Performance Comparison of Rasa and Amazon LEX Bots

Greeshma Damera
MS Computer Science
University of Houston
Houston,USA
gdamera@cougarnet.uh.edu

Krithikashree Lakshminarayanan
MS Computer Science
University of Houston
Houston,USA
krithikasln99@gmail.com

Rahul Tandon
MS Computer Science
University of Houston
Houston,USA
rahul.tandon98@outlook.com

***Abstract*— Chabot's are intelligent conversational agents that understands and responds in human understandable language. They enhance the user experience and can be trained to automate tedious tasks. With the help of many powerful, open source tools like RASA, Amazon LEX we are now able to build and train these intelligent agents easily. Through this project we try to analyze and evaluate the performance, verify the quality issues and attributes related to development and implementation. Finally we examine the approach we have used to evaluate the performance of the bots.**

Keywords—Chabot, contextual assistant, RASA, Amazon LEX, performance evaluation

I. INTRODUCTION

Current advances in Natural Language Processing (NLP) and enhanced computing power have facilitated the rapid improvement and deployment of chatbots in various industries.[1] The purpose of development of chatbot can range from solving generic customer service issues to building it to resolve specific operation issues

In our project we are trying to build a Contextual AI Assistants go far beyond simple FAQ interactions, they consider the context of what has been said before, which is a very important part of every natural conversation. With this smart chatbot will send an emergency pop-up to the nearest help center – be it police station, fire station or hospital in just few quick texts, with your location for quick support. We are trying to build the bot using RASA and Amazon LEX, with these bots we Analyze these modern chatbots based on their functionality, architecture, and categorization (as well as their evaluative aspects).[1]

II. DESCRIPTION

A. Rasa – Contextual AI assistant

We will be using RASA, an open-source ML tool for developers. We will use RASA's two components: Natural Language Understanding (NLU) and Dialogue Model Component (Core).

NLU is an open-source NLP tool for intent classification and entity extractions. It is the ear of the assistant which is helping to understand what is being said, it takes user input in an unstructured human language and extract structured data in a form of intents and entities - intents is labels that represent the overall goal of user message, like - "Good Morning! My name is Rahul" can have an intent - "greet" while entities are pieces of information that an assistant may need in a certain context or something to remember throughout the conversation to keep the interaction natural, so name "Rahul" here is an entity. All this is achieved by training a named entity recognition model.

Dialogue Model Component (Core) is a framework for ML-based, contextual decision making. It is the brain which makes decision on how an assistant should respond based on the state of the conversation as well as the context. Core learns by observing patterns from example conversational data between the user and an assistant which is called stories.

NLU enables your assistant to understand what the user says, Core component predicts how an assistant should respond and Rasa X enables your assistant to continue learning from a real conversational data. All these Rasa component work together to make a Contextual AI Assistant.[2]

We will use open-source libraries of NLP and ML like SpaCy and scikit-learn with default parameters optimized for most common NLP tasks. We will need labelled user data in JSON format for training our NLU or interpreter part. Sample can be seen below.

```
{
  "text": " I need a General Physician in
70254"
  "intent": "find_doctor",
  "entities": [
    {
      "doctor_type": "General Physician",
      "zipcode": 70254
    }
  ]
}
```

While it is always better to have more data to train our ML models but in case, we don't have it we will use free services such as Chatito or Tracy to synthetically generate one to suite our use case. To train a dialogue model which in our case is Rasa Core python library we first create a stories.md file. Sample file would look like –

```
## sample story
* greet
   - utter_greet
* goodbye
   - utter_goodbye
```

We then complete dialogue model training. To create stories suitable to our use case, we will apply online training recommended by Rasa Core. Once we have enough sample conversations to be suitable to our use case we will repeat the step to train our dialogue management model on the newly updated stories.md file. [3]

B. Amazon Lex

We would also be building a conversational assistant using Amazon Lex. Amazon lex is an Artificial intelligence service which is fully managed with advanced natural language models that can be used to build conversational assistants. It provides us with all the tools to solve challenging deep learning problems, like language understanding with one easy to use service.

It provides natural language understanding technologies for creating a Language Understanding system. Lex learns from various ways users express their intent on the basis of sample utterances provided by the developer. The language understanding system takes natural language speech and text as inputs, understands the intent from the input and fullfills the user's intent by invoking the appropriate response.

Lex naturally supports context management so we can manage the context as the conversation develops without having to build a custom code.

Assistants built using Amazon lex provide an ability for multi turn conversations, that is, once the intent of the user has been identified, the user will be automatically prompted with the information that is required for the intent fulfilment. For Instance, in our project, If the intent of the user is a Fire Emergency, he/she would be prompted for providing location, contact number. We can achieve this by listing the slots, that we are expecting from the users, and their corresponding prompts.

