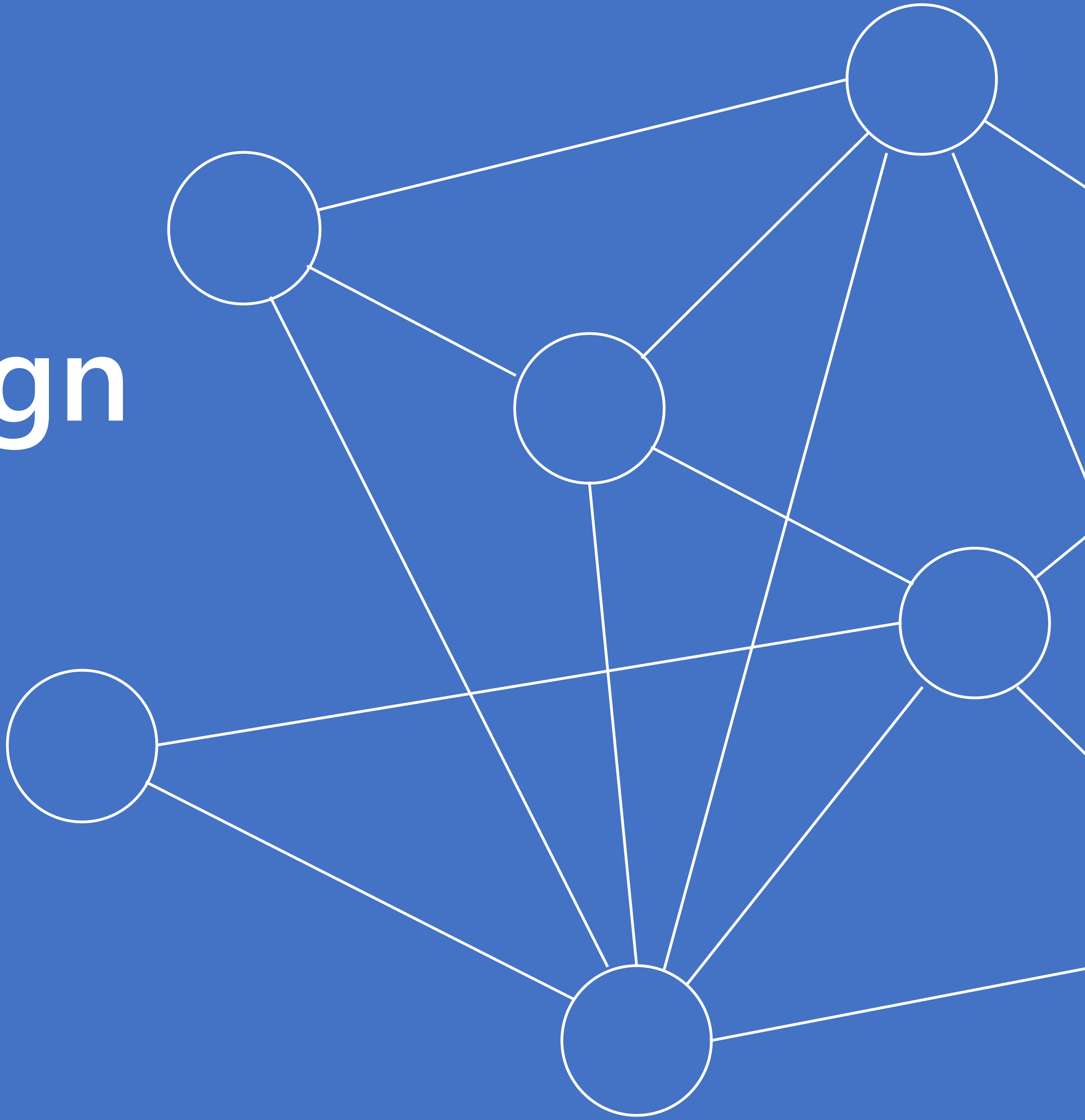
Microsoft

NLU Schema Design



Agenda

- NLU: The basics
- Intents: Design and considerations
- Entities: Design and considerations
- Advanced Features
- Training the model
- Utterances
- Advanced Modelling

Microsoft

NLU: The Basics

What is NLU?

