

Abstract geometric lines in the top left corner, consisting of several overlapping, irregular polygons and lines in a light beige color.

CHATBOT PROJECT

By Sahaja Segu

Abstract:


In this project, we implement a chatbot that can answer questions based on a "story" given to the bot.

We are using a subset of the BaBi dataset released by Facebook research.

There are 10,000 data in the training set and 1,000 data in the testing set.


Each data in the training/testing set consists of 3 components:

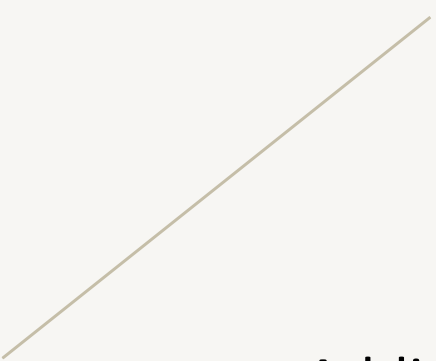
- Story - consists of single or multiple sentences
- Question - single sentence query related to the story
- Answer - "yes" or "no" answer to the question



The model for our chatbot is a RNN network with attention mechanism. It includes the following layers: Embedding, LSTM, Dropout, Dense and Activation. The design of the model pretty much follows the idea in the paper "End-to-End Memory Networks" Our model achieved pretty high accuracy on training/testing set and performs really good on run time generated data.


Chatbots are AI-driven conversational agents that simulate human-like interactions through natural language processing (NLP). This abstract provides an overview of chatbots, including their evolution, applications, and underlying technologies. It highlights the various types of chatbots, discusses ethical considerations, and explores their potential across industries.

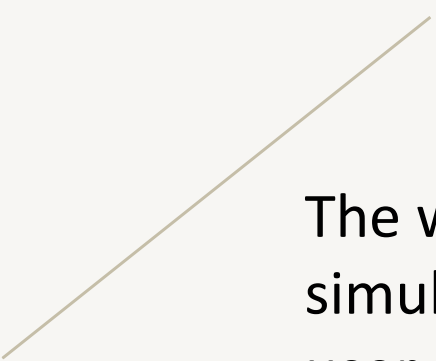




Additionally, it discusses the ethical considerations surrounding chatbot development, such as privacy concerns and bias mitigation. Furthermore, it examines the future potential of chatbots in fields ranging from customer service and healthcare to education and entertainment.


Overall, this abstract provides a comprehensive overview of chatbots, shedding light on their impact on society and their role in shaping the future of human-computer interaction.





The web based platform provides a vast intelligent base that can help simulate problem- solving for humans. This proposed chatbot identifies the user context which triggers the particular intent for a response. Since it is responding dynamic response the desired answer will be generated for the user. The proposed system used machine learning algorithms to learn the Chatbot by experiencing various user's responses and requests

Nowadays chat-bot is started to becoming so robust because Artificial Intelligence aids the human touch in every conversation, chat-bot understand the user's query, and trigger an accurate response. The objective of this project is that chatbots can help to reduce the dependency of an organization on humans and also minimize the need for a different system for different processes.






OBJECTIVE:

A chatbot can communicate with a real person behaving like a human.

Let's list down objectives and purpose of chatbots.


You can create chatbots for any business the same as you recruit a person for any department of your company. Whether you are :

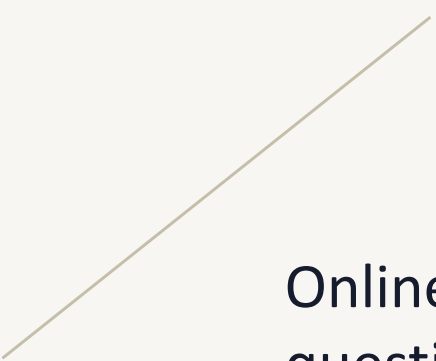
- Insurance Assistant
 - Education Consultant
 - Legal Assistant
 - A real estate business
 - Recruiter
 - Travel Agency
 - Hospital
 - Wedding planner
- 



Chatbots allow businesses to connect with customers in a personal way without the expense of human representatives. For example, many of the questions or issues customers have are common and easily answered. That's why companies create FAQs and troubleshooting guides.


Chatbots provide a personal alternative to a written FAQ or guide and can even triage questions, including handing off a customer issue to a live person if the issue becomes too complex for the chatbot to resolve. Chatbots have become popular as a time and money saver for businesses and an added convenience for customers.





Online chatbots can save time by automatically answering customer questions, providing accurate responses to lead generation prompts, and even engaging prospects on social media for increased engagement.

Chatbot can make online shopping more enjoyable by ensuring. However, from a technological point of view, a chatbot only represents the natural evolution of Question Answering systems leveraging Natural Language Processing. Formulating responses to questions in natural language is one of the most common Examples of Natural Language Processing being applied in various enterprises' end-use applications.







