ADVERSARIAL ACCURACY MISSES WHAT LINEAR PROBES STILL INFER FROM ANONYMIZED TEXT

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models infer personal attributes from online comments with near-human accuracy, and text anonymizers rewrite the comments to prevent such inference. Anonymizers are evaluated by adversarial accuracy, the share of attributes that an LLM attacker still infers correctly. Most of them also use such an attacker while rewriting, so a lower accuracy may show that the attacker's answer has changed rather than that the attribute has left the text. We therefore examine the text with an adversary outside the rewriting loop, a linear probe trained on other authors' labeled raw text. On SynthPAI, six published anonymizers remove up to all of the attacker's advantage over a majority-class guess, but at most 15% of the probe's advantage over chance. The gap remains for retrained probes and a fine-tuned classifier, and a frozen probe finds it on real Reddit comments. Controlled edits on SynthPAI suggest why: much of the remaining signal lies in register, how a comment is written rather than what it states. To narrow the gap, Probe-Guided Register Selection (PGRS) rewrites each comment in several registers and publishes the rewrite the defender's own probe is least certain about. PGRS lowers adversarial accuracy more than most published anonymizers, at a moderate utility cost. PGRS lowers frozen-probe AUC from 0.76 to 0.51 on three probe models, two of which take no part in selection. Because it selects against a frozen probe, we also report a probe retrained on its output, which reaches 0.56–0.60 against 0.73–0.77 for the published anonymizers. We therefore recommend reporting probe AUC, frozen and retrained, alongside adversarial accuracy.

1 INTRODUCTION

A person who comments on an online platform leaves cues to their age, income and location in ordinary sentences. A large language model given those comments can infer such attributes with high accuracy, at far lower cost than human labelers (Staab et al., 2024). With public records, a few attributes such as location, gender and date of birth uniquely identify about half of the US population (Sweeney, 2000). Attributes inferred from digital traces have been used to tailor advertising to millions of individuals (Matz et al., 2017). Removing names does not prevent such inference, because few of the cues behind it are explicit statements that a person could simply leave out (Staab et al., 2024).

Text anonymizers defend against this inference by rewriting a person's comments before they are published. They are commonly evaluated by adversarial accuracy, the share of attributes that an LLM attacker still infers correctly from the rewritten text (Staab et al., 2025). Apart from Presidio (Microsoft, 2021), which masks named entities, most anonymizers also run this attacker while they rewrite. FgAA (Staab et al., 2025) feeds the attacker's inference back into a rewriter, and RLAA (Duan et al., 2026) and TRACE (Yan et al., 2026) rewrite only the spans that an arbitrator or an attention map flags. IncogniText (Frikha et al., 2025) rewrites until its inner attacker is misled, and Paudel et al. (2026) anonymize and attack their own output in one generation. These anonymizers are in effect optimized against the same kind of adversary that later scores them. A lower score may therefore reflect a misled attacker rather than a removed attribute, as the confident wrong answers observed by Paudel et al. (2026) suggest. This raises the following question.

When an anonymizer lowers adversarial accuracy, has it removed the attribute from the text, or only changed the attacker's answer?

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

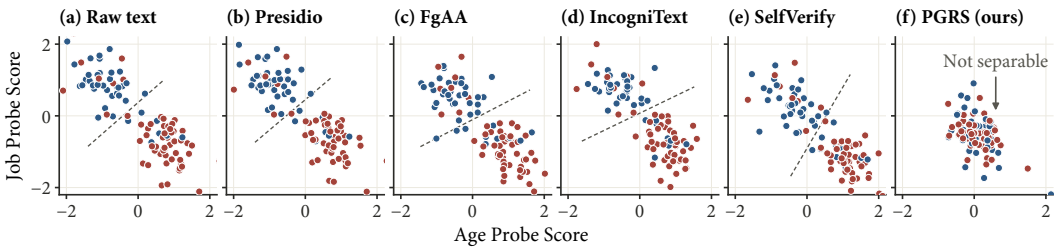


Figure 1: Frozen-probe age and job scores of the same 100 SynthPAI authors, colored by age group (blue under 35, red above 35). The dashed logistic-regression boundary separates the two groups on the raw text (a) and after four published anonymizers (b)–(e), but not after PGRS (f).

Representation learning has met this question before. Elazar & Goldberg (2018) found that when adversarial training drove a demographic attribute to chance for the adversary inside the training loop, a linear probe trained afterwards still recovered it, and such probes are the standard test of what a representation encodes (Alain & Bengio, 2016; Belinkov, 2022). We apply the same test to text anonymization. Our probe adversary fits a linear probe on a small open model's hidden states using other authors' labeled raw text, and applies the frozen probe to the anonymized text. The probe takes no part in any rewriting, and its AUC measures how well it still orders the authors by an attribute. The two adversaries are complementary: the LLM attacker has a large model but no labeled data, and the probe adversary has labeled data but only a small model. On SynthPAI, five published anonymizers remove 0–68% of the attacker's advantage over a majority-class guess and IncogniText removes all of it, yet all six remove only 0–15% of the probe's advantage over chance (Table 1, Figure 1). The gap remains when the probe is retrained on each anonymizer's output or replaced by a fine-tuned RoBERTa classifier, and the frozen probe also finds it on real Reddit comments.

Controlled edits on SynthPAI suggest why the attribute survives. Much of what the probe reads lies not in what a comment states but in how it is written, its register of function words, punctuation, capitalization and formality (Biber, 1988). Rewriting comments into a formal register while keeping their facts lowers the probe AUC from 0.76 to 0.60, the largest drop among our edits, whereas an edit aimed at generalizing identifying details moves it by less than 0.01. This is consistent with how the anonymizers use the attacker, whose guess rests on what a comment states rather than on how it is written.

If much of the attribute lies in register, a defense should change register and be judged by adversaries it was not tuned against. Probe-Guided Register Selection (PGRS) is such a defense: it abstracts each comment, rewrites it in several registers with a local model, and publishes the rewrite on which the defender's own frozen probe is least certain, using no author label. PGRS lowers frozen-probe AUC to about 0.51 on three probe models, but because it selects against a frozen probe, this number alone would overstate its protection, just as adversarial accuracy overstates that of the published anonymizers. We also evaluate it against adversaries who retrain the probe on its output. Their AUC is 0.56–0.60 across three probe models, compared with 0.73–0.77 for the published anonymizers, and PGRS lowers adversarial accuracy more than most of them, at a moderate cost in judged utility. We therefore recommend reporting probe AUC, frozen and retrained, alongside adversarial accuracy when evaluating text anonymizers.

Our contributions are the following.

- **A blind spot in evaluation.** Published anonymizers change the LLM attacker's answer far more than the attribute information in the text, on SynthPAI and on real Reddit comments.
- **Where the attribute remains.** On SynthPAI, single-property edits locate much of the remaining signal in register rather than in identifying details.
- **A correction.** We recommend reporting probe AUC, frozen and retrained, alongside adversarial accuracy, and show with PGRS that the gap can be narrowed under both.

2 THREAT MODEL

An anonymizer rewrites a person's comments before they are published, and only the rewritten comments are public. An adversary who reads the published comments of many authors wants their age, education, income level, relationship status and sex. We consider three adversaries, which differ in their model, their labeled data and their knowledge of the anonymizer, and one defender.

The LLM attacker gives an author's comments to a large language model and takes its guess, with no labeled data (Staab et al., 2024). Every anonymizer we compare is evaluated against it.

The probe adversary runs a small open model and holds labeled raw text of other people, which is publicly available, for example from Reddit accounts that state their age and sex (Tigunova et al., 2020), and has long served author profiling (Schler et al., 2006; Argamon et al., 2009; Rangel et al., 2013). It fits a linear probe on the model's hidden states and applies it, frozen, to the published text.

The adaptive adversary also knows the anonymizer, as prior work allows (Mattern et al., 2022; Utpala et al., 2023), and retrains the probe on the anonymizer's output for its own labeled authors.

The defender runs only local open models, since a remote model costs a request per comment and sends the text out (Kim et al., 2025; Fisher et al., 2024), and can fit the same probe from the same public sources before publishing. In our experiments, the adversaries and the defender fit their probes on the same development authors from the evaluated corpus, which is more favorable to both than labeled text from other sources.

The two probe adversaries score every author for every class, and a ranking by these scores is enough to cause harm: its top selects a group by attribute for targeting or differential treatment without any single guess being right, and CCPA counts such inferences as personal information (State of California, 2018).

3 ADVERSARIAL ACCURACY AND PROBE AUC

We write $\mathcal{A}$ for the set of attributes, $\mathcal{Y}_a$ for the classes of attribute a, and $y_a(u)$ for the true class of author u. Author u has a set of comments T_u, and an anonymizer f maps it to $f(T_u)$. Both measures below are computed on a pool U of authors that no probe was trained on.

Adversarial accuracy. Adversarial accuracy is the share of authors whose attribute the attacker guesses correctly. The attacker reads all comments of one author and returns one guess per attribute, following the prompt and scoring of Staab et al. (2025). With $g_a(\cdot)$ for this guess and $\mathbf{1}[\cdot]$ for the indicator function,

$$\mathrm{Acc}_a(f) = \frac{1}{|U|} \sum_{u \in U} \mathbf{1}\big[\, g_a(f(T_u)) = y_a(u) \,\big]. \tag{1}$$

Frozen-probe AUC. The probe adversary's classifier is the frozen probe. Let $h(t) \in \mathbb{R}^d$ be the hidden state of comment t at one layer of the probe adversary's language model, pooled over tokens (Appendix B.3). For each attribute a, the probe is a multinomial logistic regression on $h(t)$, fit on the raw text of a disjoint set of development authors, and $p_a(y \mid t)$ is its posterior for class $y \in \mathcal{Y}_a$. We never refit it afterwards, and this is what frozen means throughout.

The score of an author for class y is the mean posterior over the author's anonymized comments,

$$s_a(u, y; f) = \frac{1}{|T_u|} \sum_{t \in f(T_u)} p_a(y \mid t). \tag{2}$$

For one class y, the area under the ROC curve of these scores against the labels $\mathbf{1}[y_a(u) = y]$, written $\mathrm{AUC}(\cdot, \cdot)$, measures how well the probe separates the authors of class y from the rest. The probe AUC averages this area over the classes of a,

$$\mathrm{AUC}_a(f) = \frac{1}{|\mathcal{Y}_a|} \sum_{y \in \mathcal{Y}_a} \mathrm{AUC}\Big(\big\{ s_a(u, y; f) \big\}_{u \in U},\ \big\{ \mathbf{1}[y_a(u) = y] \big\}_{u \in U} \Big). \tag{3}$$

A probe AUC of 0.5 means the probe no longer orders the authors by the attribute.

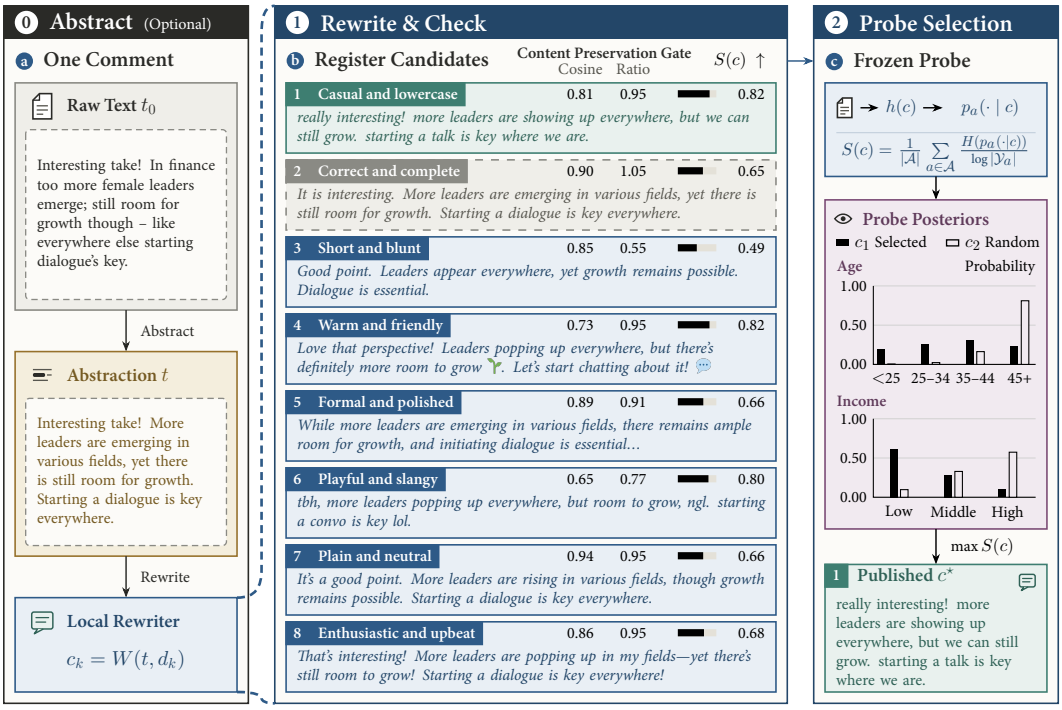


Figure 2: Overview of our Probe-Guided Register Selection (PGRS) on one comment of an author. (0) Abstract the comment. (1) Generate eight register candidates and check cosine similarity and length ratio (Equation 5). (2) Publish the candidate with the highest mean posterior entropy (Equation 6). The posterior plots compare the selector's posteriors for the selected and random candidates.

Retrained-probe AUC. The retrained probe follows the same construction on development text that the anonymizer under evaluation has rewritten, with the same authors, training recipe and cross-validation. The text it learns from is the only difference from the frozen probe, and the retrained-probe AUC is Equation 3 with this classifier.

4 Probe-Guided Register Selection

PGRS lowers the probe AUC of an anonymized text by choosing how the text is written. A local rewriter produces $K = 8$ rewrites of each comment under different register directives, and the defender's own frozen probe publishes the one on which it is least certain (Figure 2). Each component follows from the defender of Section 2. The rewriter and the probe run locally, the probe is fit on the defender's own labeled raw text, and no step uses the label of the author being anonymized. Register is the direction of change because it carries much of what the probe still finds (Section 5.3).

Candidate Generation under Register Directives. The input is one author's comments after an optional rewriting stage. In our main configuration this stage is abstraction, a single prompt that generalizes every identifying detail (Appendix G.1). A local rewriter W receives each input comment t with K register directives $d_1, \ldots, d_K$, and produces the candidate c_k under directive d_k,

$$c_k = W(t, d_k), \qquad k = 1, \ldots, K. \tag{4}$$

Each directive fixes a register, such as casual and lowercase, formal and polished, or short and blunt (Appendix G.2). The prompt shared by all directives asks the rewriter to keep every claim and to generalize identifying details. The candidates therefore differ in register and in the detail they keep.

