Grenze International Journal of Engineering and Technology, July Issue

Cross Language Application for Disease Prediction

Shivani Butala[1], Advait Lad[2], Niket Parekh[3] and Kiran Gawande[4]
[1-4] Sardar Patel Institute of Technology/Department of Computer Engineering, Mumbai, India
Email: shivani.butala@spit.ac.in, advait.lad@spit.ac.in, niket.parekh@spit.ac.in, kiran_gawande@spit.ac.in

***Abstract*—The infrequent and inadequate medical facilities for the people residing in the remote villages of India due to the lack of medical experts is a glaring medical issue facing the country. The proposed cross language application aims to solve the aforementioned problem with the use of technology by utilizing Translation techniques and Machine Learning models. With the usage of the proposed application, the users can vocally input their experienced symptoms into the application in their local language and get as output, the disease they possibly might be suffering from within seconds. To increase the accuracy of the results, an ensemble approach is followed in the paper which is a combination of five Machine Learning Models and a majority decision technique is applied to get the most accurate results. The proposed application is able to predict the disease with an accuracy of 90% in return of the symptoms provided by the user.**

***Index Terms*— Disease, Symptoms, Local Language, Natural Language Processing, Machine Learning, Chatbot.**

I. INTRODUCTION

Access to a proper medical health case is one of the basic necessities of man. However, it is a known fact that a lot of underdeveloped regions in our country suffer from a lack of immediate medical help. According to a recent article in Hindustan Times, in India, the ratio of doctors to people is very small. To be more specific, there is a single government medical expert for every 10,189 accounted people, one government supported emergency clinic bed for every 2,046 individuals and one government subsidized medical clinic for every 90,343 individuals. This proves it very difficult for the people from underdeveloped regions to get access to good medical help quickly. By and large, India has just a little more than one million current medication (allopathy) specialists to treat its populace of 1.3 billion individuals. Of these, just around 10% work in the general wellbeing and the public sector area. The lack of this basic necessity leads to either no timely diagnosis of the disease or at the best, wrong diagnosis of the disease which can have serious repercussions. So if we compare India's medical care system to some of established medical care systems, India has a lot of ground to cover, especially in the space of Emergency Medical Care System (EMS). A report in Business Standard suggests that nearly 5 million people die due to medical errors or negligence which leads to wrongful diagnosis of the disease. This puts medical errors in the top 10 killers or the nation. However, with the development of technology, the problem of lack of proper medical treatment can now be tackled. Even if the residents of the underdeveloped region cannot reach doctors or hospitals easily, the medical help can be taken to these people. So with a view to use technology to solve a social issue, we propose a chatbot that first will take the input (symptoms) from the patient in their local language and then predict the disease based on

Grenze ID: 01.GIJET.6.2.10

those symptoms using Natural Language Processing and Machine Learning.
The rest of the paper is organized as follows - Section II describes the background work and literature review. Section III delineates the methodology of the proposed system. Section IV illustrates the result analysis and discussion. Section V mentions the conclusion and the future scope.

