Advanced Dialog Systems

Dr. Demetrios Glinos
University of Central Florida

CAP6640 – Computer Understanding of Natural Language

Today

- Dialog-State Systems
- Dialog Acts
- Interpreting Dialoge Acts
- Dialog Policy
- Natural Language Generation

Excerpt: Abbott & Costello's "Who's on First"

C : I want you to tell me the names of the fellows on the St. Louis team.

A : I'm telling you. Who's on first, What's on second, I Don't Know is on third--

C : You know the fellows' names?

A : Yes.

C : Well, then who's playing first?

A : Yes.

C : I mean the fellow's name on first base.

A : Who.

C : The fellow playin' first base.

A : Who.

C : The guy on first base.

A : Who is on first.

C : Well, what are you askin' me for?

A : I'm not asking you--I'm telling you. Who is on first.

C : I'm asking you--who's on first?

A : That's the man's name.

C : That's who's name?

A : Yes.

Speech or Dialog Acts

- Speech or dialog acts
  - what we do in conversation
  - making a statement, asking a question, giving an order, uttering an exclamation

- GUS-style frame-based dialog systems cannot handle different dialog acts
  - but frames and slots will be useful here, too

- Dialog-state systems
  - also called belief-state or information-state systems
  - also based on filling frames and slots
  - but can understand and generate a variety of dialog acts
  - can track who has the initiative
  - what to say next is based on a dialog policy (more complex than filling the next slot)

