



CS 224S / LINGUIST 285

Spoken Language Processing

Andrew Maas

Stanford University

Spring 2022

**Lecture 4: Dialog system design. GUS
and frame-based dialog systems.
Alexa Skills Kit.**

Original slides by Dan Jurafsky

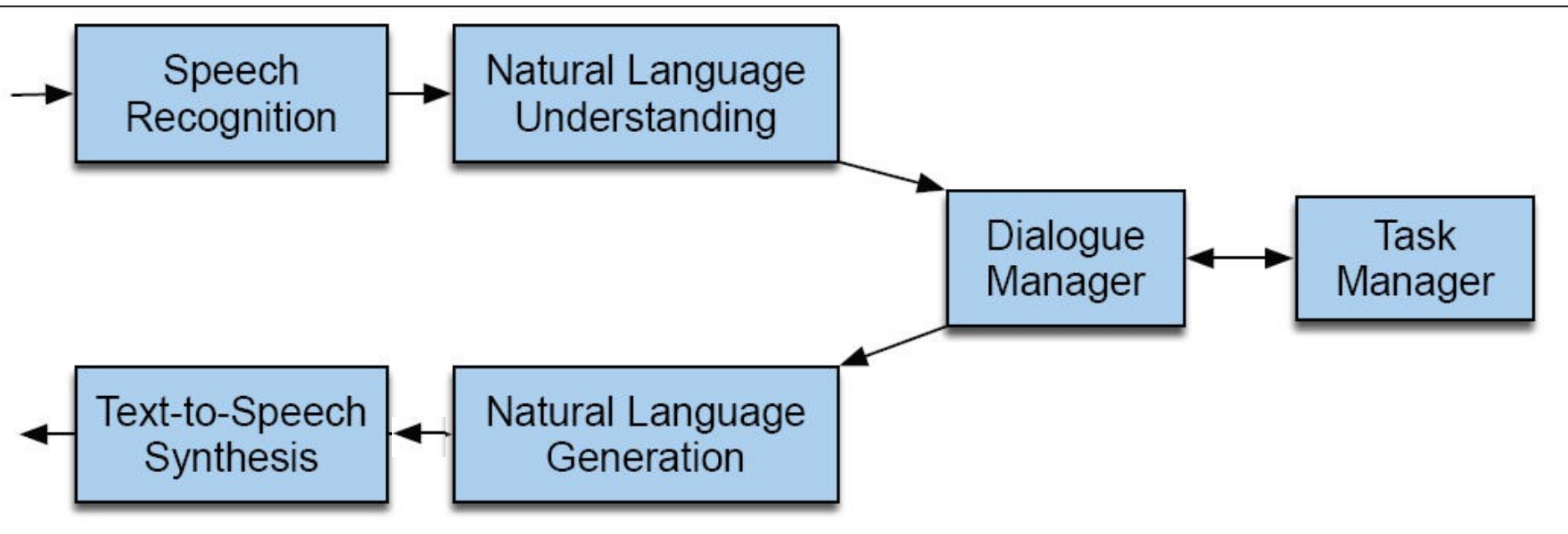
Outline

- Dialog system design
- GUS and frame-based systems
- NLU and NLG considerations
- Alexa skills kit overview

Homework 1 due Monday 11:59pm

Extra Office Hours Monday

Spoken Dialog Agent Conceptual Architecture



Dialog System Design: User-centered Design

Gould and Lewis 1985

1. Study the user and task
2. Build simulations
"Wizard of Oz study"
3. Iteratively test the design on users
4. Build a system to meet most valuable (and feasible) needs



System design considerations

- Goal and scope of overall system?
 - What tasks/actions are supported?
 - What state do the dialog/task managers track?
- What level of interaction complexity?
 - Initiative? Back-tracking? NLU support for paraphrasing?
- Need a solution for each module (ASR, TTS, NLU, NLG, task/dialog manager)
- What is the interface / data structure between modules?
 - e.g. Does ASR module send transcripts only? Emotion labels? Audio?

Case study: GoButler

- Text chat interface
- *Human* operator could complete any task!
 - Canceling cable subscriptions, booking restaurants etc.
- Wave of startups 2013-2017
 - Magic, Operator, Facebook M
- Idea: Gather enough data to automate most tasks
- With enough data, NLP + connected services would allow positive unit economics



Case study: GoButler

Current product (2022)

What happened to
do anything for me?



GoButler,
your assistant who knows
everyone

GoButler can tell you the contact information for nearly anyone. Phone numbers, email addresses, social media profiles and more.

[Phone](#) [Email](#) [Domain](#)

[Search](#)

no credit card required

Your Personal Data Assistant

GoButler crawls the internet and public records and remembers all the contact information he comes across.

He does this so he can be the best possible assistant for you. Whoever you need to find or get a hold of, GoButler is ready to serve you.

Secure Free People Search

GoButler provides an incredible amount of information for free, and aims to be dead simple to work with.

If you want additional information, GoButler can refer you to others he trusts to get you what you need.