INTRODUCTION:

A chatbot is a program that communicates with you. It is a layer on top of, or a gateway to, a service. Sometimes it is powered by machine learning (the chatbot gets smarter the more you interact with it). Or, more commonly, it is driven using intelligent rules (i.e. if the person says this, respond with that).

The services a chatbot can deliver are diverse. Important life-saving health messages, to check the weather forecast or to purchase a new pair of shoes, and anything else in between.

The term chatbot is synonymous with text conversation but is growing quickly through voice communication... “Alexa, what time is it?” (other voice-chatbots are available!)





The chatbot can talk to you through different channels; such as Facebook Messenger, Siri, WeChat, Telegram, SMS, Slack, Skype and many others. Consumers spend lots of time using messaging applications (more than they spend on social media). Therefore, messaging applications are currently the most popular way companies deliver chatbot experiences to consumers.

Aside from buying shoes, here are a few more examples of companies using chatbots:

- Uber to book a taxi
- KLM to deliver flight information
- CNN to keep you up-to-date with news content
- TechCrunch to keep you up-to-date with techie content
- Pizza Hut to help you order a pizza
- Sephora to provide beauty tips and a shopping experience
- Bank of America to connect customers and their finances

The possibilities are (almost) limitless.








METHODOLOGY:

The present Methodology provides an entry-level knowledge about the chatbot technologies and how they could be used in adult education, with a focus on online and blended learning environments. This way, we want to equip educators and training professionals with general theoretical knowledge about the specifics of applying such digital tools in the educational process and how to incorporate them into classrooms.

The Methodology consists of three parts. The first one is dedicated to the basic terms and definitions, purposes and fields of use of chatbot technologies. The second part is focused on chatbot-based learning and how to incorporate chatbot technologies into the educational process and for self-learning.





The last part includes links to additional resources and references on the topic. Although some authors distinguish ‘chatbots’ and ‘bots’, as the first ones are based on text-message interaction with users, while latter might include voice or even video-based communication and inclusion of artificial intelligence, for the purposes of the present Methodology, we will use these two terms as interchangeable.

The layers of the model looks like:

1.Input:

- A. input_sequence (fits on input_train): shape = (batch_size, max_story_len))
- B. question (fits on queries_train: shape = (batch_size, max_query_len))

2.Embedding:

- A. embedding for input_encoder_m: take input from input_sequence. Output shape: (batch_size, story_maxlen, embedding_dim)
- B. embedding for input_encoder_c: take input from input_sequence. Output shape: (batch_size, story_maxlen, max_question_len)
- C. embedding for question_encoder: take input from question. Output shape: (batch_size, query_maxlen, embedding_dim)

3.Layers after Embedding

- A. dot: dot input_encoded_m and question_encoded (output from 2A and 2C) along axes of embedding dimension. Output shape: (batch_size, story_maxlen, query_maxlen)
- B. add: add output from 3A and 2B. Output shape: (batch_size, story_maxlen, query_maxlen)
- C. Permute: permute 2nd and 3rd axes from output of 3B. Output shape: (batch_size, query_maxlen, story_maxlen)
- D. concatenate: concatenate output from 3C and 2B. Output shape: (batch_size, query_maxlen, story_maxlen+embedding_dim)

