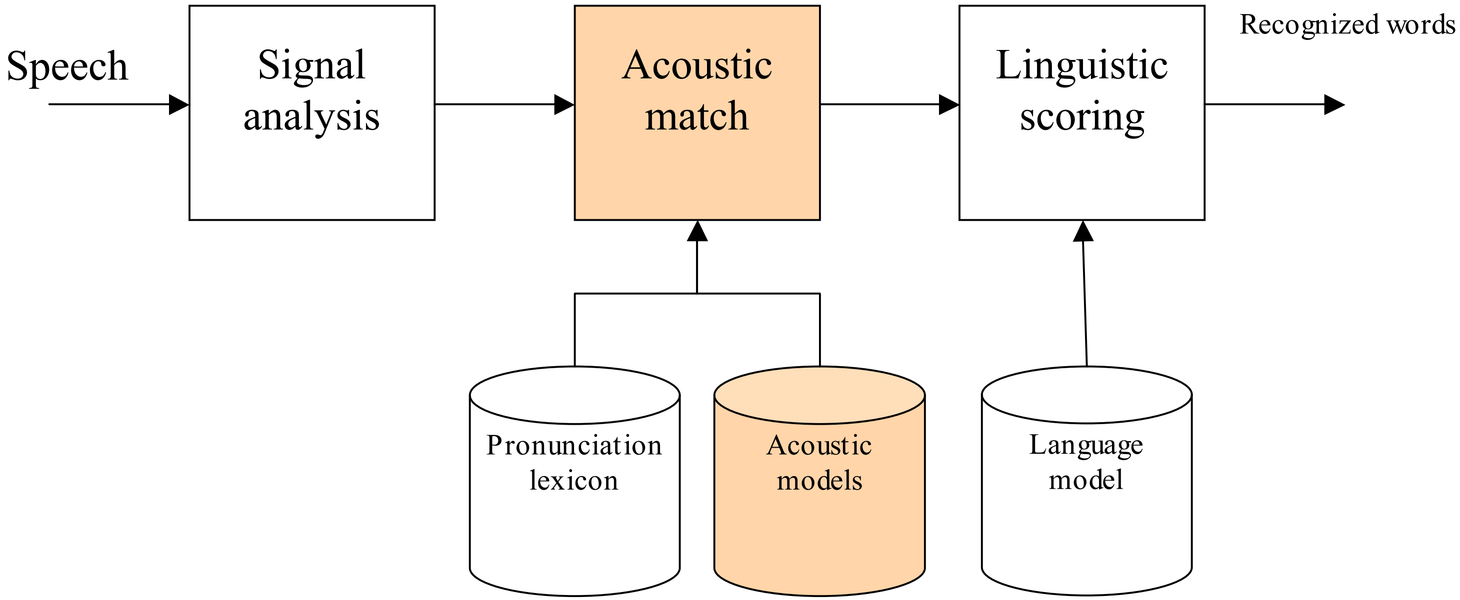
Statistical pattern matching: Outline

- Introduction
- Markov processes
- Hidden Markov Models
  - Basics
  - Applied to speech recognition
  - Training issues
- Pronunciation lexicon
- Large vocabulary speech recognition

ASR step-by-step: Acoustic match (2)

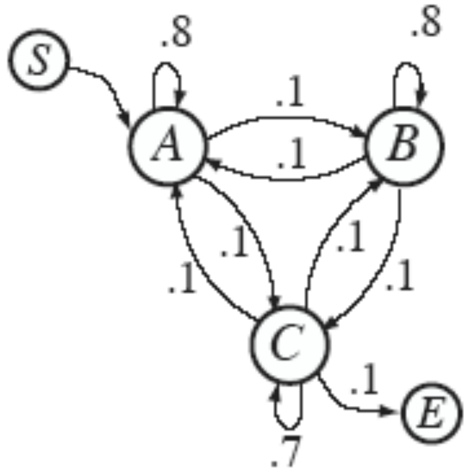


Statistical pattern recognition

- DTW is fine for small vocabulary or isolated word recognition
- Lacks the capability to model naturally occurring variations in continuous speech
- Variations in spoken language (acoustic and maybe also lexical) can be regarded as statistical fluctuations
- If we can find a suitable statistical model for speech production, it can also be applied to speech recognition

