# Introduction to
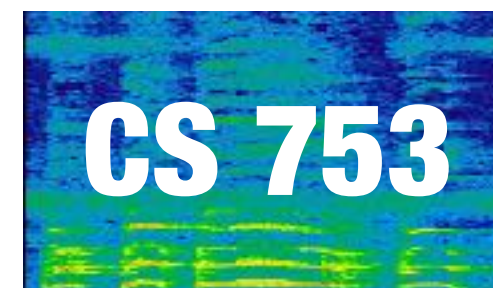# CS753: Automatic Speech Recognition

## Lecture 0



Instructor: Preethi Jyothi

# Automatic Speech Recognition

- Problem statement: T̶...̶nce into a sequence of tokens (words, syllab̶...̶s)

- Many downstream a̶...̶les:
  - Speech understanding
  - Spoken translation

# Automatic Speech Recognition



- Proble... ... ...e of tokens...

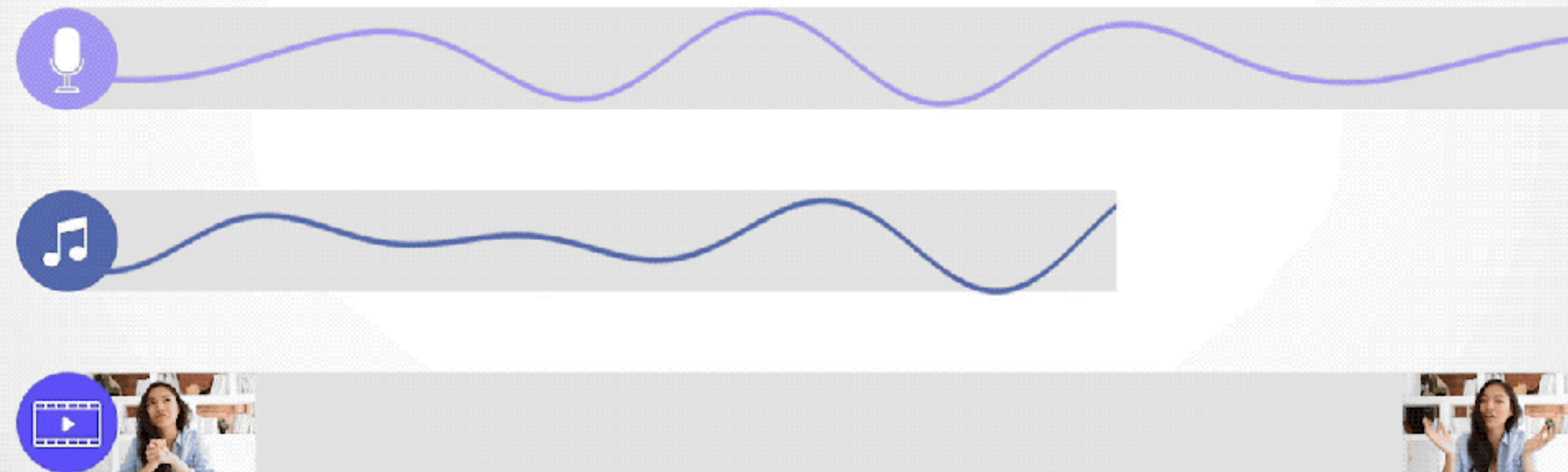- Many ...
  - Spee...
  - Spok...

# Automatic Speech Recognition

- Problem statement: Transform a spoken utterance into a sequence of tokens (words, syllables, phonemes, characters)

- Many downstream applications of ASR. Examples:

  - Speech understanding

  - Spoken translation

  - Intelligent video editing

# Automatic Speech Recognition

- Prob‌‌ of toke

- Mar
  - S
  - S
  - In

# Automatic Speech Recognition

- Problem statement: Transform a spoken utterance into a sequence of tokens (words, syllables, phonemes, characters)

- Many downstream applications of ASR. Examples:

  - Speech understanding

  - Spoken translation

  - Intelligent video editing

- Speech demonstrates variabilities at multiple levels: Speaker style, accents, room acoustics, microphone properties, etc.

# Automatic Speech Recognition

- Problem s... ...ce of tokens (wo...

- Many dow...

  - Speech u...

  - Spoken tr...

  - Intelligent...

- Speech de... ...yle, accents, r...

| Device | Channels | Use-cases |
|---|---|---|
| Matrix Voice | 7 | Internet of Things (IoT) systems, voice assistants, smart home products |
| ReSpeaker | 7 | Internet of Things (IoT) systems, voice assistants, smart home products |
| PlayStation Eye | 4 | Gaming Consoles |
| USB microphone | 1 | Embedded-scale IoT systems (e.g., with Raspberry pi) |
| Google Nexus 6 | 3 | Smartphone interaction, voice assistants |
| Shure MV5 | 1 | Desktop microphone for podcasts, video conferencing |

| | Matrix | ReSpeaker | USB | Nexus | Shure | PS Eye |
|---|---|---|---|---|---|---|
| Matrix | **0.055215** | 0.155436 | 0.073249 | 0.110685 | 0.069024 | 0.119291 |
| ReSpeaker | 0.807440 | **0.056819** | 0.154067 | 0.158762 | 0.127232 | 0.144229 |
| USB | 0.312770 | 0.098500 | **0.044086** | 0.094666 | 0.055685 | 0.096603 |
| Nexus | 0.461204 | 0.108495 | 0.092945 | 0.081738 | **0.054355** | 0.087136 |
| Shure | 0.622235 | 0.126587 | 0.257692 | 0.115106 | **0.040585** | 0.088368 |
| PS Eye | 0.612455 | 0.119135 | 0.257711 | 0.110959 | 0.055802 | **0.043578** |

Image from: https://github.com/akhilmathurs/libriadapt/

# How are ASR Systems Evaluated?

- Error rates computed on an unseen test set by comparing W* (predicted sentence) against $W_{ref}$ (reference sentence) for each test utterance

  - Sentence/Utterance error rate (trivial to compute)

  - Word/Phone error rate

- Word/Phone error rate (ER) uses the edit distance measure: What are the minimum number of edits (insertions/deletions/substitutions) required to convert $W^*$ to $W_{ref}$?

Reference ($W_{ref}$): hello world

ASR Prediction ($W^*$): hell o world

Edit distance: 2

# How are ASR Systems Evaluated?

- Error rates computed on an unseen test set by comparing W* (predicted sentence) against W$_{ref}$ (reference sentence) for each test utterance

  - Sentence/Utterance error rate (trivial to compute!)

