Natural Language Processing
IST 664 / CIS 668

RoverBot

Natural Language Processing Application Investigation Report

Ananya Bhupathipalli
Koushik Nagaraj Godbole
Omkar Mutreja
Ritesh Dhakad

Table of Contents

INTRODUCTION

What are chatbots?

Bots are simple man-made artificial intelligence systems that you interact by the means of text or audio communication. Those interactions can be clear and straightforward, for example asking a bot to give you flight details and make hotel reservations for you or more complex, such as having one investigate an issue with your web service.
One of the most important reason for the increasing demand of chatbots is that nowadays many people are just tired of downloading numerous applications for performing mundane tasks. These activities can be performed by a bot.

How are chatbots different from applications?

The facts confirm that, the lines between applications and chatbots can turn into somewhat obscured if chatbots cooperate by means of a user interface. A chatbot anyway can be separated from an application in the manner in which that the interactions with the bot occur, pretty much sequentially (as a conversation), and the bot is utilized inside a chat application.
Another way by which a chatbot is not quite the same as an application is somewhat more reminiscent of the sci-fi example, and that is the chatbot as illustration for a automated agent. A chatbot unlike an application has an "identity" that is isolated from its interaction with the users. This is similarly that the human operator exists freely of their interaction with users.
This representation can be reached out to the point where a single chatbot could collaborate with the user over different communication channel, for example, yet continually keeping up
In short a chatbot is another method for people interacting with software. While there are overlaps with usefulness offered by websites and applications, communicating with a chatbot is diverse to cooperating with a site or with an application.

Travel Bots - Future of Travel Industry

The applications of chatbots are widespread in different industries and one such industry is travel, tourism and hospitality. There are many top companies who have successfully deployed chatbots for customer facing roles. Chatbots are ideal for travel and tourism industry which is a customer facing, and there is a need for customer assistance 24/7.
Chatbots can fill in the roles of the front-end customer care representatives. Expedia is the leading firm in travel industry which has successfully deployed chatbots on their websites to assist customer in their travel bookings and queries. Another greatly loved bot in travel industry is ‘Kayak’, which provides us with all the necessary information about flight booking, hotel bookings and other travel related queries/bookings.

Advantages of using chatbots:

- It eliminates the waiting time of customers to speak to the next available representative for enquiries
- Chatbots can be made multilingual, thereby creating a diverse market.
- From business perspective, it is easy for them to work on the data collected through chatbots for analyzing and understanding their customer requirements and improve their services.
- The firms can invest more time and man power on other areas as the chatbots are capable enough to handle majority of customer queries efficiently.

NLP Chatbots - Research and Techniques

How chatbots work?

Before understanding how chatbots work, it is essential to understand three main terminologies widely used in the chatbot architectures:

1. Natural Language Processing (NLP)
2. Natural Language Understanding (NLU)
3. Natural Language Generation (NLG)

Natural Language Processing (NLP)

NLP refers to a range of computational technologies that help machines comprehend the input given by a user and respond in a way which has human-like language processing capabilities and is easily understood by the user.

Natural Language Understanding (NLU)

Natural language understanding (NLU) or natural language interpretation (NLI) is a subtopic of natural language processing with respect to artificial intelligence which deals with machine reading comprehension. This involves in the extraction of important aspects from the user inputs.
In Natural Language Understanding stage, system performs various language analysis and meaning is conveyed by each and every level of language and since humans have been shown to use all levels of language to gain understanding, the more capable an NLP system is, the more levels of language it will utilize.

Levels of Natural Language Analysis Chatbots Performs:

Phonology: This level deals with the interpretation of speech sounds within and across words. There are, in fact, three types of rules used in phonological analysis: 1) phonetic rules – for sounds within words; 2) phonemic rules – for variations of pronunciation when words are spoken together, and; 3) prosodic rules – for fluctuation in stress and intonation across a sentence.

Morphology: This level deals with the componential nature of words, which are composed of morphemes – the smallest units of meaning. For example, the word preregistration can be morphologically analyzed into three separate morphemes: the prefix pre, the root registra, and the suffix tion. Since the meaning of each morpheme remains the same across words, humans can break down an unknown word into its constituent morphemes in order to understand its meaning.

Lexical: At this level, humans, as well as NLP systems, interpret the meaning of individual words. In this processing, assignment of a single part-of-speech tag to each word is done, words that can function as more than one part-of-speech are assigned the most probable part-of-speech tag based on the context in which they occur.

Syntactic: This level focuses on analyzing the words in a sentence so as to uncover the grammatical structure of the sentence. This requires both a grammar and a parser. The output of this level of processing is a representation of the sentence that reveals the structural dependency relationships between the words.

Semantic: Semantic processing determines the possible meanings of a sentence by focusing on the interactions among word-level meanings in the sentence. This level of processing can include the semantic disambiguation of words with multiple senses; in an analogous way to how syntactic disambiguation of words that can function as multiple parts-of-speech is accomplished at the syntactic level.

Discourse: While syntax and semantics work with sentence-length units, the discourse level of NLP works with units of text longer than a sentence. That is, it does not interpret multi sentence texts as just concatenated sentences, each of which can be interpreted singly. Rather, discourse focuses on the properties of the text as a whole that convey meaning by making connections between component sentences.

Pragmatic: This level is concerned with the purposeful use of language in situations and utilizes context over and above the contents of the text for understanding.

Natural Language Generation (NLG)

NLG involves the generation of response for the chatbots, depending on the handling done in the NLU step. NLG processes turn structured data into text.

Natural Language Generation Techniques:

AIML (Artificial Intelligence Markup Language)

This type of model is very popular for entertainment bots. The simplest technology is using a set of rules with patterns as conditions for the rules. AIML is a widely used language for writing patterns and response templates. Bot developers write code in AIML language, the code can include multiple units like this:

```
<category>
<pattern>WHAT IS YOUR NAME</pattern>
<template>My name is Michael N.S Evanious.</template>
</category>
```

When the chatbot receives a message, it goes through all the patterns until finds a pattern which matches user message. If the match is found, the chatbot uses the corresponding template to generate a response.

ChatScript

ChatScript is a modern implementation of this idea. It is an open source chatbot engine which allows defining a chatbot in a rule-based language. Each rule contains a pattern and an output.

ChatScript engine has a powerful natural language processing pipeline and a rich pattern language. Using ChatScript we can do much more than with AIML. It will parse user message, tag parts of speech, find synonyms and concepts, and find which rule matches the input. In addition to NLP abilities, ChatScript will keep track of dialog, so that we can design long scripts which cover different topics. It won't do anything fancy, though. It won't run machine learning algorithms and won't access external knowledge bases or 3rd party APIs unless we do all the necessary programming.

NLP Chatbot Architecture

The current tools available for building a NLP based chatbot have a similar architecture as shown in the below. The main units of a Chatbot are as follows:

Presentation Layer

The first layer is the presentation layer which consists of the interface to chat with the chatbot. The user can only view the presentation layer. The user gives their input message and receives the bot's response on this interface. The presentation layer could be a Facebook Messenger, any Windows app etc.

Machine Learning Layer

The input message is sent to the Machine Learning Layer. This layer consists of NLP/NLU unit and a Decision Engine. In the NLP/NLU unit, the machine understands and interprets the user's message and provides codified commands for itself. These commands are then sent to the decision engine, which decides what the user needs, what is the context and what is the most suitable response for the query.

