Positive Effects of Redundant Descriptions in an Interactive Semantic Speech Interface

Lane Schwartz, Luan Nguyen, Andrew Exley, William Schuler
University of Minnesota

June 24, 2011
Motivation

NSF project: build working interactive model of speech/language processing

(parsing, word recognition depend on semantics, pragmatics in context)
Motivation

NSF project: build working interactive model of speech/language processing (parsing, word recognition depend on semantics, pragmatics in context)

- **Cognitive appeal:** Tanenhaus et al. ’95 eye-tracking evidence
  People interactively constrain search through interpretation in context
Motivation

NSF project: build working interactive model of speech/language processing (parsing, word recognition depend on semantics, pragmatics in context)

- **Cognitive appeal:** Tanenhaus et al. ’95 eye-tracking evidence
  People interactively constrain search through interpretation in context

- **Practical appeal:** context-dependent speech interfaces

To artificial agent in ‘content creation’ domain:
1. ‘Add new folder *coling*’ (fix pronunciation?)
2. ‘Go to the *coling* folder and add new item *semrec*’ (fix pronunciation?)
3. ...
4. ‘Select the *semrec* in the *coling* folder’ (recognition should be reliable)

Interface uses context to improve recognition, in lieu of training corpus
Motivation

NSF project: build working interactive model of speech/language processing (parsing, word recognition depend on semantics, pragmatics in context)

- **Cognitive appeal:** Tanenhaus et al. ’95 eye-tracking evidence
  People interactively constrain search through interpretation in context

- **Practical appeal:** context-dependent speech interfaces

  To artificial agent in ‘content creation’ domain:
  1. ‘Add new folder *coling*’ (fix pronunciation?)
  2. ‘Go to the *coling* folder and add new item *semrec*’ (fix pronunciation?)
  3. ...
  4. ‘Select the *semrec* in the *coling* folder’ (recognition should be reliable)

Interface uses context to improve recognition, in lieu of training corpus

**This talk:** extended model allows redundancy to improve accuracy (only one semrec, but similar to sentry/timrec/... so add ‘in coling’)

Schwartz, Nguyen, Exley, and Schuler

Redundant Descriptions in Interactive Speech Interpreter
Interactive semantics: Hierarchic Hidden Markov Model (Murphy, Paskin’01)

Elements hold hypoth. stacked-up **incomplete constituents**, dep. on parent
(incomplete constituent: e.g. $S/VP = \text{sentence lacking verb phrase to come}$)
Probabilistic Time-Series Model

Interactive semantics: Hierarchic Hidden Markov Model (Murphy,Paskin’01)

Elements hold hypoth. stacked-up **incomplete constituents**, dep. on parent
(incomplete constituent: e.g. $S/VP = $ sentence lacking verb phrase to come)

Hypothesized mem elements generate **observations**: words / acoust. features
Interactive semantics: Hierarchic Hidden Markov Model (Murphy, Paskin’01)

Elements in memory store may be composed (reduced) w. element above
Probability depends on antecedent vars (e.g. Det, Noun reduce to NP)
Probabilistic Time-Series Model

Interactive semantics: Hierarchic Hidden Markov Model (Murphy,Paskin’01)

Non-reduced elements carry forward or transition (e.g. NP becomes S/VP)
Probabilistic Time-Series Model

Interactive semantics: Hierarchic Hidden Markov Model (Murphy, Paskin’01)

Non-reduced elements carry forward or transition (e.g. NP becomes S/VP)
Reduced elements may be expanded again (e.g. S/VP expands to Verb)
Probabilistic Time-Series Model

Interactive semantics: Hierarchic Hidden Markov Model (Murphy, Paskin’01)

Non-reduced elements carry forward or transition (e.g. NP becomes S/VP)
Reduced elements may be expanded again (e.g. S/VP expands to Verb)
Process continues through time
Probabilistic Time-Series Model

Interactive semantics: Hierarchic Hidden Markov Model (Murphy, Paskin’01)

Alternate hypotheses (memory store configurations) compete w. each other:

$$\hat{s}_{1..T}^{1..D} \overset{\text{def}}{=} \arg \max_{s_{1..T}^{1..D}} \prod_{t=1}^{T} P_{\Theta_{LM}}(s_t^{1..D} | s_{t-1}^{1..D}) \cdot P_{\Theta_{OM}}(o_t | s_t^{1..D})$$
Probabilistic Time-Series Model

