Comp349 Spoken Language Dialogue Systems
Week 11: Semantics and Meaning

Robert Dale
Robert.Dale@mq.edu.au
Context: The Architecture of an SLDS

Speech Recognition → Language Understanding → Dialog Management → Database → Language Generation → Speech Synthesis

Speech Recognition

Language Understanding

Dialog Management

Database

Language Generation

Speech Synthesis
Today's Focus: From Word Sequence to Meaning

Phoneme Sequence via an Acoustic Model

Word Sequence via the Lexicon and a Language Model

Literal Interpretation

Pragmatic Interpretation

\[
\text{you're standing on my foot}
\]

\[
\exists x \exists y \text{hearer}(x) \land \text{foot}(y) \land \text{standing-on}(x,y)
\]

\[
\text{REQUEST(get-off(x,y))}
\]
But First: Some Notes on Writing Grammars

Common problems in the tutorial submissions:

• Always spell out words: you are recognizing speech, not typed out
• Think about the best and clearest factorisations of your rules
• Capture generalities at the right level
• Watch out for overgeneration
Today’s Agenda

- Some Last Words on Grammars and Tuning
- N-Best Processing
- Semantics: Deriving the Meaning of an Utterance
- The Assignment
How Do You Know Your Grammar Coverage is Sufficient?

• Crucial aspect: have you predicted the caller's responses correctly?

• Ways to deal with this:
  – Early in the design process: Wizard of Oz experiments
  – Initial deployment: release a pilot system
  – Full deployment: regular tuning cycles
What's Involved in Tuning

- Log data on when calls fail
  - At which states do things go wrong?
  - Easy, completely automatic
  - Used to create a Hotspot Report
- Record calls made to the system
  - Transcribe the caller utterances so they can be compared with what the recogniser heard
  - Expensive, requires manual labour
  - Used to create an Accuracy Report
HotSpot Reports

- Data provided per recognition state:
  - Did the caller hang up at this state?
  - Did the recogniser reject the utterance as unrecognizable?
  - Was there a No Speech Timeout?
  - Was there a Too Much Speech timeout?
  - Did the system abort at this state?
<table>
<thead>
<tr>
<th>Grammar</th>
<th>DestinationSuburb</th>
<th>MainFilter</th>
<th>NumberOfPassengers</th>
<th>PickUpDateTime</th>
<th>YesNo</th>
<th>YesNoFinalConfirm</th>
<th>YesNoReadyNow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>37795</td>
<td>70472</td>
<td>30214</td>
<td>4995</td>
<td>36201</td>
<td>26451</td>
<td>28592</td>
</tr>
<tr>
<td>% ABORTED</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% TMS</td>
<td>0.40%</td>
<td>0.00%</td>
<td>0.10%</td>
<td>1.10%</td>
<td>0.20%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% NST</td>
<td>2.10%</td>
<td>7.30%</td>
<td>0.80%</td>
<td>1.70%</td>
<td>1.20%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% Rejets</td>
<td>11.60%</td>
<td>8.10%</td>
<td>6.10%</td>
<td>3.10%</td>
<td>3.10%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% Hangups</td>
<td>6.60%</td>
<td>15.20%</td>
<td>4.70%</td>
<td>6.30%</td>
<td>3.10%</td>
<td>96.20%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% Rejects</td>
<td>0.00%</td>
<td>0.00%</td>
<td>7.30%</td>
<td>8.10%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.40%</td>
</tr>
<tr>
<td>% Hangups</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% ABORTED</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% TMS</td>
<td>0.40%</td>
<td>0.00%</td>
<td>0.10%</td>
<td>1.10%</td>
<td>0.20%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% NST</td>
<td>2.10%</td>
<td>7.30%</td>
<td>0.80%</td>
<td>1.70%</td>
<td>1.20%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% Rejets</td>
<td>11.60%</td>
<td>8.10%</td>
<td>6.10%</td>
<td>3.10%</td>
<td>3.10%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% Hangups</td>
<td>6.60%</td>
<td>15.20%</td>
<td>4.70%</td>
<td>6.30%</td>
<td>3.10%</td>
<td>96.20%</td>
<td>0.00%</td>
</tr>
<tr>
<td>% Rejects</td>
<td>0.00%</td>
<td>0.00%</td>
<td>7.30%</td>
<td>8.10%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.40%</td>
</tr>
<tr>
<td>% Hangups</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
</tbody>
</table>
Classifying Recognition Results