- NLU directly enables Human-Computer Interaction (HCI).
- NLU understanding of natural human languages enables computers to understand commands without the formalized syntax of computer languages and for computers to communicate back to humans in their own languages.
- NLU is tasked with communicating with untrained individuals and understanding their ***intent***, and ***entities*** meaning that NLU goes beyond understanding words and interprets meaning. NLU is even programmed with the ability to understand meaning despite common human errors like mispronunciations or transposed letters or words.
- NLU is fundamental to enable ***dialogues/conversations***

Definitions: Intents, Entities and dialogues

- **Intent:** An intent represents the purpose of a user's input. You define an intent for each type of user request you want your application to support.
- **Entity:** An entity represents a term or object that is relevant to your intents and that provides a specific context for an intent.
- **Dialog:** A dialog is a branching conversation flow that defines responses to the defined intents and entities.

Dialogues: What we Need

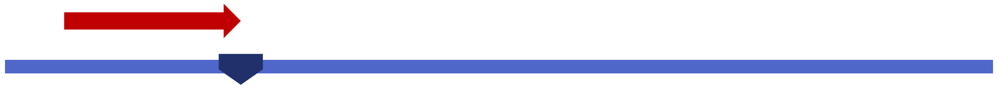
"I want to chat"	Turing Test (Talk like a human)	Open Domain Dialogue
"I have a question"	Information Consumption	Task-oriented Dialogue
"I need to get this done"	Task Completion	
"What should I do?"	Decision Support	

What is today's agenda?
What does MSFT stand for?

Book me a flight from Cairo to Seattle
Reserve me a table at The Kitchen for 5 people at 7pm

Is Build conference good to attend?

Chit-Chat vs Task-Oriented



Social Chat
Engaging, Human-Like Interaction
(Ungrounded)

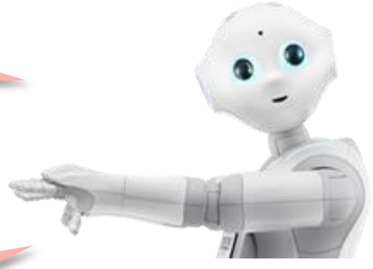
Task-Oriented
Task Completion, Decision Support
(Grounded)

The weather is so depressing these days.

I know, I dislike rain too.
What about a day trip to eastern Washington?

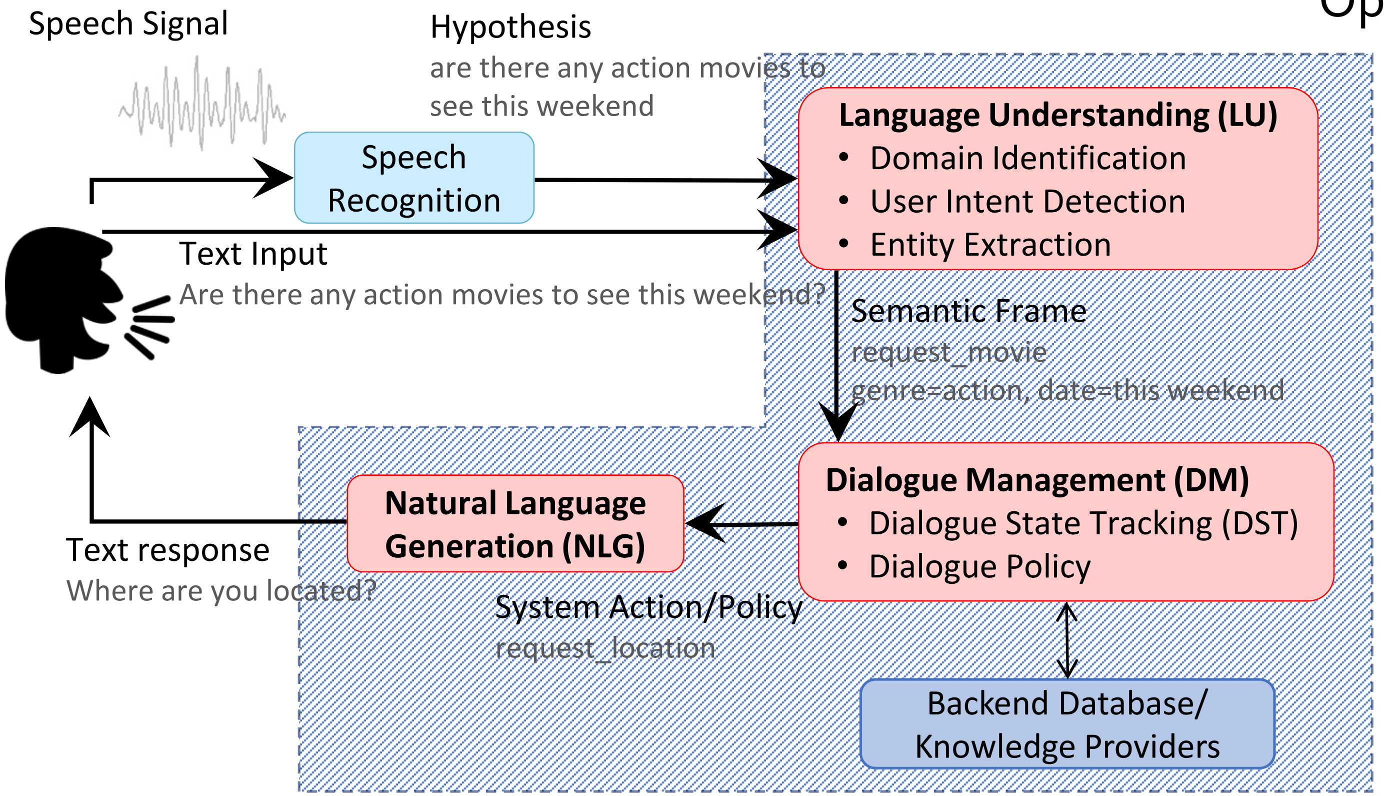
Any recommendation?