- Hidden Markov models (HMM) are the basis for current state-of-the-art in speech recognition

(First order) Markov process



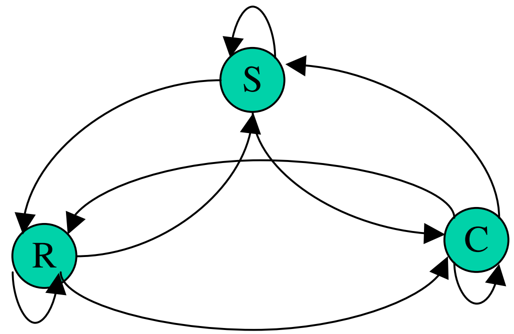
$p(q_{n+1}\|q_n)$		q_{n+1}: S	A	B	C	E
q_n	S	0	1	0	0	0
	A	0	.8	.1	.1	0
	B	0	.1	.8	.1	0
	C	0	.1	.1	.7	.1
	E	0	0	0	0	1

(from Ellis)

S A A A A A A A A B B B B B B B B B C C C C B B B B B B C E

- Time discrete random process where state is directly associated with the output
- Next state is only dependent on current state and the transition probabilities
- Transition matrix defines the probability of state at next time instance given the current state
- Ergodic process means that any state is reachable in a single step from any other state
- Left-to-right topology suitable for the temporal structure of speech

Example: Weather



- Assume that the weather can be modeled as a 1st order Markov process, i.e.:
  - The weather today has a dependency on the weather yesterday, but is not dependent on the weather on any other previous day
  - P(weather today | weather history)=P(weather today | weather yesterday)
- Three types: Sunny (S), Rain (R), Cloudy (C)
- P(S|S)=2/6; P(R|S)=2/6; P(C|S)=2/6;
  P(S|R)=1/6; P(R|R)=3/6; P(C|R)=2/6;
  P(S|C)=3/6; P(R|C)=1/6; P(C|C)=2/6
- P(S)=2/6; P(C)=3/6; P(R)=1/6
- Probability of week with S;S;S;S;C;C;R given that the last day of previous week had rain:

P(R)P(S|R)P(S|S) P(S|S)
P(S|S)P(C|S)P(C|C)P(R|C)=

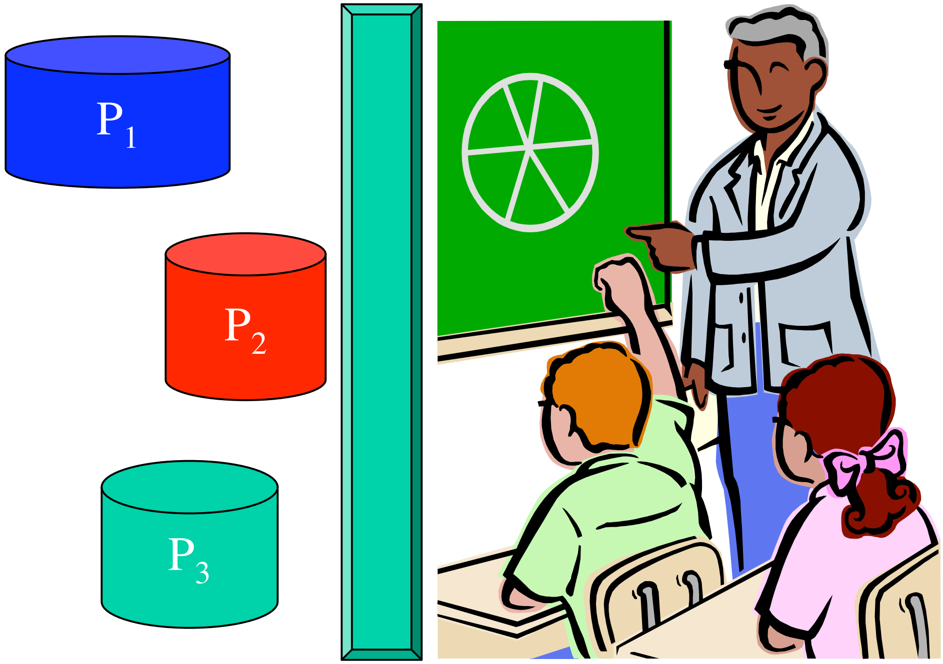
1/6*1/6*2/6*2/6*2/6*2/6*2/6*1/6=0.000152

