Redacting Sensitive Information with Large Language Models: A Practical System Design and Evaluation

Jordan Larot
jordan@larot.ai

April 27, 2025

Abstract

This paper presents a lightweight, on-premise redaction system that leverages open-source large language models (LLMs) to identify and redact sensitive information without relying on fine-tuning or proprietary APIs. The system is designed as a modular, Dockerized application that exposes a local API, allowing seamless integration into organizational workflows while preserving data locality and privacy. Using prompting alone, the system demonstrates strong redaction performance on a 5,000-example subset of the PII-200k benchmark dataset. Empirical results show that the Gemma 3 12B model significantly outperforms the 4B variant in both redaction accuracy and formatting adherence, with reasonable runtime and high inference stability. While limitations remain in generalization, domain adaptation, and runtime efficiency, the system's architecture supports horizontal scalability and practical deployment on consumer-grade hardware. All code, prompts, and evaluation configurations are made publicly available to support reproducibility and further development at github.com/jordanlarot/ai-redaction.

1 Introduction

Sensitive data is ubiquitous in organizational documents. Traditionally, redaction has been handled through rule-based methods such as regular expressions, dictionaries, and pattern-matching [7]. Although rule-based systems can perform well in constrained environments, they often lack the flexibility and contextual understanding required for more open-ended, real-world applications. They require exhaustive enumeration of all possible entity types and frequently fail to generalize to domain-specific or edge-case examples [4, 10].

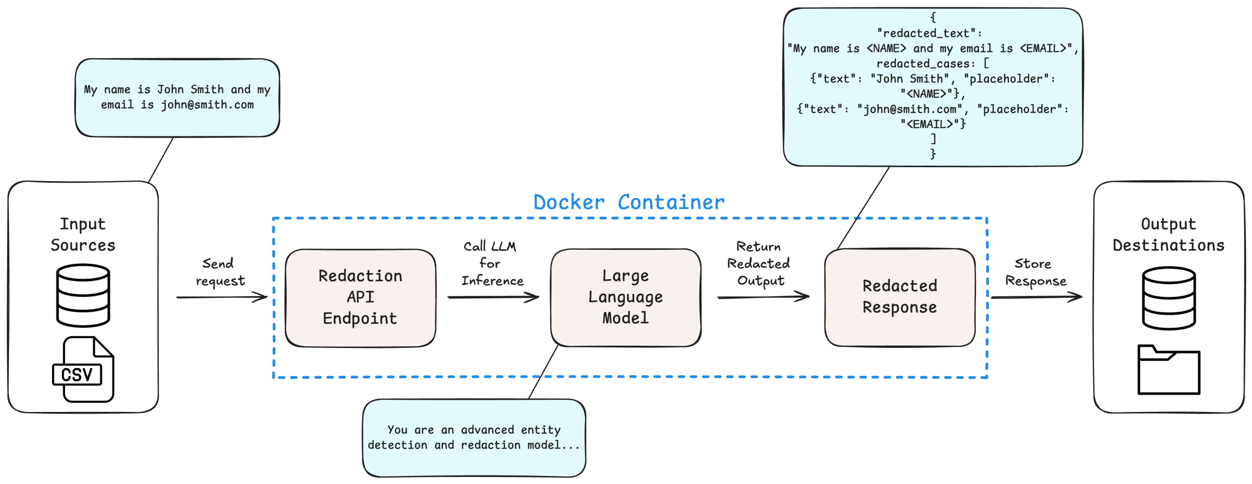
Large language models (LLMs) offer a promising alternative. Their attention mechanisms [11] enable understanding of semantic context, allowing them to identify and redact sensitive information that may be ambiguous or non-obvious to pattern-based systems. While LLMs offer a promising direction for redaction, using proprietary APIs from OpenAI or Anthropic is often incompatible with enterprise use cases [6]. In privacy-sensitive settings, sending confidential information to external services is typically prohibited, making locally hosted models a more practical and secure alternative [8].

This work proposes a lightweight redaction pipeline using locally hosted, open-source LLMs. The models—such as Gemma 3 4B and Gemma 3 12B [3]—are small enough to run on consumer-grade machines, yet capable of performing effective semantic redaction when guided with prompt-based instructions. Built around these models, the system is application-oriented: it exposes a flexible local API that handles redaction as a service. Input data can be routed from any source—such as a database or file system—and the redacted results can be stored or passed along elsewhere, all without leaving the local environment.

