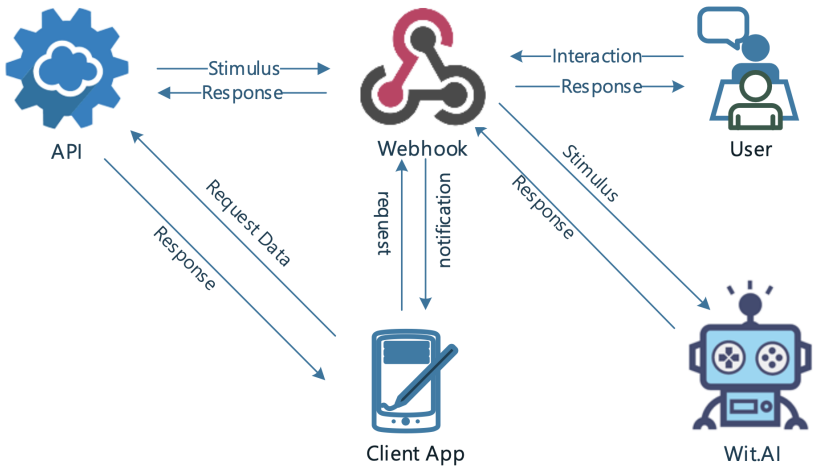
Summary on Ticketing Chatbot using Serverless NLP Technology

Introduction

- Chat is a meaningful form of communication between humans and is becoming a more natural and widely used interaction for humanising computer interactions.
- Chatbots interpret user messages and respond based on the meaning captured from the message.
- Social media platforms like Facebook, LINE, and Telegram have allowed developers to build chat automation platforms and utilise chat dialogue for promoting features and products.
- Natural Language Processing (NLP) is essential for chatbot interactions and is often built into cloud-based cognitive services offered by platforms like IBM Watson, Wit.AI, and Dialog Flow.
- Serverless architecture is a solution for building chatbot functions that are stateless, scalable, and maintainable, with functions arranged in a conversational context.
- The Serverless model used in this work is Webhook, which receives direct messages from Facebook pages and connects with Wit.AI for NLP features.
- NLP features, such as location, intent, or number, are parsed to serverless functions to provide specific responses.

- Serverless functions can interact with external APIs, such as ticketing APIs, to retrieve relevant information.

Methodology



Serverless Architecture Chatbot using Wit.AI

The system can divide into 3 parts:

1. Node JS Webhook
2. Wit.AI NLP services
3. Ticket.com Order API

Node JS Webhook

- Webhook is like an HTTP callback, which means it can perform certain actions when triggered by a POST or GET request.
- It works with Node JS to get specific responses from HTTP requests.
- GET requests are used to register applications to webhook services on Facebook, while POST requests occur when there's a new chat incoming to a Facebook Page.
- Webhook allows easy data integration and processing, with data interchange in response form.
- The data can be saved into a variable and used in other applications.
- Its main purpose is real-time data processing and direct sending of responses.

- NLP services can utilize specific functions defined in webhooks, making data processing more flexible.