Interactive semantics: Hierarchic Hidden Markov Model (Murphy, Paskin’01)

\[
\begin{align*}
\Pr_{\text{LM}}(s_{t}^{1..D} | s_{t-1}^{1..D}) &= \sum_{r_{t}^{1..D}} \Pr_{\text{Reduce}}(r_{t}^{1..D} | s_{t-1}^{1..D}) \cdot \Pr_{\text{Shift}}(s_{t}^{1..D} | r_{t}^{1..D} s_{t-1}^{1..D}) \\
&\overset{\text{def}}{=} \sum_{r_{t}^{1..D}} \prod_{d=1}^{D} \Pr_{\rho}(r_{t}^{d} | r_{t}^{d+1} s_{t-1}^{d} s_{t-1}^{d-1}) \cdot \Pr_{\sigma}(s_{t}^{d} | r_{t}^{d+1} r_{t}^{d} s_{t-1}^{d} s_{t-1}^{d-1})
\end{align*}
\]
Add interactive semantics — simply factor HHMM states:

— factor $r, s$ into interdependent syntactic ($q/f$) and referential ($e$) states:
Add interactive semantics — simply factor HHMM states:

— factor $r, s$ into interdependent syntactic ($q/f$) and referential ($e$) states:

  ▶ incomplete syntactic states: e.g. $q = S/VP$ (with $f$ as a reduce flag)
Add interactive semantics — simply factor HHMM states:

— factor \( r, s \) into interdependent syntactic (\( q/f \)) and referential (\( e \)) states:
  
  - **incomplete syntactic states**: e.g. \( q = S/VP \) (with \( f \) as a reduce flag)
  - **incomplete referential states**: e.g. \( e = \{ i_{\text{coling}}, i_{\text{naacl}} \} \) (entity set/class)
Sequences of memory stores correspond directly to familiar phrase structure:
(trees from Penn Treebank, modified to featurize empty categories)

Correspondence requires flatter, more memory-efficient representation...
‘Right-corner transform’ map right-embedded sequence $\rightarrow$ left-embedded seq. (allows new constituents to be immediately composed)
Right-Corner Transform

Transform is simple — **three cases** on right-embedded sequence:
($\eta, \mu$ are paths of 0:left/1:right)

**Beginning case:**

\[
\begin{array}{c}
\begin{array}{c}
A_\eta \\
A_{\eta.0} & A_{\eta.1}
\end{array}
\end{array}
\Rightarrow
\begin{array}{c}
\begin{array}{c}
A_\eta / A_{\eta.1} \\
A_{\eta.0}
\end{array}
\end{array}
\]

(Left child $\rightarrow$ top/right)
Right-Corner Transform

Transform is simple — **three cases** on right-embedded sequence:
(η, µ are paths of 0:left/1:right)

**Beginning case:**
\[
\begin{align*}
A_\eta &\quad \Rightarrow \quad A_\eta/1_A_{\eta.1} \\
A_{\eta.0} &\quad A_{\eta.1} \\
\alpha &\quad \beta
\end{align*}
\]
(Left child \(\rightarrow\) top/right)

**Middle case:**
\[
\begin{align*}
A_\eta &\quad \Rightarrow \quad A_\eta/1_A_{\eta.\mu.1} \\
A_{\eta.\mu.0} &\quad A_{\eta.\mu.1} \\
\alpha &\quad \beta
\end{align*}
\]
(Left child \(\rightarrow\) top/right)

Schwartz, Nguyen, Exley, and Schuler
Redundant Descriptions in Interactive Speech Interpreter
Right-Corner Transform

Transform is simple — **three cases** on right-embedded sequence:

$(\eta, \mu$ are paths of 0:left/1:right)

**Beginning case:**

$$
\begin{align*}
A_\eta & \quad \Rightarrow \\
A_{\eta\cdot0} & \quad A_{\eta\cdot1} & \quad A_\eta/ A_{\eta\cdot1} \\
\alpha & \quad \beta & \quad (\text{left child } \rightarrow \text{top/right})
\end{align*}
$$

**Middle case:**

$$
\begin{align*}
A_\eta & \quad \Rightarrow \\
A_{\eta\cdot0} & \quad A_{\eta\cdot\mu} & \quad A_{\eta\cdot\mu\cdot0} \\
\alpha & \quad \beta \quad \gamma & \quad A_\eta/ A_{\eta\cdot\mu} \\
\beta & \quad \gamma & \quad (\text{left child } \rightarrow \text{top/right})
\end{align*}
$$

**Ending case:**

$$
\begin{align*}
A_\eta & \quad \Rightarrow \\
A_{\eta\cdot0} & \quad A_{\eta\cdot\mu} & \quad A_{\eta\cdot\mu} \\
\alpha & \quad a_{\eta\cdot\mu} & \quad (\text{left child } \rightarrow \text{top/right})
\end{align*}
$$
Align levels to a grid, to train HHMM:

Time-order parsing based on familiar phrase structure grammar rules
Interactive Interpretation

Add interactive meaning:

- last year: first-order objects (individual files/directories)
  runs in real time w. 4000 individuals, 4000 words
Add interactive meaning:

- last year: first-order objects (individual files/directories) runs in real time w. 4000 individuals, 4000 words
- now: redundancy requires second-order objects (sets of individuals) runs in real time w. 100 individuals, 1000 words
Add interactive meaning:

- last year: first-order objects (individual files/directories)
  runs in real time w. 4000 individuals, 4000 words

- now: redundancy requires second-order objects (sets of individuals)
  runs in real time w. 100 individuals, 1000 words

  syntax, semantics defined w. familiar grammar rules, set operations
  (someday, by user – fully extensible language model)
Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $\mathcal{E}$
Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $\mathcal{E}$
- assume functions true or false over individuals:
  \[
  \lambda x. \text{file}(x) \\
  \lambda y. \lambda x. \text{contain}(x, y)
  \]
Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $\mathcal{E}$
- assume functions true or false over individuals:
  
  $\lambda x. \text{file}(x)$
  $\lambda y. \lambda x. \text{contain}(x, y)$

- functions define transitions/operators from set to set:
  
  $\text{ExeFile} : \lambda X. \{ x \mid x \in X \land \text{executable}(x) \}$
  $\text{Contain} : \lambda X. \{ y \mid x \in X \land \text{contain}(x, y) \}$
  $\text{Contain'} : \lambda Y. \lambda X. \{ x \mid x \in X \land y \in Y \land \text{contain}(x, y) \}$
Interactive Interpretation – transitions

Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $\mathcal{E}$
- assume functions true or false over individuals:
  - $\lambda x. file(x)$
  - $\lambda y. \lambda x. contain(x, y)$
- functions define transitions/operators from set to set:
  - $\text{ExeFile} : \lambda X. \{ x \mid x \in X \land \text{executable}(x) \}$
  - $\text{Contain} : \lambda X. \{ y \mid x \in X \land \text{contain}(x, y) \}$
  - $\text{Contain'} : \lambda Y. \lambda X. \{ x \mid x \in X \land y \in Y \land \text{contain}(x, y) \}$
Interactive Interpretation – transitions

Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $E$
- assume functions true or false over individuals:
  \[
  \lambda x. \text{file}(x) \\
  \lambda y. \lambda x. \text{contain}(x, y)
  \]
- functions define transitions/operators from set to set:
  \[
  \text{ExeFile} : \lambda X. \{ x \mid x \in X \land \text{executable}(x) \} \\
  \text{Contain} : \lambda X. \{ y \mid x \in X \land \text{contain}(x, y) \} \\
  \text{Contain'} : \lambda Y. \lambda X. \{ x \mid x \in X \land y \in Y \land \text{contain}(x, y) \}
  \]
- operators can be associated with rules
  \[
  \text{RC} \rightarrow (\text{Contain}) \text{ containing NP (Contain') } \\
  \text{NP} \rightarrow (\text{ExeFile}) \text{ executables}
  \]
Interactive Interpretation – transitions

Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $E$
- assume functions true or false over individuals:
  \[ \lambda x. \text{file}(x) \]
  \[ \lambda y. \lambda x. \text{contain}(x, y) \]
- functions define transitions/operators from set to set:
  \[ \text{ExeFile} : \lambda X. \{ x \mid x \in X \land \text{executable}(x) \} \]
  \[ \text{Contain} : \lambda X. \{ y \mid x \in X \land \text{contain}(x, y) \} \]
  \[ \text{Contain}' : \lambda Y. \lambda X. \{ x \mid x \in X \land y \in Y \land \text{contain}(x, y) \} \]
- operators can be associated with rules, traversed/composed in order:
  \[ \text{RC} \rightarrow (\text{Contain}) \text{ containing NP (Contain')} \]
  \[ \text{NP} \rightarrow (\text{ExeFile}) \text{ executables} \]
  \[ \{ d_1 d_2 d_3 \} \circ \text{Contain} = \{ f_2 f_3 \} \ldots \] (start w. directories, get contents)
Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $E$
- assume functions true or false over individuals:
  $\lambda x. \text{file}(x)$
  $\lambda y. \lambda x. \text{contain}(x, y)$
- functions define transitions/operators from set to set:
  $\text{ExeFile} : \lambda X. \{ x \mid x \in X \land \text{executable}(x) \}$
  $\text{Contain} : \lambda X. \{ y \mid x \in X \land \text{contain}(x, y) \}$
  $\text{Contain'} : \lambda Y. \lambda X. \{ x \mid x \in X \land y \in Y \land \text{contain}(x, y) \}$
- operators can be associated with rules, traversed/composed in order:
  $\text{RC} \rightarrow (\text{Contain})$ containing NP $(\text{Contain'})$
  $\text{NP} \rightarrow (\text{ExeFile})$ executables
  $\{ d_1 d_2 d_3 \} \circ \text{Contain} \circ \text{ExeFile} = \{ f_2 \} \ldots$ (get executable contents)
Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $\mathcal{E}$
- assume functions true or false over individuals:
  $\lambda x. \text{file}(x)$
  $\lambda y. \lambda x. \text{contain}(x, y)$

