НИТУ «МИСиС»

Институт ИТАСУ

Кафедра инженерной кибернетики

Направление подготовки: 01.03.04 «прикладная математика»

Квалификация (степень): бакалавр

КУРСОВАЯ РАБОТА

по учебной дисциплине «Методы искусственного интеллекта»
VIII семестр 2020 у.г.

Студент: Малынковский О.В.

Группа: БПМ-16-2

Преподаватель: доц., к.т.н. А.С. Кожаринов

Оценка:

Дата:

Москва 2020

Оглавление

Постановка задачи

Вопросно-ответная система (QA-система) — информационная система, способная принимать вопросы и отвечать на них на естественном языке, другими словами, это система с естественно-языковым интерфейсом.

Необходимо создать прототип такой системы. Разрабатываемое пользовательское приложение относится к предметной области «изобразительное искусство», поэтому создаваемая система является узкоспециализированной. Например, система должна давать информацию о каких-то фактах из биографии известных художников.

Диалог с пользователем должен происходить на русском языке.

Использованные средства разработки и системные требования

Наиболее удобным и подходящим средством для развертывания такой системы является язык Python. Исходя из такой задачи средством обработки пользовательских запросов и поиск ответов по базе выбираем библиотеку cdQA и NLTK. В качестве источника знаний для создания базы знаний будем использовать статьи из Википедии.

Приложение должно работать в Windows 10.

Создание вопросно-ответной системы

Архитектура

Современные QA-системы обычно включают особый модуль — **классификатор вопросов**, который определяет тип вопроса и, соответственно, ожидаемого ответа. После этого анализа система постепенно применяет к предоставленным документам все более сложные и тонкие методы NLP (обработка на естественном языке), отбрасывая ненужную информацию. Самый грубый метод — **поиск в документах** — предполагает использование системы поиска информации для отбора частей текста, потенциально содержащих ответ. Затем **фильтр** выделяет фразы, похожие на ожидаемый ответ (например, на вопрос «Кто …» фильтр вернет кусочки текста, содержащие имена людей).

Представление знаний и поиск ответов

Производительность вопросно-ответной системы зависит от эффективности используемых методов анализа текстов и от качества текстовой базы — если в ней нет ответов на вопросы, QA-система мало что сможет найти. Чем больше база — тем лучше, но только если она *содержит* нужную информацию. Большие хранилища (такие как Интернет) содержат много избыточной информации. Это ведёт к следующим моментам:

1. Для получения знаний по предметной области выгрузим статьи по теме из Википедии помощью API интерфейса данной платформы. Далее производится **токенизация** текста, то есть разбиение длинных строк текста в более мелкие: абзацы делим на предложения, предложения на слова. **Нормализация** — серия операций, в результате которых текст приводиться к нормализованному виду: все слова приводятся к одному регистру, удаляются знаки пунктуации, расшифровываются сокращения, числа приводятся к их текстовому написанию и т.д. Нормализация необходима для унификации методов обработки текста.
2. **Стеммизация** — устранение придатков к корню, то есть отделение суффикса, приставки, окончания. **Лемматизация** — близка к стеммизации. Отличие в том, что приводит слово к смысловой канонической форме слова (инфинитив для глагола, именительный падеж единственного числа — для существительных и прилагательных). Это более сложная операция. Например: зафрахтованный — фрахтовать, ценами — цена, лучший — хороший.
3. **Part-of-Speech tagging** — это процесс назначения тега токенизированной части предложения. Наиболее популярная разметка определяет слова как имена существительные, прилагательные, глаголы и другие части речи.

4. **Statistical Language Modeling** (статистическое моделирование языка) — процесс построения статистической модели языка, целью которой является создать текст, максимально близкий к натуральной речи. Математически SLM — это распределение вероятности появления строки S как целого предложения.
5. **«Сумка слов» (bag of words)** — это детальная репрезентативая модель, используемая для упрощения обработки содержания выделенного текста. Эта модель не берет во внимание грамматику или порядок слов. Главная задача — определение количества вхождений слов в данный текст.
6. **n-граммы.** Это другая модель для упрощения распознавания содержание текста. В отличии от моделей, где не учитывается порядок слов, n-граммная модель определяет и сохраняет смежные последовательности слов в тексте.

Описание клиентского приложения

Общие сведения

Разработан прототип вопросно-ответная система (QA-система) для предметной области «изобразительное искусство». Например, система должна давать информацию о каких-то фактах из биографии известных художников. Диалог с пользователем осуществляется на русском языке.

Описание экранных форм

При запуске приложения программа приветствует пользователя и приглашает к заданию вопросов. Открывается форма для ввода интересующего вопроса.

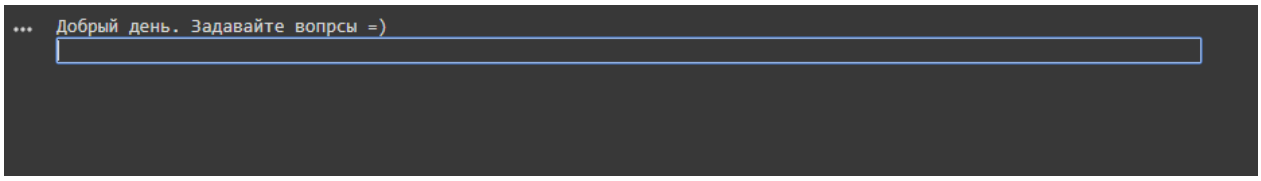

Рисунок 1

После ввода вопроса нужно нажать клавишу Enter и начнется поиск ответа. Далее система выведет найденный ответ на экран.

