International Journal on Applications in Basic and Applied Sciences
Volume 9: Issue 4 : September 2023, pp 1 – 6 www.aetsjournal.com ISSN (Online) : 2455 – 1007

Artificial Intelligence and Machine Learning Based Healthcare Chatbot System

[1]Mrs. Abhirami.J.S , [2]Mrs. M.S.Vinu
Assistant Professor
[1]Department of Artificial intelligence and Data Science
[2]Department of Computer Science and Engineering
Nehru Institute of Engineering and Technology
nietjsabhirami@nehrucolleges.com

***Abstract*—** With increasing population of India, increasing birth rate and decreasing death rate due to advancement in the medical field it's found that numbers of doctors are less to serve the need of the increasing population. This scenario can be better understood while walking through the cities government hospitals where the less availability of the doctors is the major cause behind the improper treatment of the patients and in certain scenario the resultant death. Sometime even doctors can make mistake in providing the correct treatment result in death of patient. To encounter such cases there is a need of the smart and Intelligent Chabot who can provide advice to the doctors and sometime even patients about what to do in such cases which ultimately results in the saving the life of hundreds of people. The AI based medical Chabot on which this research topic is based deals with providing medical advice in such scenario because sometime doctors can even make mistake while observing the symptoms but the machine which is specifically developed for it can't make such mistake. This AI based medical Chabot can take decision as per the request of the patient.

***Keywords*—** Health care,Artificial Intelligence,Chabot, Symptoms.

I. INTRODUCTION

Through chat bots one can communicate with text or voice interface and get reply through artificial intelligence. Typically, a chat bot will communicate with a real person. Chat bots are used in applications such as e-commerce customer service, call centers and Internet gaming. Chat bots are programs built to automatically engage with received messages. Chat bots can be programmed to respond the same way each time, to respond differently to messages containing certain keywords and even to use machine learning to adapt their responses to fit the situation. A developing number of hospitals, nursing homes, and even private centers, presently utilize online Chat bots for human services on their sites. These bots connect with potential patients visiting the site, helping them discover specialists, booking their appointments, and getting them access to the correct treatment. An ML model has to be created wherein we could give any text input and on the basis of training data it must analyze the symptoms. A Supervised Logistic Regression machine learning algorithm can be implemented to train the model with data sets containing various diseases CSV files. The goal is to compare outputs of various models and suggest the best model that can be used for symptoms in real world inputs. Data set contains CSV file having all diseases compiled together. The logistic regression algorithm in ML allows us to process the data efficiently. The goal here is to model the underlying structure or distribution of the data in order to learn more from the training set. In any case, the utilization of artificial intelligence in an industry where individuals' lives could be in question, still starts misgivings in individuals. It brings up issues about whether the task mentioned above ought to be assigned to human staff. This healthcare chat bot system will help hospitals to provide healthcare support online 24 x 7; it answers deep as well as general questions. It also helps to generate leads and automatically delivers the information of leads to sales. By asking the questions in series it helps patients by guiding what exactly he/she is looking for.

Almost everyone kept on hold while operators connect you to a customer care executive. On an average people spend around 7 minutes until they are assigned to a person. Gone are the frustrating days of waiting in a queue for the next available operative. They are replacing live chat and other forms of slower contact methods such as emails and phone calls. Since chat bots are basically virtual robots they never get tired and continue to obey your command. They will continue to operate every day throughout the year without requiring to take a break

II. LITERATURE SURVEY

[1]Mohammed Javed et al.[2016] 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 charactersThey 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.

[2]Naeun Lee et al. [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.

[3]Tao Jaing [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.

[4]Jerome R. Bellegarda [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

. [5]Liner Yang et al. [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 predefined 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.

.

[6]Bo Chen [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.

[7] Zhenghua Li [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.

[8]LinHua Gao et al. [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.

[9]Sijun Qin [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.

[10] Sachin S. Gavankar et al [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.

III. PROPOSED SYSTEM

A. RASPBERRY PI:

The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2 BLE, faster Ethernet, and PoE capability via a separate PoE HAT The dual-band wireless LAN comes with modular compliance certification, allowing the board to be designed into end products with significantly reduced wireless LAN compliance testing, improving both cost and time to market. The Raspberry Pi 3 Model B+ maintains the same mechanical footprint as both the Raspberry Pi 2 Model B and the Raspberry Pi 3 Model B.

