

Wiki Page Summarize

Text Processing with Web Scraping

In this assignment, you are expected to scrape a Wikipedia webpage and then summarize the content while retaining the original headings and sections.

Dataset

Since this involves web scraping, you'll be generating your own dataset from Wikipedia.

- Wikipedia Page to Scrape: https://en.wikipedia.org/wiki/Alexander_the_Great

Guidelines for Web Scraping and Text Summarization

1. Use BeautifulSoup 4 to scrape the content of a Wikipedia page. Retrieve not just the text but also the headings and sub-headings.
2. Once scraped, summarize the text under each section while retaining the original headings.
3. Create prompts chains for generating summaries that are coherent and contextually accurate.

Expectations

1. By the end of the assignment, you should be proficient in scraping web data and summarizing textual information.
2. Create a pipeline that goes from web scraping to summarization, ensuring that the summarized output retains the original structure of headings and sections.
3. Propose metrics for evaluating the quality of the summarized content. Consider factors such as information retention, readability, and coherence.

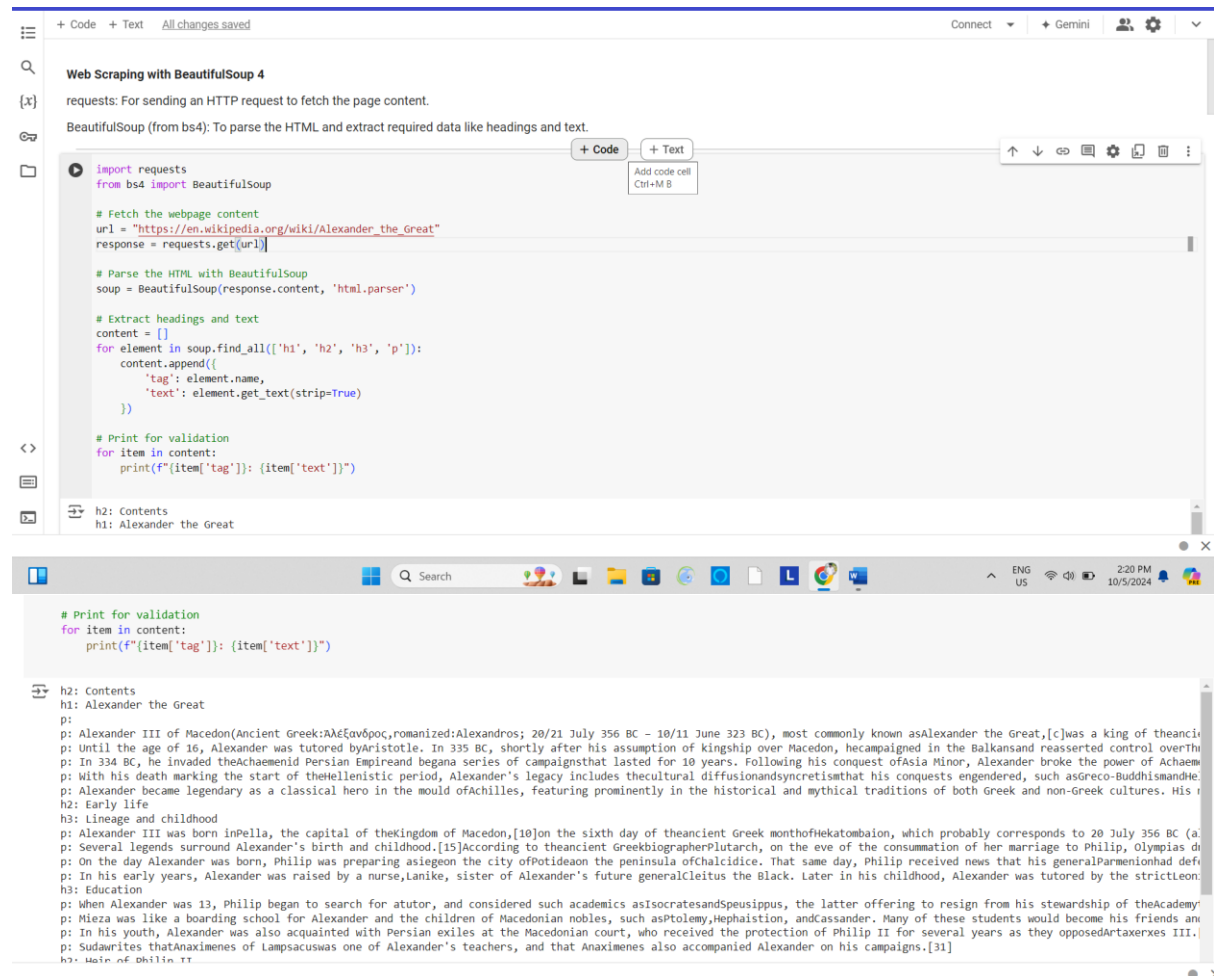
Submission

1. **Code:** Code for web scraping using BeautifulSoup 4, and the summarization pipeline.
2. **Report:** A document outlining the design decisions, challenges encountered, and an error analysis.
3. **Output:** The summarized Wikipedia page, retaining the original headings and sections.
4. **Reference Material:** Additional resources for BeautifulSoup 4 and text summarization can be found in the Appendix.

Web Scraping with BeautifulSoup 4

requests: For sending an HTTP request to fetch the page content.

BeautifulSoup (from bs4): To parse the HTML and extract required data like headings and text.



```
import requests
from bs4 import BeautifulSoup

# Fetch the webpage content
url = "https://en.wikipedia.org/wiki/Alexander_the_Great"
response = requests.get(url)

# Parse the HTML with BeautifulSoup
soup = BeautifulSoup(response.content, 'html.parser')

