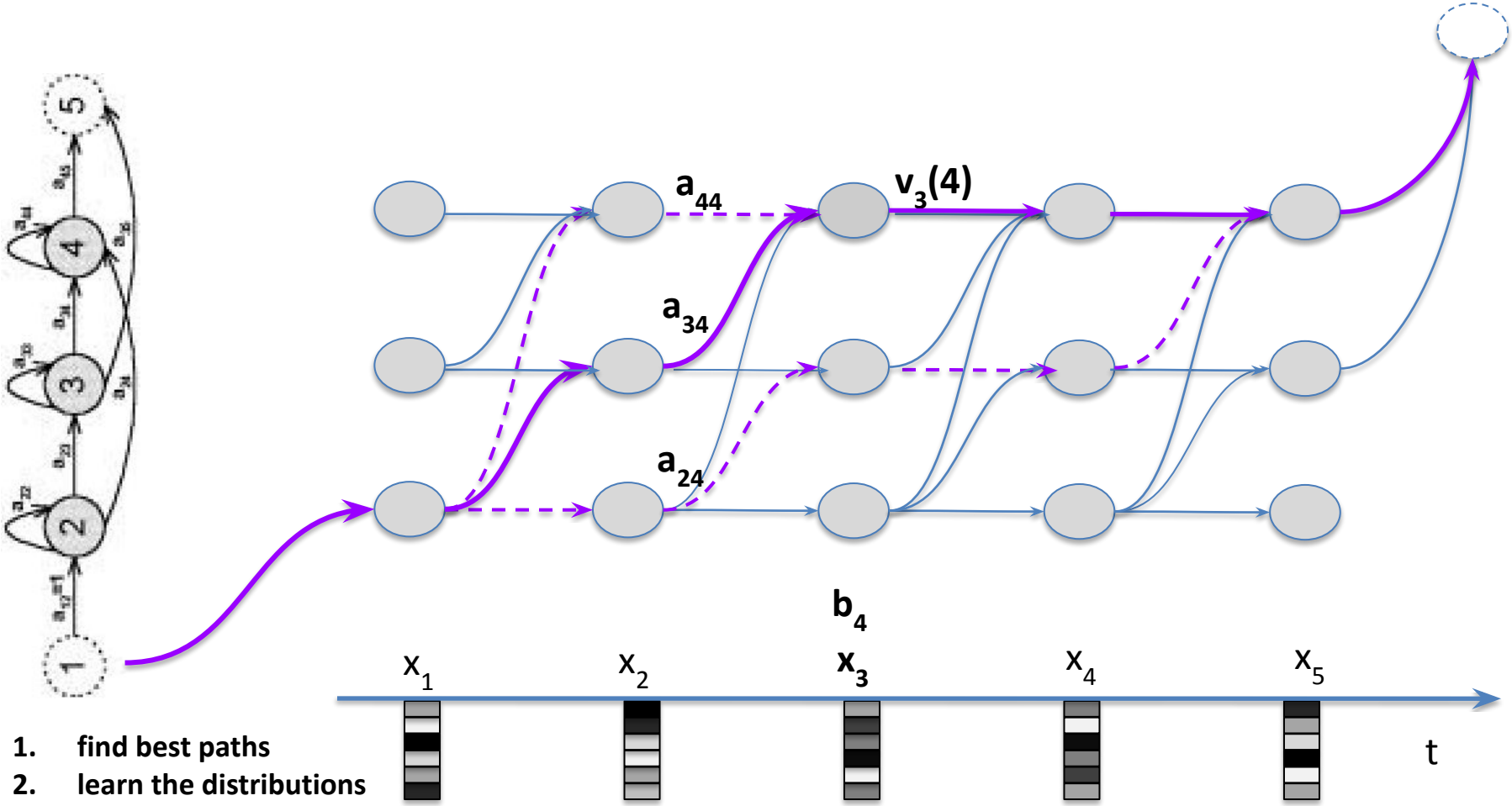
Learning: the Forward-Backward algorithm!





1. find best path
2. learn the distributions

Learning: the Forward-Backward algorithm!



1. find best paths
2. learn the distributions

Reminder: the fundamental equation of ASR

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

$X_T = x_1, ..., x_T$

A sequence of features frames

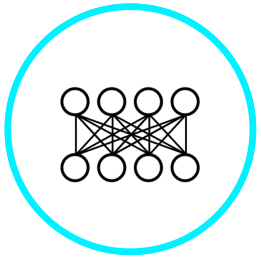
$W_M = w_1, ..., w_M$

A sequence of words

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

ACOUSTIC MODEL

LANGUAGE MODEL



Language modeling for ASR

- HMMs (again!)

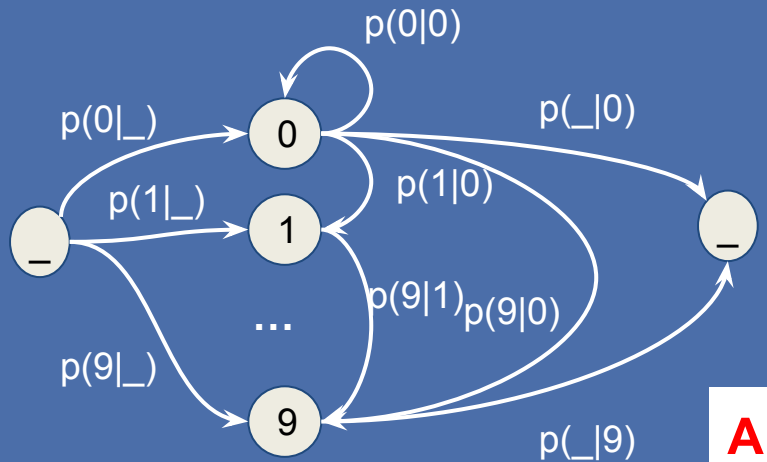
Reminder: bigram language models

$$P(W_n \mid W_1, W_2, \ldots, W_{n-1}) \approx P(W_n \mid W_{n-1})$$

Reminder: bigram language models

$$P(W_n \mid W_1, W_2, \ldots, W_{n-1}) \approx P(W_n \mid W_{n-1})$$

Eg, with digits zero-ten



Another wFST!!

Reminder: trigram language models

$P(W_n|W_1 \ldots W_{n-1}) \sim P(W_n|W_{n-2}W_{n-1})$

A larger wFST!!

$$P(w_1, \ldots, w_L) = \prod_{i=1}^{L+1} P(w_i|w_{i-2}w_{i-1})$$

$$P(w_i|w_{i-2}w_{i-1}) = \frac{c(w_{i-2}w_{i-1}w_i)}{c(w_{i-2}w_{i-1})}$$

count in the corpus

$P(W_n|W_1 \ldots W_{n-1}) \sim P(W_n|W_{n-2}W_{n-1})$

Learning ngram models:

$$P(w_1, \ldots, w_L) = \prod_{i=1}^{L+1} P(w_i|w_{i-2}w_{i-1})$$

$$P(w_i|w_{i-2}w_{i-1}) = \frac{C(w_{i-2}w_{i-1}w_i)}{C(w_{i-2}w_{i-1})}$$

count in the corpus

Reminder: smoothing, etc

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

- unobserved ngrams
  - most ngrams have been seen 0 time
  - problem grows exponentially with n
  - true 0s versus sampling
  - solutions:
    - smoothing
    - backoff, interpolation,

    clustering

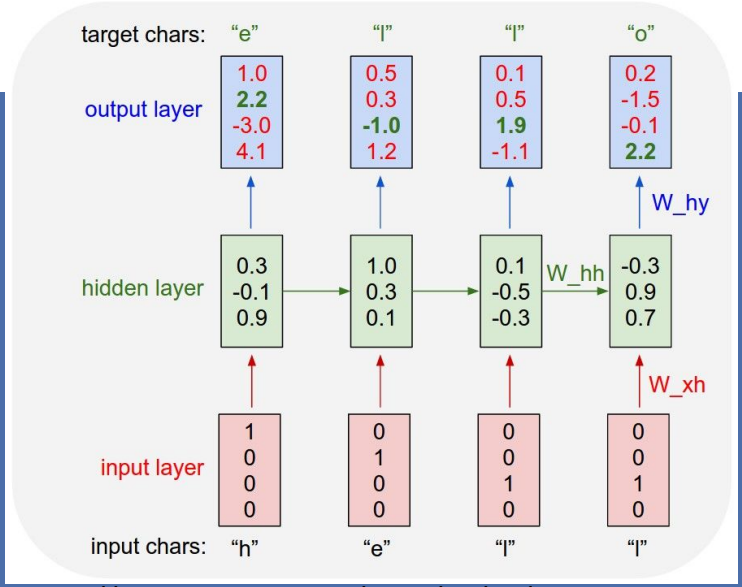
$$P_{\text{Laplace}}(w_i) = \frac{c_i + 1}{N + V}$$

$$c_i^* = (c_i + 1)\frac{N}{N + V}$$

$$\begin{aligned}\hat{P}(w_n|w_{n-2}w_{n-1}) &= \lambda_1 P(w_n|w_{n-2}w_{n-1}) \\ &+ \lambda_2 P(w_n|w_{n-1}) \\ &+ \lambda_3 P(w_n)\end{aligned}$$

A word on RNN language models

- RNNs can approximate P(W) better than ngram models by relying on indefinite amount of past information
- But, the Markov assumption no longer holds
- Hence Viterbi lo longer possible (greedy search, beam search)



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Summary so far: HMM-GMM for large vocabulary ASR

- Using phonemes as building blocks for Acoustic Modeling (A)
  - *modeling temporal dynamics with three states HMMs*
  - *modeling coarticulation by replacing phonemes with tied triphones*
- Building a pronunciation model for words (P)
  - *Modeling pronunciation variations through wFSTs*
- Building a language model (G)
  - *Using n-gram wSFT*
- Combining everything
  - *a single model (a large wFST!): A∘P∘G*
  - *Decoding using Viterbi*
  - *Learning : forward-backward (A), by hand (P), by counting (G)*

Limits

A lot of linguistic expertise

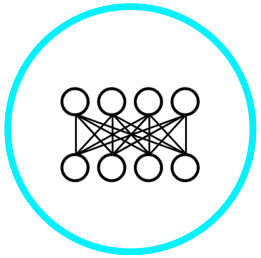
- List of phonemes
- Pronunciation dictionary
- List of phonological rules

Reasonable amounts of annotations (50h+)

