



Just A Rather Very Intelligent Chatbot

# Contents

- Why J.A.R.V.I.C.? 4 people, really? Back-End
- Overview Seq2Seq Model
- Target audience Encoder & Decoder
- Use Case diagram Why Seq2Seq?
- SDLC Emotion Classification
- Project Timeline Model Integration
- Environments MySQL
- Front-End Previous Chat History
- Connection Setup & Authentication Emergency
- Examples Challenges Overcome
- Chat Session References
- Examples Demonstration



# Why J.A.R.V.I.C. ? 4 people, really ?



J.A.R.V.I.C. is a step towards creating an Automated Conversational Agent that imparts Cognitive Behavior Therapy. Studies show that individuals show more freedom while share personal information with a chatbot rather than with a human.

J.A.R.V.I.C. is an effort to show the effectiveness of chatbots to act as therapists to offer low-cost treatment to people who suffer from depression and other mental diseases.



# Overview

We have developed an automated conversational agent that classifies the emotion of the user and forms replies based on the predicted emotional state of the user. The product is an android based application. It connects to a remote server that runs the models for emotion classification and reply-text generation. The app allows the user to signup and login into the system. His details are stored in a database in the server. On login, the past chat history is loaded into the chat session and the colours on the GUI are modelled on the emotion of the user. The user also has the options to reset password clear chat.





# Target audience

Depression is the leading cause of disability worldwide, and it can kill you. Yet scientists know surprisingly little about why it happens and how best to treat it.

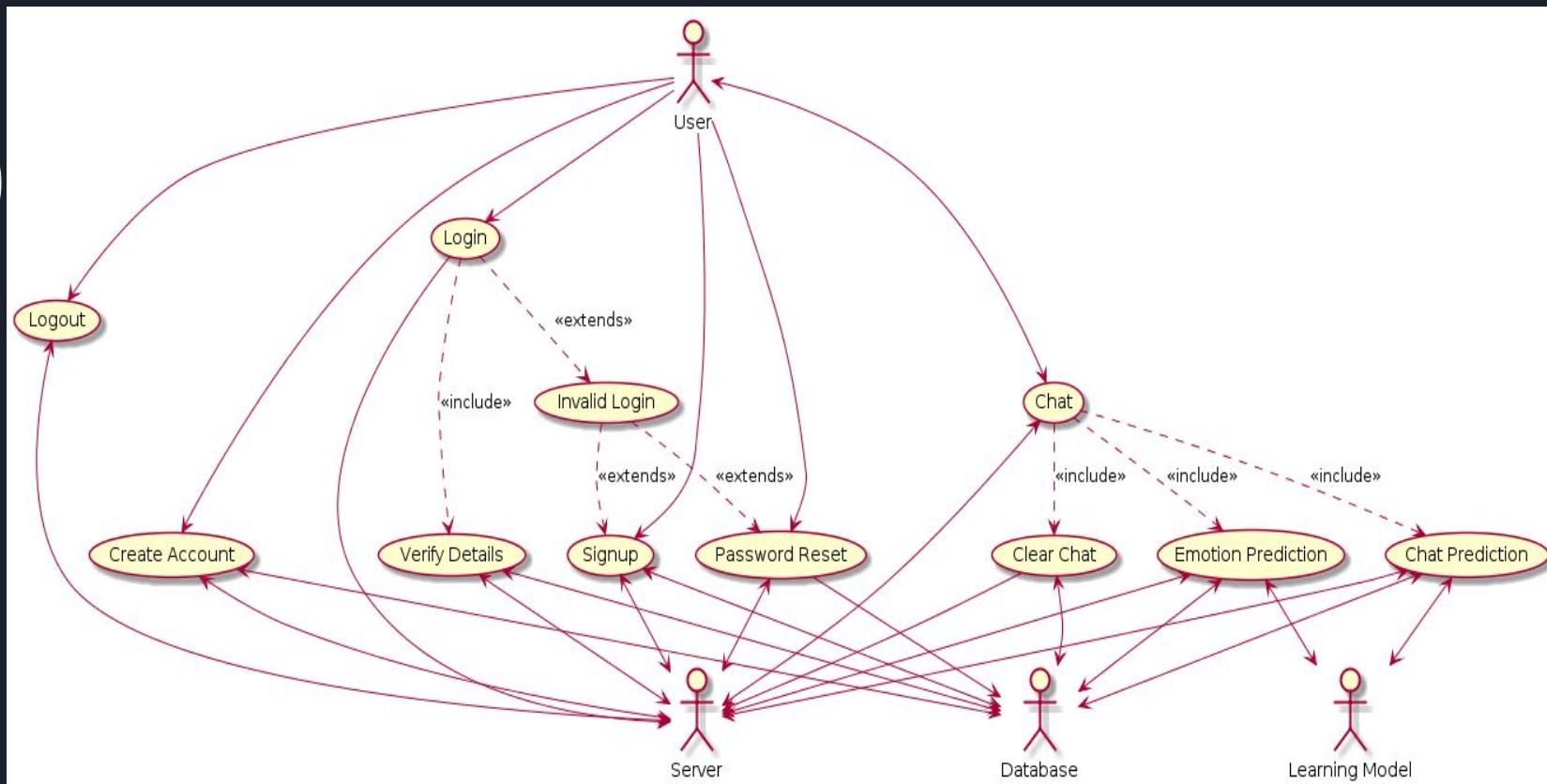
We do know that talking seems to help - especially under the guidance of a licensed mental health professional. But therapy is expensive, inconvenient, and often hard to approach.