We write integration code and AWS Lambda runs the code when required to send / retrieve data from a database or any external system. We would use Amazon SNS for notifying receivers.

We can deploy the assistant in just one click without having to worry about the infrastructure and the hardware. Additionally, connect with other services of AWS for executing business logic, monitoring the performance and querying data.

C. Functionality (RASA and LEX)

Our bot will be communicating with the user, based on the context and the conversation it determines their emergency. The bot will send request to the corresponding department along with the user details and GPS location for quick help. We will use open-source GPS tracking API integrated with our bots to send users location. Since we cannot send the request now to the departments, so we will create 3 web apps (For Police, Hospital and Fire station).

D. Evaluation Metrics

Load Testing- We are trying to compare the efficiency of RASA and LEX bots based on the load it can handle at a given time. Our range of request lies between 10k to 1000k request and we are planning to extend it from 1000k to 100M.

AI specific testing- We will test our bots response to determine the correct situation and its accuracy rate on action taken. We are planning to keep our accuracy rate to be 75%.

CORE testing- Engagement rate: How many talked to it? Lead capture rate: What were the outcomes of those chats?

E. What we learnt

We learned how to build bots using powerful open source tools like RASA and LEX. We learned about architecture and internal NLU processes inside these AI driven bots. we did extensive research about cloud services and learned about aws services like s3 buckets,EC2 instance ,LEX , lambda function and open source ML frameworks. We learned about scalability and load testing using Docker.

F. Experiments:

After deploying the assistant for a wider audience, there is a possibility that it will see messages it hasn't seen before in the training data. To simulate this, we kept aside testing data. After making significant changes to the data set, to evaluate and train our NLU model, we split the data in folds and trained the NLU model extensively. Data was split as Training/Testing like 80%/20%, this went on in multiple sets of folds – 70%/30, 60%/40%, 50%/50% till 20%/80%. [Split can be seen in figure 2A] We then did the full NLU evaluation by doing cross-validation with different folds of data. We also prepared a special test dataset using examples from real conversation. We realized that there can be more than one way to declare an intent. So, we managed to prepare a test dataset with sentences that intent the same but are conveyed differently, just to keep the conversation real and provide plausible data to improve our NLU model. [Dataset can be seen in figure 2B].

The intents created for the bot on RASA and on Amazon Lex were consistent, they were trained on the same dataset. We evaluated the performance of both the bots based on the results from NLU training. We compared the performance based on the factors like model tweaking, adoption, Intent entity model which are discussed below.

G. Results

The result are represented below as – confusion matrix, hits-misses graphs and intent-entity classification report. All these representation were generated both for RASA bot and Amazon Lex for comparison.

Confusion Matrix: The confusion matrix [Fig 3A and Fig 3B] for RASA bot was built using the command rasa test nlu – cross-validation while Amazon Lex's confusion matrix was manually generated with manually testing of dataset. The confusion matrixes clearly shows that on small amount of training/testing data cross-validation RASA bot clearly shows better labelled-predicted intent prediction with lesser to none mismatch on comparison to Amazon Lex.

Hits and Misses Graph: The framework generated graphs [Figure 4A –RASA bot and 4B –Amazon Lex] indicates the performance of the bots on the dataset mentioned above. This is another representation on how the bot performance improves as the bots are tested on different folds of dataset and on testing of special dataset created for enhancement of NLU model. The real comparison appears to be on lesser number of samples where misses are few to negligible for RASA bot while there clearly are more misses encountered for Amazon Lex for same lesser sample of data.

Intent Entity Classification Report: The classification report [Fig 5A and Fig 5B] could only be generated for RASA bot for the complete dataset. However, there was no provision in Amazon Lex to generate such a report as classification report could only be generated for per test sample. However as a

result we realized this to be one of the limitation of Amazon Lex bots over RASA generated bots.

H. Comparison

Intent/Entity Model: Rasa allows to add our own custom model into the pipeline for any task. Lex is a solid platform with good models and pre-trained entities but it does not support custom models.

Model Tweaking: With Rasa we can evaluate the model and configure it based on our needs. However, with Lex, models are black box, we don't have any option of improving or evaluating the models.

Language Support: Rasa provides language support using pipeline concept, we can add our own component to support multiple languages. Currently Lex only supports only English and US Spanish, is not available in most places in the world.

Adoption: Rasa has been widely adopted in a diverse range of industries to creation of their AI Assistants. Amazon Lex is not even present in most of the countries at the moment.

I. Conclusion/Verdict:

Lex help you building a chat bot easily and it does not require any knowledge of AI. But if you want to have a wide range of customizations and more control over how you build your chatbot, Rasa would be the better option.

