Crash Course in ASR (An attempt)

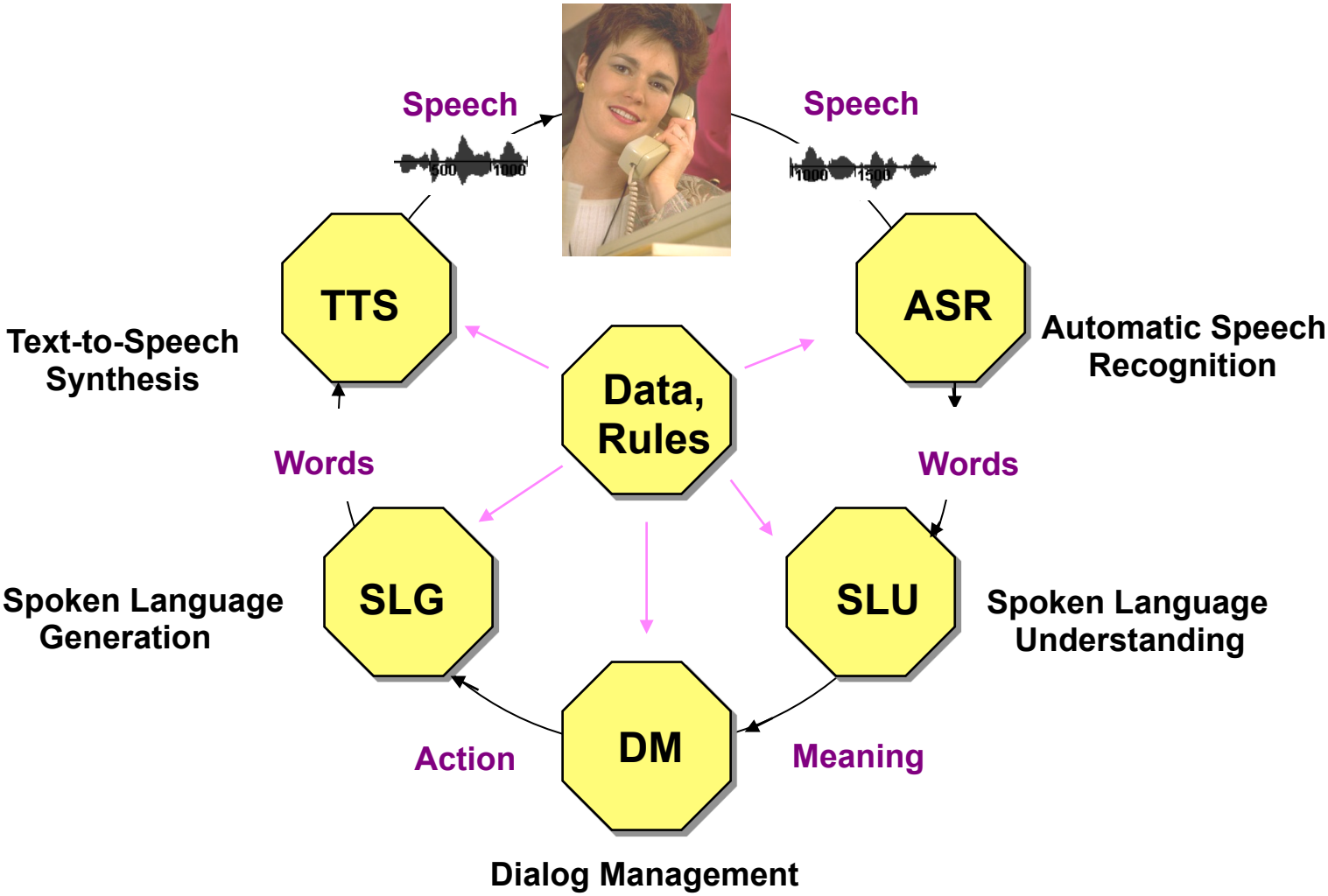
Automatic Speech Recognition (ASR)

Marco Siniscalchi, Ph.D.
Kore University of Enna
94100 Enna, Italy

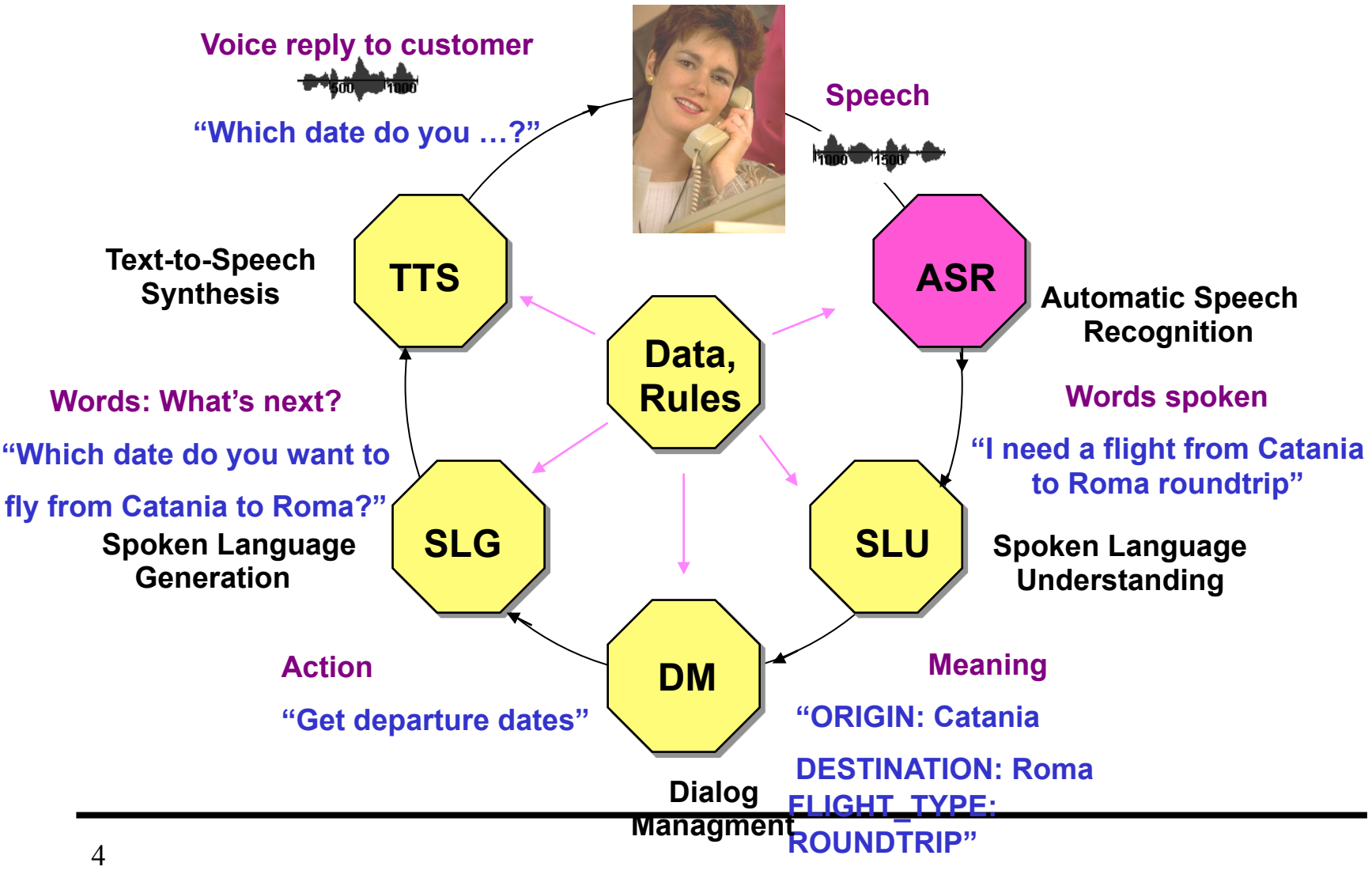
Lecture Outline

- The Speech Dialog Circle
  - System components
- Automatic Speech Recognition (ASR)
  - System components and technology dimensions
  - Implementation issues
- Theory of Markov Models
  - Discrete Markov processes
  - Hidden Markov processes
- Solutions to the Three Basic Problems of HMM's
  - Computation of observation probability
  - Determination of optimal state sequence
  - Optimal training of model

Voice-Enabled System Technology Components



The Speech Dialog Circle

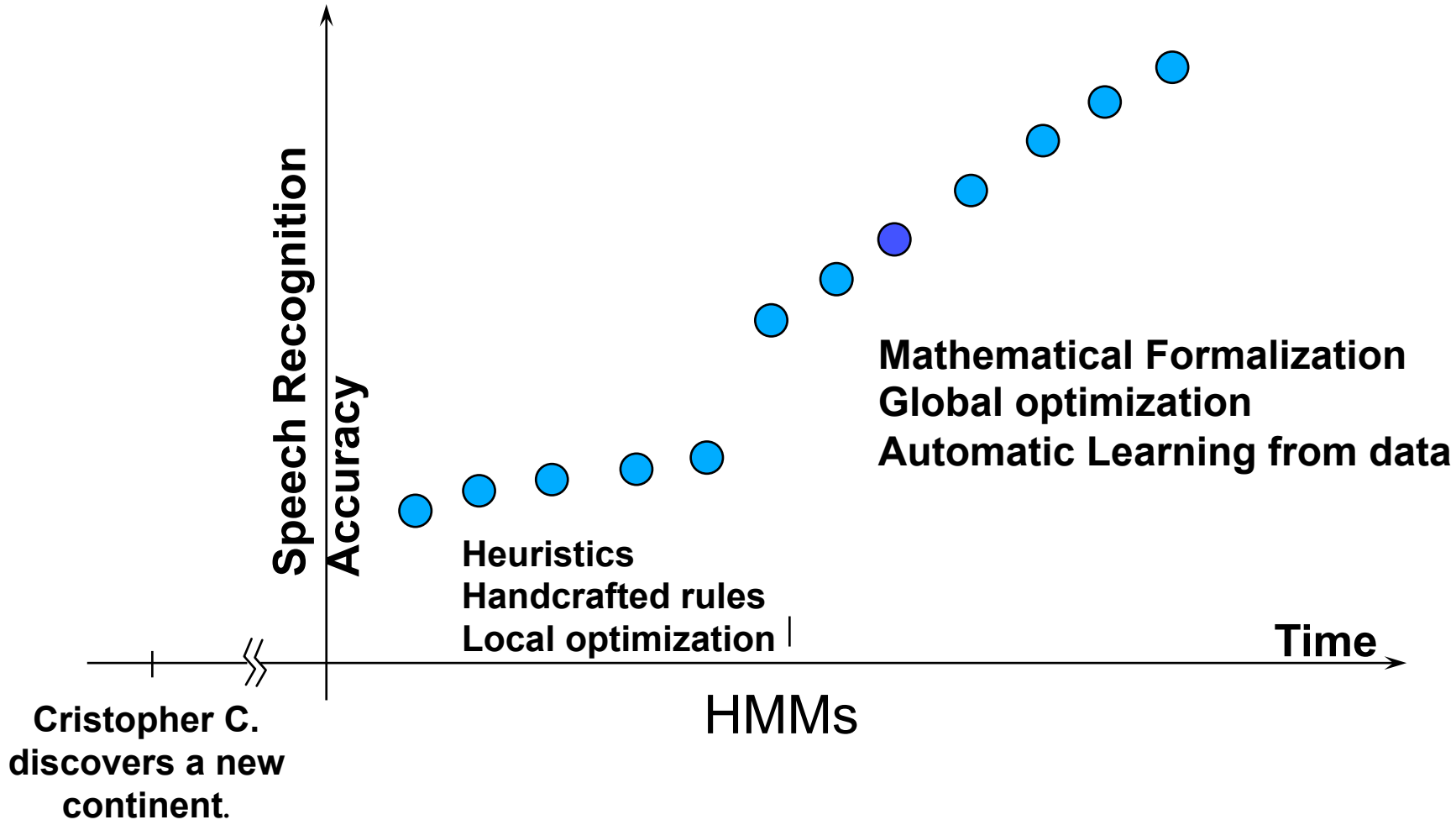


Automatic Speech Recognition (ASR)

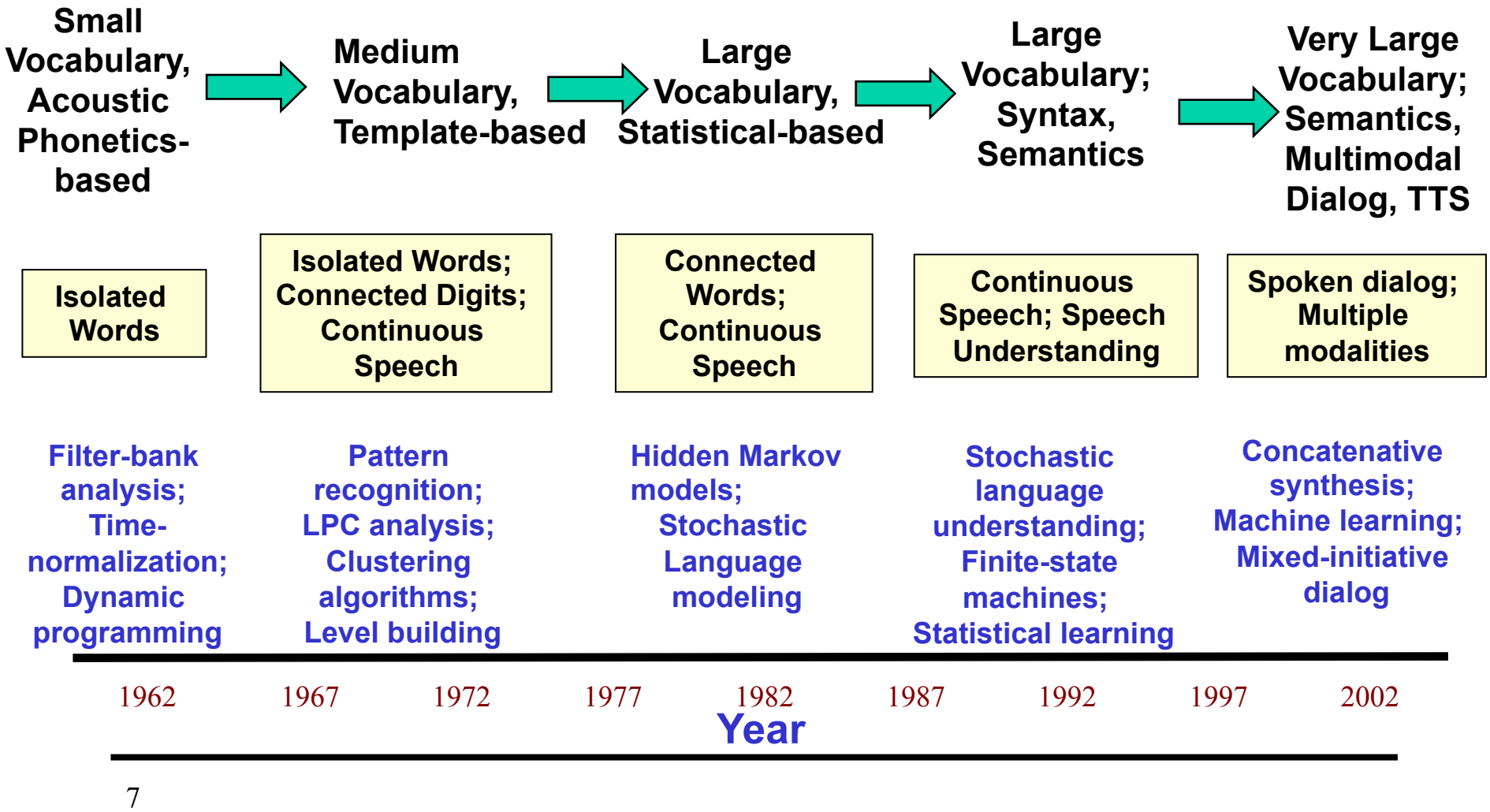
Goal: *Accurately* and *efficiently* convert a speech signal into a text message independent of the device, speaker or the environment.

Applications: Automation of complex operator-based tasks, e.g., customer care, dictation, form filling applications, provisioning of new services, customer help lines, e-commerce, etc.

