



CUSTOMER SERVICE CHAT BOT FOR TUPL.



STUDENTS: YIFU WU, BOB FAN, KEYI ZHOU

Objective

Tupl, Inc. is a tech company providing service for different companies. The company is planning to have an intelligence chatbot that can response to customers problem. It is common for companies to use a chatbot for customer support today. The chatbot is usually powered by natural language processing and machine learning. The chatbot we are developing aim to serve for phone service company. The chatbot should be able to identify user intent from text input, classify user's problem, respond with logical answers, and take correct actions according to different customer's situation.

Requirement for chatbot

- The chatbot should identify and make proper response to the user's intent like chitchat, problem complaints, or providing information.
- When the user makes complaints, the chatbot should identify the exact problem and its complexity and make proper reactions to those conditions. If the problems are too complex or the description by the user is too ambiguous, the chatbot will connect the user with an agent.
- During the entire process, the chatbot will record the information from the user such as the phone number, address, and country name and save them into the company's database. It will identify the exact cause of user complaints and save it to the database as well.
- The entire project will be deployed on Docker. The process of deployment is expected to be easy and straightforward.

ADDRESS AND PHONE IDENTIFICATION

- ShipEngineer API
 - Parse address data from unstructured text
 - Use function to send request to API and process feedback object
 - REST API and HTTPS Request
- Twilio API
 - Parse phone number from unstructured text
 - support different formats of phone number
 - output a standard format of phone number
 - able to detect different carrier, countries, and type

```
s = "i live in 4545 8th Ave NE seattle, WA 98105"
getaddress(s)
```

```
{'address_line1': '4545 8th Ave NE',
'address_residential_indicator': 'unknown',
'city_locality': 'Seattle',
'postal_code': '98105',
'state_province': 'WA'}
```

```
s = "1234 South Kenth Street is my adress"
getaddress(s)
```

```
{'address_line1': '1234 South Kenth Street',
'address_residential_indicator': 'unknown'}
```

```
[ ] get_number('My phone number is +1 206 777-7777')
```

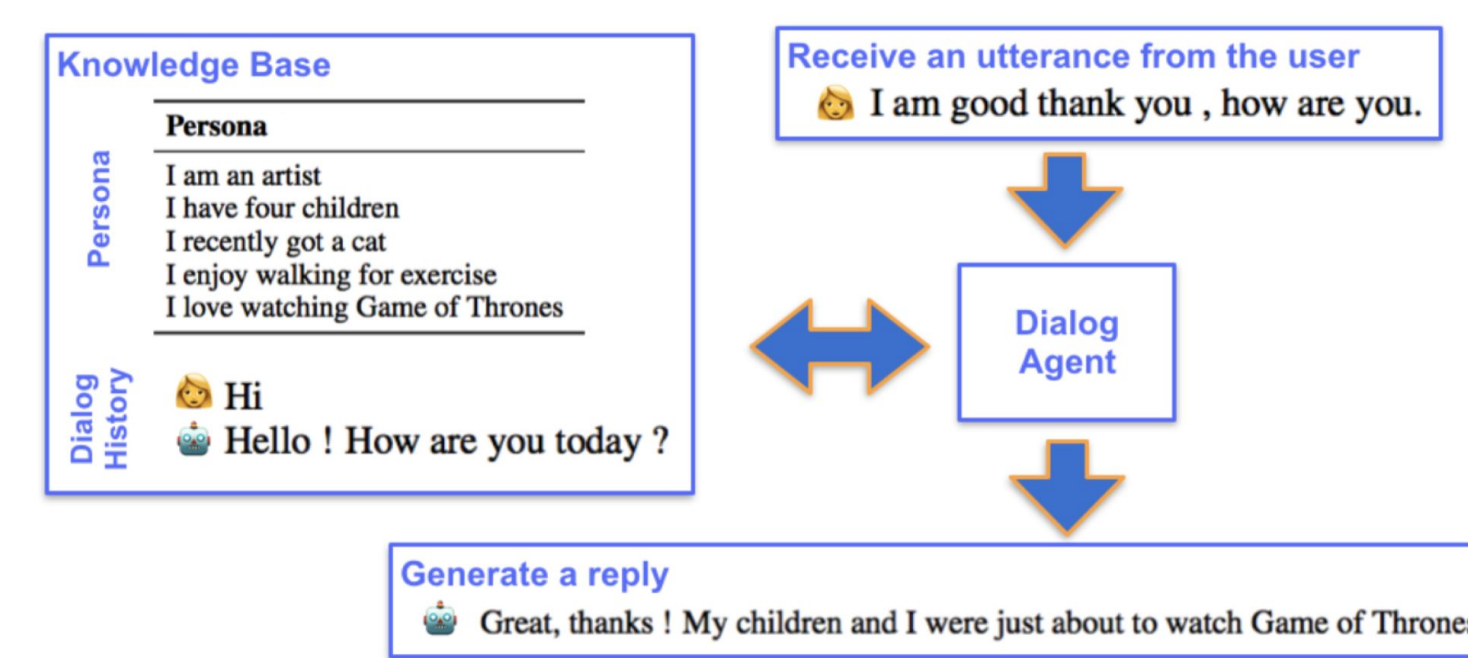
```
{'carrier_name': 'T-Mobile USA, Inc.',
'country_code': 'US',
'phone_number': '(206) 777-7777',
'type': 'mobile'}
```

```
s = "currently in Seattle, WA"
getaddress(s)
```

```
{'address_residential_indicator': 'unknown',
'city_locality': 'Seattle',
'state_province': 'WA'}
```