Case study: GoButler

- The “personal assistants to do anything” startup hype wave settled down 2017+
- Collecting data isn’t enough to cover *all* possible tasks
 - Acting on the task often complex. Requires a human.
 - NLP/Dialog aspects work fairly well for simpler requests
- What came out of these experiments?
 - Narrow-domain chat assistants for valuable services
 - Tools for quickly designing and building dialog assistants
 - Lots of VC money spent on users getting free personal assistants for a while 😬

Rough system design process:

Design phase

1. Overall system goal.
2. Define set of task actions system can perform
3. Create example interactions

Rough system design process:

Technology choices after design

1. Overall system goal.
2. Define set of task actions system can perform
3. Create example interactions
4. Define dialog manager approach (actions + dialog acts/state of system)
5. Choose NLU approach matching complexity of tasks and approach to initiative + dialog acts
6. Define NLG approach and dialog state -> NLG interface
7. Create a dialog policy (choosing next dialog action and sending to NLG)
8. Choose ASR/TTS approach. Update NLU/NLG if needed

System design considerations

- Not all systems require support for complex interactions
- Frameworks like Alexa force some choices about multiple modules to simplify overall development
- ASR/TTS components often be treated as black-box, but great systems are sensitive to ASR uncertainty
- Okay to redefine/combine modules based on problem (e.g. a smart NLG module might simplify dialog manager)

Outline

- Dialog system design
- **GUS and frame-based systems**
- NLU and NLG considerations
- Alexa skills kit overview

Frame-based dialog agents

- One or more **frames**
 - Each a collection of **slots**
 - Each slot having a **value**
- **Domain ontology** is full collection of frames for a (sub)system
- A frame defines the task data structure

The Frame

- A set of **slots**, to be filled with information of a given **slot type**
- Each associated with a **question** to the user

Slot	Type	Question
ORIGIN	city	What city are you leaving from?
DEST	city	Where are you going?
DEP DATE	date	What day would you like to leave?
DEP TIME	time	What time would you like to leave?
AIRLINE	line	What is your preferred airline?

Frame-based dialogue agents

1977:

GUS, A Frame-Driven Dialog System¹

**Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay,
Donald A. Norman, Henry Thompson and
Terry Winograd**

*Xerox Palo Alto Research Center, 3333 Coyote Hill Road,
Palo Alto, CA 94304, U.S.A.*

Artificial Intelligence Journal, 1977

- Still the industrial state of the art
- Alexa roughly follows GUS architecture

GUS: Hello. My name is GUS. I can help you plan a simple trip by air. Where do you want to go ?

**The state of the
art in 1977 !!!!**

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: 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

Slot types can be complex, hierarchical

- The type *DATE*

DATE

MONTH NAME

DAY (BOUNDED-INTEGER 1 31)

YEAR INTEGER

WEEKDAY (MEMBER (SUNDAY MONDAY TUESDAY WEDNESDAY THURSDAY FRIDAY SATURDAY))]

Simple action policy for frame-based dialog

Consider a trivial airline travel system:

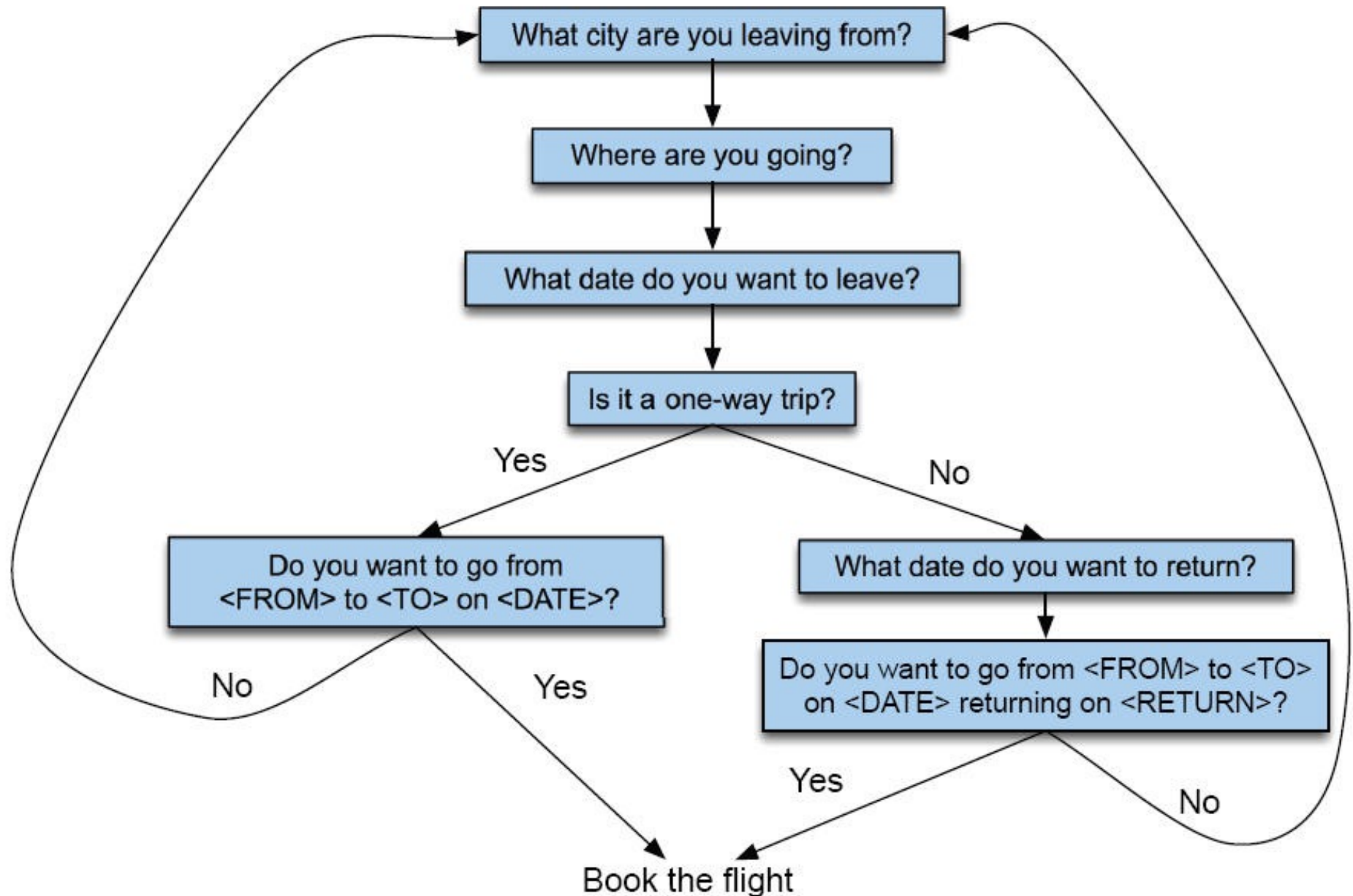
- Ask the user for a departure city

- Ask for a destination city

- Ask for a time

- Ask whether the trip is round-trip or not

Finite State Dialog Manager



Finite-state dialog managers

- System completely controls the conversation with the user.
- It asks the user a series of questions
- Ignoring (or misinterpreting) anything the user says that is not a direct answer to the system's questions

Frames and mixed initiative

- System asks questions of user, filling any slots that user specifies
 - When frame is filled, do database query
- If user answers 3 questions at once, system can fill 3 slots and not ask these questions again!
- Frame structure guides dialog

Mixed Initiative

- Conversational initiative can shift between system and user
- Simplest kind of mixed initiative: use the structure of the **frame** to guide dialogue

Slot

Question

ORIGIN

What city are you leaving from?

DEST

Where are you going?

DEPT DATE

What day would you like to leave?

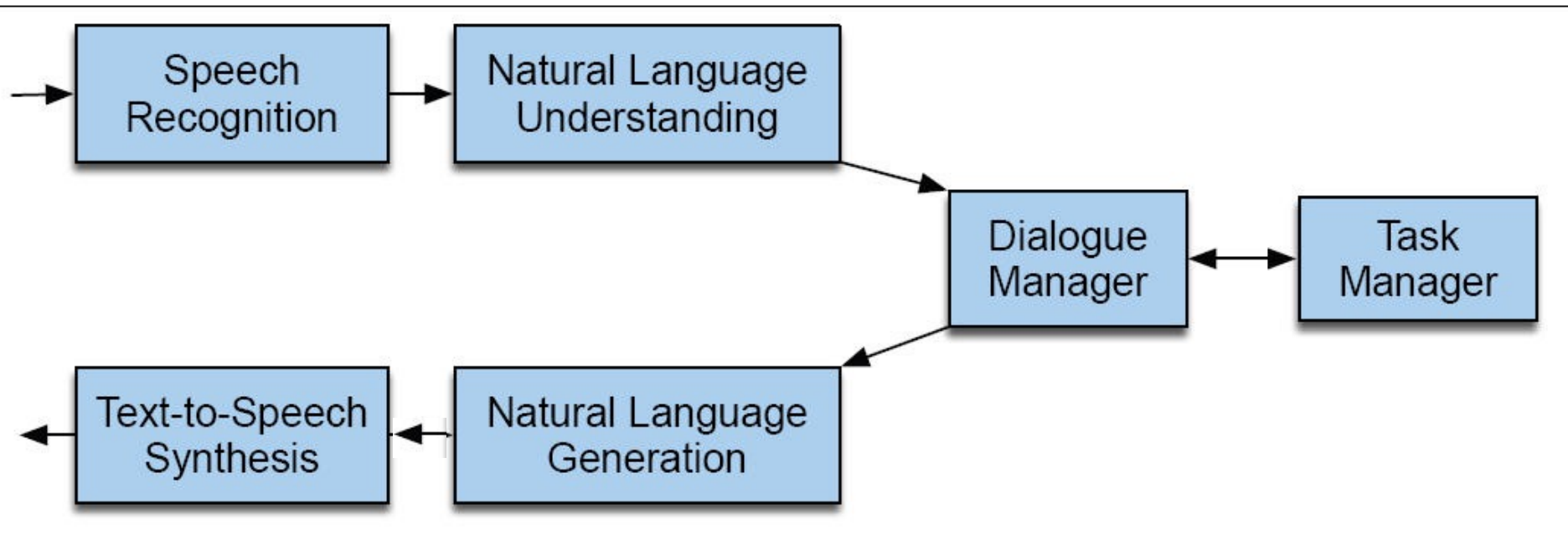
DEPT TIME

What time would you like to leave?

AIRLINE

What is your preferred airline?

NLU and NLG with frame-based systems



Natural Language Understanding for filling dialog slots

1. Domain classification

Asking weather? Booking a flight?
Programming alarm clock?