Hidden Markov models

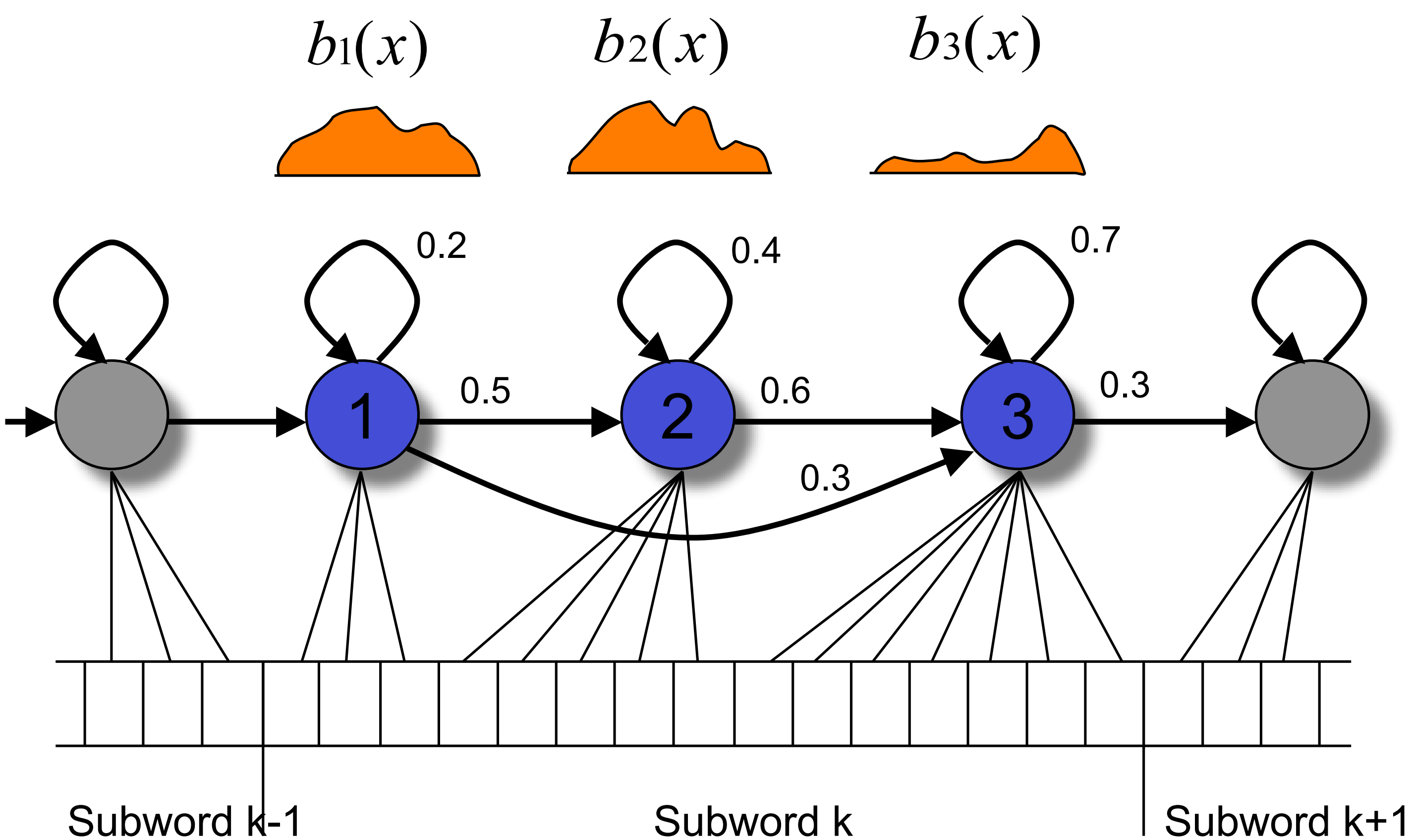
- In a Markov process, the observation is directly linked to the emitting state
- In a *hidden* Markov model, the observation is a probabilistic function of the state.
  - The HMM is a doubly stochastic process
  - Each state has an associated probability density of the emission symbols
  - If the process is in a given state, output symbols are emitted according to this probability density
- If we observe a sequence of symbols, the underlying state sequence is not known
- But we can estimate the most *likely* state sequence for an observed sequence of symbols, if the model parameters are known

