Collecting, Correcting Speech Errors

Mark Johnson
Brown University

Joint work with Eugene Charniak and Matt Lease

Supported by NSF grants IIS 9720368 and IIS0095940

March, 2005
Talk outline

• What are speech repairs, and why are they interesting?
• A noisy channel model of speech repairs
• Training and evaluating the model of speech errors
• Reranking using machine-learning techniques

Rough copy dependencies, context-free and tree adjoining grammars

• A novel model of interpreting ill-formed input – combines two very different kinds of structures

RT04F evaluation
Speech errors in (transcribed) speech

• Speech errors in (transcribed) speech

  • Restarts and repairs

  • Filled pauses

  • Parentheticals

  • "Ungrammatical" constructions

  • But, you know, I was reading the other day.

  • I think it’s, uh, refreshing to see the, uh, support.

  • Why didn’t he, why didn’t she stay at home?

  • I want a flight to Boston, uh, to Denver on Friday.
Why focus on speech repairs?

• Filled pauses are easy to recognize (in transcripts at least)
• Parentheticals are handled by current parsers fairly well
• Ungrammatical constructions aren’t necessarily fatal
• Statistical parsers learn constructions in training corpus

... but speech repairs warrant special treatment, since the best parsers badly misanalyse them...
Statistical models of language are incredibly useful!

Early statistical models focused on dependencies between adjacent words (n-gram models) and most lexicalized (capture some semantic dependencies) and most are robust.

Probabilities estimated from real corpora.

If model permits every word sequence to occur with non-zero probability, model is robust.

Probability distinguishes “good” from “bad” sentences. These simple models work surprisingly well because they are lexicalized model is robust.

Probabilities estimated from real corpora.

$\leftarrow$ the $\leftarrow$ man $\leftarrow$ in $\leftarrow$ the $\leftarrow$ hat $\leftarrow$ drinks $\leftarrow$ red $\leftarrow$ wine $\leftarrow$ red

(adjacent words (n-gram models))

Early statistical models focused on dependencies between n-gram models.

Statistical regularities are incredibly useful.

Statistics models of language.
Most probable tree is "best guess" at correct syntactic structure.

Probability of a tree is the product of the probabilities of its rules.

Rules are associated with probabilities.

The diagram represents a probabilistic context-free grammar for the sentence "the man drinks red wine."
Head to head dependencies

Backoff and smoothing are central issues

Lexicalization captures a wide variety of syntactic (and semantic)

Rules:

Np drinks

S

...
The structure of repairs

- Repairs are not always copies
- Repairs are typically short
- The Reparandum is often a "rough copy" of the Repair
- The Interregnum is usually lexically and prosodically marked, but can be empty
- The Reparandum is often not a syntactic phrase

... and you get, uh, you can get a system...
But Charniak's parser never finds any EDITED node!

(Interrogation and repair are also annotated)

Each reparandum is indicated by an EDITED node

The Switchboard treebank contains the parse trees for 1M words of spontaneous telephone conversations

The Switchboard treebank contains the parse trees for 1M words of spontaneous telephone conversations
... may be hard to formalize ... but without a good model of "conceptual representation", this... starts incrementally generating syntax and phonology again... and "backs up", i.e., partially deconstructs syntax and phonology, recognizes that what is said doesn't mean what was intended, ... Speaker incrementally generates syntax and phonology, ... Speaker generates intended "conceptual representation"... and you get, uh, you get a system...
Approximating the “true model” (1)

- Dependencies are very different to those in “normal” language
- Reparandum results from attempt to generate Repair structure
  - Intended to say
  - Tree with reparandum and interregnum excised is what speaker

Approximate semantic representation by syntactic structure

Approximating the “true model” (1)
use model of normal language to interpret ill-formed input

- probability
- after correcting the error, what’s left should have high probability

String with reparandum and interregnum excised is well-formed

- involves crossing (rather than nested) dependencies
- Repair string is “rough copy” of Repair string

