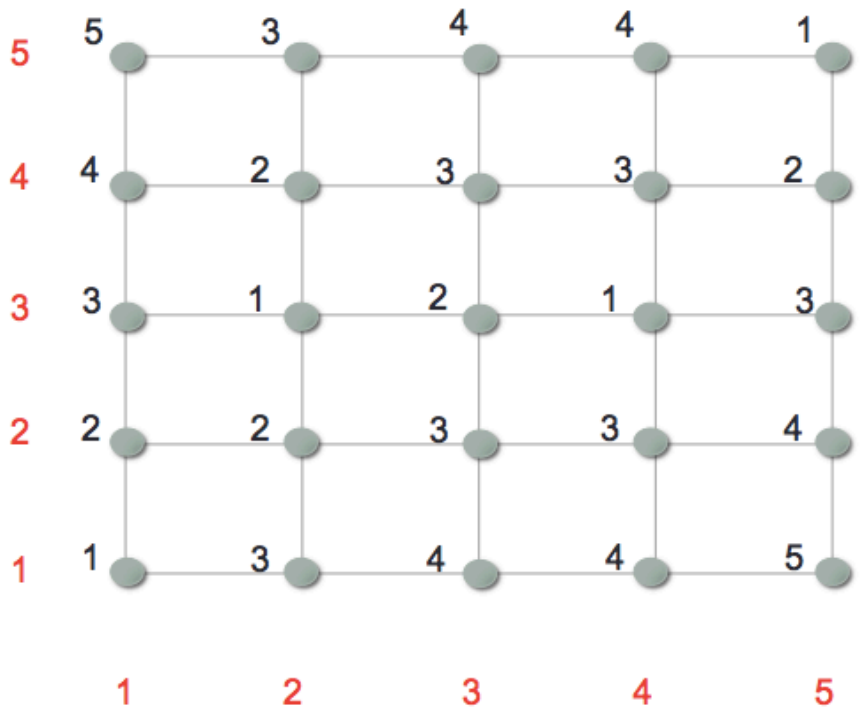
LEXICON & FST

Many illustrations provided by courtesy of James Glass

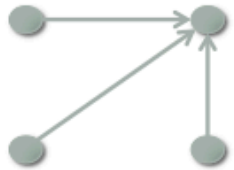
Quiz

Problem 1



Global constraint
$|x - y| <= 1$

Local constraint



Find the minimum distance and the warp.

Problem 2

Put the following steps from the MFCC derivation in the correct order.
DCT, Log, FFT, Windowing with Hamming window, Mel filterbank, Pre-emphisis filter (high pass filter), DC offset removal

Recap++

- LM (n-gram)
  - How to evaluate a sentence
  - How to estimate LM
  - How to compare – Perplexity
- Sparsity problem
  - Discounting
  - Back-off
  - Good Turing
  - KN discounting
  - Interpolation

Counts	I	eat	rice	</s>
I	10	5	7	8
eat	5	0	2	15
rice	3	6	0	20
<s>	1	0	30	60

T	H	E	R	E		I	S		N	O		R	E	V	E	R	S	E
1	1	1	5	1	1	2	1	1	2	1	1	15	1	17	1	1	1	2
	O	N		A		M	O	T	O	R	C	Y	C	L	E		A	…
1	3	2	1	2	2	7	1	1	1	1	4	1	1	1	1	1	3	…

$$P(w_i|w_{i-1}) = \begin{cases} f(w_i|w_{i-1}) & c(w_{i-1}w_i) \geq \alpha \\ f_d(w_i|w_{i-1}) & \alpha > c(w_{i-1}w_i) > 0 \\ q(w_{i-1})P(w_i) & c(w_{i-1}w_i) = 0 \end{cases}$$

San Francisco is the largest city. San Francisco is in California…

The rice ___ very good
P(grows| rice) vs P(Francisco | rice)
q(rice) P(grows) vs q(rice) P(Francisco)

Beyond n-grams

- Class n-grams
- N-gram adaptation

Class n-grams

- Many words has similar behavior
  - Days of the week, numbers, cities
- N-gram performance can be improved by clustering words and replace the words with the cluster
- Cluster can be created manually or automatically
- Can easily add new words to the cluster later

Class n-grams example

fly to Thailand
fly from Australia
fly to Japan
Australia is hot

fly to COUNTRY
fly from COUNTRY
fly to COUNTRY
COUNTRY is hot

- P(fly to Australia) = ?

COUNTRY = { Thailand, Australia, Japan}
P(Australia | Fly to) = P(COUNTRY | Fly to)P(Australia | COUNTRY)
= 2/2 * 2/4

Automatic word clustering

- Cluster similar words into the same category
  - How to measure similarity?
- Key observation : you can guess the meaning of the word from the words around it (context)

วัน ____ เรียน เสร็จ ต้อง ดู ~~เดอะ แมส ซิงเก้อ~~ บุพเพสันนิวาส

- How to measure context?
  - Mutual Information

Point-wise mutual information (PMI)

- PMI measures how the occurrence of x gives information about the occurrence of y

$$pmi(x = a; y = b) = log \frac{p(x = a; y = b)}{p(x = a)p(y = b)}$$

- PMI is 0 if x and y are independent.
  - Can be positive or negative (positive/negative correlation)

Top PMI values from 50 million words wikipedia corpus

word 1	*word 2*	*count word 1*	*count word 2*	count of co-occurrences	PMI
puerto	rico	1938	1311	1159	10.0349081703
hong	kong	2438	2694	2205	9.72831972408
los	angeles	3501	2808	2791	9.56067615065
carbon	dioxide	4265	1353	1032	9.09852946116

https://en.wikipedia.org/wiki/Pointwise_mutual_information

Clustering based on MI

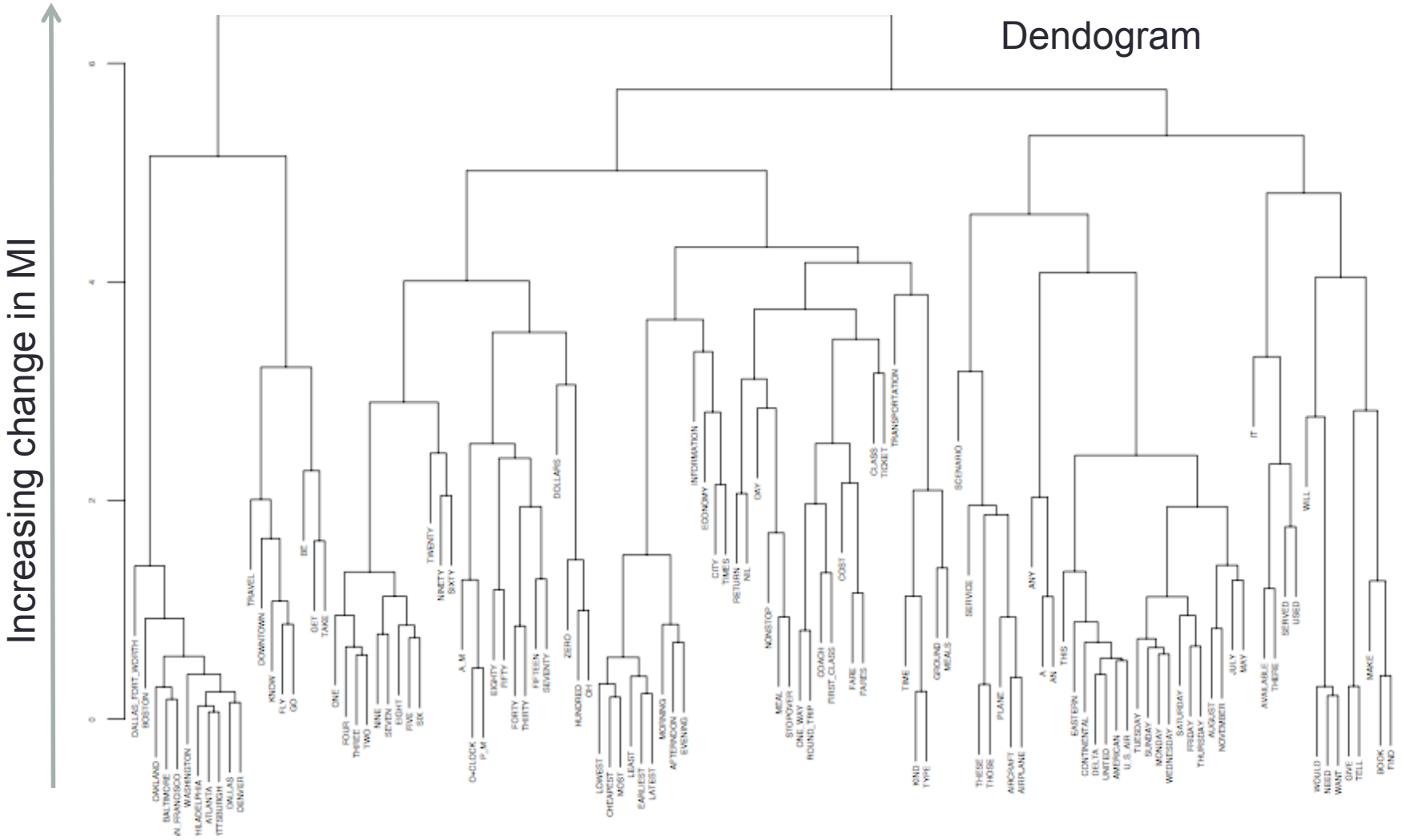
- Mutual Information (MI) is the average of PMI over all outcomes/words.

