远程科研指导项目总结报告

聊天机器人项目总结

姓　名：　翁希祥

学　校：　俄亥俄州立大学

专　业：计算机与信息科学

2019 年 8月

目录

1. 项目概述

本项目通过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 Openstock Python package，构造了一个股票机器人问答系统，以便人们可以通过聊天的方式快速、精准地获取到指定股票的介绍、开盘价、成交量等信息，系统将具有一定的理解人类语言的能力，具有可重用性，并且与股票实时数据建立联系，根据问题动态地获取数据、构造答案、返回答案。

2. 背景知识

2.1 自然语言处理

自然语言处理（Natural Language Processing）是人工智能领域中的一个研究方向，它被用于处理人类的语言。语言是人类区别其他动物的本质特性。在所有生物中，只有人类才具有语言能力。人类的多种智能都与语言有着密切的关系。人类的逻辑思维以语言为形式，人类的绝大部分知识也是以语言文字的形式记载和流传下来的。因而，NLP 是人工智能的一个重要，甚至核心的部分。语言具有模糊、上下文相关和隐晦的特性，甚至有时并不准确，同时人类的语言是基于人类认知的，而在表达时，同一个意思还可以有多种不同的表达方式。所以让机器处理人类的语言是非常困难的，而自然语言处理技术就旨在解决相关的问题。

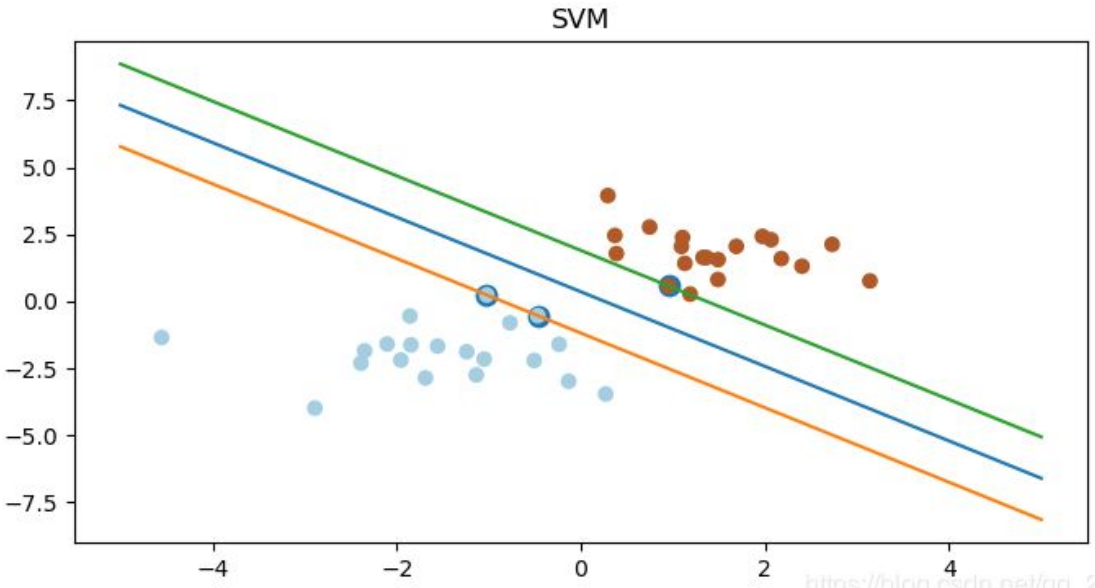
2.2 支持向量机 (SVM)

分类作为数据挖掘领域中一项非常重要的任务，它的目的是学会一个分类函数或分类模型(或者叫做分类器)，而支持向量机本身便是一种监督式学习的方法，它广泛的应用于统计分类以及回归分析中。

支持向量机（SVM）是90年代中期发展起来的基于统计学习理论的一种机器学习方法，通过寻求结构化风险最小来提高学习机泛化能力，实现经验风险和置信范围的最小化，从而达到在统计样本量较少的情况下，亦能获得良好统计规律的目的。

