Correcting Typographical Error and Understanding User Intention in Chatbot by Combining N-Gram and Machine Learning Using Schema Matching Technique

Mikael L. Tedjopranoto, Andreas Wijaya, Levi Hanny Santoso, and Derwin Suhartono

***Abstract*—Purpose of this research is to make chatbot based system to help Small and Medium Enterprise business. Initially, we build this application only to help Small and Medium Enterprise owner to monitor their business and report. Yet, we realize that we can make our chatbot to be more effective and efficient using machine learning technique. N-gram and machine learning using schema matching are embedded to the chatbot to understand user intention and correct typographical error inside the sentences. Finally, the chatbot has been successfully achieved those objectives. It can be concluded that the chatbot can drive the users' feeling to be more convenient and help Small and Medium Enterprise owner to monitor their business.**

***Index Terms*—Chatbot, n-gram, machine learning, schema matching, typographical error, user intention.**

I. Introduction

There are no doubts that the digital era is developing quickly, with different platform reaching the users such as social media, email, and websites. We have a lot of ways of communication, the process of sending messages to become more and more interactive than it used to be. In the last few years, digital communication was about social media and information technology is influencing everything.

In the recent years, several emerging issues are posing serious challenges to the small and medium-sized enterprises (SME's). These enterprises enter the new era, and one challenge that worth to be addressed is globalization. Small medium enterprises also need to adapt to globalization, one way to do that is to digitalize their business with information technology.

As we can see, the role of IT is very big in small medium enterprises, 45% said that it is a necessary cost, the other 35% said that it is an enabler of business efficiency and the rest said it is a driver of competitive advantage or differentiation. In this research, we want to propose chatbot as a tool to help small medium enterprises in Indonesia. Chatbot is a computer program that has the ability to hold a conversation with a human using Natural Language Speech [1]. Basically, chatbot has an ability to send text messages and it feels like a human who sends it. These days people might be using these daily needs, for example, Siri and Cortana are intelligent personal assistants in the form of chatbot by Apple and Windows [2]. Using chatbot, people can easily send any information to their customers by creating some defined conversations. So, chatbots can be so convenient and easy to use.

The problem that business owner faced these days, is they need to come to their store to check their income, employees, and their stock. With chatbot, those things are not necessary anymore, business owner only needs to send a message to Business Bot, to do all of those things.

Chatbot is our future, however, people in Indonesia is not used to asking questions or do their daily basis using chatbot. Our idea is to introduce or SME (Small Medium Enterprise) to chatbot, so they can manage their business as easy as sending a chat message.

To make it more convenient and easy to use, we will attempt to develop a chatbot using machine learning. Machine learning is a subfield of Artificial Intelligence that gives the machine the capabilities to learn from data. Using machine learning algorithm, we aim to produce a model that will correct the typo of users and understand user's intent without typing the exact keyword.Typo itself is, Typing errors occurs when the typist or the author knows the actual and correct spelling of the word but mistakenly or by slip of finger presses an invalid key [3]. User intent is the identification and categorization of what a user intended or wanted when they typed their search terms [4]. To correct the typo, we will use N-gram technique. An N-gram is an N-character slice of a longer string, N-gram-based matching has had some success in dealing with ASCII. If we count N-grams that are common to two strings, we get a measure of their similarity that is resistant to a wide variety of textual errors [5]. In our system, we use N-grams of several different lengths simultaneously and compare the results of the N-gram to correct the typo. The user will have chatting experience such that it looks like doing chat with real person.

Based on the introduction, we identify some problems, does this application help owner of Small Medium Enterprise? does the user feel more convenient with our application using Artificial Intelligence technique? are the evaluations for the technique good enough to satisfy the user? We hypothesize that by implementing artificial intelligence technique, such as N-gram and machine learning will help the chatbot become convenient and easy to use.