- Parallel corpus of speech+transcription

Assumptions and approximations

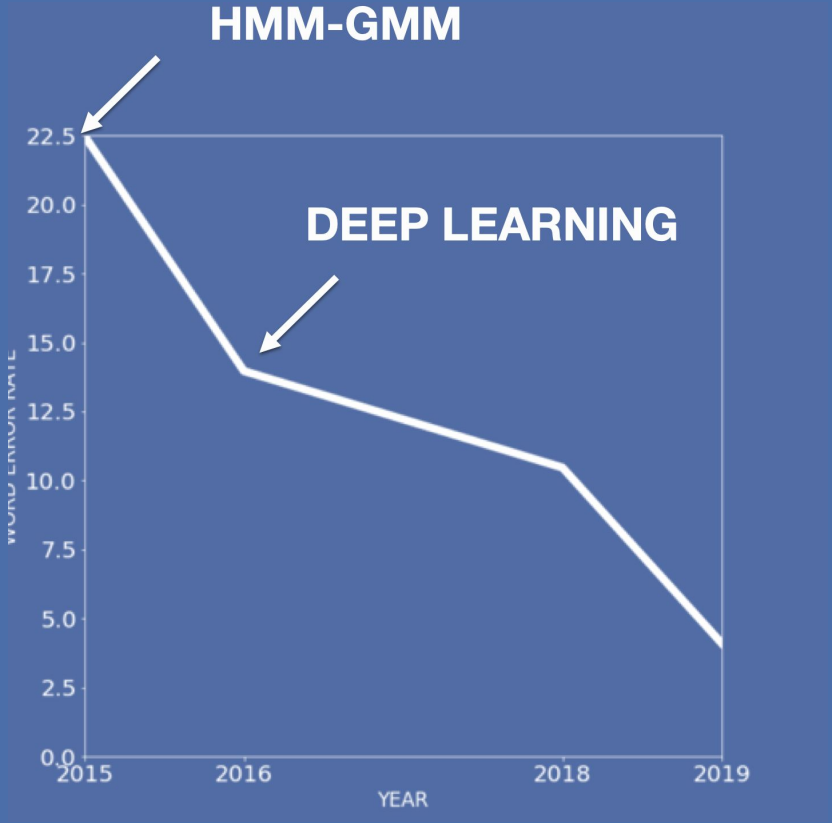
- Phone states= stationary emission probability
- Local context assumption (Markov property)
- Linguistic knowledge on phonemes, rules, etc



Towards end-to-end

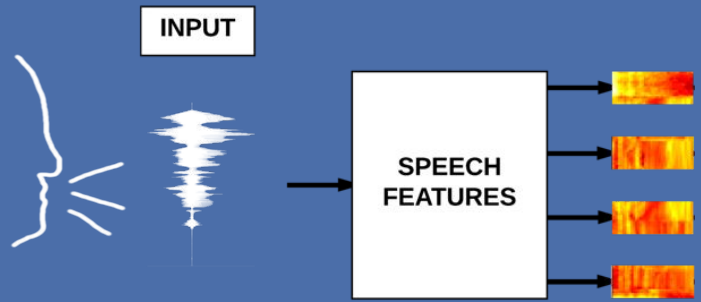
- Hybrid Models: HMM-DNN
- Learning audio features
- CTC

HMM-DNN



General idea: reduce the amount of expertise and assumptions by using more generic models and more data

HMM-DNN



Acoustic Model (A)

tristate HMM GMM

Pronunciation & Language Model (G°P)

HMM

“MY NAME IS NEIL”

$X_T = x_1, ..., x_T$

A sequence of features frames

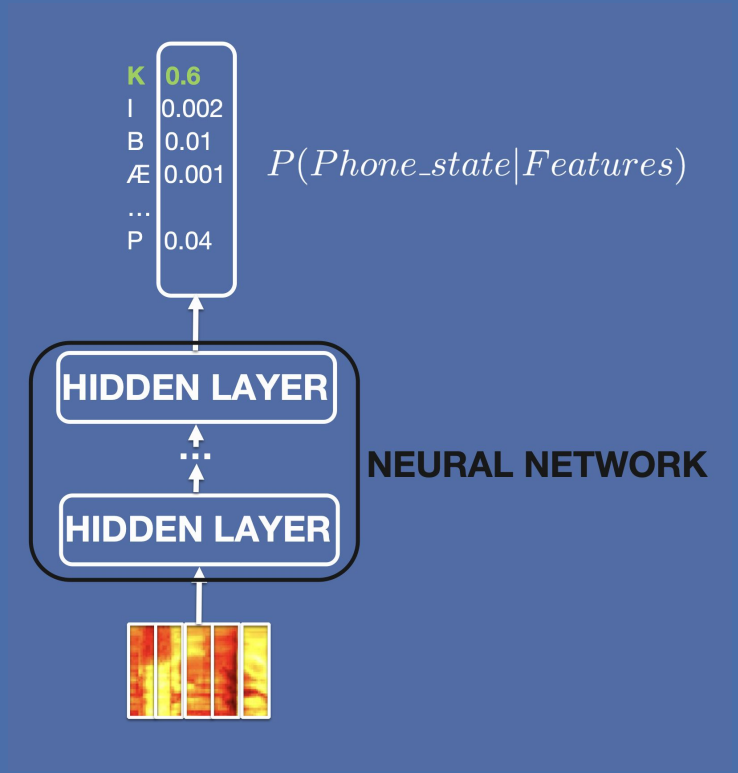
$Q_P=q_1, q_P$

A sequence of phonemes states

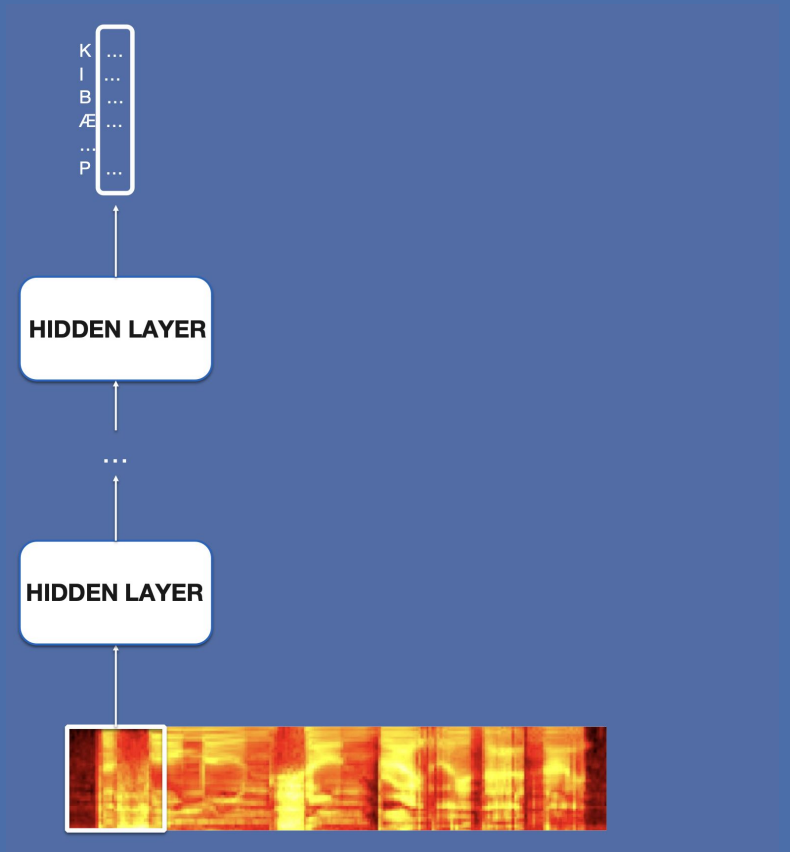
$W_M = w_1, ..., w_M$

A sequence of words

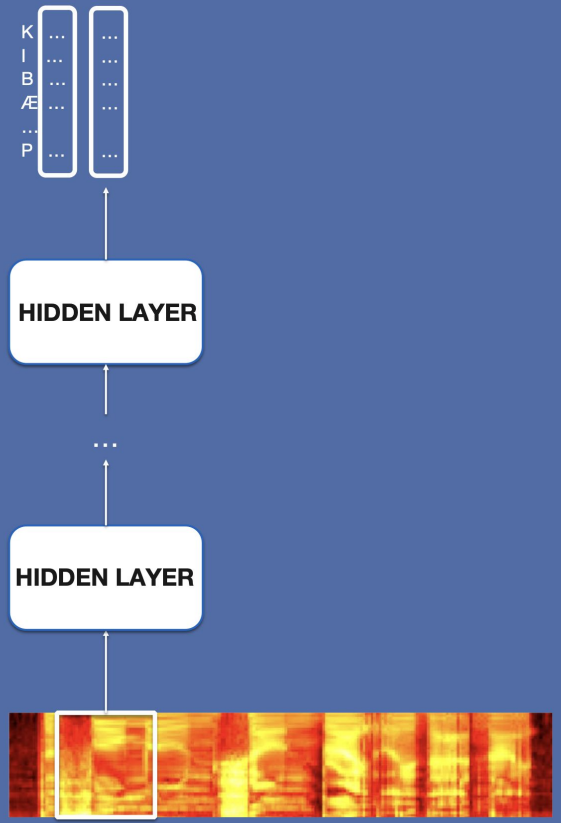
The neural network takes features as input and outputs probabilities over phone states



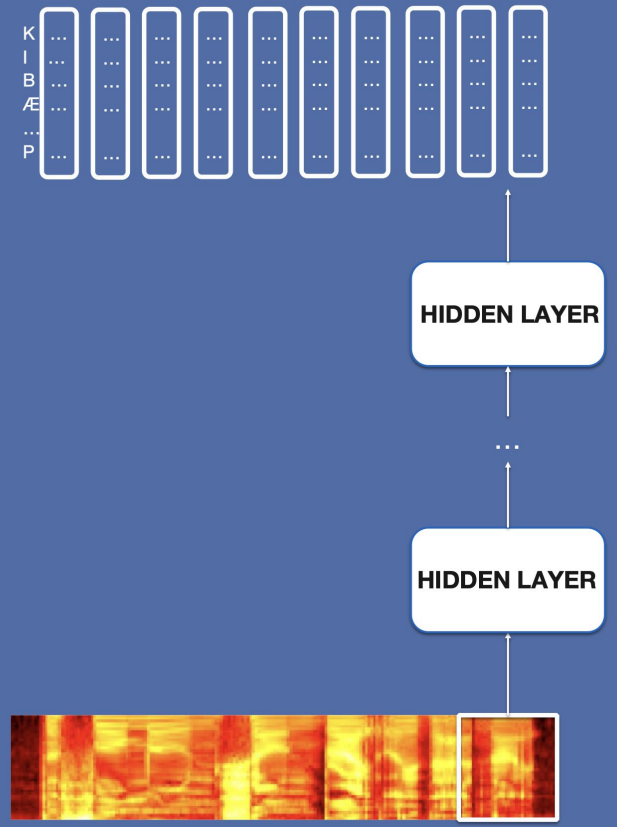
The neural network takes features as input and outputs probabilities over phone states



The neural network takes features as input and outputs probabilities over phone states