通俗来讲，它是一种二类分类模型，其基本模型定义为特征空间上的间隔最大的线性分类器，即支持向量机的学习策略便是间隔最大化，最终可转化为一个凸二次规划问题的求解。

图 1 SVM图解



2.3 Rasa

Rasa 是一个用于自动文本和基于语音的对话的开源机器学习框架，支持多种语言，内置支持 Spacy、MITIE、Jieba 等多种开源 NLP 工具，可将各种组件进行组合配置，构造定制化的 NLP Pipeline 以便适合需求。

Rasa 配套了 Rasa NLU 和 Rasa Core，Rasa NLU 用于实体提取和意图识别；Rasa Core 是会话引擎，它用真实的对话文本和机器学习算法训练得到模型，从而预测在收到消息后机器人该作何回复（动作）。

Rasa 有封装好的服务器 Server 程序，运行 Server 后，可以使用 Http API 来调用功能。

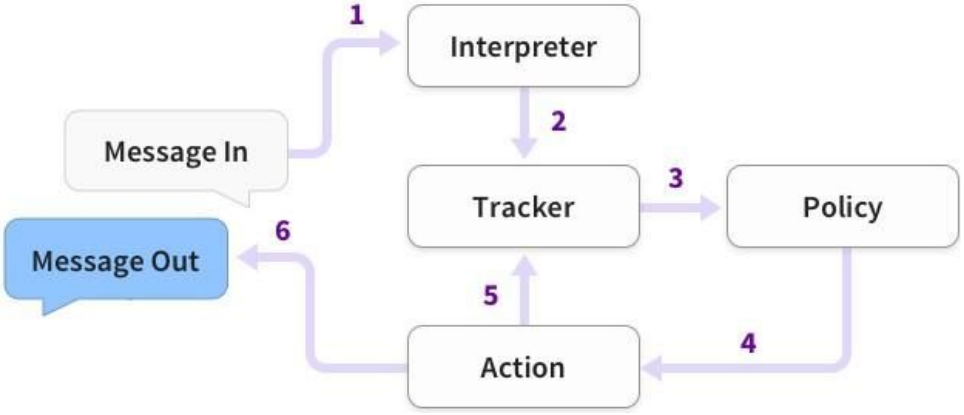


图 2 Rasa 架构机器人消息响应流程

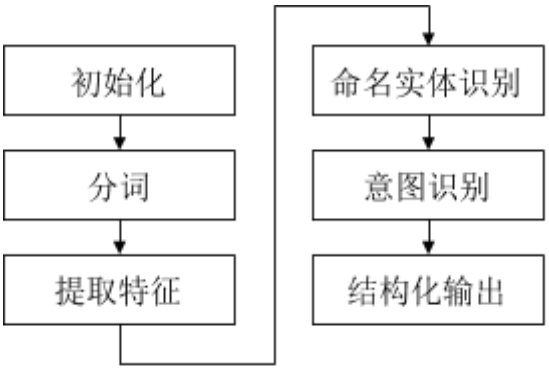


图 3 Rasa NLU Pipeline 流程

3. 过程分析

3.1 获取数据

考虑到目标是要做一个英文的股票行情的问答系统，所以需要去搜集开盘价、股票代码等有关的数据， 按照老师上课所给的使用iexfinance的API的示例来进行查询实时股票价格，交易量，开盘价等各种信息的方法，我虽然能成功的安装，但是在调取API的时候发现有些API无法调取成功，考虑到我在美国的缘故，应该不是网络IP所造成的需要翻墙问题。

```
In [7]: a.get_balance_sheet()
---------------------------------------------------------------------------
IEXQueryError                             Traceback (most recent call last)
<ipython-input-7-0a6a2c468a21> in <module>
----> 1 a.get_balance_sheet()

~/gitsrc/wxpydemo/mypy3/lib/python3.6/site-packages/iexfinance/stocks/base.py in get_balance_sheet(self, **kwargs)
    191             return pd.DataFrame(data)
    192
--> 193         return self._get_endpoint("balance-sheet", fmt_p=fmt_p, params=kwargs)
    194
    195     def get_book(self):

~/gitsrc/wxpydemo/mypy3/lib/python3.6/site-packages/iexfinance/stocks/base.py in _get_endpoint(self, endpoint, params, fmt_p, f
    101         self.endpoints = [endpoint]
    102
--> 103         data = self.fetch(fmt_j=fmt_j, fmt_p=no_pandas)
    104         # IEX Cloud returns multiple symbol requests as as a list of dicts
    105         # so convert to dict of dicts

~/gitsrc/wxpydemo/mypy3/lib/python3.6/site-packages/iexfinance/base.py in fetch(self, fmt_p, fmt_j)
    207         """
    208         url = self._prepare_query()
--> 209         data = self._execute_iex_query(url)
    210         return self._output_format(data, fmt_j=fmt_j, fmt_p=fmt_p)
    211

~/gitsrc/wxpydemo/mypy3/lib/python3.6/site-packages/iexfinance/base.py in _execute_iex_query(self, url)
