Work done by Diksha Saxena from ( 8th - 19th June 2023)

* Smarana got me up to speed with the project (Social Media Image Recontextualization) objective and DARPA deadlines.
* I was asked to take up Web Scraping to collect Data from various news providers for the sub task of Recontextualization Detection.
* Here’s the box link where everything is documented and collected: (<https://buffalo.app.box.com/folder/212263692608>)

Techniques Explored:

1. BeautifulSoup
2. RSS Feeds
3. API calls and BeautifulSoup
4. Newspaper3k Library

Sources for Data:

1. The Guardian
2. Reuters
3. BBC
4. NewsCatcher

Codes:

1. Using a Developer API Key to make API calls to fetch political article data from the Guardian.

**import** requests

**from** bs4 **import** BeautifulSoup

**import** json

**from** unidecode **import** unidecode

**import** re

*# API key*

api\_key = 'ac66b0b4-572e-4810-ac72-d9e6b7237ac6'

*# URL for fetching articles*

endpoint = 'https://content.guardianapis.com/search'

*# Query parameters*

params = {

'api-key': api\_key,

'q': 'politics', *# search query*

'section': 'politics', *# section filter*

'page-size': 50, *# Number of articles per page*

'order-by': 'newest' *# Sort articles by newest*

}

*# GET request to the API*

response = requests.get(endpoint, params=params)

*# Parse the JSON response*

data = response.json()

*# Extract information*

articles = []

**for** result **in** data['response']['results']:

article\_url = result['webUrl']

article\_response = requests.get(article\_url)

article\_html\_content = article\_response.content

*# BeautifulSoup object creation*

soup = BeautifulSoup(article\_html\_content, 'html.parser')

content\_elements = soup.find\_all('p')

content = ' '.join([unidecode(element.get\_text(separator=' ')) **for** element **in** content\_elements])

*# Extract the image captions*

image\_caption\_elements = soup.find\_all('figcaption')

image\_captions = list(set([unidecode(caption.get\_text(separator=' ')) **for** caption **in** image\_caption\_elements]))

*# Cleanup- Remove excessive spaces*

image\_captions = [re.sub(r'\s+', ' ', caption) **for** caption **in** image\_captions]

*# Dictionary to store the scraped data*

article = {

'headline': unidecode(result['webTitle']),

'url': article\_url,

'section': unidecode(result['sectionName']),

'date': result['webPublicationDate'],

'content': content.replace('\n', ''),

'image\_captions': [caption.replace('\n', '') **for** caption **in** image\_captions]

}

articles.append(article)

*# Storing in JSON*

json\_data = json.dumps(articles, indent=4)

*# Write to a new file*

filename = '/Users/dikshasaxena/Documents/2nd sem/Research-NLP/scrapedData-TheGuardian.json'

**with** open(filename, 'w') **as** file:

file.write(json\_data)

Output:

A screenshot of a computer screen

Description automatically generated with medium confidence

a) Using the RSS feed to scrape political articles from The Reuters (It does not provide an API)

**import** feedparser

**import** json

*# URL of the RSS feed for the political section*

rss\_feed\_url = "https://www.reutersagency.com/feed/?best-topics=political-general&post\_type=best"

*# Parse the RSS feed*

feed = feedparser.parse(rss\_feed\_url)

*# Extract article details*

articles = []

**for** entry **in** feed.entries:

article = {

"title": entry.title,

"link": entry.link,

"published": entry.published,

"summary": entry.summary

}

articles.append(article)

*# Convert to JSON format*

json\_data = json.dumps(articles, indent=4)

*# Write to a file*

filename = "/Users/dikshasaxena/Documents/2nd sem/Research-NLP/scrapedData-Reuters-RSS.json"

**with** open(filename, "w") **as** file:

file.write(json\_data)

I am still working on code to fetch real time data directly into the box folder using the RSS feed.

Note: The RSS feed has a limit of approx. 10 articles that can be fetched at a time and although it works better than the beautifulSoup approach where we had to manually find class tags and extract, it still is cumbersome, hence, we are contemplating using the Newspaper3k Library to extract data one by one, or the NewsCatcher API instead.

b) BeautifulSoup to extract data from the Reuters.