Data Layer

The data layer consists of a data connector and custom data source unit. The custom data source consists of all the custom and built-in responses, training phrases etc. All the data we use to customize our chatbot is present in the custom data source unit. The data connector acts as a connector between the decision engine and custom data source unit and also connects the custom data source unit to the NLG unit.

Natural Language Generation(NLG)

NLG unit represents machine output in text format so as to be understood by the user.

Messaging Backend

The responses from NLG go to the messaging backend unit which then sends the response back to the presentation layer.

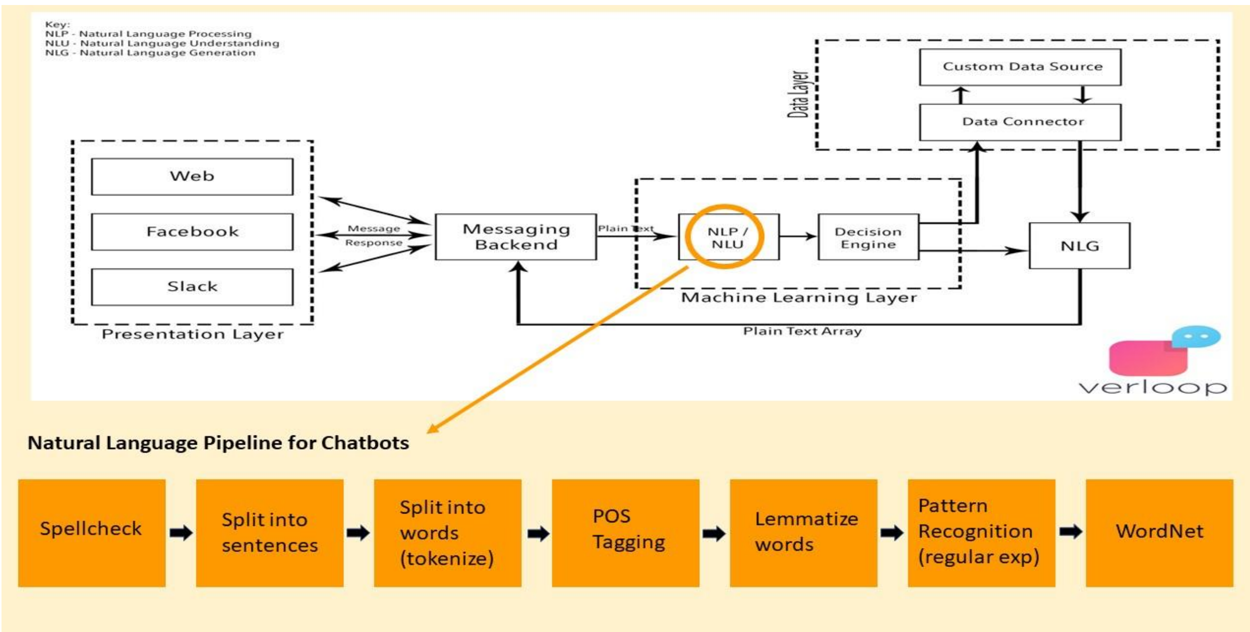

Fig 1. Architecture Diagram for Chatbots

The main unit of the chatbots is a pre-processing NLP/NLU unit. The Natural Language Pipeline for chatbots include these steps:

1. **Spell check:** In this we get the input text from the user and check for the spelling. If there are any spelling errors, we can fix the errors with simple method or build a spell checker using deep learning
2. **Split the input into sentences:** It is a best practice to split the input into sentence as it is very helpful to analyze every sentence separately. Splitting the text into sentences is easy, you can use one of NLP libraries, e.g. NLTK, StanfordNLP, SpaCy.
3. **Split the sentence into words/Tokenize:** This is the next part in text processing, it is a best practice to split the sentence into words as because hardcoded rules typically operate with words. The NLP libraries can do it.

4. **POS tagging:** Some words have multiple meanings, it is necessary to get the correct meaning of the word in the given context. Knowing the parts of speech can help us overcome this ambiguity. You can use same NLP libraries or Google SyntaxtNet.
5. **Lemmatize words:** In many cases, the form of the word is not an important aspect for writing a hardcoded rule. It the preprocessing code able to identify a lemma, a canonical form of the word, it will help in simplifying the rule. Lemmatization is identifying the lemmas, which is based on dictionaries where we will get all forms of word. WordNet is the most popular dictionary for English. NLTK and some other libraries allow using it for lemmatization.
6. **Pattern Recognition/Entity Recognition:** To identify entities we match it with a given set of patterns. There are entities like dates which can be given in different formats and thus it helps to use unified formats for matching the patterns for such entities. We can also use regular expressions to customize and identify certain entities.
7. **WordNet/Find Concepts:** There are pre-processing codes which can be used to identify the concepts. To find concepts/synonyms we can use WordNet.

Types of chatbots

Chatbots can be classified based on various categories like the market they are used in -

Entertainment/Chit-Chat Bots:
These bots are supposed to be fun, creative and smart enough to continue a conversation but need not remember all the details of the dialogue. The quality of these bots can be assessed by how long the conversations last. Short conversation imply the bots were not successful in holding a conversation.

Business/Task-Oriented Bots:
The task-oriented bots are used for transactional purposes such as placing an order, booking a flight, helping with troubleshooting etc. The conversation with these bots are generally short. These bots can be assessed based on how many users reach the end goal - like booking an Uber etc. Travel bots fall in this category.

Bots can also be classified based on the kind of information they are expected to provide as -

Open Domain Chabots:
These bots answer all sorts of queries like - “How will the weather be tomorrow?”, “What the the different species of flowers?” etc. These bots are hard to train and perfect.

Closed Domain Chatbots:
These bots are domain-specific and provide information for specific scenarios such as providing information to tourists about the activities in a place, perform tasks related to applications on a device such as Siri, Cortana, Alexa etc. These bots can be well-trained to answer domain specific queries and

hold the ability to perform really well in real-time settings. Our chatbot RoverBot can be classified as closed domain chatbot.

Another useful classification is based on the type of responses given by the chatbot. They are Generative models and Retrieval-based models.

Generative Models:

The future of chatbots are going to be based on the generative model. These are the smarter bots and does not use any pre-defined repository. It learns from scratch and can make grammatical errors and are less rigid. These can respond better to surprising demands and questions from the customer. However, it is difficult to obtain perfection with a generative model chatbot.

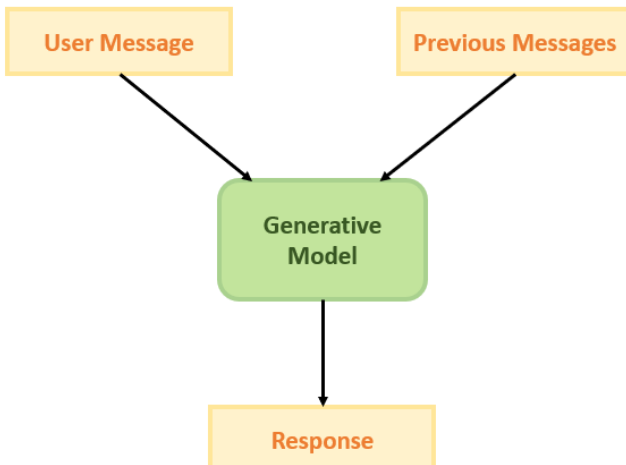

Fig 2. Generative Model Chatbots Workflow

Retrieval-based Models

Retrieval-based models are more practical and widely available in the current market. Most of the algorithms and API available for developing a chatbot are based on Retrieval-based model. This model has a predefined set of responses and uses the user message/input and context of the message to select the best possible response.