II. Literature Review

Jayshri A.Todase and S. Shelkel. in their paper they have proposed a new Machine Translation System for translating Devanagari to English language. In their method, a rule-based approach is used to translate multiple Devanagari sentences to equivalent English sentences. The process used in the proposed methodology is firstly identifying parts of speech, then tokenization of words followed by identifying the English meaning of those Marathi words and then getting a direct translation of the sentence. The translated English sentence is then connected to English grammar rules to have proper meanings [1]. Jatin C.Modh, Saini and Jatinderkumar R. in their survey have given a brief description of some of the approaches that have been implemented to achieve successful Machine Translation. Along with that, they have also discussed the work done on translation of Gujarati language in particular [2].
Harini D K and Natesh M. propose machine learning algorithms to effectively predict diseases of various kinds. Additionally, to overcome the aforementioned problem of missing data, latent factor models are used by them. Also, the information that is received for analysis from the hospitals is in both structured and unstructured format. To nullify the issues because of those, a new multimodal disease risk prediction algorithm is designed which is based on convolutional neural networks (CNN-MDRP) [3]. Jayashree Nair. K. A. Krishnan and R. Deetha. propose an HMT Architecture by the combination of declension Ruled Based MT and SMT systems. The proposed system includes nine steps - Splitting, Parsing, POS tagger, Declension tagger, Sentence rules, Reordering (SOV/SVO), Lexical dictionary (wordnet dictionary), Translation and Morphological analyzer [4]. Md. Tahmid Rahman Laskar, Md. Tahmid Hossain, Abu Raihan Mostofa Kamal and Nafiul Rashid make use of decision tree approach for symptom-based disease prediction which is based on the user input. Average accuracy of 88% was obtained by the system but it can be further improved by considering larger datasets and building deeper decision trees [5].
Sandeep Saini and V. Sahula conducted a survey of the Machine Translation Techniques pertaining to Indian Languages and compared the rule based, corpus based and hybrid machine translation methods along with their subtypes. The rule based techniques include the Direct MT, interlingual MT and the Transfer based MT. The Direct MT technique can be used for languages which are similar as word by word translation is conducted. Interlingual MT constructs an abstract intermediate representation between source and target languages. Database of translation rules are used in Transfer based MT. Corpus based techniques have two types - Statistical MT and Example based MT.[6]. Min Chen,Y. Hao, K. Hwang, L. Wang and L. Wang proposed the problem of missing data that has been taken care of. Latent factor model is used to reconstruct the missing data and there has been some focus to region specific disease outbreaks as well. The various outbreaks and viruses in a particular region has been considered as a part of the dataset for better accuracy of the model. So, both, the structured and the unstructured data is considered and a convolutional neural network based disease prediction model (CNN-MDRP) is developed. An accuracy of 94.3% is achieved using this model and run time has been considerably improved when compared with similar kinds of other prediction algorithms [7].
Son Doan, Elly Wang, Sameer Tilak and Manabu Torii proposed a novel methodology for ailment status distinguishing proof, one of the PHIs, in clinical NLP content mining dependent on profound neural models. Likewise, for gauge examination, the conventional/shallow AI models were additionally considered. The profound neural models, in view of single station N-gram word embeddings and CNN gathering seem promising as they are adapting consequently, with programming including revelation and learning, without a requirement for manual hand creating and highlighting designing of content highlights [8]. Kunal Rajput, Girija Chetty and Rachel Davey assessed a way to deal with extricating causal relations from tweets utilizing natural language handling (NLP) strategies. They concentrated on three well being related points: 'stress', 'sleep deprivation', and 'cerebral pain'. They proposed a lot of lexico-syntactic examples dependent on reliance parser yields to extricate causal data. An enormous dataset comprising 24 million tweets were utilized [9].
Dhiraj Dahiwade, G. Patle and E. Meshram propose a disease prediction system based upon the symptoms given by the user based on the K-Nearest Neighbour (KNN) and Convolutional neural network approaches. The dataset for the prediction was taken from the UCI machine learning website. The dataset undergoes pre-

processing after which it is split into training and testing datasets. The training dataset undergoes Feature Selection after which it is subject to Ensemble Classification Techniques - KNN and CNN. The two algorithms are further compared on the basis of the performance time, accuracy and memory requirements. Convolutional neural networks prove to be more accurate with an accuracy of 84.5%. Additionally, It is also more computationally efficient and requires a performance time of approximately 11000 milliseconds while that of K-Nearest Neighbours is close to 13000 milliseconds [10]. Allen Daniel Sunny et. al developed a system to predict diseases in the system using K - nearest neighbours, Decision Tree, Naive Bayes approach and the Apriori methods. Scikit library implementations of KNN and decision trees were used which did not give good results because of the many-many structure of the symptom-diseases. Naive Bayes was implemented but was limited in the results due to the conditional probability approach. Apriori algorithms gave the best results as it based on the frequent item set approach and also listed the probability of occurrence of multiple diseases [11].

