Demo: Sanitizing Medical Documents with Differential Privacy using Large Language Models

Rushil Thareja[1], Gautam Gupta[2], Preslav Nakov[1], Praneeth Vepakomma[1,3], Nils Lukas[1]
[1]Mohamed bin Zayed University of Artificial Intelligence (MBZUAI)
[2]Indraprastha Institute of Information Technology Delhi (IIIT Delhi)
[3]Massachusetts Institute of Technology (MIT)
first_name.last_name@mbzuai.ac.ae

Abstract

Medical documents often contain sensitive information such as disease history and symptoms. Regulations like GDPR strictly prohibit leakage of such content. A natural solution is to sanitize documents with large language models (LLMs) before sending them to untrusted providers. However, LLM-based paraphrasing remains vulnerable to membership inference attacks (MIA), which can reveal what private tokens were present in the input. Differentially Private Inference (DPI) offers formal guarantees against such leakage, but standard approaches severely degrade utility. Recent methods improve trade-offs by applying DP only to private tokens, yet this requires accurate tagging of private spans. In practice, privacy in medical text is highly context dependent and varies across organizations/jurisdictions, leading existing taggers to perform poorly. LLM-based taggers achieve higher accuracy but require costly fine-tuning and risk leaking private data through memorization. We address this by introducing *constitutional classifiers* for private information tagging. Here, we learn a *constitution* i.e a set of natural language rules, directly from a small annotated subset, achieving stronger performance than existing taggers while requiring no fine-tuning. Importantly, the learned rules remain interpretable and auditable, allowing human experts to verify or edit them for compliance. We integrate our constitutional tagger with DPI through DP-Fusion, yielding an end-to-end pipeline for utility-preserving medical document sanitization using LLMs. The system is deployed and publicly available at www.documentprivacy.com.

1 Introduction

Healthcare datasets are uniquely vulnerable to privacy leakage, as they contain highly sensitive personal information and are frequently exposed through third-party breaches. Recent reports show that healthcare accounts for over a quarter of all disclosed data breaches, with a significant fraction originating from vendors [1]. Such incidents illustrate how even indirect access paths can compromise patient confidentiality. In this context, deploying large language models (LLMs) over medical records raises acute risks: private tokens such as names, diagnoses, or treatment identifiers may be inferred or reconstructed from model outputs [16], resulting in regulatory violations and erosion of patient trust.

Consider, for example, a hospital that wants to deploy LLMs to assist users in matching their symptoms to historical records from a large document dataset. An LLM could help retrieve relevant case histories, suggest possible diagnostic procedures, or summarize treatment outcomes across similar patients. Such functionality has clear utility in supporting medical staff and improving patient care. However, these records inevitably contain sensitive information, including patient demographics, medical histories, and unique symptom descriptions. Regulations such as GDPR classify health data under a *special category* of highly sensitive information (Article 9), and any leakage of such data to untrusted parties (e.g., through the use of closed-box LLM APIs) is strictly prohibited [17, 11].

A solution is to first paraphrase documents with an LLM and then share the sanitized outputs with external providers [8, 18, 15]. Yet, paraphrasing alone does not eliminate leakage: private tokens can still be inferred from the transformed text, through existing membership attacks [16]. Differentially Private Inference (DPI) offers protection against such leakage, but existing methods severely reduce utility by introducing noise into the LLM output distribution [7, 18]. Recent work [16] achieves stronger trade-offs by restricting guarantees to only the private tokens, as identified by a tagger.

Therefore, the tagger is critical, as it determines which tokens in a document are actually covered under the DP guarantees offered through DPI. Existing pre-trained taggers [9, 23] focus on a narrow set of personally identifiable information (PII), which is insufficient for health data, where diseases, symptoms, and clinical details are also privacy-sensitive [2]. Moreover, definitions and requirements of privacy vary across data types, organizations, and regulatory contexts. Fine-tuning taggers is an option, but it raises three major concerns: (i) limited availability of annotated medical data and sufficient compute, (ii) risk of memorization and leakage through model weights, and (iii) lack of transparency, making models non-auditable, a critical issue for compliance.

To address this, we adopt *constitutional classifiers*, a technique from LLM safety [13] that enables robust classification using a set of natural language rules (a constitution), which are then applied by an LLM for tagging. A good constitution should balance precision and recall while covering diverse edge cases, but creating such a rule set is difficult for experts and must be tailored separately for each dataset. In contrast, private text annotation is far easier for organizations to perform. We therefore aim to *learn* the constitution automatically from these annotations using a proposed multi-agent framework, rather than relying on manually supplied rules.

2 Background

Membership Inference Attacks (MIA) [24] aim to decide whether a record was part of the model's input by analyzing its output. In the context of paraphrasing a document d_i, the adversary inspects the generated text d_i' to test if a private token x appeared in d_i. Two methods have demonstrated robust performance in the context of LLMs:

The *Min-k% attack* [14] computes the average log-likelihood of the least probable k% tokens in a sequence $x = (x_1, \ldots, x_N)$: $\text{MIN-}k\%\text{PROB}(x) = \frac{1}{|\text{Min-}k\%(x)|} \sum_{x_i \in \text{Min-}k\%(x)} \log \Pr(x_i \mid x_{<i})$.

