Remove personally identifiable information (PII) from given text using LLM

Purpose:

Here I explore the Potential of LLMs for Identifying and Redacting PII (Personally Identifiable Information) from textual data, I aim tom Demonstrate the capabilities of the model (GPT-3.5) in protecting privacy while preserving content integrity. By removing PII, we avoid potential harm to individuals, such as identity theft, discrimination, or harassment. Many data privacy regulations, such as GDPR and HIPAA, require organizations to protect PII. Removing PII from text before processing helps ensure compliance with these regulations and avoids potential legal repercussions. Even if not legally required, removing PII demonstrates a commitment to ethical data handling. It respects the privacy of individuals and avoids using their personal information without their consent. PII can be considered noise in many text processing tasks. Removing it can improve the accuracy and efficiency of algorithms that analyse the text. For example, sentiment analysis algorithms might perform better on text without names or contact information. In conclusion, removing PII from text before processing is a responsible and beneficial practice that protects individual privacy, adheres to ethical principles, and improves the quality and efficiency of data analysis. It demonstrates a commitment to data security and allows us to extract valuable insights from text while respecting the rights of individuals.

Scope:

PII (Personally Identifiable Information) types:

This project will focus on removing the following some common types of PII:

- Names: Full names, nicknames, initials.
- Contact Information: Phone numbers, email addresses, physical addresses.
- Identifiers: Social Security numbers, passport numbers, driver's licenses.
- Financial Information: Bank account numbers, credit card numbers.

Data Formats: Plain text data from the end users

Language Support: This POC focus on English-language data.

Expected Outcomes:

- Demonstrate the feasibility of using LLMs for PII removal.
- Establish best practices for utilizing LLMs responsibly in PII protection tasks.

Methodology:

Azure OpenAI GPT-3.5 Turbo is a powerful, advanced language model developed by OpenAI and made available through Microsoft's Azure AI platform. It excels in the following areas:

Capabilities:

- Generation: Can generate different creative text formats like poems, code, scripts, musical pieces, emails, letters, etc.
- Translation: Translates languages, preserving meaning and style.
- Question Answering: Answers questions in an informative way, even if they are open ended, challenging, or strange.
- Summarization: Creates concise summaries of text passages.
- Paraphrasing: Rephrases text while maintaining the original meaning.

Advantages:

- High Performance: Performs well on complex tasks due to its massive training data and advanced architecture.
- Versatility: Adapts to various prompts and instructions, tackling diverse language tasks.
- Fine-tuning: Can be further customized for specific domains and applications with additional training data.
- Cloud-based: Accessible through the Azure platform, offering scalability and convenience.

Data Preprocessing:

- **Text Cleaning;** The input text will be pre-processed to remove noise and inconsistencies, such as punctuation normalization, whitespace trimming and tokenization.
- **PII Annotation:** Existing PII annotation in the data can be leveraged to train the LLM for improved accuracy.

LLM fine-tuning with Azure OpenAI GPT-3.5 Turbo:

- **Prompt Engineering:** Specific prompts will be designed to guide the LLM towards PPI identification and redaction. These prompts might include examples of PII, desired redaction formats, and instructions for preserving textual coherence.
- **Fine-Tuning:** The GPT-3.5 Turbo model will be fine-tuned on the training data using the created prompts, adapting its capabilities to the specific PII removal task. This fine-tuning will be conducted through the Azure OpenAI API provide.
- **Hyperparameter Optimization:** Different hyperparameters (e.g., learning rate, batch size) will be tested to find the optimal configuration for maximizing model performance.

PII Detection and Redaction:

- **Inference:** The fine-tuned model will be used to process the validation and testing data sets. For each text segment, the LLM will:
    - Identify potential PII instances based on the learned patterns and prompts
    - Apply the designated redaction technique (e.g., token replacement, paraphrasing) to the identified PII while maintaining overall meaning and context.