Manish Bali, Samahit Mohanty, Subarna Chatterjee, Manash Sarma and Rajesh Puravankara propose a 'Diabot' which is a medical chatbot using Ensemble learning which is a hybrid algorithm which combines the strengths of several Machine Learning prediction algorithms and does not include their weaknesses. The diabot - diagnosis chatbot makes use of Natural Language Understanding (NLU) using RASA NLU Engine to interact with the patient and the Front End interface ie. the React UI platform. The Ensemble classifier which includes Multinomial Naive Bayes, Bernoulli Naive Bayes, Support Vector Machine, Random Forest and Decision tree has an average accuracy of 86% [12]. Papachristou and D, Barnaghi P expanded on the idea of Support Vector Regression efficiency along with Neural Network based Non-linear Canonical Correlation (n-CCA) for predicting the seriousness of the previously mentioned symptoms between two distinctive time focused during a pattern of chemotherapy (CTX). Their outcomes exhibit that these two strategies delivered proportionate outcomes for each of the three symptoms. These kinds of prescient models can be utilized to recognize high hazard patients, teach patients about their indication experience and improve the planning of preemptive and customized symptoms management mediations [13].

Bohra et. al have reviewed the applicability of Bayes algorithm for the prediction of disease from its respective symptoms. System used Naïve Bayes algorithm and depending on the symptoms predicted the diseases and for a normal person it predicted the daily hygiene diet and routines which he can follow. Users were also able to contact the specialist doctors nearby [14]. Prem, Guru, M. A. Hema, Laharika Basava and Anjali Mathur apply different AI innovations to discover the yield. Their exploration work manages decision trees, Naive Bayes hypothesis, artificial neural network and k-mean clustering and random forest algorithm. Disease improvement relies upon three conditions-have plants helpless to illness, ideal condition and reasonable pathogen. The nearness of every one of the three conditions is an unquestionable requirement for a sickness to happen. 11 traits and 310 columns of information have been thought of and examination of strategies like decision tree, Naive Bayes, Neural system and various plots like box plot, bar plot are performed [15].

III. Experimentation Of Multi - Linguistic Chatbot

Three modules namely the Dialgflow module, Natural Language Processing module and Prediction module were used for the development of the chatbot. The Dialogflow module is responsible for taking the user input in the form of audio and in the local language of the end user and providing the possible diseases as a response. The end user input is going to be in a raw unstructured format and will consist of certain symptoms. This data is provided to the Natural Language Processing module which is responsible for pre-processing the user input and extracting the relevant keywords from the raw data. Finally, the prediction module will utilize the below mentioned dataset and Machine Learning models to predict the disease as output while taking the user symptoms as the input. This output will then be given back to the Dialogflow module which will provide the result to the end user through the chatbot. The system design can be seen in Figure 1.

A. Google DialogFLow API for chatbot generation

The proposed chatbot was implemented with the help of the Google Dialogflow API. A Dialogflow agent is a Natural Language Processing module that helps to build and design conversational platforms. It takes the input from the user in the form of text or audio and translates it into a systematic and structured format which can be used for various services. Dialogflow utilizes Google Machine Learning which helps in Natural Language Understanding of the user's intent. The main purpose of using the Dialogflow API was to make

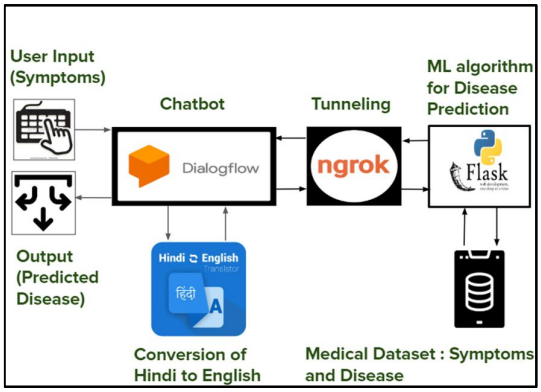

Figure 1. System Design

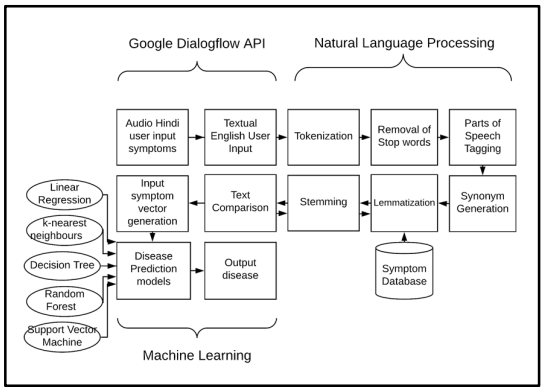

Figure 2. Flowchart of proposed chatbot

use of its inbuilt handling of over 20 languages since the focus of the proposed system is to build a chatbot in the local language of people in the rural regions. For the proposed system, the chatbot will be built using the Hindi language. Another important feature of the API is its ability to handle audio user input ie. voice interactions and generate voice outputs using the Google Speech-To-Text engine which is another core component of the proposed chatbot. The user input which is in audio format and in Hindi language will be converted into text and handled internally by Dialogflow. A Dialogflow agent has several major components but for the scope of the proposed system, mainly 2 components are used which are described as follows:

