lab1_tokenization

Harito ID

2025-09-16

Lab 1: Text Tokenization

Setup

Before starting this lab, ensure you have installed all necessary project dependencies by running:

```
pip install -r requirements.txt
```

Objective

The goal of this lab is to understand and implement a fundamental NLP preprocessing step: tokenization. You will create a simple tokenizer and then a more advanced one.

Task 1: Simple Tokenizer

1. **Define the Interface:** In `src/core/interfaces.py`, define an abstract base class for a `Tokenizer`. It should have a single method, `tokenize(self, text: str) -> list[str]`.
2. **Implement a Simple Tokenizer:**
    - Create a new file: `src/preprocessing/simple_tokenizer.py`.
    - Inside this file, create a class `SimpleTokenizer` that inherits from the `Tokenizer` interface.
    - Implement the `tokenize` method. This method should:
        - Convert the text to lowercase.
        - Split the text into tokens based on whitespace.
        - Handle basic punctuation (e.g., `.` , `?` `!`) by splitting them from words. For example, `"Hello, world!"` should become `["hello", ",", "world", "!"]`.

Task 2: Regex-based Tokenizer (Bonus)

1. **Implement a Regex Tokenizer:**
    - Create a new file: `src/preprocessing/regex_tokenizer.py`.
    - Inside this file, create a class `RegexTokenizer` that also inherits from the `Tokenizer` interface.
    - Implement the `tokenize` method using a single regular expression to extract tokens. This is more robust than simple splitting. A good starting regex could be `\w+|[^\w\s]`.

Evaluation

- Create a `main.py` file in the root directory to test your tokenizers.
- Instantiate your `SimpleTokenizer` and `RegexTokenizer`.
- Test them with the following sentences and print the output:
  - `"Hello, world! This is a test."`
  - `"NLP is fascinating... isn't it?"`
  - `"Let's see how it handles 123 numbers and punctuation!"`

Task 3: Tokenization with UD_English-EWT Dataset

1. **Load Dataset:**
   - In your `main.py`, import `load_raw_text_data` from `src.core.dataset_loaders`.
   - Load the raw text from the `UD_English-EWT` dataset:

```python
from src.core.dataset_loaders import load_raw_text_data
# ... (your tokenizer imports and instantiations) ...

dataset_path = "/Data/HaritoWork/Teaching/VNU_HUS/Tu_NLP/data/UD_English-EWT/en
raw_text = load_raw_text_data(dataset_path)

# Take a small portion of the text for demonstration
sample_text = raw_text[:500] # First 500 characters

print("\n--- Tokenizing Sample Text from UD_English-EWT ---")
print(f"Original Sample: {sample_text[:100]}...")

simple_tokens = simple_tokenizer.tokenize(sample_text)
print(f"SimpleTokenizer Output (first 20 tokens): {simple_tokens[:20]}")

regex_tokens = regex_tokenizer.tokenize(sample_text)
print(f"RegexTokenizer Output (first 20 tokens): {regex_tokens[:20]}")
```

   - Observe and compare the output of your tokenizers on a real-world dataset.