SOFTWARE REQUIREMENT SPECIFICATION

1. Introduction

1.1 Purpose

The purpose of this project is to develop **REDACT**, a machine learning-based redaction tool designed to conceal, anonymize, and mask sensitive data across multiple formats such as text files, documents and PDFs. As privacy and security needs grow in industries like healthcare, finance, and law, **REDACT** provides an efficient and customizable redaction solution.

1.2 Scope

REDACT will support data redaction across various formats, allowing users to define the degree of redaction using a gradational redaction scale. The system ensures that the redacted data retains its structural similarity to the original, making it usable for analytical and operational purposes. Leveraging advanced natural language processing (NLP) techniques and deep learning models, the tool will anonymize identifiable content while providing additional functionality to generate realistic synthetic data. This allows for the creation of shareable and collaboration-friendly datasets, enhancing utility while protecting sensitive information.

1.3 Definitions, Acronyms, and Abbreviations

NLP: Natural Language Processing

NER: Named Entity Recognition

BERT: Bidirectional Encoder Representations from Transformers

ML: Machine Learning

API: Application Programming Interface

1.4 References

[1]Cumby, C., & Ghani, R. (2011, August). A machine learning based system for semi-automatically redacting documents. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 25, No. 2, pp. 1628-1635).

[2] Yermilov, O., Raheja, V., & Chernodub, A. (2023). Privacy-and utility-preserving nlp with anonymized data: A case study of pseudonymization. arXiv preprint arXiv:2306.05561.

[3] Peng, S., Huang, M. J., Wu, M., & Wei, J. (2024). Transforming Redaction: How AI is Revolutionizing Data Protection. arXiv preprint arXiv:2409.15308.

[4] Lison, P., Pilán, I., Sánchez, D., Batet, M., & Øvrelid, L. (2021, August). Anonymisation models for text data: State of the art, challenges and future directions. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers) (pp. 4188-4203).

[5] Larbi, I. B. C., Burchardt, A., & Roller, R. (2022). Which anonymization technique is best for which NLP task?--It depends. A Systematic Study on Clinical Text Processing. arXiv preprint arXiv:2209.00262.

2. System Overview

2.1 Overall Description

The automated redaction system is designed to process text documents, identify sensitive or personally identifiable information (PII), and replace such data with synthetic equivalent data. The system leverages Natural Language Processing (NLP) techniques to detect entities and ensures data privacy by generating synthetic alternatives through a Private Aggregation of Teacher Ensembles (PATE)-GAN model. This application is particularly useful for industries like healthcare, finance, and legal, where sensitive data must be handled securely and in compliance with regulations.

The primary objective of this system is to ensure privacy preservation in document-sharing processes without compromising the utility of the redacted documents. The system aims to automate and streamline the redaction process, which is traditionally manual, error-prone, and time-intensive.

2.2 Key Features

2.2.1 Named Entity Recognition (NER)

The system utilizes pre-trained and fine-tuned Named Entity Recognition (NER) models to detect sensitive information such as names, dates, addresses, and phone numbers. It also supports customizable entity detection to cater to domain-specific requirements, allowing users to tailor the recognition process for unique needs.

2.2.2 Synthetic Data Replacement

The system uses the PATE-GAN model to generate realistic synthetic data for replacing identified tokens, ensuring privacy by incorporating differential privacy techniques during data generation. This approach safeguards sensitive information while maintaining the integrity of the dataset.

2.2.3 Automation

The system fully automates the redaction process, from tokenizing the text document to generating the redacted output. This automation reduces manual effort and minimizes errors in handling sensitive data, ensuring efficiency and accuracy.

2.2.4 Customizable Redaction Rate

The system allows users to define which types of tokens need redaction and which can be retained, providing flexibility to adapt to various regulatory and operational requirements. This customization ensures the redaction process aligns with specific needs and compliance standards.

