

# Chatbots and Dialogue Systems

Berlin Chen

Department of Computer Science & Information Engineering  
National Taiwan Normal University

Reference:

Speech and Language Processing (3rd ed.), Chapter 15 & Teaching Material

# Prologue



- Language is the mark of humanity and sentience (知覺能力), and conversation or dialogue is the most fundamental and specially privileged arena of language
  - Conversation between humans is an intricate and complex joint activity
  - It is crucial to understand something about how humans converse with each other
- Dialogue systems (or conversational agents)
  - Programs are designed and implemented to communicate with users in natural language (text, speech, or both), which fall into two classes
    1. Chatbots
    2. Task-oriented dialogue agents

## Two Broad Kinds of Conversational Agents

- **Chatbots**: systems which are designed for extended conversations
  - Set up to mimic the unstructured conversation or chat characteristic of human-human interaction (e.g., for fun)
  - Mainly for entertainment, but also for practical purposes like making task-oriented agents more natural (e.g., for therapy)
- **Task-oriented dialogue agents**: systems which use **conversation with users** to help complete tasks
  - **Digital assistants** (Siri, Alexa, Google Now/Home, Cortana, etc.): give directions, control appliances, find restaurants, or make phone calls
  - **Conversational agents**: can answer questions on corporate websites, interface with robots, and even be used for social good (公益)

# Properties of Human Conversation

- Speech Acts
- Speaker Turns
- Grounding
- Initiatives
- Inference and Implicature

The subtle characteristics of human conversations (like the aforementioned properties) are among the reasons it is difficult to build dialogue systems that can carry on natural conversations with humans.

# Properties of Human Conversation: Speaker Turns

- **Speaker Turns:** A dialogue is a sequence of turns (C1, A2, C3, and so on), each of which is a single contribution from one speaker to the dialogue
  - A turn can consist of **a sentence** (like C1), although it might be as short as **a single word** (C13) or as long as **multiple sentences** (A10)

single sentence

very short sentence

multiple sentences

C1: ... I need to travel in May.  
A2: And, what day in May did you want to travel?  
C3: OK uh I need to be there for a meeting that's from the 12th to the 15th.  
A4: And you're flying into what city?  
C5: Seattle.  
A6: And what time would you like to leave Pittsburgh?  
C7: Uh hmm I don't think there's many options for non-stop.  
A8: Right. There's three non-stops today.  
C9: What are they?  
A10: 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.  
C11: OK I'll take the 5ish flight on the night before on the 11th.  
A12: On the 11th? OK. Departing at 5:55pm arrives Seattle at 8pm, U.S. Air  
flight 115.  
C13: OK.  
A14: And you said returning on May 15th?  
C15: Uh, yeah, at the end of the day.  
A16: OK. There's #two non-stops ... #  
C17: #Act... actually #, what day of the week is the 15th?  
A18: It's a Friday.  
C19: Uh hmm. I would consider staying there an extra day til Sunday.  
A20: OK...OK. On Sunday I have ...

**Figure 15.1** Part of a phone conversation between a human travel agent (A) and human client (C). The passages framed by # in A16 and C17 indicate overlaps in speech.

# Properties of Human Conversation: Speech Acts

- **Speech Acts**: Each utterance in a dialogue is a kind of action being performed by the speaker. These actions are commonly called speech acts or **dialog acts**
  - One taxonomy consisting of 4 major classes (Bach and Harnish, 1979):

斷言/確認	<b>Constatives</b> : committing the speaker to something's being the case (answering, claiming, confirming, denying, disagreeing, stating)
建議/請求	<b>Directives</b> : attempts by the speaker to get the addressee to do something (advising, asking, forbidding, inviting, ordering, requesting)
許諾	<b>Commissives</b> : committing the speaker to some future course of action (promising, planning, vowing, betting, opposing)
致謝/回應	<b>Acknowledgments</b> : expressing the speaker's attitude regarding the hearer with respect to some social action (apologizing, greeting, thanking, accepting an acknowledgment)

# Properties of Human Conversation: Grounding

- **Common Ground**: It is important for the participants (interlocutors) to establish what they both agree on, called the **common ground** (Stalnaker, 1978)
  - **Grounding** means acknowledging that the hearer has understood the speaker
- An **adjacency pair** of a conversation is composed of a first pair part and a second pair part (Schegloff, 1968), and these expectations can help systems decide what actions to take
  - **QUESTIONS** set up an expectation for an **ANSWER**
  - **PROPOSALS** are followed by **ACCEPTANCE** (or **REJECTION**)
  - **COMPLIMENTS** (“Nice jacket!”) often give rise to **DOWNPLAYERS** (“Oh, this old thing?”)

不會雞同鴨講

An adjacency pair can be separated by a side sequence (Jefferson 1972) or sub-dialogue.

## Properties of Human Conversation: Initiatives

- **Initiatives**: A conversation may be completely controlled by one participant, who is the **initiative of the conversation**. For example,
  - A reporter interviewing a chef asks questions, and the chef responds
  - This reporter has the **conversational initiative** (Walker and Whittaker 1990)
- Most human conversations have **mixed initiative**
  - I lead, then you lead, then I lead
- Human-Machine Conversations
  - **System-Initiative**: System asks user questions to fill out a form, user cannot change the direction
  - **User-Initiative**: User asks or commands, then system responds. In turn, the user can specify another query
  - **Mixed-Initiative**: while the norm for human-human conversations, mixed-Initiative is very difficult for dialogue systems to achieve



## Properties of Human Conversation: Inference and Implicature

- **Inference** is also important in dialogue understanding, for example

**Agent:** And, what day in May did you want to travel?

**Client:** OK, uh, I need to be there for a meeting that's from the 12th to the 15th.

Notice that the client does not in fact answer the agent's question. The client merely mentions a meeting at a certain time. What is it that licenses the agent to infer that the client is mentioning this meeting so as to inform the agent of the travel dates?

- **Implicature:** What enables hearers to draw these inferences is that conversation is guided by a set of maxims (經驗法則?), general heuristics that play a guiding role in the interpretation of conversational utterances

情溢乎辭 ? Or 辭溢乎情 ?

# Chatbot Architectures

- Rule-based systems (early approaches)
  - Pattern-action rules (ELIZA)
  - Plus a mental model (PARRY)  
(The first system to pass the Turing Test!)
- Corpus-based (present-day approaches)
  - Information Retrieval (Xiaolce)
    - Mine large datasets of human-human conversations, which can be done by using information retrieval to copy a human response from a previous conversation
  - Neural encoder-decoder (BlenderBot)
    - Generate a response from a user utterance using encoder-decoder models

# Present-Day Chatbots: Facebook's Blender-Bot

Roller et al. 2020

Will you sing me a song?

