Chapter 1

Introduction

1.1 Background

We are in the age, where the intelligent machines have arrived. There are number of new kinds of machines and software available which are collectively know as Bots. Bots are the computer program which simulates human conversation through voice commands or text chats or both.

One of the ultimate goals in the field of AI is to build computer systems that can have human-like conversations with users. Recently there are advances in AI technologies, which has got us closer to achieving this goal.

Chatbots continue to grow in popularity with 80% of businesses expecting to be using one by 2020. Though it may feel like the term 'chatbot' has only recently entered the public lexicon, they actually have a longer history than you might expect.

1950 - THE TURING TEST

Alan Turing Theorized that a truly intelligent machine would be indistinguishable from a human during text only conversation. These ideas essentially laid the foundations for the chatbot revolution.

1. ELIZA*:* It is considered to be the first chatbot in the history of Computer Science which was developed by **Joseph Weizenbaum** at Massachusetts Institute of Technology (MIT). It was in 1994 that the term 'Chatterbot" was coined. ELIZA operates by recognizing key words or phrases from the input to reproduce a response using those keywords from pre-programmed responses.

2. ALICE: It was developed in 1995 by Richard Wallace. Unlike Eliza, the ALICE chatbot was able to use natural language processing, which allowed for more sophisticated conversation. It was revolutionary, though, for being open-source. Developers could use AIML (artificial intelligence markup language) to create their own Chatbots powered by ALICE.

Today there are many chatbot like Siri, Alexa, Google Now, Google Assistant, Cortana etc.

1.2 Objective

1. To understand the human behavior which leads to depression.

2. To identify depression symptoms via questionnaires.

3. To design a chatbot which helps depressed people in expressing and exchanging their emotional sentiments without fear.

4. To motivate and boost them which helps to overcome depression.

1.3 Need of RoboRelief

- Depression is the leading cause of disability, and the cost of mental illness to society has doubled in the last 10 years in the region of every world.
- A study found that people are more likely to open up to a talking computer than a human.
- The aim of this chatbot is to motivate a person going through a low phase of his life and to avoid ill-effects of depression.
- The rate of depression is growing at an alarming rate. There are simply not enough mental health professionals to meet this demand.

1.4 Introduction to Deep Learning

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called **artificial neural networks**.

Andrew Ng commented on the important point that it is all about scale. That as we construct larger neural networks and train them with more and more data, their performance continues to increase. This is generally different to other machine learning techniques that reach a plateau in performance.

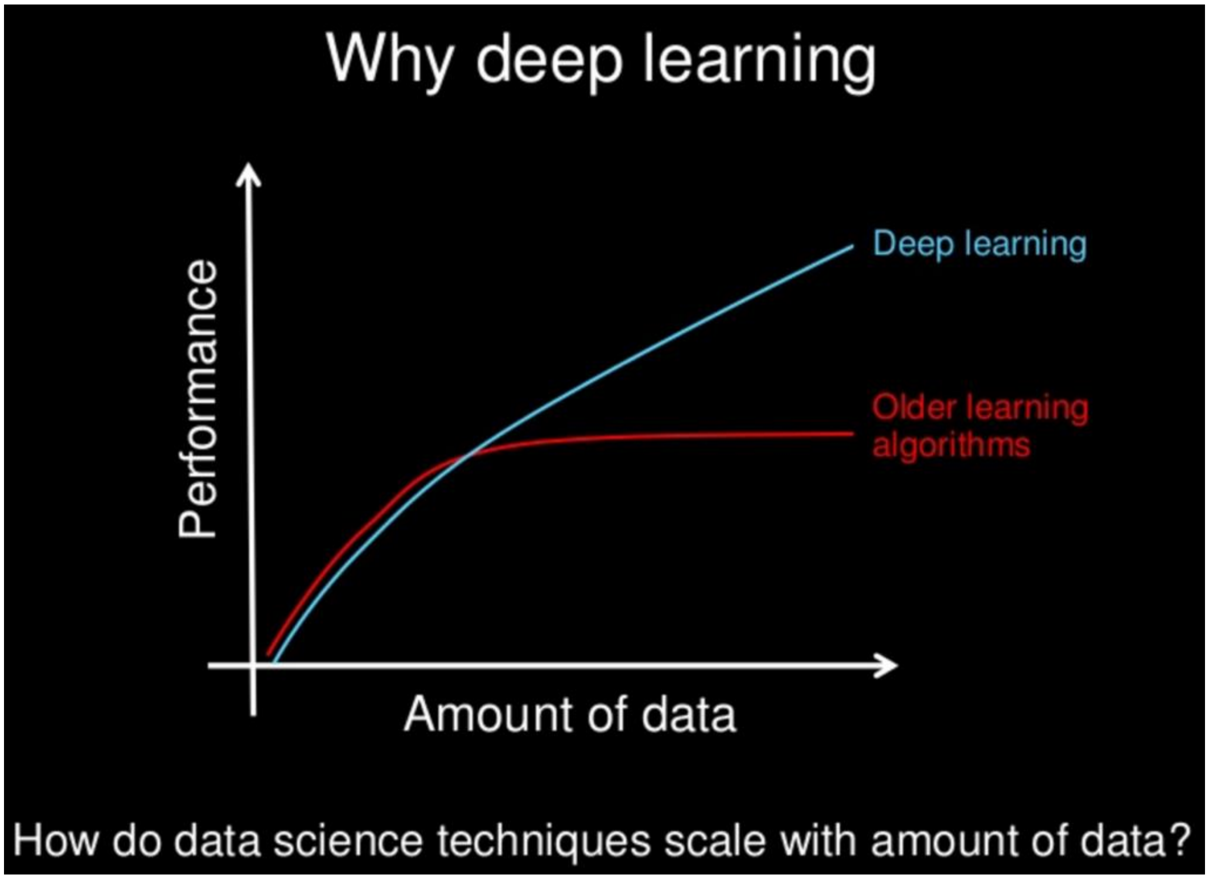


Figure 1.1 :- Graph showing comparison between Older learning algorithms and Deep Learning

In addition to scalability, another often cited benefit of deep learning models is their ability to perform automatic feature extraction from raw data, also called feature learning.

Deep Learning Algorithm:

1. MLP(Multilayered Perceptron) -

   one of the basic Algorithms of Deep Learning Multilayer Perceptron or MLP. In the perceptron we just multiply with **weights** and add **Bias,** but we do this in one layer only.

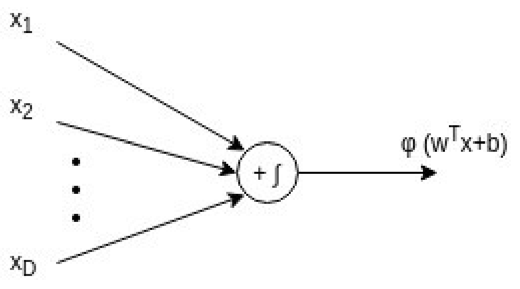


Figure 1.2 :- Single Layer perceptron

In the Multilayer perceptron, there can be more than linear layer or neurons. If we take the simple example the three-layer first will be the *input layer* and last will be *output layer* and middle layer will be *hidden layer.* We feed the input data to the input layer and take the output from the output layer. We can increase the number of the hidden layer as much as we want, to make the model more complex.

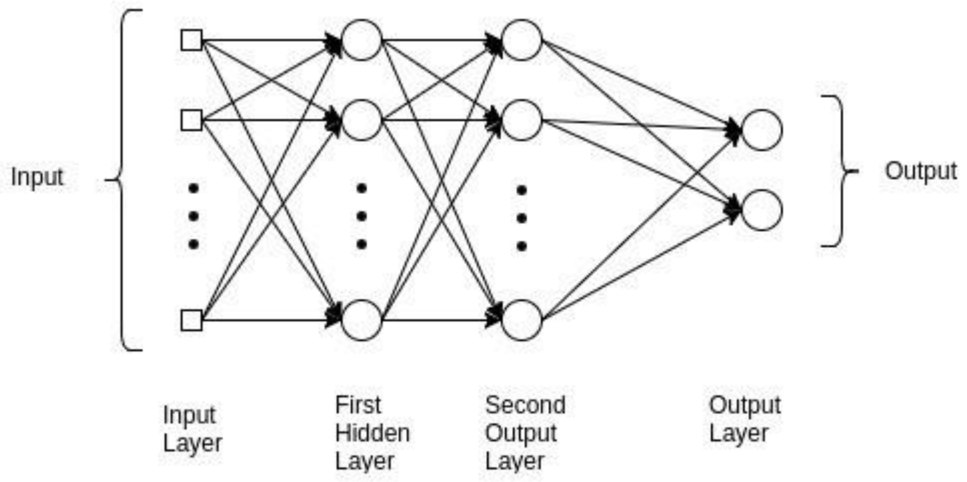


Figure 1.3: Multi-Layer Perceptron

2. LSTM

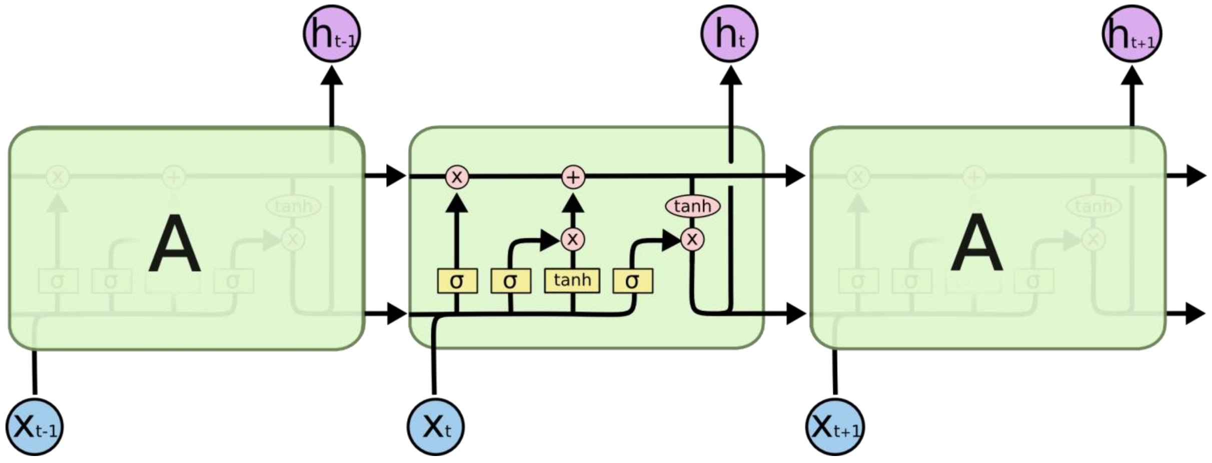


Figure 1.4: LSTM Architecture

Long Short Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997)

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

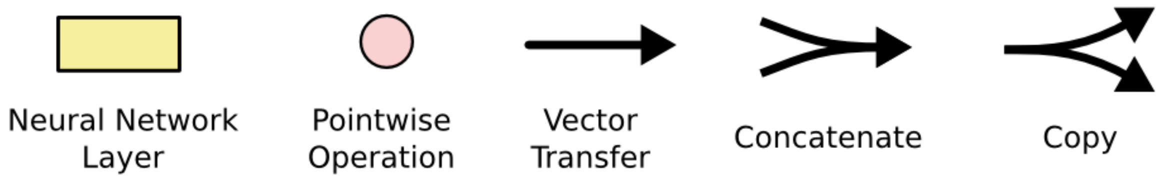


Figure 1.5: Notation used in LSTM Architecture

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram.

The cell state is kind of like a conveyor belt. It runs straight down the entire chain, with only some minor linear interactions. It's very easy for information to just flow along it unchanged.

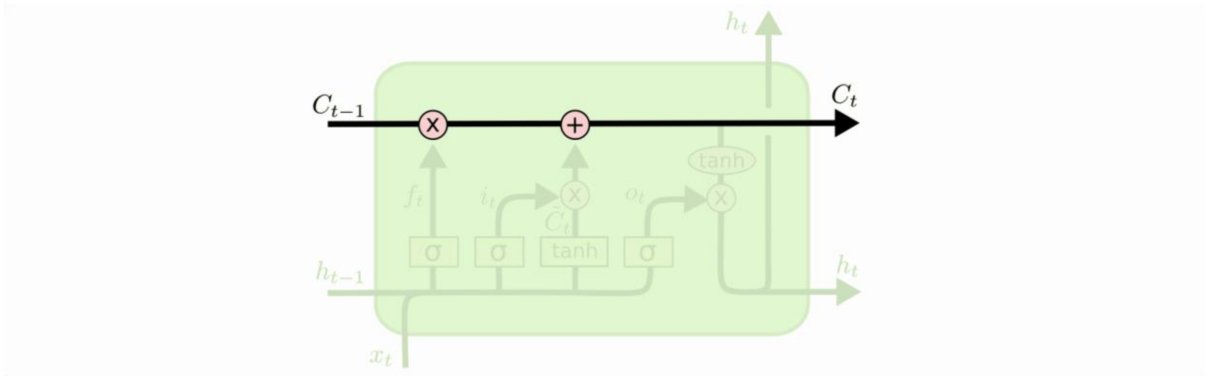


Figure 1.6: Skip Connection in LSTM

The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates.

Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation.

The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means “let nothing through,” while a value of one means “let everything through!”

An LSTM has three of these gates, to protect and control the cell state.

1.5 Introduction to Project entitled RoboRelief Application

This application is Android OS based application. It also uses deep learning algorithm like LSTM (Long Short Term Memory) which is a type of RNN (Recurrent Neural Network). It uses a special type of architecture of LSTM, which is sequence-to-sequence model or encoder-decoder model.

In this, we have two LSTM: -

1. Encoder - One LSTM is known as encoder and it is used to encode the user response.
2. Decoder - Another LSTM is known as decoder which is used to give response to user

Both of these used together to build chatbot. This type of model is also used in Language Translation Task.

It is used to boost the confidence of people to overcome their depression so that suicide rate due to depression could be decreased. Any person can chat with this chatbot to relax his mind.

1.6 Summary

This chapter, Introduction has provided a brief description of our project including several captions including background, motivation and objectives Introduction to Project entitled RoboRelief Application, In the next chapter, we will discuss about prevailing similar applications and point out the specialty, performance and benefits of our proposed application while comparing to them.

Chapter 2

Problem Identification

2.1 History

Mental healthcare is in Crisis. Depression is the leading cause of disability, and the cost of mental illness to society has doubled in the last 10 years in the region of every world. Yet, the global median spending on mental health is just 2.8% of government health spending.

Depression is a common illness characterized by persistent sadness and a loss of interest in activities that one normally enjoys, accompanied by an inability to carry out daily activities, for at least two weeks.

In addition, there may be a loss of energy; a change in appetite; sleeping more or less; anxiety; reduced concentration; indecisiveness; restlessness; feelings of worthlessness, guilt, or hopelessness; and thoughts of self-harm or even committing suicide.

Depending on the number and severity of symptoms, a depressive episode can be categorized as mild, moderate, or severe. There are effective treatments for depression.

The WHO estimates that one in four people in the world will be affected by mental or neurological disorders at some point in their lives. Around 450 million people currently suffer from such conditions, placing mental disorders among the leading causes of ill health and disability worldwide. Globally, the total number of people with depression was estimated to exceed 300 million in 2015, equivalent to 4.3% of the world's population. Depression is ranked as the single largest contributor to global disability (7.5% of all years lived with disability in 2015). At its worst, depression can lead to suicide, over 800 000 people die due to suicide every year. Suicide is the second leading cause of death in 15-29-year-olds.