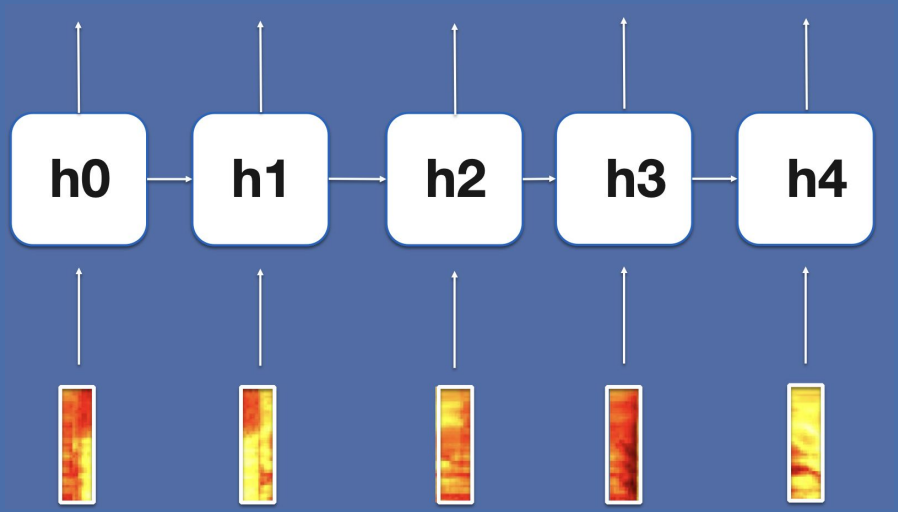


The neural network takes features as input and outputs probabilities over phone states



Recurrent acoustic model

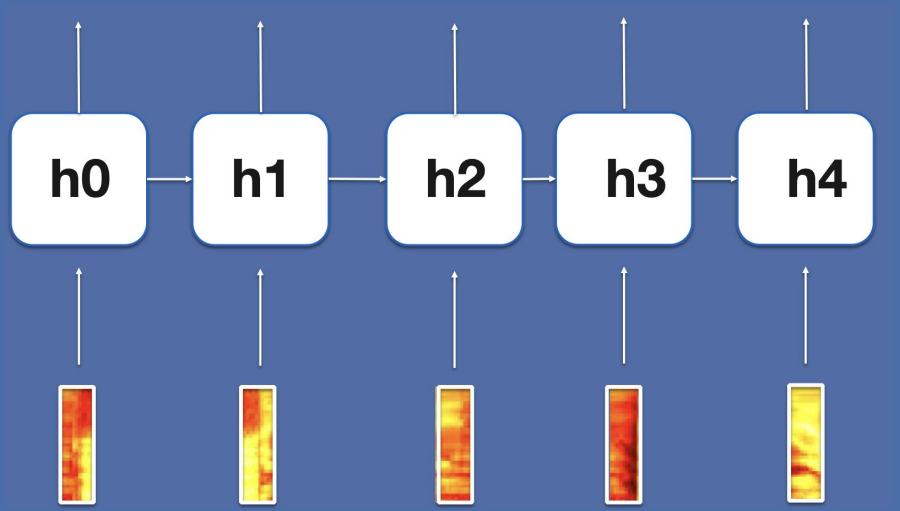
- can model any sequences (can model non stationarity)
- can encode dependencies (coarticulation)



