Dialog Systems and Chatbots

Dr. Demetrios Glinos
University of Central Florida

CAP6640 – Computer Understanding of Natural Language

Today

- Conversational Agents
- Rule-Based Chatbots
- Corpus-Based Chatbots
- Frame-Based Dialog Agents
- Evaluating Dialogue Systems
- Dialogue System Design

Conversational Agents and Dialog

- Conversation
  - The hallmark of what it means to be human
  - Everyone engages in conversation
  - Topics range from serious to trivial

- Conversational agents (dialog systems)
  - Communicate with users in natural language
  - text and/or speech
  - useful for interface to robots

- Dialog "turns"
  - converstion is treated like a game
  - each participant's turn can be a word, sentence, or more
  - useful measure for dialog systems

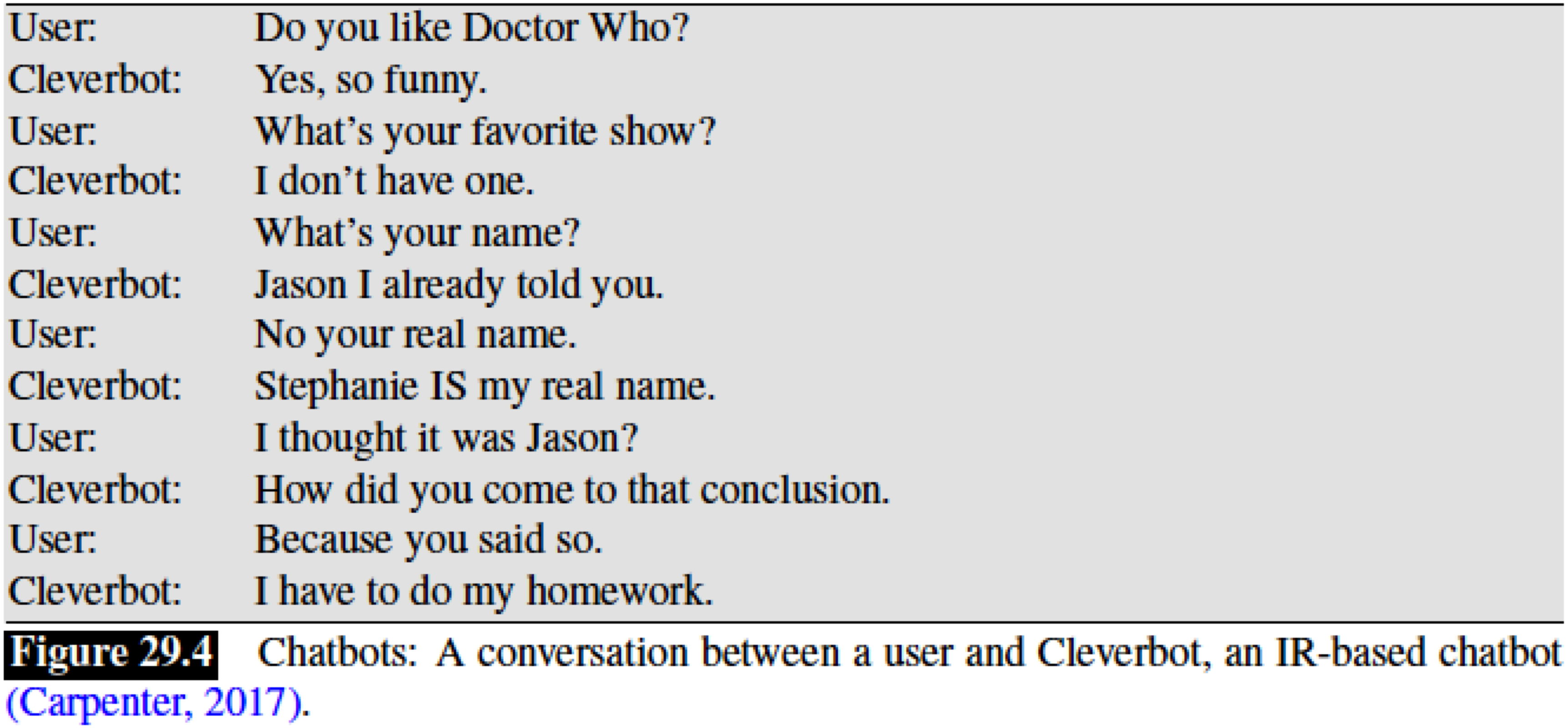
Task-oriented dialog agents

- Designed for a particular task
- Set up to have short conversations (usually < 6 interactions)
- Current crop of phone/home digital assistants
  - Siri, Cortana, Alexa, Google Now/Home, etc.
  - travel directions, find restaurants, make calls, send texts, play music, control home appliances, etc.
- Commercial conversational agents
  - answer customer questions, address problems
  - tutoring systems

Chatbots

- Designed for extended conversations
  - although often used to refer to even task-oriented agents
- Mimic unstructured "chats" typical of human-human conversations
  - not focused on a particular task like booking air travel
- Often used for Turing test competitions
- Often used for entertainment
  - e.g., Microsoft's Xioaice (Little Bing) service which chats in Chinese on text messaging platforms like Weibo
  - e.g., Cleverbot

Example: Cleverbot conversation



User: Do you like Doctor Who?
Cleverbot: Yes, so funny.
User: What's your favorite show?
Cleverbot: I don't have one.
User: What's your name?
Cleverbot: Jason I already told you.
User: No your real name.
Cleverbot: Stephanie IS my real name.
User: I thought it was Jason?
Cleverbot: How did you come to that conclusion.
User: Because you said so.
Cleverbot: I have to do my homework.

