دانشگاه قم
University of Qom

گزارش فنی پروژه‌ی پایانی بازیابی اطلاعات

با موضوع: پیاده‌سازی یک موتور جستجوی ساده با داده‌های وبسایت خبری " خبر آنلاین "

مجتبی نبوی

شماره دانشجویی: ۹۷۵۳۰۸۸۰۰۶

استاد راهنما: آقای حسین امیرخانی

خرداد ۱۴۰۱

چکیده

در این پروژه یک موتور جستجوی ساده با داده‌های وب سایت خبری "خبر آنلاین" و با زبان پایتون پیاده‌سازی گردید. با توجه به این موضوع که داده‌ها بطور آماده در دسترس نبودند، در ابتدا یک خزنده نوشته شد که داده‌های لازم برای ادامه‌ی فرایند را جمع‌آوری کند. داده‌های جمع‌آوری شده، اخبار ۴ ماه ابتدای سال ۱۴۰۰ با تعداد ۲۵۴۱۱ خبر و دارای ۹ ویژگی و با حجمی حدود ۱۰۰ مگابایت می‌باشد. پس از برخی پردازش‌ها روی این داده‌ها، یک پرس و جو به برنامه داده می‌شود و براساس آن تعدادی اخبار مرتبط بعنوان پاسخ نمایش داده می‌شوند.

خزش و جمع‌آوری داده‌ها

برای جمع‌آوری داده‌ها، ابتدا صفحه‌ی آرشیو خبرها را پیدا کرده و در آن بدنبال الگویی برای خزش بودیم. یکی از الگوهای بسیار خوب که از آن در این پروژه استفاده گردید در تصویر ۰ در ادامه آمده است.

```
archive_url_template = f'https://www.khabaronline.ir/page/archive.xhtml?date={date}&pi={page}'
```

تصویر ۰ - الگوی بدست آمده برای شروع خزش و جمع آوری آدرس خبرها

در این الگو می‌توان با دادن یک تاریخ شمسی با کمک کتابخانه‌ی jdatetime (برای تبدیل تاریخ میلادی به شمسی)، خلاصه‌ی اخبار آن روز را مشاهده و در صفحات آن حرکت کرد. در جمع‌آوری داده‌های این پروژه تمام تاریخ‌های بین ۱۴۰۰/۰۱/۰۱ تا ۱۴۰۰/۰۴/۳۱ استفاده شد و در نهایت، آدرس تمام اخبار آن هر روز با حرکت در تمام صفحات و با کمک کتابخانه‌های requests و beautifulSoup بدست آمد.

در ادامه، به تمام آدرس‌های جمع‌آوری‌شده در مرحله‌ی قبل مراجعه می‌کنیم و داده‌های مورد نیاز خود را از بخش‌های مختلف هر سند، مانند متن و ابرداده‌ها (metadata) با استفاده از کتابخانه‌ی newspaper استخراج و ذخیره می‌کنیم.

