A Literature Review On Chatbots In Healthcare Domain

Conference Paper · August 2019

CITATIONS
0

READS
986

4 authors, including:

Nivedita Bhirud
Vishwakarma Institute Of Information Technology, Pune
8 PUBLICATIONS **8** CITATIONS
SEE PROFILE

Subhash Tatale
Savitribai Phule Pune University
10 PUBLICATIONS **6** CITATIONS
SEE PROFILE

A Literature Review On Chatbots In Healthcare Domain

Nivedita Bhirud, Subhash Tataale, Sayali Randive, Shubham Nahar

Abstract: Research says 60% of visits to a doctors are for simple small-scale diseases, 80% of which can be cured at home using simple home remedies. These diseases mostly include common cold and cough, headache, abdominal pains, etc. They may be caused due to the changes in the weather, intake of improper diet, fatigue, etc. and can be cured without the intervention of a doctor. There are a number of chatbots which provide services for the healthcare domain. But the problem with these chatbots is that they only provide answers for general healthcare FAQs. That is, these systems are unable to provide a natural communication with the user just as a doctor can. Work is being carried out to enable the chatbots to communicate in a way similar to the communication carried out between two humans. That is, the user must experience the feel of communicating to a person and not to a bot. This makes the chatbot a virtual communicating friend of the user. This type of smart communication (usually used in healthcare counselling) can be achieved by inclusion of NLU, NLP and ML techniques in the conventional scripted chatbots. There are a number of domains wherein the smart chatbots provide their functionalities. This paper briefs about the chatbot system for the healthcare domain. Also, it specifies various NLU, NLG and ML techniques to be incorporated in the chatbot and the comparison of the same.

Index Terms: Chatbot, Healthcare Domain, ML (Machine Learning), NLG (Natural Language Generation), NLU (Natural Language Understanding), Smart Communication, Virtual Communicating Friend

1 Introduction

CHATBOTS are automated systems which replicate users behavior on one side of the chatting communication. They are mimic systems which imitate the conversations between two individuals. They provide a simulating platform for effective and smart communications with the user on the other end. They copy marketers, sales person, counsellors and other mediators and work to provide services that the above-mentioned people provide. There are wide ranges of chatbots catering in many domains some of them are as follows: business, market, stock, customer care, healthcare, counselling, recommendation systems, support system, entertainment, brokering, journalism, online food and accessory shopping, travel chatbots, banking chatbots, recipe guides, etc. The most famous chatbots like Alexa or Google assistant are the best examples that can be given for smart communicating chatbots. These are general purpose chatbots that provide services for all domains and are not restricted to a specific domain. There are also domain-specific chatbots which provide functionalities to the above-mentioned domains. Some of them are as follows: Botsify is a chatbot which helps developers to create smart Facebook Messenger Chatbots and is used to collect information from Facebook users. Imperson is a chatbot which helps developers to create business chatbots and provide customer care services. NBC is a chatbot which helps the newsreaders to navigate quickly through top headlines.

- *Sayali Randive has completed bachelor's degree in Computer Science from Pune University. E-mail: sayalirandive906@gmail.com*
- *Shubham Nahar has completed bachelor's degree in Computer Science from Pune University*
- *Nivedita Bhirud is Assistant Professor at Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune, India*

The above mentioned chatbots were the systems for business or market domains. The aim of this paper is to discuss the need and usage of chatbots in the healthcare domain. There are a lot of existing chatbots for healthcare domain serving different functionalities. Endurance is a chatbot which deals with users suffering from Dementia (disease). The chatbot Casper helps people suffering from insomnia to pass their nights which are sleepless due to loneliness. MedWhat is a question-answering chatbot which gives answers to basic healthcare FAQs and also provides information about various diseases and its symptoms. The problem with these chatbots is that they just provide monotonous answers to users' questions. They are not capable of establishing a smart communication with the user just as a doctor does. These systems are also not able to predict the problems (diseases) faced by the user. To know what the patient is suffering from, the system should be friendly to the user, so that the user can communicate all the problems faced by him to the system. The paper aims at proposing a chatbot system, which is capable of establishing a smart communication. This can be achieved by implementing a smart and a responsive chatbot which is able to communicate with the user just as another human can communicate. To make conventional chatbots function like virtual friends, techniques of NLU, NLG and ML require to be incorporated into the system. These techniques make the system more communicative in the natural language, proves fruitful for counselling, and can also be modelled for prediction of diseases.