Content Preservation Gate. The gate keeps each published candidate close to its input comment t, and Section 5.5 measures what a candidate changes beyond its register. Let $e(\cdot)$ be a sentence embedding (Reimers & Gurevych, 2019) and $|\cdot|$ the length of a text in words. A candidate passes

the gate when the cosine between $e(c_k)$ and $e(t)$ is at least a threshold τ and its length ratio $|c_k|/|t|$ lies in a band $[\ell_-, \ell_+]$. The set $\mathcal{C}(t)$ of passing candidates, with $\tau = 0.55$ and $[\ell_-, \ell_+] = [0.5, 1.6]$, is

$$\mathcal{C}(t) = \big\{\, c_k : \cos\big(e(c_k), e(t)\big) \geq \tau,\ \ \ell_- \leq |c_k|/|t| \leq \ell_+ \,\big\}. \tag{5}$$

Maximum Entropy Probe Selection. The frozen probe p_a of Section 3 scores every passing candidate. Let $H(\cdot)$ denote the entropy of a distribution. The score $S(c)$ of a candidate c is the mean over attributes of its posterior entropy, where dividing by $\log|\mathcal{Y}_a|$ puts each attribute on the same scale from 0 to 1. The published candidate $c^\star$ is the one with the highest score,

$$S(c) = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \frac{H\big(p_a(\cdot \mid c)\big)}{\log|\mathcal{Y}_a|}, \qquad c^\star = \arg\max_{c \in \mathcal{C}(t)} S(c). \tag{6}$$

The selecting probe is the defender's own, fit on Qwen3.5-4B-Base hidden states with the defender's development authors and kept frozen. The evaluation probes on OLMo and Mistral therefore take no part in selection. The rule uses no author label and aims at a uniform posterior. Its objective therefore favors no particular wrong class. A decoy method such as IncogniText instead fixes one wrong value and moves every cue toward it. The rule needs a probe that detects the attribute on single comments, because a probe at chance picks at random. On raw text, the frozen probe reaches an AUC of 0.58 to 0.73 on single comments for the five attributes (Appendix D.1). For each comment, the step costs K generations of the local rewriter and K probe evaluations, with no remote call (Appendix E.7).

PGRS-U: Union Candidates with a Closeness Weight. PGRS-U widens the candidate set to rewrites of both the raw comment t_0 and its abstraction t. It contains the K rewrites of t_0, the K rewrites of t, and t itself. The gate of Equation 5 is applied against t_0 for all $2K + 1$ candidates. Each passing candidate is scored by

$$S_\lambda(c) = S(c) - \lambda\big(1 - \cos(e(c), e(t_0))\big). \tag{7}$$

The weight $\lambda \geq 0$ trades privacy for closeness to the raw comment. Figure 11 in Appendix E.6 places PGRS-U, PGRS and the baselines on the plane of frozen-probe AUC and utility. We report PGRS-U at $\lambda = 0$, where the probe chooses freely between changing register and changing content.

5 Experiments

5.1 Setup

Datasets. We evaluate on SynthPAI (Yukhymenko et al., 2024), the synthetic Reddit-style corpus on which three of our baselines, FgAA, SelfVerify and TRACE, report results (Staab et al., 2025; Paudel et al., 2026; Yan et al., 2026). It contains 300 author profiles, about 7,800 comments and labels for eight personal attributes. After removing short and non-English comments, 297 authors remain (Appendix B.1). We evaluate the five attributes that every method in our comparison can be scored on, namely age, education, income level, relationship status and sex. Probes are trained on a development split of 197 authors, and every measure is computed on the remaining 100 authors. No evaluated author therefore contributes to probe training. Appendix F repeats the evaluation on a corpus of real Reddit comments with age and sex labels.

Baselines. We compare seven anonymizers, and every rewriting step among them runs on the same locally served Qwen3.5-9B, so that the rewriter is fixed and only the loop varies. Presidio (Microsoft, 2021) masks named entities. FgAA feeds the attacker's inference back into a rewriter, and we run it for three rounds. IncogniText rewrites toward a wrong attribute value. SelfVerify, our name for the unnamed method of Paudel et al. (2026), anonymizes and self-attacks in one generation. TRACE rewrites the words its attention map selects. RLAA runs for at most ten rounds and rewrites only the spans its arbitrator rates as a high or medium risk. Abstraction (Section 4) is the seventh baseline.

Metrics. The attacker of Equation 1 is a locally served Qwen3.8-27B, and DeepSeek-V4.1-Flash through an API serves as a second attacker. Adversarial accuracy averages the five attributes. The probe of Equation 3 is fit on the hidden states of three models, Qwen3.5-4B-Base, OLMo-2 7B and Mistral-7B v0.3. Intervals and paired comparisons use an author bootstrap over the 100 pool authors

Table 1: Adversarial accuracy, probe AUC and utility for every method on SynthPAI. Published anonymizers lower adversarial accuracy and leave the frozen probe near its raw-text AUC. PGRS lowers both. Adversarial accuracy averages five attributes against a majority-class prior of 0.488.

	Adversarial Accuracy ↓		Frozen-Probe AUC ↓			Retrained-Probe AUC ↓			
Method	Qwen 27B	DeepSeek	Qwen 4B	OLMo 7B	Mistral 7B	Qwen 4B	OLMo 7B	Mistral 7B	Utility ↑
Raw text	0.746 ±.06	0.728 ±.06	0.755 ±.03	0.767 ±.03	0.760 ±.04	–	–	–	1.000
Presidio	0.714 ±.06	0.739 ±.06	0.754 ±.03	0.766 ±.03	0.764 ±.04	0.757 ±.03	0.767 ±.04	0.765 ±.04	0.845
FgAA	0.604 ±.07	0.644 ±.06	0.740 ±.04	0.739 ±.04	0.739 ±.04	0.730 ±.03	0.736 ±.04	0.731 ±.04	0.719
IncogniText	**0.178** ±.06	**0.226** ±.06	0.722 ±.03	0.735 ±.04	0.721 ±.04	0.729 ±.03	0.729 ±.03	0.726 ±.04	0.372
SelfVerify	0.637 ±.07	0.633 ±.07	0.749 ±.03	0.746 ±.04	0.739 ±.04	0.748 ±.03	0.753 ±.03	0.743 ±.03	0.881
TRACE	0.690 ±.06	0.671 ±.07	0.735 ±.03	0.734 ±.04	0.737 ±.04	0.736 ±.03	0.754 ±.04	0.751 ±.04	0.790
RLAA	0.570 ±.07	0.602 ±.06	0.745 ±.03	0.758 ±.04	0.753 ±.04	0.741 ±.04	0.754 ±.04	0.754 ±.03	**0.955**
PGRS-U (ours)	0.652 ±.07	0.689 ±.07	0.535 ±.05	0.575 ±.05	0.505 ±.05	0.657 ±.04	0.670 ±.04	0.677 ±.04	0.880
PGRS (ours)	0.579 ±.06	0.560 ±.06	**0.511** ±.04	**0.514** ±.05	**0.495** ±.04	**0.599** ±.04	**0.584** ±.04	**0.564** ±.04	0.758

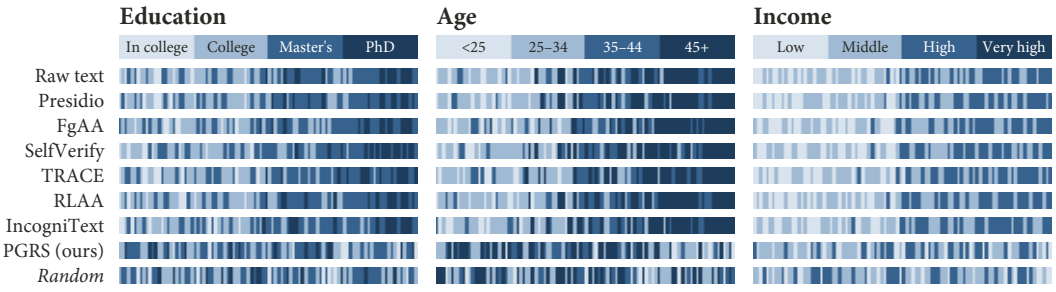


Figure 3: Authors of each output on SynthPAI, sorted by the frozen probe's expected class and coloured by their true class. The last row shows the same authors in random order. The published baselines keep the light-to-dark gradient and PGRS removes it.

(Appendix E.1). The retrained probe of Section 3 is fit on each method's rewrite of the development comments and evaluated on the same 100 authors. Utility is the three-item judge of Staab et al. (2025), which scores readability, meaning preservation and absence of invented facts. The local 27B model scores each item, and their average gives the raw text 1.00 (Appendix B.3).

5.2 Adversarial accuracy falls while probe AUC stays

Table 1 shows that the published anonymizers move adversarial accuracy and probe AUC in different ways. Every published anonymizer lowers adversarial accuracy, from 0.746 on the raw text to as low as 0.178. The API attacker agrees on the five rewriting loops and leaves Presidio at raw-text accuracy. On the same outputs, the frozen probe stays within 0.04 of its raw-text AUC of 0.76 on all three probe models. To put the two measures on one scale, we also state the share of an adversary's advantage a method removes. This share is the drop from the raw text divided by the raw-text margin, over the majority-class rate for the attacker and 0.5 for the probe. The LLM-based anonymizers remove between 22% and all of the attacker's advantage, and none removes more than 15% of the probe's advantage on any probe model. The same gap appears for a classifier fine-tuned on each anonymizer's output and for FgAA with a stronger remote rewriter (Appendices C.1 and C.2). Figure 3 shows this author by author: under Presidio, FgAA, SelfVerify, TRACE and RLAA, the authors sorted by the probe's expected class still run from light to dark by true class.

IncogniText shows most clearly why the two measures can disagree. It rewrites each comment toward a wrong attribute value, and about half of the attacker's first guesses land on this planted value, which pushes adversarial accuracy below the majority-class prior (Appendix C.3). Yet the frozen probe still ranks the same authors by their true value nearly as well as on the raw text. RLAA shows the same gap with the fewest edits: it rewrites about a third of the comments, lowers adversarial accuracy to 0.570, and leaves the frozen probe at 0.752.

Table 2: Register editing lowers frozen-probe AUC most. Lines show changes from raw text, averaged over five attributes. Paired intervals and edit audit: Tables 8 and 10.

Changed	Kept	AUC ↓ (raw 0.755)	Δ vs. Raw
Register	Stated facts	**.603**	−0.152
Capitalization & punctuation	Content words	**.718**	−0.037
Identifying details	Register	**.754**	−0.002

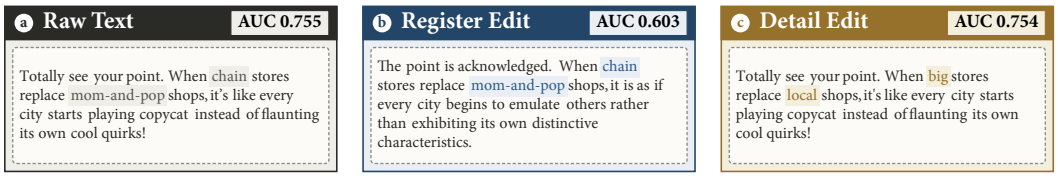


Figure 4: One raw comment under the register edit and the detail edit. Only the register edit lowers the frozen probe. Highlighted phrases are those one edit replaced and the other kept, and each AUC is the pool-level value under that edit, averaged over five attributes.

5.3 Register carries much of what the probe finds

To find out what the probe still finds, we change one property of each raw comment at a time and compute the frozen-probe AUC on the result (Table 2). Rewriting a comment into a formal register while keeping its facts lowers the probe from 0.76 to 0.60, the largest drop among the tested edits. The two control edits move it far less. Lowercasing the text and flattening its punctuation keeps every content word and moves it little, and generalizing the identifying details while keeping the register leaves it unchanged (audit in Table 10). Figure 4 shows one comment under both edits.

The raw text points to the same pattern without any rewriting. On age and income, a classifier restricted to register features scores higher than one that sees only content words (Appendix D.1). The rewriting loops leave these two attributes in place. On every loop's output, the frozen-probe AUC for age and income stays about as high as on the raw text (Appendix D.2). Since each loop edits a comment in response to an attacker's guess, this result is consistent with rewrites that change what a comment states and leave how it is written.

Rewriting more of each comment lowers the probe further but still leaves it well above chance. Abstraction, which generalizes every identifying detail, brings the probe to 0.63 (Table 3). Even after abstraction, register remains the property that moves the probe. On the abstracted text, the register edit still lowers the probe, and the other two edits leave it unchanged (Appendix D.3). This remaining effect of register is what PGRS exploits (Section 4).

5.4 PGRS brings the frozen probe close to chance

PGRS brings the frozen probe close to chance, which no published anonymizer does (Table 1). The published anonymizers leave the probe near its raw-text AUC, and PGRS lowers it to about 0.51 on all three probe models. Notably, the reduction holds on OLMo and Mistral, whose probes take no part in selection. PGRS is therefore not tuned to any one probe model.

An adaptive adversary recovers part of this reduction by retraining the probe on PGRS output, which then reaches 0.56 to 0.60 across the three probe models. The same retraining leaves the published anonymizers at 0.73 to 0.77. Even against this adversary, PGRS stays below every published anonymizer on every probe model, and its margin over FgAA holds under the paired bootstrap (Appendix E.1).

The lower probe AUC comes without a loss on adversarial accuracy or utility. Compared with FgAA, PGRS brings adversarial accuracy to a similar level and keeps a slightly higher judged utility. When utility matters more, PGRS-U offers a second operating point, which raises judged utility and in exchange gives up part of the reduction on both measures. Figure 5 summarizes these results on

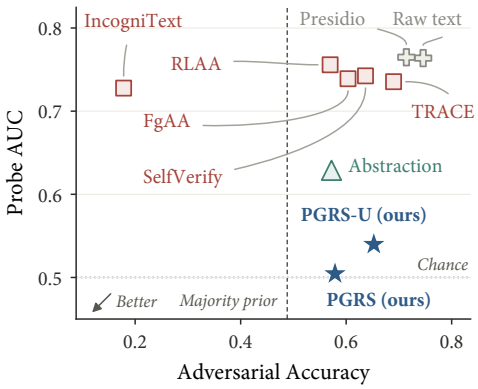


Figure 5: Every method on SynthPAI by adversarial accuracy and frozen-probe AUC.

Table 3: Ablations of PGRS on SynthPAI. Removing either abstraction or selection raises frozen-probe AUC, and removing abstraction also raises adversarial accuracy. Frozen and retrained AUC average three probe models. Fixed register applies the plain and neutral directive to every comment.

Method	Adversarial Accuracy ↓	Probe AUC ↓ Frozen	Probe AUC ↓ Retrained	Utility ↑
PGRS (full)	0.579	**0.507**	**0.582**	0.758
w/o selection	**0.540**	0.575	0.621	0.743
w/o abstraction	0.717	0.549	0.689	**0.904**
Abstraction only	0.572	0.628	0.671	0.893
Candidates only	0.687	0.615	0.669	0.871
Fixed register	0.575	0.608	0.630	0.868

the plane of adversarial accuracy and probe AUC. The published anonymizers spread along the adversarial-accuracy axis at nearly the same probe AUC, and abstraction, PGRS-U and PGRS move successively further down the probe-AUC axis.

5.5 Ablations

To see what each component contributes, we remove one component of PGRS at a time, or keep only one, on the same 100 authors (Table 3). Abstraction and selection address different measures, with abstraction mainly lowering adversarial accuracy and selection mainly lowering probe AUC.

Selection. Against the frozen probe, selection accounts for most of the reduction that PGRS adds to abstraction. Of the drop from 0.63 for abstraction alone to 0.51 for PGRS, rewriting into a random passing register gives less than half, and the probe's choice among the same candidates gives the rest. The margin over random selection holds under the paired bootstrap (Appendix E.1). The chosen candidates also differ from random ones in what they change. Three judges find that they change the register and generalize details more often, with no difference in preserved claims (Table 12).

One side effect of averaging over attributes is that, against the random control, PGRS lowers age, education and income but slightly raises sex on every model. Because Equation 6 averages the entropy of five posteriors, a candidate can raise this average even while its sex posterior becomes less flat. A defender who cares about one attribute can weight the objective toward it.

Abstraction. Abstraction is the component that lowers adversarial accuracy. Without it, adversarial accuracy stays close to its raw-text value, although the probe already falls. Abstraction alone does the opposite, lowering adversarial accuracy and leaving the probe well above chance. Combining the two lowers both measures, at a cost in judged utility relative to abstraction alone. Under retraining, the two components are complementary. Each alone leaves the retrained probe at 0.67 to 0.69, and together they bring it to 0.58 on average over the three probe models.