The *LOSS attack* [22] instead relies on perplexity, assuming that lower loss implies membership.

To defend against such attacks, Differentially Private Inference (DPI) adapts the classical notion of DP [3] to model outputs, ensuring that the influence of any single token on the generated text is provably bounded [16]. Therefore, DPI provides a mechanism to prevent adversaries from inferring whether a private token t was present in d_i by observing its paraphrase d_i'.

Definition 1 (Differentially Private Inference (DPI))**.** *Let $m : \mathcal{X} \to \mathcal{Y}$ be a prediction model and fix $\varepsilon > 0$, $\delta \in [0, 1]$. A randomized algorithm A provides (ε, δ)-Differentially Private Inference for m if the induced mechanism $\mathcal{M}(D) = A(m, D) \in \mathcal{Y}, \quad D \in \mathcal{X}$, satisfies the (ε, δ)-DP guarantee [3]:*

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] + \delta, \quad \forall\, D, D' \text{ neighbors}, \; S \subseteq \mathcal{Y}. \tag{1}$$

While traditional DPI approaches in LLMs often degrade utility [18, 7], recent work has shown that focusing guarantees only on sensitive spans can preserve utility while still bounding leakage. We adopt this approach through *DP-Fusion* [16], a token-level DPI mechanism that provides practical privacy without producing incoherent outputs. In DP-Fusion, the document D is decomposed into a public component X_{pub} (with private spans removed) and m private groups $X_1, \ldots, X_m$, each containing disjoint sensitive tokens. For each decoding step t, we compute logits on the public context p_{pub}, and on each private variant $p_{\text{priv},i}$, which is the distribution obtained from the public document with the tokens of group X_i reinserted. Then, a mollification step selects $\lambda_i \geq 0$ such that the Rényi divergence (D_α) [10] is bounded, i.e $D_\alpha\big(\lambda_i p_{\text{priv},i} + (1 - \lambda_i) p_{\text{pub}} \,\big\|\, p_{\text{pub}}\big) \leq \alpha\beta_i$, with $\alpha\beta_i$ the per-group budget [4]. By bounding Rényi divergence, the mechanism satisfies differential privacy (Eq. 1), since Rényi DP can be reduced to standard (ε, δ)-DP [10]. Finally, the fused distribution is then averaged: $p_{\text{final}} = \frac{1}{m} \sum_{i=1}^{m} (\lambda_i p_{\text{priv},i} + (1 - \lambda_i) p_{\text{pub}})$ and the next token d_t is sampled as $d_t \sim p_{\text{final}}$. Repeating this process produces the privatized paraphrase of the document i.e D'.

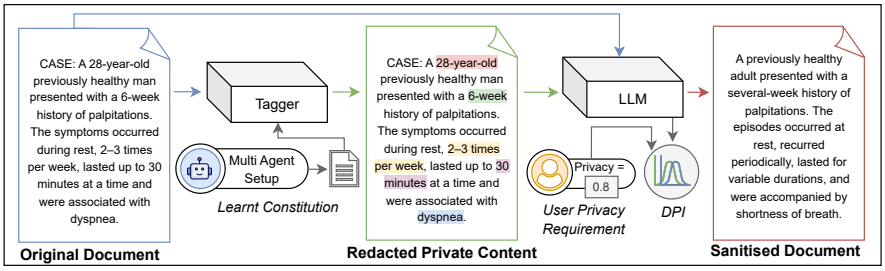


Figure 1: Overview of our medical dataset privatization pipeline.

3 Methodology

An overview of our proposed end-to-end pipeline for health document sanitization with differential privacy is shown in Figure 1. Uploaded documents are first parsed locally and passed through a tagger that marks private phrases, which are then converted to tokens by tokenization; since we can move interchangeably between phrases and tokens, we use the two terms synonymously throughout the paper. This produces two versions of the text: a *public* one with sensitive spans redacted, and the original document. We then apply a single-group variant of DP-Fusion, distinct from the existing implementation [16], that interpolates directly between the private (original) and public (redacted) distributions under a user-specified privacy parameter $\alpha\beta$, which can be converted to the standard (ε, δ) DP guarantee (1) via Theorem 4 in [16]. This yields higher utility, lower computational cost, and smoother privacy–utility trade-offs compared to group-level allocations.

3.1 Tagging Private Tokens

As a core component of our private token tagging method, we use an `Annotator` Agent prompted with a learned constitution, i.e., a set of natural language rules in system prompts to guide tagging. Recent work on constitutional classifiers [13] shows that this constitution, can drive downstream LLM tasks with high accuracy while letting auditors inspect and modify the rules directly.

Prior work assumes an external *oracle* (e.g., human annotators) defines the constitution. From stakeholder interactions, we find this difficult: medical practitioners struggle in designing rules that yield low FN yet high F1 when applied by an LLM tagger for DP-Fusion. Thus, inspired by recent work on using LLMs as optimizers [21], we *learn* a constitution from a small, hand-labelled subset where stakeholders annotate spans of private information in text. Annotating such spans is considerably easier for stakeholders than writing explicit rules, and this process implicitly encodes their notion of privacy without requiring them to define the rules directly, which our learned constitution then aims to capture. We also include a baseline with a human-expert authored constitution created by an expert, but our learned constitution outperforms this version.

