**Construction of New Data**

**1. Data Source and Overview**

**Dataset**: **BBC News Dataset**

* **Description**: The BBC News Dataset consists of news articles across five primary categories: **Business**, **Entertainment**, **Politics**, **Sport**, and **Tech**.
* **Source**: The dataset was sourced from University College Dublin's Machine Learning Group, where each article is available in plain text format and organized by category.
* **File Structure**: The dataset is structured into folders, each representing a category. Each file within these folders is an individual news article.

**2. Preprocessing and Preparation**

**Objective**: To prepare the dataset for segmentation tasks, articles were processed to ensure clear boundaries and minimal text distortion.

**Processing Steps**:

* **Step 1: Load Articles**: Each article was loaded from its respective category folder and concatenated into a single document.
* **Step 2: Apply Text Cleaning**: Basic text preprocessing was applied, including lowercasing, normalization of whitespace, and minimal punctuation handling to retain readability.
* **Step 3: Insert Segmentation Markers**: After each article, a marker (===END===) was added to indicate segmentation boundaries, ensuring each category transition was clearly defined.

**Code Snippet for Preprocessing**:

import os

import re

def clean\_text(txt):

txt = txt.lower()

txt = re.sub(r"\n+", " ", txt) # Normalize newlines

txt = re.sub(r"\s+", " ", txt).strip() # Normalize whitespace

return " ".join(["START", txt, "END"])

def preprocess\_bbc\_dataset(dataset\_folder, output\_path):

segmented\_text = []

for category in ['business', 'entertainment', 'politics', 'sport', 'tech']:

category\_path = os.path.join(dataset\_folder, category)

files = os.listdir(category\_path)

for file in files:

file\_path = os.path.join(category\_path, file)

with open(file\_path, 'r', encoding='utf-8') as f:

article = f.read().strip()

article = clean\_text(article)

segmented\_text.append(article)

segmented\_text.append("===END===") # Boundary marker

with open(output\_path, 'w', encoding='utf-8') as output\_file:

output\_file.write("\n\n".join(segmented\_text))

**Explanation of Code**:

* **clean\_text**: This function performs basic normalization to retain full words without stemming, converting all text to lowercase, removing extra spaces, and ensuring sentences are readable.
* **preprocess\_bbc\_dataset**: Loads, processes, and combines each article, adding segmentation markers to separate each article.

**3. Data Sample and Snippet**

Below is a sample snippet from the preprocessed dataset demonstrating the structure:

START the stock market showed significant growth as several major companies posted high earnings. investors are optimistic about future economic growth. END

===END===

START in a landmark decision, the government passed a policy reform aimed at reducing carbon emissions. this move has been praised by environmentalists but faces opposition from industry leaders. END

===END===

**Explanation**: Each article starts with "START" and ends with "END" to denote boundaries within the segment. The "===END===" marker represents the end of each article, maintaining a clear and consistent structure.\

**4. Human Annotation and Agreement**

**Annotation**: No additional human annotation was required, as the dataset is already categorized into clear topics (business, entertainment, etc.), and each category serves as a distinct segment for testing.

**Agreement Measure**: Since no manual annotations were required, inter-annotator agreement measures are not applicable in this instance.