a. Intents: Intents are used to map the user inputs into responses. The user input for one conversational turn is categorized into an 'Intent'. Every end user expression is classified into the best matching intent by Dialogflow. Here, 'userinput' and 'result' are the intents created. The 'userinput' intent is used for the end user expressions containing the symptoms which are used for prediction of the disease which is handled by the prediction module. Once the prediction module has finished computing, the 'result' intent will deliver the response to the user. Additionally, Dialogflow has a 'Default Welcome Intent' which will handle the basic user interactions such as 'Hi', 'Hello', 'How are you' etc. A 'Default Fallback Intent' is also available to handle user inputs which did not match any intent. Every intent consists of four basic parts:

i. Training phrases: Every intent consists of certain example phrases for the end-users expressions. When a user input resembles one of these phrases, Dialogflow will match the expression to the corresponding intent. Dialogflow handles other similar phrases using its built-in machine learning module. The 'userinput' intent has a list of symptoms present in the dataset as the training phrases. Thus, whenever the user expression consists of a symptom which is present in the training phrases or any other similar phrase, it will be matched to the 'userinput' intent. The 'result' intent is simply to provide the results to the user and has 'yes', 'no', 'okay', etc. as training phrases when the user is asked if he/she is ready to receive the results.

ii. Action: An action can be defined for each intent. When a user input is matched to an intent, Dialogflow conveys the action to the system which can be used to trigger other actions. The 'userinput' intent has 'input.mp' as its action and the 'result' intent has 'input.result' as the action. Once the intent is triggered, we can obtain the action name as a parameter and if the action name is 'input.mp', the user expression which consists of synonyms will be used as the input data for the prediction module. If the action name is 'input.result', the computed results will be sent back to the Dialogflow API from the back end.

iii. Responses: It is the output provided by Dialogflow in the form of text, speech, or a visual response to the end-user. The responses can be used to ask the end user for more information, provide certain answers or even end the conversation. The 'userinput' intent asks the user if he/she is ready to receive the results as a response whereas the 'result' intent gives the actual predicted disease as the response.

b. Fulfillment: Fulfillment is used to give more dynamic responses to the user input. Every intent has the option to enable a fulfillment which when enabled will call upon a defined service for the action of the called upon intent. Fulfillment is deployed as a webhook which are user defined HTTP call backs. Webhooks are called when certain criteria is fulfilled such as code being pushed onto a repository or new information posted on a site. The source site will make an HTTP request to the URL assigned for the webhook when such an event occurs. Fulfillment enabled intents use webhooks to fetch results from the server by passing the user data of the matched intent through the webhook service. The Flask framework was used to create a python webhook and a flask app was developed. The 'ngrok' tunnelling software was used to run the Flask app and

relay the information between the Dialogflow API and the local development environment. The command ‘ngrok http <port-number>’ is used where the port number stands for the port defined in the Flask app. The result is a unique Ngrok domain name which is used as the webhook URL by the Dialogflow API in the Fulfillment section. As a result, Fulfillment provides connectivity between the Dialogflow module and the other two modules. Thus, the user input obtained via the chatbot through the Dialogflow API will be given to the local environment wherein it will be subjected to the Natural Language Processing and Prediction module.