2 Research Gaps

The chatbots used by the telecom and marketing sectors for customer service are scripted types of chatbots. They help the customers on some predefined customer care questions. Research is being carried out in making the conventional monotonous chatbots to be communicative, responsive and carry out the communication in a natural (conversational) language. This requires the inclusion of NLP and ML techniques in the system. There are a number of ways to do so. Selection of an appropriate method is based on the domain of the chatbot, the functionalities it intends to provide, the language of communication, the end user, etc. All the above-mentioned issues need to be considered while

implementing a chatbot system. Some of the methods are briefed about in the following section

3 Literature Review

Smart chatbots made up of NLU, NLG and ML engines have the following components:

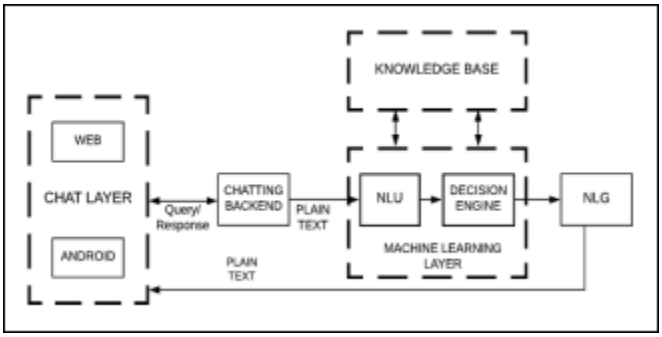

Fig. 1. *Architecture of Chatbot System*

3.1 A review on Chat Interface

This unit is the front end of the system. It is responsible for collecting the user queries from the user which are the input to the system. It is also responsible for displaying the system generated results to the user. Therefore, it can be said that the chat interface is the face of the system through which the entire communication takes place. It is the mediator of conversation between the system and the user. The query that user fires on the chat interface is passed on to the chatting backend which acts as a message delivering system between the Chat interface and the Machine Learning Layer. This interface can be accessed either as a website or as a smart phone app. The type of interface depends on the requirements of the user that are to be satisfied by the system. If the system is accessed from a smartphone, the interface will be in the form of an app and if the system is accessed from a website, then the interface will be in the form of a website. For building apps on the smartphone, it will require to use android for android phones or Swift for iOS. In this case, only the interfacing platform will be programmed on android and the complete backend processing of the system will take place on a server on which the system will be deployed. For making a website, either Java or Python web frameworks can be used. Java provides Spring and Struts as the most advanced and latest web frameworks. Similarly, Python allows usage of Django and Flask frameworks for building of a website. The criteria for selection of the programming language depends upon the functionalities that the system intents to provide, the requirements of the users that will use the system, the algorithms that are to be used by the system, etc. Selection of an appropriate programming language makes it simpler for developers to develop a system which provides maximum functionality to the user with high accuracy and minimum complexity.

3.2 A review on NLU Engine

NLU i.e. Natural Language Understanding is a subpart of NLP (Natural Language Processing) which enables the system to understand the natural language or the conversational language spoken by the users. The conversational language used by humans for day to day conversations is not as perfect as the formal language. It does not focus much on the vocabulary and the grammar. Hence, it becomes difficult for a system to understand the intent of the sentence. The input received from the user is in unstructured text format which cannot be understood by the system directly. It understands input only in structured formats. The unstructured text received from the user is converted to structured format by extracting important words and patterns from the user text using the NLU techniques. Humans are capable of understanding any mispronunciations, homophones, swapped words, shortened form of words (like ‘it’s’ for ‘it is’), slang words or phrases and also words which are not used in formal vocabulary but exist in regular conversations. NLU techniques enables the system to identify these twerks if the user makes use of them while conversing with the chatbot, so as to make the user feel that the conversation is taking place between two humans and not between a human and a bot. NLU systems do not directly understand the meaning of the user sentences. It involves a sequence of processes to derive the actual intent of the sentence. To understand a complete sentence, the NLU system needs to understand each word of that sentence. It means that the initial task is the segmentation of the sentences into individual words. Next, to understand the word, the system needs to understand the grammar of the sentence. This can be done by knowing the parts of speech of each word in that sentence. Here comes the POS (Parts-Of-Speech) tagger into picture. After knowing the grammatical weightage of each word, all of them are parsed to know the dependency among them. This is the most important step wherein the word with the highest dependency is extracted, from which the intent of the system can be known. It is not possible that the knowledge base would contain the exact sentence that the user has sent. It might contain a sentence with the same intent but with different words used in it. To match these types of synonymic sentences, synonym determination and sentence matching are required. The different tasks to be implemented under the NLU Engine and the methods to do the same have been discussed further.