Dialog-State System Architecture

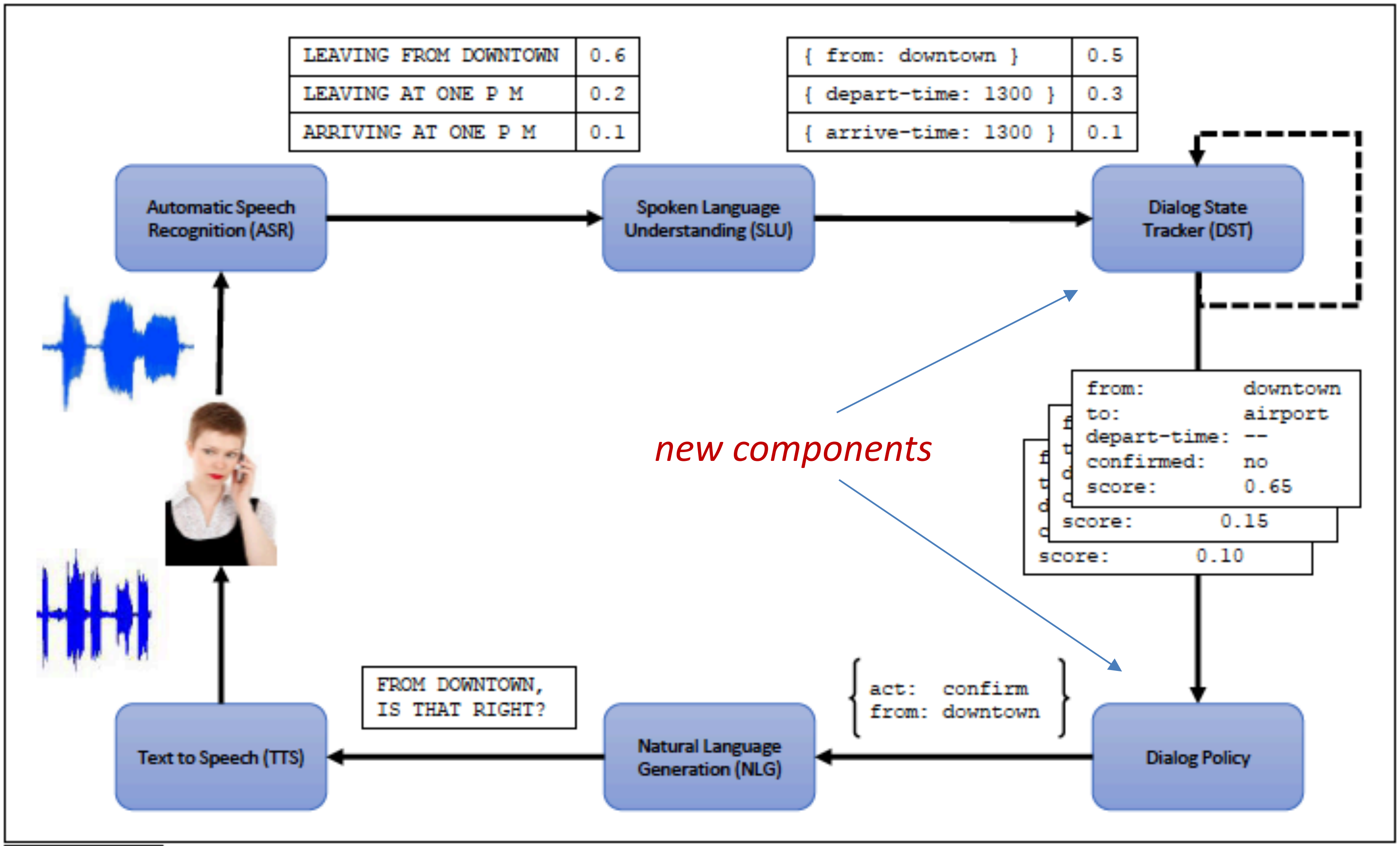


Figure 30.1 Architecture of a dialog-state system for task-oriented dialog from Williams et al. (2016).

source: J&M (3d Ed. draft)

Today

- Dialog-State Systems
- Dialog Acts
- Interpreting Dialoge Acts
- Dialog Policy
- Natural Language Generation

Dialog Acts

- Key insight: Utterances are actions, commonly called speech acts

- Speech acts (Bach and Harnish, 1979):

  Constatives: committing the speaker to something being the case
  e.g., *answering, claiming, confirming, denying, denying, stating*

  Directives: attempts by the speaker to get the addressee to do something
  e.g., *ordering, asking, advising, forbidding, inviting, requesting*

  Commissives: committing the speaker to some future course of action
  e.g., *promising, planning, vowing, betting, opposing*

  Acknowledgments: expressing the speaker's attitude regarding the hearer with respect to some social action
  e.g., *apologizing, greeting, thanking, accepting an acknowledgment*

Common Ground

- Dialog is not a series of independent speech acts
- It is a collective act performed by speaker and hearer

- Common ground
  - the set of things that are mutually believed by both speakers
  - must be continually established
  - hearer must *ground* the speaker's utterances
    - i.e., acknowledge them to make it clear that the hearer has understood the speaker's meaning and intention

- Principle of closure: Agents performina an action require evidence, sufficient for current purposes, that they have succeeded in performing it

  - humans need this, too
  - this is why a well-designed elevator button lights up when we press it
  - grounding is also important to tell the speaker that he has *not* succeeded

Grounding Methods

- A continuum of grounding methods, ordered from weakest to strongest (Clark and Schaefer, 1989):

Continued attention	B shows she is continuing to attend and therefore remains satisfied with A's presentation
Next contribution	B starts in on the next relevant contribution
Acknowledgment	B nods or says a continuer like *uh-huh*, *yeah*, or the like, or an assessment like *that's great*
Demonstration	B demonstrates all or part of what she has understood A to mean, by reformulating (paraphrasing) A's utterance or by collaboative completion of A's utterance
Display	B displays verbatim all or part of A's presentation

Examples of Grounding

C_1:	. . . I need to travel in May.	
A_1:	And, what day **in May** did you want to travel?	*display*
C_2:	OK uh I need to be there for a meeting that's from the 12th to the 15th.	
A_2:	And you're flying into what city?	
C_3:	Seattle.	*next contribution*
A_3:	And what time would you like to leave Pittsburgh?	*acknowledgment*
C_4:	Uh hmm I don't think there's many options for non-stop.	
A_4:	Right. There's three non-stops today.	
C_5:	What are they?	
A_5:	The first one departs PGH at 10:00am arrives Seattle at 12:05 their time. The second flight departs PGH at 5:55pm, arrives Seattle at 8pm. And the last flight departs PGH at 8:15pm arrives Seattle at 10:28pm.	
C_6:	OK I'll take the 5ish flight on the night before on the 11th.	
A_6:	On the 11th? OK. Departing at 5:55pm arrives Seattle at 8pm, U.S. Air flight 115.	
C_7:	OK.	