Expected value E[x] = Sum of p(x)x

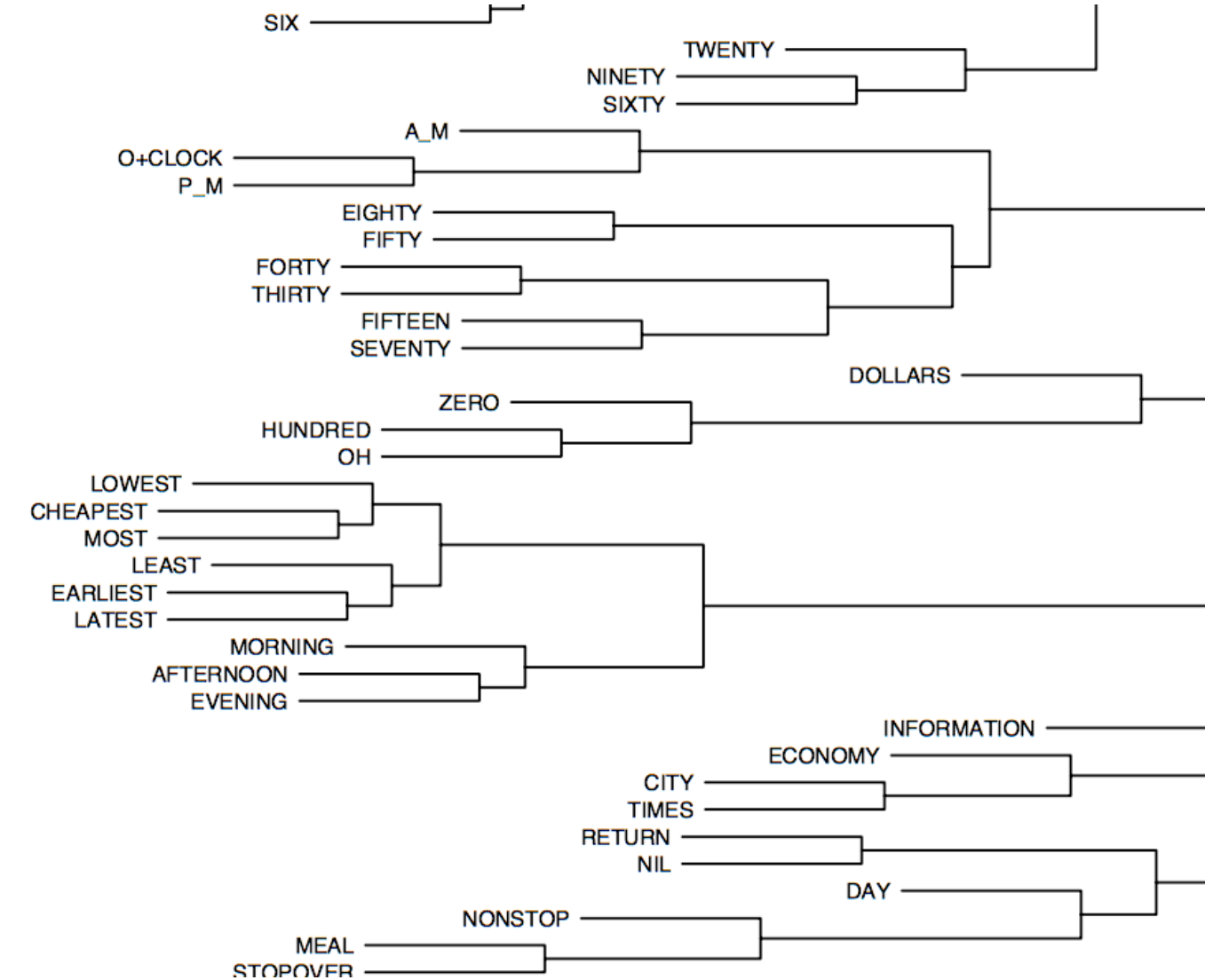
$$I = \Sigma_{x,y} p(x,y) pmi(x,y)$$

- Clustering w_1 and w_2 into a cluster c_1 is good if the change in MI is the small.
- Can do hierarchical clustering (bottom-up)

Clustering example



Increasing change in MI



SRILM clustering example

Class	Words
34	apparently, do, could, . . .
37	concord, cleveland, sacramento, philadelphia, baltimore, pittsburgh, cincinnati, rochester,. . .
39	nice, friend, toll, digital, takeoff, landed, gets, helen, arrived, right, left, analog, requests
63	will
66	arriving
78	orange, des, little, salt, baton, westchester, las
81	last, first, second, third
82	thank
89	taken, takes, take, taking, someone
168	feel, want, wanted, need, meant
170	between, from
223	around, approximately, after

N-gram adaptation

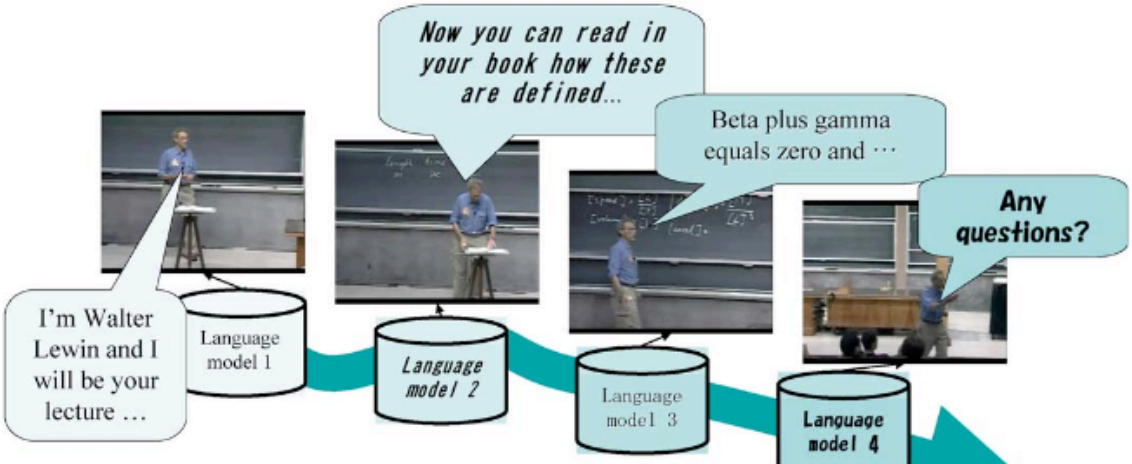
- We can change the n-gram according to the current flow of the conversation/document.
- Usually interpolate with a static n-gram
- Cache-based n-gram
  - Create an n-gram based on the previous X words.

$$P(w_i|h_i) = \lambda P_c(w_i|h_i) + (1-\lambda)P_s(w_i|h_i)$$

cache n-gram Global n-gram

Topic-based n-gram

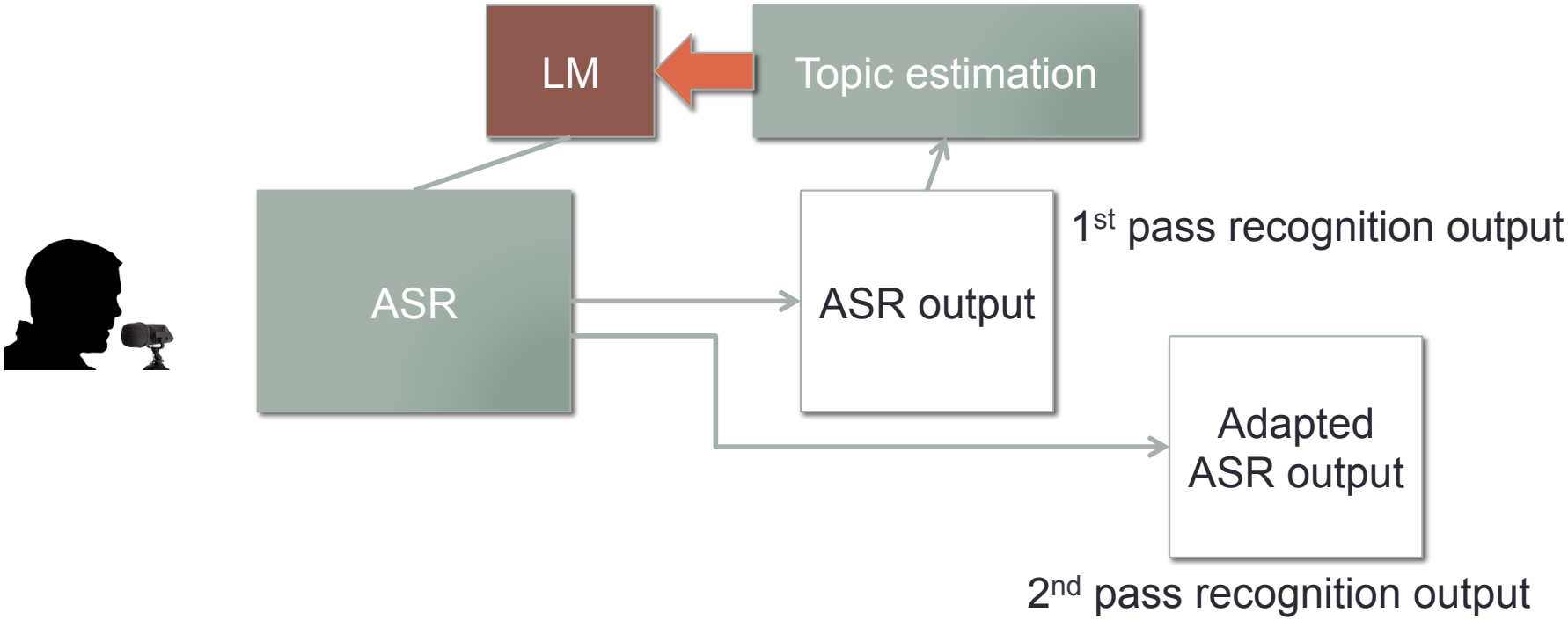
- Build multiple n-gram based on various topics – textbooks from various subjects
- Change interpolation weight according to the predicted topic
- Can also learn topics automatically from data



*Walter Lewin, 8.01 Physics I: Classical Mechanics, Fall 1999. (Massachusetts Institute of Technology: MIT OpenCourseWare), http://ocw.mit.edu (Accessed Oct., 2007). License: Creative Commons BY-NC-SA.

Topic-based n-gram example

- Corpus of 1 hour lectures
- Estimate topic from ASR output -> Adapt LM
  - Adapt for each sentence
  - Adapt for whole lecture



Topic-based n-gram example

- Adaptation helps – but not much. Larger gain on ppl than WER
- Improvement in ppl does not always improve WER
  - Always evaluate your final metric!

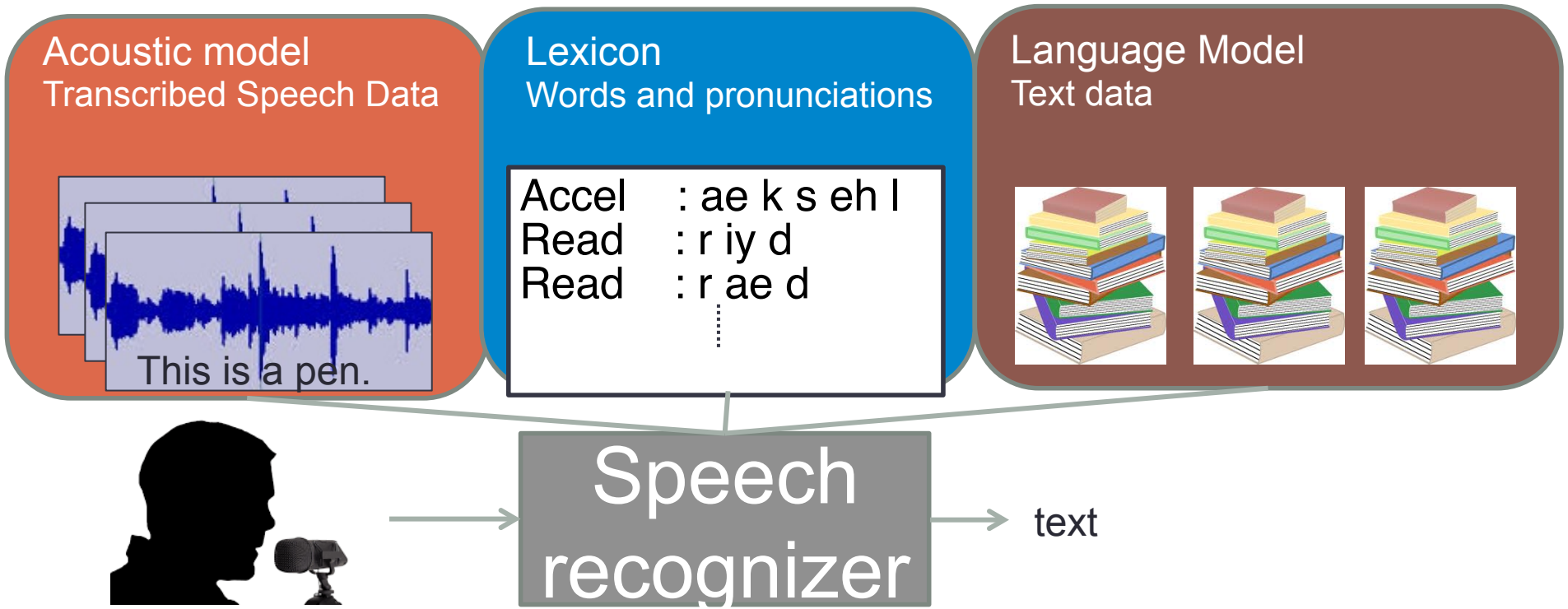
	Unigram ppl	Trigram ppl	WER
1st pass results	611	208	41.4
Whole lecture method1	543	188	41.3
Whole lecture method2	549	202	41.0
sentence level method1	521	185	41.0
sentence level method2	504	179	40.6
Adapt on transcription	482	171	39.4