To learn the constitution we employ a multi-agent setup. An `Annotator` Agent applies the constitution to mark private tokens. A `Processor` Agent then evaluates these predictions to identify false negatives (FN) and false positives (FP). A `Decision` Agent determines whether the next update should be an *add*, *remove*, or *edit* operation. When the action requires modifying an existing rule (*remove* or *edit*), a `Target Selector` Agent identifies the exact rule to update. Finally, a `Rule Editor` Agent applies the chosen update using the context of the observed errors. All agents produce *reasoning for their actions*, which is passed along to guide the subsequent agent. To preserve explainability, a `Summariser` Agent collects the reasonings from all agents and produces a single consolidated justification for each action. The agentic optimisation proceeds in an epoch–batch setup analogous to standard training loops.

The complete algorithm for constitution learning is given in Algorithm 1. Full prompts for the `Annotator`, `Decision`, `Target Selector`, and `Rule Editor` agents are provided in Appendices E, F, G, and H, respectively. The `Processor` is a rule-based system that employs bipartite matching to compute metrics, with details shown in Algorithm 2 in the Appendix. We use GPT-4.1 as the underlying LLM for this agentic framework, though recent open-source models with comparable performance can also be used locally [20].

Algorithm 1 Multi-Agent Framework for Constitution Learning

function BUILDCONSTITUTION($D_{\text{train}}, D_{\text{val}}, \{G_i\}, D_R, E$)
▷ D_{train}: training docs; D_{val}: validation docs; $\{G_i\}$: gold phrase labels; D_R: rule budget; E: epochs
  $C, C^\star \leftarrow \varnothing$; F1$^\star \leftarrow 0$; $\mathcal{L} \leftarrow \varnothing$; Shuffle D_{train} into batches $\{B_j\}$ ▷ Init Constitution, Logs, and Batches
  for epoch $= 1$ to E **do** ▷ Iterating across document subset to learn the constitution
    for each batch B_j **do**
      $P \leftarrow$ ANNOTATOR(B_j, C) ▷ Loops over all docs in B_j and return predicted private phrases.
      $\{FN, FP\} \leftarrow$ PROCESSOR$(P, \{G_i : d_i \in B_j\})$ ▷ Computes FN/FP (with context) for B_j
      $(a, r_{\text{dec}}) \leftarrow$ DECISIONAGENT(FN, FP, C, D_R) ▷ Chooses ADD, REMOVE, EDIT and reason
      if $a \in \{\text{EDIT}, \text{REMOVE}\}$ **then**
        $(t, r_{\text{tgt}}) \leftarrow$ TARGETSELECTOR(a, C, FN, FP, r_{dec}) ▷ Agent picks which rule to modify
      else
        $t \leftarrow \perp$; $r_{\text{tgt}} \leftarrow \varnothing$
      end if
      $(C, r_{\text{rule}}) \leftarrow$ RULEEDITOR$(C, a, t, FN, FP, r_{tgt})$ ▷ Applies the actual update
      SUMMARISER$(P, C, a, t, r_{\text{dec}}, r_{\text{tgt}}, r_{\text{rule}})$; $\mathcal{L} \leftarrow \mathcal{L} \cup \{\cdot\}$ ▷ Logs decisions and reasoning
    end for
    $P_v \leftarrow$ ANNOTATOR(D_{val}, C) ▷ The Annotator tags the validation set with the proposed constitution
    $\{FN_v, FP_v, \text{F1}\} \leftarrow$ PROCESSOR$(P_v, \{G_i : d_i \in D_{\text{val}}\})$ ▷ Computes metrics for the validation set
    if F1 $>$ F1* **then** $C^\star \leftarrow C$; F1$^\star \leftarrow$ F1 ▷ Track best constitution on validation
    end if
  end for
  return $C^\star, \mathcal{L}$ ▷ Return best constitution, final predictions, and audit log
end function

4 Experiments

4.1 Dataset

Statistic	Value
Documents	200
Private Chars	316,712 (56.0%)
Public Chars	249,004 (44.0%)
Private Entities	25,189
Private Entity Groups	41
Avg. Entities / Group	614.37
Avg. Chars / Group	7,724.68
Avg. Chars / Entity	12.57

To approximate real-world medical documents, we evaluate on the MACCROBAT dataset for medical private information [2]. This dataset includes 49 private information types, such as `Biological structure`, `Detailed description` and `Diagnostic procedure`. It is curated for clinical natural language processing tasks, providing annotated private entities. Full statistics are provided in Table 4.1, and an example annotated document is included in Appendix A.

4.2 Performance of Tagger

We construct a dataset of 176 documents (20 batches of 8) for constitution learning, 16 documents (2 batches of 8) for validation, and 16 documents (2 batches of 8) for testing, corresponding to an apx. 85:8:8 split of the full 192 documents (after discarding 8 very short documents with fewer than 500 characters). To baseline existing pretrained taggers, we evaluate widely used PII taggers from the Microsoft Presidio library [9], which has been used in prior work [15]. We select the best performing models within Presidio: BERT-NER (`dslim/bert-base-NER`), SpaCy (`en_core_web_lg`), and Flair (`flair/ner-english-ontonotes-large`).