The objectives of this research are: (1) to succeed in correcting the typo from the user's input based on our pre-set keywords, (2) to implement machine learning to define the user's intent after the user types something, and (3) to create an application that can help Small Medium Enterprises owner

Manuscript received October 22, 2018; revised April 15, 2019. This work was supported by Bina Nusantara University.

The authors are with Computer Science Department, School of Computer Science, Bina Nusantara University, Jakarta, Indonesia 11480 (e-mail: mikael.tedjopranoto@binus.ac.id, andreas.wijaya002@binus.ac.id, levi.santoso@binus.ac.id, dsuhartono@binus.edu).

retrieves information about their business. The benefits of this research are: (1) trained machine learning architecture can be used for another similar chatbot, and (2) Small Medium Enterprises owner can use this application to monitor their business.

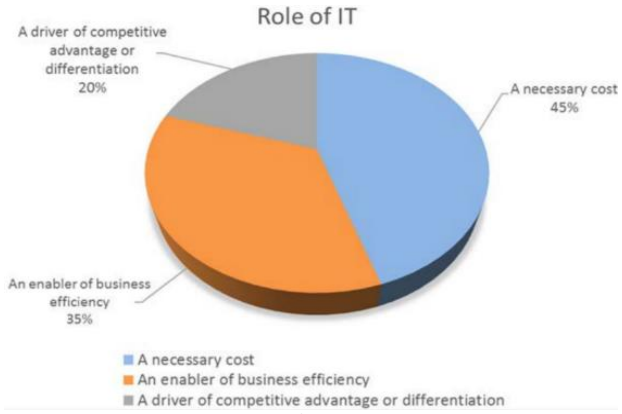

Fig. 1. SMEs survey of IT services (IDC, 2017).

For objective evaluation, we provide N-gram score table for every related keyword and the accuracy of the output. For schema matching, we provide top five of our users' input and count the accuracy, precision, recall and F1 score. They are calculated based on True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). True positive is number of correctly predicted data as positive values which means the value of actual class is "Yes" and the value of predicted class is also "Yes". True negative is number of correctly predicted data as negative values which means that the value of actual class is "No" and value of predicted class is also "No". False positive is "No" for actual class and "Yes" for predicted class. False negative is "Yes" for actual class but "No" for predicted class. Accuracy is simply the ratio of correctly predicted observations, it is the number of correct prediction divided by total number of prediction [6]. Recall is the number of retrieved relevant items as a proportion of all relevant items. Therefore, recall is a measure of effectiveness in retrieving performance and can be viewed as a measure of effectiveness in including relevant items in the retrieved set. Precision is the number of retrieved relevant items as a proportion of the number of retrieved items. Precision is, therefore, a measure of purity in retrieval performance, a measure of effectiveness in excluding non-relevant items from the retrieved set. The F1 score is the weighted average of precision and recall. Therefore, this score takes both false positives and false negatives into account. It works best if false positives and false negatives have similar cost [7].

This research writing systematics is divided into 5 chapters. Chapter 1 explains the background, identification of problems, scope of research, uses and objectives of the research, research methodology, and writing systematic. Chapter 2 elaborates several previous related methods that is done by other researchers. Chapter 3 describes the explanation on how we run our experiments. Research framework is shown here. Chapter 4 shows and explains the experiment results, user interface, score of N-gram model and result of schema matching. Finally, chapter 5 provides conclusion, analysis of the previous chapter, and suggestions that will help further development of the systems.

II. Method Comparison

SuperAgent is a powerful customer service chatbot leveraging large-scale and publicly available e-commerce data. Nowadays, large e-commerce websites contain a great of in-page product descriptions as well as user-generated content, such as Amazon.com, Ebay.com and many others. Take an Amazon.com product page as an example, which contains detail Product Information (PI), a set of existing customer Questions & Answers (QA), as well as sufficient Customer Reviews (CR). This crowd-sourcing style of data provides appropriate information to feed into chat engines, accompanying human support staff to deliver better customer service experience when online shopping [8].

