

PROJECT REPORT

Chatbot (Hope) for Mental Healthcare

- Chatbot that helps and assists users for their mental and health queries



TEAM MEMBERS:

- Aparna Bhat
- Priyanka Singh

Under the guidance of:

Dr. Surendra Sarnikar

1. INTRODUCTION AND PROBLEM DESCRIPTION:

Today in the present era, the major challenge that people are facing after the lockdown from COVID 19 is Loneliness, Anxiety and Mental health disorder. Mental health disorders affected an estimated 792 million people, based on data from 2017 alone. Current figures of those struggling with mental health illnesses amid the COVID-19 pandemic could be exponentially higher. The aim of our project is to design a Conversational AI Chatbot (Hope) using deep learning techniques which would build your knowledge on mental health disorders as well as support your peers with mental illness.

2. DATA DESCRIPTION/DATA PREP

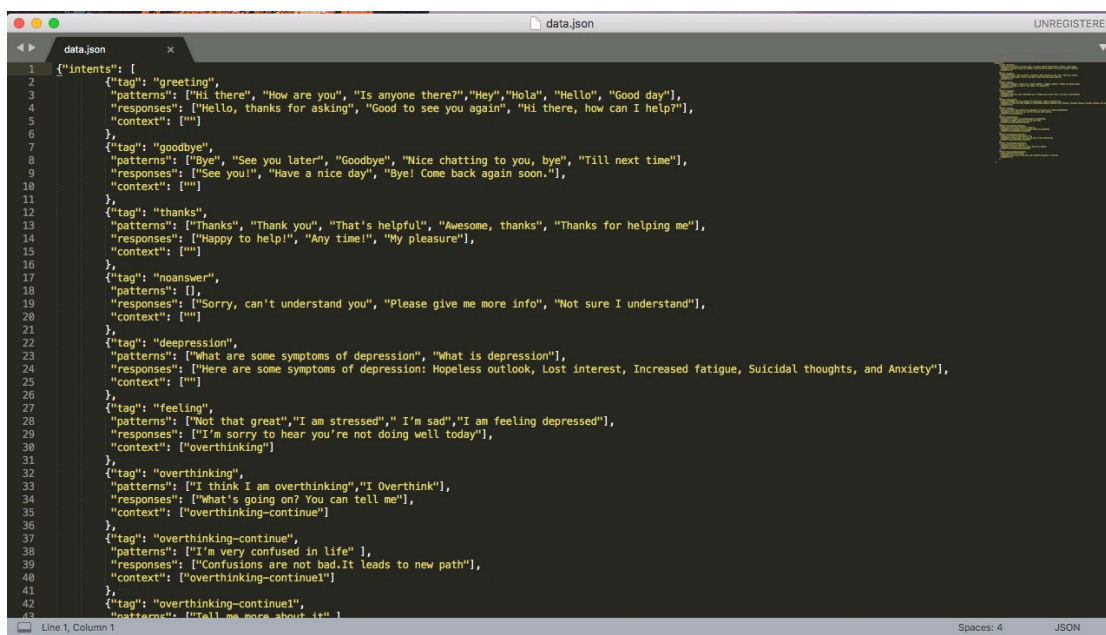
For this project, a dataset has been generated by taking reference of research papers and conversations between a user and a therapist available on the internet. As this area is quite new, there is no major availability of ready datasets for such problems.

For data preparation text preprocessing is required. Text preprocessing is executed to get quality information that can be utilized for acquiring better quality outcomes.

We have taken the "tag" and "patterns" from the data file and stored it in an assortment of extraordinary words in the attempt to make a Bag of Words (BoW) vector. Below are the techniques that are used:

1. Tokenization
2. Lemmatization
3. Vectorization

The data we have used is a json file (JavaScript objects that lists different tags that correspond to different types of word patterns) where we store all of our "intents" which have predefined patterns and responses.



```
1 [{"intents": [
2   {"tag": "greeting",
3     "patterns": ["Hi there", "How are you?", "Is anyone there?", "Hey", "Hola", "Hello", "Good day"],
4     "responses": ["Hello, thanks for asking", "Good to see you again", "Hi there, how can I help?"],
5     "context": [""]
6   },
7   {"tag": "goodbye",
8     "patterns": ["Bye", "See you later", "Goodbye", "Nice chatting to you, bye", "Till next time"],
9     "responses": ["See you!", "Have a nice day", "Bye! Come back again soon."],
10    "context": [""]
11  },
12  {"tag": "thanks",
13    "patterns": ["Thanks", "Thank you", "That's helpful", "Awesome, thanks", "Thanks for helping me"],
14    "responses": ["Happy to help!", "Any time!", "My pleasure"],
15    "context": [""]
16  },
17  {"tag": "noanswer",
18    "patterns": [],
19    "responses": ["Sorry, can't understand you", "Please give me more info", "Not sure I understand"],
20    "context": [""]
21  },
22  {"tag": "depression",
23    "patterns": ["What are some symptoms of depression", "What is depression"],
24    "responses": ["Here are some symptoms of depression: Hopeless outlook, Lost interest, Increased fatigue, Suicidal thoughts, and Anxiety"],
25    "context": [""]
26  },
27  {"tag": "feeling",
28    "patterns": ["Not that great", "I am stressed", "I'm sad", "I am feeling depressed"],
29    "responses": ["I'm sorry to hear you're not doing well today"],
30    "context": ["overthinking"]
31  },
32  {"tag": "overthinking",
33    "patterns": ["I think I am overthinking", "I Overthink"],
34    "responses": ["What's going on? You can tell me"],
35    "context": ["overthinking-continue"]
36  },
37  {"tag": "overthinking-continue",
38    "patterns": ["I'm very confused in life"],
39    "responses": ["Confusions are not bad. It leads to new path"],
40    "context": ["overthinking-continue1"]
41  },
42  {"tag": "overthinking-continue1",
43    "patterns": ["Tell me more about it"]
44  }
45 ]}]
```

3. MODEL DESCRIPTION AND ALTERNATIVE MODEL ANALYSIS

For this project, 3 models are trained and tested. The first model is DNN model, second model is RNN LSTM and the last model is Seq2Seq.

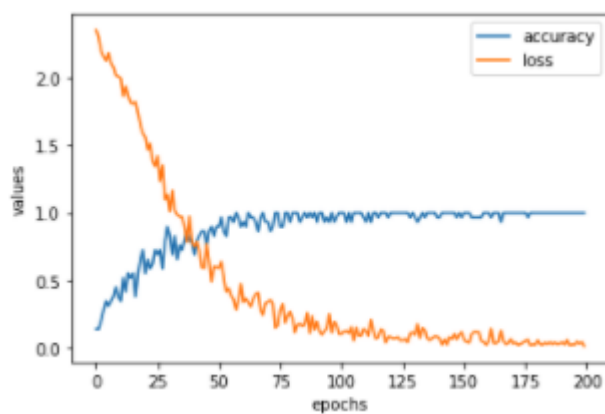
Dense Neural Network:

DNN model has 3 layers with the first layer of 128 neurons, second layer of 64 neurons and third output layer containing the number of neurons equal to the number of intents to predict output intent. The activation function used is 'softmax' as the output is multi class. For model compilation, 'adam' optimizer is used with loss function as 'categorical_crossentropy'. Model performance is measured by 'accuracy' of the model by running it for 200 epochs. The accuracy achieved by this model after 200 epochs is 100% meaning this model is able to correctly predict the output of the test dataset.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 128)	14336
dropout_2 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 64)	8256
dropout_3 (Dropout)	(None, 64)	0
dense_5 (Dense)	(None, 15)	975

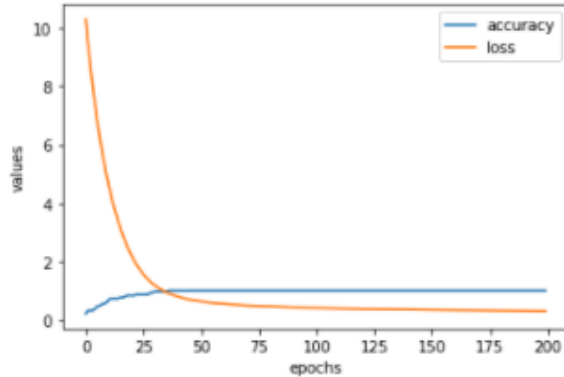
Total params: 23,567
Trainable params: 23,567
Non-trainable params: 0



Above plot shows that the model achieved 100% accuracy after 48 iterations only. Also, the model loss is reduced after every epoch with 0.05 at 200th epoch. Though the model is

performing good, irregularities are there in accuracy and loss after every iteration. To smoothen the learning, L2 Regularization is performed on the model.

