

自主學習報告

利用Python實作一款屬於自己的聊天機器人

三壬45 廖培君

在現代的科技領域中，聊天機器人已然成為人類與機器互動的重要工具，而其被廣泛應用於客戶服務、資訊獲取等領域。這些經由程式及資料分析後，能夠去模擬人與人之間的對話。而它能夠透過語言處理和機器學習等技術，去理解並回應使用者的需求。

聊天機器人的種類

根據技術和功能的不同，聊天機器人主要分為以下三類：

- 1.基於規則的聊天機器人：這種類型的機器人會遵循預先設定的規則和條件，當使用者輸入相似的問題時，機器人會檢查是否符合這些規則，並給出相應的回答。此類機器人適用於問答固定、答案明確的場景，如簡單的、時常出現在各種服務業官網的「常見問題解答」。
- 2.基於機器學習的聊天機器人：這類的機器人具備學習和理解語言的能力，能夠從大量數據中學習並記憶不同使用者的語言模式，處理更複雜的對話，並生成更自然且符合上下文的回答。
- 3.混合式聊天機器人：顧名思義，其結合了機器學習的方法及規則服從（將對話限制在框架中，給予一定範圍）。這類機器人透過規則引導對話流程，並使用機器學習模型生成回答。適用於存在較多變因的複雜對話場景，兼具控制對話流程和生成自然回應的優點，將對話侷限在一定大小的框架內，同時給予了用戶一定的自由度。

聊天機器人簡述

在這次的自主學習中，我學會了如何透過Python及相關模組去實作能夠具備基本語言能力的機器人，並透過訓練及導入數據去使其能夠有高精度。

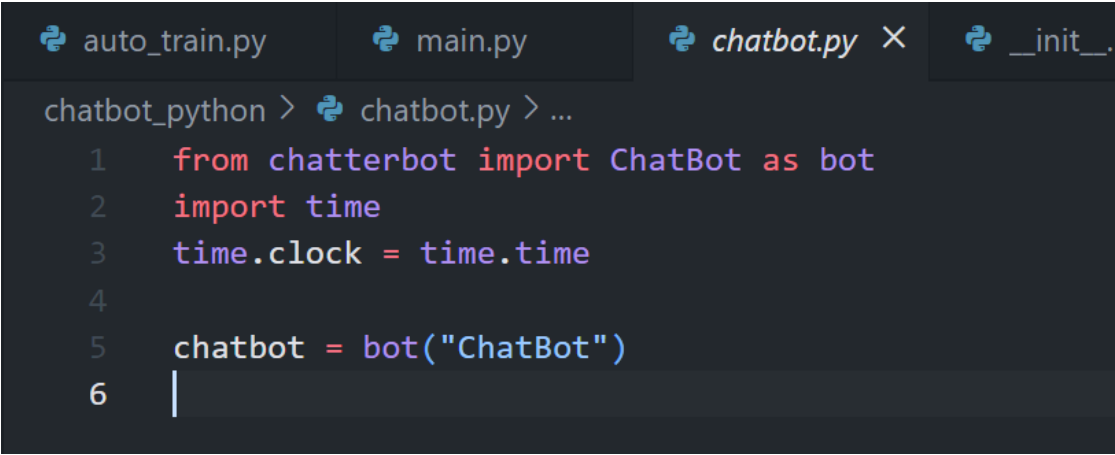
而本次，我所實作的AI是利用了預測及模仿的原理去訓練資料及與使用者對話，使其能夠在與用戶的對話中同時獲得訓練的機會。而其會根據對於不同的對話中用戶所回答去記憶並學習，以此達到「對話」的效果。

而更高階的語言模型，以ChatGPT-4o為例，其透過了整合演算法及字節之間的邏輯性，去預測並且回答所輸入的問題，並整合語言、音訊及影像，去帶給使用者更好的體驗。