2.2.5 Scalable and Efficient Processing

The system is capable of handling large volumes of documents efficiently, making it suitable for high-scale operations. It is also designed for seamless integration into existing document management workflows, enhancing overall productivity without disrupting current processes.

2.2.6 User-Friendly Interface

Offers a simple interface for uploading documents, viewing detected entities, and previewing redacted outputs.

3. Operating Environment

3.1 Software Requirements

Operating System: Windows/MacOS

Libraries: Python 3.8+, PyTorch/TF 2.0+, Hugging Face Transformers, NumPy, Scikit-learn

Development Environment: Google Colab, VS Code

3.2 Hardware Requirements

Processor: Multi-core CPU

RAM: Minimum 8 GB

GPU: NVIDIA CUDA-enabled GPU with at least 8 GB VRAM

Storage: 512 GB SSD for storing large pre-trained models and datasets.

4. Functional Requirements

4.1 Multi-Format File Handling

The system should support file uploads in various formats, including text files (TXT, CSV, JSON) and both text-based and scanned PDFs. It should allow batch uploads for processing

multiple files at once and provide real-time feedback on the upload status, such as progress bars and error messages for unsupported formats.

4.2 Level of Redaction

Users can define the degree of redaction using a gradational scale which shows **Low** for minimal redaction with more context retained, **Medium** for a balanced approach that hides sensitive details while maintaining utility, and **High** for full redaction to ensure maximum privacy. Additionally, custom redaction rules can be configured based on specific patterns or entity types like names, emails, and addresses, providing precise control over the redaction process.

4.3 Text Extraction

PyPDF2 is used for text extraction from scanned documents. Automated text extraction involves leveraging Optical Character Recognition (OCR) to convert visual content in scanned PDFs into text. For text-based PDFs, PyPDF2 extracts content directly, ensuring the preservation of the original document's formatting. The focus is on achieving accurate extraction of both structured data (e.g., tables, lists) and unstructured data (e.g., paragraphs), maintaining the document's integrity and usability.

4.4 Redaction

Automatically identify and redact sensitive information using Named Entity Recognition (NER) for detecting Personally Identifiable Information (PII) such as names, emails, and phone numbers, as well as predefined rules and regular expressions for domain-specific redaction, including financial or legal terms. Apply redaction consistently across different file types, ensuring that text, visual elements, and metadata are properly anonymized. Additionally, visualize redacted content to allow users to review the changes before finalizing. This approach ensures thorough and accurate redaction while maintaining user oversight and control.

4.5 Output Document

Generate redacted outputs in the same format as the original file (e.g., PDF to redacted PDF). Provide users with an option to export redacted text in plain text or structured formats such as JSON. Ensure that the output retains structural similarity to the original document, maintaining

layout and format wherever possible. Additionally, log changes made during redaction for audit and compliance purposes. This approach ensures consistency, flexibility, and accountability in the document redaction process.

5. Non- Functional Requirements

5.1 Performance

The system must ensure efficient processing speeds for various file types to meet performance standards. For text-based files, the system should be capable of handling a rate of at least 1,000 tokens per second. Additionally, for PDFs, the processing speed should average 5 seconds per page, which includes Optical Character Recognition (OCR) processing time. These benchmarks are crucial for maintaining optimal performance and user satisfaction.

5.2 File Format Handling

Format Detection: Implement robust file format detection to identify and validate supported formats (PDF, TXT, DOCX) before processing.

5.3 Usability

User-Friendly Interface: The system features an intuitive and accessible interface with clear instructions, tooltips, and easy upload of files.