Figure 29.4 Chatbots: A conversation between a user and Cleverbot, an IR-based chatbot (Carpenter, 2017).

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

Today

- Conversational Agents
- Rule-Based Chatbots
- Corpus-Based Chatbots
- Frame-Based Dialog Agents
- Evaluating Dialogue Systems
- Dialogue System Design

The first chatbot: ELIZA

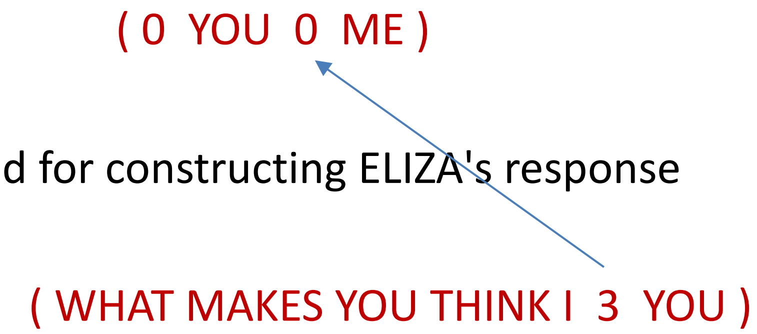
- ELIZA (Weizenbaum, 1966)
    - the most important chatbot dialog system
    - architecture still used in some current chatbots (e.g., ALICE)

- simulated a Rogerian psychotherapist
    - method involves reflecting a patient's statements back at them
    - e.g., PATIENT : "I went for a long boat ride."
      ELIZA : "Tell me about boats."

→ It is easy to assume not that ELIZA doesn't know what a boat is, but rather that she had some conversational purpose in mind

ELIZA Operation

- ELIZA uses a pattern/transform set of rules
  - pattern - used for extracting key words from user input



(0 YOU 0 ME)

  - transform – used for constructing ELIZA's response

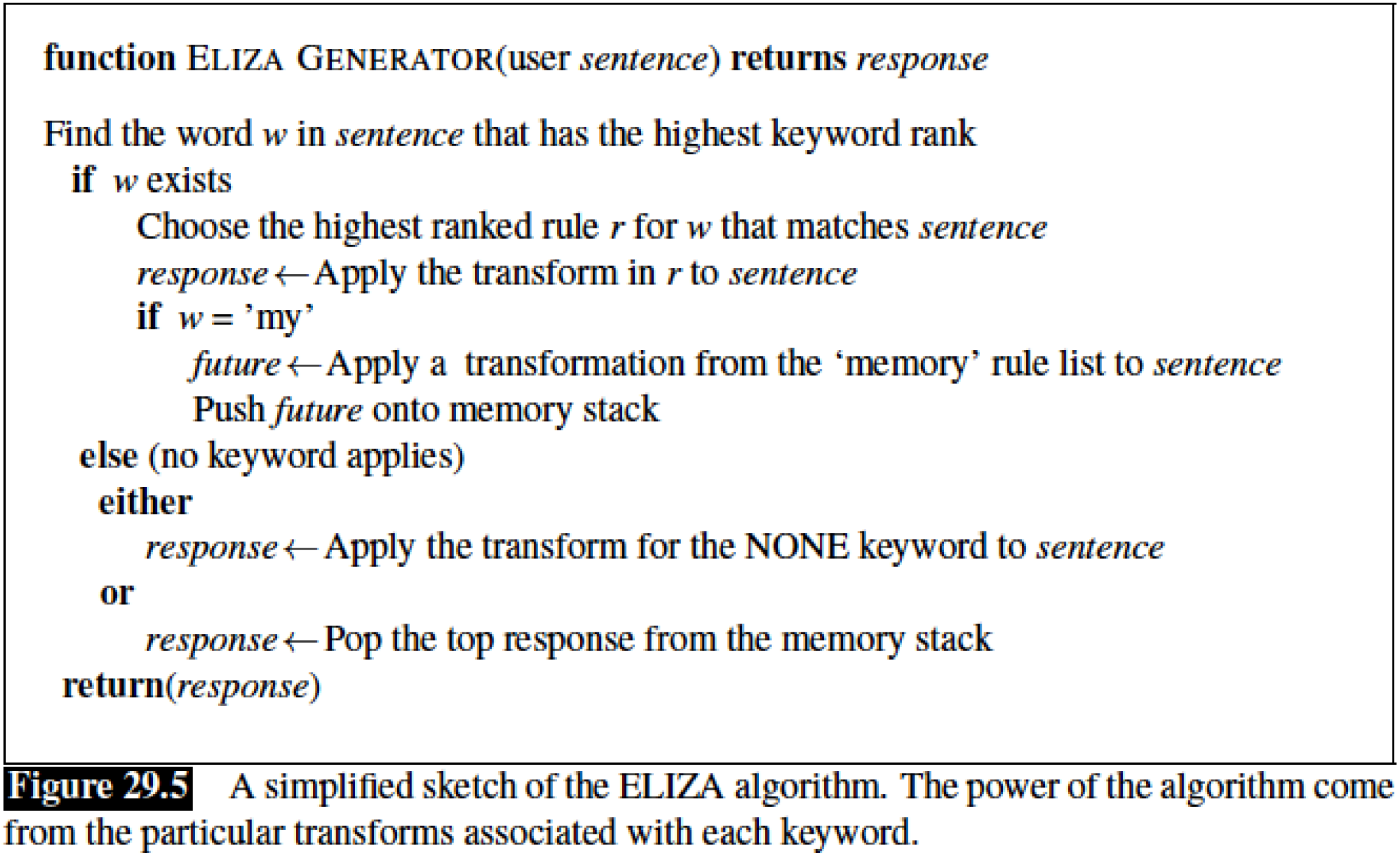
(WHAT MAKES YOU THINK I 3 YOU)