- **In Grammar**
  - The utterance provided by the caller is covered by the grammar
- **Out of Grammar**
  - The utterance provided by the caller is outside the scope of the grammar
But ... 

- You can't classify a user utterance as in-grammar or out-of-grammar unless you know what they said.
- What the recognizer posits as a hypothesis may not be what the caller actually said.
- So: to determine what is in-grammar and what is out-of-grammar, we need to manually transcribe utterances and compare the human transcriptions with what the system thought was said.
In-Grammar vs Out-of-Grammar

• Grammar:
  YesNo → yes | no

• Caller utterances:

<table>
<thead>
<tr>
<th>In-grammar</th>
<th>Out-of-grammar</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes, no</td>
<td>yep, yeah, sure, nope</td>
</tr>
</tbody>
</table>
Tuning via Transcriptions

.YesNo: Would you like fries with that?
[T] yes
[R] yes
[ca-in]
<action add-fries>
{Conf 75}
Tuning via Transcriptions

The prompt

YesNo: Would you like fries with that?
[T] yes
[R] yes
[ca-in]
<action add-fries>
{Conf 75}
Tuning via Transcriptions

The prompt

State or grammar name

. YesNo: Would you like fries with that?

[T] yes

[R] yes

[ca-in]

<action add-fries>

{Conf 75}
Tuning via Transcriptions

The prompt

State or grammar name

Human transcription

YesNo: Would you like fries with that?

[T] yes
[R] yes
[ca-in]
<action add-fries>
{Conf 75}
Tuning via Transcriptions

The prompt: 
State or grammar name: 
Human transcription: 
Recognizer hypothesis:

YesNo: Would you like fries with that?
[T] yes
[R] yes
[ca-in]
&action add-fries>
{Conf 75}
Tuning via Transcriptions

The prompt

State or grammar name

Human transcription

Recognizer hypothesis

Categorisation of recognition

YesNo: Would you like fries with that?

[T] yes

[R] yes

[ca-in]

<action add-fries>

{Conf 75}
Tuning via Transcriptions

The prompt: 
State or grammar name
Human transcription
Recognizer hypothesis
Categorisation of recognition
Semantic slot fill

YesNo: Would you like fries with that?

[T] yes
[R] yes
[ca-in]
<action add-fries>
{Conf 75}
The prompt: "YesNo: Would you like fries with that?"

- **State or grammar name**
- **Human transcription**
- **Recognizer hypothesis**
- **Categorisation of recognition**
- **Semantic slot fill**
- **Confidence Level**

**The Process:**
- [T] yes
- [R] yes
- [ca-in]
- `<action add-fries>`
- `{Conf 75}`
Five Possible Categorisations of a Recognition Hypothesis

- CA-in: Correct Accept, In-Grammar
- CR-out: Correct Reject, Out-of-Grammar
- FA-in: False Accept, In-Grammar
- FA-out: False Accept, Out-of-Grammar
- FR-in: False Reject, In Grammar
Classifying Recognition Results: CA-In

- CA-in: Correct Accept, In-Grammar
  - System correctly recognises something the caller said as being within the scope of the grammar
- So:
  - Caller's actual utterance (as determined by the human transcriber) is in grammar
  - System correctly recognized what the caller said
A Correct Accept

.DeliveryOrPic: Would you like delivery or pickup?
[T] pickup
[R] pickup
[ca-in] <action pickup> {Conf 75}
Classifying Recognition Results: FA-In

• FA-in: False Accept, In-Grammar
  – Caller says something in-grammar but it is recognised as something else in-grammar

• So:
  – What the caller said, according to the human transcriber, was in grammar
  – The system recognised the caller as saying something that was in grammar, but did not correctly recognise what the caller said
A False Accept that is In-Grammar

.SpellSurname: Please spell your surname for the pick-up order.