$$h_0 = \tanh(W_x x_0)$$

$$h_t = tanh(W_x x_t + W_h h_{t-1} + b_h) \text{ for } t > 0$$

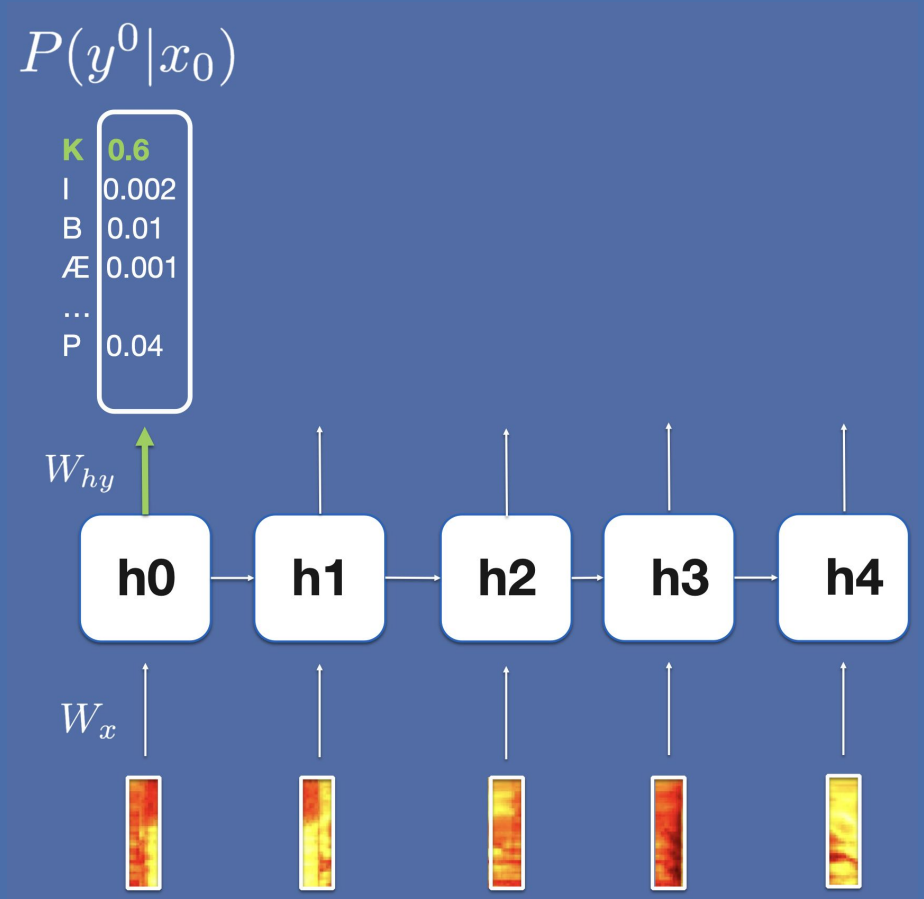
$$y^t = Softmax(W_{hy} h_t + b_y)$$



$$h_0 = \tanh(W_x x_0)$$

$$h_t = tanh(W_x x_t + W_h h_{t-1} + b_h) \text{ for } t > 0$$

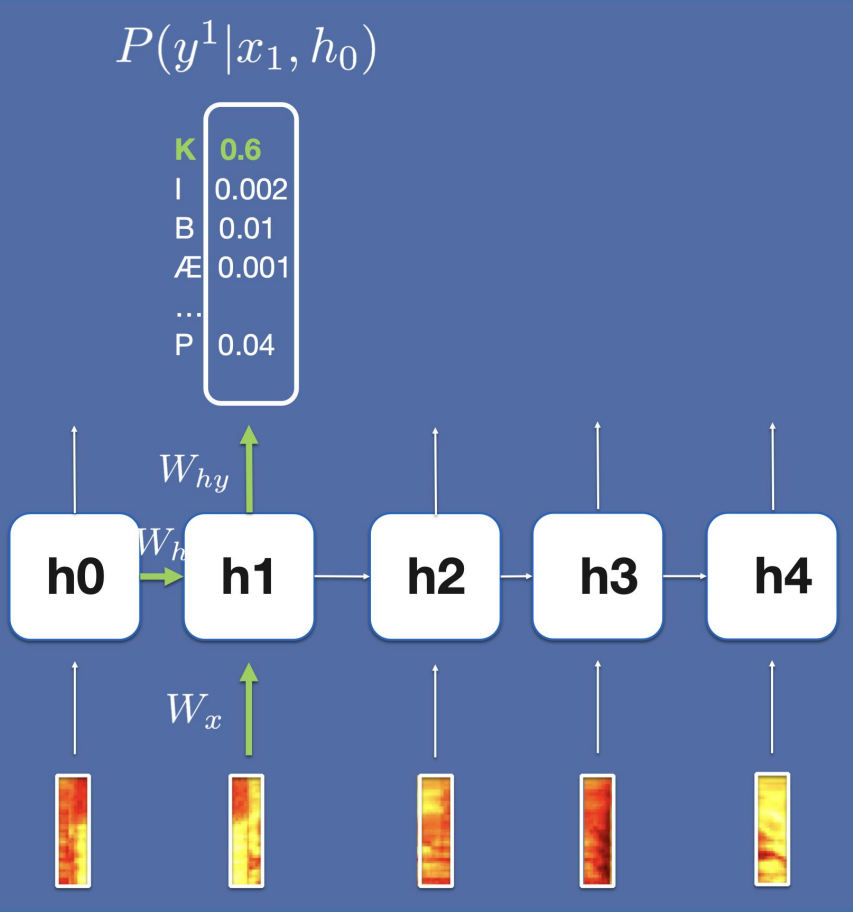
$$y^t = Softmax(W_{hy} h_t + b_y)$$



$$h_0 = \tanh(W_x x_0)$$

$$h_t = tanh(W_x x_t + W_h h_{t-1} + b_h) \text{ for } t > 0$$

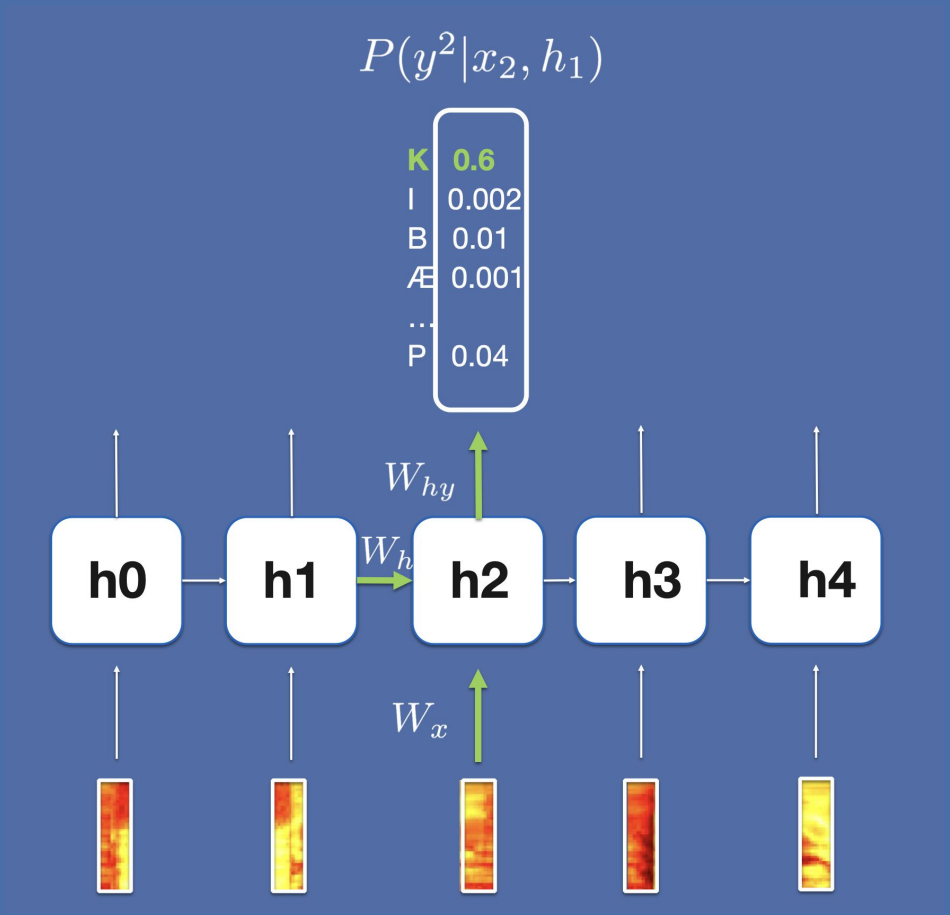
$$y^t = Softmax(W_{hy} h_t + b_y)$$



$$h_0 = \tanh(W_x x_0)$$

$$h_t = tanh(W_x x_t + W_h h_{t-1} + b_h) \text{ for } t > 0$$

$$y^t = Softmax(W_{hy} h_t + b_y)$$

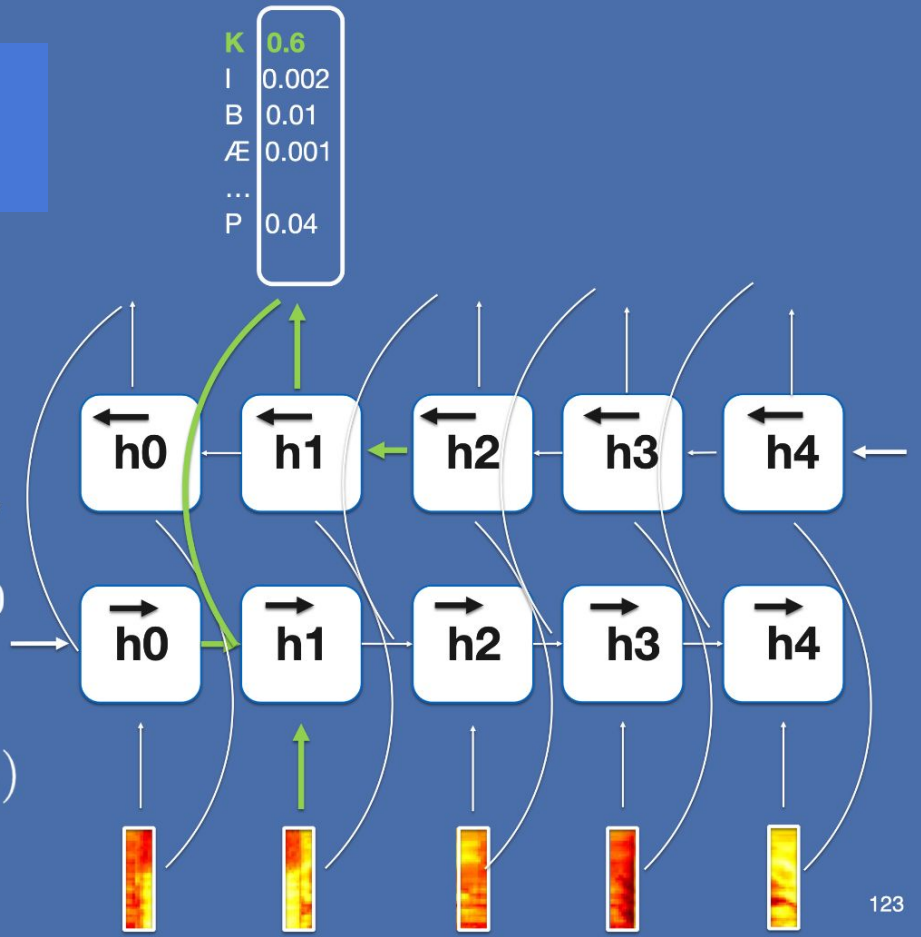


Improving modeling of context with bi-directionnal RNNs

$$\overrightarrow{h}_0 = tanh(W_x x_0)$$
$$\overleftarrow{h}_T = tanh(W_x x_T)$$

$$\overleftarrow{h}_t = tanh(W_x x_t + W_{\overleftarrow{h}} \overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}) \text{ for } t < T$$
$$\overrightarrow{h}_t = tanh(W_x x_t + W_{\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}) \text{ for } t > 0$$

$$y^t = Softmax(W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y)$$

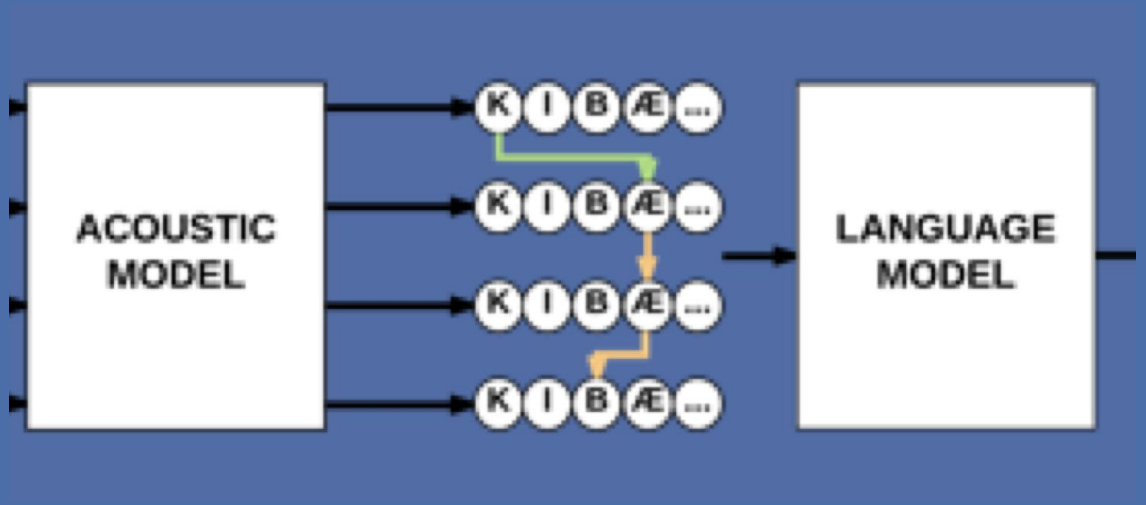


HMM-DNN

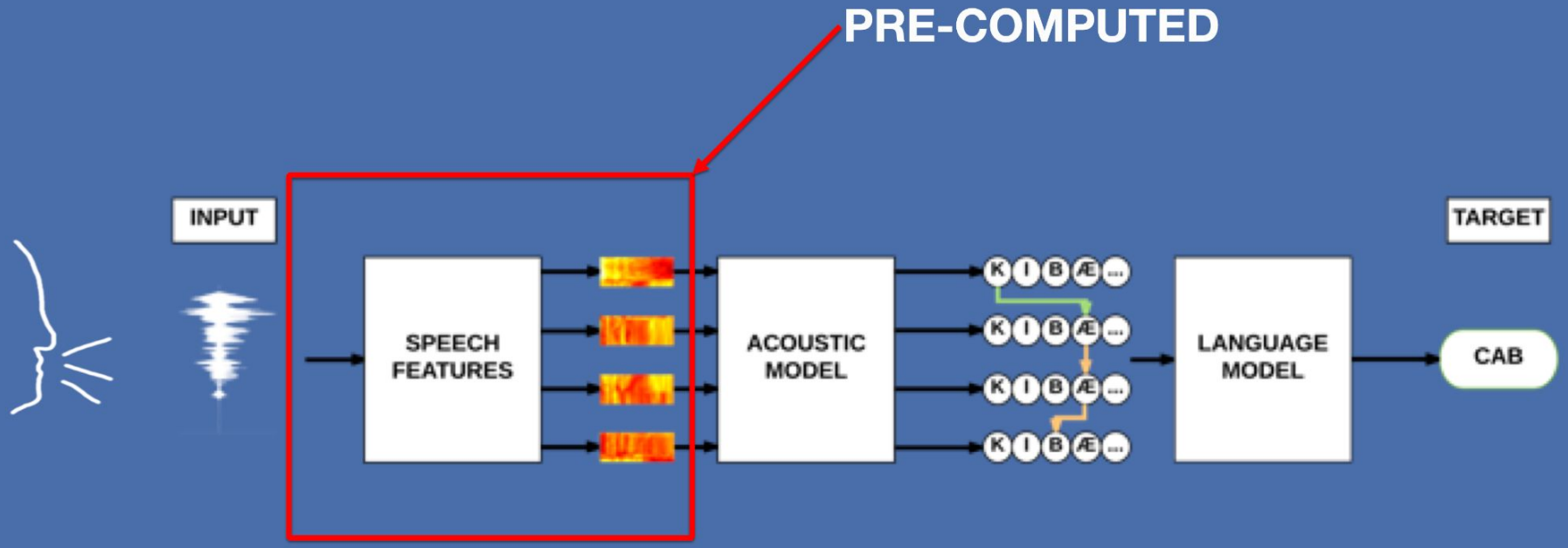
Labels are the phone states obtained after forced alignment with Viterbi

Train with a cross-entropy loss by backpropagation

Decode by building a decoding lattice and passing on to the Pronunciaton and Language model



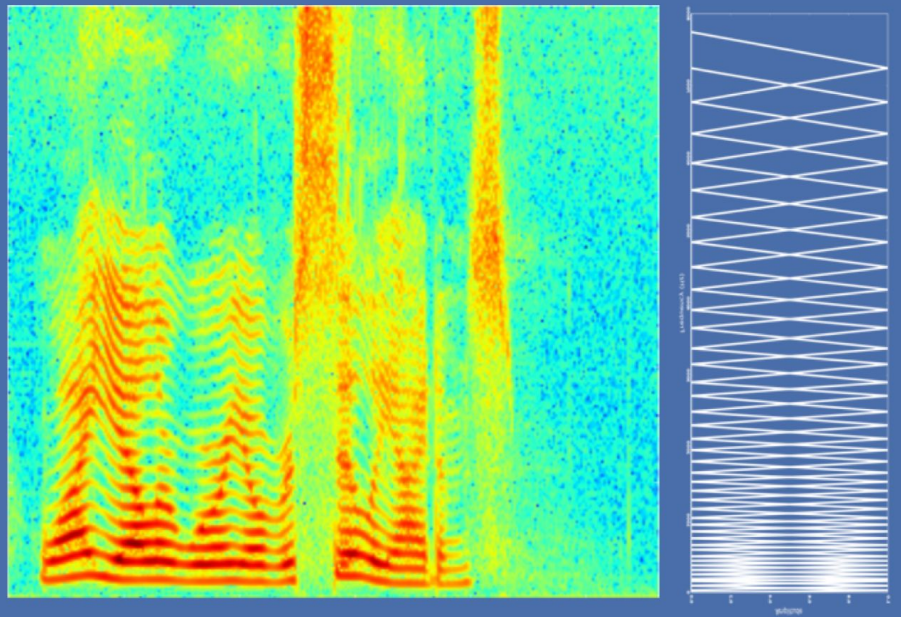
HMM-DNN


PRE-COMPUTED
INPUT
SPEECH FEATURES
ACOUSTIC MODEL
K I B Æ ...
K I B Æ ...
K I B Æ ...
K I B Æ ...
LANGUAGE MODEL
TARGET
CAB

Can we learn the mel-filterbanks?