E. Chuangsuwanich, S. Watanabe, T, Hori, T. Iwata, and J. Glass, "Handling Uncertain Observations in Unsupervised Topic-Mixture Language Model Adaptation," ICASSP 2012

The ASR equation

$$= argmax_{W,L} P(X \mid L) P(L \mid W) P(W)$$

X - waveform, L - pronunciation, W - words



The Lexicon

- Mapping between words and pronunciations
- Traditionally generated by linguists – still is 99% generated by linguists
- Time consuming
  - Many words can be pronounced differently depending on dialect or meaning, Nice (place) vs nice (adj)
- Requires expert knowledge
  - Cannot recruit a random person to do like doing transcription

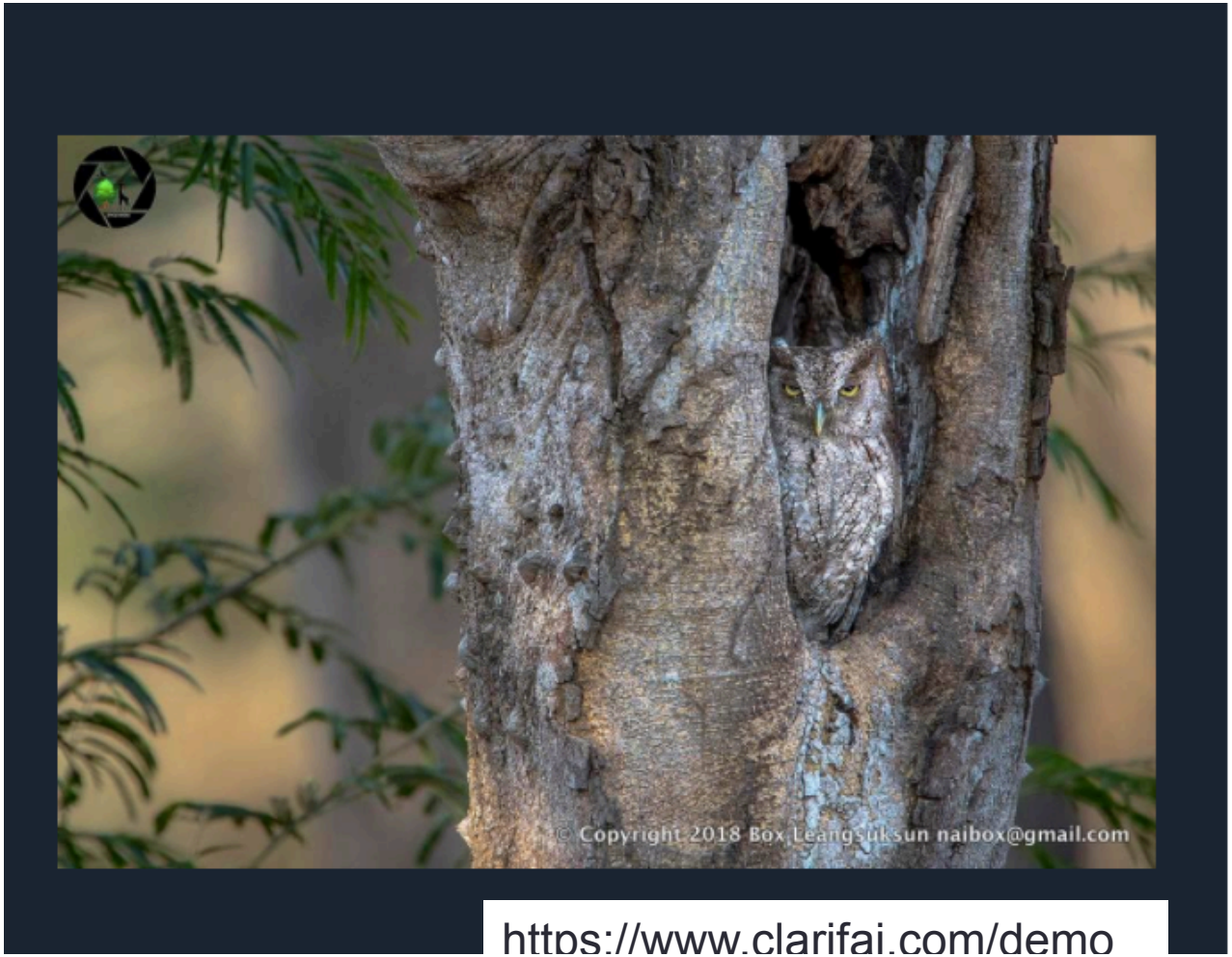
"Every time I fire a linguist, the performance of the speech recognizer goes up"

Frederick Jelinek - speech recognition pioneer

Machines and humans think differently.

Can a neural network recognize the owl?

Yes, easily



https://www.clarifai.com/demo

LANGUAGE

English (en)

PREDICTED CONCEPT	PROBABILITY
tree	0.997
no person	0.990
nature	0.988
wildlife	0.971
outdoors	0.961
wood	0.956
bird	0.942
wild	0.927
leaf	0.893

Yes, easily

https://www.clarifai.com/demo

wood	0.956
bird	0.942
wild	0.927
leaf	0.893
animal	0.891
tropical	0.875
closeup	0.863
branch	0.855
bark	0.815
environment	0.815
desktop	0.809
owl	0.806

But it fails at this

https://iotsecurity.eecs.umich.edu/#roadsigns

Adversarial attacks

The lexicon, with less linguists

- Grapheme lexicon
- G2P model

The grapheme lexicon

- Represent letters (graphemes) as sound units
- The pronunciation is just the sequence of letter spelling
- Does not work for character-based languages (Chinese, Japanese)
- Works because we model sequence of sounds (triphones)

กรรไกร : k a n^ kr ai z^

Phoneme lexicon

กรรไกร : ก ร ร ไ ก ร

Grapheme lexicon

Grapheme vs Phoneme

- Not the end of the world if you do not have a lexicon
- Can be slightly improved with some knowledge about the language (rule-based)
- The more regular the spelling is the closer the gap

Language (WER)	Phoneme lexicon	Grapheme lexicon	diff
Kazakh	76.8%	77.0%	+0.2
Kurmanji	85.5%	85.1%	-0.4
Telugu	86.3%	87.0%	+0.7
Cebuano	75.7%	75.9%	+0.2
Lao	67.3%	69.9%	+2.6
Haitian	52.0%	52.3%	+0.3
Assamese	58.6%	58.5%	+0.1
English	8.0%	8.5%	

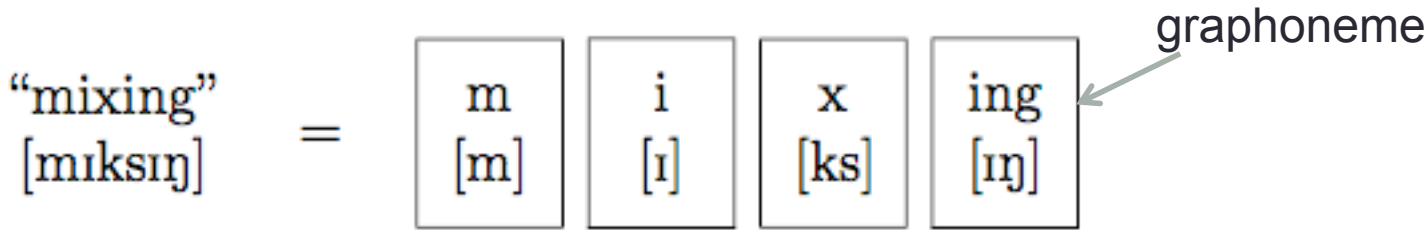
D. Harwath and J. Glass. Speech recognition without a lexicon-bridging the gap between graphemic and phonetic systems. In Proc. InterSpeech, 2014.
V. Le, L. Lamel, A. Messaoudi, W. Hartmann, J. Gauvain, C. Woehrling, J. Despres, and A. Roy. Developing STT and KWS systems using limited language resources. In Proc. InterSpeech, 2014.
E. Chuangsuwanich, *Multilingual Techniques for Low Resource Automatic Speech Recognition*, MIT, June 2016.

Automatic Grapheme-to-Phoneme (G2P)

- Given a small lexicon generated by linguists, learn a model to predict the pronunciation of new words
- Sometimes called L2S model (Letter-to-Sound)
- Can use acoustic data to improve performance of generated pronunciation

Joint Sequence Models (graphonemes)

- Learn mapping between sequence of graphemes to sequence of phonemes. (many-to-many)



- $$P(b|w) = \max_{G} \prod_{G_i} P(g_i)$$

- where w is the grapheme representation of the word, b is the pronunciation, G is the set of all possible graphoneme segmentation, g_i is a single graphoneme.

Bisani, Maximilian, and Hermann Ney. "Joint-sequence models for grapheme-to-phoneme conversion." *Speech communication* 50.5 (2008): 434-451.

We meet again

- I have a sequence of graphemes
- I have a sequence of phonemes

- We want to align them for the best alignment

w	=	c	o	u	p		l	e
b	=	kcl_k	ah		pcl_p	ax	l	
	=	kcl_k		ah	pcl_p	ax	l	
g_1	=	c/kcl_k	o/ah	u/ϵ	p/pcl_p	ϵ/ax	l/l	e/ϵ
g_2	=	c/kcl_k	o/ϵ	u/ah	p/pcl_p	ϵ/ax	l/l	e/ϵ

and again

- Trained by EM and Viterbi/Dynamic programing
  - Align with Viterbi
  - Update P(graphonemes)
  - Repeat

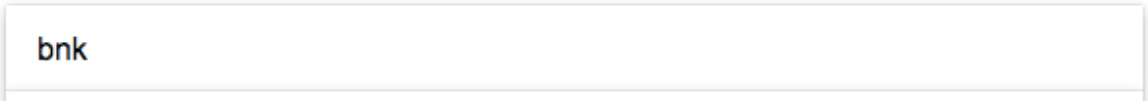
G2P example

- Trained with 5k pronunciations using Sequitur (already in your SRILM+kaldi docker image)
- Can produce multiple candidate pronunciations

```
ฟรุ้งฟริ้ง    0       0.394294        fr u ng^ x p^ fr i ng^
ฟรุ้งฟริ้ง    1       0.082894        fr u ng^ x p^ fr i xx ng^
ฟรุ้งฟริ้ง    2       0.070312        fr u ng^ fr i ng^
มุ้งมิ้ง      0       0.641720        m u ng^ m i ng^
มุ้งมิ้ง      1       0.134912        m u ng^ m i xx ng^
มุ้งมิ้ง      2       0.056220        m u ng^ m i e ng^
สัตว์โลกอมตีน       0       0.192856        s a t^ l oo k^ m^ t ii n^
สัตว์โลกอมตีน       1       0.164017        s a t^ l oo k @@ m^ t ii n^
สัตว์โลกอมตีน       2       0.116318        s a t^ l oo k^ @ m^ t ii n^
รู้หมือไร่    0       0.780650        r uu m vv r a j^
รู้หมือไร่    1       0.192681        r uu m vva r a j^
รู้หมือไร่    2       0.002743        r uu h m vv r a j^
```

Learning from real acoustic data

- Can identify the correct pronunciation(s) from acoustic data
- Possible usage example
  - A new trending word appears in text search
  - Generate multiple pronunciations of the new word using G2P
  - Add it to the lexicon. Update the LM using text search data
  - Acquire Voice Search examples from users
  - Viterbi align to identify the real pronunciation



Weighted lexicon from acoustic data

Linguists pronunciations weighted with acoustic data

G2P pronunciations weighted with acoustic data

Word	Weighted Expert Pronunciations		PMM Pronunciations	
bangalore	0.38	bcl b aa ng g el ao r	0.53	bcl b ae ng g el ao r
			0.28	bcl b aa ng gcl g el ao r
general	0.40	dcl jh eh n axr el	0.54	dcl jh eh n axr el
general	0.30	dcl jh eh n r el	0.19	dcl jh eh n axr l
general	0.25	dcl jh eh n axr l	0.17	dcl jh eh n r el
istanbul	1.0	ih s tcl t aa n bcl b uw l	0.41	ih s tcl t ae n bcl b el
			0.20	ih s tcl t aa n bcl b el
			0.14	ih s tcl t ax n bcl b el
ottawa	0.95	aa dx ax w ax	0.51	aa dx ax w aa
	0.04	aa tcl t ax w ax	0.42	aa dx ax w ax
nice	0.75	n ay s	0.75	n ay s
	0.24	n iy s	0.24	n iy s

McGraw, I. Badr, and J. Glass, "Learning Lexicons From Speech Using a Pronunciation Mixture Model," *IEEE Transactions on Audio, Speech, and Language Processing*, February 2013, Volume 21, Issue 2, pp. 357-366.

