Natural language generation for dialogue systems

David DeVault

CSCI 599 Special Topic: Natural Language Dialogue Systems
Spring 2013
Overview

- What is NLG in dialogue systems, and why is it challenging?
- Some approaches to NLG in dialogue systems
- NLG pipelines and sub-problems
- NLG evaluation and challenge tasks
- NLG-related special topics
What is NLG in dialogue systems?

- Automatic Speech Recognition (ASR)
- Natural Language Understanding (NLU)
- Dialogue Manager (DM)
- Natural Language Generation (NLG)
- Text-to-Speech Synthesis (TTS)

Speech → Text → Semantic Representation → Context → Semantic Representation

Speech → Text → Text-to-Speech Synthesis (TTS) → Speech

Speech → Text → Natural Language Generation (NLG) → Text → Context → Semantic Representation
Why is NLG challenging in dialogue systems?

- May need to achieve **fluent output for large semantic spaces**
  - A large space of possible meanings may need to be covered
  - Fluent output requires that numerous grammatical rules be respected

**Good Text:**  
we don't have medical supplies here captain

**Bad Text:**  
- we don't has medical supplies here captain
- he don't has medical supplies here captain
- we captain don't have medical supplies here

- Simple “string splicing” approaches often yield ungrammatical texts
Why is NLG challenging in dialogue systems?

- NLG often involves **grammar development**
- Requires substantial linguistic expertise
- Can be hard to reuse grammars across different systems

Text: `we don't have medical supplies here captain`

Diagram:
- Dialogue Manager (DM)
- Semantic Representation
- Natural Language Generation (NLG)
- Text-to-Speech Synthesis (TTS)
- Speech
Why is NLG challenging in dialogue systems?

- NLG often requires connecting grammatical structures to application semantics

Text: we don't have medical supplies here captain

Application frame for desired utterance:

<table>
<thead>
<tr>
<th>speech-act.action</th>
<th>assert</th>
</tr>
</thead>
<tbody>
<tr>
<td>speech-act.content.polarity</td>
<td>negative</td>
</tr>
<tr>
<td>speech-act.content.attribute</td>
<td>resourceAttribute</td>
</tr>
<tr>
<td>speech-act.content.value</td>
<td>medical-supplies</td>
</tr>
<tr>
<td>speech-act.content.object-id</td>
<td>market</td>
</tr>
<tr>
<td>addressee</td>
<td>captain-kirk</td>
</tr>
<tr>
<td>dialogue-act.addresssee</td>
<td>captain-kirk</td>
</tr>
<tr>
<td>speech-act.addresssee</td>
<td>captain-kirk</td>
</tr>
</tbody>
</table>

Grammatical structure for desired utterance:

```
S
  | VP
  |  \\
  |   \\
  |   \\
  |   \\
  |   \\
  |   we
  |   do
  |   not
  |   \\
  |   \\
  |   \\
  |   \\
  |   have
  |   \\
  |   \\
  |   \\
  |   \\
  |   medical supplies
  |   \\
  |   \\
  |   \\
  |   \\
  |   here
```
Why is NLG challenging in dialogue systems?

- **Context-sensitivity** of language
  - People use context to communicate efficiently while coordinating their understanding
  - Very salient cases: pronouns and referring expressions

Text:
- he can bring you medical supplies
- my lieutenant can bring you medical supplies
- Lieutenant Jones can bring you medical supplies
Example of a referential ambiguity

Candidate Objects

Which object?

Your scene

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

History

s2: the object is a circle
Agent: ok
[s2 clicks Skip this object]
s2: the object is the beige diamond

Present: [s2, Agent], Active: []
Example of a color term ambiguity

Candidate Objects

Your scene

a19: click continue
[ Agent clicks Continue (next object) ]
Agent: the target is dark orange
a19: click continue
[ Agent clicks Continue (next object) ]
Example of a color term ambiguity

Candidate Objects

Your scene

[ Agent clicks Continue (next object) ]
Agent: the target is dark orange
a19: click continue
[ Agent clicks Continue (next object) ]
Agent: the solid brown square

You (a19:)

or
Skip this object
Example of a color term ambiguity

Candidate Objects

which color?