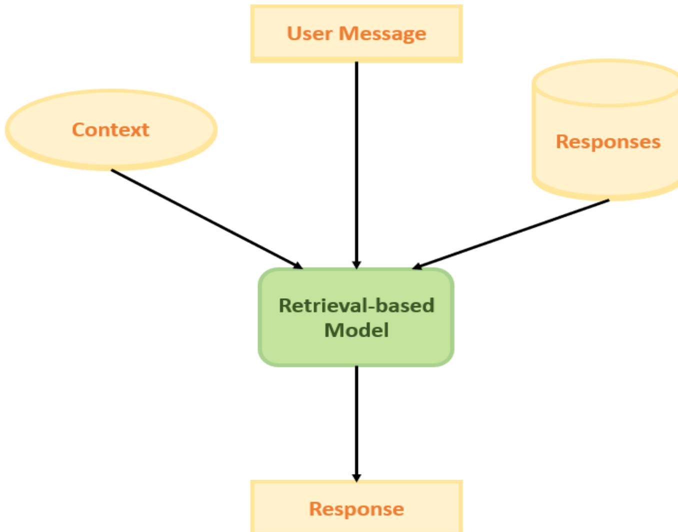

Fig 3. Retrieval based Model Chatbots Workflow

Tools for building Chatbots

There are multiple NLP based tools/platforms available in the market to build chatbots. Some of the well-known chatbot building tools are:

1. Dialogflow (api.ai)
2. LUIS
3. AmazonLex
4. IBM Watson
5. Wit.ai
6. PandoraBots
7. ChatFuel
8. Recast.ai

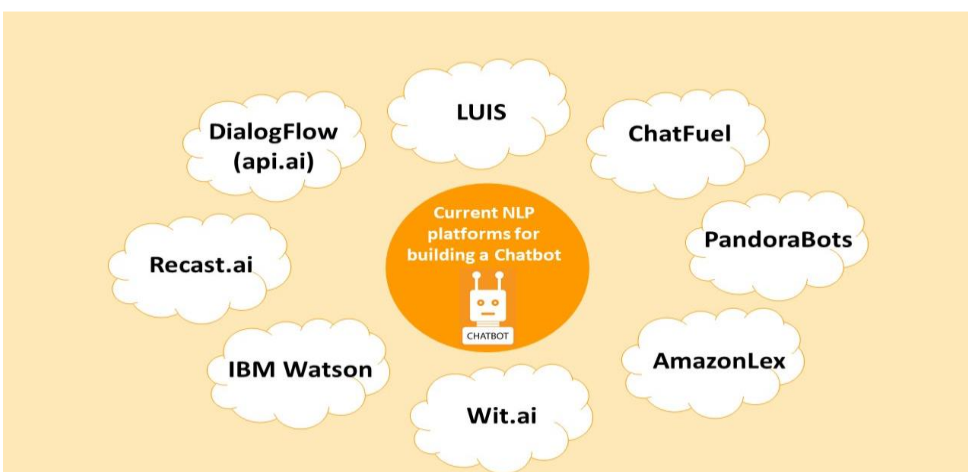

Fig 4. Tools for building Chatbots

All the above tools for building Chatbots are retrieval-based models.

Microsoft LUIS

Language Understanding (LUIS) enables your application to comprehend what a person needs in their own language. LUIS utilizes machine learning to enable engineers to construct applications that can receive user input in natural language and extract significant meaning from it. A customer application that chats with users, who can pass user input to a LUIS application and get pertinent, detailed, point by point information back.

What is a LUIS app?

A LUIS application is a domain-specific dialect display planned by you and custom fitted to your requirements. You can begin with a pre-built domain model, manufacture your own, or mix bits of a pre-built domain with your own custom data. A model begins with a rundown of general user intentions, for example, “Book a Flight” or “Book a Hotel.” Once the intentions are recognized, you supply precedent expressions called utterances for the intents. At that point you name the expressions with any specific details you need LUIS to pull out of the utterance. Pre-built domain models incorporate all these these pieces for you and is a great method to begin utilizing LUIS rapidly. After the model is designed, trained, and tested, it is prepared to receive and process utterances. The LUIS application gets the utterance as a HTTP request and reacts with extracted user intentions. Your client application sends the expression and gets LUIS’s assessment as a JSON object. Your client application would then be able to make proper move.

Key LUIS concepts

Intents: - An intent represents actions that the client needs to perform. The intent is a reason or objective communicated in a user’s input, for example, booking a flight, paying a bill, or finding a news article. You characterize and name intents that relate to these activities. A travel application may characterize a purpose named “Book a Flight”.

Utterances: - An utterance is a text input from the user that your application needs to get it. It might be a sentence, similar to “Book a ticket to India”, or a piece of a sentence, such as “Booking” or “India flight.” Utterances aren’t always well-formed, and there can be numerous expression varieties for a specific intent.

Entity:- An entity can be defined as detailed pieces of information that is important in the utterance. For instance, in the expression “Book a ticket to India”, “India” is a location. By perceiving and labeling the entities that are referenced in the user’s utterance, LUIS encourages you pick the explicit action to take to answer a client’s request.

Examples:

Intent	**Sample User Utterance**	**Entities**
Book a Flight	"Book a flight to **New York**?"	New York
Book a Hotel	"Book an **expensive** 5-star hotel with **Wi-Fi**?"	Expensive, Wi-Fi
Book a Flight	"Book a business **class ticket** from **Syracuse** to **California**"	Class, Ticket, Syracuse, California

How to customize your LUIS model?

Start your LUIS model with the intents your customer application can resolve. Intents are simply names, for example, "Book a Flight" or "Book a Hotel."

After an intent is recognized, you require test expressions that you need LUIS to guide to your intent, for example, "Purchase a ticket to Boston tomorrow." Then, label the parts of the utterance that are relevant to your application space as entities and set a type, for example, date or location.

For the most part, an intent is utilized to trigger an action and an entity is utilized as a parameter to execute an action.

For instance, a "Book a Flight" intent could trigger an API call to an external service for booking a plane ticket, which requires entities like the travel destination, date, and airline information. See Plan your application for examples and direction on how to choose intents and entities to reflect the functions and connections in an application.

How to identify entities?

Entity identification decides how effectively the end user finds the correct answer. LUIS provides different ways to distinguish and categorize entities.

Prebuilt Entities: LUIS has numerous prebuilt domain models including intents, utterances, and prebuilt entities. You can utilize the prebuilt entities without utilizing the intents and utterances of the prebuilt model. The prebuilt entities save you time.

Custom Entities: LUIS gives you different ways to recognize your very own custom entities including simple entities, composite entities, list entities, and hierarchical entities.

Phrases LUIS: Gives phrase lists, which additionally help recognize entities.

How to improve performance?