WER Performance

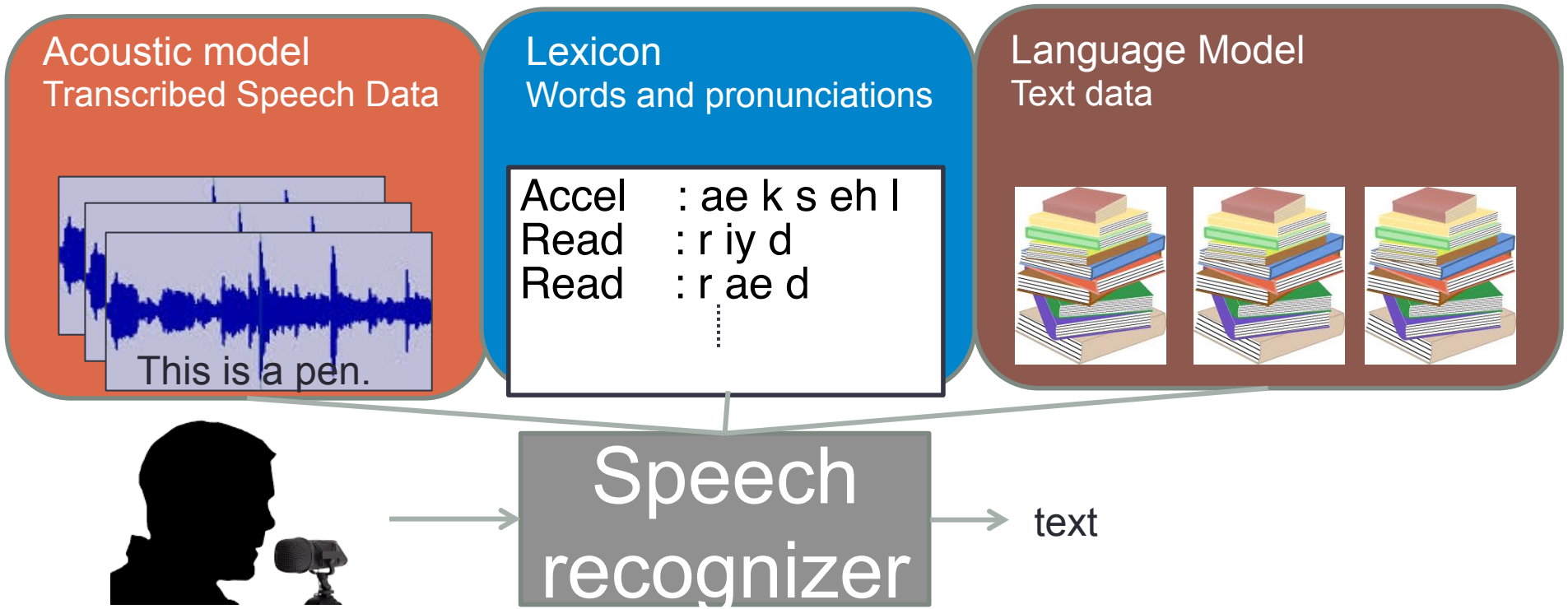
Lexicon	WER
G2P	10.1
Linguist	9.5
Weighted linguist	9.2
Weighted G2P	8.2

McGraw, I. Badr, and J. Glass, "Learning Lexicons From Speech Using a Pronunciation Mixture Model," *IEEE Transactions on Audio, Speech, and Language Processing*, February 2013, Volume 21, Issue 2, pp. 357-366.

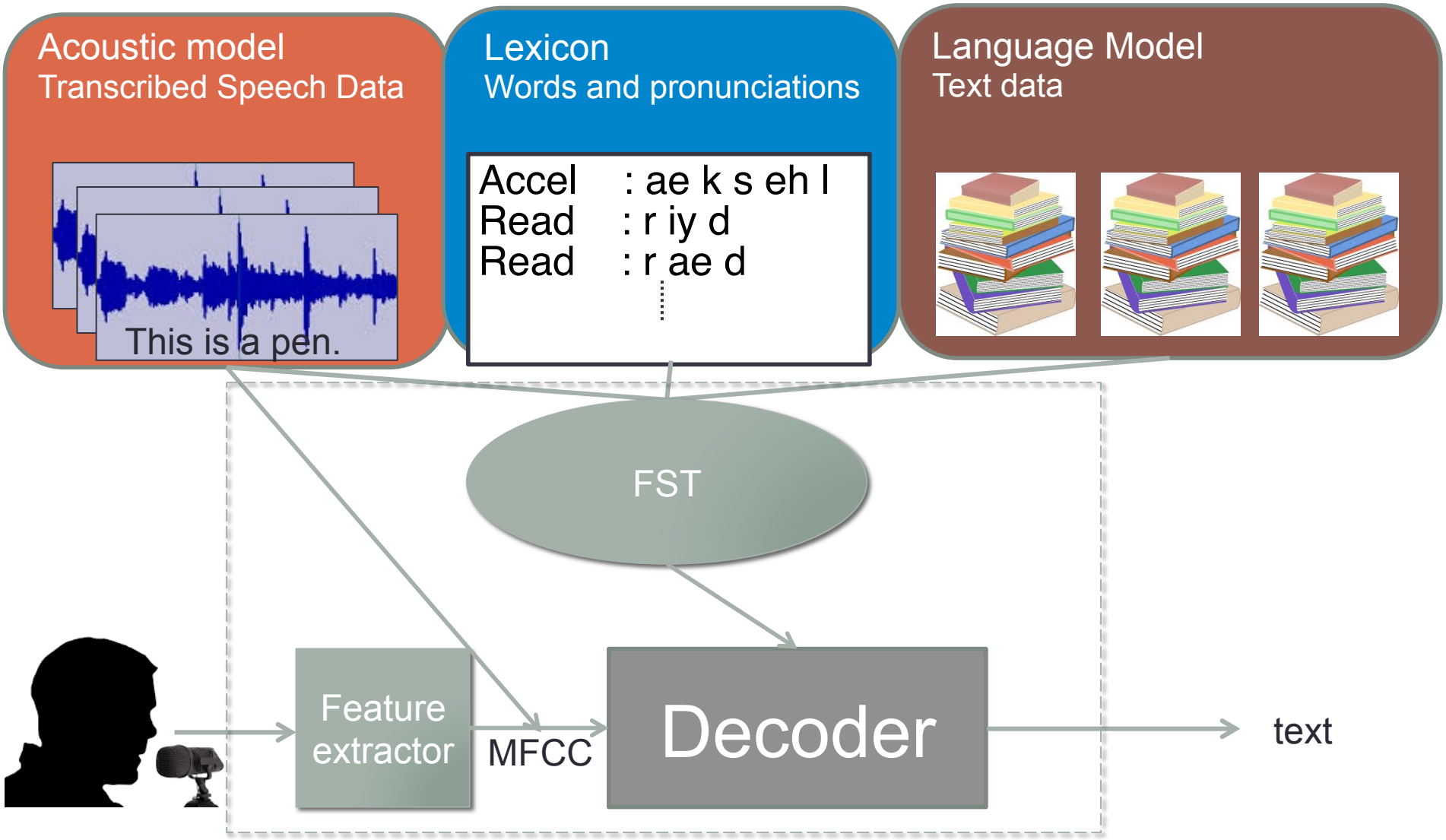
The ASR equation

$$= argmax_{W,L} P(X \mid L) P(L \mid W) P(W)$$

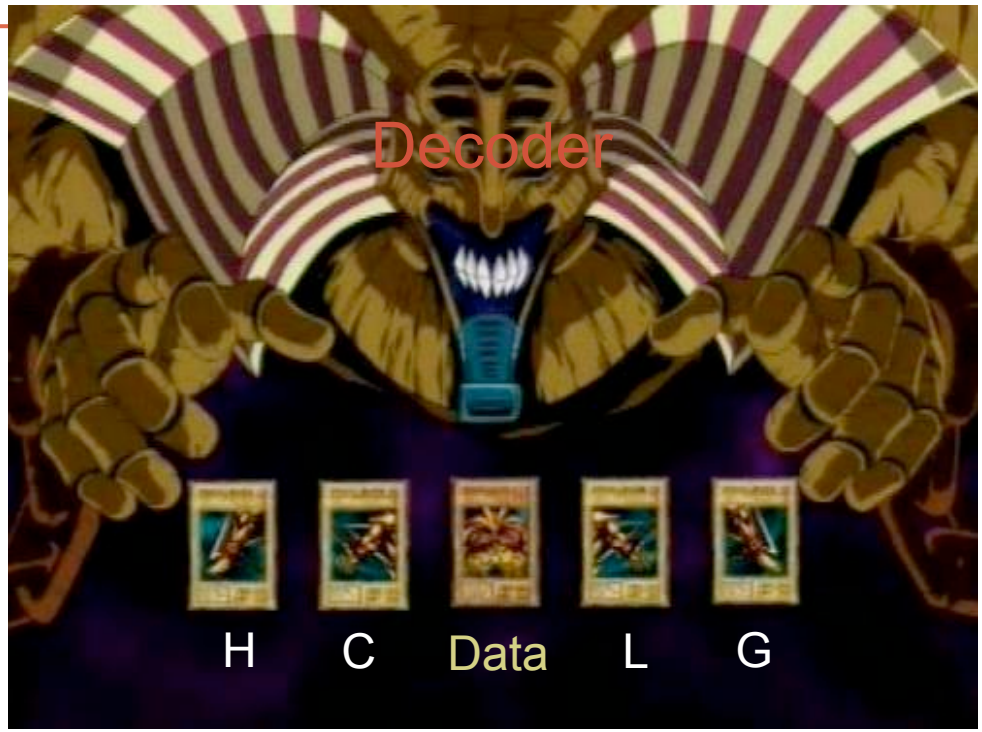
X - waveform, L - pronunciation, W - words