Fig. 2 shows the system overview of SuperAgent. As the figure shows, when the product page is visited, SuperAgent crawls the HTML information and scrapes PI+QA+CR data from the webpage. Given an input query from a customer, different engines are processed in parallel. If one of the answers from the first three engines has high confidence, the chatbot will return the answer as response. Otherwise, the chit-chat engine will generate a reply from the predefined permitted response sets.

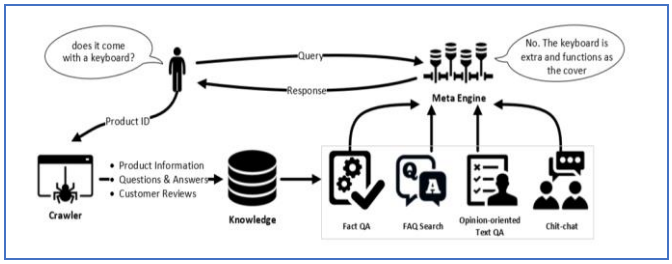

Fig. 2. The system overview of SuperAgent.

SuperAgent uses regression forest model to choose between their techniques while our chatbot will automatically use between n-gram or schema matching technique after knowing the users' intention. N-gram is "an n-token sequence of words" [9]. N-grams originate from the field of computational linguistics. Schema matching refers to problem of finding similarity between elements of different database schemas. Schema matching uses machine learning techniques to find the correct equivalence between the input schemas [10]. For the training system, we use K-Nearest Neighbor (KNN). KNN algorithm is a method for classifying objects based on closest training examples in the feature space [11].

III. Research Methodology

We use a simple system for our chatbot, we set a certain keyword to let users proceed to another state by typing the keywords. The keywords are set according to the user's state, category, and intent value. To check the intention, category, and state of a user, our system will connect to the database where all information of users and data related to the business are stored. If our system detected a new user, the system will add new user information in the database with zero intention value, null category value, and zero as state value. All data that business bot shows is provided by the database, which is the data from the cashier machine or manually input. Detail of business bot method is shown in Fig. 3.

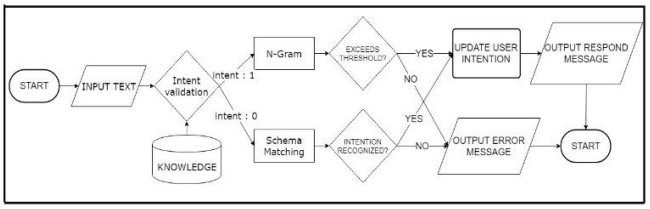

Fig. 3. Business bot method.

Upon receiving an input which is in form of a text sent by a user through LINE, our system will check whether the user's intent value is 0 or 1, depends on the input that already settled. If a user's intent value is 1, the input will run in n-gram engine. Otherwise, the input will run in the schema matching package engine called Flexmatcher. We use machine learning technique inside Flexmatcher with KNN Algorithm. The details inside n-gram and schema matching will be mentioned further later in this chapter. If the users' intention is recognized, our system will update the users' information according to their intention.

For n-gram, our system will constantly check a user's information on the database every time the user sends a message to the chatbot. N-gram will run if the value of the user's intent value is 1. Our system will use trigram (n-gram with n=3) to calculate the similarity between the keywords and user's input. The first step of the N-gram engine is to receive the input, state, and category. Then, N-gram engine will validate the state and category. After validating state and category, certain keywords are chosen to be compared with the input for example:

1. State: 1; Category: Laporan, keywords that will be compared are: "pemasukan" (income), "pengeluaran" (outcome), "report", and "tambah pemasukan" (add income).
2. State: 1; Category: List, keywords that will be compared are: "stok" (stock), "pemasok" (supplier), "pelanggan" (customer), "bahan" (material), "tambah stok" (add stock), "tambah pemasok" (add supplier), "tambah pelanggan" (add customer), and "tambah bahan" (add material).
3. State: 2, keywords that will be compared are: "bulan" (month) and "tanggal" (date).

After comparing all keywords with the input, n-gram will save the highest value. If the value that n-gram saved exceeds the threshold that we set which is 0.25, then the system will send the keyword back to the system.

