Overview

Introducing CR-nlp, your all-in-one solution for natural language processing (NLP) tasks. Designed to simplify the complexities of text analysis, CR-nlp offers a comprehensive suite of tools tailored for developers and researchers alike.

At its core, CR-nlp harnesses the power of industry-leading NLP libraries, including NLTK and Hugging Face's Transformers. By leveraging pre-trained models and robust algorithms, CR-nlp empowers users to streamline common NLP tasks with ease.

Key Features:

- Tokenize Text: Seamlessly break down text into tokens using state-of-the-art models from Hugging Face's Transformers library. With CR-nlp, text tokenization becomes effortless and efficient, enabling users to extract valuable insights from textual data.

- Analyze Sentiment: Gain deeper understanding into textual sentiment with CR-nlp's sentiment analysis capabilities. Leveraging both Transformers and NLTK's VADER model, CR-nlp provides accurate sentiment analysis, empowering users to gauge sentiment polarity with precision.

- Lemmatize Words: Transform words into their base forms based on their part of speech with CR-nlp's lemmatization functionality. By identifying and normalizing word forms, CR-nlp facilitates clearer and more accurate analysis of textual data.

- Stem Words: Reduce words to their root forms using the Porter Stemming algorithm. With CR-nlp's stemming capabilities, users can efficiently preprocess textual data, simplifying subsequent analysis tasks.

From tokenization to sentiment analysis, lemmatization, and stemming, CR-nlp equips users with the tools they need to unlock the full potential of natural language processing. Whether you're a developer seeking to enhance text-based applications or a researcher exploring linguistic patterns, CR-nlp is your go-to companion for all things NLP.

Tokenize a given text using the Hugging Face Transformers library.

```
    Parameters:
    - text (str): The input text to tokenize.
    - model_name (str): The name of the pre-trained model to use for tokenization.
                  Default is "bert-base-uncased".

    Returns:
    - tokens (list): List of tokens obtained by tokenizing the input text.
```

FUNCTIONS-

- tokenize_text

  Tokenize a given text using the Hugging Face Transformers library.

  Parameters:
  - text (str): The input text to tokenize.
  - model_name (str): The name of the pre-trained model to use for tokenization.
    Default is "bert-base-uncased".

  Returns:
  - tokens (list): List of tokens obtained by tokenizing the input text.

- analyze_sentiment

  Analyze sentiment of a given text using a pre-trained sentiment analysis model.

  Parameters:
  - text (str): The input text for sentiment analysis.
  - model_name (str): The name of the pre-trained sentiment analysis model.
    Default is "nlptown/bert-base-multilingual-uncased-sentiment".

  Returns:
  - sentiment (str): The predicted sentiment (e.g., "POSITIVE", "NEGATIVE", "NEUTRAL").
  - confidence (float): The confidence score associated with the predicted sentiment.

- lemmatize_words

  Lemmatize a list of words.

  This function takes a list of words, determines the part of speech for each word, and then lemmatizes it (converts it to its base or dictionary form) according to its part of speech. It utilizes the NLTK library's WordNetLemmatizer and the part-of-speech tagging to accurately lemmatize each word.

  Parameters:
  - words: A list of words (strings) that you want to lemmatize.

  Returns:
  - A list of lemmatized words

- analyze_sentiment_vader

  Analyzes the sentiment of a given text using VADER sentiment analysis.

  Parameters:
  - text: A string containing the text to analyze.

  Returns:
  - A dictionary containing the scores for negative, neutral, positive, and compound sentiments.

- stem_words

  Stems a list of words.

  This function applies the Porter Stemming algorithm to a list of words, reducing each word to its root or stem form. It's particularly useful in natural language processing and search applications where the exact form of a word is less important than its root meaning.

  Parameters:

  - words: A list of words (strings) to be stemmed.

  Returns:

  - A list containing the stemmed version of each input word.

Step into a realm of unparalleled NLP innovation with CR-nlp. Our library sets the stage for transformative text analysis, providing a range of specialized functions meticulously crafted to simplify and elevate your NLP workflows. Effortlessly segment text into tokens, gauge nuanced sentiments, and refine words with precision using our advanced tokenization, sentiment analysis, lemmatization, and stemming functionalities. With seamless integration of state-of-the-art models and algorithms, CR-nlp empowers you to uncover deeper insights and unlock new dimensions of textual analysis and understanding.

Experience the versatility of sentiment analysis, precise word normalization, and root reduction with CR-nlp. Whether you're a seasoned developer or a burgeoning researcher, our library offers a cohesive suite of functionalities to enrich your text analysis endeavors. Embark on a journey of discovery with CR-nlp—where every text holds a story waiting to be revealed.

Requirements:

nltk
transformers