Additionally, we benchmark an LLM-based tagger fine-tuned for NER, the 7B LLaMA model from UNI-NER [23]. All methods are evaluated on the same held-out test split. For our constitutional classifier (Algorithm 1), we run for 15 epochs. We compare two setups: an expert-supplied constitution derived from manual inspection and annotation, and a learned constitution learnt through our multi-agent optimisation framework. The complete evaluation results are reported in Table 2. Our constitutional tagger achieves the best overall performance, with higher recall and F1 than the expert-supplied constitution. Therefore, learning rules through our multi-agent optimisation framework yields stronger tagging accuracy without tuning the model.

4.3 Performance of Full Pipeline

We use our Constitutional Classifier (Learnt) to mark private tokens, which are then passed to DP-Fusion as shown in Figure 1. We adopt the same experimental setting as DP-Fusion and use the same QWEN 2.5 7B model [12] for paraphrasing.

Table 1: DP-Fusion with our Constitutional Tagger (Learnt Const.), Mean ASR (*Min-k%*, and *LOSS*) and cosine similarity to the original document.

$\alpha\beta$	LOSS	Min5%	Min10%	Min20%	Min40%	Cosine Sim.
0.1	15.0	11.7	13.3	15.0	13.3	0.6546
1.0	16.7	15.0	15.0	16.7	16.7	0.6629
5.0	18.3	16.7	16.7	18.3	18.3	0.7222
10.0	20.0	18.3	18.3	18.3	20.0	0.8452

Table 2: Performance of taggers on MACCROBOT

Tagger	**P**	**R**	**F1**
UNI-NER (Llama)	**0.896**	0.443	0.583
Microsoft Presidio (SpaCy)	0.400	0.100	0.154
Microsoft Presidio (Flair)	0.543	0.031	0.055
Microsoft Presidio (BERT-NER)	0.187	0.014	0.024
Constitutional Classifier (Expert)	0.858	0.470	0.607
Constitutional Classifier (Learnt)	0.884	**0.545**	**0.674**

Following DP-Fusion, we measure privacy via the attack success rate (ASR), computed under two attacks, Min-k and LOSS (Section 2). All attacks are mounted on the 4 most frequent entity groups: `Biological structure`, `Detailed description`, `Diagnostic procedure`, and `Sign/symptom` with candidate set size $|C| = 5$. For utility, we compute cosine similarity between the privatized and original documents using MiniLM-v2 embeddings [19]. Prior work has shown this to be a reliable proxy for paraphrase quality [6], whereas LLM-based judges and perplexity can vary significantly. This metric was not included in the original DP-Fusion implementation. Table 1 reports results for different values of $\alpha\beta$. This parameter specifies the maximum allowed divergence acting as the privacy–utility trade-off knob. As $\alpha\beta$ increases, ASR rises across all attacks ($\downarrow$ privacy), while cosine similarity to the original document also increases ($\uparrow$ utility). Lower theoretical (DP) privacy budgets increase leakage but yield outputs that more closely resemble the input text i.e are expected to have higher-downstream utility.

5 Demonstration

We deploy the complete document sanitization pipeline at www.documentprivacy.com, with a focus on medical text. The workflow proceeds as follows: (1) Users upload documents to be sanitized; the parser supports `.txt`, `.pdf`, `.docx`, and `.pptx` formats. (2) Users configure two components: the tagger and the paraphrasing LLM. The tagger (our constitutional classifier or Presidio models: SpaCy, BERT, Flair) identifies private tokens. The LLM is then used with DP-Fusion to generate privatized paraphrases; in deployment, this is restricted to use the 3B QWEN-2.5 model, which offers lower performance but runs efficiently on available GPUs. (3) The system provides an overview of detected private entities, which would be covered under the theoretical (DP) guarantees. (4) The user specifies a privacy requirement in $[0, 1]$, internally mapped to $[0.1, 10]$ and passed as $\alpha\beta$ to DP-Fusion. (5) The paraphrase is generated iteratively, with token-progress and ETA displayed. (6) Finally, a side-by-side comparison of the original and privatized documents is shown, with the option to download the output. A full visual overview of the demonstration is showcased in Appendix I.

6 Conclusion

In this paper, we address privacy leakage from medical documents when processed with external, and often untrusted, GenAI providers. This problem poses a barrier to building robust and policy-compliant GenAI technologies for healthcare. To provide sanitization with theoretical guarantees, we construct a pipeline that first employs a tagger to identify private tokens and then applies Differentially Private Inference (DPI), through DP-Fusion, which paraphrases the text using an LLM while bounding leakage of these marked private tokens. A key limitation of existing approaches lies in the inaccuracy of current taggers in medical contexts. Therefore, we propose constitutional classifiers, where the constitution of rules is learned through a multi-agent optimization framework. This is then used by an annotator agent to reliably mark privacy-sensitive spans in medical documents. By integrating accurate tagging with token-level DPI, our end-to-end document sanitization pipeline enables the use of medical documents with potentially untrusted or non-compliant GenAI providers while ensuring that patient privacy is not compromised.

At present, our experiments focus on a closed-box model (GPT-4.1). Future work will extend the framework to open-source models deployed locally, and we are actively testing the transferability of constitutions across different LLMs. We are also working on comparisons with fine-tuning approaches and establishing stronger baselines, as well as improving the pipeline to more efficiently identify optimal constitutions.

