BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**NIÊN LUẬN**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Đề tài**

**CHATBOT CƠ BẢN**

**Sinh viên: Trần Đức Thắng**

**Mã số: B1910450**

**Khóa: K45**

**Cần Thơ, 11/2022**

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**NIÊN LUẬN**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Đề tài**

**CHATBOT CƠ BẢN**

**Người hướng dẫn Sinh viên thực hiện**

Ths. Lê Hoàng ThảoTrần Đức Thắng

MSSV: B1910450

Khóa: K45

***Cần Thơ, 12/2021***

**LỜI CẢM ƠN**

Em xin cảm ơn……...

Em xin chân thành cám ơn!

**MỤC LỤC**

[PHẦN 1: GIỚI THIỆU 1](#_Toc121250257)

[I. CHATBOT LÀ GÌ? 1](#_Toc121250258)

[II. MỤC TIÊU ĐỀ TÀI 1](#_Toc121250259)

[1. Mục tiêu về kiến thức 1](#_Toc121250260)

[2. Mục tiêu về sản phẩm 1](#_Toc121250261)

[III. ĐỐI TƯỢNG VÀ PHẠM VI NGHIÊN CỨU 1](#_Toc121250262)

[IV. NỘI DUNG NGHIÊN CỨU 2](#_Toc121250263)

[PHẦN 2: NỘI DUNG 2](#_Toc121250264)

[I. ĐẶC TẢ YÊU CẦU 2](#_Toc121250265)

[II. PHÂN TÍCH VÀ THIẾT KẾ GIẢI PHÁP 2](#_Toc121250266)

[PHẦN 3: CƠ SỞ LÝ THUYẾT 3](#_Toc121250267)

[I. NGÔN NGỮ LẬP TRÌNH PYTHON VÀ TRÍ TUỆ NHÂN TẠO 3](#_Toc121250268)

[1. Ngôn Ngữ Lập Trình Python 3](#_Toc121250269)

[2. Trí Tuệ Nhân Tạo 3](#_Toc121250270)

[II. THƯ VIỆN CHÍNH 3](#_Toc121250271)

[1. Keras 3](#_Toc121250272)

[2. NLTK 3](#_Toc121250273)

[PHẦN 4: CÀI ĐẶT GIẢI PHÁP 4](#_Toc121250274)

[I. TIỀN XỬ LÝ DỮ LIỆU 4](#_Toc121250275)

[1. Mô tả giải pháp 4](#_Toc121250276)

[2. Xử lý bằng Python 4](#_Toc121250277)

[II. XÂY DỰNG MÔ HÌNH 5](#_Toc121250278)

[1. Mô tả giải pháp 5](#_Toc121250279)

[2. Xử lý bằng Python 5](#_Toc121250280)

[PHẦN 5: KẾT LUẬN 6](#_Toc121250281)

[1. Kết quả đạt được 6](#_Toc121250282)

[2. Hướng phát triển 6](#_Toc121250283)

[3. Tài liệu tham khảo 6](#_Toc121250284)

**Danh Mục Bảng**

**Danh Mục Hình**

[Hình 1. Chia văn bản thành các từ (tokenizing) 4](#_Toc121333959)

[Hình 2. Bổ sung, xoá từ trùng và sắp xếp lại 4](#_Toc121333960)

[Hình 3. Xuất các từ đã tách ra file 5](#_Toc121333961)

[Hình 4. Chuyển đổi văn bản thành số và tạo tập training 5](#_Toc121333962)

[Hình 5. Tạo mạng neural cho mô hình 6](#_Toc121333963)

[Hình 6. Training và lưu mô hình 6](#_Toc121333964)

[Hình 7. Import dữ liệu 7](#_Toc121333965)

[Hình 8. Hàm tách từ 7](#_Toc121333966)

[Hình 9. Hàm xác định phần tử 7](#_Toc121333967)

# PHẦN 1: GIỚI THIỆU

## I. CHATBOT LÀ GÌ?

Chatbot là một phần mềm thông minh có khả năng giao tiếp và thực hiện các hành động tương tự như con người. Chatbots được sử dụng rất nhiều trong tương tác với khách hàng, tiếp thị trên các trang mạng xã hội và trả lời tin nhắn tự động với khách hàng... Có hai loại mô hình chatbot cơ bản là: Retrieval based và Generative based models.