  - Word/Phone error rate

- Word/Phone error rate (ER) uses the edit distance measure: What are the minimum number of edits (insertions/deletions/substitutions) required to convert $W^*$ to $W_{ref}$?
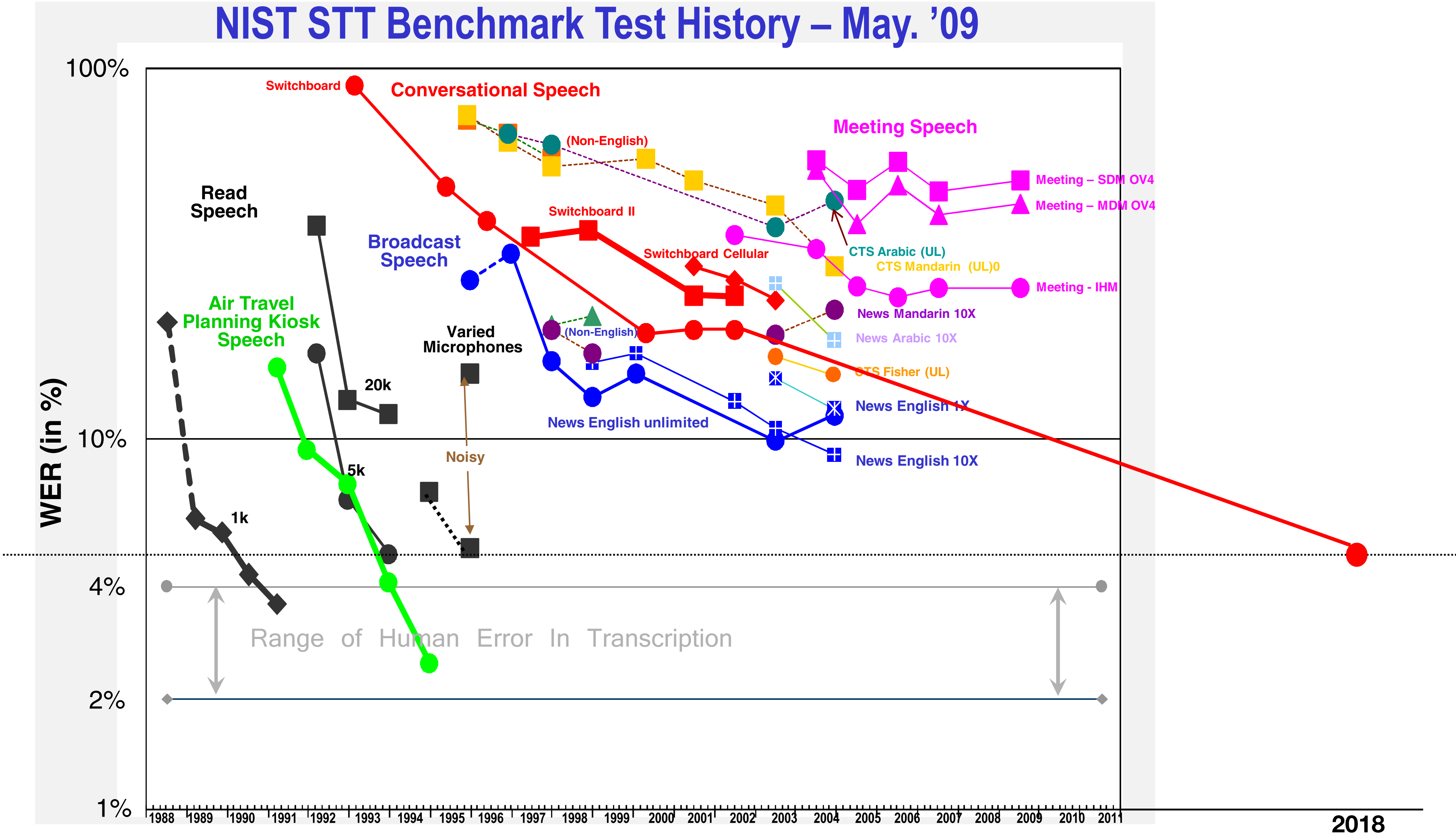
On a test set with $N$ instances:

$$\text{ER} = \frac{\sum_{j=1}^{N} \text{Ins}_j + \text{Del}_j + \text{Sub}_j}{\sum_{j=1}^{N} \ell_j}$$
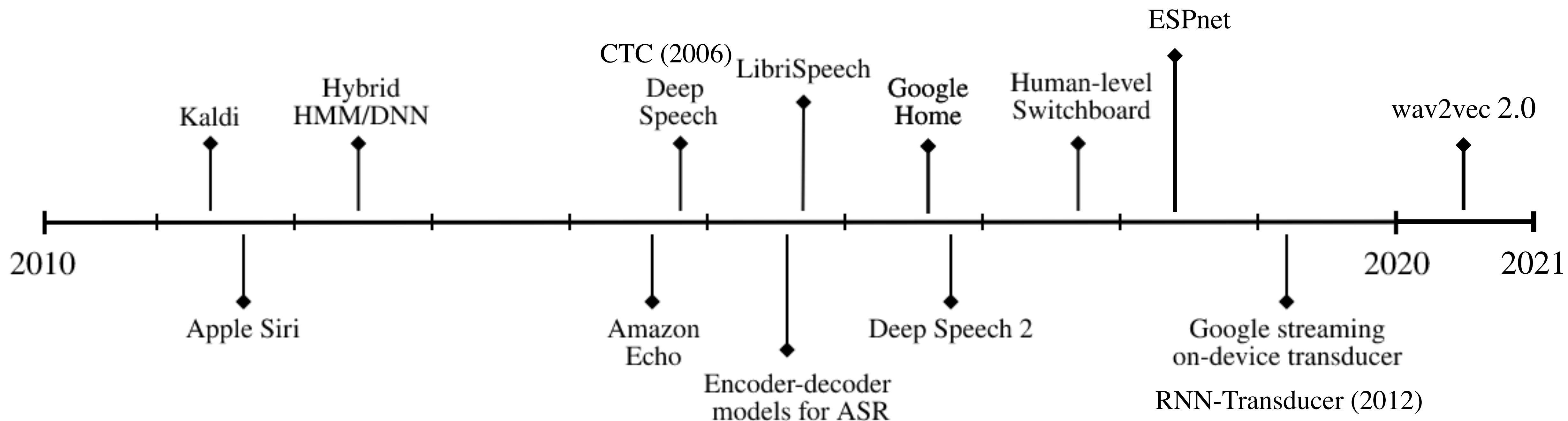
Ins$_j$, Del$_j$, Sub$_j$ are number of insertions/deletions/substitutions in the jth ASR output

$\ell_j$ is the total number of words/phones in the jth reference

# WERs over the Years



NIST STT Benchmark Test History – May. '09

Image from: http://www.itl.nist.gov/iad/mig/publications/ASRhistory/

# Progress in ASR Over the Last Decade

# Statistical Speech Recognition

Pioneer of ASR technology, Fred Jelinek (1932 - 2010): Cast ASR as a channel coding problem.

Let $\mathbf{O}$ be a sequence of acoustic features corresponding to a speech signal. That is, $\mathbf{O} = \{O_1, \ldots, O_T\}$, where $O_i \in \mathbb{R}^d$ refers to a $d$-dimensional acoustic feature vector and $T$ is the length of the sequence.

Let $\mathbf{W}$ denote a word sequence. An ASR decoder solves the foll. problem:

$$\mathbf{W}* = \arg\max_W \Pr(\mathbf{W} \,|\, \mathbf{O})$$

$$= \arg\max_W \Pr(\mathbf{O} \,|\, \mathbf{W}) \Pr(\mathbf{W})$$

# Simple example of isolated word ASR

- Task: Recognize utterances which consist of speakers saying either "up" or "down" or "left" or "right" per recording.

- Vocabulary: Four words, "up", "down", "left", "right"