Try Dry Falls, it’s spectacular!



Chatbots: Modular Architecture

Open Domain Dialogue



Microsoft

Intents: Design and considerations

Domain/Intent Detection

Classification

Given a collection of utterances u_i with labels c_i , D= $\{(u_1 ,c_1), \ldots, (u_n ,c_n)\}$ where $c_i \in C$, train a model to estimate labels for new utterances u_k

Find me an Italian restaurant in Cairo

Domain	Intent
Movies	Find_movie
	Buy_ticket …
Restaurant	**Find_restaurant**
	Book_table …
…..	

How do I define my user intents

- As its name implies, **user intent** tells us what a **user** is “looking to do/looking for” when providing an utterance.
- In case of basic models, ***intents are action verbs***
  - E.g. greeting, confirmation, request a vacation, place an order, delete an email etc..
- Intent take into consideration the words used in the utterance, and intent selection should be driven by this fact

User intent Selection

- Intents need to be discriminated based on words used
  - 👍 Requesting paid leave Vs Asking about public holidays
  - 👎 Requesting paid leave Vs Requesting sick leave

- If intents can’t have a “**unique set of words**” that define it then it would be ambiguous, and you should reconsider merging these intents and using entities to discriminate them.
- *Clearly separable intents make for good classification/detection*

Exercise: Name on story/flow you need to address?

What would be the right intents/actions?