با توجه به امکان بروز مشکلات در فرایند جمع‌آوری وذخیره‌ی داده‌ها مانند قطعی برق، اینترنت و پرشدن تمام فضای حافظه‌ی اصلی (ram)، ذخیره‌سازی داده‌ها، پس از جمع‌آوری کامل داده‌های یک روز انجام و پس از آن حافظه‌ی میانجی (buffer) برای داده‌های سایر روزها خالی می‌شود. کدهای خزنده (crawler) در تصویر ۱ آمده است.

```python
collected_data = []
main_url = 'https://www.khabaronline.ir'

def get_html_content(date, page):

  date = date.strftime('%Y-%m-%d')
  page_url = f'{main_url}/page/archive.xhtml?date={date}&pi={page}'

  return requests.get(url= page_url).text

def get_article_links(date, page):

  html_content = get_html_content(date= date, page= page)

  html_parser = BeautifulSoup(markup= html_content, features= 'html.parser')

  html_a_links = html_parser.select(selector= 'ul li.news h3 a')

  article_links = []

  for a_link in html_a_links:

    article_links.append(a_link['href'])

  return article_links

def parse_article_content(url):

  article_url = f'{main_url}{url}'

  article = Article(url= article_url, language= 'fa')

  article.download()

  article.parse()

  article_content = {
    'url': article.url,
    'id': article.meta_data['nastooh']['nid'],
    'title': article.title,
    'image': article.top_img,
    'summary': article.meta_description,
    'text': article.text,
    'tags': article.tags,
    'publish': article.publish_date,
    'keywords': article.meta_keywords,
  }

  return article_content

def save_collected_data(current_date, from_date, to_date):

  try:

    df = pandas.DataFrame(data= collected_data)

    from_date = from_date.strftime("%Y-%m-%d")
    to_date = to_date.strftime("%Y-%m-%d")

    data_file_name = f'khabaronline-{from_date}-{to_date}.csv'

    df.to_csv(data_file_name, mode='a', index=False, header=False)

    print(f'\n\nAll colleted data in date: {current_date} has been saved.\n\n')

  except:
    print(f'\n\nSomething went wrong when trying to save colleted data in date: {current_date}. \n\n')

def crawl_khabaronline(from_date, to_date):

  page = 0
  current_date = from_date

  while True:

    try:

      page += 1
      print(f'Collecting articles in date: {current_date.strftime("%Y-%m-%d")} and page: {page}:')
      article_links = get_article_links(date= current_date, page= page)

      if len(article_links) != 0:

        for article_url in article_links:

          try:
            article_content = parse_article_content(url= article_url)
            print(f'\tParsed article with id: {article_content["id"]}')
            collected_data.append(article_content)
          except:
            print(f'Something went wrong when trying to parse article with (article_url:{article_url}) parameters.')
            continue

      else:
        if current_date == to_date:
          break
        else:
          page = 0
          save_collected_data(current_date= current_date, from_date= from_date, to_date= to_date)
          current_date = current_date + jdatetime.timedelta(days= 1)
          collected_data.clear()

    except:
      print(f'Something went wrong when trying to get article links with (date:{current_date.strftime("%Y-%m-%d")}, page: {page}) parameters.')

      continue
```

تصویر ۱ - کدها و توابع خزنده

نحوه‌ی استفاده از توابع و کدهای خزنده

همان‌طور که در تصویر ۱ مشاهده کردید، برای اعمال مختلف، توابعی در نظر گرفته شده است. این توابع، بطور عادی کاری انجام نمی‌دهند و باید براساس نیاز خود آن‌ها را فراخوانی کنیم. برای شروع عمل خزش و ذخیره‌ی داده‌ها باید بصورت زیر عمل می‌کنیم:

```
from_date = jdatetime.date(year= 1400, month= 1, day= 1)
to_date = jdatetime.date(year= 1400, month= 4, day= 31)

crawl_khabaronline(from_date= from_date, to_date = to_date)
```

تصویر ۲ - نحوه‌ی استفاده از خزنده

ویژگی داده‌های جمع آوری شده

با استفاده از خزنده‌ی نوشته شده، تعداد ۲۵۴۱۱ خبر دارای ۹ ویژگی و با حجمی حدود ۱۰۰ مگابایت، از تاریخ ۱۴۰۰/۰۱/۰۱ تا ۱۴۰۰/۰۴/۳۱ جمع‌آوری شد. در تصویر ۳، می‌توان ۴ خبر اول را در قالب یک جدول مشاهده کرد که به درک بهتر از داده‌ها کمک می‌کند. هر خبر شامل داده‌های زیر است که با کمک کتابخانه‌ی newspaper استخراج شده‌اند.

۱. آدرس خبر
۲. آدرس تصویر خبر
۳. شناسه
۴. عنوان
۵. خلاصه
۶. متن اصلی
۷. تاریخ انتشار
۸. تگ‌ها
۹. کلمات کلیدی