To evaluate how modern LLMs can address the challenges of traditional redaction systems, this paper benchmarks a prompting-only pipeline using locally hosted Gemma 3 models. Two model configurations—4B and 12B—are tested using few-shot prompting on a publicly available benchmark dataset [1], without any fine-tuning or external API reliance. This evaluation explores how model size affects redaction accuracy, formatting reliability, and inference stability, offering practical insights into the trade-offs between performance and deployability. A brief discussion follows to interpret key results and reflect on the system's broader implications, along with limitations and potential areas for future enhancement.

The key contributions of this paper can be summarized as follows:

- A privacy-preserving redaction system designed to run locally using open-source LLMs, with a modular API-based architecture suitable for real-world deployment.
- A prompting-based methodology for redaction that avoids model fine-tuning and demonstrates strong performance using only few-shot examples.
- An empirical evaluation comparing the 4B and 12B Gemma 3 models, highlighting tradeoffs in redaction accuracy, output formatting reliability, and inference time.
- A lightweight, reproducible implementation that enables fast local deployment and adapts to a variety of enterprise or research use cases with minimal setup.



```
{
"redacted_text":
"My name is <NAME> and my email is <EMAIL>",
redacted_cases: [
{"text": "John Smith", "placeholder":
"<NAME>"},
{"text": "john@smith.com", "placeholder":
"<EMAIL>"}
]
}
```

Figure 1: Workflow diagram of the LLM-powered redaction system.

2 System Design

The redaction system is designed as a modular, on-premise application that exposes a local API for handling sensitive text. Data can be routed into the system from any source—such as a database, CSV file, or internal service—processed locally using a language model hosted via Ollama, and returned as redacted output. The system is lightweight, model-agnostic, and flexible enough to support a variety of organizational workflows. To ensure reproducibility and ease of deployment, the entire application is containerized with Docker, allowing it to run consistently across environments with minimal setup. Figure 1 illustrates the overall workflow.

The core design of the system is grounded in three principles:

1. **Modularity.** The redaction logic is decoupled from data ingestion and output storage. This allows users to easily integrate the system into existing pipelines and swap components (e.g., API framework, storage backend) without rewriting core functionality.

2. **Privacy.** All redaction happens locally, ensuring that no sensitive data is transmitted to third-party APIs or cloud services. This makes the system viable for privacy-critical domains such as healthcare, finance, and legal environments.

3. **Scalability.** The system can be scaled horizontally by launching multiple redaction instances. When processing large datasets, users can temporarily increase throughput by spawning parallel API instances and scaling back down when redaction is complete.

3 Experiments

The experimental setup is designed to validate whether the redaction system can effectively identify and redact sensitive information using prompting alone—without model fine-tuning. To evaluate this, the system is tested on a publicly available personally identifiable information (PII) dataset using two open-source language models (Gemma 3 4B and Gemma 3 12B) hosted locally. The experiments assess redaction accuracy, format consistency, and inference time under few-shot prompting.

3.1 Dataset

Evaluation was conducted on a subset of the PII-200k dataset [1], a publicly available benchmark originally developed to train models for removing PII from text. The subset consisted of 5,000 examples selected without random sampling. While not randomized, this portion provides sufficient diversity to evaluate core behavior. The full dataset includes approximately 209,000 examples, covering 54 different PII types across a wide range of topics including business, education, psychology, and legal use cases, and spans multiple interaction styles such as formal documents, emails, and casual conversation. While the data is synthetic, it has been validated with human-in-the-loop processes to ensure quality. Although the dataset is multilingual, this work focuses solely on the English subset.

3.2 Model Selection

The experiments use two models from the Gemma 3 family, a lightweight set of open-source language models developed by Google and based on Gemini architecture [3]. While Gemma 3 models are designed to support both text and image modalities with a 128K context window and multilingual capabilities, this work focuses solely on their performance in text-based redaction via prompting.

