DEVELOPMENT OF A CONVERSATIONAL CHATBOT FOR BANKING SYSTEM USING ARTIFICIAL INTELLIGENCE AND NATURAL LANGUAGE PROCESSING

Sky-bits Technologies Private Limited, Kolkata, West Bengal

20th May, 2019 – 19th July, 2019

Soumya Sen Debomit Dey Akanksha Ghosh

Computer Science and Technology

IIEST, Shibpur

Introduction

A chatbot is an artificial intelligence (AI) software that can simulate a conversation (or a chat) with a user in natural language through messaging applications, websites, mobile apps or through the telephone. A chatbot is often described as one of the most advanced and promising expressions of interaction between humans and machines. However, from a technological point of view, a chatbot only represents the natural evolution of a Question Answering system leveraging Natural Language Processing (NLP). Formulating responses to questions in natural language is one of the most typical examples of Natural Language Processing applied in various enterprises' end-use applications.

Here we created a conversational chatbot on Banking system.

Training and Test Dataset

We have taken 6 topics of Query related to Banking System –

1. Account Balance Enquiry
2. Account Money Transfer
3. Account Transaction Details
4. Account General Query
5. Card Related Query
6. Loan Related Query

Training Datasets: For each topic, we created a dataset. The datasets are consisted of all possible queries including the various alternative ways of writing the same queries. We have taken around 90 types of question and around 500 queries related to all 6 sections to train the model. We have created a database where we put all the training datasets in different tables.

Test Dataset: We have taken 8 to 10 questions from all 6 sections to create the test dataset.

Train The Model

After creating the training datasets, we used an online platform Google's dialogue-flow to train the model. Based on the topics on which we created the queries, we trained the model by creating the intent and entities for each topic. We trained the model using queries that were part of our training data. We created 6 intent based on each training dataset-

1. account_balance_enquiry
2. account_general_query
3. account_money_transfer
4. account_transaction_details
5. card
6. loan

After training the model we exported the zip file of trained model from dialogue-flow and extracted the files in our local pc.

Installation of Rasa-Nlu-Traniner

Python Package Installation: To run the trained model we have to install a offline nlu-trainer i.e Rasa nlu trainer. To install Rasa we need some python packages to be installed in our pc. We have created a text file 'Requirements.txt' where we included all the necessary python packages with their required version. To install all the packages follow the steps:

1. Open command prompt normal mode in your windows.
2. Go to the folder where the file is saved. (cd location)
3. Run the command -> ***pip install -r requirements.txt***
4. After successful installation close the command prompt.

Spacy English Language Download: Now, we have installed all the required packages. We have to install spacy English language on which we will work. To install spacy English language, follow the steps:

1. Open command prompt again as administrator mode.
2. Run the command-> ***python -m spacy download –en***
3. After successful installation close the command prompt.

Nodejs Installation: To install the Rasa-Nlu-trainer we need to install Nodejs software in our pc. Download link: https://nodejs.org/en/ . Download the Nodejs software, install it and add path to the environmental variables.

<u>Rasa-Nlu-Trainer Installation</u>: Now we have to install Rasa-Nlu-Trainer. For that follow the steps:

1. Open command prompt in normal mode
2. Run the command-> ***npm i -g rasa-nlu-trainer***
3. After successful installation, Rasa-Nlu-Trainer is installed in the pc locally.

- ***<u>Suggestion:</u>*** ***Install Anaconda for smooth installation of Rasa-Nlu-Trainer***

<u>Training Queries with Answers</u>

We have created a file nlu_model.py. To create the model in Rasa, run the nlu_model.py file. Now we have the chatbot model offline. We have created database using python SQL module Sqlite3 and inside the database we created 6 tables which has the same name as intents where we store the queries and their answers. In the first column of a row, we stored the number of question of same type, in second column all the queries of same type separated by '||' and in third column the answer. Run the following commands to create the database:

1. ***python account_balance_enquiry.py***
2. ***python account_general_query.py***
3. ***python account_money_transfer.py***
4. ***python account_transaction_details.py***
5. ***python card.py***
6. ***python loan.py***

Now we have 6 tables in the database. We have stored all the details (account number, name, balance, last deposit, last withdrawal) of account holders in a table named as database. To create the table run ***python database.py*** .

Vectorize the Datasets

To check similarity between the training query and the user's query we use tfidf vectorizier to vectorize all the tables from the database. After vectorize each table, we get a vectorizor and we store the vectorize values in text file named as same the table name.

Answer of the Queries

Now, Run ***chatbot.py*** file to get the final layout. There are 3 situation based on the possibility whether a chatbot can answer the query or not –

1. Chatbot can directly answer the query of user.
2. Chatbot can answer the query of user by fetching the details from database table.
3. Chatbot can not give answer.

When a query comes, we first extract intent and entity of that query. When we get the intent, we vectorize the query using the trained vectorizer and compared that query with the vectorizer value of that intent-vector file. For similarity check, we used Jackard similarity method. With the most similar query, we check 3 conditions of the answer column:

1. If answer is ‘fetch’, then we have to fetch details from database table.
2. If answer is ‘Not_available’, then chatbot can not give the answer. It generates a random ticket, gives to the user, stores that query with intent in a text file to forward the query to respective department of the bank.
3. Else chatbot will give the answer available in the answer column.

Accuracy: We get about 92% overall accuracy.