Your scene

[ Agent clicks Continue (next object) ]
Agent: the target is dark orange
a19: click continue
[ Agent clicks Continue (next object) ]
Agent: the solid brown square

You (a19:)

or Skip this object
Example of a color term ambiguity

Candidate Objects

Your scene

Agent: the target is dark orange
a19: click continue
[ Agent clicks Continue (next object) ]
Agent: the solid brown square
a19: light brown?
Example of a color term ambiguity

Candidate Objects

Your scene

a19: click continue
[Agent clicks Continue (next object)]
Agent: the solid brown square
a19: light brown?
Agent: yeah
Why is NLG challenging in dialogue systems?

- **Context-sensitivity** of language
  - People use context to communicate efficiently while coordinating their understanding
  - Very salient cases: pronouns and referring expressions
  - An NLG module needs to select appropriate surface expressions for the context of the utterance
    - Need to find a way to represent the relevant aspects of linguistic context
    - Need to circumscribe the possible misunderstandings in context
Why is NLG challenging in dialogue systems?

- **Ambiguity** of natural language in context
  - The same surface expression can often mean several different things
  - Ambiguity is pervasive in NLP
    - syntactic ambiguity: *I saw the man with the binoculars*
    - referential ambiguity: *The object is the beige diamond*
    - word sense ambiguity: *I need to go to the bank*
    - ambiguity in implication: *It's cold outside*
    - ...
  - Recall that an NLU module often needs to resolve ambiguity and identify the user's specific meaning
  - Conversely, an NLG module needs to avoid (problematic) ambiguity so the user can identify the system's meaning
Why is NLG challenging in dialogue systems?

- Challenges with **semantic representations**
  - Ideal input format for NLG is not settled in the field
    - Wide-coverage NLG systems may expect input in a format that is alien to your application
  - As discussed for NLU, it can be hard to find a general-purpose semantic representation that is practical for use in your specific system
    - The semantic distinctions that matter tend to vary from application to application
    - System builders often create idiosyncratic representations for their domains
    - Can make it harder to reuse NLG work across systems
Why is NLG challenging in dialogue systems?

- NLG potentially involves *large search spaces*
  - Large number of possible grammatical structures
  - Need good heuristics to keep search tractable
Why is NLG challenging in dialogue systems?

- NLG potentially involves *large search spaces*
  - Large number of possible grammatical structures
  - Need good heuristics to keep search tractable
Why is NLG challenging in dialogue systems?

- Phenomena of *spontaneous speech* are complicated in a standard pipeline architecture
  - Result is that dialogue systems don't really talk like people
    - They aren't disfluent
    - They don't start speaking until they know exactly what they're going to say
    - They don't react to user backchannels or other user feedback while talking
Example of spontaneous dialogue
(from Meteer 1995, Switchboard)

Annotated:

A: he’s pretty good. / He stays out of the street / {C and, } {F uh, } if I
catch him I call him / {C and } he comes back. / {D So } [ he, + he’s ] pretty
good about taking to commands [ and + --

B: {F Um. } /

A: -- and ] things. /

B: Did you bring him to a doggy obedience school or --

A: No -- /

B: -- just --

A: -- we never did. /

B: -- train him on your own / {C and, } --/

A: [ I, + I ] trained him on my own / {C and, } {F uh, } this is the first dog
I’ve had all my own as an adult. /

B: Uh-huh. /
Overview

- What is NLG in dialogue systems, and why is it challenging?
- Some approaches to NLG in dialogue systems
- NLG pipelines and sub-problems
- NLG evaluation and challenge tasks
- NLG-related special topics
Some approaches to NLG in dialogue systems

- Canned text
- Template-based generation
- Distractor sets and generation
- Hand-authored grammars
- Example-based approaches
- Wide coverage realizers
Canned text

- Some dialogue systems use pre-authored complete utterances, or “canned text”
  - As in NPCEditor...