Sure, what do you want to be sung to? I can sing you a song about baking.

Yes, sing me a song about baking!

Do you want me to sing it to the tune of "Sing Along with Mitch"?

Yes, I would love that.

OK, I'll try to get it to you by the end of the day. What else do you like to do?

**User** No, sing it to me now!

OK, give me a few days. I'm a teacher, so I don't have much free time.

Can you at least tell me what the lyrics are?

Sure, it goes like this: "Baking is my favorite thing to do in the world."

How does the chorus go?

It goes: BAKING IS MY FAVORITE THING TO DO IN THE WORLD!

Wow, sounds like a bop.

I know, right? It's so catchy. I can't get it out of my head.

**Blender-Bot**

# Present-Day Chatbots: Microsoft's Xiaoice System

Zhou et al., 2020

[人类] 小冰你好	[Human] Hello, XiaoIce.
[小冰] 你好呀，你头像好有趣，感觉在哪见过	[XiaoIce] Hello, your profile pic is interesting. Like I've seen it somewhere.
[人类] 是吗	[Human] Really?
[小冰] 提不是哪部动画里的呀	[XiaoIce] Is it a comic character?
[人类] 你怎么知道的	[Human] How do you know that?
[小冰] 你猜	[XiaoIce] Guess.
[人类] 难道你是真人吗？	[Human] Are you human?
[小冰] 这个，你和我多聊聊就知道了	[XiaoIce] Well, you'll know it after chatting with me more.
[人类] 你不是人工智能机器人么	[Human] Aren't you an AI chatbot?
[小冰] 是又怎样，我不会冷冰冰的对待你的	[XiaoIce] So what? I won't be indifferent to you.
[人类] 你都有什么功能啊	[Human] What functions do you have?
[小冰] 我有200多个skills，来，说出你的需求	[XiaoIce] I have over 200 skills. Just name one you need.

# Chatbots in Old Days: ELIZA (1/2)

- ELIZA is the most important rule-based chatbot system in the history of the field (Weizenbaum, 1966)
  - Rogerian psychology: assume the pose (裝腔作勢) of knowing almost nothing of the real world

ELIZA seems to give very sophisticated therapeutic advice.

```
Men are all alike.
IN WHAT WAY
They're always bugging us about something or other.
CAN YOU THINK OF A SPECIFIC EXAMPLE
Well my boyfriend made me come here.
YOUR BOYFRIEND MADE YOU COME HERE
He says I'm depressed much of the time.
I AM SORRY TO HEAR YOU ARE DEPRESSED
::
WHO ELSE IN YOUR FAMILY TAKES CARE OF YOU?
My father
YOUR FATHER
You are like my father in some ways.
WHAT RESEMBLANCE DO YOU SEE
You are not very aggressive but I think you don't want me to notice that.
WHAT MAKES YOU THINK I AM NOT AGGRESSIVE
You don't argue with me.
WHY DO YOU THINK I DON'T ARGUE WITH YOU
You are afraid of me.
DOES IT PLEASE YOU TO BELIEVE I'M AFRAID OF YOU
My father is afraid of everybody.
WHAT ELSE COMES TO MIND WHEN YOU THINK OF YOUR FATHER
Bullies.
DOES THAT HAVE ANYTHING TO DO WITH THE FACT THAT YOUR
BOYFRIEND MADE YOU COME HERE
```

Rules are organized by **keywords**  
Each keyword has a pattern and  
a list of possible **transforms**.

Keyword: YOU  
Pattern: (0 YOU 0 ME)  
Transforms: (WHAT MAKES YOU THINK I 3 YOU)  
(WHY DO YOU THINK I 3 YOU)

Figure 15.4 Sample ELIZA dialogue from Weizenbaum (1966).

## Chatbots in Old Days: ELIZA (2/2)

- ELIZA works by pattern/transform rules

```
function ELIZA GENERATOR(user sentence) returns response  
  
Find the word w in sentence that has the highest keyword rank  
if w exists  
    Choose the highest ranked rule r for w that matches sentence  
    response ← Apply the transform in r to sentence  
    if w = 'my'  
        future ← Apply a transformation from the 'memory' rule list to sentence  
        Push future onto memory queue  
    else (no keyword applies)  
        either  
            response ← Apply the transform for the NONE keyword to sentence  
        or  
            response ← Pop the oldest response from the memory queue  
return(response)
```

**Figure 15.5** A simplified sketch of the ELIZA algorithm. The power of the algorithm comes from the particular transforms associated with each keyword.

## Chatbots in Old Days: PARRY

Colby et al., 1972

- PARRY was another chatbot with a clinical psychology focus, and used to study schizophrenia (精神分裂症)
  - In addition to ELIZA-like regular expressions, PARRY included a model of its own **mental state**, with **affective variables** for the agent's levels of fear and anger
  - Certain topics of conversation might lead PARRY to become more angry or mistrustful

If PARRY's anger variable is high, he will choose from a set of "hostile" outputs. If the input mentions his delusion topic, he will increase the value of his fear variable and then begin to express the sequence of statements related to his delusion (妄想).

- Parry was the first known system to pass the Turing Test (in 1972!)
  - Psychiatrists (精神科醫生) could not distinguish **text transcripts of interviews with PARRY** from **transcripts of interviews with real paranoids** (偏執狂患者)

## Two Architectures for Corpus-based Chatbots

- Response by retrieval
  - Use information retrieval to grab a response (that is appropriate to the context) from some corpus
- Response by generation
  - Use a language model or encoder-decoder to generate the response given the dialogue context

- Modern corpus-based chatbots are very data-intensive, and commonly require hundreds of millions or billions of words.



## What Human-Human Conversations to Draw on?

- Transcripts of telephone conversations between volunteers
  - Switchboard corpus of American English telephone conversations
- Movie dialogues
  - Various corpora of movie subtitles, resembling natural conversation in many ways
- Hire human crowd-workers to have conversations, e.g.,
  - The Topical-Chat dataset has 11K crowdsourced conversations on 8 topics
  - EMPATHETICDIALOGUES includes 25K crowdsourced conversations grounded in a situation where a speaker was feeling a specific emotion

## More on Data Curation

- Many systems first **pretrain** on large datasets of pseudo-conversations drawn from social media
  - Twitter (Ritter et al., 2010a), Reddit (Roller et al., 2021), Weibo (微博), and other social media platforms; **tend to be noisy**
- Another common technique is to extract possible responses from knowledge sources (Wikipedia, news stories)
  - By doing so, a chatbot can tell stories or mention facts acquired in that way
- Finally, once a chatbot has been put into practice
  - The turns that humans use to respond to the chatbot can be used as additional conversational data for training or finetuning
  - **Confidence metrics** are required to make sure that these turns come from conversations that are going well

It is also crucial in these cases to remove personally identifiable information (PII).

