

Large vocabulary ASR

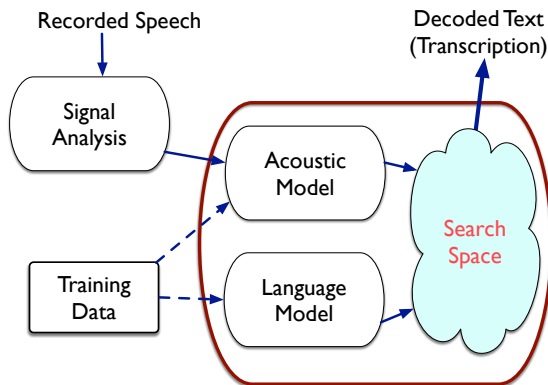
Peter Bell

Automatic Speech Recognition – ASR Lecture 8
8 February 2024

Large-vocabulary recognition

- The Viterbi algorithm for isolated and connected words
- Decoding with bigram and trigram language models
- Methods for efficient search: pruning, tree-structured lexicons, look-ahead

HMM Speech Recognition



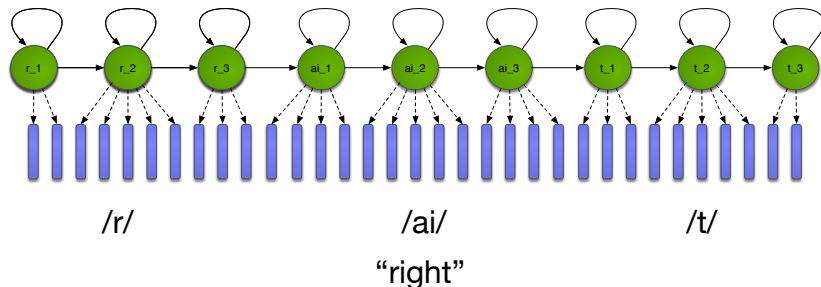
The Search Problem in ASR

- Find the most probable word sequence $\hat{W} = w_1, w_2, \dots, w_M$ given the acoustic observations $X = x_1, x_2, \dots, x_T$:

$$\begin{aligned}\hat{W} &= \arg \max_W P(W|X) \\ &= \arg \max_W \underbrace{p(X | W)}_{\text{acoustic model}} \underbrace{P(W)}_{\text{language model}}\end{aligned}$$

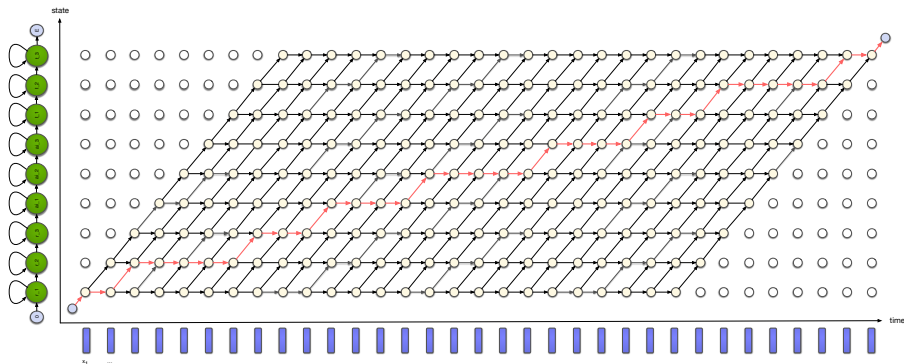
- Use pronunciation knowledge to construct HMMs for all possible words
- Finding the most probable state sequence allows us to recover the most probable word sequence
- Viterbi decoding* is an efficient way of finding the most probable state sequence, but even this is infeasible as the vocabulary gets very large or when a stronger language model is used

