Return to Main

Objectives

Review:
Noisy Channel Model
Chomsky Hierarchy

N-grams:
Derivation
Examples

Complexity:
Perplexity
Examples

On-Line Resources:
XML
W3C
Software: SRILM

LECTURE 32: N-GRAM LANGUAGE MODELS

- Objectives:
  - Communication theory model of speech recognition
  - Statistical language models
  - N-gram language models
  - Perplexity

This lecture combines material from the course textbook:

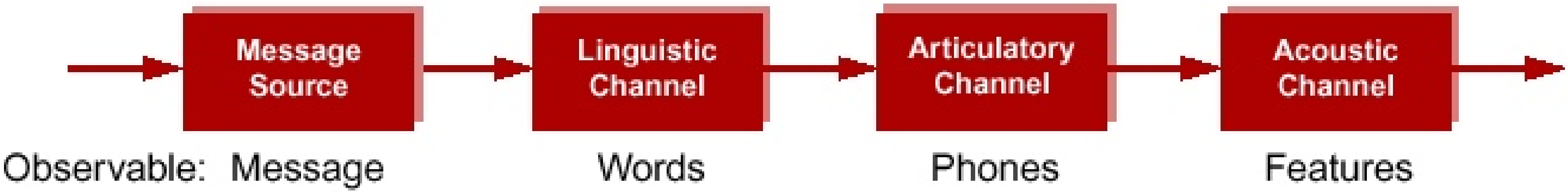
> X. Huang, A. Acero, and H.W. Hon, *Spoken Language Processing - A Guide to Theory, Algorithm, and System Development*, Prentice Hall, Upper Saddle River, New Jersey, USA, ISBN: 0-13-022616-5, 2001.

and from this source:

> F. Jelinek, *Statistical Methods for Speech Recognition*, MIT Press, Boston, Massachusetts, USA, ISBN: 0-262-10066-5, 1998.

A NOISY COMMUNICATION CHANNEL MODEL
OF SPEECH RECOGNITION

A noisy communication theory model for speech production and perception:



Bayesian formulation for speech recognition:

$$P(W|A) = P(A|W)P(W)/P(A)$$

Objective: minimize the word error rate by maximizing $P(W|A)$

Approach: maximize $P(A|W)$ (training)

Components:

- $P(A|W)$: acoustic model (hidden Markov models, mixture of Gaussians)
- $P(W)$: language model (statistical, N-grams, finite state networks)
- $P(A)$: acoustics (ignore during maximization)

The language model typically predicts a small set of next words based on knowledge of a finite number of previous words (N-grams) — leads to search space reduction.

THE CHOMSKY HIERARCHY

We can categorize language models by their generative capacity:

Type of Grammar	Constraints	Automata
Phrase Structure	A -> B	Turing Machine (Unrestricted)
Context Sensitive	aAb -> aBb	Linear Bounded Automata (N-grams, Unification)
Context Free	A -> B Constraint: A is a non-terminal. Equivalent to: A -> w A -> BC where "w" is a terminal; B,C are non-terminals (Chomsky normal form)	Push down automata (JSGF, RTN, Chart Parsing)
Regular	A -> w A -> wB (Subset of CFG)	Finite-state automata (Network decoding)

- CFGs offer a good compromise between parsing efficiency and representational power.
- CFGs provide a natural bridge between speech recognition and natural language processing.

N-GRAM LANGUAGE MODELS

Consider a word sequence $\boldsymbol{W} = w_1 w_2 w_3 \ldots w_n$. The probability of this word sequence can be decomposed as follows:

$$
\begin{aligned}
P(\boldsymbol{W}) &= P(w_1 w_2 w_3 \ldots w_n) \\
&= P(w_1) P(w_2 | w_1) P(w_3 | w_1, w_2) \ldots P(w_n | w_1, w_2, \ldots, w_{n-1}) \\
&= \prod_{i=1}^{n} P(w_i | w_1, w_2, \ldots, w_{i-1})
\end{aligned}
$$

The choice of w_i thus depends on the history, which we define as the preceding $i-1$ words.

Clearly, estimating $P(w_i | w_1, w_2, \ldots, w_{i-1})$ for every unique history is prohibitive. Why?