Maximum File Size:The system can efficiently handle file uploads of up to 200 MB without performance degradation

6. SYSTEM DESIGN

6.1 SYSTEM ARCHITECTURE

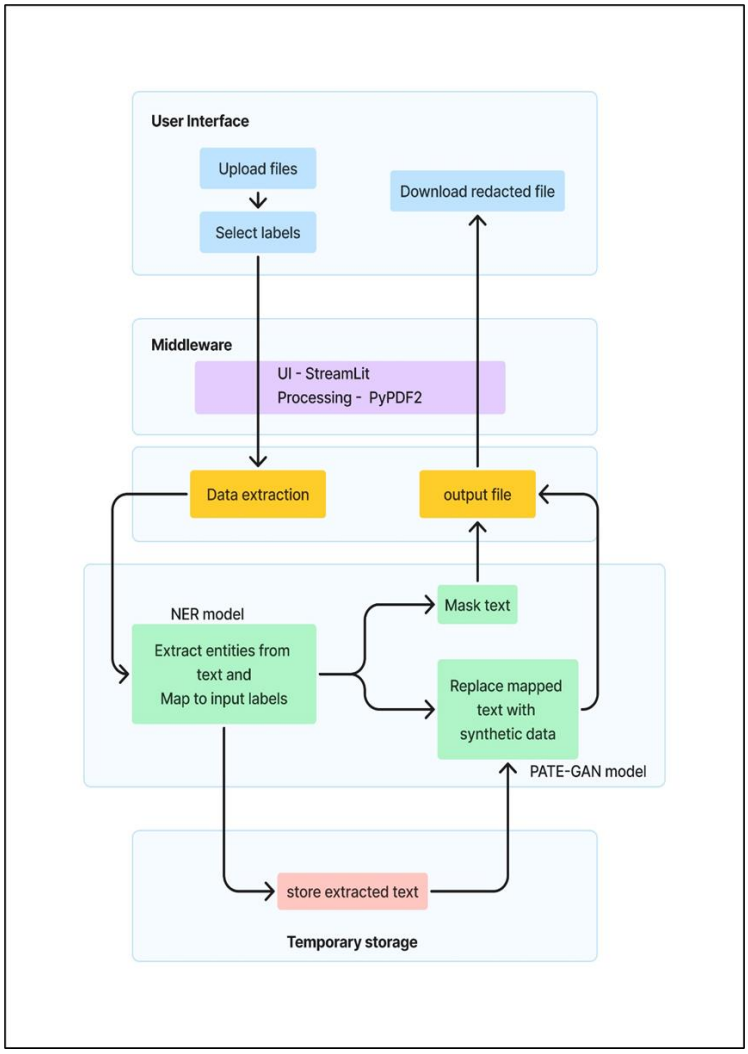

Fig 6.1 System Architecture

The above diagram (Fig 6.1) outlines the architecture of a redaction tool. The **User Interface (UI)** enables users to upload files, select specific labels for redaction, and download the processed files. The UI is built using **Streamlit**, with **PyPDF2** handling file processing. In the **Middleware**, data is extracted from the uploaded files, and an **NER model** identifies and maps entities based on the selected labels. The extracted text is either masked or replaced with synthetic data generated by a **PATE-GAN model**. The intermediate extracted text is stored

temporarily, and the final output file, containing redacted or anonymized data, is returned to the user for download.