**import** requests

**from** bs4 **import** BeautifulSoup

**import** json

**def** **scrape\_article**(url):

*# GET request to fetch the HTML content of the article*

response = requests.get(url)

html\_content = response.content

*# Create a BeautifulSoup object*

soup = BeautifulSoup(html\_content, 'html.parser')

*# Find the image caption, content, heading of the article*

image\_caption\_element = soup.find('figcaption', class\_='primary-image\_\_caption\_\_3Ayt0')

image\_caption = image\_caption\_element.text.strip() **if** image\_caption\_element **else** None

content\_elements = soup.find\_all('p')

content = '\n'.join([element.text **for** element **in** content\_elements])

heading = soup.find('h1').text.strip()

*# Store in a dictionary*

data = {

'url': url,

'heading': heading,

'image\_caption': image\_caption,

'content': content

}

*# Convert to JSON format*

json\_data = json.dumps(data, indent=4)

**return** json\_data

*# Provide the article URL*

article\_url = "https://www.reuters.com/technology/us-congress-consider-two-new-bills-artificial-intelligence-2023-06-08/"

article\_data = scrape\_article(article\_url)

*# Write to a new file*

**with** open('/Users/dikshasaxena/Documents/2nd sem/Research-NLP/scrapedData-Reuters-1.json', 'w') **as** file:

file.write(article\_data)

Note: for each article on the Reuters website, the image caption was displayed using different CSS classes, which meant extracting them, would require a manual effort in inspecting the classes and modifying the code accordingly for each article, hence, scrapped this idea.

1. Newspaper3k – Python Module to extract data and summarize it, finding keywords along the way.

Link to code : <https://www.geeksforgeeks.org/newspaper-article-scraping-curation-python/#>

Note: Working on modifying the code to get it’s URL inputs from the NewsCatcher API, so that it can be automated and we wouldn’t have to fetch the data from each article one by one.

A screenshot of a computer

Description automatically generated

1. NewsCatcher- It provides us with an API to fetch news articles and data by inputting a query. We can specify the sources of the news providers required, limit the number of articles from each, and use multiple related keywords to fetch relevant articles.

This might help us find a more coherent closely related dataset, for our baseling submission, after which we can diversity topics and train the model to give a more robust output.

import requests

url = "https://api.newscatcherapi.com/v2/search"

querystring = {

"q": "\"Ukraine AND Russia\"",

"lang": "en",

"sort\_by": "relevancy",

"page": "1"

}

headers = {

"x-api-key": "GN8nvzQj7XzIW1IP8OiS7GiaajXM2CCPT7CKDyp2YYg"

}

**try**:

response = requests.get(url, headers=headers, params=querystring)

response.raise\_for\_status()

**print**(response.text)

except requests.exceptions.RequestException **as** **e**:

**print**("Error occurred:", **e**)

Link to Documentation: https://docs.newscatcherapi.com/code-snippets/scroll-through-all-pages-to-get-all-found-articles (To play around with the query string)

For example, for keywords Ukraine AND Russia, it would give out articles that contain both these keywords from the sources specified, which we haven’t yet in the code.

The API Keys are private to me for both the Guardian and the NewsCatcher.

Note: The outputs can be found for all of these approaches in the box folder, most of them are in JSON format.

Summer Plans:

Social Media Image Recontextualization-

1. We plan to submit a project baseline by the 28th June 2023 for Recontextualization Detection, which is the sub task A of the whole project.
2. I have worked on web scraping and dataset collection up to now. The summaries are to be generated using the Newspaper3k library. Plan for next two weeks up till 28th is to finalize the pipeline with Smarana, and implement it, and get baseline results. I will also be working on pushing the baseline onto a docker container for the submission.
3. We plan to have a Github Repo up and running for tracking and submission purposes.
4. Post 28th, we will be working on finishing up the Recontextualization Detection sub task and proceed with Recontextualization Localization and Classification throught the rest of the summer or after that till the completion.
5. Smarana mentioned to me that the project will run through to the end of Fall and beyond, and I would love to be a part of it.
6. We have a meeting with Professor Doermann on Thursday (22nd June) to discuss the pipeline and brainstorm ideas on how to proceed further.

A picture containing text, screenshot, receipt

Description automatically generated

Conversational AI-

1. I would like to work on any ongoing project that you may have in the conversational AI domain.
2. I am open to researching on ways to build an Empathetic Chatbot, which would be a continuation from the work that I had done in the NLP and Text Mining Course last semester.
3. I am also open to any other projects you might have available.