Disease Diagnostic ChatBot

Team Ultron(Group 9)
Kanhaiya Rathi(130101033), Kenil Tanna(130101036) , Mayank Gupta(130101086), Naman Jain(130101051)

***Abstract*— Health care is one of the most important sector which also has high cost and high risk associated with it. This paper focuses on developing an AI bot that can help user to identify and diagnose possible disease if he feels unwell. User can pose the symptoms to the bot and bot can ask follow up questions and determine possible diseases. We explain the reason for choosing the bot framework that we have used and give a unique conversational flow diagram to show how it will function. We have used both Random Forests and Naive Bayes method for disease diagnosis and conclude that Naive bayes even though being a simpler method performs better for our dataset . Our bot also provides follow up questions to the patient regarding the symptoms to better diagnose the disease and provides the nearest location of the hospital.**

I. INTRODUCTION

It's difficult to remove a doctor-patient relationship using AI bots, but it can be used to gain more information about what kind of disease a patient is having or the symptoms that the patient is suffering with.

We focus on developing one such chat bot, which takes symptoms from the user and than asks for different possible symptoms as follow up questions. Based on the symptoms, we tried different algorithms such as Random decision forests, Naive Bayes to identify the most possible disease the patient can have and then display it with the corresponding probabilities which is calculated based on historical data.

These are the problems which one can relate to while using the chat bot to identify the disease:

Firstly, although there are some standards for some diseases, having those symptoms doesn't mean that patient have that disease. Also, single symptom can be a sign of multiple different diseases. Moreover, because of the questionnaire about Disease A, the patient might eliminate the chances of Disease B or C, and he might try to give wrong paths and mislead the physician as he might have gotten convinced that he/she has disease A. Many diseases also have different stages, and during those stages those symptoms might get changed, which also leads to problem in estimation. Patient personality is another concern. Some might exaggerate their condition. It might be possible that patient is not sure about the disease but still try to give any details, and if he suspects some disease than he might give misleading information. There are also some habits (normal, cultural habits) which may be cause of a symptom but not a disease. Likewise, currently used medication, past operations are needed to be considered. Medical imaging and testing methods are needed for diagnosis.

Despite all of the above problem, we try to come up with the probable disease from the set of symptoms with the motivation that the patient will get to know more about the condition in which he is going through. A software or formal questionnaire to help doctors and others cover all the bases and diagnose patients would be helpful. And one can use this bot's prediction to get alerted, and meet the doctor soon if required.

II. RELATED APPS AND METHODS

Symptom Checker is one such already existing app, which tries to predict the cause for symptoms and then tells what steps to try next. They take different factors such as gender, age, weight, etc. into account along with the symptoms. They also use the historical stored symptoms as well as the family history checkup and take it into account while predicting the possible diseases. They also state the possible reasons for why the patient should or should not meet a doctor, and provides lots of information about the symptoms as well as possible diseases.

A lot of research is going on in this field from a very long time. One of the naive methods is to create database for every possible disease and there corresponding symptoms. Though this solution has a drawback in terms of it's time and space efficiency. This paper uses the symptoms with the ratings given by the patient. They predict the disease based on the severity of the patient and they also do cure time prediction based on the real-life data. To each symptoms they give different coefficient depending on the disease. They use the similarity measures to identify that which symptoms can lead to what kind of diseases.[1]

People also have used different kind of techniques, and one such is using the neural network. [2]

They have used artificial neural network for typical disease diagnosis. They selected symptoms of eight different diseases and created a dataset of such few hundred cases. After putting the data into machinable format MLP neural network is applied. Based on automatic medical diagnosis system, they find the role of selection of effective symptoms and the advantages of data fuzzification on neural network very vital. Using the dataset from Health care cost and Utilization Project (HCUP) which is available publicly, National Inpatient Sample (NIS) data, they have used random forest to classify the symptoms into eight disease classes. There Random Forest ensemble techniques outperforms SVM, bagging, boosting algorithms. Due to highly imbalance in dataset, they uses random sub-sampling in their learning approach. One problem in this paper is they have only eight classes of diseases in their dataset. [3] They have used data-mining techniques for their disease prediction system. Mafia

algorithms are used for classification purposes, the data is estimated using entropy based cross validations and partition techniques.[4]

A. Survey of Different Chat Bot Frameworks:

A chat bot is a program which conducts conversation with a human being to make it seem as if the other person is talking to another person. They are generally used for various different purposes like customer support, information acquisition, doing various bookings like hotel and flights.

