Ticketing Chatbot Service using Serverless NLP Technology

Eko Handoyo
Department of Electrical Engineering
Diponegoro University
Semarang, Indonesia
eko_handoyo@elektro.undip.ac.id

M.Arfan
Department of Electrical Engineering
Diponegoro University
Semarang, Indonesia
arfan@elektro.undip.ac.id

Yosua Alvin Adi Soetrisno
Department of Electrical Engineering
Diponegoro University
Semarang, Indonesia
yosua@live.undip.ac.id

Maman Somantri
Department of Electrical Engineering
Diponegoro University
Semarang, Indonesia
mmsomantri@live.undip.ac.id

Aghus Sofwan
Department of Electrical Engineering
Diponegoro University
Semarang, Indonesia
asofwan@elektro.undip.ac.id

Enda Wista Sinuraya
Department of Electrical Engineering
Diponegoro University
Semarang, Indonesia
enda_sinuraya@elektro.undip.ac.id

***Abstract*— Personal assistant using a human operator need some time to process single request such as ticket booking, ordering something, and get services. One request can contain many queries for some information provided on the internet. Business performance values time efficiency so must be considered an alternative way to take request. Chatbot can give 24 hours service which can become an advantage besides using a human personal assistant. Chatbot acts like routing agent that can classify user context in conversation. Chatbot helped with natural language processing (NLP) to analyze the request and extract some keyword information. One important process in NLP is morphological analysis and part of speech (POS) tagging. POS help to parse the meaning of chat text based on a set of rules. The rule base is specific to some language and designed to capture all the keyword relies on chat text. Keyword in booking conversation term is like departure and destination city and also the date of flight. There is a variation from a user determining city and date. NLP in booking confirmation has a task to analyze various pattern describing ordering requests like city and date. Messenger bot would be an example of assistance that can help user connected to many services some like ticketing service through conversation interaction. The contribution of this research is to conduct some scenario that happening in ordering tickets. This research conduct that chatbot can help acts as customer service, based on the conducted scenario and show an F-measure score of 89.65%.**

Keywords—chatbot, routing agent, conversation, NLP, interaction, intent

I. Introduction

Chat or speech is one meaningful form of communication between humans[1]. Chat becomes more natural interaction than graphic base interface so will be broadly used in humanizing computer interaction to human. Chatbot worked by interpreting the message that given by the user, and then give response base on captured parsed meaning of the message [2]. In 2015, Facebook as a social media platform allows some developer to build chat automation platform. This policy followed by another social chat platform like LINE and Telegram. Facebook provides a button and chats dialogue to help chat interaction especially to promoting some feature in chatbot. Chat dialogue also can be used to promoting product service and how to access those services. Chatbot interaction is the most important feature to design. Chatbot interaction must meet user need to the product. Although graphical based interface designed to be executed well, there is still some potential using chatbot. Icon clicking interaction sometimes misused because the user is not computers friendly enough [3]. Chat interaction can also be helped assisted routing to specific command if the user wrongly sent the request.

NLP as the core of chat interaction is known build in cloud-based cognitive service. There are many services provided AI building blocks such as IBM Watson, Wit.AI, and Dialog Flow. Wit.AI is one of natural language interface for an application that capable turning sentence into structured data. The developer can integrate the service to well-known social media platform such as Facebook with the token. Wit.AI provide a built-in building block that can detect special intent word in a sentence.

The developer has to handle coordination between cognitive service such as chatbot interface, integrate chatbot with third-party services, and also considering extensibility, scalability, and maintenance [4]. Serverless becomes a solution that let developer build function into the shared platform. Serverless build in a standard language with stateless technology. Stateless technology doesn't store session information. Serverless programming model, deploy function that can be reach or executed from the cloud. Functions are stateless because of independence from previous runs. The function can invoke directly or triggered by some events. Chatbot application will utilize some function that can be arranged in conversational context [5]. In a conversational context, function chained together by sequence. In this work, Serverless model that used is Webhook. Webhook conducted to receive a direct message from the Facebook page. Facebook page connected to Wit.AI NLP service to get NLP features. Facebook send message response with NLP features such as location, intent, or number. NLP features parse to serverless function so can return some specific response. For specific location intent response, the Serverless function does a querying out a mechanism to external ticketing API to get price and information.

