International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-5, January 2020

Adaptive Machine Learning Chatbot for Code-Mix Language (English and Hindi)

Rushabh Sancheti, Sunil Upare, Nivedita Bhirud, Subhash Tatale

Abstract*: The humanoid assistant system can be described as the system resembling or imitating the human behaviour. These systems can be called as chatbots. There are a large number of conventional scripted types of chatbots. The problem with these chatbots is that they provide a monotonous type of communication i.e. they provide the user with a predefined set of options for any of its query. This scripted nature limits the scope of the chatbot systems, to provide smart and effective services to the users. This problem restricts the system efficiency. Efforts are being made to improve the scripted nature of chatbots and enable them to converse in a manner similar to the conversation between two humans. This makes the system more user-friendly, and provides better solutions to them. Chatbots providing health care services imitate the conversation between the doctor and the patients to give them general information about diseases, remedies, precautions, etc. and also provides a prediction of the diseases depending upon the symptoms provided by the user. Here, the chatbot behaves as a virtual doctor. This can be achieved by incorporating NLU, ML and NLG techniques in the system. Here, in this paper, we have briefed about the chatbot system architecture and adaptive self-learning algorithm for providing services in healthcare domain.

Keywords: Chatbot, Healthcare Domain, ML (Machine Learning), NLG (Natural Language Generation), NLU (Natural Language Understanding), Virtual Doctor.

I. INTRODUCTION

Research says 60% of visits to a doctor are for simple small-scale diseases, out of which 80% can be cured at home if they have the right information. 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 any intervention of the doctor. This humanoid assistant provides users smart communication with basic information related to health care, predicts the disease suffered by the user by knowing the symptoms faced by him/her and also recommends precautions to be taken or remedies to cure the same. If the disease predicted is a high scale disease, which would require medical assistance, then the system suggests that he should consult his general physician. The basic intention of the system is to communicate with the user in the same way as a doctor does so that the user feels free to talk out any type of problem faced by him. The system then accordingly consults the user. The system acts as a virtual communicating friend, to achieve the health care counselling.

There are a wide number of chatbots which cater to provide services in diverse fields for various requirements of the user. Chatbots in the health care domain prove to be beneficial to predict the diseases faced by the users. Also, there is a need felt to build a system which can provide smart and friendly communication with the users. The aim is to replace the scripted chatbots (which can only predict the diseases) with smart chatbots which make use of NLU, ML and NLG techniques.

The system primarily aims at achieving a smart and natural communication with the user. Both the user and the system should understand what the other wants to convey. The system should be able to interpret the intent of the user's message. Also, the system should be capable of gathering useful information (symptoms, duration, etc.) from the user and predict the disease suffered.

The paper walks through an entire architecture of the chatbots which are built using NLP and ML algorithms. Section 2 provides a description of the system architecture, the modules included in the system, the knowledge base used, and also the algorithms used for prediction. Section 3 shows the modularization done while building the system. Section 4 and Section 5 provide the Conclusions, Result Analysis and Future Work respectively.

II. ARCHITECTURE

This section explains the architecture of the proposed system. The following diagram depicts the same.

Manuscript published on January 30, 2020.
* Correspondence Author
Rushabh Sancheti*, Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune, India.
Email: rushabhpsancheti@gmail.com
Sunil Upare*, Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune, India.
Email: sunilupare45@gmail.com
Nivedita Bhirud, Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune, India.
Email: nivedita.bhirud@viit.ac.in
Subhash Tatale, Department of Computer Engineering Vishwakarma Institute of Information Technology, Pune, India.
Email: subhash.tatale@viit.ac.in