- example:

PATIENT: You hate me
ELIZA : What makes you think I hate you

- Each pattern/rule is linked to a keyword that might occur in a user sentence
- Keywords are given a priority order (rank)

Siimplified ELIZA Algorithm



```
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 stack
  else (no keyword applies)
    either
      response ← Apply the transform for the NONE keyword to sentence
    or
      response ← Pop the top response from the memory stack
  return(response)
```

Figure 29.5 A simplified sketch of the ELIZA algorithm. The power of the algorithm come from the particular transforms associated with each keyword.

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

ELIZA: Special Cases

- If no keyword matches, ELIZA chooses a non-committal response or pop a response from the MEMORY stack

- Non-committal responses
  - PLEASE GO ON
  - THAT'S VERY INTERESTING
  - I SEE

- MEMORY stack
  - if "my" is the highest ranked word, ELIZA randomly selects a stransform on the MEMORY list, applies the transform, and saves it on the MEMORY stack
  - (0 MY 0 = LETS DISCUSS FUTHER WHY YOUR 3)
  - (0 MY 0 = EARLIER YOU SAID YOUR 3)
  - (0 MY 0 = DOES THAT HAVE ANYTHING TO DO WITH THE FACT THAT YOUR 3)

PARRY

- PARRY (Colby et al., 1971)
  - written by a psychiatrist
  - simulated a patient with pararoid schizophrenia

- System included a model of its own mental state
  - included affect variables for the agent's levels of fear and anger
  - certain topics could result in PARRY being more angry or mistrustful
  - if anger threshold exceeded, would choose from a set of "hostile" outputs
  - if inputs mentioned PARRY's delusion topic, fear variable would increase, etc.

- The first known system to pass a version of the Turing test (1972)
  - real psychiatrist were asked to judge transcripts of conversations between other psychiatrists and either real patients or PARRY
  - the psychiatrists were able to guess which were human and which were PARRY only 48% of the time

Today

- Conversational Agents
- Rule-Based Chatbots
- Corpus-Based Chatbots
- Frame-Based Dialog Agents
- Evaluating Dialogue Systems
- Dialogue System Design

Corpus-Based Chatbots

- Instead of rules, mine conversations from corpora
  - human-human conversations
  - human side of human-computer conversations
  - even from non-dialog text

- Corpora
  - e.g., social media platform data, Twitter, movie dialog

- Basic types
  - information retrieval
  - supervised machine learning

- Like rule-based:
  - reflexive - conversational context not modeled
  - often called response generation systems (similar to QA systems)

IR-Based Chatbots

- Basic idea:
  - Respond to user's turn X by repeating some turn Y from a corpus
  - e.g., Twitter, Sina Weibo, movie dialog
  - can also mine actual human interactions with the chatbot

- Given user query q and conversational corpus C:

1. Return the response to the most similar turn (using, e.g., cosine similarity)

$$r = response\left(\operatorname*{argmax}_{t \in C} \frac{q^T t}{|q||t|}\right)$$

2. Return the most similar turn

$$r = \operatorname*{argmax}_{t \in C} \frac{q^T t}{|q||t|}$$

← This one seems to work better in practice

Sequence-to-Sequence Chatbots

- Classic supervised machine learning requires choosing
    - size of input layer
    - size of output layer
    - everything in between
- But conversational inputs and responses can be variable and different lengths
- Solution: **seq2seq** ("sequence-to-sequence") modeling
    - encoder/decoder with fixed-size intermediate representation

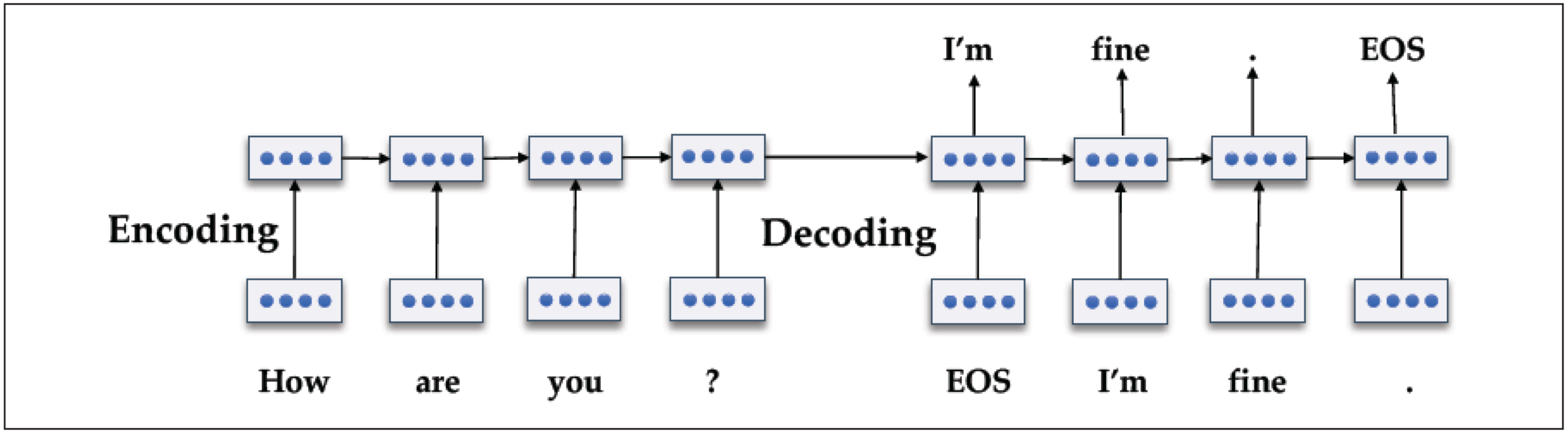


Figure 29.6 A sequence to sequence model for neural response generation in dialog.