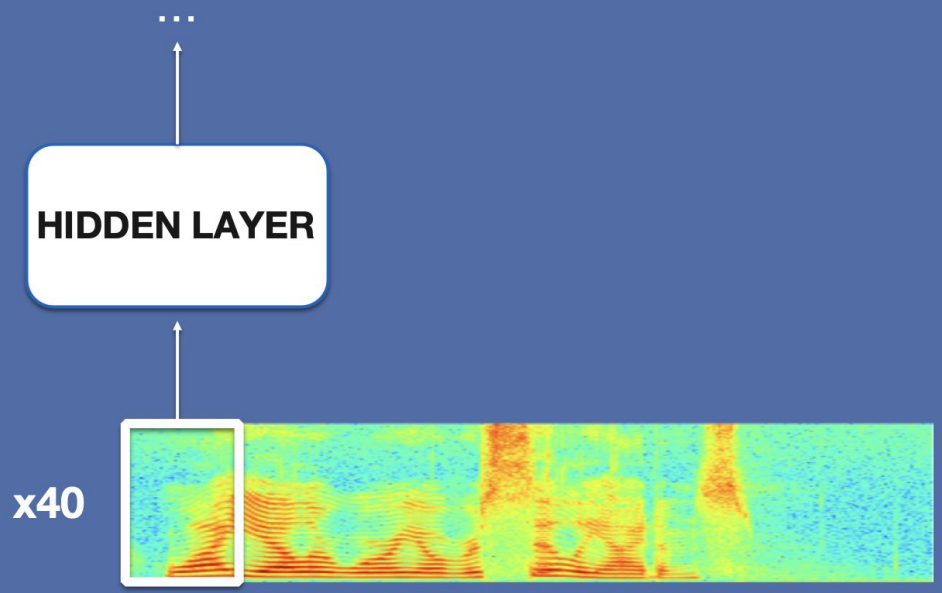
$$Melfbank_j(k) = \sum_{\omega=0}^{256} Spectrogram(k,\omega) Melfilter_j(\omega)$$

They are convolutions over the frequency axis

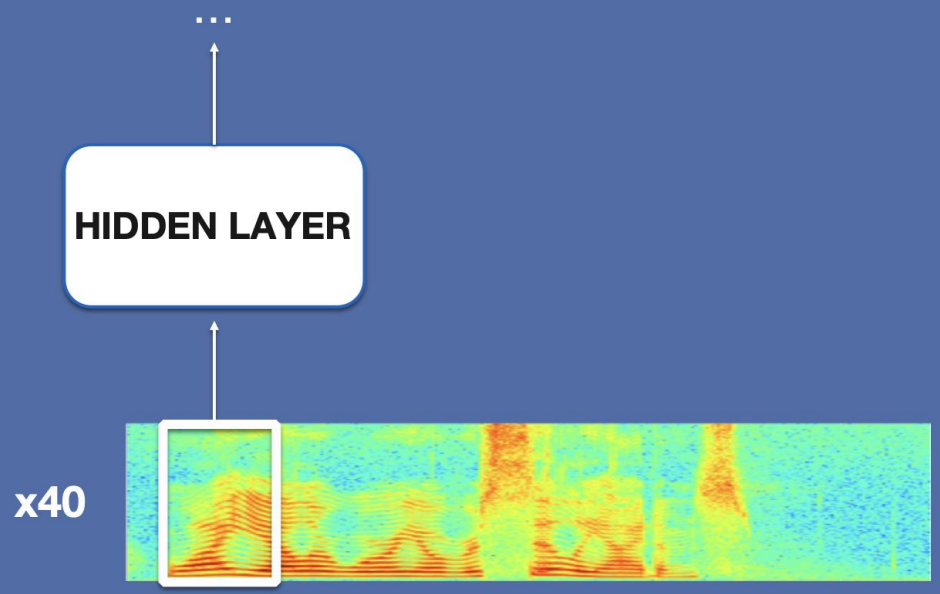


(Sainath et al. learning filterbancks with a deep NN)

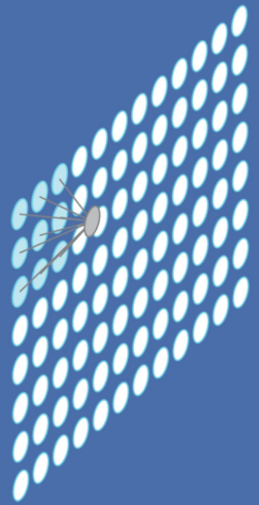
Can we learn the mel-filterbanks?



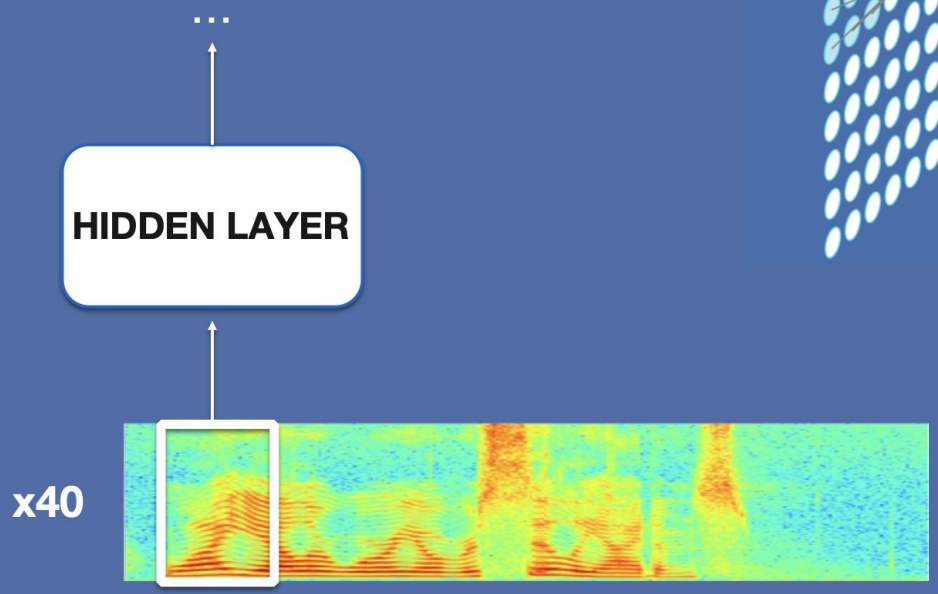
Can we learn the mel-filterbanks?



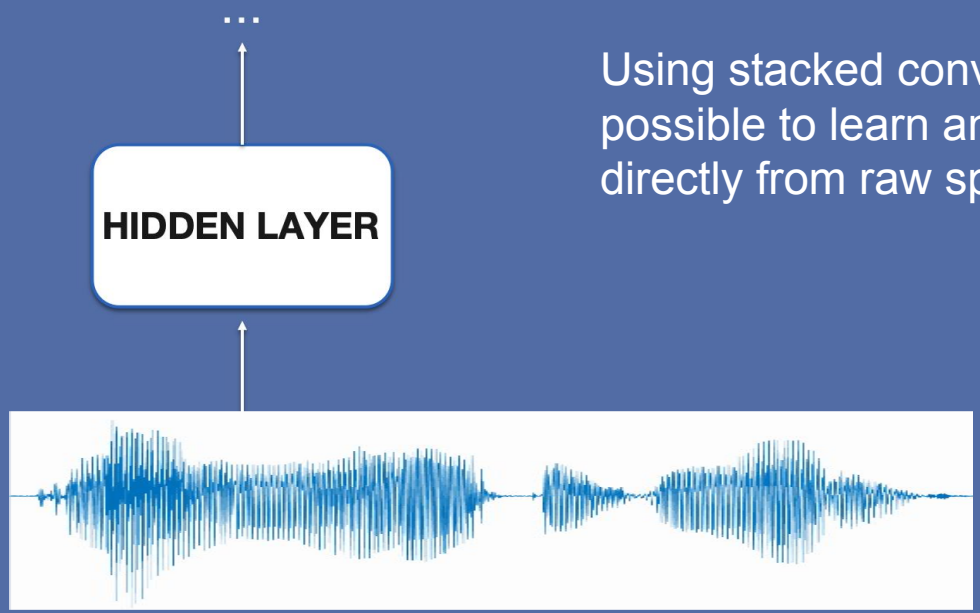
Can we learn the mel-filterbanks?



Kernel size : 1 x 3 x 3
Number of kernels : 1

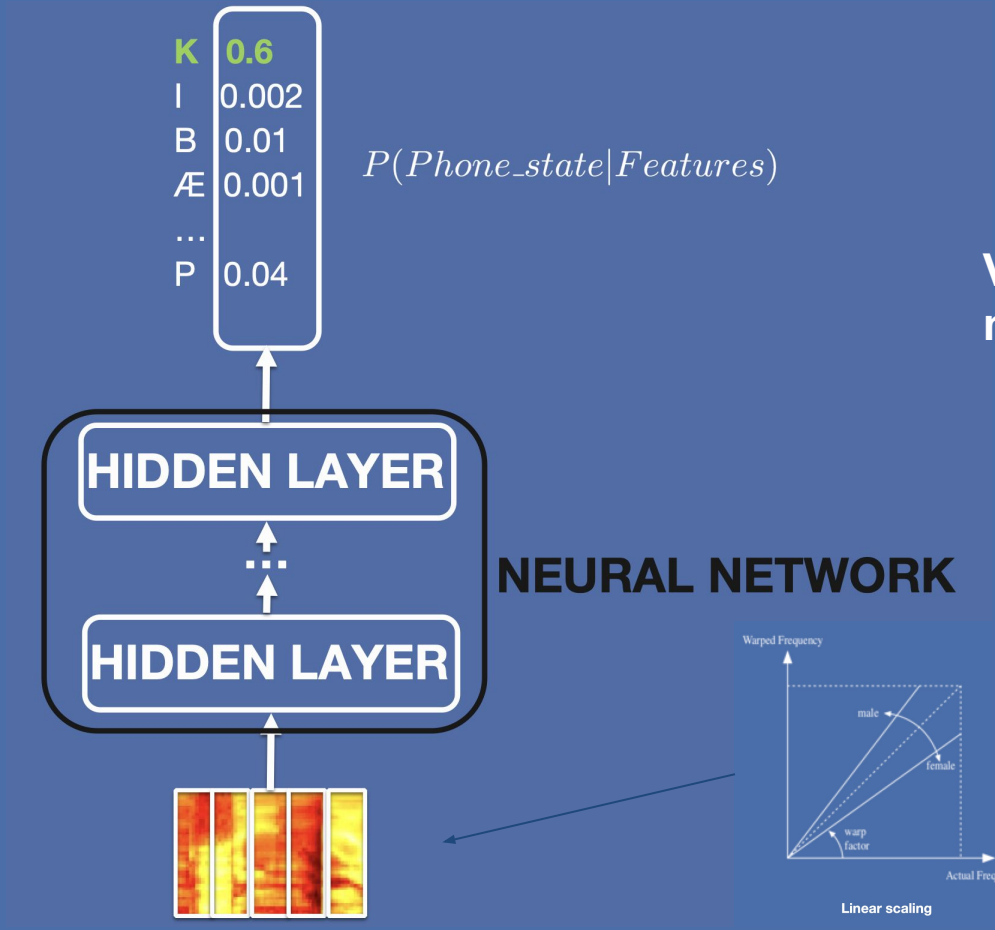


Can we learn the spectrogram itself?