- Advantages:
  - no need to develop a grammar or detailed semantic representation
  - can use a voice actor instead of TTS (higher quality speech)
  - can be adequate for systems with finite output needs

- Disadvantages:
  - system output sharply limited
    - only have complete utterances anticipated at design time
  - no context sensitivity
Canned text

- Some dialogue systems use pre-authored complete utterances, or “canned text”
  - As in NPCEditor... (classification based on canned texts)

- Advantages:
  - no need to develop a grammar or detailed semantic representation
  - can use a voice actor instead of TTS (higher quality speech)
  - can be adequate for systems with finite output needs

- Disadvantages:
  - system output sharply limited
    - only have complete utterances anticipated at design time
  - no context sensitivity
Template-based generation

- Template-based generators use rules to instantiate slots within partial texts (templates)
  
  we don't have ___$value___ here ___$addressee___
  
  we don't have *medical supplies* here *captain*

- Many systems use some form of *ad hoc* template-based NLG

- More feature rich template-based systems have also been developed
  
  - YAG (McRoy, Channarukul, and Ali 2003)
  - TG/2 (Busemann and Horacek 1998)
  - D2S (van Deemter and Odijk 1997; Theune et al. 2001)
  - EXEMPLARS (White and Caldwell 1998)
  - XTRAGEN (Stenzhorn 2002)

- Complex template-based systems have some grammar-like capabilities; see van Deemter, Krahmer, and Theune (2005) for discussion
Template-based generation

• Advantages:
  – Requires less linguistic expertise than grammar-based approaches
  – Enables some context-sensitivity and productive capabilities
  – Fast at run-time

• Disadvantages:
  – Can be hard to reuse templates across systems
  – May get more limited coverage than grammar-based approaches
  – Maintaining fluent output can be more complicated than you expect
Distractor sets and generation

- Distractor sets are a way to formalize possible ambiguities in context
  - Each *target object* is assigned a set of *distractor objects*
  - An unambiguous referring expression matches the target but none of the distractors; see Dale & Reiter (1995)
Generating referring expressions with distractor sets

“the rabbit”

“the rabbit on the right”

“the rabbit in the hat”
Hand-authored grammars

- A grammar can be hand-authored to handle the generation needs of a specific system
- Many grammar formalisms exist (CFG, TAG, CCG, HPSG, ...)
- The grammar defines the space of syntactic structures that are available in NLG
- Systems need to somehow link these syntactic structures to context and semantic representations
Example of NLG with a hand-authored grammar

SPUD (Sentence Planning Using Description) is a sophisticated approach to context-sensitive NLG (Stone et al., 2003)

A SPUD grammar consists of a set of linguistic resources.

Each linguistic resource consists of
- syntax
- presuppositions
- assertions
- a syntactic operation

SPUD uses a lexicalized tree adjoining grammar (LTAG).
How SPUD combines linguistic resources

NP [ref X]  
D  
N [ref X]  
meeting  

meeting(X)  
comp NP [ref X]  

PP  
with  
NP [ref Z]  

with(Y,Z)  
postmod NP [ref Y]  

NP [ref X]  
D  
N [ref X]  
meeting  
with  
NP [ref Z]  

meeting(X) & with (X, Z)  
comp NP [ref X]  

=  

+  

Interpreting a linguistic resource in context

SPUD requires a **knowledge interface** that answers queries about which individuals in the context that satisfy a set of constraints, e.g.

\[ \text{meeting}(x) \, \& \, \text{with}(x,y) \]

The output of these queries is a set of binding sets.
SPUD requires a **dialogue manager** to provide distractor sets for arbitrary individuals in the context.

- "the rabbit"
- "the rabbit on the right"
- "the rabbit in the hat"
SPUD’s search algorithm

Search states are *communicative intents*:
- a linguistic resource
- an intended interpretation of each semantic variable
- a set of supported interpretations

Search states are expanded through syntactic operations & queries to the knowledge base and dialogue manager.

SPUD uses greedy search, with $k$ steps of lookahead.