## Response by Retrieval: Classic IR Method

- Given a user turn  $q$ , and a training corpus of conversations  $C$
- Find in  $C$  the turn (response)  $r$  that is most similar (e.g., in terms of **tf-idf cosine**) to  $q$ , say  $r$

$$\text{response}(q, C) = \arg \max_{r \in C} \frac{\vec{\mathbf{q}} \cdot \vec{\mathbf{r}}}{|\vec{\mathbf{q}}| |\vec{\mathbf{r}}|}$$

- Generally,  $C$  is the training set for the system, and we score each turn  $r$  in  $C$  as a potential response to the context  $q$  selecting the highest-scoring one
- **An alternative way** is to return the response to the turn resembling  $q$ 
  - That is, we first find the most similar turn  $t$  to  $q$  and then return as a response the following turn  $r$

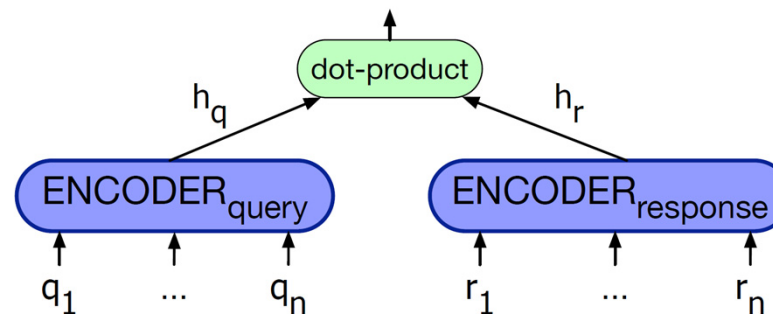
# Response by Retrieval: Neural IR Method

- A **bi-encoder model** is adopted
  - In which we train two separate (BERT) encoders, one to encode the user query and one to encode the candidate response
  - Use the dot product between these two vectors as the score

$$h_q = \text{BERT}_Q(q) [\text{CLS}]$$

$$h_r = \text{BERT}_R(r) [\text{CLS}]$$

$$\text{response}(q, C) = \underset{r \in C}{\text{argmax}} h_q \cdot h_r$$



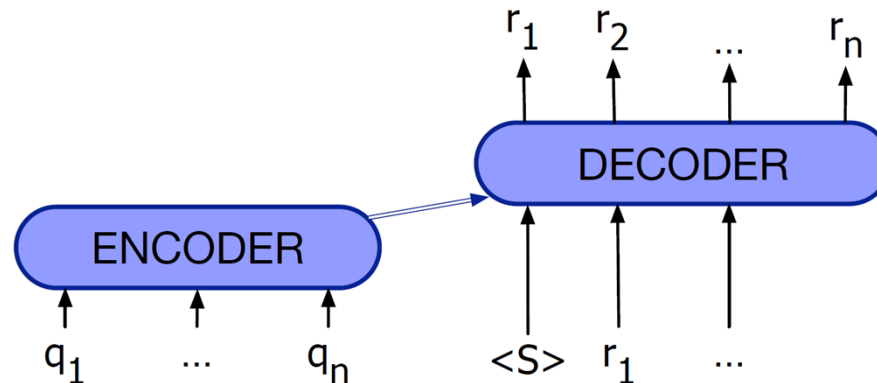
## More on Response by Retrieval

- The IR-based approach can be extended in various ways, such as by using more sophisticated neural architectures (Humeau et al., 2020)
- **Conversational History**: we can use a longer context for the query than just the user's last turn, up to the whole preceding conversation
- Information about the **user** or **sentiment** or other information can also play a role

## Response by Generation (1/2)

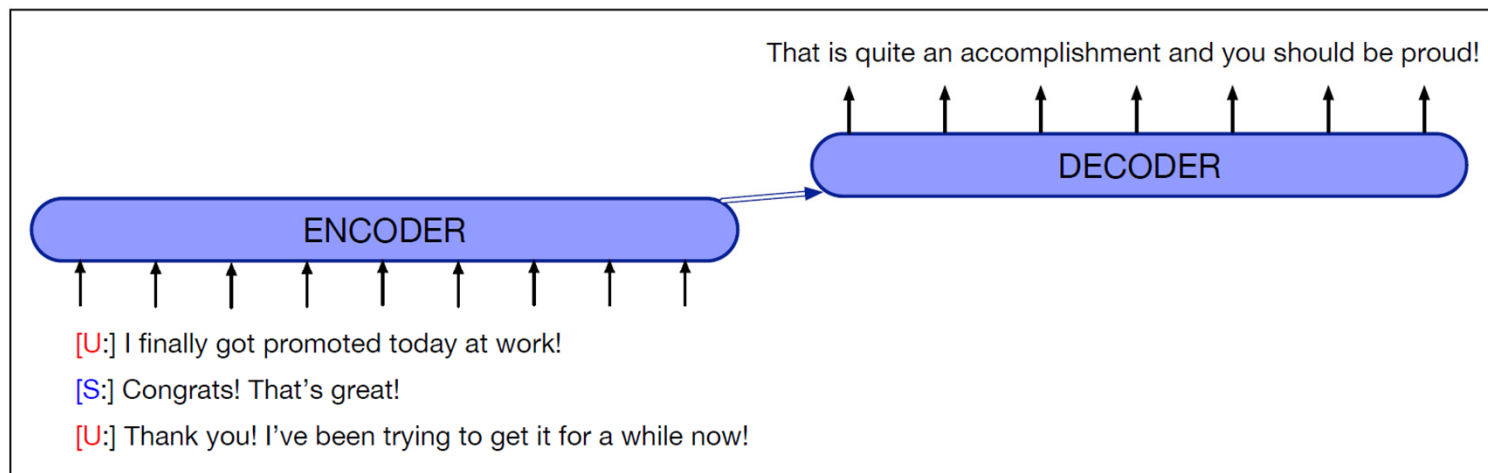
- Think of response production as an encoder-decoder task
- Generate each token  $r_t$  of the response by conditioning on the encoding of the entire query  $q$  and the response so far  $r_1, \dots, r_{t-1}$

$$\hat{r}_t = \operatorname{argmax}_{w \in V} P(w|q, r_1, \dots, r_{t-1})$$



## Response by Generation (2/2)

- An alternative to the encoder-decoder architecture is to fine-tune a large language model (LLM) on a conversational dataset and use the language model directly as a response generator, for example:
  - In the Chirpy Cardinal system (Paranjape et al., 2020), the neural chat component generates responses from GPT-2 (Radford et al., 2019), finetuned on the EmpatheticDialogues dataset (Rashkin et al., 2019)



**Figure 15.7** Example of encoder decoder for dialogue response generation; the encoder sees the entire dialogue context.