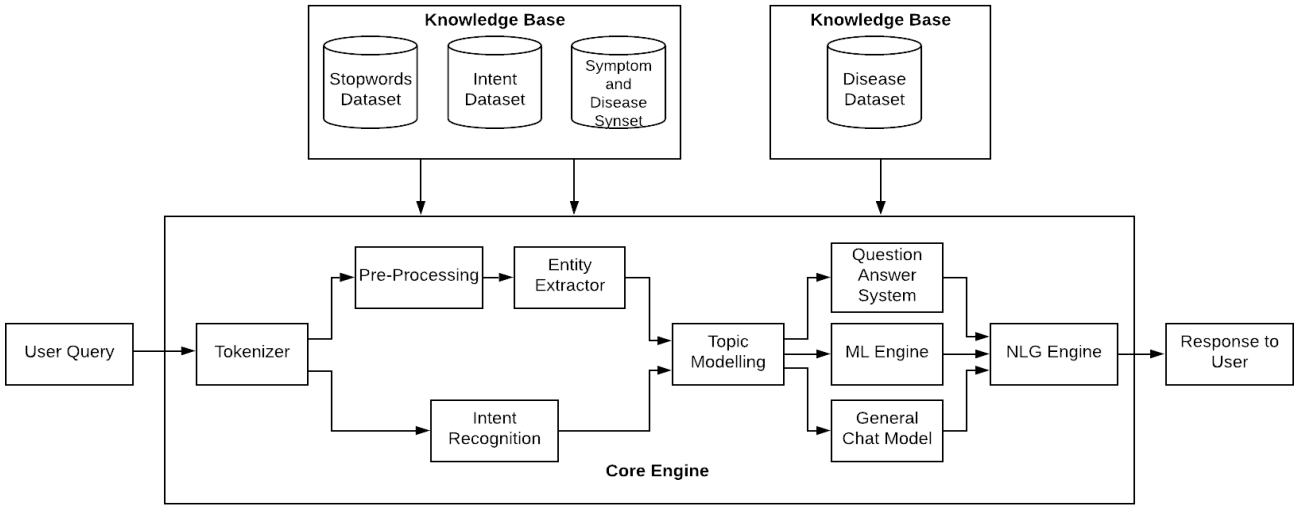

Fig. 1. **Architecture Diagram (Architecture Diagram)**

A. Knowledge Base

The proposed system uses four types of datasets, each with one function. The different datasets used are as follows:

- Disease Dataset

This dataset consists of all the information regarding various diseases ranging from small scale (home-curable) diseases to large scale diseases. As the system is intended to be used for health counselling and disease prediction using the symptoms of the user, the Chabot definitely requires data about diseases and their symptoms. For a collection of data regarding the diseases, site scrapping has been used. Scrapped data is added in the disease dataset. This dataset consists of all the information about various diseases. Information about each disease is stored with fields like disease_name, symptoms, descriptions, precautions, remedies, causes, etc.

JSON Format of Disease Dataset

```
{
   Disease_Name :{},
   Description :{},
   Symptoms :{},
   Precautions :{},
   Remedies :{},
   When_to_visit_your_doctor :{},
}
```

This dataset is stored in the JSON format which is as shown above. The contents of this dataset are also accessed in the JSON format.

- **Symptom and Disease Synset**

Initially, the names of all common symptoms faced by the user were extracted. As this data can be in the structured format so the data can be stored in the row-column format. Symptom names are stored in the symptom table as a row for each symptom. Also, there could be various ways used by user to express different symptoms. For example, if any user wants to say that he/she has fever, he can convey the same in many ways as follows: high temperature, body warm, heating up, feverishness, etc. or if user is conversing in Hindi or code-mix language, user can say: jwar, bukhar etc. All of the above words mean the same i.e. it conveys that the user has fever. Hence, the symptoms table also consists of the synonyms or synset of the corresponding symptoms in both the English and Hindi languages.

There can be multiple spellings of the word jwar like jwer or jwr and this totally depends on the user. So, all such variations are also added in the synonym table.

This symptom dataset is used when the user uses some synonym of a particular symptom in his/her query.

- Intent Dataset

For this system, the user queries can express the following intents: Greetings, affirmations i.e. positive statements, negations i.e. sentence in which the user uses words like not, never, neither and so on, uncertainty i.e. the user is not sure about something, farewells i.e. saying goodbyes and exiting, bad words i.e. user query which consists of a slang, and the intents of the user like ‘give description’, ‘give symptoms’, ‘user symptoms detection’, ‘give remedies’, etc. Each of these intents consists of a group of sentences, which justify the same intent.

- **Stopwords Dataset**

Stop words are the words used in sentences which don’t convey any specific meaning but used for denoting any entity or for specifying any relationship between entities in the statement. These words can be removed from sentence which will be used for the process of meaning extraction. Removal of these words will increase the speed of processing of the statement.

Stop words can be the articles, punctuations, prepositions, conjunctions and all such words which do not specify the meaning of a sentence. Dataset maintains the list of these stop-words. There are two different datasets which contains stop words of both Hindi and English language respectively.