References

[1] Steve Alder. Healthcare experiences more third-party data breaches than any other sector. https://www.hipaajournal.com/healthcare-highest-third-party-breaches/, March 2024. Accessed: 2025-08-28.

[2] J. Harry Caufield. Maccrobat2020 dataset. https://doi.org/10.6084/m9.figshare.9764942.v2, 2020. Version 2 of the MACCROBAT2018 dataset on Figshare.

[3] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[4] James Flemings, Meisam Razaviyayn, and Murali Annavaram. Differentially private next-token prediction of large language models. *arXiv preprint arXiv:2403.15638*, 2024.

[5] John E Hopcroft and Richard M Karp. An n^5/2 algorithm for maximum matchings in bipartite graphs. *SIAM Journal on computing*, 2(4):225–231, 1973.

[6] Hiu Ting Lau and Arkaitz Zubiaga. Understanding the effects of human-written paraphrases in llm-generated text detection. *Natural Language Processing Journal*, page 100151, 2025.

[7] Jimit Majmudar, Christophe Dupuy, Charith Peris, Sami Smaili, Rahul Gupta, and Richard Zemel. Differentially private decoding in large language models, 2022. URL https://arxiv.org/abs/2205.13621.

[8] Justus Mattern, Benjamin Weggenmann, and Florian Kerschbaum. The limits of word level differential privacy. *arXiv preprint arXiv:2205.02130*, 2022.

[9] Microsoft. Presidio: Data protection and de-identification sdk. https://microsoft.github.io/presidio/, 2025. Accessed: 2025-08-28.

[10] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pages 263–275. IEEE, 2017.

[11] Miranda Mourby, Katharina Ó Cathaoir, and Catherine Bjerre Collin. Transparency of machine-learning in healthcare: The gdpr & european health law. *Computer Law & Security Review*, 43: 105611, 2021.

[12] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

[13] Mrinank Sharma, Meg Tong, Jesse Mu, Jerry Wei, Jorrit Kruthoff, Scott Goodfriend, Euan Ong, Alwin Peng, Raj Agarwal, Cem Anil, et al. Constitutional classifiers: Defending against universal jailbreaks across thousands of hours of red teaming. *arXiv preprint arXiv:2501.18837*, 2025.

[14] Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023.

[15] Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. Large language models are advanced anonymizers. *arXiv preprint arXiv:2402.13846*, 2024.

[16] Rushil Thareja, Preslav Nakov, Praneeth Vepakomma, and Nils Lukas. Dp-fusion: Token-level differentially private inference for large language models, 2025. URL https://arxiv.org/abs/2507.04531.

[17] Maria Tzanou. *Health Data Privacy Under the GDPR*. Taylor Francis Limited, 2023.

[18] Saiteja Utpala, Sara Hooker, and Pin-Yu Chen. Locally differentially private document generation using zero shot prompting. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8442–8457, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.566. URL https://aclanthology.org/2023.findings-emnlp.566.

[19] Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. *arXiv preprint arXiv:2012.15828*, 2020.

[20] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[21] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*, 2023.

[22] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018.

[23] Wenxuan Zhou, Sheng Zhang, Yu Gu, Muhao Chen, and Hoifung Poon. Universalner: Targeted distillation from large language models for open named entity recognition. *arXiv preprint arXiv:2308.03279*, 2023.

[24] Zhanke Zhou, Jianing Zhu, Fengfei Yu, Xuan Li, Xiong Peng, Tongliang Liu, and Bo Han. Model inversion attacks: A survey of approaches and countermeasures, 2024. URL https://arxiv.org/abs/2411.10023.

A Example Annotated Document from MACCROBAT

To illustrate the dataset and annotation process, we include an example document from the MACCROBAT corpus [2]. The document is shown with its annotated private spans highlighted. This provides a representative view of the type of medical narratives used for constitution learning and evaluation.

CASE: A 28-year-old previously healthy man presented with a 6-week history of palpitations . The symptoms occurred during rest , 2–3 times per week , lasted up to 30 minutes at a time and were associated with dyspnea . Except for a grade 2/6 holosystolic tricuspid regurgitation murmur (best heard at the left sternal border with inspiratory accentuation), physical examination yielded unremarkable findings. An electrocardiogram (ECG) revealed normal sinus rhythm and a Wolff– Parkinson– White pre-excitation pattern (Fig.1: Top), produced by a right-sided accessory pathway . Transthoracic echocardiography demonstrated the presence of Ebstein's anomaly of the tricuspid valve , with apical displacement of the valve and formation of an " atrialized " right ventricle (a functional unit between the right atrium and the inlet [inflow] portion of the right ventricle) (Fig.2). The anterior tricuspid valve leaflet was elongated (Fig.2C, arrow), whereas the septal leaflet was rudimentary (Fig.2C, arrowhead). Contrast echocardiography using saline revealed a patent foramen ovale with right-to-left shunting and bubbles in the left atrium (Fig.2D). The patient underwent an electrophysiologic study with mapping of the accessory pathway , followed by radiofrequency ablation (interruption of the pathway using the heat generated by electromagnetic waves at the tip of an ablation catheter). His post-ablation ECG showed a prolonged PR interval and an odd "second" QRS complex in leads III, aVF and V2–V4 (Fig.1Bottom), a consequence of abnormal impulse conduction in the " atrialized " right ventricle . The patient reported no recurrence of palpitations at follow-up 6 months after the ablation.