## Response by Retrieving and Refining Knowledge

- Seek to generate responses from informative text rather than dialogue, e.g.:
  - To respond to turns like “Tell me something about Beijing”
    - Xiaolce collects sentences from public lectures and news articles
    - And searches them using IR based on query expansion from user’s turn
  - To augment an encoder-decoder model
    - Use IR to retrieve potentially useful passages from **Wikipedia**
    - Then create multiple candidates by concatenating each retrieved Wikipedia sentence to the dialogue context with a separator token
    - Each candidate can be given as the encoder context to **the encoder-decoder model**, which learns to incorporate text from the Wikipedia sentence into its generated “**knowledge-aware**” response

```
Context <SEP> Candidate 1
.
.
Context <SEP> Candidate n
```

Chatbots can be much more interesting and informative if they can response from text knowledge sources other than the dialogue alone.



## One Ongoing Research Problem: Neural Chatbots often Get Repetitive and Boring

A: Where are you going?

B: I'm going to the restroom.

A: See you later.

B: See you later.

A: See you later.

B: See you later.

.

.

A: How old are you?

B: I'm 16.

A: 16?

B: I don't know what you are talking about.

A: You don't know what you are saying.

B: I don't know what you are talking about.

A: You don't know what you are saying.

### Human-Machine Conversations

A number of modifications are thus required to the basic encoder-decoder model to adapt it for the task of response generation.

- Diversity-enhanced versions of beam search (Vijayakumar et al., 2018)
- Diversity-focused training objectives (Li et al., 2016)

## Hybrid Architectures (1/2)

- **Chirpy Cardinal** (Paranjape et al., 2020) response generation from **a series of different generators**:
  - GPT-2 finetuned on EmpatheticDialogues
  - GPT-2 finetuned to paraphrase content from Wikipedia
  - Rule-based movie or music generators that produce scripted conversation about a movie or a musician
    - Asking the user's opinion about a movie
    - Giving a fun fact
    - Asking the user their opinion on an actor in the movie

## Hybrid Architectures (2/2)

- Part of a Conversation with Chirpy Cardinal (Paranjape et al. 2020)

#	Utterance	Entity	Commentary
1	<b>User:</b> let's chat <b>Bot:</b> Hi, this is an Alexa Prize Socialbot. I'd love to get to know you a bit better before we chat! Is it all right if I ask for your name? [ Launch ]	None None	[ Launch ] gives <i>FORCE_START</i> response to begin the conversation.
2	<b>User:</b> my name is chris <b>Bot:</b> Well it's nice to meet you, Chris! I'm excited to chat with you today. [ Launch ] I hope your afternoon is going well. What are your plans for the rest of today? [ Neural Chat ]	None None	[ Launch ] gives a <i>STRONG_CONTINUE</i> response that needs a prompt, which is supplied by [ Neural Chat ] with <i>FORCE_START</i> priority (scripted transition).
3	<b>User:</b> hang out with my friends <b>Bot:</b> That sounds great, I love spending time with friends. What will you do together? [ Neural Chat ]	None None	[ Neural Chat ] generates a GPT-2 response with <i>STRONG_CONTINUE</i> priority.
4	<b>User:</b> maybe watch a movie <b>Bot:</b> Me too! I love watching movies; I get to learn so much about what the world is like outside of the cloud! Have you seen any movies recently? [ Movies ]	None Film	[ Movies ] detects the 'movie' keyword in the user's utterance and gives a <i>FORCE_START</i> response (overriding [ Neural Chat ]), setting current entity to <i>Film</i> .
5	<b>User:</b> i saw the matrix <b>Bot:</b> Nice! Did you like The Matrix? [ Movies ]	The Matrix The Matrix	The [ Entity Linker ] and [ Entity Tracker ] identify <i>The Matrix</i> as the current entity.
6	<b>User:</b> i loved it neo is amazing <b>Bot:</b> Oooh, yeah, I agree. Hey, isn't Keanu Reeves in that movie? What do you think about Keanu Reeves? [ Movies ]	The Matrix Keanu Reeves	[ Movies ] sets current entity to <i>Keanu Reeves</i> .
7	<b>User:</b> i want to talk about something else	None	[ Navigational Intent ] detects <i>negative intent</i> , so [ Entity Tracker ] rejects the current entity.

# Chatbots: Pros and Cons

- Pros:
  - Fun
  - Good for narrow, scriptable applications
- Cons:
  - Systems do not really understand
  - Giving the appearance of understanding may be problematic
  - Rule-based chatbots are expensive and brittle
  - IR-based chatbots can only mirror training data

One important next step in research is figuring out ways to integrate chatbot-type abilities into frame-based dialogue systems (agents).

# Frame-based Dialogue Agents

Bobrow et al., 1977

- Sometimes called "**task-based dialogue agents**"
  - Systems that have the goal of helping a user solve a task like **making a travel reservation** or **buying a product**
- Architecture:
  - First proposed in the GUS system of 1977
  - A knowledge structure representing user intentions
  - One or more **frames** (each consisting a set of slots with values)

## The Frame

- A frame is a kind of knowledge structure (sometimes called **domain ontology**) representing part of the information about user intentions that the system can extract from user sentences
- A frame consists of a collection of slots (to be filled with information of a given **semantic type**), where the semantic type of a slot may be hierarchical
  - Each slot of a frame can take a set of possible **values**, and each associated with a **question** to the user

Slot	Type	Question Template
ORIGIN CITY	city	“From what city are you leaving?”
DESTINATION CITY	city	“Where are you going?”
DEPARTURE TIME	time	“When would you like to leave?”
DEPARTURE DATE	date	“What day would you like to leave?”
ARRIVAL TIME	time	“When do you want to arrive?”
ARRIVAL DATE	date	“What day would you like to arrive?”

**Figure 15.10** A frame in a frame-based dialogue system, showing the type of each slot and a question used to fill the slot.

