CS 224S / Linguist 285
Spoken Language Processing

Andrew Maas | Stanford University | Spring 2024

Lecture 18: Dialog system design. Alexa Skills Kit in the era of LLMs

Poster session on Wednesday!

- **Today is our final lecture, next time is poster session. Wednesday June 5**
  - Set up poster by 12:30pm
  - Present poster with group 12:30 - 2:00pm
  - Please try to have everyone present the full time
    - Worst case, >=1 group member at least
  - Space available for laptop/phone demos of audio samples. Please help pack up afterwards!
- **Huang Engineering Building, Mackenzie Room**
- **Be ready with a ~2 minute verbal summary of your poster (+ demo!)**
  - Teaching staff will circulate. Make sure to speak with at least one of us
- **Final project reports are due Saturday by 11:59pm. No late days.**
  - Okay to include results in final report that didn't make it in time for poster

Project Milestone Feedback

- **Overall great progress on ambitious projects!**
- **Median progress**
  - Dataset selected + working with it
  - Baseline methods implemented and tested
  - Many debugging complex baselines / off the shelf models
- **Some people are getting lost in complex, multi-module systems**
  - Get *something* working end-to-end. Then improve on it
  - You do not need to fine tune an ASR model just for this class. Identify where you can improve and do it
- **Comments in gradescope**
  - *Even if you earned full points on milestone, your final report must have:*
    - *Additional experiment results*
    - *More complete writeup. Especially related work + methods description*
    - *Address any feedback from us about insufficient results/experiments, examples from your system*

Project Milestone Feedback

- **Get a complete (data prep, train, evaluate) experiment pipeline working with simple models first.**
  - Clearly articulate your research hypothesis and how your chosen dataset, model, metrics, fit in
  - For demo systems. What should the demo achieve? For who? Compared to what baseline for achieving that task
- **When deciding what to try next:**
  - Form a hypothesis about what is broken and how to improve (include in your final paper)
  - Don't lose track of your high level project goal!
- **See course page for final report expectations:**
  - Present your project motivation, what you tried, and progress you made in context of overall project goal
  - It's okay if various deep learning approaches don't work as well as you hope
  - Make sure you describe why you tried what you tried, and control/debugging experiments as needed
- **100% on final report == we evaluate your project + paper as on par with published short paper**

Project Milestone Feedback: Related work citations

- Related work / citing previous work is NOT a “book report” where you summarize each
- Instead, cite papers relevant to the points you are making to ground your proposed approach, claims about novelty, and results analysis in the context of previous work
  - Related approaches on the same dataset, same task on different datasets, similar demo system
  - Cite papers that introduced methods/models you use, and works that applied similar models to similar data
- What should it look like? An ACL short paper! 4 pages published in top tier conference

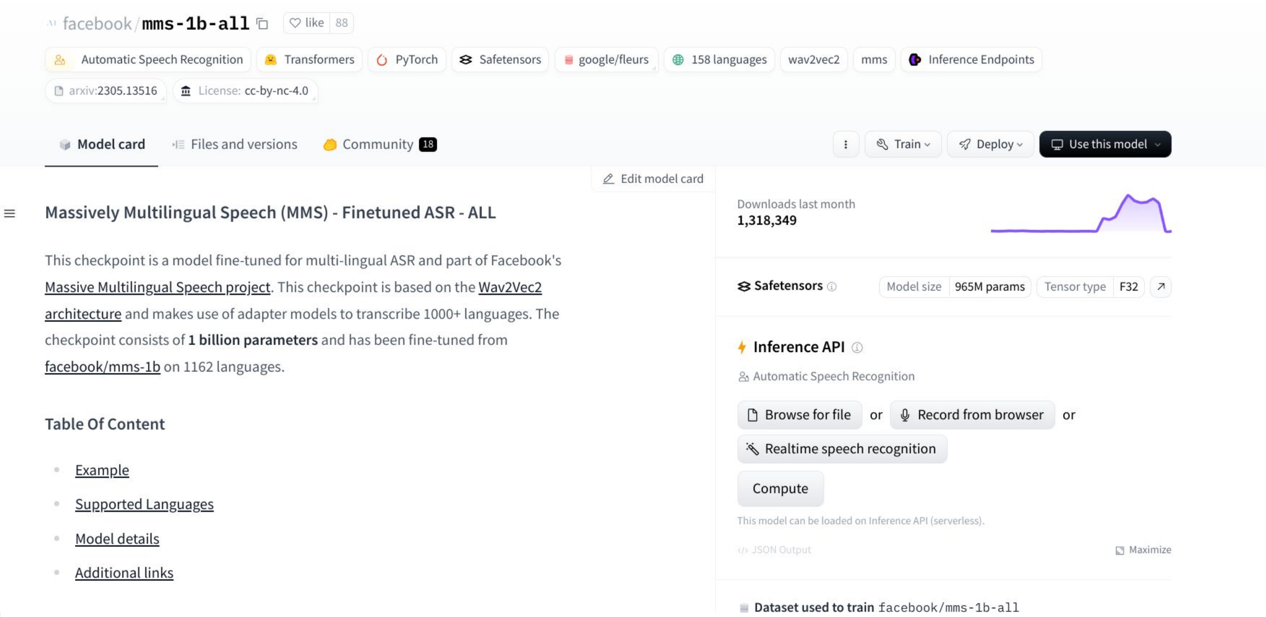
systems. Moreover, the logical form should be suitable to support feasible reasoning, for which also theorem provers, model builders, and model checkers can be used. Several semantic representations have been proposed that take these aspects into account, such as for example Quasi Logical Forms [1] and Dynamic Predicate Logic [6]. For the approach presented here, a simplified first order logic is used similar to quasi logical forms. The dialogue data that is used for semantic interpretation consists of recorded interactions with a help desk on how to operate a fax device. Examples of resulting utterances and their corresponding semantic content, expressed by λ-expressions of first-order logic, are illustrated in the following table:

The approach reported here has several aspects in common with that of Bos [2], who uses a CCG based parser [5] and assigns Discourse Representation Structures (DRSs) to the lexical categories used by the parser

Examples from (Geertzen, 2009)

Homework 3

How did I get the baseline improved WERs?



How did I get the baseline improved WERs?

Other methods people tried

Used large language models

GPT - 4

Trained a kNN classifier

Raghav Mittal Garg

- Train a KNN model to do language prediction using wav2vec feature extraction to determine which language the utterance is from.
- We then route the utterance to a finetuned model, if it existed.