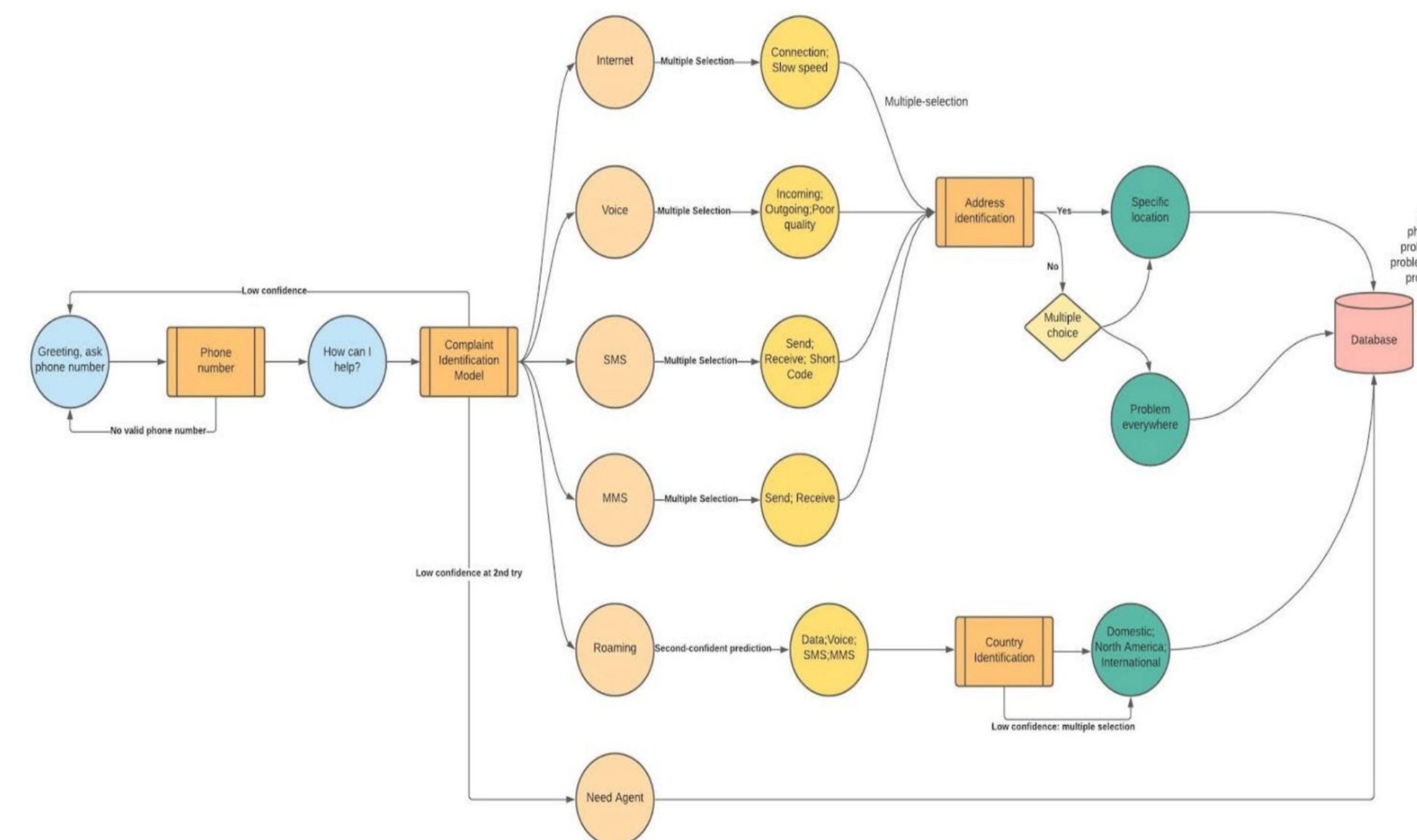
Data Science Implementation

- Greeting and general chat is done by chatbot trained by transfer learning.
- We used a large scale pretrained language model, OpenAI GPT combined with a Transfer Learning fine-tuning technique.
- The chatbot we trained is an AI with personality. We set fine tuned the chatbot to have personality of a customer chatbot.
- The chatbot uses BERT model to embed sentences.