To calculate the N-gram we use the following formula:

$$\frac{(X-Y)}{X} \quad (1)$$

where X is the total number of distinct n-gram across the two strings and Y is the number of characters (duplicate characters counted) not shared by the two strings. We set 0.25 as threshold because this is the most logical calculation. It will correct one letter typographical error in a less than four letters word, and two letters typographical error in a word less than 7 letters.

Upon sending back a keyword, our system will update the user's state and category in accordance with the keyword.

If the user's intent value is 0, schema matching engine will be called to process the input data. Schema matching is a package in Python which handles the problem of matching multiple schemas to a single mediated schema. Before our system process the input, there are few things that need to be prepared:

1. **Make dummy datasets in an array or more as a foundation to train**

For schema matching, we need few words as foundation to train for. We will prepare dummy datasets containing commands and keywords that will be trained in schema matching engine. We will collect the data by group discussion about commands and keywords that are related to the business. Given multiple dummy datasets listing every command related to business, we must set the mediated schema in the first array of every dummy datasets.

2. **Set headers and arrange the training datasets**

In this step, we must turn the previous datasets of array into tables and fill the first column with the mediated schemas of the datasets. The number of tables made is according to the amount of the datasets

3. **Specify headers into name classifiers**

The next step is to determine headers of the tables into name classifiers. We manually specify every header into the name classifiers and inserted into an array variable. Headers with the same meaning or category but different words will be classified into the same name classifier. The utility of these name classifiers is to recognize the input from the prediction variable

4. **Create and map the prediction dataset**

After classifying the headers, we will make the prediction dataset to predict the intention. The prediction dataset will be mapped exactly like the other datasets. Like the other datasets, prediction dataset has to be an array.

5. **Combine training data into an array variable**

After the previous step, we combine the training datasets that have been converted into tables into an array variable called "schema_list".

6. **Prepare, train, and retrieve the data**

The next step is to prepare the data to train. A schema matching package called Flexmatcher will be used to train the data. Before training the data, we have to make a variable containing the training data, ".train()", for example:

- **data.train()**

where "data" is the training data that was prepared in the previous step right before ".train()" is called. After the data is trained, we will retrieve the prediction by using "make_prediction" command on the mapped prediction table.

- **predicted_mapping = data.make_prediction(data3)**

where "predicted_mapping" is the variable made to contain the prediction and "data" is the training data, and "data3" is the prediction table. To retrieve the name classifier of the prediction, we just have to print "predicted_mapping" array of "data3" header.

- **print(predicted_mapping["kata"])**

where "kata" is the header of prediction table. The predicted name classifier of the header input will be printed after "print" is called.

After all these steps, our system can run this schema matching engine and retrieve the prediction by sending the input text from the user. However, there are few cases that

schema matching engine received an unpredictable or unrecognized input text. Therefore, we ask our system to save the unrecognized input to an error log. Then, our system will respond with a message telling the user that it did not recognize the input and ask the user to send another input.

Every unrecognized input text from users will be filtered and manually added to training datasets. First, our system receives unknown users' input. For example, the user sends "income" and our system sends the input to the schema matching engine to process the word "income". The engine does not understand what that word is and gives an error message. The unknown users' input will automatically be stored in error log.

However, if the engine understands or recognizes that input, it will respond with a message and update the user's state, category, and intent. To update user's information, our system will need the prediction from schema matching engine. After updating user's information, our system will push a text message to the user according to the user's current intent, state, and category. After proceeding from state to state, there will be a time where there are no more state to continue, our system will automatically update the user's information to default. Where intent and state values become 0 and category becomes NULL.

IV. Result and Discussions

In this chapter, we explore more about what result that we got. We use dummy data for our testing, which is suitable for clothing business. We attempted to experiment with 2 different methods: N-gram and Schema Matching. We will show the numbers that we got from those experiments. We choose N-gram and Schema Matching for this research, because of how useful N-gram and Schema Matching for the user to experience chatting with another human while using the bot. Bahasa (Indonesia language) is used for the whole experiment. For each Bahasa mentioned in the examples, we provide its English inside the brackets afterwards.