2. Intent Determination

Find a Movie, Show Flight, Remove
Calendar Appt

3. Slot Filling

Extract the actual slots and fillers

Natural Language Understanding for filling slots

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

Natural Language Understanding for filling slots

Wake me tomorrow at six.

DOMAIN: ALARM-CLOCK

INTENT: SET-ALARM

TIME: 2017-07-01 0600-0800

Rule-based Slot-filling

Write regular expressions or grammar rules

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

Do text normalization

Time consuming and brittle NLU capabilities

With modern NLP tools/features, only use rules alone in special cases

Machine learning for slot-filling

I want to fly to San Francisco on
Monday afternoon please

Use 1-of-N classifier for Domain/Intent. Use sequence
model to tag words/phrases with slot names

- Input:

features like word N-grams

- Output:

Domain: AIRLINE

Intent: SHOWFLIGHT

Destination-City: "San Francisco"

Depart-Date: "Monday"

More sophisticated algorithm for slot filling: IOB Tagging

- IOB Tagging
 - tag for the beginning (B) and inside (I) of each slot label,
 - plus one for tokens outside (O) any slot label.
 - $2n + 1$ tags, where n is the number of slots.

B-DESTINATION
I-DESTINATION
B-DEPART_TIME
I-DEPART_TIME
O

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

Slot-filling is information/entity extraction from text NLP

- Conditional Random Field (CRF) with word vector features, or neural classifiers both work well

Back in 2000 , **People Magazine** **PUBLISHER** highlighted **Prince Williams'** **PERSON** style who at the time was a little more fashion-conscious , even making fashion statements at times .

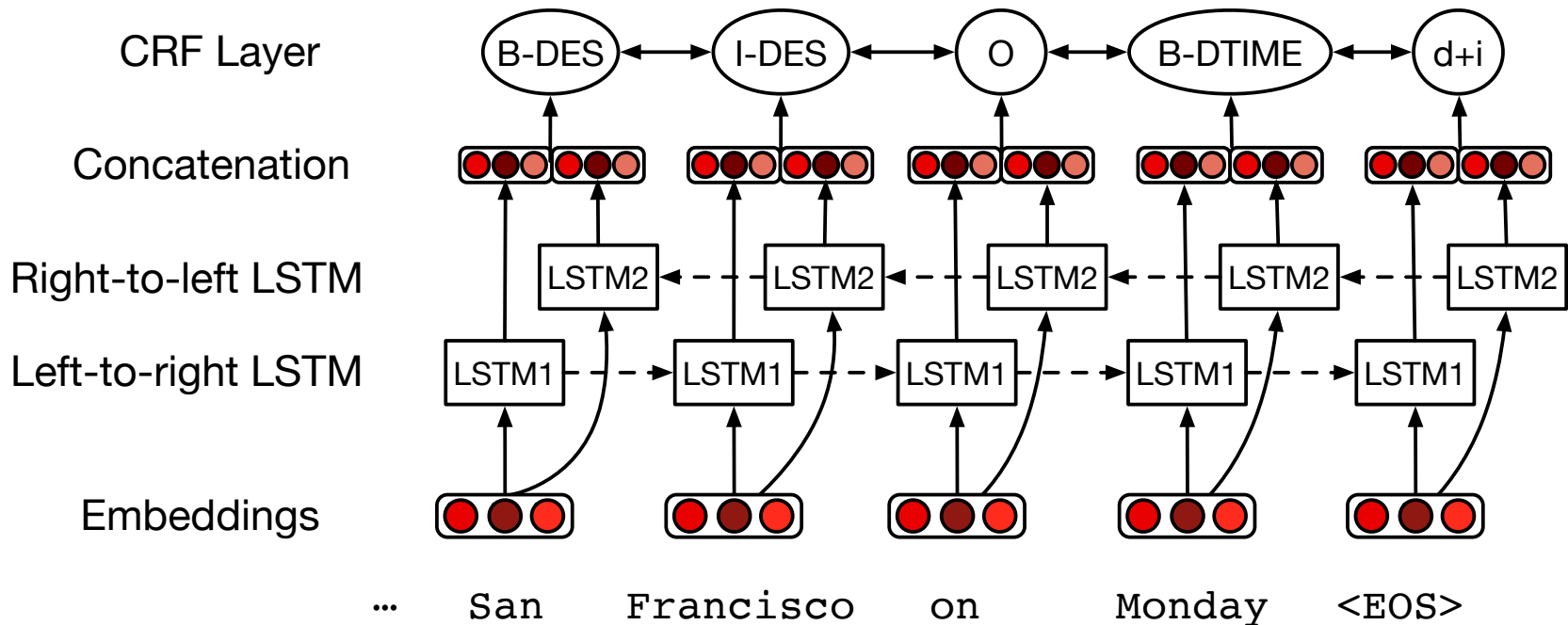
Now-a-days the prince mainly wears **navy** **COLOR** **suits** **ITEM** (sometimes **double-breasted** **DESIGN**) , **light blue** **COLOR** **button-ups** **ITEM** with **classic** **LOOK** **pointed** **DESIGN** **collars** **PART** , and **burgundy** **COLOR** **ties** **ITEM** .

But who knows what the future holds ...