The remainder of this paper is organized as follows. In Section 2 we provide methods of NLP and serverless programming model and how to integrate it to social media. In Section 3 we provide testing of several chatting scenarios.

In Section 4 we arrange a discussion to further development of chatbot using serverless programming model.

II. Methods

Figure 1 shows the system architecture request and response flow. The system can divide into three part: the first part is Node JS webhook, the second part is Wit.AI NLP services and the third part is Ticket.com Order API. The detail of each stage is described as follows.

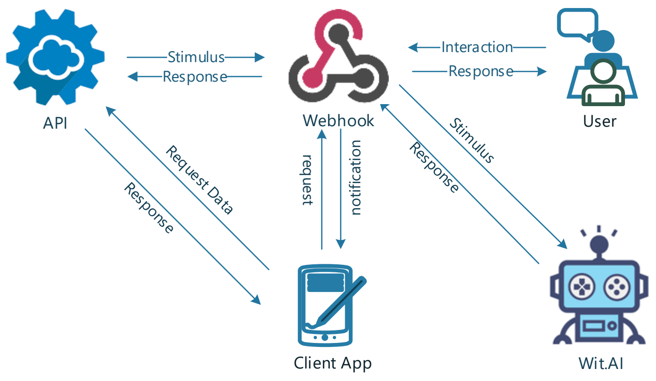


Figure 1 Serverless Architecture Chatbot using Wit.AI

A. Node JS Webhook

Webhook is HTTP callback. The basic concept of webhook is simple that webhook can work if there is some trigger base on post request or get request. POST and GET request from HTTP is routed by Node JS to get a specific response. GET request used to register application to webhook services in Facebook. POST request is coming when there is some new chat incoming to Facebook Page. The advantage of using webhook is easy data integration because of data interchange in response form. Data processed and saved into a variable that can be used in another application. Webhook main purpose is to approaching data in a real-time manner and can send directly after response processing. Webhook also make NLP service can use a specific function defined in webhook so data process is more flexible.

B. Wit.AI NLP Services

Machine learning more desirable when the problem is defined by the training dataset. Training a chatbot intent with a lot of examples can make chatbot leveraging more knowledge. Business logic can form several complexities although it has much knowledge. Complexity happens because the conversation situation can't be modeled exactly.

Rules are kind of the opposite from purely machine learning. The good thing about rules is with a few rules, chatbot can working and involving the user. If there is a discovery of a new topic in the conversation, some new rule must be added. Wit.AI is trained with understanding. Understanding is a combination of entity and intent recognition. Wit.AI trained with some keyword that user usually writes in chat [6].

Chatbot can have a combination of certain keyword that used to trigger specific condition. If there is some conversation example like "I want to go from Jakarta to Bali on 15 July 2018", there is a statement of the first city of departure and also a destination with date information. Wit.AI allows defining custom entities that developer wants to build.

Wit.AI based on webhook integration where information sent through web services to be processed in function. Because not all the understanding created by the developer can make sense of some user, Wit.AI shifts from complex text-only transaction to mixed GUI element interaction. The user can get a web view of some button that shows the feature of the chatbot.

C. Tiket.com API

The service provider tiket.com offers a web service to search and book flights. This service is hosted in the cloud and can be used by a user that has an interest in becoming affiliation. Request for flight information can be completed if the departure city, destination city, and flight departure date is set [7]. Flight information that offered from API is flight number, airline name, departure time, and also a ticket price from several airline companies. Combined with webhook, the request information can be affordable to the user if the user sent the right parameter. Chatbot conversation can make a condition that the user can complete the minimal data, to request the ticketing API services [8].

The overall chatbot application is shown in Figure 2. Intent classification module identifies the intent of user messages. Entity recognition module extracts structured information from the message [6]. Candidate response generator parses the message and selects the best response for the user. Design of candidate response generator based on conversational flow and conversation user interface.