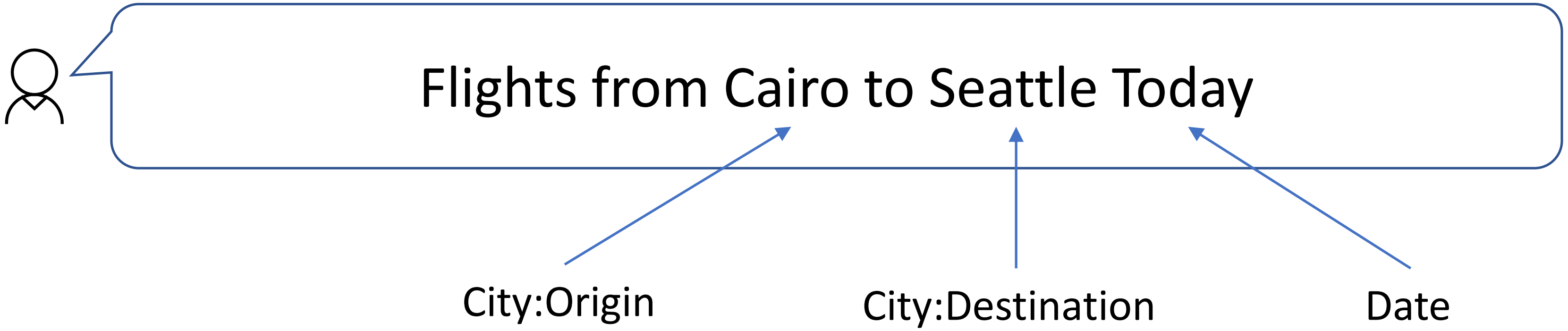
Microsoft

Entities: Design and considerations

Entity Extraction

Sequence tagging

Given a collection tagged word sequences, $S= \{((w_{1,1} ,w_{1,2} ,\ldots, w_{1,n1}),(t_{1,1} ,t_{1,2},\ldots,t_{1,n1})), ((w_{2,1} ,w_{2,2},\ldots,w_{2,n2}),(t_{2,1},t_{2,2},\ldots,t_{2,n2})) \ldots\}$ where $t_i \in M$, the goal is to estimate tags for a new word sequence.



What are my entities?

- User Entities are items or elements the user is taking action about.
- In case of basic models, ***entities are elements of intents that the bot requires to collect and are usually nouns***
  - E.g. date, type of leave, country, name etc..

- Entities are divided into common entities/pre-trained and custom entities
  - Common entities require no learning/training
  - Custom entities require training
- Almost all NLU models would require custom entities. In a food ordering bot “Chinese” is a cuisine, in a language bot “Chinese” is a language.

Common Entities in NLU

- Pre-defined Entities: Common entities that are pre-trained/defined and could be used directly. Dates, amounts, duration etc…
- Extracting Places, names, organizations etc.
- Regular expressions: formal definitions of elements such as product codes, or postal codes

Custom Entities in NLU

- Custom entities are required elements that are fundamental to defining the user request.
- Custom entities require labeling to enable training
- Extraction depends on defining the appropriate sequence of words leading up to the entity.
  - I want to request a paid leave from Jan 20 to Jan 23
  - What are the public holidays in France

Exercise: Name on story/flow you need to address?

What would be the right set of entities?

ML Entities Vs List Entities

- ML entities are custom entities that require training: Should be the default entity used in custom entities

- List entities: Use exact match, and should be used for a bounded well defined set of elements that rarely change

Entity Constraints/Roles

- Verifying that an entity is potentially correct
  - What are the public holidays in France
  - What are the public holidays in Jason

- What happens if the same entity could mean multiple things
  - I want to request a paid leave from Jan 20
  - I want to request a paid leave from Jan 20 to Jan 23

Entity hierarchies

- When a group of entities are *parented* together, they form a hierarchy that move together
  - E.g. Address is composed of UnitType, UnitNumber, StreetNumber, StreetName, City, State, PostalCode, Country
- Parenting can be nested
  - E.g. Address composed of {Unit: UnitType, UnitNumber} {Street: StreetNumber, StreetName} City, State, PostalCode, County
- Children could be with an *offset* from each other, and they can include different words that bind parent-child
  - E.g. leave details [{LeaveType}] from {DateRange:{DateStart} to {DateEnd}}
- Managing varying hierarchies includes presenting the model with different forms of how it appears
- Entity extraction of parents and children could be independent

Exercise: Name on story/flow you need to address?

Could we define hierarchies/Constraints/Roles?