[T] s m i t h
[R] s m y p h
[fa-in] <action name> <name SMYPH> {Conf 68}
False Accept, In-Grammar

.PizzaSpec: What would you like for your second pizza?

[T] @hes@ super supreme
[R] large super supreme
[fa-in] <action pizzaSpec> <pizzaSize large>
False Accept, In-Grammar

.PizzaSpec: What pizza would you like?

[T] @hes@ one small hawaiian pizza

[R] @hes@ onions olives and bacon pizza

[fa-in] <action pizzaSpec>

<pizzaToppingCodes1 ON BO BA>
<pizzaToppings1 onion, blackOlives, bacon>
<pizzaVariety1 create-your-own>
<pizzaVarietyCode1 CY> {Conf 46}
False Accept, In-Grammar

.PickupTimeOK:  Is that pick-up time OK?

[T] yes
[R] nah

[fa-in] <action no> {Conf 72}
Classifying Recognition Results: FA-Out

- **FA-out:** False Accept, Out-of-Grammar
  - Caller says something out-of-grammar but it is recognised as something else in-grammar

- **So:**
  - According to the transcriber, what the caller said was out-of-grammar
  - The system incorrectly recognised what was said as something that was in-grammar
False Accept, Out-of-Grammar

.SuggestiveSel: Would you like 2 large pizzas, a garlic bread and a bottle of Pepsi from twenty four dollars and ninety five cents?

[T] no i want three
[R] no one pizza
[fa-out] <action numPizzas> <number 1> {Conf 48}
False Accept, Out-of-Grammar

.PizzaSpec: What would you like for your first pizza?
[T] how much is it
[R] @hes@ mushrooms
[fa-out]
<action pizzaSpec>
<pizzaToppingCodes1 MU>
<pizzaToppings1 mushrooms>
<pizzaVarietyCode1 CY>
<pizzaVariety1 create-your-own>
{Conf 54}
False Accept, Out-of-Grammar

.PickupTimeOK:  The pickup time at Newport is fifteen minutes. Is that OK?

[T] yes i suppose [fragment]
[R] yeah it is
[fa-out] <action yes> {Conf 60}
.PizzaBase: Which base would you like?
[T] thin and crispy and pan fried
[R] thin and crispy pizza

[fa-out]
<action pizzaSpec>
<pizzaBase thin>
<pizzaBaseCode T>
{Conf 50}
False Accept, Out-of-Grammar

.PizzaSpec: What would you like for your second pizza?
[T] shit [noise] super supreme
[R] thin super supreme
[fa-out] <action pizzaSpec> <pizzaBase thin>
An Accuracy Report