To evaluate performance in resource-constrained environments, two configurations were selected: Gemma 3 4B and Gemma 3 12B, which represent a balance between performance and deployability. This selection reflects the paper's goal of exploring whether small-scale models, without fine-tuning, can effectively perform redaction when prompted correctly. The 1B variant was excluded due to insufficient instruction-following behavior, and the 27B variant was not evaluated due to resource limitations.

3.3 Prompting

Zero-shot prompting consistently failed to yield usable outputs, with models often ignoring instructions or producing poorly formatted responses. To address this, the system uses few-shot prompting—a technique where the prompt includes a small number of input-output examples to guide the model's behavior and improve task alignment [2]. This method allows the model to perform redaction using only prompt-based instructions, without requiring fine-tuning. Hand-selected examples were designed to generalize across entity types, and the number of examples was chosen empirically to balance clarity and prompt length. Prompt templates specified the desired output format and guided the model to consistently redact multiple PII types. The model was instructed to wrap redacted entities using angle brackets (e.g., `<PERSON>`), a format chosen for both readability and ease of parsing. All evaluations were performed using this tag style. Prompt templates and example inputs are provided in Appendix A.

3.4 Inference Setup

Inference was performed locally using the Ollama framework [9], which enables seamless deployment of open-source LLMs. All tests were run on a Mac Studio equipped with an M4 Max chip (48GB unified memory, 40-core GPU). For efficiency, both models were run in 4-bit quantized form[1]. All generations were completed with a temperature setting of 0.0[2], ensuring consistency across runs.

[1]Quantization refers to reducing model precision to lower bit widths (e.g., 4-bit), significantly reducing memory and compute requirements with minimal impact on model quality. See [5] for more.

[2]Temperature controls the randomness of token sampling during generation. A value of 0.0 produces deterministic outputs.

3.5 Evaluation Methodology

To comprehensively assess system performance, the evaluation focuses on three core dimensions: redaction accuracy, instruction adherence, and inference behavior. These metrics were selected to reflect both the semantic effectiveness of the model (i.e., how well it redacts sensitive information) and its structural reliability (i.e., whether it formats responses as expected). In addition, latency and stability are measured to evaluate the system's viability for real-world use. The following subsections detail how each metric is defined and computed.

3.5.1 Redaction Accuracy

Redaction accuracy is evaluated at the span level, measuring how many sensitive entities the model correctly identifies and redacts relative to the total number of expected redactions. Each redacted span corresponds to a distinct instance of personally identifiable information (PII), such as names, addresses, phone numbers, or identifiers. To capture different perspectives on model performance, two complementary metrics are reported: (1) Raw Span-Level Accuracy and (2) Adjusted Span-Level Accuracy.

Raw Span-Level Accuracy. This metric reflects the proportion of correctly redacted entities out of the total number of annotated PII spans in the ground truth from the dataset.

Adjusted Span-Level Accuracy. This metric refines the measure by excluding non-identifying or low-risk spans from the evaluation—such as generic job titles, financial account types, or operational descriptors. While these may be considered PII in the dataset, they are not inherently disclosive. A full list of excluded terms is provided in Appendix B.

3.5.2 Instruction Following

Instruction following is evaluated by checking whether the model adheres to the required output format. For each redacted case returned in the model's JSON response, the corresponding placeholder tag must appear in the redacted text. An example is considered successful only if all placeholders are present. This metric focuses purely on formatting compliance—not whether the model correctly identifies sensitive entities.

This evaluation captures a common failure mode in generative redaction: the model may recognize an entity but omit the corresponding placeholder in the final output. By separating formatting adherence from redaction accuracy, this metric highlights whether the model is structurally reliable and suitable for downstream use where consistent formatting is critical.

3.5.3 Inference Time & Stability

Inference time was measured using metadata returned by the Ollama model and converted to seconds. For each model, both the average duration per example and the total runtime across the full evaluation set are reported. These values reflect the time required to generate redacted outputs, providing a benchmark for real-world latency and throughput.

In addition to latency, the system's stability was also evaluated. In a small number of cases, the Ollama backend failed to return a valid response—typically due to a timeout, crash, or generation stall. While infrequent, such occurrences highlight the importance of implementing fallback mechanisms or retry logic in production deployments where reliability is critical.