Eliza and Parry were some classic chat bots that were used to simulate strictly typed conversations. Recently there has been a surge in the demand for on-line assistants for various tasks as the use and the extent of the Internet is increasing. There are chat bots for sports highlights to ordering food on-line to booking your travel plans. There are various applications out there like Facebook Messenger, Kik, Slack, Telegram etc. which are competing for monopoly in the virtual assistant or the chat bots market. For our purpose here we use the facebook messenger platform because it is easily accessible and it provides a great messenger api to leverage its different functionalities.

The general chat-bot architecture should be to understand what the user says and respond with either static or dynamic responses or generate some responses using deep learning methods or otherwise and in doing all of this it should also maintain the context of the conversation. We divide the type of chat-bot frameworks into three parts:

1) No Programming Platforms: These are non-technical user oriented platforms, where machine learning or natural language processing and the developer does not need to worry about the coding skills. There are various platforms for this like ChatFuel, Octane.ai, Motion.ai etc. The pros of these kind of platforms are that there is a low learning curve and it is very easy to build one. But the cons are glaring the most important being they are not able to address complex applications or even extract simple things such as cities as they do not use natural language processing.

2) Conversational Platforms: Here the aim is to have a conversation with the person and not consider a task-oriented scenario. The main drawback of these platforms is that it is difficult to scale the patterns if they are manually build.

3) Platforms that are the Industry Standards: We look at some platforms that have become the industry standard or are supported by the major players. We looked at the following frameworks or platforms:

- **Lex**: This is a framework provided by Amazon which aims at building a conversational bot mainly for alexa. Now as we are targeting Facebook Messenger Platform this is not much of a use for us.
- **Watson**: This is a IBM provided platform which is powered by the Watson AI that IBM is currently developing and working on. This framework was too complex for our requirements and not as much as we needed it to be.
- **LUIS**: Luis was introduced by Microsoft in 2016 with main aim to provide a framework to create bots for skype. Though we found that Luis has several limitations like it allows to create only 10 entities per application and it is difficult to integrate it with the facebook messenger platform. Also, it had to be semi-learned or trained to identify the
- **Wit.ai**: This chatbot platform was recently aquired by Facebook, and it uses the concept of stories to create conversational flows. It allows the controlling of flow, has a section of "Understanding" which allows the chatbot to train on the examples. But the major problem we observed here was that the GUI was very buggy, and it is proves difficult control the flow of the conversation.
- **Api.ai**: This is a framework is the best one we could find in terms of the utility, ease of use and the function we wanted it to perform. Using intents and contexts, it allows a very powerful way of modeling large and complex flows. Also, it provides easy integration capabilities with several platforms such as facebook messenger,slack,telegram etc. Another thing we found was that it had a small talk support etc. which makes it a better experience for the person using it.

III. DATASET EXPLANATION

We have collected the dataset from the following source: Dataset

It contains information about Diseases with the corresponding symptoms with count of disease occurrences. This knowledge base is automatically generated by methods based on information in discharge summaries(text format) of patients at New York Presbyterian Hospital in 2004. From this particular site we downloaded the data and parsed it in the required format. Here are the dataset information:

- Number of relations: 2,130
- Number of diseases: 148
- Number of symptoms: 405

IV. OUR CHATBOT

We have built our chat bot using api.ai, chat-bot framework of google. We have currently integrated it with facebook messenger applications, as it is being used by billions of user across globe. Our chatbot takes symptoms from user and predict based on these symptoms. For prediction we have trained our model on available dataset. Main Modules in our Solutions are NLP module, Prediction Model module.

A. NLP(Natural Language Processing) Module

This module handles natural language processing part of our problem. When user poses some query first we need to detect what user want us to do and second problem is extracting the data given in the query to solve that problem. We have used api.ai chat-bot framework for this task. To extract these information api.ai uses intents, entities and contexts. We defined all these possible intents, entities and contexts with a logical flow. and when user enters some message api.ai process that and returns a Json object with all the information like intent, entity and context in that message.

- **Intents**: Intents are the methods that are triggered corresponding to the work user wants us to do. Suppose he just says Hi then welcome intent will be detected. We have defined all possible intents needed for our conversation flow.
  - **Welcome Intent**: This is detected whn user first opens the application and says Hi, hello type queries.
  - **Fetch Symptoms**: This is detected in a sentence when user want to pose some symptom to our bot. ex: I have pain chest.
  - **predict disease**: This is detected when user wants us to predict the disease based on previously posed symptoms.
  - **User Location**: This is detected when user shares their location using the quick reply feature we have provided and He wants to the nearby doctors.
  - **GoodBye**: this is detected when user poses query to exit from the conversation like bye, goodnight etc.
- **Entities** entities are the features in a sentence that user give in order to fulfill their request. Like in our case user gives symptoms to get the prediction of disease. So we want to detect diseases from user sentences.
  - **Symptoms**: In our bot we only needs symptoms as entities. We have defined 405 symptoms and uploaded all in entity list. We can also give synonyms in that list in order to make better detection.