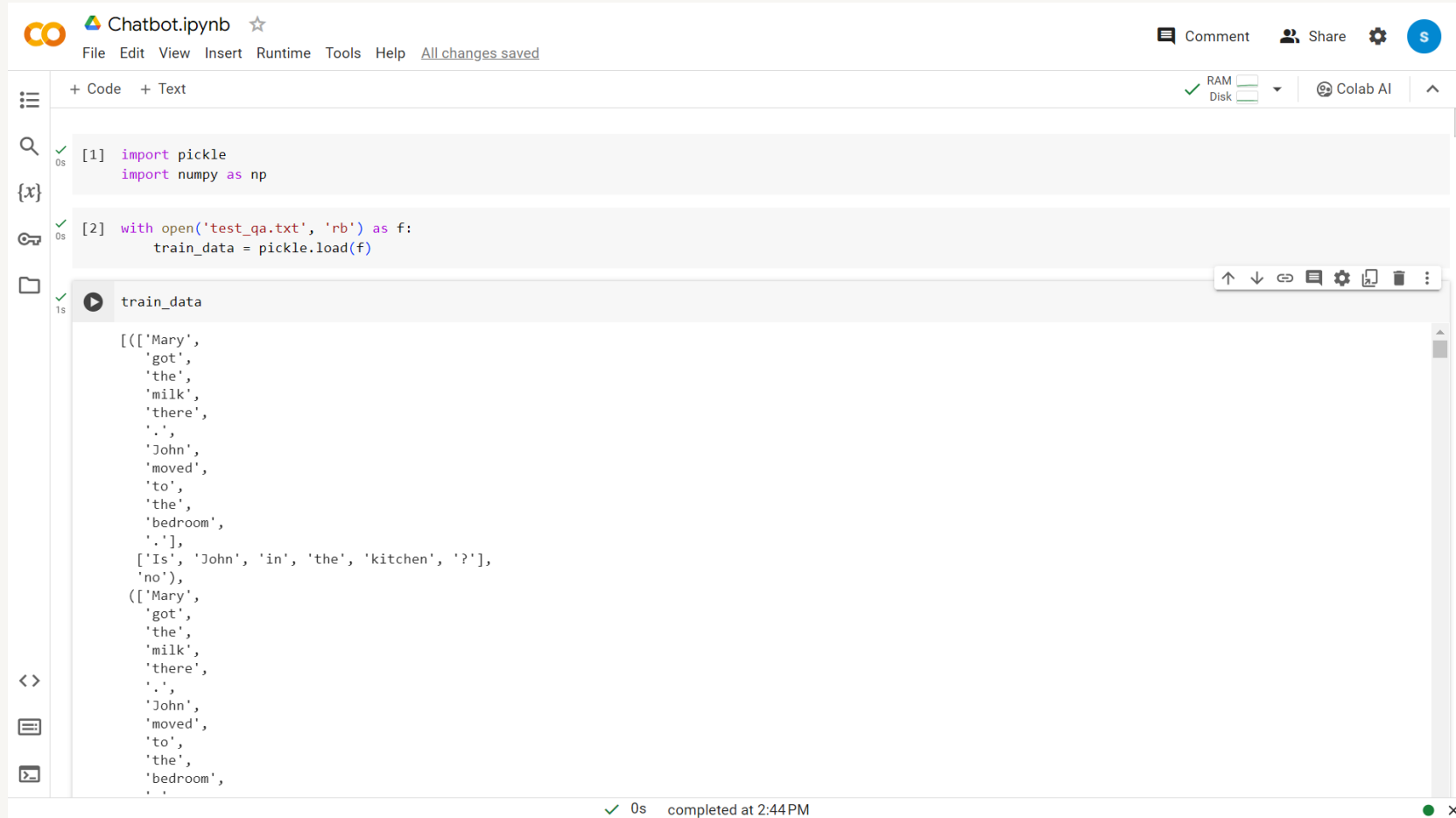
4.LSTM (hidden_unit = 32)

5.Dropout (rate = 0.3)

6.Dense (output = vocab_size)

7.Activation (activation function = 'softmax')

CODE:



The screenshot shows a Google Colab notebook interface. The top bar includes the Colab logo, the notebook title "Chatbot.ipynb", and a star icon. Below this is a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help", followed by a link "All changes saved". On the right side of the top bar are icons for "Comment", "Share", "Settings", and a user profile icon labeled "S".

The notebook contains two code cells. The first cell, labeled "[1]", contains the following code:

```
[1] import pickle
import numpy as np
```

The second cell, labeled "[2]", contains the following code:

```
[2] with open('test_qa.txt', 'rb') as f:
    train_data = pickle.load(f)
```

Below the code cells is a variable inspector showing the variable "train_data". It has a play button icon and a "1s" execution time. The output of the variable is a list of lists of words:

```
[(['Mary',
  'got',
  'the',
  'milk',
  'there',
  '.',
  'John',
  'moved',
  'to',
  'the',
  'bedroom',
  '.'],
 ['Is', 'John', 'in', 'the', 'kitchen', '?'],
 'no'),
 (['Mary',
  'got',
  'the',
  'milk',
  'there',
  '.',
  'John',
  'moved',
  'to',
  'the',
  'bedroom',
  '.'],
 ['Is', 'John', 'in', 'the', 'kitchen', '?'],
 'no')]
```

At the bottom of the notebook, there is a status bar showing a green checkmark, "0s", and "completed at 2:44 PM".

Chatbot.ipynb

CommentShareColab AI

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAMDisk

Colab AI

↑ ↓ ↺ ⚙ 📄 🗑 ⋮

1s

✓

```
'Mary',
'discarded',
'the',
'milk',
',',
'John',
'went',
'to',
'the',
'garden',
',',
['Is', 'John', 'in', 'the', 'kitchen', '?'],
'no'),
(['Mary',
'got',
'the',
'milk',
'there',
',',
'John',
'moved',
'to',
'the',
'bedroom',
',',
'Mary',
'discarded',
'the',
'milk',
',',
'John',
'went',
```

0s

✓

```
[4] with open('test_qa.txt', 'rb') as f:
    test_data = pickle.load(f)
```