Once your application is ready to publish and real user utterances are entered, LUIS utilizes active learning to enhance distinguishing and identification capability. In the dynamic learning process, LUIS provides real expressions that it is relatively uncertain of for you to review. You can label them as indicated by intents and entities, retrain, and republish.

This iterative procedure has lot of advantages. LUIS recognizes what it is uncertain of, and your assistance leads the maximum improvement in framework execution. LUIS adapts quickly and takes minimum amount of your time and effort. LUIS is a functioning machine learning software taking care of business.

DIALOGFLOW

Dialogflow gives us a chance to assemble conversational interfaces over our products and services by giving a powerful natural language understanding (NLU) engine to process and understand natural language input.

When we utilize Dialogflow, we make specialists that can comprehend the immense and shifted subtleties of human language and make a translate that to standard and organized implying that our applications and services can understand.

Dialogflow uses intents, entities, actions with parameters, contexts, speech to text, and text to speech capabilities, along with machine learning that works silently and trains our model. Dialogflow has built-in knowledge on topics like casual talks, weather, and wisdom. It means we don't have to train the agent for these intents. Dialogflow returns the output as JSON data.

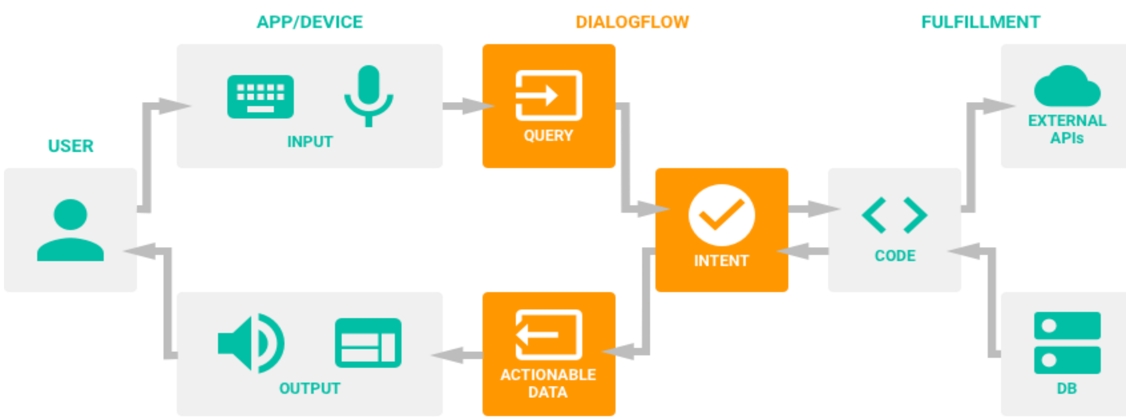

Fig 5. Working of Dialogflow tool

Important Dialogflow Concepts

Agents: - Agents are best described as NLU (natural language understanding) modules. These can be included in our app, product, or service and transform natural user requests into actionable data.

This transformation occurs when a user input matches one of the intents inside our agent. Intents are the predefined or developer-defined components of agents that process a user's request.
Agents can also be designed to manage a conversation flow in a specific way. This can be done with the help of contexts, intent priorities, slot filling, responsibilities, and fulfillment via webhook.

Intents: - In order to define how the conversation has to be, we provide intents in our agent that map user input to responses. In each intent, we define examples of user utterances which can trigger the matching intent, what need to be extracted from the utterance, and how to that particular respond.

Entity: - Entities are powerful tools used for extracting parameter values from natural language inputs. Any important data we want to get from a user's request will have a corresponding entity.
There are three types of entities: system (defined by Dialogflow), developer (defined by a developer), and user (built for each individual end-user in every request).

Automatic Annotation: - When we add examples to the User says section, they are annotated automatically. The system detects the correspondence between words (or phrases) and existing developer and system entities and highlights such words and phrases. It also automatically assigns a parameter name to each detected entity.

Action: - The action name is defined manually. It will be the trigger word for our app to perform a particular action.

Response: - This is just a response to what the user says. We can improve your agent's eloquence by adding several variations of the text response per intent. When the same intent has been triggered more than once, different text response variations will be unrepeatable until all options have been used. It'll help make our agent speech more human-like.

Fallback Intents: - Fallback intents are triggered if a user's input is not matched by any of the regular intents or enabled built-in small talk.

Making of RoverBot using LUIS and Dialogflow

Intents:

1. Bookflight: This intent allows the user to perform the action of booking a flight and allows them to specify some extra pieces of information like locations, amount etc.
2. Bookhotel: This intent allows the user to perform the action of booking a hotel and allows them to specify some extra pieces of information like amount, facilities etc.

Entities: Some entities like class, location, and flight ticket were created for Bookflight intent which helps LUIS and Dialogflow in understanding class of flight, Source and destination location respectively from the input text given by the user. Similarly, facility and map-sort entities were created for Bookhotel which helps LUIS and Dialogflow in understanding the facility user want from the input text of user. Some inbuilt entities like age, number, phone number, URL, email id etc. were used which improved the understanding level of LUIS and Dialogflow and made LUIS and Dialogflow capable of understanding a new text given from user efficiently.

Utterances:

Bookflight utterances:

1. Book a flight from Seattle to Cairo.
2. Book a ticket of business class on 31st March 2019.
3. Book the cheapest flight from www.cheapoair.com and send me a confirmation mail at abs@gmail.com.

Bookhotel utterances:

1. Book a cheapest 5 star hotel.
2. Book a best 7 star hotel nearest to the airport in Seattle.
3. Book a cheapest 3 star hotel with Wi-Fi and pool facility.

Features:

Extra features were defined in LUIS in order to make it more efficient in understanding a new test from user. We created cityname in phrase list where cities like San Diego and New Delhi were defined so that LUIS take San Diego as one complete location.