Figure 30.2 Part of a conversation between a travel agent (A) and client (C).

source: J&M (3d Ed. draft)

Speech Acts in Practical Systems

- Practical systems must distinguish
  - statements
  - directives asking for information
  - directives that are orders to do something
- Speech acts are important here, too
- Example

Without acknowledgment	With acknowledgement
System: Do you wish to review more of your personal profile? Caller: No. System: What's next?	System: Do you wish to review more of your personal profile? Caller: No. System: Okay, what's next?

Dialog Acts

- Dialog acts
  - combine speech acts and grounding
  - serve as tags for entire sentences
  - tagset is generally custom built for each particular application

Tag	Sys	User	Description
HELLO($a=x, b=y, \ldots$)	✓	✓	Open a dialog and give info $a=x, b=y, \ldots$
INFORM($a=x, b=y, \ldots$)	✓	✓	Give info $a=x, b=y, \ldots$
REQUEST($a, b=x, \ldots$)	✓	✓	Request value for a given $b=x, \ldots$
REQALTS($a=x, \ldots$)	χ	✓	Request alternative with $a=x, \ldots$
CONFIRM($a=x, b=y, \ldots$)	✓	✓	Explicitly confirm $a=x, b=y, \ldots$
CONFREQ($a=x, \ldots, d$)	✓	χ	Implicitly confirm $a=x, \ldots$ and request value of d
SELECT($a=x, a=y$)	✓	χ	Implicitly confirm $a=x, \ldots$ and request value of d
AFFIRM($a=x, b=y, \ldots$)	✓	✓	Affirm and give further info $a=x, b=y, \ldots$
NEGATE($a=x$)	χ	✓	Negate and give corrected value $a=x$
DENY($a=x$)	χ	✓	Deny that $a=x$
BYE()	✓	✓	Close a dialog

slot fillers

Figure 30.4 Dialogue acts used by the HIS restaurant recommendation system of Young et al. (2010). The Sys and User columns indicate which acts are valid as system outputs and user inputs, respectively.

source: J&M (3d Ed. draft)

Example: Sample HIS System Dialog

	Utterance	Dialogue act
U:	Hi, I am looking for somewhere to eat.	`hello(task = find,type=restaurant)`
S:	You are looking for a restaurant. What type of food do you like?	`confreq(type = restaurant, food)`
U:	I'd like an Italian somewhere near the museum.	`inform(food = Italian, near=museum)`
S:	Roma is a nice Italian restaurant near the museum.	`inform(name = "Roma", type = restaurant, food = Italian, near = museum)`
U:	Is it reasonably priced?	`confirm(pricerange = moderate)`
S:	Yes, Roma is in the moderate price range.	`affirm(name = "Roma", pricerange = moderate)`
U:	What is the phone number?	`request(phone)`
S:	The number of Roma is 385456.	`inform(name = "Roma", phone = "385456")`
U:	Ok, thank you goodbye.	bye()

Figure 30.5 A sample dialog from the HIS System of Young et al. (2010) using the dialog acts in Fig. 30.4.

source: J&M (3d Ed. draft)

Conversational Analysis

- Conversational analysis (CA)
  - analyzes structure of conversations by examining adjacency pairs
    - consists of a "first pair part" and a "second pair part"
  - because dialog acts are not independent
  - examples:
    - question and answer
    - proposal and acceptance (or rejection)
    - compliment and downplayer
    - greeting and greeting
- adjacency pair structure can help model decide which action to take

Adjacency Pair Separation

- adjacency pair components can be separated by a side sequence or a pre-sequence

- side sequence

  User: What do you have going to UNKNOWN_WORD on the 5th?