    163                 return self._validate_response(response)
    164             time.sleep(self.pause)
--> 165         return self._handle_error(response)
    166
    167     def _handle_error(self, response):

~/gitsrc/wxpydemo/mypy3/lib/python3.6/site-packages/iexfinance/base.py in _handle_error(self, response)
    176                 raise auth_error(auth_msg)
    177             else:
--> 178                 raise IEXQueryError("The query could not be completed. "
    179                                     "There was a client-side error with your "
    180                                     "request.")
```

图4 iexfinance官网exampleAPI无法调用

```
In [10]: appl.get_company_name
Out[10]: <bound method Stock.get_company_name of <iexfinance.stocks.base.Stock object at 0x11cf73c50>>

In [11]: appl.get_company_name()
Out[11]: 'Apple, Inc.'

In [12]: appl.get_company()
Out[12]:
{'symbol': 'AAPL',
 'companyName': 'Apple, Inc.',
 'exchange': 'NASDAQ',
 'industry': 'Telecommunications Equipment',
 'website': 'http://www.apple.com',
 'description': 'Apple, Inc. engages in designing, manufacturing, and marketing of mobile communication, media devices, perso
 operates through the following geographical segments: Americas, Europe, Greater China, Japan, and Rest of Asia Pacific. The
e Europe segment consists of European countries, as well as India, the Middle East, and Africa. The Greater China segment com
sia Pacific segment includes Australia and Asian countries. Apple was founded by Steven Paul Jobs, Ronald Gerald Wayne, and S
ed in Cupertino, CA.',
 'CEO': 'Timothy Donald Cook',
 'securityName': 'Apple Inc.',
 'issueType': 'cs',
 'sector': 'Electronic Technology',
 'employees': 132000,
 'tags': ['Electronic Technology', 'Telecommunications Equipment']}
```

图5 部分API可以调用而部分无法返回正常结果

数次尝试无果后，我发现了一个免费的可以代替iexfinance的python包，openstock，发现能很好地应用于我的项目中。只需用一个自带的函数即可返回所有我所需要整合进机器人的信息。

```python
# 获取实时行情
def get_stock_quote(stock_code):
  stock_quote = stock_client.get_stock_quote(query=[stock_code])
  return stock_quote
```