Duchess Kate **PERSON** did wear an **Alexander McQueen** **BRAND** **dress** **ITEM** to the **wedding** **OCCASION** in the **fall of 2017** **SEASON** .

Sequence models for slot filling: IOB Tagging

- IOB Tagging is done by a sequence model
- Typical:



- Extracted strings can then be normalized (San Fran->SFO)

Generation Component (NLG)

- **Content Planner**
 - Decides what content to express to user
(ask a question, present an answer, etc)
 - Often merged with dialogue manager
- **Language Generation**
 - Chooses syntax and words
 - TTS
- **In practice:** Template-based w/most words prespecified

What time do you want to leave CITY-ORIG?

Will you return to CITY-ORIG from CITY-DEST?

More sophisticated NLG

- Dialogue manager builds representation of meaning of utterance to be expressed
- Passes this to a “generator”
- Mixing chatbot-like NLG constrained to convey dialog representation can improve user satisfaction

NLG in Dialog

- Great area to leverage pre-trained large language models / text generation
- Critical aspect: Ensure correctness of what we convey to the user!

NLG in Dialog

- Great area to leverage pre-trained large language models / text generation
- Critical aspect: Ensure correctness of what we convey to the user!

Deep learning NLG conditioned on dialog semantics

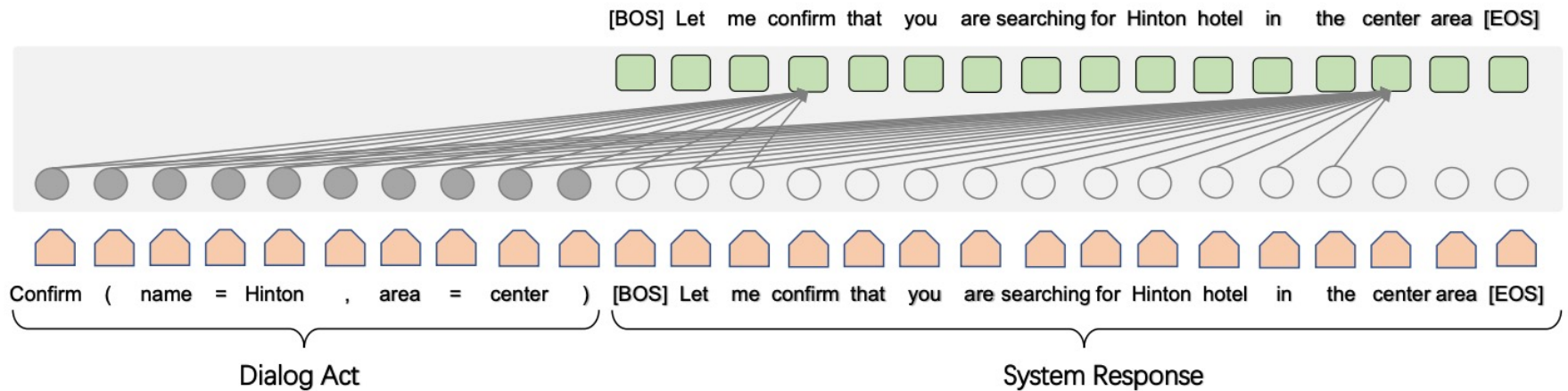


Figure 2: Illustration of SC-GPT. In this example, SC-GPT generates a new word token (*e.g.*, “confirm” or “center”) by attending the entire dialog act and word tokens on the left within the response.

Semantically conditioned GPT for dialog NLG

$$\mathcal{A} = [\underbrace{\mathbf{I}}_{\text{Intent}}, \underbrace{(s_1, v_1), \dots, (s_P, v_P)}_{\text{Slot-value pairs}}]$$

$$p_{\theta}(\mathbf{x}|\mathcal{A}) = \prod_{t=1}^T p_{\theta}(x_t|x_{<t}, \mathcal{A})$$