User Query

The system begins its functioning once the user enters the chat interface and inputs its initial query. The system can take input in string format or can also use speech to text API to take audio input. This audio input is then converted into the string format.

Retrieval Number: E6489018520/2020©BEIESP
DOI:10.35940/ijrte.E6489.018520
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

These queries of the user are captured by the system and passed to the core engine in the string format.

B. Tokenizer

Firstly, the user query is tokenized into the separate sentences if there is two or more sentences. After sentence tokenization, each sentence is again tokenized into the individual words. This step also includes the removal of stop words. These include articles, prepositions, punctuations, etc. Also, the main words from the sentence are used to find the intent of the user query. Furthermore, pre-processing and intent recognition is carried out to classify the type of user query.

C. Pre-processing

This module will perform transformations on the data for smooth and simplified processing in the further module. In this module, POS-tagging and stop-words removal process will be applied. POS-tagging will tags each word with its respective Part-Of-Speech tag (like Noun, Pronoun, Verb, etc.) POS tagging

Stop words dataset is used to remove stop words of Hindi language. If a word in stop word dataset matches with the word in user query then it will be removed. Removal of stop words reduces the processing required in an entity extractor.

D. Entity Extractor

This module is used for the purpose of the symptom detection and disease detection from the user query. It will extract if any symptoms or disease is specified in the sentence. Output of pre-processing module contains POS tagged and stop word removed query. It uses symptom and disease synset for entity extraction.

E. Intent Recognizer

This module of the system is the most important one, because the functioning of the chatbot further will depend on the intent of the user query.

The intent recognizer works on the principle of similarity. Similar meaning sentences for respective intents have been added into the intent dataset. But it is not necessary that the user would frame those sentences exactly in the same manner. The important words that have been extracted in the previous module are checked in the database. The nearest match sentence or group of words is looked up for in the dataset. When the nearest match is found, the module returns the corresponding intent of the sentence.

An example of intent recognition could be identification of greetings used by the user. The user can just greet system by saying Good morning, Hello, Hi, Heyya, etc. The intent dataset consists of an intent named Greetings which has sub greetings like good morning, good afternoons, good nights as one row, hi, hiee, hy and all different versions of 'hi' that can be used by the user in the next row, versions of 'hello' in the next and so on. All possible greetings have been listed here. Suppose the user comes up with 'Good Morning', the chatbot will first check the time of the system on which the chatbot is running and will send an appropriate greeting i.e. Good morning or afternoon or night.

If user requests remedies for some disease then the keywords from the input query are matched with intent dataset. Remedies column of intent dataset consists of keywords which suggests that user need remedies. If system founds perfect match or any nearby match then it recognizes the intent as 'give remedies'.

Similarly, system will detect the other intents of the user from user queries. These intents can be 'give description', 'give symptoms', 'user symptoms detection', 'give remedies' , 'affirmation detection' , 'negation detection' , 'uncertainty detection'.

F. Topic Modelling

User can input any type of query. This query can be a friendly chat or some question or user specifying about his problem. According to the type of query, system will decide the response. Entity extraction and intent recognition are used to find out the type of user query. There are three conditions that are handled by this module. They are explained as follows:

If the user query consists of intents like 'give remedies' or 'give description' along with a disease name and a question or imperative statement for the same then it is understood that the user is trying to get information about the disease i.e. trying to know the remedies or the details of the disease as mentioned, and this query is sent to question answer module.

Secondly, if the user query consists of symptom name and intent is recognized as 'user specifying symptoms'. In this case, the entity extractor module passes this entity set to the ML Engine for disease prediction.

If a query is not identified in above two modules then it is passed to general chat model.

G. Question Answer Module

This module will respond if user asks any question related to any disease. The system first looks for that disease in the disease dataset and extracts the required information and produces the information as per its intent. If the information is not available in the knowledge base then web-site scrapping can be used to extract information from web.

H. ML Engine

This module is the heart of the system i.e. the core engine. It is used for deciding which processing algorithm will be used on the user query depending upon the type of user query. Type of the query is decided in the topic modelling section. If query is apart from general chat and question, ML Engine takes input of the entity extractor, intent recognizer and topic modelling unit, and processes that information. If information from these components is sufficient enough for required output generation, then it generates the output. If the information collected is not sufficient then it will generate tokens. These tokens will contain the information required for ML engine for generating output. NLG unit will convert these tokens in appropriate question to get the required information from the user.

