

Assignment 04

The 1980s saw a shift from Natural Language Processing techniques aiming to codify the grammatical rules of natural language towards techniques aiming to use statistical models to generate text. One early idea which technically isn't "AI" seeing as it is "memorizing" the training data and yet introduces us to the power contained in statistical techniques of text generation is the idea of Markov chains. Write a python function `generate(filename: str, start_words: list[str], chain_length: int, num_generated: int) -> str` which takes a filename, a chain length, a list of start words which has to be exactly as long as the `chain_length` (why?), and an integer `num_generated` and returns a sentence `num_generated` words long which sounds similar to the text contained in `filename`.

PA You

The 1980s saw a shift from Natural Language Processing techniques aiming to codify the grammatical rules of natural language towards techniques aiming to use statistical models to generate text. One early idea which technically isn't "AI" seeing as it is "memorizing" the training data and yet introduces us to the power contained in statistical techniques of text generation is the idea of Markov chains. Write a python function `generate(filename: str, start_words: list[str], chain_length: int, num_generated: int) -> str` which takes a filename, a chain length, a list of start words which has to be exactly as long as the `chain_length` (why?), and an integer `num_generated` and returns a sentence `num_generated` words long which sounds similar to the text contained in `filename`.



ChatGPT

Creating a text generator using Markov chains is a fascinating application of statistical models in the field of Natural Language Processing. The reason the list of start words has to be exactly as long as the chain length is because the Markov chain needs an initial state (or states) to start generating the subsequent text. In a Markov chain of length n , the next state (word, in this context) is determined based on the previous n states. Hence, if your chain length is n , you need n initial states (words) to predict the next state according to the probabilities learned from the training data.

Here's a basic implementation of such a function in Python. This script reads from a file, builds a Markov chain model, and generates text based on this model. Note that for simplicity and readability, this script is quite basic and can be improved in many ways, including handling edge cases and optimizing memory usage.

This script works as follows:

1. **build_markov_chain** reads the file and builds a Markov chain where each state (a tuple of words) maps to a dictionary of possible next words and their probabilities.
2. **generate** then uses this model to generate text. It starts with the given initial words, selects the next word based on the probabilities, and updates the current state to include this new word while dropping the oldest word in the state.

This implementation assumes the file is plaintext and uses simple space-based tokenization. For real-world applications, you might want to consider more sophisticated preprocessing and tokenization to handle punctuation, capitalization, and other linguistic nuances.

PA

You

give me some test cases in code for above code

Make sure you have test text files (`'test_text_1.txt'`, `'test_text_2.txt'`, `'test_text_3.txt'`) containing some sample text in the same directory as your Python script. Adjust the test cases according to your file content and desired parameters. These test cases check if the `'generate'` function returns a string and also tests for invalid input cases such as incorrect chain length and start words length.



You

give sample content of each test_text_1.txt, test_text_2.txt, test_text_3.txt



ChatGPT

Sure, here are some sample contents for the test text files:

``test_text_1.txt`:`

bash

Copy code

The **cat** sat on the mat. The **cat** meowed loudly. The dog barked at the **cat**. The **cat** ran

``test_text_2.txt`:`

css

Copy code

I love to read books. I love to go for walks in the park. I love spending time with n

``test_text_3.txt`:`

csharp

Copy code

Machine learning **is** a subfield of artificial intelligence. Machine learning algorithm

These sample texts contain diverse sentences that can be used to test the Markov chain generator function. Adjust them as needed for your testing purposes.



You

do not use unittest. write normal functions



ChatGPT

Sure, I'll provide you with a simple script to test the Markov chain text generator without using ``unittest``. Instead, this script directly calls the generator function with predefined parameters and prints the results. This approach is straightforward and suitable for quick checks or demonstrations.

First, ensure your generator function and necessary imports are correctly defined in your script. Then, add the following testing functions:

Remember to create the text files `test_text_1.txt`, `test_text_2.txt`, and `test_text_3.txt` in the same directory as your script, with the content provided earlier. This setup will let you see how the generator performs with different inputs and also how it handles error conditions.