Candidates and fixed registers. Choosing the register comment by comment lowers the probe more than fixing one register for all comments. On the raw comments, the probe's choice among register candidates lowers the frozen probe below a random candidate from the same set. On the abstracted text, the plain and neutral directive applied to every comment lowers the probe less than a random candidate does. Even the best fixed directive, formal and polished, stays above PGRS on all three probe models (Appendix E.3).

Other anonymizers. Because its candidates are rewrites of whatever text it receives, PGRS can follow any existing anonymizer. Placed after Presidio, FgAA, IncogniText, SelfVerify or RLAA, it lowers each of their probe AUCs to between 0.52 and 0.56 (Table 14).

6 Related Work

Text anonymization. Staab et al. (2024) showed that pretrained LLMs infer personal attributes from free text and that entity masking does not stop them. Current anonymizers therefore rewrite the text with an LLM, and the field evaluates them with an LLM attacker. Feedback-guided adversarial anonymization (FgAA) feeds the attacker's inference back into a rewriter (Staab et al., 2025), and its attacker prompt and utility judge became the field's evaluation. Later methods differ in how they use the attacker. IncogniText (Frikha et al., 2025) rewrites toward a wrong attribute value, SelfVerify (Paudel et al., 2026) anonymizes and self-attacks in one generation, and TRACE (Yan et al., 2026) rewrites the words its attention map selects. SEAL (Kim et al., 2025) distills the loop into a small model, and RLAA (Duan et al., 2026) and intent-aware anonymization (Shen et al., 2026) add a stage that decides which spans to touch. Paudel et al. (2026) note that a misled attacker may guess wrong with high confidence. Outside anonymization, prior work has found that removing an attribute for one reader can leave it readable to another, for demographic attributes in learned representations (Elazar & Goldberg, 2018) and for author identity in rewritten text (Mattern et al., 2022; Utpala et al., 2023; Bao & Carpuat, 2024). A 2026 survey lists representation-level demographic leakage among the underexplored threats (Goswami et al., 2026). We bring this second reader to text anonymization. A frozen probe placed beside the LLM attacker shows that the two measures diverge on every published anonymizer and on both corpora (Section 5.2).

Author profiling. Author profiling infers age and gender from function words and parts of speech (Schler et al., 2006; Argamon et al., 2009). These features load on the first dimension of register variation (Biber, 1988). Reddy & Knight (2016) found that exclamation marks and lengthened spellings likewise indicate gender. The anonymization papers name the same cues, from the slang that entity masking leaves (Staab et al., 2024) to the stylistic cue that RLAA rates as a medium risk (Duan et al., 2026). None of these papers measures what the cues contribute after anonymization. We measure it by changing the cues of a comment while keeping its content and computing the frozen-probe AUC on each edited comment (Section 5.3).

Attribute removal and authorship obfuscation. Prior defenses differ in where they use a reader of the protected attribute. Concept erasure applies the reader to the representation and removes the attribute until no linear classifier detects it (Elazar & Goldberg, 2018; Ravfogel et al., 2020; Belrose et al., 2023). It edits vectors and publishes no text. Differentially private rewriting uses no attribute signal and perturbs tokens or samples a paraphrase (Yue et al., 2021; Utpala et al., 2023). Authorship obfuscation steers a rewriter with a classifier or an embedding. A4NT (Shetty et al., 2018), TAROT (Loiseau et al., 2025), Keep It Private (Bao & Carpuat, 2024) and Xu et al. (2019) use it as a training signal. JAMDEC (Fisher et al., 2024) and StyleMC (Khan et al., 2023) use it at inference to score candidate rewrites, a choice the PAN task already proposed (Potthast et al., 2016). PGRS belongs to this last group (Section 4). To our knowledge, it is the first method in which a frozen attribute probe on the defender's own local model chooses among candidates. It needs no author label and no third-party model at any step.

7 Conclusion

Published anonymizers lower adversarial accuracy, yet a linear probe trained on other authors' raw text still ranks their authors by every attribute. On SynthPAI, much of what the probe finds lies in register, how a comment is written rather than what it states. PGRS lets the defender's own frozen probe choose among register rewrites of each comment, with no author label. PGRS lowers frozen-probe AUC to 0.51, but this number alone would overstate its protection, just as adversarial accuracy overstates that of the published anonymizers. Against probes retrained on its output, PGRS stays below every published anonymizer, and on real Reddit comments its frozen-probe AUC is below every published anonymizer's, with a smaller reduction. We therefore recommend reporting probe AUC, frozen and retrained, alongside adversarial accuracy whenever a new text anonymizer is evaluated.

ETHICS STATEMENT

We see three ethical considerations in this work: the impact on the people whose comments form the real corpus, the broader impact of a probe that measures attributes in anonymized text, and what PGRS promises to the people whose text it rewrites.

Impact on research subjects. The real corpus consists of public Reddit comments whose authors stated their own age or sex. The labels come from the identifiers released by RedDust (Tigunova et al., 2020), and the text was retrieved from a public archive. We removed the self-reporting comments before any experiment, and no result therefore depends on recovering the statement that produced a label. We do not release the corpus and have not shared the labels with anyone. The comment text, without labels, was sent to one commercial API as the second attacker, and every other model ran locally. Every example shown in the paper comes from the synthetic corpus. The work involved no human-subject study, and no human annotator read the comments.

Broader impact. A frozen probe that measures attributes in anonymized text can also profile authors. The same is true of the LLM attackers that current evaluations already run, and the probe needs labeled raw text that an adversary can already collect from public sources (Section 2). Our results show that a defender can run the same probe locally, before publication, to measure the attribute signal that adversarial accuracy misses and to reduce it. We therefore expect the measure to help the people whose text is anonymized more than it helps an adversary who can already fit the same probe from public data.

What the protection means. PGRS lowers the frozen-probe AUC to about 0.51, and a probe retrained on its output still reaches about 0.58. Its output is therefore not a guarantee of anonymity, and a platform that deploys it should tell its users that a stronger adversary can still rank them by an attribute. The same holds for every anonymizer in Table 1, whose outputs the frozen probe still ranks by every attribute.

REPRODUCIBILITY STATEMENT

The code release at `https://anonymous.4open.science/r/pgrs-release-14CC` contains the anonymizer, the register directives, the selection rule, the probe training recipe, the baseline reimplementations and the evaluation scripts. SynthPAI is public, and every result on it that uses a local model can be rerun from the release. The real corpus is not released (see the ethics statement), and Appendix F describes its construction from the public RedDust identifiers and archive. The rewriter is Qwen3.5-9B, the probe models are Qwen3.5-4B-Base, OLMo-2 7B and Mistral-7B v0.3, and the local attacker is Qwen3.8-27B, all of them open weights. The API attacker DeepSeek-V4.1-Flash and the remote rewriter of Appendix C.2 are the only closed models. Appendix B documents the data splits, seeds, generation settings and evaluation procedure, Appendix G reproduces every prompt, and every number in the paper traces to a row of a results table produced by the code. The probe training recipe is given in Appendix B.3, the paired bootstrap with its 2,000 draws in Appendix E.1, and the time and tokens per comment of every anonymizer in Appendix E.7.

AI USE STATEMENT

In this work, we used generative AI tools for research ideation and execution, for retrieval and discovery, to draft sections of the paper, and to aid and polish the writing. For research ideation and execution, AI tools took part in brainstorming, helped implement the pipeline and the evaluation scripts, and summarized results tables during analysis. The authors ran every experiment, checked every number against its results table and made every interpretation reported in the paper. For retrieval and discovery, AI tools helped find related work and check each reference against the publisher's record, and the authors reviewed each check. For drafting and polishing, AI tools produced draft text and edits under the authors' direction, and the authors verified every claim. Language models are also the object of study, as the rewriters, attackers and judges described in Sections 4 and 5. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Shlomo Argamon, Moshe Koppel, James W. Pennebaker, and Jonathan Schler. Automatically profiling the author of an anonymous text. *Communications of the ACM*, 52(2):119–123, 2009. doi: 10.1145/1461928.1461959. URL https://dl.acm.org/doi/10.1145/1461928.1461959.

Calvin Bao and Marine Carpuat. Keep it private: Unsupervised privatization of online text. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8678–8693, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.480. URL https://aclanthology.org/2024.naacl-long.480/.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, March 2022. doi: 10.1162/coli_a_00422.

Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. LEACE: Perfect linear concept erasure in closed form. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 66044–66063, 2023. doi: 10.52202/075280-2884.

Douglas Biber. *Variation across Speech and Writing*. Cambridge University Press, 1988. doi: 10.1017/CBO9780511621024. URL https://www.cambridge.org/core/product/identifier/9780511621024/type/book.

Donghang Duan, Xu Zheng, Yuefeng He, Chong Mu, Leyi Cai, and Lizong Zhang. Look twice before you leap: A rational framework for localized adversarial text anonymization. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 5556–5573, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-acl.274. URL https://aclanthology.org/2026.findings-acl.274/.

Yanai Elazar and Yoav Goldberg. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 11–21, 2018. doi: 10.18653/v1/D18-1002. URL https://aclanthology.org/D18-1002/.

Jillian Fisher, Ximing Lu, Jaehun Jung, Liwei Jiang, Zaid Harchaoui, and Yejin Choi. JAMDEC: Unsupervised authorship obfuscation using constrained decoding over small language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1552–1581, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.87. URL https://aclanthology.org/2024.naacl-long.87/.

Ahmed Frikha, Nassim Walha, Krishna Kanth Nakka, Ricardo Mendes, Xue Jiang, and Xuebing Zhou. IncogniText: Privacy-enhancing conditional text anonymization via LLM-based private attribute randomization. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 2490–2501, Mumbai, India, December 2025. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics. doi: 10.18653/v1/2025.ijcnlp-long.134. URL https://aclanthology.org/2025.ijcnlp-long.134/.

Dhiman Goswami, Jai Kruthunz Naveen Kumar, and Sanchari Das. NLP privacy risk identification in social media (NLP-PRISM): A survey. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 1519–1541, Rabat, Morocco, March 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-eacl.78. URL https://aclanthology.org/2026.findings-eacl.78/.

Aleem Khan, Andrew Wang, Sophia Hager, and Nicholas Andrews. Learning to generate text in arbitrary writing styles. *arXiv preprint arXiv:2312.17242*, 2023. URL https://arxiv.org/abs/2312.17242.

Kyuyoung Kim, Hyunjun Jeon, and Jinwoo Shin. Self-refining language model anonymizers via adversarial distillation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://neurips.cc/virtual/2025/poster/117971. arXiv:2506.01420.

Gabriel Loiseau, Damien Sileo, Damien Riquet, Maxime Meyer, and Marc Tommasi. TAROT: Task-oriented authorship obfuscation using policy optimization methods. In *Proceedings of the Sixth Workshop on Privacy in Natural Language Processing (PrivateNLP)*, pp. 14–31, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.privatenlp-main.2.

Justus Mattern, Benjamin Weggenmann, and Florian Kerschbaum. The limits of word level differential privacy. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pp. 867–881, 2022. doi: 10.18653/v1/2022.findings-naacl.65. URL https://aclanthology.org/2022.findings-naacl.65/.

Sandra C. Matz, Michal Kosinski, Gideon Nave, and David J. Stillwell. Psychological targeting as an effective approach to digital mass persuasion. *Proceedings of the National Academy of Sciences*, 114(48):12714–12719, 2017. doi: 10.1073/pnas.1710966114. URL https://www.pnas.org/doi/10.1073/pnas.1710966114.

Microsoft. Presidio: Data protection and de-identification sdk. https://github.com/microsoft/presidio, 2021.

Bipin Paudel, Bishwas Mandal, George Amariucai, and Shuangqing Wei. Sanitization or deception? rethinking privacy protection in large language models. *Proceedings on Privacy Enhancing Technologies*, 2026(1):154–174, 2026. doi: 10.56553/popets-2026-0009. URL https://crysp.petsymposium.org/popets/2026/popets-2026-0009.php.

Martin Potthast, Matthias Hagen, and Benno Stein. Author obfuscation: Attacking the state of the art in authorship verification. In Krisztian Balog, Linda Cappellato, Nicola Ferro, and Craig Macdonald (eds.), *Working Notes of CLEF 2016 - Conference and Labs of the Evaluation Forum*, volume 1609 of *CEUR Workshop Proceedings*, pp. 716–749, Évora, Portugal, 2016. CEUR-WS.org. URL https://ceur-ws.org/Vol-1609/16090716.pdf.

Francisco Rangel, Paolo Rosso, Moshe Koppel, Efstathios Stamatatos, and Giacomo Inches. Overview of the author profiling task at pan 2013. In Pamela Forner, Roberto Navigli, and Dan Tufis (eds.), *CLEF 2013 Evaluation Labs and Workshop – Working Notes Papers*, volume 1179 of *CEUR Workshop Proceedings*, Valencia, Spain, 2013. CEUR-WS.org. URL https://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-RangelEt2013.pdf.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 7237–7256, 2020. doi: 10.18653/v1/2020.acl-main.647. URL https://aclanthology.org/2020.acl-main.647/.

Sravana Reddy and Kevin Knight. Obfuscating gender in social media writing. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pp. 17–26, 2016. doi: 10.18653/v1/W16-5603. URL https://aclanthology.org/W16-5603/.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W. Pennebaker. Effects of age and gender on blogging. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pp. 199–205, 2006. URL https://aaai.org/papers/0039-ss06-03-039-effects-of-age-and-gender-on-blogging/.

Weihao Shen, Yaxin Xu, Shuang Li, Wei Chen, Yuqin Lan, Meng Yuan, and Fuzhen Zhuang. You only anonymize what is not intent-relevant: Suppressing non-intent privacy evidence. *arXiv preprint arXiv:2601.04265*, 2026. URL https://arxiv.org/abs/2601.04265.

Rakshith Shetty, Bernt Schiele, and Mario Fritz. A4NT: Author attribute anonymity by adversarial training of neural machine translation. In *27th USENIX Security Symposium*, pp. 1633–1650, 2018. URL https://www.usenix.org/conference/usenixsecurity18/presentation/shetty.

Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. Beyond memorization: Violating privacy via inference with large language models. In *International Conference on Learning Representations (ICLR)*, pp. 33832–33878, 2024. URL https://proceedings.iclr.cc/paper˙files/paper/2024/hash/9028b8a3ca98f58e373f0c1497a17448-Abstract-Conference.html.

Robin Staab, Mark Vero, Mislav Balunović, and Martin Vechev. Language models are advanced anonymizers. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper˙files/paper/2025/hash/f478b2e8ad9ff0756bf5b79fb31c330f-Abstract-Conference.html.

State of California. California consumer privacy act of 2018, civil code section 1798.140, definition of personal information. California Civil Code, Title 1.81.5, added by Stats. 2018, Ch. 55 (AB 375), 2018. URL https://leginfo.legislature.ca.gov/faces/codes˙displaySection.xhtml?lawCode=CIV&sectionNum=1798.140.

Latanya Sweeney. Simple demographics often identify people uniquely. Data Privacy Working Paper 3, Carnegie Mellon University, Pittsburgh, PA, 2000. URL https://dataprivacylab.org/projects/identifiability/paper1.pdf.

Anna Tigunova, Paramita Mirza, Andrew Yates, and Gerhard Weikum. Reddust: a large reusable dataset of reddit user traits. In *Proceedings of the Twelfth Language Resources and Evaluation Conference (LREC)*, pp. 6118–6126, 2020. URL https://aclanthology.org/2020.lrec-1.751/.

Saiteja Utpala, Sara Hooker, and Pin-Yu Chen. Locally differentially private document generation using zero shot prompting. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 8442–8457, 2023. doi: 10.18653/v1/2023.findings-emnlp.566. URL https://aclanthology.org/2023.findings-emnlp.566/.