Hidden Markov process

- Each urn contains colored balls
- Color distribution is different for each urn
- Movement of person drawing balls is not seen
- Estimate the movement based on the observed sequence of ball colors



Hidden Markov Models - HMM



HMM specification

- Number of states, N
- Initial probabilities, i.e. the probability of being in a state at time t=0
- Transition probabilities, $\{a_{ij}\}$, i,j=1,...,N
  - a_{ij}=P(state *j* at $t=n+1$ | state *i* at $t=n$)
  - Can be written as a NxN matrix
  - Observing the left-right temporal structure of speech, the matrix will be upper triangular (i.e. probability of going backwards is zero)
- Observation probabilities/densities, $\{b_j(\mathbf{x})\}$
  - $b_j(\mathbf{x})=p(\mathbf{x}$ | state *j*)

HMM assumptions

- Conditional independence assumption
  - The observation at time *t* is only dependent on the current state and is independent of previous observations
  - Known to be incorrect - from theory of speech production
- The durations of each state is implicitly modeled from the self-transition probabilities
  - I.e. - a geometric duration distribution
  - Does not fit known duration distribution
- The Markov assumption:
  - The state at time *t* is only dependent on the state at time *t-1*
  - $P(s_t \mid s_1^{t-1}) = P(s_t \mid s_{t-1})$
  - Second order models would alleviate some of the duration modeling deficiencies but are computationally very expensive

- In spite of this, they work!

HMMs for speech recognition

- The error rate will be minimized if the MAP criterion is employed:

$$M^* = \underset{M_j}{\operatorname{argmax}}\ p(M_j | X, \Theta)$$

  – I.e. Select the model that has the highest probability of having generated the observations

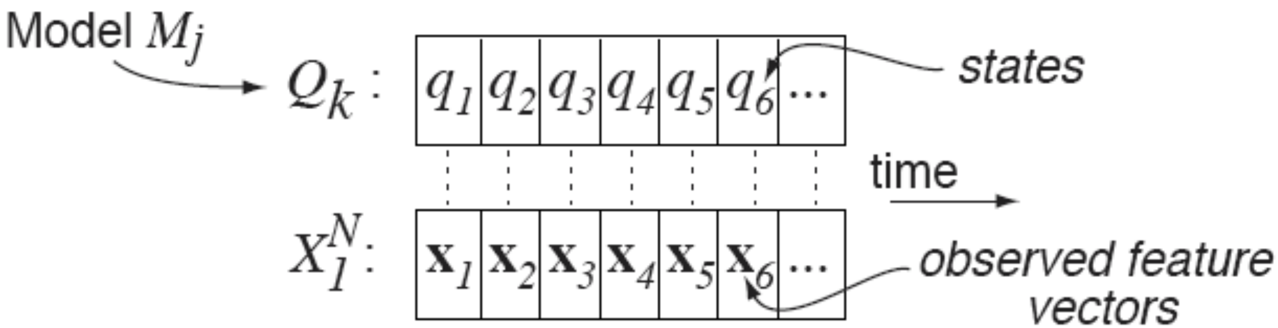
- We can rewrite the above expression using Bayes' rule

$$\begin{aligned} M^* &= \underset{M_j}{\operatorname{argmax}}\ p(M_j | X, \Theta) \\ &= \underset{M_j}{\operatorname{argmax}}\ p(X | M_j, \Theta_A)\, p(M_j | \Theta_L) \end{aligned}$$