但是話又說回來，我們一般人的電腦也沒有如此大規模的設備，包括儲存空間、算力以及執行速度。所以，除了利用連線的API或是只擷取部分高精度的資料去回答部分問題外，在我思考了幾天後，沒有想到任何方法，於是只好退而求其次，尋找一個較為簡單但可以實作並且闡明其原理的方式。（由於Python為直譯式的語言，所以運算速度也比編譯式的慢，但是其擁有較豐富的函式庫，所以最後選擇使用它）

實作聊天機器人之步驟

第一步，我先去網路上尋找了能夠輔助我去利用Python撰寫語言處理AI的相關函式庫，而在這其中，我找到了Chatterbot，其不僅可以透過與其對話去訓練，也可以利用其提供的資料庫去獲得大量的語言資料。這樣一來，解決了許多的對訓練資料方面的困擾。（但是，「導入資料」這個方法是我後期才發現



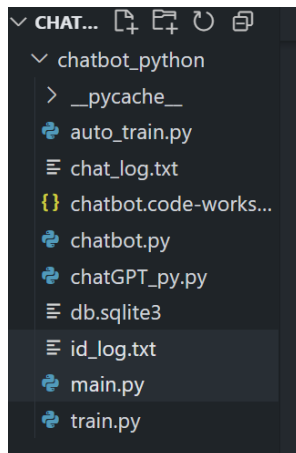
```
auto_train.py  main.py  chatbot.py X  __init__.py
chatbot_python > chatbot.py > ...
1  from chatterbot import ChatBot as bot
2  import time
3  time.clock = time.time
4
5  chatbot = bot("ChatBot")
6  |
```

如上圖，步驟其實非常簡單，只是利用其給予語法去建立一個機器人之對話數據模型而已。

第二步，開始撰寫程式：我將訓練用及對話用的不同功能寫入了不同的檔案

中，分別是train.py及chat.py。而我打算將程式化為函式，去整合到main.py

中。而我也將AI的本體寫在另外一個程式中，方便其他的程式碼去調用。



如左圖所示，可以看到我將程式分開儲存，以利辨識。而右圖中是我本次的主要程式碼，可以看到，再利用了函式

```
from chatterbot import ChatBot as bot
import time

time.clock = time.time
chatbot = bot("ChatBot")
exit_conditions = (":q", "quit", "exit")
while True:
    query = input(">")
    if query in exit_conditions:
        break
    else:
        print(f"{chatbot.get_response(query)}")
```

庫之後，變得十分輕鬆。感謝作者。

第三步，訓練資料：前期利用了土法煉鋼的方式，去利用操控滑鼠及鍵盤的程式去讓其與ChatGPT進行對話。後來發現，由於我的ChatGPT沒有升級，導致其不僅效率差，也同時浪費了許多的時間在這上面。而後來，我發現可以透過導入的方式，直接引用他人（這邊是以撰寫這個函式庫的作者所提供的基本數據去進行導入）

```
auto_train.py  main.py  train.py  ×  __init__.py
chatbot_python > train.py > ...
1  from chatbot import chatbot
2  from chatterbot.trainers import ChatterBotCorpusTrainer
3
4  trainer = ChatterBotCorpusTrainer(chatbot)
5
6  trainer.train(
7      "chatterbot.corpus.english"
8  )
```

如上圖，利用其給予的內建資料庫中之訓練資料，能夠輕鬆解決訓練過程耗時、訓練步驟繁瑣等問題。

總結

總而言之，聊天機器人作為近代科技不可或缺的重要組成部分，透過不斷的技術進步，正逐漸改變人們的生活和工作方式。而我們應當將其作為一個工具去使用，以此來讓我們的生及工作增添便利性，並利用它去為我們打造一個更美好的世界。

在這次的自主學習中，我學會的不只是一個觀念，更是一個對未來的構想，一項無法取代的技能。

引用：

<https://www.wati.io/blog/聊天機器人/>

<https://openai.com/index/hello-gpt-4o/>

<https://github.com/gunthercox/ChatterBot>