    System: Let's see, going *where* on the 5th?

    User: Going to Hong Kong.

  System: OK, here are some flights ...

- pre-sequence

  User: Can you make train reservations?

  System: Yes I can.

  User: Great, I'd like to reserve a seat on the 4pm train to New York.

Today

- Dialog-State Systems
- Dialog Acts
- Interpreting Dialoge Acts
- Dialog Policy
- Natural Language Generation

Interpreting Dialogue Acts

- Dialog-state tracker (DST) determines
  - current fillers for the current frame
  - current set of user constraints
  - the user's most recent dialog act
- Example (Mrksic et al., 2017)

Conversation	DST output
User: I'm looking for a cheaper restaurant	inform(price=cheap)
System: Sure. What kind – and where?	
User: Thai food, somewhere downtown	inform(price=cheap, food=Thai, area=centre)
System: The House serves cheap Thai food	
User: Where is it?	inform(price=cheap, food=Thai, area=centre); request(address)
System: The House is at 106 Regent Street	

Using Surface Syntax

- Surface syntax seems informative
  - yes-no question: auxiliary verb precedes subject

    e.g., *Will breakfast be served on the flight?*
  - statement: no aux-inversion

    e.g., *I don't care about lunch.*
  - command: no syntactic subject

    e.g., *Show me flights from Orlando to New York.*
- But there are common exceptions
  - request disguised as a question: *Can you give me flights from Miami to Austin?*
  - question disguised as a statement: *You said you wanted to travel on Friday?*

Learning Dialog Actions

- Dialog actions constrain the possible slots and values involved in an utterance

  I'd like Cantonese food near Soho

  → inform(food=cantonese, area=soho)

- First-pass classifier chooses dialog act (*inform* above)
  - standard supervised machine learning on labeled corpus
  - lots of features, including unigrams, bigrams, etc.
    - e.g. show me → request, are there → question

- Second-pass classifier fills appropriate slots
  - standard supervised machine learning on labeled corpus
  - e.g., CRF or RNN-based IOB tagging
  - or a multinomial classifier for all possible slot-value pairs (since ontology finite)

Detecting Correction Acts

- User correction acts
  - When a system fails to recognize or misunderstands an utterance, the user generally repeats or reformulates the utterance

- If system contains an ASR component, detection is often more difficult
  - users typically use *hyperarticulation* (exaggerated energy, duration, etc.) when stating corrections
  - e.g., *I said BAL-TI-MORE, not Boston*

- Corrective utterences are typically
  - exact repetitions
  - repetitions with one or more words omitted
  - paraphrases
  - these can be learned by a standard machine learning classifier

Today

- Dialog-State Systems
- Dialog Acts
- Interpreting Dialoge Acts
- Dialog Policy
- Natural Language Generation

Dialog Policy

- Dialog policy
  - determines the next action the system should take
  - i.e., what dialog act the system should generate

- Running example: Confirmation and Rejection
  - systems make mistakes and often need to be sure they have correctly understood the user
  - methods
    - confirming understandings with the user
    - rejecting utterances that the system is likely to have misunderstood

Confirmation Methods

- explicit confirmation (generally sounds non-natural and non-human)

S:	Which city do you want to leave from?
U:	Baltimore.
S:	**Do you want to leave from Baltimore?**
U:	Yes.
U:	I'd like to fly from Denver Colorado to New York City on September twenty first in the morning on United Airlines
S:	**Let's see then. I have you going from Denver Colorado to New York on September twenty first. Is that correct?**
U:	Yes

- implicit confirmation (generally sounds much more conversationally natural)

	U:	I want to travel to Berlin
demonstration →	S:	**When do you want to travel to Berlin?**
	U2:	Hi I'd like to fly to Seattle Tuesday Morning
display →	A3:	**Traveling to Seattle on Tuesday, August eleventh in the morning.** Your full name?