DNN Model Regularization:



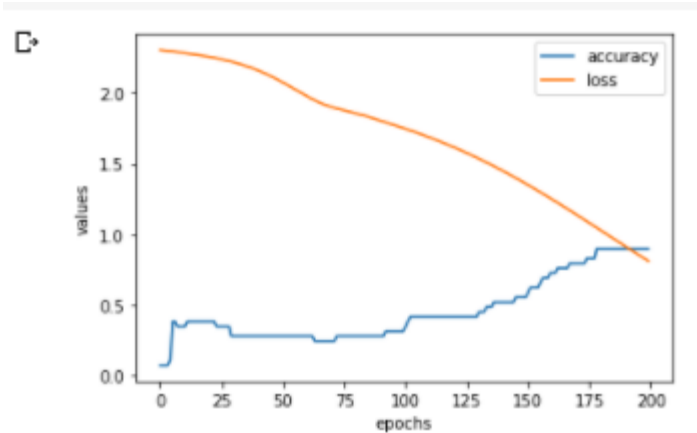
After L2 Regularization and running the model for 200 epochs, the model has improved a lot and shows no spikes. Also 100% accuracy is achieved after 37 iterations only. Also, the model loss is 0.29 at 200th epoch.

RNN LSTM:

The LSTM model is developed with 4 layers with the first layer as the Embedding layer having timesteps equal to 9 and channels equal to 10. For the LSTM layer, the channel size is 10 and timestep is 9. To activate the timestep, return_sequences is kept as True so that it communicates with the next layer. The third layer is the Flatten layer which connects the LSTM layer output with the fourth Dense layer. The number of neurons in the Dense layer is the same as the Output_length. The activation function used is 'softmax'. For model compilation, 'adam' optimizer is used with loss function as 'sparse_categorical_crossentropy'. Model performance is measured by 'accuracy' of the model by running it for 200 epochs. The accuracy achieved by this model after 200 epochs is 97% meaning this model is able to correctly predict the output of the test dataset by 97%.

Model: "model"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 9)]	0
embedding (Embedding)	(None, 9, 10)	900
lstm (LSTM)	(None, 9, 10)	840
flatten (Flatten)	(None, 90)	0
dense (Dense)	(None, 9)	819
Total params: 2,559		
Trainable params: 2,559		
Non-trainable params: 0		



Above plot shows that the model achieved 97% accuracy after 143 iterations and is training slowly after every iteration. Also, the model loss is reduced after every epoch with 0.24 at 200th epoch.

Below is the snapshot of the chat between User and Bot using RNN LSTM model.

```
User:Hi there
Bot: Hello, thanks for asking
User:I am sad
Bot: I'm sorry to hear you're not doing well today
User:Thanks for the help
Bot: Any time!
User:goodbye
Bot: See you!
```

Seq2Seq:

The seq2seq model also called the encoder-decoder model uses Long Short Term Memory- LSTM for text generation from the training corpus. In our rule based chatbot we have used this approach for text generation given in the user input. The encoder outputs a final state vector (memory) which becomes the initial state for the decoder. Teacher forcing method is used to train the decoder which enables it to predict the following words in a target sequence given in the previous words. For target sequences we have added the <START> at the beginning and <END> at the end of the sequence so that our model knows where to start and end text generation. For training the model the hyper parameters we have used are: RMS_PROP as optimizer, categorical cross entropy as loss function. After training finishes the accuracy we got is 96%, and the accuracy fluctuates at some epochs. The reason for that is that we have used only a few pairs of the dataset.

Below is the snapshot of the chat between User and Bot using Seq2Seq model.

```
Enter question : Hi there
WARNING:tensorflow:Model was constructed with shape
hi there how can i help end
Enter question : I am stressed
i'm sorry to hear you're not doing well today end
Enter question : I'm very confused in life
confusions are not bad it leads to new path end
Enter question : 
```

Below is the summary table for the hyperparameters used in both the models.