# Extract headings and text
content = []
for element in soup.find_all(['h1', 'h2', 'h3', 'p']):
    content.append({
        'tag': element.name,
        'text': element.get_text(strip=True)
    })

# Print for validation
for item in content:
    print(f'{item["tag"]}: {item["text"]}')

h2: Contents
h1: Alexander the Great

p:
p: Alexander III of Macedon(Ancient Greek:Ἀλέξανδρος,romanized:Alexandros; 20/21 July 356 BC – 10/11 June 323 BC), most commonly known asAlexander the Great,[c]was a king of theancient Greek kingdom of Macedonia, who became one of the greatest military leaders in world history. He was born in Pella, the capital of the kingdom of Macedonia, and was tutored by the philosopher Aristotle. In 336 BC, he became king at the age of 13, and in 334 BC, he began his conquest of the Persian Empire, which lasted for 10 years. Following his conquest of Asia Minor, Alexander broke the power of the Achaemenid Empire, and in 323 BC, he died at the age of 32, leaving a vast empire that stretched from the Balkans to the Indian subcontinent. His empire was the largest in the history of the world, and his conquests engendered the Hellenistic period, a period of Greek culture that spread across the world. Alexander's legacy includes the cultural diffusion and syncretism that his conquests engendered, such as Greco-Buddhism and the Hellenistic period. He is also known for his military genius, his leadership, and his character. His life and conquests have been the subject of many books, films, and other works of art. His empire was the largest in the history of the world, and his conquests engendered the Hellenistic period, a period of Greek culture that spread across the world. Alexander's legacy includes the cultural diffusion and syncretism that his conquests engendered, such as Greco-Buddhism and the Hellenistic period. He is also known for his military genius, his leadership, and his character. His life and conquests have been the subject of many books, films, and other works of art.
```

Text Summarization

Retain the original structure of headings and sub-headings.