A Brief History of Speech Recognition

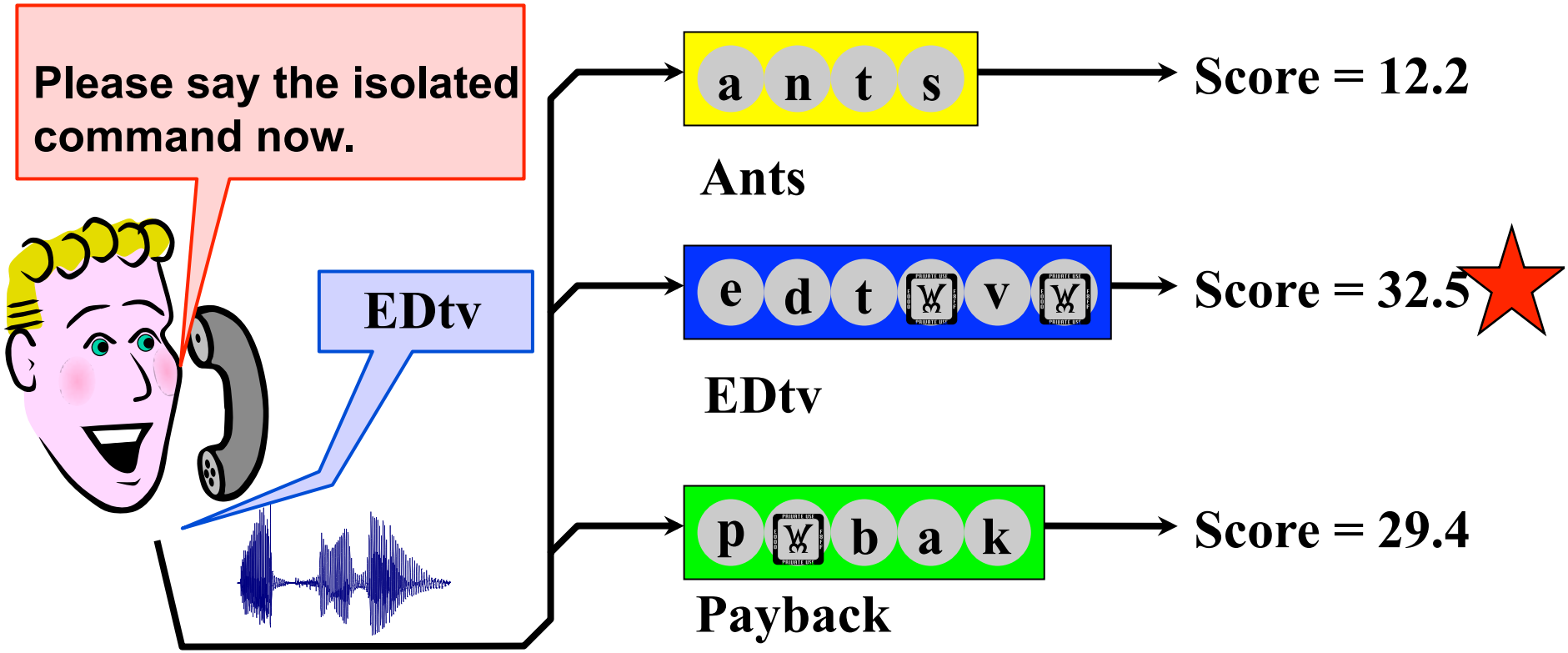


Milestones in ASR Technology Research

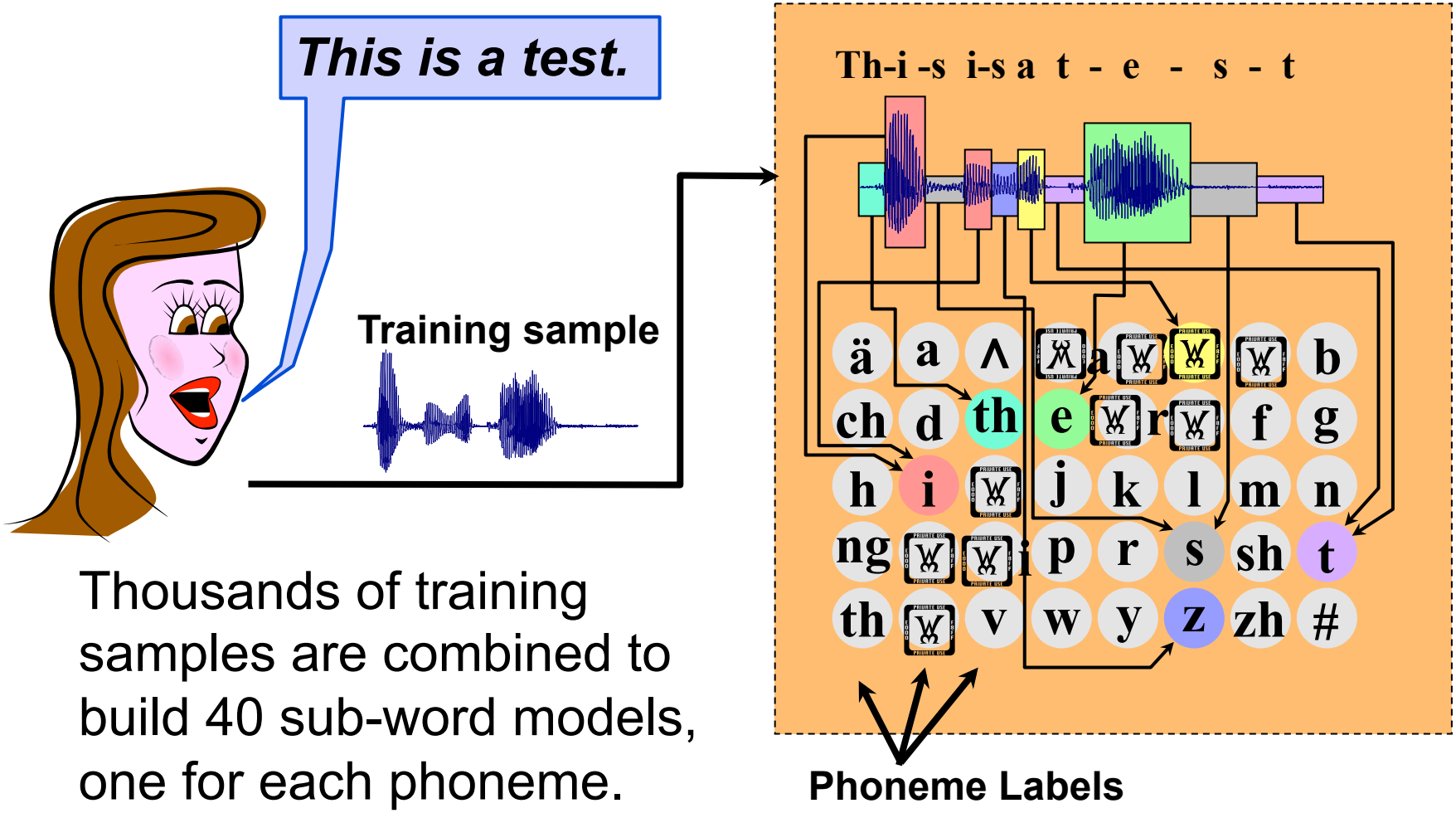


Isolated Word Recognition (IWR)

- Computing word score using a sequence of phone scores

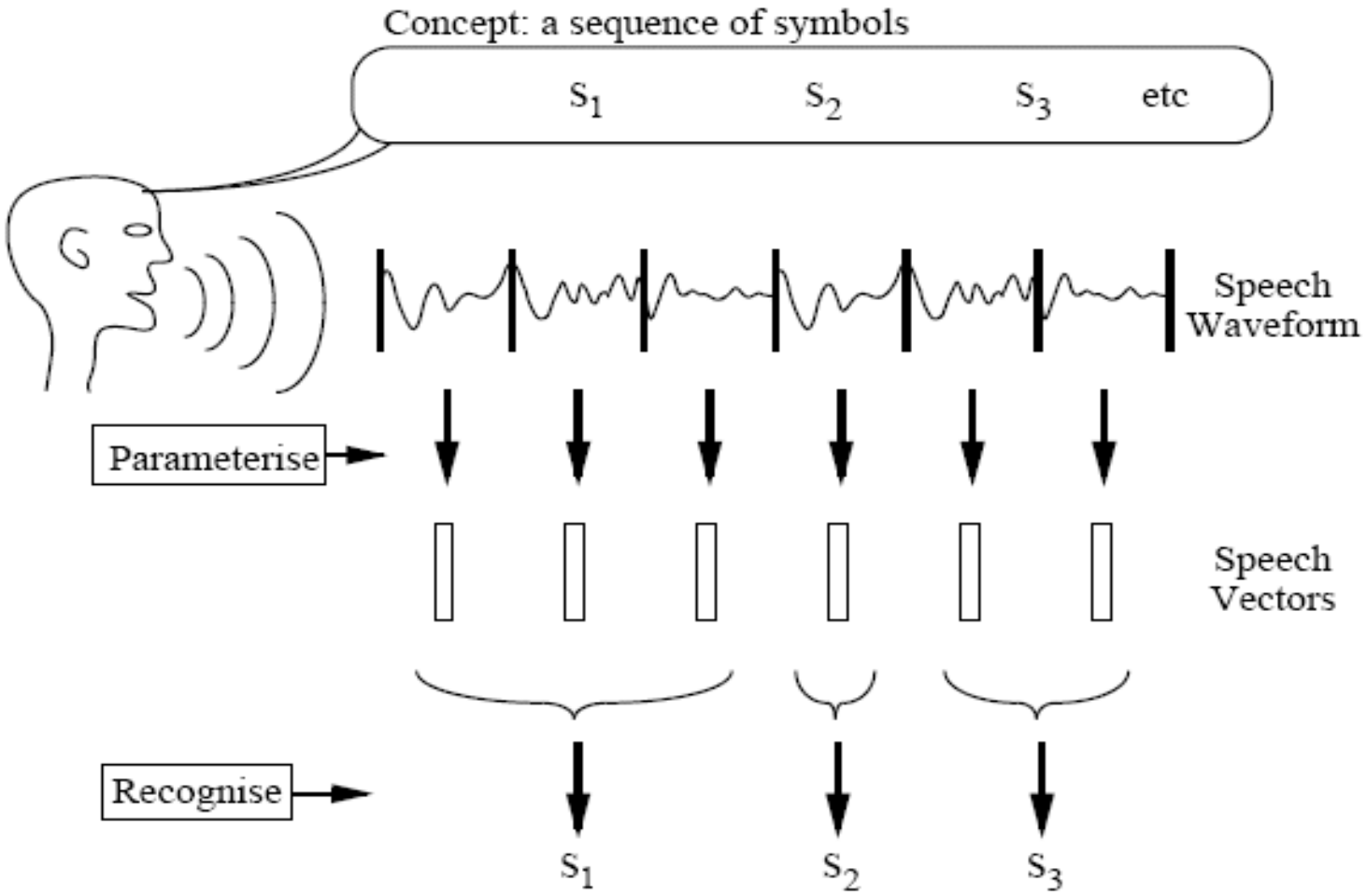


Training Speech Recognizer



Thousands of training samples are combined to build 40 sub-word models, one for each phoneme.

Automatic Speech Recognition (ASR)



ASR System Components

- Feature Extraction
  - Framing and short-time spectral/cepstral analysis
- Acoustic Modeling of Speech Units
  - Fundamental speech unit selection
  - Statistical pattern matching (HMM unit) modeling
- Lexical Modeling
  - Pronunciation network
- Syntactic and Semantic Modeling
  - Deterministic or stochastic finite state grammar
  - *N*-gram language model
- Search and Decision Strategies
  - Best-first or depth-first, DP-based search
  - Modular vs. integrated decision strategies

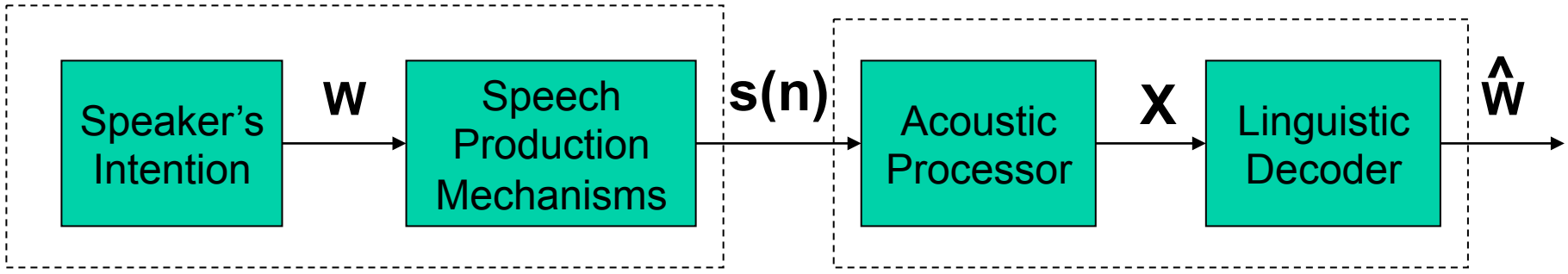
ASR Terminology

- Vocabulary
  - Words that can be recognized in an application
  - More words imply more errors and more computation
- Word spotting
  - Listening for a few specific words within an utterance
- Extraneous speech screening (rejection)
  - Capability to decide whether a candidate key word is a close enough match to be declared a valid key word
- Grammars
  - Syntax (word order) that can be used
  - The way words are put together to form phrases & sentences, some are more likely than others
  - Can be deterministic or stochastic
- Semantics
  - Usually not properly modeled or represented

ASR System and Technology Dimensions

- Isolated word vs. continuous speech recognition
  - Isolated = pauses required between each word
  - Continuous = no pauses required
- Speech unit selection: whole vs. sub-word
  - Whole word = requires data collection of the words to be recognized
  - Sub-word = recognizing basic units of speech (phonemes), so word recognition is easier for new applications
- Read vs. spontaneous (degree of fluency)
- Small vs. medium or large vocabulary
- Multilingual vs. dialect/accent variations

Basic ASR Formulation (Bayes Method)



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

$$= \arg\max_{W} \frac{P(X \mid W)P(W)}{P(X)}$$

$$= \arg\max_{W} P_A(X \mid W)P_L(W)$$

Step 3 (arg max) Step 1 ($P_A(X \mid W)$) Step 2 ($P_L(W)$)

Steps in Speech Recognition

Step 1- acoustic modeling: assign probabilities to acoustic realizations of a sequence of words. compute $P_A(X|W)$ using hidden Markov models of acoustic signals and words

Step 2- language modeling: assign probabilities to sequences of words in the language. Train $P_L(W)$ from generic text or from transcriptions of task-specific dialogues

Step 3- hypothesis search: find the word sequence with the maximum a posteriori probability. Search through all possible word sequences to determine arg max over W

Step 1-The Acoustic Model

- We build acoustic models by learning statistics of the acoustic features, X, from a training set where we compute the variability of the acoustic features during the production of the sounds represented by the models
- It is impractical to create a separate acoustic model, $P_A(X \mid W)$, for every possible word in the language--it requires too much training data for words in every possible context
- Instead we build acoustic-phonetic models for the ~50 phonemes in the English language and construct the model for a word by concantenating (stringing together sequentially) the models for the constituent phones in the word
- We build sentences (sequences of words) by concatenating word models

Step 2-The Language Model

- The **language model** describes the probability of a sequence of words that form a valid sentence in the language
- A simple **statistical method** works well based on a Markovian assumption, namely that the probability of a word in a sentence is conditioned on **only** the previous N-words, namely an N-gram language model

$$P_L(W) = P_L(w_1, w_2, \ldots, w_k)$$

$$= \prod_{n=1}^{k} P_L(w_n \mid w_{n-1}, w_{n-2}, \ldots, w_{n-N})$$

where $P_L(w_n \mid w_{n-1}, w_{n-2}, \ldots, w_{n-N})$ is estimated by simply counting up the relative frequencies f of N-tuples in a large corpus of text

Step 3-The Search Problem

- The **search** problem is one of searching the space of all valid sound sequences, conditioned on the word grammar, the language syntax, and the task constraints, to find the word sequence with the maximum likelihood
- The **size of the search space** can be astronomically large and take inordinate amounts of computing power to solve by heuristic methods
- The use of methods from the field of **Finite State Automata Theory** provide **Finite State Networks (FSN)** that reduce the computational burden by orders of magnitude, thereby enabling exact solutions in computationally feasible times, for large ASR problems

Speech Recognition Processes (I)

- **Choose the task** => sounds, word vocabulary, task syntax (grammar), task semantics

- **Example**: isolated digit recognition task
  - Sounds to be recognized—whole words
  - Word vocabulary—zero, oh, one, two, …, nine
  - Task syntax—any digit is allowed
  - Task semantics—sequence of isolated digits must form a valid telephone number

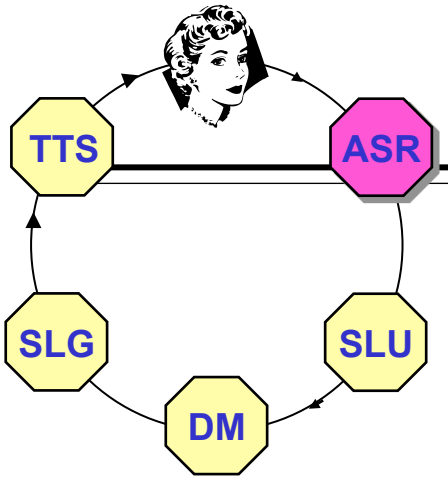
Speech Recognition Processes (II)

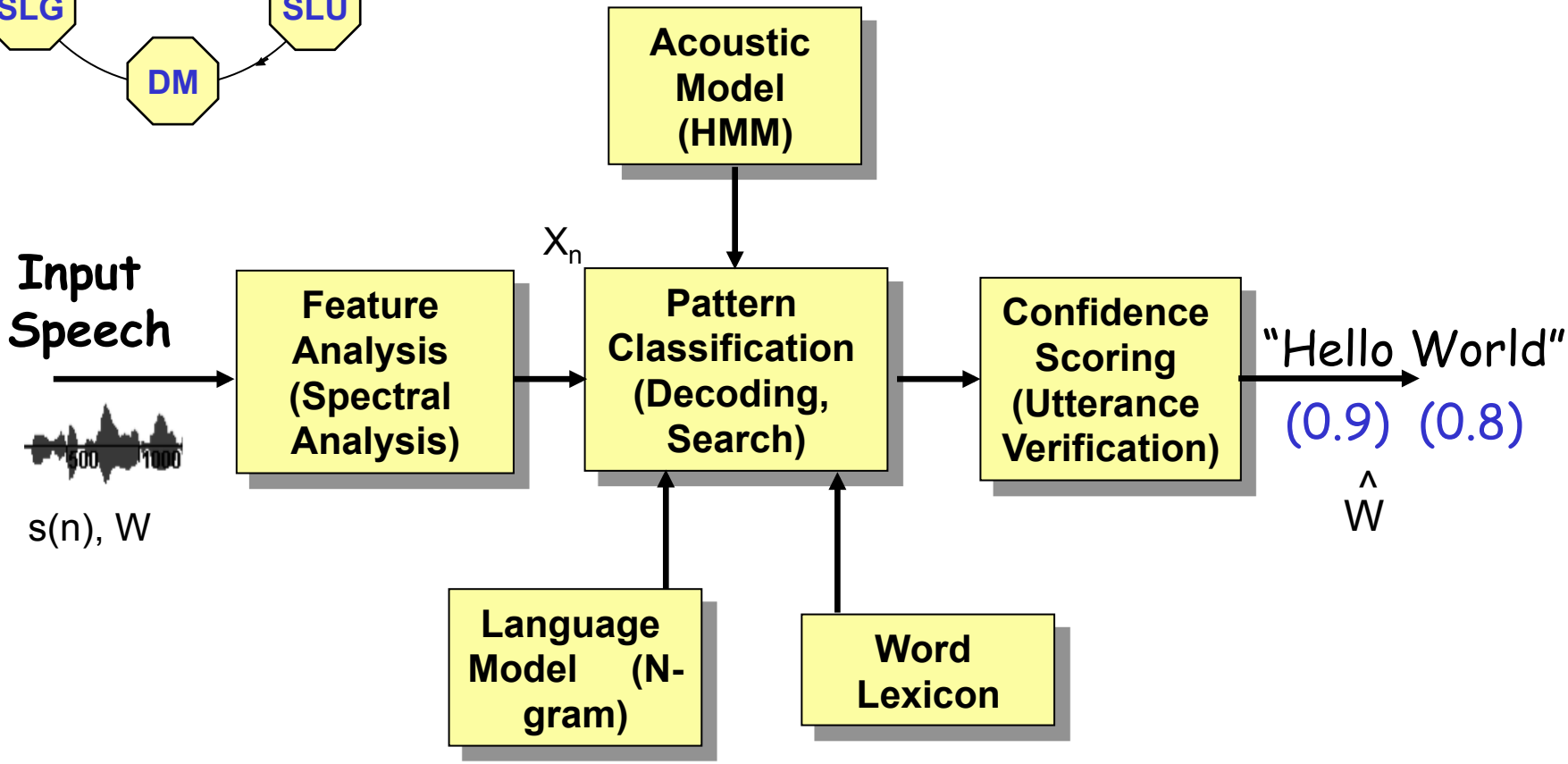
- **Train the Models** => create a method for building **acoustic word models** from a speech training data set, a **word lexicon**, a **word grammar** (language model), and a **task grammar** from a text training data set
  - Speech training set— e.g., 100 people each speaking the 11 digits 10 times in isolation
  - Text data training set— e.g., listing of valid telephone numbers (or equivalently algorithm that generates valid telephone numbers)

Speech Recognition Processes (III)

- **Evaluate performance** => determine word error rate, task error rate for recognizer
  - Speech testing data set—25 new people each speaking 10 telephone numbers as sequences of isolated digits
  - Evaluate digit error rate, phone number error rate
- **Testing algorithm** => method for evaluating recognizer performance from the testing set of speech utterances

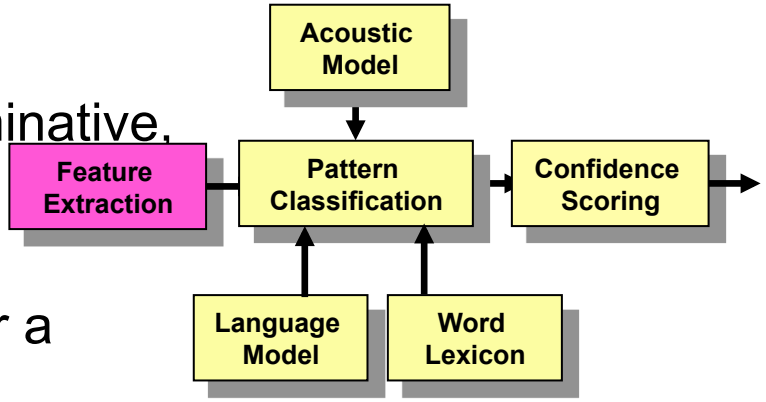
Speech Recognition Process





Feature Extraction

Goal: extract a representation for speech sounds. Ideally, speech features are discriminative, robust, and parsimonious



Method: **spectral analysis** through either a bank-of-filters or through LPC followed by non-linearity and normalization (cepstrum).