图6 获取实时行情

3.2 制作训练集

老师上课时使用的是JSON各式来标注训练数据中的实体，意图。

但我以前对JSON并不熟悉，由于我的时间比较紧，我就选用了一种我已经使用过很多次，比较熟悉的Markdown格式来进行训练数据的标记。

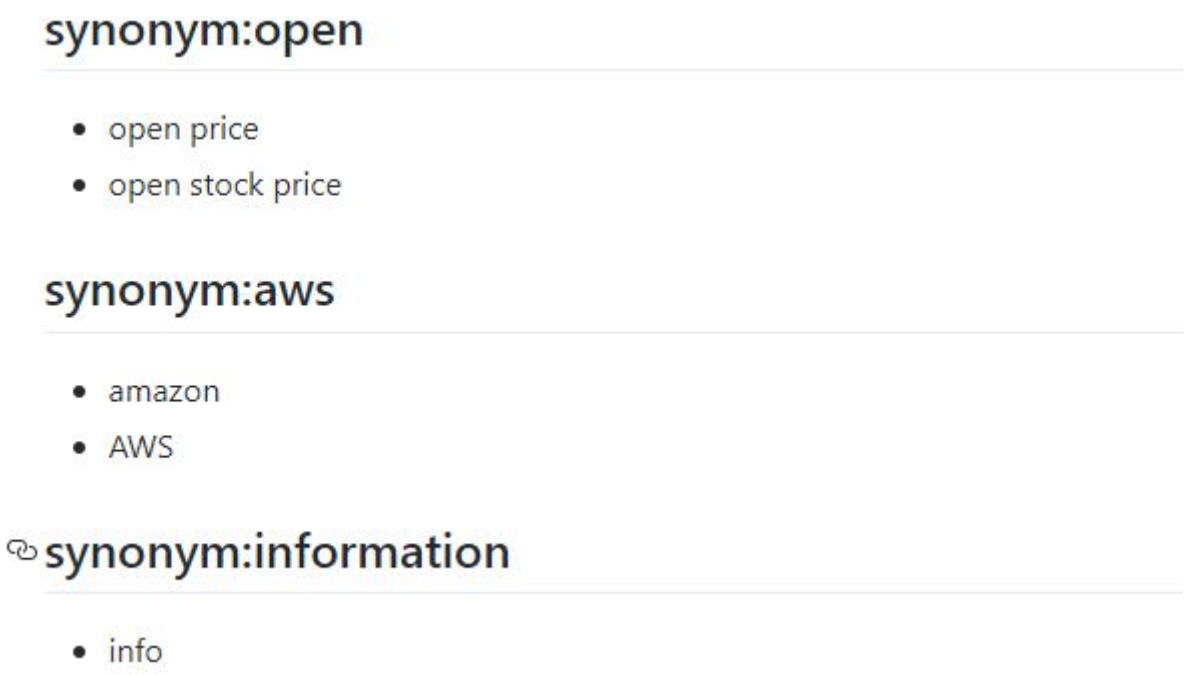


图7 部分Markdown格式

3.3 Rasa 训练数据

要让电脑对人类的语言做出回应，就需要使用Rase对训练数据及进行学习，这样机器人才能正确地理解用户的意图以及用户发送信息中的实体

```
def train(config_file,train_data):
  import warnings
  warnings.filterwarnings('ignore')