Recap: the word HMM



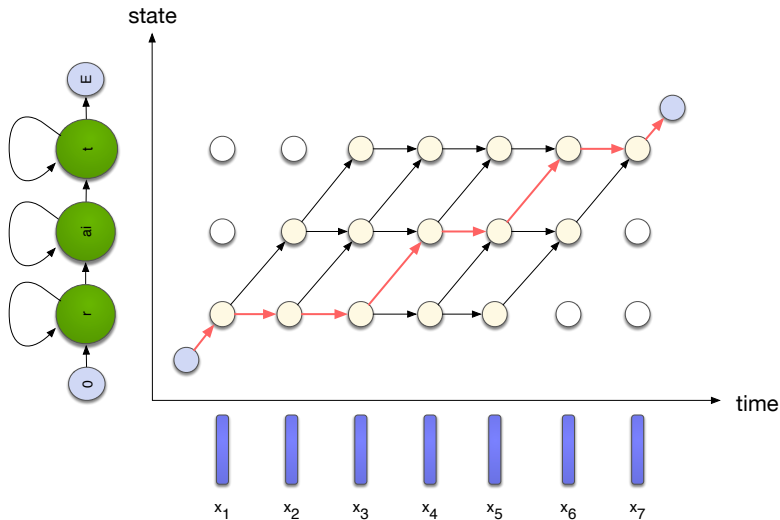
HMM naturally generates an alignment between hidden states and observation sequence

Viterbi algorithm for state alignment

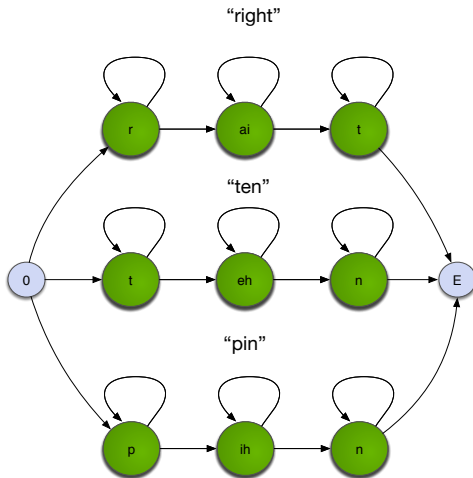


Viterbi algorithm finds the best path through the trellis – giving the highest $p(X, Q)$.

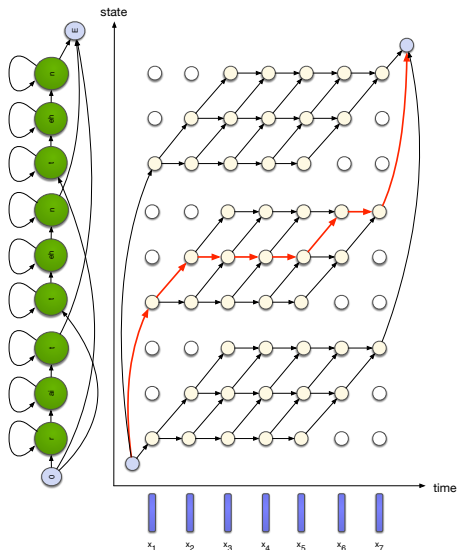
Simplified version with one state per phone



Isolated word recognition



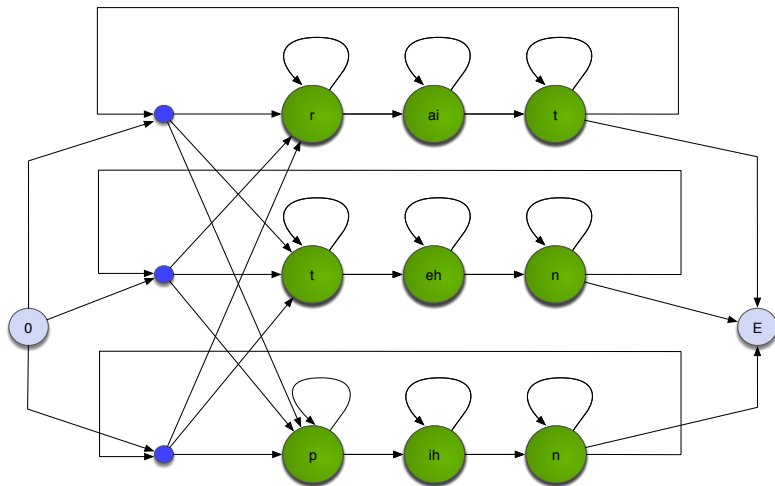
Viterbi algorithm: isolated word recognition