For the N-gram method, we used $N = 3$ namely trigram, and it was categorized as good if the score was higher than 0.25. We use trigram, because it may be identified by the set of the characters they contain, and completely ignore the ordering information.

For Schema Matching we use Flexmatcher and pandas as our plugin, which include several examples of texts or sentences that might be typed by the users also from the respondents that already tried to use this bot. Flexmatcher is a Schema Matching package in Python which handles the problem of matching multiple schemas to a single mediated schema.

We use Flexmatcher to calculate the user's input value and pandas to make a table consisting our array of data using pandas data-frame. So, after the bot received the user's input, it can determine what is the user's "real" intention and then place the user in a specific state.

Table Ipresents the result for correct the typographical error. To measure the user intention, we used some of our users input as our sample inputs for this result. Table I provides the accuracy, precision, recall and f1 score based on those sample inputs. We attempted for each input for 5 times.

TABLE I: Result for N-Gram

Word	N=3		
	Typo	Keyword	Accuracy
Laporan	1 Typo	Laporan vs Laporin	Pass
	2 Typo	Laporan vs Lapurin	Not Pass
	3 Typo	Laporan vs Lepurin	Not Pass
List	1 Typo	List vs Lust	Pass
	2 Typo	List vs Ludt	Not Pass
	3 Typo	List vs Luds	Not Pass
Pemasukan	1 Typo	Pemasukan vs Pemasukin	Pass
	2 Typo	Pemasukan vs Prmasukin	Pass
	3 Typo	Pemasukan vs Prmssukin	Not Pass
Pengeluaran	1 Typo	Pengeluaran vs Pengeluarsn	Pass
	2 Typo	Pengeluaran vs Pengeluarsm	Pass
	3 Typo	Pengeluaran vs Pnngeluarsm	Not Pass
Report	1 Typo	Report vs Raport	Pass
	2 Typo	Report vs Rapory	Not Pass
	3 Typo	Report vs Rapery	Not Pass
Stok	1 Typo	Stok vs Stak	Pass
	2 Typo	Stok vs Stal	Not Pass
	3 Typo	Stok vs Srak	Not Pass
Pemasok	1 Typo	Pemasok vs Pemasuk	Pass
	2 Typo	Pemasok vs Pemasul	Pass
	3 Typo	Pemasok vs Prmasul	Not Pass
Pelanggan	1 Typo	Pelanggan vs Pelanggin	Pass
	2 Typo	Pelanggan vs Pelanggim	Pass
	3 Typo	Pelangang vs Pelangcen	Not Pass
Bahan	1 Typo	Bahan vs Bahin	Pass
	2 Typo	Bahan vs Bagin	Pass
	3 Typo	Bahan vs Begin	Not Pass
Tambah Stok	1 Typo	Tambah Stok vs Tembah Stok	Pass
	2 Typo	Tambah Stok vs Tembih Stok	Pass
	3 Typo	Tambah Stok vs Tembih Stak	Not Pass
Tambah Pemasok	1 Typo	Tambah Pemasok vs Tembah Pemasok	Pass
	2 Typo	Tambah Pemasok vs Tembih Pemasok	Pass
	3 Typo	Tambah Pemaosk vs Tembih Pemasog	Pass
Tambah Pelanggan	1 Typo	Tambah Pelanggan vs Tambih Pelanggan	Pass
	2 Typo	Tambah Pelanggan vs Tembih Pelanggan	Pass
	3 Typo	Tambah Pelanggan vs Tembih Pelanggin	Pass
Tambah Bahan	1 Typo	Tambah Pelanggan vs Tambih Bahan	Pass
	2 Typo	Tambah Pelanggan vs Tambih Bahsn	Pass
	3 Typo	Tambah Pelanggan vs Tembih Bahsn	Pass

