Methodologies for Language Modeling and Search in Continuous Speech Recognition

by

Neeraj Deshmukh
Dept. of EE
Boston University
Boston, MA 02215
neeraj@engc.bu.edu

Joseph Picone
Instt. for Signal & Info. Processing
Mississippi State University
MS State, MS 39762
picone@ee.msstate.edu

Southeastcon, '95
Visualizing the Future

March 27, 1995

The Problem of Continuous Speech Recognition

♠ **Statistical Pattern Recognition**

Mathematical representation:

$$p(\widehat{W}/A) = \max_{W} p(W/A)$$

Bayes' Theorem:

$$p(\widehat{W}/A) = \arg\max_{W} p(A/W)p(W)$$

♠ **Automatic Speech Recognition**

- $W = w_1, w_2, \ldots, w_N$
- Acoustic Model $p(A/W)$
- Language Model $p(W)$
- Search

♠ **Complex Applications ⇒ Hierarchical Modeling**

♠ **Hidden Markov Models**

- Training
- Decoding

Continuous Speech Recognition System

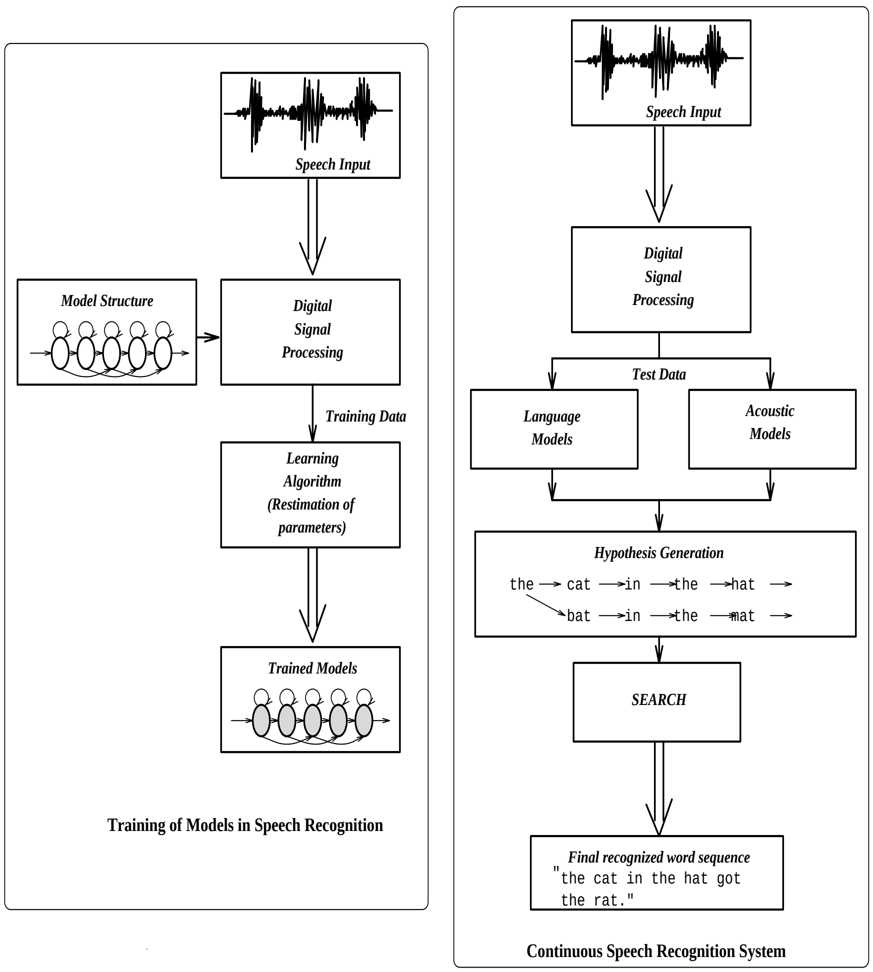

Figure 1: Training and Recognition

Schematic of training and recognition systems

Statistical Language Modeling

♠ **Motivation**

Provides constraints on the occurrence of particular words and word sequences, thus determining the search space.

♠ **Goodness Criterion**

- **Perplexity**

$$P = 2^{H(w)}$$

where $H(w)$ is the **entropy** of the language model.