Use Repair string as approximation to intended meaning

- I want a flight to Boston, uh, I mean, to Denver on Friday

Approximating the “true model” (2)
Helical structure of speech repairs

Reparandum

Interregnum

Reparandum

... a flight to Boston, uh, I mean, I mean, I mean, I mean, to Denver on Friday...
The Noisy Channel Model

\[
\frac{(n)p}{(x)p(x|n)} = (n|x)p
\]

Bayes rule describes how to invert the channel

Noisy channel models combines two different submodels

- Noisy channel model $P(U|X)$
- Source model $P(X)$ (statistical parser)

... and you get... you can get a system... and you get, you can get a system...

Source signal $x$

Noisy signal $u$
The channel model

Reparandum is "rough copy" of repair

unigram model of interregnum phrases ⇐

only 62 different phrases appear in interregnum (uh, I mean)

• a flight:flight: to:Boston:∅:I:mean to:Denver:Denver

• Channel model is a transducer producing source:output pairs

I want a flight to Boston, uh, I mean, to Denver on Friday
CFGs generate nested dependencies between a string $w$ and its reverse $w_R$. 

\[(1)\]
CFGs generate nested dependencies between a string $w$ and its reverse $w_R$. (2)
CFGs generate nested dependencies between a string \( w \) and its reverse \( w^R \).
CFGs generate nested dependencies between a string $w$ and its reverse $w_R$. 

\[
\begin{array}{c}
\text{CFGs generate } w_R \\
\end{array}
\]
TAGS generate web dependencies (1)
TAGS generate dependencies (2)
TAGS generate dependencies (3)
TAGS generate \( \text{dependencies} \)
Derivation of a flight... (1)
Derivation of a flight...
Derivation of a flight to Boston on Friday.

I mean to go to Denver on Friday.
Derivation of a flight ... (4)
Derivation of a flight (5)
Derivation of a flight... (6)
Derivation of a flight... (7)
Derivation of a flight . . . (8)
Derivation of a flight

(9)

I mean to Denver on Friday.

32
Derivation of a flight (10)
Derivation of a flight \( \ldots \) (11)
Training data (1)

10K (0.8%), repair 53K (4%), too complicated 24K (1.8%)

Number of training words: reparandum 50K (3.8%), interregnum

31K repairs, average repair length 1.6 words

5.4% of words are in a reparandum

Punctuation and partial words ignored

Trained on Switchboard files sw[23]*.dps (1.3M words)

Switchboard corpus annotates reparandum, interregnum, and repair

... a flight to Boston, uh, I mean, to Denver on Friday ...

Training data (1)
Training data (2)

148 of 352 substitutions (42%) in heldout are not in training

2.9K (5%) are substitutions with same POS

- 5.6K (10%) are substitutions
- 9K (16%) are deletions
- 7K (12%) are insertions
- 3.5K (62%) are identities

Of the 57K alignments in the training data:
- 35K (62%) are identities
- 7K (12%) are insertions
- 9K (16%) are deletions
- 2.9K (5%) are substitutions with same POS

Reparandum and repair word-aligned by minimum edit distance

Reparandum
Repair
Interregnum

... a flight to Boston, uh, I mean, to Denver on Friday

Training data (2)
Estimating the channel model.

Channel model is defined in terms of several simpler distributions:

- \( P_{\text{tomorrow} | \text{Boston}, \text{Denver}} \): Probability that next reparandum word is tomorrow.
- \( P_{\text{m} | \text{Boston}, \text{Denver}} \): Probability of \( m \) after reparandum Boston and repair Denver, where \( m \in \{ \text{copy}, \text{substitute}, \text{insert}, \text{delete}, \text{end} \} \).
- \( P_{\text{repair} | \text{flight}} \): Probability of a repair starting after flight.