Age, Biological_structure, Clinical_event, Coreference, Date, Detailed_description, Diagnostic_procedure, Disease_disorder, Duration, Frequency, History, Lab_value, Sex, Sign_symptom, Therapeutic_procedure

Figure 2: Example MACCROBAT document with annotated private spans.

B Measuring Tagger Performance

Simple string-based matching can lead to incorrect counts of false negatives and false positives. To address this, we align predicted and ground-truth spans via maximum bipartite matching [5], as outlined in Algorithm 2.

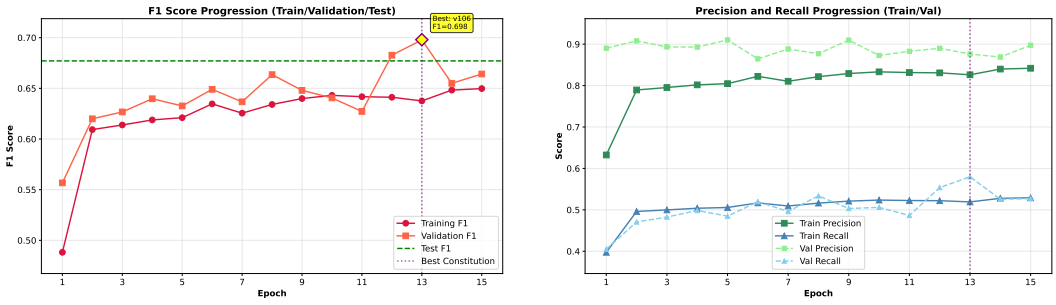
Algorithm 2 One-to-One Substring Matching for P/R/F1

Require: Predicted phrases $P = \{p_1, \ldots, p_m\}$, Gold phrases $G = \{g_1, \ldots, g_n\}$
Ensure: Precision P, Recall R, F1, counts TP, FP, FN, TN

function CANONICALIZE(s)
  return lowercase(NFKC(s)) with collapsed whitespace and unified dashes
end function
function MATCHES(p,g)
  $p \leftarrow$ CANONICALIZE(p); $g \leftarrow$ CANONICALIZE(g)
  return (g is a substring of p)
end function
Build bipartite graph: left nodes $1..m$ (predicted), right nodes $1..n$ (gold)
Adj$[u] \leftarrow \varnothing$ for all $u \in \{1, \ldots, m\}$
for $u \leftarrow 1$ to m **do**
  for $v \leftarrow 1$ to n **do**
    if MATCHES(p_u, g_v) **then**
      add edge $(u \rightarrow v)$ to Adj$[u]$
    end if
  end for
end for
Maximum matching: (pairU, pairV, TP) $\leftarrow$ HOPCROFTKARP(Adj, m, n) ▷ Any maximum-cardinality bipartite matching algorithm is fine.
FP $\leftarrow m -$ TP ▷ predicted with no matched gold
FN $\leftarrow n -$ TP ▷ gold with no matched prediction
TN $\leftarrow 0$ ▷ undefined in open-set extraction
if TP + FP > 0 **then**
  $P \leftarrow$ TP/(TP + FP)
else
  $P \leftarrow 0$
end if
if TP + FN > 0 **then**
  $R \leftarrow$ TP/(TP + FN)
else
  $R \leftarrow 0$
end if
if $P + R > 0$ **then**
  F1 $\leftarrow 2PR/(P + R)$
else
  F1 $\leftarrow 0$
end if
return $(P, R,$ F1, TP, FP, FN, TN)

C Learning a constitution

Constitutional learning shows consistent gains over 15 epochs, with F1 improving from 0.488 to 0.698 (43% relative). Performance rises quickly in early epochs and stabilizes near epoch 13 (constitution v106). Precision remains high (>0.9) throughout, while recall steadily increases, indicating effective rule refinement without loss of accuracy.



(a) F1 Score Progression

(b) Precision and Recall Progression

Figure 3: Constitution learning progress showing performance metrics across epochs. The purple line marks the best validation performance (epoch 13, constitution v106) with 103 rules.

D Expert-Supplied Constitution

The expert constitution was derived through an iterative process where domain experts inspected data, drafted natural-language rules, tested them in annotation trials, and refined based on observed false positives and false negatives. Their pipeline for deriving a constitution served as inspiration for our multi-agent framework for rule induction.