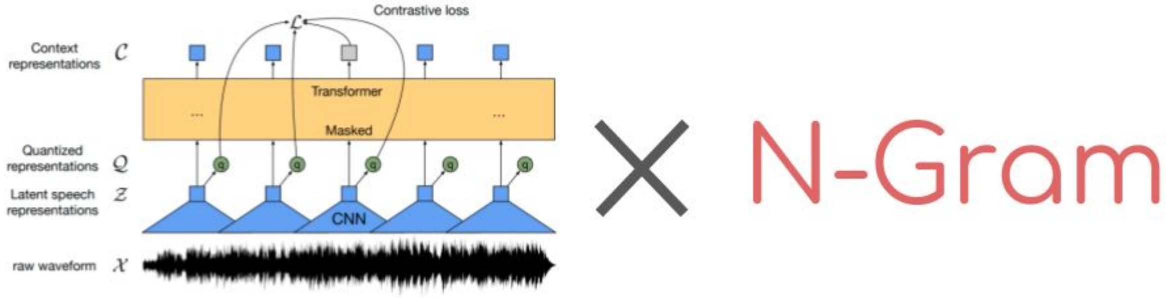
Confusion Matrix

	Amharic	siSwati	isiXhosa	Yoloxóchitl Mixtec
Amharic	77	0	0	0
siSwati	2	90	0	0
isiXhosa	0	1	108	30
Yoloxóchitl Mixtec	0	0	24	91

Classification Report

	Precision	Recall	F1-Score	Support
Amharic	0.97	1.00	0.99	77
siSwati	0.99	0.98	0.98	92
isiXhosa	0.82	0.78	0.80	139
Yoloxóchitl Mixtec	0.75	0.79	0.77	115

Used an n-gram language model



Finetuned with similar languages

3.1 Languages #389

Anonymous
2 weeks ago in **Homework**

PIN STAR WATCH 89 VIEWS

Are we allowed to use training data from similar languages for 3.1 or are we only allowed to use the 1h subsets in the HF dataset you provided?

Comment Edit Delete Endorse •••

Learnt about the languages they were using

Pete Warden 2w

Thanks Tolulope! I traced the dataset back to the original paper, and section 2.3 describes the orthography used in the transcription.

> (a) Transcription Level: The YMC-EXP corpus presently has two levels of transcription: (1) a practical orthography that represents underlying forms; (2) surface forms. The underlying form marks prefixes (separated from the stem by a hyphen), enclitics (separated by an = sign), and tone elision (with the elided tones in parentheses). All these "breaks" and phonological processes disappear in the surface form.

> The practical, underlying orthography mentioned above was chosen as the default system for ASR training for three reasons: (1) it is easier than a surface representation for native speakers to write; (2) it represents morphological boundaries and thus serves to teach native speakers the morphology of their language; and (3) for a researcher interested in generating concordances for a corpus-based lexicographic project it is much easier to discover the root for 'house' in be03 e 3=an4 and be03 e (3)= 2 than in the surface forms be03a˜ 4 and be03 e 2

This helps me understand what's going on in this case.

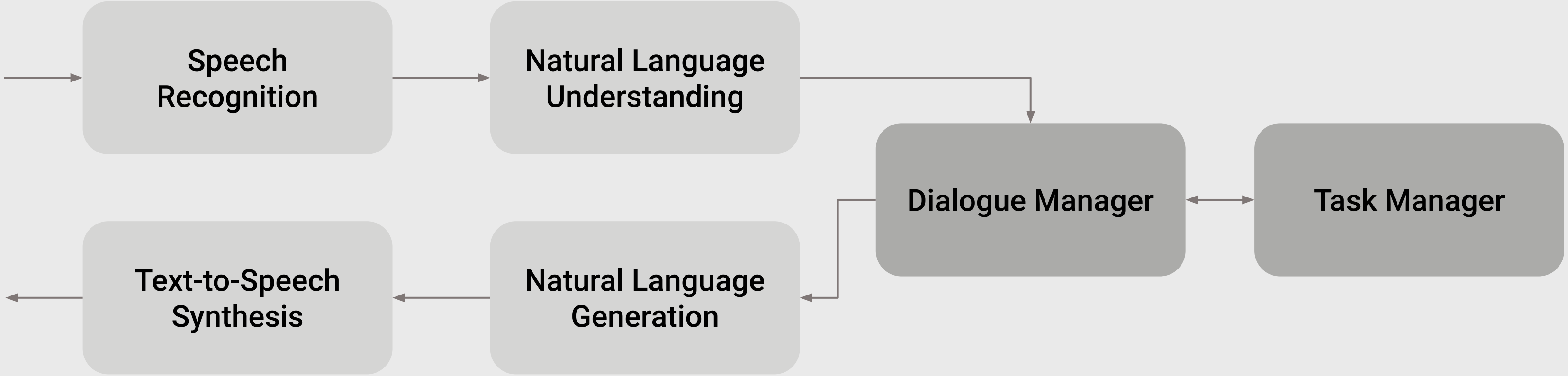
Homework 3 - Hopefully it was fun!

Outline

- Dialog System Design
- Case studies on dialog paradigms
  - Asynchronous text-based personal assistants (e.g. GoButler)
  - Alexa Skills Kit

Dialog System Design

Spoken Dialog Agent Conceptual Architecture



Dialog System Design: User-centered Design

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

Figure: Gould and Lewis (1985)

User-focused 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?
- **What is the interface / data structure between modules?**
  - e.g. Does ASR module send transcripts only? Emotion labels? Audio?
  - How does the system connect with external actions / APIs?
- **Focus on design questions and desired interactions to build requirements**
  - Don't constrain yourself by what is easy to implement at first
  - Gather requirements before spending much time planning implementation solutions (what to use for each module (ASR, TTS, NLU, NLG, task/dialog manger)

System Design Process: Design Phase

1

Define overall system goal

2

Define set of task actions system can perform

3

Create example interactions

System Design Process: Technology Choices

1. Define 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 (sometimes voice commands work fine)
- 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)

- Big modern questions in design:
  - Focus on the user experience. What dialog interactions do you need to support? Synchronous? Etc.
  - Once UX defines requirements. What is the task/action space? (what actions the system can take in the world)
  - Design the conversational system to bridge between the UX and task outputs you want to support

Emerging state of the art implementation approach

- **Use deep learning approaches for ASR/TTS. Keep these modules separate at first**
- **Use LLM for NLU and NLG**
    - Create custom markup representation for slots and their values in both NLU and NLG
    - Fine tune LLM for your tasks once initial system is stable
- **Dialog control + state tracking: Use specialized, smaller LLMs**
    - LLM can guess about next action to take
    - Track semi-structured representation of dialog so far using LLM to update state
    - Interface with tasks / actions / external APIs using structured output from LLM
- **Training options (depends on availability):**
    - Use supervised learning to optimize per-task outputs
    - Interact with live/simulated users for reinforcement learning
    - Specialize NLU/NLG and ASR/TTS to domain-specific vocabulary for your task