  # 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_file))

  # Load the training data
  training_data = load_data(train_data)

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

图 8 使用Rasa训练数据

3.4 与Telegram集成

首先我必须注册一个Telegram的官方机器人，同事挥霍的一个Token用于我接下来调用Telegram API的操作。

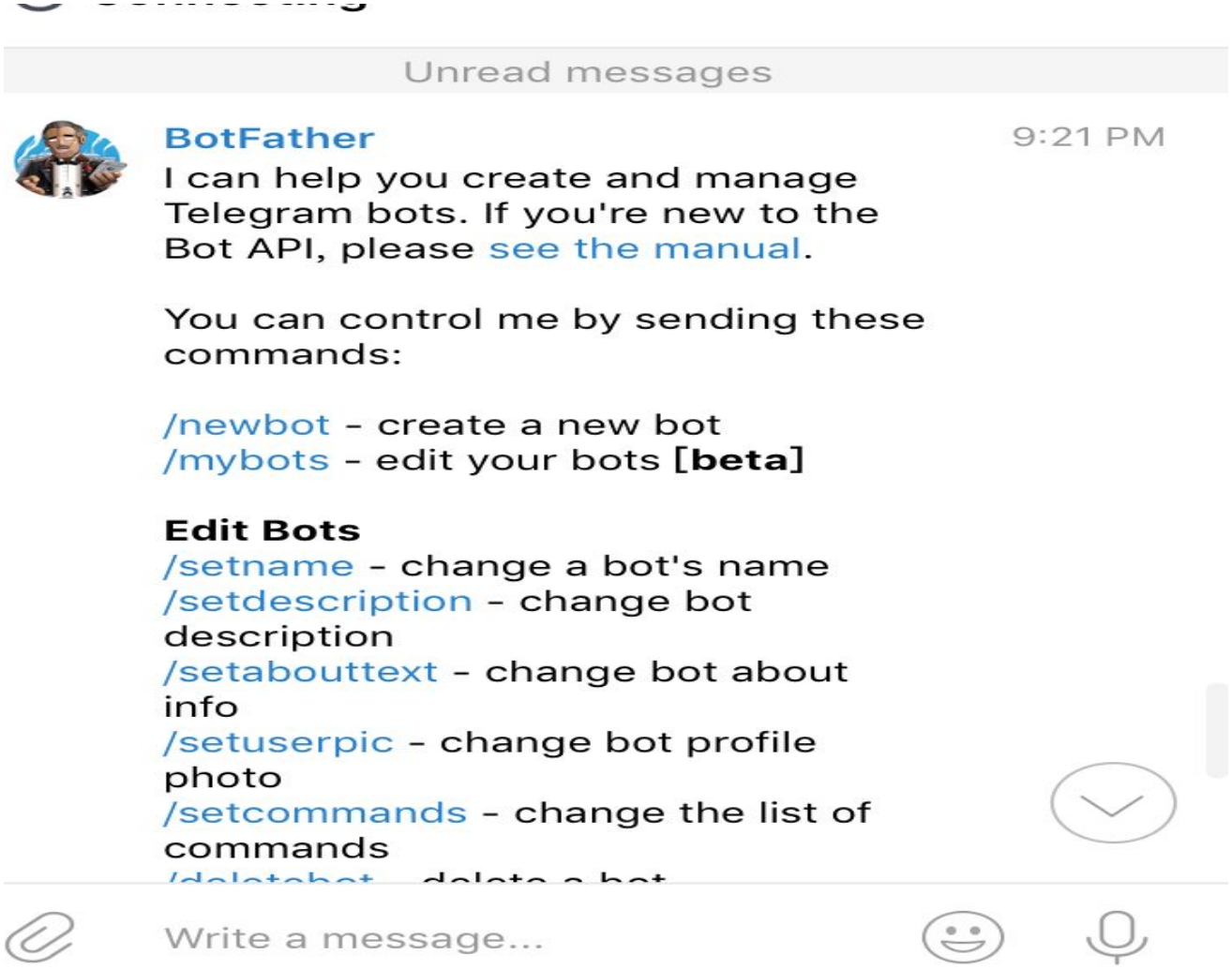


图 9 与Telegram中的BotFather对话以获得Token

第二步，我考试调用API来启动我的Telegram 机器人，然而在import Updater的时候我就遇到了问题。