✓ 0s

completed at 2:44PM

Chatbot.ipynb

CommentShareColab AI

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAMDisk

Colab AI

↑ ↓ ↺ ⚙ 📄 🗑 ⋮

0s

✓

```
[5] train_data[0]
```

0s

✓

```
(['Mary',
'got',
'the',
'milk',
'there',
',',
'John',
'moved',
'to',
'the',
'bedroom',
',',
['Is', 'John', 'in', 'the', 'kitchen', '?'],
'no')
```

0s

✓

```
[6] print(type(train_data))
    print(type(test_data))
```

0s

✓

```
<class 'list'>
<class 'list'>
```

0s

✓

```
[7] print("Length of the train data: ", len(train_data))
    print("Length of the test data: ", len(test_data))
```

0s

✓

```
Length of the train data: 1000
Length of the test data: 1000
```

0s

✓

```
[8] ' '.join(train_data[0][0])
```

0s

✓

```
'Mary got the milk there . John moved to the bedroom .'
```

✓ 0s

completed at 2:44PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

```
[9] ' '.join(train_data[0][1])

'Is John in the kitchen ?'

[10] train_data[0][2]

'no'

[11] all_data = test_data + train_data

[12] len(all_data)

2000

all_data

['journeyed',
 'to',
 'the',
 'bathroom',
 '.',
 'John',
 'went',
 'back',
 'to',
 'the',
 'garden',
 '.',
 'Sandra',
 'went',
 'back',
 'to',
 'the',
```

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

```
[13] ['Daniel',
      'journeyed',
      'to',
      'the',
      'hallway',
      '.',
      'Daniel',
      'went',
      'to',
      'the',
      'kitchen',
      '.',
      'Sandra',
      'went',
      'to',
      'the',
      'office',
      '.',
      'Sandra',
      'journeyed',
      'to',
      'the',
      'bathroom',
      '.',
      'Daniel',
      'moved',
      'to',
      'the',
      'bedroom',
      '.',
      'Daniel',
      'drowned']

[14] type(all_data)

list
```

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Colab AI

+ Code + Text

RAM Disk Colab AI

[15] set(train_data[0][0])

{'.', 'John', 'Mary', 'bedroom', 'got', 'milk', 'moved', 'the', 'there', 'to'}

[16] vocab = set()

for story, question, answer in all_data:
vocab = vocab.union(set(story))
vocab = vocab.union(set(question))

[17] vocab.add('no')

[18] vocab.add('yes')

vocab

{'.',
'?',
'Daniel',
'Is',
'John',
'Mary',
'Sandra',
'apple',
'back',
'bathroom',
'bedroom',
'discarded',
'down',
'dropped',
'football',
'garden',
'got',
'grabbed',
'hallway',
'in',
'journeyed',
'kitchen',
'left',
'milk',
'moved',
'no',
'office',
'picked',
'put',
'the',
'there',
'to',
'took',
'travelled',
'up',
'went',
'yes'}

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Colab AI

+ Code + Text

RAM Disk Colab AI

[19]

'garden',
'got',
'grabbed',
'hallway',
'in',
'journeyed',
'kitchen',
'left',
'milk',
'moved',
'no',
'office',
'picked',
'put',
'the',
'there',
'to',
'took',
'travelled',
'up',
'went',
'yes'

[20] vocab_len = len(vocab) + 1

[21] all_story_len = [len(data[0]) for data in all_data]
max_story_len = max(all_story_len)
max_question_len = max([len(data[1]) for data in all_data])

max_story_len

86

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

✓ RAM Disk Colab AI

[23] max_question_len

6

[24] from keras.preprocessing.sequence import pad_sequences
from keras.preprocessing.text import Tokenizer

[25] tokenizer = Tokenizer(filters=[])
tokenizer.fit_on_texts(vocab)

tokenizer.word_index

{'down': 1,
'discarded': 2,
'garden': 3,
'office': 4,
'?': 5,
'hallway': 6,
'bedroom': 7,
'dropped': 8,
'got': 9,
'moved': 10,
'daniel': 11,
'travelled': 12,
'put': 13,
'john': 14,
'back': 15,
'no': 16,
'up': 17,
'grabbed': 18,
'left': 19,
'sandra': 20,
'football': 21,
'in': 22}

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

✓ RAM Disk Colab AI

[26] 'travelled': 12,
'put': 13,
'john': 14,
'back': 15,
'no': 16,
'up': 17,
'grabbed': 18,
'left': 19,
'sandra': 20,
'football': 21,
'in': 22,
'picked': 23,
'yes': 24,
'.'': 25,
'there': 26,
'milk': 27,
'apple': 28,
'kitchen': 29,
'went': 30,
'journeyed': 31,
'took': 32,
'bathroom': 33,
'to': 34,
'is': 35,
'the': 36,
'mary': 37)

train_story_text = []
train_question_text = []
train_answers = []

for story, question, answer in train_data:
train_story_text.append(story)
train_question_text.append(question)
train_answers.append(answer)

