Chatbot Application on Cryptocurrency

Qitao Xie[1], Qingquan Zhang[2], Dayuan Tan[1], Ting Zhu[1], Shen Xiao[3], Beibei Li[4], Lei Sun[5], Ping Yi[6], Junyu Wang[3]
[1]*Department of Computer Science and Electrical Engineering, UMBC, USA*, {qxie1, dayuan1, zt}@umbc.edu
[2]*University of Illinois, Urbana Champaign*, qingquan@illinois.edu
[3]*Hunan University*, xiaosheng@hnu.edu.cn
[4]*Carnegie Mellon University*, beibeili@andrew.cmu.edu
[5]*Shanghai Graduate School of Chinese Academy of Social Sciences*, lsun3@chicagobooth.edu
[6]*Shanghai Jiaotong University*, yiping@sjtu.edu.cn

***Abstract*—Many chatbots have been developed that provide a multitude of services through a wide range of methods. A chatbot is a brand-new conversational agent in the high-speed changing technology world. With the advance of Artificial Intelligence and machine learning, chatbots are becoming more and more popular. A chatbot is the extension of human interface mediums such as the phone and social platforms. Similarly, Cryptocurrency is a new extension of digital or virtual currency designed to work as a medium of exchange. In the current digital exchanging world, investors and interested parties are eager to know more information about, and the capabilites of, this new type of currency. One of the potential paths to retrieve the info automatically and quickly is through a chatbot. We explored the open source python library, Chatterbot, to apply Itchat API (a *WeChat* interface) with the aim of building a robot chatting application, I&C Chat, on the topic of cryptocurrency. First, we collected question and answer pairs datasets from Quora websites. Furthermore, we also created API calls to query the real time quote for the top 25 cryptocurrencies. Then we used the collected data to train our chatbot and implemented a logic adapter to receive the price quote of cryptocurrencies based on the incoming question. The Itchat API method will return the best matched answer to the asking party automatically. The response time of different questions has been investigated. The results imply that this application is quite useful, feasible and beneficial to the digital currency world.**

***Index Terms*—Chatbot, Cryptocurrency**

I. Introduction

The interaction in the format of speech or text between humans and computers is gaining more and more in popularity nowadays. People expect to have similar experiences when they talk to machines as when they talk to human beings. In order to provide suitable responses based on phrases or keywords taken from questions, a dialogue system or program is needed, which is often called a *chatbot* or a *chatterbot*. The chatbot is a computer program that has an ability to communicate with people by providing answers to questions. People input the natural language speech or text, while the program provides the most feasible intelligent response in the form of text or speech.

The history of chatbots started in the 1950s when computer scientist Alan Turing experimented with the concept of communicating like humans do. In the 1960s Joseph Weizenbaum published the *ELIZA* program [1]. At that time researchers aimed to examine whether these systems can imitate the human speech so that users would think that they are interacting with other people instead of computers. Technological advancement in recent decades, like faster computers, faster internet service, and the invention of smartphones, makes the information communication and sharing feasible in reasonable manners. People created machines and software to simulate a human-like conversation, which was called *ChatterBot* by Michael Mauldin in 1994 [2], shortened to *chatbot* later. Since then, the term *chatbot* has been used to refer to conversational agents that primarily use text-based or speech-based input. The idea of combining artificial intelligence and the interaction of natural form sped up the development of new chatbot platforms. Chatbots are immensely popular at the moment. Tech giants' core products such as *IBM Watson, Amazon Alexa, Apple Siri* are some examples of popular chatbots. Facebook created an online assistant named *M* and already has bots for small things, like booking a haircut or sending flowers, on its *Messenger* app. *IBM* created *Watson*, a question answering computer system, which proved its power by winning the popular quiz show *Jeopardy*. *Google* is also working on a chat-bot based app to answer user questions.

Another thing making chatbots popular is the ease with which anyone can create one themselves, without any programming skills needed. As of now, some of the popular chatbot platforms are: *Microsoft Bot platform, Pandorabots, Facebook Bots for Messenger*. Today chatbots are used in education, business, and in information retrieval processes. Today, commercial applications of chatbots include: travel agents, customer service support [3], technical support, personal assistants, education, marketing, entertainment, legal and financial fields, and so on. The communication between the companies and their clients has reached a new level of intensity. With the rapid development of technology, the customer experience is changing dramatically. Therefore, the use of chatbots in business can be a crucial issue of improving customer communication and customer support. Finally, information retrieval processes [4] in the legal and financial fields are gaining more and more attention.