Screenshots of LUIS

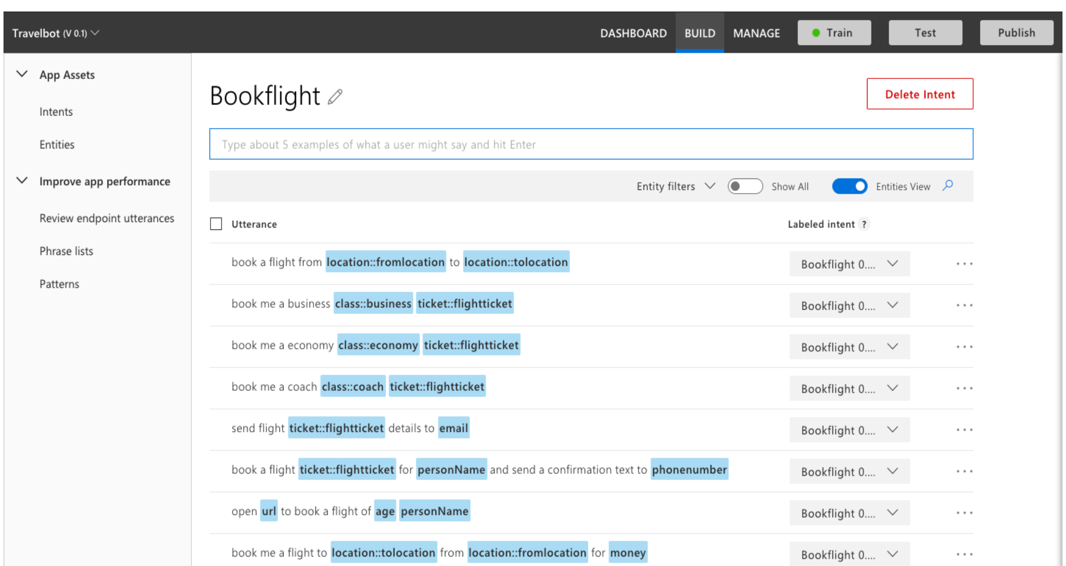

Fig 6. Bookflight Intent with Utterances

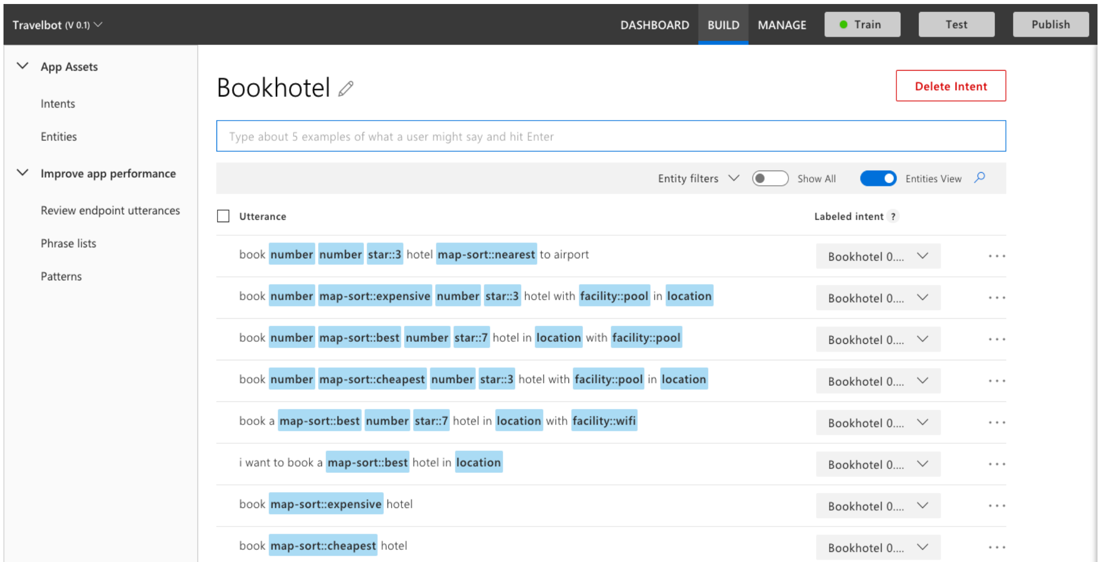

Fig 7. Bookhotel Intent with Utterances

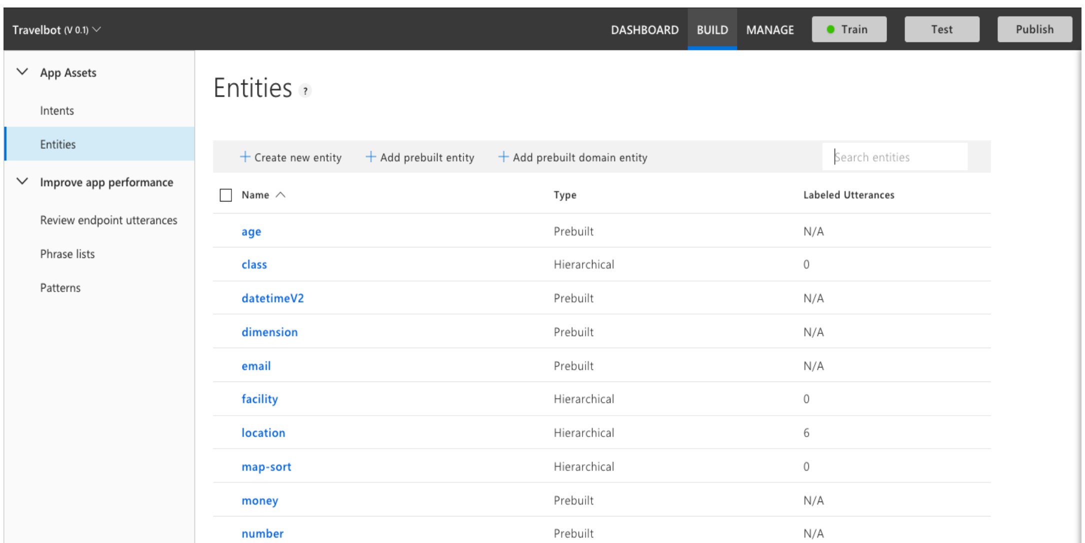

Fig 8. Entities for Bookflight and Bokkhotel Intent

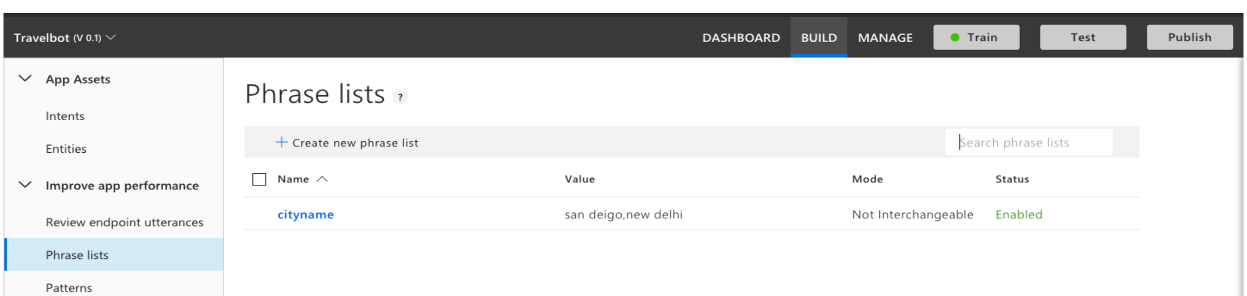

Fig 9. Phrase Lists (creating city name)

```
{
  "query": " book a business class ticket from san deigo to new delhi from www.cheapoair.com for $200 and send the confirmation mail at abc@gmail.com and a text at (123)-567-8904",
  "topScoringIntent": {
    "intent": "Bookflight",
    "score": 0.9982713
  },
  "entities": [
    {
      "entity": "class",
      "type": "class::business",
      "startIndex": 17,
      "endIndex": 21,
      "score": 0.719262838
    },
    {
      "entity": "san deigo",
      "type": "location::fromlocation",
      "startIndex": 35,
      "endIndex": 43,
      "score": 0.8606499
    },
    {
      "entity": "new delhi",
      "type": "location::tolocation",
      "startIndex": 48,
      "endIndex": 56,
      "score": 0.9246054
    },
    {
      "entity": "ticket",
      "type": "ticket::flightticket",
      "startIndex": 23,
      "endIndex": 28,
      "score": 0.9860398
    },
    {
      "entity": "abc@gmail.com",
      "type": "builtin.email",
      "startIndex": 124,
      "endIndex": 136,
      "resolution": {
        "value": "abc@gmail.com"
      }
    },
    {
      "entity": "$200",
      "type": "builtin.currency",
      "startIndex": 85,
      "endIndex": 88,
      "resolution": {
        "unit": "Dollar",
        "value": "200"
      }
    },
    {
      "entity": "(123)-567-8904",
      "type": "builtin.phonenumber",
      "startIndex": 152,
      "endIndex": 165,
      "resolution": {
        "score": "1",
        "value": "(123)-567-8904"
      }
    },
    {
      "entity": "www.cheapoair.com",
      "type": "builtin.url",
      "startIndex": 63,
      "endIndex": 79,
      "resolution": {
        "value": "www.cheapoair.com"
      }
    }
  ],
  "sentimentAnalysis": {
    "label": "neutral",
    "score": 0.5
  }
}
```