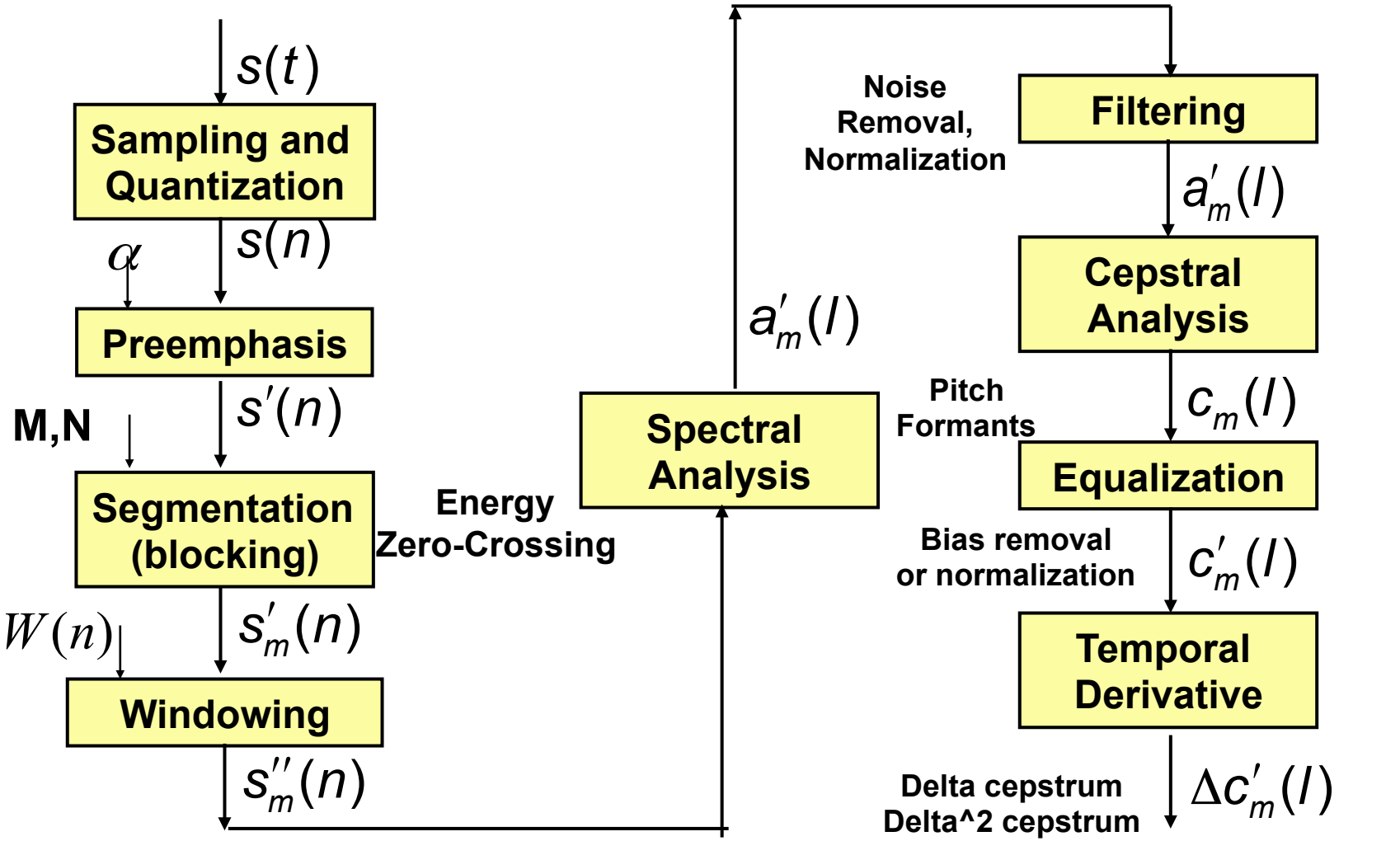
Result: **signal compression** where for each window of speech samples where 30 or so cepstral features are extracted (64,000 b/s -> 5,200 b/s).

Challenges: **robustness** to environment (office, airport, car), devices (speakerphones, cellphones), speakers (acents, dialect, style, speaking defects), noise and echo. **Feature set** for recognition—cepstral features or those from a high dimensionality space.

What Features to Use?

- **Short-time Spectral Analysis**:
  Acoustic features:
    - Mel cepstrum (LPC, filterbank, wavelets), energy
    - Formant frequencies, pitch, prosody
  Acoustic-Phonetic features:
    - Manner of articulation (e.g., stop, nasal, voiced)
    - Place of articulation (e.g., labial, dental, velar)
  Articulatory features:
    - Tongue position, jaw, lips, velum
  Auditory features:
    - Ensemble interval histogram (EIH), synchrony
- **Temporal Analysis**: approximation of the velocity and acceleration typically through first and second order central differences.

Feature Extraction Process



ASR: Statistical Methodology

- We assume that the speech features are generated according to the probability models of the underlying linguistic units
- During the training phase, the model parameters are learned from labeled data (features).
- During the testing phase, the learned models are used to find the hypothesis with the maximum a posteriori (MAP) probability given speech features

ASR Solution

$$\hat{W} = \arg\max_{W \in \Gamma} \bar{P}_{\Omega}(W) \cdot \bar{p}_{\Lambda}(X \mid W)$$

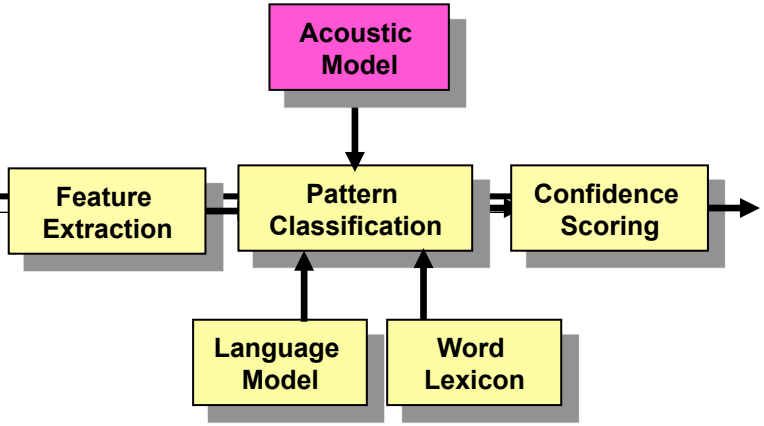
$\bar{p}_{\Lambda}(X \mid W)$ — Acoustic Model (AM): gives the probability of generating feature *X* when *W* is uttered

- Need a model for every *W* to model all speech features from *W* à HMM is an ideal model for speech units
  - Sub-word unit is more flexible (better)

$\bar{P}_{\Omega}(W)$ — Language Model (LM): gives the probability of *W* (word, phrase, sentence) is chosen to say.

- Need a flexible model to calculate the probability for all kinds of *W* à Markov chain model (*n*-gram)

Acoustic Model



- **Goal:** map acoustic features into distinct phonetic labels (e.g., /s/, /aa/) or word labels.

- **Hidden Markov Model (HMM):** statistical method for characterizing the spectral properties of speech by a parametric random process. A collection of HMMs is associated with a phone or a word. HMMs are also assigned for modeling extraneous events.

- **Advantages:** powerful statistical method for dealing with a wide range of data and reliably recognizing speech.

- **Challenges:** understanding the role of classification models (ML Training) versus discriminative models (MMI training). What comes after the HMM—are there data driven models that work better for some or all vocabularies.

Markov Models

- A Markov model consists of a set of states, an initial distribution, and a transition probability matrix

- Two model assumptions
  - The probability of state S_t given S_{t-1} is independent of any state prior to (t -1)

$$p(S_t \mid S_{t-1}, S_{t-2}, \ldots, S_1) = p(S_t \mid S_{t-1})$$

  - The transition probability does not vary with t

$$p(S_t = j \mid S_{t-1} = i) = a_{ij} \quad \text{for all } t$$

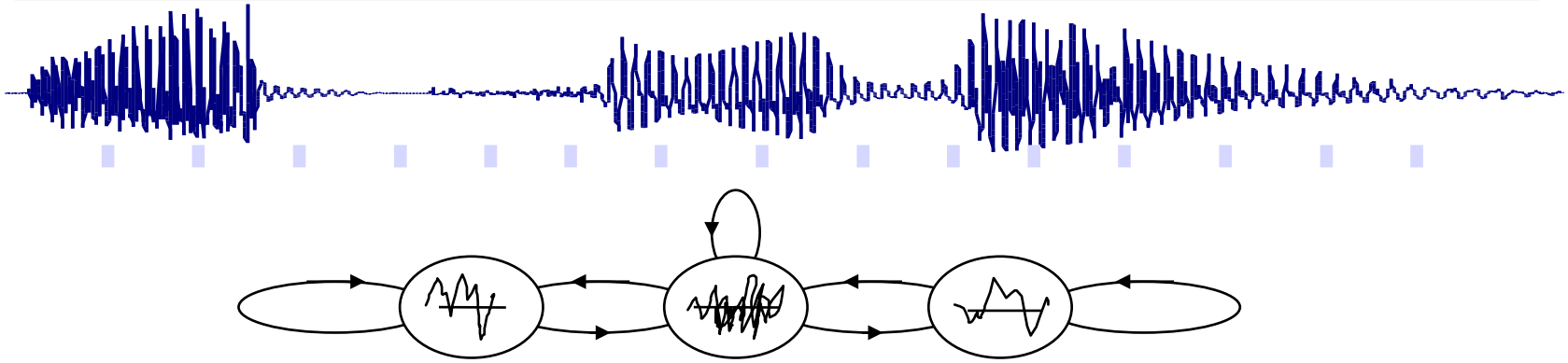
Hidden Markov Models (HMMs)

- In an HMM, the state identities are hidden and the observed sequences depends probabilistically on the state sequence
- In addition to the components required in a Markov model, in HMM there are the observation likelihoods, denoted by $\boldsymbol{b}_i(\boldsymbol{X}_t)$, representing the probability of observing X_t when the state $S_t = i$

Coin-toss Models

- Suppose there are a number of coins, each with its own bias
- One of the coins, coin S_t, is randomly selected. The selection probability is dependent on the identity of the previous coin, S_{t-1}
- Coin S_t is tossed and the outcome (head or tail) X_t is recorded, but not the coin

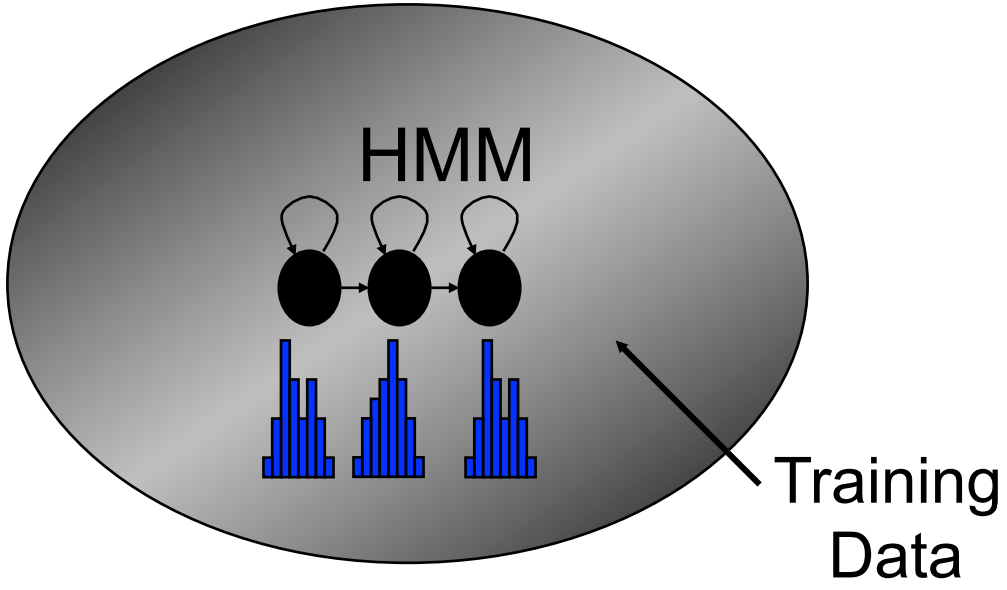
HMM: An Ideal Speech Model

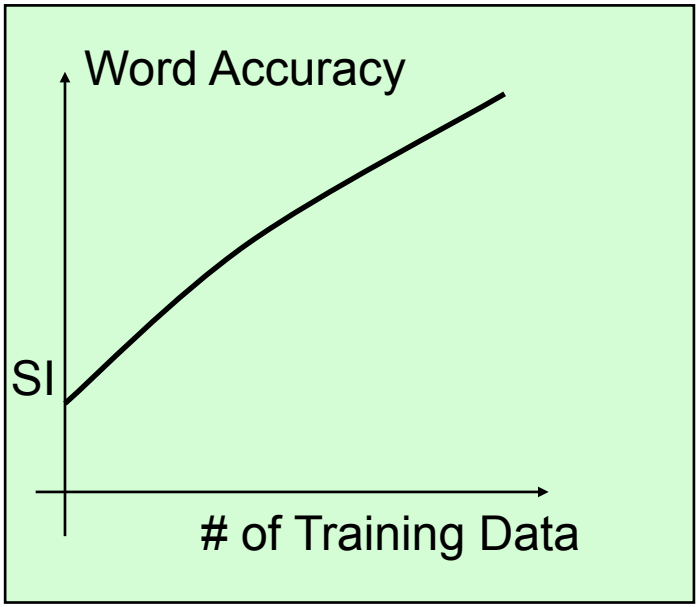


- Variations in speech signals: temporal & spectral
- Each state represents a process of measurable observations
- Inter-process transition is governed by a finite state Markov chain
- Processes are stochastic and individual observations do not immediately identify the hidden state.