- Perplexity does not represent effect of similar-sounding words
- Perplexity vs. accuracy of recognition

♠ **Popular Language Modeling Techniques**

- Static models
- Dynamic models

Static Language Models

♠ **Uniform language model**

- Probability of all words is equal ⇒ No constraints

♠ **The n-gram**

- The information about the identity of a word depends on the document history i.e. words preceding it.

$$p(W) = \prod_{i=1}^{N} p(w_i/w_1, \ldots, w_{i-1})$$

- Use the previous $n-1$ words to determine probability of occurrence of each word.

$$p(W) = \prod_{i=1}^{N} p(w_i/w_1, \ldots, w_{i-n+1})$$

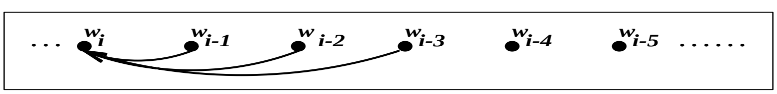

Figure 2: n-gram for $n = 3$

- Limit on value of n —– 2 or 3 at most
- Perplexity and accuracy with n

♠ **Limitations**

- Cannot adapt to style of document or topicality of data

Dynamic Language Models

♠ **Motivation**
- Exploit domain-specific nature of data
- Increase modularity by sub-language modeling
- Capture long-range linguistic phenomena

♠ **Prevalent Techniques**
- Long-distance n-grams
- Triggers
- Cache models
- Class grammars
- Tree-based models
- Mixtures

♠ Practical Issues
- Size for large vocabulary
- Computational cost for training
- Convergence of training algorithms

Search Strategies

♠ **Search Paradigm**
To choose a word sequence with the highest likelihood score for the acoustic and language models given the observed data.

♠ **Motivation**
The number of hypotheses (choices for the correct pattern) grows exponentially with length of the utterance. Hence a strategy that saves on computation and storage requirements is sought.

♠ **Approaches to restructure search**

- Optimization of hypothesis generation
- Reduction in problem space
- Search reduction
- Application of external knowledge sources

♠ **Sub-optimal choices**

♠ **Popular search techniques**

- Viterbi Search
- Viterbi Beam Search
- $A^\star$ Stack Decoding
- N-best Search
- Generalized N-best Search

Viterbi Algorithms

♠ **Viterbi decoding**
- At every instant, compute scores for all possible state transitions in the models
- Update scores of all states that give a better score on transition
- Keep track of the top scoring state at each instant
- Once end of utterance is reached, trace back to get final solution

♠ **Viterbi beam search**
- Viterbi search where only those hypotheses that have score above some threshold (or beam) value are propagated

♠ **Frame-synchronous Viterbi search**
- To optimize hypothesis generation, applies frame-level pruning in addition to the state level Viterbi beam
- Attractive for memory-critical and hierarchical systems

♠ **Implementation issues**
- Time-synchronous
- Computationally extensive for larger problems
- Inherently one-best

$A^\star$ Stack Decoding

♠ **Salient Features**

- Constructs an ordered stack of all hypotheses above a certain score
- For the hypothesis on top, shortlists possible next words using fast-match techniques
- Computes new hypothesis scores for these using detailed matches
- Re-orders the stack with these new hypotheses

♠ **Implementation issues**

- Depth-first search
- Problems of robustness and speed for large problems
- Allows use of cheaper models for fast-matches

N-best Search

♠ **Algorithm**

- Similar to Viterbi beam search
- Maintains *all* hypotheses within specified beam
- Propagates top N hypotheses at each state
- N is independent of Viterbi beam

♠ **Practical issues**

- Tool to integrate information from multiple sources
- Partial towards shorter hypotheses

♠ **Generalized N-best**

- Lattice N-best
  - ▷ Builds a lattice of word (or sentence) hypotheses in an initial pass
  - ▷ Subsequent passes eliminate poor hypotheses and downsize this lattice
  - ▷ Obtains N-best hypotheses by recursive search tracing back through this lattice
- Forward-backward search
  - ▷ Forward pass search using cheap, efficient models eliminates very poor hypotheses
  - ▷ Backward search using complex models picks the N top scoring hypotheses

Conclusion

- ♠ Current state of the art in speech recognition allows modest applications
- ♠ Prohibitive constraints of computational and memory requirements for real-life situations
- ♠ Need to develop better techniques for modeling speech, like a hierarchy of HMMs
- ♠ Need to include long-distance linguistic effects on word occurrence in efficient, practicable dynamic language models
- ♠ Search algorithms should be suitably modified to handle magnified search spaces within the bounds of real-time implementation
- ♠ Our future research will be directed primarily at developing such efficient search strategies for hierarchical systems