**Retrieval-based Chatbots:** sử dụng các mẫu đầu vào và các phản hồi tương ứng đã được định nghĩa từ trước. Sau đó, nó sử dụng một số loại phương pháp tiếp cận heuristic để chọn câu trả lời thích hợp nhất với đầu vào (input) và bối cảnh (context). Heuristic ở đây có thể đơn giản là các kĩ thuật khớp luật (rule-based expression matching). Retrieval-based chatbot không thể tạo ra bất kỳ từ mới nào khác, chúng chỉ lấy một số câu trả lời từ một tập có sẵn.

**Generative-based Chatbots:** Những mô hình như thế này không dựa vào tập phản hồi định nghĩa trước. Chúng dựa trên mạng nơ-ron Sequence to Sequence, được phát triển dựa trên bài toán Dịch Máy (Machine translation). Nhưng thay vì dịch từ ngôn ngữ này sang ngôn ngữ khác, mô hình này sẽ dịch từ input ra output, tức là chúng có thể truy xuất thông tin từ input và phản hồi lại một cách ấn tượng như bạn đang nói chuyện với con người. Generative based Chatbot sẽ cần một lượng lớn dữ liệu và training trên mạng Deep Neural.

## II. MỤC TIÊU ĐỀ TÀI

### 1. Mục tiêu về kiến thức

* Nắm được kiến thức Python để cài đặt giải thuật và thư viện tkinter để thiết kế giao diện.
* Nắm được kiến thức cơ bản và giải thuật của thư viện Keras để áp dụng thuật toán máy học Retrieval-based.

### 2. Mục tiêu về sản phẩm

* Phân tích và xử lý dữ liệu (Tiền xử lý dữ liệu)
* Xây dựng mô hình máy học
* Tạo Chatbot dựa trên mô hình
* Chạy Chatbot

## III. ĐỐI TƯỢNG VÀ PHẠM VI NGHIÊN CỨU

* Sử dụng ngôn ngữ lập trình bậc cao Python và một số thư viện như Keras, Tensorflow, NLTK để tạo ra Chatbot

## IV. NỘI DUNG NGHIÊN CỨU

|  |  |
| --- | --- |
| **STT** | **Nội dung** |
| **1** | Tìm hiểu Machin Learning và tiền xử lý dữ liệu |
| **2** | Tìm hiểu về Keras và NLTK |
| **3** | Tạo tập dữ liệu học |
| **4** | Xây dựng mô hình |
| **5** | Tạo Chatbot với mô hình |

Bảng : Nội dung nghiên cứu

# 

# PHẦN 2: NỘI DUNG

## I. ĐẶC TẢ YÊU CẦU

Chatbot có khả năng giao tiếp, trả lời các câu hỏi của người dùng một cách chính xác.

## II. PHÂN TÍCH VÀ THIẾT KẾ GIẢI PHÁP

Tạo Chatbot chúng ta chia ra làm hai phần là mô hình máy học và huấn luyện tạo Chatbot.

Mô hình máy học, sử dụng một tập huấn luyện để tăng dộ chính xác cho Chatbot. Tập huấn luyện trước khi được sử dụng trải qua giai đoạn tiền xử lý dữ liệu do máy không đọc được kỹ tự cần chuyển về số (0,1).

Huấn luyện tạo Chatbot, sử dụng mô hình đã tạo ra để huấn luyện. Sau đó tạo giao diện để người dùng có thể giao tiếp với Chatbot.

# PHẦN 3: CƠ SỞ LÝ THUYẾT

## I. NGÔN NGỮ LẬP TRÌNH PYTHON VÀ TRÍ TUỆ NHÂN TẠO

### 1. Ngôn Ngữ Lập Trình Python

Python là ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng, do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python được thiết kế với ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Cùng với đó là cộng đồng cũng như là kho thư viện khổng lồ do cộng đồng cung cấp đã góp phần làm cho Python trở thành ngôn ngữ lập trình phổ biến nhất thế giới.

Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình. Được dùng rộng rãi trong phát triển trí tuệ nhân tạo.Python được phát triển trong một dự án mã nguồn mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

### 2. Trí Tuệ Nhân Tạo

Trí tuệ nhân tạo hay AI (Artificial Intelligence) là trí thông minh được thể hiện bằng máy móc, trái với trí thông minh tự nhiên của con người. Dùng để mô tả các máy móc có khả năng bắt chước các chức năng nhận thức của con người.

Trí thông minh nhân tạo được chia làm ba loại hệ thống khác nhau: trí tuệ nhân tạo phân tích, lấy cảm hứng từ con người và nhân tạo.