A goal state has a linguistic resource for which the intended interpretation of its presuppositions and assertions is the only supported interpretation in the context.
Example SPUD search space

Each node in the search space at right represents a communicative intent
Example communicative intent

{Y \leftarrow t_3, \text{PossVarVal} \leftarrow \text{inTargetDomain}, A \leftarrow \text{inFocus}, Set \leftarrow \text{setPrag}, \text{Equals} \leftarrow \text{equal}, X \leftarrow e_2, \text{Circle} \leftarrow \text{circleFigureObject}, M \leftarrow \text{addrC}, \text{Pink} \leftarrow \text{palevioletredFigureObject}}

1 interpretation:
{Y \leftarrow t_3, \text{PossVarVal} \leftarrow \text{inTargetDomain}, A \leftarrow \text{inFocus}, Set \leftarrow \text{setPrag}, \text{Equals} \leftarrow \text{equal}, X \leftarrow e_2, \text{Circle} \leftarrow \text{circleFigureObject}, M \leftarrow \text{addrC}, \text{Pink} \leftarrow \text{palevioletredFigureObject}}

\text{addrC}[t_3, \text{equal}(t_3,e_2)],[\text{setPrag}[\text{inFocus}(Z),\text{inFocus}(t_3)]]
Example-based approaches

• One of the readings described an example-based approach to NLG (DeVault et al., 2008)
  – See also (Wong and Mooney, 2007; Stone, 2003; Varges and Mellish, 2001)

• Trade-offs with example-based approaches
  – Pros: can tailor syntactic/semantic modeling to application, less linguistic knowledge required
  – Cons: complex relation between output, examples, and learning algorithms

• Overall such approaches provide a practical approach to productive output capabilities with less grammar development effort
Practical NLG for dialogue systems: requirements

• Fast enough for real-time interaction
• Adequate coverage for domain
  – Output fluency
  – Fidelity to the requested meaning
  – Variety of alternative outputs
  – Tolerance for generation failures
• Low development costs
  – Building dialogue systems is complex/expensive enough!
• Easy for dialogue manager to formulate an NLG request
Practical NLG for dialogue systems: options

- Grammar-based generation may not meet these requirements
  - Options: wide-coverage realizer, hand-crafted grammar, example-based method
  - Get productive coverage
  - But can be slow, expensive to build, difficult to formulate NLG request

- Simpler solutions often adopted
  - Canned text
  - Template-based output
Example dialogue system: Doctor Perez

Captain: I have orders to move this clinic to a camp near the US base.

Perez: I need to stay and help the patients here.

... 

Captain: Would you be willing to move downtown?

Perez: We don't have medical supplies downtown captain.

... 

Captain: The U.S. Army could move you.

Perez: I must remain neutral captain.
Practical NLG for Doctor Perez: requirements

- Fast
  - Output in \( \leq 200 \text{ms} \)
- Productive coverage
  - Thousands of semantic inputs
  - Some disfluency is acceptable
- Quick and easy NLG authoring
  - NL team has over 10 members: programmers, testers, linguists, computational linguists, domain experts, annotators
- Interface to available semantic representations
  - Needs to be easy to formulate an NLG request given a semantic frame
Doctor Perez's existing semantic representation

**semantic frame:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>addressee</td>
<td>captain-kirk</td>
</tr>
<tr>
<td>dialogue-act.addressee</td>
<td>captain-kirk</td>
</tr>
<tr>
<td>dialogue-act.type</td>
<td>assign-turn</td>
</tr>
<tr>
<td>dialogue-act.actor</td>
<td>doctor-perez</td>
</tr>
<tr>
<td>speech-act.actor</td>
<td>doctor-perez</td>
</tr>
<tr>
<td>speech-act.addressee</td>
<td>captain-kirk</td>
</tr>
<tr>
<td>speech-act.action</td>
<td>assert</td>
</tr>
<tr>
<td>speech-act.content.type</td>
<td>state</td>
</tr>
<tr>
<td>speech-act.content.polarity</td>
<td>negative</td>
</tr>
<tr>
<td>speech-act.content.time</td>
<td>present</td>
</tr>
<tr>
<td>speech-act.content.attribute</td>
<td>resourceAttribute</td>
</tr>
<tr>
<td>speech-act.content.value</td>
<td>medical-supplies</td>
</tr>
<tr>
<td>speech-act.content.object-id</td>
<td>market</td>
</tr>
</tbody>
</table>