<table>
<thead>
<tr>
<th>Grammar</th>
<th>Total</th>
<th>IG</th>
<th>CA-in</th>
<th>FA-in</th>
<th>FR-in</th>
<th>OOG</th>
<th>CR-out</th>
<th>FA-out</th>
</tr>
</thead>
<tbody>
<tr>
<td>.DestinationSuburb</td>
<td>806</td>
<td>59.90%</td>
<td>74.90%</td>
<td>14.90%</td>
<td>10.10%</td>
<td>40.10%</td>
<td>44.90%</td>
<td>55.10%</td>
</tr>
<tr>
<td>.NumberOfPassengers</td>
<td>580</td>
<td>91.20%</td>
<td>88.70%</td>
<td>6.00%</td>
<td>5.30%</td>
<td>8.80%</td>
<td>47.10%</td>
<td>52.90%</td>
</tr>
<tr>
<td>.PickUpDateTime</td>
<td>67</td>
<td>77.60%</td>
<td>84.60%</td>
<td>15.40%</td>
<td>0.00%</td>
<td>22.40%</td>
<td>73.30%</td>
<td>26.70%</td>
</tr>
<tr>
<td>.ReadyToGoNow</td>
<td>513</td>
<td>88.70%</td>
<td>83.50%</td>
<td>14.30%</td>
<td>2.20%</td>
<td>11.30%</td>
<td>34.50%</td>
<td>65.50%</td>
</tr>
<tr>
<td>.YesNo</td>
<td>2358</td>
<td>88.30%</td>
<td>84.50%</td>
<td>12.20%</td>
<td>3.30%</td>
<td>11.70%</td>
<td>64.00%</td>
<td>36.00%</td>
</tr>
<tr>
<td>.YesNoSuburb</td>
<td>231</td>
<td>69.70%</td>
<td>82.00%</td>
<td>18.00%</td>
<td>0.00%</td>
<td>30.30%</td>
<td>0.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>Overall</td>
<td>4555</td>
<td>82.60%</td>
<td>83.70%</td>
<td>12.20%</td>
<td>4.10%</td>
<td>17.40%</td>
<td>47.50%</td>
<td>52.50%</td>
</tr>
</tbody>
</table>
What the Headings Mean

When Do You Tune?

- Typically two or three tuning phases after system deployment
- First tuning phase 1-2 weeks after deployment
- Second phase three months after deployment
- Third phase six months after deployment
Predicting User Responses

System: Are you ready to go now?
• yes
• yes yes
• yes yes I am
• right now
• absolutely
• absolutely yes
• we are yes
Unpredictable User Responses

in five minutes
ah i'd like to book for ten for ten thirty please
as soon as possible
no could you pick me up at um
no i want the taxi to pick us up at
no i'd like a taxi at eleven thirty please
um yes i need a station wagon
um maybe
ten minutes
straight away
sorry
ready to go now yes

yes i'm in a hurry
yes i'm going to rose avenue thank thank you
yes hurry up
yes eleven thirty
yes and its senior citizen
no i want to make a time booking please
no i want it for twelve noon today please
it's for ah quarter past eight tonight
hello
i am and i have a fifty dollar note please
OOG Responses for ‘What Suburb?’

<table>
<thead>
<tr>
<th>Response</th>
<th>Count</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>oxford street</td>
<td>14</td>
<td>1.34%</td>
</tr>
<tr>
<td>central railway</td>
<td>12</td>
<td>1.15%</td>
</tr>
<tr>
<td>taylor square</td>
<td>9</td>
<td>0.86%</td>
</tr>
<tr>
<td>chinatown</td>
<td>7</td>
<td>0.67%</td>
</tr>
<tr>
<td>no</td>
<td>7</td>
<td>0.67%</td>
</tr>
<tr>
<td>airport domestic</td>
<td>6</td>
<td>0.58%</td>
</tr>
<tr>
<td>i don't know</td>
<td>6</td>
<td>0.58%</td>
</tr>
<tr>
<td>sydney international airport</td>
<td>6</td>
<td>0.58%</td>
</tr>
<tr>
<td>airport international airport</td>
<td>6</td>
<td>0.58%</td>
</tr>
<tr>
<td>randwick racecourse</td>
<td>5</td>
<td>0.48%</td>
</tr>
<tr>
<td>cockle bay</td>
<td>4</td>
<td>0.38%</td>
</tr>
</tbody>
</table>

...
Grammar Development

- Aim to get most of the app right with good design: manipulate the user into saying things you will recognize
- Recognize that you will never get it completely right and that tuning will be required
- Start with small grammars, extend them on the basis of evidence from real calls
Today’s Agenda

• Some Last Words on Grammars and Tuning
• N-Best Processing
• Semantics: Deriving the Meaning of an Utterance
• The Assignment
Confidence Levels

• A recognizer typically returns a result along with a confidence level
  – |a large pepperoni pizza| 72% |