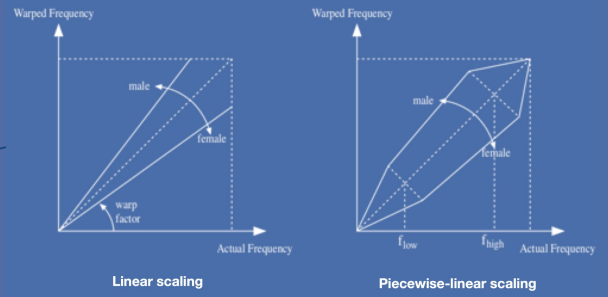


Using stacked convolutions, it is possible to learn an acoustic model directly from raw speech

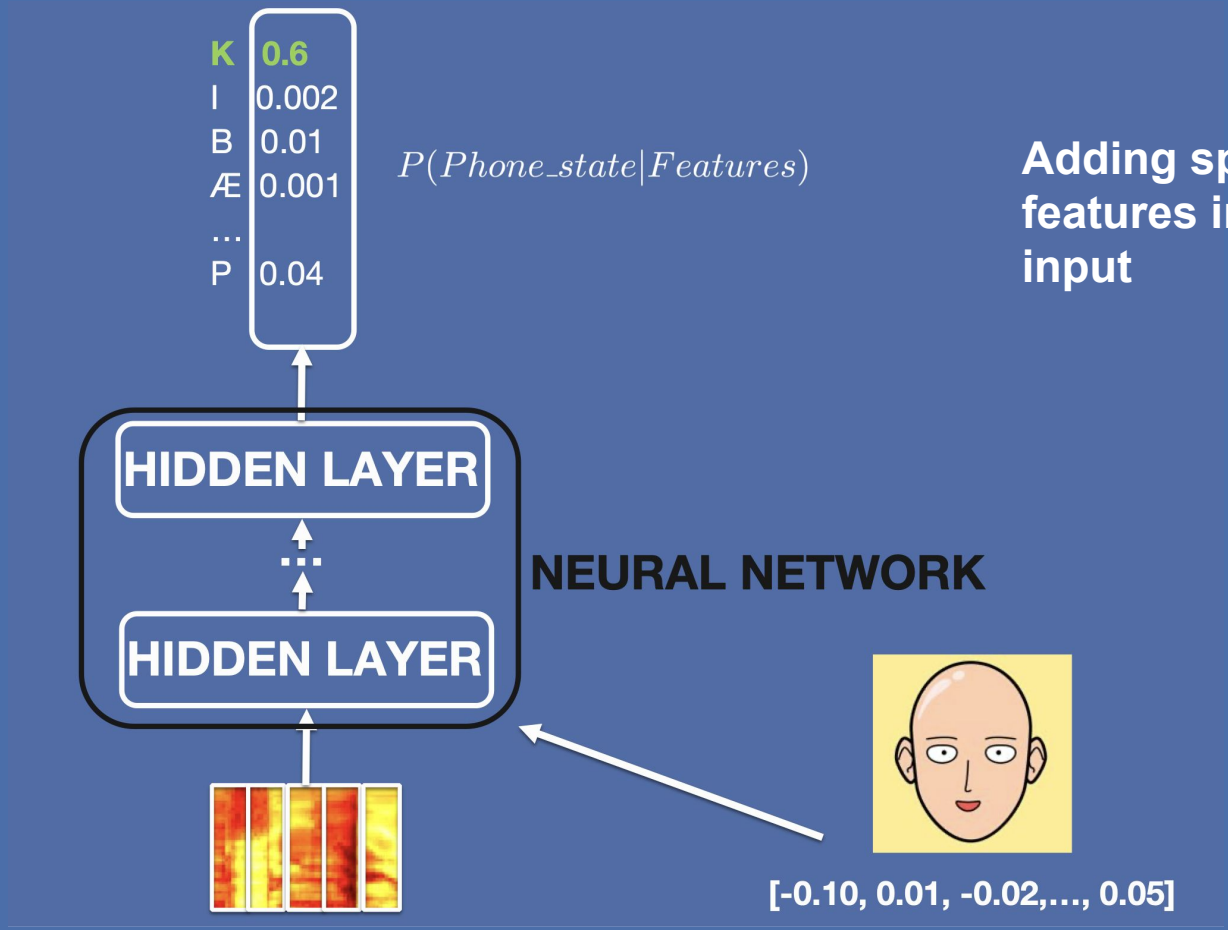
Handling variability



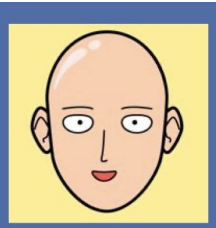
Vocal tract length normalization



Handling variability



Handling variability



[0, 0, 1,..., 0]

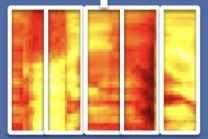
K	0.6
I	0.002
B	0.01
Æ	0.001
…	
P	0.04

$P(phoneme|features)$

HIDDEN LAYER

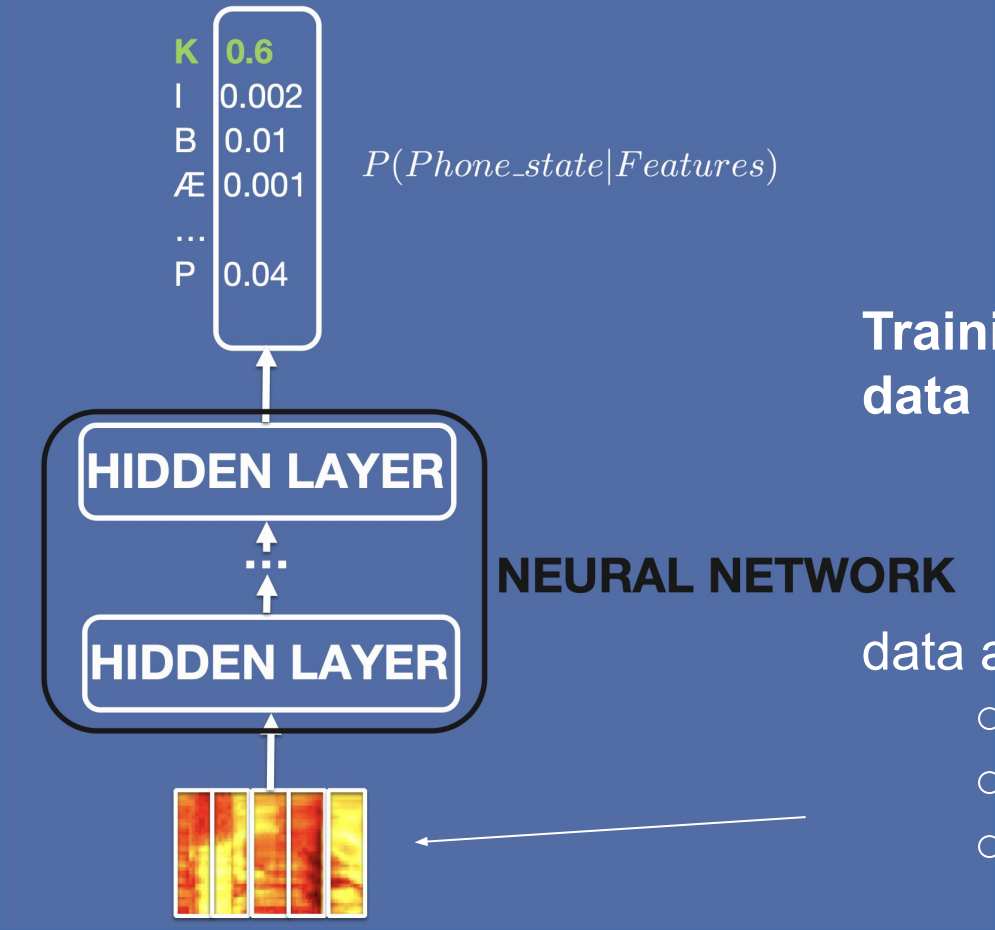
HIDDEN LAYER

NEURAL NETWORK



Multitask learning

Handling variability



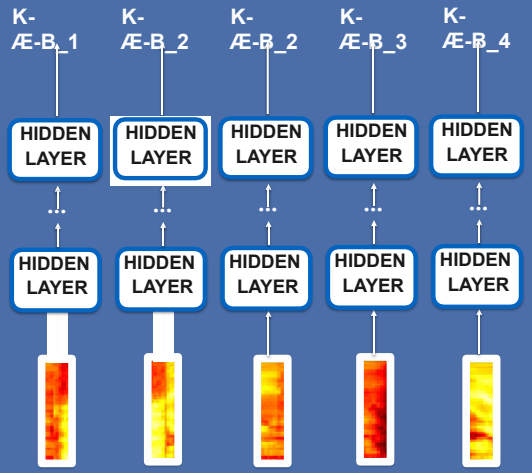
Training on LOTS of data

data augmentation

- pitch change
- using TTS
- adding noise, reverb

HMM-DNN (Hybrid) Recap

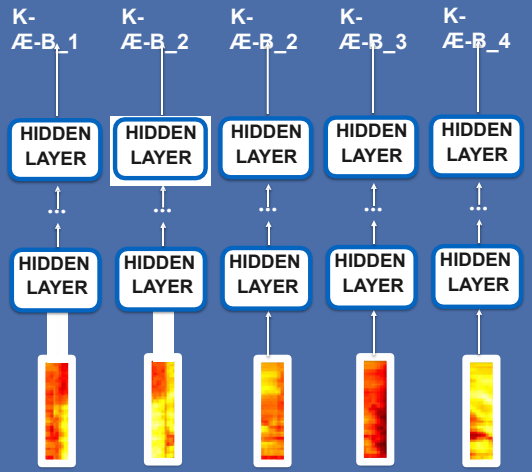
- Extract speech features
- Train a Hidden Markov Model with Gaussian Mixture Model for emissions
- Use the Viterbi algorithm to extract an alignment of feature frames with states
- Train a Neural Network to predict each state from its frame (or from the raw audio)



HMM-DNN Recap

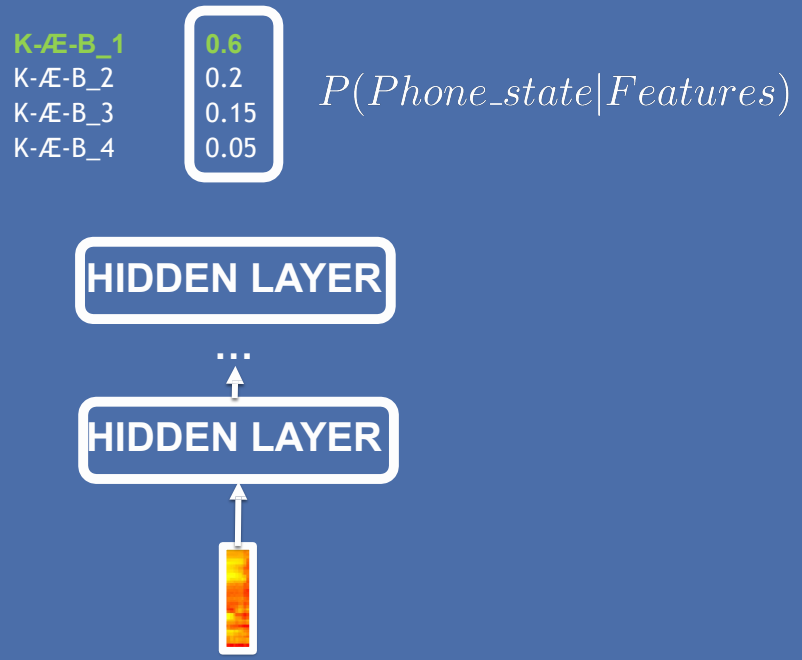
- **Extract speech features**
- **Train a Hidden Markov Model with Gaussian Mixture Model for emissions**
- **Use the Viterbi algorithm to extract an alignment of feature frames with states**
- Train a Neural Network to predict each state from its frame (or from the raw audio)

Could we get rid of these?