B. Natural Language Processing For Data Preparation

Once the vocal input from the user is converted into Hindi text using Dialogflow, it is translated into English using the python library ‘translator’. Using the ‘langdetect’ python library, the language of the user input can be identified and then further communications can be carried out using the identified language. The translated text is then acted upon by a few Natural Language Processing techniques to bring it into a more standard format and to extract the more vital words such as symptoms, duration of symptoms and any other keywords which might be useful for disease prediction. The techniques which are used in the proposed cross language application are as follows:

a. Tokenization : The input from the user is a sentence or more than one sentence with multiple words stringed together.In order to extract keywords, these sentences need to be first broken down into individual words which is done by tokenization. For eg. “I have fever. I am shivering too.” The tokens for this input will - ‘I’, ‘have’, ‘fever’, ‘.’, ‘I’, ‘am’, ‘shivering’, ‘too’ and ‘.’. This makes work easier for the following process as dealing with individual tokens is less complex than dealing with longer sentences.

b. Stop Word Removal : The tokenized words or tokens are then passed for stop word removal. This list of words have certain stop words which are of no significance or value to us but are just added in the sentences to make the sentence grammatically accurate. Such stop words are removed in the beginning itself to avoid processing of extra words in the following steps. ‘The’, ‘a’, ‘an’, ‘and’ are some of the common stop words which are removed.

c. Parts of Speech Tags : Once we have a list of individual words, those words are tagged according to the part of speech they belong to by calling NLTK's .pos_tag function. There are several inbuilt tags available in Python NLTK libraries, the most common of which are noun, verb, adjective, conjunctions and digits. From observation, it can be concluded that the only words that are useful for disease prediction are the words that are tagged as nouns, adjectives, verb or adverb. So, only those words are passed forward to further processing.

d. Synonym Generation : Since one symptom can have more than one word to describe it, for highly accurate prediction of disease, it is important to consider all possible synonyms of the symptom and not just the symptom itself. Every word is passed through the .synsets function and the resulting list of synonyms are stored in a list of lists and passed forward.

e. Stemming : The process of reducing a word to its most simplest form by removing all kinds of suffixes and prefixes is done during Stemming. All the words along with their synonyms are passed through the .stem function and the stemmed words are passed on for lemmatization. Stemming is also performed on the column names of the data set to ensure an accuracy during the formation of input vectors later.

f. Lemmatization : Lemmatization is the process of achieving the root form of words. All the stemmed words are lemmatized here into their root forms. Lemmatization is not the most crucial step in text prepossessing but it is done to achieve greater accuracy.

C. Prediction of diseases using Symptoms

Dataset Description: The prediction of diseases by the proposed cross language application is a result of the machine learning models that are trained on a large dataset. The dataset that is considered for training the models for the proposed application is obtained from Kaggle - the world's data science community, which provides datasets related to a range of topics across domains. The dataset has 133 columns with each column name indicating the symptoms and the rows indicating the test cases for various diseases. The contents of the rows of the dataset are binary, i.e., the intersection of the row and column will have value ‘1’ if the said symptom, which is the column name, is indeed a symptom of the considered disease in the test case for that row. If the disease does not have the column name as one of its symptoms, the intersection of the row and column in consideration will have the value ‘0’. The dataset is split into two parts - training and testing. The training dataset consists of 4920 rows and the Testing data which is used for testing the accuracy of the models has 40 rows.

Dataset Preparation: Some of the symptoms contained more than one word and were separated by underscores. Such symptoms were split into single word symptoms using underscore as the delimiter. Duplicate column names were integrated into a single column and the resultant column will have a value given by the logical 'OR' operation performed row wise on the duplicated columns. The final result is a dataset with 181 columns.

Generation of input vector: Whenever a user gives an input to the system, the input goes through translation (to convert the input into English) and a number of Natural Language Processing techniques to make it easier to use. What we get as a result are a list of words which are either symptoms or they are words which will have some weightage during the prediction of disease. The database contains a huge list of diseases along with their respective symptoms which are stemmed during the Stemming process. To conclusively predict the disease, it is necessary to match the stemmed and lemmatized symptoms received from the user input to the stemmed symptoms in the dataset. This is done using an 'input vector'. The input vector is initialized having 'N' zeros, where 'N' is the number of symptoms in the dataset. Then, the stemmed input symptoms are passed through the list of symptoms in the dataset and the index of the matched symptom from the dataset is captured. The recorded index is changed to '1' in the input vector. This process is repeated for all the symptoms received from the user. The end result is a user input vector which is passed to the prediction module as the test tuple. But before being passed as the test tuple, the sparsity of the input vector needs to be handled.