Disease prediction is done using the symptoms provided by the users. This disease prediction works on two algorithms: Apriori Algorithm for most frequent symptom detection and Eager Decision tree Algorithm for disease prediction.

Initially, the engine accepts all the symptoms faced by the user. It then checks for those symptoms in the disease dataset. All the diseases that contains those symptoms are extracted by the engine. The system now can have a single disease or a set of diseases remaining. If there is only single disease left, the system predicts this disease to the user. If there are more than one disease remaining, the engine then uses the Apriori algorithm to find out most frequently occurred symptom in those diseases.

1. Apriori Algorithm for most frequent Symptom Detection

This is a machine learning algorithm which helps to find frequently occured item-sets. In the disease prediction, this algorithm is used to find the most frequent symptom in the set of predicted diseases. The most frequent symptom predicted by this algorithm is then produced to the user and asked if he/she is facing that symptom. If the user says 'yes', the diseases containing that symptom are kept and all others are discarded and the system finds for the next most frequent symptom. If the user says 'no', the symptom is discarded and the next frequent symptom is calculated. This process continues until there is only one disease remained.

Algorithm

Step 1: Start
Step 2: Make the list of symptoms faced by the user.
Step 3: Extract all the diseases which have all the symptoms in the symptoms list as a symptom.
Step 4: Make the list of all the symptoms of those disease excluding symptoms faced by the user.
Step 5: Return the most frequently occurred symptom from this list.

Apriori algorithm has complexity of $\Theta (mn^2)$ where m is number of diseases and n is number of symptoms in the disease. Using hash table and dictionaries data structure it can be optimized in $\Theta (mn)$.

2. Eager Decision Tree Algorithm for he Disease Prediction

Every time using scanning of entire disease data for disease prediction will require a lot of computational complexity. Hence machine learning can be used to train model and use this trained model for further prediction. This will reduce computational complexity. Decision tree can be used to build this machine learning model.

The extraction and discarding of the diseases which is based on the symptoms, is done using the decision tree. But, this chatbot does not make use of the normal decision tree algorithm, instead it uses the Eager Decision Tree Algorithm. This algorithm is dynamic in nature. This tree is built based on the user queries. The nodes in this tree contain the symptom names and the edges contain 'yes' or 'no'. There can be multiple nodes of the same symptom but there will be unique paths for the diseases. There can also be more than one path for the diseases, but the paths are unique as they might contain slight changes in symptoms or also different order of symptoms.

Algorithm

Step 1: Start
Step 2: Make the list of symptoms listed by the user.
Step 3: Check if the list matches with Sub lists of the root node list.
  a) If yes, traverse to child_node of that node
  b) Check if it is a leaf node
    i) If yes, user is suffering from that Disease.
    ii) END
  c) Else ask user if he/she is facing from the symptom of that node
  d) Go to step 5
Step 4: If not, add the list of the symptoms to the root list of the tree
  a) Find most frequent occurring symptom using Apriori Algorithm
  b) Add that symptom as a symptom child to that list in the node.
  c) Ask user if he is facing that problem
  d) Go to step 5
Step 5: If user is facing from that symptom traverse to the yes_symptom of the node else traverse to the no_symptom of the node.
Step 6: If that node is a leaf node
  a) If yes, user is suffering from that Disease
  b) END
Step 7: If node is not present in the tree
  a) Add the current symptom of the node to the list of symptoms faced by the user
  b) Find most frequent occurring symptom using Apriori Algorithm
  c) Add that symptom as a yes_symptom or no_symptom as per answer given by user to question 5
  d) Ask user if he is facing that problem
  e) Goto step 5
Step 8: If node is present in the tree
  a) Ask user if he is facing that symptom
  b) Go to step 5

Decision tree has the complexity $\Theta (\log n)$ as it is a binary decision tree.

I. General Chat Model

This modules handles all other queries. This queries can be answered by storing queries and their responses. There are various python libraries available for implementing this model.

J. NLG Engine

This module works on the preparation of responses to be provided to the user. It prepares sentences using entity sets and makes the proper arrangement of the same. In this system, the entity set to be provided depends on the intent of the user query.

If the intent detected id 'greetings' then the system finds an appropriate response matching to the user's greeting and sends the same. If the user is asking about the system, then the system provides his introduction to the user. If the recognized intent is the general information question about a disease, then the system provides the appropriate answer by extracting it from the disease dataset.