In India, the National Mental Health Survey 2015-16 reveals that nearly 15% Indian adults need active intervention for one or more mental health issues and one in 20 Indians suffers from depression. It is estimated that in 2012, India had over 258,000 suicides, with the age-group of 15-49 years being most affected.

The Indian government's commitment is reflected in the National Mental Health Program (NMHP), which encompasses life skills training and counselling in educational institutions, workplace stress management and suicide prevention services, among others.

2.2 Problem Statement

In a country where people are averse to talking about mental and behavioral illness such as depression, a chatbot by our team is proving to be revolutionary by aiding those suffering from it. RoboRelief by AvkGroup is designed to be an emotionally intelligent chatbot that acts as a virtual coach, helping in managing and improving mental health, and is claimed to be the first of its kind by the company.

According to the WHO, one in four people globally suffer from mental distress at some point of their lives, and 50% of those depressed are not even identified. In India, the prevalence of mental disorder is nearly 10%, with suicide and self-harm being the leading cause of death for adolescents who are being exposed to increasing levels of stress and anxiety. RoboRelief, an AI enabled coach touch bases of this serious concern at hand by providing AI enabled coach and chat platform.

Based on the above data we decided to build RoboRelief, which is a chatbot. This chatbot will trained on some questions, which can be phrased as Depression dataset. To get to any psychiatrist regarding depression issue is very expensive but this will not be expensive and also doctors regarding depression issues are not easily available but this chatbot will be easily accessible to anyone who have android phone and in today's world almost everyone has android phone.

Therefore, we decided to build this type of chatbot.

2.3 Scope of the Project

We want our project RoboRelief to work as an emotional resilient tool for mental health. People can talk to "Reliefo" the bot in RoboRelief when they do not want to talk to another human being. We want to help people who are in stress and are in tension. Our aim is that Reliefo helps people to reflect and learn more about themselves, their own well-being and the things they can do to strengthen their well-being. Initially, we want this chat based function to allow the users to be more reflective. It is quite safe and secure because this is anonymous as it is just the friend of the person, who has RoboRelief in his or her phone. We want to make the world more mentally resilient. Technology has always been seen as a part of the problem, an introducer, an enhancer of the problems and now we want to make it the solution.

Sometimes life can be overwhelming, as we all know, we want RoboRelief to help you to get your mojo back. RoboRelief will be a bot, which can help in taking back the life from being stressed to being normal. We want RoboRelief to make the world healthier.

As we, all know how we think affects the way we feel and by changing the way we think, we can feel or act better in challenging circumstances. Cognitive Behavioral Therapy or CBT is a way for people to understand problematic thinking patterns and challenge the way we think about things. We want our RoboRelief to adapt this effective technique. We know this technique is not an easy task to work on in our under graduation study but we affirm that we will surely work for its betterment by attaining the technique of CBT in the future which is the future scope of this project.
Most of the times it has been seen that people who are depressed or stressed do not go to check out a therapist or psychiatrist. The chief reason apart from money is that people do not want to expose themselves to another human being as being emotionally weak. Reliefo would use AI to detect what the person is saying and then give responses accordingly. The responses have been checked from over various platforms.

People who have been bullied on social media platforms the reason for their stress would probably not be the place where one would go for stress removal. One would not go to Facebook messenger to relief the stress. Moreover, nowadays adolescents do not want to talk about their problems to their parents. Generally, they talk to their best friends about their problems but they do not feel safe to disclose that. Therefore, we decided to keep RoboRelief as anonymous and secure. Moreover, it can help those also who do not have anyone to talk, people who are lonely.

Challenging the stigma around mental illness and getting everyone to open up and talk is a difficult challenge. For those who are reluctant to talk to a real person then perhaps talking to a machine might provide a step in the right direction.

2.4 Encoder-Decoder Model

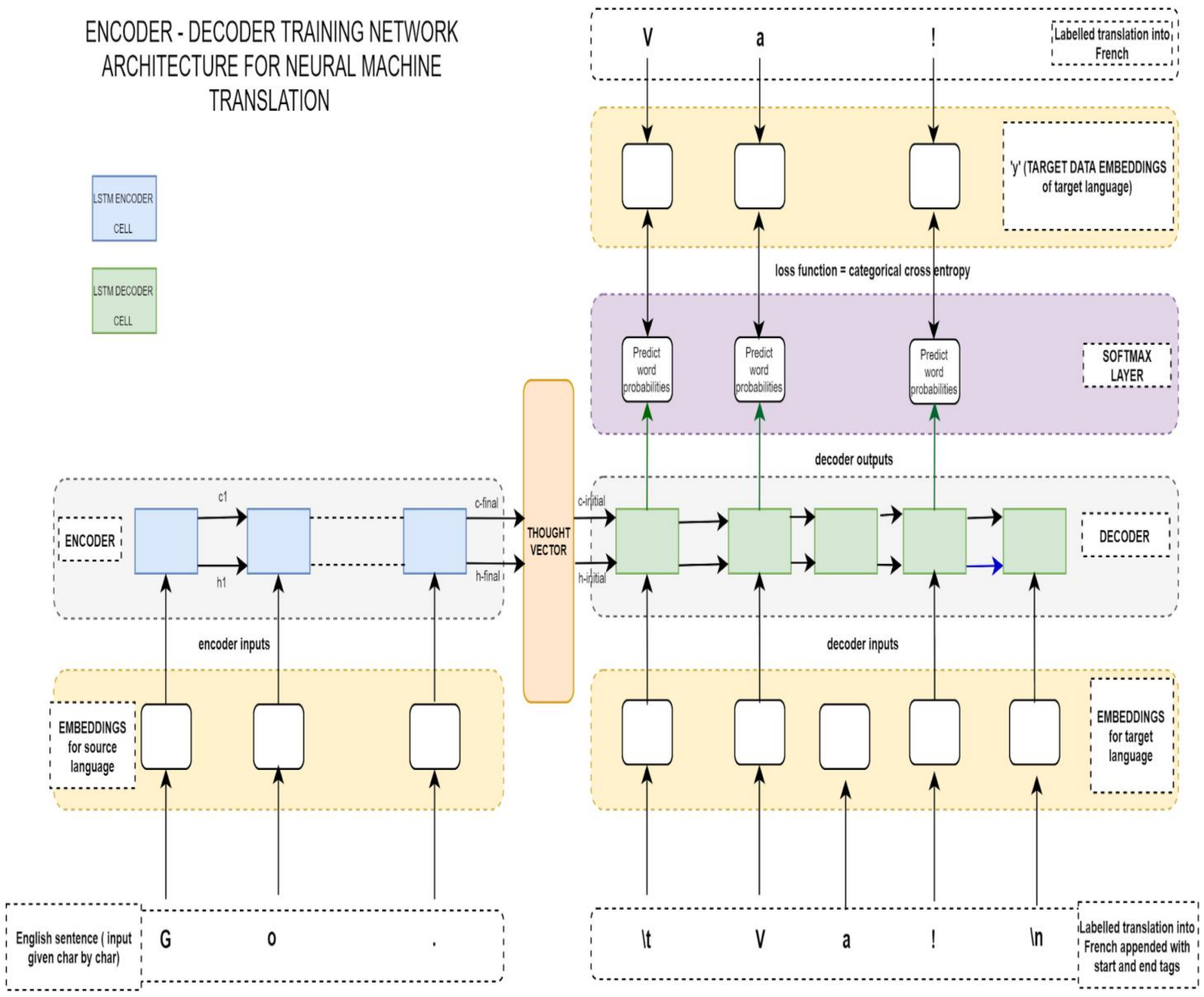

ENCODER - DECODER TRAINING NETWORK ARCHITECTURE FOR NEURAL MACHINE TRANSLATION
LSTM ENCODER CELL
LSTM DECODER CELL
V a !
Labelled translation into French
'y' (TARGET DATA EMBEDDINGS of target language)
loss function = categorical cross entropy
Predict word probabilities
SOFTMAX LAYER
decoder outputs
c1
h1
c-final
h-final
THOUGHT VECTOR
c-initial
h-initial
ENCODER
DECODER
encoder inputs
decoder inputs
EMBEDDINGS for source language
EMBEDDINGS for target language
English sentence (input given char by char)
G o .
\t V a ! \n
Labelled translation into French appended with start and end tags

Figure 2.1 Diagram showing Encoder-Decoder Model

Sequence prediction often involves forecasting the next value in a real valued sequence or outputting a class label for an input sequence.

There is a more challenging type of sequence prediction problem that takes a sequence as input and requires a sequence prediction as output. These are called sequence-to-sequence prediction problems, or seq2seq for short.

One modeling concern that makes these problems challenging is that the length of the input and output sequences may vary. Given that there are multiple input time steps and multiple output time steps, this form of problem is referred to as many-to-many type sequence prediction problem.

One approach to seq2seq prediction problems that has proven very effective is called the Encoder-Decoder LSTM.

This architecture is comprised of two models: one for reading the input sequence and encoding it into a fixed-length vector, and a second for decoding the fixed-length vector and outputting the predicted sequence. The use of the models in concert gives the architecture its name of Encoder-Decoder LSTM designed specifically for seq2seq problems.

The Encoder-Decoder LSTM was developed for natural language processing problems where it demonstrated state-of-the-art performance, specifically in the area of text translation called statistical machine translation.

In one of the first applications of the architecture to English-to-French translation, the internal representation of the encoded English phrases was visualized. The plots revealed a qualitatively meaningful learned structure of the phrases harnessed for the translation task.

The input array to be fed into the LSTM should be three dimensional. Lets look at this in the context of feeding several rows of sentences to be fed into the LSTM where each sentence is a collection of words and the size of the sentence can be either fixed / variable.

- The first dimension is the number of sentences in our corpus.
- Second dimension specifies the number of time steps. The timesteps in this context can be visualized as the number of words in that particular sentence *(*assuming each word is converted to a vector). Or else, it can be the number of characters in the sentence (assuming each character is converted to a vector). Also note that this can be fixed length or varied length because each sentence can have varied

number of words/characters. We can make this a fixed length by padding all the sequences with zeros wherever needed.

- Third dimension specifies the number of features. It can be the number of words in the corpus, or it can be the number of characters in case we are doing a character level translation, for example.

But when we define the input shape of LSTM, we only define it as 2D and not 3D — we do not specify the number of samples or the batch size. We only specify the number of time-steps and the number of features. In keras we define this as —

- Input(shape=(nb_timesteps, nb_features))**.** As noted, the nb_timesteps can correspond to either the fixed or the variable sequence length of the input array. If we want it to be a variable length, then we specify this as None.In that case, it will be *Input(shape=(*None*, nb_features))***.**

When the number of timesteps is None**,** then LSTM will dynamically unroll the timesteps till it reaches the end of the sequence. This is typical of Neural machine translation architectures involving encoder-decoder networks.

We can initialize LSTM with an initial state while feeding the input. Again this is typical of encoder-decoder architectures, where the decoder LSTM needs to be initialized with the final encoder cell state and hidden state. In such cases, we pass the intial_state argument along with the input.

- decoder_LSTM (decoder_input, initial_state=encoder_states)

2.5 Applications of Encoder-Decoder LSTMs

The list below highlights some interesting applications of the Encoder-Decoder LSTM architecture.

- Machine Translation, e.g. English to French translation of phrases.
- Image Captioning, e.g. generating a text description for images.
- Conversational Modeling, e.g. generating answers to textual questions.
- Movement Classification, e.g. generating a sequence of commands from a sequence of gestures.