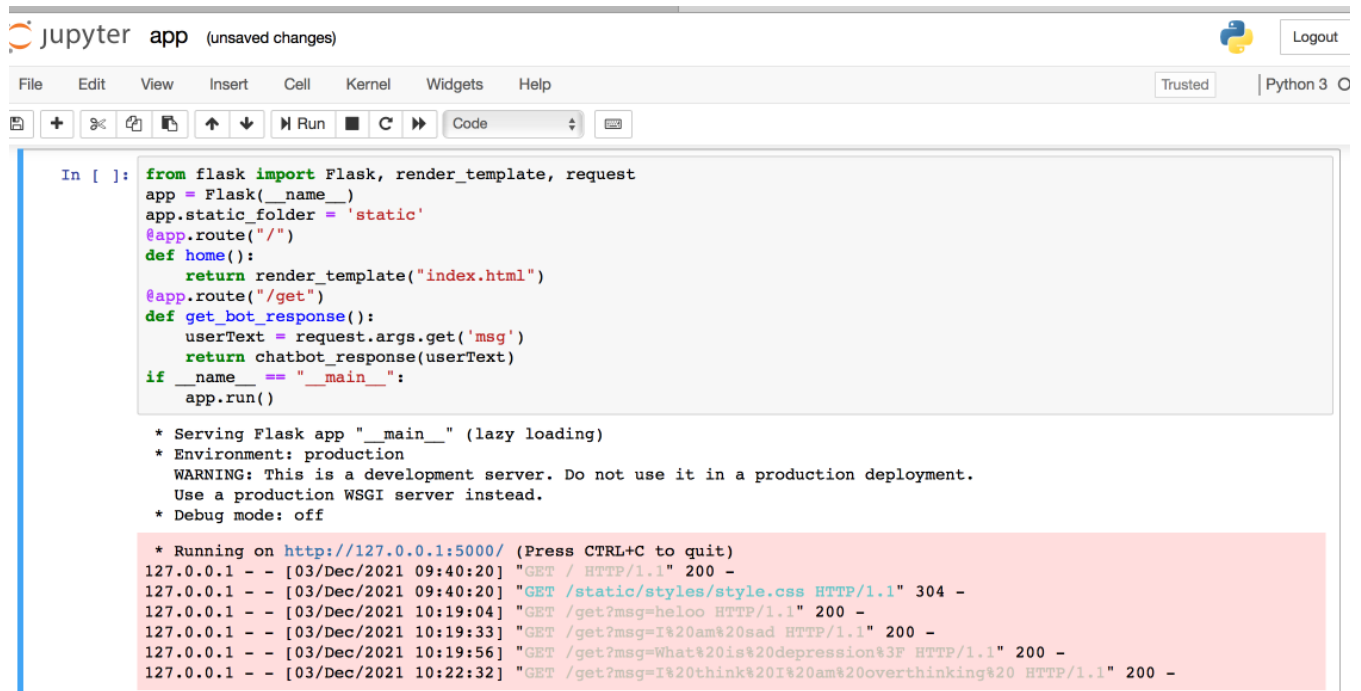
Model	Activation Function	Optimizer	Loss Function	Accuracy	Model Loss
DNN	softmax	adam	categorical_crossentropy	100%	0.04
LSTM	softmax	adam	sparse_categorical_crossentropy	97%	0.24
Seq2Seq	softmax	rmsprop	categorical_crossentropy	96%	0.8

4. MODEL IMPLEMENTATION

Out of all the models we are taking, the DNN model as the best model and moving forward with its integration with flask for front end. For front end, we have created app.py which is calling the DNN model.

For Backend we have created the data.json file based on the research which has some patterns with matching responses. The data is first preprocessed and cleaned by using some NLP techniques. After the data is cleaned the model is trained and that's how the bot sends the response back to the user. Below is the high level architecture of the project.