6.2 DATA FLOW DIAGRAM(DFD)

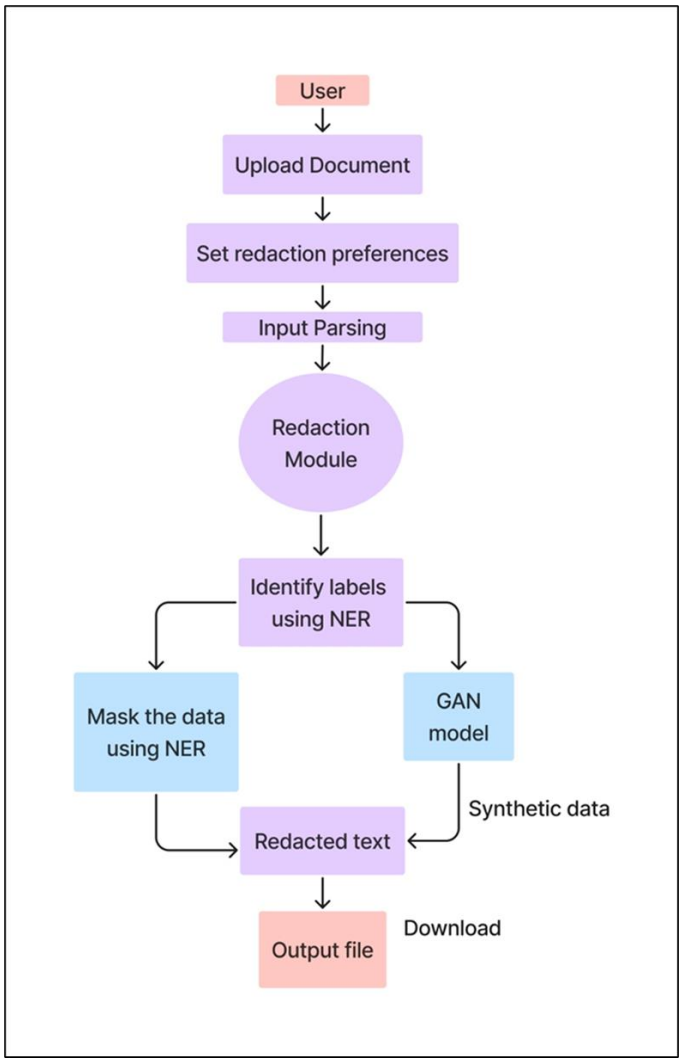

Fig 6.1 Data Flow Diagram

The above diagram(**Fig** 6.1) illustrates a streamlined process for document redaction. The user starts by uploading a document and setting their redaction preferences. Through **input parsing**, the data enters the **Redaction Module**, where an **NER (Named Entity Recognition)** model identifies the specified labels. The data is then either masked using the NER model or replaced

with synthetic data generated by a **GAN model**. The resulting redacted text is compiled into an **output file**, which the user can download.