Expert supplied constitution

1. **Span policy:** Extract minimal standalone tokens; prefer exact mention boundaries. If uncertain, choose the shorter atomic span.
2. **Demographics / IDs:** Mark exact age phrases, sex/gender terms, ethnicity/religion, fine-grained locations/facilities, and family relations; atomic only. *Private:* age phrase, gender term. *Not:* generic role noun, broad region.
3. **Time & quantitative markers:** Mark numeric dates, durations, frequencies, ages-at-event, doses, sizes, counts, grades, and lab values. Exclude vague time words and non-clinical numbers; atomic only. *Private:* numeric duration or dose. *Not:* vague time word.
4. **Clinical conditions & events:** Mark disease/syndrome names, signs/symptoms, lifestyle/exposure/history, outcomes/status, and medical encounter/event terms in clinical context. Include severity/acuity modifiers attached to these; atomic only. *Private:* disease or symptom token. *Not:* everyday verb out of context.
5. **Anatomy / phenotype / morphology & care setting:** Mark anatomical sites (including laterality/regions), cell types, morphology descriptors, and care/scene-of-care phrases linked to care; atomic only. *Private:* site with laterality or morphology token. *Not:* laterality word alone.
6. **Meds / procedures / tests / markers:** Mark drug names/brands/regimens, named procedures/surgeries/biopsies, imaging/lab/IHC markers, and device/guidance terms. Exclude bare words like "test/examination" without modality/target; atomic only. *Private:* drug or named test token. *Not:* generic "test".

E Prompt for Annotator Agent

The annotator agent is guided by a structured system prompt that embeds the constitution rules and instructs the model to extract all private phrases from a given document. The complete prompt is shown below.

System prompt for annotator agent

System: You are a constitutional privacy annotation agent. You will be given some rules in natural language in the form of a constitution, use that to annotate private leaking stuff. Apply the constitution rules systematically to identify ALL private phrases with maximum precision and recall.

User: `CONSTITUTION RULES:` {{CONSTITUTION_TEXT}}

`ANNOTATION TASK:` Apply constitution rules systematically to extract ALL matching private phrases:

```
{
 "private_phrases": ["phrase1", "phrase2", "phrase3"]
}
```

Now based on the provided rules annotate the below provided text to annotate privacy leaking phrases. Whenever confused you should stick to marking something as private, because it is always safer to mark something not private as private, rather than marking something private as not. Basically, it is better to optimise for FN (false negatives) being lower than FP (false positives).
The rules will have examples; understand and use them. But do not memorise and mark ONLY based on examples, you should always extrapolate and try to estimate what is private. If no constitution is available, just predict based on your estimate.

Text: {{TEXT_CONTENT}}

F Prompt for Decision Agent

We design a decision agent that explicitly optimizes for *practical utility*. The full system prompt is lengthy and omitted here for brevity, but its key logic can be summarised as follows:

1. **Error Balance Analysis:** The agent first compares false negatives (FN) and false positives (FP). - If FP greatly outnumbers FN (e.g., 3+ FP per FN), the system is too aggressive and over-flags. - If FN is high but FP is reasonable, the system is too conservative and misses privacy.
2. **Trend Interpretation:** The agent considers recent error trends. - If FP is rising faster than FN is falling, rules are becoming too broad. - If F1 declines or both FN and FP stay high, existing rules are ineffective.
3. **Constitution Rule Check:** The agent examines whether: - FN phrases are already covered but poorly specified, - FP phrases result from overly broad rules, or - missing coverage leaves clear gaps.
4. **Strategic Action Choice:** - **ADD** when FN is problematic but FP remains controlled. - **REMOVE** when FP dominates and harms usability. - **EDIT** when both FN and FP arise in the same semantic area and can be refined.
5. **Step-by-Step Reasoning:** The agent must explain (i) the current FN–FP balance, (ii) the main issue, and (iii) why the chosen action best addresses it.

The final output is returned in structured JSON:

```
{
  "action": "add|edit|remove",
  "reasoning": "Step-by-step analysis:

  [1] Current balance assessment
  [2] Main problem identified
  [3] Why this action best addresses it"
}
```

G Prompt for Target Selector Agent

The role of the Target Selector Agent is to choose *which* rule in the constitution should be modified once a strategic action (ADD, REMOVE, or EDIT) has been selected. The full prompt is omitted for brevity; its logic can be summarised as follows:

1. **Error Attribution:** The agent analyses how each rule contributes to false negatives (FN) and false positives (FP). - High-FP rules are overly broad or vague, often lacking qualifiers. - Rules linked to FN patterns may be too narrow or missing context.
2. **Removal Case:** If the action is **REMOVE**, the agent selects the rule that: - Produces the most false positives, - Uses general patterns with little context restriction, - Contributes little to reducing FN, and - Would significantly cut false alarms if removed.
3. **Edit Case:** If the action is **EDIT**, the agent selects the rule that: - Partially addresses FN but also generates FP, - Has boundary issues (catching both private and non-private spans), - Can be refined by adding qualifiers or narrowing scope, - Offers clear improvement potential with minimal changes.
4. **Selection Criteria:** Across both cases, the chosen rule should maximize impact on FN/FP balance, show a clear connection to observed error patterns, and yield the greatest utility gain with a single modification.
5. **Step-by-Step Reasoning:** For the selected rule, the agent must explain (i) how it contributes to current FN/FP errors, (ii) what effect removal or refinement would have, and (iii) why it is preferred over alternatives.

