NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR
Cachar, Silchar

B.Tech. VIIth Semester
Artificial Intelligence, CS-401

Subhojit Ghimire, 1912160
Computer Science and Engineering, CSE-B

ABSTRACT:

Be it support bots like Amazon/Paypal Support Assistant or more smart virtual personal assistant like Google Assistant or Alexa, we all have used an automated chatbot system in one way or the other. These AI chat bots have a way of not only giving a helpful response to the questions asked, but to perform some actions like booking a movie ticket, flight ticket, making food orders etc. These days, it is easier and efficient to get tasks done through Intelligent Bots than it is through humans. RASA is one such open-source implementation that can interact with the database, API, conversational flow, interactive learning with reinforcement Neural Network. The sole purpose of RASA is to provide the best of conversational AIs for customer experience in today's digital globe. In this project, I will be implementing a simple AI Chatbot using RASA Framework and challenge it with my regional language– Nepali.

STEPS INVOLVED:

1. Firstly, Anaconda Distribution was installed on the machine as Anaconda offers easiest way to perform Python and Machine Learning.
2. Anaconda Prompt was started and the following line of command was run to set-up rasa in python v.3.6 environment in conda:
   > conda create -n rasa python=3.6
3. In the same prompt, the following line of command was run to install rasa dependencies:
4. After Rasa was successfully installed on the machine, the following line of command was executed to set-up a basic rasa chatbot and carry on a basic conversation with it:
   > rasa init
5. In the folder the basic rasa project was initiated, the following files were opened and made changes to:
   a. domain.yml: Here, new responses can be created as an action from the bot. I added and modified the responses for greeting user; asking wellbeing of the user; to cheer up the user if the user's feeling down; ask the user if, as a bot, was I helpful in assisting the user; be happy for the user if the user's happy; utter goodbye; prove, as a bot, that I am indeed a bot; ask user if they want to see something fun; share a prank video if user is looking for something for some fun; utter gladness if, as a bot, I was able to help the user.
   b. nlu.yml: This is the dataset area to record some ways the user can bend the question or chat. The program will recognise some keywords and reply accordingly, so, this area stores synonyms of the same word, various ways the same question can be asked in, various ways users will choose to reply and more. Basically, this yml file trains confidence of the question or emotion recognition. Some basic intents stored here are: greeting, wellbeing, goodbye message, affirmation, deny, good mood, sad mood, prank and bot challenge
   c. rules.yml: This section stores some general situations, like how to react should that specific situation arise.
   d. stories.yml: Unlike rules.yml, stories.yml stores elaborate situations and trains a bot to follow one of many story paths and even bend the conversation in another direction should the need arise.
6. Run the following command line to train the bot according to the newly made changes:
   > rasa train
7. Run the following command line to start chatting with the chatbot:
   > rasa shell