B. Disease Prediction Model

This module handles the task of prediction of disease given some symptoms. We get the symptoms by NLP module and keep storing them using session id of the conversation. This model takes input the list of symptoms posed by user over time and outputs probabilities of all diseases in our database. We have explored two models Random forest and random forest to predict these probabilities. We found that Multinomial Baysian is performing well. Though we don't have testing data we only have training data but by giving similar symptoms as testing for a disease we experimented many cases and found this conclusion.

1) Random forest: Random decision forest is an ensemble learning method for classification, regression and other tasks. In this method we create a number of decision trees each differing by other by split criteria, goodness measure, or cut-off probability which we randomize, hence the name random forest. We then take a majority vote or other such metrics to get the final classification output. Random forests by virtue of this randomization, correct for decision trees' habit of over fit for our data set fitting to their training set

2) Multinomial Naive Bayes: This is a Probabilistic classifier which uses strong independence assumption between all features. This model is well suited for discrete features like word count features, or as in our case symptoms present. Naive Bayes model assumes strong independence between all its feature given some class. But it does not say anything about individual features probabilistic distribution. In Multinomial Naive Bayes each feature distribution with respect to class is multinomial distribution.
Though these strong assumption of conditionally independence are rarely true, in practice this model performs really well. So people still uses it.
If we have n features $f_1, f_2...f_n$ and a class c then Bayesian assumption can be written as

$$p(f_1, f_2..f_n|c) = \prod_{i=1}^{n} p(f_i|c)$$

Here for our problem $f_1, f_2..f_n$ are symptoms. We have taken f_i as 1 when ith symptom is present else 0. and we want to predict the probability of each class corresponding to these features.

$$p(c|f_1, f_2..f_n) \propto p(c)p(f_1|c)...p(f_n|c)$$

Here p(c) is the prior of the class c. In our case classes are diseases. and $p(f_i|c)$ are probabilities learned by training data. Using this equation we can calculate probabilities of each disease for given symptoms. and can give user the top 3 highest probable diseases.

C. Conversation Flow of our Bot

Figure1 shows the conversation or dialog flow of our chat-bot. First User poses welcome dialogs then he poses symptoms he is having. then our bot asks for any more symptoms and then it predicts diseases and shows to the user. Then again it suggest some more symptoms in order to give more accurate result and finally it asks for location and give nearby hospital details where user can visit for the concerned problem.

D. Components Structure of our Bot

Figure2 shows how API.AI is related to other components and how it processes data. In the left It shows that user interact with the chat-bot or application using some input method and on some platform like messenger bot or slack or something else. then that bot sends user message to api.ai for NLP task. Then api.ai extract NLP features and send these to fulfillment service (web api) as Json object for processing. We need to code this fulfillment logic and host it as api on some server. this server returns result as json object in the format accepted by api.ai. then api.ai returns this result to the respective bot and in turn showed to user.

E. Special Features of Our ChatBot

Following are the special features that we added in out bot:

1) Suggesting related symptoms: Based on the symptoms that the patient provides, the bot suggests the related symptoms to predict more specific diseases. In the dataset, we have the count that in how many medical report the disease is present with their corresponding symptoms. So, this essentially gives us the co-occurrence matrix, which helps us to find the relation between the symptoms.