Figure 1.The Raspberry Pi 3 Model B+

The Raspberry Pi Compute Module 3+ (CM3+) is a range of DDR2-SODIMM-mechanically-compatible System on Modules (SoMs) containing processor, memory, eMMC Flash (on non-Lite variants) and supporting power circuitry. These modules allow a designer to leverage the Raspberry Pi hardware and software stack in their own custom systems and form factors. In addition these modules have extra IO interfaces over and above what is available on the Raspberry Pi model A/B boards, opening up more options for the designer. The CM3+ contains a BCM2837B0 processor (as used on the Raspberry Pi 3B+), 1Gbyte LPDDR2 RAM and eMMC Flash. The CM3+ is currently available in 4 variants, CM3+/8GB, CM3+/16GB, CM3+/32GB and CM3+ Lite, which have 8, 16 and 32 Gigabytes of eMMC Flash, or no eMMC Flash, respectively. The CM3+ Lite product is the same as CM3+ except the eMMC Flash is not fitted, and the SD/eMMC interface pins are available for the user to connect their own SD/eMMC device. Note that the CM3+ is electrically identical and, with the exception of higher CPU z-height, physically identical to the legacy CM3 products. CM3+ modules require a software/firmware image dated November 2018 or newer to function correctly.

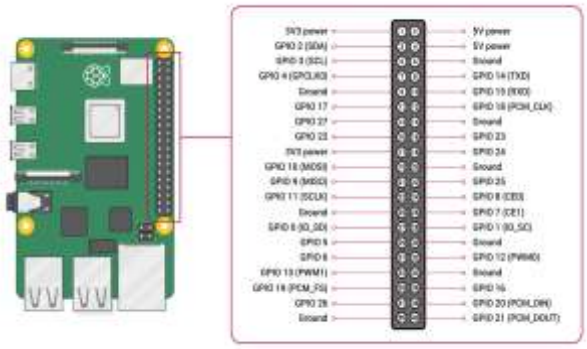

Figure 2:Raspberry pi pin diagram

B. I2C LCD DISPLAY:

The I2C 1602 LCD module is a 2 line by 16 character display interfaced to an I2C daughter board. The I2C interface only requires 2 data connections, +5 VDC and GND For in depth information on I2C interface and history, visit: http://www.wikipedia/wiki/i2c Specifications: 2 lines by 16 character I2C Address Range 0x20 to 0x27 (Default=0x27, addressable) Operating Voltage 5 V DC Backlight White Adjustable by potentiometer on I2c interface 80mm x 36mm x 20 mm 66mm x 16mm Power: The device is powered by a single 5Vdc connection

C. POWER SUPPLY:

This document presents the solution for a 12V 1A flyback converter based on the Infineon OPTIREG™ TLE8386-2EL controller and IPD50N08S4-13 OptiMOS™-T2. The user is guided through the component selections, the circuit design and, finally, an overview of the experimental results are presented. The TLE8386-2EL is part of the Automotive OPTIREG™ family and it implements a low-side-sense current mode controller with built in protection features. The device is AECQ-100 qualified. The IPD50N08S4-13 is an AEC-Q101 qualified 80V N-channel enhanced mode MOSFET; it is part of the OptiMOS™-T2 family. Intended audience This document is intended for power supply design engineers, application engineers, students, etc., who need to design a Flyback converter for automotive power applications where a galvanic isolation between two voltage domains is required. In particular the focus is on a battery connected flyback that delivers up to 12W at 12V output voltage; the intention is to provide the user with all of the needed information to fully design and characterize the SMPS bringing it from an engineering concept to its production. Specific features and applications are: - 48V to 12V

Automotive applications - Isolated current mode SMPS – Flyback regulators with auxiliary sensing.