Fig 10. JSON output for a user text input (Bookflight)


```
{
  "query": "book a best 5 star hotel with wifi in Seattle",
  "topScoringIntent": {
    "intent": "Bookhotel",
    "score": 0.9667664
  },
  "entities": [
    {
      "entity": "wifi",
      "type": "facility::wifi",
      "startIndex": 30,
      "endIndex": 33,
      "score": 0.40487197
    },
    {
      "entity": "seattle",
      "type": "location",
      "startIndex": 38,
      "endIndex": 44,
      "score": 0.867347836
    },
    {
      "entity": "best",
      "type": "map-sort::best",
      "startIndex": 7,
      "endIndex": 10,
      "score": 0.8488126
    },
    {
      "entity": "star",
      "type": "star::5",
      "startIndex": 14,
      "endIndex": 17,
      "score": 0.5647553
    },
    {
      "entity": "5",
      "type": "builtin.number",
      "startIndex": 12,
      "endIndex": 12,
      "resolution": {
        "subtype": "integer",
        "value": "5"
      }
    }
  ],
  "sentimentAnalysis": {
    "label": "positive",
    "score": 0.9125426
  }
}
```


Fig 11. JSON output for a user text input (Bookhotel)

Screenshots of Dialogflow

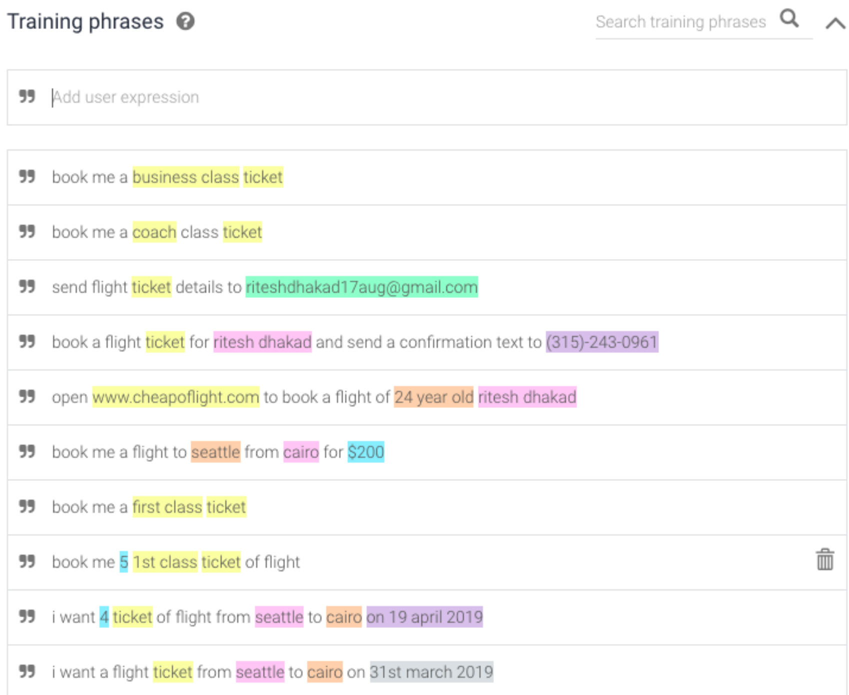

Fig 12. flightbook Intent with Training Phrases

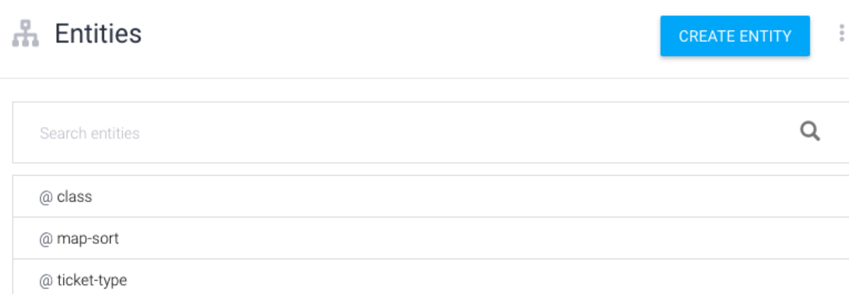

Fig 13. flightbook Entities

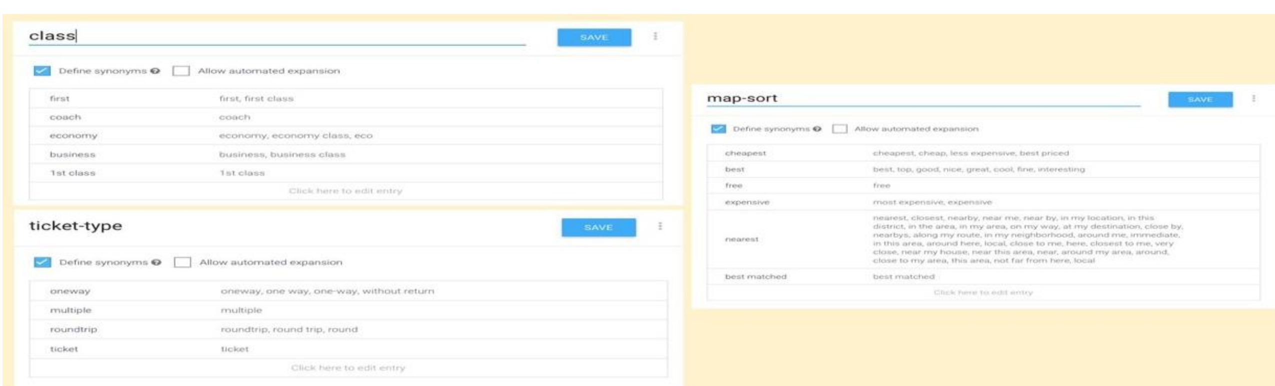

Fig 14. Class, map-sort and ticket-type Entities

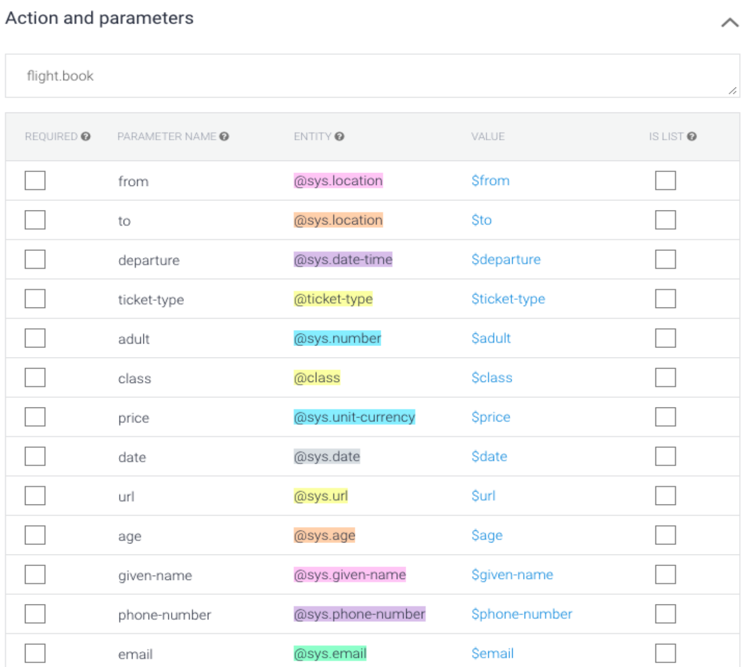

Fig 15. Action and parameters

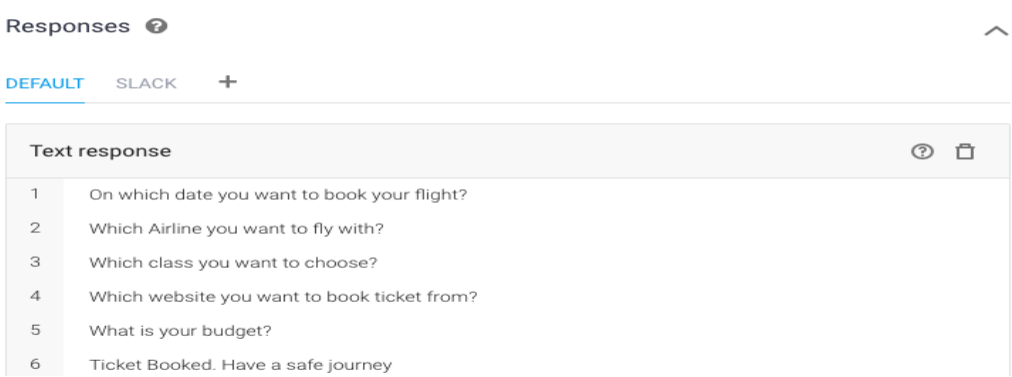

Fig 16. Responses (Chatscript)

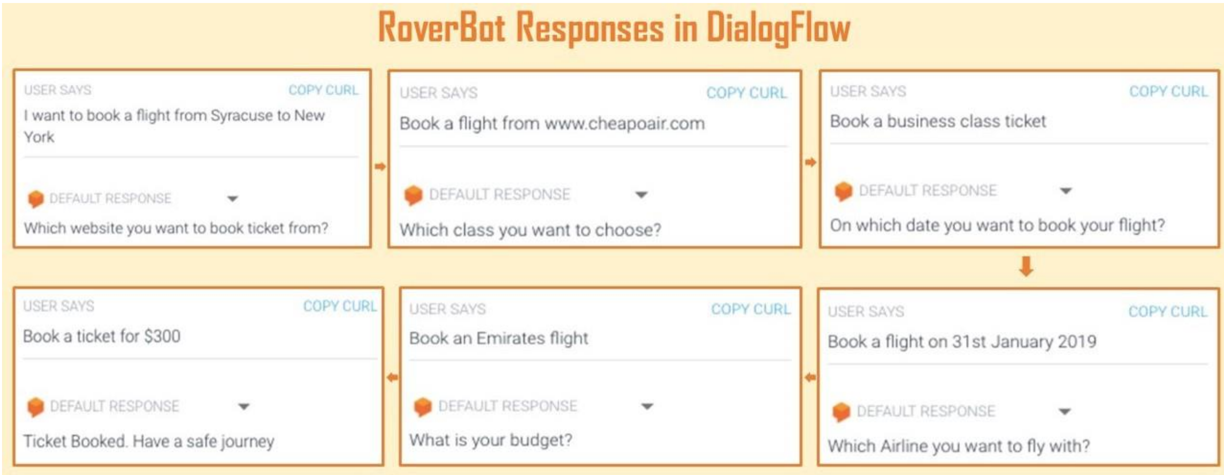

Fig 17. Conversation with RoverBot in Dialgflow

Comparing LUIS and Dialogflow based on Built and Functionality

LUIS	Dialogflow (API.AI)
Microsoft LUIS (Language Understanding Intelligent Service) uses - Intent: Represents action the user wants to perform Utterances: User input Entity: Identifies & extracts useful information from user utterances Features: Features help LUIS recognize both intents and entities by providing hints to LUIS that certain words and phrases are a part of a category or follow a pattern	**Acquired by Google, Dialogflow uses - Agent: They are the NLU modules Intent: Represents action the user wants to perform Entities: Identifies & extracts useful information from user input Context: These are basically training phrases Responses: Responses are speech to text and text to speech capabilities**
The NLG unit for LUIS uses AIML (Artificial Intelligence Markup Language)	**The NLG unit for LUIS uses ChatScript**
Microsoft LUIS understands the intent, and provides detailed metrics on how it matches the intents and entities, using which we can also measure true positives, true negatives, false positives and false negatives. However, building a conversational bot with LUIS required us to use UWP (Universal Windows Platform) apps	**Microsoft LUIS understands the intent, and provides detailed metrics on how it matches the intents and entities, using which we can also measure true positives, true negatives, false positives and false negatives. However, building a conversational bot with LUIS required us to use UWP (Universal Windows Platform) apps**