3.2.1 A review on Word Segmentation

Segmentation, also referred to as tokenization is the process of splitting text into smaller and meaningful units. These units could be paragraphs, sentences, clauses, phrases, words or letters. The smallest unit are the letters. Word segmentation is the splitting of sentences into individual words separated by blank spaces. The tokenized units of the sentences are called as tokens. The tokenizers split the sentences into words and punctuations marks as independent units. The most commonly used tokenizer is of space type, i.e. it splits the sentences into words at the blank spaces. It is also required that the tokenizer should consider abbreviations, acronyms, dates, numbers in decimal formats, etc., which cannot split at punctuations and blank spaces, as they will lose their meaning if done so.

Mohammed Javed et al. [1] [2015] explained a method to implement word segmentation. He proposed in his algorithm to calculate character spaces in the sentences. The character spaces should include all types of gaps between characters.

They include the gaps between letter, punctuations and the words. The algorithm functions on the basis of the amount of gap or character space between each unit in the sentence. After the calculation of character spaces, an average of the gaps is calculated to know the mean average between characters in the sentence. This average gap distance is then applied to the sentence which is to be segmented. The places at which the character space is more than the average character space are said to be the points of tokenization. The gap between words is always more than the average gap and hence tokenization takes place at the blank spaces between words in the sentences.

Naeun Lee et al. [2] [2017] proposed the implementation of word segmentation using NLTK. Natural Language ToolKit (NLTK) is a python package which caters to provide services for NLP. It has inbuilt tokenizers. Users need to import the package and use the required type of tokenizer which is present in the form of functions. The NLTK includes a wide range of tokenizers which are as follows standard, letter, word, classic, lowercase, N-gram, pattern, keyword, path, etc. The most commonly used tokenizer is the word-punkt tokenizer which splits the sentences at the blank spaces. The accuracy, speed and efficiency of the NLTK tokenizers is commendable. Also, it does not require any algorithm implementation as the package executes them at the backend.

Tao Jaing [3] [2011] explains the usage of CRF (Conditional Random Fields) Algorithm for word segmentation. This algorithm trains the system for spaces between the characters. Using this training, the system identifies the gap between characters in the test sentence. The system keeps a threshold value for the gap distance. If the value of gaps in the test sentence is more than the specified threshold, then the sentence splits at those points. CRF requires a lot of training to be given to the system, which makes the process time consuming. Comparing the three methods illustrated above, the NLTK proves to be more efficient in all aspects as compared to the other two. The usage of NLTK does not require the implementation of any algorithm as everything is taken care by the package itself. Also, the accuracy, speed and diversity provided by the package is better than the two algorithms.

3.2.2 A review on POS Tagging

POS Tagging is the process of assigning grammatical annotations to individual words in the sentences. These annotations include the Parts-Of-Speech Tags. They denote the grammatical importance of the word in the sentence based on the dependency of that word with other words in that phrase, clause, sentence, paragraph, etc. The common POS tags are noun, verb, pronoun, etc. There are number of ways which can be used to perform the POS Tagging. Some of them are explained below.

Jerome R. Bellegarda [4] [2010] proposed a method called latent analogy for POS Tagging. In this algorithm, latent semantic mapping (LSM) technique is used. It requires the training on the available corpus. The LSM maintains a feature space of the trained corpus which has been tagged. Now, new sentences are provided to the LSM for tagging and the analysis is performed so as to determine the sentences from the training data which are closest to the test sentence. This is called as sentence neighbourhood. Sentence neighbourhood holds true for two sentences if they share the same intent matter. Once the intent matching sentences are found from the trained data, the POS tags attached to those sentences are then mapped to the test sentences.

