股票查询机器人项目报告

陈意宁　2019/8/16

一 项目介绍

本次的股票查询机器人项目的是运用自然语言处理技术制作一个简易的股票查询机器人。用户可查询股票的信息有：最新价格、实时价格、开盘价格、收盘价格以及最新成交量。

其中运用的主要技术是基于 Rasa NLU 的本地基础聊天机器人系统的构建；同一个问题多种选择性的回答，并提供缺省回答的方案；能通过正则表达式、模式匹配、关键词提取、句法转换等来回答问题；实现状态机的多轮多次查询技术，并能基于语境问题提供解释和回答。

本报告中还包含未运用到的课程中重点技术的学习报告。

二 主要技术运用

1. Rasa NLU

Rasa 是一个基于多轮对话的框架，其中包含两个模块 Rasa core 与 Rasa nlu。Rasa nlu 是用来理解语义的，包括意图识别，实体识别，它会把用户的输入转换为结构化的数据。首先，我们需要训练数据集来训练模型，使得我们可以进行意图识别和实体识别，我这里使用的是 training-data.json。根据我们的目的—股票机器人，我列了以下几种意图：greet（问候），ask_usage（询问功能），stock_search（查询股票信息），thankyou（感谢）， bye（使用完毕），default（其他）。实体包括 company（公司名），price（价格），volume（销量）。有了这些意图及实体，我们可以更好得得出用户想要查询的东西并给出相应的回复。

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

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

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

2. 正则表达式

正则表达式是对字符串（包括普通字符（例如，a 到 z 之间的字母）和特殊字符（称为“元字符”））操作的一种逻辑公式，就是用事先定义好的一些特定字符、及这些特定字符的组合，组成一个“规则字符串”，这个“规则字符串”用来表达对字符串的一种过滤逻辑。正则表达式是一种文本模式，该模式描述在搜索文本时要匹配的一个或多个字符串。

使用正则表达式可以更加精准的找到想要形式的字符串，进行搜索或者替换十分便捷。在本项目中我还加入一个简易的反问形闲谈回复，即是找到匹配的字符

```
def replace_pronouns(message):

    message = message.lower()
    if re.search(r'\bme\b', message):
        return re.sub(r'\bme\b', 'you', message)
    if re.search(r"\b(i)\b", message):
        return re.sub(r"\b(i)\b", 'you', message)
    elif re.search(r'\bmy\b', message):
        return re.sub(r'\bmy\b', 'your', message)
    elif re.search(r'\byour\b', message):
        return re.sub(r'\byour\b', 'my', message)
    elif re.search(r'\byou\b', message):
        return re.sub(r'\byou\b', 'me', message)
    return message
```

串，如“I”，“you”在回复中将其替换为“you”，“me”然后进行回复。而且，使用 r'\bme\b'这样形式的正则表达式可以避免将 remember 中的 me 替换成 you。使得回复更加准确。

3. Spacy

Spacy 是由 cython 编写。因此它是一个非常快的库。spaCy 提供简洁的接口用来访问其方法和属性。要想使用 Spacy 和 访问其不同的 properties， 需要先创建 pipelines。 通过加载模型来创建一个 pipeline。Spacy 提供了许多不同的模型，模型中包含了语言的信息--词汇表，预训练的词向量，语法和实体。

在课程中，我们使用 spaCy 可以准确地提取出一些特定实体，如姓名，公司名称，日期时间等。也可以用利用词向量其测试两个词汇或词组的相似度，也可以识别形容词归属的名词，在自然语言识别技术中是一个非常多功能且实用的库。

4. 实现状态机的多轮多次查询技术

此项技术根据用户的的输入判定当前状态，并根据输入判断是否进入下一状态，且是否需要待定状态。如初始状态时，需要的待定状态是认证状态，即用户必须登录才可进行下一步操作，否则将永远停留在初始状态。