Inside the recognizer

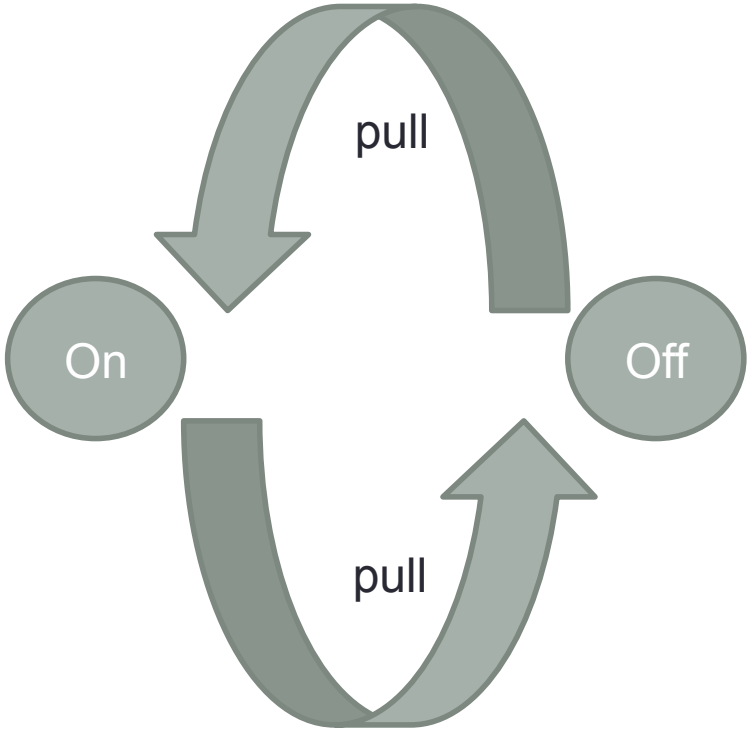


FINITE STATE TRANSDUCERS AND DECODING



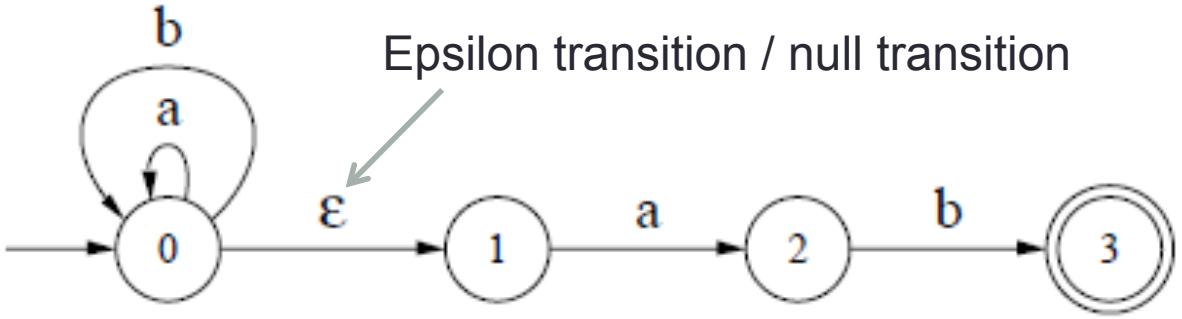
Finite State Automata (FSA)

- A model of computation
- The machine is in a certain state
- States change according to the input



Defining a state machine

- Set of states (nodes)
- Set of possible inputs (alphabet)
- Set of transition between states (arcs)
- Starting state
- Set of end states (final states)

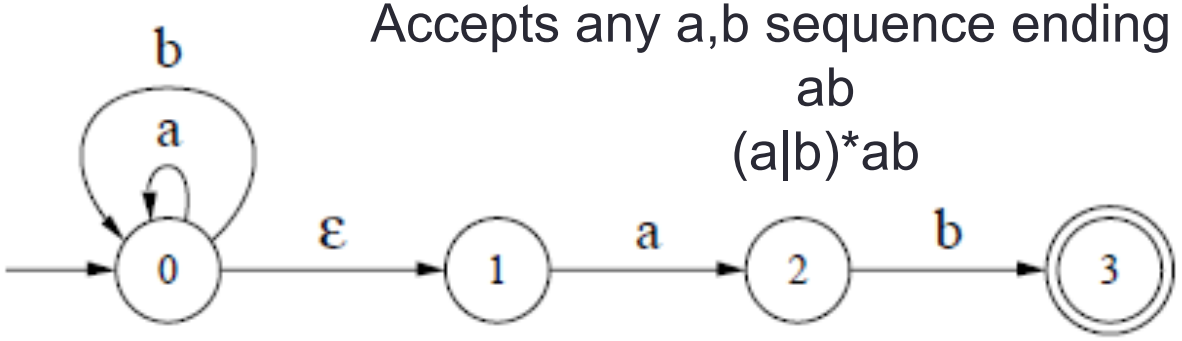


Defining a state machine

- A state machine accepts the input if it ends at an end state
- Does the model accept:
  - a b a a
  - a a b
  - a
- Sometimes called Finite State Acceptor (FSA)

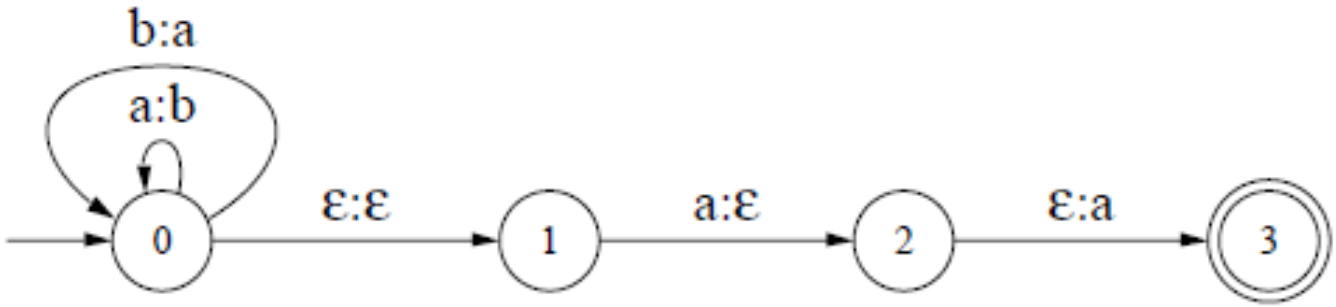
Accepts any a,b sequence ending with ab

(a|b)*ab



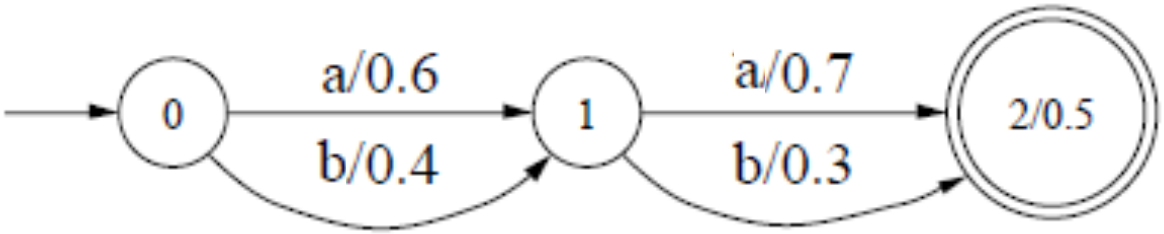
Finite State Transducer (FST)

- Similar to FSA, but the arcs now have input-output pairs
- What is the output of
  - a a b a
  - b



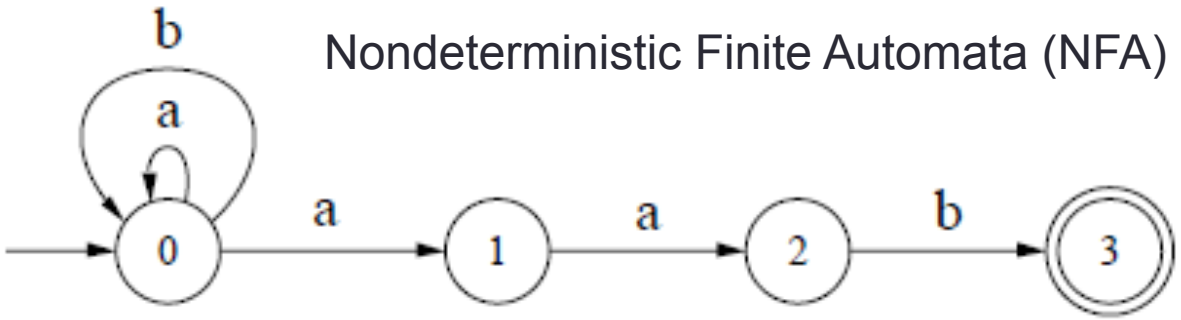
Weighted FST/FSA

- Each arc can be weighted with scores, probabilities, costs, etc.
- P(aa) = 0.6*0.7*0.5
- P(ab) = 0.6*0.3*0.5
- P(bb) = 0.4*0.3*0.5
- P(ba) = 0.4*0.7*0.5



Deterministic Finite Automata (DFA)

- A FSA/FST is deterministic if any input sequences uniquely determines the state sequence
  - No epsilon transition
  - At most one transition per input label for all states
- Algorithms exist to convert NFA to DFA (powerset construction)
- Deterministic property is desirable because it makes search more efficient



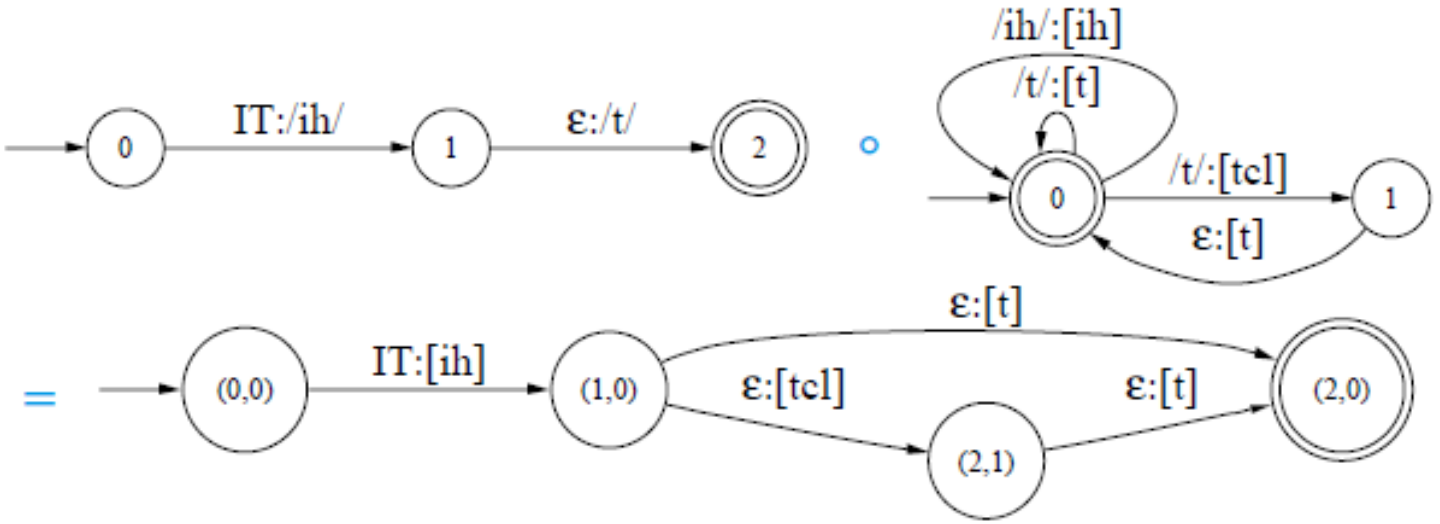
Nondeterministic Finite Automata (NFA)