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

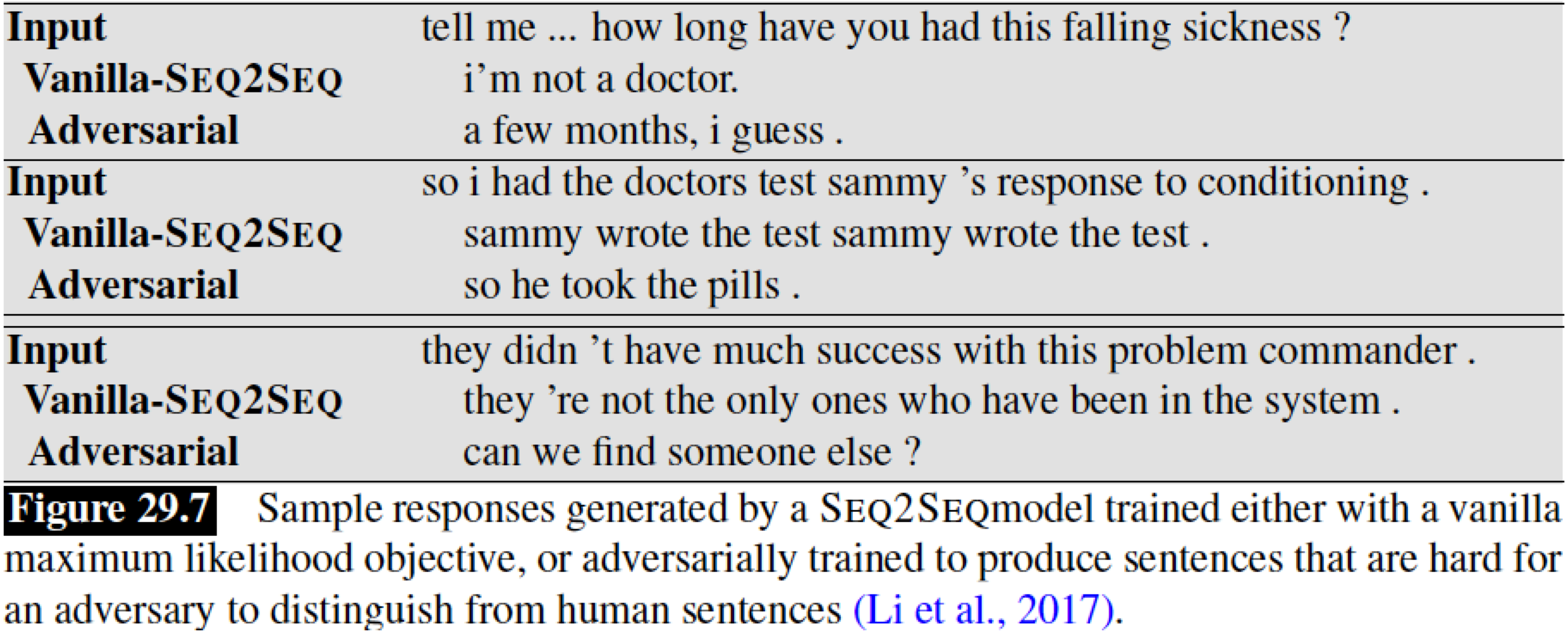
Issues with seq2seq Modeling

- Basic models tend to produce predictable, but repetitive and dull responses
  - e.g., "I'm OK", "I don't know"
  - solution: change the objective function for training to a mutual information objective, or using a beam decoder to keep more diverse responses in the beam

- Modeling the context
  - solved by allowing the model to see prior turns or summaries of prior turns

- Coherence
  - responses don't generally cohere across multiple turns
  - solution: reinforcement learning in adversarial context

Model Comparison



Input	tell me ... how long have you had this falling sickness ?
Vanilla-SEQ2SEQ	i'm not a doctor.
Adversarial	a few months, i guess .
Input	so i had the doctors test sammy 's response to conditioning .
Vanilla-SEQ2SEQ	sammy wrote the test sammy wrote the test .
Adversarial	so he took the pills .
Input	they didn 't have much success with this problem commander .
Vanilla-SEQ2SEQ	they 're not the only ones who have been in the system .
Adversarial	can we find someone else ?

Figure 29.7 Sample responses generated by a SEQ2SEQmodel trained either with a vanilla maximum likelihood objective, or adversarially trained to produce sentences that are hard for an adversary to distinguish from human sentences (Li et al., 2017).