- Data splits

  - Training data: 30 utterances

  - Test data: 20 utterances

- Let's parameterize $\Pr_\theta(\mathbf{O} \mid \mathbf{W})$ using a Markov model with parameters $\theta$.
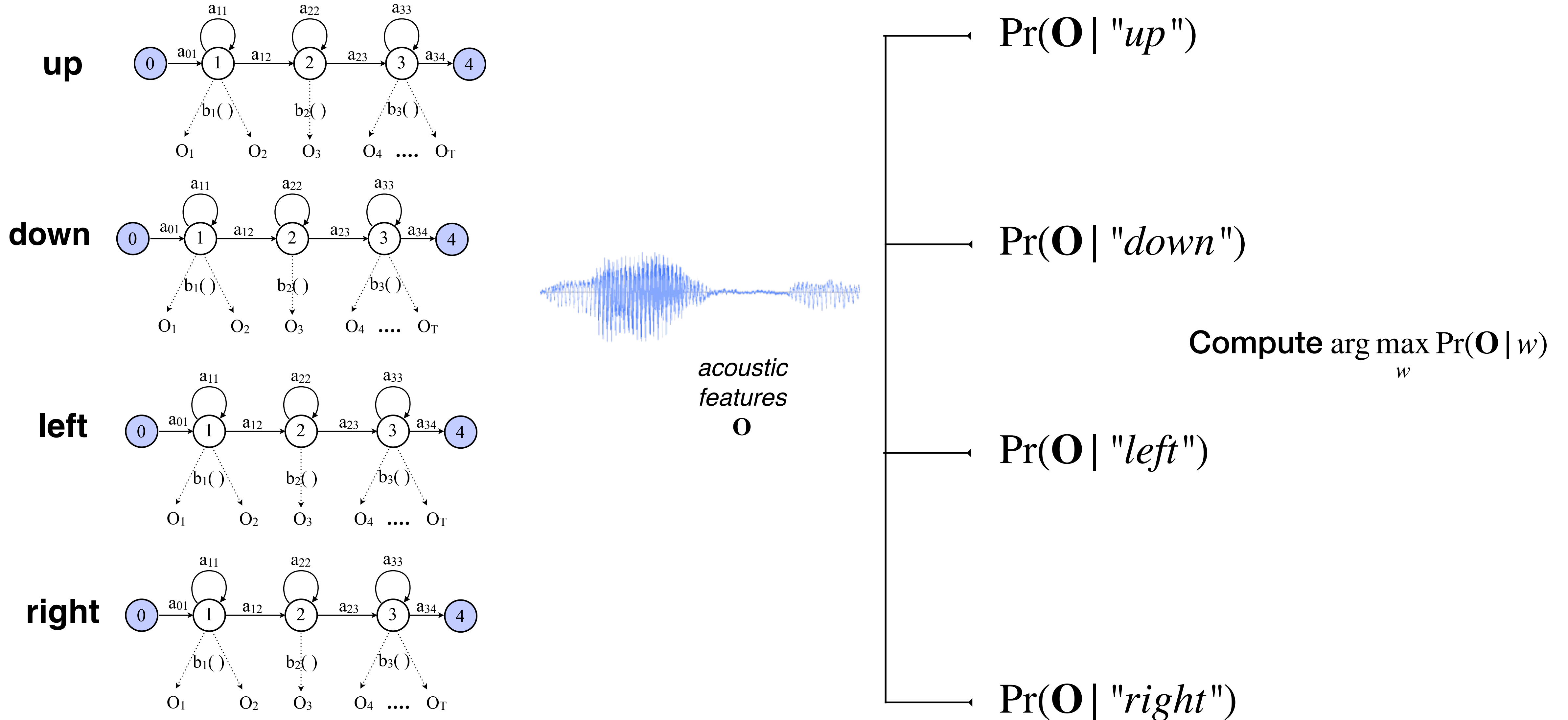
# Word-based acoustic model
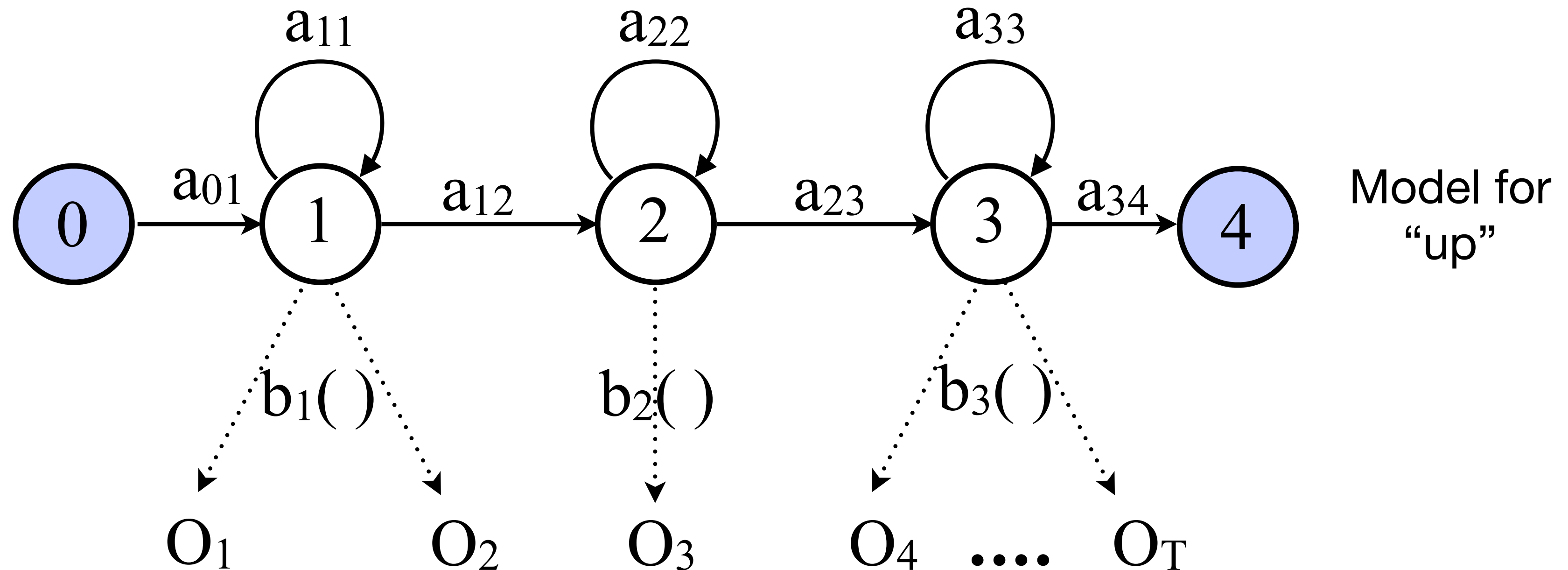