Handling Sparse Vectors: The dataset used for research has 181 symptoms as columns and the user input vector satisfies 5 to 8 symptoms on an average. This number proves out to be too low to give out accurate results which is the most vital part of the proposed system. Thus, this sparse vector is made denser by using Implication techniques. The available 181 symptoms are initially clustered into 7 clusters using the K-Means algorithm. The sparse input vector is then passed through these 7 clusters to find the best fit for the input vector. The cluster which is found to be the closest fit for the input vector is selected for increasing the density of the input vector. The symptoms from the selected cluster are appended to the input vector and the resulting vector is a much denser vector with a greater number of 1's. This improved input vector is passed on as the test tuple to the classifiers. The 181 columns are the input variables to each of the classifiers and there is 1 output variable which has 42 possible values which are the name of the diseases. The prediction module consists of 5 Machine Learning models namely - Random Forest, Decision Tree, K-Nearest Neighbours, Support Vector Machine and Logistic Regression, each of which is trained autonomously on the dataset. The python 'sklearn' library is utilized for training and testing the classifiers and calculating the probabilities of the predicted diseases. The generated input vector having 181 elements is then classified into a disease category by each of the classifiers and the final choice is made by using the concept of majority voting. Thus, if all five classifiers agree on the predicted disease, the output given to the user through Dialogflow will be a single disease (returned in the local language of the user), otherwise all the predicted diseases will be returned to the user with the probability of the occurrence of each disease.

IV. Result Analysis And Discussion

The proposed method can be highly beneficial for the people in the rural regions since the entire chatbot will communicate in the preferred language of the user. Consider the end user input to be in Hindi as shown in Figure 1.

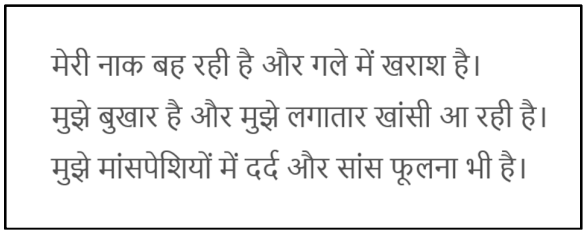

Figure 3. End-user input in Hindi

This user expression contains several symptoms which the user will communicate to the chatbot through the Dialogflow API using either audio or textual format. Dialogflow handles the conversion of audio data into

textual data. This user expression will be matched to the ‘user input’ intent since the data contains several symptoms such as ‘runny nose’, ‘fever’, ‘sore throat’, ‘muscle ache’, ‘shortness of breath’ and ‘cough’ which are present as the training phrases of the ‘user input’ intent. Once the user phrase is matched to the intent, the webhook will relay this data to the local development environment. This raw unstructured data is acquired using a json query at the back end i.e. the Flask app. The data will first be converted into English using the ‘translator’ library and gives the result - ‘I have a runny nose and a sore throat. I have a fever and I am constantly coughing. I also have muscle aches and shortness of breath.’
These sentences are then passed through the tokenizer function followed by the removal of stop words. The output tokens obtained are as follows :‘I’, ‘runny’, ‘nose’, ‘sore’, ‘throat’, ‘.’, ‘I’, ‘fever’, ‘I’, ‘constantly’, ‘coughing’, ‘.’, ‘I’, ‘also’, ‘muscle’, ‘aches’, ‘shortness’, ‘breath’, ‘.’. These tokens are then tagged individually with their respective parts of speech. Table I highlights the tokens and the part of speech they are tagged to by the .postag function.

TABLE I. PARTS OF SPEECH TAGGING, SYNONYM AND INPUT VECTOR GENERATION

Token	Parts of Speech Tag	Tag Full Form	Synonym	Matched Column from Dataset
I	PRP	Personal Pronoun	-	-
runny	NN	Noun (Singular)	runny, fluid	runny, fluid
nose	RB	Adverb	-	-
throat	NN	Noun (Singular)	pharynx, throat	throat
sore	NN	Noun (Singular)	afflictive, huffy, sore, raw, painful, mad, tender, sensitive	sore, pain
fever	NN	Noun (Singular)	pyrexia, febricity, fever, febrility, feverishness	fever
constantly	RB	Adverb	-	-
coughing	VBG	Verb (Gerund)	cough, coughing	cough
Full Stop	-	-	-	-
also	RB	Adverb	-	-
muscles	NN	Noun (Singular)	muscular tissue, muscularity, muscle, sinew, heftiness, brawniness, musculus, brawn, muscleman	muscle
aches	NNS	Noun (Plural)	languish, yen, yearn, aching, ache, pine, suffer, hurt, pain	pain
shortness	NN	Noun (Singular)	shortness, curtness, truncation, gruffness, brusqueness, abruptness	-
breath	NN	Noun (Singular)	shortness of breath	breathless