Liner Yang et al. [5] [2018] put forth the technique of implementing the POS Tagger using Neural Networks. This algorithm consists of ‘n’ numbers of hidden layers. Theses layers are determined by the number of iterations or combinations required to tag the required sentence correctly. At each layer of the algorithm, each word in the sentence is tagged with an appropriate POS tag and then passed to the next later for checking the correctness of the tags. This keeps happening unless the next layer provides the same tags as provided by the previous layer. Another technique to implement the POS tagger is following the traditional approach i.e. of maintaining a dictionary of tags for the given language. Python NLTK provides an inbuilt Tagger which can be used just by importing the NLTK package. The NLTK has a pre-defined set of tags and a trained data of its own. It tests the sentence and applies an appropriate tag to it. On comparing the above three algorithms, the NLTK tagger proves to be speed and usage efficient. But highest accuracy is provided by the neural network algorithm as it undergoes many iterations.

3.2.3 A review on Dependency Parsing

A dependency parser is used to establish the relationship between words in a sentence based on the grammatical tags attached to it. It is the next step after parsing. A dependency tree or graph is created for every sentence. This tree is called as the parsing tree or the dependency tree. There are a number of ways by which the parsing can be implemented. The comparison of the same is expressed below.

Bo Chen [6] [2011] proposed a method for implementing the dependency tree. It initially finds out the dependencies among the words in the sentence. Each word is checked for its relationship or dependency with the other word. The word with the highest dependency is selected to be the root. The other words with a relation with the root node are attached to it as the child nodes. This keeps on continuing until all the words are placed in the tree. The tree form of the sentence is called the dependency parser tree. The dependencies among the words are found out by using the POS tags.

Zhenghua Li [7] [2014] provided a further improvised model of the dependency parser. In the traditional method mentioned above the parser creates a parsed tree for the required sentence. In the graph-based dependency parser, the tree created is converted to a graph where the words in the sentences are the vertices and the dependency between the words are the represented by the edges. This data structure shows a better representation of the parsed sentence. Parsing is always to be performed by the traditional method. But graph-based parser improves the visibility, readability and understandability of the parser.

3.2.4 A review on Synonym and Pattern Recognition

For information retrievals, no matter how big our data is, no sentence sent by the user can be perfectly same to any sentence in the database. But there can be sentences with the same intent. After understanding the intent of the user sentence, the database is checked for a sentence with the same intent. The matched sentences have difference of words which are used to express the same content. They use alternative words or synonyms. This makes synonym detection necessary for the system. Synonyms for a particular word may be domain independent or domain dependent. Domain independent synonyms are synonyms for a word in the entire vocabulary. But domain-dependent synonyms are synonyms for a word in that respective domain only. There are various algorithms used for the detection and extraction of synonyms, some of which are reviewed below.

LinHua Gao et al. [8] [2018] explains the traditional dictionary method of synonym extractions. In this method, the system database maintains a dataset of synonyms for important keywords in that domain. The sentence sent by the user is then mapped on to that synonym dataset. The keywords detected from the sentence are then checked in that synonym set to check for same intent. All possible synonyms of that keyword are then looked out for a match in the main database. The sentence which is closest to the user sentence is extracted. This method is time consuming and requires more of storage and complexity.

Sijun Qin [9] [2015] proposed a feature selection method for synonym extraction. In this method, among all the parts of speech tags, words having the tags as noun, verbs and adjectives are marked as positive tags and the others as negative tags. The polarity for each feature (word) is then carried out by using the POS tags. If the overall feature polarity is positive, then it can be identified categorically. All the positive features are then grouped together and the synonyms detection for the group of features will be relatively strong, as an entire clause is checked for its synonymic meaning. The synonym sets which are extracted for that clause of features is then calculated for information gain. The one with the highest information gain is the strongest synonym extracted.