$a_{ij} \rightarrow$ Transition probabilities going from state $i$ to state $j$

$b_j(O_i) \rightarrow$ Probability of generating $O_i$ from state $j$

Compute $\Pr(\mathbf{O} \mid \text{"up"}) = \sum_{\mathbf{Q}} \Pr(\mathbf{O}, \mathbf{Q} \mid \text{"up"})$
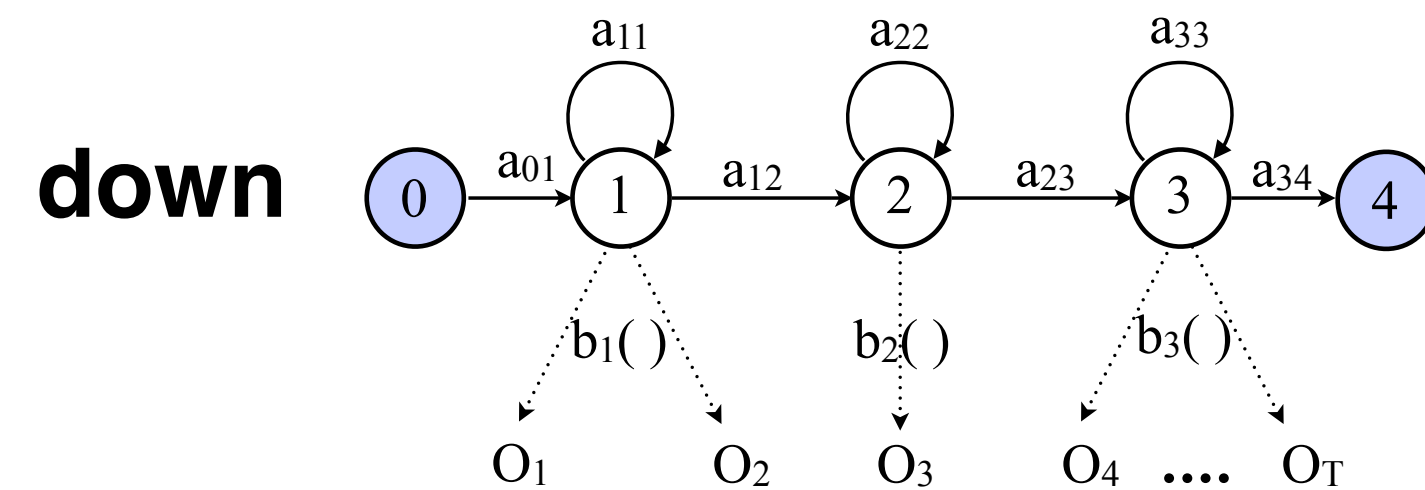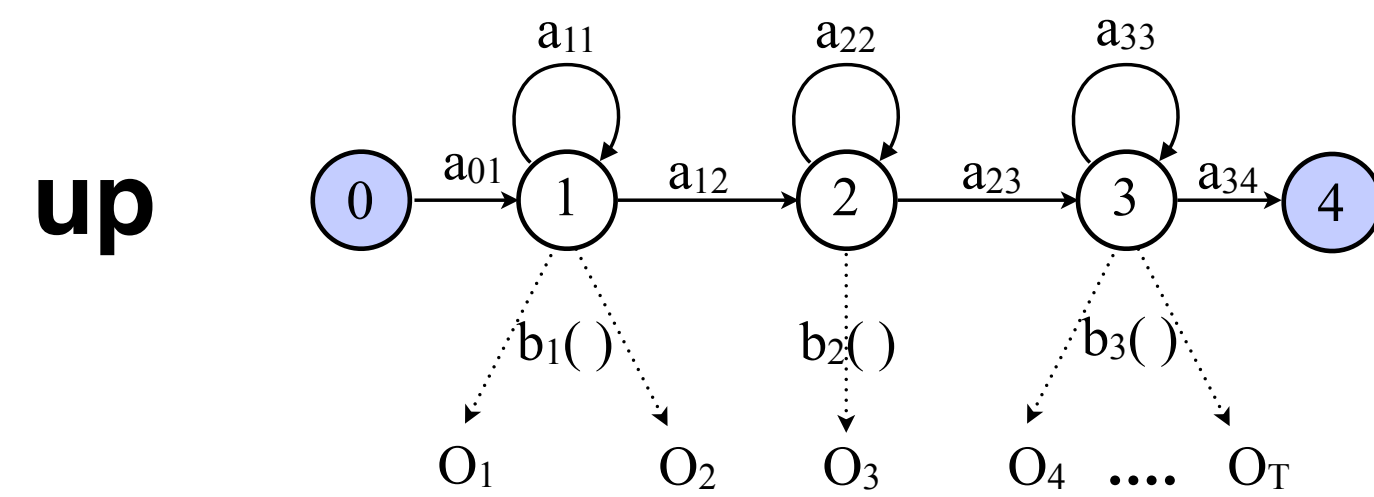
Efficient algorithm exists.
Will appear in a later class.
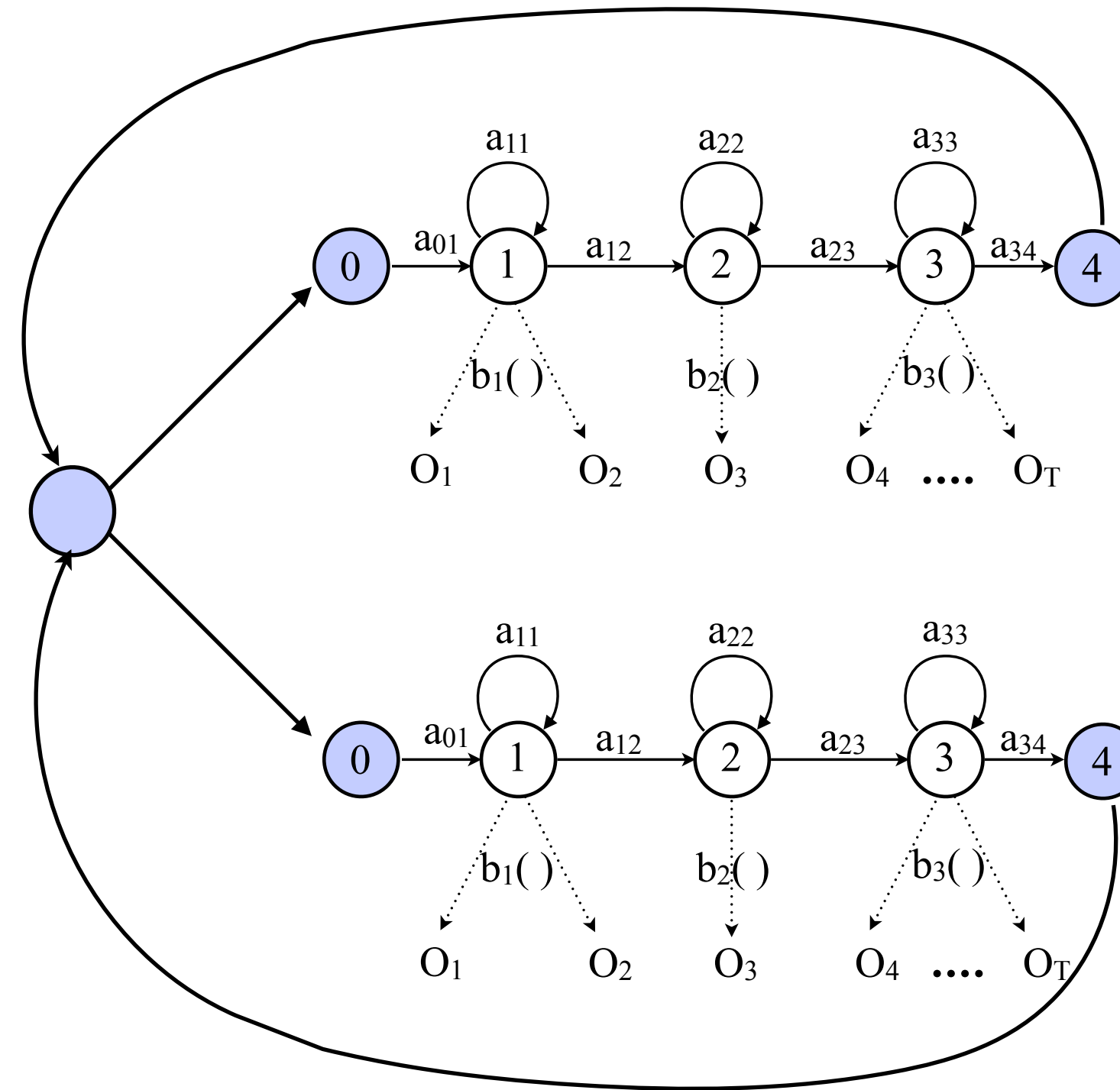
# Isolated word recognition

# Small tweak

- Task: Recognize utterances which consist of speakers saying either "up" or "down" **multiple times** per recording.

**up**



**down**

# Small tweak

- Task: Recognize utterances which consist of speakers saying either "up" or "down" **multiple times** per recording.



Search within this graph

# Small vocabulary ASR

- Task: Recognize utterances which consist of speakers saying one of 1000 words **multiple times** per recording.

- Not scalable anymore to use words as speech units

- Model using phones instead of words as individual speech units

  - Phonemes are abstract, subword units that distinguish one word from another (minimal pair; e.g. "pan" vs. "can")

  - Phones are actually sounds that are realized and not language-specific units

- What's an obvious advantage of using phones over entire words?
  Hint: Think of words with zero coverage in the training data.

# Statistical Speech Recognition

Let $\mathbf{O}$ be a sequence of acoustic features corresponding to a speech signal. That is, $\mathbf{O} = \{O_1, \ldots, O_T\}$, where $O_i \in \mathbb{R}^d$ refers to a $d$-dimensional acoustic feature vector and $T$ is the length of the sequence.

Let $\mathbf{W}$ denote a word sequence. An ASR decoder solves the foll. problem:

$$\mathbf{W}^* = \arg\max_W \Pr(\mathbf{W} \,|\, \mathbf{O})$$
$$= \arg\max_W \Pr(\mathbf{O} \,|\, \mathbf{W}) \Pr(\mathbf{W})$$