• You can use this information to avoid excessive confirmation dialogs: for example
  – if confidence > 80%
    then assume okay
  else if confidence > 70%
    then ask for confirmation
  else repeat question
  endif
N-Best Rankings

- A recognizer may return several results with different confidence levels:
  - one two seven five three five | 79% |
  - one two seven nine three five | 72% |
  - one two seven three five | 71% |
  
- You don't have to accept the recognizer's best guess.
Using Application Intelligence

- Review n-best rankings on the basis of the question asked
  - How many digits were required?
  - Is there an external check you can perform on the data?
  - Can you use frequency data?
  - Can you use caller specific information?
Examples of External Knowledge in N-Best Reranking

• Using the check digit in a credit card
• Checking that the day number in a month is valid
  – ‘30th February’ is probably a misrecognition
• Checking that a day name is consistent with the specified date
  – ‘Wednesday 25th October’ isn’t a day in 2007
• Checking that the specified destination city is not the same as the already-specified source city
Today’s Agenda

• Some Last Words on Grammars and Tuning
• N-Best Processing
• Semantics: Deriving the Meaning of an Utterance
  – What is Semantics?
  – Using Grammars to Return Semantic Values
  – Dialog Semantics via Multiple Returned Values
  – Meaning in Statistical Language Models
• The Assignment
What is Semantics?

• Semantics = Meaning

• Meaning is a representation of the content of an utterance, abstracted away from superficial variations

• Two aspects of meaning: sense and reference

• Sense = 'the semantic content' of an utterance

• Reference = the things referred to in the world
Sense and Reference

1. the guy who teaches the second half of Comp349
2. the lecturer for the second half of the Spoken Language Dialog Systems unit
3. the only Scotsman in the Department of Computing

1 and 2 are identical (or at least similar) in sense
1, 2 and 3 are identical in reference
## Language and the World

<table>
<thead>
<tr>
<th>The World</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concepts, classes of objects</td>
<td>Nouns</td>
</tr>
<tr>
<td>Objects/entities/things, either concrete or abstract</td>
<td>Noun phrases</td>
</tr>
<tr>
<td>Properties or qualities of objects</td>
<td>Adjectives</td>
</tr>
<tr>
<td>Events, states of affairs</td>
<td>Sentences, clauses, sometimes NPs</td>
</tr>
</tbody>
</table>
Semantics and Pragmatics

• Semantics = literal meaning
• Pragmatics = meaning in context
• Common examples:
  – Can you pass the salt?
  – It’s cold in here.
  – You’re standing on my foot.
  – His handwriting is good.
'Real' Semantics

- The cat sat on the mat

\[ \exists e \exists y \text{ event-type78}(e) \cap \text{PAST}(e) \cap \text{entity-type122}(x) \cap \text{entity-type157}(y) \cap \text{AGENT}(e, x) \cap \text{LOCATION}(e, \text{spatial-relationship12}(y)) \cap \text{SITTING}(e) \cap \text{PAST}(e) \cap \text{CAT}(x) \cap \text{MAT}(y) \cap \text{AGENT}(e, x) \cap \text{LOCATION}(e, \text{ON}(y)) \]
Why Semantics?

- Unambiguous representation
- Formal, well-defined
- General purpose, domain-independent
- Allows inference and reasoning
Today’s Agenda

• Some Last Words on Grammars and Tuning
• N-Best Processing
• Semantics: Deriving the Meaning of an Utterance
  – What is Semantics?
  – Using Grammars to Return Semantic Values
  – Dialog Semantics via Multiple Returned Values
  – Meaning in Statistical Language Models
• The Assignment
Restricting the Range of Inputs

- Strict Yes/No:
  YesNoStrict
  [ yes no ]

- Loose Yes/No:
  YesNoLoose
  [ yes yep yeah (sure thing) (no problem) no nope (no way) ]
Using Literal Utterances as Return Values