3.3 A review on Decision or ML Engine

Scripted or monotonous chatbots have predefined replies to be given. They provide replies to the user from a set of predefined replies categorized on the basis of the query given by the user. Inclusion of ML in chatbots enables it to compute the replies from scratch. It is used to make predictions to predict the responses for the user queries and also to update the system from its experiences. It keeps updating the databases as and when it encounters something new from the user. This engine uses supervised or unsupervised or both techniques to analyze what the user requires. It further uses a model to interpret the intent of the user and provides the appropriate results. The results may be in the form of predictions or any form of analysis which are based on the execution and analysis of mathematical models. Most of the machine learning models are based on statistical and probabilistic evaluations of the instances occurring and the calculations makes a prediction for the test instance. The decision engine not only includes models for predictions, but also includes algorithms for information retrievals like entity extractions, multiple text classifications, etc. Also, the inclusion of a machine learning layer in a chatbot system, is used to create an ontological relationship for entities extracted, and also associate them with context-specific questions along with their alternatives, synonyms and machine-enabled classes. These features of machine learning, converts a static and basic FAQ system to a smart and more personalized communicating experience. For chatbots that provide services in diverse domains, the machine learning layer adds on to the services that it can provide. It intends to increase the accuracy of the responses provided by the system to the users and also extends the scope of the system. The system is enabled to update itself by learning from its experiences. This makes the system less prone to false predictions. The chatbots that are used in healthcare domain for disease predictions can use a wide range of algorithms, some of which are clustering, Bayesian networks, decision trees, etc. The methods of their execution and the comparison of the algorithms for the appropriate selection of the same is briefed here. A decision engine is the brain of the system. It includes the incorporation of ML algorithms for predictions, statistical and probabilistic calculations, etc. Also, ML enables the system to learn from its past experiences, so as to provide better and revised results. The chatbots for health care domain require disease predictions algorithm. Prediction can be carried out in many ways some of which are reviewed below.

Sachin S. Gavankar et al. [10] [2017] proposed the eager decision tree algorithm for prediction. This type of decision tree is the improvised version of the traditional decision tree. It creates this tree at runtime, based on the user's queries and keeps updating the tree on new user messages. Consider its working for disease prediction. In this algorithm, the symptoms detected in the user query are added as child nodes to the root node. The nodes keep on getting added for new symptoms detected. Further for every symptom, the algorithm checks for the second symptom which has the highest occurrence with the earlier symptom and asks the user for that symptom. If he says yes, then the system traces that path to check for the disease present at the root node. This will keep iterating for all users and the tree keeps getting updated for new entries or traces the path available.

Naganna Chetty et al. [11] [2015] put forth a fuzzy approach for predictions. In this algorithm, the system follows the clustering mechanism. It means that, the algorithm extracts that data from the knowledge base which is the closest to the user query. When the user fires a query, the algorithm searches for the best matches in the knowledge base and provides the same to the user. In the next iteration, when the user gives the second query, the best matches are further searched for relevance. Each query of the user, filters the matches on every iteration. This keeps on continuing until a single best match is found and that match is provided to the user as the result of prediction. Comparing the two algorithms we come to know that prediction using fuzzy logic (clustering) is easy to implement and involves less complexity. On the

other hand, eager decision tree algorithm involves more complexity and requires more time for execution. But the accuracy provided by eager decision trees is more as compared to the fuzzy approach.

3.4 A review on NLG Engine

NLG performs the reverse task of NLU. It is the process of converting the system produced results into natural language representations which can be easily understood by the user. In other words, NLG is the process of generating text/speech from system generated patterns. The results produced by the system are in the structured format so that they can be easily understood and processed by the system. NLG represents the system knowledge base in a natural or conversational language representation which can be easily understood by the user. There can be a number of ways in which in which a same sentence can be said. The sentences can have two voices i.e. active or passive voice. Also, there can be similarity between two sentences, but they might involve the usage of synonyms. Hence, while providing a response to the user, the NLG unit needs to calculate all the possibilities to interpret the same sentence, and then select the most appropriate one. NLG engine also performs a sequence of tasks to generate sentences. The initial task is to determine the content. It involves the selection of response to be given to the user. This step decides the appropriate content (or set of words) that should be present in the sentence. Also, it deals with the position of the words in the sentences based on its POS Tag (placements of verbs, nouns, adjectives, prepositions, etc.). In all, this step deals with the organization of a basic sentence right from the choice of words to their placement in the sentence. The next task is the choice of sentences. As already said, there can be a variety of sentences that can be used to express the same situation, this step deals with the selection of the appropriate sentence, which is the best for that instance. The sentences taken into consideration for possibilities are in their abstract format and are not perfect sentences. They require the addition of grammar rules to make them grammatically correct. This section checks the semantic correctness of sentences based on the grammar rules defined by the system. Last and the most important is the morphology check, wherein the sentence generated from the previous steps is checked upon for its correctness. This step validates the correctness of the sentence.