Dialog Act

Intent: *Confirm*

Slot-value pairs:

```
[ name = Hilton ], [ area = center ]
```

Response

Let me confirm that
you are searching for Hinton hotel
in the center area

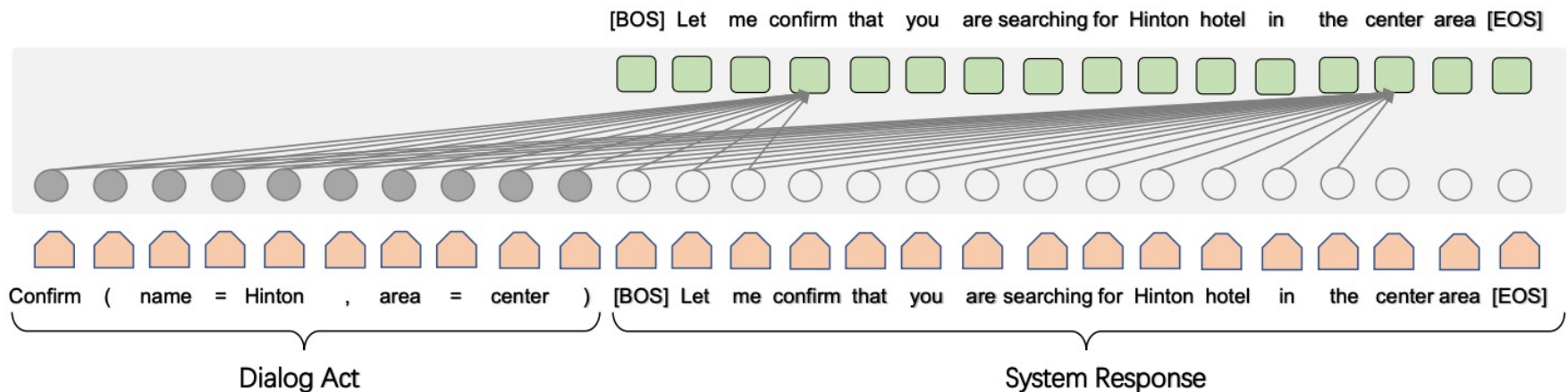


Figure 2: Illustration of SC-GPT. In this example, SC-GPT generates a new word token (*e.g.*, “confirm” or “center”) by attending the entire dialog act and word tokens on the left within the response.

Deep learning NLG conditioned on dialog semantics

3	Input DA	Train{inform(time=50 minutes)}, hotel{request(stars=?; area=?), inform(choice=5)}
	Reference	travel time is 50 minutes . i found 5 hotels you may like . do you have a star rating request or an area you prefer
	SC-LSTM	there are 5 trains that are 50 minutes . do you have a preferred departure location ? % [train{inform(choice=5),request(departure=?)}] [hotel{request(stars=?; area=?), inform(choice=5)}]
	HDSA	there are 5 trains that meet your criteria . do you have a preference on the area or star rating ? % [train{inform(choice=5, time=50 minutes)}] [hotel{inform(choice=5)}]
	SC-GPT	there are 5 hotels that meet your criteria . the trip will last 50 minutes . do you have an area preference or star rating you would like ?

Table 8: Examples of generated utterances from different models, along with its corresponding dialog acts (DAs) and references. The first two examples are sampled from FEWSHOTWOZ and the last one is from MultiWOZ. Each generated utterance is followed by a brief description explaining the errors (starting with “%”). (Better viewed in color. wrong , redundant , missing information)

Deep learning NLG conditioned on dialog semantics

1	Input DA	Laptop{inform(name=satellite proteus 84; type=laptop; memory=8 gb; drive=1 tb; weight=2.3 kg)}
	Reference	the satellite proteus 84 is a laptop with a 1 tb drive , 8 gb memory and weighs in at 2.3 kg
	SC-LSTM	the satellite proteus 84 is a laptop with 8 gb , with a 1 tb drive , and is for business computing , and is for business computing % [businesscomputing=true] [weight=2.3kg]
	GPT-2	the satellite proteus 84 laptop has a 1 tb drive , weighs 2.3 kg and is used for business computing % [businesscomputing=true] [memory=8 gb]
	SC-GPT	the satellite proteus 84 is a laptop with 8 gb memory , 1 tb drive , and a weight of 2.3 kg
2	Input DA	Restaurant{inform_only_match(name=marnee thai and thanh long; pricerange=moderate; near=outer sunset; goodformeal=dinner)}
	Reference	marnee thai and thanh long are the only 2 moderately priced restaurants near outer sunset that are good for dinner
	SC-LSTM	marnee thai and thanh long is is moderate restaurant restaurant and good good good dinner % [near=outer sunset] [inform_only_match]
	GPT-2	there is a moderately priced restaurant called marnee thai and thanh long that is near the outer sunset area % [goodformeal=dinner] [inform_only_match]
	SC-GPT	marnee thai and thanh long is the only restaurant that serves moderately priced food near outer sunset and good for dinner

Outline

- Dialog system design
- GUS and frame-based systems
- NLU and NLG considerations
- **Alexa skills kit overview**

Alexa Skills Kit

- A *Skill* is a top level command for Alexa.
 - “Alexa open 224S Homework 2”
 - Skill → **domain ontology**
- A skill contains *intents* which are distinct task actions.
 - Intent → **frame**
 - Design intents with built-in capabilities per intent and ASK interaction model in mind
- Each intent contains *slots* which each have a *slot type* and take on a *slot value*
- Not quite this simple (e.g. ASK built-in intents are not simple to define in the frame/slot abstraction)

Alexa Skills Kit

- Dialog management is complex, partially handled with built-in features (clarification, value verification, cancel skill, etc)
- NLU through grammars and examples.
 - ASK trains models for you based on examples
 - Many rich slot types (dates, numbers, lists)
- Task management is custom! ASK provides a dialogue API to your web server, you implement server-side task execution.
- NLG is template-based with ASK adding variety
- ASR/TTS handled by ASK. Interface is text/transcripts
- Overall framework is API/SDK oriented like web dev