Dialog State Tracking As a Task

- Predict state (slot values + dialog act) from

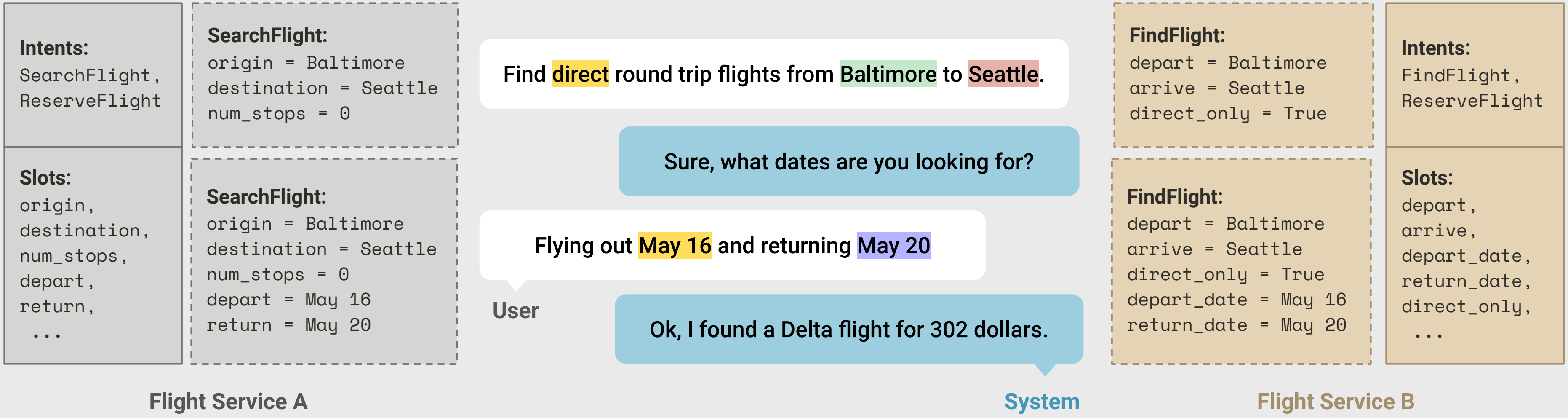


Figure: Dialogue state tracking labels after each user utterance in a dialogue in the context of two different flight services. Under the schema-guided approach, the annotations are conditioned on the schema (extreme left/right) of the underlying service). [DSTC8 overview]

Dialog State Tracking

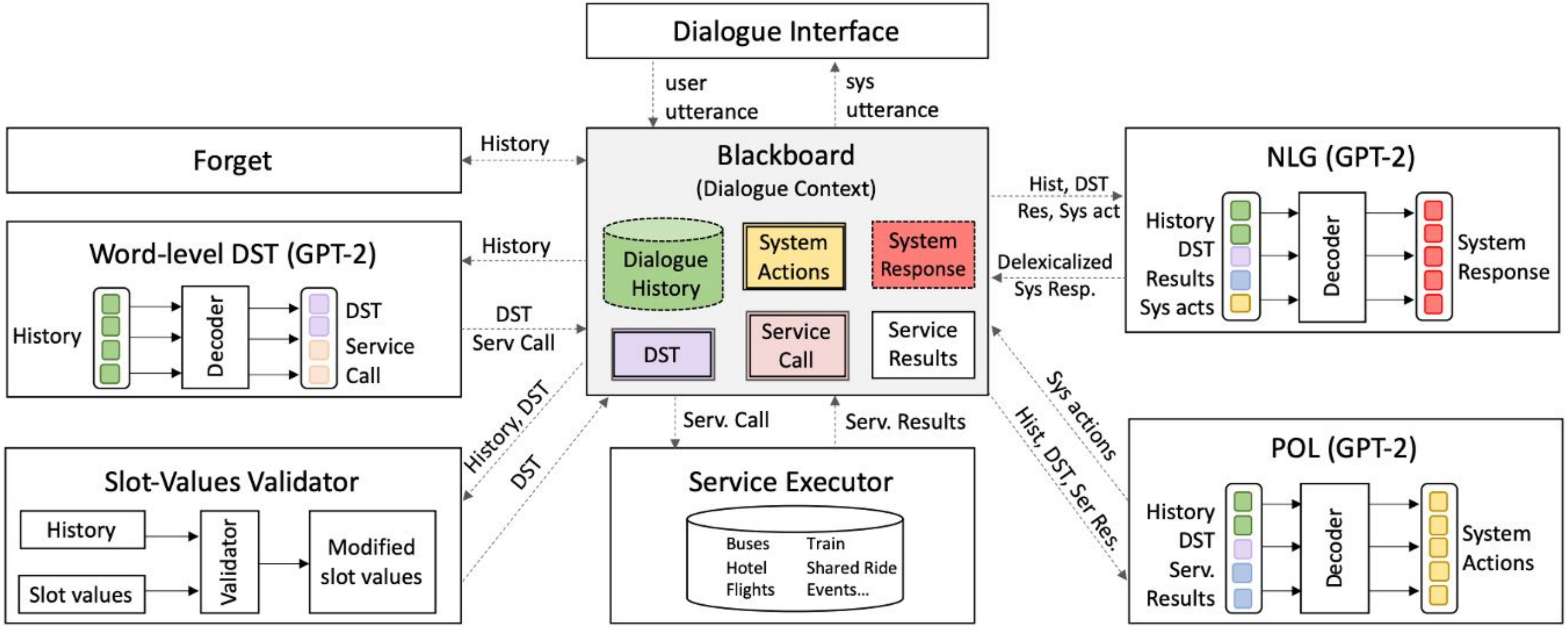
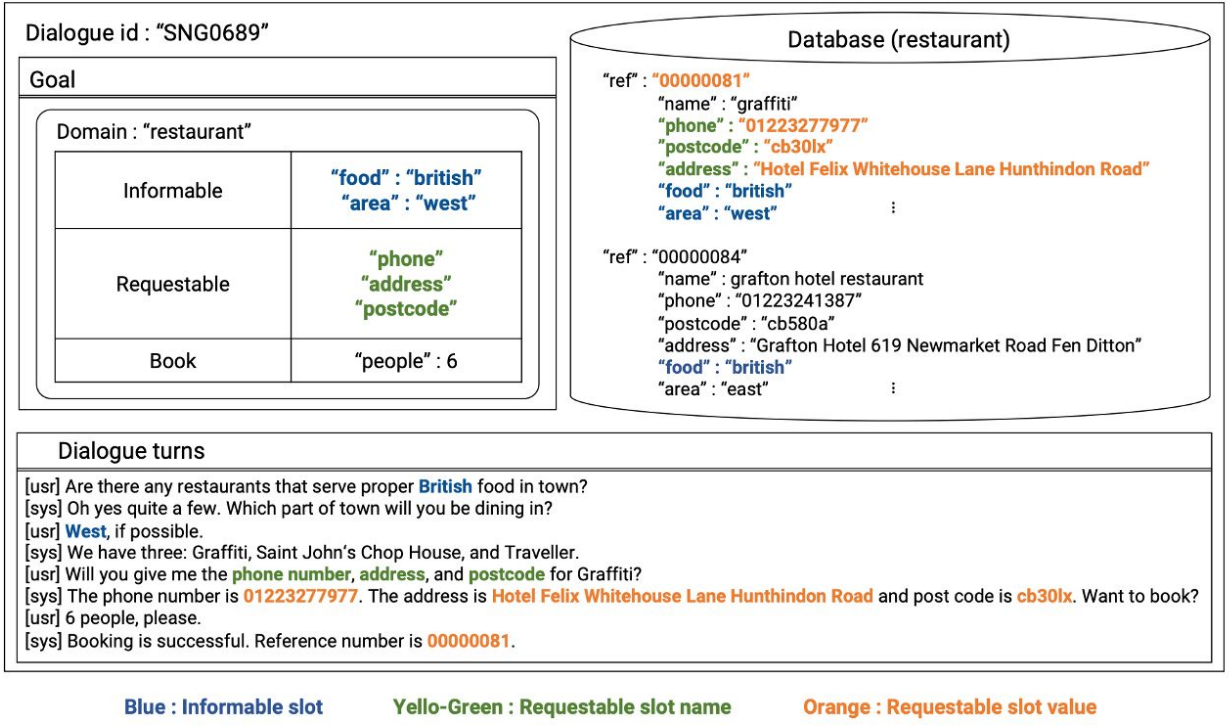