# Statistical Speech Recognition

Let $\mathbf{O}$ be a sequence of acoustic features corresponding to a speech signal. That is, $\mathbf{O} = \{O_1, \ldots, O_T\}$, where $O_i \in \mathbb{R}^d$ refers to a $d$-dimensional acoustic feature vector and $T$ is the length of the sequence.
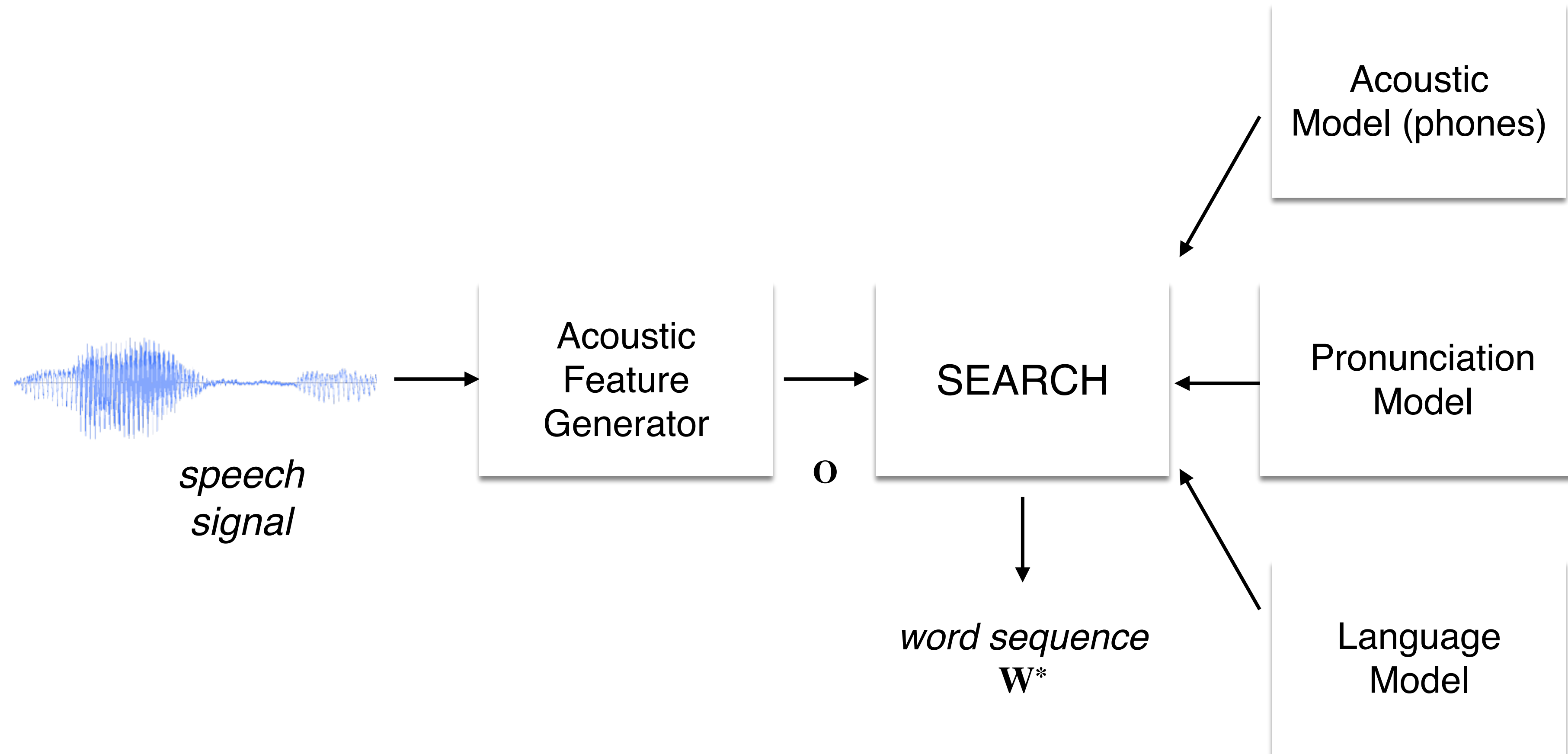
Let $\mathbf{W}$ denote a word sequence. An ASR decoder solves the foll. problem:

$$\mathbf{W}^* = \arg\max_W \Pr(\mathbf{W} \,|\, \mathbf{O})$$

$$= \arg\max_W \Pr(\mathbf{O} \,|\, \mathbf{W}) \Pr(\mathbf{W})$$

$$\approx \arg\max_W \sum_Q \Pr(\mathbf{O} \,|\, \mathbf{Q}) \Pr(\mathbf{Q} \,|\, \mathbf{W}) \Pr(\mathbf{W})$$
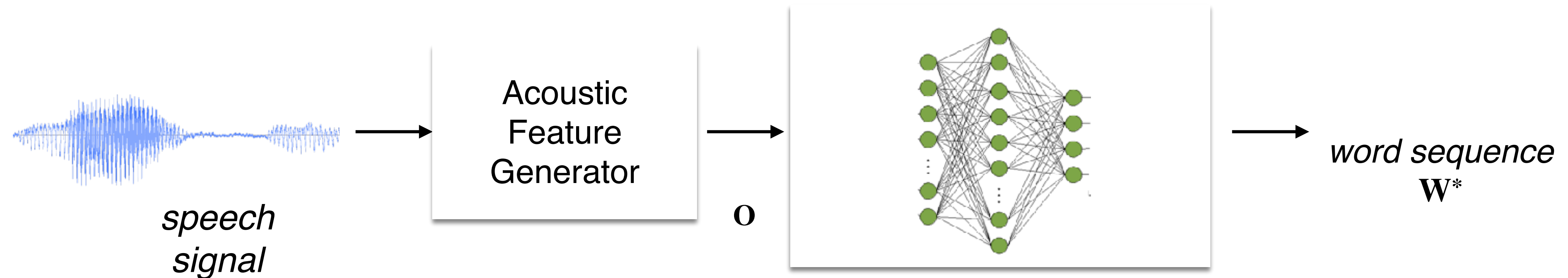
Language Model

Acoustic Model

Pronunciation Model

# Architecture of an ASR system

# Cascaded ASR $\Rightarrow$ End-to-end ASR



Single end-to-end model that directly learns a mapping from speech to text

# End-to-End ASR Systems

- All components trained jointly as a single end-to-end model

- Trained using pairs of speech clips and their corresponding text transcripts

- End-to-end models, with sufficient data, sometimes outperform conventional ASR systems



Speech Waveform

Acoustic Analysis

Acoustic Features

| Frame # | Acoustic Features | | | |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |
| : | | | | |

**Output Transcription**

|  | dev | test |
|---|---|---|
| DNN-HMM | 4.0 | 4.4 |
| E2E (Attention) | 4.7 | 4.8 |

Librispeech-960

|  | dev | test |
|---|---|---|
| DNN-HMM | 5.0 | 5.8 |
| E2E (Attention) | 14.7 | 14.7 |