If the intent is symptom detection, then the predicted disease along with its remedies are given to the user as a response.

In this system inbuilt python NLG tool is used. Entity set is provided to the NLG module and it makes the proper arrangement of the nouns, verbs, punctuations, prepositions according to the required tense and sends it to the user.

III. MODULARIZATION

The work to be done on HACI is divided into modules as follows:

1. Module 1 – Data
   - Collection of data from trusted sources like 'vikaspedia.in'
   - Cleaning and storing the same in structured format in JSON format
   - Addition of all symptoms and their synonyms in the database SQLite

2. Module 3 – Pre-processing
   - Receiving the user input in string format
   - Tokenizing the string
   - Removing the stop words like articles, punctuations, prepositions, etc.
   - Part of speech tagging
   - Extracting the important words

3. Module 2 – Intent Classification and Entity extraction
   - Gathering all possible intents that the user can have
   - Addition of the intents and the ways to express the intents in the database
   - Extracting entities from user query

4. Module 4 – Algorithm Selection
   - From the important words extracted, the engine identifies the intent of the user query whether it is a general chat, greetings, affirmative sentence, negations, etc.
   - Based on the intent the appropriate algorithm is selected.

IV. RESULT ANALYSIS

Implementation of architecture into application as specified in diagram (Fig. 1) have shown promising results. Application is able to predict disease with almost more than 90% accuracy. However, diseases who requires lab tests are not included in the scope and application is only providing homemade remedies. As this architecture aims for a wide variety of conversation, it requires lots of data processing and analysis, but algorithm learns from each experience. This self-learning nature of architecture gives better response time with more experience. Use of Apriori algorithm helps to provide more smart and accurate conversation.

Dialogue efficiency results calculated from the application have also shown the encouraging results.

Table I - Dialog Efficiency Results (For Specified Scope)

User Query Type	Accurate Query Type Prediction (in %)	Accurate Prediction of Response (in %)
General Chat other than HealthCare Domain	95%	62%
Question related to HealthCare	99%	81%
HealthCare Related Chat	89%	93%
Accurate Prediction Of Disease	94%	94%

Accurate prediction of responses in case of general chat depends on Knowledge base of the system. Efficiency of general chat queries by can increased by adding more relative information in knowledge base data. Question related responses are generated from the knowledge base as well if not found in knowledge base it can be searched on internet. Natural Language Processing and algorithm explained above have also shown promising results. Implementation of these algorithm have shown almost 93% of accuracy.

As application needs huge knowledge base, mining of the useful information from this data set takes lots of time. Due to adopted self-learning mechanism of the eager decision tree algorithm, application is able to cut down the time required for mining of the related data by 80%.

V. CONCLUSION

Using the NLU and ML algorithms, the chatbot system being implemented can be made smart and user friendly to make it behave as a virtual friend. Also, algorithms explained above makes the chatbots more efficient, as if it looks like not a virtual system but a human that you are talking to. Basic NLU and NLG make it possible to make the communication between the user and the system as natural as possible. Introduction of ML in the system helps in making the system learn from its past experiences. This enables the system to keep learning and updating its knowledge which provides better and more accurate results. Use of Apriori algorithm helps system for making smart and efficient conversation by reducing redundant talks. But this system limits to using the NLU and ML at the mediocre level and it can be extended to use ML for learning the ethics of the users and perform analysis on the emotional quotient to understand each human separately. Also, as the system can understand both the English and the Code-mix language for communication which makes user more comfortable to talk to.

On the whole, the use of NLP and ML in domain specific chatbots boosts its capability to understand what the humans actually want to say and accordingly provide more accurate responses.

VI. FUTURE WORK

Chatbot can communicate in English and Code-mix language. The scope can be further expanded to include Indian languages like Hindi, Marathi, Gujrati, etc. so that a wide range of users can use the system. Also, the people who are not comfortable with English can use the system in their mother tongue. Chatbot includes the use of basic ML in the core engine. ML and NLP are used in Chatbot just to understand what the user actually wants to say, convert it into machine understandable language and then predict the most probable disease. The use of ML can be extended to understand the ethics and emotional quotient of the user to provide better medical treatment. Also, there can be an inclusion of suggestion of doctors which can deal with the predicted disease. The suggestions should be provided according to the specialization of the doctor, disease faced by the user and the location of the doctor and the user.