source: J&M (3d Ed. draft), Ch. 30

Rejection Strategies

- simple rejection: *I'm sorry, I didn't understand that.*

- progressive prompting (escalating detail):
  - instead of repeating the same question

System:	When would you like to leave?
Caller:	Well, um, I need to be in New York in time for the first World Series game.
System:	<reject>. Sorry, I didn't get that. Please say the month and day you'd like to leave.
Caller:	I wanna go on October fifteenth.

source: J&M (3d Ed. draft), Ch. 30

- rapid reprompting: *I'm sorry?* or *What was that?*
  - used for the first rejection instead of a detailed follow-up question
  - preferred by users

Features for Decision Policy

- Decision (in our running example) is to choose:
  - explicit confirmation
  - implicit confirmation
  - rejection
- Common features
  - ASR confidence in what is recognized from utterance
  - cost of making an error (e.g., when booking flight or moving money)
- Example: using thresholds $\alpha, \beta,$ and γ

$< \alpha$	low confidence	reject
$\geq \alpha$ and $< \beta$	some confidence	confirm explicitly
$\geq \beta$ and $< \gamma$	high confidence	confirm implicitly
$\geq \gamma$	very high confidence	do not confirm at all

A Simple Policy Based on Local Context

- Basic policy:
  - choose the best system action A_i based on the entire dialog state
  - can include entire history of conversation by both parties

$$\widehat{A_i} = \operatorname{argmax}_{A_i \in A} P(A_i \mid A_1, U_1, \ldots, A_{i-1}, U_{i-1})$$

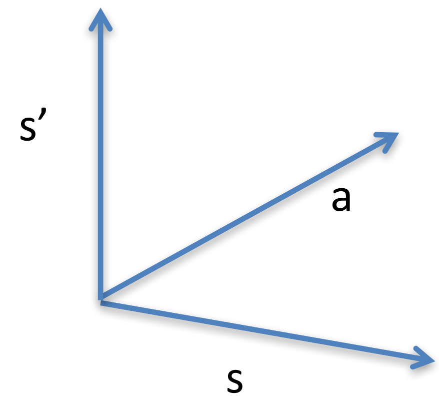
- Now, there are may different conversational paths that could lead to the same set of filled slots, so we can simplify by using the state of the affected frame

$$\widehat{A_i} = \operatorname{argmax}_{A_i \in A} P(A_i \mid Frame_{i-1}, A_{i-1}, U_{i-1})$$

- Given a sufficiently large corpus, we can estimate these probabilities using a classifier
- In practice, often need to build simulators to generate artificial conversations on which to train

Markov Decision Processes

- Markov Decision Process (MDP)
  - A set of states $s \in S$
  - A set of actions $a \in A$
  - A transition model $P(s' \mid s, a)$
  - A reward function $R(s)$
  - A start state s_0
  - Possibly one or more terminal states



- Transitions have the Markov property
  - $P(s' \mid s, a)$ does not depend on how the agent got to state s
  - we can visualize this a 3-D table of probabilities
- A solution to an MDP is a policy, π
  - $\pi(s)$ is the action to take for state s
- Executing a policy produces nondeterministic (stochastic) results
- An optimal policy must *maximize expected utility*

Modeling Dialog as an MDP

- States
  - can include entire history, but state space would be exceptionally large
  - so, typically use
    - same features as in our simplified model (frame + last turn)
    - plus ASR confidence, etc
- Actions
  - the particular speech acts to generate
  - also possibly performing a database query to find information (e.g., for flights)
- Reward function: $R = -(w_i n_i + w_e n_e + w_f n_f)$

where

$$n_i = \#turns$$
$$n_e = \#errors$$
$$n_f = \#slots\ filled$$
$$w_i = weights$$

Learning the Optimal Policy

We can use reinforcement learning

Representing a dialog as a trajectory in state space $s_1 \xrightarrow{a_1, r_1} s_2 \xrightarrow{a_2, r_2} s_3 \xrightarrow{a_3, r_3} s_4 \cdots$