Has an in-built feature to perform Sentiment Analysis	**Does not have an in-built feature for Sentiment Analysis**

Evaluating RoverBot performance on LUIS and Dialogflow platforms

There are different evaluation metrics available for comparing a chatbot performance. However, since we were looking for a language-understanding evaluation metric to compare our chatbot performance, we chose F-Score as our main evaluation metric.

F-Score is being widely used as benchmark in the current market for evaluating the intent classification of a chatbot. Intent/Entity classification is an important aspect of the Natural Language Understanding (NLU) system and is thus an important evaluation metric to evaluate a chatbot.

'F-Score' is basically the measure of test's accuracy and uses both Precision rate and Recall rate.

F-Score can be calculated using the following formula:

$$\text{F-Score} = 2 \times \frac{\text{Precision x Recall}}{\text{Precision + Recall}}$$

To use this evaluation metric, we first need to understand what precision and recall rates mean and what do they signify.

Precision Rate: When given a certain set of input messages to the chatbot, precision signifies what proportion of positive identifications of intents and entities by the chatbot were actually correct.

Recall Rate: When given a certain set of input messages to the chatbot, recall explains what proportion of actual positives of intents and entities were identified correctly by the chatbot.

These can be better explained by looking at the below formulas:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \qquad \text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

In order to understand this better, we need to understand the terms used in the above formulas.

True Positive: Cases where the chatbot has correctly identified the intent/entity from the given input message.

False Positive: Cases where the chatbot has wrongly classified input word as intent/entity.

False Negative: Cases where the chatbot has failed to identify the intent/entity from a given input message.

The below table shows the show to classify a given test case as true positive, false positive, false negative.

Under the 'Actual' section:

'Positive' refers to the word from input message being an intent/entity.

'Negative' refers to the word from input message not being an intent/entity.

Under the 'Predicted' section:

'Positive' refers to the word from input message being classified as an intent/entity by the chatbot.

'Negative' refers to the word from input message not being classified an intent/entity by the chatbot.