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings

+ Code + Text

RAM Disk Colab AI

train_story_text[:2]

[['Mary',
'got',
'the',
'milk',
'there',
'.',
'John',
'moved',
'to',
'the',
'bedroom',
'.',
'Mary',
'got',
'the',
'milk',
'there',
'.',
'John',
'moved',
'to',
'the',
'bedroom',
'.',
'Mary',
'discarded',
'the',
'milk',
'.',
'John',
'went',
'to',
'the',
'garden',
'.',
'']]]

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings

+ Code + Text

RAM Disk Colab AI

[29] train_story_seq = tokenizer.texts_to_sequences(train_story_text)

[30] print(len(train_story_seq))
print(len(train_story_text))

1000
1000

train_story_text[:2]

[['Mary',
'got',
'the',
'milk',
'there',
'.',
'John',
'moved',
'to',
'the',
'bedroom',
'.',
'Mary',
'got',
'the',
'milk',
'there',
'.',
'John',
'moved',
'to',
'the',
'bedroom',
'.',
'

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

✓ [31]

```
'the',  
'bedroom',  
'.',  
'Mary',  
'discarded',  
'the',  
'milk',  
'.',  
'John',  
'went',  
'to',  
'the',  
'garden',  
'.']]
```

✓ train_story_seq[:2]

```
[[37, 9, 36, 27, 26, 25, 14, 10, 34, 36, 7, 25],  
 [37,  
  9,  
  36,  
  27,  
  26,  
  25,  
  14,  
  10,  
  34,  
  36,  
  7,  
  25,  
  37,  
  2,  
  36,  
  27,  
  25,  
  14,  
  30,
```

✓ 0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

✓ [32]

```
37,  
2,  
36,  
27,  
25,  
14,  
30,  
34,  
36,  
3,  
25]]
```

✓ [33]

```
def vectorize_stories(data, word_index=tokenizer.word_index, max_story_len=max_story_len, max_question_len=max_question_len):  
    X = []  
    Xq = []  
    Y = []  
    for story, query, answer in data:  
        x = [word_index[word.lower()] for word in story]  
        xq = [word_index[word.lower()] for word in query]  
  
        y = np.zeros(len(word_index)+1)  
        y[word_index[answer]] = 1  
  
        X.append(x) # X holds list of lists of word indices for stories.  
        Xq.append(xq) # Xq holds list of lists for word indices for questions.  
        Y.append(y) # Y holds lists of lists of (38) binary numbers, only 1 of them is 1.  
  
    return (pad_sequences(X, maxlen=max_story_len), pad_sequences(Xq, maxlen=max_question_len), np.array(Y))
```

✓ inputs_train, queries_train, answers_train = vectorize_stories(train_data)

✓ [35] inputs_test, queries_test, answers_test = vectorize_stories(test_data)

✓ 0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings

RAM Disk Colab AI

+ Code + Text

[35] inputs_test, queries_test, answers_test = vectorize_stories(test_data)

[36] inputs_test

array([[0, 0, 0, ..., 36, 7, 25],
[0, 0, 0, ..., 36, 3, 25],
[0, 0, 0, ..., 36, 3, 25],
...,
[0, 0, 0, ..., 36, 28, 25],
[0, 0, 0, ..., 36, 3, 25],
[0, 0, 0, ..., 28, 26, 25]], dtype=int32)

[37] queries_test

array([[35, 14, 22, 36, 29, 5],
[35, 14, 22, 36, 29, 5],
[35, 14, 22, 36, 3, 5],
...,
[35, 37, 22, 36, 7, 5],
[35, 20, 22, 36, 3, 5],
[35, 37, 22, 36, 3, 5]], dtype=int32)

answers_test

array([[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.],
...,
[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.]])

[39] sum(answers_test)

0s completed at 2:44PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings

RAM Disk Colab AI

+ Code + Text

[38] answers_test

array([[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.],
...,
[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.],
[0., 0., 0., ..., 0., 0., 0.]])

[39] sum(answers_test)

array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0., 503., 0., 0., 0., 0., 0.,
0., 0., 497., 0., 0., 0., 0., 0., 0., 0., 0.,
0., 0., 0., 0., 0.])