	url	id	title	image	summary	text	tags	publish	keywords
0	https://www.khabaronline.ir/news/1497657/%D8%A...	1497657	ایران اکنون» به» روایت شهروند گراوسی و همسر ا...	https://media.khabaronline.ir/d/2021/03/21/4/5...	ایران اکنون»،» روایت مصور تمدن، جاذبه‌های تار...	به گزارش خبرگزاری خبرآنلاین، کتاب «ایران اکنو...	عکس', 'عکاسی',', 'معرفی کتاب', '[بازار کتاب	2021-03-21 20:28:00+00:00	بازار کتاب',', 'عکاسی', 'معرفی کتاب', '[کتاب', 'عکس
1	https://www.khabaronline.ir/news/1497694/%D8%A...	1497694	تصاویر	ادای احترام مردم به آرامگاه شجریان و ...	https://media.khabaronline.ir/d/2021/03/21/4/5...	مردم مشهد و گردشگرانی که به این شهر سفر کرده‌ا...	به گزارش خبرگزاری خبرآنلاین، در حالی که دیروز ...	محمدرضا", 'شجریان', 'مهدی اخوان ثالث', '... ', 'توس	2021-03-21 20:15:43+00:00	محمدرضا"] شجریان', 'مهدی اخوان ثالث', 'ابوالقا...
2	https://www.khabaronline.ir/news/1497692/%D8%A...	1497692	دختر و پسران سردار سلیمانی در مراسم تولد پدر ش...	https://media.khabaronline.ir/d/2020/02/16/4/5...	تصویری از حضور خانواده حاج قاسم سلیمانی در مرا...	به گزارش خبرگزاری خبرآنلاین ،امروز اول فروردین...	شهید سپهبد قاسم سلیمانی', 'ترور	2021-03-21 19:59:39+00:00	شهید سپهبد قاسم سلیمانی"] ترور
3	https://www.khabaronline.ir/news/1497683/%D8%A...	1497683	اولین تصویر از کارت ملی سردار سلیمانی	https://media.khabaronline.ir/d/2020/12/13/4/5...	پاشگاه خبرنگاران جوان نوشت: همزمان با زادروز ش...	امروز اول فروردین ماه سال ۱۴۰۰ است و زادروز نو...	ایران و آمریکا", 'شهید سپهبد قاسم سلیمانی	2021-03-21 19:41:56+00:00	شهید سپهبد قاسم سلیمانی', 'ترور', 'ایران و آ...

تصویر ۳ - چهار خبر اول از داده‌های جمع‌آوری شده

تبدیل اسناد خام به ماتریسی از ویژگی‌های TF-IDF و تشکیل واژگان

در این مرحله با استفاده از داده‌های بدست آمده، واژگان خود را ایجاد کرده و تعداد تکرار هر کلمه در اسناد را نیز بدست می‌آوریم؛ سپس برای هر یک از اسناد، براساس معیار TF-IDF یک عدد را به آن اختصاص می‌دهیم. از این عدد برای برسی میزان شباهت پرس و جوی کاربر و اسناد، توسط معیار Cosine که در یک فضای برداری، زاویه‌ی کسینوسی میان هر یک از اسناد و پرس و جوی کاربر را حساب می‌کند، استفاده می‌شود. برای ایجاد واژگان و حساب‌کردن TF-IDF برای هر سند، از کتابخانه‌ی معروف SciKit-Learn و کلاس TfidfVectorizer استفاده می‌کنیم.

```
# Reading Collected Data

data_file_path = '..../khabaronline-1400-01-01-1400-04-31.csv'

documents = pandas.read_csv(filepath_or_buffer= data_file_path)

# Tf-Idf Documents Vectorization

vectorizer = TfidfVectorizer()

vectorized_documents = vectorizer.fit_transform(raw_documents= documents['summary'])
```

تصویر ۴ - خواندن داده‌ها، ایجاد واژگان و برداری‌کردن اسناد

ویژگی واژگان ایجاد شده بر اساس اسناد

واژگان ایجاد شده در مرحله‌ی قبل دارای ۲۴۵۶۵ عبارت می‌باشد. در تصویر ۵ می‌توان ۸ عبارت اول واژگان را مشاهده کرد.

```
list(vectorizer.vocabulary_.keys())[:8]

['تاریخی', 'های', 'جاذبه', 'تمدن', 'مصور', 'روایت', 'اکنون', 'ایران']
```

تصویر ۵ - پنج عبارت اول در واژگان موتور جستجو

تبدیل پرس و جوی کاربر به ماتریسی از ویژگی‌های TF-IDF

در این مرحله نیز مانند مرحله‌ی قبل و براساس واژگان بدست آمده، برای هر یک از عبارات (term) پرس و جو TF-IDF را حساب کرده و از آن در مرحله‌ی بعد، برای بررسی شباهت پرس و جو و اسناد استفاده می‌کنیم.

```
query = 'قیمت دلار امروز'

vectorized_query = vectorizer.transform(raw_documents= [query])[0]
```