The accuracy of .postag is function is 71.42% since10 of the 14 tokens are correctly classified. The tokens ‘runny’, ‘nose’, ‘sore’ and ‘also’ are incorrectly tagged as Noun, Adverb, Noun and Adverb respectively. Once all the tokens are tagged with their respective parts of speech, only the nouns, adjectives and verbs are selected for further processing. The selected tokens are passed through the .synsets function and the output obtained are the synonyms of the tokens. The output is recorded in Table II below. The generated synonyms of each word along with the original word are then subjected to stemming and lemmatization to obtain their roots. Thus, the stemmed form of ‘fever’ becomes ‘fev’, ‘trouble’ becomes ‘troubl’, ‘painfulness’ becomes pain’, ‘sore’ becomes ‘sor’ whereas the stemmed form of ‘runny’ and ‘throat’ is the original word itself. Once the roots are obtained, they are compared with the roots of the symptoms present in the dataset. If a match is found, the value of the input vector at the position where the corresponding matched symptom is present is marked as ‘1’, the rest of the positions are given the value ‘0’. The matched word from the dataset for each of the user uttered words and their synonyms is shown in Table I. It can be seen that ‘shortness’ did not match with a synonym from the dataset. The rest of the words are matched and thus we get an input vector with the value ‘1’at 8 positions as seen in Table I.
The input vector has a total length of 181 and will be passed to the Machine Learning classifiers as the test tuple. The accuracy of the Machine Learning classifiers can be seen in Table II and in Table III. Table II illustrates the results for the classifiers without the improved denser input vector. It can be seen the Logistic

Regression is able to predict the disease with the highest accuracy of 90.2% followed by K-Nearest Neighbours, Support Vector Machine, Random Forest and Decision Tree With accuracies of 63.41%, 58.53%, 56.09% and 46.34%respectively. Even with the varying accuracies, all the classifiers were able to accurately predict the test case disease which was 'Coronavirus'.

The input vector is now made denser by the use of Clustering and Implication as mentioned earlier in the paper. On introducing the resulting improved denser input vector, the accuracies for the classifiers are observed to rise significantly. Table III shows the results when the denser input vector was used by the classifiers. From both Table II and Table III, it can be seen that Logistic Regression has the highest accuracy with or without the denser input vector. Even though the order for the rest of the classifier changes with the introduction of the denser input vector, for either case, all the classifiers give a uniform predicted disease i.e. 'Coronavirus'.

TABLE II. EVALUATION METRICS OF ML CLASSIFIERS WITH SPARSE INPUT VECTOR

Classifier	Accuracy	Precision	Recall	F-1 Score
Logistic Regression	0.9024	0.87	0.9	0.88
K Nearest Neighbours	0.6341	0.54	0.63	0.56
Decision Tree	0.4634	0.43	0.49	0.45
Random Forest	0.5609	0.40	0.51	0.43
Support Vector Machine	0.5853	0.48	0.59	0.50

TABLE III. EVALUATION METRICS OF ML CLASSIFIERS WITH DENSE INPUT VECTOR

Classifier	Accuracy	Precision	Recall	F-1 Score
Logistic Regression	0.9245	0.89	0.92	0.90
K Nearest Neighbours	0.8341	0.81	0.83	0.81
Decision Tree	0.7634	0.73	0.79	0.75
Random Forest	0/8609	0.80	0.85	0.82
Support Vector Machine	0.8853	0.88	0.89	0.88

V. CONCLUSIONS AND FUTURE SCOPE

In this paper, we used our knowledge of technology to solve a social issue and enhance social health. The sole purpose of the chatbot is to fill the void that is caused by the lack of doctors and the proposed system is capable of conversing with the user to diagnose the patient based on the symptoms provided. The system integrates Natural Language Processing with Machine Learning algorithms for disease prediction and it also includes comparison of different algorithms to get more accurate results and predictions. Later, we could modify the chatbot to include more regional languages, thereby, increasing the reach of the application.