J. Figures and Tables

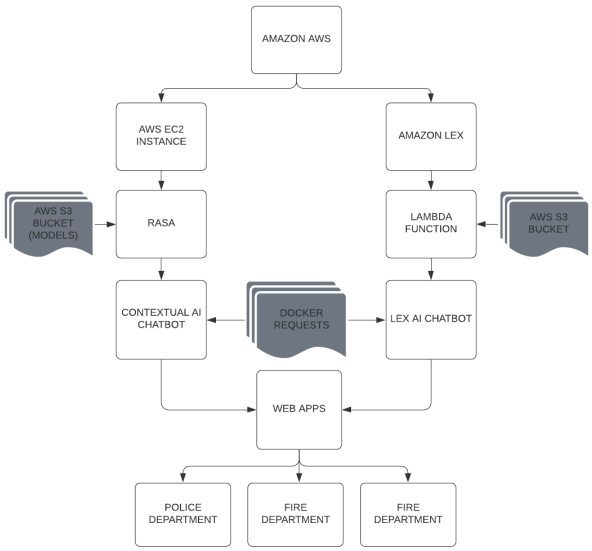


Fig. 1 Process flow diagram

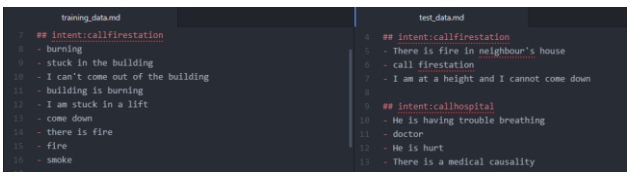
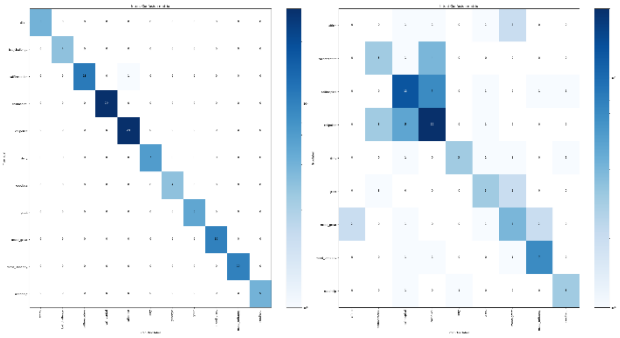


Fig. 2A Training/Testing dataset

```
Policestation:
1. Someone is following me
2. I am being followed by someone
3. I feel there is someone following
4. A man is following me
5. A woman is following me
```

Fig. 2B Special Training/Testing dataset



Confusion Matrix - Fig 3A RASA Fig 3B Amazon Lex

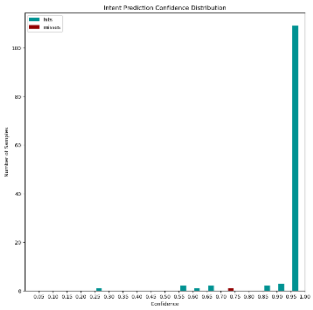
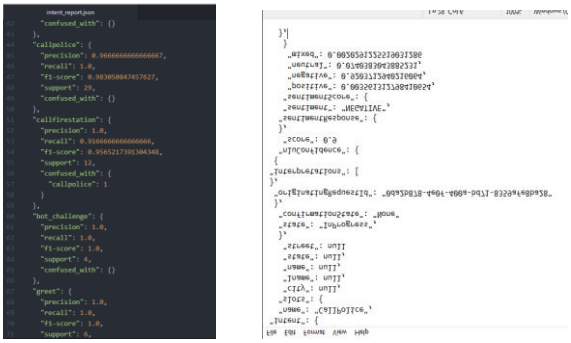


Fig. 4A RASA Hits and Misses Graphs

Fig. 4B Amazon Lex Hits and Misses Graphs



Entity-Intent Classification Report –

Fig. 5A RASA Fig. 5B Amazon Lex

III. ROLES

Rahul, Krithika and Greeshma will work on dataset collection ,documentation and testing. Greeshma will be working on training the model in both the bots. Rahul will be developing the API for action required from the user end. Krithika will be working on development of API for the response and actions from receivers end. All three of us will actively participate in quality assessment and load testing.

ACKNOWLEDGMENT

We thank our professor Dr. Weidong Shi for approving and supporting our project. We would like to express our special thanks and gratitude to our TA for his guidance and support throughout the development of the project. Finally we thank University of Houston for providing us this opportunity.We also thank the anonymous references and suggestions used.

REFERENCES

[1] Wari Maroengsit ,Thanarath Piyakulpinyo ,Pimwadee Chaovalit ,Thanaruk Theeramunkong , “A survey on evaluation odel on chatbots,” 2019 7th International Conference (references)

[2] By ©2020, Rasa Technologies

[3] By © Copyright 2020 Open Source Libs

[4] By © 2021, Amazon Web Services, Inc. or its affiliate