The final output is returned in structured JSON:

```
{
  "rule_index": <number>,
  "reasoning": "Analysis:
  [1] Contribution to errors
  [2] Expected improvement from action
  [3] Why this rule over others"
}
```

H Prompt for Rule Editor Agent

The Rule Editor Agent is responsible for creating and refining constitutional rules once a specific action and target have been identified. Its goal is to maximize privacy detection performance through systematic rule design while maintaining domain generality and explainability. The full prompt is omitted for brevity, but its logic can be summarized as follows:

1. **Domain-Adaptive Rule Design:** The agent derives rules that generalize across documents of the same domain, avoid overfitting to specific samples, and capture intrinsic properties of privacy-sensitive spans.
2. **Create New Rules:** When tasked with **ADD**, the agent: - Identifies domain-relevant privacy patterns, - Generalizes them to apply across similar documents, - Focuses on intrinsic properties that make spans private, - Designs concise rules that scale to large datasets.
3. **Edit Existing Rules:** When tasked with **EDIT**, the agent: - Targets the specified rule, - Performs gap analysis to identify precision/recall issues, - Applies minimal yet effective modifications, - Ensures refinements expand coverage without introducing regressions.
4. **Design Principles:** - Rules must be interpretable, concise ($<$300 characters), and domain-scoped. - Rule text must never contain actual private content. - Focus on improving recall while preserving precision. - Output rules are standardized natural language instructions.
5. **Output Format:** The agent always responds in strict JSON with a single optimized rule, e.g.:

   ```
   {"rule_text": "your single optimized rule here"}
   ```

I Demonstration

We provide a visual walkthrough of the deployed system for medical document sanitization. The full workflow is illustrated in Figure 4.

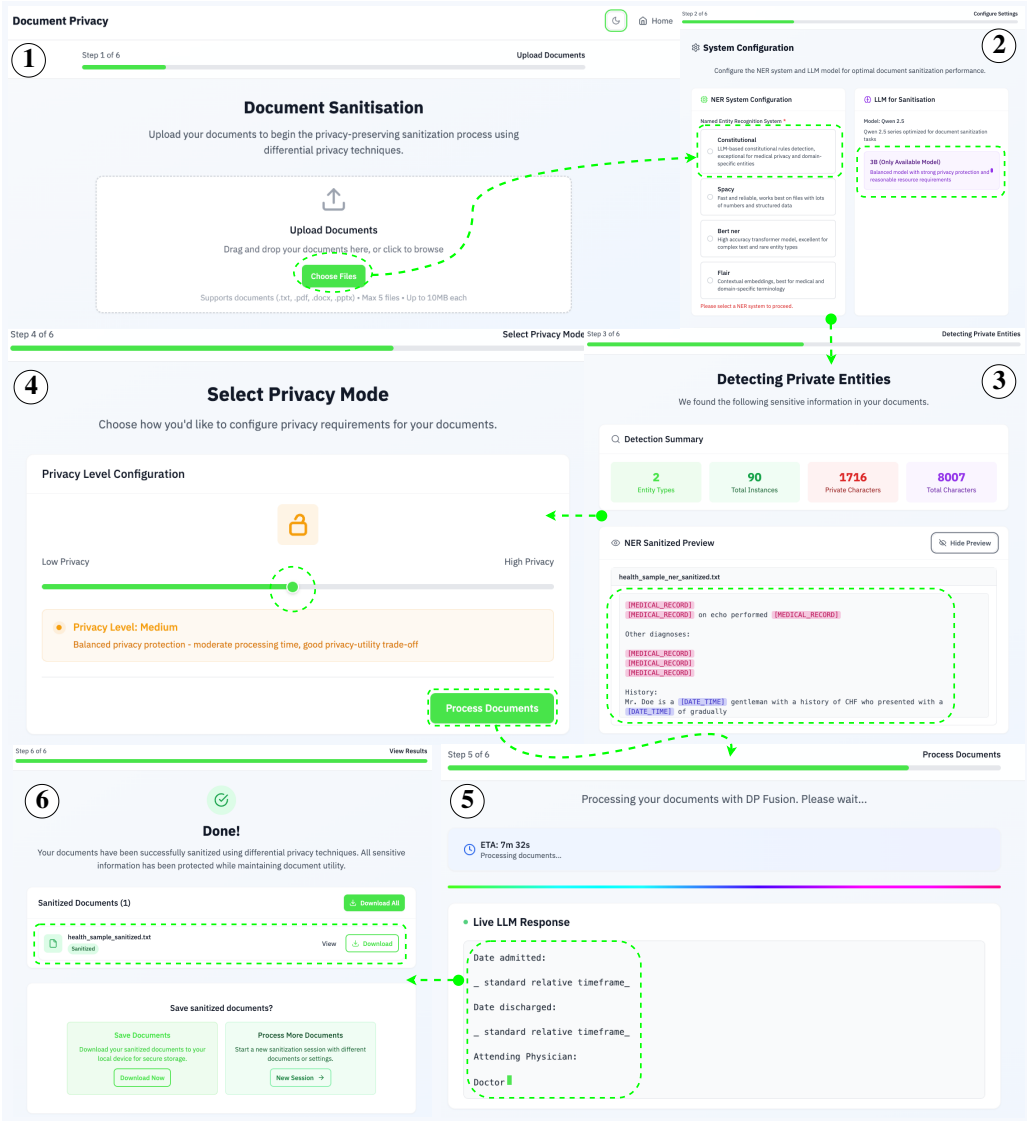


Figure 4: End-to-end demonstration of the deployed pipeline: (1) document upload, (2) tagger and LLM selection, (3) private entity overview, (4) privacy level choice, (5) iterative DP-paraphrase generation, and (6) Downloading the resultant privatised document.