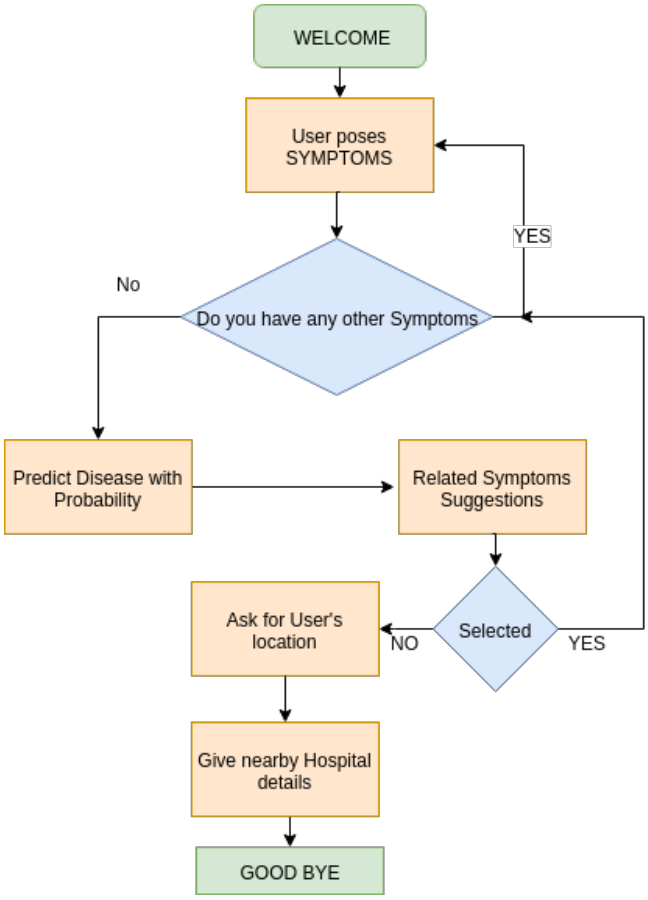

Fig. 1. flowchart

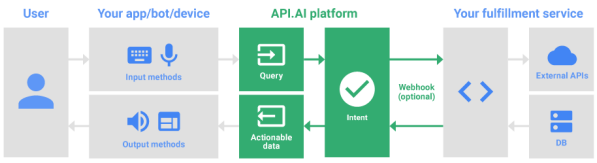

Fig. 2. Shows how API.AI is related to other components and how it processes data

2) Getting User Location and showing nearby hospital: After predicting the possible diseases with their corresponding probability scores, the bot asks the patient for their current location. Using google places api, it generates the location of nearby hospitals and suggests the patient to meet nearby doctors if required.

V. RESULT AND ANALYSIS

We first decided to apply a Random Forest as it was a fairly complex model and it used decision trees to predict the disease. To check how good the model was performing, we gave all the symptoms of gastritis to it and it still predicted it with a comparatively low probability of 0.68 while it should have been much higher. We also found that the model was over-fit for certain diseases as on giving certain symptoms it predicted diseases that looked non-intuitive.
Hence, we decided to apply a much simpler model in Multinomial Naive Bayes. On giving all the symptoms of gastritis it predicted the disease with a probability of 0.9998. Also, on providing symptoms like fever and cough it gave intuitive predictions like bronchitis and influenza.
We experimented for influenza disease by varying number of symptoms. In original data we had 21 symptoms and we gave all 21 symptoms then 18 symptom, and we gave all 15, 10 and 5 symptoms and analyzed which model was giving how much accuracy.

- When given all 21 symptoms Multinomial Bayes returned influenza disease with probability 0.9999 while random forest returned 0.63.
- when we gave first 18 symptoms MN Bayes gave influenza with probability 0.997 while RF(random forest) gave 0.51.
- For first 15 features MN Bayes gave 0.98 and RF returned 0.41
- for first 10 features MN Bayes returned 0.73 and RF returned just 0.2 probability.
- For first 5 features MN Bayes gave 0.11 probability for influenza while RF couldn't detect the disease and gave it as second most probable disease instead of first.

Hence we decided to go with Multinomial Naive Bayes for predicting the disease.

VI. CONCLUSIONS AND FUTURE WORKS

As we have already mentioned, we try to estimate the most probable disease using the given set of symptoms. We use api.ai google chat bot framework to have a smooth conversation with the patient. The Bot also comes up with the follow up questions so as to specifically determine the cause for the symptoms. It also suggests location for the nearby hospitals and doctor's clinic. It's decision helps patient for his knowledge base and it is also beneficial at initial level for general physician, but it cannot replace a doctor's position.

Future work can include more information about patient to take into account, such as the patient history, age, sex, weight, etc. This extra information will help in more precise identification of disease. To further improve the system's reliability, we can use test results of various medical conditions. And for the further development of the chat bot, we can include online appointment setting with the doctor, maintaing prescription for the patient and giving reminders for the medication, and also providing more details about the medicines the patient is using.

REFERENCES

[1] Shankar, M., Pahadia, M., Srivastava, D., Ashwin, T.S. and Reddy, G.R.M., 2015, May. A Novel Method for Disease Recognition and Cure Time Prediction Based on Symptoms. In Advances in Computing and Communication Engineering (ICACCE), 2015 Second International Conference on (pp. 679-682). IEEE.

[2] Mahajan, S. and Shrivastava, G., Effective Diagnosis of Diseases through Symptoms Using Artificial Intelligence and Neural Network. International Journal of Engineering Research and Applications, pp.2248-962.
[3] Khalilia, M., Chakraborty, S. and Popescu, M., 2011. Predicting disease risks from highly imbalanced data using random forest. BMC medical informatics and decision making, 11(1), p.51.
[4] Banu, M.N. and Gomathy, B., 2013. Disease Predicting System Using Data Mining Techniques. International Journal of Technical Research and Applications, 1(5), pp.41-45.