Chapter 3
Detail Design

3.1 Activity diagram

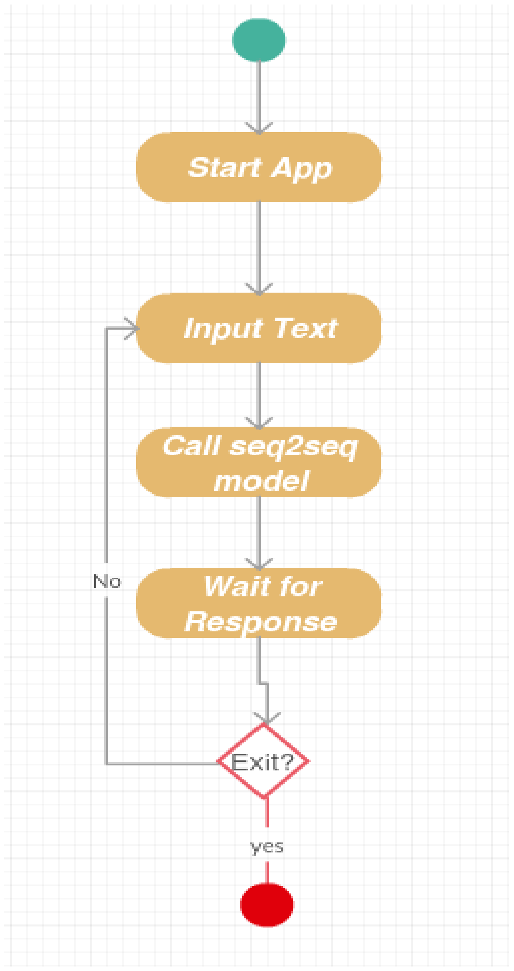


Fig 3.1 Activity diagram of RoboRelief

3.2 DFD

3.2.1 Level-0 DFD

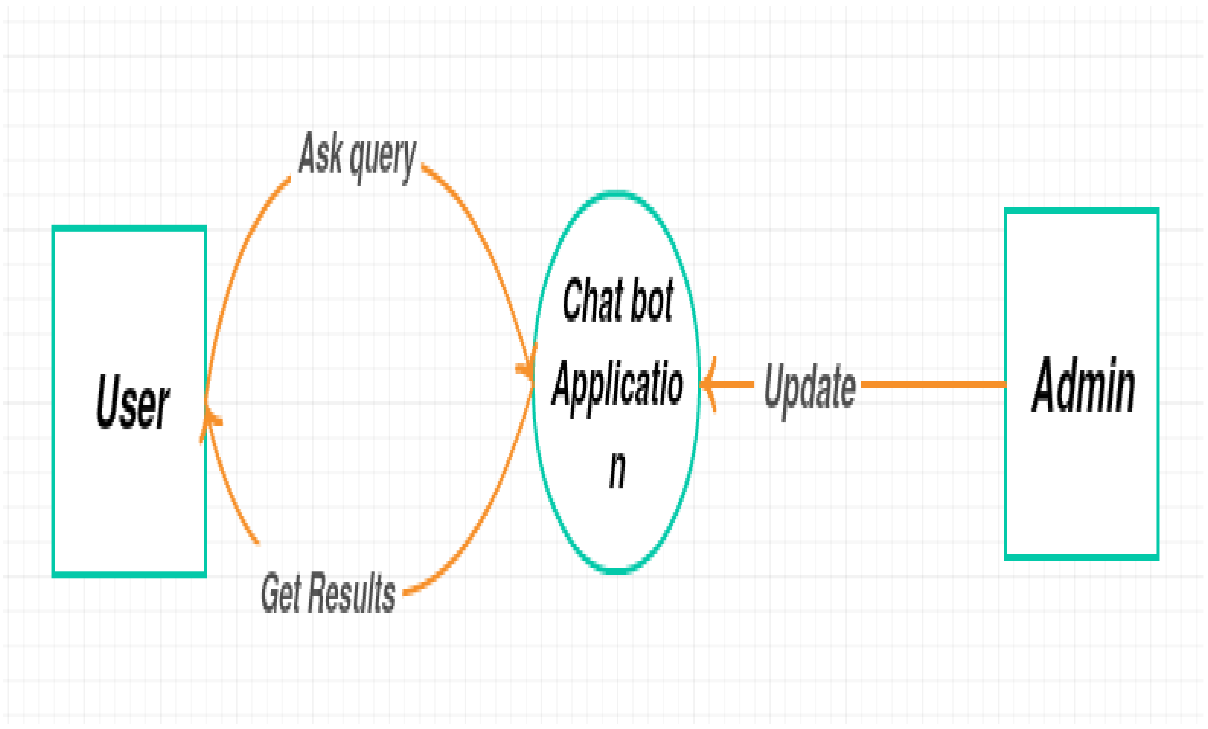


Fig 3.2.1 Level-0 DFD of RoboRelief

3.2.2Level-1 DFD

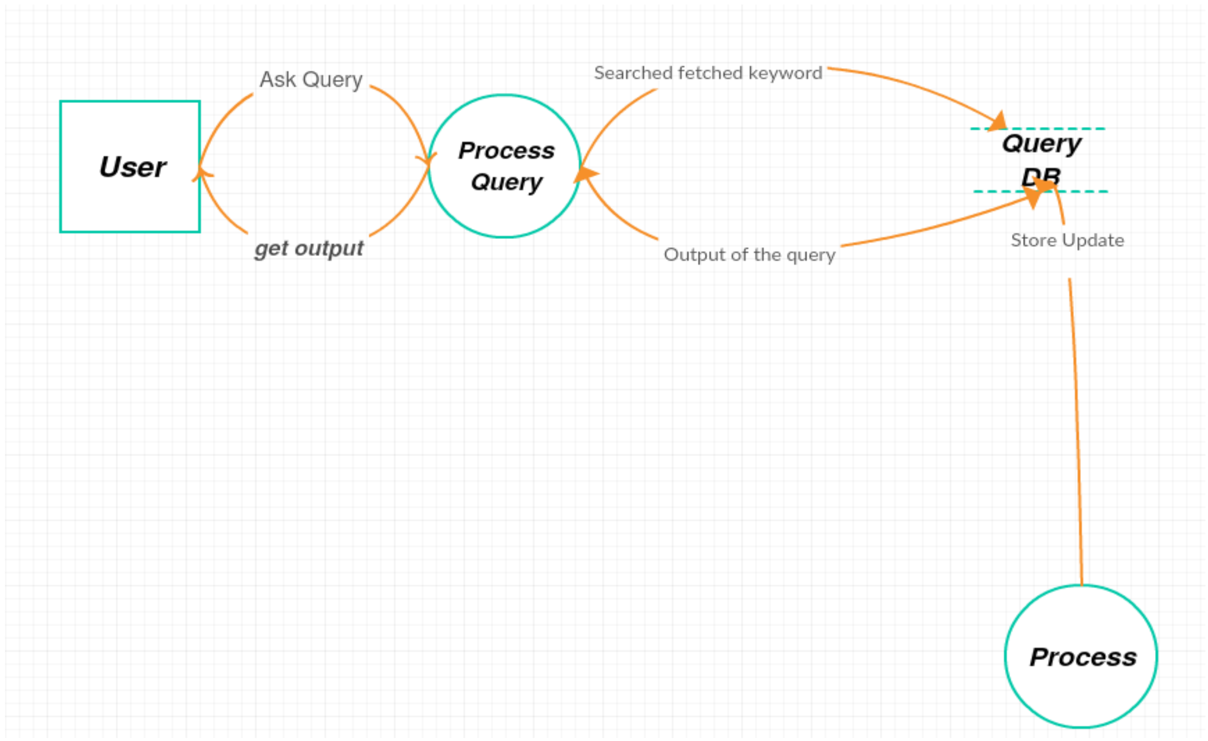


Fig 3.2.2 Level-1 DFD of RoboRelief

3.3 Sequence diagram

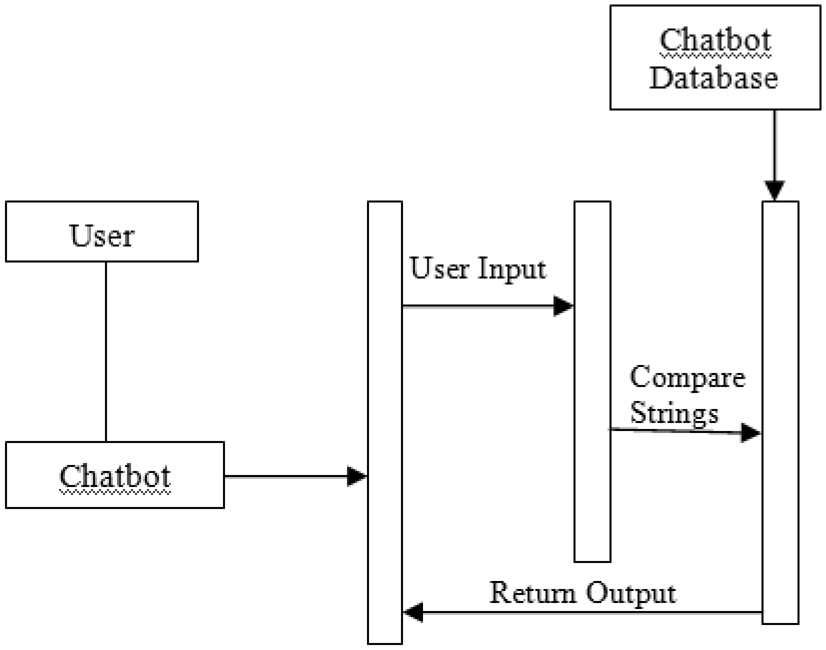


Fig 3.3 Sequence Diagram of RoboRelief

3.4 RoboRelief Chatbot Architecture

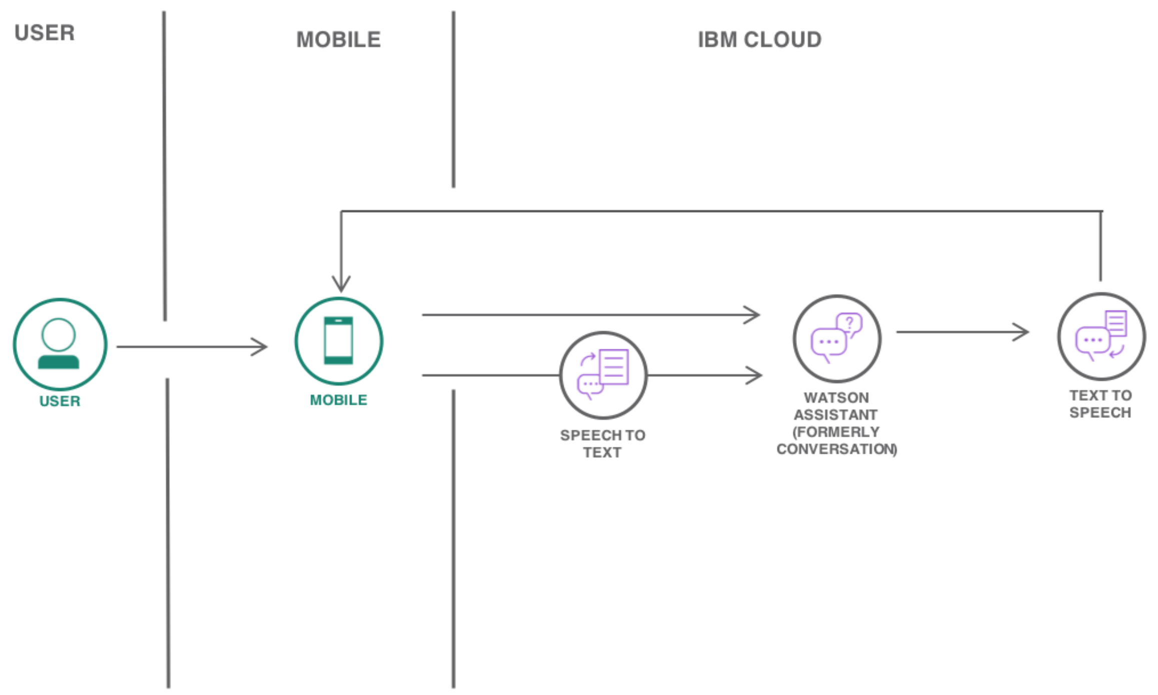


Figure 3.4 RoboRelief chatbot architecture

Chapter 4 Testing

4.1 Android Testing

The Android framework includes an integrated testing framework that helps you test all aspects of your application and the SDK tools include tools for setting up and running test applications. Whether you are working in Eclipse with ADT or working from the command line, the SDK tools help you set up and run your tests within an emulator or the device you are targeting.

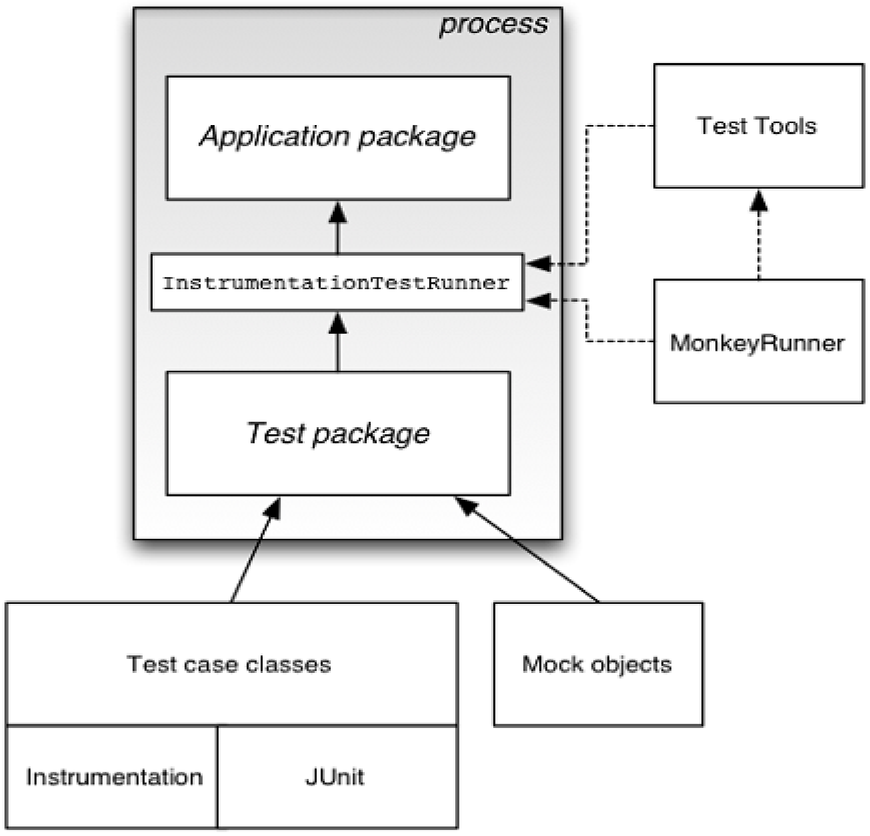


Figure 4.1 Android Testing

4.2 Testing Tools in Android

Many tools can be used for testing android applications. Some are official like Junit,Monkey and some are third party tools that can be used to test android applications. In this testing part we are going to explain these two tools to test android applications.

- JUnit

- Monkey

4.2.1 JUnit

You can use the JUnit TestCase class to do unit testing on a class that doesn't call Android APIs. TestCase is also the base class for AndroidTestCase, which you can use to test Android-dependent objects. Besides providing the JUnit framework, AndroidTestCase offers Android-specific setup, teardown, and helper methods.

4.2.2 Monkey

The UI/Application Exerciser Monkey, usually called "monkey", is a command-line tool that sends pseudo-random streams of keystrokes, touches, and gestures to a device. You run it with the Android Debug Bridge (adb) tool.
You use it to stress-test your application and report back errors that are encountered. You can repeat a stream of events by running the tool each time with the same random number seed.

4.2.2.1 Monkey features

Monkey has many features, but it can all be summed up to these four categories.

- Basic configuration options
- Operational constraints
- Event types and frequencies
- Debugging options

4.2.2.2 Monkey Usage

In order to use monkey, open up a command prompt and just navigate to the following directory.

```
android ->sdk ->platform-tools
```

Once inside the directory, attach your device with the PC , and run the following command.

```
adb shell monkey -p your.package.name -v 500
```

This command can be broken down into these steps.

- adb - Android Debug Bridge. A tool used to connect and sends commands to your Android phone from a desktop or laptop computer.
- shell - shell is just an interface on the device that translates our commands to system commands.
- monkey - monkey is the testing tool.
- v - v stands for verbose method.
- 500- it is the frequency count or the number of events to be sent for testing.

Table 4.1 Android Studio Testing

Steps	Description
1	You will useAndroid studio to create an Android application under a package com.tutorialspoint.myapplication.
2	Modify src/MainActivity.java file to add Activity code.
3	Modify layout XML file res/layout/activity_main.xml add any GUI component if required.
4	Create src/second.java file to add Activity code.
5	Modify layout XML file res/layout/view.xml add any GUI component if required.
6	Run the application and choose a running android device and install the application on it

	and verify the results.

Let us try to run your Android Testing application. I assume you have connected your actual Android Mobile device with your computer. To run the app from Android studio, open one of your project's activity files and click Run icon from the toolbar. Before starting your application, Android studio will display following window to select an option where you want to run your Android application.
Select your mobile device as an option and then check your mobile device which will display application screen. Now just follow the steps mentioned at the top under the monkey section in order to perform testing on this application.

4.3 Testing with test cases

4.3.1 Testing of Android Activities

Sl. no.	Test Case id	Test Cases	Description	Steps to execute	Test data/ input	Expecte d result	Actual Result	Pass /Fail
1.	TC1	Splash Screen	To check the working of splash screen activity	Launching the app by pressing the app icon.	Splash screen image	Proper working of splash screen.	Splash screen worked properl y.	Pass
2.	TC2	Launch of Main Activity	To check smooth loading of the app after the splash screen activity.	Note the time taken for loading of smoothnes s	Main Activity	Proper and smooth working of Main Activity	Main Activit y works fine.	Pass

3.	TC3	ChatBot	Smooth Working of the Chatbot button and hence its activity.	Note the time required for the reply to come from Bot side.	Query Input by the user.	Bot responses should be quick.	Bot responses were not quick many a times. But sometimes they were instant.	Pass
4.	TC4	Stress buster deactivated.	Check if pressing the Stress Buster button app make the app crash.	Press the stress buster button multiple times.	Stress Buster Button click.	Application should not crash.	The app does not crashes. The button is deactivated.	Pass
5.	TC5	Profiles	Check if all the person's pics are linked to their professional profiles.	Press the image of the team members.	Image button clicks	Image button when pressed should redirect to their	Profiles are opened when the images are	Pass

						profiles.	clicked.	

4.3.2 Testing of chatbot

Sl. No.	Test Case id	Test cases	Description	Steps to execute	Test data/ input	Expected result	Actual result	Pass/Fail
1.	TC_01	Unit test	Check the chatbot when it is not integrated with android platform.	Run your query inputs on IBM platfor m.	Various query inputs to check the chatbot.	Query responses should be appropriate.	We get appropriate results for many set of inputs but some results are not up to the mark.	Pass
2.	TC_02	NLP check	To check if the sentences sharing the same intent are considered same	Query to be inputted in the text box and send button is to be clicked	Query serving same intent but in various way.	Query with same intent should be handled in the same manner.	Query with same intentsare handled in the same manner. For reference, check figure 5.3	Pass
3.	TC_03	Intent	To check	Query	Query	Queries	Not all the	Pass

		check	which intents are supported	to be inputted in the text box and send button needs to be clicked	serving various intents.	with different intents should be handled.	intents are handled. However, we get appropriate results for some query.	