Below is the screenshot of app.py file

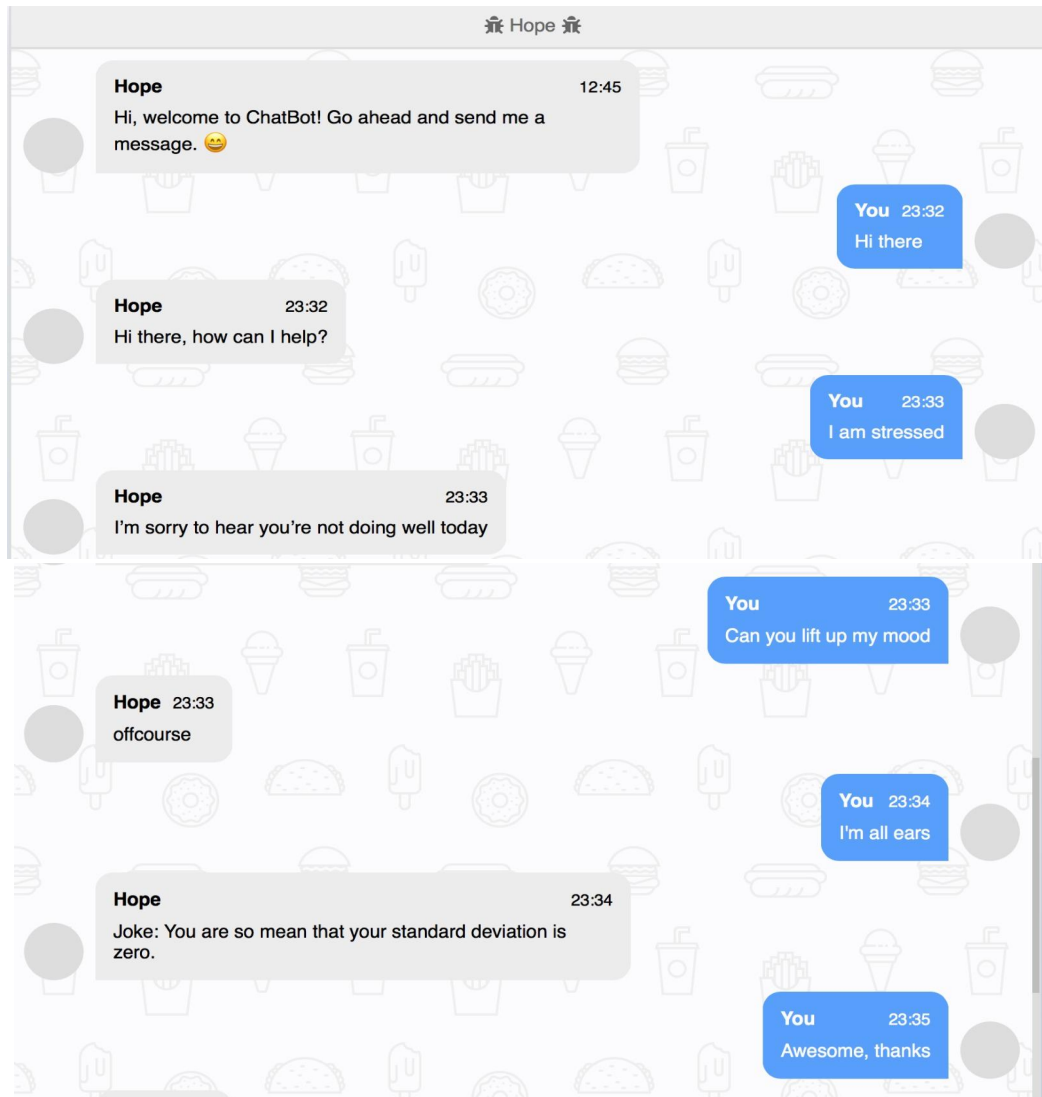


```
In [ ]: from flask import Flask, render_template, request
app = Flask(__name__)
app.static_folder = 'static'
@app.route("/")
def home():
    return render_template("index.html")
@app.route("/get")
def get_bot_response():
    userText = request.args.get('msg')
    return chatbot_response(userText)
if __name__ == "__main__":
    app.run()

* Serving Flask app "__main__" (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off

* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [03/Dec/2021 09:40:20] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [03/Dec/2021 09:40:20] "GET /static/styles/style.css HTTP/1.1" 304 -
127.0.0.1 - - [03/Dec/2021 10:19:04] "GET /get?msg=heloo HTTP/1.1" 200 -
127.0.0.1 - - [03/Dec/2021 10:19:33] "GET /get?msg=I%20am%20sad HTTP/1.1" 200 -
127.0.0.1 - - [03/Dec/2021 10:19:56] "GET /get?msg=What%20is%20depression%3F HTTP/1.1" 200 -
127.0.0.1 - - [03/Dec/2021 10:22:32] "GET /get?msg=I%20think%20I%20am%20overthinking%20 HTTP/1.1" 200 -
```

Here are some snapshots of Hope interacting with the user.



5. DISCUSSION

BUSINESS IMPACT

During the pandemic breakdown, Artificial intelligence has been proven to be an effective tool source or support for people. There are many different ways hope chatbot can positively support users with bad mental state and also drive business growth:

1. Advertisement platform for mental health therapy.
2. Monitor Customer Data to gain insights

A recent study found that companies with mental-health programmes in place for one year had median annual return of investment of \$1.62 for every dollar invested.

This isn't to say that technology will replace human beings. Robots and chatbots can't empathize with us, and they can't even diagnose our mental illnesses. We still need human therapists – today, perhaps more than ever – but technology can supplement the valuable work of human beings. Because even therapists can feel anxiety, and they could use a little help from our robot friends.

POTENTIAL IMPROVEMENTS:

1. Improvement in sentiment analysis for increasing communication efficiency.
2. Integration of voice bot.
3. Integration of therapists connecting to the user based on the location keywords the user provides.
4. Surveys could be conducted for the chatbot questionnaire.

CHALLENGES

1. The biggest challenge in this project was to provide context with meaningful responses.
2. Continuous Chatbot Testing based on the model accuracy.
3. Not enough resources for mental health disorder queries(Patient-Therapist conversations)

REFERENCES

http://timalthoff.com/docs/althoff-2016-mental_health.pdf

<https://feminisinindia.com/2020/05/04/conversations-therapist-mental-health-lockdown/>