Centre Tapped Transformer Specifications
Step-down Centre tapped Transformer
Input Voltage: 220V AC at 50Hz
Output Voltage: 24V, 12V or 0V
Output Current: 1A
Vertical mount type
Low cost and small package

A centre-tapped transformer also known as two phase three wire transformer is normally used for rectifier circuits. When a digital project has to work with AC mains a Transformer is used to step-down the voltage (in our case, to 24V or 12V) and then convert it to DC by using a rectifier circuit. In a center-tapped transformer the peak inverse voltage is twice as in bridge rectifier hence this transformer is commonly used in full wave rectifier circuits.

The operation and theory behind a Center tapped transformer is very similar to a normal secondary transformer. A primary voltage will be induced in the primary coil (I1 and I3) and due to magnetic induction the voltage will be transferred to the secondary coil. Here in the secondary coil of a centre tapped transformer, there will be an additional wire (T2) which will be placed exactly at the center of the secondary coil, hence the voltage here will always be zero.If we combine this zero potential wire (T2) with either T1 or T2, we will get a voltage of 12V AC. If this wire is ignored and voltage across T1 and T2 is considered then we will get a voltage of 24V AC. This feature is very useful for the function of a full wave rectifier.

D.THONNY IDE:

Thonny is as mall and light weight Integrated Development Environment. It was developed to provide a small and fast IDE, which has only a few dependencies from other packages. Another goal was to be as independent as possible from a special Desktop Environment like KDE or GNOME, so Thonny only requires the GTK2 toolkit and therefore you only need the GTK2 runtime libraries installed to run it.

For compiling Thonny yourself, you will need the GTK (>= 2.6.0) libraries and header files. You will also need the Pango, Gliband ATK libraries and header files. All these files are available at http://www.gtk.org. Furthermore you need, of course, a C compiler and the Make tool; a C++ compiler is also required for the included Scintilla library. The GNU versions of these tools are recommended.Compiling Thonny is quite easy. The following should do it:

There are also some compile time options which can be found in src/Thonny. Please see Appendix C for more information. In the case that your system lacks dynamic linking loader support, you probably want to pass the option --disable-vte to the configure script. This prevents compiling Thonny with dynamic linking loader support to automatically load libvte.so.4 if available. Thonny has been successfully compiled and tested under Debian 3.1 Sarge, Debian 4.0 Etch, Fedora Core 3/4/5, Linux From Scratch and FreeBSD 6.0. It also compiles under Microsoft WindowsAt startup, Thonny loads all files from the last time Thonny was launched. You can disable this feature in the preferences dialog (see Figure 3-4). If you specify some files on the command line, only these files will be opened, but you can find the files from the last session in the file menu under the "Recent files" item. By default this contains the last 10 recently opened files. You can change the amount of recently opened files in the preferences dialog. You can start several instances of Thonny , but only the first will load files from the last session. To run a second instance of Thonny , do not specify any file names on the command-line, or disable opening files in a running instance using the appropriate command line option.Thonny detects an already running instance of itself and opens files from the command-line in the already running instance. So, Thonny can be used to view and edit files by opening them from other programs such as a filemanager. If you do not like this for some reason, you can disable using the first instance by using the appropriate command line optionIf you have installed libvte.so in your system, it is loaded automaticallyThonny , and you will have a terminal widget in the notebook at the bottom. If Thonny cannot find libvte.so at startup, the terminal widget will not be loaded. So there is no need to install the package containing this file in order to run Thonny . Additionally, you can disable the use of the terminal widget by command line option, for more information see Section3.2.You can use this terminal (from now on called VTE) nearly as an usual terminal program like xterm. There is basic clipboard support. You can paste the contents of the clipboard by pressing the right mouse button to open the popup menu and choosing Paste. To copy text from the VTE, just select the desired text and then press the right mouse button and choose Copy from the pop up menu. On systems running the X Window System you can paste the last selected text by pressing the middle mouse button in the VTE (on 2-button mice, the middle button can often be simulated by pressing both mouse buttons together).As long as a project is open, the Make and Run commands will use the project's settings, instead of the defaults. These will be used whichever document is currently displayed. The current project's settings are saved when it is closed, or when Thonny is shut down. When restarting Thonny , the previously opened project file that was in use at the end of the last session will be reopened.