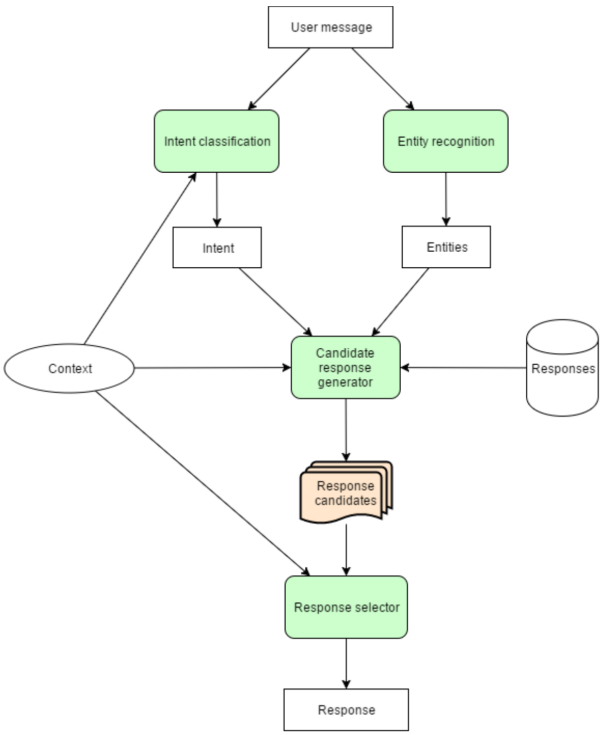
Wit.AI NLP services

- Machine learning is great when the problem is well-defined by the training dataset. For instance, training a chatbot with many examples helps it gain more knowledge.
- Business logic can get complicated even with a lot of knowledge because it's challenging to perfectly model all possible conversation situations.
- On the other hand, using rules offers simplicity. A few rules can make a chatbot functional and interactive with users. For example, if a chatbot discovers a new topic in the conversation, new rules must be added to handle it.
- Wit.AI is a system trained with understanding, combining entity and intent recognition based on commonly used keywords in chats. For example, if a user writes, "I want to go from Jakarta to Bali on 15 July 2018," Wit.AI can recognize the entities "Jakarta," "Bali," and the intent of "travel."
- Chatbots can be triggered by specific keywords to activate certain conditions. For example, specific keywords like "order," "book," or "find" can trigger different actions in the chatbot.
- Wit.AI allows developers to define custom entities to enhance chatbot capabilities. For example, a travel chatbot may define custom entities for airline names, hotel preferences, and travel dates.
- Wit.AI integrates with webhooks, where information is sent through web services and processed in functions. This shift allows users to interact with the chatbot through a mix of text and graphical user interface (GUI) elements. For example, users can see buttons in the chatbot interface to access specific features or options.

Tiket.com API

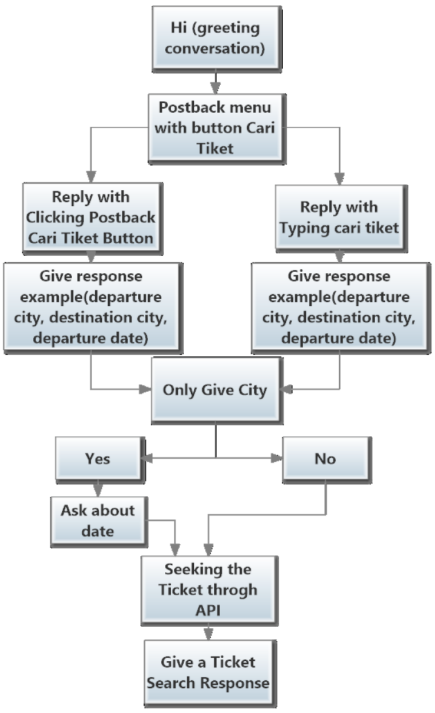
- tiket.com offers a cloud-hosted web service for searching and booking flights.
- Users interested in becoming affiliates can use this service by sending requests with departure city, destination city, and flight departure date.
- The API provides flight information such as flight number, airline name, departure time, and ticket prices from various airlines.

- By combining webhook, chatbot can request flight information from the API when the user provides the right parameters.
- The chatbot application consists of :
  - An Intent classification module to identify user intent.
  - An Entity recognition module to extract structured information from the message.
  - A Candidate response generator that selects the best response based on the conversational flow and user interface.



Chatbot architecture

Conversational Flow



- Conversational flow guides the conversation in NLP services, ensuring it follows specific rules.
- The user initiates the conversation by sending a message to the chatbot.
- The chatbot's functionality becomes accessible after the user sends a message.
- Ready-made applications like Botika provide a basic framework for conversational flow.
- Conversation agent processes user requests and maps them to relevant intents, which represent the domain of the user's conversation by keywords.
- Wit.AI is a platform that performs Natural Language Understanding (NLU), grouping requests based on captured keywords or intents.
- Intents can have synonym keywords representing specific statements, like "hi" for greetings.
- Wit.AI uses NLP for parsing user requests, employing a linear model with vectorised text from the DBpedia dataset to classify the user's message.
- The linear model stores words as vectors with defined positions (subject, verb, object) for better understanding.
- The Conversation User Interface (CUI) is provided to connect the chatbot with the user, and it may contain interactive buttons for easy interactions.