Table 1: *A Summary of Literature Review*

S.N.	Authors	Problem discussed and solved	Method/ Algorithm/ Tools Used	Results
1	Mohammed Javed et al. [1] ,[15]	To implement word segmentation (tokenization)	Calculating all character spaces	It involves mathematical calculations hence proves to be slower than the others.
2	Naeun Lee et al. [2], [17]	To implement word segmentation (tokenization)	Using NLTK package which involves inbuilt tokenizer	Easy to implement, as does not require any coding. Faster and more accurate
3	Tao Jiang et al. [3], [11]	To implement word segmentation (tokenization)	Using Conditional Random Fields	This algorithm proves to be more accurate and less complex than the first but less efficient as compared to NLTK.
4	Jerome R. Bellagarda [4] ,[10]	To implement POS Tagging	Using the latent analogy algorithm	Requires training of large amount of data. Hence involves complexity.
5	Liner Yang et al. [5], [18]	To implement POS Tagging	Using neural network algorithm	As the algorithm works in layers, it provides high accuracy, but is not time efficient.
6	None	To implement POS Tagging	Using NLTK	Provides above average accuracy at minimum complexity.
7	Bo Chen et al. [6], [1]	To create a dependency parser	Using a dependency tree to understand the dependencies.	Traditional method. Accuracy depends on the training of the data.
8	Zhenghua Li et al. [7], [14]	To create a dependency parser	Using a graph data structure for the implementation of the parser	Improvised version of the above- mentioned algorithm. Provides higher visibility, understandability and improves accuracy.
9	LinHua Gao et al. [8],	Synonym detection and	Dictionary method	Traditional method. Requires to maintain a

	[18]	extraction		dictionary of synonyms wordwise. Provides less accuracy then self-training mod-els.
10	Sijun Qin et al. [9], [15]	Synonym detection and extraction	Feature selection method.by calculating feature polarity	Provides high accuracy and less complexity as compared to dictionary method
11	Sachin S. Gavankar et al. [10], [17]	Implementing disease predictions.	Eager decision tree.	The dynamic nature of the tree makes it more efficient. Provides high rate of accuracy due to the updating mechanism pre-sent in it.
12	Naganna Chetty et al. [11] ,[15]	Implementing disease predictions.	Fuzzy approach	Provides high accuracy, but is not efficient to implement as it involves the scanning of the entire database for each iteration.Though the amount of data decreases on each iteration, but yet initially the data is quite large to be scanned.

4 Conclusion

This paper provides a critical review of the tasks involves in NLU and ML for inclusion of them in chatbot systems to make them smart. This paper covers the review of 24 research papers. This indicates that a lot more research paper can be published in this research area in future. It is found that there are a wide range of algorithms for implementation of all the tasks involved in NLU and ML. The selection of the appropriate algorithm depends upon the functionalities to be provided by the chatbot and also domain in which the services are to be provided. Also, the data format plays a vital role for the selection of an algorithm.

Acknowledgment

This survey was supported by CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING (CDAC), PUNE, under guidance of Mrs. Priyanka Jain (Joint Director, CDAC). We are thankful to our colleagues Rushabh Sancheti and Sunil Upare who provided expertise and greatly assisted the survey.

References

[1] Mohammed Javed, P. Nagabhushan, B.B. Chaudhari, “A Direct Approach for Word and Character Segmentation in Run-Length Compressed Documents with an Application to Word Spotting”, 13th International Conference on Document Analysis and Recognition (ICDAR), 2015.

[2] Naeun Lee, Kirak Kim, Taeseon Yoon, “Implementation of Robot Journalism by Programming Custombot using Tokenization and Custom Tagging”, 2017.

