1. Problem Understanding and Requirements

The task is to calculate and add several metrics to an existing DataFrame (df2) after the extraction of the data from the article links that are provided in the input.xlsx file. These metrics include sentiment scores (POSITIVE SCORE, NEGATIVE SCORE, POLARITY SCORE, SUBJECTIVITY SCORE), text complexity metrics (AVG SENTENCE LENGTH, PERCENTAGE OF COMPLEX WORDS, FOG INDEX, AVG NUMBER OF WORDS PER SENTENCE, COMPLEX WORD COUNT, WORD COUNT, SYLLABLE PER WORD, PERSONAL PRONOUNS, AVG WORD LENGTH).

2. Initial Setup and Data Loading

Data Extraction:

- Used BeautifulSoup for the extraction of the data.
- Use requests.get(url) to fetch HTML content from a given URL.
- Parse the content using BeautifulSoup ('html.parser').
- Find all <p> tags to extract text paragraphs.
- Join paragraph texts into a single string and return.

Loaded the necessary libraries (pandas, nltk) and downloaded required NLTK resources (punkt for tokenization, cmudict for syllable counting, stopwords for common words).

Defined paths and directories for loading additional resources like stopwords and sentiment dictionaries.

3. Data Cleaning and Preparation

Implemented functions to load stopwords from various files (load_stop_words).

Loaded positive and negative word lists from master dictionaries and removed stopwords (load_words).

Defined functions for:

- count_syllables: to count syllables in a word using NLTK's CMU Pronouncing Dictionary and a simple heuristic.
- count_personal_pronouns: to count personal pronouns (e.g., 'I', 'we') in text using regex.
- calculate_additional_metrics: to compute various metrics including text complexity and sentiment scores for each text in df2.

4. Metric Calculation Functions

- calculate_scores: computes sentiment scores (positive, negative, polarity, subjectivity) based on the presence of positive and negative words in the text.
- calculate_avg_sentence_length: computes the average sentence length in terms of words.
- calculate_percentage_complex_words: calculates the percentage of complex words in the text.
- calculate_fog_index: computes the Fog Index, a measure of text readability.

- calculate_syllable_per_word: calculates the average number of syllables per word.
- calculate_avg_word_length: computes the average word length in characters.

5. Conclusion

This approach ensures that all specified metrics are computed accurately and efficiently using functions tailored to handle each metric's calculation. By leveraging pandas' vectorized operations and NLTK's text processing capabilities, the solution is designed to handle text data effectively and generate comprehensive insights into each text's characteristics. Adjustments can be made as needed based on specific data nuances or additional requirements.