6.3 USE-CASE DIAGRAM

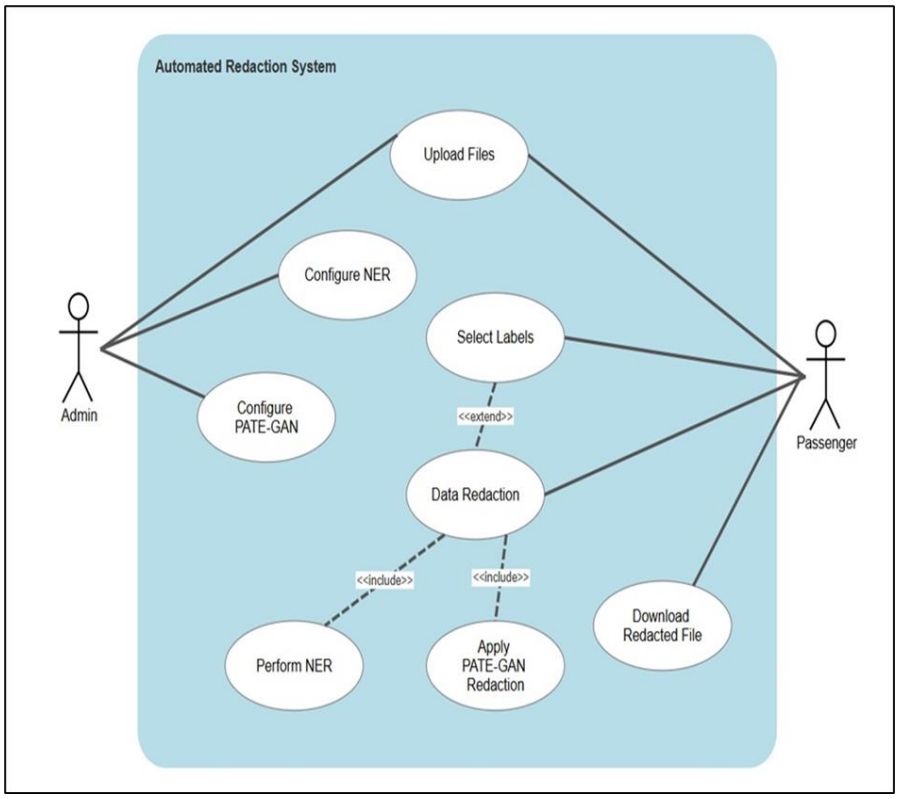

Fig 6.3 Use case Diagram

The use-case diagram(**Fig** 6.3) depicts an **Automated Redaction System** involving two user roles: **Admin** and **Passenger**. The Admin can upload files, configure the Named Entity Recognition (NER) model, and set up the PATE-GAN (Privacy-preserving Adversarial Training with GANs) model. Passengers can upload files and select labels for redaction. The core functionality, **Data Redaction**, includes performing NER to detect sensitive data, applying PATE-GAN for privacy-preserving redaction, and enabling users to download the redacted files. This system streamlines the process of secure and automated data redaction, ensuring sensitive information is handled effectively.

6.4 CLASS DIAGRAM

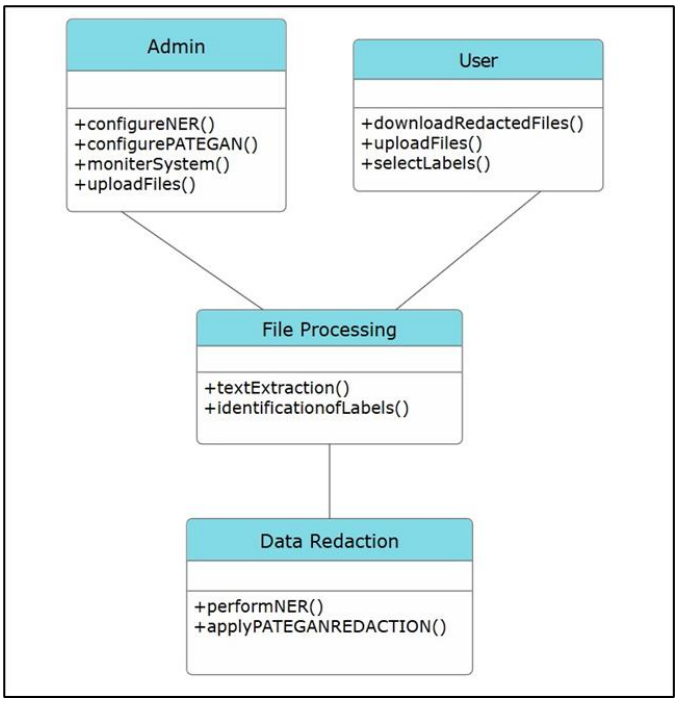

Fig 6.4 Class Diagram

This class diagram**(Fig** 6.4) outlines the structure of an Automated Redaction System. The Admin class includes functionalities such as configuring the NER and PATE-GAN models, monitoring the system, and uploading files. The User class allows file uploads, label selection, and downloading redacted files. Both interact with the File Processing class, which handles text extraction and label identification. Finally, the Data Redaction class performs NER and applies PATE-GAN-based redaction to ensure sensitive data is securely handled.