I want a flight to Boston, uh, I mean, to Denver on Friday.
Estimated repair start probabilities
Don’t know how to efficiently search for best analysis using parser LM ⇒ find 25-best hypothesized sources for each sentence using a simpler bigram LM

• Use them as features in a machine learning algorithm
  – Add them (noisy channel model)
  – Add them probabilities

Two ways of combining channel and language model log probabilities:

• Calculate probability of each hypothesized source using parsing LM

  simpler bigram LM

• Find 25-best hypothesized sources for each sentence using a

LM parser

(1) Implementation details (1)
MaxEnt reranker

Parses and probabilities for source hypotheses

Parsing language model

25 highest scoring source hypotheses

Noisy channel model with bigram LM

Input string

Most likely source hypotheses

Implementation details (2)
Evaluation of model's performance

<table>
<thead>
<tr>
<th>Model Configuration</th>
<th>f-score</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>NCM + bigram LM</td>
<td>0.75</td>
<td>0.45</td>
</tr>
<tr>
<td>NCM + parser LM</td>
<td>0.81</td>
<td>0.35</td>
</tr>
<tr>
<td>MaxEnt reranker using NCM + parser LM</td>
<td>0.87</td>
<td>0.25</td>
</tr>
<tr>
<td>MaxEnt reranker alone</td>
<td>0.78</td>
<td>0.38</td>
</tr>
</tbody>
</table>

Evaluated on an unseen portion of Switchboard corpus

-f-score is a geometric average of edited words precision and recall (bigger is better)

Error rate is the number of word errors made divided by number of true edited words
RT04F competition

The document outlines a deterministic SU segmentation algorithm, with a focus on noisy channel models and parser-based language models. The process includes:

1. Input words and IP probs from SRI, ICSI, and UW.
2. Segmentation of input words into SUs.
3. 25 best edit hypotheses generated.
4. Parses and string probabilities for each edit hypothesis.
5. Best edit hypotheses selected by MaxEnt reranker.
6. Deterministic FW and IP rule application.
7. EW, FW, and IP labels for input words.

Test material was unsegmented speech, and Meta-data extraction was performed.

The input words and IP probs from SRI, ICSI, and UW were used.
<table>
<thead>
<tr>
<th>Task/Task</th>
<th>Oracle words</th>
<th>ASR words</th>
<th>Error rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>RT04F evaluation results</td>
<td>55.9</td>
<td>40.0</td>
<td>33.7</td>
</tr>
<tr>
<td></td>
<td>28.6</td>
<td>76.3</td>
<td>64.1</td>
</tr>
</tbody>
</table>

- Interruption point detection combined these two models.
- Filler word detection used deterministic rules.
- EDITE word detection used noisy channel reranker.
Darpa runs a competitive evaluation (RT04) of speech understanding systems. Our system was not designed to deal with the RT04 data. OCR word detection was one task in this evaluation.

<table>
<thead>
<tr>
<th></th>
<th>Error Rate on Dev2 Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full model</td>
<td>0.525</td>
</tr>
<tr>
<td>Repair model</td>
<td>0.567</td>
</tr>
<tr>
<td>Parsing model</td>
<td>0.541</td>
</tr>
<tr>
<td>Prosodic features</td>
<td>0.541</td>
</tr>
<tr>
<td>ASR words</td>
<td>0.722</td>
</tr>
<tr>
<td>Oracle words</td>
<td>0.790</td>
</tr>
</tbody>
</table>

Evaluation of models' performance
Conclusion and Future Work

• Syntactic parsers make good language models
• Grammars are useful for lots of things besides syntax!
• Noisy channel model can combine very different kinds of models

Novel way of modeling robust language comprehension — can exploit prosodic and other kinds of information

Modern machine learning techniques are very useful

– a TAG model of "rough copy" dependencies in speech repairs
– a lexicalized CFG model of syntactic structure
– a lexicalized CFG model of syntactic structure

Performs well in practice

Noisy channel model can combine very different kinds of models

Grammar are useful for lots of things besides syntax!

Syntactic parsers make good language models