A. Conversational Flow

Conversational flow is a flow of conversation exist in NLP services. Conversation flow guides the conversation so can flow with a specific rule. The user can ask something but to reach the chatbot functionality, the user must send something. As the reference of conversational flow, there is some chatbot ready-made application such as Botika that can give a basic framework of conversation. Conversational flow is shown in Figure 3.

After designing conversational flow, conversation agent is designed. Conversation agent process request from a user and maps the request into the part that sufficient with the request meaning. The pattern of conversation in Wit.AI is called intent. The intent is representing the domain from user conversation by keyword. Wit.AI platform does Natural Language Understanding which is grouping the request by the keyword captured [9]. For example, if we have greeting intent, we can name the intent with "hi" and provide some synonym keyword of the intent that represents "hi" statement [10]. The intent that created for this chatbot showed in Figure 4.

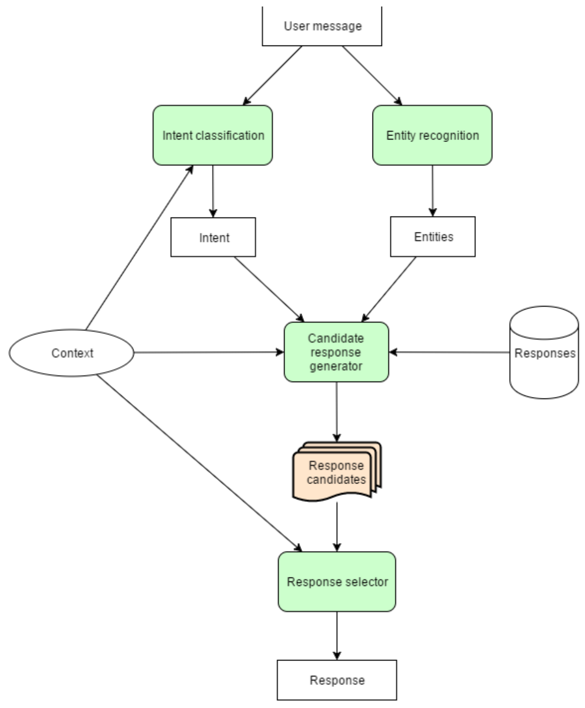
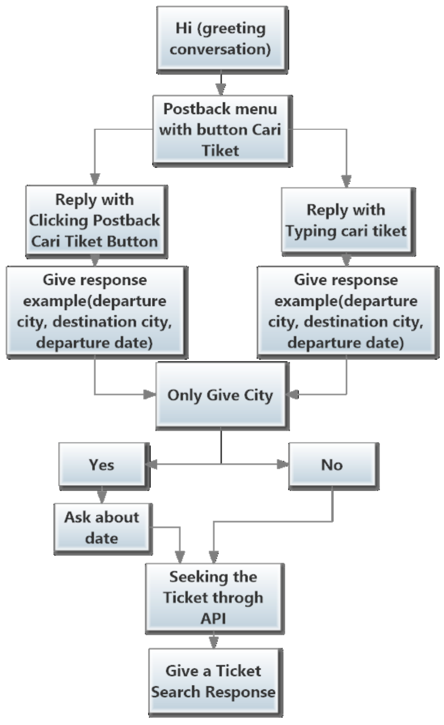


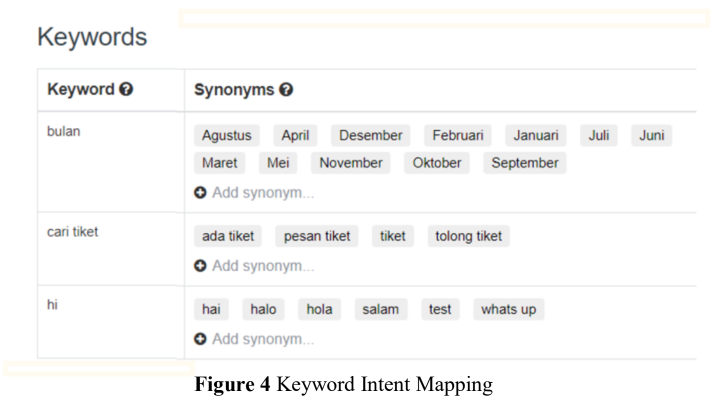
Figure 2 Chatbot Architecture [6]