Figure: Dialogue System Architecture. Arrows illustrate retrieving/updating information from/to the blackboard. Boxes labeled GPT2 (DST, POL, NLG) represent the same neural module which is invoked multiple times using different aggregated inputs. Dotter boxes contain symbols and double-line boxes contain natural language. (Romero *et al*, 2021)

MultiWOZ Single Domain Example

Figure:
(Ham et al. ACL 2020)



End-to-end Neural Pipeline for Goal-oriented Dialogue Systems Using GPT2

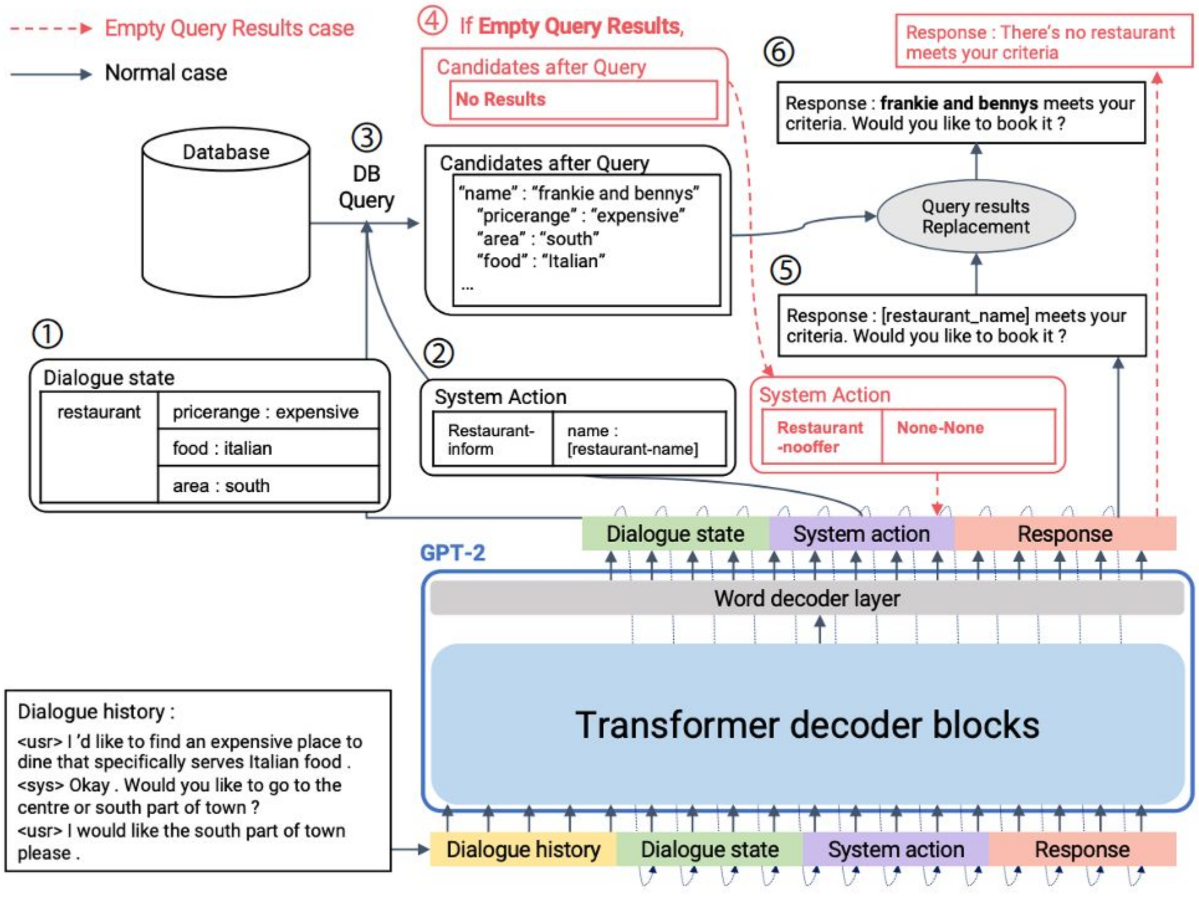


Figure: The overview of end-to-end neural dialogue model. For the transformer, we use fine-tuned GPT=2. The dashed line represents the information to and from DB query, which invoked when the system action needs to fetch an actual value from the database. (Ham et al. ACL 2020)