Chapter 5

Results

5.1 Screenshots of RoboRelief Application

5.1.1 Splash Screen

Description: - Splash screen is fired whenever application is launched and the app is not in recent apps.

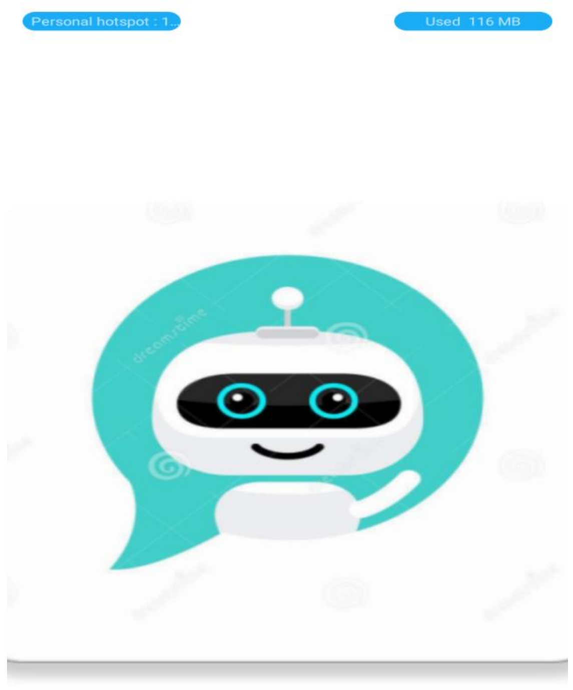


Figure 5.1 Splash screen of RoboRelief

5.1.2 Home page (Main Activity)

Description: - It contains three modules, which will lead to the three different fields. The three fields are namely as follows:

- Wanna Chat
- Stress Buster
- Meet the team

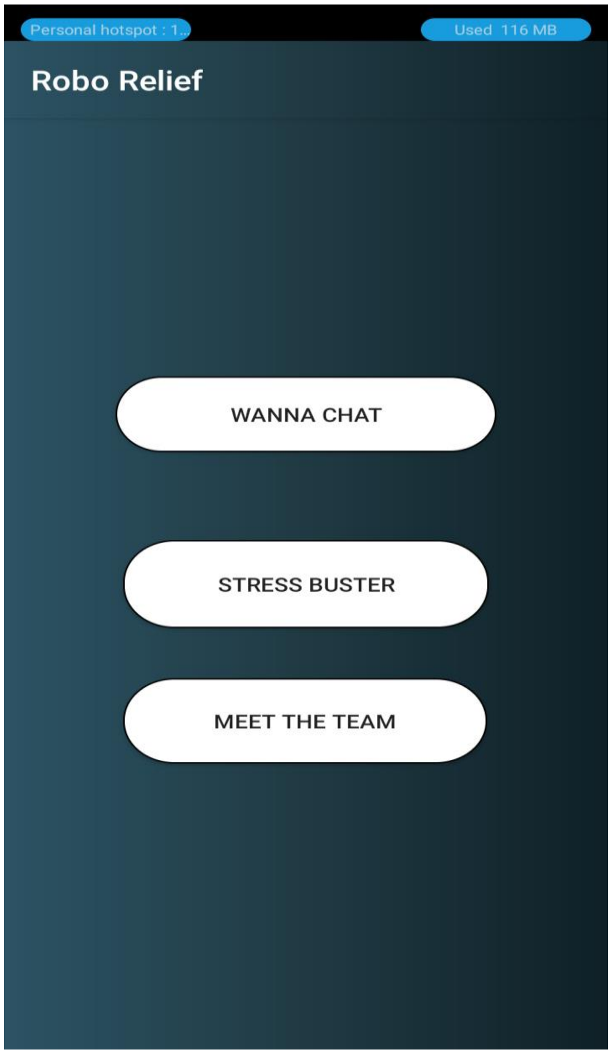
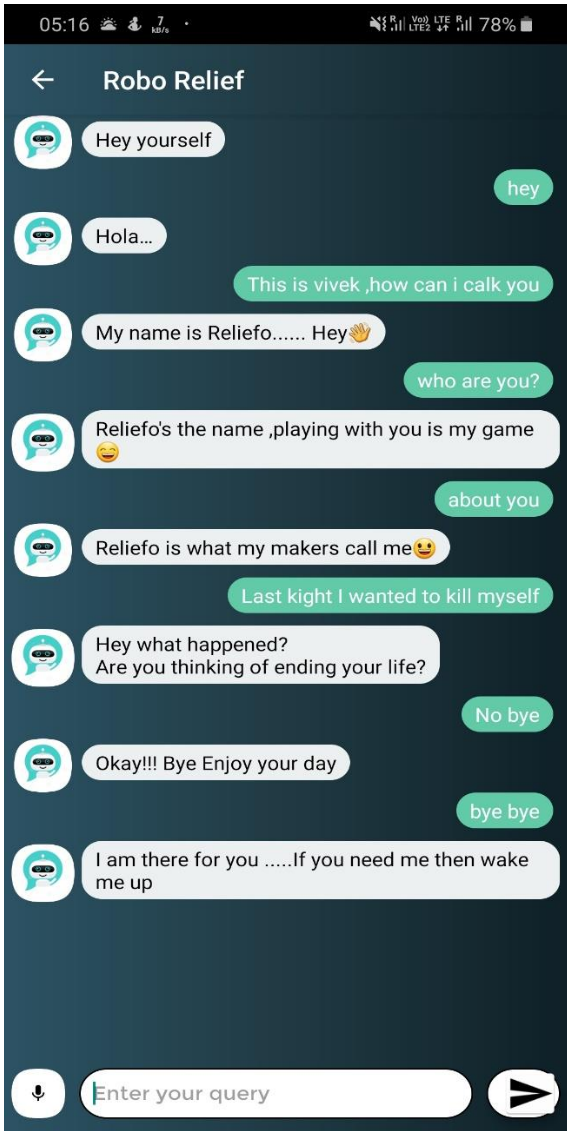


Figure 5.2 Home page (Main Activity page of RoboRelief)

5.1.3 Wanna chat

Description: -It is the first and foremost activity in our project. It is the most important activity of the project as it covers up the objective of the project. It is a chatbot, which will respond to the user and effectively handle all the situations. Starting from the greetings, it will cover up all the depression and stress related response so that who all uses this chatbot feel stress relieved. Since, this part uses dataset and we have data in less number so there is a scope of improvement in this part.


05:16 78%

Robo Relief

Hey yourself

hey

Hola...

This is vivek ,how can i calk you

My name is Reliefo...... Hey👋

who are you?

Reliefo's the name ,playing with you is my game 😄

about you

Reliefo is what my makers call me😃

Last kight I wanted to kill myself

Hey what happened?
Are you thinking of ending your life?

No bye

Okay!!! Bye Enjoy your day

bye bye

I am there for you .....If you need me then wake me up

Enter your query

Figure 5.3 Wanna chat activity

5.1.4 Stress Buster

Description: - It is the second activity in our project. It also covers a part of the objective of the project. It will use the conversation to know what your current mood is and will give you the music recommendations, which will try to heal you mentally. This part of the application is under construction since we need to have online storage for the music so that the application size is reasonable and uses least CPU of the user's device. This will help in smooth execution of the application.

5.1.5 Meet the Team

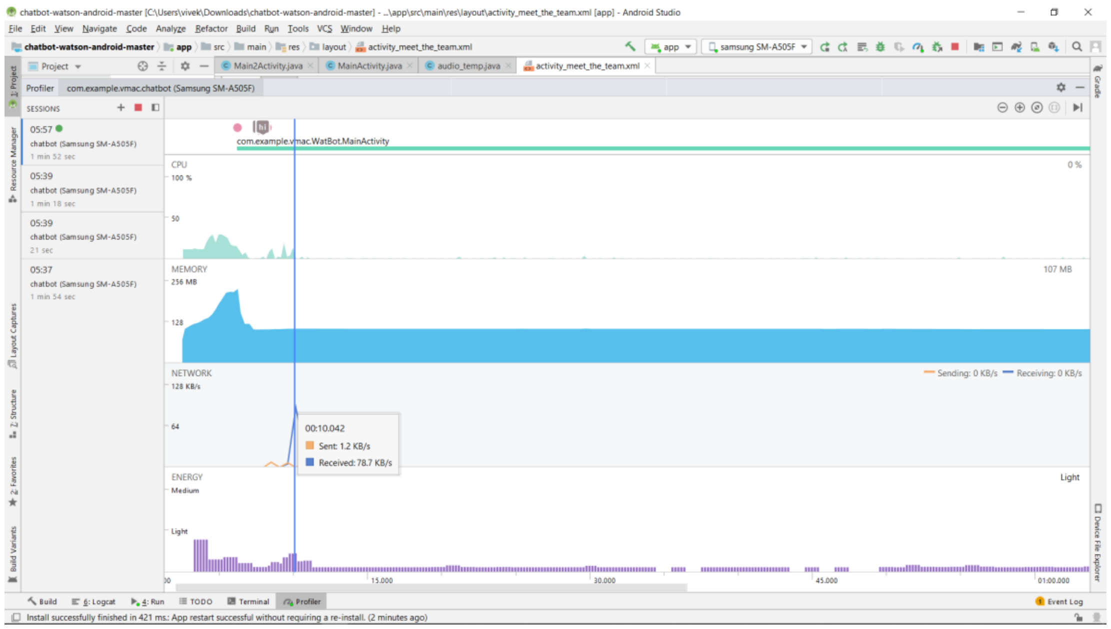
Description: - It is the third activity of our project. It is just a basic activity, which has all the team members' profiles. All the team members'pictures has been linked to their professional profiles. Here, professional profiles means LinkedIn profile. It can be seen in figure5.4.

5.2 Performance

The Android Profiler in Android Studio 3.0 and higher replaces the Android Monitor tools. The Android Profiler tools provide real-time data to help you to understand how your app uses CPU, memory, network, and battery resources. The Android Profiler is compatible with Android 5.0 (API level 21) and higher.When you launch a debuggable app, that process is selected by default.

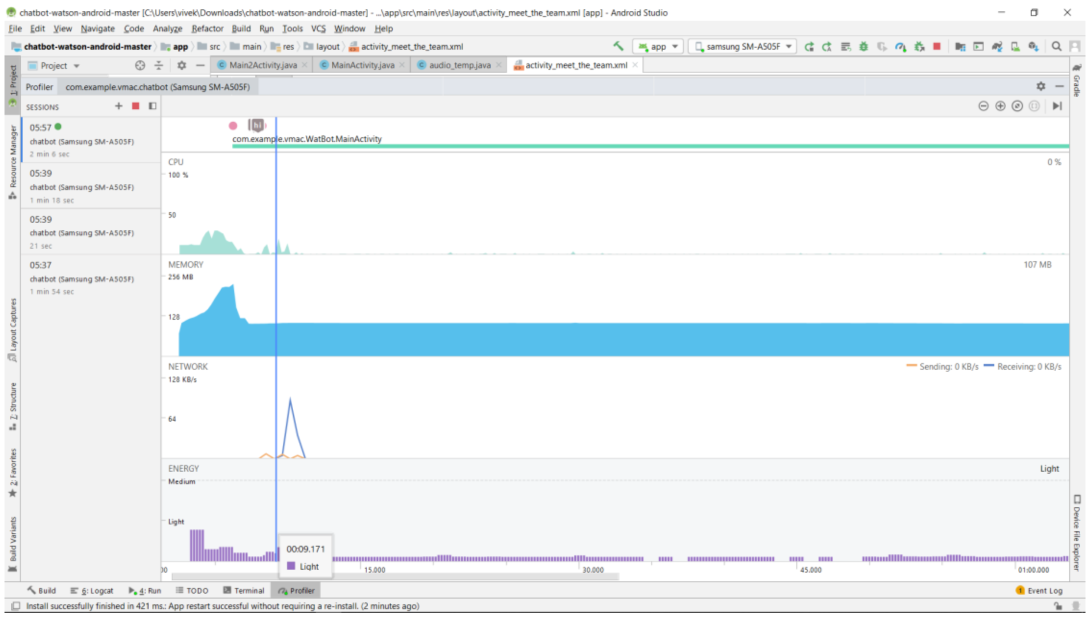
Android Profiler continues to collect profiling data until you disconnect the device or click **End Session**. The profiler section of the application that is used to monitor the performance of app is shown in Figure 5.5, 5.6, 5.7.

Figure 5.4 Meet the team


chatbot-watson-android-master [C:\Users\vivek\Downloads\chatbot-watson-android-master] - ...\app\src\main\res\layout\activity_meet_the_team.xml [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Main2Activity.java
MainActivity.java
audio_temp.java
activity_meet_the_team.xml
Profiler com.example.vmac.chatbot (Samsung SM-A505F)
SESSIONS
05:57 chatbot (Samsung SM-A505F) 1 min 52 sec
05:39 chatbot (Samsung SM-A505F) 1 min 18 sec
05:39 chatbot (Samsung SM-A505F) 21 sec
05:37 chatbot (Samsung SM-A505F) 1 min 54 sec
com.example.vmac.WatBot.MainActivity
CPU 100 % 50 0 %
MEMORY 256 MB 128 107 MB
NETWORK 128 KB/s 64
Sending: 0 KB/s Receiving: 0 KB/s
00:10.042
Sent: 1.2 KB/s
Received: 78.7 KB/s
ENERGY Medium Light
15.000 30.000 45.000 01:00.000
Build 6: Logcat 4: Run TODO Terminal Profiler Event Log

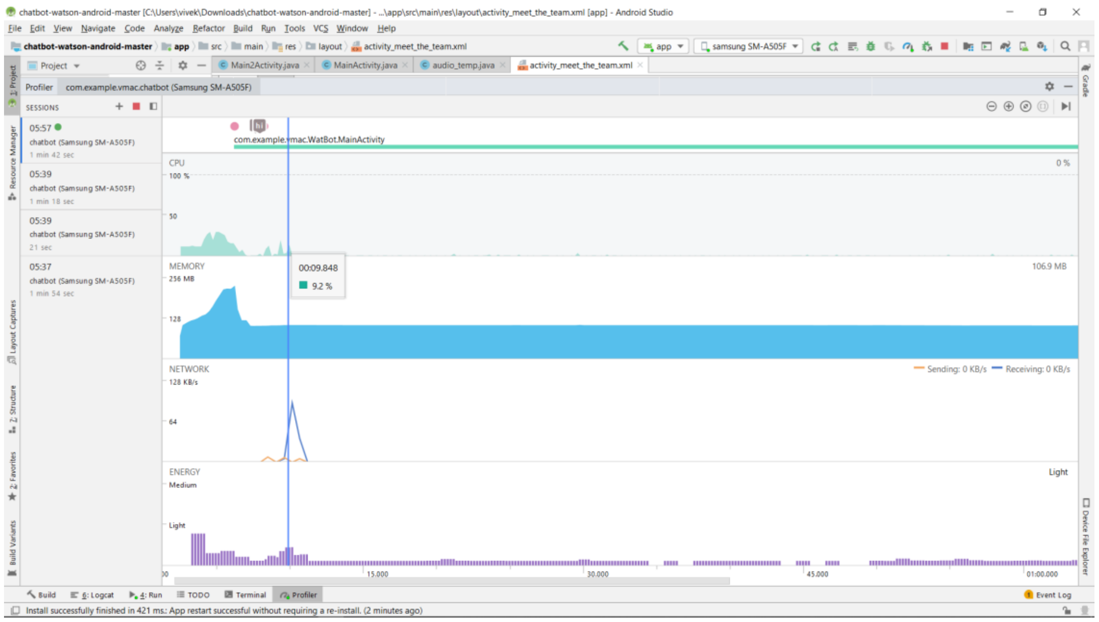
Install successfully finished in 421 ms.: App restart successful without requiring a re-install. (2 minutes ago)