[3] Tao Jiang, Hongzhi Yu, Yangkyi Jam, “Tibetan Word Segmentation Systems based on Conditional Random Fields”, 2011.

[4] Jerome r. Bellagarda, “Parts-Of-Speech tagging by Latent Analogy”, IEEE Journal of Selected Topics in Signal Processing, Vol. 4, No. 6, 2010.

[5] Liner Yang, Meishan Zhang, Yang Liu, Maosong Sun, Nan Yu, Guohong Fu, “Joint POS Tagging and Dependency Parsing with Transition-based Neural Networks”, 2018.

[6] Bo Chen, Donghong Ji, “Chinese Semantic Parsing based on Dependency Graph and Feature Structure”, International Conference on Electronic and Mechanical Engineering and Information Technology, 2011.

[7] Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, and Wenliang Chen, “Joint Optimization for Chinese POS Tagging and Dependency Parsing”, IEEE/ACM transactions on audio, speech, and language processing, Vol. 22, No. 1, January 2014.

[8] LinHua gao, HePing Chen, “An Automatic Extraction Method Based on Synonym Dictionary for Web Reptile Question and Answer” 2018.

[9] Sijun Qin, Jia Song, Pengzhou Zang, Yue Tan, “Feature Selection for Text Classification Based on Parts-Of-Speech Filter and Synonym Merge”, 12th International Conference on Fuzzy Systems and Knowledge Discover (FSKD), 2015.

[10] Sachin S. Gavankar, Sudhirkumar D. Sawarkar, “Eager Decision Tree”, 2nd International Conference for Convergence in Technology (I2CT), 2017.

[11] Naganna Chetty, Kunwar Singh Vaisla, Nagamma Patil, “An improved Method for Disease Prediction using Fuzzy Approach”, 2nd International Conference on Advances in Computing and Communication Engineering, 2015.

[12] Kyo-Joong, DongKun Lee, ByungSoo Ko, Ho-Jin, Choi, “A Chatbot for Psychiatric Counseling in Mental Healthcare Service Based on Emotional Dialogue Analysis and Sentence Generation”, IEEE 18th International Conference on Mobile Data Management, 2017.

[13] Bhavika R. Ranoliya, Nidhi Raghuwanshi, Sanjay Singh, “Chatbot for University FAQs”.

[14] Ming-Hsiang Su, Chung-Hsien Wu, Kun-Yi Huang, Qian-Bei Hong, Hsin-Min Wang, “A Chatbot Using LSTM-based Multi-Layer Em- bedding for Elderly Care”.

[15] Cyril Joe Baby, Faizan Ayyub Khan, Swathi J. N., “Home Automation using IOT and a Chatbot using Natural Language Processing”,

[16] International Conference on Innovations in Power and Advanced Computing Technologies.

[17] [Ashay Argal, Siddharth Gupta, Ajay Modi, Pratik Pandey, Simon Shim, Chang Choo, “Intelligent Travel Chatbot for Predictive Recommendation in Echo Platform”.

[18] [Oliver Pietquin, Thierry Dutoit, “Dynamic Bayesian Networks for NLU Simulation with Applications to Dialog Optimal Strategy Learning”.

[19] Fco Mario Barcala, Jesus Vilares, Miguel A. Alonso, Jorge Grana, Manuel Vialres, “Tokenization and Proper Noun Recognition for Information Retrieval”.

[20] Soo H. Kim, Chang B. Jeong, Hee Kwag, Chin Y. Suen, "Word Segmentation of Printed Word Lines Based on Gap Clustering and Special Symbol Detection".
[21] Xiaofei Li, Xusheng Xie, "Research of Intelligent Word Segmentation and Information Retrieval", 2nd International Conference on Education Technology and Computer (ICETC)", 2010.
[22] Meishan Zhang, Nan Yu, Guohong Fu, "A Simple and Effective Neural Model for Joint Word Segmentation and POS Tagging", 2018.
[23] [Liner yang, Meishan Zhang, Yang Liu, Maosong Sun, Nan Yu, Guohong Fu, "Joint POS Tagging and Dependency Parsing with Transition Based Neural Networks.", 2018.
[24] Sagar Verma, Sukhad Anand, Chetan Arora, Atul Rai, "Diversity in Fashion Recommendation using Semantic Parsing", 2018.