End-to-end Neural Pipeline for Goal-oriented Dialogue Systems Using GPT2

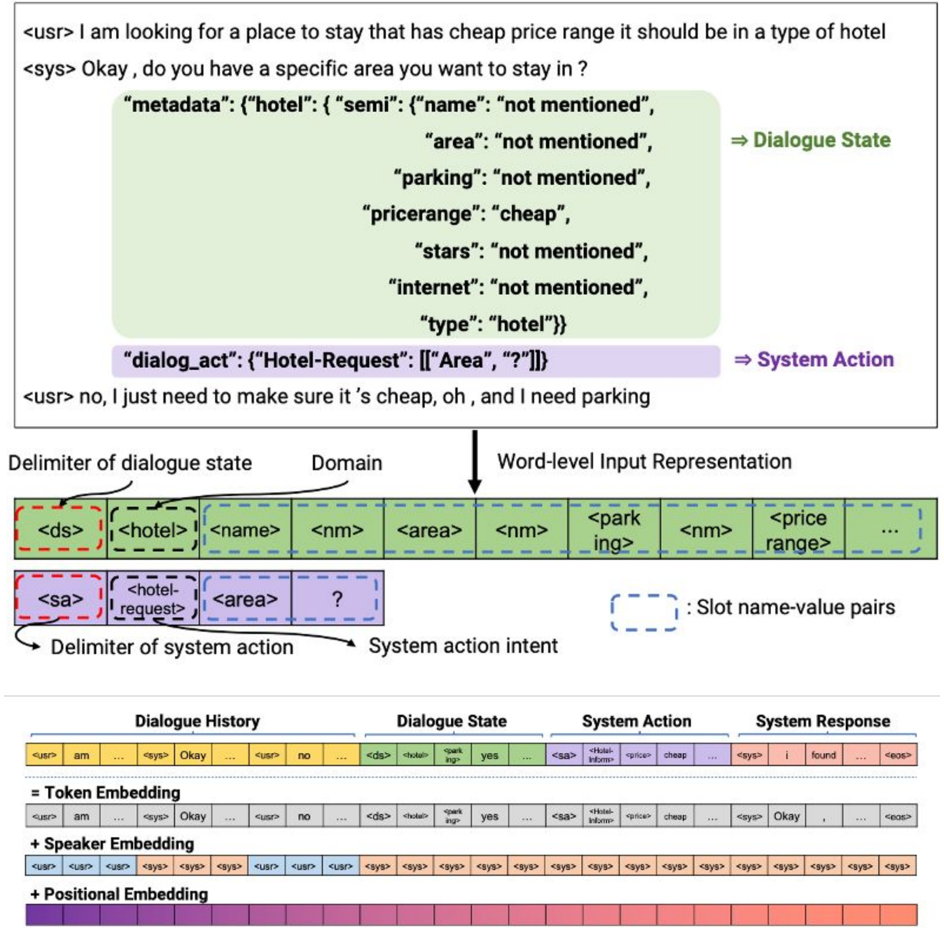


Figure 1: The MultiWOZ dataset, the 'metadata' is treated as the dialogue state and the 'dialogue act' is treated as the system action.

Figure 2: Input representation for fine-tuning GPT-2.
(Ham et al. ACL 2020)

End-to-end Neural Pipeline for Goal-oriented Dialogue Systems Using GPT2

Rank	Team ID	Success Rate ↑	Language Understanding ↑	Response Appropriateness ↑	Turns ↓
1	OURS(504430)	**68.32%**	**4.149**	**4.287**	19.507
2	504429	65.81%	3.538	3.632	15.481
3	504563	65.09%	3.538	3.840	**13.884**
4	504651	64.10%	3.547	3.829	16.906
5	504641	62.91%	3.742	3.815	14.968
N/A	Baseline	56.45%	3.097	3.556	17.543

Table 1: Overall results of the human baseline evaluation carried out by DSTC8 organizers. Only top 5 teams and the baseline are compared.

Model	Joint Acc.	Slot Acc.
GLAD (Zhong et al., 2018)	35.57	95.44
GCE (Nouri and Hosseini-Asl, 2018)	36.27	98.42
SUMBT (Lee et al., 2019a)	46.64	96.44
TRADE (Wu et al., 2019)	**48.62**	**96.92**
OURS + greedy	44.03	96.07

Table 2: Performance comparison with the state-of-the-art models in Dialogue State Tracking benchmark of MultiWOZ dataset.

Model	Inform	Success	BLEU
BASELINE (Budzianowski et al., 2018)	71.29	60.96	18.80
TOKENMOE (Pei et al., 2019)	75.30	59.70	16.81
HDSA (Chen et al., 2019)	82.9	68.90	**23.60**
STRUCTURED FUSION (Mehri et al., 2019)	82.70	72.10	16.34
LARL (Zhao et al., 2019)	**82.78**	**79.20**	12.80
OURS + greedy	77.00	69.20	6.01

Table 3: Performance comparison with the state-of-the-art models in Dialogue-Context-to-Text-Generation benchmark of MultiWOZ dataset.
(Ham et al. ACL 2020)