There is no doubt that digital service and marketing are developing rapidly, with different platforms reaching the audience through things such as website, e-mail, instant messaging, and social networks. However, a new exciting communication

channel, chatbot, is playing the starring role. People already use it daily – *Siri* and *Cortana*, which are intelligent personal assistants in the form of chatbots used in Apple and Windows devices. With chatbot, you can easily deliver any information to the potential end-user by creating some pre-defined conversations.

Cryptocurrency is new to many people but it has been getting very hot recently. A cryptocurrency is a digital or virtual currency designed to work as a medium of exchange [5]. Different from the conventional currencies, a cryptocurrency uses cryptography to secure and verify transactions as well as control the creation of new units of a particular cryptocurrency. It was introduced in 2009 but did not grab mainstream media and people's eyes until 2012. It may have the potential to compete against traditional payment methods like cash or check, and other online payment method like credit and debit cards and *PayPal* [6].

The construction of a new chatbot steeped in the topic of cryptocurrency may prove more than just a novel project. On the one hand, people eagerly want to obtain information on the topic of cryptocurrency and a quick price quote. On the other hand, chatbot is a very good conversational agent to meet this requirement, which can provide the information in a human-to-human manner. Our goal in this study was to identify a technology for building a chatbot that fulfills the technical requirements and needs of today's exciting crypto-market. The ideal technology should meet the following requirements: be based on an open source platform, to be simple enough and easy enough to implement, and facilitate the support from an open source community. The selected technology should be deployed onto platforms that can be easy to access.

There are some cryptocurrency information bots in the market, but most of them are hosted on a specific platform such as Facebook messenger. Some of them also limit one-on-one chatting. For some bots, knowledge information is collected on the fly so that it slows down the response. As a dot on the horizon, we would like to have the chatbot that can answer any question, but for now we want to create an achievable chat system on this interesting topic. In this paper, we will propose our chatbot system on cryptocurrency, which can tell people at any moment what is happening in the crypto-market. The bot can do the following: 1) AI market, a bot can detect some patterns happening right now in the major cryptocurrencies and 2) people query, there are lots of discussions about the cryptocurrencies. We can filter out some good questions and answers from the popular sites such as *Quroa, Google*, or even *WeChat* accounts. We are continuously scanning all the conversations and announcements related to cryptocurrencies to find patterns that could be useful in the crypto-market. Questions and answers collected from different channels and real-time price quote APIs have been embedded into our chatbot system. Focusing on the cryptocurrency field, we designed and implemented our chatbot system to answer any users' any question on the cryptocurrency, from the real time price to definition of different knotty terms used in this brand-new field. Our cryptocurrency chatbot has the features of multi-platform deploying, group chat usage, knowledge-acquisition beforehand, and access to external resources.

The paper is organized as follows: in section II, we will introduce why we want to design and implement a chatbot system on the cryptocurrency topic, then in section III the details of each part of our design will be explained, section IV illustrates our evaluation and results, and we will conclude in section V.

II. Motivation and Background Study

People pay attention to cryptocurrency for a multitude of reasons. One of the most important motives is that in the past three years the market capitalization and volume have had a massive increase. As of May 6, 2018, the total market capitalization of cryptocurrencies is bigger than 450 billion USD and the record high daily volume is larger than 25 billion USD [7], which is almost 1400% bigger than it was in March 2017. Huge capitalization comes along with a huge profile margin. More and more people are attracted to this field in the hope of cashing in on the trend.

On the way of getting on board there are so many new terms and concepts in the cryptocurrencies field, as well as detailed operations corresponding to this topic, especially for beginners. Meanwhile that information on the internet is so scattered that it takes a long time to search and collect all of what you want. It would be very convenient if the quick and accurate answers to what we want to know were readily available. One of the best approaches is through a chatbot, a human-like conversation application.

According to the report [8], the chatbot interface has been very popular and maintains a steady growth. It has been used in Instant Messaging systems, such as *Facebook Messenger, Google Assistant, Amazon Alexa*, and so on. Indeed, Instant Messaging has more active users than any other social network or mail application [9]. Nearly 4 billion users are accounted for the top 10 IM platforms. The gradual worldwide acceptance of chat-based interfaces makes the ease of adoption and diffusion of newer technologies built on the top of the pre-existing platforms feasible.

The ease of integration is also a key motivation to developing platforms and frameworks which can synchronize chatbot applications on the field of cryptocurrency. Hence, we decided to implement a chatbot application on cryptocurrency.

A chatbot is basically a system of machine-to-human conversational interactions. The main goals are to solve the following:

1) How to get the most information from the input question?
2) How to process the question to get the best-fit response?
3) What is the best way to present the agent?