Alexa domain classification

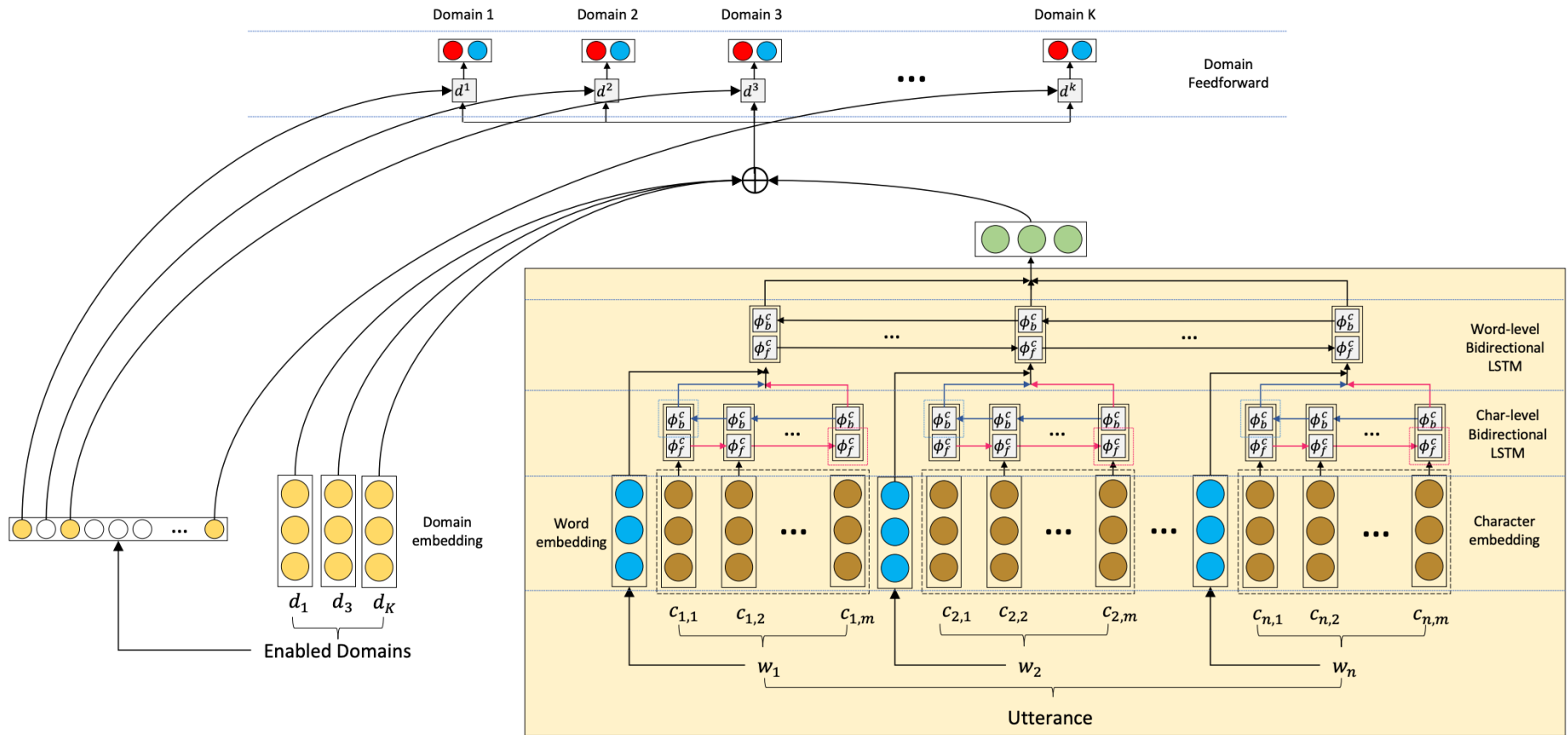


Figure 1: The overall architecture of the personalized dynamic domain classifier.

ASK interaction schema

Interaction Model

Field	Type	Description	Required?
languageModel	object	Conversational primitives for the skill	yes
dialog	object	Rules for conducting a multi-turn dialog with the user	no
prompts	array	Cues to the user on behalf of the skill for eliciting data or providing feedback	no

languageModel [↗](#)

Field	Type	Description	Required?
invocationName	string	Invocation name of the skill	yes
intents	array	Intents and their slots	yes
types	array	Custom slot types	no
modelConfiguration	object	Optional settings for the interaction model. Available in supported locales .	no

languageModel_intents

Field	Type	Description	Required?
name	string	Name of the intent. For details about intent names, see Intent and slot name requirements .	yes
slots	array	List of slots within the intent.	no
samples	array	Sample utterances for the intent	no