Figure 5.5Network used by the application


chatbot-watson-android-master [C:\Users\vivek\Downloads\chatbot-watson-android-master] - ...\app\src\main\res\layout\activity_meet_the_team.xml [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Main2Activity.java
MainActivity.java
audio_temp.java
activity_meet_the_team.xml
Profiler com.example.vmac.chatbot (Samsung SM-A505F)
SESSIONS
05:57 chatbot (Samsung SM-A505F) 2 min 6 sec
05:39 chatbot (Samsung SM-A505F) 1 min 18 sec
05:39 chatbot (Samsung SM-A505F) 21 sec
05:37 chatbot (Samsung SM-A505F) 1 min 54 sec
com.example.vmac.WatBot.MainActivity
CPU 100 % 50 0 %
MEMORY 256 MB 128 107 MB
NETWORK 128 KB/s 64
Sending: 0 KB/s Receiving: 0 KB/s
ENERGY Medium Light
00:09.171
Light
15.000 30.000 45.000 01:00.000
Build 6: Logcat 4: Run TODO Terminal Profiler Event Log

Install successfully finished in 421 ms.: App restart successful without requiring a re-install. (2 minutes ago)

Figure 5.6 Power used by the application


chatbot-watson-android-master [C:\Users\vivek\Downloads\chatbot-watson-android-master] - ...\app\src\main\res\layout\activity_meet_the_team.xml [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
chatbot-watson-android-master app src main res layout activity_meet_the_team.xml
app
samsung SM-A505F
Project
Main2Activity.java
MainActivity.java
audio_temp.java
activity_meet_the_team.xml
Profiler com.example.vmac.chatbot (Samsung SM-A505F)
SESSIONS
05:57 chatbot (Samsung SM-A505F) 1 min 42 sec
05:39 chatbot (Samsung SM-A505F) 1 min 18 sec
05:39 chatbot (Samsung SM-A505F) 21 sec
05:37 chatbot (Samsung SM-A505F) 1 min 54 sec
com.example.vmac.WatBot.MainActivity
CPU
100 %
50
0 %
00:09.848
9.2 %
MEMORY
256 MB
128
106.9 MB
NETWORK
128 KB/s
64
Sending: 0 KB/s
Receiving: 0 KB/s
ENERGY
Medium
Light
Light
15.000
30.000
45.000
01:00.000
Build
6: Logcat
4: Run
TODO
Terminal
Profiler
Event Log

Install successfully finished in 421 ms.: App restart successful without requiring a re-install. (2 minutes ago)

Figure 5.7 CPU usage of the application

Chapter 6

Deployment

6.1Overview of deployment process

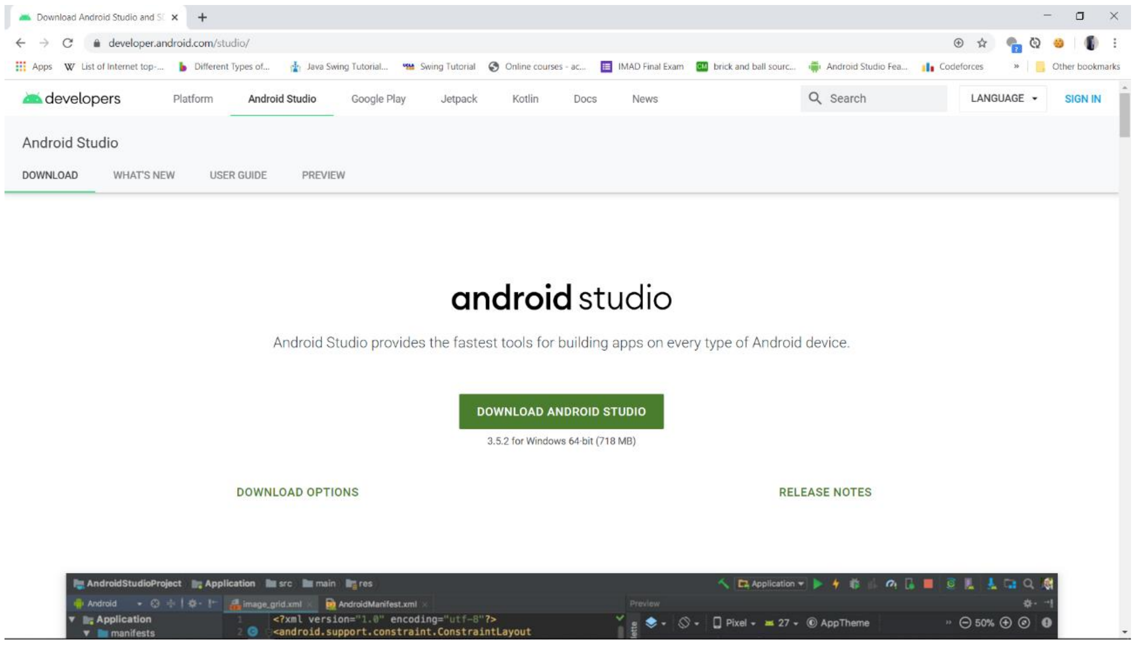
Deployment of a project is the accumulation of all the activities, which make a software available for use. So, keeping things short we will tell how to deploy our android chatbot on your system. This is an android application unlike the other software deployments, our project deployment can be done in any of the two ways:-

- Installing the RoboRelief application through its apk file, which you can download from the GitHub link.
- Downloading our project from the GitHub repository named RoboRelief and then opening it with Android Studio and then configuring it as guided below.

First, we need to have android studio in our system (Laptop or pc) so that you can run our project. If you do not have the software you need to install it first or else you need to use the first way to get our application in your android smartphone by downloading it from our GitHub link.

6.2Installation of Android Studio

First, you need to download Android Studio for your system. If you are using a windows system, make sure that you get a windows installer for Android Studio and so on if you are using a Linux based system or a Mac. After successful download of Android Studio installer you will have a .exe file in your downloads of your pc.



android studio

Android Studio provides the fastest tools for building apps on every type of Android device.

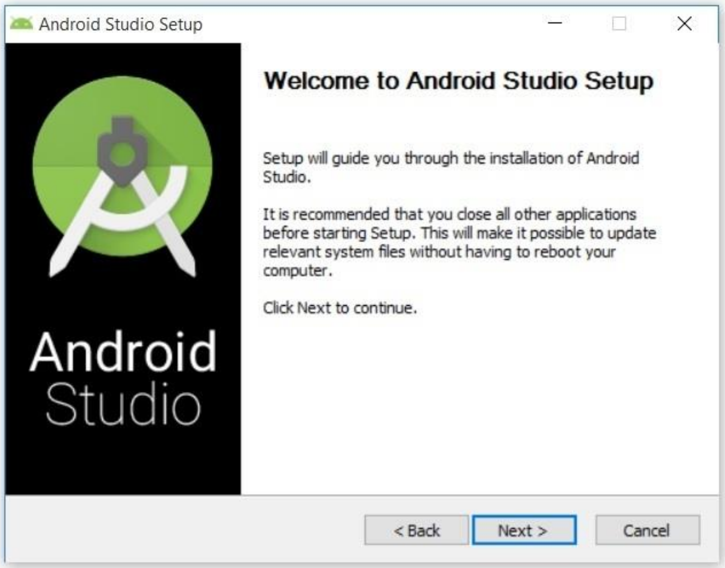
3.5.2 for Windows 64-bit (718 MB)

Figure 6.1 Downloading android studio

To install Android Studio on Windows, proceed as follows:

Step 1: If you downloaded an .exe file (recommended), double-click to launch it.

If you downloaded a .zip file, unpack the ZIP, copy the **android-studio** folder into your **Program Files** folder, and then open the **android-studio > bin** folder and launch studio64.exe (for 64-bit machines) or studio.exe (for 32-bit machines).



Welcome to Android Studio Setup

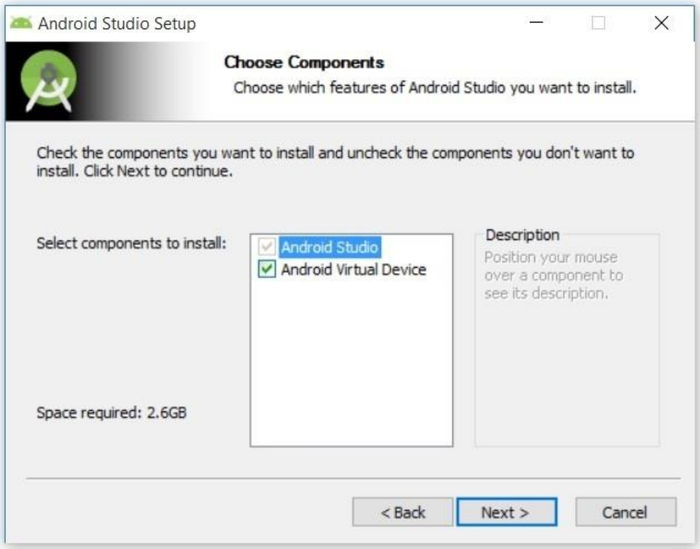
Setup will guide you through the installation of Android Studio.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Figure 6.2 Android Studio Setup dialog box

Step 2: Clicking Next will take you to the following panel, which provides the option to decline installing an Android Virtual Device (AVD).


Android Studio Setup

Choose Components

Choose which features of Android Studio you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

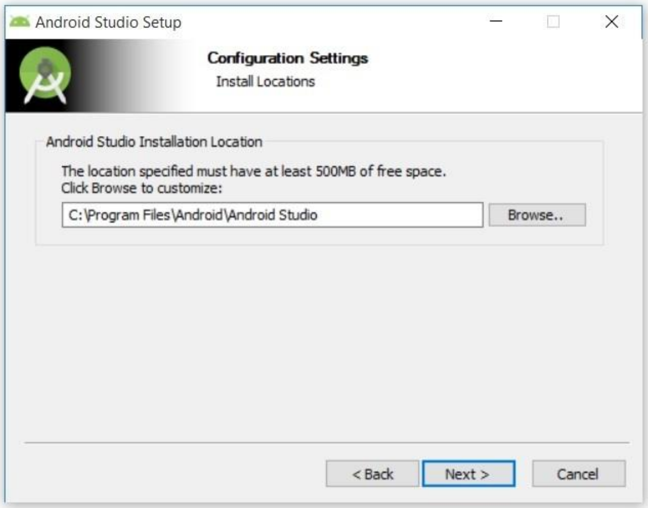
Select components to install:
- [✓] Android Studio
- [✓] Android Virtual Device

Description: Position your mouse over a component to see its description.

Space required: 2.6GB

Figure 6.3 Install an Android AVD?

Step 3: Choose to keep the default settings (Recommended). After clicking Next, you will be taken to the Configuration Settings panel, where I was asked to choose where to install Android Studio.


Android Studio Setup

Configuration Settings

Install Locations

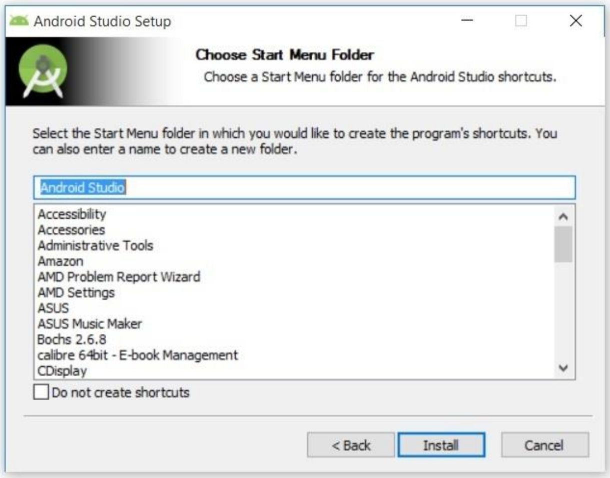
Android Studio Installation Location

The location specified must have at least 500MB of free space. Click Browse to customize:

C:\Program Files\Android\Android Studio

Figure 6.4 The installation location must have at least 500MB free space

Step 4: Keep the default installation location and click Next, and you will be greeted with the Choose Start Menu Folder panel.



Android Studio Setup

Choose Start Menu Folder

Choose a Start Menu folder for the Android Studio shortcuts.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

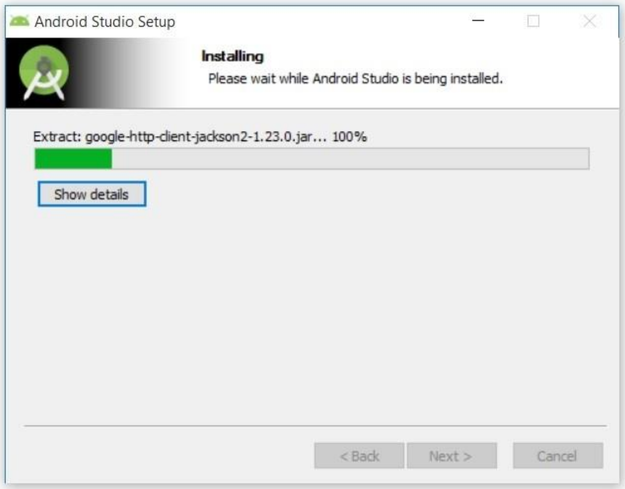
Android Studio

Accessibility
Accessories
Administrative Tools
Amazon
AMD Problem Report Wizard
AMD Settings
ASUS
ASUS Music Maker
Bochs 2.6.8
calibre 64bit - E-book Management
CDisplay

☐ Do not create shortcuts

Figure 6.5 select the folder in which to store Android Studio shortcuts

Step 5: Keep the default setting and then click Install. The following Installing panel will appear:



Android Studio Setup

Installing

Please wait while Android Studio is being installed.

Extract: google-http-client-jackson2-1.23.0.jar... 100%

Figure 6.6 This panel shows the progress of the installation

Step 6: Clicking Show details causes the names of files being installed and other activities to be displayed. When installation finishes, the Installation Complete panel will appear.

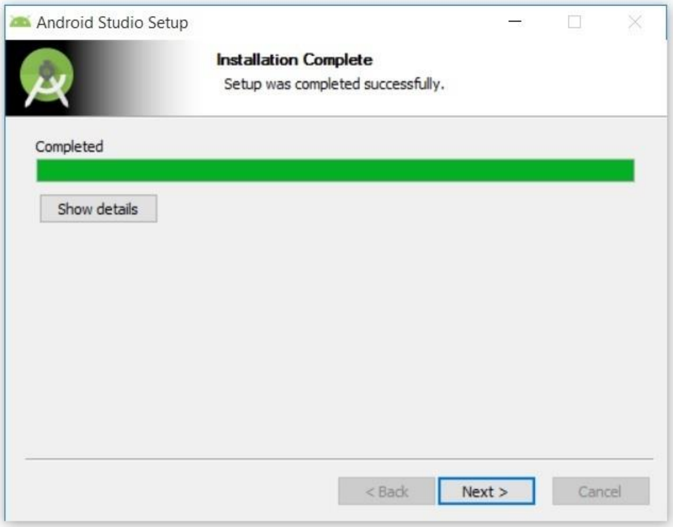


Figure 6.7 The Next button is enabled when installation completes

Step 7: After clicking Next, the installer will present the Completing Android Studio Setup panel.