## II. THƯ VIỆN CHÍNH

### 1. Keras

Keras là một open source cho Neural Network được viết bởi ngôn ngữ lập trình Python. Nó là một library được phát triển vào năm 2005 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể được sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano. Một số ưu điểm Keras như:

* Dể sử dụng, dung đơn giản hơn Tensor, xây dựng model nhanh.
* Run được trên cả CPU và GPU
* Hỗ trợ xây dựng CNN, RNN hoặc cả hai.

### 2. NLTK

Thư viện NLTK – Natural Language Toolkit là một trong những thư viện open source xử lý ngôn ngữ tự nhiên được viết bằng Python. Với ưu điểm là dễ sử dụng nên thu viện này trở nên phổ biến và có cộng đồng ngày càng lớn mạnh. Thư viện cung cấp hơn 50 kho dữ liệu văn bản khác nhau (corpora) và nhiều chức năng để xử lý dữ liệu văn bản để phục vụ cho nhiều mục đích khác nhau.

# PHẦN 4: CÀI ĐẶT GIẢI PHÁP

## I. TIỀN XỬ LÝ DỮ LIỆU

### 1. Mô tả giải pháp

Để xử lý dữ liệu đầu vào là những câu mẫu trong tập dữ liệu (input) dùng để train cho mô hình máy học ta chia toàn bộ dữ liệu văn bản thành các phân tử nhỏ như các tư rồi sau đó chuyển các từ về dạng số để máy có thể hiểu được.

### 2. Xử lý bằng Python

#### 2.1. Tách từ

Để có thể chuyển từ chữ thành số cho máy hiểu thì ta cần phải chuyển một câu thành từng để có thể dễ dàng chuyển đổi. Tokenizing là điều cơ bản nhất và đầu tiên có thể làm trên dữ liệu văn bản. Tokenizing là quá trình chia toàn bộ văn bản thành các phần nhỏ như các từ. Thư viện NLTK của Python là biện pháp đơn giản nhât củ thể là hàm nltk.sent\_tokenize.

for intent in intents['intents']:

    for pattern in intent['patterns']:

        # Mã hoá từng từ

        w = nltk.word\_tokenize(pattern)

        words.extend(w)

        # Thêm tài liệu vào kho dữ liệu

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

        # Thêm vào danh sách lớp cho máy học

        if intent['tag'] not in classes:

            classes.append(intent['tag'])

Hình . Chia văn bản thành các từ (tokenizing)

# Bổ sung, xoá từ trùng lặp và sắp xếp từ

words = [w.lower() for w in words if w not in ignore\_words]

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

Hình . Bổ sung, xoá từ trùng và sắp xếp lại

Sau khi tách từ ra và gán từ đó thuộc vào một class để sau này máy có thể nhận diện được. Xuất những từ được đã được lưu vào words và classes vào tập tin để sử dụng lại cho các bước sau.

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

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

Hình . Xuất các từ đã tách ra file

#### 2.2 Tạo tập dữ liệu đào tạo

Tạo tập dữ liệu training bào gồm các input và output, input là các **patterns** và output là các **responses** tương ứng. Như đã nói ở trên, máy tính không thể hiểu văn bản nên ta sẻ thực hiện chuyển văn bản dạng chữ về dạng số.

Để chuyển về dạng số ta tạo mảng **pattern\_words** chứa các câu đã chia từ. Sau đó ta tạo 1 mảng **bag** chứa các từ, nếu có thì là **1** ngược lại là **0** (Hơi khó hiểu, nhưng cài đặt lại rất dễ). Sau khi chuyển đổi xong thì thêm từ và lớp vào mảng **training**.

for doc in documents:

    bag = []

    # Danh sách các từ được mã hoá (documents chứa các từ vựng trong phần pattern)

    pattern\_words = doc[0]

    # Biểu thị các từ có liên quan

    pattern\_words = [word.lower() for word in pattern\_words]

    for w in words:

        bag.append(1) if w in pattern\_words else bag.append(0)

    # Các vị trí ko xét = 0

    output\_row = list(output\_empty)

    # Đánh dấu vị trí hiện tại = 1

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

    # Tạo tập training với bag chứa các mẫu từ đã chuyển số đựng trong bag và đầu ra là câu trả lời tương ứng

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

Hình . Chuyển đổi văn bản thành số và tạo tập training

## II. XÂY DỰNG MÔ HÌNH

Sau khi xây dựng được tập train, xây dựng 1 mạng neural với 3 layers bằng cách sử dụng Keras sequential API. Layer thứ nhất có 128 nơron với input là tập **train\_x**. layer thứ 2 có 64 nơron là một hidden layer. Layer thứ 3 chứa output là **train\_y** kết quả của quá trình dự đoán.