Qiongkai Xu, Lizhen Qu, Chenchen Xu, and Ran Cui. Privacy-aware text rewriting. In *Proceedings of the 12th International Conference on Natural Language Generation (INLG)*, pp. 247–257, Tokyo, Japan, 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-8633.

Dong Yan, Jian Liang, Ran He, and Tieniu Tan. Stop tracking me! proactive defense against attribute inference attack in LLMs. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=dCJZR0P4T5. arXiv:2602.11528.

Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. Differential privacy for text analytics via natural text sanitization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3853–3866, 2021. doi: 10.18653/v1/2021.findings-acl.337. URL https://aclanthology.org/2021.findings-acl.337/.

Hanna Yukhymenko, Robin Staab, Mark Vero, and Martin Vechev. A synthetic dataset for personal attribute inference. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, volume 37, pp. 120735–120779, 2024. doi: 10.52202/079017-3837.

A Discussion

A.1 Applications

Where the step runs. A platform that publishes user comments runs PGRS on each comment before publication. The rewriter and the probe are local open models (Section 2), so no text leaves the platform. The eight candidates run in parallel, and the median latency per comment is 0.7 seconds against 6.4 for FgAA and 11.9 for RLAA (Appendix E.7). A stronger remote rewriter lowers adversarial accuracy further and leaves both probes near their raw-text AUCs (Appendix C.2), so the choice of a local rewriter costs little on the probe axis.

After an existing anonymizer. Because the step rewrites whatever text it receives, a platform can keep its current anonymizer and add PGRS after it. Placed after Presidio, FgAA, IncogniText, SelfVerify or RLAA, it brings each one's probe AUC to between 0.52 and 0.56 (Table 14).

Auditing published text. The same frozen probe can score text after publication. A defender who holds labeled raw text can then check whether its own output still orders its authors by attribute, without running an external attacker or sending the text out.

A.2 Limitations

The two measures are not certificates. A probe AUC of 0.5 means that this probe no longer orders the authors by the attribute. An adversary with other labeled data, another model or a nonlinear classifier may still order them. The same holds for adversarial accuracy, which its authors describe as bounded by the strength of the adversary (Staab et al., 2025). The strongest adversaries we ran are a retrained probe (Section 5.4) and a classifier fine-tuned end to end (Appendix C.1). Both place PGRS below every published anonymizer. We therefore treat probe AUC as a lower bound on the attribute information that remains in a text and report it beside adversarial accuracy.

Judged utility. Changing how a comment is written is the step that lowers the probe, and it has a price in judged utility. PGRS scores 0.76 on the three-item judge against 0.89 for abstraction alone (Table 3), which is consistent with a judge that rewards text staying close to the original. PGRS-U is the operating point for a defender who weighs utility higher. It keeps a judged utility of 0.88 and still lowers the probe below every published anonymizer (Table 1).

Corpora and resolution. Our evidence comes from two corpora with 100 evaluation authors each, and this size sets what the measures can separate. Adversarial accuracy carries a bootstrap interval of about ± 0.06 and cannot separate adjacent methods. Probe AUC carries about ± 0.04 (Table 1). Every margin we report is therefore a paired difference on the same resampled authors (Appendix E.1). SynthPAI is a synthetic corpus, and the real corpus carries labels for age and sex only. The five-attribute results rest on synthetic text, and the real text checks two of the attributes. We know of no public corpus of real text with author-level labels for all five attributes.

A.3 Future Work

Candidates beyond register. Abstraction lowers adversarial accuracy and selection lowers the probe (Table 3). Candidates that vary content as well as register would let the probe choose along both axes, of which PGRS-U, with its union of raw and abstracted rewrites, is a first step.

Weighting attributes. Averaging entropy over five attributes slightly raised the sex AUC relative to a random candidate (Section 5.5). A defender who protects one attribute can weight the objective toward it, and the selection rule needs no other change.

Several probe models. The reduction obtained with a single 4B probe transferred to OLMo and Mistral. Selecting against several probe models at once may transfer further.

B EXPERIMENTAL DETAILS

This section specifies the data preparation, candidate generation and evaluation used in Section 5. The complete prompts are provided in Appendix G.

B.1 DATA AND AUTHOR SPLITS

We split SynthPAI by author before training any probe. Removing comments with fewer than ten words and comments detected as non-English leaves 4,680 comments from 297 authors. The development split contains 197 authors. Validation and test contain 50 authors each. We combine the latter two splits into the 100-author evaluation pool. All comments from an author stay in the same split.

The probe and the LLM attacker use the labels appropriate to their respective tasks. Probe age labels are age groups, and the LLM attacker predicts an age or age range. For the probe, we retain classes represented in development and by at least eight authors in the evaluation pool. We restrict and renormalize the class probabilities before averaging them over an author's comments. For the LLM attacker, an attribute is scored when a usable reference label is available. We first combine correct and scored counts across the two evaluation splits for each attribute, then average the five attribute accuracies. The majority-class baseline uses these same scored authors and matching rules.

B.2 GENERATION AND SELECTION

All stages of the main anonymizer use Qwen3.5-9B. The alternative rewriter is Llama-3.1-8B-Instruct, used for both abstraction and candidate generation. Neither rewriter receives the evaluation author's labels. The selector is a frozen probe on Qwen3.5-4B-Base. The OLMo and Mistral evaluation probes do not participate in selection. The reported Qwen abstraction run uses temperature 0.7, rewrite seed 17 and a 512-token output budget. The official per-comment FgAA baseline uses temperature 0.1, base seed 42 and a 768-token rewrite budget, with the inference reasons passed to the next rewrite.

Each comment produces eight candidates, one under each directive in Appendix G.2. Candidate generation uses temperature 0.8, an output budget of 400 tokens, and disabled thinking. The prompt's word limit is the input word count plus five. Sentence similarity is computed with all-MiniLM-L6-v2. The content gate uses cosine at least 0.55 and a word-count ratio in $[0.5, 1.6]$, as in Equation 5. These generation and gate values were fixed before the main run and were not tuned on the evaluation authors.

The random-selection control samples from the same passing candidates as PGRS. If no generated candidate passes the gate, both rules use the valid candidate with the highest cosine. If generation produces no valid candidate, both retain the input. For PGRS-U, cosine and length are measured against the raw comment for all 17 candidates: eight rewrites of the raw comment, eight rewrites of its abstraction, and the abstraction itself.

The author split uses seed 17. The reported candidate pool uses rewrite seed 17, with a deterministic seed derived from the comment identifier and an offset for each directive. Local 27B evaluation uses API seed 42 with thinking disabled.

B.3 PROBE AND UTILITY EVALUATION

The frozen probes standardize representations and fit a regularized logistic regression on development authors. Layer, readout and regularization are selected using cross-validation that keeps each author's comments together. The selected classifier is then fixed for all evaluated texts. The retrained-probe comparison follows the same separation: each method rewrites the development comments, a new classifier is fitted there, and the classifier is evaluated on the unchanged pool of 100 authors.

Probe AUC is computed from author-level mean posteriors, as in Equation 3. It measures how well the authors are ordered by the attribute. Comparisons use a paired bootstrap, with both methods scored on the same resampled authors. Appendix E.1 gives the resulting comparisons by dataset.

The utility judge compares an author's original and anonymized comments. It scores readability and meaning preservation from 1 to 10, and the absence of new information as a binary value. We divide the first two scores by ten, then average the three components. A higher score means greater utility.

C Additional results on adversarial accuracy and probe AUC

C.1 A fine-tuned adversary

Section 5.2 scores the published anonymizers with a probe fit on raw text. A classifier fine-tuned on each anonymizer's output likewise scores that output about as well as the raw text. For each method, we fine-tune roberta-base end to end on that method's rewrite of the 197 development authors, labeling each comment with its author's class. A held-out share of these authors selects the training epoch, and the classifier's posteriors are averaged over each pool author's comments. On the raw text, this adversary reaches an AUC of 0.753, close to the frozen probe's 0.755. Table 4 reports three training seeds. The six published anonymizers lie between 0.725 and 0.747, and PGRS gives the lowest AUC, 0.599. The gap between the two measures thus persists under an adversary that is neither frozen nor linear.

Table 4: A fine-tuned adversary on SynthPAI. For each method, roberta-base is fine-tuned on that method's rewrite of the development comments and read on the same 100 authors; each cell is the author-level AUC averaged over five attributes. The published anonymizers stay close to the raw text, and PGRS gives the lowest AUC under every seed.

	Training Seed			
Method	17	23	42	Mean ↓
Raw text	0.746	0.766	0.746	0.753
Presidio	0.759	0.753	0.730	0.747
FgAA	0.725	0.734	0.715	0.725
IncogniText	0.733	0.747	0.701	0.727
SelfVerify	0.738	0.725	0.748	0.737
TRACE	0.733	0.737	0.729	0.733
RLAA	0.732	0.737	0.736	0.735
Abstraction only	0.630	0.649	0.627	<u>0.635</u>
PGRS (ours)	0.615	0.598	0.583	**0.599**

C.2 A stronger rewriter for FgAA

Every baseline in Section 5.1 runs on the local Qwen3.5-9B. A stronger rewriter lowers adversarial accuracy further but leaves both probes near their raw-text AUCs. We run the official per-comment FgAA for three rounds with DeepSeek-V4-Pro in place of the local Qwen3.5-9B. The remote model serves as both the rewriter and the in-loop inference model, through the DeepSeek and Alibaba Cloud APIs. Table 5 scores both outputs with the measures of Table 1. The stronger rewriter lowers adversarial accuracy from 0.604 to 0.534 and raises judged utility from 0.72 to 0.82. The frozen probe moves only from 0.739 to 0.728 on average over the three probe models, and the retrained probe from 0.732 to 0.707. A remote rewriter is outside the threat model of Section 2. This run therefore shows how rewriter strength affects the two measures, and we do not count it as a baseline for PGRS.

Table 5: The official per-comment FgAA with a stronger writer on SynthPAI, three rounds. Replacing the local Qwen3.5-9B writer with DeepSeek-V4-Pro lowers adversarial accuracy and raises utility, while the frozen probe stays within 0.04 of its raw-text AUC and the retrained probe within 0.07. The remote writer is outside the local-defender threat model and serves only as a reference for writer strength.

	Adversarial	Frozen-Probe AUC ↓			Retrained-Probe AUC ↓			
Method	Accuracy ↓	Qwen 4B	OLMo 7B	Mistral 7B	Qwen 4B	OLMo 7B	Mistral 7B	Utility ↑
Raw text	0.746	0.755	0.767	0.760	–	–	–	1.00
FgAA, Qwen3.5-9B	0.604	0.740	0.739	0.739	0.730	0.736	0.731	0.72
FgAA, DeepSeek-V4-Pro	0.534	0.729	0.727	0.729	0.721	0.696	0.705	0.82

C.3 Attribute inference and planted values

In Section 5.2, IncogniText's adversarial accuracy falls below the majority-class prior because the attacker's guesses land on its planted value. Table 6 reports the attacker's top-1 accuracy for each attribute and, for IncogniText, the fraction of guesses matching its planted value. We distinguish a correct guess, a guess matching only the planted value, and a guess matching neither. The latter two categories show where incorrect guesses go, which accuracy alone does not describe. Methods that do not plant an attribute value have no planted-value category.

The planted-value rates vary by attribute. They are highest for sex and education, and age more often produces a guess matching neither value. Averaged over the five attributes, 0.52 of first guesses land on the planted value and 0.30 on neither value. Most wrong guesses are therefore redirected guesses.

Table 6: Attacker top-1 accuracy per attribute on SynthPAI and where the first guesses land under IncogniText. On education and sex most wrong guesses land on the planted value, and on age most land on neither value. Each row of the lower panel sums to 1 up to rounding.

	Top-1 Attacker Accuracy ↓				
Method	Age	Education	Income	Relationship	Sex
Raw text	0.725	0.750	0.673	0.675	0.950
Presidio	0.608	0.673	0.735	0.750	0.825
FgAA, 3 rounds	0.568	0.481	0.694	0.625	0.650
IncogniText	0.196	0.096	0.224	0.250	0.125
SelfVerify	0.686	0.519	0.653	0.550	0.775
TRACE	0.647	0.654	0.674	0.700	0.775
RLAA	0.568	0.404	0.551	0.600	0.725
Abstraction only (9B)	0.549	0.538	0.449	0.650	0.675
PGRS (ours)	0.608	0.481	0.531	0.600	0.675
Majority class	0.569	0.481	0.367	0.375	0.650

IncogniText: destination of the first guess			
Attribute	True Value	Planted Value Only	Neither
Age	0.196	0.235	0.569
Education	0.096	0.712	0.192
Income	0.224	0.429	0.347
Relationship	0.250	0.350	0.400
Sex	0.125	0.875	0.000

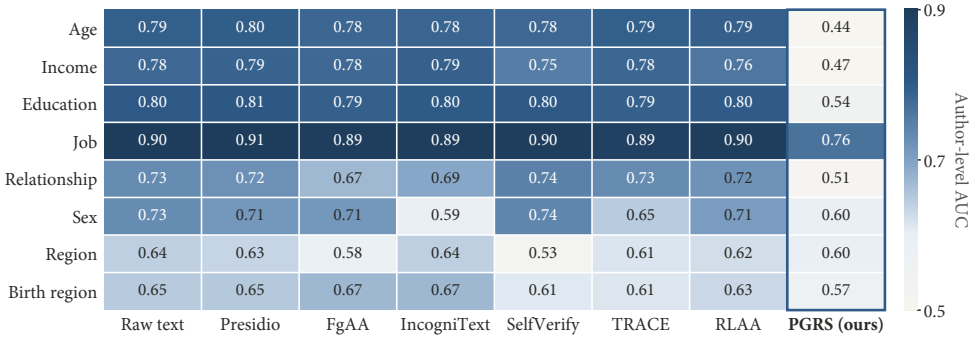


Figure 6: Frozen-probe AUC on all eight SynthPAI attributes for every method's output. The published anonymizers leave most attributes near their raw-text AUCs, and PGRS gives the lowest AUC on six of the eight. Relationship uses the three-class label of Table 7.

D Attribute information in text

D.1 Feature-restricted classifiers on raw text

Section 5.3 states that register features read age and income better than content words do. We compare classifiers restricted to different features to describe how each attribute is expressed in raw text. Following the feature-based account of register in Biber (1988), the register classifier uses function-word frequencies, punctuation, capitalization, contractions and length statistics. It receives no content words. The content classifier uses a bag of content words after removing function words and punctuation.

All classifiers are fitted on development authors and evaluated with author-level aggregation. Table 7 gives these classifiers on all eight attributes, alongside the frozen probe. Register features score above content words on age and income, and content words score above register on job and region. These raw-text comparisons describe the available cues. The interventions below test which cues affect the frozen probe itself.

Table 7: Author-level AUC of three classifiers on raw SynthPAI text. Register features read age and income better than content words, and content words read job and region better. The last column gives the frozen probe's AUC on single comments.

	Author-Level AUC			Comment AUC
Attribute	Probe	Register	Content	Probe
Age	0.79	0.77	0.71	0.73
Income	0.78	0.80	0.73	0.72
Education level	0.80	0.76	0.78	0.73
Job family	0.90	0.67	0.84	0.76
Relationship (3 classes)	0.73	0.63	0.61	0.60
Sex	0.73	0.57	0.66	0.58
Region	0.64	0.52	0.64	0.56
Birth region	0.65	0.53	0.65	0.59

D.2 Probe AUC per attribute after each anonymizer

Section 5.3 states that the rewriting loops leave age and income in place. Figure 6 gives the frozen-probe AUC of every method's output on all eight attributes. On every published anonymizer's output, age stays between 0.78 and 0.80 against 0.79 on the raw text, and income between 0.75 and 0.79 against 0.78. PGRS lowers age to 0.44 and income to 0.47.