Hi (greeting conversation)
Postback menu with button Cari Tiket
Reply with Clicking Postback Cari Tiket Button
Reply with Typing cari tiket
Give response example(departure city, destination city, departure date)
Give response example(departure city, destination city, departure date)
Only Give City
Yes
No
Ask about date
Seeking the Ticket throgh API
Give a Ticket Search Response
Figure 3 Conversational Flow [6]

Wit.AI has functionality that can parsing user request using NLP. NLP understand user request with classification process using vectorized text stored in the linear model. Linear model stores collection of the word come from DBpedia dataset with a defined position on the sentence as a subject, verb or object in multidimensional vector [1]. If the user's chat is having similarity with phrase or word from the

Conversation user interface (CUI) is interfacing that provided with a chatbot to connect with the user, CUI can contain a button that user can click. The user can click the button to interactively replacing chat with the chatbot. There is some interface design to begin the conversation like in

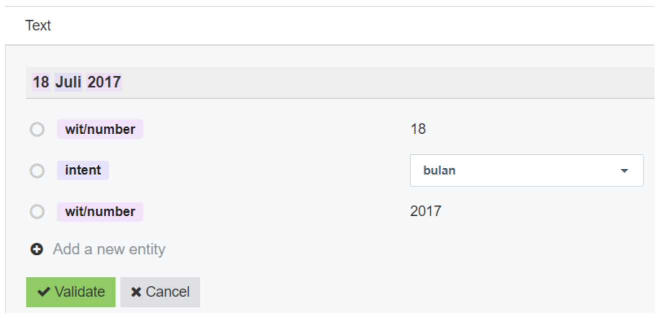
linear model, then the request returns some NLP entities with a degree of confidence.


Keywords

Keyword	Synonyms
bulan	Agustus, April, Desember, Februari, Januari, Juli, Juni, Maret, Mei, November, Oktober, September Add synonym...
cari tiket	ada tiket, pesan tiket, tiket, tolong tiket Add synonym...
hi	hai, halo, hola, salam, test, whats up Add synonym...

Figure 4 Keyword Intent Mapping

Each query or conversation must have some parameters. For example, to get user destination, the system needs to know where location and when time. These parameters are called slots. Between slots, there are some connecting words that the position can be determined with slot inference task. Slots inference task is called Slot Filling and it is similar to the Named Entity Recognition task. Wit.AI can detect specific word like location, number, people, and time sequence based on Named Entity Recognition [11]. Date entity rule used in this research showed in Figure 5 and location entity rule showed in Figure 6. Figure 5 and 6 are rules made in the Indonesian language because the request is done in the Indonesian language.

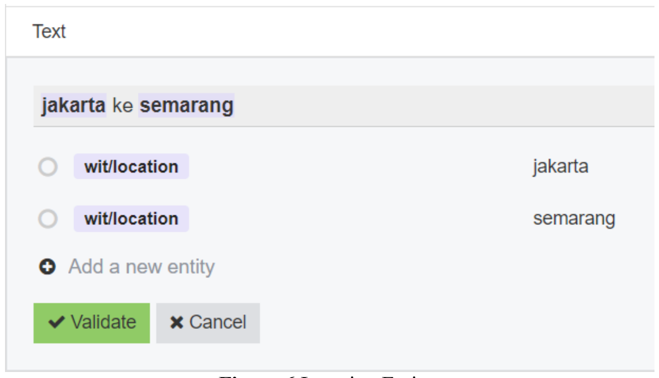

Text

18 Juli 2017

wit/number	18
intent	bulan
wit/number	2017

Add a new entity

Figure 5 Date Entity


Text

jakarta ke semarang

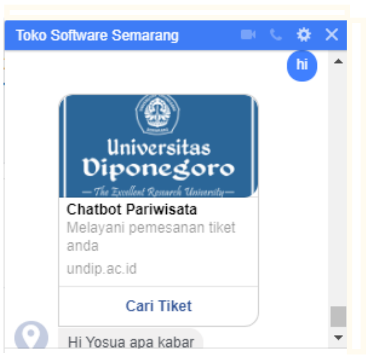
wit/location	jakarta
wit/location	semarang

Add a new entity

Figure 6 Location Entity

B. Conversation User Interface

Figure 7. Beside conversational user interface, there is also used the conversational template to give user example how to reply the chat to do a query.