Execute will run the corresponding executable file, shell script or interpreted script in a terminal window. Note that the Terminal tool path must be correctly set in the Tools tab of the Preferences dialog - you can use any terminal program that runs a Bourne compatible shell and accept the "-e" command line argument to start a command. After your program or script has finished executing, you will be prompted to press the return key. This allows you to review any text output from the program before the terminal window is closed.By default the Compile and Build commands invoke the compiler and linker with only the basic arguments needed by all programs. Using Set Includes and Arguments you can add any include paths and compile flags for the compiler, any library names

and paths for the linker, and any arguments you want to use when running Execute.Thonny has basic printing support. This means you can print a file by passing the filename of the current file to a command which actually prints the file.However, the printed document contains no syntax highlighting.

E. SYSTEMDESIGN

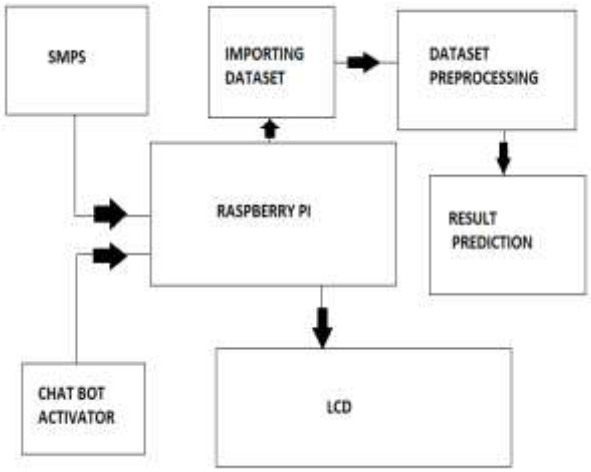

Figure.3:System Architecture

F. WORKING

1. The raspberry pi is interfaced with chatbot activator button and once the button gets activated and the chat bot starts asking the question once we starts replying to the question the raspberry pi performs Machine learning and compares the answers with the symptom database
2. If any of the symptom gets matches the respective symptom is shown through LCD and the chatbot also suggest the Respective doctor to visit.

IV. RESULTS

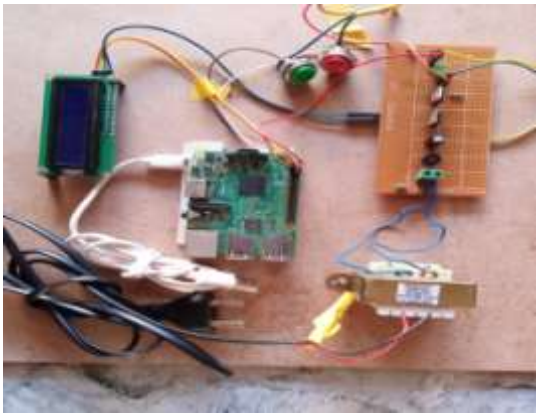

Figure 4: LCD, Transformer and SMPS Connected with Raspberry Pi

Figure 5: Hardware result for Desease with certainty of the desease

S.No	Question	Answer
1	Is there any pain in certain part of the body?	Yes
2	Do you have cold?	No
3	Feeling pain behind the eyes?	No
4	Are you feeling nauseous?	Yes
5	Is there pain in upper abdominal area?	Yes
6	Are you suffering indigestion?	Yes
7	Is there any vomiting?	Yes
8	Is there any bleeding from mouth?	No
9	Is there any pain sensation in middle of the stomach?	Yes

Table 1: Prediction using the corresponding symptoms

.No	Disease	Certainly percentage	Preferred doctor
1	peptic ulcer	79	Dr.Malathi sathyanathan MBBS,MD-Peadiatrics

Table 2: Certainty of the desease and preferred

V.CONCLUSION

Thus, we can conclude that this system giving the accurate result. As we are using large dataset which will ensures the better performance. Thus, we build up a system which is useful for people to detect the disease by typing symptoms

Future Scope:

Chat bots are a thing of the future which is yet to uncover its potential but with its rising popularity and craze among companies, they are bound to stay here for long. Machine learning has changed the way companies were communicating with their customers. With new platforms to build various types of chat bots being introduced, it is of great excitement to witness the growth of a new domain in technology while surpassing the previous threshold.