- functions define transitions/operators from set to set:
  $\text{ExeFile} : \lambda X. \{ x \mid x \in X \land \text{executable}(x) \}$
  $\text{Contain} : \lambda X. \{ y \mid x \in X \land \text{contain}(x, y) \}$
  $\text{Contain}' : \lambda Y. \lambda X. \{ x \mid x \in X \land y \in Y \land \text{contain}(x, y) \}$

- operators can be associated with rules, traversed/composed in order:
  $\text{RC} \rightarrow (\text{Contain})$ containing $\text{NP}$ ($\text{Contain}'$)
  $\text{NP} \rightarrow (\text{ExeFile})$ executables
  $\{ d_1 d_2 d_3 \} \circ \text{Contain} \circ \text{ExeFile} \circ \text{Contain}' = \{ d_2 s_1 \} \ldots$ (containers)
Interactive Interpretation – transitions

Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $\mathcal{E}$
- assume functions true or false over individuals:
  \[ \lambda x. \text{file}(x) \]
  \[ \lambda y. \lambda x. \text{contain}(x, y) \]
- functions define transitions/operators from set to set:
  \[ \text{ExeFile} : \lambda X. \{ x \mid x \in X \land \text{executable}(x) \} \]
  \[ \text{Contain} : \lambda X. \{ y \mid x \in X \land \text{contain}(x, y) \} \]
  \[ \text{Contain}' : \lambda Y. \lambda X. \{ x \mid x \in X \land y \in Y \land \text{contain}(x, y) \} \]
- operators can be associated with rules, traversed/composed in order:
  \[ \text{RC} \rightarrow (\text{Contain}) \text{ containing NP (Contain') } \]
  \[ \text{NP} \rightarrow (\text{ExeFile}) \text{ executables } \]
  \[ \{ d_1 d_2 d_3 \} \circ \text{Contain} \circ \text{ExeFile} \circ \text{Contain}'(d_1 d_2 d_3) = \{ d_2 \} \ (\lambda X : \cap) \]
Interactive Interpretation – transitions

Interactive interpretation defined using $e \rightarrow e$ transitions in HHMM

- assume finite domain of individuals $\mathcal{E}$
- assume functions true or false over individuals:
  $\lambda x.\text{file}(x)$
  $\lambda y.\lambda x.\text{contain}(x, y)$
- functions define transitions/operators from set to set:
  $\text{ExeFile} : \lambda X.\{x \mid x \in X \land \text{executable}(x)\}$
  $\text{Contain} : \lambda X.\{y \mid x \in X \land \text{contain}(x, y)\}$
  $\text{Contain'} : \lambda Y.\lambda X.\{x \mid x \in X \land y \in Y \land \text{contain}(x, y)\}$
- operators can be associated with rules, traversed/composed in order:
  $\text{RC} \rightarrow (\text{Contain})$ containing NP $(\text{Contain'})$
  $\text{NP} \rightarrow (\text{ExeFile})$ executables
  $\{d_1d_2d_3\} \circ \text{Contain} \circ \text{ExeFile} \circ \text{Contain'}(d_1d_2d_3) = \{d_2\}$ $(\lambda X: \cap)$

Semantics now ‘ride along’ through transform as operator/transition chains
Right-Corner Transform on Operator Chains

Operator chains prior to transform (dot arcs show \( \lambda X \) dependencies):

\[
\{d_1, d_2, d_3, \ldots\} \quad \text{NP} \quad \{d_2\} \\
\{d_1, d_2, d_3\} \quad \text{NP} \quad \{f_2\} \\
\{f_2, f_3\} \quad \text{RC} \quad \{f_2\} \\
\{f_2\} \quad \text{NP} \quad \{f_2\} \\
\]