J.A.R.V.I.C. is aimed to be used by the poor masses who cannot afford mental healthcare and is presented as an alternative to the conventional human therapist.

J.A.R.V.I.C. can also be used by organizations like the government to find out the general sentiments of the community.



# Use Case diagram:



# Software Development Life Cycle Model Used: Rapid Application Development

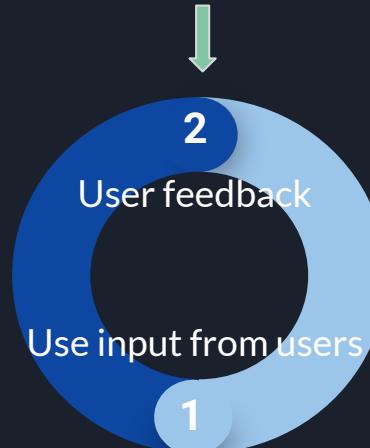


Development:

Build the system.

RAS and FS:

Identify objectives and information requirement



Design:

Work with user to design  
J.A.R.V.I.C.

Introducing J.A.R.V.I.C.



# Project timeline



1st Feb- 15th Feb      16th Feb- 2nd March      2nd March- 15th March      15th March - 25th March      25th March-31st March      1st April-10th April



## RAS and FS

Prepared the System Requirement Specification Document and did extensive feasibility study

## System Design

Prepared an initial design for the system.

## Organising solutions

All the possible solutions to the problem are carefully studied

## Development of prototypes and reviewing

Prototypes were developed and the best suitable solution was selected and the SRS was reviewed

## Development of final product

The final product is developed based on the final prototype.

## Testing

Use cases are tested and corrections are made if necessary.

# Environments Used:

## Languages Used:

- Python 3.5
- Java 1.8
- XML

## Database Management

- MySQL
- mysql-connector for Python3.5



## App Development:

- Android Studio

## Machine Learning:

- Cuda 9.0
- CuDNN 7.4
- Pytorch 1.0
- Scikit-learn
- Numpy
- Pickle

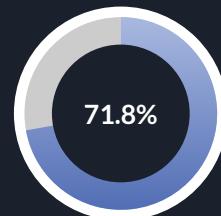


Group subdivision-1: Deepank Agrawal & Jyotisman Das

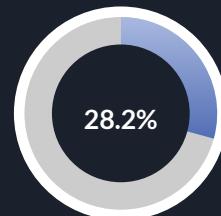


# Front-end development

- Development of Android Application
- Features:
  - Connection Setup:
    - Server Host
    - Server Port
  - Authentication:
    - Login
    - Signup
    - Reset Password
  - Chat Session:
    - Clear Chat
    - History
    - Colour of GUI based on emotion



Java



XML



# Connection Setup & Authentication



- Connection Setup page.
  - Connection is based on Local Network instead of the web.
  - Tackles multiple WiFi hosts problems.
- Login/signup after authentication
  - After successful login, the chat history is restored and made available to the user.
  - Failed attempt to login is detected and the user is provided with the option to reset the password.
  - After verification the password is reset and the user is redirected to the login page.
  - For a new user, there is an option to signup after which the user can login using the credentials he/she provided.