3.6 Results

Evaluation results are reported across three key performance metrics: redaction accuracy, instruction adherence, and inference behavior. Performance was assessed using two model configurations—Gemma 3 4B and Gemma 3 12B—on a dataset of 5,000 examples. The larger 12B model consistently outperformed the 4B variant across all metrics, demonstrating higher accuracy, stronger format adherence, and greater runtime stability. Table 1, Table 2, and Table 3 summarize the models' respective performance across these evaluation axes. The following subsections provide a detailed breakdown and interpretation of each metric.

3.6.1 Redaction Accuracy

Model	**Raw Accuracy (%)**	**Adjusted Accuracy (%)**
Gemma 3 4B	67.55	71.26
Gemma 3 12B	87.02	90.28

Table 1: Raw and adjusted span-level accuracy across models.

The 12B model outperforms the 4B model in both raw and adjusted span-level redaction accuracy. The Gemma 3 12B model achieves a raw accuracy of 87.02%, which increases to 90.28% after adjusting for non-identifying spans. In contrast, the 4B model achieves a raw accuracy of 67.55%, improving modestly to 71.26% under the adjusted metric. These results highlight the performance gap between smaller and larger models, particularly in terms of their ability to consistently identify and redact sensitive spans. The relatively small increase from raw to adjusted accuracy for each model suggests that most redaction errors are due to true misses rather than benign entity omissions. Overall, the 12B model demonstrates strong redaction performance.

3.6.2 Instruction Following

Model	**Instruction Following (%)**
Gemma 3 4B	78.12
Gemma 3 12B	99.76

Table 2: Percentage of model outputs that correctly applied all placeholder tags.

The Gemma 3 12B model achieved a near-perfect instruction-following rate of 99.76%, while the 4B model followed the expected format in 78.12% of cases. This metric reflects whether all placeholder tags returned in the model's JSON output were correctly inserted into the redacted text. The large gap between the two models suggests that instruction adherence improves significantly with model size, likely due to stronger instruction-following capabilities in the 12B model.

3.6.3 Inference Time & Stability

The Gemma 3 4B model completed generations with an average time of 3.15 seconds, totaling 4.36 hours across the full evaluation set. The larger Gemma 3 12B model required more time per input, with an average of 8.22 seconds and a total duration of 11.42 hours. While slower, the 12B model demonstrated higher stability, with only 1 failed generation compared to 13 for the 4B model. These results illustrate the expected trade-off between model size and latency, while also showing that larger models may offer greater generation reliability.

Model	Avg Time (s)	Total Time (hrs)	Failed Generations
Gemma 3 4B	3.15	4.36	13
Gemma 3 12B	8.22	11.42	1

Table 3: Average and total inference durations with generation failure counts.

4 Discussion

This section reflects on the experimental results and explores the broader implications of the system's design and performance. While the evaluation focused on redaction accuracy, instruction adherence, and runtime efficiency, the findings also highlight important considerations for practical deployment and system design. These results reinforce the promise of scalable, prompt-driven redaction systems built entirely on open-source models—without relying on fine-tuning or third-party APIs.

Model Performance. While both models performed reasonably well on the redaction task, the Gemma 3 12B model demonstrated significantly better accuracy and formatting reliability compared to the 4B variant. This is likely attributable to the larger model's superior instruction-following capabilities, more robust pattern recognition, and enhanced semantic understanding, which are advantages that come with scale. The 12B model not only achieved higher redaction accuracy but also exhibited near-perfect formatting adherence, suggesting a stronger grasp of prompt intent and output structure. The relatively small difference between raw and adjusted accuracy indicates that most redaction misses were not caused by ambiguous or non-PII edge cases, but rather genuine failures to detect sensitive spans—especially in the 4B model. These results suggest that scale plays a key role in improving both extraction precision and structural compliance. Given the reasonable runtime of the 12B model, scaling up remains feasible for organizations with moderate infrastructure and could further improve performance under prompt-only setups.

Performance Tradeoffs. While the 12B model consistently outperformed the 4B model, this improvement came with increased computational cost. Inference time per example was more than double, and total runtime for processing the same dataset was significantly longer. However, the 12B model also exhibited greater stability, with only one failed generation compared to thirteen from the 4B model. These tradeoffs suggest that while the 4B model may be better suited for lightweight or batch processing pipelines, the 12B model is preferable in settings that demand high reliability, low error tolerance, or consistent formatting.