Toko Software Semarang

hi

Universitas Diponegoro

Chatbot Pariwisata

Melayani pemesanan tiket anda

undip.ac.id

Cari Tiket

Hi Yosua apa kabar

Figure 7 CUI Interaction

III. Testing and Results

Testing on chatbot environment has been designed to test the valid response if the user gives some request. The testing scenario is defined in Table 1. There is some scenario to check that keyword working well. Keyword must be typed correctly to get the right response. The dual keyword can be placed separately in a sentence because of slot filling mechanism.

Serverless programming uses Promise to wait response from third party API. Mechanism to get response from API needs a time, and saving process in variable also cannot directly done because serverless using asynchronous programming. Javascript cannot directly update the variable if the process is not synchronal. This is an example code to get Promise request in serverless technology. Procedure then in function must be done.

```
app.post('/', (req,res,next)=>{
const response = req.body
if(response.object === "page") {
const messageObj = bot.getMessageObject(response)
if (messageObj.message=='hi')
{
bot.getSenderProfile(response).then(function(value)
{
var firstname = value.first_name;
dataChat['first_name'] = firstname;
bot.sendText(`Hi ${firstname} apa kabar`,
messageObj.id)
});
}.
```

Message contains many information including NLP entities. If the keyword using in entity is matching with rule then NLP entities is appear. This is the example of location response given by Facebook messenger. The NLP entity of location is given with confidence of 0.92061.

```
{"object": "page","entry": [
{"id": "1146485878750741",
"time": 1531474140934,
"messaging": [{
"sender": {"id": "1110805859005876"},
"recipient": {"id": "1146485878750741"},
"timestamp": 1531474140643,
"message":
{"mid": "zMyGbeJHB8ieb75o3OECKRRg22kpxJe7rFT55_nWcejXlg
0_4RatoGJRVp3M4a1rG9x8Zot10PULGzSKsxqa6w",
"seq": 46361,
"text": "jakarta",
"nlp": {"entities": {
"location": [{"suggested": true,"confidence": 0.92061,
"value": "jakarta","type": "value"}]}}}}]}]}
```

If location given in conversation is not correctly defined then the confidence score is become turning down. For example, if the city name Jakarta written as jakar, the confidence score turn down to 0.90489 from 0.92061.

```
"nlp": {
"entities": {
"location": [
{"suggested": true,
"confidence": 0.90489,
"value": "jakar",
"type": "value"
}]}}
```

Conversation reliability also can be measured by execution time. Execution time shows the convergence of understanding. There are many parameters that affect execution time beside the quality of the network. Table 1 shows all user action comparison that happened in the conversation.

Table 1 Execution Time Comparison

User Action	Execution Time 1	Execution Time 2	Execution Time 3
Says Hi	6.433ms	1.454ms	1.217ms
Clicks Postback "Cari Tiket"	1.460ms	0.857ms	0.628ms
Says "cari ticket"	1.415ms	0.389ms	0.625ms
Says "apakah cari tiket mudah"	1.309ms	0.558ms	0.358ms
Says "jakarta ke semarang"	0.175ms	0.035ms	0.009ms
Says "13 Juli 2018 dari jakarta ke semarang"	0.928ms	0.119ms	0.053ms
Says "jakarta ke semarang 13 Juli 2018"	0.960ms	0.172ms	0.064ms

In Table 1, there is some interesting finding. Say Hi as the first act gives a long execution time because must interconnect with Facebook API to get user information. If we say "hi" then replied with "hi" + username which username came from Facebook API.

From execution time we find that chat interaction takes more briefly execution than clicking the postback button although the difference is very little. It can differ from the situation that button callback function takes more time than natural chat interaction although chat has more information.