تصویر ۶ - تبدیل پرس و جوی کاربر به بردار با استفاده از واژگان بدست آمده

محاسبه‌ی میزان شباهت هر سند و پرس و جو

در این مرحله با استفاده از کلاس cosine_similarity که در کتابخانه‌ی SciKit-Learn قرار دارد؛ بررسی می‌کنیم که هر سند چقدر با پرس و جوی کاربر شباهت دارد. این شباهت در یک فضای برداری و با استفاده از اختلاف زاویه‌ی پرس و جوی کاربر با هر یک از اسناد بدست می‌آید که عددی بین ۰ تا ۱ داده خواهد بود. هر چقدر این امتیاز (عدد) بیشتر باشد، میزان شباهت نیز بیشتر خواهد شد.

```
query_document_similarities = []

for document in vectorized_documents:
  similarity = float(cosine_similarity(document, vectorized_query))
  query_document_similarities.append(similarity)
```

تصویر ۷ - محاسبه‌ی میزان شباهت هر سند و پرس و جو

انتخاب نتایج برتر و نمایش به کاربر

پس از تمام مراحل قبل، از جمع‌آوری داده تا بررسی میزان شباهت پرس و جوی کاربر با اسناد، نوبت آن رسیده است که اسناد با شباهت بیشتر که احتمالا نتایج مورد نظر کاربر نیز هستند را به او نشان دهیم. در مرحله‌ی قبل، ما میزان شباهت هر یک از اسناد را بدست آوردیم و یک امتیاز به آن‌ها اختصاص دادیم؛ اکنون باید اسناد با بیشترین امتیازها را بعنوان پاسخ به کاربر نمایش دهیم. برای این منظور مطابق تصویر ۸ عمل می‌کنیم.

```
result_count = 20
sorted_indexes = numpy.argsort(query_document_similarities)

for i in range(result_count):
  current_index = sorted_indexes[-i-1]
  current_document = documents.iloc[current_index]
  print('similarity:', query_document_similarities[current_index], 'title:', current_document['title'], 'url:', current_document['url'])
```

تصویر ۸ - انتخاب اسناد با امتیاز برتر و نمایش آن به کاربر

همانطور که در خط آخر تصویر ۸ دیده می‌شود، پاسخ شامل: میزان شباهت، عنوان و آدرس سند است. تعداد اطلاعات هر سند و نوع نمایش را می‌توان به راحتی تغییر و آن را بهبود داد.

برخی از نتایج موتور جستجو

در مثال زیر، پرس و جوی کاربر "قیمت امروز دلار" بوده و ۲۰ نتیجه‌ی زیر بدست آمده‌اند. در میان این ۲۰ سند مرتبط با پرس و جوی کاربر، بیشترین شباهت ۰.۷۰۸ یعنی حدود ۷۰ درصد و کمترین آن ۰.۴۲۷ یا به عبارتی حدود ۴۲ درصد می‌باشد.

تصویر ۹ – برخی از نتایج موتور جستجوی نوشته شده با پرس و جوی "قیمت امروز دلار"

بهینه‌سازی و افزایش سرعت

برای پیاده‌سازی موتور جستجوی معرفی شده برای وب سایت خبر آنلاین، از ساده‌ترین روش‌ها استفاده شده است و در بسیاری از موارد می‌توان آن را از ابعاد مختلف بهینه کرد. بطور مثال یکی از بهینه‌سازی ها می‌تواند در هنگام بررسی میزان شباهت اسناد و پرس و جوی کاربر صورت بگیرد؛ بگونه‌ای که این بررسی و اختصاص امتیاز شباهت برای تمام اسناد محاسبه نشود؛ زیرا قسمت بزرگی از اسناد ما به پرس و جوی کاربر ارتباطی ندارند و دخالت آن‌ها در فرایند امتیازدهی بی‌معنی است.

کدها و داده‌های استخراج شده

تمام کدهای نوشته شده و داده‌های استخراج شده، برای بررسی بیشتر در گیت‌هاب قرار داده شده‌اند که لینک آن در ادامه آمده است.

https://github.com/mojtabanabavi/Information-Retrieval-Project