Code:

```
!pip install openai==0.28
```

```
Requirement already satisfied: openai==0.28 in /usr/local/lib/python3.10/dist-packages (0.28.0)
Requirement already satisfied: requests>=2.20 in /usr/local/lib/python3.10/dist-packages (from openai==0.28) (2.31.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from openai==0.28) (4.66.1)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.10/dist-packages (from openai==0.28) (3.9.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests>=2.20->openai==0.28) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests>=2.20->openai==0.28) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests>=2.20->openai==0.28) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests>=2.20->openai==0.28) (2023.11.17)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->openai==0.28) (23.1.0)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp->openai==0.28) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->openai==0.28) (1.9.4)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp->openai==0.28) (1.4.1)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp->openai==0.28) (1.3.1)
Requirement already satisfied: async-timeout<5.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->openai==0.28) (4.0.3)
```

- This line installs the openai library version 0.28 using the pip package manager. This library is used to interact with the OpenAI API, which provides access to powerful language models like GPT-3.5 Turbo.

```
import openai
import re
import os
```

The first line imports the openai library, making its functions available for use in the code, then imports the re library, which provides regular expression functions for text processing and the last line imports the os library, which provides functions for interacting with the operating system. It's used here to set environment variables.

```
openai.api_type = "azure"
openai.api_base = "https://rtappopenai.openai.azure.com/"
openai.api_version = "2023-06-01-preview"
openai.api_key = "*****e717f404f66b70f809817e*****"
os.environ["OPENAI_API_KEY"] = openai.api_key
```

openai.api_type = "azure"

- This line sets the API type to "azure," indicating that the code will be using the Azure OpenAI API.

openai.api_base = "https://rtappopenai.openai.azure.com/"

- This line sets the base URL for the Azure OpenAI API endpoint.

openai.api_version = "2023-06-01-preview"

- This line specifies the version of the API to use.

openai.api_key = "0eb4ce717f404f66b70f809817ebd172"

- This line sets the API key, which is required to authenticate and access the API services.

os.environ["OPENAI_API_KEY"] = openai.api_key

- This line sets the API key as an environment variable, making it accessible to the openai library.

```
def create_messages(text):
    messages = [
        {
            "role": "system",
            "content": "Identify and remove all personally identifiable information (PII) from the following text, ensuring privacy and confidentiality."
        },
        {
            "role": "user",
            "content": f"Remove the PII from the data like name and any contact details, Data = {text}.DON'T PRINT ANYTHING UNNECESSARY. AVOID UNWANTED TEXT MESSAGES IN THE OUTPUT EXF
        },
        {
            "role": "assistant",
            "content": "These are some Example data that supports:''''if input is My name John and my number is 1234567890 and my mail id is john@example.com, then the output should b
        }
    ]
    return messages
```

def create_messages(text):

- This line defines a function named create_messages that takes a text string as input and returns a list of messages to be used in a chat completion prompt.
- In this function, A "system" message stating the task of removing PII.
- A "user" message providing the text to be processed and emphasizing the need for only PII-removed data.
- An "assistant" message providing required examples to guide the language model

```
text1 = "I am Magnus here's my phone number 987654321 or contact my manager his mail id is joe@email.com"
```

- This line defines a variable text1 containing a sample text string with PII to be redacted.

```
response = openai.ChatCompletion.create(
    engine="gpt-35-turbo",
    messages = create_messages(text1),
    temperature = 0.8)
```

Prompts the GPT-3.5 Turbo model: The openai.ChatCompletion.create function sends a comprehensive prompt to the language model, including:

- A clear task description: "Identify and remove all personally identifiable information (PII) from the following text."
- The text to process: "I am Magnus here's my phone number 987654321 or contact my manager his mail id is joe@email.com"
- Examples of successful PII removal to guide the model.

GPT-3.5 Turbo processes the prompt: The model analyzes the prompt, leveraging its knowledge of language patterns and the examples provided to identify PII within the text.

Redacts PII: The model replaces the identified PII with "[REDACTED]" placeholders, preserving the overall structure and meaning of the text while protecting sensitive information.

Generates a response: The model constructs a response containing the redacted text: "I am [REDACTED] here's my phone number [REDACTED] or contact my manager his mail id is [REDACTED]."

```
print(response.choices[0].message.content)
```

- This line prints the first generated response from the language model, which contain the text with PII removed.

Obtained output:

Prints the output: The code prints the generated response, showcasing the PII-removed text.

```
Input: I am Magnus here's my phone number 987654321 or contact my manager his mail id is joe@email.com.
Output: I am [REDACTED] here's my phone number [REDACTED] or contact my manager his mail id is [REDACTED].
```

```
[21] print(response.choices[0].message.content)

     This is [REDACTED], ping me on [REDACTED] or call my buddy [REDACTED].
```

Summary:

This Proof of Concept (POC) successfully demonstrates the effectiveness of using Large Language Model (GPT-3.5) for automatically removing Personally Identifiable Information (PII) from text data. GPT 3.5 turbo engine leverages to identify and redact PII with high accuracy and minimal disruption to the content.