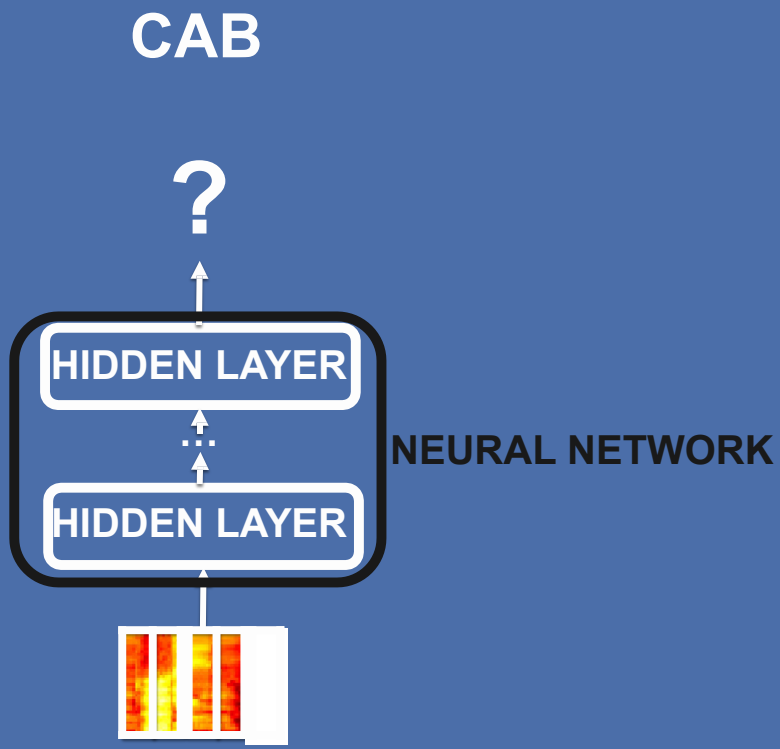
Training a DNN without alignment?

- When a state/phoneme is attributed to each frame it is trivial to define the loss function: classification loss



Training a DNN without alignment?

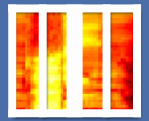
- When a state/phoneme is attributed to each frame it is trivial to define the loss function
- Without alignment, we have the word transcription and the feature frames
- Train a speech recognition system directly from the features to the phoneme sequence (no need for alignment)



Problem: no alignment available

- Phonemes last dozens to hundreds of miliseconds, while frames are sampled every 10ms
- Many more frames than phonemes
- We need to learn classification and alignment jointly
- A loss function allows learning both at the same time: Connectionist Temporal Classification (CTC)

CAB



Connectionist Temporal Classification

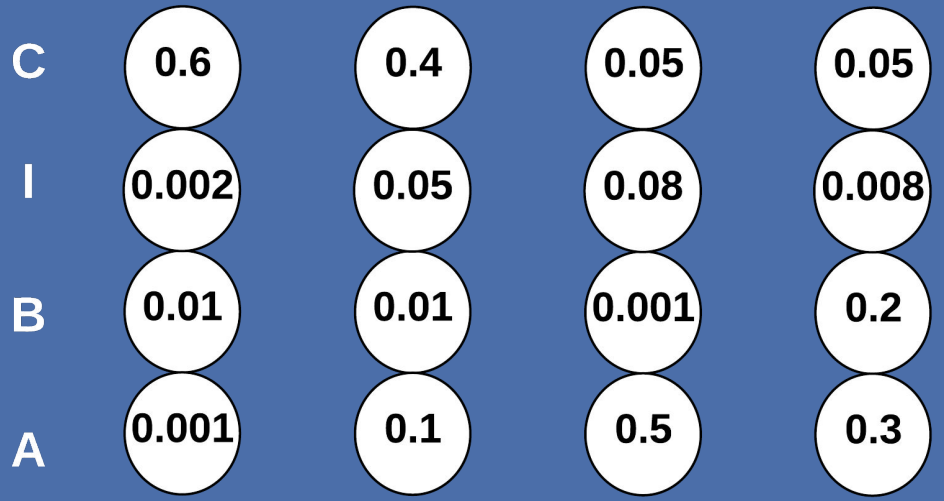
- Given that the final layer of your network is a softmax with as many dimensions as there are phonemes, it can be seen as a probability distribution over phonemes

y_k^t the probability of character k at time t

$$y_k^t = Softmax(h_{N-1}(x_t))$$

$$= \frac{e^{w_k^T h_{N-1}(x_t)+b}}{\sum_{k=1}^{K} e^{w_k^T h_{N-1}(x_t)}}$$

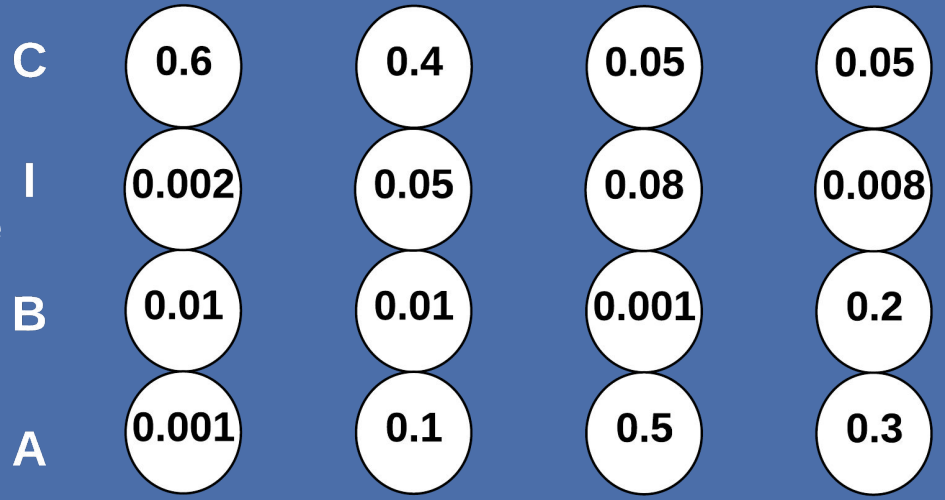
Y=phoneme-set U {0}



« Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks », Graves et al 2006.

Connectionist Temporal Classification

- Each output dimension is a node
- A path is a sequence of nodes
- Probability of a path is the product of the probability of the nodes (independence of time steps)

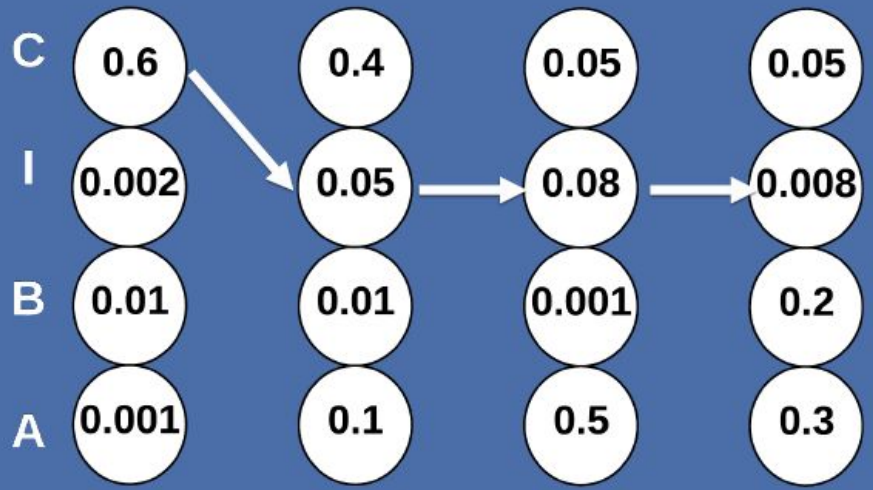


« Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks », Graves et al 2006.