在本次项目中，我一共写了五种状态：

INIT=0（初始状态）

AUTHED=1（认证状态）

STOCK_QUERY=2（股票查询状态）

ORDERED=3（查询状态）

END=4（退出状态）

这些状态根据用户的输入判定状态并转移至下一状态，为用户更好地提供查询服务。

```
# Define the states
INIT=0
AUTHED=1
STOCK_QUERY=2
ORDERED=3
END=4

# Define the policy rules
policy_rules = {
    (INIT, "stock_search"): (INIT, "you'll have to log in first, what's your phone number?", AUTHED),
    (INIT, "number"): (AUTHED, "perfect, welcome! You can start searching stock information now. Here's the information
    (AUTHED, "stock_search"): (AUTHED, "Please tell me which stock you want to know.", STOCK_QUERY),
    (AUTHED, "stock_search"): (STOCK_QUERY, "What information do you want to know?", None),
    (STOCK_QUERY, "stock_search"): (ORDERED, "Here's the information. What else do you need?", None),
    (ORDERED, "stock_search"): (STOCK_QUERY, "What else do you want to know?", None),
    (ORDERED, "stock_search"): (ORDERED, "What else do you want to know?", None),
    (ORDERED, "end"): (END, "Byebye, hope to see you again!", None),
    (STOCK_QUERY, "end"): (END, "Ok bye, hope to see you again!", None),
}
```

5. 基于增量过滤器的单轮多次增量查询技术以及甄别否定实体技术

基于增量过滤器的单轮多次增量查询技术的作用是通过提取用户输入的信息，进行查询，并逐步限制条件，缩小查询范围，最终得出用户想要的信息。

甄别否定实体技术是判断是否含有‘not’，‘n’t’等词汇，即是否是否定，若是就将其加入 excluded list，排除在答案之外，继续进行查询的技术。

这两项技术的目的是精准明确用户的查询需求，通过逐步缩小范围来得到最优的答案。

6.与股票 API 的集成

在此项目中，我使用了 iexfinance.stocks API 接口取调取股票信息，根据用户的查询意图，返回相关信息。

```
def respond_latestPrice(company):
    aapl = Stock(company, token = "pk_05a3d93860d54ae4a2c0f9a01ed362d5")
    info = str(aapl.get_quote()['latestPrice'])
    bot_message =  "Its latest price is " + info +"$"
    return bot_message

def respond_latestVolume(company):
    aapl = Stock(company, token = "pk_05a3d93860d54ae4a2c0f9a01ed362d5")
    info = str(aapl.get_quote()['latestVolume'])
    bot_message =  "Its latest volume is " + info +"$"
    return bot_message

def respond_realTimePrice(company):
    aapl = Stock(company, token = "pk_05a3d93860d54ae4a2c0f9a01ed362d5")
    info = str(aapl.get_quote()['iexRealtimePrice'])
    bot_message =  "Its real time price is " + info +"$"
    return bot_message

def respond_openPrice(company):
    aapl = Stock(company, token = "pk_05a3d93860d54ae4a2c0f9a01ed362d5")
    info = str(aapl.get_quote()['open'])
    bot_message =  "Its open price is " + info +"$"
    return bot_message

def respond_closePrice(company):
    aapl = Stock(company, token = "pk_05a3d93860d54ae4a2c0f9a01ed362d5")
    info = str(aapl.get_quote()['close'])
    bot_message =  "Its close price is " + info +"$"
    return bot_message
```

三 结论

通过短暂的四周学习，我学到了许多自然语言处理相关的技术，并运用这些技术制作出了一个简易的股票查询机器人。从对 AI 相关知识几乎零了解到现在学到了这么多优秀的自然语言处理技术，我感到受益匪浅。虽然我现在制作的股票机器人并不完美，但我相信通过未来更深入的学习，我会制作出更优秀的聊天机器人。

在此感谢张老师的指导，虽然只有四个星期，但张老师教授给了我许多知识，让我感受到自然语言处理的强大，并提高了使用 python 及相关技术的能力。我会继续努力学习这些技术，将他们更为熟练地掌握。