- Interface design facilitates the conversation's initiation through the linear model, returning NLP entities with confidence levels for user queries.
- Each conversation requires parameters called slots, like location and time, which can be determined through Slot Filling, similar to Named Entity Recognition (NER) tasks.
- Wit.AI can detect specific words like locations, numbers, people, and time sequences using Named Entity Recognition for slot inference tasks.

Conversation User Interface

Conversational Templates and Testing Scenario:

- Conversational templates provide examples for users on how to respond to the chatbot's queries.
- Keywords must be typed correctly for the chatbot to respond accurately.
- Dual keywords can be placed separately in a sentence due to slot filling mechanism.

Serverless Programming and Promises:

- Serverless programming uses Promises to handle responses from third-party APIs asynchronously.
- Getting a response from an API takes time, so asynchronous programming is used.
- JavaScript cannot directly update variables if the process is not synchronous.

NLP Entities and Location Response:

- The message contains NLP entities with information about the user's request.
- Matching keywords in NLP entities triggers specific responses.
- Facebook Messenger example shows a location response with a confidence level of 0.92061.
- Inaccurate location definitions lower confidence scores. For instance, if "Jakarta" is written as "jakar", the confidence score decreases from 0.92061 to 0.90489.

Findings from the Conversation

Summary of interesting findings in chatbot execution:

- Saying "Hi" as the first act takes longer execution time due to interconnecting with Facebook API to fetch user information.
- Chat interaction with natural language understanding and slots takes less time than keyword-based conversations.
- Clicking the postback button has slightly longer execution time compared to natural chat interactions.
- Location and date intents show longer execution time in the first conversation but reduce in subsequent interactions, thanks to chat history saving.
- Combining intents in one chat can lead to reduced execution time compared to separating them.
- Cyclomatic complexity in serverless technology arises due to nested if statements when waiting for callbacks, affecting execution time.

Chatbot Testing

- Rule of thumb suggests that information searching requires a minimum of 50 examples for testing.
- In specific conditions with narrower templates and rules, testing with less than 50 examples might be acceptable.
- To handle mistyping, some processing is required to tolerate minor errors if the meaning remains the same.
- Measurement in chatbot testing uses a binary value, with valid responses (relevant answers) counted as 1 and invalid responses (non-relevant answers) as 0.
- Table shows the notation for relevant testing.

	Relevant	Non-Relevant
Retrieved	13	3
Non Retrieved	0	0

Measurement Inference:

- Precision rate for intent testing is 81.25%, and recall is 100% as the chat always responds.

- F-measure (harmonic mean of precision and recall) is 89.65%, derived from twice the product of precision and recall divided by the sum of precision and recall.

Future Work:

- NLP algorithms can be improved with memory cells to remember early conversations, saving user behaviour in a hidden Markov model.
- User behaviour that deviates from the chatbot's instructions can be overcome using this memory cell.
- By modelling the relationships between requests, wrong typing can be predicted and provide smoother responses.

Conclusion

- Specific domain chatbots redefine chat experiences with automated responses and guided Conversational User Interface (CUI) interactions.
- The novelty lies in the chatbot's intelligence, covering out-of-topic and mistyping situations.
- Ticketing chatbot shows good response and direction but requires a more sophisticated algorithm to handle all user requests.
- Intent classification in sentences should be smoother to consider confidence rates.
- Analysing chat history can enhance the understanding of user behaviour.
- Integrating with other services like Google Home Cloud AI can add interesting features, such as sound capabilities.