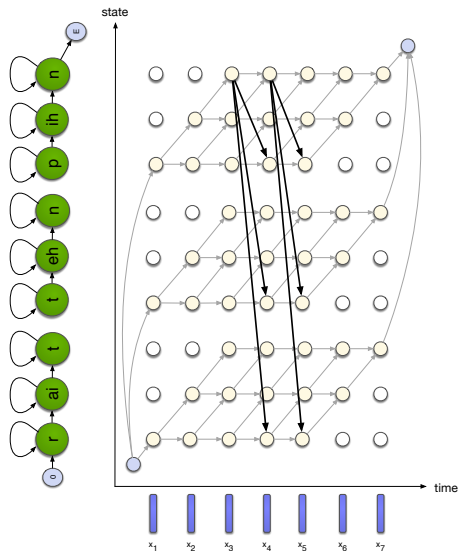
Connected word recognition

- Even worse when recognising connected words...
- The number of words in the utterance is not known
- Word boundaries are not known: any of the V words may potentially start at each frame.

Connected word recognition

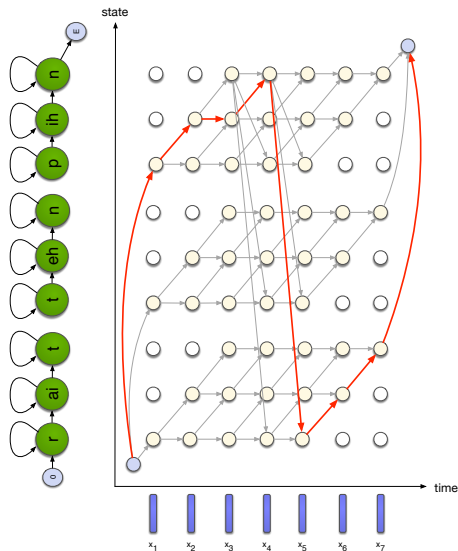


Viterbi algorithm: connected word recognition



Add transitions between
all word-final and
word-initial states

Connected word recognition



Viterbi decoding finds the best word sequence

BUT: have to consider $|V|^2$ inter-word transitions at every time step

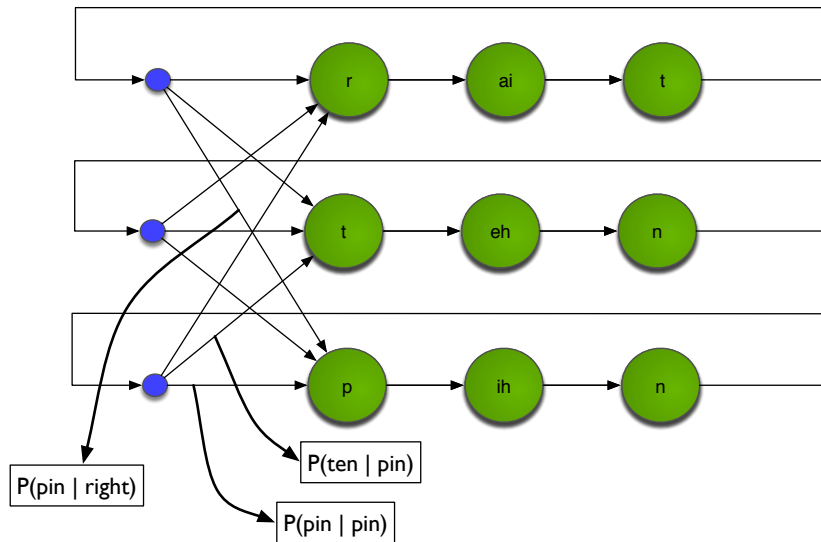
Integrating the language model

- So far we've estimated HMM transition probabilities from audio data, as part of the acoustic model
- Transitions *between words* → use a language model
- n -gram language model:

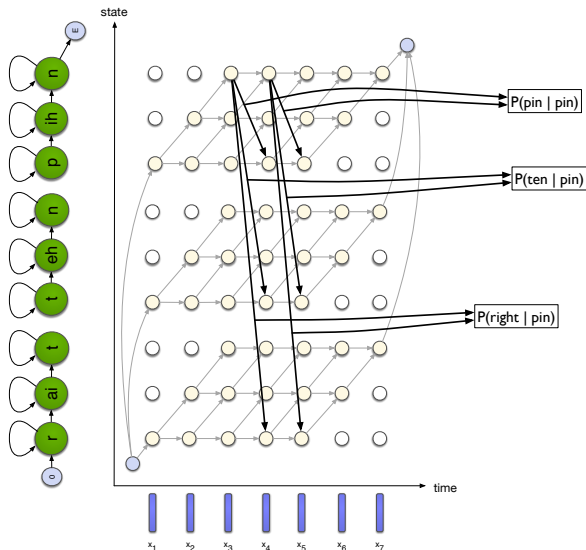
$$p(w_i|h_i) = p(w_i|w_{i-n}, \dots w_{i-1})$$