# Examples



20:22 12% Connection Setup

10.145.54.45

2003

**DONE**

This screenshot shows the Connection Setup screen. It features a green header bar with the text "Connection Setup". Below the header are two input fields: one for the IP address containing "10.145.54.45" and another for the port number containing "2003", both of which have red validation lines underneath. At the bottom is a large red button labeled "DONE".

20:23 11% Signup

Rajat

rajatkj11@gmail.com

.....

9582027877

**REGISTER**

Already registered? Login

This screenshot shows the Signup screen. It has a green header bar with the text "Signup". The form includes fields for a user name ("Rajat"), email ("rajatkj11@gmail.com"), password ("....."), and a phone number ("9582027877"). Each field has a red validation line. A large red "REGISTER" button is at the bottom. Below the button is a link for existing users: "Already registered? Login".

20:21 12% Login

rajatkj11@gmail.com

.....

Forgot password?

**LOGIN**

This screenshot shows the Login screen. It has a green header bar with the text "Login". It features fields for an email ("rajatkj11@gmail.com") and a password ("....."). A "Forgot password?" link is located between the fields. A large red "LOGIN" button is at the bottom. Below the button is a link for new users: "Not Registered? Signup".

20:22 11% Password Reset

rajatkj11@gmail.com

9582027877

.....

**RESET**

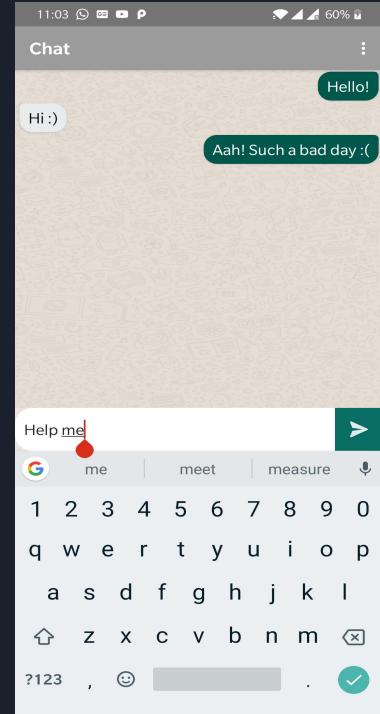
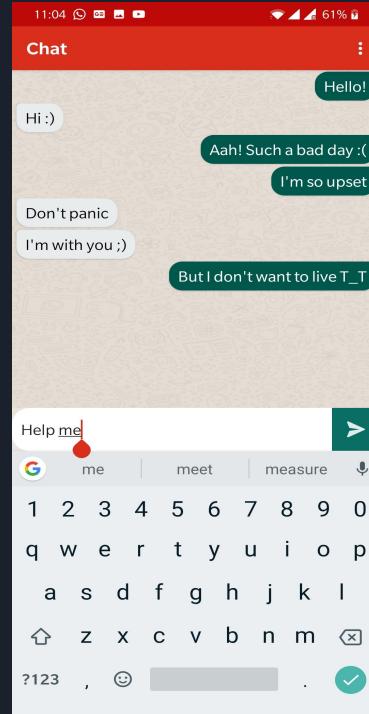
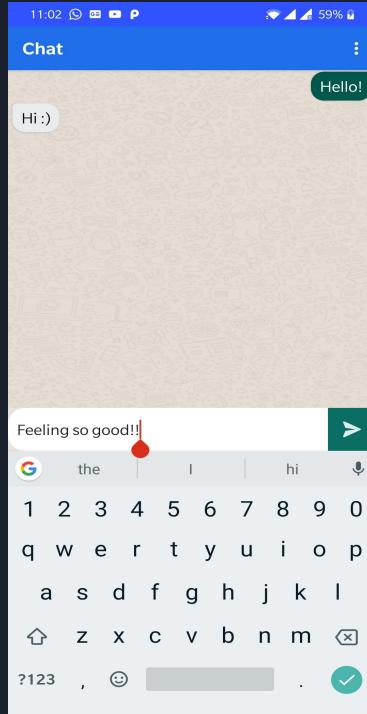
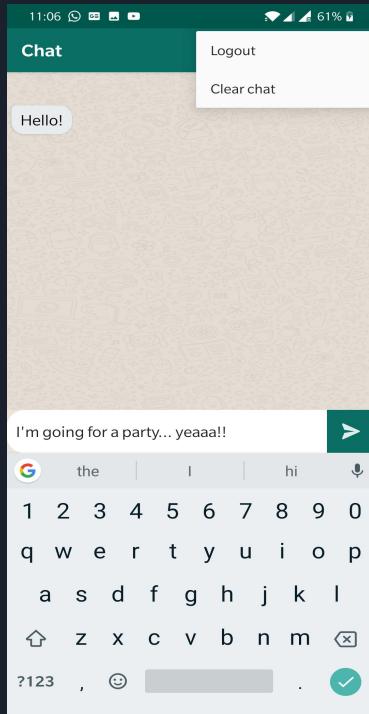
This screenshot shows the Password Reset screen. It has a green header bar with the text "Password Reset". It features fields for an email ("rajatkj11@gmail.com"), a phone number ("9582027877"), and a password ("....."). A large red "RESET" button is at the bottom. Below the button is a link for new users: "Not Registered? Signup".