TABLE II: Average and F1 Score from Some of the User's Input

Keyword	ACCURACY	RECALL	PRECISION
List	100%	100%	100%
Laporan	80%	80%	100%
Pemasukan	100%	100%	71%
Pengeluaran	100%	100%	100%
Report	100%	100%	100%
Tambah Pemasukan	80%	80%	100%
Stok	100%	100%	100%
Pemasok	100%	100%	100%
Pelanggan	100%	100%	100%
Bahan	100%	100%	100%
Tambah Pemasok	100%	100%	100%
Tambah Stok	100%	100%	100%
Tambah Bahan	100%	100%	100%
Tambah Pelanggan	100%	100%	100%
Average	**97%**	**97%**	**98%**
F1 Score		**98%**	

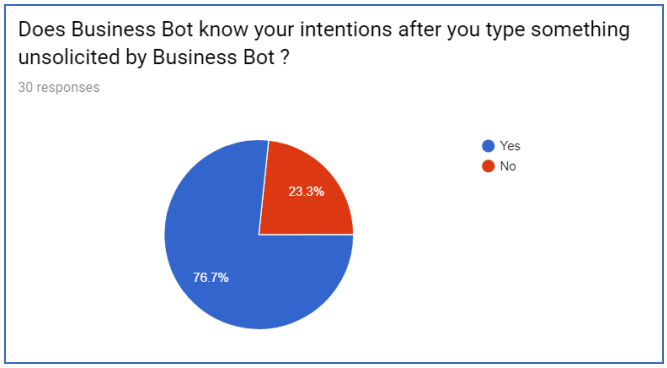

Fig. 4. Flexmatcher functionality for respondent's input.

For evaluating user's satisfaction, we used subjective

evaluation. In this case, we use the questionnaire from Google Form. We use Google Form because of the simplicity, respondent's familiarity, and how easy to share the questionnaire to our targeted individuals. For our targeted individuals, we involve university students, store owners, and working people. The questionnaire is comprised of 7 questions about our chatbot, but we will only show 3 of them because other question is not quite relevant with this research. We shared the questionnaire after we showed our application and asked the user to try it by him/herself. In the questionnaire, we told them to act as if they have a cloth business and answer the question based on their role. We got a total of 30 responses for our subjective evaluation.

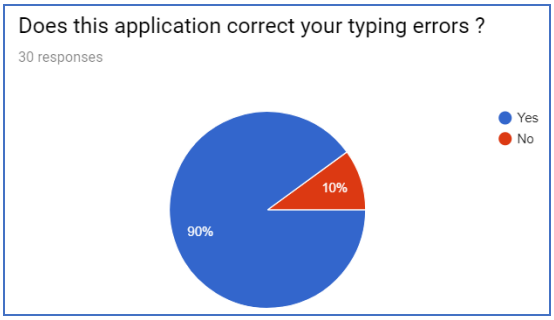

Fig. 5. N-gram functionality on respondent's input.

As presented in Fig. 4 out of 30 responses, 76.7% respondents stated that Business Bot knew their intentions after they typed something unsolicited by Business Bot. The questions are about our N-gram and Flexmatcher functionality in our chatbot, and most of the time our chatbot knows the respondent's intention even if they type something unsolicited. In Fig. 5, out of 30 responses, Business Bot successfully corrected 90% of the respondents typing errors. As we can see from the chart, our N-gram works well even though some errors still occur. In Figure, 6 out of 30 responses, number of letters that were corrected by Business Bot until it did not fix respondents' words anymore are: 1 Letter – 13.3%, 2 Letters – 43.3%, 3 Letters – 20%, More than 3 Letters – 23.3%.

For the discussion, the first one is about the chit-chat conversation model in the chatbot. Our system has not understood a chit-chat conversation input like "How are you?" or "I love you". We have done our research about chit-chat model, and in the SuperAgent's paper, they used seq2seq model trained on twitter conversation.