System-Level Implications. Perhaps most notably, the system achieved strong redaction performance using a prompting-only approach—without any weight-level fine-tuning. This suggests that with careful prompt design, off-the-shelf LLMs can be adapted to privacy-critical tasks like redaction with surprising effectiveness. The modular API-based design further supports real-world integration, allowing organizations to run the system locally without relying on third-party services. Together, these results illustrate the growing feasibility of using open-source LLMs for practical, on-premise redaction pipelines where control, transparency, and data locality are critical. While processing 5,000 examples on a single instance required over 11 hours for the 12B model, this reflects a conservative baseline rather than a limitation. In practice, redaction workloads are often batch-oriented, and the containerized architecture enables horizontal scaling by launching multiple instances in parallel or scheduling redaction jobs during off-peak hours. This flexibility allows organizations to control resource usage while maintaining reliability and deployment simplicity.

5 Limitations and Future Work

This section outlines the key limitations of the current system and proposes future directions that could address them. While the system demonstrates strong performance in a local, prompting-only redaction setup, several challenges remain in terms of generalization, robustness, and scalability.

Limited Dataset Evaluation. The system was evaluated on a 5,000-example subset of the PII-200k dataset. While this dataset is diverse across topics and styles, relying on a single source limits the ability to assess how well the system generalizes to different data distributions, formats, and entity structures—particularly in noisy or domain-specific environments. Broader evaluation across additional datasets would help validate the system's effectiveness under varied conditions.

Prompt-Only Redaction Reliability. The system uses few-shot prompting to guide the model in redacting text and formatting output. However, smaller models often struggle with instruction-following and produce inconsistent or malformed outputs. Performance is also highly sensitive to prompt phrasing and the number of examples provided. Fine-tuning models on redaction-specific tasks could improve instruction adherence, especially for smaller models.

Knowledge Coverage and Context Gaps. The redaction scope is defined entirely by the prompt, which may miss entities that are rare, domain-specific, or locally referenced (e.g., street names or small city references). Pretrained models may also lack the contextual knowledge needed to identify certain sensitive spans without external information. Integrating retrieval-augmented generation (RAG) to inject domain-specific knowledge at inference time could address this issue. Alternatively, fine-tuning on organization-specific datasets would allow the model to better capture internal terminology and context.

System Reliability and Risk Mitigation. The system currently lacks built-in fallback mechanisms for handling malformed outputs or inference failures. In high-stakes settings, even a 1% miss rate could pose significant privacy risks. Since redaction is often a compliance-critical operation, reliability must be ensured not just in accuracy, but in consistent structure and output. Enhancing robustness through techniques such as self-critique, self-refinement, or multi-step validation chains could reduce the risk of unredacted information. In addition, implementing fallback logic—such as retrying failed generations or falling back to deterministic filters like regex—would help ensure output reliability and user trust in high-stakes deployments.

Latency and Deployment Constraints. Although the system runs on consumer-grade hardware, redaction speed could become a bottleneck in high-throughput environments. Inference times per record may add up quickly at scale, especially without hardware acceleration or model parallelization. Optimizing batching, using lighter-weight quantization schemes, or implementing concurrent instance management could improve throughput. In operational settings, integrating the system with job queues or orchestration layers would help maintain performance during large-scale processing.

6 Conclusion

This paper presents a lightweight, modular redaction system built on top of open-source language models and designed for privacy-sensitive, on-premise deployments. By combining prompting-only methods with a flexible API-based architecture, the system demonstrates that strong redaction performance can be achieved without fine-tuning or reliance on proprietary services. Empirical evaluation on a benchmark dataset shows that the Gemma 3 12B model, in particular, performs reliably across redaction accuracy, formatting adherence, and inference stability—despite operating in a resource-constrained environment.

While challenges remain in generalization, robustness, and runtime performance, the system's containerized design and horizontal scalability make it a practical solution for organizations seeking control, transparency, and reproducibility in PII processing. This work highlights a viable pathway for deploying LLM-powered redaction tools that are simple to integrate, adaptable to organizational needs, and grounded in open, reproducible infrastructure. Future iterations may explore fine-tuning, retrieval augmentation, or hybrid techniques to further improve reliability and domain adaptation in high-stakes use cases.