REFERENCES

[1] J. A. Todase and S. Shelke, "Script Translation System For Devanagari To English," 2018 International Conference on Information , Communication, Engineering and Technology (ICICET), Pune, 2018, pp. 1-4.

[2] Modh, Jatin C., Saini, Jatinderkumar R., "A Study Of Machine Translation Approaches For Gujarati Language," International Journal of Advanced Research in Computer Science, Jan/Feb2018, Vol. 9 Issue 1, pp. 285-288.

[3] Harini D K, Natesh M, "Prediction Of Probability Of Disease Based On Symptoms Using Machine Learning Algorithm," International Research Journal of Engineering and Technology (IRJET), 2018, Vol.5 Issue 5, pp. 392-395.

[4] J. Nair, K. A. Krishnan and R. Deetha, "An efficient English to Hindi machine translation system using hybrid mechanism," 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, 2016, pp. 2109-2113.

[5] Md. Tahmid Rahman Laskar, Md. Tahmid Hossain, Abu Raihan Mostofa Kamal,, Nafiul Rashid : "Automated Disease Prediction System (ADPS): A User Input-based Reliable Architecture for Disease Prediction," International Journal of Computer Applications (0975 – 8887) Volume 133 – No.15, January 2016, pp. 24-29

[6] S. Saini and V. Sahula, "A Survey of Machine Translation Techniques and Systems for Indian Languages," 2015 IEEE International Conference on Computational Intelligence & Communication Technology, Ghaziabad, 2015, pp. 676-681.

[7] M. Chen, Y. Hao, K. Hwang, L. Wang and L. Wang, "Disease Prediction by Machine Learning Over Big Data From Healthcare Communities," in IEEE Access, vol. 5, pp. 8869-8879, 2017.

[8] Son Doan, Elly Wang, Sameer Tilak and Manabu Torii, “Using natural language processing to extract health-related causality from Twitter messages,” 2018 IEEE International Conference on Healthcare Informatics Workshop.
[9] Kunal Rajput, Girija Chetty and Rachel Davey, “Deep Neural Models for Chronic Disease Status Detection in Free Text Clinical Records,” 2018 IEEE International Conference on Data Mining Workshops (ICDMW).
[10] D. Dahiwade, G. Patle and E. Meshram, “Designing Disease Prediction Model Using Machine Learning Approach,” 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2019, pp. 1211-1215.
[11] Allen Daniel Sunny, Sajal Kulshreshtha, Satyam Singh, Srinabh, Mr. Mohan Ba, Dr. Sarojadevi H. “Disease Diagnosis System By Exploring Machine Learning Algorithms,” International Journal of Innovations in Engineering and Technology (IJIET), May 2018, Volume 10, Issue 2, pp. 14-21.
[12] Manish Bali, Samahit Mohanty, Subarna Chatterjee, Manash Sarma, Rajesh Puravankara, “Diabot: A Predictive Medical Chatbot using Ensemble Learning,” International Journal of Recent Technology and Engineering (IJRTE), Volume-8 Issue-2, July 2019, pp. 6334-6340.
[13] Papachristou N, Puschmann D, Barnaghi P, “Learning from data to predict future symptoms of oncology patients,” PLoS One. 2018;13(12):e0208808. Published 2018 Dec 31. doi:10.1371/journal.pone.0208808
[14] Bohra, Himdeep & Arora, Amol & Gaikwad, Piyush & Bhand, Rushabh & Patil, Manisha. (2017), “Health Prediction and Medical Diagnosis using Naive Bayes,” IJARCCE. 6. 32-35. 10.17148/IJARCCE.2017.6407.
[15] Prem, Guru, M. A. Hema, Laharika Basava and Anjali Mathur, “Plant Disease Prediction using Machine Learning Algorithms,” International Journal of Computer Applications, Volume 182, No. 25, November 2018, pp. 1-7.