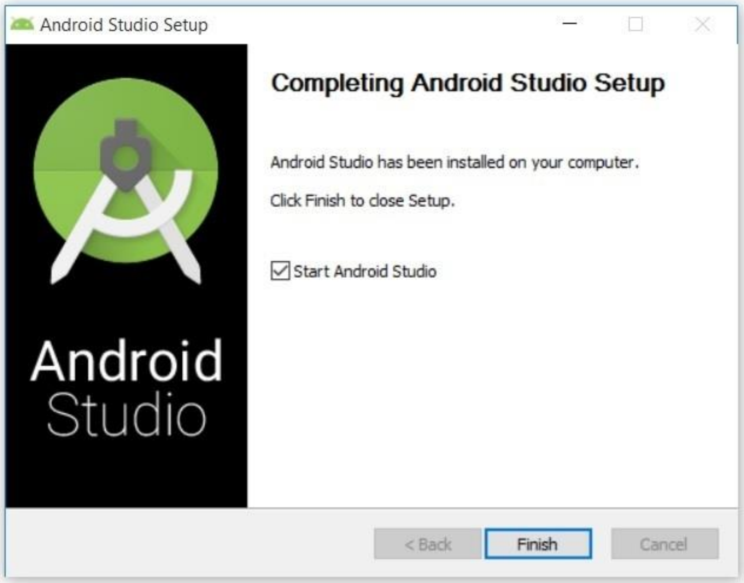

Android Studio Setup
Completing Android Studio Setup
Android Studio has been installed on your computer.
Click Finish to close Setup.
Start Android Studio
Android Studio
< Back
Finish
Cancel

Figure 6.8 Uncheck the Android Studio checkbox

To complete the installation, I unchecked the Start Android Studio box and clicked Finish.

6.3Installing RoboRelief from apk file

Step 1: Open the GitHub link in your web browser.

Link: - https://github.com/vivekvvk143/RoboRelief/blob/master/RoboRelief.apk

Step2: If you are using your desktop or laptop for opening this link follow this step.

Step 2.1: Click the "Clone or Download" link option and then click "download Zip"

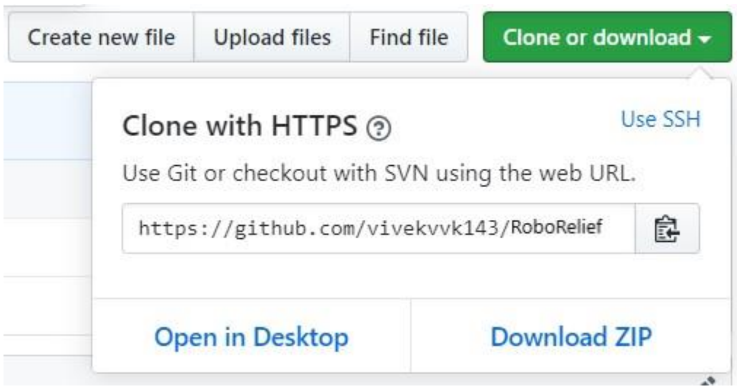


Figure 6.9 Downloading apk zip from GitHub Link

Step 2.2: You need to have an android smartphone for running our application. Get an Android phone and connect to your pc or laptop and transfer the apk to your mobile phone and thereby install the application there itself.

Step 3: If you are using an Android smartphone to open the above link follow this step

Step 3.1: After opening this link, click "Download ZIP" and downloading process will start.

Step3.2: After the downloading has completed, Go to the Downloads folder of Internal Storage in your android smartphone and click on the RoboRelief.apk file as shown in figure 6.2.

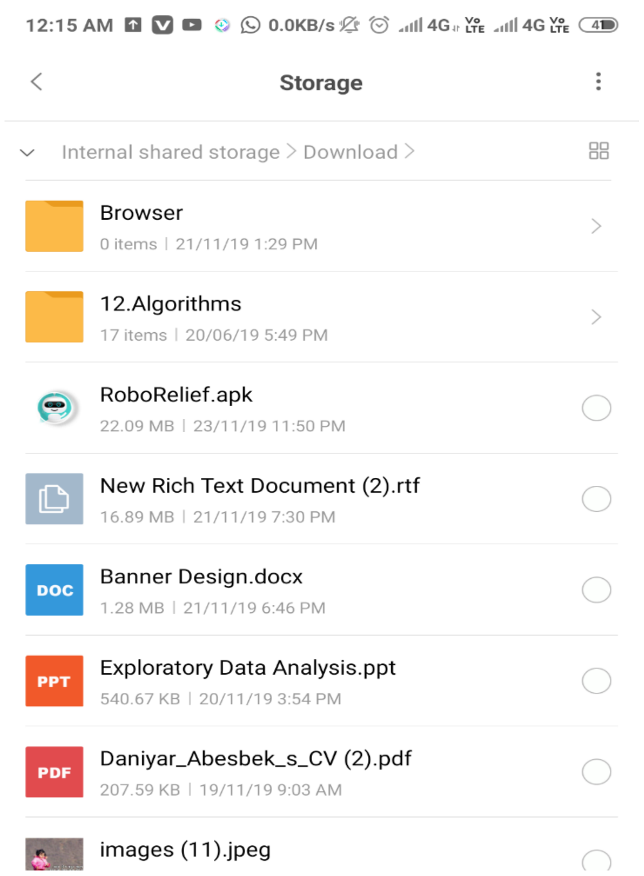

12:15 AM
Storage
Internal shared storage > Download >
Browser
0 items | 21/11/19 1:29 PM
12.Algorithms
17 items | 20/06/19 5:49 PM
RoboRelief.apk
22.09 MB | 23/11/19 11:50 PM
New Rich Text Document (2).rtf
16.89 MB | 21/11/19 7:30 PM
Banner Design.docx
1.28 MB | 21/11/19 6:46 PM
Exploratory Data Analysis.ppt
540.67 KB | 20/11/19 3:54 PM
Daniyar_Abesbek_s_CV (2).pdf
207.59 KB | 19/11/19 9:03 AM
images (11).jpeg

Figure 6.10 Downloads section on your Android phone

6.4Download the project.

Step 1: Open the link in your desktop browser and click "clone or download" and download it

Link :https://github.com/vivekvvk143/RoboRelief

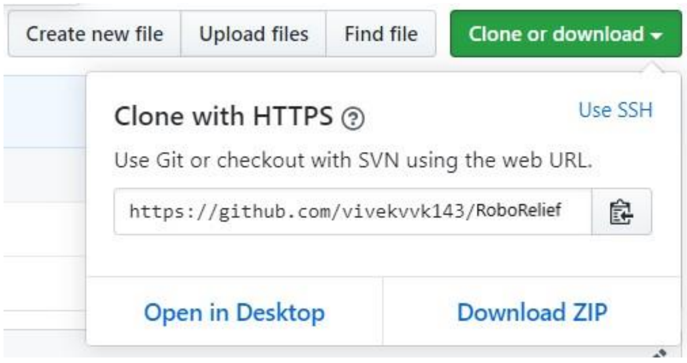
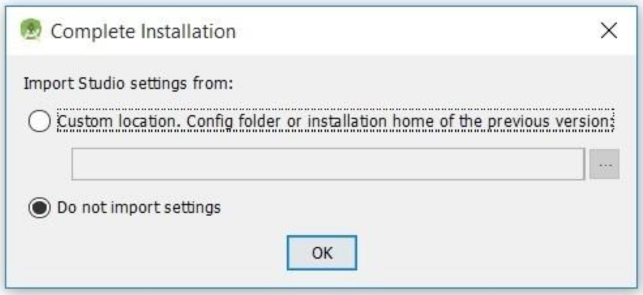


Figure 6.11 Downloading project ZIP on your desktop or Laptop

Step 2: Go to the section, where your downloaded files goes and extract the zip, where you want.

Step 3: Now you need to complete the installation of Android Studio to use our application. If you are following our deployment process sequentially, you must have installed it on your system as per the installation guide discussed in section 6.2.

Step 4: Run android studio on your system.The first time Android Studio runs, it presents a **Complete Installation** dialog box that offers the option of importing settings from a previous installation.



Complete Installation

Import Studio settings from:

( ) Custom location. Config folder or installation home of the previous version:

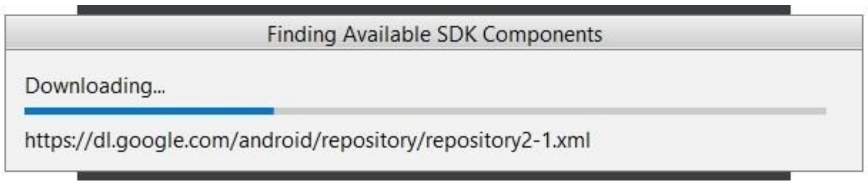
(●) Do not import settings

Figure 6.12 A previous installation's settings can be imported

Step 5: Choose not to import settings (the default selection) and then click OK, and you will be rewarded with the following splash screen:

Figure 6.13 Android Studio's splash screen

You will observe the following Finding Available SDK Components message box. Let the downloading of the SDK components complete.



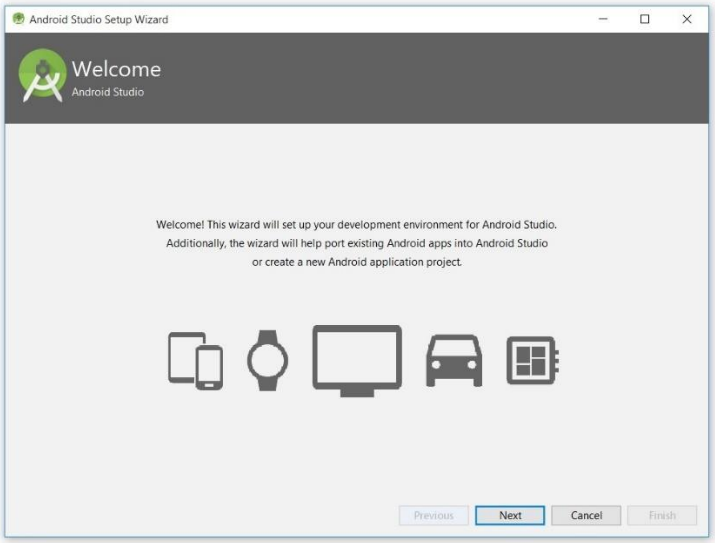
Finding Available SDK Components

Downloading...

https://dl.google.com/android/repository/repository2-1.xml

Figure 6.14 Android Studio downloads any SDK components that are needed (and available)

At this point, Android Studio presents the following Android Studio Setup Wizard dialog box:

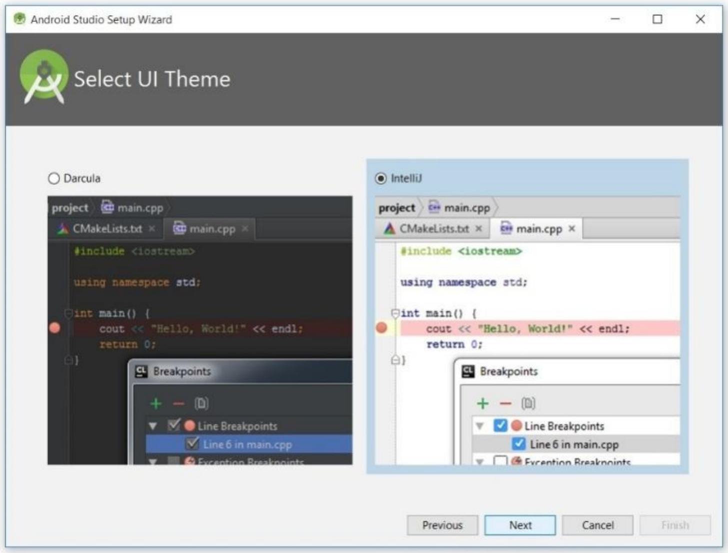


Welcome! This wizard will set up your development environment for Android Studio. Additionally, the wizard will help port existing Android apps into Android Studio or create a new Android application project.

Figure 6.15 The wizard provides setup and app-porting capabilities

I clicked Next, and the wizard will invite you to select an installation type. Keep the default standard setting.

Choose the theme, which interests you as shown in figure 6.16



Darcula [✓ IntelliJ]

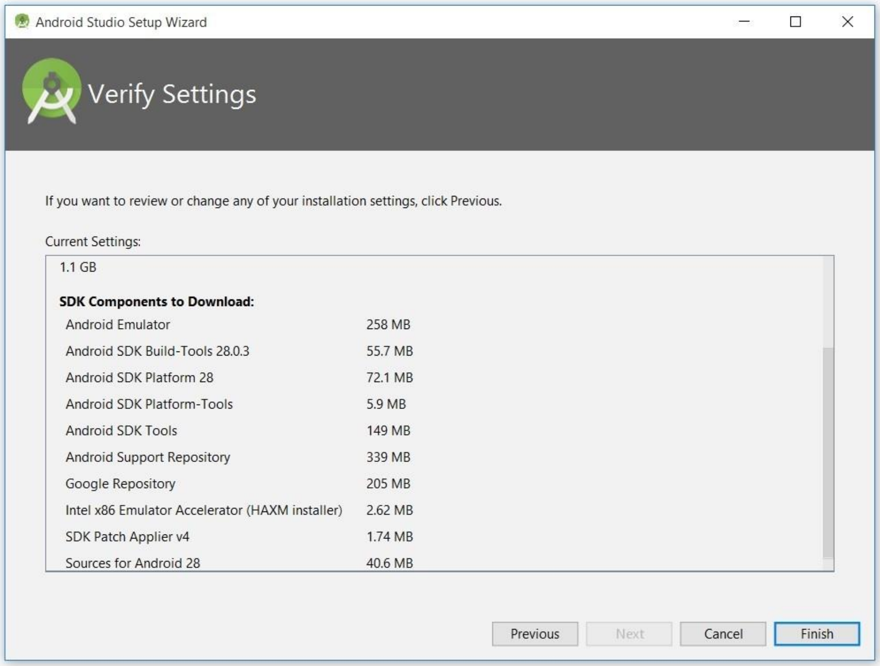
```
#include <iostream>

using namespace std;

int main() {
    cout << "Hello, World!" << endl;
    return 0;
}
```

Figure 6.16 Choose your theme

Keep the default theme setting and click Next. Android Studio next provides the opportunity to verify settings.



If you want to review or change any of your installation settings, click Previous.