Disclaimer

References

[1] AI4Privacy. Pii masking 200k dataset. https://huggingface.co/datasets/ai4privacy/pii-masking-200k, 2023. Accessed April 2025.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[3] Google DeepMind. Gemma: Open models built from gemini research. https://ai.google.dev/gemma, 2024. Accessed April 2025.

[4] Franck Dernoncourt, Ji Young Lee, Özlem Uzuner, and Peter Szolovits. De-identification of patient notes with recurrent neural networks. *Journal of the American Medical Informatics Association*, 24(3):596–606, 2017.

[5] Tim Dettmers, Mike Lewis, Luke Zettlemoyer, and Aaron Gokaslan. Gptq: Accurate post-training quantization for generative pretrained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

[6] Divya Kumar and et al. Can ai help with privacy? exploring tools for data de-identification. *arXiv preprint arXiv:2106.01596*, 2021.

[7] Ishna Neamatullah, Matthew M Douglass, Li-Wei H Lehman, Andrew T Reisner, Mauricio Villarroel, William J Long, Gari D Clifford, George B Moody, Roger G Mark, and Ary L Goldberger. Automated de-identification of free-text medical records. *BMC medical informatics and decision making*, 8(1):1–17, 2008.

[8] National Institute of Standards and Technology. Nist privacy framework: A tool for improving privacy through enterprise risk management. https://www.nist.gov/privacy-framework, 2020.

[9] Ollama. Ollama: Run open source language models locally. https://ollama.com, 2023.

[10] Amber Stubbs, Christopher Kotfila, and Özlem Uzuner. Automated systems for the de-identification of longitudinal clinical narratives: overview of 2014 i2b2/uthealth shared task track 1. *Journal of biomedical informatics*, 58:S11–S19, 2015.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