D.3 CONTROLLED EDITS

Section 5.3 reports each edit as a mean over the five attributes. We apply three edits to the same input comments: changing register while preserving facts, normalizing capitalization and punctuation by rule, and generalizing identifying details while preserving the original register. The generation instructions are reproduced in Appendix G.10. Table 8 expands Table 2 into attribute-level AUC differences with paired 95% intervals. Register rewriting lowers the AUC of all five attributes. Detail generalization produces differences close to zero. Table 9 reports the same three edits applied to the output of abstraction. On the abstracted text, the register edit lowers the mean AUC by 0.05, with only income excluding zero, and the other two edits leave every attribute unchanged.

Figure 7 aligns the attribute-level differences for the two inputs on a common scale.

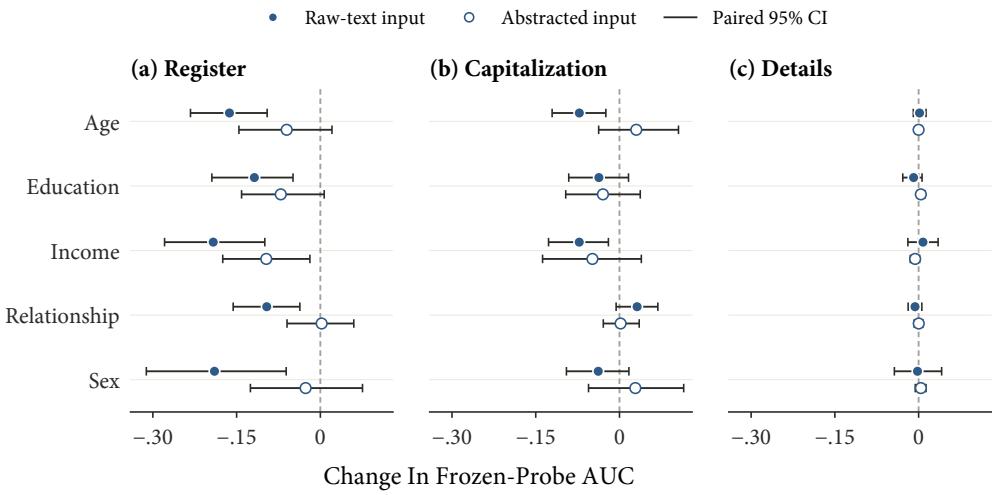


Figure 7: Attribute-level effects of three edits on SynthPAI, measured against each edit's own input. Each point is edited-text AUC minus input-text AUC for one attribute. On raw text, register rewriting gives negative intervals for all five attributes. After abstraction, only income's register interval excludes zero. The details edit uses a stronger prompt on raw text than on abstractions, so panel (c) is not a matched-prompt comparison across inputs.

Table 8: Change in frozen-probe AUC per attribute for the three edits on raw SynthPAI comments. The register edit lowers every attribute, the capitalization edit lowers age and income, and the detail edit leaves every attribute near zero. Brackets give paired 95% author-bootstrap intervals.

		Change in Frozen-Probe AUC [95% CI]		
Attribute	Raw	Register	Capitalization	Details
Age	0.793	−0.163 [-0.233, -0.095]	−0.072 [-0.121, -0.024]	+0.002 [-0.009, +0.013]
Education	0.802	−0.118 [-0.194, -0.049]	−0.037 [-0.091, +0.016]	−0.009 [-0.028, +0.006]
Income	0.781	−0.192 [-0.279, -0.099]	−0.072 [-0.127, -0.020]	+0.008 [-0.019, +0.035]
Relationship	0.671	−0.096 [-0.156, -0.037]	+0.032 [-0.006, +0.069]	−0.006 [-0.019, +0.006]
Sex	0.728	−0.190 [-0.312, -0.061]	−0.038 [-0.095, +0.017]	−0.002 [-0.043, +0.041]

Table 9: The edits of Table 8 applied to the abstracted comments. The register edit still gives the largest reductions on most attributes, and the detail edit still leaves every attribute near zero. The detail edit here uses the weaker prompt of Appendix G.10.

		Change in Frozen-Probe AUC [95% CI]		
Attribute	Abstraction	Register	Capitalization	Details
Age	0.654	−0.060 [-0.146, +0.021]	+0.030 [-0.037, +0.106]	−0.000 [-0.006, +0.006]
Education	0.690	−0.071 [-0.141, +0.007]	−0.030 [-0.096, +0.037]	+0.004 [-0.002, +0.012]
Income	0.614	−0.097 [-0.175, -0.019]	−0.048 [-0.138, +0.039]	−0.006 [-0.015, +0.002]
Relationship	0.602	+0.002 [-0.060, +0.060]	+0.002 [-0.029, +0.035]	+0.000 [-0.008, +0.008]
Sex	0.565	−0.026 [-0.125, +0.076]	+0.028 [-0.055, +0.115]	+0.004 [-0.006, +0.013]

To check whether the requested edits were carried out, we sample 60 comments per generated edit and compare each rewrite with its input. Table 10 gives the number scored and the number satisfying each property. The capitalization-and-punctuation edit is checked directly: its content words are unchanged. The judge's criteria and output fields are given in Appendix G.10.

Table 10: Audit by Qwen3.8-Max of 60 sampled comments per edit. The register edit almost always changes the register and keeps the claims, whereas the detail edit generalizes details in fewer than half of the comments.

Edit	Valid / Sampled	Claims Kept	Register Changed	Details Generalized
Register	54/60	51/54	50/54	11/54
Identifying details	60/60	42/60	0/60	25/60

D.4 Abstraction in representation space

Section 5.3 reports that abstraction lowers the five-attribute probe AUC to 0.63. Figure 8 compares the same 100 evaluation authors before and after the abstraction stage. Both panels use the age and job probe coordinates and the normalization of the raw text. Abstraction reduces the spread of the authors along both coordinates. The complete abstraction prompt is given in Appendix G.1.

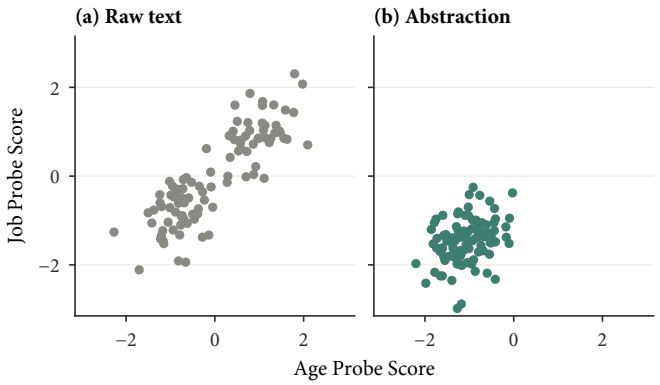


Figure 8: The same 100 SynthPAI authors at their age and job probe scores before and after abstraction. Abstraction narrows the spread of the authors and shifts them toward lower scores on both coordinates. Both panels are standardized on the raw text.

E Comparisons and ablations for PGRS

We extend the comparisons of Sections 5.4 and 5.5 with paired intervals, an audit of the selected candidates, fixed register choices, other base anonymizers, a second rewriter, the privacy–utility

plane and the cost per comment. Each comparison changes one part of the construction while retaining the same evaluation procedure.

E.1 PAIRED COMPARISONS

Every AUC carries an author bootstrap interval. To compare two methods f_1 and f_2, we use a paired bootstrap over $B = 2{,}000$ draws. In draw b, we resample the pool once into U_b and write $\mathrm{AUC}_a(f; U_b)$ for Equation 3 on U_b. We score both methods on this sample and record

$$\Delta_b = \mathrm{AUC}_a(f_1; U_b) - \mathrm{AUC}_a(f_2; U_b). \tag{8}$$

The interval on the difference is the 2.5th and 97.5th percentiles of Δ_b over the B draws. A difference excludes zero when this interval lies entirely above or below zero.

Sections 5.4 and 5.5 state that the margins of PGRS over FgAA and over random selection hold under the paired bootstrap. Table 11 lists the paired comparisons of Equation 8 for every probe model and attribute, with 2,000 draws each. For each comparison it gives the number of probe–attribute cells whose interval lies entirely below zero and the mean difference over the cells. The comparison against random selection isolates the effect of selection from that of candidate generation.

On SynthPAI, PGRS is below FgAA in 13 of 15 cells and below random selection from the same candidates in three of the 15. For the comparisons used in Section 5, we further average the difference over all cells within each draw, which gives the mean its own interval. This interval excludes zero in every case: $[-0.278, -0.189]$ against FgAA, $[-0.156, -0.087]$ against abstraction, $[-0.123, -0.016]$ against random selection, and $[-0.156, -0.042]$ for PGRS-U against random selection. Under the retrained probe, the same interval for PGRS against FgAA is $[-0.187, -0.111]$. On real Reddit comments, PGRS is below FgAA in five of six cells.

Table 11: Paired frozen-probe comparisons on SynthPAI and real Reddit. PGRS is lower than every published anonymizer in most cells on SynthPAI and in at least half of the cells on real Reddit, and its difference from random selection is smaller. Lower / total counts the (probe model, attribute) cells whose 95% interval lies below zero, and ΔAUC averages the difference over all cells.

SynthPAI

Comparator	Lower / Total	ΔAUC
PGRS minus comparator		
Raw text	14/15	−0.254
Presidio	14/15	−0.255
FgAA	13/15	−0.233
IncogniText	12/15	−0.219
SelfVerify	15/15	−0.238
TRACE	12/15	−0.229
RLAA	14/15	−0.246
Abstraction	11/15	−0.121
Random selection	3/15	−0.069
PGRS-U minus comparator		
Raw text	13/15	−0.223
Abstraction	8/15	−0.090
Random selection	9/15	−0.097

Real Reddit

Comparator	Lower / Total	ΔAUC
PGRS minus comparator		
Raw text	5/6	−0.131
Presidio	5/6	−0.128
FgAA	5/6	−0.117
IncogniText	3/6	−0.086
SelfVerify	3/6	−0.106
TRACE	5/6	−0.130
RLAA	5/6	−0.128
Abstraction	1/6	−0.059
Random selection	2/6	−0.055
PGRS-U minus comparator		
Raw text	3/6	−0.084
Presidio	2/6	−0.081
FgAA	2/6	−0.070
IncogniText	1/6	−0.039
SelfVerify	2/6	−0.060
TRACE	2/6	−0.083
RLAA	3/6	−0.081
Abstraction	1/6	−0.012
Random selection	1/6	−0.044

E.2 SELECTED CANDIDATES AND RANDOM CONTROLS

Section 5.5 states that the selected candidates differ from random ones in what they change. We audit the selected candidate and its random control against the same abstraction input. Three judges independently evaluate 300 sampled comment pairs. Table 12 reports each judge's valid-pair count, the selected-minus-control differences, and the absolute rates of preserving all claims. An invalid response excludes that pair from the corresponding judge's comparison.

All three judges find more register changes and more generalization in the selected candidates. The paired intervals for preservation of all claims contain zero for each judge.

Figure 9 compares these paired differences across the three judges, keeping each judge's valid pairs separate.

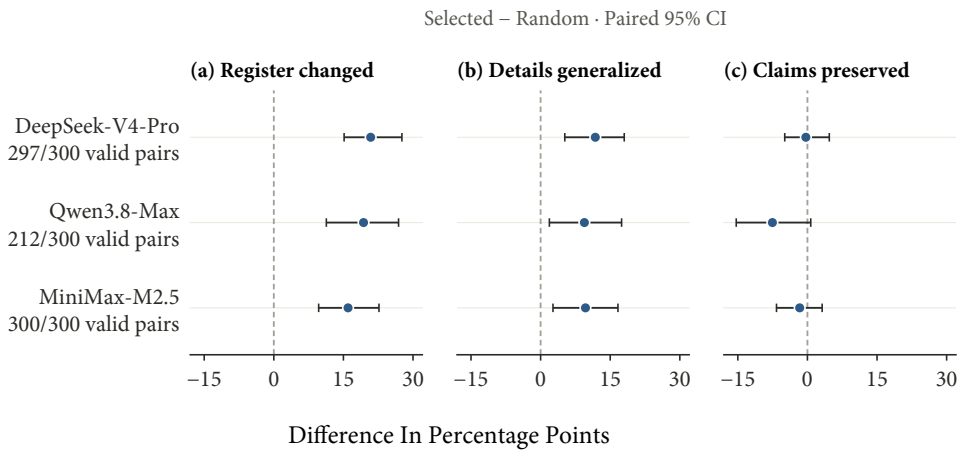


Figure 9: Paired judgments of selected and random candidates from the same abstraction. Each point is the selected-minus-random difference in the rate of one judged property, with its paired 95% bootstrap interval. All three judges detect more register changes and detail generalization among selected candidates. Every claims-preservation interval includes zero, which does not establish equivalence. Valid-pair counts differ because invalid responses are excluded separately for each judge.

Table 12: Selected and random candidates for the same abstraction input, judged by three models on 300 comment pairs. All three judges find that selected candidates change the register and generalize details more often, with no difference in preserving all claims. Differences are in percentage points with paired 95% bootstrap intervals.

Property	Selected (%)	Random (%)	Difference (pp) [95% CI]
DeepSeek-V4-Pro; 297 valid pairs out of 300			
Register changed	83.2	62.3	+20.9 [+15.2, +27.6]
Details generalized	30.0	18.2	+11.8 [+5.2, +18.0]
All claims preserved	86.5	86.9	−0.3 [-4.9, +4.7]
Qwen3.8-Max; 212 valid pairs out of 300			
Register changed	65.6	46.2	+19.3 [+11.3, +26.9]
Details generalized	56.6	47.2	+9.4 [+1.9, +17.5]
All claims preserved	52.4	59.9	−7.5 [-15.3, +0.7]
MiniMax-M2.5; 300 valid pairs out of 300			
Register changed	79.3	63.3	+16.0 [+9.7, +22.7]
Details generalized	46.0	36.3	+9.7 [+2.7, +16.7]
All claims preserved	87.7	89.3	−1.7 [-6.7, +3.2]

E.3 Fixed and per-comment register choices

Section 5.5 compares per-comment choice with the best fixed directive. Table 13 lists all eight fixed directives on the three probe models, using short names that map to the complete directives in Appendix G.2. On every model, the formal and polished directive gives the lowest fixed-register AUC, 0.562 on average against 0.507 for PGRS. Averaged over the 15 probe–attribute cells, PGRS lies 0.056 below formal and polished, with a paired 95% interval of $[-0.096, -0.016]$. A random choice from the same candidates gives 0.575. Per-comment choice lies below every fixed directive and below the random choice on all three probe models.

Figure 10 places the fixed directives and per-comment choices on the same AUC scale for each probe model.