Current Settings:

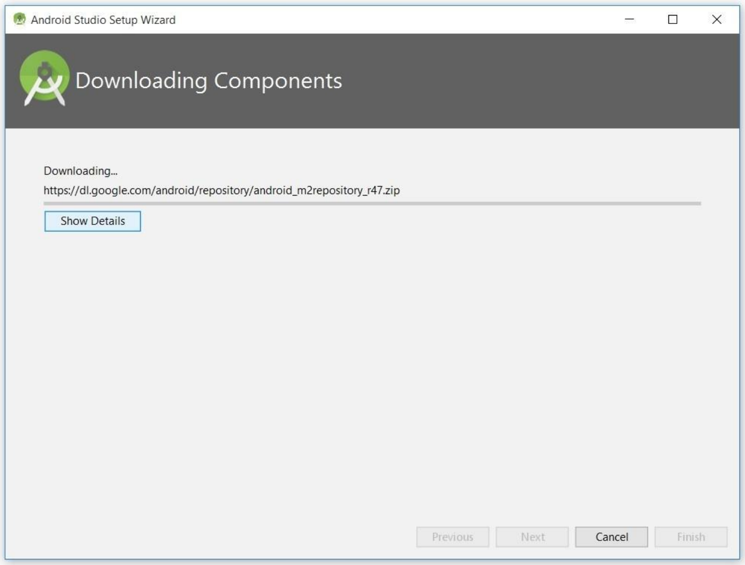
1.1 GB

SDK Components to Download:

Android Emulator	258 MB
Android SDK Build-Tools 28.0.3	55.7 MB
Android SDK Platform 28	72.1 MB
Android SDK Platform-Tools	5.9 MB
Android SDK Tools	149 MB
Android Support Repository	339 MB
Google Repository	205 MB
Intel x86 Emulator Accelerator (HAXM installer)	2.62 MB
SDK Patch Applier v4	1.74 MB
Sources for Android 28	40.6 MB

Figure 6.17 Android Studio identifies additional SDK components that will be downloaded

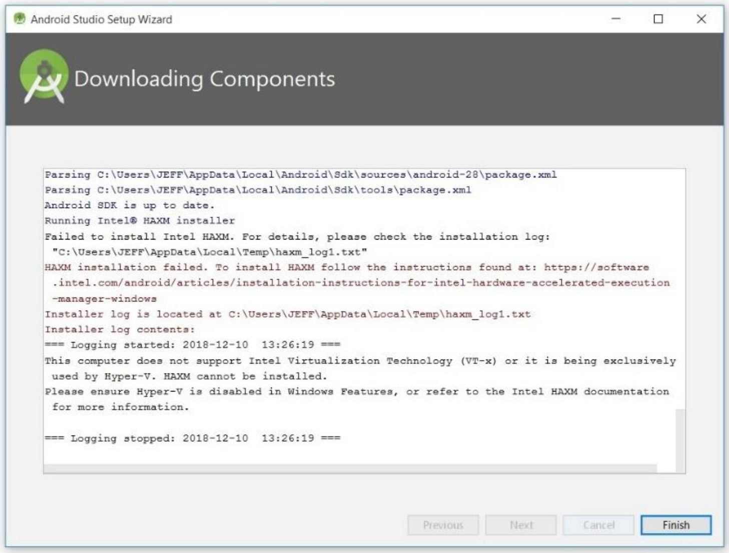
Click Finish and Android Studio will begin the process of downloading SDK components.



Downloading...
https://dl.google.com/android/repository/android_m2repository_r47.zip

Figure 6.18 The wizard downloads and unzips SDK components

It can take several minutes for this part of the setup to finish. Clicking Show Details might relieve some boredom by revealing the various files being downloaded and unzipped.



```
Parsing C:\Users\JEFF\AppData\Local\Android\Sdk\sources\android-28\package.xml
Parsing C:\Users\JEFF\AppData\Local\Android\Sdk\tools\package.xml
Android SDK is up to date.
Running Intel® HAXM installer
Failed to install Intel HAXM. For details, please check the installation log:
 "C:\Users\JEFF\AppData\Local\Temp\haxm_log1.txt"
HAXM installation failed. To install HAXM follow the instructions found at: https://software
 .intel.com/android/articles/installation-instructions-for-intel-hardware-accelerated-execution
 -manager-windows
Installer log is located at C:\Users\JEFF\AppData\Local\Temp\haxm_log1.txt
Installer log contents:
=== Logging started: 2018-12-10  13:26:19 ===
This computer does not support Intel Virtualization Technology (VT-x) or it is being exclusively
 used by Hyper-V. HAXM cannot be installed.
Please ensure Hyper-V is disabled in Windows Features, or refer to the Intel HAXM documentation
 for more information.

=== Logging stopped: 2018-12-10  13:26:19 ===
```

Figure 6.19 Downloading Components finished

After clicking, finish button to complete the wizard, the Welcome to Android Studio dialog box will appear**.**

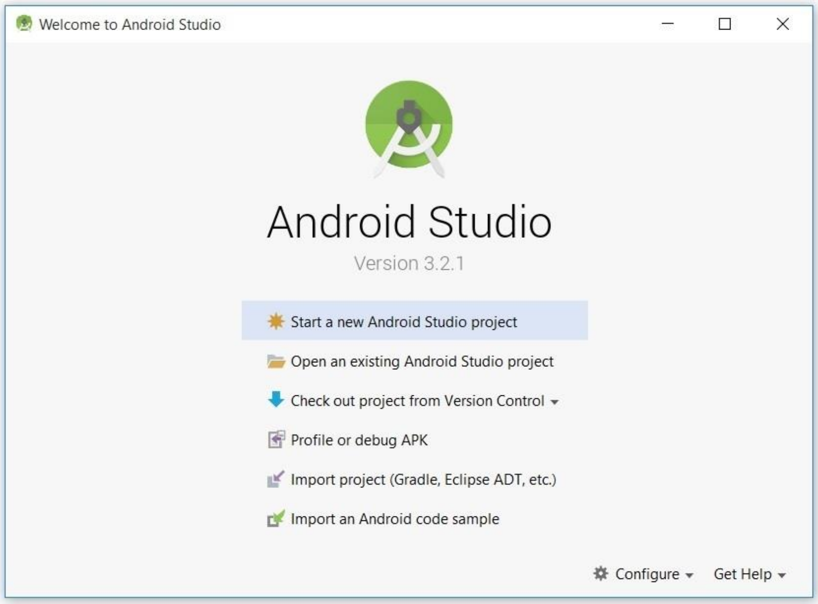
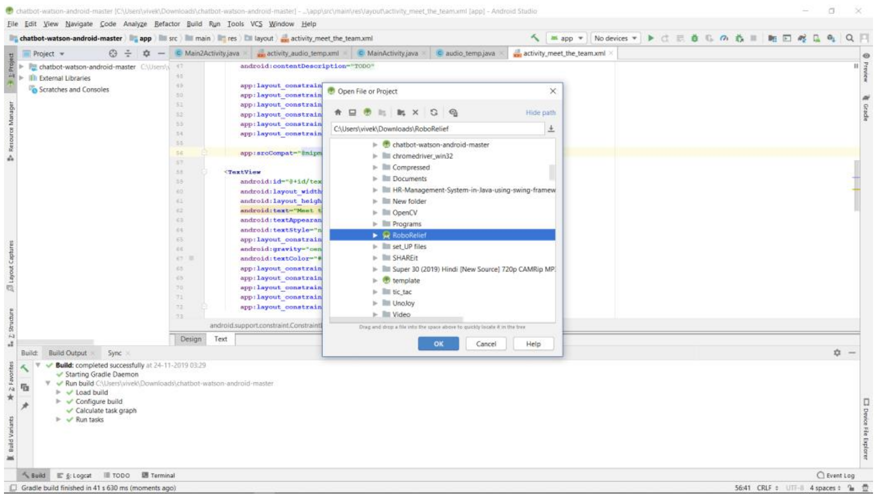

Welcome to Android Studio
Android Studio
Version 3.2.1
Start a new Android Studio project
Open an existing Android Studio project
Check out project from Version Control
Profile or debug APK
Import project (Gradle, Eclipse ADT, etc.)
Import an Android code sample
Configure
Get Help

Figure 6.20 Welcome to Android Studio

Step 6: Select "Open an existing Android Studio project" and then guide it to the place where you have the unpacked zip of the downloaded project.


chatbot-watson-android-master [C:\Users\vivek\Downloads\chatbot-watson-android-master] - ...\app\src\main\res\layout\activity_meet_the_team.xml [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
chatbot-watson-android-master app src main res layout activity_meet_the_team.xml
app No devices
Project
chatbot-watson-android-master
External Libraries
Scratches and Consoles
Main2Activity.java activity_audio_temp.xml MainActivity.java audio_temp.java activity_meet_the_team.xml
Open File or Project
Hide path
chatbot-watson-android-master
chromedriver_win32
Compressed
Documents
HR-Management-System-in-Java-using-swing-framew
New folder
OpenCV
Programs
RoboRelief
set_UP files
SHAREit
Super 30 (2019) Hindi [New Source] 720p CAMRip MP
template
tic_tac
UnoJoy
Video
Drag and drop a file into the space above to quickly locate it in the tree
OK
Cancel
Help
Design
Text
Build: Build Output Sync
Build TODO Terminal Event Log
56:41 CRLF UTF-8 4 spaces

```
android:contentDescription="TODO"

app:layout_constrain
app:layout_constrain
app:layout_constrain
app:layout_constrain
app:layout_constrain
app:layout_constrain

app:srcCompat="@nipm

<TextView
    android:id="@+id/tex
    android:layout_width
    android:layout_heigh
    android:text="Meet t
    android:textAppearan
    android:textStyle="n
    app:layout_constrain
    android:gravity="cen
    android:textColor="#
    app:layout_constrain
    app:layout_constrain
    app:layout_constrain
    app:layout_constrain
    app:layout_constrain
android.support.constraint.ConstraintL
```

```
C:\Users\vivek\Downloads\RoboRelief
```

```
Build: completed successfully at 24-11-2019 03:29
  Starting Gradle Daemon
  Run build C:\Users\vivek\Downloads\chatbot-watson-android-master
    Load build
    Configure build
    Calculate task graph
    Run tasks
```

```
Gradle build finished in 41 s 630 ms (moments ago)
```

Figure 6.21 Open the extracted project RoboRelief

Step 7: After clicking the "OK", the project will open up. Wait the gradle to synchronize. After the gradle project sync is completed and if it completes successfully then it is, very fine but if any error shows up click on the link, which pops up in the output window for error resolution. Errors if any will be resolved after the required files will be downloaded. Let all the required files to be downloaded for proper execution of the project.

Step 8: To run the application on your smartphone you need to connect your android smartphone. Make sure that your android version is android 5.0 (Lollipop) or above and has the developer options enabled. If it is not enabled, go to step 8.1.

Step 8.1: Tap on Build Number 7 times to enable developer options on any android phone. This rule is applicable for devices with Android 8.0 or higher.

Step 8.2: Return to previous screen to see Developer options near the bottom. Go into developer options and enable USB debugging mode.

Step 9: Connect your android smartphone to your device via USB and click the run button in the Android Studio.

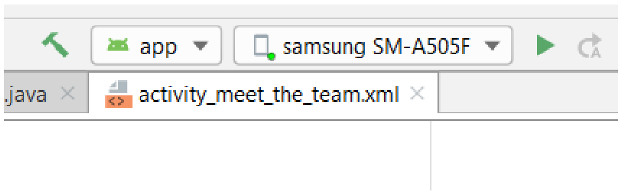


Figure 6.22 Green inverted triangle button is the run button

Make sure your device shows up in place of Samsung SM-A505F. As soon as you click the run button, the following figure 6.23 and figure 6.24 respectively shows up the screenshot of the splash screen and the main activity of RoboRelief.

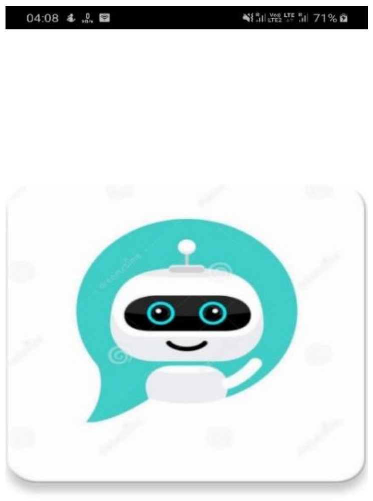


Figure 6.23 RoboRelief Icon would be displayed on your device's screen on app launch

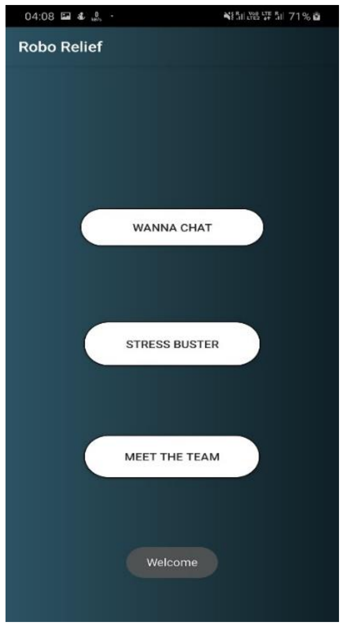


Figure 6.24 Main Activity of RoboRelief that will be displayed on your device

CONCLUSION

Now-a-days Chatbots are looked at from an assisting approach. This makes them unable to establish any friendly connection with the user. In addition, purely AI chat bots do not have set responses for set statements. Our approach will provide a response based on users' way of chatting as well as based on positively prepared responses. A study introduced a personalization framework using long-term memory. Another study used word2vec to generate natural sounding phrases. This will help establish a friendly connection with the user as well as provide positive thoughts to user. If the user keeps giving negative response over time, the chatbot can have a functionality to alert the users close ones. Doing this project gave us boost and a platform to use case our theory knowledge and thus implementing practical knowledge. We gained some insights of Android application making which we had not experienced earlier.

FUTURE SCOPE

- This application aims to connect with therapists and psychiatrist so that they can check the responses. We can connect with them so that they can justify and check all the query responses so that they can be justified.
- We aim to use proper CBT or Cognitive Behavioral Therapy to enhance our model and make it think more like a human.
- As mentioned above, the application will need some psychiatrists who will not only check the query response of the bot but also administer or look after the patients who will apply for a coach. We mean to say that we will have a paid section where at nominal cost the needy persons who need a coach can get help directly from the doctors directly through our platform.
- Since our Android application, has a section (activity) which is under development phase. We look forward to include this feature in our application in the next version.