For example, the date type in GUS is itself a frame with slots with types like integer or members of sets of weekday names

## Two Basic Dialogue Architectures

- The GUS architecture
  - Sometimes just called "frame-based" architecture
  - Over 40 years old, but still used in most industrial task-based dialogue agents
  - Focus on a set of hand-build production rules for filling frames and taking actions

D. G. Bobrow et al., Daniel G., GUS, a frame-driven dialog system." *Artificial Intelligence*, 1997.

- The dialogue-state architecture
  - Extension of GUS
  - More common in research systems
  - Some aspects are making their way into industrial systems

## Control Structure for Frame-based Dialogue

- System asks questions of user, filling any slots that user specifies
- User might fill many slots at a time:

I want a flight from San Francisco to Denver one way leaving after five p.m. on Tuesday.

- The system fills all the relevant slots, and then continues asking questions to fill the remaining slots, skipping questions associated with filled slots
- When frame is filled, perform database querying



## Attachment of Condition-Action Rules

自動填入資訊到相關slots

- For example, some rules can be attached to the DESTINATION slot for the plane booking frame
  1. Once the user has specified the destination
    - Enter that city as the default *StayLocation* for the hotel booking frame
  2. Once the user has specified **DESTINATION DAY** for a short trip
    - Automatically copy as **ARRIVAL DAY**

I want a flight from San Francisco to Denver one way leaving after five p.m. on Tuesday.

# Natural Language Understanding for Filling Dialog Slots (1/2)

## 1. Domain classification

- Asking weather? Booking a flight? Programming alarm clock?

## 2. Intent Determination

- Find a Movie, Show a Flight, Remove Calendar Appointments

## 3. Slot Filling

- Extract the actual slots and fillers from a user's utterances

## Natural Language Understanding for Filling Dialog Slots (2/2)

- Example 1:

Show me morning flights from Boston to SF on Tuesday.

```
DOMAIN:      AIR-TRAVEL
INTENT:      SHOW-FLIGHTS
ORIGIN-CITY: Boston
ORIGIN-DATE: Tuesday
ORIGIN-TIME: morning
DEST-CITY:   San Francisco
```

- Example 2:

Wake me tomorrow at six.

```
DOMAIN:  ALARM-CLOCK
INTENT:  SET-ALARM
TIME:    2017-07-01 0600-0800
```

## Rule-based Slot Filling

- First, write regular expressions or grammar rules
  - We might just define a regular expression for recognizing the SET-ALARM intent

Wake me (up) | set (the|an) alarm | get me up

- Second, do text normalization
  - The fillers are put into some sort of canonical form, for example by normalizing dates

## Other Key Components of a Dialogue System

- The **ASR** (automatic speech recognition) component takes audio input from a phone or other device and outputs a transcribed string of words
  - The ASR component can also be made dependent on the dialogue state
  - A language model that is completely dependent on dialogue state is called a restrictive grammar
  - Real systems usually mix restrictive grammars with more general language models
- The **NLG** (natural language generation) module produces the utterances that the system says to the user
  - Frame-based systems tend to use **template-based generation** in which all or most of the words in the sentence to be uttered to the user are prespecified by the dialogue designer
  - **Sentences created by these templates are often called prompts**
  - Templates might be completely fixed (like 'Hello, how can I help you?'), or can include some variables that are filled in by the generator

What time do you want to leave CITY-ORIG?  
Will you return to CITY-ORIG from CITY-DEST?

## More on NLG

- It is possible to **do some simple grounding** even in templated generation

System: Did you want to review some more of your personal profile?  
Caller: No.  
System: What's next?



System: Did you want to review some more of your personal profile?  
Caller: No.  
System: *Okay* what's next?

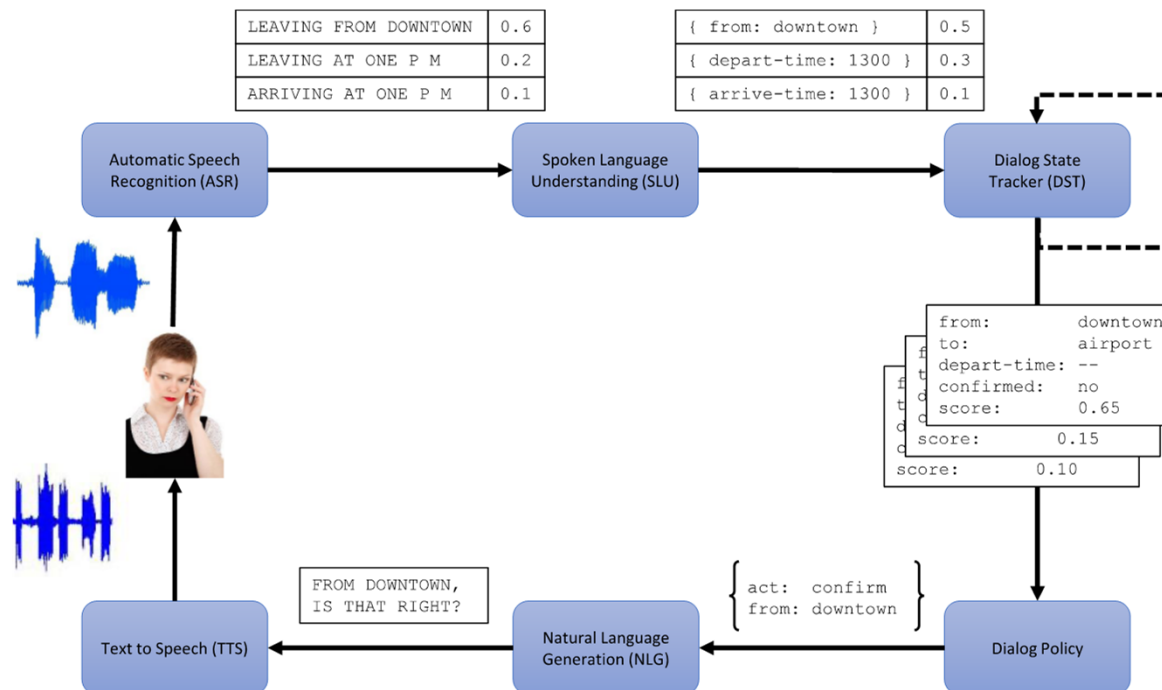
grounding (acknowledgement)

## Summary: Simple Frame-based Architecture

- Like many rule-based approaches
- Pros:
  - High precision
  - Can provide coverage if the domain is narrow
- Cons:
  - Can be expensive and slow to create rules
  - Can suffer from recall problems

# The Dialogue-State Architecture

- Modern research systems for task-based dialogue are based on a more sophisticated version of the frame-based architecture called the **dialogue-state** or **belief-state** architecture