Librispeech-100
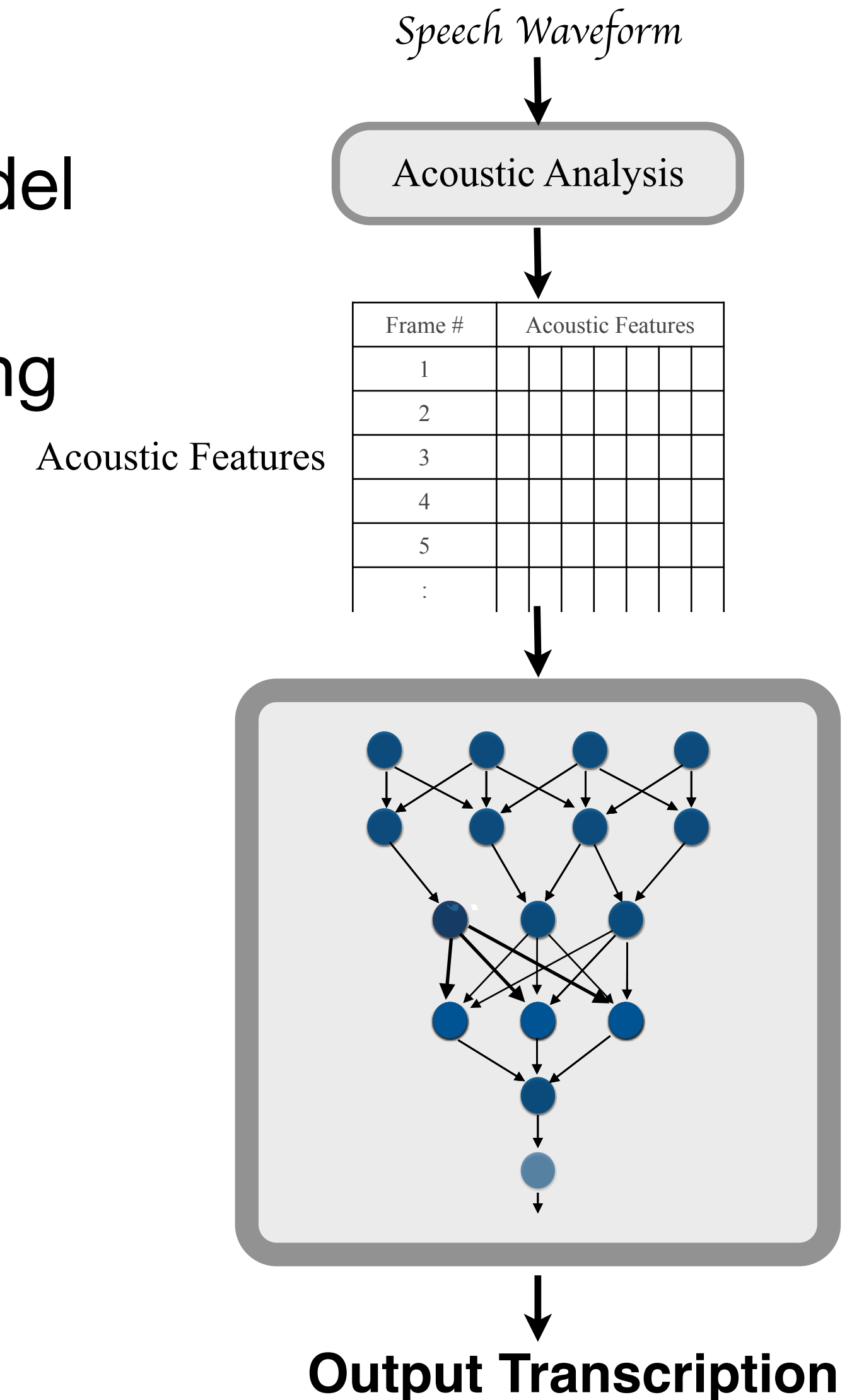
# ASR Progress

Voice Recognition Software Finally Beats Humans At Typing, Study Finds

**AUG 16**

Microsoft researchers achieve new conversational speech recognition milestone

**AUG 17**

Amazon's AI system could cut Alexa speech recognition errors by 15%

**MAR 19**

https://venturebeat.com/2019/04/22/amazons-ai-system-could-cut-alexa-speech-recognition-errors-by-15/

https://www.microsoft.com/en-us/research/blog/microsoft-researchers-achieve-new-conversational-speech-recognition-milestone/

https://www.npr.org/sections/alltechconsidered/2016/08/24/491156218/voice-recognition-software-finally-beats-humans-at-typing-study-finds

# Exciting Time to do Speech Research
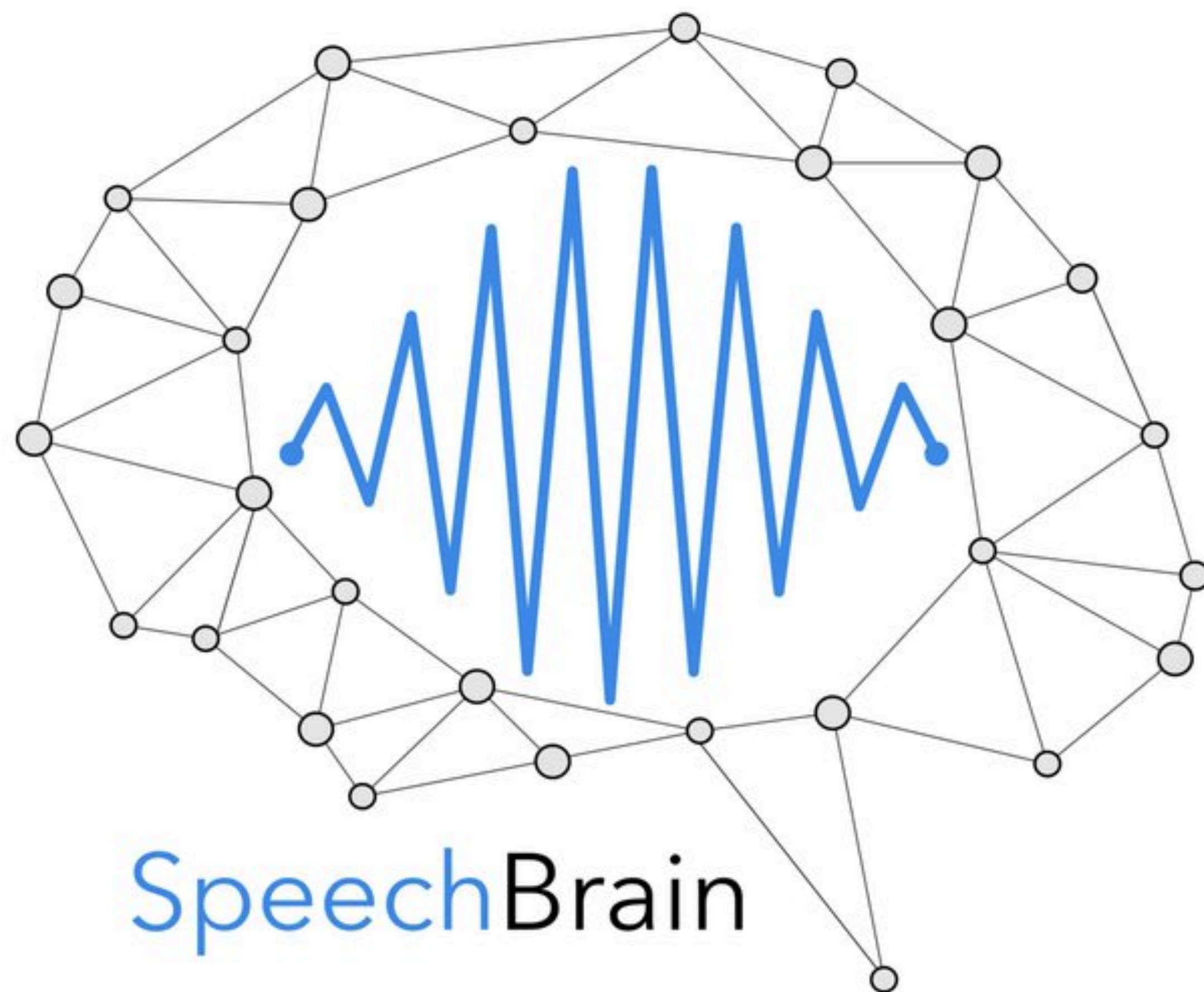
# Progress in ASR Over the Last Decade
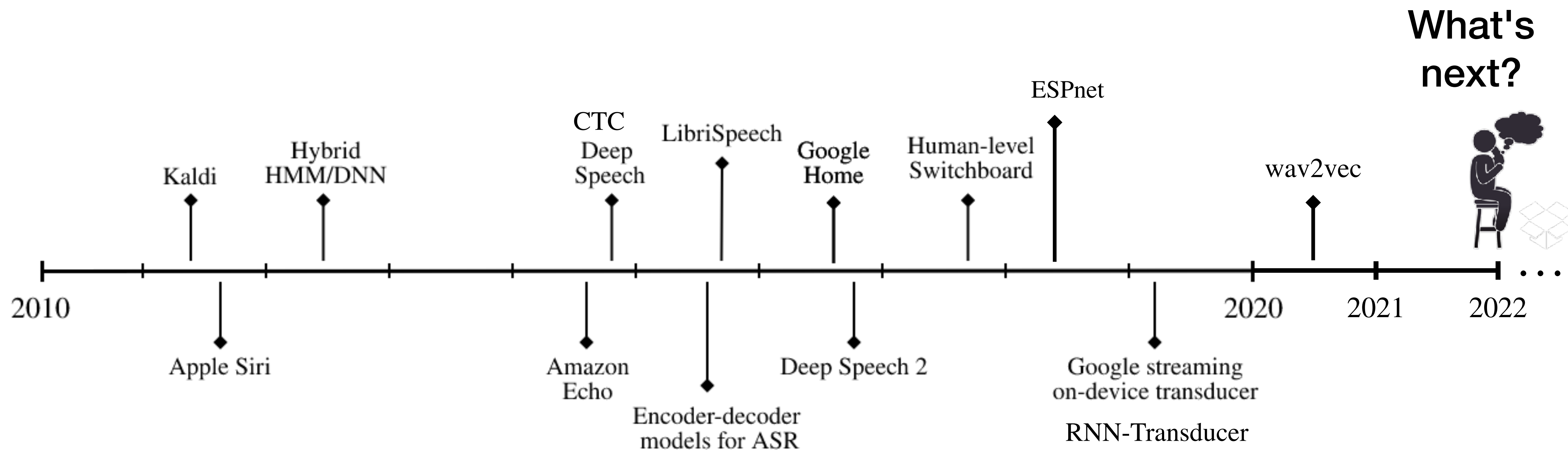
# Many Unsolved Problems Related to ASR

- State-of-the-art ASR systems do not work well on heavy regional accents, dialects.

- Code-switching is hard for ASR systems to deal with.

- How do we rapidly build competitive ASR systems for a new language?
  Low-resource ASR, multilingual ASR and pretrained models.

- Long-form ASR is still a challenge.

- How do we recognize speech from meetings where a primary speaker is speaking amidst other speakers?