**possible realization:**
captain there are no medical supplies at the market
Technical approach of DeVault et al. (2008)

- Training examples
- Automatic grammar induction
- Automatic search strategy optimization
Training examples

- A training example $e = (\text{utterance, semantic links, syntax})$
we don't have medical supplies here captain
Semantic links

- We assume application semantics is a set \( M = \{ m_1, ..., m_n \} \)

**we don't**
- speech-act.action: assert
- speech-act.content.polarity: negative

**have**
- speech-act.content.attribute: resourceAttribute

**medical supplies**
- speech-act.content.value: medical-supplies

**here**
- speech-act.content.object-id: market

**captain**
- addressee: captain-kirk
- dialogue-act.addressee: captain-kirk
- speech-act.addressee: captain-kirk
We don't have medical supplies here, Captain.
Automatic grammar induction

- We induce a grammar that associates linked application semantics to fragments of phrase structure
  - Our grammar is a variant of PTAG (Chiang, 2003)
  - Domain-independent rules used to assign derivations to training examples
  - Given derivations, can recursively detach grammar entries that link tree structures directly to application semantics
More details on grammar induction

- Grammar is a variant of PTAG (Chiang, 2003)
  - Adding application semantics to trees
  - Allowing multiple lexical anchors per tree (for flexible semantic granularity)
- Use rules to assign derivations to training examples
  - Lexical anchor(s) for non-terminals
  - Complement/adjunct status for arguments (Magerman, 1995; Chiang, 2003; Collins, 1999)
  - Additional syntactic features for verb inflectional agreement, finiteness in VP and S complements, grammatical role constraints
- Detach grammatical entries
Example: inferred lexical entries

- **speech-act.action**: assert
- **speech-act.content.polarity**: negative
- **speech-act.content.attribute**: resourceAttribute
- **speech-act.content.value**: medical-supplies
- **speech-act.content.object-id**: market

**Captain Kirk** wants to **have** **medical supplies** *here*.
Generation as search

- Input: desired semantic representation $M = \{m_1, \ldots, m_n\}$
- Output: utterance $u$ that expresses (at least) $M$
- Algorithm:
  - Beam search in space of derived trees
  - Rank expansions using weighted features:
    - Goal nodes express (at least $M$)
- Anytime algorithm:
  - Accrue goal states until timeout
  - Return list of alternatives ranked by derivation probability
Search strategy optimization

• Select beam size (semi-automatic)
  – Select size so can search the beam exhaustively before the timeout (200ms)

• Select features and weights (automatic)
  – Use domain-independent rules to define potential features
  – Try to generate the training examples using the induced grammar
  – Update weights when mistakes occur (Daumé and Marcu, 2005; Collins and Koo, 2005)
Wide coverage realizers

• A range of wide coverage realizers have been developed
  – See e.g. (White et al., 2007; Cahill and van Genabith, 2006; Zhong and Stent, 2005; Langkilde-Geary, 2002; Langkilde and Knight, 1998; Elhadad, 1991)

• Trade-offs with wide coverage realizers
  – Pros: free syntactic/language model, linguistically sound/robust
  – Cons: extra information needed to specify realizer input
Some approaches to NLG in dialogue systems

- Canned text
- Template-based generation
- Distractor sets and generation
- Hand-authored grammars
- Example-based approaches
- Wide coverage realizers
Overview

- What is NLG in dialogue systems, and why is it challenging?
- Some approaches to NLG in dialogue systems
- NLG pipelines and sub-problems
- NLG evaluation and challenge tasks
- NLG-related special topics
NLG pipelines and sub-problems

• Some NLG systems decompose the NLG problem into a number of internal steps
  – content planning
    • selection of communicative goals
  – sentence planning / microplanning
    • selection of syntactic structures
    • aggregation
    • referring expression generation
    • lexical choice
  – surface realization
    • production of surface text
• The “correct” way to decompose NLG into subtasks is subject to debate; see e.g. optional reading by Stone & Webber (1998)
Example from Stone & Webber (1998)

Figure 1: “Remove the rabbit from the hat.”