# Components in a Dialogue-State Architecture

- **NLU/SLU**
  - Extract **slot fillers** from the user's utterance using machine learning
- **Dialogue State Tracker**
  - Maintain the **current state** of the dialogue (user's most recent dialogue act, set of slot-filler constraints from user)
- **Dialogue Policy (Dialogue Manager)**
  - Decide **what the system should do** or say next
    - GUS policy: Ask questions until the frame was full and then report back the results of some database query (or information retrieval)
  - More sophisticated: know **when to answer questions, when to ask a clarification question**, etc.
- **NLG**
  - Produce more natural, less templated utterances

## Dialogue Acts (1/2)

- Combine the ideas of **speech acts** and **grounding** into a single representation
  - **An example:** 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

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

# Intents and Dialogue Acts

- **Intents** and **dialogue acts** are two concepts that are often used in the design and development of conversation agents
  - An intent refers to the purpose or goal of a user's message, or the user's intended meaning
  - For example, a user message like "What is the weather like today?" might be mapped to an intent like "get weather information".
- Dialogue acts, on the other hand, refer to the role or function that a message plays in a conversation
  - Dialogue acts can include things like making a request, giving a command, making a statement, or asking a question
  - Dialogue acts are often used to help the chatbot understand the context and purpose of a user's message, and to determine how to respond appropriately
- Intents and dialog acts are often used together in chatbot development
  - They both help to inform the chatbot's understanding of a user's message and the context in which it was sent

## Dialogue Acts (2/2)

- A sample dialogue from [the HIS System](#) of Young et al. (2010) using the dialogue acts defined in the previous page

Utterance	Dialogue act
U: Hi, I am looking for somewhere to eat.	hello( <b>task = find, type=restaurant</b> )
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()

user intent

## Slot Filling: A Machine Learning Approach

- Machine learning classifiers to map words to **semantic frame-fillers**
- Given a set of labeled sentences

Input: "I want to fly to San Francisco on Monday please"

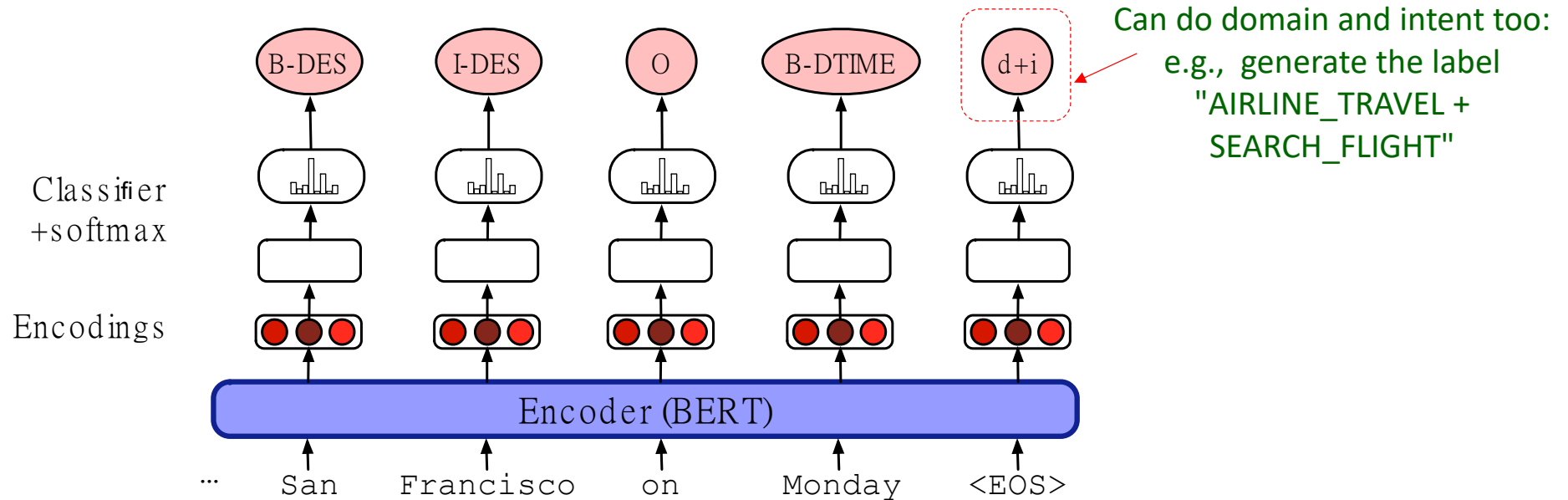
Output: Destination: SF

Depart-time: Monday

- Build a classifier to map from one to the other
- Requirements: Lots of labeled data

# Slot Filling: BIO Tagging

- The idea of the BIO tagging paradigm is to train a classifier to label each input word with a tag that tells us what slot (if any) it fills
  - A tag for the beginning (B) and inside (I) of each slot label, and one for tokens outside (O) any slot label



## Once We Have the BIO Tag of a Sentence

0 0    0 0    0 B-DES I-DES    0 B-DEPTIME I-DEPTIME 0  
I want to fly to San Francisco on Monday afternoon please

- We can extract the filler string for each slot
- Then, **normalize** it to the correct form in the ontology like "SFO" for San Francisco using **homonym dictionaries** (SF=SFO=San Francisco, are the same place)

## The Task of Dialogue State Tracking

- The job of the **dialogue-state tracker** is to determine both the **current state** of the frame (the fillers of each slot), as well as the user's most recent dialogue act
  - The dialogue-state thus includes more than just the slot-fillers expressed in the current sentence; **it includes the entire state of the frame at this point**, summarizing all of the user's constraints

```
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
```



## More on Dialogue State Tracking

- An example: “I'd like Cantonese food near the Mission district.”

```
inform(food=cantonese, area=mission) .
```

Dialogue Act



- Dialogue act interpretation algorithm:
  - 1-of-N supervised classification to choose `inform`
  - Based on encodings of current sentence + prior dialogue acts
- Simple dialogue state tracker:
  - Run a slot-filler after each sentence

## Dialogue Policy (1/2)

- The goal of the **dialogue policy** is to decide what **action** the system should take next, that is, what **dialogue act** to generate
- At turn  $i$  in the conversation, we want to predict which action  $A_i$  to take, based on the entire dialogue state (history)
  - The state could mean the entire sequence of dialogue acts from the system ( $A$ ) and from the user ( $U$ )

$$\hat{A}_i = \operatorname{argmax}_{A_i \in A} P(A_i | A_1, U_1, \dots, A_{i-1}, U_{i-1})$$

- Simplify the above equation by just conditioning on the current dialogue state (filled frame slots) and the last turn and turn by system and user

$$\hat{A}_i = \operatorname{argmax}_{A_i \in A} P(A_i | \text{Frame}_{i-1}, A_{i-1}, U_{i-1})$$

## Dialogue Policy (2/2)

- The aforementioned probabilities can be estimated by a **neural classifier**
  - Use neural representations of the slot fillers (for example as spans) and the utterances (for example as sentence embeddings computed over contextual embeddings)
- More sophisticated models train the policy via **reinforcement learning (RL)**
  - To decide which action to take, a reinforcement learning system gets a reward at the end of the dialogue, and uses that reward to train a policy to take actions
- For example, in a movie-recommendation dialogue system, the action space could have only three actions: **EXECUTE**, **CONFIRM**, and **ELICIT**
  - EXECUTE sends a query to the database and answers the user's question
  - CONFIRM clarifies the intent or slot with the users (e.g., “Do you want movies directed by Christopher Nolan?”)
  - ELICIT asks the user for missing information (e.g., “Which movie are you talking about?”)