REFERENCES

- Pratik Kataria, Kiran Rode, Akshay Jain, Prachi Dwivedi4 and Sukhada Bhingarkar Department of Computer Engineering, MIT-COE Pune India: International Journal of Pure and Applied Mathematics, Special Issue. User Adaptive Chatbot for Mitigating Depression

- World Health Organization, Depression: Let us talk. [Online]. Available: http://www.who.int/mental health/management/depression/en/

- The PHQ-9 :Validity of a Brief Depression Severity Measure .Link :- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1495268/Journal List>J Gen Intern Med>v.16(9); 2001 Sep> PMC1495268

- Chatbot testing Idea :https://chatbotslife.com/chatbot-testing-1f359b5459a

- IBM Watson documentation : https://www.ibm.com/cloud/watson-assistant/docs-resources/

- Android Studio Documentation :https://developer.android.com/docs

- https://pdfs.semanticscholar.org/26fb/6b4f917156760488025ae423e598e65417f2.pdf

- http://trap.ncirl.ie/2697/1/karllyons.pdf

- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6286427/

- Two founders left plush jobs London create AI chatbot fights mental illness depressionhttps://analyticsindiamag.com/two-founders-left-plush-jobs-london-create-ai-chatbot-fights-mental-illness-depression/

- Livemint.com, Over 5 crore people suffer from depression in India: WHO, 2017. [Online]. Available: http://www.livemint.com/Specials/Ysja8QtaVqjRpKg7eAFJfL /Over-5-crore-people-suffer-from-depression-in-IndiaWHO.html

- https://colah.github.io/posts/2015-08-Understanding-LSTMs/

APPENDIX A

Get started with Android Studio

Android Studio is Google's officially supported IDE for developing Android apps. This IDE is based on IntelliJ IDEA, which offers a powerful code editor and developer tools. Android Studio 3.2.1 includes the following features:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Instant Run to push changes to your running app without building a new APK
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to help you catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and Google App Engine
- Plugin architecture for extending Android Studio via plugins

Download Android Studio

Google provides Android Studio for Windows, Mac OS X, and Linux platforms. You can download Android Studio from the Android Studio homepage, where you'll also find the traditional SDKs with Android Studio's command-line tools. Before downloading Android Studio, make sure your platform meets the following requirements:

Windows requirements

- Microsoft Windows 7/8/10 (32-bit or 64-bit)
- 3 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution

Mac OS requirements

- Mac OS X 10.10 (Yosemite) or higher, up to 10.13 (High Sierra)
- 3 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)

- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution

Linux OS requirements

- GNOME or KDE desktop. Tested on Ubuntu 14.04 LTS, Trusty Tahr (64-bit distribution capable of running 32-bit applications)
- 64-bit distribution capable of running 32-bit applications
- GNU C Library (glibc) 2.19 or later
- 3 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution

APPENDIX B

Android Manifest.xml :

```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

   package="com.example.vmac.WatBot">



   <uses-permission android:name="android.permission.INTERNET" />

   <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>

   <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

   <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />

   <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />

   <uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />

   <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

   <uses-permission android:name="android.permission.RECORD_AUDIO" />

   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

   <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
   <application
       android:allowBackup="true"
       android:icon="@mipmap/ic_face"
       android:label="@string/app_name"
       android:supportsRtl="true"
       android:theme="@style/AppTheme">
       <activity android:name=".Stress_buster"
           android:theme="@style/team"></activity>
       <activity
           android:name=".Meet_the_team"
           android:parentActivityName=".Main2Activity"
           android:theme="@style/team">

           <!-- The meta-data tag is required if you support API level 15 and lower --
>

           <meta-data
               android:name="android.support.PARENT_ACTIVITY"
               android:value=".Main2Activity" />
       </activity>
       <activity android:name=".Main2Activity" />
       <!--
       meta-data
           android:name="com.google.android.actions"
           android:resource="@xml/actions" /
       -->
       <activity
           android:name=".MainActivity"
           android:parentActivityName=".Main2Activity"
           android:screenOrientation="portrait"
           android:theme="@style/AppTheme">
           <meta-data
```

```
                android:name="android.support.PARENT_ACTIVITY"

                android:value=".Main2Activity" />

        </activity>

        <activity

            android:name=".SplashActivity"

            android:screenOrientation="portrait"

            android:theme="@style/SplashTheme">

            <intent-filter>

                <action android:name="android.intent.action.MAIN" />


                <category android:name="android.intent.category.LAUNCHER" />

            </intent-filter>

        </activity>

    </application>


</manifest>
```

Activity_Main.xml :

```
<?xml version="1.0" encoding="utf-8"?>

<android.support.design.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"

   xmlns:tools="http://schemas.android.com/tools"

   android:layout_width="match_parent"

   android:layout_height="match_parent"

   android:fitsSystemWindows="true"

   android:background="@drawable/gradient"

   tools:context="com.example.vmac.WatBot.MainActivity">


   <!--<android.support.design.widget.AppBarLayout

        android:layout_width="match_parent"

        android:layout_height="wrap_content">
        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"/>

    </android.support.design.widget.AppBarLayout>-->

    <include
        layout="@layout/content_chat_room"
        android:visibility="visible" />

</android.support.design.widget.CoordinatorLayout>
```

Activity_Main.java :

```
package com.example.vmac.WatBot;

import android.Manifest;
import android.content.Context;
import android.content.pm.PackageManager;
import android.graphics.Typeface;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.DefaultItemAnimator;
import android.support.v7.widget.LinearLayoutManager;

import android.support.v7.widget.RecyclerView;

import android.util.Log;

import android.view.View;

import android.view.Window;

import android.widget.EditText;

import android.widget.ImageButton;

import android.widget.Toast;


import com.ibm.cloud.sdk.core.http.HttpMediaType;

import com.ibm.cloud.sdk.core.http.ServiceCall;

import com.ibm.cloud.sdk.core.security.IamAuthenticator;

import com.ibm.watson.developer_cloud.android.library.audio.MicrophoneHelper;

import com.ibm.watson.developer_cloud.android.library.audio.MicrophoneInputStream;

import com.ibm.watson.developer_cloud.android.library.audio.StreamPlayer;

import com.ibm.watson.developer_cloud.android.library.audio.utils.ContentType;

import com.ibm.watson.assistant.v2.Assistant;

import com.ibm.watson.assistant.v2.model.CreateSessionOptions;

import com.ibm.watson.assistant.v2.model.MessageInput;

import com.ibm.watson.assistant.v2.model.MessageOptions;

import com.ibm.watson.assistant.v2.model.MessageResponse;

import com.ibm.watson.assistant.v2.model.SessionResponse;

import com.ibm.watson.speech_to_text.v1.SpeechToText;

import com.ibm.watson.speech_to_text.v1.model.RecognizeOptions;

import com.ibm.watson.speech_to_text.v1.model.SpeechRecognitionResults;

import com.ibm.watson.speech_to_text.v1.websocket.BaseRecognizeCallback;

import com.ibm.watson.text_to_speech.v1.TextToSpeech;

import com.ibm.watson.text_to_speech.v1.model.SynthesizeOptions;


import java.io.InputStream;

import java.util.ArrayList;
```

```java
public class MainActivity extends AppCompatActivity {


 private RecyclerView recyclerView;

 private ChatAdapter mAdapter;

 private ArrayList messageArrayList;

 private EditText inputMessage;

 private ImageButton btnSend;

 private ImageButton btnRecord;

 StreamPlayer streamPlayer = new StreamPlayer();

 private boolean initialRequest;

 private boolean permissionToRecordAccepted = false;

 private static final int REQUEST_RECORD_AUDIO_PERMISSION = 200;

 private static String TAG = "MainActivity";

 private static final int RECORD_REQUEST_CODE = 101;

 private boolean listening = false;

 private MicrophoneInputStream capture;

 private Context mContext;

 private MicrophoneHelper microphoneHelper;


 private Assistant watsonAssistant;

 private SessionResponse watsonAssistantSession;

 private SpeechToText speechService;

 private TextToSpeech textToSpeech;


 private void createServices() {

   watsonAssistant = new Assistant("2018-11-08", new
IamAuthenticator(mContext.getString(R.string.assistant_apikey)));

   watsonAssistant.setServiceUrl(mContext.getString(R.string.assistant_url));


   textToSpeech = new TextToSpeech(new
IamAuthenticator((mContext.getString(R.string.TTS_apikey))));
```

```
   textToSpeech.setServiceUrl(mContext.getString(R.string.TTS_url));



   speechService = new SpeechToText(new
IamAuthenticator(mContext.getString(R.string.STT_apikey)));

   speechService.setServiceUrl(mContext.getString(R.string.STT_url));

 }



 @Override

 protected void onCreate(Bundle savedInstanceState) {

   super.onCreate(savedInstanceState);

   setContentView(R.layout.activity_main);



   mContext = getApplicationContext();



   inputMessage = findViewById(R.id.message);

   btnSend = findViewById(R.id.btn_send);

   btnRecord = findViewById(R.id.btn_record);

   String customFont = "Montserrat-Regular.ttf";

   Typeface typeface = Typeface.createFromAsset(getAssets(), customFont);

   inputMessage.setTypeface(typeface);

   recyclerView = findViewById(R.id.recycler_view);



   messageArrayList = new ArrayList<>();

   mAdapter = new ChatAdapter(messageArrayList);

   microphoneHelper = new MicrophoneHelper(this);



   LinearLayoutManager layoutManager = new LinearLayoutManager(this);

   layoutManager.setStackFromEnd(true);

   recyclerView.setLayoutManager(layoutManager);

   recyclerView.setItemAnimator(new DefaultItemAnimator());

   recyclerView.setAdapter(mAdapter);
```

```
    this.inputMessage.setText("");
    this.initialRequest = true;


    int permission = ContextCompat.checkSelfPermission(this,
      Manifest.permission.RECORD_AUDIO);


    if (permission != PackageManager.PERMISSION_GRANTED) {
      Log.i(TAG, "Permission to record denied");
      makeRequest();
    } else {
      Log.i(TAG, "Permission to record was already granted");
    }


    recyclerView.addOnItemTouchListener(new
RecyclerTouchListener(getApplicationContext(), recyclerView, new ClickListener() {
      @Override
      public void onClick(View view, final int position) {
        Message audioMessage = (Message) messageArrayList.get(position);
        if (audioMessage != null && !audioMessage.getMessage().isEmpty()) {
          new SayTask().execute(audioMessage.getMessage());
        }
      }


      @Override
      public void onLongClick(View view, int position) {
        recordMessage();


      }
    }));


    btnSend.setOnClickListener(new View.OnClickListener() {
      @Override

      public void onClick(View v) {

        if (checkInternetConnection()) {

          sendMessage();

        }

      }

    });


    btnRecord.setOnClickListener(new View.OnClickListener() {

      @Override

      public void onClick(View v) {

        recordMessage();

      }

    });


    createServices();

    sendMessage();

  }


  ;


  // Speech-to-Text Record Audio permission

  @Override

  public void onRequestPermissionsResult(int requestCode, @NonNull String[]
 permissions, @NonNull int[] grantResults) {

    super.onRequestPermissionsResult(requestCode, permissions, grantResults);

    switch (requestCode) {

      caseREQUEST_RECORD_AUDIO_PERMISSION:

        permissionToRecordAccepted = grantResults[0] ==
 PackageManager.PERMISSION_GRANTED;

        break;

      caseRECORD_REQUEST_CODE: {
        if (grantResults.length == 0
          || grantResults[0] !=
          PackageManager.PERMISSION_GRANTED) {


          Log.i(TAG, "Permission has been denied by user");
        } else {
          Log.i(TAG, "Permission has been granted by user");
        }
        return;
      }


      case MicrophoneHelper.REQUEST_PERMISSION: {
        if (grantResults.length > 0 && grantResults[0] !=
 PackageManager.PERMISSION_GRANTED) {
          Toast.makeText(this, "Permission to record audio denied",
 Toast.LENGTH_SHORT).show();
        }
      }
    }
    // if (!permissionToRecordAccepted ) finish();


  }


  protected void makeRequest() {
    ActivityCompat.requestPermissions(this,
      new String[]{Manifest.permission.RECORD_AUDIO},
      MicrophoneHelper.REQUEST_PERMISSION);
  }


  // Sending a message to Watson Assistant Service
  private void sendMessage() {
    final String inputmessage = this.inputMessage.getText().toString().trim();

    if (!this.initialRequest) {

      Message inputMessage = new Message();

      inputMessage.setMessage(inputmessage);

      inputMessage.setId("1");

      messageArrayList.add(inputMessage);

    } else {

      Message inputMessage = new Message();

      inputMessage.setMessage(inputmessage);

      inputMessage.setId("100");

      this.initialRequest = false;

      Toast.makeText(getApplicationContext(), "Tap on the message for Voice",
 Toast.LENGTH_LONG).show();



    }



    this.inputMessage.setText("");

    mAdapter.notifyDataSetChanged();



    Thread thread = new Thread(new Runnable() {

      public void run() {

        try {

          if (watsonAssistantSession == null) {

            ServiceCall<SessionResponse> call = watsonAssistant.createSession(new
 CreateSessionOptions.Builder().assistantId(mContext.getString(R.string.assistant_id)).
 build());

            watsonAssistantSession = call.execute().getResult();

          }



          MessageInput input = new MessageInput.Builder()

            .text(inputmessage)
```

```
          .build();
        MessageOptions options = new MessageOptions.Builder()
          .assistantId(mContext.getString(R.string.assistant_id))
          .input(input)
          .sessionId(watsonAssistantSession.getSessionId())
          .build();
        MessageResponse response =
watsonAssistant.message(options).execute().getResult();
          Log.i(TAG, "run: "+response);
        final Message outMessage = new Message();
        if (response != null &&
          response.getOutput() != null &&
          !response.getOutput().getGeneric().isEmpty() &&
          "text".equals(response.getOutput().getGeneric().get(0).responseType())) {
          outMessage.setMessage(response.getOutput().getGeneric().get(0).text());
          outMessage.setId("2");

          messageArrayList.add(outMessage);

          // speak the message
          new SayTask().execute(outMessage.getMessage());

          runOnUiThread(new Runnable() {
            public void run() {
              mAdapter.notifyDataSetChanged();
              if (mAdapter.getItemCount() > 1) {
                recyclerView.getLayoutManager().smoothScrollToPosition(recyclerView,
null, mAdapter.getItemCount() - 1);

              }

            }
          });
        }
      } catch (Exception e) {
        e.printStackTrace();
      }
    }
  });

  thread.start();

}

private class SayTask extends AsyncTask<String, Void, String> {
  @Override
  protected String doInBackground(String... params) {
    streamPlayer.playStream(textToSpeech.synthesize(new SynthesizeOptions.Builder()
      .text(params[0])
      .voice(SynthesizeOptions.Voice.EN_US_LISAVOICE)
      .accept(HttpMediaType.AUDIO_WAV)
      .build()).execute().getResult());
    return "Did synthesize";
  }
}

//Record a message via Watson Speech to Text
private void recordMessage() {
  if (listening != true) {
    capture = microphoneHelper.getInputStream(true);
    new Thread(new Runnable() {
      @Override
      public void run() {
         try {

           speechService.recognizeUsingWebSocket(getRecognizeOptions(capture), new
MicrophoneRecognizeDelegate());

         } catch (Exception e) {

           showError(e);

         }

       }

     }).start();

     listening = true;

     Toast.makeText(MainActivity.this, "Listening....Click to Stop",
Toast.LENGTH_LONG).show();



   } else {

     try {

       microphoneHelper.closeInputStream();

       listening = false;

       Toast.makeText(MainActivity.this, "Stopped Listening....Click to Start",
Toast.LENGTH_LONG).show();

     } catch (Exception e) {

       e.printStackTrace();

     }



   }

 }



 /**

  * Check Internet Connection

  *

  * @return

  */

 private boolean checkInternetConnection() {

   // get Connectivity Manager object to check connection

   ConnectivityManager cm =
     (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);


   NetworkInfo activeNetwork = cm.getActiveNetworkInfo();

   boolean isConnected = activeNetwork != null &&

     activeNetwork.isConnectedOrConnecting();


   // Check for network connections

   if (isConnected) {

     return true;

   } else {

     Toast.makeText(this, " No Internet Connection available ",
Toast.LENGTH_LONG).show();

     return false;

   }


 }


 //Private Methods - Speech to Text

 private RecognizeOptions getRecognizeOptions(InputStream audio) {

   return new RecognizeOptions.Builder()

     .audio(audio)

     .contentType(ContentType.OPUS.toString())

     .model("en-US_BroadbandModel")

     .interimResults(true)

     .inactivityTimeout(2000)

     .build();

 }


 //Watson Speech to Text Methods.

 private class MicrophoneRecognizeDelegate extends BaseRecognizeCallback {

   @Override
   public void onTranscription(SpeechRecognitionResults speechResults) {

     if (speechResults.getResults() != null && !speechResults.getResults().isEmpty())
{

       String text =
speechResults.getResults().get(0).getAlternatives().get(0).getTranscript();

       showMicText(text);

     }

   }



   @Override

   public void onError(Exception e) {

     showError(e);

     enableMicButton();

   }



   @Override

   public void onDisconnected() {

     enableMicButton();

   }



 }



 private void showMicText(final String text) {

   runOnUiThread(new Runnable() {

     @Override

     public void run() {

       inputMessage.setText(text);

     }

   });

 }



 private void enableMicButton() {
   runOnUiThread(new Runnable() {
     @Override
     public void run() {
       btnRecord.setEnabled(true);
     }
   });
 }


 private void showError(final Exception e) {
   runOnUiThread(new Runnable() {
     @Override
     public void run() {
       Toast.makeText(MainActivity.this, e.getMessage(), Toast.LENGTH_SHORT).show();
       e.printStackTrace();
                     }
              });
       }
}
```