ACKNOWLEDGMENT

This research 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 colleague Ms. Sayali Randive (bachelor's degree in Computer Science from Pune University.) and Mr. Shubham Nahar (bachelor's degree in Computer Science from Pune University) who provided expertise and greatly assisted the research.

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 Ranom 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 Embedding for Elderly Care".
15. Cyril Joe Baby, Faizan Ayyub Khan, Swathi J. N., "Home Automation using IOT and a Chatbot using Natural Language Processing", International Conference on Innovations in Power and Advanced Computing Technologies.
16. Ashay Argal, Siddharth Gupta, Ajay Modi, Pratik Pandey, Simon Shim, Chang Choo, "Intelligent Travel Chatbot for Predictive Recommendation in Echo Platform".
17. Oliver Pietquin, Thierry Dutoit, "Dynamic Bayesian Networks for NLU Simulation with Applications to Dialog Optimal Strategy Learning".
18. Fco Mario Barcala, Jesus Vilares, Miguel A. Alonso, Jorge Grana, Manuel Vialres, "Tokenization and Proper Noun Recognition for Information Retrieval".
19. Soo H. Kim, Chang B. Jeong, Hee K.Kwag, , Chin Y. Suen, "Word Segmentation of Printed Word Lines Based on Gap Clustering and Special Symbol Detection".
20. Xiaofei Li, Xusheng Xie, "Research of Intelligent Word Segmentation and Information Retrieval", 2nd International Conference On Education Tachnology and Computer (ICETC)", 2010.
21. Meishan Zhang, Nan Yu, Guohong Fu, "A Simple and Effective Neural Model for Joint Word Segmentation and POS Tagging", 2018.
22. Liner yang, Meishan Zhang, Yang Liu, Maosong Sun, Nan Yu, Guohong Fu, "Joint POS Tagging and Dependency Parsing with Transition Based Neural Networks.", 2018.
23. Sagar Verma, Sukhad Anand, Chetan Arora, Atul Rai, "Diversity in Fashion Recommendation using Semantic Parsing", 2018

AUTHORS PROFILE

Rushabh Sancheti, is Research and Development Engineer at Dassault Systemes solutions lab private ltd. He has completed a Bachelor of Engineering in Computer Science from Pune University. He has deeply researched in the field of Machine Learning in association with the Centre for Development of Advanced Computing (C-DAC). His current research topics include Natural Language Processing and Machine Learning specifically in the field of Healthcare Domain. He conducts various guidance and technical lectures at the university. He has also headed the Computer Society of India – VIIT Chapter in the year 2017-18. His research in the field of Machine Learning has been praised by Vishwakarma University.

Sunil Upare is Software Engineer at Persistent Systems Private ltd. He has graduated from Pune University with a Bachelor of Engineering in Computer Science. He has conducted various research in the fields of Machine Learning and Natural Language Generation in association with Centre for Development of Advanced Computing (C-DAC). His research in the fields of Machine Learning and Natural Language Generation have been appreciated by Vishwakarma University. He has also done research in the field of Use of 3D Convolution Neural Network for Human Action Recognition in Videos. His current research areas include Machine Learning and Natural Language Generation specifically in Healthcare Domain.

Nivedita S. Bhirud finished her B.E. in Computer Engineering from Pune University (2010), followed by M.E. in Computer Engineering from Mumbai University (2013). She has 7 years of teaching experience in the field of Computer Engineering. Currently, she is working as Assistant Professor at Department of Computer Engineering, Vishwakarma Institute of Information Technology, and Pune. Her research interest include Natural Language Processing (NLP), Machine Learning. She has published over 10 journal papers and 5 conference papers. She is life time member of International Association of Engineers (IAENG).

Subhash Tatale has completed B.E. in Computer Engineering from Pune University, M.Tech. in Information Technology from Pune University and pursuing Ph.D. in Computer Science & Engineering. He is having 12 years of teaching and IT industry experience. His research area includes Software Engineering, UML and Artificial Intelligence. He has published more than 15 papers in national and international conferences and journals. Now, he is working as an Assistant Professor at Vishwakarma Institute of Information technology, Pune. He is lifetime member of International Association of Engineers (IAENG). Currently he is also working on research project with Board of College and University Development (BCUD).