HMM models spectral and temporal variations simultaneously

Acoustic Modeling of Speech Units and System Performance



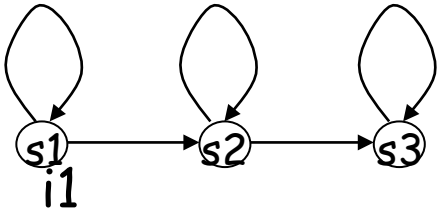


In a typical system, each phoneme in the language is modeled by a 3-state left-to-right continuous density HMM (CDHMM), and background noise is modeled by a 1-state CDHMM

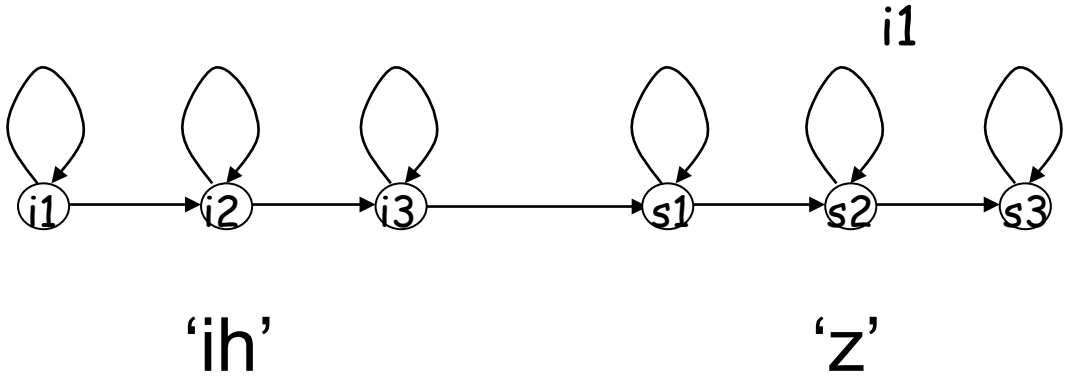
Up to thousand of hours of speech data have been used to train HMM's

HMM for Speech

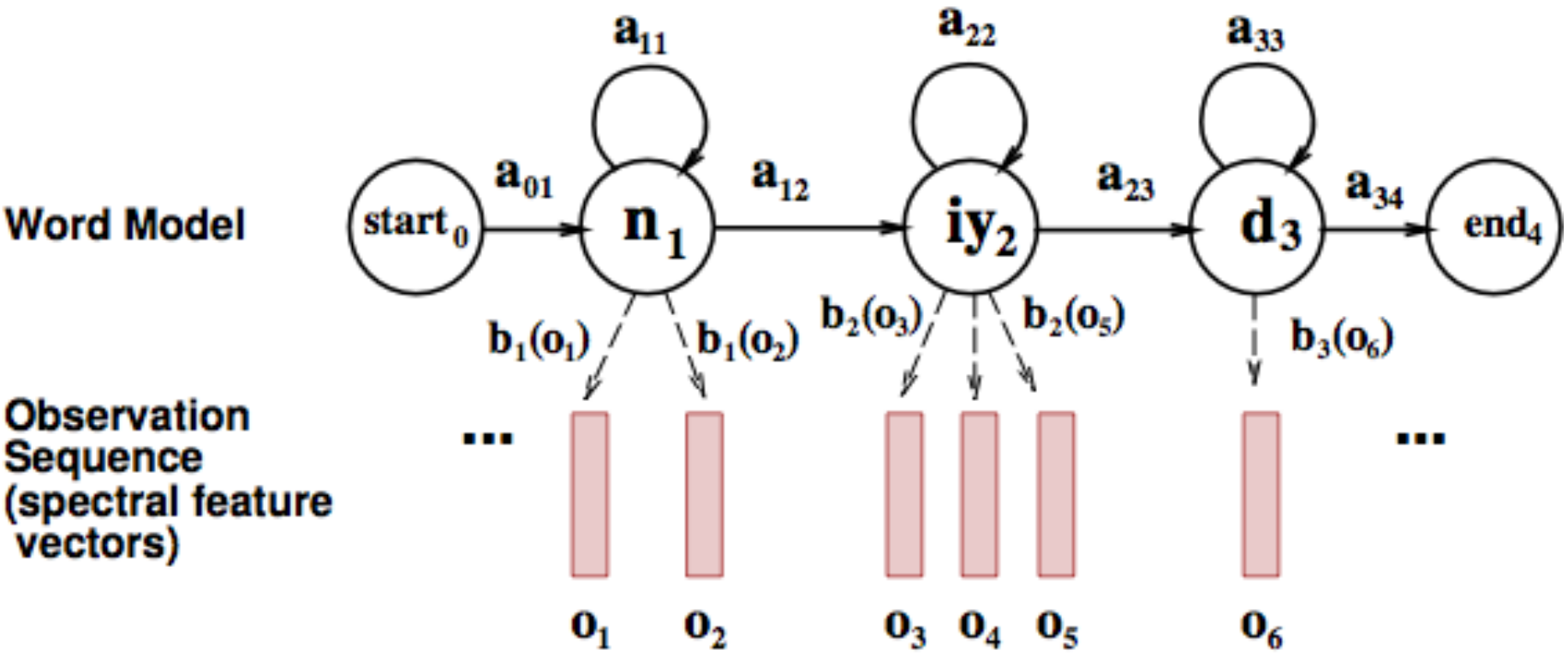
- Phone model : ‘s’



- Word model: ‘is’ (ih-z)



Isolated Word HMM



Left-right HMM – highly constrained state sequences

Basic Problems in HMMs

- Given acoustic observation *X* and model Φ*:*

Evaluation: Given the observation *X* and the model parameters Φ, *P*(*X* | Φ)

Decoding: choose optimal state sequence, that is, given the observation *X and* Φ, determine the optimal state sequence S*

$$S^* = \arg\max_S p(X, S \mid \Phi)$$

Re-estimation: adjust Φ to maximize *P*(*X*|Φ)

Evaluation: Forward-Backward Algorithm

- Denote the parameters in HMM by
  - *The initial probability* $\pi_i = a_{1i}$
  - *The transition probability* a_{ij}
  - *The observation probability* $b_i(X_t)$

Evaluation: Forward-Backward Algorithm

- Define the forward probability α as

$$\alpha_i(t) = p(X_1,\ldots,X_t,S_t = i)$$

$$Then$$

$$\alpha_j(1) = a_{1j}b_j(X_1)$$

$$\alpha_j(t) = \sum_{i=2}^{N-1}\alpha_i(t-1)a_{ij}b_j(X_t),$$

$$\alpha_N(T) = \sum_{i=2}^{N-1}\alpha_i(T)a_{iN}.$$

Evaluation: Forward-Backward Algorithm

- Define the backward probability β as

$$\beta_i(t) = p(X_{t+1},\ldots,X_T \mid S_t = i)$$

$$Then$$

$$\beta_i(T) = a_{1N}$$

$$\beta_i(t) = \sum_{j=2}^{N-1} a_{ij} b_j(X_{t+1}) \beta_j(t+1),$$

$$\beta_1(1) = \sum_{j=2}^{N-1} a_{1j} b_j(X_1) \beta_j(1)$$

Evaluation: Data Likelihood

- The joint probability of S_t=j and X is

$$p(X_1^T, S_t = j) = \alpha_j(t)\beta_j(t)$$

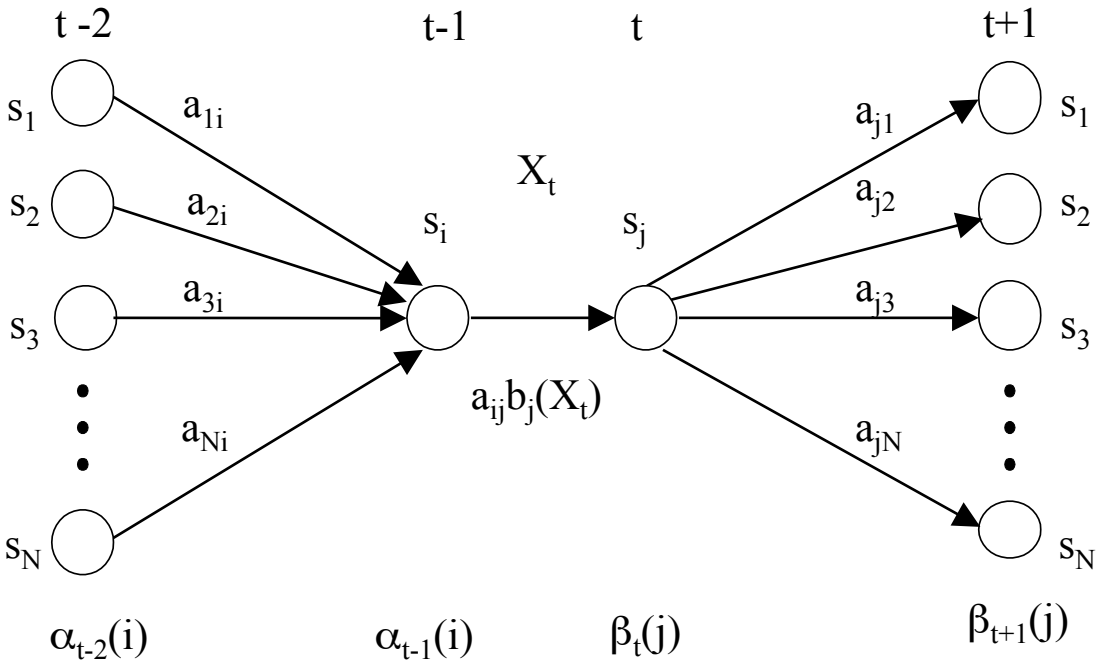
The data likelihood is

$$p(X_1^T) = \sum_j p(X_1^T, S_t = j) = \sum_j \alpha_j(t)\beta_j(t)$$

Evaluation: The Baum-Welch Algorithm

$$\alpha_t(i) = P(X_1^t, s_t = i \mid \Phi) = \left[\sum_{j=1}^{N} \alpha_{t-1}(j) a_{ji}\right] b_i(X_t) \qquad 2 \le t \le T;\ \ 1 \le i \le N$$

$$\beta_t(j) = P(X_{t+1}^T \mid s_t = j, \Phi) = \left[\sum_{i=1}^{N} a_{ji} b_i(X_{t+1}) \beta_{t+1}(i)\right] \qquad t=T-1\ldots1;\ \ 1 \le j \le N$$

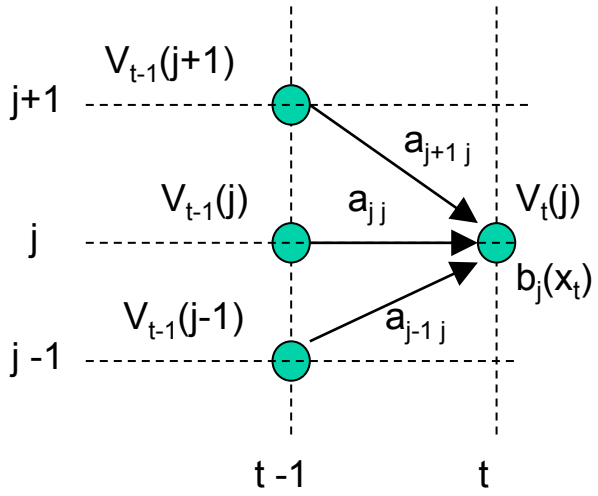


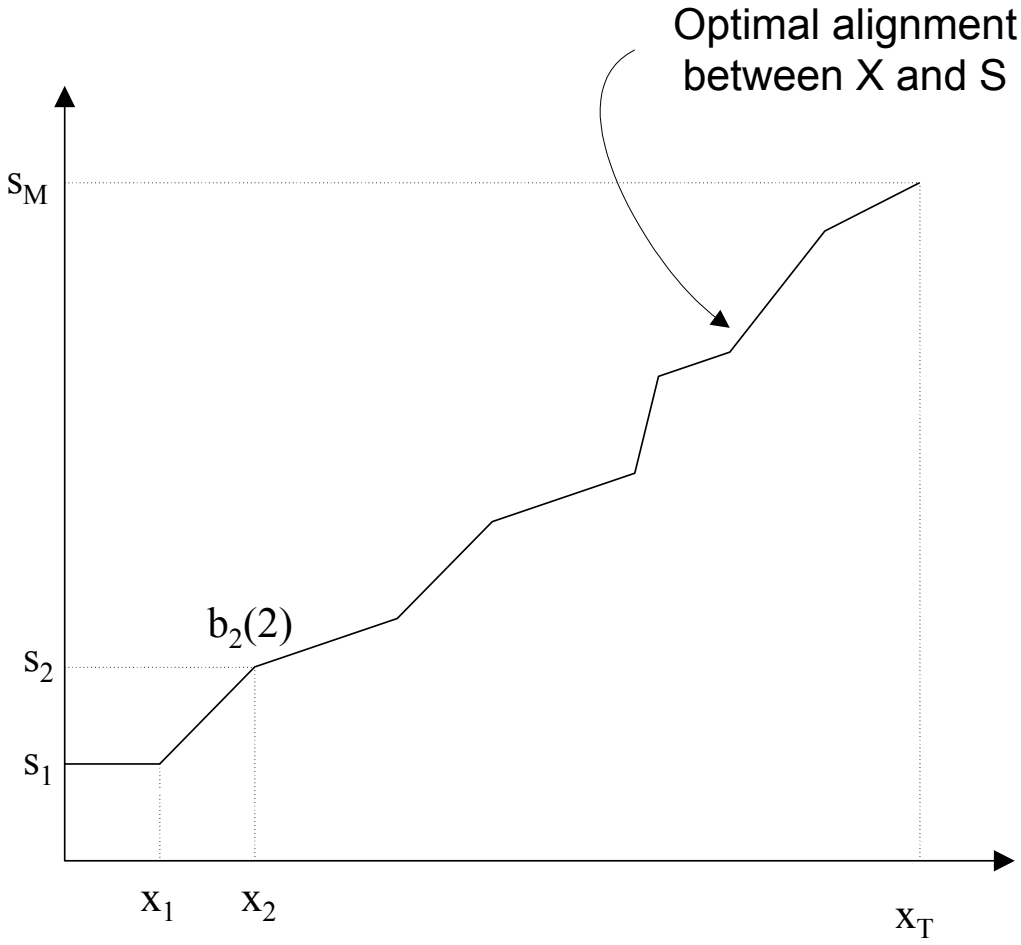
$$\gamma_t(i,j) = P(s_{t-1} = i, s_t = j \mid X_1^T, \Phi)$$

$$= \frac{P(s_{t-1} = i, s_t = j, X_1^T \mid \Phi)}{P(X_1^T \mid \Phi)}$$

$$= \frac{\alpha_{t-1}(i) a_{ij} b_j(X_t) \beta_t(j)}{\sum_{k=1}^{N} \alpha_T(k)}$$

Decoding: Viterbi Algorithm





Reestimation: Training Algorithm

- Find model parameters Φ=(A, B, π) that maximize the probability p(X| Φ) of observing some data X:

$$P(\mathbf{X} \mid \Phi) = \sum_{\mathbf{S}} P(\mathbf{X}, \mathbf{S} \mid \Phi) = \sum_{\mathbf{S}} \prod_{t=1}^{T} a_{s_{t-1} s_t} b_{s_t}(X_t)$$

- There is no closed-form solution and thus we use the **EM algorithm**:

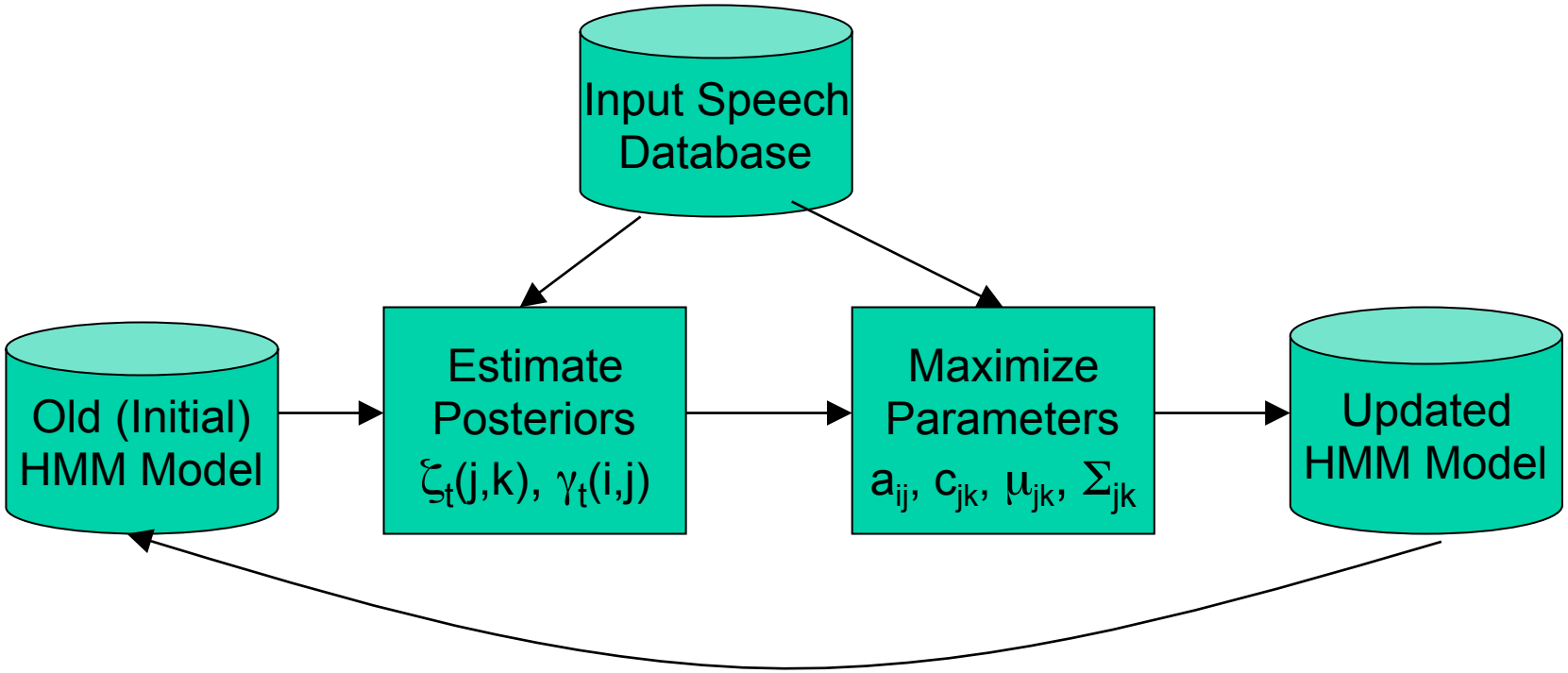
  given Φ, we compute model parameters $\hat{\Phi}$ that maximize the following auxiliary function Q

$$Q(\Phi, \hat{\Phi}) = \sum_{\text{all } \mathbf{S}} \frac{P(\mathbf{X}, \mathbf{S} \mid \Phi)}{P(\mathbf{X} \mid \Phi)} \log P(\mathbf{X}, \mathbf{S} \mid \hat{\Phi}) = Q_{\mathbf{a}_i}(\Phi, \hat{\mathbf{a}}_i) + Q_{\mathbf{b}_j}(\Phi, \hat{\mathbf{b}}_j)$$

  it is guaranteed to have increased (or the same) likelihood

Iterative Training Procedure

- Several iterations of the process:



Continuous Observation Density HMM's

Most general form of pdf with a valid re-estimation procedure is:

$$b_j(x) = \sum_{m=1}^{M} c_{jm} \bullet \left[x, \mu_{jm}, U_{jm} \right], \quad 1 \le j \le N$$

x = observation vector=$\{x_1, x_2, \ldots, x_D\}$

M = number of mixture densities

c_{jm} = gain of m-th mixture in state j

$\bullet$ = any log-concave or elliptically symmetric density (e.g., a Gaussian)

μ_{jm} = mean vector for mixture m, state j

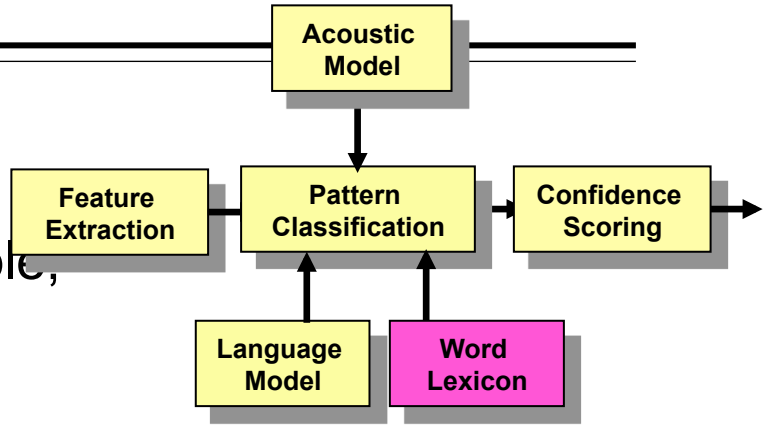
U_{jm} = covariance matrix for mixture m, state j

$$c_{jm} \ge 0, \quad 1 \le j \le N, \quad 1 \le m \le M$$

$$\sum_{m=1}^{M} c_{jm} = 1, \quad 1 \le j \le N$$

$$\int_{-\infty}^{\infty} b_j(x) dx = 1, \quad 1 \le j \le N$$

Word Lexicon



Goal:

Map legal phone sequences into words according to **phonotactic** rules. For example,

David /d/ /ey/ /v/ /ih/ /d/

Multiple Pronunciation:

Several words may have multiple pronunciations. For example

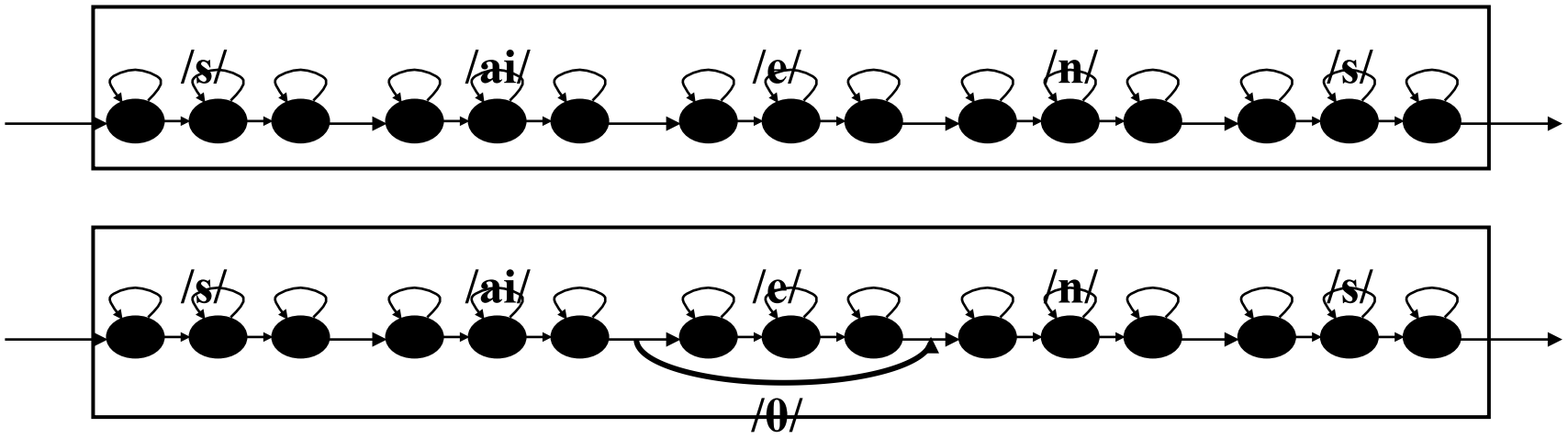
Data /d/ /ae/ /t/ /ax/
Data /d/ /ey/ /t/ /ax/

Challenges:

How do you generate a word lexicon automatically; how do you add new variant dialects and word pronunciations.

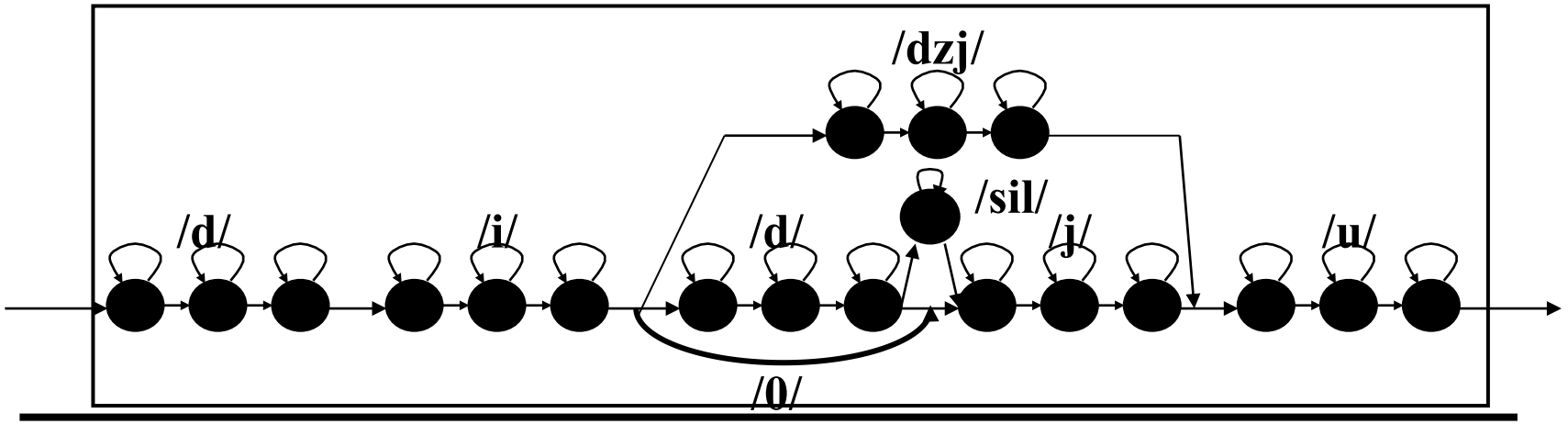
Lexical Modeling

- Assume each HMM → a monophone model
  American English: 42 monophone → 42 HMMs
  - Concatenation of phone models (phone HMM's)
  - Lexicon: /science/ = /s/+/ai/+/e/+/n/+/s/ or /s/+/ai/+/n/+/s/
  - Multiple pronunciations and pronunciation network



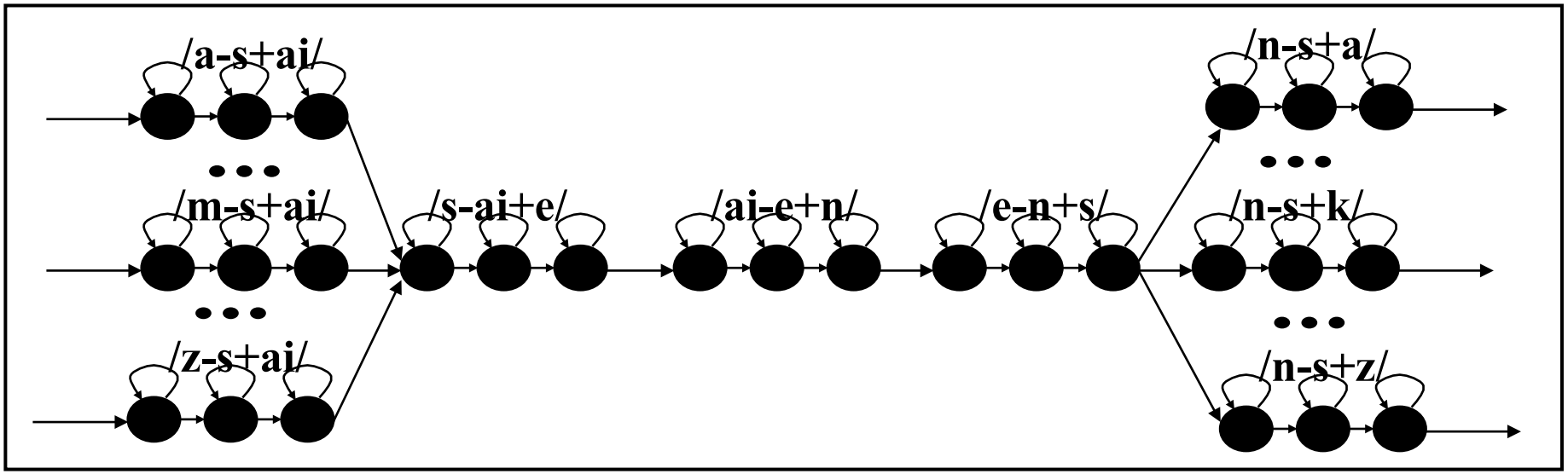
Word-Juncture Modeling

- Co-articulation Effect
  - Simple concatenation of word models (word HMM's)
  - Hard change: "did you" = /d/+/i/+/dzj/+/u/
  - Soft change: possible pronunciation variations
  - Source of major errors in many ASR systems
  - Easier to handle in syllabic languages with open syllables (vowel or nasal endings, e.g. Japanese, Mandarin)

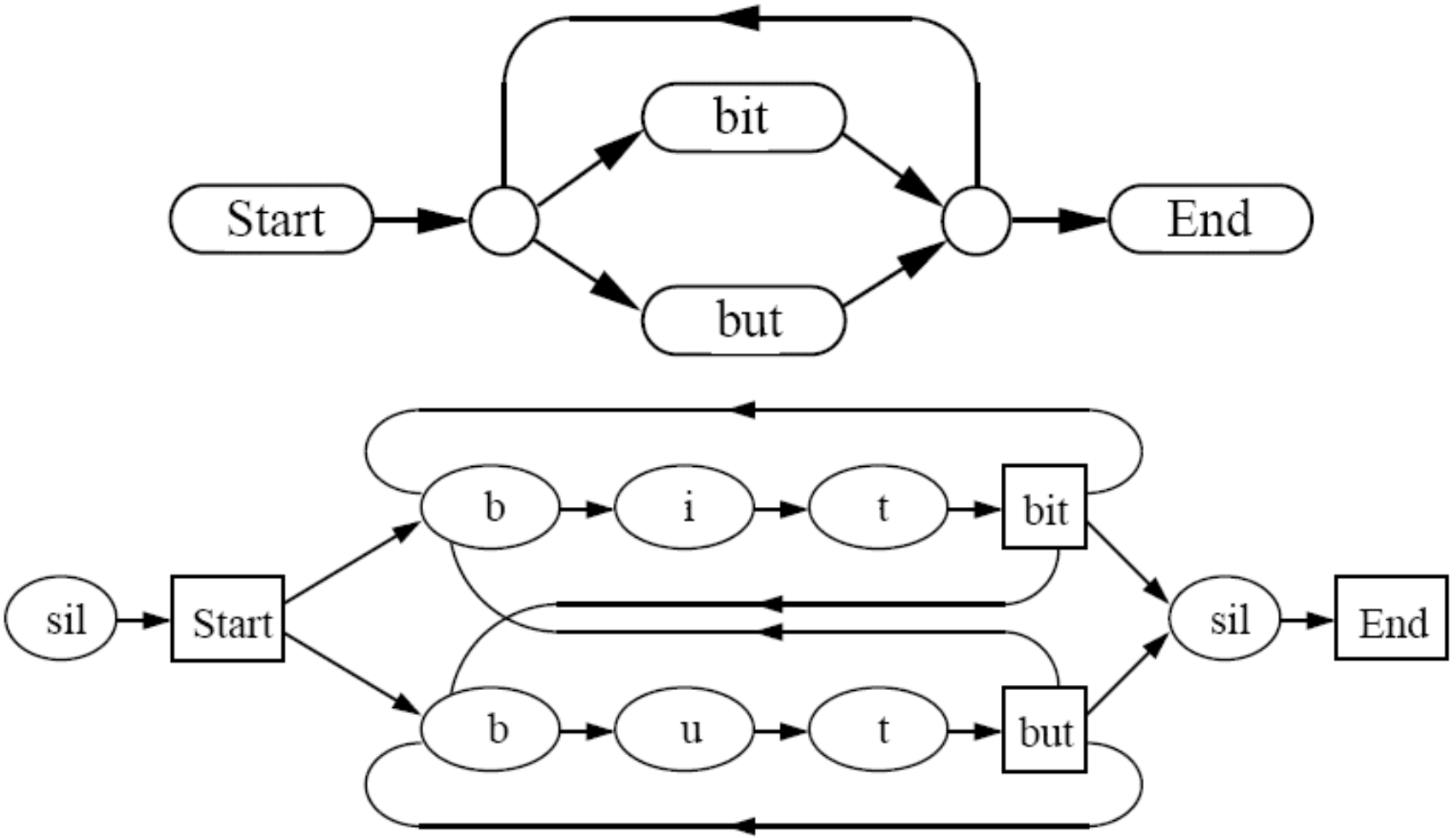


Modeling Triphone

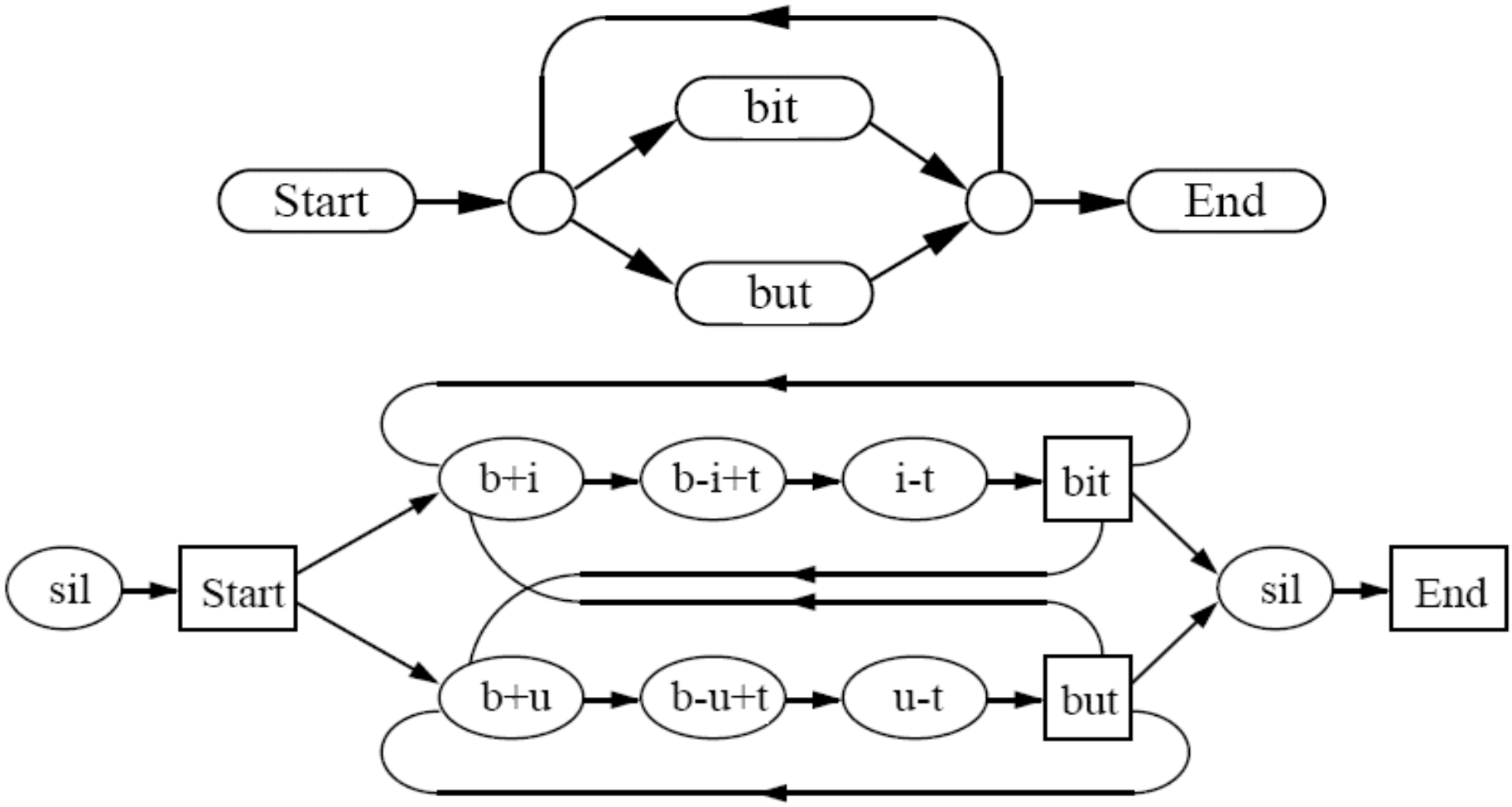
- Monophone modeling is too simple to model coarticulation phenomenon ubiquitous in speech
- Modeling context-dependent phonemes: triphone
  - American English: 42X42X42 triphones → 74,088 HMMs



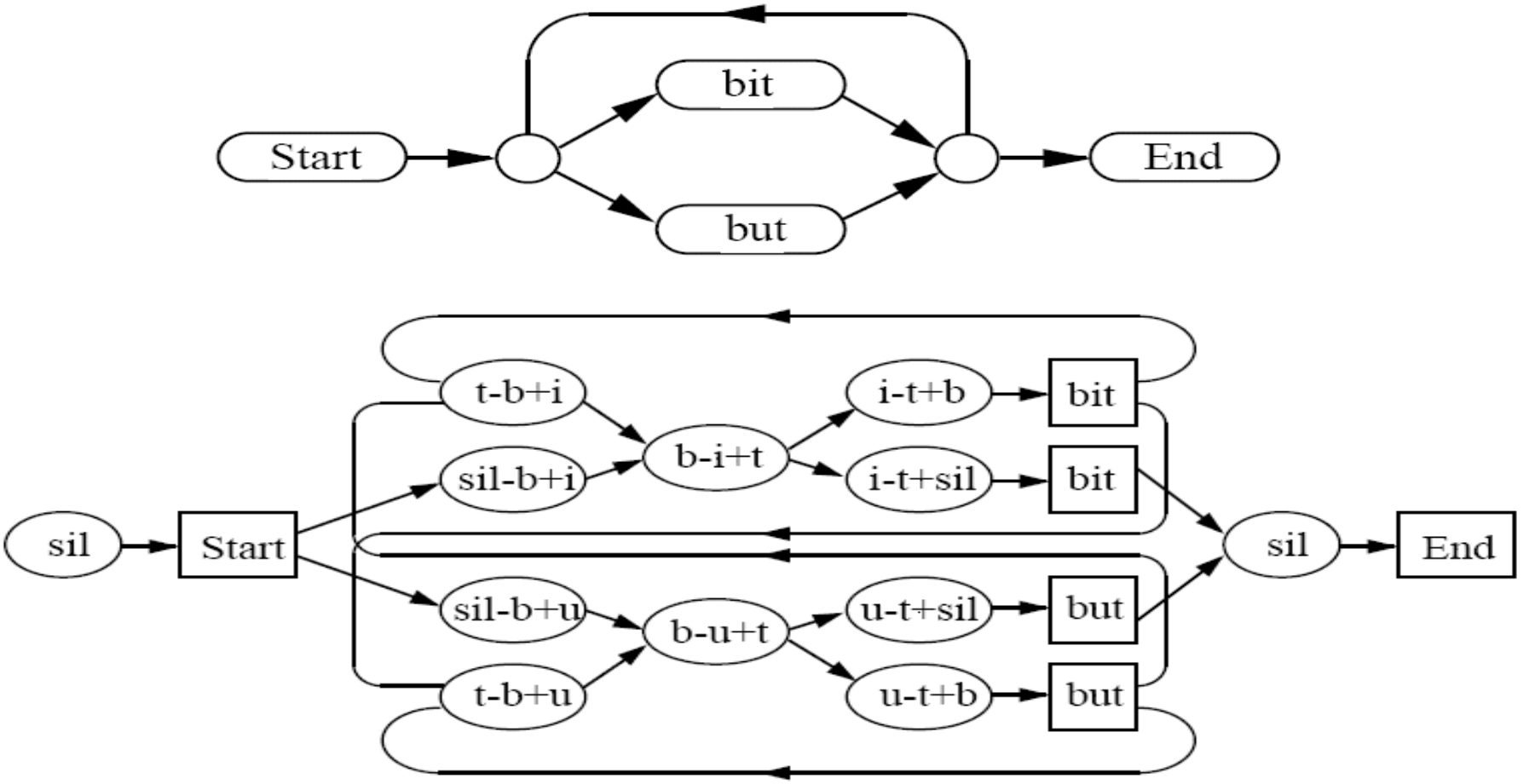
Grammar Network Expansion: Monophones



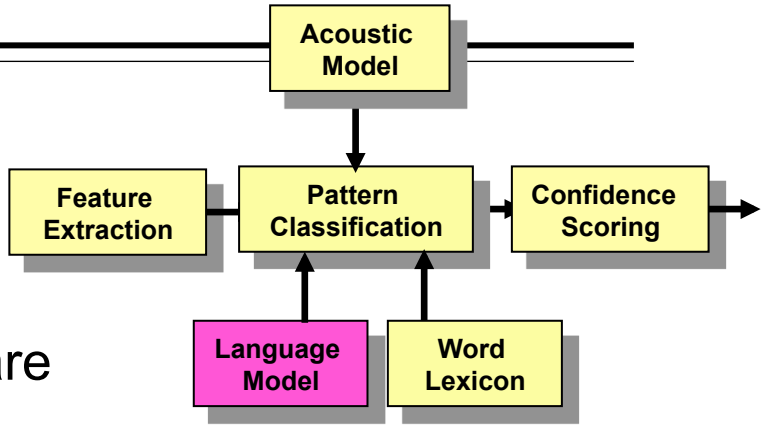
Grammar Network Expansion: Triphones



Grammar Network Expansion: Cross-Word



Language Model



Goal: Model "acceptable" spoken phrases, constrained by task syntax.