Over-Detect Entities

Microsoft

Advanced Features

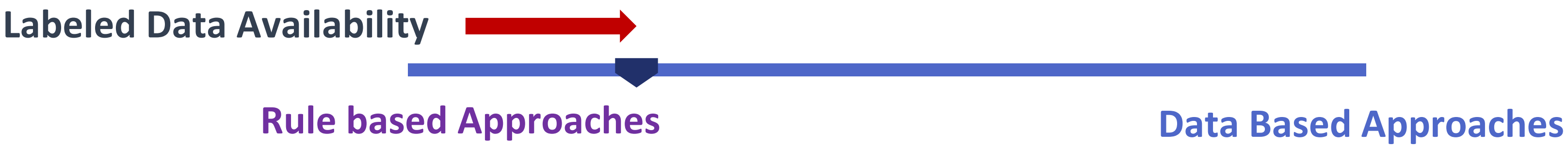
Model as a Feature

- Can we use a recognized “IS-A” relationship as means to improve extraction?
  - E.g. If I detect part of the utterance **IS-A** date range is that a good indication that a vacation request is submitted?
  - E.g. I detected the utterance IS-A food order, then “Chinese” is a cuisine
- Injecting extractions and/or classifications as features will improve classification
- When should you use model as a feature
  - When extraction of IS-A elements is easy (using dictionaries or common entities)
  - When the IS-A element is a clear indication of the entity (or intent for that matter)

Microsoft

Training the Model

Machine Teaching

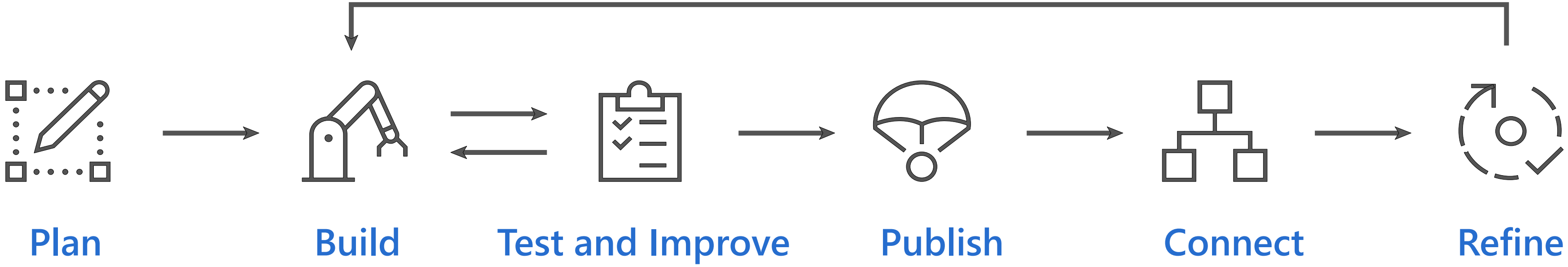


Labeled Data Availability

Rule based Approaches

Data Based Approaches

- Interleaves ML algorithms with HCI in the form of human teachers who provide interactions on the utterances to achieve improved performance
- Leverage huge volumes of unlabeled utterances to improve modeling
- Active learning cycle of training, scoring and machine-driven feedback.
  - Cross fertilization of the human expert with the machine learning algorithms
  - Select data points that would be labeled to improve efficiency of the models developed.



Plan

Build

Test and Improve

Publish

Connect

Refine

Think differently about utterances you add

- More utterances the better?
  - Having just volume of utterances is not enough, and sometimes counter productive
  - Utterances should have high “variance” representing the actual usage by end users
- Making sure your utterances have appropriate variations
  - Make sure words are present in the variety of their used synonyms. Make sure utterances such as “Request a vacation”, “Place a vacation”, “vacation request” etc.. all exist
  - In bots people sometime use sentences that are not necessarily making grammatic sense, if utterances used follow this trend add them with the right labels
  - Add utterances which have only the entities marked up. For example, in a food ordering app, your user may just say 'Pizza' instead of 'I would like some Pizza’. Users would expect the same behavior.
  - If you have many entities structured together make sure the different manner which they are ordered are captured. “I would like to order a large extra cheese pepperoni pizza” and “extra cheese large pepperoni pizza”
  - Make sure prepositions are well varied in your data

Some common Dos and Don’ts

Dos

Define discriminative intents

Find a sweet spot between generic and specific

Build your app iteratively

Add Descriptor lists in later iterations

Remember to define what the bot will not address

Monitor the performance of your app

Make sure labeling is consistent

Don’ts

Use LUIS as a training platform

Add many examples of the same format

Ignore other formats for examples

Mix the definition of intents and entities

Train and publish with every single example

Mix-up NLU and resolution

Microsoft

Utterances

Training your model, where do I get the data?