End-to-end Neural Pipeline for Goal-oriented Dialogue Systems Using GPT2

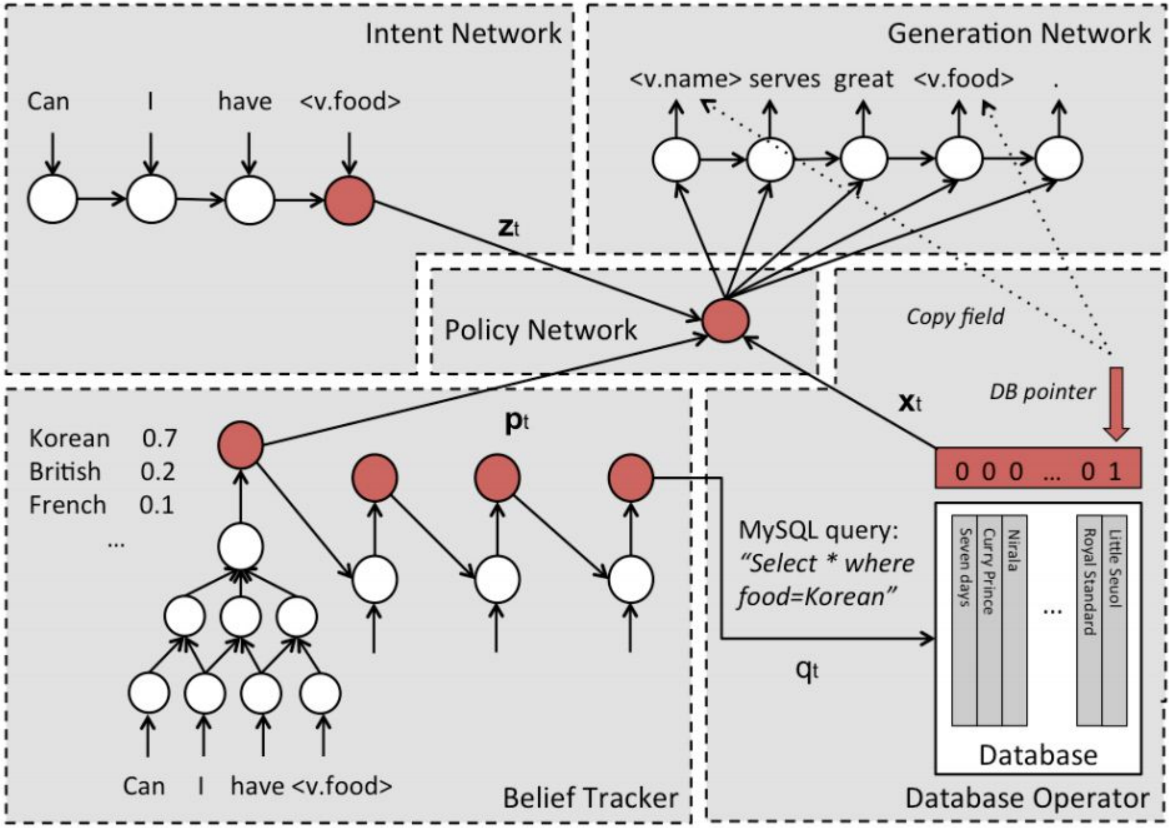
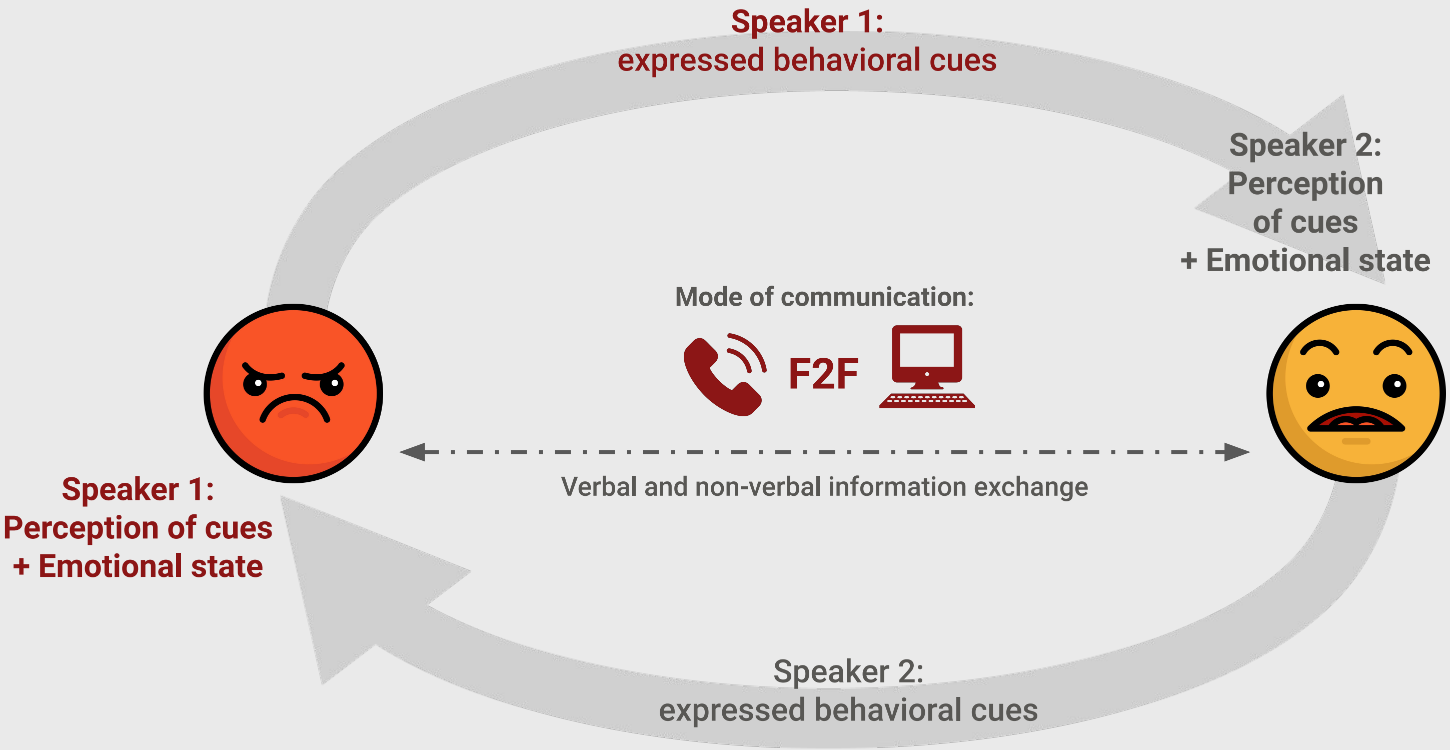


Figure: The proposed end-to-end trainable dialogue system framework.

(Wen et al. EACL 2017)

Final note on design: Behavior cues and emotional content



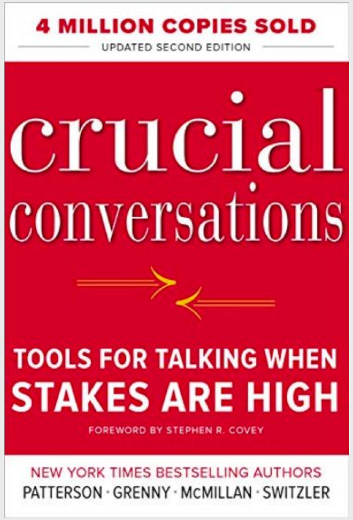
Full human dialog + emotion

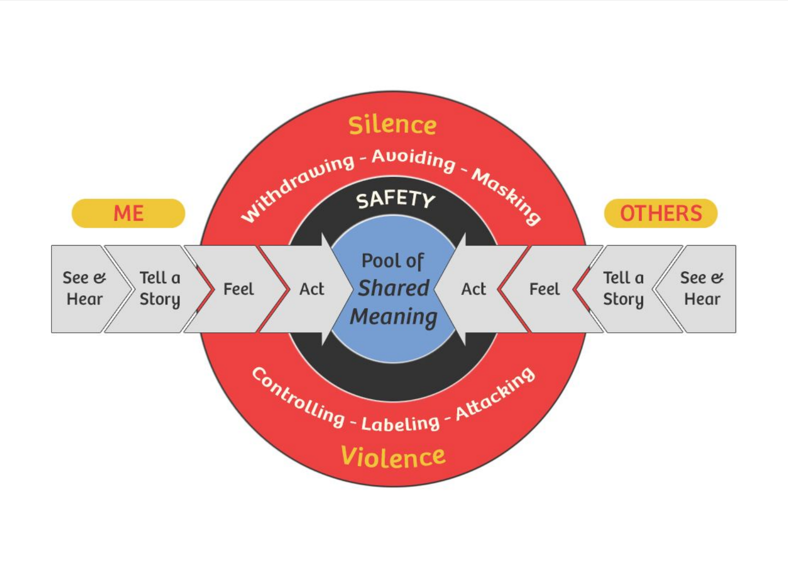
- True dialogs are a mix of information exchange and emotional exchange
- Tracking and addressing emotional dynamics in conversations is critical for contentious topics