Rule-based: Deterministic grammars that are knowledge driven. For example,

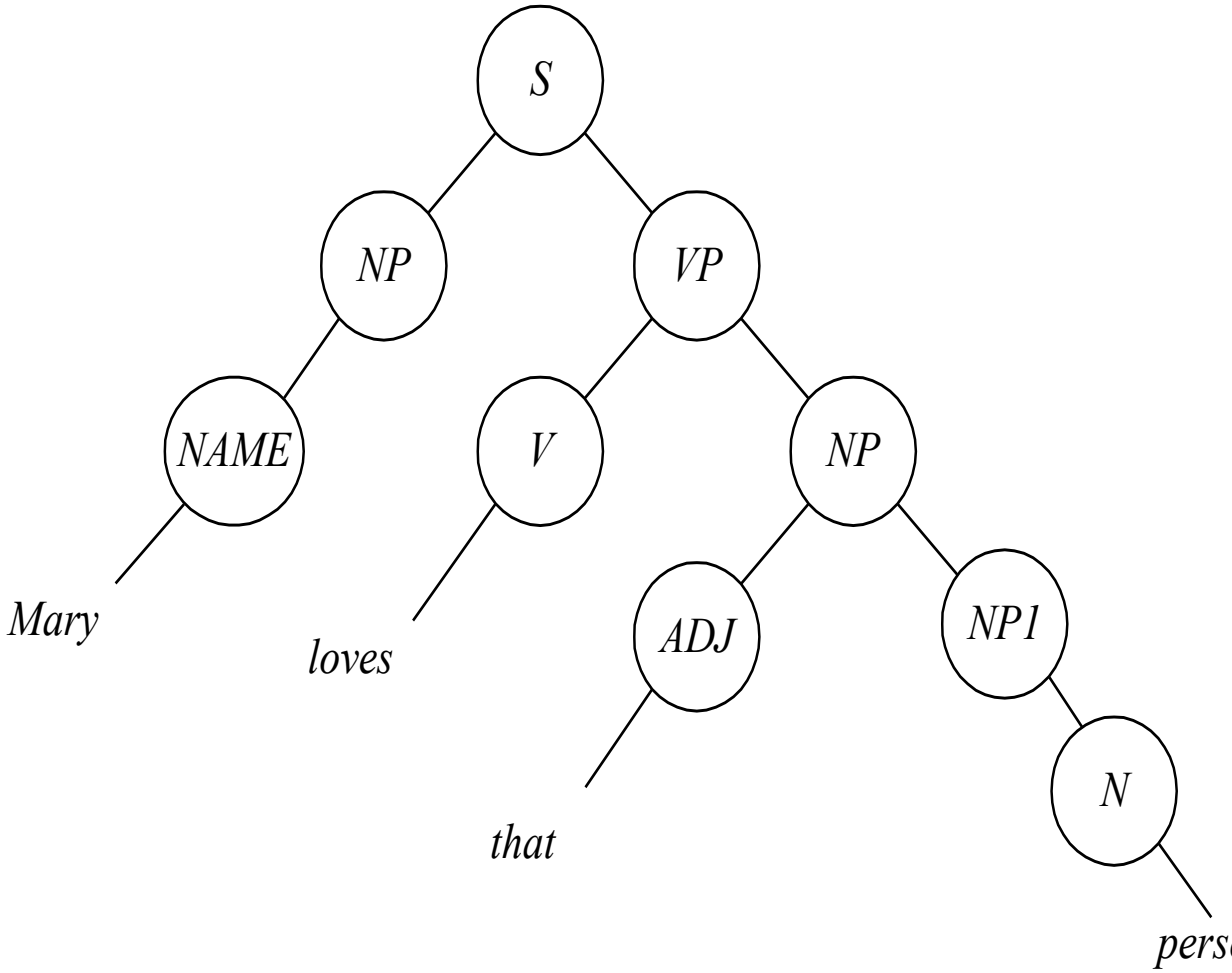
flying from $city to $city on $date

Statistical: Compute estimate of word probabilities (N-gram, class-based, CFG). For example

flying from Newark to Boston tomorrow

Challenges: How do you build a language model rapidly for a new task?

Formal Grammars



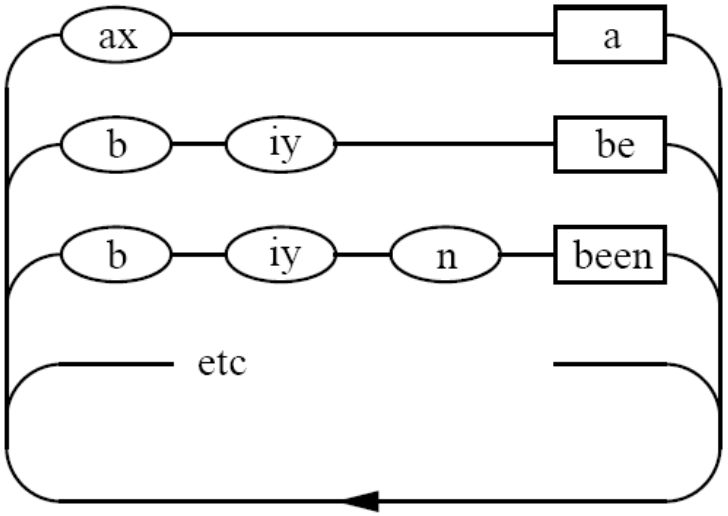
Rewrite Rules:

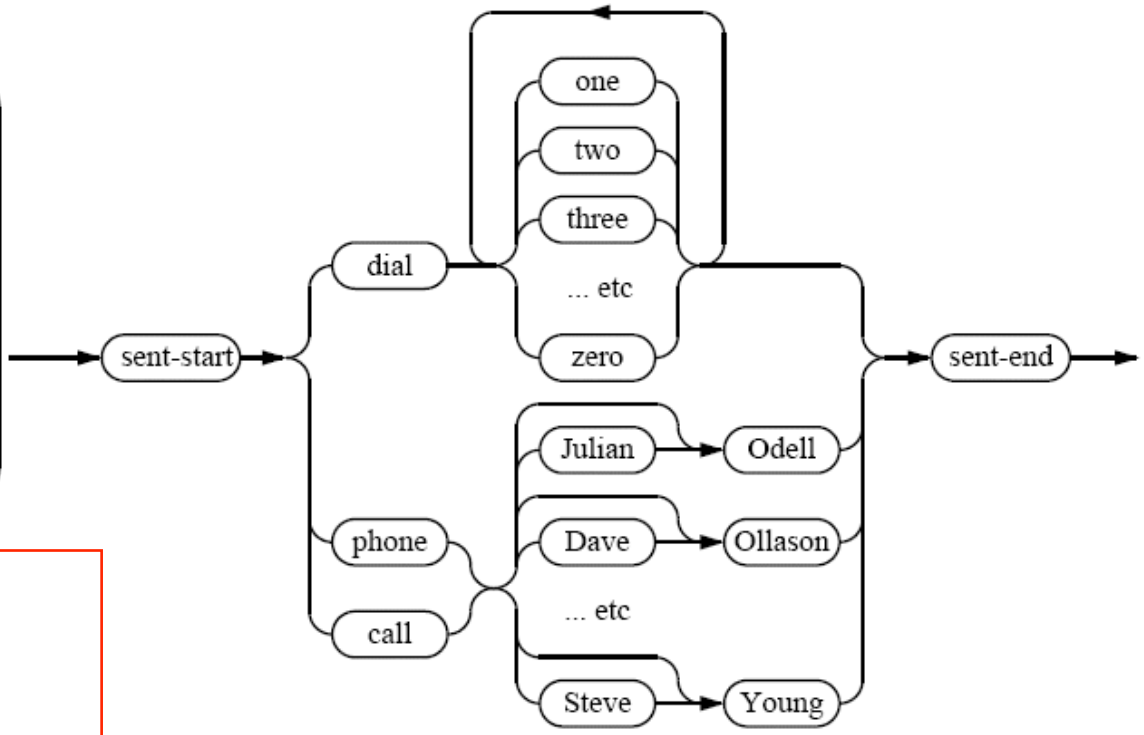
1. $S \rightarrow NP\ VP$
2. $VP \rightarrow V\ NP$
3. $VP \rightarrow AUX\ VP$
4. $NP \rightarrow ART\ NP1$
5. $NP \rightarrow ADJ\ NP1$
6. $NP1 \rightarrow ADJ\ NP1$
7. $NP1 \rightarrow N$
8. $NP \rightarrow NAME$
9. $NP \rightarrow PRON$
10. $NAME \rightarrow Mary$
11. $V \rightarrow loves$
12. $ADJ \rightarrow that$
13. $N \rightarrow person$

Chomsky Grammar Hierarchy

Types	Constraints	Automata
Phase structure grammar	$\alpha \rightarrow \beta$. This is the most general grammar.	Turing machine
Context-sensitive grammar	Subset of $\alpha \rightarrow \beta$ $\lvert\alpha\rvert \leq \lvert\beta\rvert$, where $\lvert . \rvert$ indicates the length of the string.	Linear bounded automata
Context-free grammar (CFG)	$A \rightarrow w$ and $A \rightarrow BC$, where w is a terminal and B, C are non-terminals.	Push down automata
Regular grammar	$A \rightarrow w$ and $A \rightarrow wB$	Finite-state automata

Other Examples of Grammar Network





Word-loop grammar:

- **For all possible sentences.**
- **Each branch represents a word in vocabulary**
- **May add transition probabilities from language models**

Grammar for Voice Dialing

N-Grams

$$P(\mathbf{W}) = P(w_1, w_2, \ldots, w_N)$$
$$= P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1, w_2)\cdots P(w_N \mid w_1, w_2, \ldots, w_{N-1})$$
$$= \prod_{i=1}^{N} P(w_i \mid w_1, w_2, \ldots, w_{i-1})$$

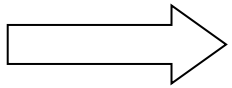
Trigram Estimation

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

Example: bi-grams

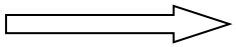
(Probability of a word given the previous word)

I want to go from Boston to Denver tomorrow
I want to go to Denver tomorrow from Boston
Tomorrow I want to go from Boston to Denver with United
A flight on United going from Boston to Denver tomorrow
A flight on United
Boston to Denver
Going to Denver
United from Boston
Boston with United tomorrow
Tomorrow a flight to Denver
Going to Denver tomorrow with United
Boston Denver with United
A flight with United tomorrow
...



I want	0.02
want to	0.01
to go	0.03
go from	0.05
go to	0.01
from Boston	0.01
Boston to	0.02
to Denver	0.01
...	
...	

... 0.03 0.02

I want to go from Boston to Denver  1.2e-12

0.02 0.01 0.05 0.01 0.02

Generalization for *N*-grams (back-off)

0.0 0.0 0.02

I want a flight from Boston to Denver ⟹ 0.0

0.02 0.0 0.01 0.02

BAD!!!

If the bi-gram ab was never observed, we can estimate its probability:

Probability of word b following word a as a fraction of probability of word b

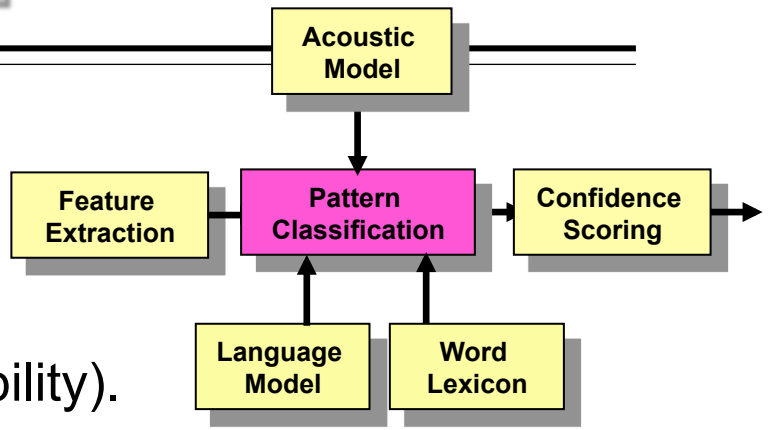
0.001 0.002 0.02

I want a flight from Boston to Denver ⟹ 4.8e-15

0.02 0.003 0.01 0.02

GOOD

Pattern Classification



Goal: Combine information (probabilities) from the acoustic model, language model and word lexicon to generate an "optimal" word sequence (highest probability).

Method: **D**ecoder searches through all possible recognition choices using a Viterbi decoding algorithm.

Challenges: How do we build efficient structures (FSMs) for decoding and searching large vocabulary, complex language models tasks;

- features x HMM units x phones x words x sentences can lead to search networks with 10^{22} states
- FSM methods can compile the network to 10^{8} states—14 orders of magnitude more efficient

what is the theoretical limit of efficiency that can be achieved ?

ASR: Viterbi Search

- Assume we build the grammar network for the task, with all trained HMMs attached in the network
- Unknown utterance → a sequence of feature vectors *Y*
- Speech recognition is nothing more than a viterbi search:
  - The whole network viewed as a composite HMM Λ
  - *Y* viewed as input data, find the optimal path (viterbi path) *S** traversing the whole network (from START to END)

$$S^* = \arg\max_{S \in \Theta} \Pr(S) \cdot p(Y, S \mid \Lambda) = \arg\max_{S \in \Theta} \Pr(W_S) \cdot p(Y, S \mid \Lambda)$$

  - Once *S** is found, the recognition results (word sequence) can be derived by backtracking the Viterbi path

ASR Issues

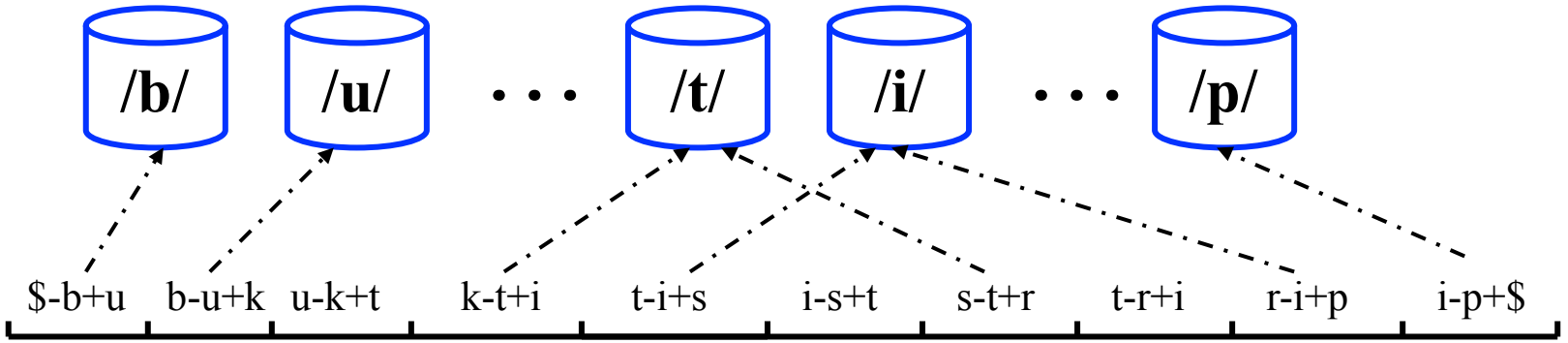
- Training stage:
  - *Acoustic modeling*: how to select speech unit and estimate HMMs reliably and efficiently from available training data.
  - *Language modeling*: how to estimate *n*-gram model from text training data; handle data sparseness problem.
- Test stage:
  - *Search*: given HMM's and *n*-gram model, how to efficiently search the optimal path from the huge grammar network.
    - Search space is extremely large
    - Call for an efficient pruning strategy