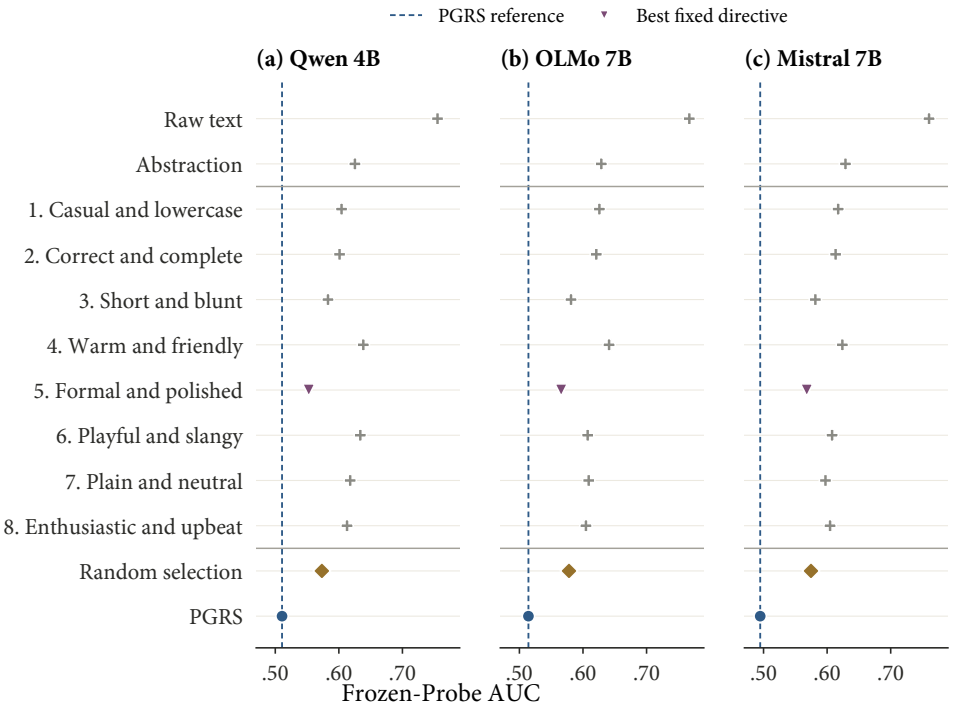


Figure 10: Fixed directives and per-comment selection evaluated on three probe models. Each point is mean author-level AUC over the five SynthPAI attributes. Directive order matches the prompts. Formal and polished is the lowest-scoring fixed directive on every model. PGRS is lower than all eight fixed directives and random selection on each model. Qwen supplies the selection probe. OLMo and Mistral do not participate in selection. These are point estimates, not interval comparisons.

Table 13: Frozen-probe AUC on the three probe models for each fixed directive and for per-comment choices on SynthPAI. On every model, formal and polished is the best fixed directive and PGRS reads lower than all of them; on the mean, a random choice per comment reads lower than every fixed directive except formal and polished. Directive numbers follow Appendix G.2.

	Frozen-Probe AUC ↓			
Published Text	Qwen 4B	OLMo 7B	Mistral 7B	Mean
Raw text	0.755	0.767	0.760	0.761
Abstraction	0.625	0.629	0.629	0.628
One directive for all comments				
1. Casual and lowercase	0.604	0.626	0.617	0.616
2. Correct and complete	0.601	0.621	0.613	0.612
3. Short and blunt	0.583	0.581	0.582	0.582
4. Warm and friendly	0.638	0.641	0.624	0.634
5. Formal and polished	0.553	0.566	0.568	0.562
6. Playful and slangy	0.634	0.607	0.608	0.616
7. Plain and neutral	0.618	0.609	0.597	0.608
8. Enthusiastic and upbeat	0.613	0.605	0.605	0.607
A choice for each comment				
Random selection	0.573	0.578	0.575	0.575
PGRS	0.511	0.514	0.495	0.507

E.4 Selection after other anonymizers

Section 5.5 states that PGRS can follow any existing anonymizer. We apply the same candidate generation and selection to the output of each base anonymizer (Table 14). The random-selection column isolates the contribution of the selector, and the paired comparison is always between selected and random outputs from the same base. Presidio has the largest reduction in mean frozen-probe AUC, from 0.761 to 0.554. The selected outputs of the six bases lie between 0.507 and 0.558. After selection, the probe AUC depends little on the base anonymizer.

Table 14: PGRS applied to the output of each base anonymizer on SynthPAI. After every base, PGRS reads well below both the base output and a random candidate from the same set. Frozen-probe AUC averages five attributes and three probe models, and Lower / total counts the 15 cells whose paired interval lies below zero.

	Adversarial Accuracy ↓		Utility ↑		Frozen-Probe AUC ↓			
Base Anonymizer	Base	+PGRS	Base	+PGRS	Base	+PGRS	Random	Lower / Total
Presidio	0.714	0.688	0.85	0.75	0.761	0.554	0.640	11/15
FgAA	0.604	0.599	0.72	0.66	0.739	0.552	0.601	6/15
IncogniText	0.178	0.216	0.37	0.38	0.726	0.518	0.570	4/15
SelfVerify	0.637	0.573	0.88	0.74	0.745	0.533	0.601	2/15
RLAA	0.570	0.549	0.96	0.86	0.752	0.558	0.640	9/15
Abstraction	0.572	0.579	0.89	0.76	0.628	0.507	0.575	3/15

E.5 A second rewriting model

The main comparison fixes the rewriter to Qwen3.5-9B (Section 5.1). We replace Qwen3.5-9B with Llama-3.1-8B-Instruct at both generation stages and retain the Qwen 4B selector (Table 15). Of the 30 paired intervals that compare the two rewriters, one detects a difference in frozen-probe AUC, and it favors the Qwen rewriter. With the Llama rewriter, selection lowers the mean probe AUC by 0.088 relative to random selection, with nine cells excluding zero. Selection lowers the probe AUC with either rewriter. The two rewriters differ in adversarial accuracy and judged utility, as reported separately in the upper panel.

Table 15: Abstraction and PGRS with Llama-3.1-8B-Instruct as the writer and the Qwen 4B selector unchanged. The new writer leaves the frozen-probe AUC of PGRS at about the same level and lowers both adversarial accuracy and utility. The lower block counts the cells whose paired interval lies below zero, and a count in parentheses is the number of cells whose interval lies above zero.

	Adversarial Accuracy ↓		Frozen-Probe AUC ↓			
Method	Qwen 27B	DeepSeek	Qwen 4B	OLMo 7B	Mistral 7B	Utility ↑
Abstraction, Qwen 9B	0.572	0.604	0.625	0.629	0.629	**0.89**
Abstraction, Llama 8B	0.514	0.507	0.616	0.598	0.624	0.87
PGRS, Qwen 9B	0.579	0.560	0.511	0.514	0.495	0.76
PGRS, Llama 8B	0.483	0.501	0.480	0.508	0.505	0.69
Random selection	0.491	0.557	0.583	0.585	0.589	0.65
Paired probe comparison					Lower / Total	Mean ΔAUC
Abstraction, Llama 8B − Abstraction, Qwen 9B					0/15	-0.015
PGRS, Llama 8B − random pick					9/15	-0.088
PGRS, Llama 8B − PGRS, Qwen 9B					0/15 (+1)	-0.009

E.6 Privacy and utility

Section 4 introduces PGRS-U as a second operating point. Figure 11 plots frozen-probe AUC against judged utility on both corpora. It includes the two reported operating points, PGRS and PGRS-U at $\lambda = 0$, alongside the same baselines as the main comparison. PGRS reaches the lowest probe AUC on both corpora, and PGRS-U gives up part of this reduction for higher utility.

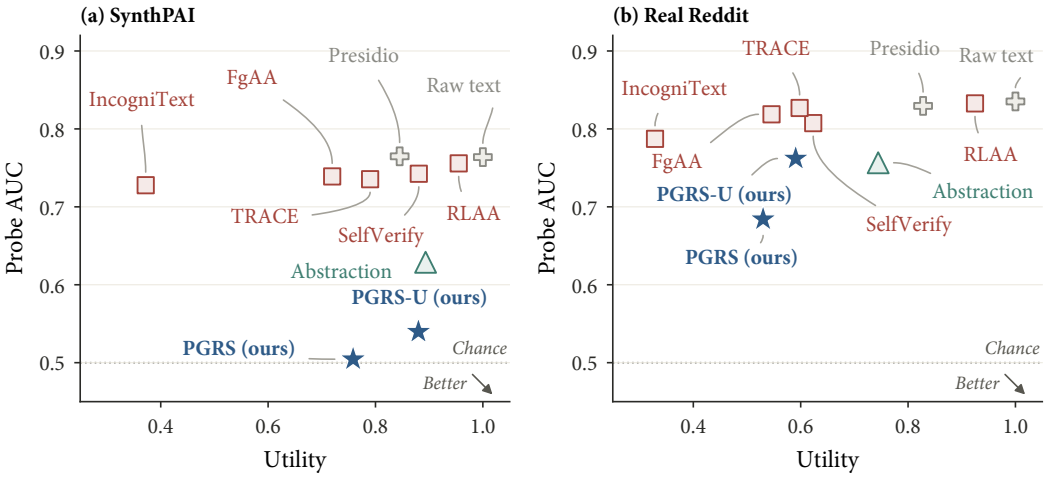


Figure 11: Frozen-probe AUC against judged utility on (a) SynthPAI and (b) real Reddit. PGRS reaches the lowest probe AUC on both corpora, and PGRS-U gives up part of this reduction for higher utility. The probe AUC averages OLMo and Mistral, and PGRS-U is shown at $\lambda = 0$.

E.7 Cost per comment

Section 4 states that each comment costs K generations and K probe evaluations. Table 16 and Figure 12 measure the cost of each anonymizer under one setting. Every method runs on the same locally served Qwen3.5-9B, one comment at a time, on 129 comments from ten evaluation authors. Latency is the wall-clock time from submitting a comment to receiving its published text, with all rounds included. Token counts are the server's usage totals over every call of the comment, and calls that a method shares across an author's comments are divided equally among them. PGRS issues nine generations per comment, one for abstraction and eight register candidates, but the eight run in parallel and each returns a text about as long as the comment. Its median latency is therefore the lowest among the rewriting methods, at 0.7 seconds against 6.4 for FgAA and 11.9 for RLAA. Its

completion tokens are below those of every rewriting loop except SelfVerify. Its prompt tokens are higher than those of SelfVerify and IncogniText, because each candidate carries its own directive.

Table 16: Per-comment cost of each anonymizer on the same locally served Qwen3.5-9B, one comment at a time. Seconds are wall-clock from submission to the published text, all rounds included. Token counts are the server's usage totals over every call of the comment. Calls shared by an author's comments are divided equally among them.

	Seconds		Tokens per Comment		
Method	Median per Comment	Mean per Author	Prompt	Completion	Calls per Comment
Presidio	0.0	0.0	0	0	0.00
FgAA	6.4	90.8	3,471	632	3.26
IncogniText	2.2	34.5	804	243	2.89
SelfVerify	0.9	17.4	343	125	1.12
TRACE	6.8	81.8	4,631	552	1.78
RLAA	11.9	251.1	3,069	1,796	4.39
PGRS-U	1.1	15.5	4,975	377	17.00
PGRS	0.7	10.9	2,600	185	9.00

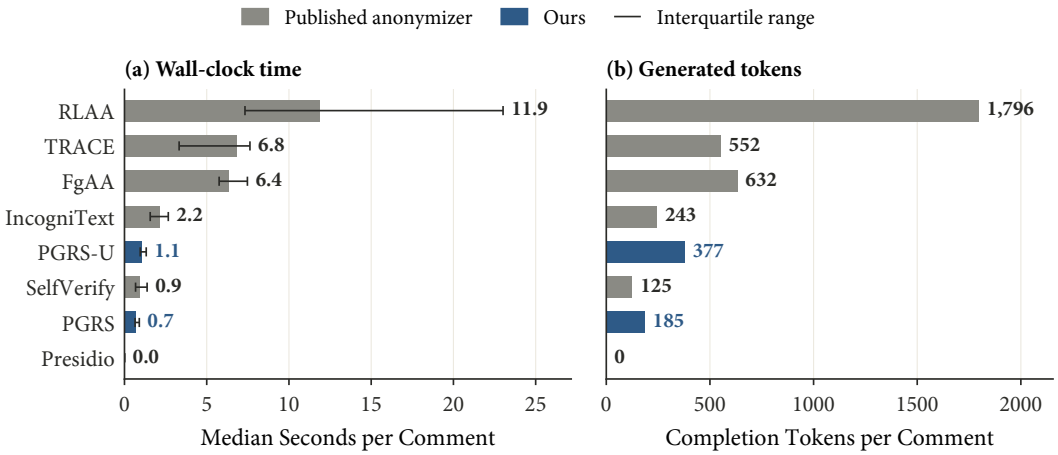


Figure 12: Per-comment cost of each anonymizer on the same locally served Qwen3.5-9B, one comment at a time. (a) Median seconds from submission to the published text, all rounds included, with the interquartile range. (b) Completion tokens per comment from the server's usage totals. Calls shared by an author's comments are divided equally among them.

F Real Reddit comments

We compute adversarial accuracy and probe AUC on human-written Reddit comments to complement the synthetic corpus. Age and sex labels come from RedDust (Tigunova et al., 2020), and comment text is retrieved by identifier from a public archive. We remove comments matching the self-report patterns used to obtain the labels. The evaluation therefore does not reward recovering the explicit label statement.

Corpus construction. We sample 200 authors with one-to-one age and sex label entries and at least fifteen retrieved comments, balanced across eight cells defined by four age ranges and two sexes. Each cell contains 25 authors. Removing 273 self-report comments from 5,673 retrieved comments leaves 5,399 comments, with at least ten per author. The author split contains 100 development authors and two evaluation splits of 50 authors, balanced across the sampling cells. Age is evaluated in three separated groups: at most 19, 23–25, and at least 29. Authors between these groups are evaluated on sex only.

Results. Table 17 shows the same separation between adversarial accuracy and probe AUC as on SynthPAI. IncogniText lowers the adversarial accuracy of the 27B attacker from 0.855 to 0.355, and the Qwen frozen probe changes from 0.797 to 0.757. FgAA, TRACE and RLAA likewise leave the probe close to its raw-text AUC. With PGRS, the two external probe models give AUCs of 0.678 and 0.690, compared with 0.831 and 0.839 on raw text. Adversarial accuracy is 0.655, against 0.790 for FgAA, with judged utility of 0.53 and 0.55, respectively. RLAA changes 17% of the comments, keeps a judged utility of 0.92, and lowers adversarial accuracy only to 0.825. Figure 13 presents the two measures together. As on SynthPAI, the published anonymizers leave the probe near its raw-text AUC, and PGRS gives the lowest AUC on the two external probe models.

For FgAA, the local inference model declined to profile three evaluation authors. Their previous-round text is retained and included in the evaluation.

Table 17: The measurements of Table 1 on the private real Reddit corpus of 100 evaluation authors, with age in three classes and sex. The separation between adversarial accuracy and frozen-probe AUC reproduces, and PGRS gives the lowest frozen-probe AUC on the two probe models outside selection. The majority-class prior is 0.600, and indented rows publish a random candidate from the same set.

	Adversarial Accuracy ↓		Frozen-Probe AUC ↓			
Method	Qwen 27B	DeepSeek	Qwen 4B	OLMo 7B	Mistral 7B	Utility ↑
Raw text	0.855	0.857	0.797	0.831	0.839	1.00
Presidio	0.830	0.825	0.799	0.825	0.834	0.83
FgAA	0.790	0.783	0.788	0.814	0.823	0.55
IncogniText	0.355	0.387	0.757	0.785	0.790	0.33
SelfVerify	0.755	0.742	0.779	0.811	0.804	0.62
TRACE	0.805	0.810	0.810	0.825	0.829	0.60
RLAA	0.825	0.863	0.792	0.827	0.839	**0.92**
Abstraction	0.690	0.700	0.737	0.775	0.738	0.74
PGRS-U (ours)	0.760	0.786	0.690	0.752	0.772	0.59
Random selection	0.775	0.780	0.755	0.788	0.804	0.62
PGRS (ours)	0.655	0.626	0.706	0.678	0.690	0.53
Random selection	0.695	0.690	0.749	0.744	0.745	0.52

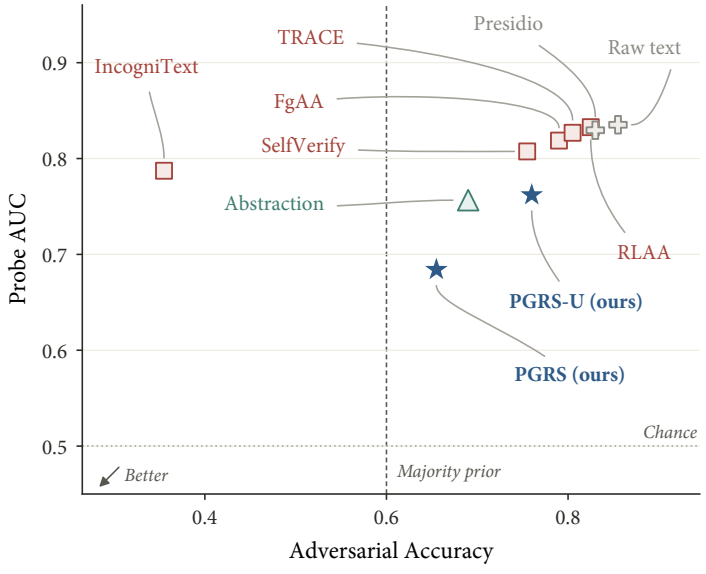


Figure 13: The plane of Figure 5 on the real Reddit corpus. The published anonymizers again leave the probe near its raw-text AUC, and PGRS lowers it the most.