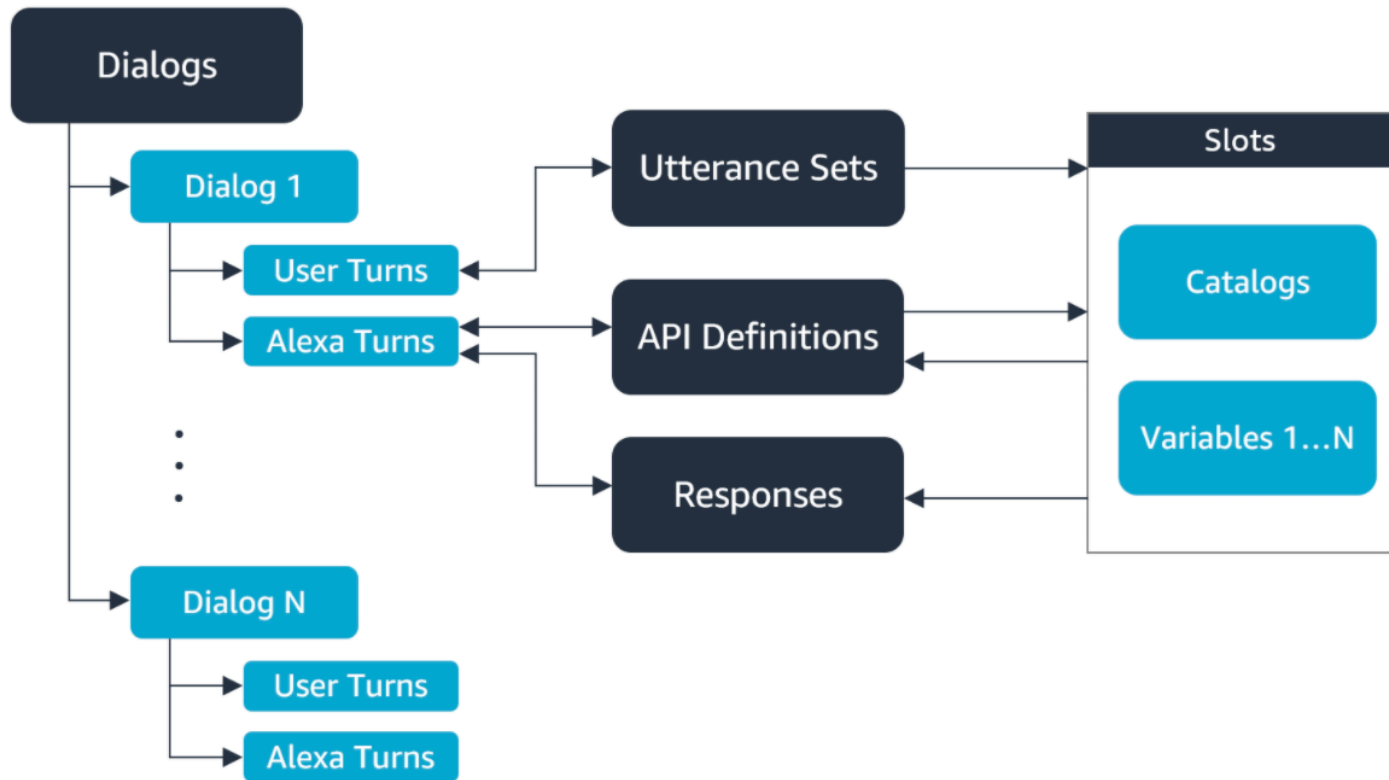
ASK Intent JSON example

This example shows a portion of the intent object for a PlanMyTrip intent. The utterances for the intent are in `interactionModel.languageModel.intents[].samples`. Each slot has its own samples array. For brevity, other properties within `interactionModel` and `languageModel` are not shown

```
{ "interactionModel": { "languageModel": { "intents":  
  [ { "name": "PlanMyTrip", "slots": [ { "name": "travelDate", "type": "AMAZON.DATE",  
    "samples": [ "I am taking this trip on {travelDate}", "on {travelDate}", "{travelDate}" ] },  
    { "name": "toCity", "type": "AMAZON.US_CITY", "samples": [ "I'm going to {toCity}", "{toCity}" ] },  
    { "name": "fromCity", "type": "AMAZON.US_CITY", "samples": [ "{fromCity}", "I'm starting from  
    {fromCity}" ] },  
    { "name": "travelMode", "type": "LIST_OF_TRAVEL_MODES", "samples": [ "I am going to {travelMode}",  
    "{travelMode}" ] },  
    { "name": "activity", "type": "LIST_OF_ACTIVITIES", "samples": [ "{activity}", "I plan to {activity}" ] } ],  
    "samples": [ "{toCity}", "I want to travel from {fromCity} to {toCity} {travelDate}", "i want to visit  
    {toCity}", "i am going on trip on {travelDate}", "I'm {travelMode} from {fromCity} to {toCity}", "i'm  
    {travelMode} to {toCity} to {activity}", "plan a trip", "plan a trip to {toCity}", "plan a trip starting from  
    {fromCity}", "I'd like to leave on {travelDate}", "I'd like to leave on the {travelDate}", "I'd like to fly out of  
    {fromCity}" ] ] ] }
```

Alexa Conversations (new in 2020)

When you build an Alexa Conversations skill, you create the following components that train Alexa Conversations how to interact with your user.



[\(Announcement, ASK docs\)](#)

English (US)

Save Model

Version

Build Model

Update live skill

Evaluate Model



CUSTOM

Invocation

Interaction Model

Intents (6)

ice_cream

Built-In Intents (5)

AMAZON.FallbackIntent

AMAZON.CancelIntent

AMAZON.HelpIntent

AMAZON.StopIntent

AMAZON.NavigateHomeIntent

Annotation Sets New

Intent History

Utterance Conflicts (0)

JSON Editor

Assets

Slot Types (4) Feedback X



Updates to sample utterances qualify for instant live updates. [Learn more](#) about live updates to your skill.

Intents / ice_cream

Sample Utterances (15) ?

Bulk Edit

Export

What might a user say to invoke this intent?



i would like {num_scoops} scoops of {flavor} in a {container} with {toppings_one} and {toppings_two}



I would like {num_scoops} scoops of {flavor} in a {container} with {toppings_one} please



a {container} with {num_scoops} scoops of {flavor}



one ice cream please



a {container} with {flavor} {num_scoops} scoops



< 1 - 5 of 15 > [Show All](#)

Dialog Delegation Strategy ?