module03-analyzing-text-content-natural-language-processing-30pct/3-1-2-computer-lab.qmd

List of Figures

List of Tables

prepare text content

This code chunk outlines a procedure for processing a collection of text files using Python, with a focus on content analysis and natural language processing (NLP). Here's a detailed explanation of each step:

```
import os
import pandas as pd
import re

# Function to clean text by removing non-ASCII characters
def clean_text(text):
    # Remove non-ASCII characters
    cleaned_text = re.sub(r'[^\x00-\x7F]+', '', text)
    return cleaned_text

# Directory containing text files
directory_path = '/content/osm-cca-nlp/res'
directory_path = '/home/sol-nhl/rnd/d/quarto/osm-cca-nlp/res'
```

1. **Importing Necessary Libraries**:
   - The code begins by importing essential Python libraries: `os`, `pandas`, and `re`. The `os` library is used for interacting with the file system, `pandas` is a powerful data manipulation library used to manage and analyze data structures like DataFrames, and `re` is the regular expressions library used for pattern matching and text processing.
2. **Defining a Text Cleaning Function**:
   - A function `clean_text(text)` is defined to clean the text data by removing non-ASCII characters. This function uses the `re.sub()` method to search the text for any character that does not fall within the ASCII range (`[^\x00-\x7F]`) and replaces it with an empty string, effectively removing these characters. This step is crucial in NLP tasks to standardize text data, making it easier to analyze.
3. **Setting Up the Directory Path**:
   - The directory containing the text files is specified with `directory_path = '/content/osm-cca-nlp/res'`. This path directs the script to the location of the text files that will be processed.

```
# Initialize an empty list to store the data
data = []

# Initialize a unique ID counter
unique_id = 1

# Iterate over the text files in the directory
for filename in os.listdir(directory_path):
    # Consider only plain text files
    if filename.endswith(".txt") or filename.endswith(".md"):
        file_path = os.path.join(directory_path, filename)

        # Read the file content
        with open(file_path, 'r', encoding='utf-8') as file:
            text = file.read()

        # Clean the text
        cleaned_text = clean_text(text)

        # Append the data as a dictionary with a unique ID
        data.append({
            'id': unique_id,
            'filename': filename,
            'original_text': text,
            'cleaned_text': cleaned_text
        })

        # Increment the unique ID
        unique_id += 1
```

4. **Initializing Data Structures**:
   - An empty list `data` is initialized to store the cleaned data, and a `unique_id` counter is set to `1` to uniquely identify each text file. These data structures are essential for organizing and managing the extracted and cleaned content.
5. **Iterating Over Files in the Directory**:

- The script iterates over each file in the specified directory using `os.listdir(directory_path)`. A conditional statement `if filename.endswith(".txt") or filename.endswith(".md")` ensures that only plain text files (`.txt`) and Markdown files (`.md`) are processed. This step is fundamental in content analysis as it focuses the analysis on relevant document types.

6. **Reading and Cleaning Text Content**:
   - For each text file, the file is opened and read into a string variable `text` using `open(file_path, 'r', encoding='utf-8')`. The content is then passed to the `clean_text()` function to remove non-ASCII characters, resulting in a cleaned version of the text stored in `cleaned_text`. This step is crucial for preparing the text data for further NLP tasks by ensuring it is in a consistent and analyzable format.

```
# Create a Pandas DataFrame
text_df = pd.DataFrame(data)

# Save the DataFrame as a TSV file in the 'csv' subdirectory
output_file_path = '/home/sol-nhl/rnd/d/quarto/osm-cca-nlp/csv/text data.tsv'
output_file_path = '/content/osm-cca-nlp/csv/text_data.tsv'

# Save the DataFrame to a TSV file
text_df.to_csv(output_file_path, sep='\t', index=False)

# Display the DataFrame
print(text_df)
```

7. **Storing the Data**:
   - The original and cleaned text, along with the filename and unique ID, are stored as a dictionary in the `data` list. This structured storage is essential for keeping track of each document's metadata and content, facilitating easy access and manipulation for later analysis.
8. **Creating a DataFrame**:
   - Once all files are processed, the list `data` is converted into a pandas DataFrame using `pd.DataFrame(data)`. This DataFrame organizes the collected data into a tabular format, where each row corresponds to a file, and columns represent the unique ID, filename, original text, and cleaned text. This step is critical in content analysis and NLP as it allows for systematic exploration, manipulation, and analysis of the textual data.
9. **Displaying the Data**:
   - Finally, the DataFrame is printed to the console using `print(df)`, providing a visual representation of the processed data. This allows for a quick inspection of the results, ensuring that the text cleaning and data aggregation processes have been correctly executed.

This procedure exemplifies a typical workflow in content analysis and NLP, where raw text data is cleaned, organized, and prepared for more sophisticated analytical tasks such as tokenization, entity recognition, or sentiment analysis. The use of Python and its libraries like `pandas` and `re` streamlines these tasks, making it easier to manage and analyze large collections of text.

code chunk

```
import os
import pandas as pd
import re

# Function to clean text by removing non-ASCII characters
def clean_text(text):
    # Remove non-ASCII characters
    cleaned_text = re.sub(r'[^\x00-\x7F]+', '', text)
    return cleaned_text

# Directory containing text files
directory_path = '/content/osm-cca-nlp/res'
directory_path = '/home/sol-nhl/rnd/d/quarto/osm-cca-nlp/res'

# Initialize an empty list to store the data
data = []

# Initialize a unique ID counter
```

```
unique_id = 1

# Iterate over the text files in the directory
for filename in os.listdir(directory_path):
    # Consider only plain text files
    if filename.endswith(".txt") or filename.endswith(".md"):
        file_path = os.path.join(directory_path, filename)

        # Read the file content
        with open(file_path, 'r', encoding='utf-8') as file:
            text = file.read()

        # Clean the text
        cleaned_text = clean_text(text)

        # Append the data as a dictionary with a unique ID
        data.append({
            'id': unique_id,
            'filename': filename,
            'original_text': text,
            'cleaned_text': cleaned_text
        })

        # Increment the unique ID
        unique_id += 1

# Create a Pandas DataFrame
text_df = pd.DataFrame(data)

# Save the DataFrame as a TSV file in the 'csv' subdirectory
output_file_path = '/content/osm-cca-nlp/csv/text_data.tsv'
output_file_path = '/home/sol-nhl/rnd/d/quarto/osm-cca-nlp/csv/text_data.tsv'

# Save the DataFrame to a TSV file
text_df.to_csv(output_file_path, sep='\t', index=False)

# Display the DataFrame
```

```
print(text_df)
```