Acoustic model

Language model

HMMs for speech recognition (2)



- Observations are time discrete sequence of feature vectors
- A sentence model is composed of a sequence of states (normally constructed by concatenating subword/phone models)

The HMM problems

- Evaluation
  - Given a model and a sequence of observations, what is the probability that the model has generated the observations?
  - Sum of probabilities of all allowed paths through model
  - Efficient solution using ”Forward” and ”backward” algorithms
  - Similar to dynamic programming
- Decoding
  - Given a model and a sequence of observations, what is the most likely state sequence in the model that produces the observations?
  - Can be evaluated efficiently using dynamic programming - the Viterbi algorithm

The HMM problems (2)

- Learning
  - Given a model an a set of observations, how can we adjust the model parameters to maximize likelihood (the probability of the observations for the given model)?
  - Two main solutions:
    - Baum-Welch algorithm
      - Guarantees that change in likelihood will be non-negative
      - Theoretically best solution
      - Efficient implementation using forward and backward algorithm
    - Viterbi training
      - Maximizes likelihood of best path, i.e. sub-optimal with respect to criterion
      - Efficient
      - Corresponds well to the recognition procedure

Recognition with acoustic models

- Evaluation of the likelihood is too costly
- Pragmatic choice:
  - Likelihood of best path dominates the likelihood score
  - Approximate likelihood with likelihood of best path
  - Can use Viterbi algorithm for recognition
  - Efficient implementation

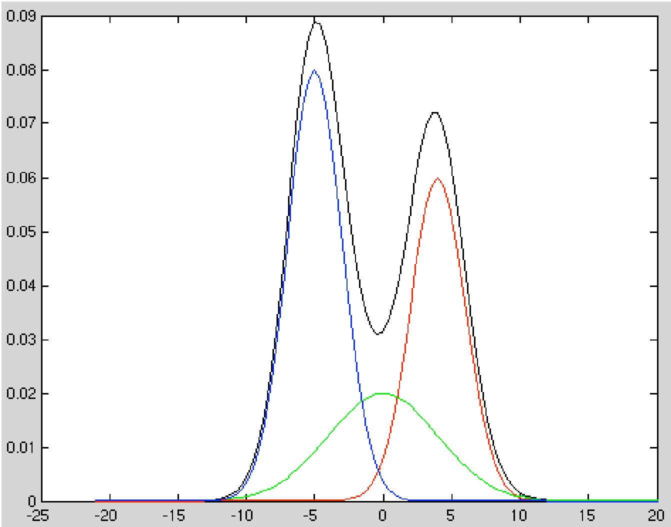
$$M^* = \underset{M_j}{\operatorname{argmax}}\, p(X \mid M_j, \Theta_A) = \underset{M_j}{\operatorname{argmax}} \sum_{\forall \{Q = q_1, \ldots, q_N\}} p(X, Q \mid M_j, \Theta_A)$$

$$\approx \underset{M_j}{\operatorname{argmax}} \left\{ \underset{Q}{\operatorname{argmax}}\, p(X, Q \mid M_j, \Theta_A) \right\}$$