Connectionist Temporal Classification

$\pi = \pi_1, .., \pi_T$ a path in the graph

$$p(\pi|x) = \prod_{t=1}^{T} y^t_{\pi_t}, \forall \pi \in L^T$$

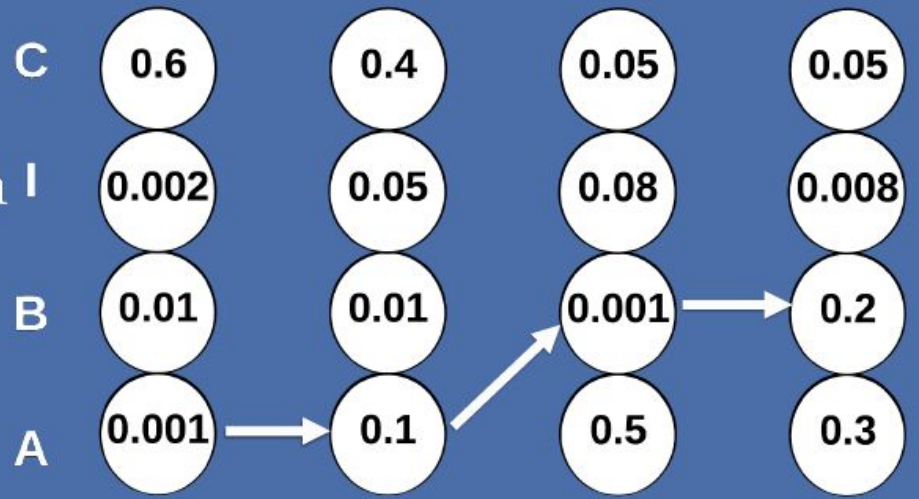


Probability(CIII) = 0.6*0.05*0.08*0.008 = 0.00000192

Connectionist Temporal Classification

$\pi = \pi_1, .., \pi_T$ a path in the graph

$$p(\pi|x) = \prod_{t=1}^{T} y^t_{\pi_t}, \forall \pi \in L^T$$



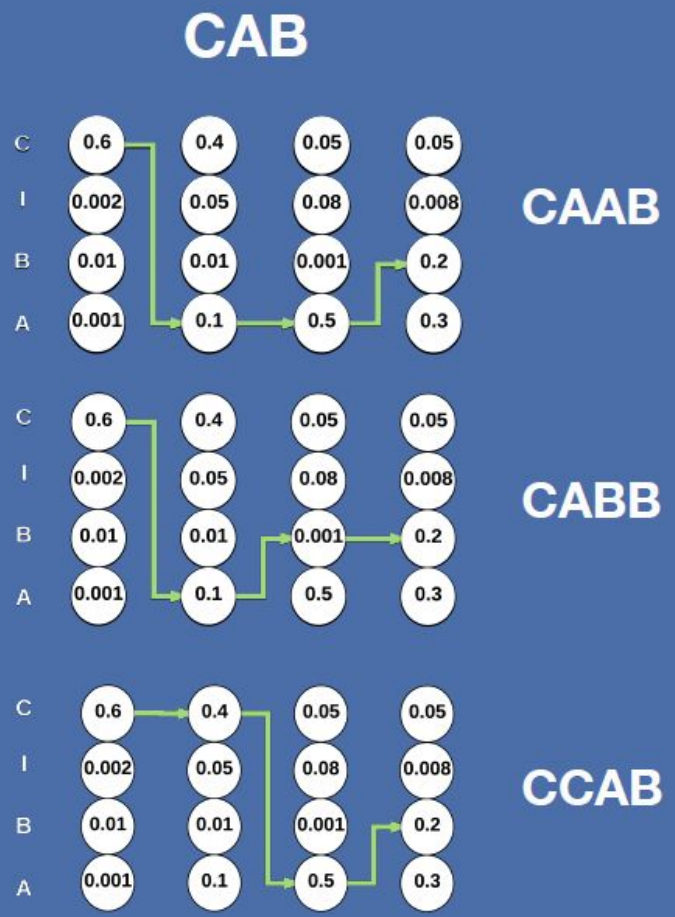
Probability(AABB) = 0.001*0.1*0.001*0.2 = 0.00000002

Training criterion

- Loss function:

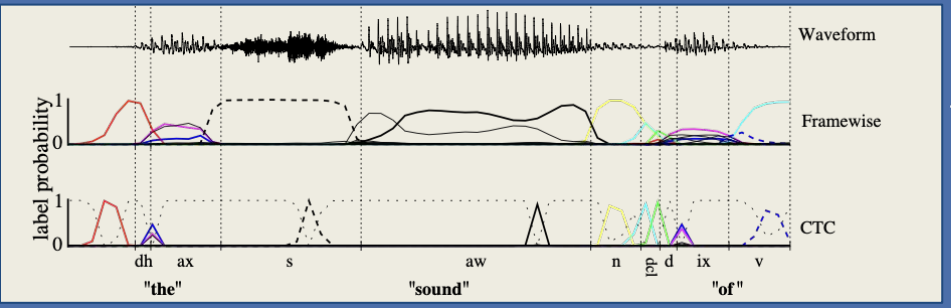
$$-\sum_{\pi \in \text{Valid paths}} P(\pi|x)$$

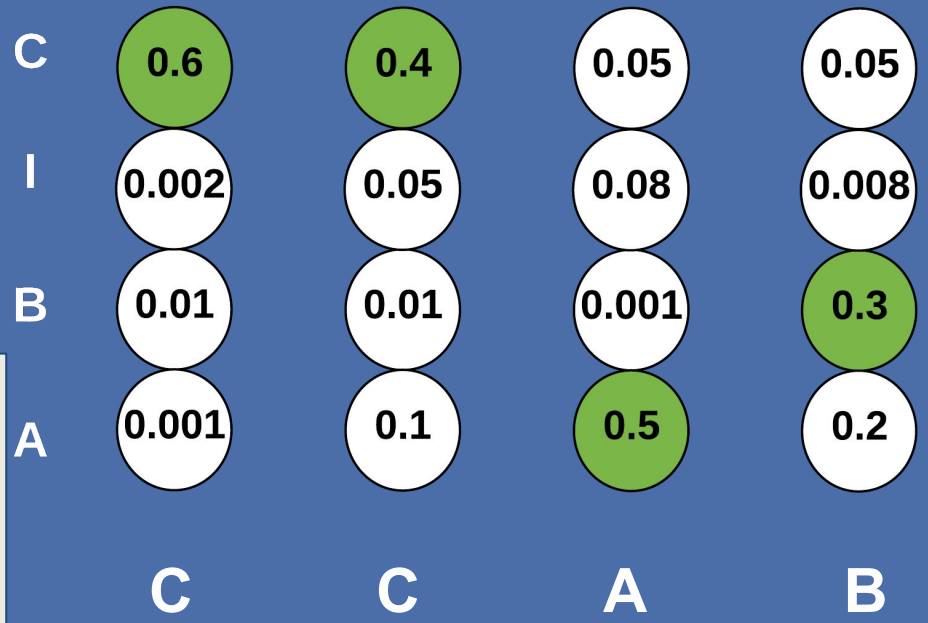
- Maximize the valid paths
- Nothing for the other paths
- Train the entire network with backpropagation



Decoding

- The basic decoding is just to take the most likely character at each step





CTC training

- List of phonemes
- Pronunciation dictionary
- List of phonological rules
- ~~train an HMM/GMM~~

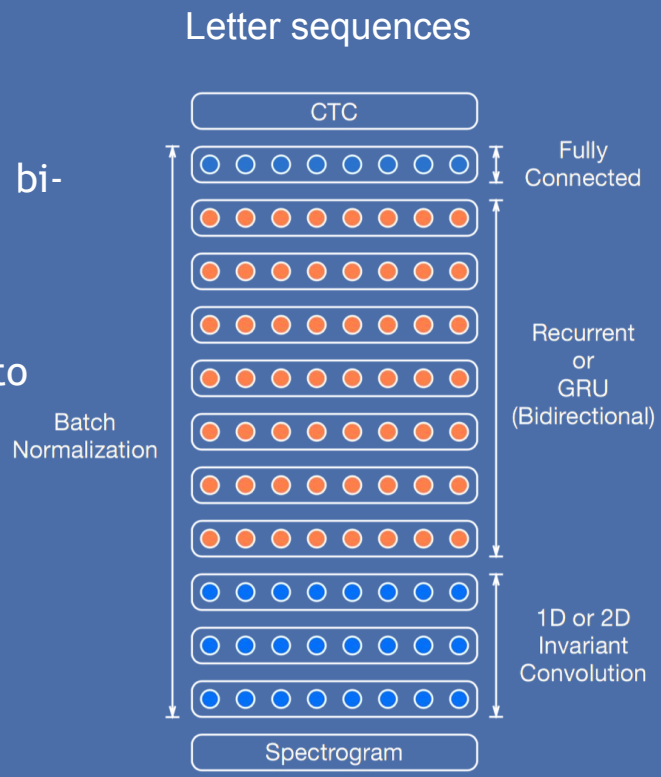
CTC training

- ~~List of phonemes~~
- ~~Pronunciation dictionary~~
- ~~List of phonological rules~~
- ~~train an HMM/GMM~~

Why not directly map speech to sequences of letters?

Deep Speech

- Baidu system
- Combines convolutions on spectrograms, bi-directional RNN and CTC
- Trained directly on letter sequences
- No speaker normalization (enough data to learn invariances!)

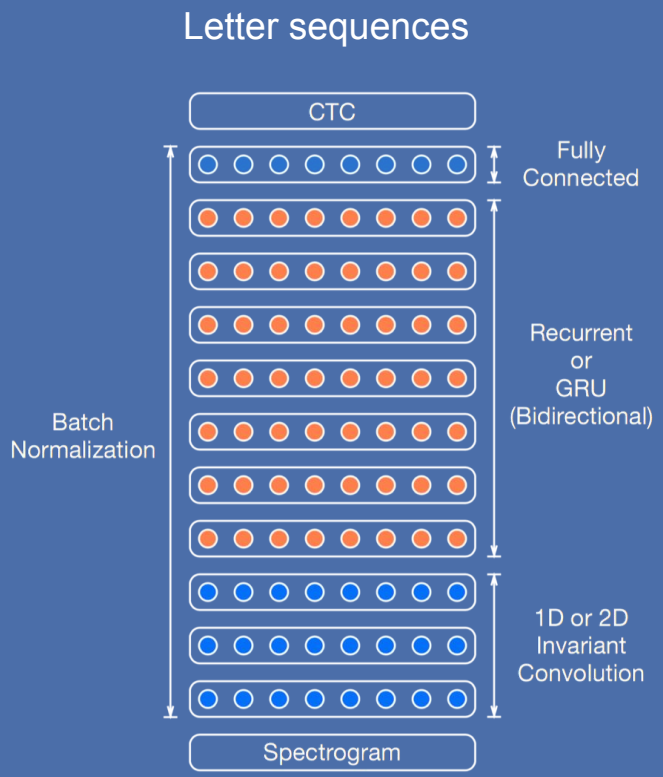


« Deep Speech: Scaling up end-to-end Speech Recognition. Hannun et al., 2014

« Deep Speech 2: End-to-End Speech Recognition in English and Mandarin », Almodei et al., 2015

Language models?

- Beam search decoding
- LM used to rescale the beam search



« Deep Speech: Scaling up end-to-end Speech Recognition. Hannun et al., 2014

« Deep Speech 2: End-to-End Speech Recognition in English and Mandarin », Almodei et al., 2015

Limits

A lot of linguistic expertise

- List of phonemes
- Pronunciation dictionary
- List of phonological rules

Some deep learning expertise

- Hyper parameter tuning

Reasonable amounts of annotations (50h+)

- Parallel corpus of speech+transcription

Large amounts of annotations (5000h++)

- Parallel corpus of speech+transcription

Assumptions and approximations

- Phone states= stationary emission probability
- Local context assumption (Markov property)
- Linguistic knowledge on phonemes, rules, etc

Assumptions and approximations

- Enough data will solve everything
- (unknown) inductive biases

References

- Language models:
  - Jurasky & Martin (2017). Chapter 4. Language modeling with N-grams
  -
- Dynamic programming:
  - Jaehyun Park (2015). Course materials for CS 97SI, Stanford University
  -
- Viterbi decoding, Baum Welch
  - Jurafsky & Martin (2017). Chapter 9. Hidden Markov Models
  -
- to know more
  - Gales & Young (2007). The application of Hidden Markow Models in Speech Recognition. *Foundations and Trends in Signal Processing*, **1(3)**, 195-304.
  - Mohri, M., Pereira, F., and Riley, M. (2002). Weighted finite-state transducers in speech recognition. Computer Speech & Language, 16(1):69–88.
- Tools
  - openFST (http://www.openfst.org)
  - Kaldi: http://kaldi-asr.org/
  - Wav2letter: https://github.com/facebookresearch/wav2letter

Question and quizz