- Integrate the language model directly in the Viterbi search

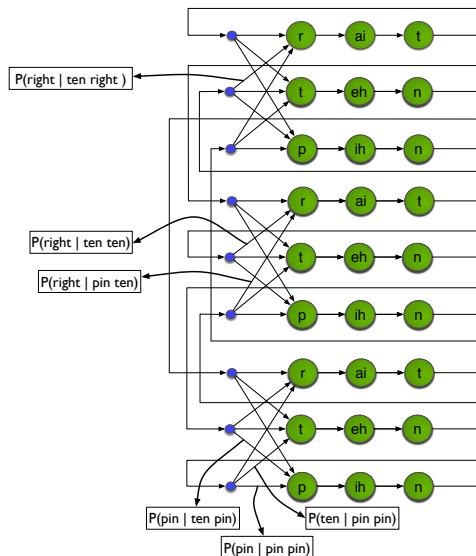
Incorporating a bigram language model



Incorporating a bigram language model



Incorporating a trigram language model

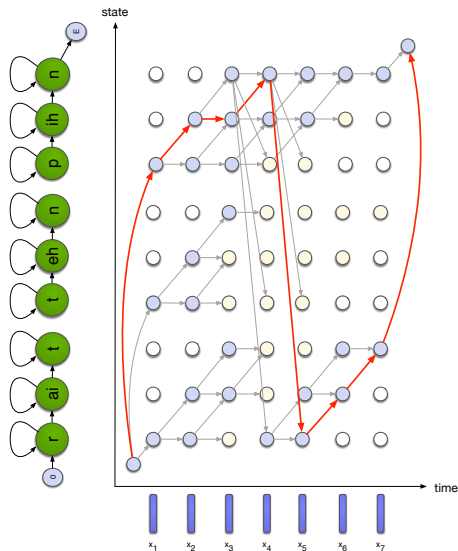


Need to duplicate HMM states to incorporate extended word history

- Viterbi decoding performs an exact search in an efficient manner
- But exact search is not possible for large vocabulary tasks
 - Long-span language models and the use of cross-word triphones greatly increase the size of the search space
- Solutions:
 - Beam search (prune low probability hypotheses)
 - Tree-structured lexicons
 - Language model look-ahead
 - Dynamic search structures
 - Multipass search (\rightarrow two-stage decoding)
 - Best-first search (\rightarrow stack decoding / A^* search)

- Viterbi decoding performs an exact search in an efficient manner
- But exact search is not possible for large vocabulary tasks
 - Long-span language models and the use of cross-word triphones greatly increase the size of the search space
- Solutions:
 - Beam search (prune low probability hypotheses)
 - Tree-structured lexicons
 - Language model look-ahead
 - Dynamic search structures
 - Multipass search (\rightarrow two-stage decoding)
 - Best-first search (\rightarrow stack decoding / A^* search)
- Next lecture: an alternative approach using weighted finite state transducers (WFSTs)

Pruning



During Viterbi decoding,
don't propagate tokens
whose probability falls a
certain amount below the
current best path