Рисунок 2

К сожалению, система не всегда способна найти нужный ответ, даже если он содержится в базе знаний. На рис.3 видно, что есть и неправильные ответы. Иногда этого связано с тем, что сами знания на английском языке и запросы пользователя могут быть переведены неточно. Также причина может быть в необученности системы работы с запросами на естественном языке.

```
Ответ:Карл Брюллов
Кто нарисовал Утро в сосновом Лесу?
Ответ:Савицкий
В каком году родился Шишкин*
Ответ:1873
Когда родился Шишкин ?
Ответ:1820
В каком году родился Иван Шишкин?
Ответ:1897
Какое имя у Шишкина?
Ответ:Иван Шишкин
В каком городе родился Иван Шишкин?
Ответ:Санкт-Петербург
В каком году Шишкин получил звание профессора?
Ответ:1863
Кто написал картину "Лесная глушь" ?
Ответ:Илья Репин
Какую награду Репин получил за эскиз «Ангел смерти избивает всех перворожденных египтян»:
Ответ:В 1863 году он был награжден двумя серебряными медалями первого класса
Куда отправился Репин летом 1870 года вместе с братом ?
Ответ:Юрьевец
Пока
```

Рисунок 3

Чтобы завершить диалог нужно ввести «Пока»

Выводы

В рамках данной курсовой работы были исследованы принципы работы вопросно-ответных систем. Реализовано пользовательское приложение в формате диалога на русском языке. В ходе тестирования были обнаружены вопросы, на которые система дает неправильные ответы. Если в базе знаний на самом деле есть нужный ответ, то это связано с тем, что сами знания на английском языке и запросы пользователя могут быть переведены неточно. Также причина может быть в необученности системы работы с запросами на естественном языке.

Приложения

Код приложения на Python

```
pip install cdqa

pip install googletrans

import pandas as pd

from ast import literal_eval


from cdqa.utils.filters import filter_paragraphs

from cdqa.utils.download import download_model, download_bnpp_data

from cdqa.pipeline.cdqa_sklearn import QAPipeline

from nltk import sent_tokenize, word_tokenize

basedir = '/content/test'

import os

import nltk

nltk.download('punkt')

files=pd.DataFrame()

files['title'] = os.listdir(basedir)

files['paths']=' '

files['paths'] = [os.path.join(basedir, name) for name in files['title']]

files['paragraphs']=' '

files['paragraphs']=[open((i), 'r', encoding="utf8", errors='ignore').read().strip() for i in files['paths'] ]

files.head()

files['paragraphs']=files['paragraphs'].apply(lambda x:sent_tokenize(x) )

files.to_csv('new_dataset.csv')

df=pd.read_csv('new_dataset.csv',converters={'paragraphs': literal_eval})

new_df=df.head(n=3)
```

```
from cdqa.pipeline import QAPipeline
import wget
url = "https://github.com/cdqa-suite/cdQA/releases/download/bert_qa/bert_qa.joblib"
wget.download(url, 'bert_qa.joblib')
cdqa_pipeline = QAPipeline(reader='bert_qa.joblib')
cdqa_pipeline.fit_retriever(df=df)
query = 'In which Academy Viktor Vasnetsov studied ?'
prediction = cdqa_pipeline.predict(query)
prediction
print('query: {}\n'.format(query))
print('answer: {}\n'.format(prediction[0]))
print('title: {}\n'.format(prediction[1]))
print('paragraph: {}\n'.format(prediction[2]))
from googletrans import Translator
translator = Translator()
#print(translator.translate('В каком году родился Л?', dest='en',src='ru').text)
query = translator.translate('Кто рисовал в стиле романтизма ?', dest='en',src='ru').text
prediction = cdqa_pipeline.predict(query, n_predictions=5)
prediction
print(translator.translate(prediction[0][0], dest='ru',src='en').text)
flag=True
#site index
print('Добрый день. Задавайте вопрсы =)')
#User introduces
#user_name=input()
#print('Edulexa: Hello ',user_name)

while(flag==True):
    i=0
```

```
    #taking user question
    query = input()
    #query=query.lower()

    if('Пока' not in query.split(' ')):
        if(query=='thanks' or query=='thank you' ):
            flag=False
            print("Edulexa: You are welcome..")
        else:
            query_eng = translator.translate(query, dest='en',src='ru').text
            prediction = cdqa_pipeline.predict(query_eng)
            print("Ответ:"+translator.translate(prediction[0], dest='ru',src='en').text)
    elif ('Пока' in query.split(' ')):
        flag=False
```

Получение данных

```
pip install wget
pip install pywikibot
# Pywikibot needs a config file
pywikibot_config = r"""# -*- coding: utf-8  -*-
mylang = 'en'
family = 'wikipedia'
usernames['wikipedia']['en'] = 'test'"""
with open('user-config.py', 'w', encoding="utf-8") as f:
    f.write(pywikibot_config)
import pywikibot
from pywikibot import pagegenerators
pip install mwparserfromhell
```

```
text = ""
site = pywikibot.Site('en', 'wikipedia')
cat = pywikibot.Category(site,'Category:19th-century Russian painters')
gen = pagegenerators.CategorizedPageGenerator(cat, content = True)
i=0
file_n = "Eng19_rus_data" + ".txt"
for page in gen:
    print(page.title)
    i += 1
    text += page.text
with open(file_n, 'w') as file:
  file.write(text)
import mwparserfromhell
full = mwparserfromhell.parse(text)
stripped = full.strip_code()
with open("temptext.txt", 'w') as file:
  file.write(stripped)
```