[40] tokenizer.word_index['yes']

24

[41] tokenizer.word_index['no']

16

from keras.models import Sequential, Model
from keras.layers import Embedding
from keras.layers import Input, Activation, Dense, Permute, Dropout, add, dot, concatenate, LSTM

[43] input_sequence = Input((max_story_len,))
question = Input((max_question_len,))

0s completed at 2:44PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

[44] vocab_size = len(vocab) + 1

[45] input_encoder_m = Sequential()
input_encoder_m.add(Embedding(input_dim=vocab_size, output_dim=64))
input_encoder_m.add(Dropout(0.3))

[46] input_encoder_c = Sequential()
input_encoder_c.add(Embedding(input_dim=vocab_size, output_dim=max_question_len))
input_encoder_c.add(Dropout(0.3))

[47] question_encoder = Sequential()
question_encoder.add(Embedding(input_dim=vocab_size, output_dim=64, input_length=max_question_len))
question_encoder.add(Dropout(0.3))

[48] input_encoded_m = input_encoder_m(input_sequence)
input_encoded_c = input_encoder_c(input_sequence)
question_encoded = question_encoder(question)

print(input_encoded_m.shape)
print(question_encoded.shape)

(None, 86, 64)
(None, 6, 64)

match = dot([input_encoded_m, question_encoded], axes=(2,2))
match = Activation('softmax')(match)

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

[51] response = add([match, input_encoded_c])
response = Permute((2,1))(response)

[52] answer = concatenate([response, question_encoded])

[53] answer

<KerasTensor: shape=(None, 6, 150) dtype=float32 (created by layer 'concatenate')>

[54] answer = LSTM(32)(answer)

[55] print(answer.shape)

(None, 32)

[56] answer = Dropout(0.5)(answer)

[57] answer = Dense(vocab_size)(answer)

[58] answer = Activation('softmax')(answer)

answer

<KerasTensor: shape=(None, 38) dtype=float32 (created by layer 'activation_1')>

[60] model = Model([input_sequence, question], answer)

0s completed at 2:44 PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

05

[51] response = add([match, input_encoded_c])
response = Permute((2,1))(response)

{x}

05

[52] answer = concatenate([response, question_encoded])

05

[53] answer

<KerasTensor: shape=(None, 6, 150) dtype=float32 (created by layer 'concatenate')>

05

[54] answer = LSTM(32)(answer)

05

[55] print(answer.shape)

(None, 32)

05

[56] answer = Dropout(0.5)(answer)

05

[57] answer = Dense(vocab_size)(answer)

05

[58] answer = Activation('softmax')(answer)

<>

05

answer

<KerasTensor: shape=(None, 38) dtype=float32 (created by layer 'activation_1')>

05

[60] model = Model([input_sequence, question], answer)

0s completed at 2:44PM

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

05

permute (Permute) (None, 6, 86) 0 ['add[0][0]']

05

concatenate (Concatenate) (None, 6, 150) 0 ['permute[0][0]',
'sequential_2[0][0]']

{x}

05

lstm (LSTM) (None, 32) 23424 ['concatenate[0][0]']

05

dropout_3 (Dropout) (None, 32) 0 ['lstm[0][0]']

05

dense (Dense) (None, 38) 1254 ['dropout_3[0][0]']

05

activation_1 (Activation) (None, 38) 0 ['dense[0][0]']

05

=====
Total params: 29770 (116.29 KB)
Trainable params: 29770 (116.29 KB)
Non-trainable params: 0 (0.00 Byte)
=====

109

[64] history = model.fit([inputs_train, queries_train], answers_train, batch_size=32, epochs=15, validation_data=([inputs_test, queries_test], answers_test))

<>

109

Epoch 1/15
32/32 [=====] - 1s 16ms/step - loss: 0.6786 - accuracy: 0.5750 - val_loss: 0.6662 - val_accuracy: 0.6250
Epoch 2/15
32/32 [=====] - 0s 14ms/step - loss: 0.6783 - accuracy: 0.5570 - val_loss: 0.6668 - val_accuracy: 0.6010
Epoch 3/15
32/32 [=====] - 0s 14ms/step - loss: 0.6775 - accuracy: 0.5850 - val_loss: 0.6654 - val_accuracy: 0.6190
Epoch 4/15
32/32 [=====] - 0s 13ms/step - loss: 0.6748 - accuracy: 0.5670 - val_loss: 0.6628 - val_accuracy: 0.6290
Epoch 5/15
32/32 [=====] - 0s 14ms/step - loss: 0.6738 - accuracy: 0.5710 - val_loss: 0.6598 - val_accuracy: 0.6360
Epoch 6/15
32/32 [=====] - 0s 13ms/step - loss: 0.6778 - accuracy: 0.5530 - val_loss: 0.6605 - val_accuracy: 0.6390
Epoch 7/15
32/32 [=====] - 0s 13ms/step - loss: 0.6699 - accuracy: 0.5830 - val_loss: 0.6629 - val_accuracy: 0.6070
Epoch 8/15