# Chat Session

- This is the most important part of our ChatBot.
- The user chats with the J.A.R.V.I.C. server here.
- Many features are provided from the front-end side, like -
  - Clear Chat option - The user can anytime clear chat done with the server by selecting the option from overflow menu.
  - Logout option - The user can log out from the chat, which get redirected to connection setup window for re-login.
  - Dynamic GUI color - As the chat progresses, JARVIC tries to know the current emotional state of the user and accordingly change GUI color.
  - Currently, JARVIC is able to identify 3 types of emotions namely - Positive, negative and critical.

# Examples



CALM

POSITIVE

CRITICAL

NEGATIVE

Group subdivision-2: Rajat Kumar Jenamani & Siddhant Agarwal



# Back-end Development

- Chatbot text generative model for reply generation.
  - Trained two models one for each emotion- happy and sad.
  - Datasets are custom created.
  - A model is selected based on the classified emotion.
- Bayesian Classifier for emotion classification.
  - Classifies the emotions assuming the data distribution as multinomial.
  - This emotion is saved in the database.
  - The predicted emotion is used to format the interface as well as to select model for generating replies.
- Database management for storing information about users.



Python



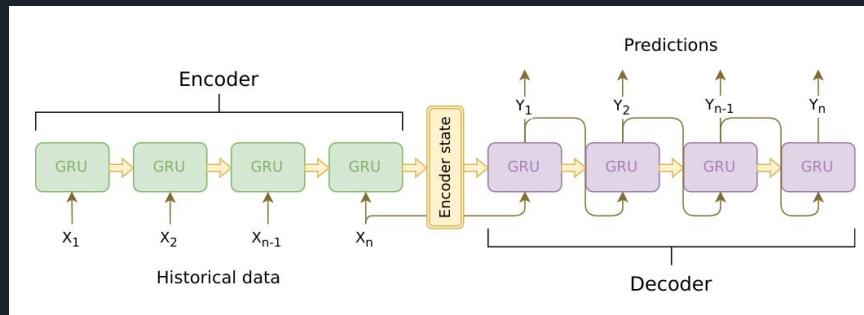
MySQL



# Chatbot text generative models: Seq2Seq model

The brain of our chatbot is a sequence-to-sequence (seq2seq) model. The goal of a seq2seq model is to take a variable-length sequence as an input, and return a variable-length sequence as an output using a fixed-sized model.

Two separate Recurrent Neural Networks are used for this purpose. One RNN acts as an **encoder**, which encodes a variable length input sequence to a fixed-length context vector. In theory, this context vector (the final hidden layer of the RNN) will contain semantic information about the query sentence that is input to the bot. The second RNN is a **decoder**, which takes an input word and the context vector, and returns a guess for the next word in the sequence and a hidden state to use in the next iteration.