Acoustic Modeling

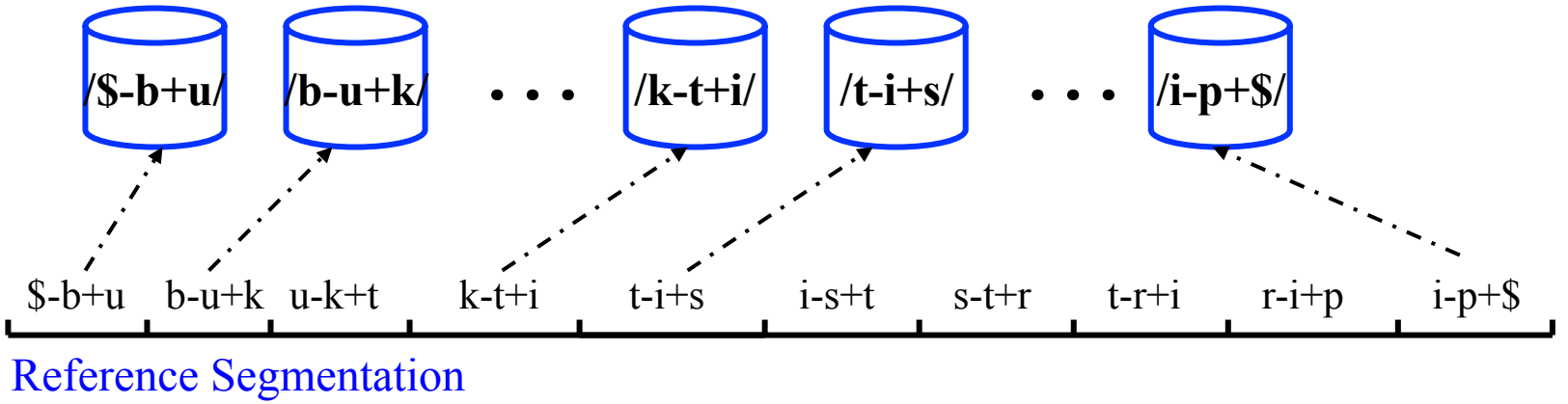
- Selection of Speech Units: modeled by a HMMDigit string recognition: a digit by a HMM ➔ 10-12 HMMs
  - Large vocabulary: monophone→biphone→triphone
- HMM topology selection
  - Phoneme: 3-state left-right without skipping state
  - Digit/word: 6-12 states left-right no state skipping
- HMM type selection
  - Top choice: Gaussian mixture CDHMM
  - Number of Gaussian mixtures in each state (e.g., 16)
- HMM parameters estimation:
  - ML (Baum-Welch algorithm) and others

Selecting Speech Segments for each HMM

Monophone HMMs

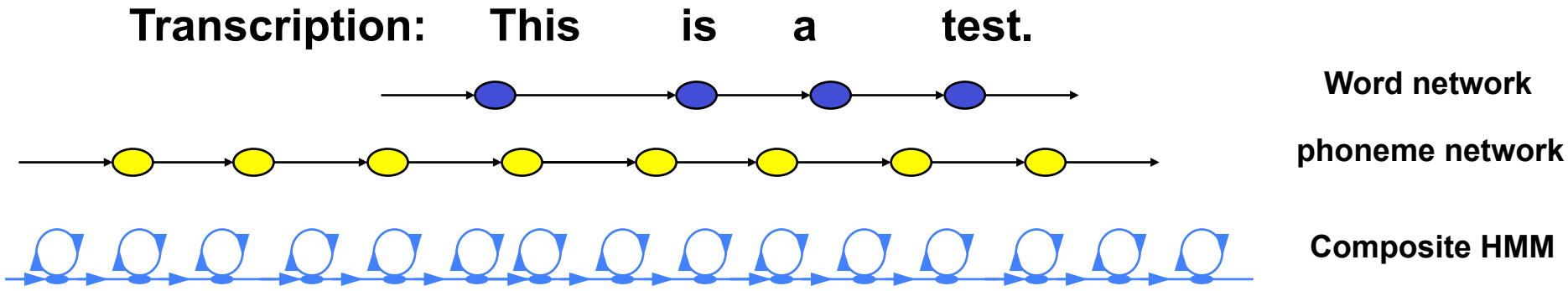


Triphone HMMs



Reference Segmentation

- Where the segmentation information comes from?
  - Human labeling: tedious, time-consuming;
    - Only a small amount is affordable; used for bootstrap.
  - Automatic segmentation if a simple HMM set is available.
    - Forced-alignment: Viterbi algorithm; Need transcription only
    - HMMs + transcription ➔ segmentation information

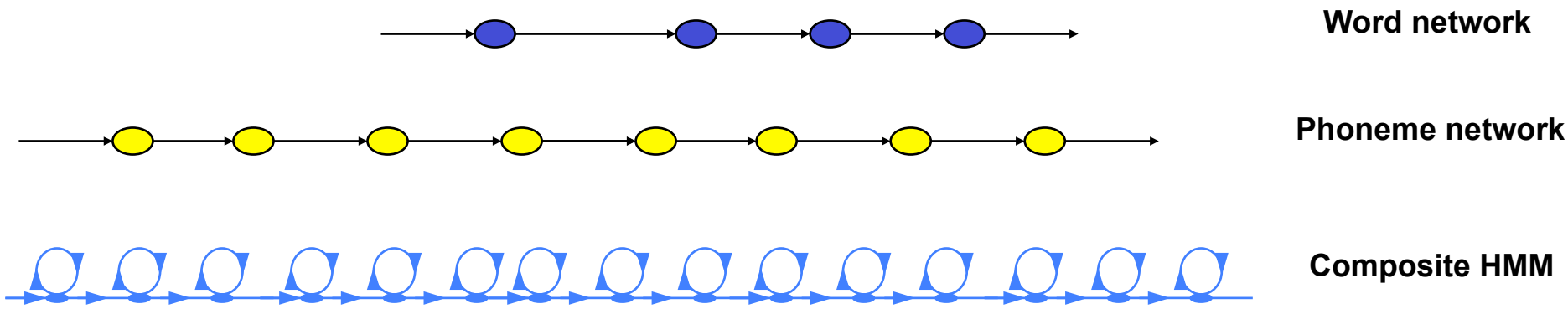


Run the Viterbi algorithm to backtrack segmentation information

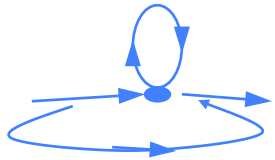
Embedded Training

- Only need transcription for each utterance; no segmentation is needed; automatically tune to optimal segmentation during training

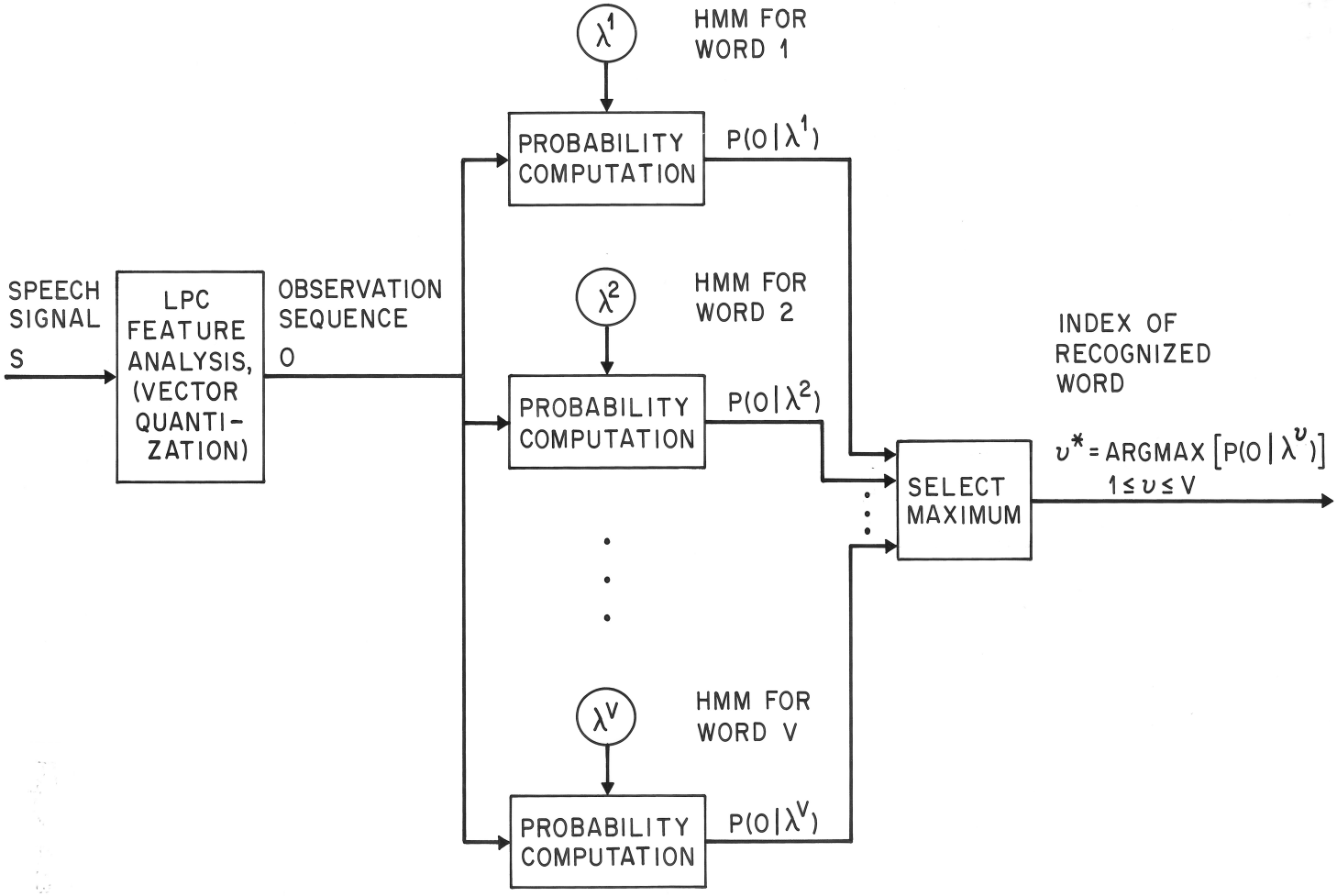
Transcription: **This** **is** **a** **test.**



Add optional 1-state silence models between words



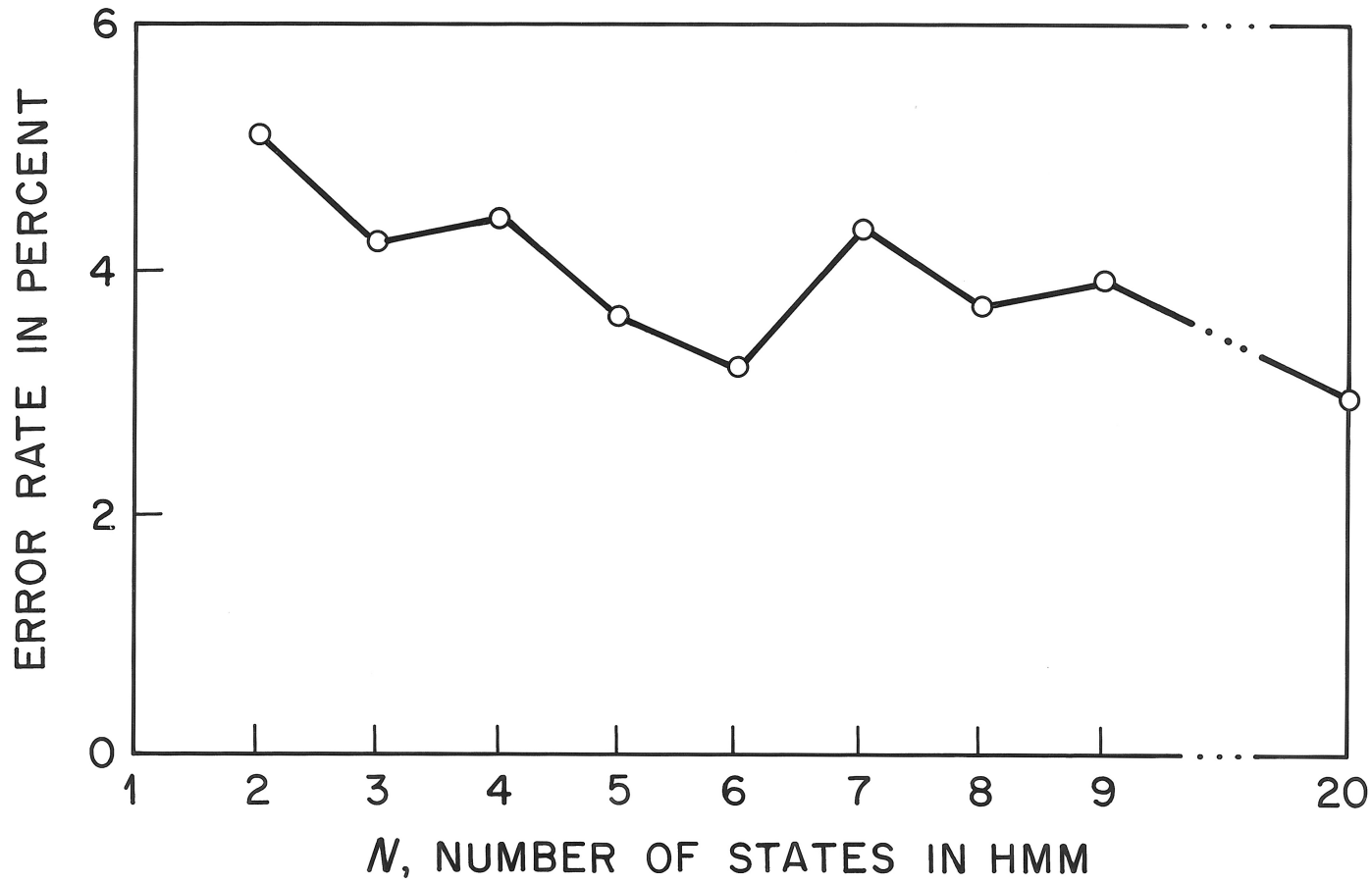
Isolated Word HMM Recognizer



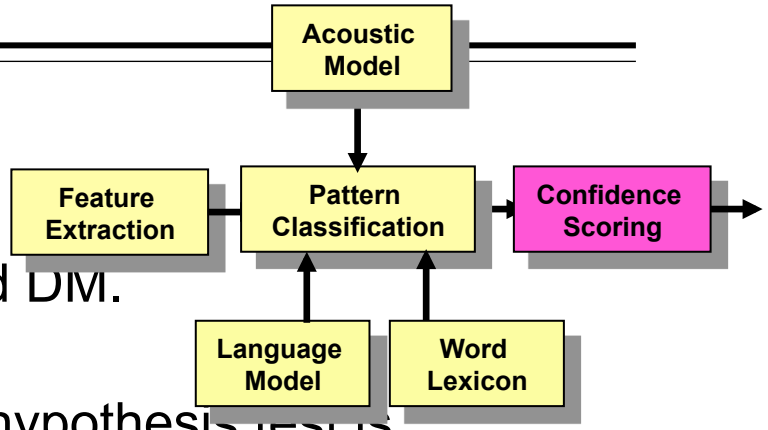
Choice of Model Parameters

- Left-to-right model preferable to ergodic model (speech is a left-right process)
- Number of states in range 2-40 (from sounds to frames)
  - Order of number of distinct sounds in the word
  - Order of average number of observations in word
- Observation vectors
  - Cepstral coefficients (and their time derivatives) derived from LPC (1-9 mixtures), diagonal covariance matrices
  - Vector quantized discrete symbols (16-256 codebook sizes)
- Constraints on $b_j(O)$ densities
  - $bj(k)>\varepsilon$ for discrete densities
  - $C_{jm}>\delta$, $U_{jm}(r,r)>\delta$ for continuous densities

Performance vs. Number of States in Model



Confidence Scoring



Goal: Identify possible recognition errors and out-of-vocabulary events. Potentially improves the performance of ASR, SLU and DM.

Method: A confidence score based on a hypothesis test is associated with
each recognized word. For example:

Label:	credit please
Recognized:	credit fees
Confidence:	(0.9) (0.3)

$$P(\mathbf{W} \mid \mathbf{X}) = \frac{P(\mathbf{W})P(\mathbf{X} \mid \mathbf{W})}{\sum_{\mathbf{W}} P(\mathbf{W})P(\mathbf{X} \mid \mathbf{W})}$$

Challenges: Rejection of extraneous acoustic events (noise, background speech, door slams) without rejection of valid user input speech.

Robustness

Problem:

a mismatch in the speech signal between the training phase and testing phase can result in performance degradation.

Methods:

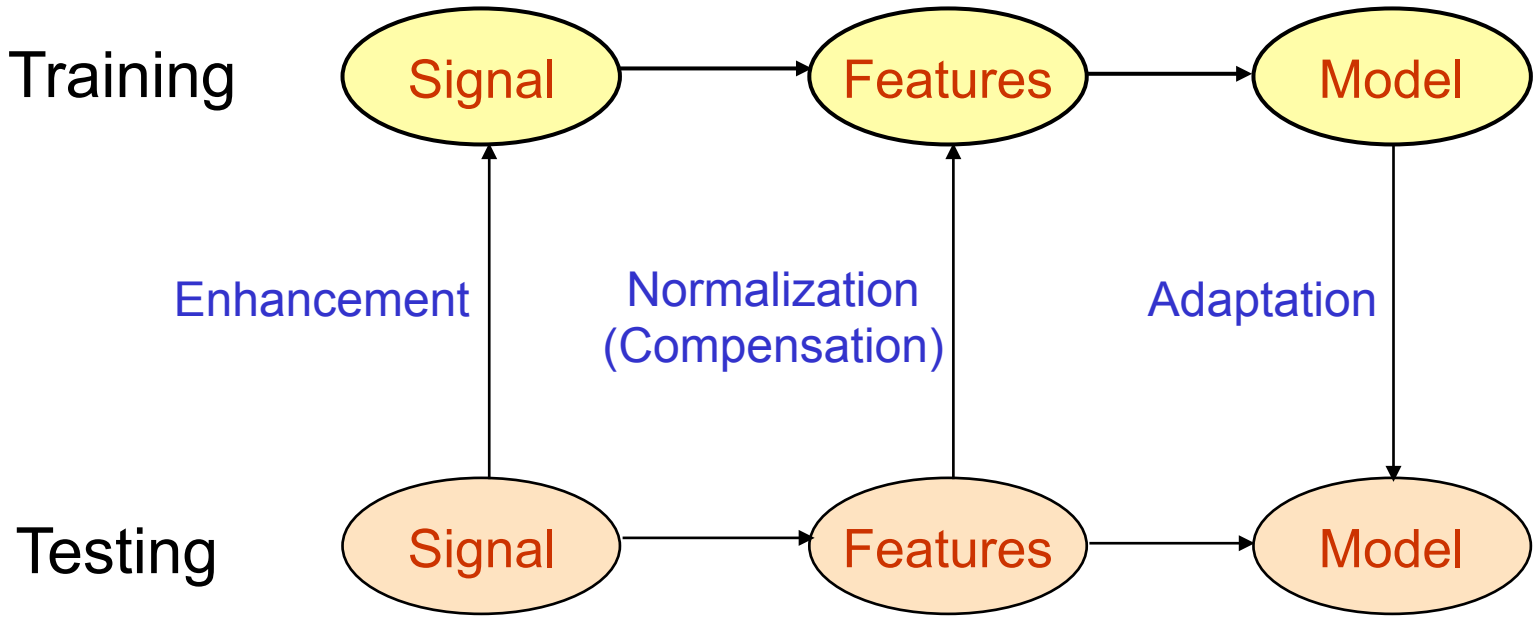
traditional techniques for improving system robustness are based on signal enhancement, feature normalization or/and model adaptation.