The five skills for effectively sharing or stating your views (particularly controversial, touchy, or unpopular views) can be easily remembered with the acronym STATE:

Share your facts
Tell your story
Ask for others' paths
— The "**What**" Skills

Talk tentatively
Encourage testing
— The "**How**" Skills

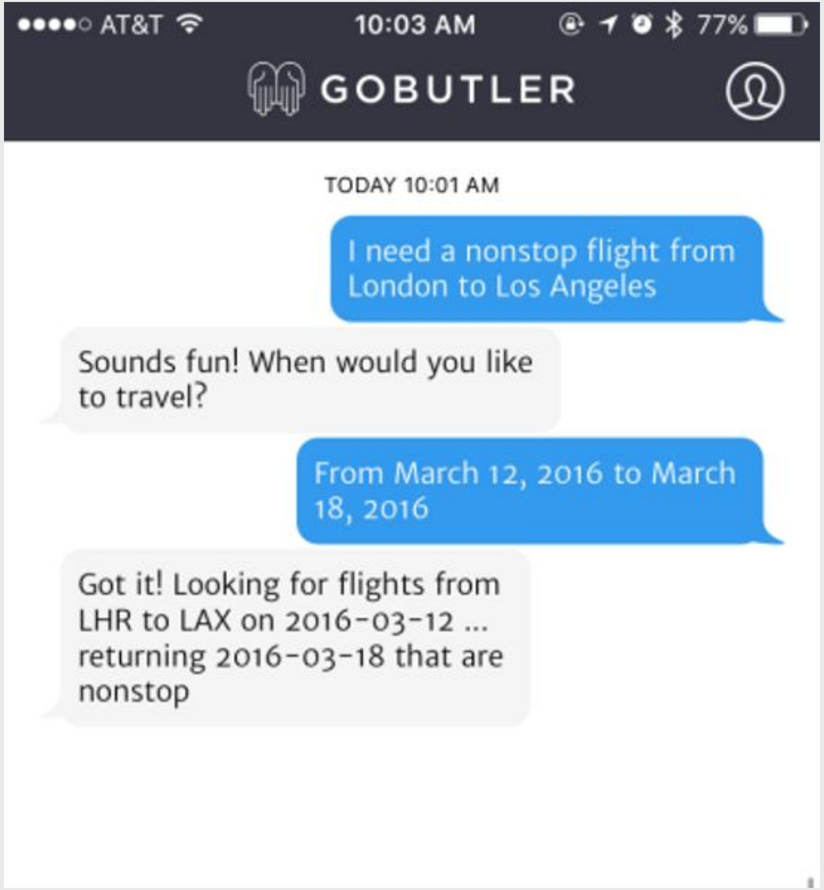




Dialog case study: Asynchronous assistants

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

415-123-1234 Search

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 😬

Alexa Skills Kit Overview

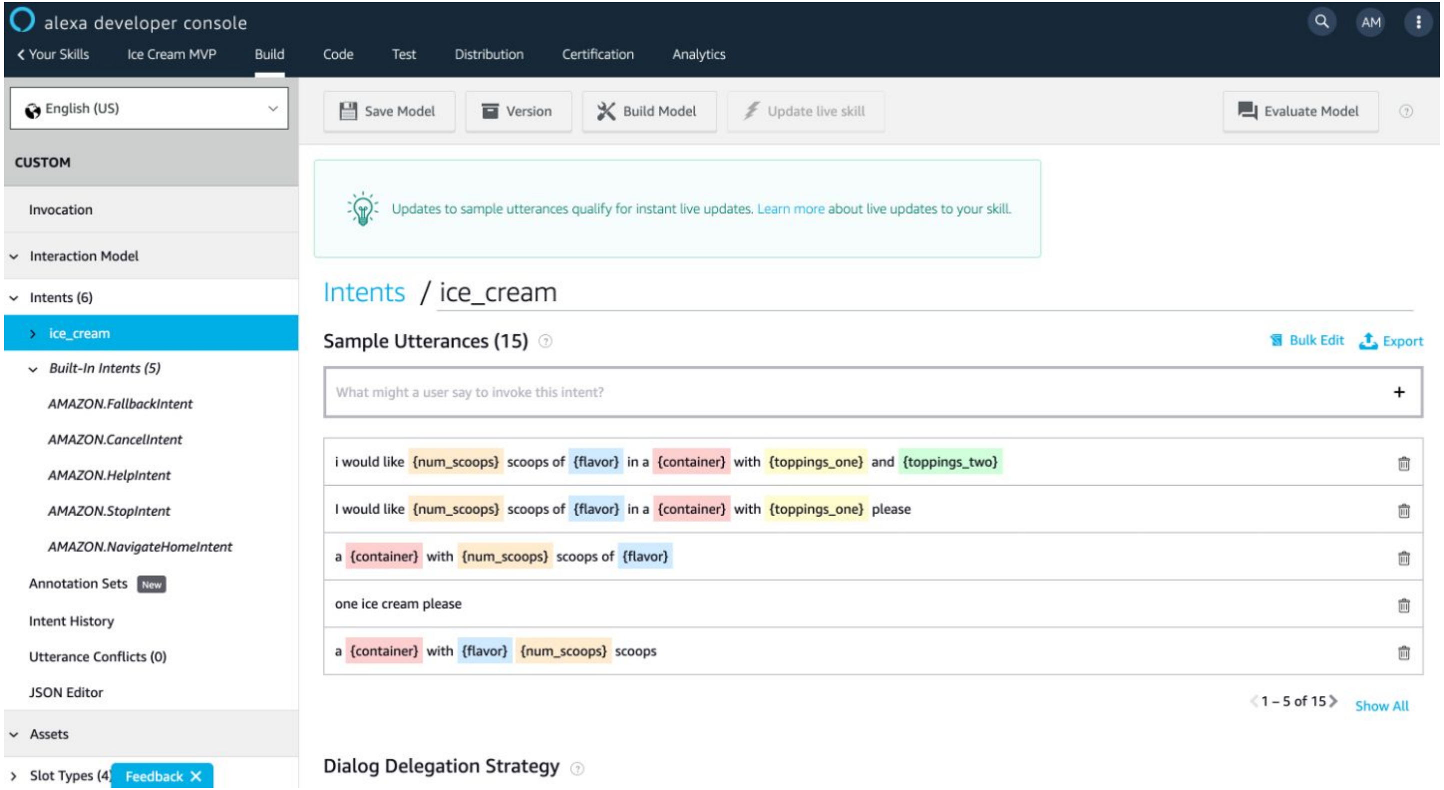
Alexa Skills Kit

- 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 intent and slot examples (2020)