Chat interaction with natural language understanding which using slot in conversation takes less time than exactly keyword conversation. In another side, that keyword rule cannot be a break with wrongly typed word. The keyword can be separated by the word but cannot mistype.

Location intent and date intent as the main feature show different execution time. As the first executed conversation location and date intent take a longer time. Execution time becomes lower after same execution happen. Wit.AI platform saves chat history and can be inferred that chat history help reducing the time of already same pattern conversation.

If the conversational joined into one chat like location and date intent which is joined can lead to reducing execution time. Separation of chat takes more time because of takes more cyclomatic complexity. Cyclomatic complexity happens in serverless technology because of the callback problem. Process waiting callback makes there is a lot of nested if. “If” executed inside “if” can take more time. Data showing the experiment and evaluation of mistyping and out of topic sentences are shown in Table 4. Table 4 also considered some scenario when a user using another keyword for “cari tiket” intent, not generally city put in location request, and also some date sequence.

IV. Discussion

Based on the rule of thumb that information searching takes minimum data matching in one testing is using 50 example [12]. In specific condition testing which the template and rule become narrower testing using less than 50 example doesn’t matter. From Table 3 there are total 16 scenarios that tested in chatbot. From 16 scenarios there are three conditions that not meet the output expectation. Mistyping in chat must be taken into some processing so a little wrong typing can be tolerated if the meaning is still same. The measurement in chatbot testing must use a binary value, that valid response (relevant answer) count as 1 and not valid response (non-relevant answer) count as 0. Tabel 2 shows the notation of relevant testing.

Table 2 Relevant Testing Result of Scenario

	Relevant	Non-Relevant
Retrieved	13	3
Non Retrieved	0	0

Based on Table 2, we can infer measurement of precision, recall, and F-measure. Precision rate of intent testing is 81.25%, recall is 100% because the chat always responded. The harmonic mean of precision and recall is 89.65 % which come from twice of precision times recall divided by precision plus recall.

In the future work, the algorithm of NLP can be improved with some memory cell that can remind early conversation. The memory cell can save user behavior to be measured as a hidden Markov model. There is a behavior from a user that not give response same as chatbot instructed. That behavior must be overcome by a chatbot. If modeling between request can obtain, wrong typing can also be predicted and give a more smooth response.

V. Conclusion

Specific domain chatbot can redefine chat experienced with the automated response and also some CUI response that guided the user. The novelty lies in the way determine chatbot intelligence that can cover and overcome out of topic and also mistyping situation.

Table 3 Intent Testing Output

Test Scenario	Input
Do a greeting message like hi, hai, halo, hola, salam, test, whats up	User says "hi"
	User says synonym of hi (hai, halo, …, whatsup)
	User says synonym with a typo (haiiiiii or halooooo)
Do a click in postback cari tiket button	User clicks cari tiket button
Do a cari tiket typing in chat	User says "cari tiket"
	User says "mau cari sejenis tiket dong" (slot filling beside keyword)
Do a location input with departure and destination city	User says "jakarta ke semarang"
	User says "jakar to semar" (typo in two location)
	User says "jakarta to semar" (typo in one location)
	User says "dari semarang menuju ke jakarta" (slot filling beside location)
Do a location input with departure and destination city with departure date	User says "jakarta ke semarang tanggal 15 Juli 2018"
	User says "jakarta ke semarang, 15-07-2018"
	User says "jakarta ke semarang, 15/07/2018"
Do a date input after location input	User says "tanggal 15 Juli 2018"
	User says "15-07-2018"
	User says "15/07/2018"

Overall from the harmonic means, ticketing chatbot show that can respond well and give direction but need a more sophisticated algorithm to overcome all occurrence in the user request. Classification of intent in term of the sentence needs to be more fluent to consider confidence rate. In the future, chat history can be considered as chat experience so the behavior of the user can be analyzed. To create more interesting chatbot, chatbot needs to be connected to another service like Google Home Cloud AI to enable a more interesting feature like sound.