Determinization of FST

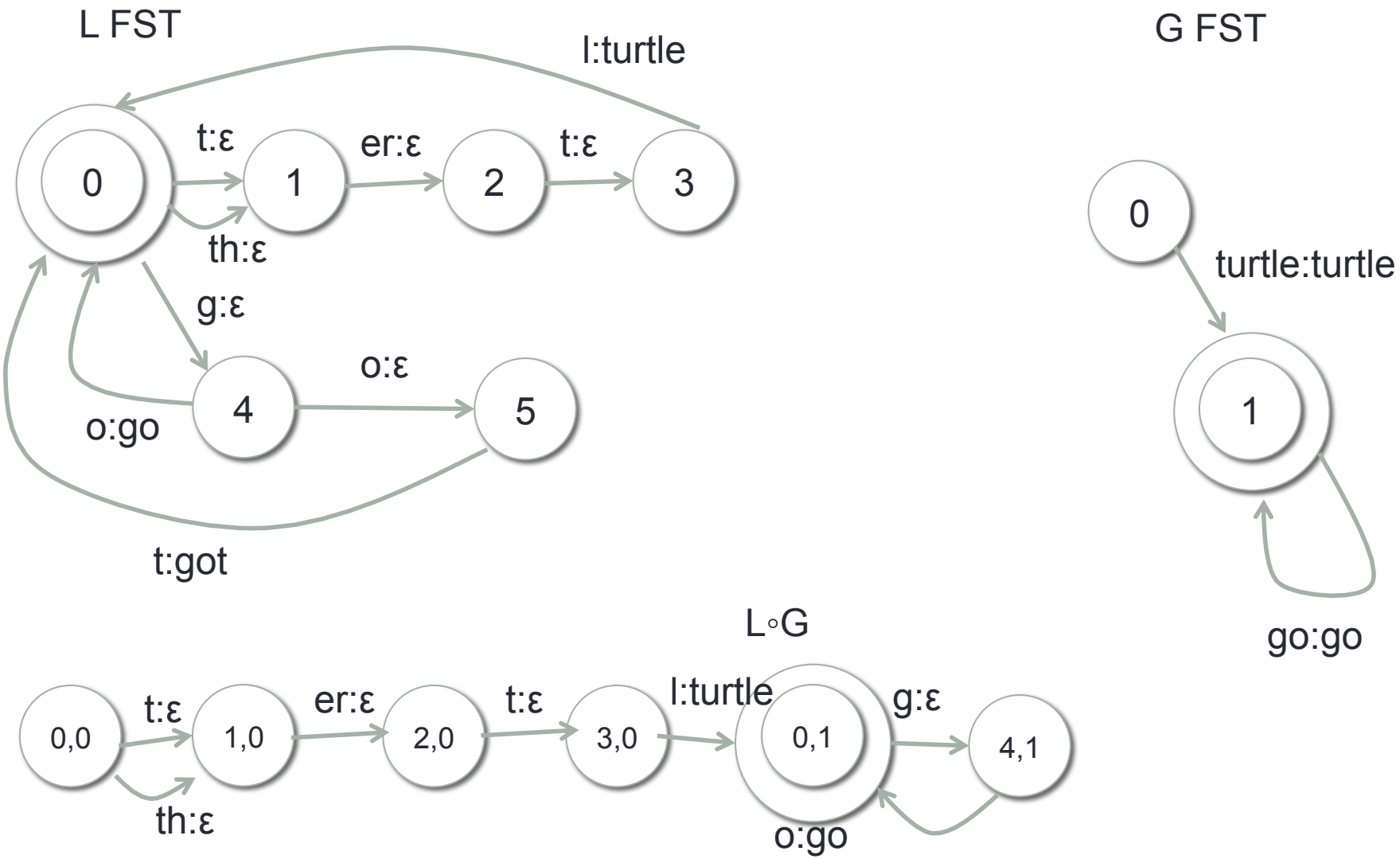
- The process of turning non-deterministic FST/FSM into a deterministic one
- Every FSM has a deterministic version
  - The deterministic FSM can have 2^N node in the worst case
- Not true for FSTs
  - There are workarounds

FST Composition A◦B

- A◦B feeds the outputs of A as inputs of B
  - Think of concatenating systems in a series
- Output states are associated with input state pairs
- Very powerful!

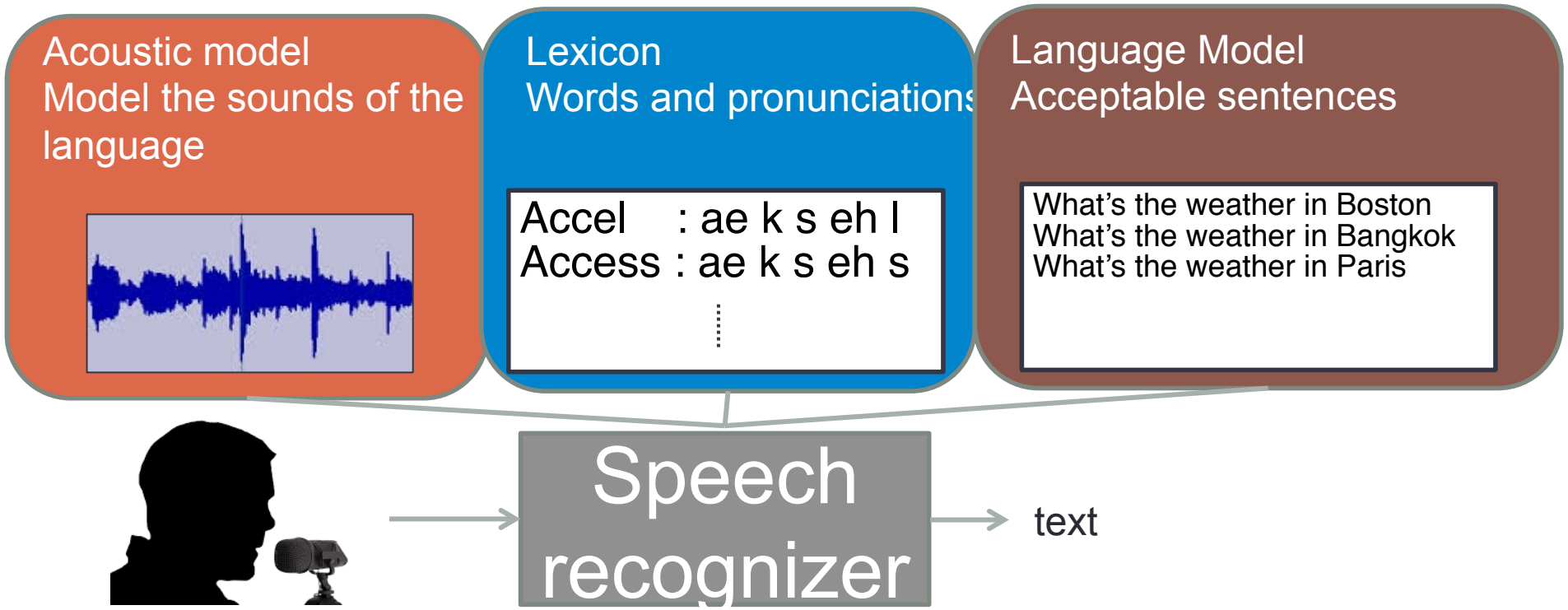


FST composition example



FST in ASR

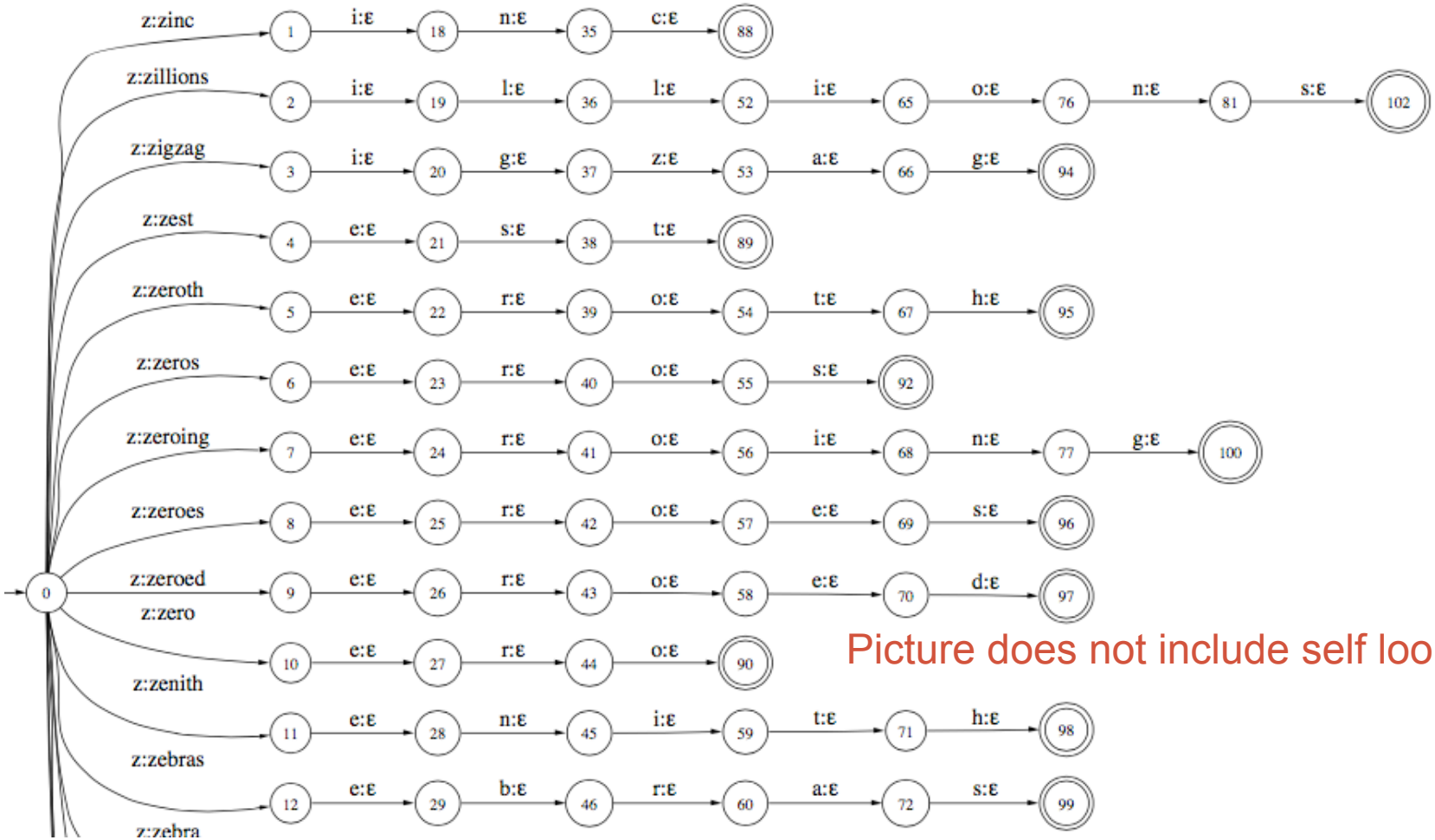
- Each component can be represented as a FST
- Final FST is the composition of all three
- Manipulating FSTs makes ASR flexible



Lexicon FST (L FST)