The most recently popular chatbots provide a free-form text input interface that allow users to "ask anything." This poses a big challenge in the development of chatbots. At least in the near future, chatbot performance depends heavily on the anticipation of what users may say to the agent. Therefore, it is critical for a successful chatbot to abstract the most important

information from the user input. Below we will discuss the mathematical concept and methodology used in our chatbot system.

A. Knowledge graph data structure

A knowledge graph data structure is a great way to store data [10]. When a data set is trained, the knowledge graph is generated for the necessary entries so that the inputs and their corresponding responses are tightly correlated. A knowledge graph can extract more meaning from a dialog to better realize how user intent relates to an application's data. A statement could have multiple responses, and a response could have multiple statements to be related with, illustrated in Figure 1. The data training stores that information into databases after the pre-loaded training data is processed. The chatbot's database knowledge graph, based on a list of statements representing conversation can be kept updated. During the conversation with chatbot, it is up to the chatbot to choose which best-fit response for the given user input to provide. In our case, we use the best match algorithm via calculating the Levenshtein distance, which we will discuss below in more detail.

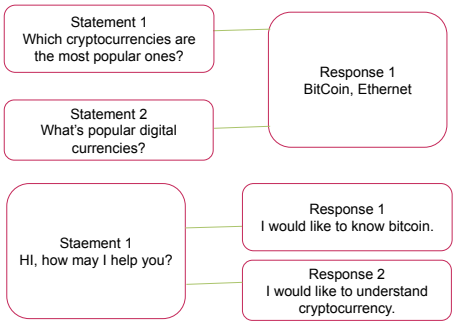

Fig. 1. Knowledge graph data structure.

B. Levenshtein distance

The Levenshtein distance, also known as edit distance, is defined as the distance value of describing the minimal changes that are required to modify one string (input) to another string (output), which is widely used in computer science, natural language processing, and information theory.

Mathematically, the Levenshtein distance can be calculated via dynamic programming as follows, where i is the length of string a, and j is the length of the string b.

If $min(i,j) \neq 0$,

$$lev_{a,b}(i,j) = min \begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+1_{(if a_i \neq b_j)} \end{cases} \quad (1)$$

Others,

$$lev_{a,b}(i,j) = max(i,j) \quad (2)$$

This method has been implemented in many realistic applications, such as handwritten character recognition [11], measuring dialect pronunciation differences [12], helping to assist natural language comprehension just like our method and so on.

The greater the Levenshtein distance, the greater are the differences between the strings. The similarity score between the two statements is based on the Levenshtein distance. The value is between 0.0 and 1.0. For example, from "chatbot" to "chatbot", the Levenshtein distance is 0 because the original and targeted strings are the same, and the sentence similarity score is 1.0. On the other hand, from the string "chatbot" to "chatbat", the distance is 1 since one substitution is needed to change from "chatbot" to "chatbat". and the sentence similarity score is 0.86. The sentence similarity score here is calculated based on the Levenshtein distance. The less the score, the less similar are the statements, as illustrated on Table 1.

TABLE I
SENTENCE SIMILARITY SCORES BASED ON THE LEVENSHTEIN DISTANCE.

Statement 1	**Statement 2**	**Similarity Score**
chatbot	chatbot	1.0
chatbot	chatbat	0.86
Where is the post office?	Look for the post office?	0.65
Price of btn	Price of etherum	0.71
dance	swim	0

C. BERT model

BERT, Bidirectional Encoder Representations for Transformers, was a big advance in the use of Machine Learning for the field of National Language Processing by the Google AI Language group in 2018 [13]. Traditionally language models consider the previous tokens and predict the next token. However, BERT is a language model considering the previous and next tokens when predicting. Especially in the question answering tasks, Jacob et al from the Google AI group found out, using BERT, learning two extra vectors marking the beginning and the end of the answer could pre-train a Q&A system. Ideally, BERT could be used to train on sentence pairs that take sentence pairs (question and answer pair) as input and then learn a specific embedding for the sentences to help the model distinguish them.

From Alex Wang and Kyunghyun Cho [14], the unnormalized log-probabilities can be calculated according to the equation below

$$\sum_{n-1}^{M} log\phi_m(x) \quad (3)$$

The probabilities can be used to find the most likely sentence according to the calculated probabilities

$$log\phi_m(x) = \begin{cases} 1h(x_m)^T f_\theta(x_{\setminus m}) & mask \notin (x_{1:m-1} \cup x_{m+1:M}) \\ 0 & otherwise \end{cases} \quad (4)$$

where $f_\theta(x_{\setminus t}) \in \mathbb{R}$, $1h(x_m)$ is a one hot vector with index x_m set to 1, and $x_{\setminus m} = (x_1, ..., x_{m-1}, [mask], x_{m+1}, ..., x_m)$.

