远程科研指导项目总结报告

聊天机器人项目总结

姓名：李文海

学校：吉林大学

专业：软件工程

目录

1． 项目背景

人工智能一直是计算机方向的一个热点，此项目通过编写一个股票查询机器人来进一步学习理解人工智能。本项目通过使用anaconda 平台中的 jupyter 编写实现。使用支持向量机学习算法，Rasa 的机器对话框架，正则表达式和 iexfinance 中的 Python package，构造了一个股票机器人问答系统，以便人们获取到指定股票的信息，系统具有一定的理解人类语言的能力。

2．相关知识

2.1 Jupyter

Jupyter Notebook 是一个交互式笔记本，其本质是一个 Web 应用程序，便于创建和共享文学化程序文档，支持实时代码，数学方程，可视化和 markdown。 用途包括：数据清理和转换，数值模拟，统计建模，机器学习等等。

2.2 自然语言处理

自然语言处理（NLP）是计算机科学，人工智能，语言学关注计算机和人类（自然）语言之间的相互作用的领域，它是人工智能领域中的一个研究方向，用于研究能实现人与计算机之间用自然语言进行有效通信的各种理论和方法。而在自然语言处理面临很多挑战，包括自然语言理解，因此，自然语言处理涉及人机交互的面积。

2.3 Rasa

Rasa_NLU 是一个自然语言理解（NLU）的一个框架。

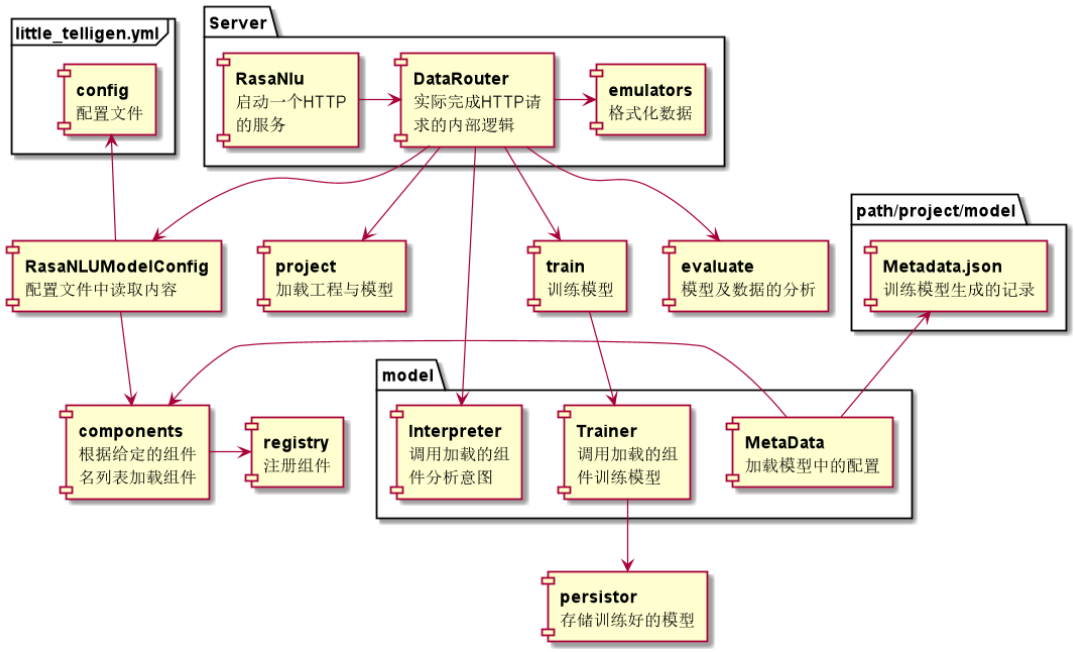
Rasa_NLU框架的主要工作是：

组件管理

训练数据管理

模型管理

提供Http服务



2.4 鲁棒性的说明

鲁棒性通常指计算机软件等在异常情况下的稳定性，如当输入非法数据时能作出处理，而不是输出莫名其妙的结果，就说明它健壮。通过数据集等的使用，增强鲁棒性。

3．流程分析

3.1 Python 正则表达式（re 模块）

```
bot_template = "{0}"
user_template = "{0}"
def send_message1(message):
    # Get the bot's response to the message
    response = respond1(message)
    # Print the bot template including the bot's response.
    print(bot_template.format(response))

import re
keywords = {
            'yes': ['yeah','ok'],
            'greet': ['hello', 'hi', 'hey'],
            'thankyou': ['thank', 'thx'],
            'stock': ['stocks','sto'],
            'company': ['frim','business'],
            'goodbye': ['bye', 'farewell']
           }
# Define a dictionary of patterns
patterns = {}

# Iterate over the keywords dictionary
for intent, keys in keywords.items():
    # Create regular expressions and compile them into pattern objects
    patterns[intent] = re.compile("|".join(keys))

responses = {
             'yes':'ok !',
             'greet': 'Hello you! , be glad to see you',
             'thankyou': 'you are very welcome',
             'default': 'default message',
             'goodbye': 'goodbye for now'
            }
```

```
# Define a function to find the intent of a message
def match_intent(message):
    matched_intent = None
    for intent, pattern in patterns.items():
        # Check if the pattern occurs in the message
        if re.search(pattern,message):
            matched_intent = intent
    return matched_intent

# Define a respond function
def respond1(message):
    # Call the match_intent function
    intent = match_intent(message)
    # Fall back to the default response
    key = "default"
    if intent in responses:
        key = intent
    return responses[key]

# Send messages
x=input()
send_message1(x)
```