# Course Syllabus (I)

- Cascaded ASR System
  - Acoustic Model (**AM**)
  - Pronunciation Model (**PM**)
  - Language Model (**LM**)

- Weighted Finite State Transducers for ASR

- **AM**: HMMs, DNN and RNN-based models

- **PM**: Phoneme and Grapheme-based models

- **LM**: Ngram models (+smoothing), RNNLMs

- Decoding Algorithms, Lattices



*Speech Waveform*

Acoustic Analysis

| Frame # | Acoustic Features |
|---------|------------------|
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| : | |

Acoustic Features

**AM**

DECODER

Acoustic model

| WORD | PRON |
|------|------|
| good | g uh d |
| like | l ay k |
| is | ih z |

Pronunciation model

**PM**

| NGRAMS | SCORE |
|--------|-------|
| Good prose | 2.5 |
| like | 0.7 |
| is like | 1.2 |
| is like a | 0.8 |

Grammar (language) model

**LM**

good prose is like a windowpane
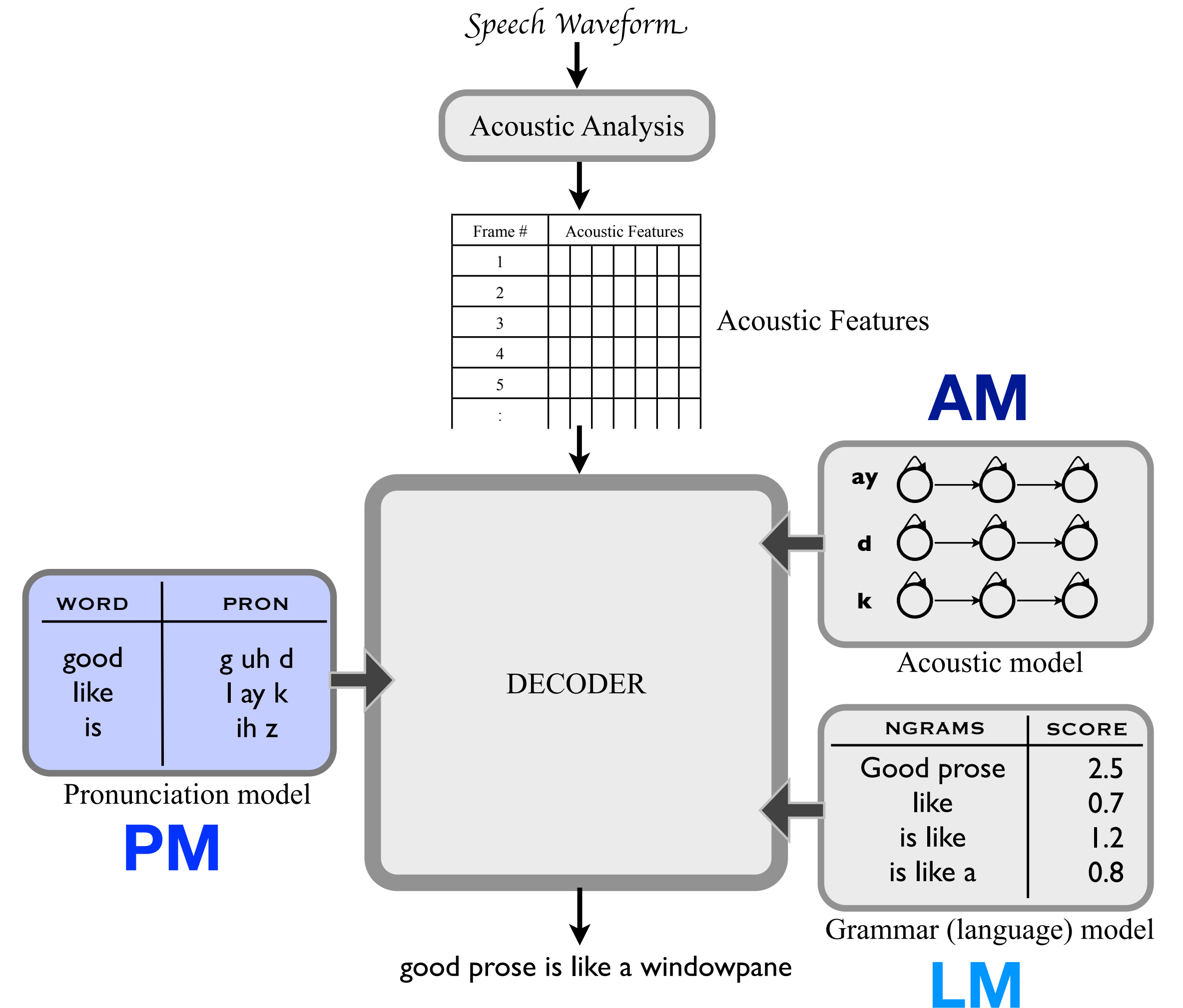
# Course Syllabus (II)



- End-to-end Neural Models for ASR

  - CTC loss function

  - Encoder-decoder models with Attention

- Acoustic Feature Analysis

- Learning Speech Representations

- Multilingual ASR

- Low-resource Speech Recognition

Image from: Chan et al., Listen, Attend and Spell: A NN for LVCSR, ICASSP 2016
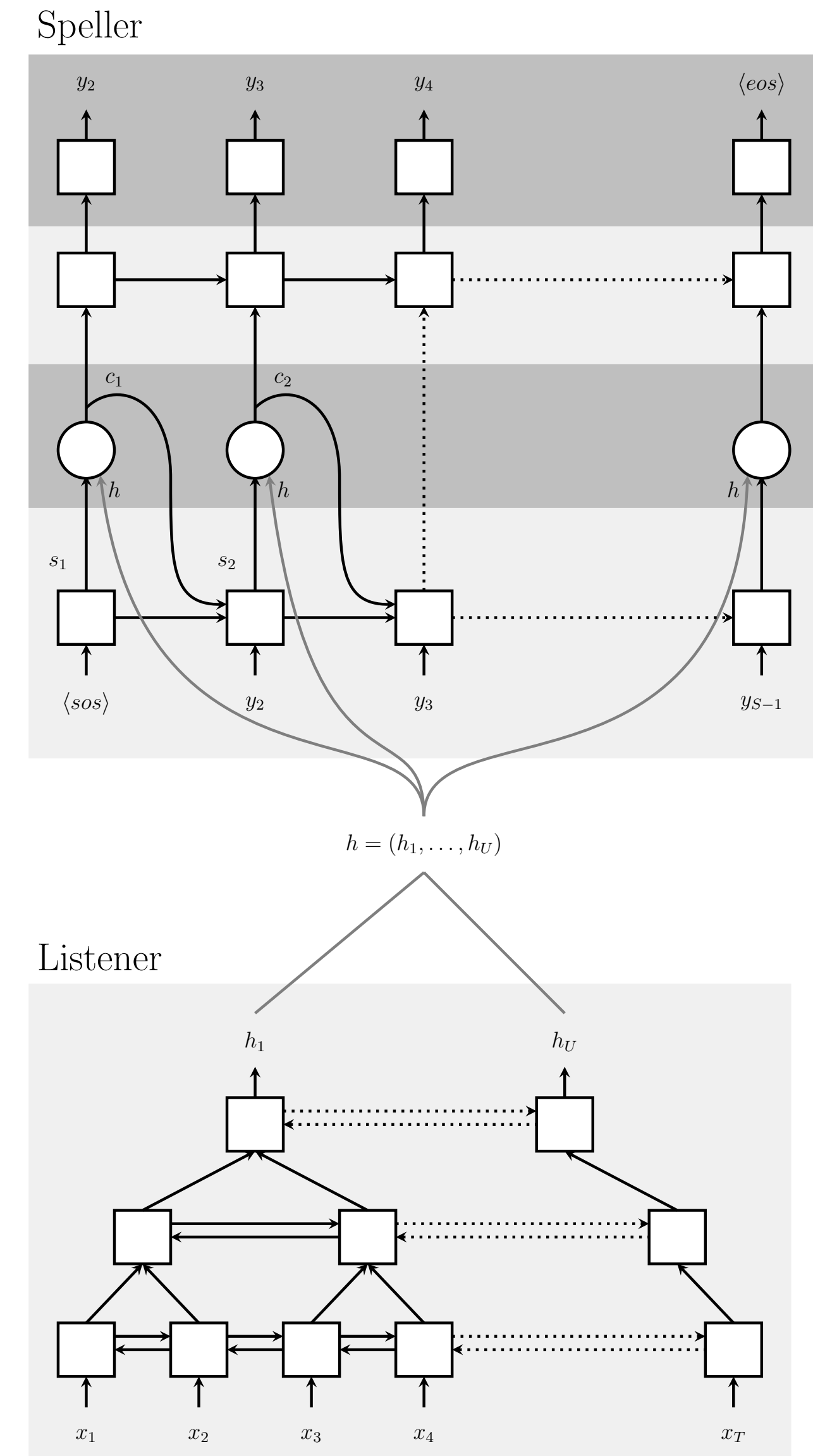
# Prerequisites

- Should have completed an introduction to ML (or equivalent) course. That is, one or more of CS 725, CS 726, CS 747, CS 419(M), EE 769, IE 663, GNR 652, GNR 638, DS 303.

