Membangun Chatbox dengan Python

**Pengenalan**

Chatbot adalah sebuah program komputer berbasis AI (Artificial Intelligence), alias robot virtual yang dapat mensimulasikan percakapan layaknya manusia. Teknologi ini juga dikenal sebagai asisten digital yang dapat memahami serta memproses permintaan pengguna, dan memberikan jawaban yang relevan dengan cepat. Chat bot menggunakan teknologi kecerdasan buatan dan bahasa pemrograman untuk memproses masukan dan responnya. Dalam proses pembuatan chatbot, tentu ada teknologi pendukung. Beberapa diantaranya adalah Machine Learning, Natural Language Processing (NLP) dan Natural Language Understanding (NLU) serta Machine Reading Comprehension (MRC).

Dalam mengimplementasikan chatbot, kita akan menggunakan Keras, yaitu library Deep Learning. NLTK (Natural Language Processing Toolkit) dan beberapa library lainnya.

**Persiapan**

Untuk mengimplementasikan chatbot, dibutuhkan Keras yaitu library Deep Learning. Untuk mempersiapkan librarinya di bawah ini adalah command untuk menginstall libari yang dibutuhkan :

*Pip install tensorflow, keras, pickle, nltk*

Struktur dari project ini adalah:

1. Train.chatbot.py. File ini berguna untuk membangun dan melatih/menjalankan deep learning model yang dapat menklasifikasi dan indentifikasi apa pengguna bertanya ke bot.
2. Gui\_chatbot.py. Difile ini berguna untuk membangun graphical user inteface untuk berinteraksi dengan chatbot.
3. Intents.json. Yang berisi seluruh data yang akan digunakan untuk train/menjalanka model yang ada di train\_chatbot.py. Berisi kumpulan tag dengan pola dan tanggapan yang sesuai.

**Membuat Chatbot**

Berikut langkah-langkah dalam pembuat chatbot :

1. Import librari Buat file python baru dan beri nama sebagai train\_chatbot dan kemudian kita akan mengimport semua modul yang diperlukan. Setelah itu, kita akan membaca file data JSON di program Python

import numpy as np

from keras.models import Sequential

from keras.layers import Dense, Activation, Dropout

from keras.optimizers import SGD

import random

import nltk

from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

import json

import pickle

intents\_file = open('intents.json').read()

intents = json.loads(intents\_file)

1. Pra-pemrosesan Data, Model tidak dapat mengambil data mentah. Harus melalui banyak proses agar mudah dipahami oleh mesin. Untuk data tekstual, ada banyak teknik preprocessing yang tersedia. Teknik pertama adalah tokenizing, di mana kita memecah kalimat menjadi kata-kata. Dengan mengamati file maksud, kita dapat melihat bahwa setiap tag berisi daftar pola dan respons. Kami menandai setiap pola dan menambahkan kata-kata dalam daftar

words=[]

classes = []

documents = []

ignore\_letters = ['!', '?', ',', '.']

for intent in intents['intents']:

for pattern in intent['patterns']:

#tokenize each word

word = nltk.word\_tokenize(pattern)

words.extend(word)

#add documents in the corpus

documents.append((word, intent['tag']))

# add to our classes list

if intent['tag'] not in classes:

classes.append(intent['tag'])

print(documents)

Teknik lainnya adalah Lemmatization. Kita dapat mengubah kata ke dalam bentuk lemma sehingga kita dapat mengurangi semua kata kanonik. Misalnya kata play, playing, plays, played, dll semuanya akan diganti dengan play. Dengan cara ini, kita dapat mengurangi jumlah total kata dalam kosa kata kita. Jadi sekarang kita lemmatize setiap kata dan menghapus kata-kata duplikat.

# lemmaztize and lower each word and remove duplicates

words = [lemmatizer.lemmatize(w.lower()) for w in words if w not in ignore\_letters]

words = sorted(list(set(words)))

# sort classes

classes = sorted(list(set(classes)))

# documents = combination between patterns and intents

print (len(documents), "documents")

# classes = intents

print (len(classes), "classes", classes)

# words = all words, vocabulary

print (len(words), "unique lemmatized words", words)

pickle.dump(words,open('words.pkl','wb'))

pickle.dump(classes,open('classes.pkl','wb'))

1. Membuat Training dan Testing Data Untuk model Train, setiap pola akan diubah dan diinput menjadi angka. Pertama, kita akan lemmatize setiap kata dari pola dan membuat daftar nol dengan panjang yang sama dengan jumlah total kata. Lalu menetapkan nilai 1 hanya untuk indeks yang berisi kata dalam pola.