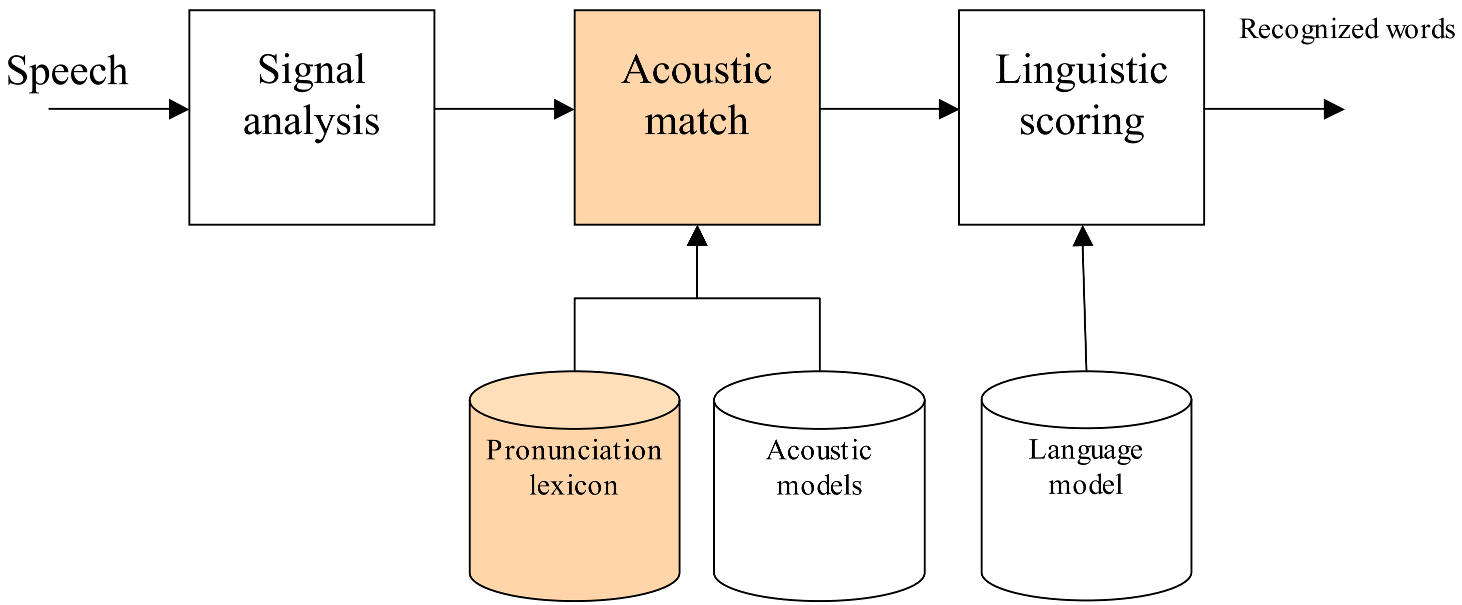
Observation probabilities

- In early HMM systems, observations were discrete (e.g. VQ indices)
- In order to avoid information loss, this was abandoned
  - **x** is a continuous multi-dimensional variable
- Efficient description of a multivariate probability density function
  - Parametric representation
  - Gaussian mulitvariate mixture density

$$b_j(\mathbf{x}) = \sum_{i=1}^{M} c_i \mathcal{N}(\mathbf{x}, \mathbf{m_{ji}}, \mathbf{C_{ji}})$$



ASR step-by-step: Acoustic match (2)



Basic unit for speech recognition

- Longer unit -> better modelling of coarticulatory effects
- Large units require extremely large amounts of training data
  - Coarticulation effects at unit boundaries
- Small units (e.g. phones) are attractive as they
  - Can describe the language with a small number of units
  - Are generalizable
  - Have a linguistic interpretation

  but they do not capture context dependent effects
- Solution: Context dependent phone models
  - Train models for all phones in all possible context
    - Immediat left-right context -> ”trigram” models

NTNU

Training issues

- Context dependent phone models lead to an explosion in the number of models that need to be estimated
  - 50 phones -> 125.000 context dependent models
- Use of Gaussian mixture models contribute further to complexity
  - Typecal parameter vector: 13 MFCC + Δ- and ΔΔ-parameters; i.e. 39 dimensional vector
  - Each mixture component requires mean vector, (diagonal) covariance matrix and mixture weight, i.e. 79 parameters
- Example: independent models for all phone models, 3-state phone models using 16 mixture components per state, 39-d feature vector:
  - 125.000*3*79*16=474 million parameters
- Large number of parameters mean
  - Problematic to obtain sufficient amount of training data for reliable estimates (note that some sound combinations are very rare)
  - High cost in recognition