Result is only an
approximation to the best
path

Tree-structured lexicon

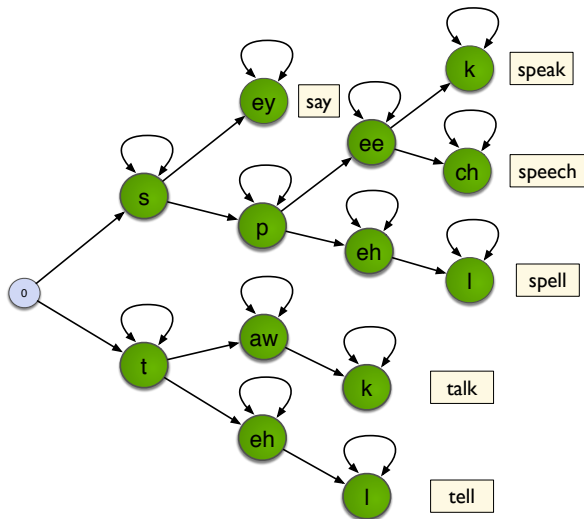
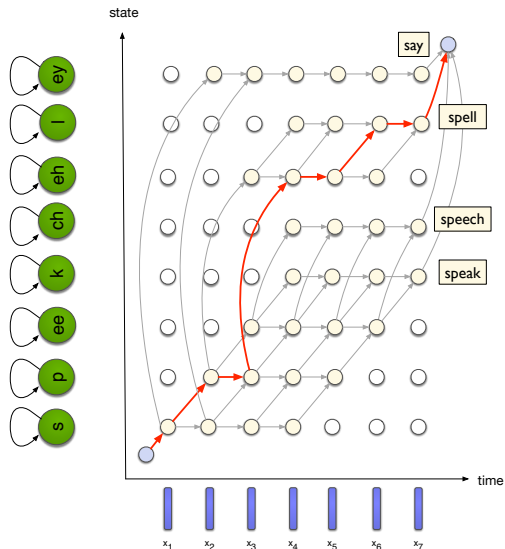


Figure adapted from Ortman & Ney, "The time-conditioned approach in dynamic programming search for LVCSR"

Tree-structured lexicon



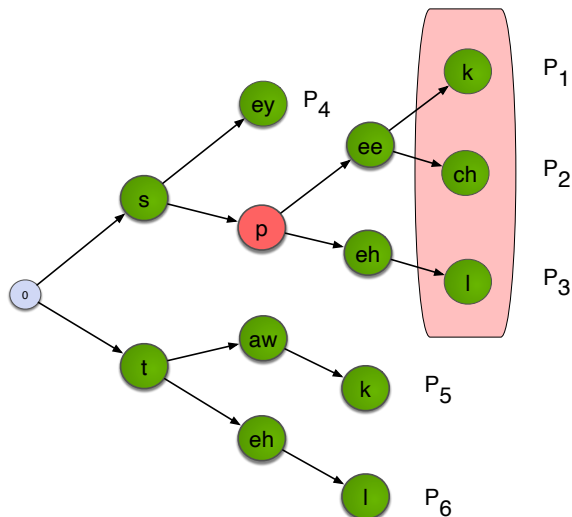
Reduces the number of
state transition
computations

For clarity, not all the connections are shown

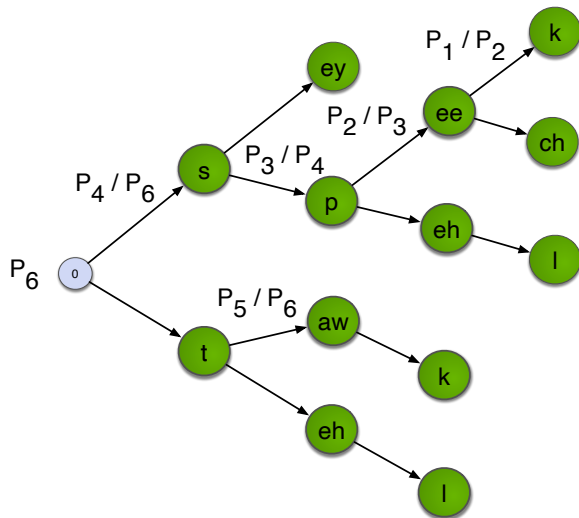
Language model look-ahead

- Aim to make pruning more efficient
- In tree-structured decoding, look ahead to find out the best LM score for any words further down the tree
- This information can be pre-computed and stored at each node in the tree
- States in the tree are pruned early if we know that none of the possibilities will receive good enough probabilities from the LM.

Language model look-ahead



Language model look-ahead



Push probabilities down the tree (assuming $P_6 > P_5 > \dots > P_1$)

- Ortmanns and Ney (2000). “The time-conditioned approach in dynamic programming search for LVCSR”. In *IEEE Transactions on Speech and Audio Processing*