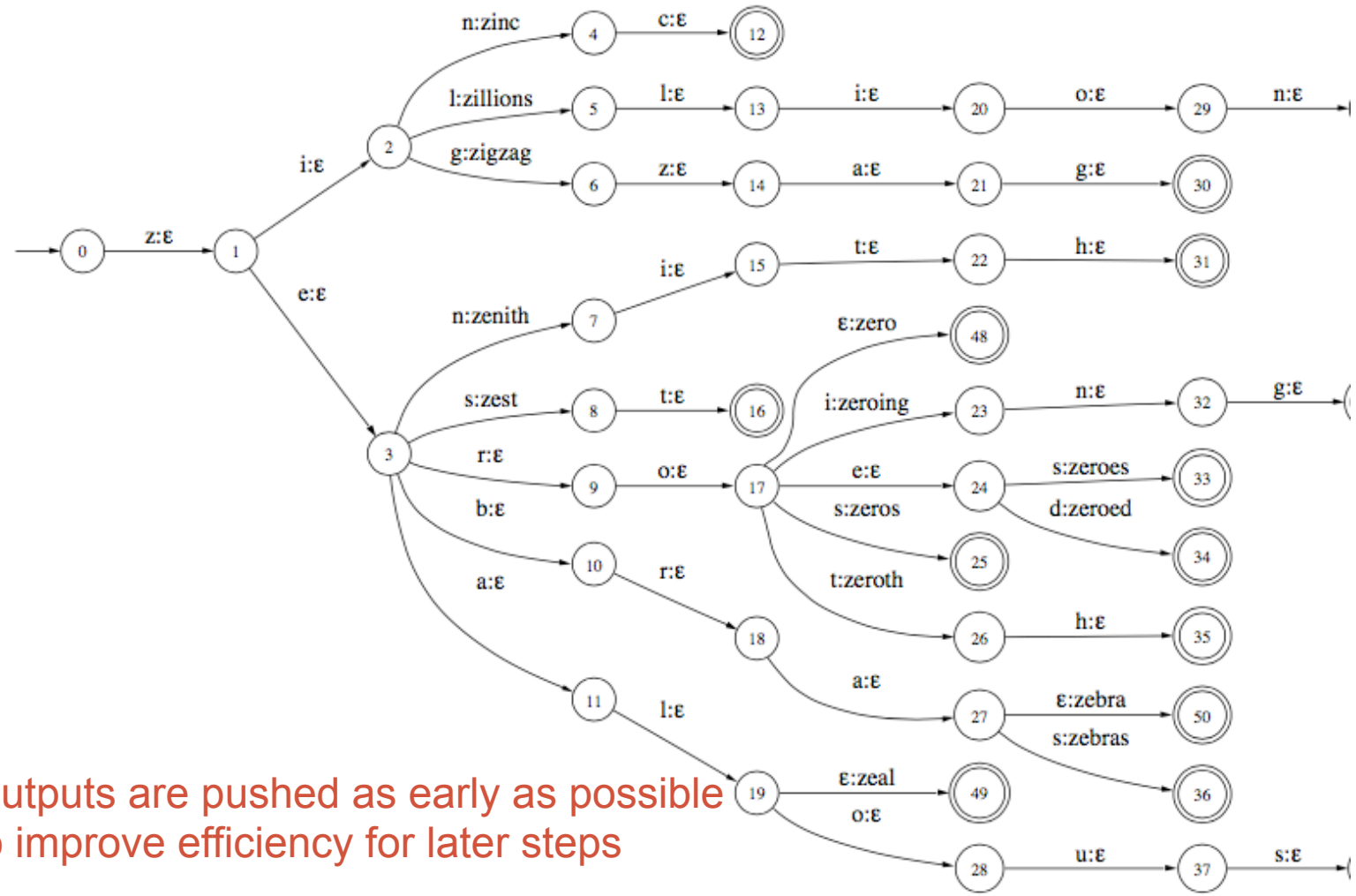
- Input phonemes, output Words
- Can weight the arc according to probability of the pronunciation
- A loop at the end (so that we can repeat words)
- Add a pronunciation for silence or noise (laugh, cough) to handle noise
- Determinize to improve efficiency

L FST example



Picture does not include self loop

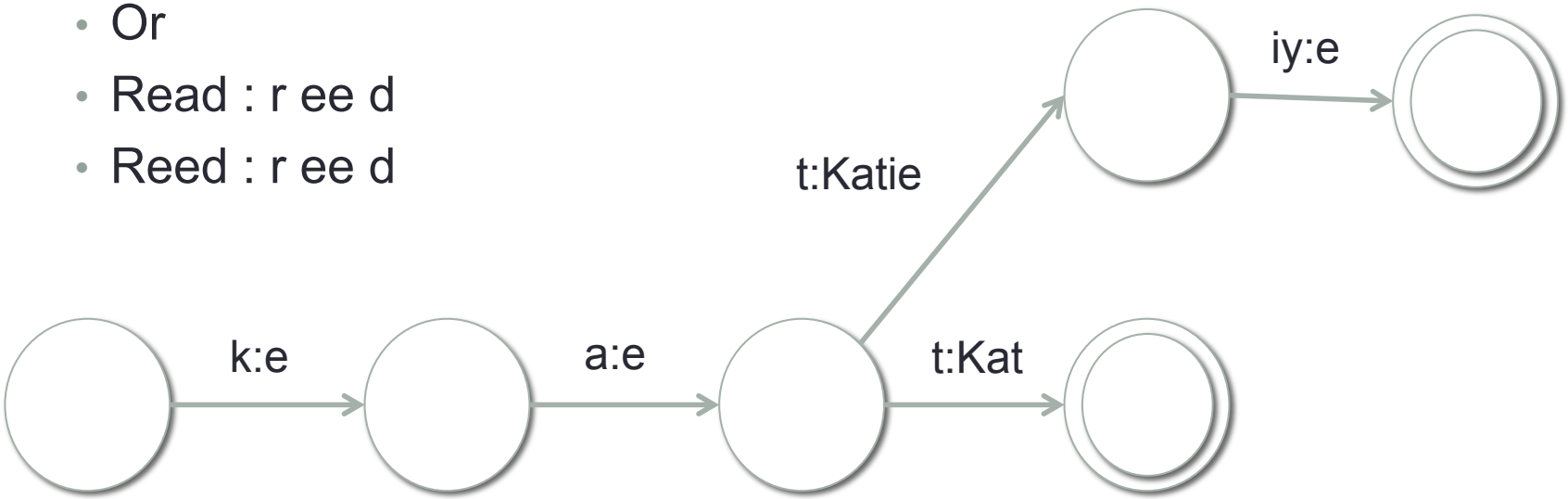
Determinized L FST



Outputs are pushed as early as possible to improve efficiency for later steps

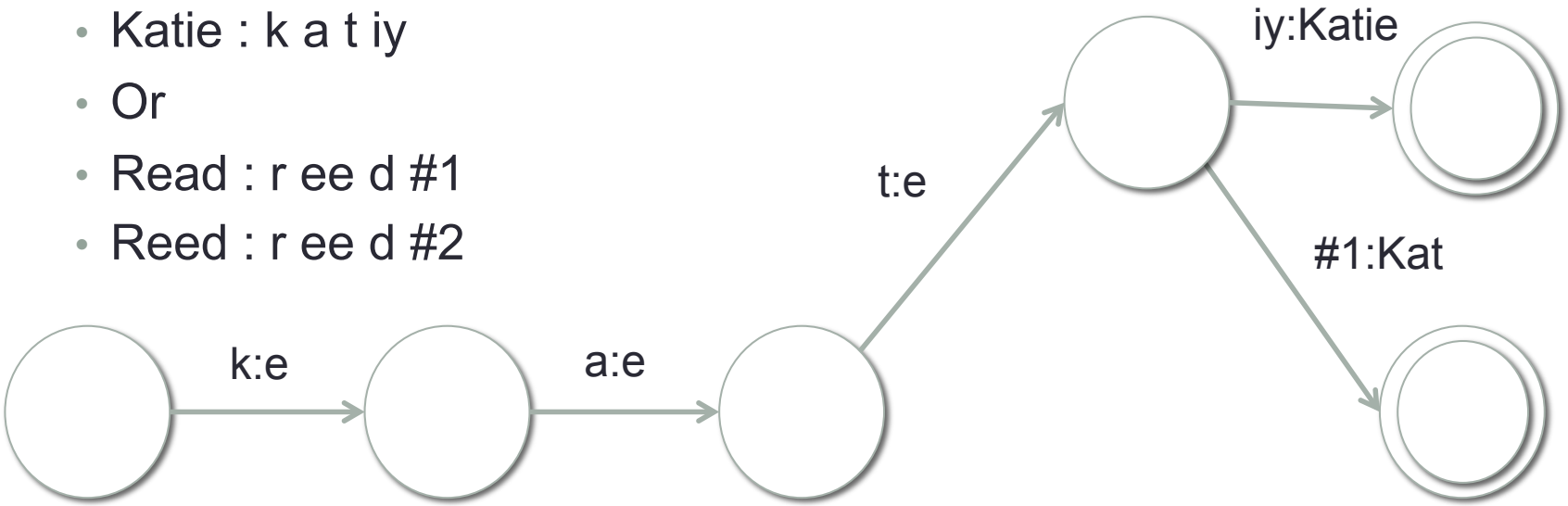
Undeterminizable L FST

- Consider
  - Kat : k a t
  - Katie : k a t iy
  - Or
  - Read : r ee d
  - Reed : r ee d



Disambiguation symbols

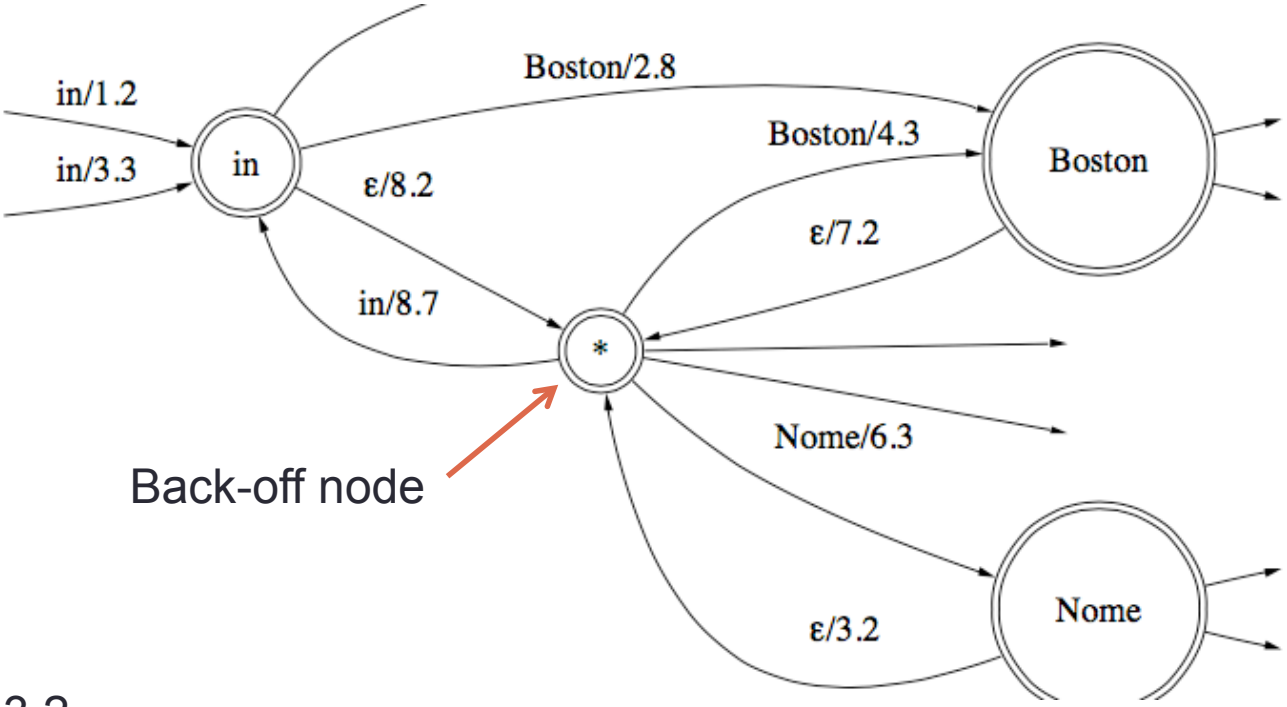
- Add extra disambiguation symbols #1, #2,… to make them different
  - Kat : k a t #1
  - Katie : k a t iy
  - Or
  - Read : r ee d #1
  - Reed : r ee d #2



- Disambiguation symbols are placeholders with no cost.