Fazel-Zarandi et al. (2017)

## Policy Example: Confirmation and Rejection

- Dialogue systems make errors, so they have to make sure they they have achieved the correct interpretation of the user's input (understood the user)
- Two important mechanisms:
  - Confirming: have understandings with the user
  - Rejecting: Utterances that the system is likely to have **misunderstood**

## Explicit Confirmation Strategy

- A system asks the user a direct question to confirm the system's understanding

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 Strategy

- A system instead can demonstrate its understanding through a grounding strategy
- For example, repeating back the system's understanding as part of asking the next question

U: I want to travel to Berlin

S: **When do you want to travel to Berlin?**

U2: Hi I'd like to fly to Seattle Tuesday Morning

A3: **Traveling to Seattle on Tuesday, August eleventh in the morning.**

Your full name?

## Comparison between Explicit and Implicit Confirmation Strategies

- Explicit and implicit confirmation have complementary strengths
- Explicit confirmation makes it easier for users to correct the system's misrecognitions
  - Since a user can just answer “no” to the confirmation question
- Explicit confirmation is awkward and increases the length of the conversation
- The explicit confirmation dialogue fragments above sound non-natural and definitely non-human
- Implicit confirmation is much more conversationally natural

## Rejection (1/2)

- A system gives the user a prompt like

*I'm sorry, I didn't understand that.*

- Sometimes utterances are rejected multiple times
  - This might mean that the user is using language that the system is unable to follow
  - Thus, when an utterance is progressively rejected, systems often follow a strategy of **progressive prompting** or **escalating prompting**

當答非所問時 ...

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.



## Rejection (2/2)

- An alternative strategy for error handling is **rapid reprompting**, in which the reprompting system rejects an utterance just by saying

*I'm sorry, I didn't understand that.*

- Only if the caller's utterance is rejected a second time does the system start applying progressive prompting
- Cohen et al. (2004) summarize experiments showing that users greatly prefer **rapid reprompting** as a first-level error prompt

## Using Confidence to Decide Whether to Confirm

- ASR or NLU systems can assign a **confidence** value, indicating how likely they are that they understood the user, for example
  - Acoustic log-likelihood of the utterance
    - Assigned by an ASR system
  - Prosodic features
    - Utterances with large F0 excursions or longer durations, or those preceded by longer pauses, are likely to be misrecognized
  - Or alternatively, the cost of making an error
    - For example, explicit confirmation is common before a flight is actually booked or money in an account is moved
- Systems could use set confidence thresholds

$< \alpha$	low confidence	reject
$\geq \alpha$	above the threshold	confirm explicitly
$\geq \beta$	high confidence	confirm implicitly
$\geq \gamma$	very high confidence	don't confirm at all

a four-tiered level of confidence

# Natural Language Generation

- NLG in information-state architecture modeled in two stages:
  - **Content Planning** (what to say)
  - **Sentence Realization** (how to say it)
- We'll focus on **sentence realization** here

## Sentence Realization (1/3)

- Assume **content planning** has been done by the **dialogue policy**, which has
  - Chosen the **dialogue act** to generate,
  - Chosen some attributes (**slots and values**) that the planner wants to say to the user
    - Either to give the user the answer, or as part of a confirmation strategy
- 2 samples of Input and Output for Sentence Realizer

```
recommend(restaurant name= Au Midi, neighborhood = midtown,  
cuisine = french
```

- 1 Au Midi is in Midtown and serves French food.
- 2 There is a French restaurant in Midtown called Au Midi.

```
recommend(restaurant name= Loch Fyne, neighborhood = city  
centre, cuisine = seafood)
```

- 3 Loch Fyne is in the City Center and serves seafood food.
  - 4 There is a seafood restaurant in the City Centre called Loch Fyne.
-

## Sentence Realization (2/3)

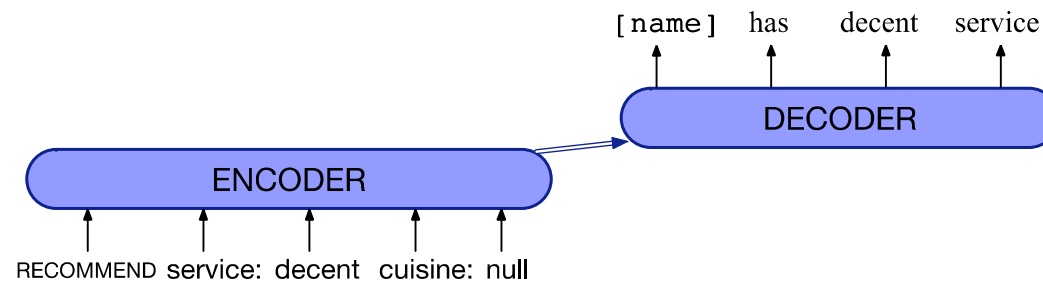
- Training data is hard to come by
  - we are unlikely to see every possible restaurant with every possible attribute in many possible differently worded sentences
- **Delexicalization**: A common way to improve generalization:
  - The process of replacing words in the training set that represent slot values with a **generic placeholder token**

```
recommend(restaurant name= Au Midi, neighborhood = midtown,  
cuisine = french
```

- 1 `restaurant name` is in `neighborhood` and serves `cuisine` food.
- 2 There is a `cuisine` restaurant in `neighborhood` called `restaurant name`.