- Programming assignments will have to be completed in Python. (We will make use of Python libraries wherever relevant.)

- Should be comfortable with basic probability and linear algebra for ML.

# Course Logistics

**Reading:** All mandatory reading will be freely available online and linked on Moodle.

**Prerecorded lectures:** All content (slides + videos) will be posted on Moodle.

**Asynchronous Q&A:** Moodle discussion forums + MS Teams class group.

**Interactive sessions:** Roughly once a week during class hours via Webex.
**Class hours:** Tue (5:30 pm - 6:55 pm), Fri (5:30 pm - 6:55 pm)

**TA office hours:** 4-5 hours spread through the week. Will be posted on Moodle.

# Other Course Information

- Teaching Assistants (TAs):

  - Barah Fazili (barah AT cse)

  - Rohit Kundu (rkundu AT cse)

  - Shubham Nemani (nshubham AT cse)

  - Pranjal Saini (pranjalsaini AT cse)
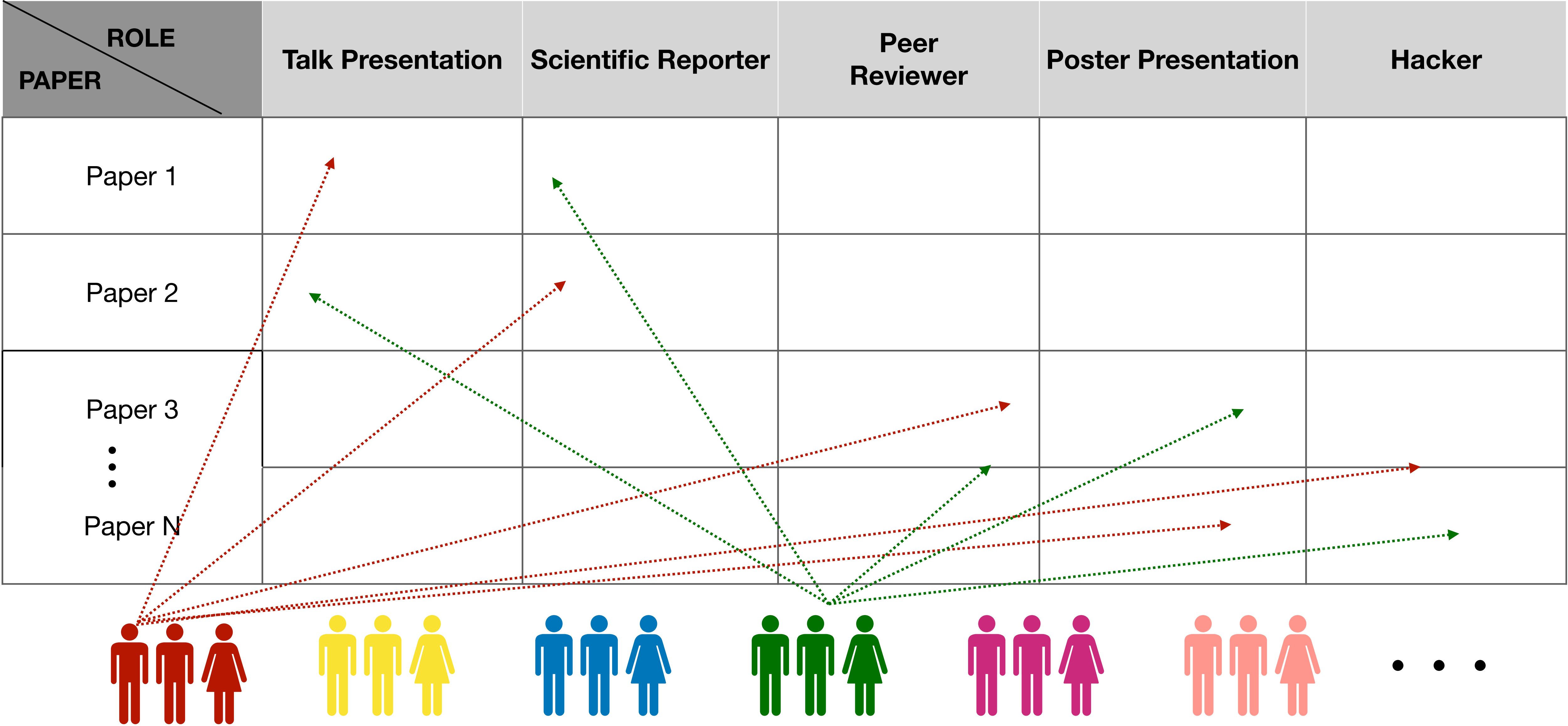
# Other Course Information

- Teaching Assistants (TAs):

  - Barah Fazili (barah AT cse)

  - Rohit Kundu (rkundu AT cse)

  - Shubham Nemani (nshubham AT cse)

  - Pranjal Saini (pranjalsaini AT cse)

- Readings:

  - No single reference book. "**Speech and Language Processing**" by Jurafsky and Martin will serve as a good overall reference. (Link is up on Moodle.)

  - All readings will be freely available and posted online on Moodle.

- Audit requirements: Complete one of two assignments and score $\geq 40\%$; Gain at least 2.5 (out of 5) participation points

# Course Evaluation

- 2 Assignments      **20%**

- Proctored Midsem Exam      **15%**

- Participation (In-class Moodle Quizzes)      **5%**

- Proctored Final Exam      **30%**

- Role-playing Seminars      **30%**

# Role-Playing Seminars[1]
## (More details will be shared on Moodle)



| ROLE / PAPER | Talk Presentation | Scientific Reporter | Peer Reviewer | Poster Presentation | Hacker |
|---|---|---|---|---|---|
| Paper 1 | | | | | |
| Paper 2 | | | | | |
| Paper 3 ⋮ Paper N | | | | | |

[1]Inspired by: https://colinraffel.com/blog/role-playing-seminar.html

# Role-Playing Seminars
## (More details will be shared on Moodle)

- ## Talk Presentation (

  - Give a 15-20 min talk (as the author) to clearly explain the main ideas in the paper.

- ## Scientific Reporter / Journalist

  - Write a technical article about the paper for a non-specialist but general CS audience (e.g., Quanta Magazine[1]).

- ## Peer Reviewer

  - Provide a full review of the paper.

- ## Poster Presentation

  - Prepare a poster visualising the main highlights of the paper.

- ## Hacker

  - Implement a small part/simplified version of the paper on a synthetic dataset or toy problem.

# Role-Playing Seminars
## (More details will be shared on Moodle)

- No more than 3 members in a team.

- List of over 50 papers will be posted on Moodle.

  - Each team gives a ranking over papers for the talk/poster presentation and hacker roles.

  - Other two roles will be randomly assigned across papers.

- Evaluation:

  - Each role will be evaluated on specific metrics.

  - Poster presentations will all be evaluated together at the end of the as a session on gather.town .

  - "*Hacker*" component will be evaluated at the end of the semester.

  - Exceptional performance in any role will be awarded extra credit points.

# Academic Integrity Policy
## Assignments/Exams



- Do not copy or plagiarise. Will incur significant penalties.

- Abide by an honour code. If caught for copying or plagiarism, name of both parties will be handed over to the Disciplinary Action Committee (DAC)[1].

- Always cite your sources (be it images, papers or existing code repos). Follow proper citation guidelines.

# Next class: HMMs for Acoustic Modeling