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

Note how responses by adversarial model seem more "natural"

Evaluating Chatbots

- Slot-filling evaluations (used for task-based dialogue) are not appropriate
  - chats are unstructured

- Word-overlap metrics like BLEU (for machine translation) do not correlate well with human judgments
  - reason: too many real-world possible responses to any given turn, unlike MT

- Human evaluation is generally required
  - However, developing automated evaluators is a current active area, training on a corpus of human judgments
  - Adversarial evaluation – idea is to train a "Turing-like" evaluator classifier to distinguish between human and computer-generated responses

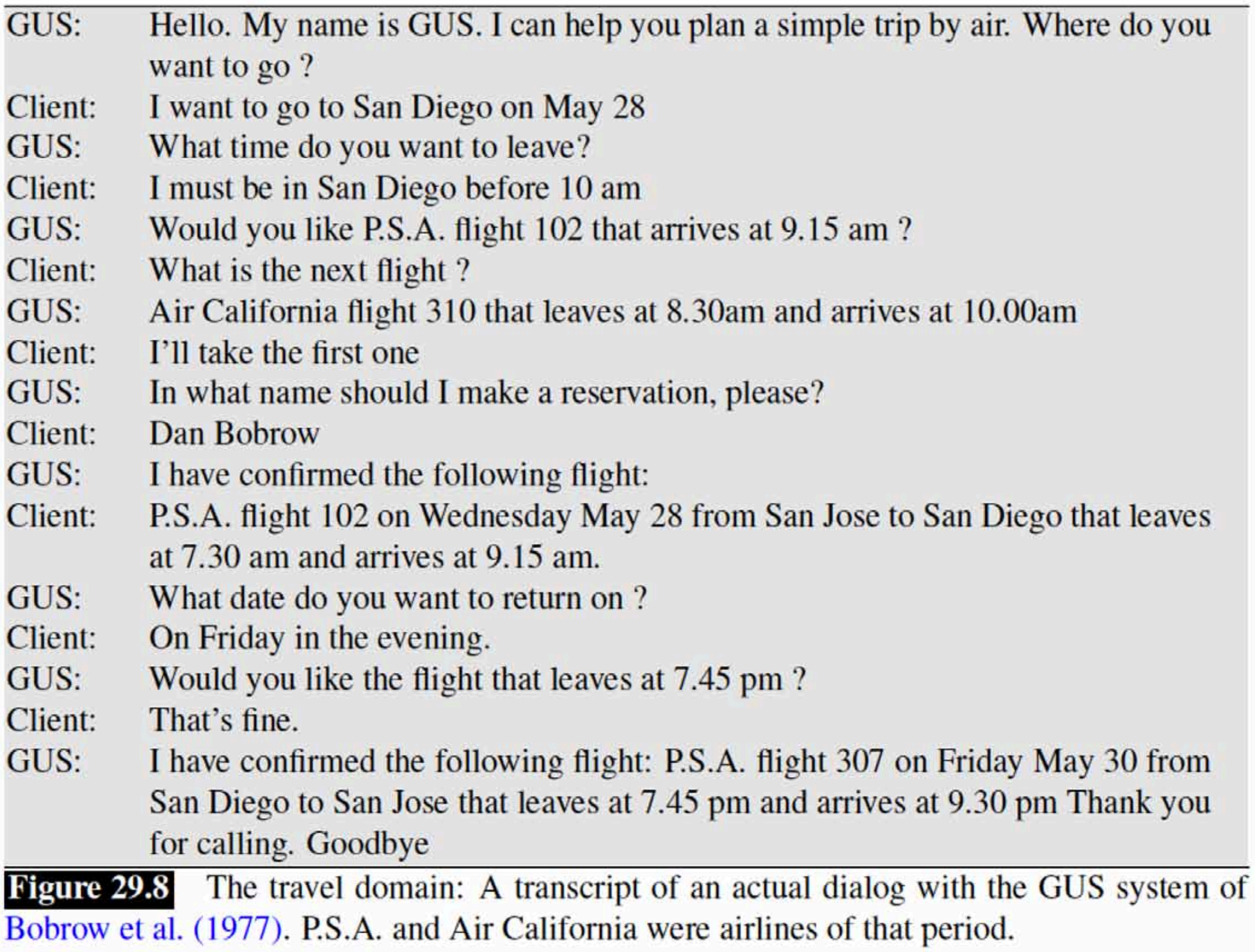
Today

- Conversational Agents
- Rule-Based Chatbots
- Corpus-Based Chatbots
- Frame-Based Dialog Agents
- Evaluating Dialogue Systems
- Dialogue System Design

Frame-Based Dialog Agents

- Used for modern task-based dialog systems
- Based on a domain ontology
  - knowledge base of kinds of intentions the system can extract from user sentences
- Basic idea:
  - The ontology defines frames
  - each frame is a collection of slots
  - each slot can take on only prescribed values
- First introduced in the GUS travel planning system (1977)
- GUS architecture is basis for most current commercial digital assistants

Sample GUS Conversation



GUS:	Hello. My name is GUS. I can help you plan a simple trip by air. Where do you want to go ?
Client:	I want to go to San Diego on May 28
GUS:	What time do you want to leave?
Client:	I must be in San Diego before 10 am
GUS:	Would you like P.S.A. flight 102 that arrives at 9.15 am ?
Client:	What is the next flight ?
GUS:	Air California flight 310 that leaves at 8.30am and arrives at 10.00am
Client:	I'll take the first one
GUS:	In what name should I make a reservation, please?
Client:	Dan Bobrow
GUS:	I have confirmed the following flight:
Client:	P.S.A. flight 102 on Wednesday May 28 from San Jose to San Diego that leaves at 7.30 am and arrives at 9.15 am.
GUS:	What date do you want to return on ?
Client:	On Friday in the evening.
GUS:	Would you like the flight that leaves at 7.45 pm ?
Client:	That's fine.
GUS:	I have confirmed the following flight: P.S.A. flight 307 on Friday May 30 from San Diego to San Jose that leaves at 7.45 pm and arrives at 9.30 pm Thank you for calling. Goodbye

Figure 29.8 The travel domain: A transcript of an actual dialog with the GUS system of Bobrow et al. (1977). P.S.A. and Air California were airlines of that period.