The cumulative reward for a given state sequence, using discount factor $\gamma \in [0,1)$

$$Q([s_0, a_0, s_1, a_1, s_2, a_2, \cdots]) = R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \cdots$$

The expected value of the cumulative reward is given by the Bellman equation

$$Q(s,a) = R(s) + \gamma \sum_{s'} P(s'|s.a) \max_{a'} Q(s',a')$$

i.e., expected cumulative reward for a given state/action pair is

- the immediate reward for the state
- plus expected discounted utility of all possible next states s'
- weighted by the probability of transitioning to such state s'

Solving the Model

- To solve the Bellman equation for the dialog action, we need to know
  - $P(s'|s.a)$
  - $R(s)$

- We can compute maximum likelihood estimates for these values from a labeled training corpus

- Given these estimates, we can solve the model using value iteration
  - start with zero estimates for values
  - iteratively compute revised values until convergence

- Optimal policy is to move in the direction of the highest next state value
  - by taking the action with the highest probability of reaching that next state

Problems with MDPs for Dialog

- MDP only practical for small models
- Can extend using Partially Observable MDPs (POMDPs)
  - add extra latent variables to represent system's uncertainty about the true state of the dialog
- Both MDPs and POMDPs suffer from computational complexity due to their reliance on simulations that don't reflect true user behavior
- Recent efforts focus on applying reinforcement learning to deep neural networks, but there is no concensus yet on a standard paradigm

Today

- Dialog-State Systems
- Dialog Acts
- Interpreting Dialoge Acts
- Dialog Policy
- Natural Language Generation

NLG Component

- Natural language generation (NLG)
  - performed once a dialog action has been decided
  - modeled in two stages
    - content planning
      - we can assume that this is determined by the policy, when it chooses the action
      - and maybe also some additional attributes (slots and values) that the system wishes to implicitly confirm to the user
    - sentence realization
      - determining how to express the content in natural language

Example: Sentence Realization

```
{
  act query                    ← dialog act
  content depart_time          ← argument to dialog act
  depart_date {
              year 2000
              month 10
              day 5
             }
  depart_airport BOS
}
=> What time on October fifth would you like to leave Boston?
```

additional slot information to include

Figure 30.6 An input frame to NLG and a resulting output sentence, in the Communicator system of Oh and Rudnicky (2000).

source: J&M (3d Ed. draft)

Steps in Sentence Realization

1. Generate delexicalized string
    - basically, a generic sentence form, with variables for specific words
        - typically for named entities, dates, times, etc.
        - e.g., *What time on* [departure date] *would you like to leave* [depart_airport] *?*

2. Relexicalize
    - fill in variables from the frame provided by the content planner

Learning Delexicalized Strings

- Use human-human dialogs labeled with dialog acts and slots
- Delexicalize entire corpus
- Divide into separate corpora, one for each dialog act

- Example training sentences:

  And what time would you like to leave depart_city ?
  When would you like to leave depart_city ?
  When would you like to leave ?
  What time do you want to leave on depart_date ?
  OK, on depart_date, what time do you want to leave ?

- Train a distinct N-gram or neural language model for each dialog act.

- Usage: Given dialog act QUERY DEPART_TIME, system will
  - sample random sentences from this model
  - rate them based on whether too short, too long, repeat or lacking slots, etc.
  - choose the highest scoring realization

Targeted Clarification

- Clarification questions
  - e.g., when ASR fails to understand some part of user's utterance

- Humans typically use targeted clarification questions that repeat elements of the misunderstanding

User:	What do you have going to UNKNOWN_WORD on the 5th ?
System:	Going where on the 5th ?

- Typically better received than generic rejection responses like "Please repeat."

- Creating targeted clarification questions
  - Can hand-craft rules: e.g., "going to UNKNOWN_WORD" → "going where"
  - Can also build a classifier to guess which slots might have been misrecognized