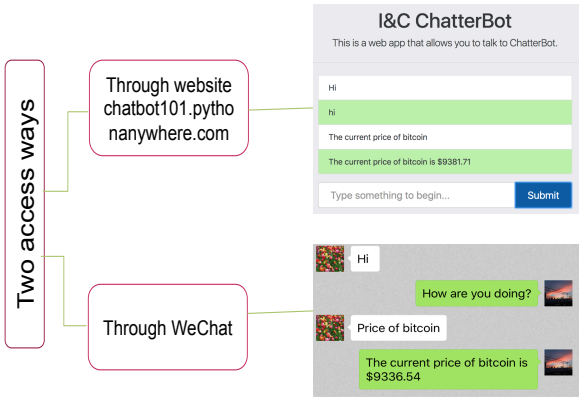

Fig. 2. The two ways to use Chatbot

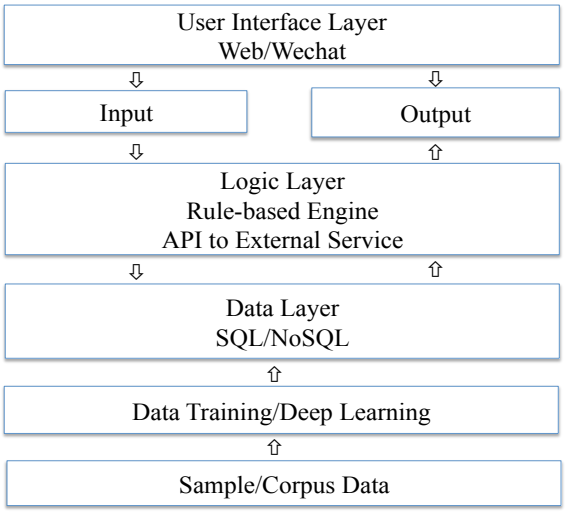

Fig. 3. A layer diagram showing our chatbot's architecture.

III. Design

Our chatbot system can provide all those information users may want in two ways. Figure 2 shows the two ways to use chatbot. One is using the web server to host chatbot, which is very convenient and familiar for most web users. The other approach is using WeChat combined with Itchat. WeChat is a very popular instant messaging tool which has more than 0.89 billion registered users and it has become a big part of people's personal social networking [15]. Itchat is an excellent open source WeChat personal number interface [16]. Using Itchat to call WeChat has never been easier. We developed a WeChat robot that can handle the information querying. Combined with Itchat/WeChat, Chatbot will help you to expand your social connections and facilitate your own working efficiency in the fields you are interested in.

Our chatbot accomplishes the stated goal through the following:

1) Collect cryptocurrency related popular questions and answers to build the dataset of Q&As.
2) Train the chatterbot using the datasets of the known statements and responses.
3) Process input in some logical ways. Using a number of machine learning methods to generate the responses. A search algorithm and a classification algorithm are deployed on logic adaptors.
4) Return the response. It could be one of several formats such as a console, API, speech synthesis, etc.

Figure 3 shows our chatbot's architecture. The logic adapter is very important here. You can use the pre-built data set to select the response via a matching logic method, which is based on the pre-built data set. The Chatterbot library [17] provides an easy way to train the chatbot through the datasets for the better responses. Another way to get the response in a logic adapter is to call an external API to get the required information. For example, to get the price quote for the cryptocurrencies, we implemented a way to call a coin-market API [8] for real-time price information. API calls can be embedded inside a logic adapter so that Chatbot can respond to the questions of price quote directly and automatically. We can use multiple logic adaptors in a chatbot system. For example, the best match adaptor uses a method to compare the input statement to the known statements. Once the closest match to the input statement is found, another method is implemented to select one of the known responses to that input statement. Another example of logic adaptors is a low confidence response adaptor. A specified default response is returned when a response cannot be determined above the threshold of confidence score. Below we discuss each step of building our chatbot in more detail.

A. Data collection

The Knowledge base is built through a series of questions and answers via *Quora* sites. Chatbots are only as intelligent as the knowledge they can have. Collecting enough training data used to train machine-learning classifiers for information retrieval [19] is crucial to achieving human-like interactions. We wrote a Python crawler program to collect multiple Q&As on the topic of cryptocurrency from Quora.