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

GUS-style Frames

- Slots identify the information needed
- Fillers for slots define the semantic types that the slot values can take
- Types may have a hierarchical structure
  - example: date is also a frame

Slot	Type
ORIGIN CITY	city
DESTINATION CITY	city
DEPARTURE TIME	time
DEPARTURE DATE	date
ARRIVAL TIME	time
ARRIVAL DATE	date

```
DATE
        MONTH NAME
        DAY (BOUNDED-INTEGER  1  31)
        YEAR INTEGER
        WEEKDAY (MEMBER (SUNDAY MONDAY TUESDAY WEDNESDAY
                                    THURSDAY FRIDAY SATURDAY ) )
```

Control Structure for Frame-Based Dialogue

- Built around the frame
  - ask pre-built questions to elicit fillers from user responses
  - when all slots filled, take appropriate action (e.g., book flight, answer question)
  - usually implemented as simple finite state automata

- Example: Simple airline booking scenario

Slot	Question
ORIGIN CITY	"From what city are you leaving?"
DESTINATION CITY	"Where are you going?"
DEPARTURE TIME	"When would you like to leave?"
ARRIVAL TIME	"When do you want to arrive?"

Example: Airline Booking

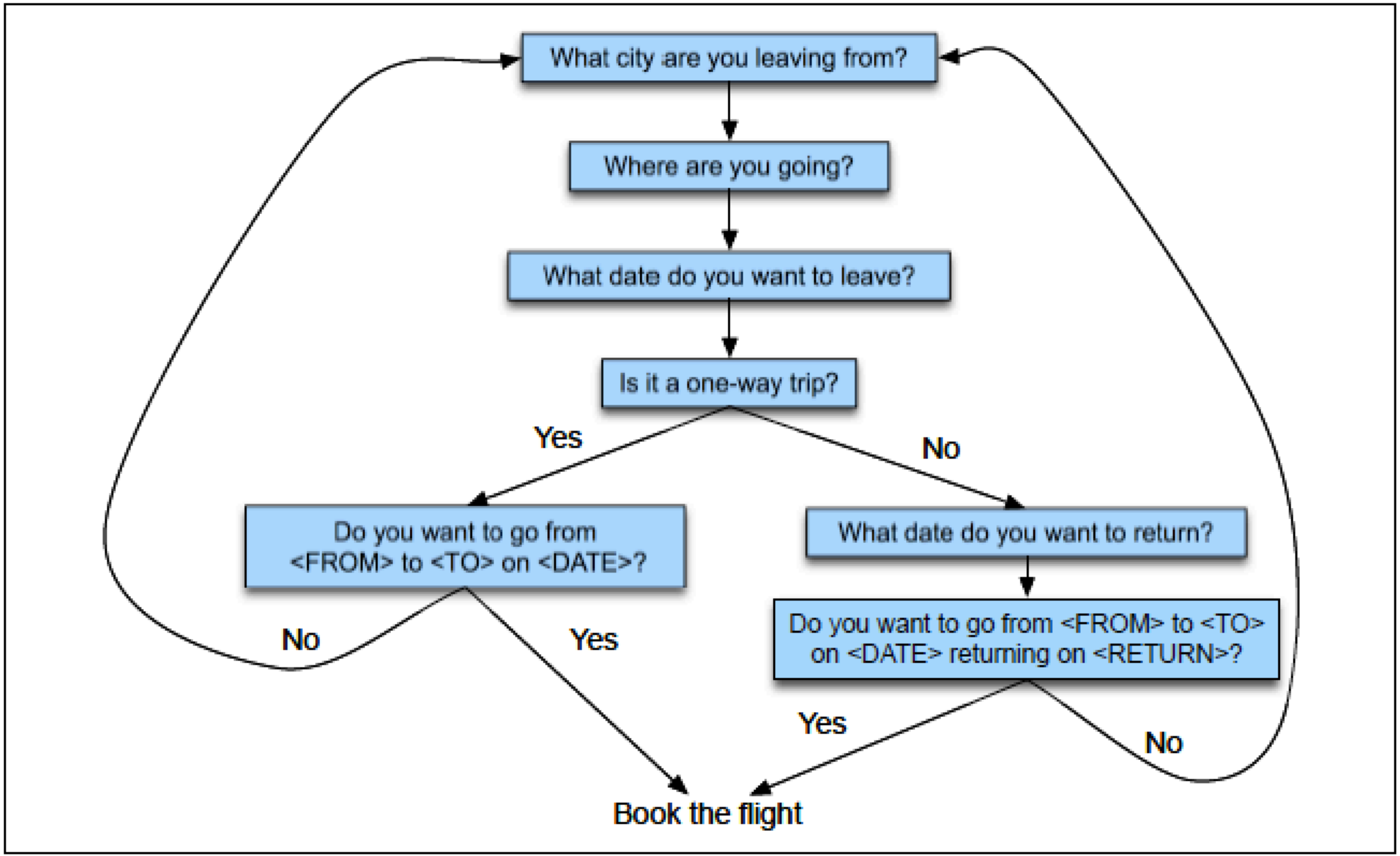

What city are you leaving from?
Where are you going?
What date do you want to leave?
Is it a one-way trip?
Yes
No
Do you want to go from <FROM> to <TO> on <DATE>?
What date do you want to return?
No
Yes
Do you want to go from <FROM> to <TO> on <DATE> returning on <RETURN>?
Yes
No
Book the flight

Figure 29.9 A simple finite-state automaton architecture for frame-based dialog.