## Sentence Realization (3/3)

- Map from **frames** to **delexicalized sentences** with **encoder-decoder models**
  - Trained on large hand-labeled corpora of task-oriented dialogue



- Output:

`restaurant_name has decent service`

- **Relexicalize**: we can then use the input frame from the **content planner** to relexicalize (fill in the exact restaurant or neighborhood or cuisine)

`Au Midi has decent service`

# Evaluating Chatbots and Task-based Dialogue Systems

- Chatbots:
  - Have goals like being enjoyable to users
  - Evaluations are mainly done by human (participant/observer) evaluation
    - **Participant**: the human who talked to the chatbot
    - **Observer**: a third party who reads a transcript of a human/chatbot conversation
- Task-based dialogue Systems
  - Have goal like the completion of a task to book a flight
  - Evaluations are mainly done by measuring task performance

# Participant Evaluation

See et al., 2019

- Human chats with a chatbot for 6 turns and rates 8 dimensions of quality:  
**avoiding repetition, interestingness, making sense, fluency, listening, inquisitiveness, humanness, engagingness**

## A few examples:

**(1) Avoiding Repetition:** How repetitive was this user?

- Repeated themselves over and over
- Sometimes said the same thing twice
- Always said something new

**(3) Making sense:** How often did this user say something which did not make sense?

- Never made any sense
- Most responses did not make sense
- Some responses did not make sense
- Everything made perfect sense

**(8) Engagingness:** How much did you enjoy talking to this user?

- Not at all
- A little
- Somewhat
- A lot



## Chatbots: Observer Evaluation (1/3)

- Sometimes we are interested in having raters assign a score to **each system turn**
  - For example, (Artstein et al., 2009) have raters mark how **coherent** each turn is
  - Or, we may just want a single high-level score to know if system A is better than system B
- The **acute-eval metric** (Li et al., 2019)
  - An observer evaluation in which annotators look at two separate human-computer conversations (A and B) and choose the one in which the dialogue system participant performed better (兩個chatbots何者較好?)

## Chatbots: Observer Evaluation (2/3)

- More on the [acute-eval metric](#)

Annotators look at two conversations (A + B) and decide which is better

- **Engagingness:** Who would you prefer to talk to for a long conversation?
- **Interestingness:** If you had to say one of these speakers is interesting and one is boring, who would you say is more interesting?
- **Humanness:** Which speaker sounds more human?
- **Knowledgeable:** If you had to say that one speaker is more knowledgeable and one is more ignorant, who is more knowledgeable?

# Chatbots: Observer Evaluation (3/3)

Hi how are you today

Hi, I'm pretty good! Just listening to some aerosmith, they're my fave :) whatre you up to?

I am listening to some italian music

Italian music, nice! What do you do for work?

I'm a professional wrestler

Oh wow! I'm a construction worker, I've been doing it since I was 16.

What do you do for a living

I'm a construction worker haha, just told you! Whats your favorite type of food?

My favorite is garlic bread

Yum! I love mexican food, so good :) do you have any hobbies?

I like to workout and go to the gym

We're a bit different- I love watching nascar and ufc. They're so fun!

Hello there, how are you?

I am doing great. How are you?

I am great, I did something crazy for me and colored my hair blue!

I have a daughter and a son who also love blue colored balls. You should meet them

Well that neat, I got a new car my mother gave so maybe I could see them!

It is a beautiful city. And, I try to be... Just cannot afford a bigger house atm.

I am sorry to hear that, I feel bad going out of town for spring break now.

Ok. I going to school in the spring for casino manager

Well I turn 29 next week, I wonder if that is a good age to apply as one.

My grandmother just died from lung cancer, sucks

**Who would you prefer to talk to for a long conversation?**

I would prefer to talk to **Speaker 1**       I would prefer to talk to **Speaker 2**

**Please provide a brief justification for your choice (a few words or a sentence)**

Please enter here...

The ACUTE-EVAL method asks annotators to compare two dialogues and choose between Speaker 1 (light blue) and Speaker 2 (dark blue), independent of the gray speaker (Li et al., 2019).

## Chatbots: Automatic Evaluation Remains an Open Problem (1/2)

- Automatic evaluations are generally not used for chatbots
  - This is because computational measures of generation performance like BLEU or ROUGE or embedding dot products between a chatbot's response and a human response correlate very poorly with human judgments
  - These methods perform poorly because **there are so many possible responses to any given turn**
  - Simple word-overlap or semantic similarity metrics work best when the space of responses is small and lexically overlapping

## Chatbots: Automatic Evaluation Remains an Open Problem (2/2)

- **Adversarial Evaluation:** One current research direction
  - Inspired by Turing Test
  - Train a “**Turing-like**” classifier to distinguish between human responses and machine responses.
  - The more successful a dialogue system is at fooling the evaluator, the better the system

## Task-based Dialogue Systems: Evaluations (1/3)

- For task-based dialogue, if the task is unambiguous, we can simply measure absolute task success (viz. **task error rate**, an extrinsic metric)
  - Did the system book the right plane flight, or put the right event on the calendar?
- Or, measure the **slot error rate** (aka **concept error rate**) for a sentence

“Make an appointment with Chris at 10:30 in Gates 104”

Slot	Filler
PERSON	Chris
TIME	11:30 a.m.
ROOM	Gates 104

- End-to-end evaluation (Task Success)
- Slot Error Rate for a Sentence
$$\frac{\text{\# of inserted/deleted/substituted slots}}{\text{\# of total reference slots for sentence}}$$

**Slot error rate:** 1/3

**Task success:** At end, was the correct meeting added to the calendar?

## Task-based Dialogue Systems: Evaluations (2/3)

- To get a **more fine-grained idea of user happiness**, we can compute a user satisfaction rating, having users interact with a dialogue system to perform a task and then having them complete a questionnaire like below

<b>TTS Performance</b>	Was the system easy to understand ?
<b>ASR Performance</b>	Did the system understand what you said?
<b>Task Ease</b>	Was it easy to find the message/flight/train you wanted?
<b>Interaction Pace</b>	Was the pace of interaction with the system appropriate?
<b>User Expertise</b>	Did you know what you could say at each point?
<b>System Response</b>	How often was the system sluggish and slow to reply to you?
<b>Expected Behavior</b>	Did the system work the way you expected it to?
<b>Future Use</b>	Do you think you'd use the system in the future?

# Task-based Dialogue Systems: Evaluations (3/3)

- Other Heuristics

- **Efficiency cost:**

- Total elapsed time for the dialogue in seconds
    - The number of total turns or of system turns
    - Total number of queries
    - “turn correction ratio”: % of turns that were used to correct errors

- **Quality cost:**

- Number of ASR rejection prompts
    - Number of times the user had to barge in (interrupt the system)



## Summary

- Conversational agents are crucial speech and language processing applications that are already widely used commercially
- Chatbots are conversational agents designed to mimic the appearance of informal human conversation
- Most commercial task-based dialogue systems use the GUS or frame-based architecture
- Dialogue systems are a kind of human-computer interaction