3.2 rasa 训练集

```
# Import necessary modules
from rasa_nlu.training_data import load_data
from rasa_nlu.config import RasaNLUModelConfig
from rasa_nlu.model import Trainer
from rasa_nlu import config

# Create a trainer that uses this config
trainer = Trainer(config.load("config_spacy.yml"))

# Load the training data
training_data = load_data('demo-rasa.json')

# Create an interpreter by training the model
interpreter = trainer.train(training_data)
```

```
# Define respond()
def respond(message):
    # Extract the entities
    entities = interpreter.parse(message)["entities"]
    # Initialize an empty params dictionary
    params = {}
    # Fill the dictionary with entities
    for ent in entities:
        params[ent["entity"]] = str(ent["value"])

    # Find hotels that match the dictionary
    results = find_hotels(params)
    # Get the names of the hotels and index of the response
    names = [r[0] for r in results]
    n = min(len(results),len(responses)-1)
    # Select the nth element of the responses array
    return responses[n].format(*names)
```

3.3 记忆对话查询股票信息

```
def send_message(policy, state, message):
    #print("USER : {}".format(message))
    new_state, response = respond(policy, state, message)
    print("{}".format(response))
    return new_state

def respond(policy, state, message):
    (new_state, response) = policy[(state, interpret(message))]
    return new_state, response

def interpret(message):
    msg = message.lower()
    if 'check' in msg:
        return 'check'
    if 'ok' in msg or 'go' in msg:
        return 'specify_stock'
    if 'what' in msg:
        return 'ask_explanation'
    return 'none'

# Define the policy rules
policy = {
    (INIT, "check"): (CHOOSE_COFFEE, "I can provide you a company with all stock information "),
    (INIT, "ask_explanation"): (INIT, "I'm a bot to help you check stock"),
    (CHOOSE_COFFEE, "specify_stock"): (ORDERED, "ok, Which company do you want to check"),
    (CHOOSE_COFFEE, "ask_explanation"): (CHOOSE_COFFEE, "I'm sorry – would you like aapl or fb?"),
}
```

```
v=input("请输入你想查询的公司:")
from iexfinance.stocks import Stock
stock= Stock(v, token="pk_0537ab872b11442f9fde606bae2a25fd")
stock.get_quote()
```

请输入你想查询的公司aapl

{'symbol': 'AAPL',
 'companyName': 'Apple, Inc.',
 'primaryExchange': 'NASDAQ',
 'calculationPrice': 'close',
 'open': 204.36,
 'openTime': 1565962200715,
 'close': 206.5,
 'closeTime': 1565985600320,
 'high': 207.16,
 'low': 203.84,
 'latestPrice': 206.5,
 'latestSource': 'Close',
 'latestTime': 'August 16, 2019',
 'latestUpdate': 1565985600320,
 'latestVolume': 28135640,
 'iexRealtimePrice': None,
 'iexRealtimeSize': None,
 'iexLastUpdated': None,
 'delayedPrice': 206.73,
 'delayedPriceTime': 1565988740053,
 'extendedPrice': 206.45,
 'extendedChange': -0.05,
 'extendedChangePercent': -0.00024,
 'extendedPriceTime': 1566086382886,
 'previousClose': 201.74,
 'previousVolume': 27883363,
 'change': 4.76,
 'changePercent': 0.02359,
 'volume': 28135640,
 'iexMarketPercent': None

4． 功能展示

```
hello
Hello you,  be glad to see you
what can you do for me
I'm a bot to help you check stock
how do you check
I can provide you a company with all stock information
ok go
ok, Which company do you want to check
请输入你想查询的公司:aapl

{'symbol': 'AAPL',
 'companyName': 'Apple, Inc.',
 'primaryExchange': 'NASDAQ',
 'calculationPrice': 'close',
 'open': 204.36,
 'openTime': 1565962200715,
 'close': 206.5,
 'closeTime': 1565985600320,
 'high': 207.16,
 'low': 203.84,
 'latestPrice': 206.5,
 'latestSource': 'Close',
 'latestTime': 'August 16, 2019',
 'latestUpdate': 1565985600320,
 'latestVolume': 28135640,
 'iexRealtimePrice': None,
 'iexRealtimeSize': None,
 'iexLastUpdated': None,
 'delayedPrice': 206.73,
 'delayedPriceTime': 1565988740053,
 'extendedPrice': 206.45,
 'extendedChange': -0.05,
 'extendedChangePercent': -0.00024,
 'extendedPriceTime': 1566086382886,
 'previousClose': 201.74,
 'previousVolume': 27883363,
```

5．项目总结

通过这个项目一个月的学习和编程经历，我对 Python 这门编程语言更加的熟悉和灵活运用，尤其是对 rasa 有了深刻的体会，对训练集的构建和编写有了深刻的认识。

首先在这个项目中，老师通过课程为我们讲解了开发环境的构建，自然语言处理用法、正则表达式的应用、词向量的构成、命名实体识别、意图识别、否定、多轮查询和有状态 对话等内容。在我实际操作过程中，由于插件的版本问题困扰了我，老师也帮我细心的指导。其次，通过项目，我对 API 的调用有了更加深入的认识。对于 rasa_nlu 有了深刻的理解，训练集的构建使我明白了为什么机器人可以如此的智能，相信在今后的学习中可以对我有很大的帮助。最后，关于异常情况的测试，有时考虑的不全面，需要继续的改进。

总而言之，通过跟随老师这一个月来的教学，我对人工智能方面有了更大的兴趣，我相信如果有机会的话，我想进一步在这一领域进行学习研究。