State tying

- Many contexts result in acoustically similar realizations
- Similar states should be able to share parameters and training material
- How to identify states with similar acoustic distributions?
  - Current wisdom: phonetic desicion trees
- Procedure:
  - Train a reasonably good set of context independent models
  - From these, generate an initial set of context dependent models
  - Use a phonetic decision tree to cluster states of contextual variants of the same ”center” phone
  - Tie these states, i.e. make them share training data and parameters
- Result: Big reduction in number of parameters (several orders of magnitude), better trained parameters

Phonetic decision trees for state tying

- Assemble a list of phonetic questions (e.g. is left context a fricative, is right context a sonorant)
- Collect all models with the same center phone at the top node
- For all (unused) quesitons, evaluate the likelihood increase by splitting the models according to that question
- Select the split that provides the highest likelihood
- For each open node, repeat the splitting procedure until a threshold in improvement is reached, or there are no further nodes to split.

Pronunciation lexicon

- Sub-word units requires need for lexicon to describe the constituents of a word
- A lexicon will contain the vocabulary words and their assoicated phone strings, e.g.

READ	r iy d
READABLE	r iy d ah b ah l
READER	r iy d er
etc.	

- Canonic baseforms only or allow pronunciation variants
- During recognition, word models can be assembled by concatenating sub-word HMMS according to the lexical description

Pronunciation lexicon issues

- Standard pronunciation lexica correspond reasonably well to how speech is pronounced when reading with a normalized pronunciation
- Important issues are
  - What to do if a pronunciation lexicon does not exist for a language
  - Representation of dialects and accents
  - Anomalities in spontaneous speech
- If TTS engine exists in a language, a first approximation lexicon can be generated from the TTS front end
- Pronunciation modeling techniques are being pursued in order to
  - Improve general performance of ASR
  - Explain and model spontaneous and accented speech
  - I.e. model the systematic differences that exist on a lexical level (as opposed to acoustic variations due to voice characteristics or environmental noise)

Large vocabulary ASR

- When the vocabulary is large, the resulting state network grows to become unmanageable
- By restricting the search, big savings in computation and memory can be achieved
- Beam search is commonly used
  - Instead of keeping score of all competing paths, discard the paths that seem unlikely to become the ultimate winner
    - Keep only the best N paths
    - Keep only the paths with likelihoods within a given percentage of the current best path
  - Can risk that the ”correct” path is discarded if beam width set too narrow
  - Other alternatives exist

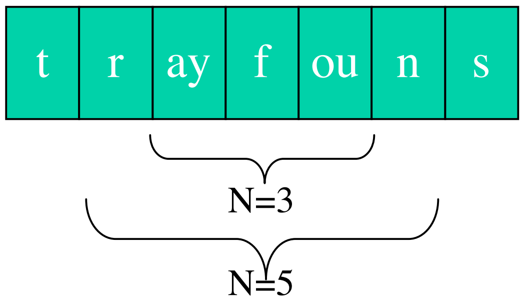
Large vocabulary ASR (2)

- Two-pass recognition
  - Perform N-best recognition using fairly crude models
    - N-best: Output the N most likely word sequences instead of only the best
    - Can be structured as a word lattice
  - Do a second pass using your best models, restricted to search among the candidates produced in the first pass
  - Significant reduction in computational demands without significant loss in recgnition performance
  - Produces additional recognition delay
- Depth-first search
  - Explore most promising path(s) first
  - Asyncronous with input
  - Stack decoding, A^* search

NTNU

Large vocabulary ASR (3)

- Increased accuracy in acoustic models
  - Cross-word ”triphones”
    - Context dependent models normally limited to intra-word contexts
    - Build acoustic models also for contexts that only occur at word boundaries
    - Use context dependency also at word boundaries
    - Improves accuracy, but increases search complexity
  - Quinphones and beyond
    - Increase context dependency beyond the immediate neighbors
    - *N*-phones: context includes *N/2* neighbors on each side
      - Triphone: *N=3;* Quinphone: *N=5*