Summarize the text under each heading while preserving key information.

```
+ Code + Text All changes saved Connect + Gemini

from transformers import pipeline

# Load the summarization pipeline
summarizer = pipeline("summarization")

# Function to split long texts into smaller chunks
def split_text(text, max_chunk_length=500):
    """Splits text into smaller chunks to fit model's input size."""
    sentences = text.split('.')
    chunks = []
    current_chunk = []
    current_length = 0

    for sentence in sentences:
        sentence_length = len(sentence)
        if current_length + sentence_length <= max_chunk_length:
            current_chunk.append(sentence)
            current_length += sentence_length
        else:
            # Join current chunk and reset
            chunks.append(' '.join(current_chunk))
            current_chunk = [sentence]
            current_length = sentence_length

    # Add any remaining chunk
    if current_chunk:
        chunks.append(' '.join(current_chunk))

    return chunks

# Function to summarize a section
def summarize_section(text, max_length=150, min_length=50):
    chunks = split_text(text)
    summaries = []

    # Summarize each chunk
    for chunk in chunks:
        summary = summarizer(chunk, max_length=max_length, min_length=min_length, do_sample=False)
        summaries.append(summary[0]['summary_text'])

    # Join the summarized chunks back together
    return ' '.join(summaries)

# Summarize paragraphs under each heading
for item in content:
    if item['tag'] == 'p': # Only summarize paragraphs
        summarized_text = summarize_section(item['text'])
        print(f"Original: {item['text'][:200]}...") # Print first 200 characters for brevity
        print(f"Summary: {summarized_text}")

warnings.warn(
    Your max_length is set to 150, but your input_length is only 3. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
```

```
+ Code + Text All changes saved Connect + Gemini

# Join current chunk and reset
chunks.append(' '.join(current_chunk))
current_chunk = [sentence]
current_length = sentence_length

# Add any remaining chunk
if current_chunk:
    chunks.append(' '.join(current_chunk))

return chunks

# Function to summarize a section
def summarize_section(text, max_length=150, min_length=50):
    chunks = split_text(text)
    summaries = []

    # Summarize each chunk
    for chunk in chunks:
        summary = summarizer(chunk, max_length=max_length, min_length=min_length, do_sample=False)
        summaries.append(summary[0]['summary_text'])

    # Join the summarized chunks back together
    return ' '.join(summaries)

# Summarize paragraphs under each heading
for item in content:
    if item['tag'] == 'p': # Only summarize paragraphs
        summarized_text = summarize_section(item['text'])
        print(f"Original: {item['text'][:200]}...") # Print first 200 characters for brevity
        print(f"Summary: {summarized_text}")

warnings.warn(
    Your max_length is set to 150, but your input_length is only 3. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
```

```
# Summarize paragraphs under each heading
for item in content:
    if item['tag'] == 'p': # Only summarize paragraphs
        summarized_text = summarize_section(item['text'])
        print(f"Original: {item['text'][:200]}...") # Print first 200 characters for brevity
        print(f"Summary: {summarized_text}")

warnings.warn(
    Your max_length is set to 150, but your input_length is only 3. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Your max_length is set to 150, but your input_length is only 122. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Original: ...
    Summary: CNN.com will feature iReporter photos in a weekly Travel Snapshots gallery . Please submit your best shots of our featured destinations for next week . Visit CNN iReport.c
    Your max_length is set to 150, but your input_length is only 58. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Your max_length is set to 150, but your input_length is only 115. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Original: Alexander III of Macedon(Ancient Greek:Ἀλέξανδρος, romanized:Alexandros; 20/21 July 356 BC – 10/11 June 323 BC), most commonly known asAlexander the Great,[c]was a king of
    Summary: Alexander III of Macedon succeeded his father Philip IIto the throne in 336 BC at the age of 20 . He spent most of his ruling years conducting a lengthymilitary campaignth
    Your max_length is set to 150, but your input_length is only 112. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Original: Until the age of 16, Alexander was tutored byAristotle. In 335 BC, shortly after his assumption of kingship over Macedon, he reasserted control over the Balkans and reasserted control
    Summary: Alexander was tutored by Aristotle until the age of 16 . In 335 BC, shortly after his assumption of kingship over Macedon, he reasserted control over the Balkans . Alexand
    Your max_length is set to 150, but your input_length is only 56. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Your max_length is set to 150, but your input_length is only 81. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Your max_length is set to 150, but your input_length is only 38. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Your max_length is set to 150, but your input_length is only 103. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Original: In 334 BC, he invaded theAchaemenid Persian Empireand begana series of campaignsthat lasted for 10 years . Following his conquest ofAsia Minor, Alexander broke the power of
    Summary: Alexander broke the power of Achaemenid Persia in a series of decisive battles, including those atIssusandGaugamela . He overthrew Darius IIIand conquered the Achaemenid Em
    Your max_length is set to 150, but your input_length is only 53. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
    Your max_length is set to 150, but your input_length is only 33. Since this is a summarization task, where outputs shorter than the input are typically wanted, you might consider de
```

pipeline

You scrape the headings and corresponding text.

You summarize the text for each section.

You output the summarized text, retaining the headings structure.

Metrics for Evaluation

Propose metrics to evaluate the quality of your summarized content. Here are a few:

Information Retention: Measure how much of the important information is preserved in the summaries.

Readability: Use readability metrics like Flesch-Kincaid to evaluate how easy the text is to read.

Coherence: Evaluate how logically the sentences flow.

Compression Ratio: The ratio of the original text to the summarized text, ensuring the summary is concise yet informative.