30°C Mostly sunny

Search

ENG IN 14:50 03-02-2024

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

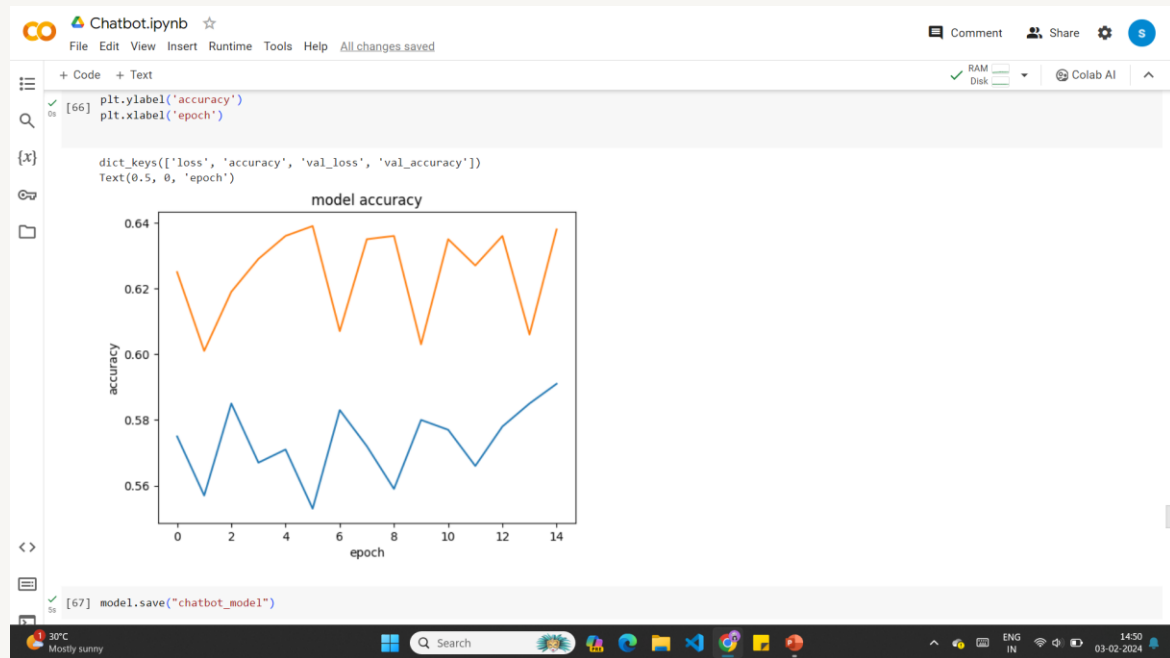
+ Code + Text

RAM Disk Colab AI

```
[64] 32/32 [=====] - 0s 14ms/step - loss: 0.6775 - accuracy: 0.5850 - val_loss: 0.6654 - val_accuracy: 0.6190
Epoch 4/15
32/32 [=====] - 0s 13ms/step - loss: 0.6748 - accuracy: 0.5670 - val_loss: 0.6628 - val_accuracy: 0.6290
Epoch 5/15
32/32 [=====] - 0s 14ms/step - loss: 0.6738 - accuracy: 0.5710 - val_loss: 0.6598 - val_accuracy: 0.6360
Epoch 6/15
32/32 [=====] - 0s 13ms/step - loss: 0.6778 - accuracy: 0.5530 - val_loss: 0.6605 - val_accuracy: 0.6390
Epoch 7/15
32/32 [=====] - 0s 13ms/step - loss: 0.6699 - accuracy: 0.5830 - val_loss: 0.6629 - val_accuracy: 0.6070
Epoch 8/15
32/32 [=====] - 0s 12ms/step - loss: 0.6755 - accuracy: 0.5720 - val_loss: 0.6576 - val_accuracy: 0.6350
Epoch 9/15
32/32 [=====] - 0s 14ms/step - loss: 0.6712 - accuracy: 0.5590 - val_loss: 0.6571 - val_accuracy: 0.6360
Epoch 10/15
32/32 [=====] - 0s 12ms/step - loss: 0.6705 - accuracy: 0.5800 - val_loss: 0.6608 - val_accuracy: 0.6030
Epoch 11/15
32/32 [=====] - 0s 13ms/step - loss: 0.6700 - accuracy: 0.5770 - val_loss: 0.6534 - val_accuracy: 0.6350
Epoch 12/15
32/32 [=====] - 1s 21ms/step - loss: 0.6732 - accuracy: 0.5660 - val_loss: 0.6534 - val_accuracy: 0.6270
Epoch 13/15
32/32 [=====] - 1s 22ms/step - loss: 0.6685 - accuracy: 0.5780 - val_loss: 0.6526 - val_accuracy: 0.6360
Epoch 14/15
32/32 [=====] - 1s 22ms/step - loss: 0.6615 - accuracy: 0.5850 - val_loss: 0.6577 - val_accuracy: 0.6060
Epoch 15/15
32/32 [=====] - 1s 16ms/step - loss: 0.6595 - accuracy: 0.5910 - val_loss: 0.6469 - val_accuracy: 0.6380
```

```
[66] import matplotlib.pyplot as plt
print(history.history.keys())
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
```