NOT “Remove the rabbit in the hat from the hat containing the rabbit”
PERSONAGE Architecture

**INPUT**: Stylistic parameters, DSyntS, Speech plans

**PERSONAGE**

1. **Content Planner**
   - Verbosity
   - Restatements
   - Content Polarity
   - Syntactic Complexity
   - Self-Reference

2. **Syntactic Template Selection**
   - Contrast: e.g. however, but
   - Justify: e.g. so, since
   - Period

3. **Aggregation**

4. **Pragmatic Marker Insertion**
   - Exclamation
   - Hedges: e.g. kind of, rather, basically, you know
   - Filled Pauses: e.g. err...
   - Swear Words: e.g. damn
   - In Group Markers: e.g. pal
   - Stuttering: e.g. Ri-Ri-River
   - Tag Questions

5. **Lexical Choice**
   - Frequency of Use
   - Word Length
   - Verb Strength

**OUTPUT UTTERANCE**

**Realization**
Surface Realization using RealPro

RealPro: off-the-shelf surface realizer

Deep-Syntactic Structure (DSyntS) for representing utterances

- Syntactic and lexical knowledge
- “Deep”: only meaning-bearing lexemes are represented, not function words

Utterance: “I don’t believe it.”

DSyntS representation in XML (format used by PERSONAGE):

```xml
<dsyntnode class="verb" lexeme="believe" polarity="neg">
    <dsyntnode lexeme="<PRONOUN>" number="sg" person="1st" rel="I"/>
    <dsyntnode lexeme="it" rel="II"/>
</dsyntnode>
```

Important to have correct DSyntS representation
PERSONAGE discussion?
Overview

- What is NLG in dialogue systems, and why is it challenging?
- Some approaches to NLG in dialogue systems
- NLG pipelines and sub-problems
- NLG evaluation and challenge tasks
- NLG-related special topics
NLG evaluation

- NLG outputs can be evaluated in many ways
- Subjective metrics look at human judgments of NLG output quality
- Objective metrics look at quantifiable aspects of NLG output
Subjective metrics from GIVE-2 (Koller et al., INLG 2010)

Q1: The system used words and phrases that were easy to understand.

Q2: I had to re-read instructions to understand what I needed to do.

Q3: The system gave me useful feedback about my progress.

Q4: I was confused about what to do next.

....
Objective metrics from GIVE-2 (Koller et al., INLG 2010)

- **task success**: Did the player get the trophy?
- **duration**: Time in seconds from the end of the tutorial until the retrieval of the trophy.
- **distance**: Distance traveled (measured in distance units of the virtual environment).
- **actions**: Number of object manipulation actions.
- **instructions**: Number of instructions produced by the NLG system.
- **words per instruction**: Average number of words the NLG system used per instruction.
Shared NLG challenge tasks

- The NLG community is moving toward having more shared challenges
- Some recent/current challenges:
  - GIVE, GIVE-2, GIVE-2.5 (2008-2011)
    - http://www.give-challenge.org/
  - GRUVE (2013, happening now!)
    - https://sites.google.com/site/hwinteractionlab/current-research/generation-challenge
  - Surface Realization Shared Task (2011)
    - http://www.nltg.brighton.ac.uk/research/sr-task/
  - REG challenge (2008)
    - http://www.itri.brighton.ac.uk/research/reg08/
Overview

• What is NLG in dialogue systems, and why is it challenging?
• Some approaches to NLG in dialogue systems
• NLG pipelines and sub-problems
• NLG evaluation and challenge tasks
• NLG-related special topics
NLG-related special topics

• referring in dialogue
• incremental speech processing
• multi-modal dialogue
• argumentation
• embodied conversational agents
Course Projects

• Start thinking about a possible course project
  – See example projects online

• One-page project proposal due in two weeks (week seven, February 27th)

• If you'd like feedback, feel free to send us a short idea (paragraph-length) in the meantime