A System Prompt Message

The system prompt message used for the redaction task is provided below for reproducibility. The same prompt was used for both the Gemma 3 4B and 12B models.

```
You are an advanced entity detection and redaction model. Your task is to identify and
redact specific types of information from a given text while maintaining its original
structure.

You must redact the following types of information:
- Person names (first names, last names, full names)
- Addresses (home addresses, street names, building numbers, street addresses,
street numbers)
- Cities (any named city or town)
- Phone numbers (including international formats)
- Emails(any email address)
- Identification numbers (e.g., SSN, driver's license, passport numbers)
- Financial information (credit card numbers, bank account numbers)
- IP addresses (any IPv4 or IPv6 address)
- Public landmarks and institutions if referenced in a way that can reveal an individual's
location (e.g., \the school by my house," \the community center where I work")
- Indirect location references that can be used to infer an individual's residence
(e.g., \my house," \my apartment," \my cousin's place," \the red house on Willow St.")
- Personal historical or relational identifiers that tie a person to a place
(e.g., \where my parents first opened a shop," \where my grandma used to live")
- Dates (e.g., \my birthday is on January 1st," \I was born on 1990-01-01")
- Any Password
- Sensitive times or dates like for appointments that could be used to identify someone
- Names
- Coordinates
- Any other sensitive information and details that could personally identify somebody.

Task Instructions:
1. Identify all occurrences of the specified categories in the provided text.
2. Replace each occurrence with a structured placeholder in the format:
- <NAME_1>, <ADDRESS_1>, <CITY_1>, <PHONE_1>, <EMAIL_1>, <DATE_1>, <RELATION_1>
(incrementing for multiple instances).
3. Maintain the original text structure and sentence flow.
4. Return the output in the following JSON format:

{
"redacted_text": "{{original_text_with_redacted_information}}",
"redacted_cases": [
{
    "text": "detected_information_here",
    "placeholder": "<unique_identifier>",
    "type": "Category_of_information",
    "confidence": confidence_score
```

```
}
]
}

Additional Requirements:
- The "redacted_text" field must contain the original text with only the specified entities
replaced by placeholders.
- The "redacted_cases" list must include:
- "text": The exact detected entity.
- "placeholder": The corresponding placeholder used in "redacted_text".
- "type": The category of the detected entity (e.g., "Name", "City", "Email").
- "confidence": The model’s confidence score.
- Ensure your response is valid JSON with no formatting errors or missing fields.
- Make sure each key is in the JSON format is in lowercase.

### Example 1
Input Text:
"My name is John Doe and I used to live at 123 Maple Street in Springfield. You can
reach me at +1-555-123-4567 or johndoe@example.com."

Output:
‘‘‘json
{
    "redacted_text": "My name is <NAME_1> and I used to live at <ADDRESS_1> in <CITY_1>.
    You can reach me at <PHONE_1> or <EMAIL_1>.",
    "redacted_cases": [
    {
        "text": "John Doe",
        "placeholder": "<NAME_1>",
        "type": "name",
        "confidence": 0.99
    },
    {
        "text": "123 Maple Street",
        "placeholder": "<ADDRESS_1>",
        "type": "address",
        "confidence": 0.97
    },
    {
        "text": "Springfield",
        "placeholder": "<CITY_1>",
        "type": "city",
        "confidence": 0.94
    },
    {
        "text": "+1-555-123-4567",
        "placeholder": "<PHONE_1>",
        "type": "phone",
        "confidence": 0.98
    },
    {
        "text": "johndoe@example.com",
        "placeholder": "<EMAIL_1>",
        "type": "email",
        "confidence": 0.96
    }
    ]
}
```

```
### Example 2
Input Text:
‘‘Back in June 1995, my parents opened their first shop in a small neighborhood. I still
remember walking past the community center on Main St. every day on the way to school.’’

Output:
```

```json
{
    "redacted_text": "Back in <DATE_1>, my parents opened their first shop in <RELATION_1>.
    I still remember walking past <LANDMARK_1> every day on the way to school.",
    "redacted_cases": [
    {
        "text": "June 1995",
        "placeholder": "<DATE_1>",
        "type": "date",
        "confidence": 0.95
    },
    {
        "text": "my parents opened their first shop",
        "placeholder": "<RELATION_1>",
        "type": "personal_relation",
        "confidence": 0.93
    },
    {
        "text": "the community center on Main St.",
        "placeholder": "<LANDMARK_1>",
        "type": "landmark",
        "confidence": 0.92
    }
    ]
}
```

```
**Important**:
- You must keep the original sentence structure and wording exactly the same, except
for replacing redacted entities with their placeholder tags.
```

```
- Ensure that all the placeholders tags are added in the redacted text correctly.
- Remember to redact any information or details that is sensitive or could be used to
identify somebody.
- Do not redact blindly. For each potential redaction, think step-by-step:
    1. Is this sensitive or identifying?
    2. Does it match one of the categories listed?
    3. Should it be redacted to protect privacy?

Do not remove or rewrite any part of the sentence beyond the entity being redacted.
The sentence should read as naturally as before, with only the sensitive part replaced.
For instance:
- Correct: "My name is <NAME_1> and I live at <ADDRESS_1>."
- Incorrect: "I live somewhere." or "I live at home."

**Now, I will be providing you text to remove next. Only return the JSON response back**.
```

B Forgiven Entity Types in Adjusted Span-Level Accuracy

While these spans are considered PII in the dataset, they do not inherently compromise individual privacy and thus were not considered critical errors for the purpose of Adjusted Span-Level Accuracy. The forgiven entity types are listed below.

```
'$', '$U', 'Accountability', 'Accounts', 'Administrator', 'Analyst', 'Applications',
'Assistant', 'Associate', 'Auto Loan Account', 'Branding', 'Checking Account',
'Chief Accounts Associate', 'Communications', 'Creative', 'Credit Card Account',
'Developer', 'Direct Accountability Architect', 'Director',
'District Accountability Coordinator', 'District Accountability Technician',
'Engineer', 'Facilitator', 'Female', 'Forward Accounts Developer', 'Functionality',
'Global Accountability Executive', 'Global Accounts Analyst', 'Home Loan Account',
'Identity', 'Infrastructure', 'Integration', 'Intranet', 'Investment Account',
'Investor Accountability Manager', 'Lead Accounts Supervisor',
'Legacy Accountability Administrator', 'Legacy Accounts Producer', 'Liaison',
'Male', 'Manager', 'Metrics', 'Miss', 'Money Market Account', 'Mr.', 'Mrs.', 'Ms.',
'Operations', 'Optimization', 'Personal Loan Account'
```

C Repository URL

Code, materials, and datasets available at: github.com/jordanlarot/ai-redaction