30°C Mostly sunny 14:50 03-02-2024



Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

[67] model.save('chatbot_model')

[68] model.load_weights('chatbot_model')

[69] pred_results = model.predict([inputs_test, queries_test])

[70] pred_results.shape

[71] test_data[0][0]

[72] story = '.join(word for word in test_data[0][0])

<tensorflow.python.checkpoint.checkpoint.CheckpointLoadStatus at 0x7b00623a4250>

32/32 [=====] - 1s 3ms/step

(1000, 38)

['Mary',
'got',
'the',
'milk',
'there',
'.',
'John',
'moved',
'to',
'the',
'bedroom',
'.']

story

30°C Mostly sunny

Search

14:50 03-02-2024

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

bedroom ,
'.'

[72] story = '.join(word for word in test_data[0][0])

[73] query = '.join(word for word in test_data[0][1])

[74] query

[75] test_data[0][2]

val_max = np.argmax(pred_results[0])

for key, val in tokenizer.word_index.items():
 if val == val_max:
 k = key

print("predicted ans is",k)
print("probability of certainty",pred_results[0][val_max])

predicted ans is no
probability of certainty 0.54654175

'Mary got the milk there . John moved to the bedroom .'

'Is John in the kitchen ?'

'no'

predicted ans is no
probability of certainty 0.54654175

30°C Mostly sunny

Search

14:51 03-02-2024

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

[78] vocab

```
{x} ['.', '?', 'Daniel', 'Is', 'John', 'Mary', 'Sandra', 'apple', 'back', 'bathroom', 'bedroom', 'discarded', 'down', 'dropped', 'football', 'garden', 'got', 'grabbed', 'hallway', 'in', 'journeyed', 'kitchen', 'left', 'milk', 'moved', 'no', 'office', 'picked', 'put', 'the', 'there', 'to', 'took']
```

30°C Mostly sunny 14:51 03-02-2024

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk Colab AI

[78] ['milk', 'moved', 'no', 'office', 'picked', 'put', 'the', 'there', 'to', 'took', 'travelled', 'up', 'went', 'yes']

[79] story = "mary dropped the football . Sandra discarded apple in kitchen"
story.split()

```
['mary', 'dropped', 'the', 'football', '.', 'Sandra', 'discarded', 'apple', 'in', 'kitchen']
```

[80] my_question = "Is sandra in the kitchen ?"

[81] my_question.split()

30°C Mostly sunny 14:51 03-02-2024

Chatbot.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Colab AI

+ Code + Text

RAM Disk Colab AI

[81] my_question.split()

['Is', 'sandra', 'in', 'the', 'kitchen', '?']

[82] mydata = [(story.split(), my_question.split(), 'yes')]

[83] my_story, my_ques, my_ans = vectorize_stories(mydata)

[84] pred_results = model.predict((my_story, my_ques))

1/1 [=====] - 0s 50ms/step

[86] val_max = np.argmax(pred_results[0])

for key, val in tokenizer.word_index.items():

if val == val_max:

k = key

print("predicted ans is",k)

print("probability of certainty",pred_results[0][val_max])

predicted ans is yes

probability of certainty 0.5499666

[88] pred_results[0][val_max]

0.5499666

30°C Mostly sunny

Search


ENG IN 14:51 03-02-2024

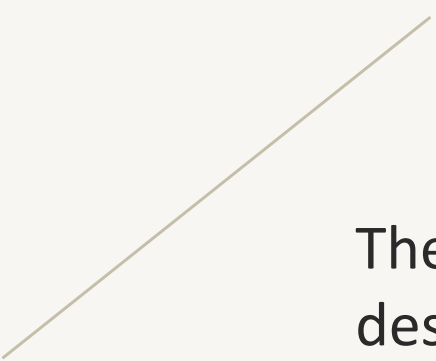


CONCLUSION:

A chatbot is one of the simple ways to transport data from a computer without having to think for proper keywords to look up in a search or browse several web pages to collect information; users can easily type their query in natural language and retrieve information. In this paper, information about the design, implementation of the chatbot has been presented.


From the survey above, it can be said that the development and improvement of chatbot design grow at an unpredictable rate due to variety of methods and approaches used to design a chatbot. Chatbot is a great tool for quick interaction with the user. They help us by providing entertainment, saving time and answering the questions that are hard to find.





The Chatbot must be simple and conversational. Since there are many designs and approaches for creating a chatbot, it can be at odds with commercial considerations. Researchers need to interact and must agree on a common approach for designing a Chatbot.

In this project, we looked into how Chatbots are developed and the applications of Chatbots in various fields. In addition comparison has been made with other Chatbots. General purpose Chatbot must be simple, user friendly, must be easily understood and the knowledge base must be compact. Although some of the commercial products have recently emerged, improvements must be made to find a common approach for designing a Chatbot.





THANK YOU