		Actual	
		Positive	Negative
Predicted	Positive	**True Positive**	**False Positive**
	Negative	**False Negative**	**True Negative**

Fig 18. Confusion Matrix

Calculating F-Score for RoverBot on LUIS and Dialogflow platforms:

We first trained our RoverBot on LUIS platform and Dialogflow platform with 25 sentences each. We then further validated our RoverBot's performance on LUIS and Dialogflow with another 16 sentences each.

Finally, we gave the below 6 sentences to RoverBot as testing data to evaluate its performance on each of the platforms and obtain their F-Scores. We tried to incorporate different types of sentences as our testing data so as to test the RoverBot capabilities to a maximum extent. Some of testing phrases were very different from what the bot was trained for and yet it gave a decent performance in correctly identifying the intents and entities.

Testing Phrases given to RoverBot on LUIS platform and Dialogflow platform:

1. Book a flight to Hyderabad on 16 Dec
2. I would like a flight from Syracuse to New Delhi tomorrow
3. Show me the cheapest flights to Seattle between 15 Dec to 25 Dec
4. Travel options to Cairo from www.studentuniverse.com in business class for two adults
5. Book flight for Ritesh Dhakad of 24 years and send a confirmation mail at rdhakad@syr.edu
6. Show me options for five star hotel in San Diego during summer

Below table shows the list of intents and entities and their classification by RoverBot, shown in terms of: **True Positive (True +), False Negative (False -)** and **False Positive (False +)**

The table also shows the overall Precision and Recall rates and the obtained F-Scores for LUIS and Dialogflow. These columns were calculated using the formulas mentioned in the above section.

	Entity Type/Intent	Type	True +	False -	False +	Precision	Recall	F Score
LUIS RoverBot	BookFlight	Intent	5	0	0	1	1	1
	BookHotel	Intent	1	0	0	1	1	1
	FromLocation	Entity	1	0	0	1	1	1
	ToLocation	Entity	4	1	0	1	0.8	0.888889
	DateTime	Entity	6	0	1	0.857143	1	0.923077
	PersonName	Entity	1	0	0	1	1	1
	Class	Entity	0	1	0	0	0	0
	Number	Entity	1	0	5	0.166667	1	0.285714
	Age	Entity	1	0	0	1	1	1
	Email	Entity	1	0	0	1	1	1
	URL	Entity	1	0	0	1	1	1
	StarHotel	Entity	1	0	0	1	1	1
	Total		**23**	**2**	**6**	**0.79**	**0.92**	**0.85**
Dialogflow RoverBot	BookFlight	Intent	5	0	0	**1**	1	1
	BookHotel	Intent	1	0	0	**1**	1	1
	FromLocation	Entity	1	0	0	**1**	1	1
	ToLocation	Entity	4	1	0	**1**	0.8	0.888889
	DateTime	Entity	2	4	0	**1**	0.333333	0.5
	PersonName	Entity	0	1	1	**0**	0	0
	Class	Entity	1	0	0	**1**	1	1
	Number	Entity	0	1	0	**0**	0	0
	Age	Entity	1	0	0	**1**	1	1
	Email	Entity	1	0	0	**1**	1	1
	URL	Entity	1	0	0	**1**	1	1
	StarHotel	Entity	1	0	0	**1**	1	1
	Total		**18**	**7**	**1**	**0.95**	**0.72**	**0.82**

Inference and Conclusion

As seen from the above table, RoverBot in Dialogflow has higher precision (95%) than the RoverBot built on LUIS (79%). However, the recall rate (92%) of the RoverBot in LUIS is higher than the recall rate (72%) of RoverBot in Dialogflow.

F-Score gives equal weightage to precision rate and recall rate and is our main evaluating metric for the performance of RoverBot. We can see that the F-Score of RoverBot using LUIS is 85%, whereas using Dialogflow it is 82%.

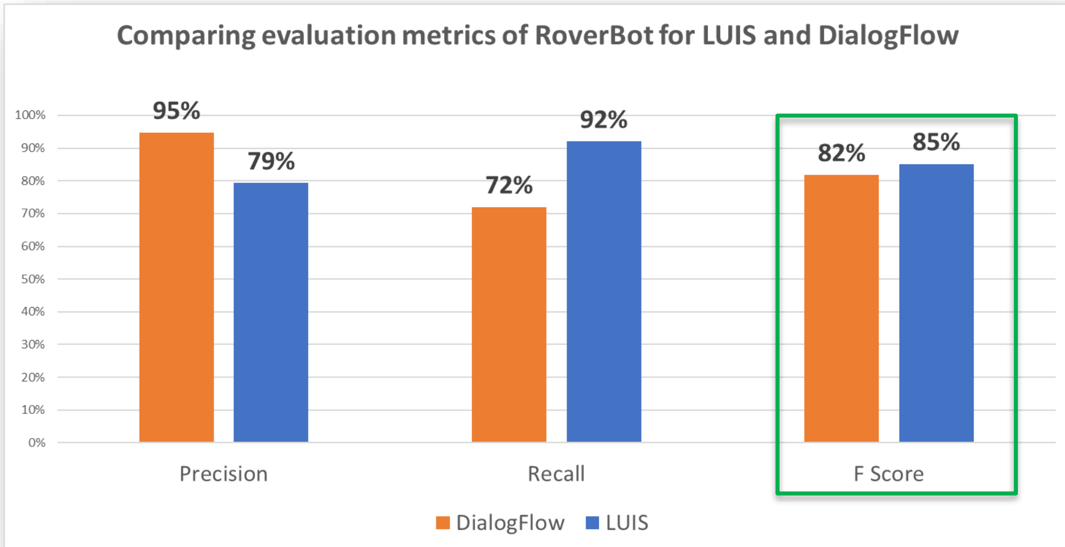

Fig 19. Comparing Evaluation Metrics of RoverBot for LUIS and Dialogflow

As seen above, the F-Score for RoverBot built using LUIS platform is higher than the F-Score for RoverBot built using Dialogflow platform.
Hence, we can conclude that for our RoverBot, LUIS platform provides a better intent classification performance than the Dialogflow.

References

https://www.nbcnews.com/tech/innovation/what-are-chatbots-why-does-big-tech-love-them-so-n572201
https://botpress.io/learn/what-and-why/
https://dialogflow.com/docs
https://medium.com/@surmenok/natural-language-pipeline-for-chatbots-897bda41482
https://dzone.com/articles/conversational-interfaces-powered-by-ai-dialogflow
https://chatbotslife.com/nlp-nlu-nlg-and-how-chatbots-work-dd7861dfc9df
https://hackernoon.com/an-analysis-of-the-best-nlp-tool-to-build-a-conversational-bot-18c4de0016c
https://www.expertsystem.com/importance-of-nlp/
https://medium.com/@surmenok/natural-language-pipeline-for-chatbots-897bda41482
https://dzone.com/articles/luis-language-understanding-intelligent-service
https://www.mytravelresearch.com/chatbots-and-the-travel-industry/
https://dialogflow.com/docs
https://www.behance.net/gallery/37453869/Designing-a-Chatbot-UX-Design-Process-Case-Study
https://studio.carnegiemuseums.org/literature-review-chatbots-conversational-experiences-566de218f92a
https://medium.com/botfuel/benchmarking-intent-classification-services-june-2018-eb8684a1e55f