With Python Requests, Selenium, and Beautiful Soup modules in place, it is easy to scrape the web pages to gather the data for the chatbot's data training that you are working with. We use algorithm 1 to collect a dataset. The following steps are used for Q&A crawling on Quora:

- The first script is used to get questions given one keyword. In our case, since we focused on the field of crptocurrency, we chose the three popular words on the field of cryptocurrency, *"cryptocurrency"*, *"bitcoin"*, and *"ethereum"* as the keywords of Quora search. The questions collected from the script are saved in a text file. As you may know, Quora only shows the 25 questions for the given keywords at a page. Using Python's Selenium module, we are able to simulate the browser's behavior of the JavaScript to move down in the page to get more

questions on the Quora querying. This is very useful for web scrawling.

- The second script is used to call Quora to get the titles of the questions and their answers. The titles and the answers are stored in a special format file. To be used for the data training in our chatbot, the data format has to follow the one shown in the Table 2.

TABLE II
DATA TRAINING FILE FORMAT FOR OUR CHATBOT.

```
categories:
- cryptocurrency
conversations:
- - What is cryptocurrency?
  - Cryptocurrency is a medium of exchange, created
      and stored electronically in the blockchain
      ...
```

We can also collect more Q&As using Itchat [13], a robot WeChat application. We developed a python application using Itchat and chatterbot API [19] to automatically collect the questions in some cryptocurrency-topic WeChat groups. We will discuss more on Itchat application below.

For the real time price, we implemented a logic adaptor inside the chatbot application using a coin-market API to retrieve the real time price of the top 25 cryptocurrencies. Whenever the input statement has a keyword such as "price of bitcoin" or other cryptocurrencies, it will try to call the coin-market API to obtain the real time price of that cryptocurrency. Currently only the top 25 cryptocurrencies are supported for this functionality. We will expand the functions to support more cryptocurrencies in the future.

Algorithm 1: Collecting Data

input : KW: keyword
output: question/answer pair list

scrape data from quora site based on keyword;
$qList \leftarrow$ http://www.quora.com/search?q=KW;
for $q \in qList$ **do**
  $answer \longleftarrow GetAnswerFromQuoraq$;
  if *unanswered* **then**
    continue;
  $ProcessAnswer(answer)$;
  $qa\_list \longleftarrow (q, answer)$

Two-dimensional string arrays are applied to build a database. Rows in the array are used for requests and responses. All the even rows contain the requests or questions and all the odd rows contain the responses or answers. Columns in the array are applied to save different types of questions that could be asked by the user and responses that a Chatbot can offer. There is one row in the array which contains default responses which is used when the matched question is not found in the array.

B. Chatbot data-training

The open source library Chatterbot comes with a corpus data and utility module which makes it easy to quickly train the bots to communicate. The library has the built-in tools that help simplify the process of training a bot instance. The basic step of data training is loading a preset example dialog into the chatbot's database. This will create the special data structure that represents the known input statement and response. With the given data set, the data trainer class will create a knowledge base so that the known input and response are correctly linked. Further, we also specify the collected data set for data training to focus on the topic of cryptocurrency.

The data for chatbot's training has to meet the specified format if using the library's data training class. You can develop your own training class for your own data format. For example, if you want to use the data collected from Twitter, you can create a new app using your twitter account. Thus, you can train your chatbot using the data from Twitter.

TABLE III
ALL THE ANSWERS RELATED WITH THE STATEMENT.

```
Conversation 1:
  Question: "How can I help you?"
  Answer: "I would like to buy a movie ticket."
Conversation 2:
  Question: "How can I help you?"
  Answer: "I would like to buy a ticket for a movie."
Graph stored in database:
  Question: "How can I help you?"
  Answer:
    - "I would like to buy a movie ticket."
    - "I would like to buy a ticket for a movie."
    - .....
```

For the given input, we find out the best-matched statement explained below. Then we find out all the answers related to the statement and either pick the first answer or pick one of the answers randomly, as shown in Table 3.

Algorithm 2 below shows the data training step in our chatbot.

Algorithm 2: Data Training

input : question/answer pairs

load data into databases;
$q\_a\_list \leftarrow$ question/answer pairs;
for $(q, a) \in q\_a\_list$ **do**
  if q *non-exists* **then**
    create a new statement;
  if a *non-exists* **then**
    create a new response record;
  if a *exists* **then**
    increase the occurerence of the response;

C. Itchat application

Itchat is a complete and graceful API for the WeChat interface [13]. Itchat can be used to respond the text messages posted by the other users in the WeChat. The simple python code used for this purpose is in Table 4.

TABLE IV
ITCHAT SAMPLE PYTHON CODE TO TALK WITH WECHAT.

```
import itchat

@itchat.msg_register(itchat.content.TEXT)
def text_reply(msg):
return msg.text

itchat.auto_login()
itchat.run()

...
```

We implemented an application to combine Itchat and chatbot, using sqlites3 to save the records including questions and corresponding answers. It can have two purposes. First you can enlarge our question pool. Secondly, if the input is known, Itchat can return the preset response right away via chatbot's algorithm.