Alexa Domain Classification

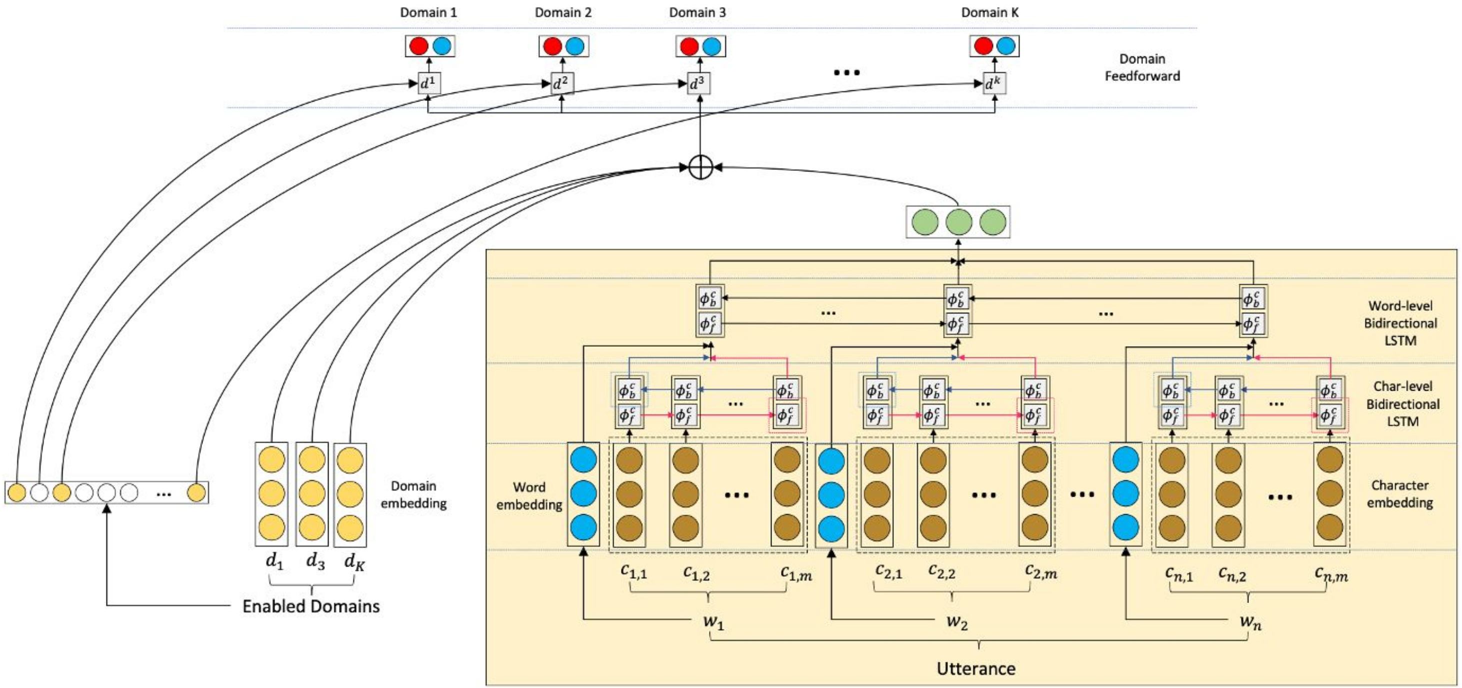


Figure: The overall architecture of personalized dynamic domain classifier. [Kim et al, 2018]

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 docs)

ASK Intent JSON Example

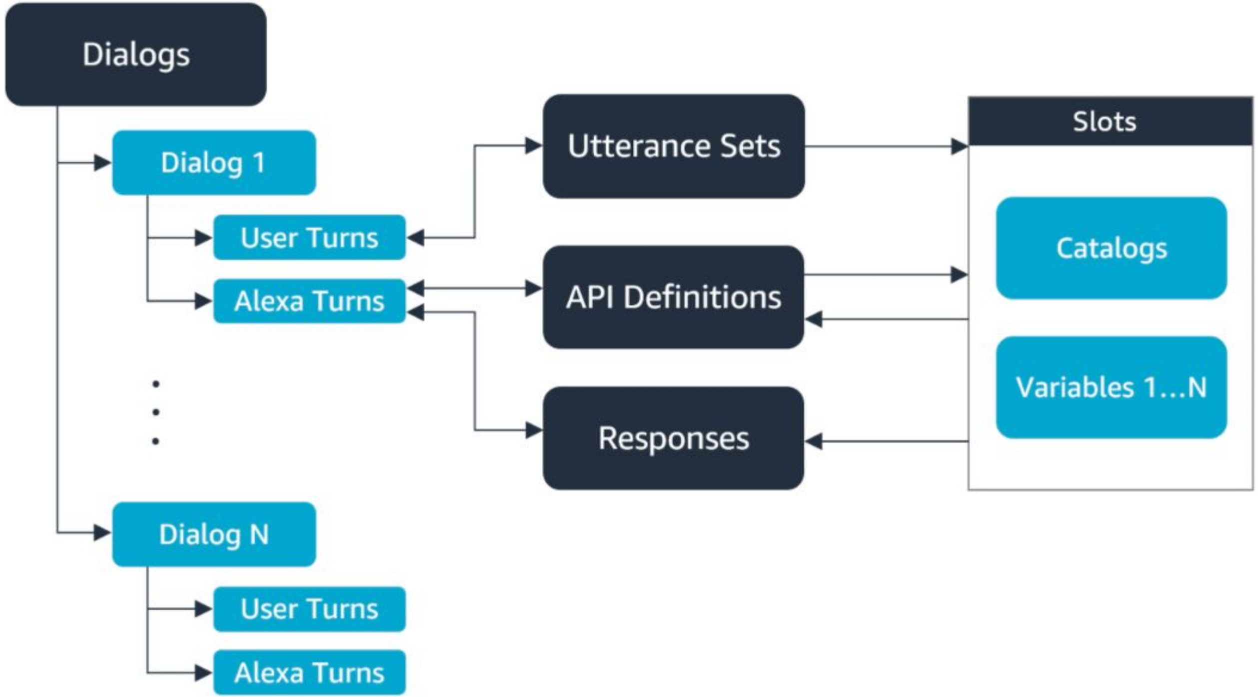
This example shows a portion of the intent object for a `PlanMyTrip` intent. The utterances for the intent are in `interactionModel.languageModel.intents[].samples`. Each slot has its own samples array. For brevity, other properties within `interactionModel` and `languageModel` are not shown


```
{ "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} " ] } ] }
```


(ASK docs)

Alexa Conversations (2020)

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



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

Thank You