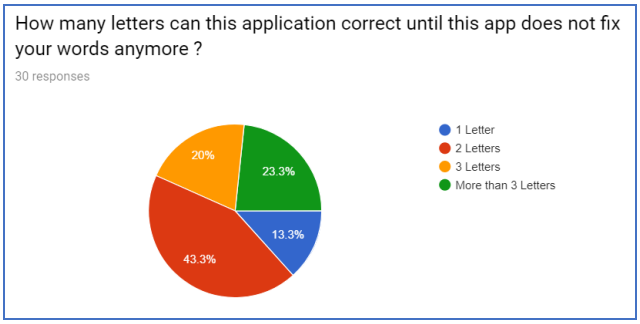

Fig. 6. N-gram and Flexmatcher error on respondent's input.

They selected 5 million most frequent non-duplicate short replies as the permitted response set. Most of which are greetings and common replies. To understand that kind of input, they need to have some deep learning technique, which we cannot perform it yet because our application database and capability is limited.

For our further improvement, we will try to implement deep learning for our chatbot. Chit-chat conversation model is a crucial point in every chatbot, because it makes the bot more human- like.

The second one, is an opinion oriented based model in the chatbot. Our chatbot does not understand an opinion-oriented input like "What do you think of this report?". Our chatbot is limited to only understand the intent of the users, because to understand opinion-oriented input, the chatbot needs to review rich information for different aspects of the report from the user's perspective.

SuperAgent's use hybrid approach method to extract the aspects from review sentences, and sentiment classifier to determine the polarity of the sentence regarding the specific aspect mentioned. We think opinion oriented based input is important for our chatbot, so the bot can give any advice or a good suggestion to the users. In conclusion, both of those points are important to achieve. However, due to our limitation in platform and databases, we cannot perform those two models.

V. Conclusions and Suggestions

Our research tried to do a new approach to chatbot by implementing machine learning and n-gram library. For N-gram, the summaries are, every time the user types something, our bot will automatically have an N-gram score. Our threshold for N-gram model is 0.25, if the score is above 0.25, the bot will be able to correct the typographical error and if the score is less than 0.25 the bot will not be able to correct the typographical error. N-gram score in every keyword is not balanced, because sometimes if we expect ten or more than ten letters in a word, the N-gram model will correct more than three miss types. It is weird, because if the users do more than three miss types, the input would be so different than the keyword that we expect.

There is a menu that expects "Tambah Bahan" (add material) keyword, but if the user type "Tenbah Bihan", the bot will correct the typographical error, because the score is more than 0.25.

The accuracy for one typographical error is 100%, two typos are 76%, and three typos are 41%.

By using machine learning, specifically the schema matching technique, our chatbot can understand the users' intent. We use top five of our users' input to count precision, recall, and F1 score. Given the keywords in our system is X, the top five inputs are : "minta X dong"(give me X please), "X please"(X please), "tampilkan X"(show me X), "permisi, tolong tunjukkan X"(excuse me, please show X), "halo, tampilkan X dong"(hello, show X please) The precision for our five top most asked question is averaging at 97%, 97% for recall, and 98 % for the F1 score. But the bot will not be able to understand something that is not trained yet. The user interface is modified to help Small Medium Enterprises owners to help their business report. Our chatbot does not understand chit-chat and opinion-oriented input, for example,

"How are you?" or "Is this good?".

For the suggestion on this research, we faced a lot of challenges, starting from the integrate Python with PHP, and LINE API, imbalance score of N-gram, lack of training data for the machine learning, and the limitation of our computing power to implement the deep learning. Some suggestions that we can give to future researchers are:

1. To make the N-gram score to be balanced, we suggest to set a different threshold for every keyword. The score would be more balanced, and it will be able to correct the typo properly.
2. To further improve the performance of the machine learning, every users' input needs to be automatically trained every time the user input something. For now, we train the data manually from our error log.