D. Input processing

Based on the different search algorithms, the closely matched response is returned with the highest confidence score. Inputs will be classified into three categories: greetings, price of cryptocurrency, and cryptocurrency related Q&As. It is a technique of artificial intelligence [20] used in the design of a Chatbot. The input is matched with the inputs saved in the database and a corresponding response is returned. Algorithm 3 is for best match logic adaptor.

To generate the response, a number of search algorithms and classification algorithm are used. For search algorithm, the following have been implemented: the similarity of an input statement to the known statement; the frequency in which similar known responses occur; the most likely category of an input statement. For classification algorithms, the chatbot uses naive Bayesian classification to determine if an input statement meets a particular set of criteria for the response to be generated from that logic adapter. Naive Bayes comes under supervised machine learning which is used to make classifications of data sets. It is used to predict things based on its prior knowledge and independent assumptions. It is a classification algorithm which makes the decision for the unknown data set. It is based on Bayes Theorem which describes the probability of an event based on its prior knowledge.

Typically, there are two types of Chatbot systems. One is a rule-based chatbot, and the other is an AI-based chatbot. The AI-based chatbot is more intelligent as it understands the natural language, but natural language processing is still a problem without satisfied solutions. The rule-based chatbot relies on the known rules. It is relatively limited depending on the size of knowledge base and the rules, but it is simple and can be easily deployed.

Algorithm 3: Best Match

input : statement
output: response

$statements \leftarrow$ statements with the known response;

for $statement2 \in statements$ **do**
  $confidence\_score \leftarrow$ Levenshtein distance between statement and statement2;
  if $confidence\_score$ *is higher* **then**
    $statement\_with\_highest\_score \leftarrow statement2$;

$responses \leftarrow$ get the responses with $statement\_with\_highest\_score$;

if *more than zero* $responses$ **then**
  get the first element from responses;

The chatbot we are building is more like a rule-based bot. We use the existing corpus data and collected data set for data-training. We also use the implemented API to get the information we want to know.

Another important feature of the chatbot is determining the context of the current user expression. Especially when the input has multiple meanings, it could reply on the history of conversation for the modern chatbot to figure out the best matched response.

For the best matched logic adaptor, it selects the response based on the best known match to a given statement. If the DB is too big, it will take a long time to search through all the statements. To speed up the best matched process, we introduced the confidence stopper. The confidence stopper uses the confidence score that can stop the search if the score is met. In our experiment, we adjust the number of the confidence stoppers to reach suitable results. Algorithm 4 is for the best match logic adaptor with a confidence stopper.

For the real time price, we implemented a logic adaptor inside the chatbot application using a coin-market API to get the real time prices of the top 25 cryptocurrencies. Whenever the input statement has a keyword such as "price of bitcoin" or other cryptocurrencies, it will try to call the coin-market API to query the real time price of that cryptocurrency. Currently only the top 25 cryptocurrencies are supported for this function. We will expand the functions to support more cryptocurrencies in the future.

E. Process flow

In chatbot, the logic adapter determines how to respond to the input statement. After one input is given, the program will compare it with each question by calculating a confidence score for each of them. Only one question statement which has highest confidence score will be selected as best matched question statement, and then the corresponding response statement to that best matching question statement will be offered as

Algorithm 4: Best Match with confidence stopper

input : statement, confidence_stopper
output: response

$statements \leftarrow$ statements with the known response;

for $statement2 \in statements$ **do**
  $confidence\_score \leftarrow$ Levenshtein distance between statement and statement2;
  if $confidence\_score$ *is higher* **then**
    $statement\_with\_highest\_score \leftarrow statement2$;
  if $confidence\_score$ *no less than* $confidence\_stopper$ **then**
    break;

$responses \leftarrow$ get the responses with $statement\_with\_highest\_score$;

if *more than zero* $responses$ **then**
  get the first element from responses;

the response to the input. A threshold of the confidence score will also be preset to make sure the best matching question statement is as good as we want - in this case, it should have a similar meaning to the input. If the response cannot be found in the database, or the confidence score is lower than the pre-set threshold, a default response sentence of "do not understand" will be offered as the return response to the input.

During the script work, if the designer wants the conversation to be as natural as it can be, open-ended questions have been proved to be a right choice. With the user typing whatever he wants to, the chatbot needs to be diligent in framing questions and processing responses. Moreover, when the chatbot does not come with AI brains, creating relevant responses to user query will become an ordeal. The designer would be better off in framing open-ended questions only when it is extremely essential. Further, open-ended questions can steer the conversation away from the end goals. Thus, it is very useful to keep its communications concise and understandable. One bot cannot cover everything since there are too many topics in the universe. That is why we only implemented logic adaptors to build the bot focusing on one small field or topic.