Perception Approach:

extract fundamental acoustic information in narrow bands of speech.
Robust integration of features across time and frequency.

Robust Speech Recognition

• A mismatch in the speech signal between the training phase and testing phase results in performance degradation



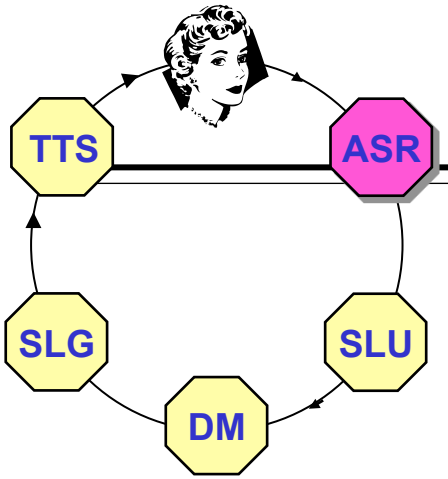
Rejection

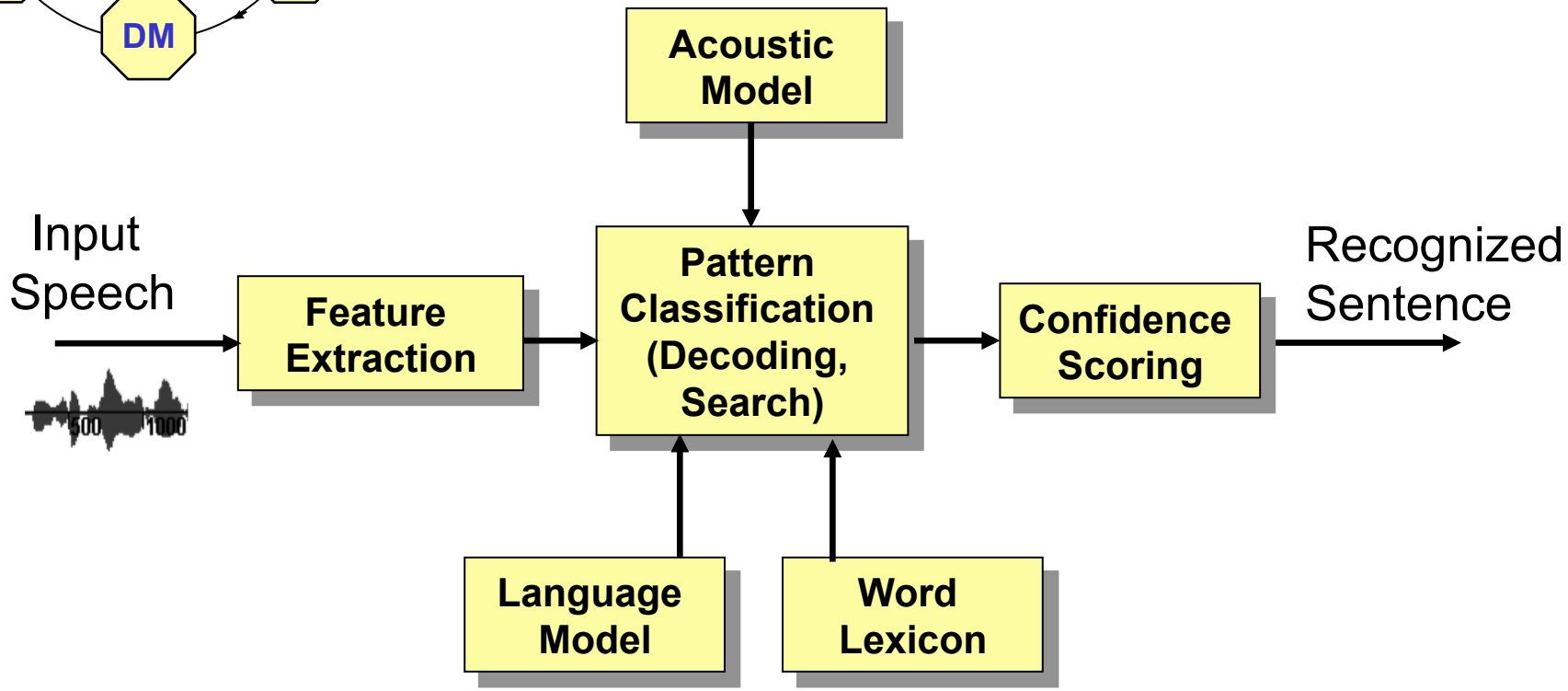
Problem:
Extraneous acoustic events, noise,
background speech and out-of-domain speech
deteriorate system performance.

Measure of Confidence:
Associating word strings with a verification cost that
provide an effective measure of confidence
(Utterance Verification).

Effect:
Improvement in the performance of the recognizer,
understanding system and dialogue manager.

State-of-the-Art Performance





How to Evaluate Performance?

- Dictation applications: Insertions, substitutions and deletions

$$\text{Word Error Rate} = 100\% \times \frac{\#Subs + \#Dels + \#Ins}{\text{No.of words in the correct sentence}}$$

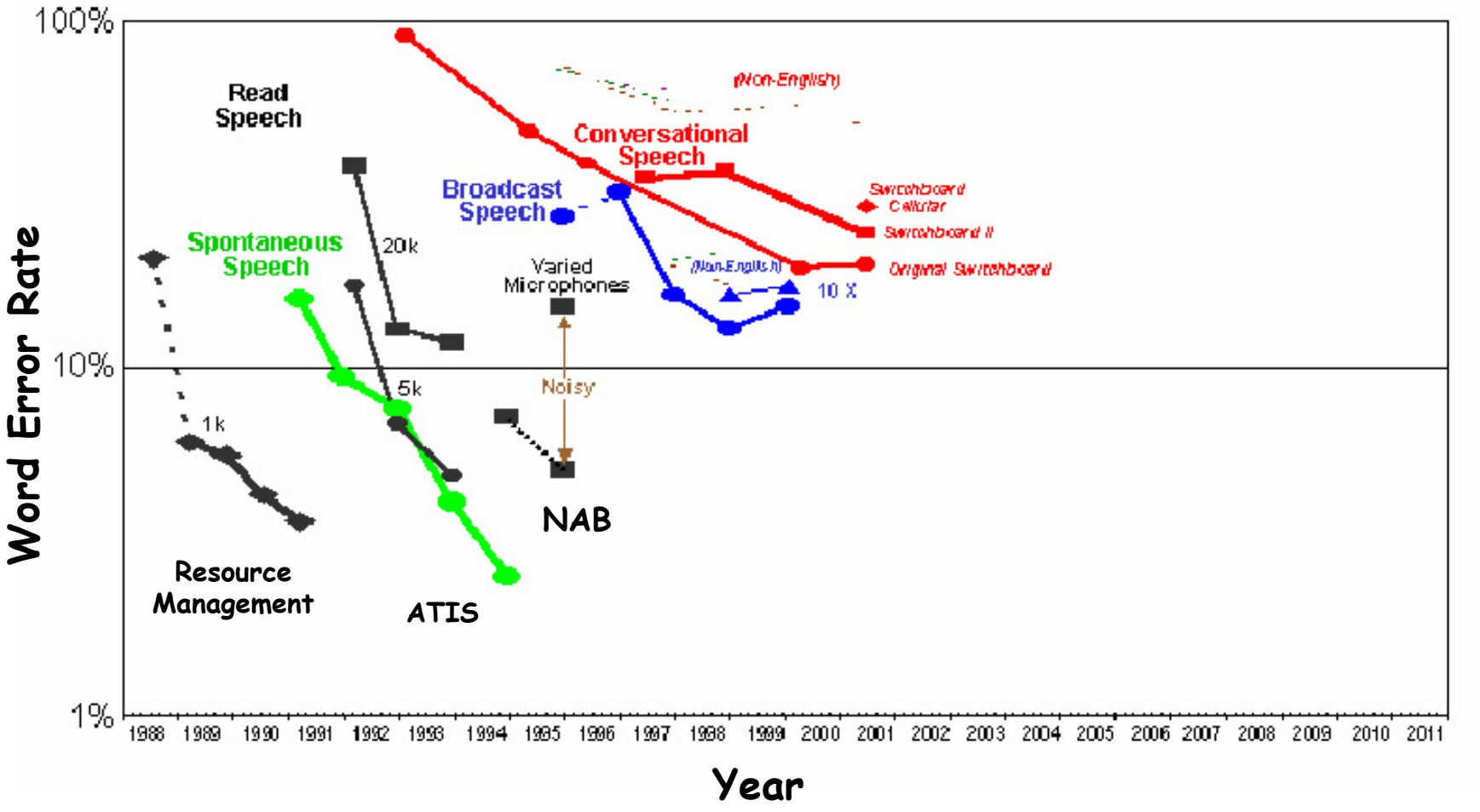
- Command-and-control: false rejection and false acceptance => ROC curves

Word Error Rates

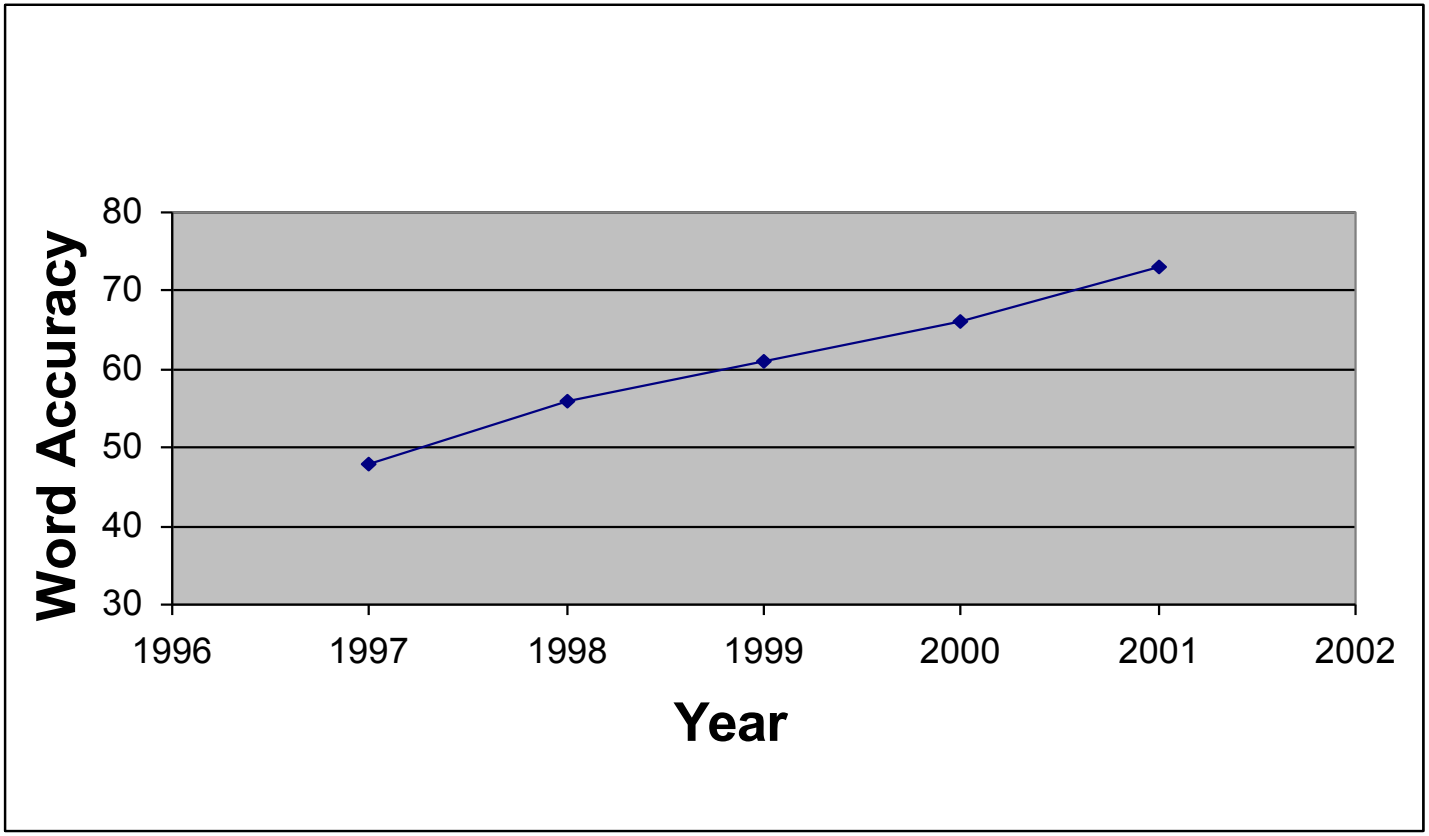
CORPUS	TYPE	VOCABULARY SIZE	WORD ERROR RATE
Connected Digit Strings--TI Database	Spontaneous	11 (zero-nine, oh)	0.3%
Connected Digit Strings--Mall Recordings	Spontaneous	11 (zero-nine, oh)	2.0%
Connected Digits Strings--HMIHY	Conversational	11 (zero-nine, oh)	5.0%
RM (Resource Management)	Read Speech	1000	2.0%
ATIS(Airline Travel Information System)	Spontaneous	2500	2.5%
NAB (North American Business)	Read Text	64,000	6.6%
Broadcast News	News Show	210,000	13-17%
Switchboard	Conversational Telephone	45,000	25-29%
Call Home	Conversational Telephone	28,000	40%

factor of 17 increase in digit error rate

NIST Benchmark Performance



Accuracy for Speech Recognition



Switchboard/Call Home Vocabulary:
40,000 words Perplexity: 85

Challenges in ASR

System Performance

- Accuracy
- Efficiency (speed, memory)
- Robustness

Operational Performance

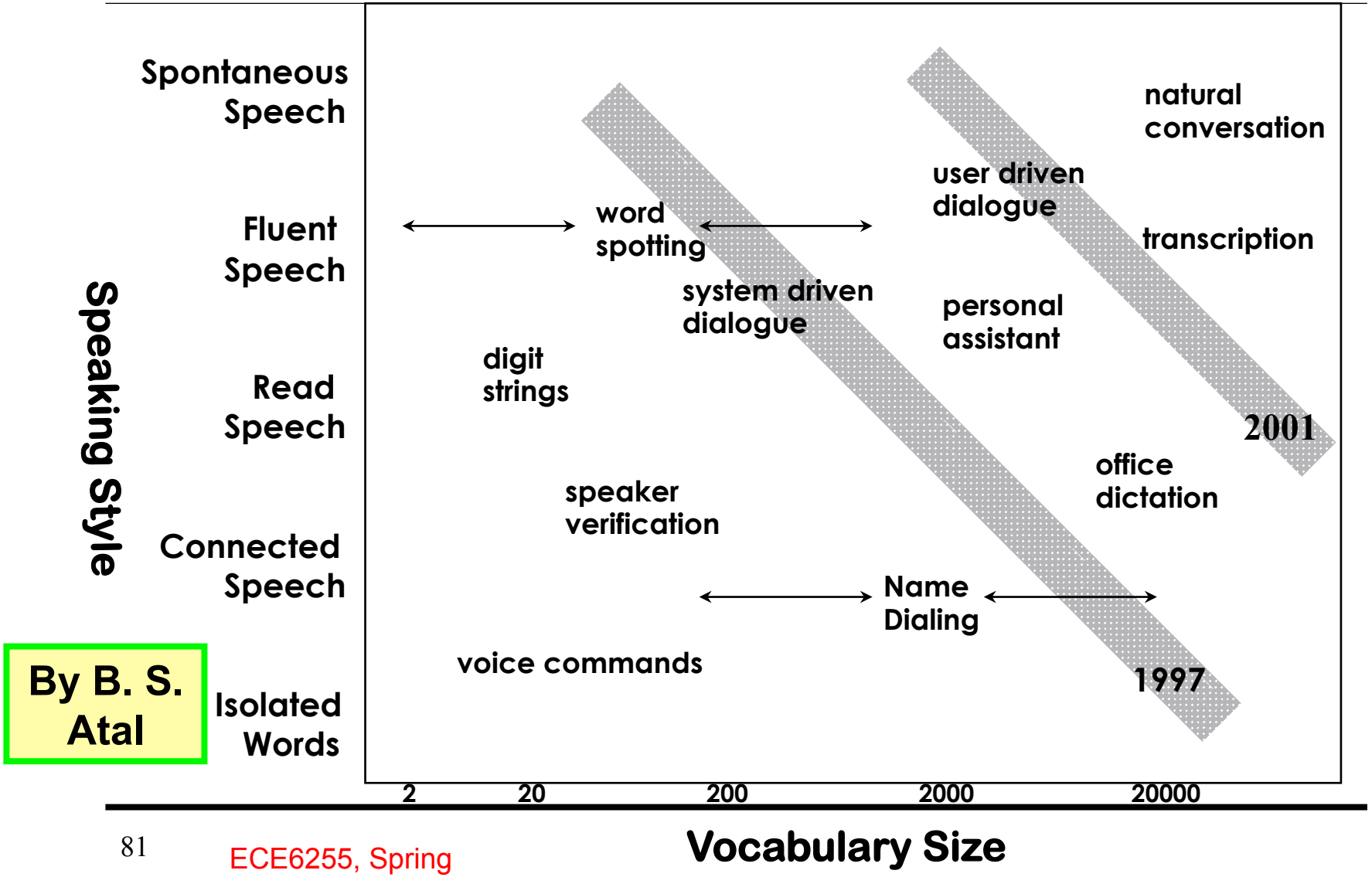
- End-point detection
- User barge-in
- Utterance rejection
- Confidence scoring

Machines are 10-100 times less accurate than humans

The End

Thank you!

Speech Recognition Capabilities



By B. S. Atal