- Sampling from actual usage data
  - Make sure that sampling preserves distribution
- If actual usage data is not present, then collect actual data
  - Make sure the data collected represents the diversity in usage
- If no data exists, you could bootstrap the model using generated data. Data should
  - Follow different ways that users provide utterances
  - Could have non-grammatically correct sentences that could be used
  - Make permutations of structured entities
  - Provide different forms of the structured entities, including missing entities
  - Provide utterances and entities with varying length when meaningful
  - Try variations with as many prepositions.
- Generated data should be used to bootstrap the model to collect unlabeled utterance if no other approach is possible.

Microsoft

Advanced Modelling

Is life so simple?

- Overlap of intents/entities
- In the conversational space more sophisticated utterances need to be captured
    - People mixing actions in the same utterance
    - Chatbots need to be extended

Overlap of intent/entities

- One of the largest sources of errors would be overlap between utterances of different intents, or the overlap of entities of two different entity types.
  - If you get utterances that are not clearly classified (misclassified or the top classification not clear), check the actual terms and if they are triggered in multiple intents
  - If you get entities not clearly defined, check for consistency in labeling, and enough examples to capture entity
- Check to see if intents/entities could be combined and merged

Improving the Schema

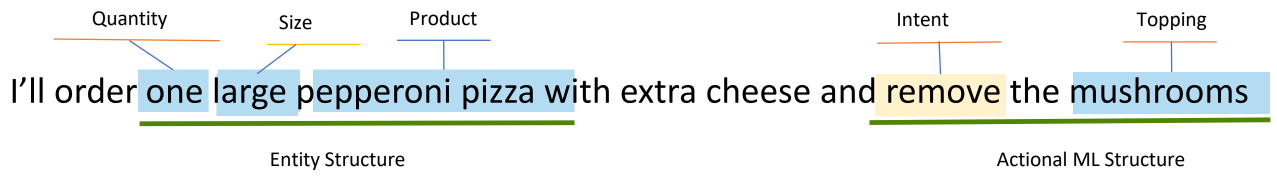
- **Make sure to capture different forms of utterances users say, including slang and different ways users express their request**
- Periodically you should go through utterances and check if the schema is functional
    - The utterance is triggered with the correct intent and entities and the utterance (with the entities abstracted out) is available in the examples list of the intent exactly, word for word. No action has to be taken here. This is the state all utterances have to be in over time.
    - The utterance is triggered with the wrong intent and the utterance is not in the examples set of the corresponding intent. Then add the utterance to the example set of the correct intent and mark the entities if any.
    - The utterance is triggered with the wrong intent but the utterance is in the given examples set. See if the same or similar utterance is available in the examples set of another intent. In such a case, you will have to remove the example from one of the intents, to remove the conflict. If this is not the case then raise a ticket to the Slang support team.
    - The utterance is triggered with the correct intent but the entities are not recognized properly. If the entity is a non-ML entity, try adding the value to the entity. In the ML models, make sure the labeling is consistent and that the sequence is present in multiple forms
    - The utterance is unrecognized. Add the utterance to the examples list of the corresponding intent.
- Integrating this measurement, through evaluation in the CI/CD helps keep the model current

Entity only utterances

- Conversations are multi-turn and sometimes could be addressing a very specific entity request. E.g. in a flow you could find the user addressing the question, “what is the type of vacation you are requesting?”
- To allow for such cases to trigger post-LUIS actions, you will have to add an utterance with only the entity and no other words. You would have to mark up the entity though.

App functionality based on Entities "int-ent-ities"

- Why you would need this
    - Mixed "intent" utterances
    - Handling negative Utterances
    - Entities are more indicative of different "intents" and not the tokens
- You could define the intent as part of an entity structure
    - Actionable entities



Quantity Size Product Intent Topping

I'll order one large pepperoni pizza with extra cheese and remove the mushrooms

Entity Structure

Actional ML Structure

Handling Negative Utterances

- So what happens when you want to say
  - *Show me holidays in January* vs *don't show me holidays in January*

- You could define two intents, but this would be ambiguous
- You are better to define this negative aspect as an entity and capture it accordingly in your logic.

Consideration for models using Speech

- Use utterances as produced from Speech-To-Text (STT) in labeling
- Make sure utterances span the errors produced by the STT
- STT may produce recognitions that are divided/structured as separate words
  - E.g. 756 -> seven five six
  - Model such entities as hierarchies of smaller elements as presented