REFERENCE

[1] Flora Amato, Stefano Marrone, "Chatbots meet e-Health: automat zing healthcare", proceeding of diet, May-2018.

[2] BenildaEleonor V. Comendador, "Pharmabot: A pediatric generic Medicine consultant Chatbot", proceeding of the JACE, April 2015.

[3] Divya, Indumathi, Ishwarya, Priyasankari, "A SelfDiagnosis Medical Chatbot Using Artificial Intelligence", proceeding MAT Journal, October-2017.

[4] Tobias Kowatsch," Text-based Healthcare Chatbots Supporting Patient and Health", 01 October 2017.

[5] Chin-Yuan Huang, Ming-Chin Yang, Chin-Yu Huang, "A Chatbot-supported Smart Wireless Interactive Healthcare System for Weight Control and Health Promotion", proceeding of the IEEE, April-2018.

[6] Boukricha, H., Wachsmuth, I.: Modeling Empathy for a Virtual Human: How, When and to What Extent. The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 3. International Foundation for Autonomous Agents and Multiagent Systems, 2011., pp. 1135–1136 .

[7] Agarwal, R., Gao, G., DesRoches, C., et al.: The Digital Transformation of Healthcare: Current Status and the Road Ahead. Information Systems Research 21, 796-809 (2010).

[8] Aron, A., Aron, E.N., Smollan, D.: Inclusion of Other in the Self Scale and the structure of interpersonal closeness. Journal of Personality and Social Psychology 63, 596-612 (1992).

[9] Bickmore, T., Cassell, J.: Social Dialogue with Embodied Conversational Agents. In: Kuppevelt, J.C.J., Bernsen, N.O., Dybkjær, L. (eds.) Advances in Natural Multimodal Dialogue Systems, vol. 30, pp. 23–54. Springer, Dordrecht (2005).

[10] Bickmore, T., Gruber, A., Picard, R.: Establishing the computer–patient working alliance in automated health behavior change interventions. Patient Education and Counseling 59, 21-30 (2005).

[11] K. Oh, D. Lee, B. KO and H. Choi, "A Chatbot for Psychiatric Counseling in Mental Healthcare Service Based on Emotional Dialogue Analysis and Sentence Generation," 2017 18th IEEE International Conference on Mobile Data Management (MDM), Daejeon, 2017, pp. 371-375. doi: 10.1109/MDM.2017.64

[12] Du Preez, S.J. & Lall, Manoj & Sinha, S. (2009). An intelligent webbased voice chat bot. 386 - 391.10.1109/EURCON.2009.5167660

[13] Bayu Setiaji, Ferry Wahyu Wibowo, "Chatbot Using a Knowledge in Database: Human-to- Machine Conversation Modeling", Intelligent Systems Modelling and Simulation (ISMS) 2016 7th International Conference on, pp. 72-77, 2016.

[14] Dahiya, Menal. (2017). A Tool of Conversation: Chatbot. INTERNATIONAL JOURNAL OF COMPUTER SCIENCES AND ENGINEERING. 5. 158-161.2017[15] C.P. Shabariram, V. Srinath, C.S. Indhuja, Vidhya (2017). Ratatta: Chatbot Application Using Expert System, International Journal of Advanced Research in Computer Science and Software Engineering,2017

[16] Mrs Rashmi Dharwadkar1, Dr.Mrs. Neeta A. Deshpande, A Medical ChatBot, International Journal of Computer Trends and Technology (IJCTT) – Volume 60 Issue 1- June 2018

[71] Farheen Naaz, Farheen Siddiqui, modified n-gram based model for identifying and filtering near-duplicate documents detection, International Journal of Advanced Computational Engineering and Networking, ISSN: 2320- 2106, Volume-5, Issue-10, Oct.-2017

[18] N-gram Accuracy Analysis in the Method of Chatbot Response, International Journal of Engineering & Technology. (2018)

[19] Shukla, V.K, Verma, A, "Enhancing LMS Experience through AIML Base and Retrieval Base Chatbot using R Language", 2019 International Conference on Automation, Computational and Technology Management (ICACTM)