Language modelling

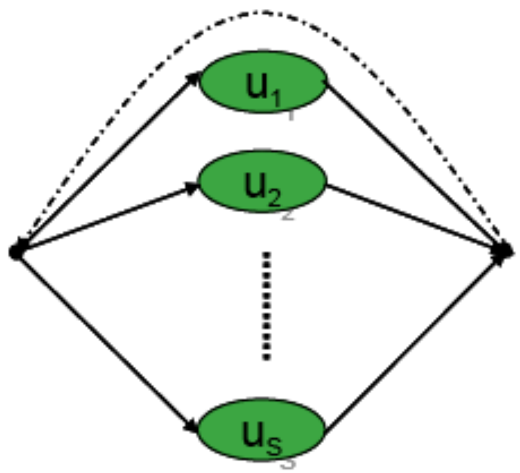
$$M^* = \underset{M_j}{\operatorname{argmax}}\, p(X \mid M_j, \Theta_A) \cdot p(M_j \mid \Theta_L)$$

Acoustic model   Language model

- The importance of the language model increase with the size of the vocabulary
  - Large vocabulary generally implies more complex language structure
  - Perplexity, average branching factor
  - A good language model can
    - Improve recognition rate
    - Reduce search complexity

Grammar

- The grammar specifies
  - The vocabulary
  - Any restrictions on the syntax
- Defined as a finite state network
- Null grammar
  - No restrictions
- Word pair grammar
  - Define all allowable word combinations
- Adding weights to arcs lead to language model
  - Uniform weights: No LM
  - Simple weighted arcs: Unigram
  - Context dependent weights: N-gram



Statistical language model - N-gram

- N-gram LM describes the probability of word *N*-tuples
- Simplification of "real-world" language complexity

$$P(W_l \mid W_1^{l-1}) = P(W_l \mid W_1 W_2 \ldots W_{l-1}) \approx P(W_l \mid W_{l-N+1} W_{l-N+2} \ldots W_{l-1})$$

- *N*=3 - trigram language model; *N*=2 - bigram language model
- Bigram example
  - Probability of a sequence of *S* words

$$\text{Bigram,N} = 2: \qquad P(W_l \mid W_1^{l-1}) = P(W_l \mid W_{l-1})$$

$$P(W_1^S) = P(W_S \mid W_{S-1}) \cdot P(W_{S-1} \mid W_{S-2}) \cdot \ldots \cdot P(W_2 \mid W_1) P(W_1)$$

$$= P(W_1) \cdot \prod_{j=2}^{S} P(W_j \mid W_{j-1})$$

N-gram language model (2)

- Power of model increses with *N*
- Complexity of decoding increase exponentially with *N*
- Data sparsity problem in training
  - Simple estimation by frequency counts
    - Trigram: $P(W_a|W_b,W_c)=Count(W_a,W_b,W_c)/Count(W_b,W_c)$
  - Uneven distribution of words in the language
    - Huge text databases required; hundres of millions of words
    - Even then, many quantities cannot be estimated
  - Need for methods to account for missing data
    - Discounting
      - Free part of probability mass for unseen events - uniform probability assignment
      - Adjust observeable probabilities
    - Back-off
      - In *N*-gram does not exist, use *N-1* gram
      - Keep going until a model exists

Last issue: The optimization criterion

- Training by maximizing the likelihood of the acoustic models
  - Models can be individually optimized
  - Does not ensure maximal discriminability
- Maximization of discrimination capability
  - Maximum mutual information (MMI)
- Minimum classification error
  - Optimization criterion: Minimize probability of error
  - Yields a more complex training procedure
- Corrective training
  - Adjust the models that make errors (and near errors)
  - Keep the rest unchanged

Current state-of-the-art (Soong&Juang, 2003)

Task	Vocabulary size	Mode	Word accuracy	Task	Vocabulary size	Perplex.	Word accuracy
Digits (0-9)	10	SI	~100%	Connected digits	10	10	~99%
Voice dialling	37	SD	100%	Naval resource management	991	<60	97%
Alphadigits+ Command words	39	SD/SI	96%/93%	Air travel information	1800	<25	97%
Air travel words	129	SD/SI	99%/97%	Business newspaper transcription	64.000	<140	94%
Japanese city names	200	SD	97%	Broadcast news transcription	64.000	<140	86%
Basic English words	1109	SD	96%				

NTNU