```
>>> updater = Updater(token='965249366:AAHg3IIvjJJ0EwlOV-9qDYbh3A5WNwAQeg0', use
_context=True)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: __init__() got an unexpected keyword argument 'use_context'
>>>
```

图 10 import Updater时发生的奇怪问题

经过我仔细检查，确定之前的操作都没问题时，我重新阅读了一遍Telegram的API教程

，发现我的Bot版本是11.1.0，与最新的API不兼容，需要参照旧的API。

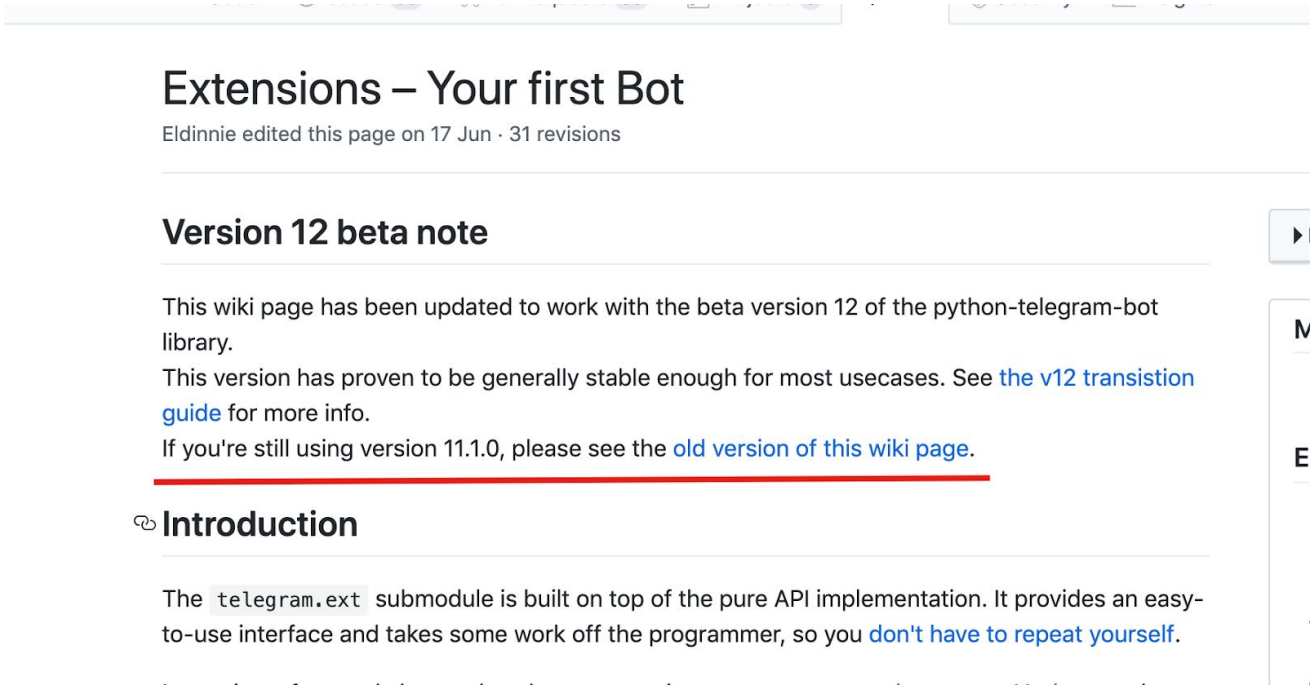


图 10 Bot与当前API不兼容

```
from telegram.ext import Updater
updater = Updater(token='TOKEN', use_context=True)
```

图 11 新API

First, you have to create an `Updater` object. Replace `'TOKEN'` with your Bot's API token.

```
from telegram.ext import Updater
updater = Updater(token='TOKEN')
```

图 12 应当使用的API

最后，我需要实现的是在Telegram里面，拿到用户输入的文本，传入我写好的response

函数，在取出新的response，写到界面回应用户输入的话，然后用户就可以根据这个回

应，再次输入。这样就完成了老师所要求的多轮多次查询。

```
class StateMachine(object):
    def __init__(self):
        self.state = 0
    def get_state(self):
        return self.state
    def set_state(self, new_state):
        self.state = new_state

stateMachine = StateMachine()
def echo(bot, update):
    input_msg = update.message.text
    logging.info(input_msg)
    # respond(policy,state,message)
    new_state, response = respond(policy_rules, stateMachine.get_state(), input_msg)
    stateMachine.set_state(new_state)
    bot.send_message(chat_id=update.message.chat_id, text=response)
```

图 13 多轮多次查询的衔接函数

```
class StockMemory(object):
    def __init__(self):
        self.remembered_stock_code = None
        self.remembered_item = None

    def update_remembered_code(self,stock_code):
        self.remembered_stock_code = stock_code

    def get_remembered_code(self):
        return self.remembered_stock_code


    def update_remembered_item(self,query_item):
        self.remembered_item = query_item

    def get_remembered_item(self):
        return self.remembered_item
```

图 14 用于记忆会话上下文中的股票信息

当然，为了应对用户可能发生的不同的多轮操作情况，我设置了状态机来指示机器人应该选择什么工作方式。