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

Initiative

- Initiative
  - which speaker has control of the conversation
  - in previous example: *system-initiative*
  - in normal conversation: initative shifts back and forth

- single-initiative FSA architecture
  - system always knows which question the user is answering

- universal commands
  - can be said anywhere in the conversation
  - typically incorporated into FSA architecture
  - e.g., "help", "start over"

Towards Greater Flexibility

- A user's answer may fill more than one slot

  e.g., "I'd like a flight from Orlando to Denver one-way leaving after 5 pm Monday."

- Or involve more than one frame

  e.g., "I'd like to book a rental car when I arrive in Dallas."

- Current standard GUS architecture uses mixed-initiative
  - e.g., Siri, Alexa, Google Assistant
  - fills in all available slots from answers
  - skips questions for slots that have already been filled
  - also uses condition-action rules
    - e.g., once travel destination known, it is used as default location for hotel booking frame
  - uses production system concepts for switching between frames

Filling the Slots

- Filling slots is a natural language understanding task
  - domain classification:
    - for modern multi-domain systems
    - e.g., booking travel, setting an alarm, calendar management, etc.
  - intent determination:
    - what is user's goal or task?
    - e.g., Find-Movie, Book-Flight, Set-Alarm, Delete-Appointment
  - slot filling:
    - extracting the information to fill the slot(s)

Semantic Understanding

- Any semantic parsing approach can be used for slot filling
- Typically use hand-written rules

Example: Regular expression for for SET-ALARM intent

`wake me (up) | set (the|an) alarm | get me up`

- Can also use full CFG grammar parsers
- Issues
  - putting fillers in canonical form, e.g., dates
  - handling negation, e.g., "any time except Tuesday morning"
  - speech recognition errors: use information from all "n-best" transcriptions

Machine Learning Approaches

- for domain and intent:
  - use word n-grams (unigrams, bigrams, and trigrams)

- for slot-filling:
  - use additional features to find which slot to fill
  - e.g., named entities or class of entities (cities, airports, days of week)
  - one classifier to determine slot
  - another classifier to determine the filler for the slot

- can also use a sequence model to directly assign slot label to each word
  - e.g., MEMM, CRF, RNN

0	0	0	0	0	B-DES	0	B-DTIME
I	want	to	fly	to	Denver	on	Monday

Deep Learning Approach

- Explicit feature extraction not used
- Input sequence of words encoded as embedding or as "1-hot" vectors
- Output is a series of IOB tags plus a final state that concatenates domain & intent

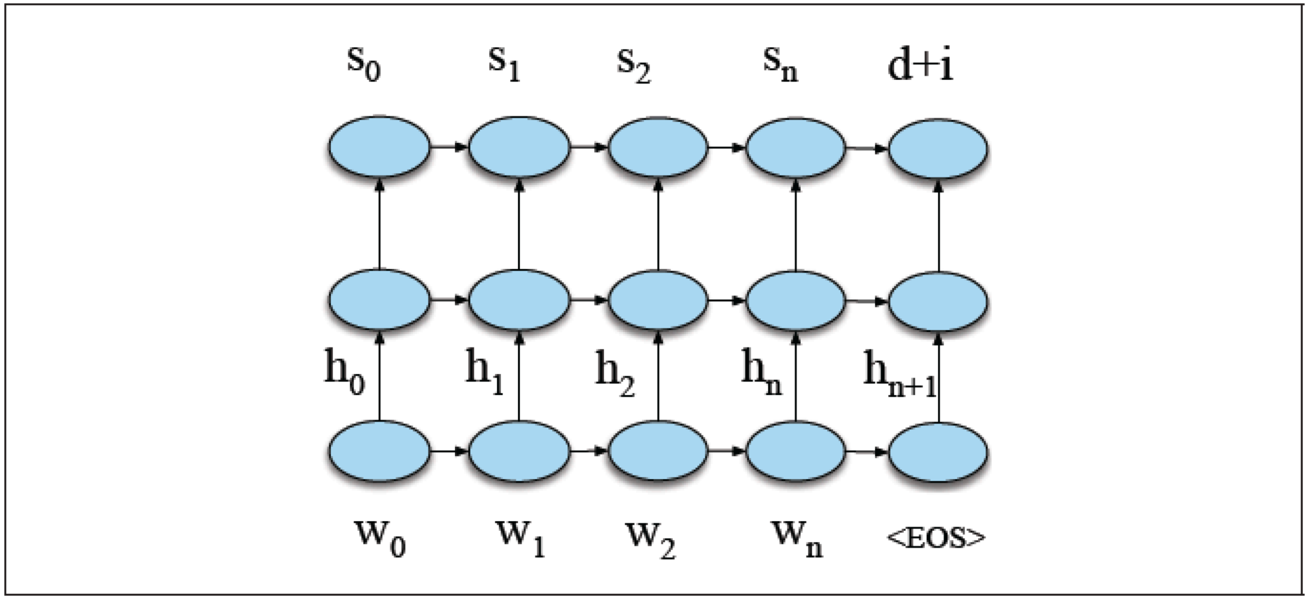


Figure 29.11 An LSTM architecture for slot filling, mapping the words in the input (represented as 1-hot vectors or as embeddings) to a series of IOB tags plus a final state consisting of a domain concatenated with an intent.