model = Sequential()

model.add(Dense(128, input\_shape=(len(train\_x[0]),), activation="relu"))

model.add(Dropout(0.5)) # Dropout tránh over-fitting

model.add(Dense(64, activation="relu"))

model.add(Dropout(0.5))

model.add(Dense(len(train\_y[0]), activation="softmax")) # softmax hàm tính trọng số cho dữ liệu

Hình . Tạo mạng neural cho mô hình

Có được mô hình như đã tạo ở trên, ta bắt đầu biên dịch mô hình và train mô hình với tập dữ liệu trên. Tiếp theo chúng ta sẽ lưu mô hình lại để có thể tái sử dụng cho ứng dụng khác.

sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)

model.compile(loss='categorical\_crossentropy', optimizer=sgd, metrics=['accuracy'])

# fit và lưu model

tots = model.fit(np.array(train\_x), np.array(train\_y), epochs=1000000, batch\_size=20, verbose=1)

model.save('train\_model.h5', tots)

Hình . Training và lưu mô hình

## III. XÂY DỰNG GIAO DIỆN

### 1.Xử lý yêu cầu (input)

Yêu cầu của người dùng nhập vào là một chỗi String hay đúng hơn là 1 câu thô cần được tiền xử lý như phần I mới có thể sử dụng được. Trước hết ta cần import những file ta đã tạo ra ở các phần trên: data\_set, train\_model.h5, words.pkl, classes.pkl.

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

intents = json.loads(open('./intents.json', encoding = "utf8").read())

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

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

Hình . Import dữ liệu

Sau khi đã import dữ liệu ta bắt đầu với chia câu giống với phần tiền dữ liệu phía trên ở đây sẻ viết một hàm

def tokenizer\_sentence(sentence):

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

    sentence\_w = [ w.lower() for w in sentence\_w ]

    return sentence\_w

Hình . Hàm tách từ

Để xác định xem câu của người dụng thuộc ý định là gì ta sẽ viết một hàm để tìm xem câu đó thuộc về những câu nào trong tập **words** mà ta đã tách ra được từ dữ liệu train.

def bag\_element(sentence, words):

    sentence\_w = tokenizer\_sentence(sentence)

    bag = [0] \* len(words)

    for s in sentence\_w:

        for i,w in enumerate(words): #enumarate sinh index bắt đầu từ 0

            if w == s:

                bag[i] = 1

    return (np.array(bag))

Hình . Hàm xác định phần tử

Viết một dự đoán xem câu người dùng nhập vào thuộc **classes** (dự định) nào. Những hàm này được Keras hỗ trợ có sẵn khi tạo model.

def predict\_classes(sentence, model):

    p = bag\_element(sentence, words)

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

    ERROR\_THRESHOLD = 0.25

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

    # print(result)

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

    result\_list = []

    for r in results:

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

    return result\_list

Hình . Dự đoán classes

Tiếp theo ta viết một hàm trả phản hồi khi đã xác định được ý định.

def getRespond (ints, intent\_json):

    tag = ints[0]['intent']

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

    for i in list\_intents:

        if i['tag'] == tag:

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

            break

    return result

def bot\_respond(text\_input):

    ints = predict\_classes(text\_input, model)

    respond = getRespond(ints, intents)

    return respond

Hình . Phản hồi ngẫu nhiên của Bot

### 2.Giao diện với tkinter

Trong phần giao diện chúng ta sẽ sử dụng một thư viện có sẵn trong Python đó là tkinter. Giúp tạo giao diện đơn giản. Trước khi viết giao diện chúng ta cần viết một hàm xử lý input của người dùng và gọi tới hàm trả lại phản hồi của Bot để in ra màn hình.

def send():

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

    EntryBox.delete("0.0",END)

    if msg != '':

            ChatLog.config(state=NORMAL)

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

            ChatLog.config(foreground="#442265", font=("Verdana", 12 ))

            res = bot\_respond(msg)

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

            ChatLog.config(state=DISABLED)

            ChatLog.yview(END)

Hình . Hàm lấy thông tin và trả phản hồi ra màn hình

# 

# PHẦN 5: KẾT LUẬN

## 1. Kết quả đạt được

## 2. Hướng phát triển

## 3. Tài liệu tham khảo