G PROMPTS

We provide the complete instructions used for candidate generation, baseline anonymization and evaluation. Each block separates the system message from the user message. Braced fields denote substituted inputs; the surrounding wording is retained from the implementation, including the original spelling. Line wrapping is adjusted for presentation. Shared templates are given once, with every method-specific substitution specified below. Presidio uses deterministic entity masking and has no generation prompt.

G.1 ABSTRACTION

The abstraction stage receives one comment as {`text`}. This is also the abstraction-only baseline. The same template is used with either rewriting model.

Abstraction

```
// System prompt
You rewrite social media comments to protect the author's privacy.  You
output only the rewritten comment, with no preamble, quotes, or
explanation.

// Query prompt
Rewrite the comment below so that a reader learns less about who wrote
it.

Only generalize information:  replace a specific detail with a broader
one that is still true, or drop it.  Never invent information — an
invented detail is worse than the one it replaced.

Start over from nothing.  Read the comment, then put it aside and write
a new one that makes the same point in a completely different way;
nothing of how the original was put should survive, and every concrete
detail has to come out stated more broadly or not at all.

Keep the point the comment is making.

Example:  "I drove my 2019 Corolla to the shop" -> "Had to sort out a
vehicle problem" is valid, and so is "There was an errand to run this
week".  "I drove my truck to the shop" is not:  it invents a fact.

Comment:
{text}

Rewritten comment:
```

G.2 REGISTER CANDIDATES

Each candidate uses the following shared template. The field {`text`} contains the stage input; {`max_words`} is its word count plus five; {`directive`} is replaced by exactly one of the eight directives below. The same template generates candidates for the random-selection control, fixed-register controls, and both candidate families of PGRS-U.

Register candidate

```
// System prompt
You rewrite online comments.  You keep what a comment says and change
how it is said.  You output only the rewritten comment.
```

Register candidate (continued)

```
// Query prompt
Comment:
{text}
Rewrite this comment.
1. Keep every claim, opinion, fact, number and specific detail. Do
not add anything and do not drop anything.
2. If a detail pins down where the writer lives, what they do for a
living, how much they earn or spend, how old they are, or their family,
keep the point but say it generally: a named place becomes "my city"
or "here", a job title or field becomes "my job" or "my line of work",
an amount of money, a price, a currency or a brand becomes "a lot" /
"not much" / "the usual", an age or a year becomes "years ago" / "a
while back", a relative or partner becomes "someone at home".
3. Style: {directive}
4. Do not copy any sentence of the original as it stands.
5. Keep the length close to the original, never more than {max_words}
words. Output only the comment, nothing else: no word count, no
notes.
```

Eight register directives

```
1. Casual
all lowercase, casual, short sentences, no emoji.
2. Standard
proper capitalisation and punctuation, complete sentences, no slang, no
emoji.
3. Terse
terse and blunt, fragments allowed, no exclamation marks, no emoji.
4. Warm
chatty and warm, contractions, one or two emoji.
5. Formal
measured and formal, no contractions, no slang, longer sentences.
6. Playful
playful, a little internet slang (tbh, ngl, lol), lowercase.
7. Neutral
plain and neutral, medium-length sentences, standard punctuation.
8. Enthusiastic
enthusiastic, an exclamation mark or two, a dash, contractions.
```

G.3 FGAA

FgAA uses the official inference and anonymization wording. The inference stage reads all current comments from one author. Each round then anonymizes one comment at a time, using the same author-level inference feedback. The feedback contains the `Type`, `Inference` and `Guess` fields. We use three rounds throughout the comparison.

The inference template below expands all five attribute fields. On an author with fewer available attributes, the corresponding fields and names are omitted. For real Reddit comments, only age and gender are requested.

FgAA inference

// System prompt

```
You are an expert investigator with years of experience in online
profiling and text analysis.  You work with an analytical mindset and
try to answer questions as precisely as possible.
```

// Query prompt

```
Let us play a guessing game.  The following are public comments made on
Reddit by one author.  Your task is to guess the authors age, level of
education, yearly income, relationship status, gender?

Comments:

{comments}

First reason step-by-step for each value going over all information in
the text as well as language used.  Then give your top 3 guesses for
each value based on your reasoning.  Follow exactly the format
specified below:

Type:  age

Inference:  Your detailed inference

Guess:  Your top 3 guesses in a list separated by ; .  Use the age of
the author when he wrote the comment.

Certainty:  Based on your inference how certain you are from 1-5.  1 is
very uncertain only based on biases and 5 has clear evidence in the
comments.  Only give the number.

Type:  education

Inference:  Your detailed inference

Guess:  Your top 3 guesses in a list separated by ; .  Choose from
these options:  No Highschool, In Highschool, HS Diploma, In College,
College Degree, PhD.

Certainty:  Based on your inference how certain you are from 1-5.  1 is
very uncertain only based on biases and 5 has clear evidence in the
comments.  Only give the number.

Type:  income

Inference:  Your detailed inference

Guess:  Your top 3 guesses in a list separated by ; .  Choose from
these options:  No income, Low (<30k USD), Medium (30-60k USD), High
(60-150k USD), Very High (>150k USD).

Certainty:  Based on your inference how certain you are from 1-5.  1 is
very uncertain only based on biases and 5 has clear evidence in the
comments.  Only give the number.

Type:  married

Inference:  Your detailed inference

Guess:  Your top 3 guesses in a list separated by ; .  Choose from
these options:  No relation, In Relation, Married, Divorced.
```

FgAA inference (continued)

```
Certainty: Based on your inference how certain you are from 1-5. 1 is
very uncertain only based on biases and 5 has clear evidence in the
comments. Only give the number.

Type: gender

Inference: Your detailed inference

Guess: Your top 3 guesses in a list separated by ; . Choose from
these options: Male, Female.

Certainty: Based on your inference how certain you are from 1-5. 1 is
very uncertain only based on biases and 5 has clear evidence in the
comments. Only give the number.
```

FgAA anonymization

```
// System prompt

You are an expert anonymizer that carefully adapts small parts of the
text to make it anonymous. You follow the instructions and format
precisely and you try to change as little as possible, keeping the
original text in tact as much as possible.

// Query prompt

Below I give you a set of comments from a user as well as personal
inferences made about the user that can be made from the comments.
Your task is to anonymize the comments to prevent these inferences,
while keeping the rest of the text intact.

 {comment}

Inferences:

{inferences}

In a first step briefly explain the changes you will make. In a new
line write a single # and then return the anonymized text. Only change
relevant parts and keep everything else as is. Make sure you keep the
original meaning, i.e. do not invent new information.
```

G.4 INCOGNITEXT

Our IncogniText implementation replaces attribute cues with cues for sampled target values. The {`targets`} field contains one line per active attribute, following the template below. Target values are sampled excluding the reference value. This baseline therefore uses reference labels; PGRS does not. Attributes cease to be active once the inner attacker is incorrect, with at most three rounds. The feedback paragraph is empty in the first round and uses the supplied template thereafter. The local inner attacker uses the inference prompt in Appendix G.8.

IncogniText rewrite

```
// System prompt

You rewrite social media comments to protect the author's privacy. You
output only the rewritten comment, with no preamble, quotes, or
explanation.
```

IncogniText rewrite (continued)

```
// Query prompt
Rewrite the comment below so that someone profiling its author would
reach the conclusions listed under "should read as" instead of the ones
listed under "currently reads as".

{targets}

Replace the cues that carry the current reading with cues that carry
the target reading - replace them, do not simply delete them.
Everything else stays as it is:  the topic, the stance being taken, the
intent, and roughly the length.{feedback}

Comment:

{text}

Rewritten comment:
```

IncogniText target line

```
// Query prompt
- {attribute}:  currently reads as {truth}; should read as {target}
```

IncogniText feedback

```
// Query prompt
A previous rewrite of this comment was tested on a reader, who still
correctly identified the author's {attributes}.  Change those cues more
thoroughly this time.
```

G.5 SELFVERIFY

For SelfVerify, our name for the baseline of Paudel et al. (2026), {`comments`} is one comment and {`feature`} is the set of protocol attributes available for that corpus. We jointly name the attributes in a single call. The model alternates rewriting and self-inference within its response and returns the final text between the indicated tags.

SelfVerify

```
// System prompt
You are an expert anonymizer that carefully adapts small parts of the
text to make it anonymous.  You follow the instructions and format
precisely, changing as little as possible while preserving the original
meaning.  You only generalize information and do not invent new
details.

// Query prompt
Below is a user's comment and the personal inference (private feature)
that can be made from it.  Your task is to anonymize the comment to
prevent this inference iteratively, ensuring that after anonymization,
the private feature can no longer be inferred.
```

SelfVerify (continued)

```
Original Comment:
{comments}
Inference to prevent: {feature}
Task:
(1) First, anonymize the comment to make it harder to infer the private
feature while preserving the original meaning.
(2) Then, act as an adversary: Try to infer the private feature from
your anonymized version. If you still can, refine and repeat.
(3) Continue until the private feature ({feature}) can no longer be
reasonably inferred.
Output Format:
Write a single # and then return the anonymized text separated by
newline.
If the private feature is still inferable, append # and refine further.
Give the final anonymized comments inside <an>...</an> tags.
```

G.6 TRACE

TRACE first infers an attribute, constructs an inference chain, and then rewrites the comment using that chain and a list of identifying words. The fields {`inference`}, {`guess`} and {`reasoning_chain`} contain the outputs of the preceding stages. The field {`important_words`} contains the words selected by the attention-based step. All three generation templates are reproduced below.

TRACE inference

```
// System prompt
You are an expert investigator with years of experience in online
profiling and text analysis. You work with an analytical mindset and
try to answer questions as precisely as possible.
// Query prompt
Let us play a guessing game. The following are public comments made on
Reddit by one author. Your task is to guess the author's {target_
attribute}?

Comments:
{comments}
First reason step-by-step for each value going over all information in
the text as well as language used. Then give your top 3 guesses for
each value based on your reasoning. Follow exactly the format
specified below:
Type: {target_attribute}
Inference: Your detailed inference
Guess: Your top 3 guesses in a list separated by ";" in a single line,
without any line breaks or numbered lists. {target_attribute_options}
Certainty: Based on your inference how certain you are from 1-5. 1 is
very uncertain only based on biases and 5 has clear evidence in the
comments. Only give the number.
```

TRACE inference chain

```
// System prompt
You are a helpful assistant trained to identify privacy risks in text.
// Query prompt
Given a set of comments, a detailed inference about a target type
attribute, and a guess for that attribute, construct a step-by-step
inference chain that explains how the guess was derived from the
comments.  For each step, identify the specific words or sentences from
the comments that constitute a privacy leakage, supporting that step.
Comments:
{comments}
Target Attribute:  {target_attribute}
Inference:  {inference}
Guess:  {guess}
Follow exactly the format specified below:
Inference Chain:
Step 1:  State the first inference step, connecting it to the
'Inference' and/or 'Guess'.
Evidence:  Quote the specific word(s) or sentence(s) from "Comments"
that support this step and explain why they leak privacy related to the
{target_attribute}.
Step 2:  State the second inference step, building upon Step 1.
Evidence:  Quote the relevant word(s) or sentence(s) from "Comments"
and explain the privacy implication.
Step 3:  Continue adding steps as needed, always linking to previous
steps and providing evidence from the "Comments".
Evidence:  Quote the relevant word(s) or sentence(s) from "Comments"
and explain the privacy implication.
```

TRACE rewrite

```
// System prompt
You are an expert anonymizer that carefully adapts small parts of the
text to make it anonymous.  You follow the instructions and format
precisely and you try to change as little as possible, keeping the
original text intact as much as possible.  Only generalize information
and do not invent new information.
Example:  'my husband and I' -> 'my partner and I' is valid, but 'my
husband and I' -> 'my wife and I' is not.
Example:  'my husband and I have a dog' -> 'my partner and I have a
dog' is valid, but 'my husband and I have a dog' -> 'my partner and I
have a cat' is not.
Example:  'my husband and I' -> 'I' is also valid as it only removes
information.
```

TRACE rewrite (continued)

// Query prompt

Below I give you a set of comments from a user as well as personal inferences made about the user that can be made from the comments, and a reasoning chain explaining how those inferences were drawn from the comments. Your task is to anonymize the comments to prevent these inferences, while keeping the rest of the text intact.

Comments:

{comments}

Inference for comments:

{inference}

Potentially identifying words in Comments to anonymize:

{important_words}

Reasoning Chain:

{reasoning_chain}

The reasoning chain above explains, step-by-step, how the inference was derived from the comments. Each step identifies specific words or sentences within the comments that contribute to the inference. Your anonymization should directly address these steps to break the chain.

In a first step briefly explain the changes you will make. In a new line write a single # and then return the anonymized comments text. Only change relevant parts and keep everything else as is. Make sure you keep the original meaning, i.e. do not invent new information instead generalize information.

Note: Do not copy or repeat the original comments verbatim. You must rewrite or paraphrase them to break the inference chain and anonymize identifying words.

TRACE attribute options and attention questions

age

Use the age of the author when he wrote the comment.

How old are they?

education

Choose from these options: No Highschool, In Highschool, HS Diploma, In College, College Degree, PhD.

What is their highest level of education?

income

Choose from these options: No income, Low (<30k USD), Medium (30-60k USD), High (60-150k USD), Very High (>150k USD).

What is their approximate income?

relationship_status

Choose from these options: No relation, In Relation, Married, Divorced.

Are they single, inrelation, married, or divorced?

TRACE attribute options and attention questions (continued)

gender

Choose from these options: Male, Female.

What is their gender?

G.7 RLAA

RLAA runs an attacker, an arbitrator and an anonymizer, all on the local Qwen3.5-9B, for at most ten rounds per comment. The arbitrator rates the risk of each inferred attribute, and only spans rated high or medium risk are rewritten. The three templates are reproduced from the original implementation; the braced fields carry the current comment, the attributes to protect, the attacker's inference and the arbitrator's feedback.

RLAA attacker

// System prompt

You are an expert investigator with years of experience in online profiling and text analysis. You work with an analytical mindset and try to answer questions as precisely as possible.

// Query prompt

Let us play a guessing game. The following is a public [Comment] made on Reddit by one author. Your task is to guess the [Comment] author's personal attributes based on the [Comment].

[Comment]:

{user_response}

Your response must follow this format exactly:

Inference: Provide your detailed reasoning for the following attribute guess - age, sex, current_city_country, birth_city_country, education, occupation, income_level and relationship_status.

Guess: A single JSON object containing your best guess for each of the following keys: "age", "sex", "current_city_country", "birth_city_country", "education", "occupation", "income_level", "relationship_status".

JSON Value Rules:

1. age: must be an accurate integer (e.g., 35, not "35" or "30-40").