Other thing that we learn is to make sure that the bot should have understood chit-chat and opinion-oriented model. The next researcher can use word embedding technique, which requires deep learning and good computing power to develop it. The bigger the training data, the bigger the required computing power the model will need, so make sure to consider the computing power that you have, and maybe renting a cloud computing is a good idea to run a big training data like word embedding model or generative model. We thought of this because chit-chat and opinion-oriented model make the bot more human-like.

REFERENCES

[1] S. A. Abdul-Kader and J. Woods, "Survey on chatbot design techniques in speech conversation systems," *International Journal on Advanced Computer Science and Applications*, vol. 6, no. 7, 2015.

[2] B. A. Shawar and E. Atwell, "Chatbots: Are they really useful?" *LDV-Forum 2007 – Band*, vol. 22, no. 1, pp. 29-49.

[3] Z. Bhatti, I. A. Ismaili, A. A. Shaikh, and W. Javaid, "Spelling error trends and patterns in Sindhi," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 3, no.10, 2012.

[4] B. Jansen and A. Spink, "Determining the user intent of web search engine queries," *WWW*, Banff, Alberta, Canada, May 8-12, 2007.

[5] K. Hornik, P. Mair, and C. Buchta, "The textcat package for n-gram based text categorizatiton in R," *Journal of Statistic Software*, vol. 52, no. 6, 2013.

[6] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS ONE*, vol. 10, no. 3.

[7] L. Egghe, "The measures precision, recall," *UHASSLET*, vol. 4, 2008.

[8] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Zhou, "SuperAgent: A Customer service chatbot for e-commerce websites," in *Proc. ACL*, System Demonstrations, 2017, pp. 97-102.

[9] D. Jurafsky and J. Martin, *Speech and Language Processing*, New Jersey: Upper Saddle, 2009.

[10] C. Chen, B. Golshan, A. Halevy, W. C. Tan, and A. Doan, "BigGorilla: An open-source ecosystem for data preparation and integration," *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, pp. 10-22, 2018.

[11] S. B. Imandoust and M. Bolandraftar, "Application of K-nearest neighbor (KNN) approach for predicting economic events," *Theoretical Background*, 2013.

Mikael L. Tedjopranoto was born in Bekasi, Indonesia on October 24, 1996. He has completed his bachelor degree in 2018. He majored in computer science at Bina Nusantara University.

He currently works as an assistant channel development manager at Unilever Indonesia. Previously he interns as a web developer at Bank Central Asia.

Mr. Tedjopranoto joined a student organization called BSSC (BINUS Square Student Committee) for 1 year.

Andreas Wijaya was born in Medan, Indonesia on November 17, 1996. He has completed his bachelor degree in 2018. He majored in computer science at Bina Nusantara University.

He currently works as a full stack developer at a wellness rewards and gamification platform called Lyfe. Previously he interns as a Web developer at a technology consultant called Sintech.

Mr. Wijaya joined a student organization called BSSC (BINUS Square Student Committee) for 2 years.

Levi Hanny Santoso was born in Jakarta, Indonesia on October 18, 1996. He has completed his bachelor degree in 2018. He majored in computer science at Bina Nusantara University.

He currently works as a software developer at PT. Adrena Solusi Insan Muda. Previously, he interns as a Web developer at PT. Media Bintang Indonesia.

Mr. Levi joined a student organization called BNCC (Bina Nusantara Computer Club) for 2 years.

Derwin Suhartono was born in Pekalongan, Indonesia on January 24, 1988. He has completed his bachelor in 2009 and master degree in 2012 majoring in computer science from Bina Nusantara University. He got his PhD in computer studies from Universitas Indonesia in 2018.

He currently serves as the head of Computer Science Program, Bina Nusantara University. Previously, he played as a coordinator of intelligent systems field in the faculty. He has published around 35 international publications, 11 national publications, and 1 book entitled "Artificial Intelligence: Konsep dan Penerapannya". His interest is in natural language processing, machine learning and applied artificial intelligence.

Dr. Suhartono actively involves in INACL (Indonesia Association of Computational Linguistics). He takes a role as reviewer in several international conferences and journals.