Directories

Containing

ExeFile

executables
Right-Corner Transform on Operator Chains

Operator chains following transform:

Directories

contain

executables

−○CONTAIN'

containing

Left legs turn into right legs, but operators keep order, structure
Right-Corner Transform on Operator Chains

Transformed operations aligned to time-series model:

Time-order interpretation from familiar grammar rules, set operations
Right-Corner Transform on Operator Chains

Transformed operations aligned to time-series model:

\[
\begin{align*}
&d=1 & \quad t=1 & \\
&d=2 & \quad t=2 & \quad NP & NP & \{\langle d_1, f_2 \rangle, \langle d_3, f_3 \rangle \} & \text{Contain} & \text{containing} & NP & NP & \{d_2\} & \text{ExeFile} & \text{executables} \\
&d=3 & \quad t=3 & \quad NP & NP & \{d_2\} & & & & & & & \\
\end{align*}
\]

Time-order interpretation from familiar grammar rules, set operations
Interpretations dynamically calculated, then used to rate hypotheses
(prob. based on denotation cardinality before and after each operation)
Evaluation: scalable to second-order denotations

Bounded model allows 2\textsuperscript{nd}order denotations in real time speech (13\% SER)

Restaurant domain: ‘select the glasses on chairs’
Evaluation: scalable to second-order denotations

Beam=100, Individuals=110, Lexicon=50 — 100 directives test:

<table>
<thead>
<tr>
<th>Subject</th>
<th>Sentence error rate</th>
<th>Corrected on 1(^{st}) retry</th>
<th>Corrected on 2(^{nd}) retry</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 / 20</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>2 / 20</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3 / 20</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4 / 20</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>2 / 20</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>13%</td>
<td>10</td>
<td>3</td>
</tr>
</tbody>
</table>
Evaluation: scalable to second-order denotations

Beam=100, Individuals=110, Lexicon=50 — 100 directives test:

<table>
<thead>
<tr>
<th>Subject</th>
<th>Sentence error rate</th>
<th>Corrected on 1\textsuperscript{st} retry</th>
<th>Corrected on 2\textsuperscript{nd} retry</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 / 20</td>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>2 / 20</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3 / 20</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4 / 20</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>2 / 20</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>13%</td>
<td>10</td>
<td>3</td>
</tr>
</tbody>
</table>

Beam=100, Individuals=110, Lexicon=1000 — 60 directives test:

<table>
<thead>
<tr>
<th>Subject</th>
<th>Sentence error rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 / 20</td>
</tr>
<tr>
<td>2</td>
<td>1 / 20</td>
</tr>
<tr>
<td>3</td>
<td>5 / 20</td>
</tr>
<tr>
<td>Total</td>
<td>13%</td>
</tr>
</tbody>
</table>
Evaluation: redundancy improves accuracy

Does interactive model let speaker be redundant to improve communication?

Monosyllabic domain: ‘select the seat [to the right of the rug]’
Evaluation: redundancy improves accuracy

Beam=100, Individuals=100, Lexicon=100 — 1000 directives test:

<table>
<thead>
<tr>
<th>Subject</th>
<th>Sentence error rate without redundancy</th>
<th>Sentence error rate with redundancy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>54 / 100</td>
<td>37 / 100</td>
</tr>
<tr>
<td>2</td>
<td>32 / 100</td>
<td>21 / 100</td>
</tr>
<tr>
<td>3</td>
<td>25 / 100</td>
<td>18 / 100</td>
</tr>
<tr>
<td>4</td>
<td>28 / 100</td>
<td>12 / 100</td>
</tr>
<tr>
<td>5</td>
<td>24 / 100</td>
<td>15 / 100</td>
</tr>
<tr>
<td>All</td>
<td>32.6%</td>
<td>20.6%</td>
</tr>
</tbody>
</table>

Natural model of using redundancy to ensure correct interpretation.
In summary: useful model of interactive comprehension!
Conclusion

In summary: useful model of interactive comprehension!
- Interactive interpretation with second-order referents (sets of indivs)
In summary: useful model of interactive comprehension!

- Interactive interpretation with second-order referents (sets of indivs)
- Runs in real time
In summary: useful model of interactive comprehension!
- Interactive interpretation with second-order referents (sets of indivs)
- Runs in real time
- Based on familiar notions of phrase structure, semantic composition
Conclusion

In summary: useful model of interactive comprehension!

- Interactive interpretation with second-order referents (sets of indivs)
- Runs in real time
- Based on familiar notions of phrase structure, semantic composition
- Interactive interpretation lets user be redundant to improve accuracy
Conclusion

Thank you!