```
INIT = 0
ROBOT_INTRO = 1
ASK_COMPANY = 2
ASK_ITEM = 3
ASK_RANDOM = 4
THANK = 5


policy_rules = {
    (INIT, "none"): INIT,
    (INIT, "greet"): ROBOT_INTRO,
    (ASK_RANDOM, "greet"): ASK_COMPANY,
    (ROBOT_INTRO, "specify_company"): ASK_COMPANY,
    (ASK_COMPANY, "specify_company"): ASK_ITEM,
    (ASK_COMPANY, "query_item"): ASK_RANDOM,
    (ASK_ITEM, "query_item"): ASK_RANDOM,
    (ASK_RANDOM, "query_item"): ASK_RANDOM,
    (ASK_RANDOM, "query_item"): ASK_RANDOM,
}
```

图 15 机器人状态机设置

同时，为了让机器人更加智能，我加入了一段代码是的机器人可以在结合训练数据的前提下，能对用户所发送的具有否定含义的信息进行很好地回复。

```
import re
negated_patterns = [
    re.compile(".* prefer (?:the)?\s*(\w+) to (?:the)?\s*(\w+)"),
    re.compile(r"(\w+) instead of (\w+)"),
    re.compile(".* (\w+), not (?:the)?\s*(\w+)")
]

def negated_ents(phrase):
    for pattern in negated_patterns:
        group = pattern.search(phrase)
        if group and (len(group.groups()) >= 2):
            return True,group.groups()