# Encoder

The encoder RNN iterates through the input sentence one token (e.g. word) at a time, at each time step outputting an “output” vector and a “hidden state” vector. The hidden state vector is then passed to the next time step, while the output vector is recorded. The encoder transforms the context it saw at each point in the sequence into a set of points in a high-dimensional space, which the decoder will use to generate a meaningful output for the given task.

# Decoder

The decoder RNN generates the response sentence in a token-by-token fashion. It uses the encoder’s context vectors, and internal hidden states to generate the next word in the sequence. It continues generating words until it outputs an *EOS\_token*, representing the end of the sentence



# Why Seq2Seq?

All generative models like GANs, seq2seq, conditional variational autoencoders etc use two different modules generally called encoders and decoders. As the name suggests, Encoders encode a given input into feature vectors. The decoder takes in the encoder outputs and states and predicts the output iteratively.

Seq2Seq models use the similar concepts of encoder and decoder, which makes them ideal for text generation. They are popularly used in language translations and several applications wherever text needs to be generated. Our chatbot is no exception. Hence we deploy this model for reply generation.

A common problem with decoders are that they solely rely on the context vectors. For this attention mechanism is used so that decoder uses some part of the input sequence as well for prediction.

Multi layered gated recurrent neural network is used in the encoders and decoders.

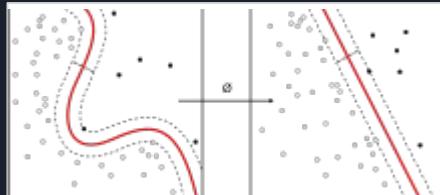




# Emotion Classification

## Bayes' Classifier

Bayes' Classifier is a probabilistic classifier based on Bayes' theorem with strong independence assumptions between features. This is a very simple machine learning technique with moderate accuracy. This model is used since the server could not support two models running simultaneously. The model classifies emotions into two classes namely positive (happiness, confidence, etc) and negative (sad, depression etc.). Based on the predicted emotion, the application changes the colour of interface.





# Generating replies according to emotions: Combining the two models:

When the user string is received at the server, the emotion is classified using the Bayes' classifier. Based on the emotion predicted by the classifier, a seq2seq model is selected. We have two models, one is in negative state and the other is in positive state. Based on the emotion of the user, an appropriate reply is generated.

# Using MySQL for database management :

The following table was used to store data of the users:

Field	Type	Null	Key	Default	Extra
name	varchar(100)	NO		NULL	
username	varchar(100)	NO	PRI	NULL	
password	varchar(100)	NO		NULL	
contact	varchar(100)	NO		NULL	
history	varchar(65000)	YES		NULL	

name	username	password	contact	history
Siddhant	agarwalsiddhant10@gmail.com	siddhant	9876543210	u\$Hi\$\$Hello
Deepank	deepank.361998@gmail.com	deepank	9785388468	u\$Hi\$\$Hello
Jyotisman Das	jeetjyotismanjeet1999@gmail.com	jyotisman	9876543210	u\$Hi\$\$Hello
Rajat Kumar Jenamani	rkjenamani@gmail.com	rkjenamani	9876543210	u\$Hi\$\$Hello

# More Features:

## PREVIOUS CHAT HISTORY →

J.A.R.V.I.C. stores the previous chat history of the user and loads it into the interface upon login.





# More Features:

## EMERGENCY DETECTION:

J.A.R.V.I.C. can detect critical situations when it realizes that the person it is chatting with is highly depressed and sends the username and contact of the person to the concerned authorities (currently on the terminal).





# Challenges Overcome



01

Prediction of emotion of user by analysing the messages received. This is really tricky as first of all there is no good dataset available that fits our use. Currently the Bayesian classifier is trained on IMDB Movie review dataset. Secondly and most importantly, only text is often quite misleading context related meanings and homonyms.

02

Prediction of reply of message sent by user depending upon message and emotion of user. Currently, our model has two states namely, user is happy or sad. The emotion classifier predicts the state and the models take the necessary actions. In future, we plan to create several states and train a decision process to take the decision for the action the model should take.

03

Predicting the replies is itself very difficult. The major hurdle we faced was we did not have enough data. We created our own data but we only able to create a total of 10 kbs in which we had to train two models. Due to this our model have very limited vocabulary. We plan to generate much more data so that the models can generate better.



# References





# DEMONSTRATION



# Thank you!

Launching our startup soon!