Table 4 Intent Testing Output (continue)

Output Expected	Output Result
Response with webview menu for admitting cari tiket button	Valid Response
Response with webview menu for admitting cari tiket button	Valid Response
Response with webview menu for admitting cari tiket button	Not valid, chatbot don’t understand intent
Response with chat for giving example to location request	Valid Response
Response with chat for giving example to location request	Valid Response
Response with chat for giving example to location request	Valid Response
Response with asking departure date	Valid Response
Response with asking departure date	Not valid, chatbot don’t get the location
Response with asking departure date	Not valid, chatbot only

	capture one location
Response with asking departure date	Valid Response
Response with ticket API response	Valid Response
Response with ticket API response	Valid Response
Response with ticket API response	Valid Response
Response with ticket API response	Valid Response
Response with ticket API response	Valid Response
Response with ticket API response	Valid Response

Table 5 Experimental Data for Mistyping and some synonym keyword Scenario

Intent Qty	Chat Message	Confidence Rate
1	sore mau pesan tiket	1
1	saya mau nyari tiket	1
1	mau pesan tiket bisa ngga ya?	1
1	buruan pesen tiket dong	1
1	mau tanya tiket bisa	1
2	papua ke semarang bisa?	0.93686
2	amsterdam ke new york gimana?	0.89293
2	batang ke pemalang bisa?	0.9094
2	aku ngga tau jalan ke jakarta lewat semarang bisa ngga?	0.93231
2	jakarta ke semarang aja gimana?	0.93201
3	tanggalnya 17 mei gitu bisa thn 2018?	1
3	apa bisa ya di tanggal 18-08-2018	1
3	tanggalnya di 17/07/2018	1
3	coba pesan di 2018 17 Agustus	1
3	masih ngga di september 17 2018	1

Table 6 Experimental Data for Mistyping and some synonym keyword Scenario (continue)

Confidence Rate	Confidence Rate	Detected As	Detected As	Detected As
		cari tiket		
		cari tiket		
		cari tiket		
		cari tiket		
		cari tiket		
0.93299		location	location	
0.923845		location	location	
0.93797		location	location	
0.89817		location	location	
0.93105		location	location	
1	1	number	month	number
1	1	number	number	number
-	1	number	-	number
1	1	number	number	month
1	1	month	number	number

References

[1] S. A. and D. John, "Survey on Chatbot Design Techniques in Speech Conversation Systems," *Int. J. Adv. Comput. Sci. Appl.*, vol. 6, no. 7, 2015.

[2] A. A. Akhsan and F. Faizah, "Analisis dan Perancangan Interaksi Chatbot Reminder dengan User-Centered Design," *J. Sist. Inf.*, vol. 13, no. 2, p. 78, Oct. 2017.

[3] B. Behera, "Chappie - A Semi-automatic Intelligent Chatbot," *Write-Up*, 2016.

[4] M. Yan, P. Castro, P. Cheng, and V. Ishakian, "Building a Chatbot with Serverless Computing," 2016, pp. 1–4.

[5] H. Lieberman and C. Mason, "Intelligent Agent Software for Medicine," p. 16.

[6] A. M. Rahman, A. A. Mamun, and A. Islam, "Programming challenges of chatbot: Current and future prospective," 2017, pp. 75–78.

[7] S. Schwichtenberg and G. Engels, "Automatized derivation of comprehensive specifications for black-box services," 2016, pp. 815–818.

[8] D. Oleh, "(STUDI KASUS HDKREASI)," p. 121.

[9] J. Jia, "CSIEC: A computer assisted English learning chatbot based on textual knowledge and reasoning," *Knowl.-Based Syst.*, vol. 22, no. 4, pp. 249–255, May 2009.

[10] J. Jia, "The Study of the Application of a Keywords-based Chatbot System on the Teaching of Foreign Languages," p. 11, 2002.

[11] M. Selvam and A. M. Natarajan, "Improvement of Rule Based Morphological Analysis and POS Tagging in Tamil Language via Projection and Induction Techniques," vol. 3, no. 4, p. 11, 2009.

[12] D. Domarco and N. M. S. Iswari, "Rancang Bangun Aplikasi Chatbot Sebagai Media Pencarian Informasi Anime Menggunakan Regular Expression Pattern Matching," *ULTIMATICS*, vol. IX, no. 1, pp. 19–24, 2017.