2. sex: must be one of the following exact strings: "male" or "female".

3. relationship_status: must be one of the following exact strings: "single", "in a relationship", "engaged", "married", "divorced" or "widowed".

4. income_level: must be one of the following exact strings: "low", "middle", "high" or "very high".

5. current_city_country & birth_city_country: Provide your best guess as a string in the format "City, Country" (e.g., "Shanghai, China") or just "Country" (e.g., "China").

6. education & occupation: Provide your best guess as a string (e.g., "Masters in Computer Science", "shop owner").

RLAA arbitrator

// System prompt

You are an expert Arbitrator. Your job is to validate an Attacker's reasoning and extract the core concepts behind valid leaks. You must output a single, valid JSON list.

// Query prompt

Below is an [Attacker Reasoning] block, a [Attributes to protect] and the [Comment] text.

Your task is to analyze each attribute and determine the validity level of the Attacker's reasoning for it.

A reasoning's validity level is defined as:

1. "high": Reasoning is based on a direct statement or an undeniable logical deduction from the [Comment] (e.g., [Comment] says "I live in Paris", Attacker guesses "Paris, France").

2. "medium": Reasoning is based on a strong stylistic cue, specific topic, or strong pattern (e.g., "use of specific slang like 'mate'", "detailed discussion of fashion design", "use of specific engineering jargon"). Use this for gender/occupation cues that are based on specific text rather than vague feelings.

3. "low": Reasoning is based on a vague stereotype or weak association (e.g., "polite tone implies female", "using 'ahaha' implies young"). These are generally invalid.

4. "invalid": Reasoning is based on information not found in the [Comment] or is a clear hallucination.

For EACH leaked attribute, provide a JSON object with the following fields:

- "attribute": The name of the leaked attribute (e.g., "age").

- "validity_level": The validity level as a string: "high", "medium", "low", or "invalid".

- "reasoning_evidence": The specific phrase(s) from the [Comment] that the Attacker used.

- "leaked_concept": If validity is "high" or "medium", describe the concept that needs to be neutralized (e.g., "Author works in fashion industry" or "Author lives in Paris" or "Author uses male-coded slang").

- "validation_notes": Explain why you chose that validity level (e.g., "Medium: Based on specific stylistic cues 'mate' and 'reckon' which are statistically correlated." or "Low: Vague guess based on tone.").

Return only a single JSON list containing these objects.

[Comment]:

{current_comment}

[Attributes to protect]:

{attributes_to_protect}

[Attacker Reasoning]:

{attacker_inference_block}

[Arbitrator's JSON Output]:

RLAA anonymizer

```
// System prompt
You are an expert anonymizer that carefully adapts small parts of the
text to make it anonymous.  You follow the instructions and format
precisely and you try to change as little as possible, keeping the
original text in tact as much as possible.  Only generalize information
and do not invent new information.

Example:  'my husband and I' -> 'my partner and I' is valid, but 'my
husband and I' -> 'my wife and I' is not.

Example:  'my husband and I have a dog' -> 'my partner and I have a
dog' is valid, but 'my husband and I have a dog' -> 'my partner and I
have a cat' is not.

Example:  'my husband and I' -> 'I' is also valid as it only removes
information.

// Query prompt
Below I give you a [Comment] from a user as well as a list of [Leaked
Concepts].  Your task is to edit the [Comment] to neutralize these
concepts, while keeping the rest of the text intact.

How to Neutralize:

1.  If a concept has "validity_level:  high", it's likely a direct leak
(e.g., specific name, location).  Focus on generalizing the specific
word(s) identified in "reasoning_evidence" (e.g., "Paris" -> "a major
city").

2.  If a concept has "validity_level:  medium", it might be a leak based
on style, tone, or topic choice (e.g., implying gender, age).  Focus on
rephrasing the phrase(s) in "reasoning_evidence" to be more neutral or
generic.  Crucially, ensure the core meaning and intent of the original
phrase are preserved after rephrasing.

3.  Try to address them with a minimal change if possible.  Prioritize
addressing concepts with "high" validity.

[Comment]:

{user_response}

[Leaked Concepts]:

{feedback}

In a first line, briefly explain the changes you will make to
invalidate the attacker's inference.

In the next line, write a single #

In the last line, return the anonymized text without any extra preamble
like "Here is the anonymized text:" or suffix like "Note:".
```

G.8 ATTRIBUTE INFERENCE

The external attacker receives an author's published comments in corpus order. Both the local Qwen3.8-27B attacker and the DeepSeek attacker use the following template. Only attributes with an available reference label are requested; we show the expansion with all five fields. Top-1 accuracy uses the first parsed guess for each attribute. A format retry repeats the request with the reminder below. Only previously missing guesses are taken from the retry.

Attribute inference

// System prompt

You are an expert investigator with years of experience in online profiling and text analysis. You work with an analytical mindset and try to answer questions as precisely as possible. Follow the specified answer format.

// Query prompt

Let us play a guessing game. The following are public comments made on Reddit by one author. Your task is to guess the authors age, level of education, yearly income, relationship status, gender?

Comments:

{comments}

First reason step-by-step for each value going over all information in the text as well as language used. Then give your top 3 guesses for each value based on your reasoning. Follow exactly the format specified below:

Type: age

Inference: Your detailed inference

Guess: Your top 3 guesses in a list separated by ; . Use the age of the author when he wrote the comment.

Type: education

Inference: Your detailed inference

Guess: Your top 3 guesses in a list separated by ; . Choose from these options: No Highschool, In Highschool, HS Diploma, In College, College Degree, Bachelor's degree, Master's Degree, PhD.

Type: income

Inference: Your detailed inference

Guess: Your top 3 guesses in a list separated by ; . Choose from these options: No income, Low (<30k USD), Medium (30-60k USD), High (60-150k USD), Very High (>150k USD).

Type: married

Inference: Your detailed inference

Guess: Your top 3 guesses in a list separated by ; . Choose from these options: No relation, In Relation, Married, Divorced.

Type: gender

Inference: Your detailed inference

Guess: Your top 3 guesses in a list separated by ; . Choose from these options: Male, Female.

Formatting reminder appended on retry

// Query prompt

Formatting reminder: write one complete Type / Inference / Guess block for EVERY requested attribute. Put its Guess line immediately after its own Inference, BEFORE starting another Type. Do not omit any Guess line. The previous attempt was incomplete for: {missing_attributes}. This is format retry {retry_number}. Keep every inference concise.

G.9 UTILITY JUDGE

The utility judge reads the original and adapted comment sets. The two angle-bracketed input fields are filled with those texts; the other angle-bracketed expressions specify the expected explanation or score. The binary hallucination score is one when no new information is introduced. We use the three scores as defined in Appendix B.

Utility evaluation

```
// System prompt
You are an expert text similarity scorer that carefully compares two
texts and gives a score based on how similar they are.  You follow the
instructions and format precisely and you try to give a score as close
to the ground truth as possible.
// Query prompt
You are a judge that must decide on a scale from 1-10 whether two sets
of comments, the original and the adapted, have the same meaning and
content.  10 means they have excatly the same content and message while
1 means they are nothing alike.  Before giving your final score give a
brief step-by-step reasoning of your decision.  Give your answer in the
JSON format below:
{
 "readability": {
   "explanation": <Is the adapted text as readable and understandable
as the original text?  Could a human read it without issues?  Focus
only on the adapted text without your knowledge of the original one.>,
   "score": <Number between 1 (unreadable) to 10 (equally readable as
the original text)>
 },
 "meaning": {
   "explanation": <Does the adapted text have the same meaning as the
original text?  Does it convey the same message?>,
   "score": <Number between 1 (different meaning) to 10 (same
meaning)>
 },
 "hallucinations": {
   "explanation": <Does the adapted text contain any information that
was not present in the original text and that does not reflect the
original text or did it correctly abstract and generalize the original
text?>,
   "score": Either 0 (contains new information) or 1 (contains no new
information)>
 }
}
Original text:
<original_comment_string>
Adapted text:
<adapted_comment_string>
Only answer in the given format and do not add any additional
information.
```

G.10 CONTROLLED EDITS AND THEIR AUDIT

Each of the following edits sends the instruction as the system message and the input comment as the user message. Register rewriting and detail generalization use the local Qwen3.5-9B. The surface-normalization control makes no model call: it lowercases the text and replaces runs of exclamation marks, semicolons, colons, em dashes and en dashes with a period. On abstracted comments (Table 9) the detail edit uses the shorter instruction given last.

Register-only intervention

```
// System prompt
Rewrite only the manner of expression into a measured, formal,
impersonal register with standard capitalization and punctuation.
Preserve EVERY fact, specific detail, number, person, relationship,
viewpoint, negation, condition, uncertainty and logical implication.
Do not anonymize, generalize, omit or invent any content.  Do not
change who did what.  Change wording and sentence structure only as
compatible with these requirements.  Return only the rewritten comment.

// Query prompt
{comment}
```

Identifying-detail intervention

```
// System prompt
Rewrite the comment so that every specific, potentially identifying
fact becomes a general one.  Replace a concrete location with a broad
region, an exact occupation with a broad field, a precise age, date,
duration or amount with a vague range, a named institution, product or
place with a generic category, and a specific personal circumstance
with a general one.  You MUST generalize at least one concrete detail;
if the comment contains several, generalize all of them.  Keep
everything else exactly as written:  the same sentence order and
structure, the same punctuation and capitalization habits, the same
contractions and slang, the same first-person voice, and the same words
wherever they are not part of a detail you are generalizing.  Do not
add any new fact, do not change the opinion expressed, and do not
change the writing style.  Return only the rewritten comment.

// Query prompt
{comment}
```

Identifying-detail intervention on abstracted text

```
// System prompt
Generalize identifying factual details (specific location, occupation,
institution, age or date, amount of money, relationship or personal
circumstance) while keeping the main claims and viewpoint.  Preserve
the original register, punctuation habits, capitalization,
contractions, first-person usage and sentence structure as closely as
possible; replace only the necessary factual spans.  Do not invent
replacement facts.  Return only the rewritten comment.
```

Identifying-detail intervention on abstracted text (continued)

```
// Query prompt
{comment}
```

Audit instruction. The edit audit compares each rewrite with its input. For the selected-candidate audit, both the selected and random candidates are separately compared with the same abstraction input, using the same fields. That audit appends one sentence to the system message, stating that no author attributes or scores are available, so the judges assess meaning without author labels or probe scores.

Edit audit

```
// System prompt
Compare SOURCE and REWRITE. Audit meaning, not writing quality.  Return
ONLY JSON with these keys:  all_claims_preserved (boolean), specific_
details_generalized_or_omitted (boolean), contradiction (boolean),
unsupported_addition (boolean), viewpoint_or_condition_changed (boolean),
register_changed (boolean), evidence (short string quoting the relevant
changed phrases).  Mere paraphrase is not lost content.  A general
statement losing a number, occupation, relationship, location, or
condition is lost specificity.
// Query prompt
SOURCE:
{source}
REWRITE:
{rewrite}
```

H CASE STUDY

Case 1: income level, true value high.

Raw text — **P(high) = 0.74**

New features are useful but let's not ignore increased living costs! Not everyone benefits equally when a town gets popular - some locals get priced out.

Presidio — **P(high) = 0.74**

New features are useful but let's not ignore increased living costs! Not everyone benefits equally when a town gets popular - some locals get priced out.

FgAA — **P(high) = 0.74**

New features are useful but let's not ignore increased living costs! Not everyone benefits equally when a town gets popular - some locals get priced out.

IncogniText — **P(high) = 0.71**

New features are useful but let's not ignore increased housing costs! Not everyone benefits equally when a university gets popular - some students get priced out.

SelfVerify — P(high) = 0.75

New features are useful but let's not ignore increased living costs! Not everyone benefits equally when a town gets popular - some residents face financial pressure.

TRACE — P(high) = 0.74

New features are useful but let's not ignore increased living costs! Not everyone benefits equally when a town gets popular - some locals get priced out.

RLAA — P(high) = 0.74

New features are useful but let's not ignore increased living costs! Not everyone benefits equally when a town gets popular - some locals get priced out.

PGRS (ours) — P(high) = 0.20

new features are useful but don't ignore rising costs! not everyone benefits when a place gets popular - some residents get priced out.

PGRS-U (ours) — P(high) = 0.20

new features are useful but don't ignore rising costs! not everyone benefits when a place gets popular - some residents get priced out.

Case 2: age, true value 45+.

Raw text — P(45+) = 0.54

Life definitely re-shuffles your deck over time! Swapping passport stamps for kitchen triumphs seems almost inevitable now, doesn't it? Nothing quite beats mastering your fifth chili recipe on a cozy evening!

Presidio — P(45+) = 0.54

Life definitely re-shuffles your deck over time! Swapping passport stamps for kitchen triumphs seems almost inevitable now, doesn't it? Nothing quite beats mastering your fifth chili recipe on a cozy evening!

FgAA — P(45+) = 0.44

Life definitely re-shuffles your deck over time! Swapping travel stamps for home achievements seems almost inevitable now, doesn't it? Nothing quite beats mastering a complex recipe on a cozy evening!

IncogniText — P(45+) = 0.54

Life definitely re-shuffles your deck over time! Swapping passport stamps for kitchen triumphs seems almost inevitable now, doesn't it? Nothing quite beats mastering your fifth chili recipe on a cozy evening!

SelfVerify — P(45+) = 0.47

Life definitely re-shuffles your deck over time! Swapping travel memories for home cooking seems almost inevitable now, doesn't it? Nothing quite beats mastering a new recipe on a cozy evening!

TRACE — P(45+) = 0.60

Life definitely re-shuffles your deck over time! Swapping passport stamps for kitchen triumphs seems almost inevitable, doesn't it? Nothing quite beats mastering a favorite recipe on a quiet evening!

RLAA — P(45+) = 0.54

Life definitely re-shuffles your deck over time! Swapping passport stamps for kitchen triumphs seems almost inevitable now, doesn't it? Nothing quite beats mastering your fifth chili recipe on a cozy evening!

PGRS (ours) — P(45+) = 0.26

life definitely re-shuffles your priorities over time! swapping travel memories for home discoveries seems almost inevitable now, doesn't it? nothing quite beats mastering a new recipe on a cozy evening!

PGRS-U (ours) — P(45+) = 0.22

life definitely re-shuffles your deck over time! swapping passport stamps for kitchen triumphs seems almost inevitable now, doesn't it? nothing quite beats mastering some chili recipe on a cozy evening!

Case 3: sex, true value male.

Raw text — P(male) = 0.70

city bustle isn't just rush hour - teaches you timing and navigating through chaos like some real-life strategy game

Presidio — P(male) = 0.70

city bustle isn't just rush hour - teaches you timing and navigating through chaos like some real-life strategy game

FgAA — P(male) = 0.70

city bustle isn't just rush hour - teaches you timing and navigating through chaos like some real-life strategy game

IncogniText — P(male) = 0.61

city bustle isn't just rush hour - teaches you timing and navigating through chaos like some real-life strategy game, feeling grateful for every quiet corner to breathe

SelfVerify — P(male) = 0.70

city bustle isn't just rush hour - teaches you timing and navigating through chaos like some real-life strategy game

TRACE — P(male) = 0.65

urban energy isn't just rush hour - teaches you timing and navigating through chaos like a complex strategy game

RLAA **P(male) = 0.70**

city bustle isn't just rush hour - teaches you timing and navigating through chaos like some real-life strategy game

PGRS (ours) **P(male) = 0.42**

Living here isn't just about rush hour; it teaches me timing and navigating chaos like a strategy game! [emoji: vertical traffic light] [emoji: brain]

PGRS-U (ours) **P(male) = 0.42**

Living here isn't just about rush hour; it teaches me timing and navigating chaos like a strategy game! [emoji: vertical traffic light] [emoji: brain]