LM FST (G FST)

- N-gram can be easily represented as FSTs
- Input : word, Output : word
- Weight is –log(P)
- The nodes represent memories of the history
- Back-off weights can be represented by a special node with epsilon transition



\1-grams:
-4.3 Boston -3.2
-6.3 Nome -7.2
-8.7 in -8.2

\2-grams:
-2.8 in Boston

$$P(w_i|w_{i-1}) = \begin{cases} f(w_i|w_{i-1}) & c(w_{i-1}w_i) \geq \alpha \\ f_d(w_i|w_{i-1}) & \alpha > c(w_{i-1}w_i) > 0 \\ q(w_{i-1})P(w_i) & c(w_{i-1}w_i) = 0 \end{cases}$$

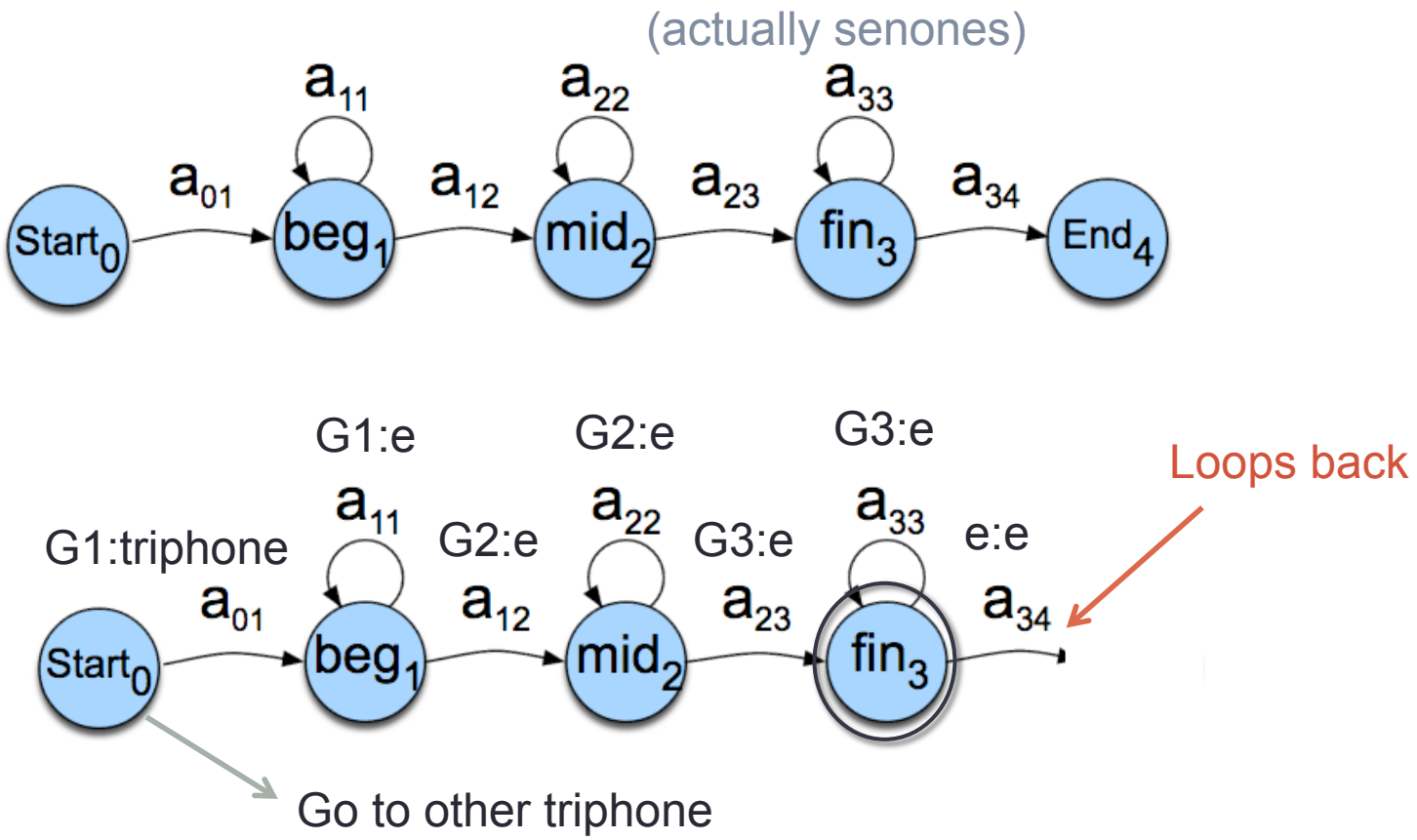
This is also not deteriminizable. Use disambiguation symbols on back-off arcs.

AM FST (H and C FST)

- Two parts : H (HMM FST) and C (Context dependent FST)
- H FST represents the three state HMM for each triphones.
- C FST maps triphones to phonemes needed by the lexicon.
  - Also enforce proper ordering
    - iy-ng-s can be followed by ng-s-*

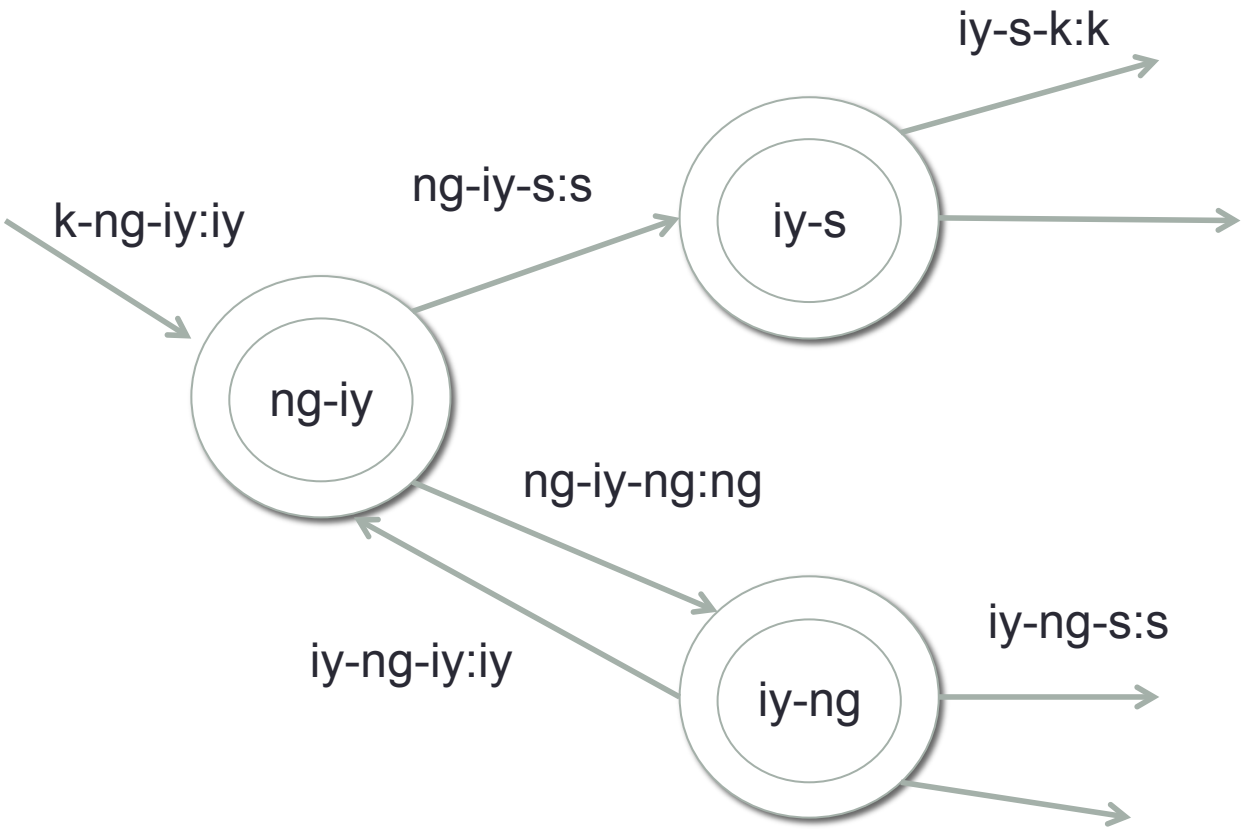
H FST

- Input : GMM id, output : triphone label i.e. iy-ng-s



C FST

- Input : triphone label, output : phoneme



HCLG FST

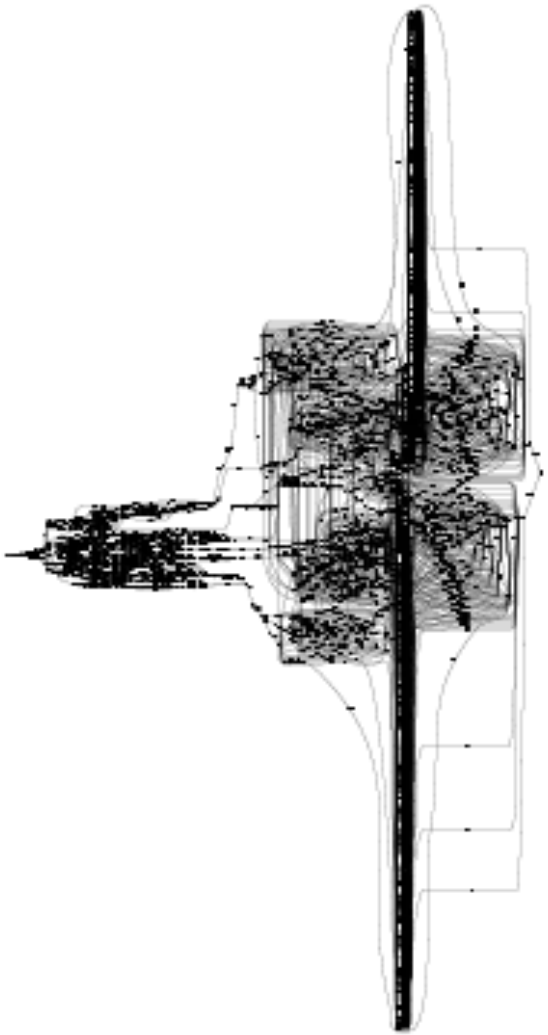
- The final FST is a composition of all 4

- HCLG = determinize(H o C o L o G)

- H input GMM ID, output triphone
- C input triphone, output phoneme
- L input phoneme, output word
- G input word, output word

HCLG FST

- Usually very large (>100Mb)
- Size goes up with
  - Vocab size
  - N-gram order
    - Limits the n-gram order used
    - Most use trigram
    - Possible to use higher n-gram by rescoring
- Size effects memory required to do ASR
  - Cannot do large vocab ASR in phones



FST for spell checker?

- Hint1 : What should the input be?
- Hint2 : What is an FST for deletion, insertion, and substitution of a single letter.

Inside the recognizer