A chat flow is the way in which a user is answered by the chatbot. In our application, a user asks a financial question and, based on the characteristics of the question, he ends up in one of three chat flows: 1. Understand the question, 2. Maybe understand the question, 3. Don't understand the question. In flow step 1, the use receives an answer to his question. In flow step 2, the bot tries to formulate the question and returns the best matched answer. If the user's question cannot be matched (with a high enough confidence score), then the chatbot "doesn't understand the question," and it goes into flow step 3. In flow step 3, the default response or the original question could be returned to the user.

IV. EXPERIMENT

We developed two ways for deploying chatbot. One approach is through WeChat. Another approach is through a web app server. To access the chatbot via WeChat, we implemented it via the Itchat API application, which can reply to the users automatically as said earlier. To access it via a web app, we deployed the chatbot via Django, which is a free and open source web framework and can be directly integrated with Chatterbot.

Datasets for training on the topic of cryptocurrencies are generated by our python script. Knowledge Base (KB) consists of more than 10,000 Question and Answer pairs corresponding to cryptocurrency. KB also consists of English corpus data based on the English greetings and conversations. The whole dataset contains disjoint sets of conversations. Our model is fully end-to-end trained without any additional supervision other than the answers themselves. We also use an open-sourced BERT model to train our datasets.

V. RESULTS

The chatbot works by adopting pattern matching. It uses pattern matching to classify the input text and generates a suitable response for the user. Chatbot knows the answer only because the input is in the associated pattern. Similarly, chatbots respond to anything related to the associated patterns, but it cannot go beyond the associated pattern.

For each kind of questions, a unique pattern must be available in the database to provide a suitable response. Naive Bayes is the classic algorithm for text classification. For an instance, a set of statements are given for a particular class. With a new input sentence, each word is counted for its occurrence and is counted for its commonality and each class is assigned a score. The highest score class is most likely to be associated with the input statement.

Without data training, the response will always return the input statement back. Table 5 shows the different responses to the input "Most popular cryptocurrencies" based on the selected logic adaptor in the chatbot after data training. The Best match logic adaptor returns the response we intend. Since the price adaptor is for the real time price of the top 25 cryptocurrencies, the response for this input to the price adaptor will be unknown. Multiple logic adaptors (combining those three adaptors) will be ideal.

TABLE V
DIFFERENT RESPONSES BASED ON LOGIC ADAPTORS FOR THE INPUT "MOST POPULAR CRYPTOCURRENCIES".

	Response
Best Match Adaptor	"Bitcoin, Ethereum, Ripple, Litecoin, Monero, Ethereum Classic, Dash, Augur, NEM, and Waves."
Price Adaptor	"Most popular cryptocurrencies."
Low Confidence Response Adaptor	"I am sorry, but I do not understand."
Multiple Adaptors (combining the three adaptors above)	"Bitcoin, Ethereum, Ripple, Litecoin, Monero, Ethereum Classic, Dash, Augur, NEM, and Waves."

Our chatbot is able to return the real time price for the top 25 cryptocurrencies correctly. Table 6 demonstrates the response of the top 3 currencies respectively, depending on the input question. The testing shows the chatbot responds to the questions consistently. It has an averaged response time of 100ms. The performance is acceptable to most users.

TABLE VI
THE RESPONSES OF REAL-TIME PRICE QUOTE FOR THE TOP THREE CRYPTOCURRENCIES.

Input	Response
"Price of bitcoin"	"The current price of bitcoin is $8386.25"
"Price of ethereum"	"The current price of ethereum is $713.123"
"Price of ripple"	"The current price of ripple is $0.710319"

To get the optimal chatbot response in a reasonable time, we adjust the confidence stopper in the best matched logic adapter. The confidence stopper is between 0.0 and 1.0. When the confidence stopper is 1.0, it was able to achieve the best matched statement for the given statement. But in the big KB, it could take a long time to get the matched statement back. Thus, we have to decrease the confidence stopper to acquire the matched statement back faster. But the tradeoff is that the matched statement may not be a good one sometimes. After many tries, we found out that the best scenario for the optimal matched statement for a given statement is to set the confidence stopper between 0.70 and 0.80.