- RASA Framework
- Open-source chatbot framework
- Understand **intent** from messages
- Easy to train
- Support Slack API

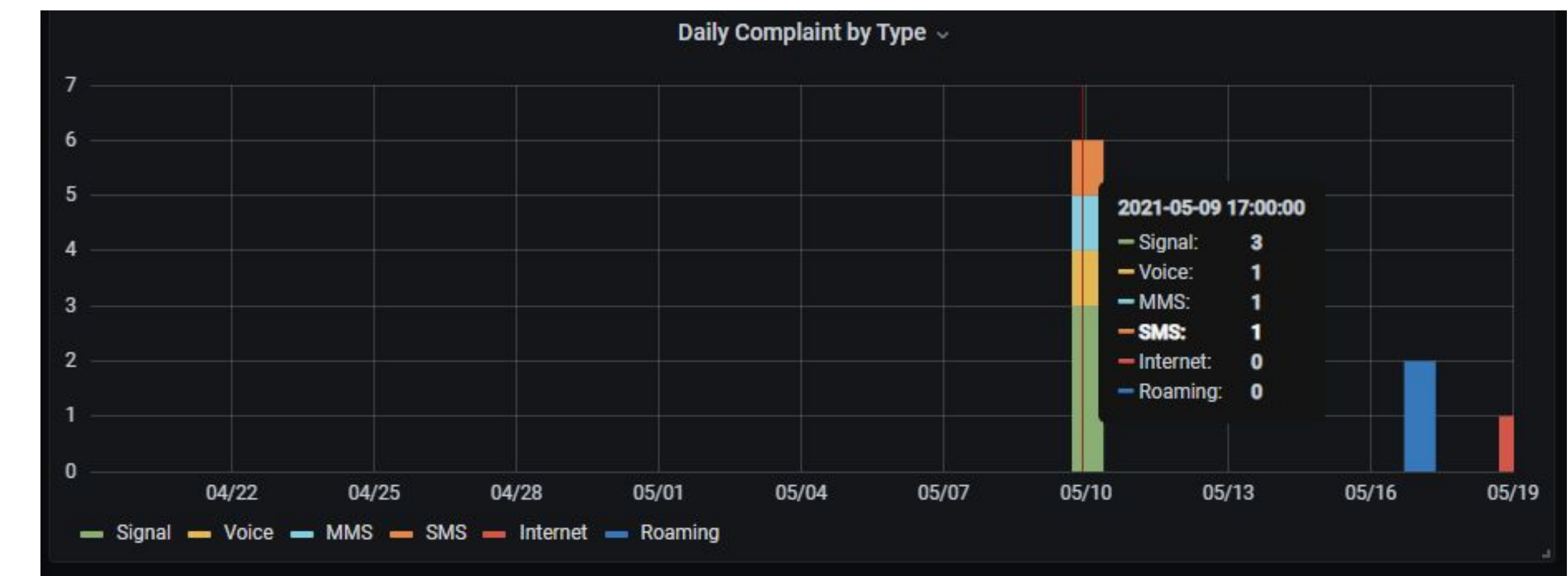
A structure to meet the requirement



The state machine diagram above shows our current design for the chatbot.

- The chitchat model will run to greet and chat with customers until a phone number is extracted.
- A deep learning based complaint identification model will identify user's question.
- Uses API and regex to extract address and country information from input text.
- Store user's information and problem into dataset.

SYSTEM INTEGRATION IMPLEMENTATION



- Data about complaints including phone numbers, problem types and timestamp will be stored into PostgreSQL database at the end of each conversation.
- A dashboard driven by Grafana will interact with the database in real time and display the amount the daily complaint by type in time series.

Yifu Wu 12:27 AM
hi

Dummy Bot APP 12:27 AM
Hello! Please enter your phone number with country code. **Chitchat model**

Yifu Wu 12:28 AM
12062215270 **Phone number identification**

Dummy Bot APP 12:28 AM
Got it! How can I help you today?

Yifu Wu 12:28 AM
I cannot access internet **problem identification model**

Dummy Bot APP 12:28 AM
Trying to identify your problem...

I see! Your problem is about: Internet.
Please select the category your problem belongs to.

You chose: 'Connection Failure'.
Where did the problem occur?

You chose: 'Everywhere'.
Your complaint has been recorded. Thank you for reaching out to us.

- Slack web client can send messages and format with block kit (in JSON).
- Slack event adapter subscribe to new messages and response to triggered event. (eg., button)

Future Work, References, and Acknowledgments

Thanks to advisors, sponsors and TAs, our team has accomplished a lot. Although we have distributed a chatbot with complete functionality, there are many improvement. For example, the response time of the models can be shorten to increase the chat bot's fluency. The chitchat model can be integrated better, too.

- [1] Slack API: Slack. 2021. Using the Slack Web API. [online] Available at: <https://api.slack.com/web> [Accessed 15 March 2021].
- [2] Slack Event Adapter: Slack. 2021. Using the Slack Events API. [online] Available at: <https://api.slack.com/apis/connections/events-api> [Accessed 15 March 2021].
- [3] Rasa Installation: Rasa.com. 2021. Installation. [online] Available at: <https://rasa.com/docs/rasa/installation/> [Accessed 15 March 2021].
- [4] Ship Engine API: Shipengine.github.io. 2021. ShipEngine API. [online] Available at: <https://shipengine.github.io/shipengine-openapi/> [Accessed 15 March 2021].