5. SEQUENCE DIAGRAM

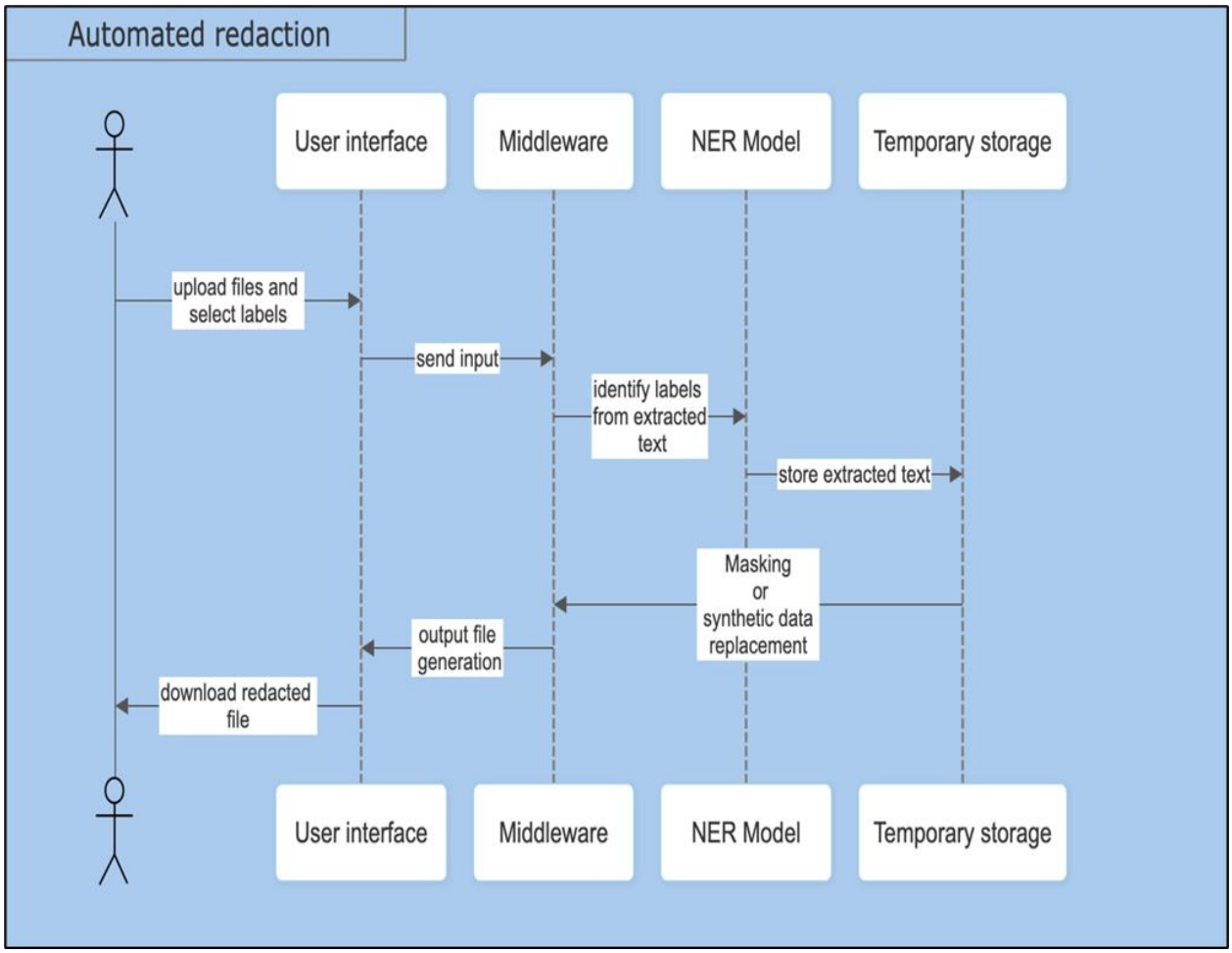

Fig 6.5 Sequence Diagram

This sequence diagram(**Fig** 6.5) illustrates the workflow of an **Automated Redaction System**. The user uploads files and selects labels through the user interface, which sends the input to the middleware. The middleware interacts with the NER model to identify labels from the extracted text and stores the extracted text in temporary storage. The system then performs masking or synthetic data replacement based on the identified labels, generating the redacted output file. Finally, the user can download the redacted file via the user interface.