# create the training data

training = []

# create empty array for the output

output\_empty = [0] \* len(classes)

# training set, bag of words for every sentence

for doc in documents:

# initializing bag of words

bag = []

# list of tokenized words for the pattern

word\_patterns = doc[0]

# lemmatize each word - create base word, in attempt to represent related words

word\_patterns = [lemmatizer.lemmatize(word.lower()) for word in word\_patterns]

# create the bag of words array with 1, if word is found in current pattern

for word in words:

bag.append(1) if word in word\_patterns else bag.append(0)

# output is a '0' for each tag and '1' for current tag (for each pattern)

output\_row = list(output\_empty)

output\_row[classes.index(doc[1])] = 1

training.append([bag, output\_row])

# shuffle the features and make numpy array

random.shuffle(training)

training = np.array(training)

# create training and testing lists. X - patterns, Y - intents

train\_x = list(training[:,0])

train\_y = list(training[:,1])

print("Training data is created")

**Interaksi dengan chatbot**

Untuk berinteraksi dengan chatbot, maka perlu ada tampilan untuk chat. Berikut kode program untuk berinteraksi dengan chatbot

import nltk

from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

import pickle

import numpy as np

from keras.models import load\_model

model = load\_model('chatbot\_model.h5')

import json

import random

intents = json.loads(open('intents.json').read())

words = pickle.load(open('words.pkl','rb'))

classes = pickle.load(open('classes.pkl','rb'))

def clean\_up\_sentence(sentence):

# tokenize the pattern - splitting words into array

sentence\_words = nltk.word\_tokenize(sentence)

# stemming every word - reducing to base form

sentence\_words = [lemmatizer.lemmatize(word.lower()) for word in sentence\_words]

return sentence\_words

# return bag of words array: 0 or 1 for words that exist in sentence

def bag\_of\_words(sentence, words, show\_details=True):

# tokenizing patterns

sentence\_words = clean\_up\_sentence(sentence)

# bag of words - vocabulary matrix

bag = [0]\*len(words)

for s in sentence\_words:

for i,word in enumerate(words):

if word == s:

# assign 1 if current word is in the vocabulary position

bag[i] = 1

if show\_details:

print ("found in bag: %s" % word)

return(np.array(bag))

def predict\_class(sentence):

# filter below threshold predictions

p = bag\_of\_words(sentence, words,show\_details=False)

res = model.predict(np.array([p]))[0]

ERROR\_THRESHOLD = 0.25

results = [[i,r] for i,r in enumerate(res) if r>ERROR\_THRESHOLD]

# sorting strength probability

results.sort(key=lambda x: x[1], reverse=True)

return\_list = []

for r in results:

return\_list.append({"intent": classes[r[0]], "probability": str(r[1])})

return return\_list

def getResponse(ints, intents\_json):

tag = ints[0]['intent']

list\_of\_intents = intents\_json['intents']

for i in list\_of\_intents:

if(i['tag']== tag):

result = random.choice(i['responses'])

break

return result

#Creating tkinter GUI

import tkinter

from tkinter import \*

def send():

msg = EntryBox.get("1.0",'end-1c').strip()

EntryBox.delete("0.0",END)

if msg != '':

ChatBox.config(state=NORMAL)

ChatBox.insert(END, "You: " + msg + '\n\n')

ChatBox.config(foreground="#446665", font=("Verdana", 12 ))

ints = predict\_class(msg)

res = getResponse(ints, intents)

ChatBox.insert(END, "Bot: " + res + '\n\n')

ChatBox.config(state=DISABLED)

ChatBox.yview(END)

root = Tk()

root.title("Chatbot")

root.geometry("400x500")

root.resizable(width=FALSE, height=FALSE)

#Create Chat window

ChatBox = Text(root, bd=0, bg="white", height="8", width="50", font="Arial",)

ChatBox.config(state=DISABLED)

#Bind scrollbar to Chat window

scrollbar = Scrollbar(root, command=ChatBox.yview, cursor="heart")

ChatBox['yscrollcommand'] = scrollbar.set

#Create Button to send message

SendButton = Button(root, font=("Verdana",12,'bold'), text="Send", width="12", height=5,

bd=0, bg="#f9a602", activebackground="#3c9d9b",fg='#000000',

command= send )

#Create the box to enter message

EntryBox = Text(root, bd=0, bg="white",width="29", height="5", font="Arial")

#EntryBox.bind("<Return>", send)

#Place all components on the screen

scrollbar.place(x=376,y=6, height=386)

ChatBox.place(x=6,y=6, height=386, width=370)

EntryBox.place(x=128, y=401, height=90, width=265)

SendButton.place(x=6, y=401, height=90)

root.mainloop()

**Menjalankan chatbot**

Berikut perintah yang dijalankan untuk menampilkan tampilan chat melalui commandline

*python train\_chatbot.py*

*Nb. Jika modul tensorflow, keras, nltk tidak berjalan bisa dengan cara membuka environment nya dengan perintah*

*python -m venv chatbot-env*

*lalu untuk mengaktifkan dengan menambahkan perintah:*

*chatbot-env\Scripts\activate*

*selanjutnya silahkan install tensorflow, keras dan nltk dengan perintah:*

*pip install tensorflow keras*

*Jika menggunakan python version 3.10 maka untuk menginstall modul bisa menggunakan perintah:*

*Py -m pip install tensorflow*

*Py -m pip install keras*

*Py -m pip install pickle*

*Py -m pip install nltk*