    return False,None
```

图 15 上下文否定式对话

4. 结果展示

4.1 可用功能

意图	含义	示例
greet	与机器人打招呼	hi，good evening
stock_search	查询股票信息	tell me something about jd company
stock_search	暂停当前查询，改为查询另外的股票（包含否定意图的识别）	Sorry, I changed for searching alibaba, not the jd
stock_search	查询开盘价	what about the open price
stock_search	查询成交量	tell me today's volume
search_search	查询公司市值	how about market cap

4.2 前端

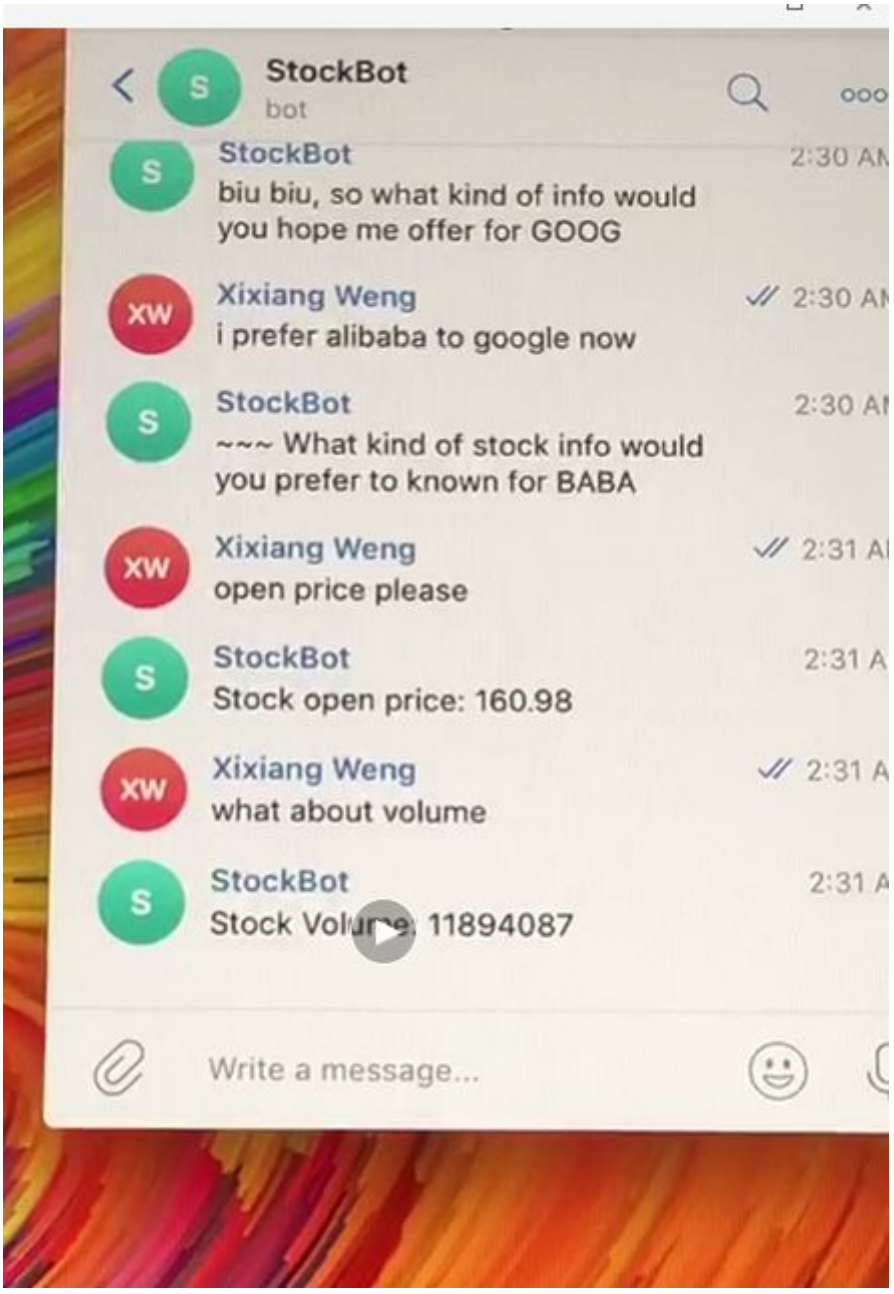


图 16 Telegram前端展示

5. 感想与收获

本次“聊天机器人”远程指导项目为期一个月，虽然“自然语言处理”的字眼已在我的学习生活中出现多次，但我一直没有实际地去自己做一个NLP的项目。

在参与项目过程中，老师讲授了自然语言处理的发展和应用场景、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

对话等知识点，上课时我还觉得这些东西很容易理解，但真正自己做起来的时候却也遇到了很多困难。

首先就是通过API获取股票信息的环节，因为老师上课时已经向我们演示了如何使用iexfinance获得股票信息，但可能每个人的电脑配置环境不一样，我急事按照老师的步骤一步步配置，却还是遇到了有一些API无法调用的问题，经过数次尝试病向老师请教解决方法以及无法解决的时候，我心情十分差，一度想推到重来重新尝试别的领域的API，如老师提及过的百度地图。但我突然想到能够查询股票信息的python包一个不会只有这一种，于是我在google上反复搜索，终于在一个github上找到了能够代替iexfinance的python包，openstock，并且发现它有一个get_quote函数能一次性返回所有我所需的信息。

解决了这个问题后，我开始思考如何配置训练集，由于老师上课时向我们展示的都是以JSON格式对训练数据进行标注，但我本人对JSON恨不熟练，我按照老师的格式照葫芦画瓢，但发现行不通。于是我就上网google我自己熟悉的Markdown能不能对训练数据进行标注的工作。大难让我十分高兴，是可以的。

之后，我在集成代码到聊天软件的时候也遇到了一些问题，一开始，我想集成到微信，但是发现扫码总是无效，由于我本人只有一个微信号，父母的微信号也应为一些原因不好弄过来当bot，于是我就按照老师给的第二条路，选择Telegram进行集成，虽然过程中也遇到了一些问题，但都能解决。最终一个能在聊天软件上运行，能够完成多轮对话，否定意图识别的股票查询机器人呈现在了我的眼前。虽然训练数据量不是很多导致有一些话Bot识别不是那么准确；状态机我考虑的情况也没有达到无懈可击的地步。但是这个完整的聊天机器人让我加深了老师上课时讲到的那些知识点的印象。让我对这一个月里学的知识有一个完整的体系认识。

最后，我的感想就是计算机的学习一定要自己上手完成才能真正算是学会了这个知识。虽然上课时老师讲的概念我好想很容易就理解了，但真正要自己用的时候才发现有很多问题自己之前都没有考虑到，知识以为自己懂了而已。

最后的最后，我还是要感谢老师这一个月来的教学，让我对NLP这个方向有了一个全面系统的认识。我现在对siri这类聊天机器人也没有之前那种不可思议的想法了。以后有机会，我还想在这个领域研究下去。