Ideally every question should get an answer in the chatbot. Unfortunately, sometimes the question may be unknown to the bot. When it happens, a default response may be returned as the answer after searching when a low confidence response adaptor is not set. If a low confidence response adaptor is set, the default response is returned. The threshold of the low confidence is configurable. In our case, we set the threshold to 0.75. It means that when the input match rate is lower than 75%, it will return the default statement as a response. Even though it is not ideal, our bot is limited to just on the topic of cryptocurrency and related knowledge. For the top 25 digital currencies, Chatbot is able to promptly provide the real-time price quotes. Aside from price quote questions, Chatbot can answer the questions successfully that are highly matched with the training data set.

Finally, as to using the BERT model to train our datasets, the results are partly promising. During our training, for the case corresponding to one question, we preselect 10 responses randomly, one of which is the correct response, it shows a very high average match rate (around 95%), but when we use a one-on-one question answering pair, the match rate is not ideal (lower than 40%). After analysis, there are two improvements that can be done on the BERT model in the near future: 1. the size of tokens is not enough. We use 64 tokens only due to our hardware limit. The results could be better if the tokens size is increased as input and 2. we can create datasets of multiple-on-one for the question answering pairs, which we assume could improve the question-answer match rate significantly.

VI. CONCLUSION

In this paper, we use the open source Chatterbot and Itchat to implement a special chatbot for the application on the field of cryptocurrency. We collect the data set for a knowledge base of cryptocurrency for data training on our chatbot. The testing shows our chatbot is excellent in responding to certain questions related with cryptocurrency. It is more convenient for the user's information acquisition especially when the user wants to follow the price trend of the cryptocurrency of interest. We will continue improving the logic adapter implementation of the response generation and enhancing the knowledge base on the cryptocurrency.

REFERENCES

[1] Weizenbaum, Joseph. "ELIZA—a computer program for the study of natural language communication between man and machine." *Communications of the ACM 9.1* (1966): 36-45.

[2] Mauldin, Michael L. "Chatterbots, tinymuds, and the turing test: Entering the loebner prize competition." *AAAI*. Vol. 94. 1994.

[3] Xu, Anbang, et al. "A new chatbot for customer service on social media." *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems.* ACM, 2017.

[4] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. "Introduction to Information Retrieval." *Cambridge University Press.* 2008.

[5] Ameer Rosic. "What is Cryptocurrency: Everything you need to know." *Blockgeeks*. blockgeeks.com/guides/what-is-cryptocurrency. Accessed Sept. 13, 2018.

[6] Gandal, Neil, and Hanna Halaburda. "Competition in the cryptocurrency market." 2014.

[7] "Cryptocurrency Market capitalizations." *Coinmarketcap.* coinmarketcap.com/api/. Accessed May 6, 2018.

[8] Manish Dudharejia. "Chatbots are the next big platform." *Entrepreneur.* www.entrepreneur.com/article/298600. Oct. 4, 2017.

[9] Princeton University. "About WordNet." *WordNet.* Princeton University. 2010.

[10] Bayu Setiaji, Ferry Wibowo."Chatbot Using A Knowledge in Database." *7th International Conference on Intelligent Systems, Modelling and Simulation* 2016.

[11] W. Mckitterick, "The Messaging App Report: How instant Messaging can be monetized," *Technical report, BusinessInsider*. 2015. 5, 6, 33, 97, 123, 2015.

[12] Oncina, Jose, and Marc Sebban. "Learning stochastic edit distance: Application in handwritten character recognition." Pattern recognition 39.9: 1575-1587, 2006.

[13] Heeringa, Wilbert Jan. Measuring dialect pronunciation differences using Levenshtein distance. Diss. University Library Groningen][Host], 2004.

[14] Jacob Devlin, Ming-Wei Chang, et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." ArXiv e-prints: 1810.04805v1, 2018.

[15] Alex Wang, Kyunghyun Cho. "BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model." ArXiv e-prints: 1902.04094v1, 2019.

[16] "2017 WeChat Users Data Report: There are more than 0.889 billion active users." *Sohu*. http://www.sohu.com/a/136382735_184641. 2017.

[17] LittleCoder, Tempdban, Chyroc. "ItChat – A complete and graceful API for Wechat." *Github*. github.com/littlecodersh/ItChat. Accessed Sept. 15, 2018.

[18] Gunther Cox."ChatterBot – Machine learning, conversational dialog engine". *Chatterbot*. chatterbot.readthedocs.io/en/stable/index.html. Accessed Sept. 15, 2018.

[19] Zhao Yan, Nan Duan, et al. "DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents." *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics.* Association for Computational Linguistics, 2016.

[20] Russell, Stuart Jonathan, Peter Norvig, John F. Canny, Jitendra M. Malik, and Douglas D. Edwards. "Artificial intelligence: a modern approach." Vol. 2. Upper Saddle River: Prentice hall, 2003.