<form id="simple_question">
  <field name="happiness">
    <grammar type="application/x-gsl" mode="voice">
      [yes no]
    </grammar>
    <prompt>Are you happy?</prompt>
    <filled>
      <prompt>You said <value expr="happiness"/></prompt>
    </filled>
  </field>
</form>
Using Literal Utterances as Return Values

• System: Are you happy?
• Caller: Yes.
• System: You said yes.
• System: Are you happy?
• Caller: No.
• System: You said no.
<form id="simple_question">
  <field name="happiness">
    <grammar type="application/x-gsl" mode="voice">
      [yes no]
    </grammar>
    <prompt>Are you happy?</prompt>
    <filled>
      <if cond="happiness=='yes'">
        <prompt>That's good.</prompt>
      </if>
      <else/>
      <prompt>That's not so good.</prompt>
    </filled>
  </field>
</form>
Using Literal Utterances as Return Values

```xml
<form id="simple_question">
  <field name="happiness">
    <grammar type="application/x-gsl" mode="voice">
      [ yes yep yeah (sure thing) (no problem) no nope (no way) ]
    </grammar>
    <prompt>Are you happy?</prompt>
    <filled>
      <if cond="happiness=='yes'">
        <prompt>That's good.</prompt>
      </if>
      <else/>
      <prompt>That's not so good.</prompt>
    </if>
  </field>
</form>
```
Using Normalized Return Values

```xml
<form id="simple_question">
  <field name="happiness">
    <grammar type="application/x-gsl" mode="voice">
      [ [yes yeah (sure thing)] {<happiness "user-is-happy"}]
      [no (no way) nope ] {<happiness "user-is-unhappy"} ]
    </grammar>
    <prompt>Are you happy?</prompt>
    <filled>
      <if cond="happiness="user-is-happy">"
        <prompt>That's good.</prompt>
      </if>
      <else/>
        <prompt>That's not so good.</prompt>
      </else/>
    </filled>
  </field>
</form>
```
Using Normalized Values

YesNoLoose

[ [yes yeah (sure thing)] {<happiness "happy">}
  [no (no way) nope] {<happiness "unhappy">}
]

• What we are really doing:
  – Mapping from surface forms to semantic values
  – The grammar acts as a way of normalising across surface variation
Surface Variation through Lexical Synonymy

- I would like a seat in cabin class
- I would like a seat in economy class
Surface Variation through Syntactic Variation

- Bill gave a book to Mary
- Bill gave Mary a book
- Mary was given a book by Bill
- It was Bill who gave a book to Mary
- It was a book that Bill gave to Mary
- It was to Mary that Bill gave a book
- What Bill gave to Mary was a book
- What Mary was given by Bill was a book
Surface Variation through Language Variation

- The blue cat sat on the mat.
- Le chat bleu s'est reposé sur la natte.
- Blåttkatten som sitts på det mattt.
- Голубой кот сидел на циновке.
Today’s Agenda

• Some Last Words on Grammars and Tuning
• N-Best Processing
• Semantics: Deriving the Meaning of an Utterance
  – What is Semantics?
  – Using Grammars to Return Semantic Values
  – Dialog Semantics via Multiple Returned Values
  – Meaning in Statistical Language Models
• The Assignment
Full Blown Semantics

- The cat sat on the mat
- $\exists e \exists x \exists y \text{event-type78}(e) \cap \text{PAST}(e) \cap \text{entity-type122}(x) \cap \text{entity-type157}(y) \cap \text{AGENT}(e,x) \cap \text{LOCATION}(e, \text{spatial-relationship12}(y))$
- $\exists e \exists x \exists y \text{SITTING}(e) \cap \text{PAST}(e) \cap \text{CAT}(x) \cap \text{MAT}(y) \cap \text{AGENT}(e,x) \cap \text{LOCATION}(e, \text{ON}(y))$
Full Blown Semantics via Recursive Attribute-Value Matrices