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

Bootstrapping

- A semi-supervised method
  - Used in industrial systems to create a machine-learning system that gets better with experience
- Start with
  - a rule-based system for the domain
  - manually labeled test set
- As new utterances come in
  - create training samples using the rule-based system
  - can use heuristics to eliminate degenerate cases
  - train a classifier on these samples
- With sufficient training samples, often outperforms the original rule-based system
  - but rule-based systems still generally higher precision

Evaluating Slot Filling

- intrinsic measure: slot error rate

$$slot\ error\ rate\ for\ sentence = \frac{\#inserted, deleted, or\ substituted\ slots}{total\ \#\ of\ reference\ slots\ for\ sentence}$$

Example: "Make an appointment with Mike at 10:30 in HEC-101"

Slot	Extracted filler
PERSON	Mike
TIME	11:30 am
ROOM	HEC-101

→ slot error rate = 1/3

Note: Can instead use slot precision, recall, and F-measure

- extrinsic measure: task error rate
  - e.g., how often the correct meeting is added to the calendar
  - less fine-grained

Additional Components

- ASR (automatic speech recognition)
  - language models for a conversational agent can depend on the dialog state
  - can train separate models for answers to specific questions
    - e.g., if system just asked the user for the departure city, an ASR model can be constrained to just model answers for that question
  - can similarly constrain FSA (RegEx) or CFG slot fillers

- language generation
  - produces what the system utters to the user
  - template-based generation
    - fixed template: "Hello, how can I help you?"
    - can include variables: "What time do you wish to depart CITY-ORIG?"
  - responses are passed to a separate TTS (text-to-speech) component

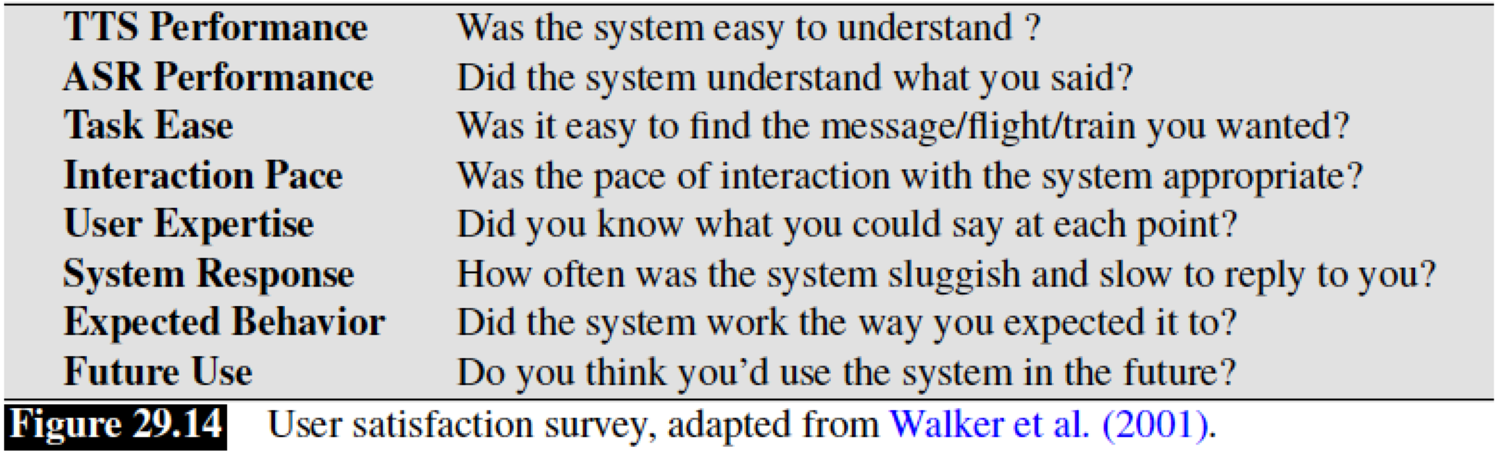
Today

- Conversational Agents
- Rule-Based Chatbots
- Corpus-Based Chatbots
- Frame-Based Dialog Agents
- Evaluating Dialogue Systems
- Dialogue System Design

Dialog System Evaluation

- absolute task success
    - for simple tasks
    - did the system book the right flight, or make the correct calendar entry

- user satisfaction rating



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

Figure 29.14 User satisfaction survey, adapted from Walker et al. (2001).

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

Heuristic Evaluaton

- Use evaluation function that maximizes task success while minimizing costs

- heuristics for task success
  - percentage of slots that were filled correctly
  - percentage of subtasks that were completed

- heuristics for efficiency cost
  - total elapsed time for dialog
  - number of total or system turns
  - total number of queries
  - number of system non-responses
  - "turn correction ratio" – percentage of turns used to correct errors

- heuristics for quality cost (user experience)
  - number of times user had to interrupt the system
  - number of times ASR failed or result rejected
  - number of times system response was too verbose, ambiguous, or incorrect
  - slot error rate

Today

- Conversational Agents
- Rule-Based Chatbots
- Corpus-Based Chatbots
- Frame-Based Dialog Agents
- Evaluating Dialogue Systems
- Dialogue System Design

System Design

- Many design choices
  - dialog strategies
  - syntax and wording for prompts
  - error messages
  - etc.

- User-centered design
  1. understand the potential users and the nature of the task
     - interview users, investigate similar systems
  2. build simulations and prototypes
     - use a Wizard-of-Oz system (human-in-loop, hidden from user) to prototype an architecture before implementation
  3. iteratively test the design on users