A practical approach is to assume this probability depends only on an equivalence class:

$$
\begin{aligned}
P(\boldsymbol{W}) &= \prod_{i=1}^{n} P(w_i | w_1, w_2, \ldots, w_{i-1}) \\
&= \prod_{i=1}^{n} P(w_i | \Phi(w_1, w_2, \ldots, w_{i-1}))
\end{aligned}
$$

There are three obvious simplifications we can make:

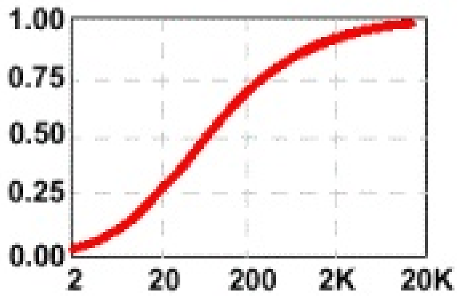
- Unigram: $\Phi(w_1, w_2, \ldots, w_{i-1}) = \phi$.
- Bigram: $\Phi(w_1, w_2, \ldots, w_{i-1}) = w_{i-1}$
- Trigram: $\Phi(w_1, w_2, \ldots, w_{i-1}) = w_{i-1}, w_{i-2}$

Of course, we can also merge histories based on linguistic considerations (e.g., grouping all nouns that describe animals, grouping all articles). What might be the advantages of doing this?

N-GRAM DISTRIBUTIONS FOR A CONVERSATIONAL SPEECH (SWITCHBOARD) CORPUS

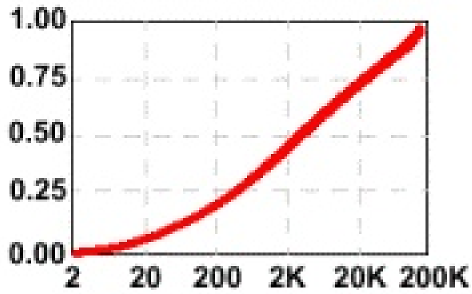
Unigrams (SWB):

- Most Common: I, and, the , you, a
- Rank-100: she, an, going
- Least Common: Abraham, Alastair, Acura



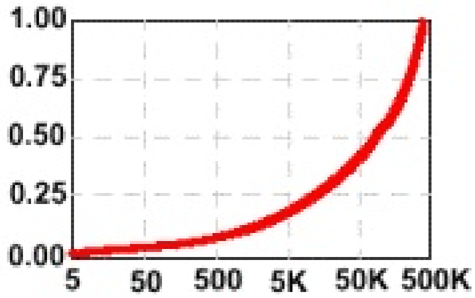
Bigrams (SWB):

- Most Common: “you know”, “yeah S!”, “!S um-hum”, “I think”
- Rank -100: “do it”, “that we”, “don’t think”
- Least Common: “raw fish”, “moisture content, “Reagan Bush”



Trigrams (SWB):

- Most Common: “!S um-hum S!”, “a lot of”, “I don’t know”
- Rank-100: “it was a”, “you know that”
- Least Common: “you have parents”, “you seen Brooklyn”



How what can measure the complexity of a language model? What is wrong with using the average branching factor?

Consider a word sequence $W = w_1 w_2 w_3 \ldots w_n = w_1^n$ as a random process. The entropy of this process is:

$$H(W) = -\lim_{n \to \infty} \frac{1}{n} E[\log(P(w_1^n))]$$
$$= -\lim_{n \to \infty} \frac{1}{n} \sum_{w_1^n} P(w_1^n) \log(P(w_1^n))$$

For an ergodic source, we can use a temporal average:

$$H(W) = -\lim_{n \to \infty} \frac{1}{n} \log(P(w_1^n))$$

Of course, we must estimate these probabilities from the training data:

$$\hat{H}(W) = -\lim_{n \to \infty} \frac{1}{n} \log(\hat{P}(w_1^n))$$

Jelinek showed that $\hat{H}(W) \geq H(W)$ if W is ergodic.

We can define **perplexity** as:

$$PP(W) = 2^{\hat{H}(W)} \approx \frac{1}{\sqrt[n]{\hat{P}(w_1^n)}}$$

Note that if all words are equally likely, and there are L words in the vocabulary:

$$PP(W) = 2^{\log_2 L} = L$$

We can define the **training-set perplexity** as a measure of how the training set fits the language model. Similarly, we can define a **test-set perplexity** as the perplexity computed over the test set. It can be interpreted as the inverse of the (geometric) average probability assigned to each word in the test set.

PERFORMANCE VS. PERPLEXITY

- Though perplexity is not the best measure for task complexity, it provides some useful insights:

Corpus	Vocabulary Size	Perplexity	Word Error Rate
TI Digits	11	11	~0.0%
OGI Alphadigits	36	36	8%
Resource Management (RM)	1,000	60	4%
Air Travel Information Service (ATIS)	1,800	12	4%
Wall Street Journal	20,000	200 - 250	15%
Broadcast News	> 80,000	200 - 250	20%
Conversational Speech	> 50,000	100 - 150	30%

- Acoustic confusibility of highly probable and interchangeable words most often dominates performance.
- WER ~= $-12.37 + 6.48*\log_2(\text{Perplexity})$ [William Fisher, NIST, May 2000]