• The cat sat on the mat

• \[
\text{index} = e \\
\text{sem} = \{ \text{time} = \text{PAST} \\
\text{eventtype} = \text{SITTING} \\
\text{agent} = \{ \text{index} = x \\
\text{type} = \text{CAT} \} \\
\text{location} = \{ \text{spatial-relationship} = \text{ON} \\
\text{object} = \{ \text{index} = y \\
\text{type} = \text{MAT} \} \} \]
Simple Case Frame Semantics

• The cat sat on the mat
  – [EVENT = SITTING
    AGENT = CAT
    LOCATION = on(MAT)]

• No logical quantification
• No explicit distinction between sense and reference
• No real support for general purpose inference
A ‘Complete’ Analysis

• I want to fly to Boston on Monday 17th June 2004.

→ {<AGENT Caller>
   <REQUEST Flight>
   <DESTINATION Boston>
   <DATE 2004-06-17>}

A Sufficient Analysis

Some attribute–values are predictable and unchanging:

• I want to fly to Boston on Monday 17th June 2004.

→ \{<\text{DESTINATION\ Boston}>\}
  <\text{DATE\ 2004-06-17}>\}
The Point

• In a limited domain system, not all dimensions of semantics are important

• We can:
  – ignore (‘compile out’) distinctions that are unnecessary
  – assume defaults for values that do not change

• ‘Domain semantics’ make the problem of handing meaning tractable
Today’s Agenda

• Some Last Words on Grammars and Tuning
• N-Best Processing
• Semantics: Deriving the Meaning of an Utterance
  – What is Semantics?
  – Using Grammars to Return Semantic Values
  – Dialog Semantics via Multiple Returned Values
    – Meaning in Statistical Language Models
• The Assignment
Grammar-based vs N-Gram Language Models

- A language model predicts what the next word will be
- Two kinds of language model are common:
  - Grammar based models: specifies alternatives via rules
  - Statistical models: trained on a large body of text, give the probability of following words based on previous input
- Finite state dialog systems generally use grammars
- Statistical models are the most common in large vocabulary systems: also referred to as n-gram models
Getting Meaning From Grammars

• Grammars allow us two ways of managing meaning:
  – Decompose structure to identify component parts
  – Aggregate superficially different responses
Getting Meaning from N-Grams

- No structure returned: just a sequence of words
- Options:
  - Parse the word sequence using a grammar
  - Statistically relate utterances to actions
Getting Meaning from N-Grams

My house has burnt down …
I need to make a claim …
My car’s just been stolen …
How do I make a claim?
Claims department please …

Auto insurance please …
I want to take our car insurance …
Can I take out car insurance?
I want to insure my car …
Please put me through to car insurance

Claims Department

New Policies: Automobiles
Take Home Messages

• ‘Real’ Semantics is complex — and overkill for most SLDSs
• Grammars provide a way of organising the recognition process to recover the important semantic distinctions
• Statistical language modelling provides a coarse-grained approach to semantics
Today’s Agenda

• Some Last Words on Grammars and Tuning
• N-Best Processing
• Semantics: Deriving the Meaning of an Utterance
• The Assignment
Assignment #2: How To Get A Pass Grade

- Design and implement a form-filling approach
- Make sure it works
- Make sure your prompts are sensible
- Make sure your grammars have reasonable coverage
- Make sure you document it properly
Assignment #2:
How To Get Better Than A Pass

Everything you need for a pass, plus:

- Incorporate some mixed initiative handling
- Incorporate sophisticated date and time grammars (more sophisticated than anything we saw in the tutorial)
- Go the extra mile: add functionalities not explicitly requested
## Assignment #2: Assessment

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation</td>
<td>5</td>
</tr>
<tr>
<td>Basic form filling system</td>
<td>2.5</td>
</tr>
<tr>
<td>Incorporation of some mixed initiative</td>
<td>2.5</td>
</tr>
<tr>
<td>Sophisticated handling of times and dates</td>
<td>2.5</td>
</tr>
<tr>
<td>Additional functionalities beyond those detailed in the specification</td>
<td>2.5</td>
</tr>
</tbody>
</table>